MONASH University

Maximizing Throughput in Multi-Function Robotic Cells

Mehdi Foumani
School of Applied Sciences and Engineering
Monash University

A Thesis Submitted for the Degree of
Doctor of Philosophy
2016

Copyright Notices

Notice 1

Under the Copyright Act 1968, this thesis may not be reproduced in any form without the written permission of the author.

Notice 2

I certify that I have made all reasonable efforts to secure copyright permissions for third-party content included in this thesis and have not knowingly added copyright content to my work without the owner's permission.

This page is intentionally left blank

This page is intentionally left blank

Abstract

Multi-functionality of robots is almost a new objective of interest, both theoretically and in practice. Recent work has shown the robot is able not only to act as a material handling device but also to inspect the part in transit between machines. Such a kind of robot and the cell in which it is applied are called the *Multi-Function Robot (MFR)* and the *Multi-Function Robotic Cell (MFRC)*, respectively. Also, the inspection scenario under this condition is named *in-line* inspection scenario.

Considering a MFRC, this thesis contains two main contributions. Firstly, we limit our study to a MFR which only measures the thickness of the part and records results in an independent computer. Accordingly, the processing route of the part is fixed although the MFR performs the inspection process of the part. Under this condition, we find a deterministic model for minimizing the cycle time. Secondly, we consider the user interface computer can be used to modify the processing route of each part based on its inspection result. This means that the number of processing of the part by the production machine is a random variable depending on the inspection result. Consequently, we should develop a stochastic model for minimizing the partial cycle time. For this case, we also focus on two other inspection scenarios in addition to in-line one: *post-process* and *in-process*. For the first scenario, the inspection process is performed by an independent inspection machine, while parts are inspected in the production machine using multiple sensors for the second scenario. Since the inspection can be performed by a MFR, we extend results for the in-line scenario. Furthermore, it is shown how a robotic cell with post-process (or in-process) inspection scenario can be converted into a robotic cell with in-line inspection scenario.

We propose an analytical method for minimizing cycle time (or expected cycle time) of cells under the aforementioned conditions. Accordingly, the thesis is organized as follows: Chapters 1 and 2 give a general overview of robotic cells, and then Chapters 3-6 present four published papers related to the situation in which the processing route of the part is fixed. The first paper is related to the origin of MFRCs. Following that, second and third papers are related to small- and large-scale MFRCs which only record the inspection results. Finally, the forth paper is related to the operational flexibility in MFRCs. Note that Chapters 3-6 are precedents for Chapters 7-9 where the processing route of each part is modified based on its inspection results. We present two papers in Chapters 7 and 8 to cover robotic cells with post-process and in-process inspection scenarios. Then, in Chapter 9, we show how cells with in-process and post-process inspection scenarios can be converted into a MFRC, which has an in-line inspection scenario. Finally, Chapter 10 presents concluding remarks and some suggestions for MFRCs operating in a dynamic environment.

This page is intentionally left blank

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Signature: Mehdi Foumani

Name: Mehdi Foumani

Date: 21st of April, 2016

This page is intentionally left blank

Publications During Enrolment

I hereby declare that this thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

This thesis includes three original papers published in peer reviewed journals, two conference papers, and two unpublished journal publications. The core theme of the thesis is the development of analytical methods for solution of different types of scheduling problems related to multi-function robotic cells. The ideas, development and writing up of all the papers in the thesis were the principal responsibility of myself, the student, working within the School of Applied Sciences and Engineering, Monash University, under the supervision of Associate Professor Indra Gunawan and Professor Kate Smith-Miles.

In the case of Chapters 3-9 my contribution to the work involved the following:

Thesis Chapter	Publication Title	Status	Nature and % of student contribution	Co-author name(s) Nature and % of Co-author's contribution*
3	Scheduling dual-gripper robotic cells with a hub machine	Published	85%. key ideas, organisation, and writing up	1) Yousef Ibrahim, 10% 2) Indra Gunawan, 5%
4	Notes on optimality conditions of small-scale multi-function robotic cell scheduling problems with pickup restrictions	Published	85%. key ideas, organisation, and writing up	1) Indra Gunawan, 5% 2) Yousef Ibrahim, 5% 3) Kate Smith-miles, 5%
5	Scheduling rotationally arranged robotic cells served by a multi-function robot	Published	90%. key ideas, development, and writing up	1) Indra Gunawan, 5% 2) Kate Smith-miles, 5%
6	Increasing throughput for a class of two-machine cell served by a multi-function robot	In press	90%. key ideas, organisation, and writing up	1) Indra Gunawan, 5% 2) Kate Smith-miles, 5%
7	Resolution of deadlocks in a robotic cell scheduling problem with post-process inspection: avoidance and recovery scenarios	Published	70%.organisation, and writing up	1) Kate Smith-miles, 20% 2) Indra Gunawan, 10%
8	Stochastic Scheduling of a Two-Machine Robotic Cell with In-Process Inspection	submitted	90%. key ideas, development, and writing up	1) Kate Smith-miles, 5% 2) Indra Gunawan, 5%
9	Two-machine robotic rework cells: in-process, post-process and in-line inspections	submitted	90%. key ideas, development, and writing up	3) Kate Smith-miles, 5% 4) Indra Gunawan, 5%

I have / have not renumbered sections of submitted or published papers in order to generate a consistent presentation within the thesis.

Student signature: Mehdi Foumani **Date: 15/04/2016**

The undersigned hereby certify that the above declaration correctly reflects the nature and extent of the student's and co-authors' contributions to this work. In instances where I am not the responsible author I have consulted with the responsible author to agree on the respective contributions of the authors.

Main Supervisor signature: **Date:** 15/4/2016

This page is intentionally left blank

To the memory of my brother, Mesam

This page is intentionally left blank

Acknowledgements

Completion of this doctoral research would not have been possible without the support and assistance of numerous people throughout the research project. I would like to express my sincere appreciation to my main supervisor, Assoc. Prof. Indra Gunawan, for the support of my Ph.D research and all memories we shared over the last four years.

The opportunities that have been available to me as a research higher degree student at Monash University have been endless. However, without a doubt, the greatest opportunity of all has been the chance to work with my associate supervisor, Prof. Kate Smith-Miles. Her guidance helped me in all aspects of my research and the writing of this thesis.

My sincere thanks also go to Prof. Mahrous Yousef for his support and encouragement during my Ph.D study. I am indebted to all people who have supported and encouraged me to strive towards my goal over the last few years especially those in Gippsland Campus. In particular, I would like to thank Asghar, Mohsen, Esmaeil, Ebrahim, Wendy, Vince, Jenny, Erica, Mai, Shakif, Jillian, Mohammed, David, Alicia, Adel, Rahul, Winston, Edward, Zhan, and Andrew.

I would like to thank people who helped me during my master and undergraduate degrees: Prof. Fariborz Jolai, Prof. Shahram Shadrokh, Prof. Babak Abbassi, and Dr. Ali Eshragh.

I would specially like to thank my inspiring parents, Ahmad and Nahid, and my wonderful siblings for all of the sacrifices they have made on my behalf. I love you all very much.

This page is intentionally left blank

Contents

List of Figures

List of Tables

This page is intentionally left blank

List of Acronyms

AD	Absolute Dominance
ALD	Atomic Layer Deposition
CMS	Cellular Manufacturing System
CNC	Computer Numerical Control
CE	Cross Entropy
FIFO	First In First Out
FSD	First-order Stochastic Dominance
FMS	Flexible Manufacturing System
GA	Genetic Algorithm
GGG	Grip-Gage-Go
IFR	International Federation of Robotics
i.i.d	Independent and identically distributed
JIT	Just In Time
MHD	Material Handling Device
MPS	Minimal Part Set
MILP	Mixed integer linear programing
MGF	Moment Generating Function
MFR	Multi-Function Robot (MFR)

MFRC	Multi-Function Robotic Cell
PCB	Printed circuit board
RCSP	Robotic Cell Scheduling Problem
RCPI	Robotic Cell with the Post-process Inspection
RRCSI	Robotic Rework Cells with Start of line Inspection
RRCEI	Robotic Rework Cells with end of line Inspection
SSD	Second-order Stochastic Dominance
SFRC	Single-Function Robotic Cell
SFR	Single-Function Robot
SE	Spectroscopic Ellipsometry
SPN	Stochastic Petri Net
TS	Tabu Search
TSP	Traveling Salesman Problem
VLSI	Very Large Scale Integrated
WIP	Work-In-Process

Part I

Introduction

Chapter 1

Introduction

Industrial technology has recently been followed by a meaningful development in areas such as computers, sensor technology, and mechatronics. This opens up a unique opportunity to develop automation solutions adapted to both small- and large-sized shops. The level of automation is a major decision for manufacturers since an effective level of automation means a stronger economy. To the best of our knowledge, robotic cells are directly the outcome of modern manufacturing industries with a high level of automation. Even though these industries apply human operators for operations, a significant improvement in productivity is obtained from maximizing levels of automation. This is because robots are safer, more programmable, less expensive, more flexible and agile, and capable of interfacing with consumers. For example, consider a press line for draw-forming of automobile body plan or an electroplating line. Obviously, it is much safer to apply a robot instead for a human operator for both of examples. Consequently, the productive use of robots in manufacturing cells is of paramount importance for production engineers due to the fact that they represent a substantial portion of the factory automation as well as the factory investment.

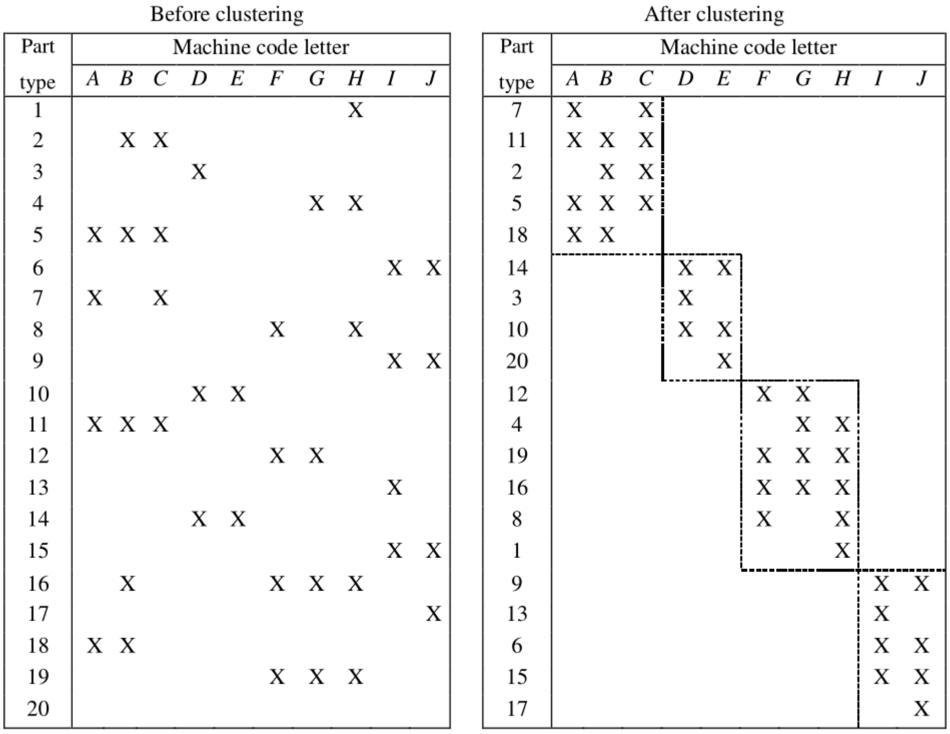
Among technical issues pertaining to real-life robotic cells, the scheduling of the robot movement between machines, the sequence of the robot operations and the part inspection strategy selection are absolutely vital as they have influence on the performance and the quality of the completed parts. Therefore, this analytical study is expended on the study of the problem of obtaining optimal sequence of robot moves for different categories of cell based on the employed industrial robot.

1.1 Cellular Manufacturing Systems

The Cellular Manufacturing System (CMS) is probably the most popular alternative for a mass production environment. It is a model for factory design which is based upon the principles of Group Technology. Actually, it goes in search of parts with similar processing routes and categorizes them as same part families. Furthermore, production machines are jointly grouped together based on the families of parts processing in a CMS, and any particular group of these machines is called a *cell*. The most important character of a CMS is that it involves processing a collection of similar parts (part families) on a dedicated cluster of machines or manufacturing processes (cells). Also, the main advantage of using CMSs is that it considerably makes the material flow better. This clearly reduces the distance travelled by materials and maximize the production rate in the factory. As our best knowledge, CMSs result in the ability for manufacturers to produce high quality products at a low manufacturing cost, without delay, and in a flexible manner regardless the applied production environment. The concepts of a CMS can be utilized in such a high variety flow demand environment to derive the economic advantages inherent in a low variety/high demand environment (Safaei and Tavakkoli-Moghaddam, 2009).

The primary purpose of using cellular manufacturing is integrating the efficiency of product layouts into functional layouts. These layouts imply two general patterns in the production line design: on one side, product layouts help manufacturing systems to act service-oriented by employment of available single-function machines to complete different parts. Considering group technology, machines are divided into small-scale clusters based on operational resemblances in their processing activities. As a result of established small-scale clusters, such an agile and highly reconfigurable system is able to product various parts, and rapidly respond to a highly competitive market. On the other side, functional layouts operationalize the concept of applying specialized machines like CNC machines for manufacturing a few standardized products. Contrary to product layouts, these layouts accomplish operational accuracy through the specialization of machines (Lim et al., 2006). A CMS achieves the benefits of both layouts.

Figure 1.1 clarifies how CMSs are incorporated into production. The objective of the cluster analysis used in this production environment is to assign 20 parts and 10 machines to four part families and four cells, respectively. As it is shown in the right side, it is possible to cluster parts and machines in a right way to have four independent cells that produce discrete parts. In more detail, machines $\{A, B, C\}$ join to form the first cell producing parts $\{7, 11, 2, 5, 18\}$, machines $\{D, E\}$ join to form the second cell producing parts $\{14, 3, 10, 20\}$, machines $\{F, G, H\}$ join to form the third cell producing parts $\{12, 4, 19, 16, 8, 1\}$, and finally machines $\{I, J\}$ join to form the forth cell producing parts $\{9, 13, 6, 15, 17\}$. Clearly, controlling four small cells is much easier than controlling a large manufacturing system consisting of 20 machines. Likewise, for complex systems, manufacturers prefer to establish methods which can be used for this purpose.



Before clustering

Part	Machine code letter									
type	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*
1								X		
2		X	X							
3				X						
4							X	X		
5	X	X	X							
6									X	X
7	X		X							
8						X		X		
9									X	X
10				X	X					
11	X	X	X							
12						X	X			
13									X	
14				X	X					
15									X	X
16		X				X	X	X		
17										X
18	X	X								
19						X	X	X		
20										

After clustering

Part	Machine code letter									
type	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*
7	X		X							
11	X	X	X							
2		X	X							
5	X	X	X							
18	X	X								
14				X	X					
3				X						
10				X	X					
20					X					
12						X	X			
4							X	X		
19						X	X	X		
16						X	X	X		
8						X		X		
1								X		
9									X	X
13									X	
6									X	X
15									X	X
17										X

Fig. 1.1. The optimal clustering of 20 parts and 10 machines.

Various approaches for scheduling and control of robotic cells have already been presented by several robot manufacturers, such as robotic cells offered by ABB Company (Lippiello et al., 2007). All these robot manufacturers incorporated CMS into the design of robotic cells. Therefore, the emphasis in this thesis is on providing a comprehensive treatment of various aspects of scheduling of robotic cells as a subdivision of CMSs.

1.2 Robotic Cells

The idea of applying robots for production in cells is a subdivision of cellular manufacturing. In fact, it is possible to use different kind of servers for material handling of a particular CMS. In the absence of any automation, trained manpower is responsible for transferring parts between machines. This can be a cause of decreasing the quality level of part and the safety of employees. As a consequence, it is not recommended for the most of production environments. An alternative way for increasing the quality level of part and the safety of employees as much as possible is that increase factory automation. On the subject of factory automation, it is common in practice to use robots instead of manpower. Generally, a computer-controlled manufacturing cell which integrates a number of production machines, a material handling robot, and some storage buffers is called a robotic cell.

The robot is one of the most popular automation device used in many industries such as the manufacture of semiconductors, printed circuit boards, pharmaceutical compounds, magnetic resonance imaging systems, fibre optics, cosmetics and glass products (Dawande et al., 2005). Financially speaking, a high worldwide customer demand, which was equal to 1.22 billion dollars, for the robotics industry was during 2005. This demand started falling in 2006 due to an undesirable economic outcome in the American and European markets, and consequently a sudden decrease in car sales. However, it was increased 36 percent in the early 2008 again. Although the robotics industry underwent the recent economic crisis, income in 2008 was 979.4 million dollars. Orders in 2010 raised to 993.2 million dollars after passing slow period in 2009 (Geismar et al., 2012). Since 2010, the total orders of robots has accelerated because of the growing trend in automation, and it ends up with the maximum annual orders ever recorded in 2014. These industry statistics tell us that companies in different sectors of the economy found out the advantages of automation so that there is a heavy demand for robots in the market (IFR International Federation of Robotics 2015).

In most practical applications, a robot is cyclically employed to serve a number of workstations by acting as a material handling device. The ever-increasing use of robots as modern material handling devices is a concerted effort to accomplish the flexibility requirement of serial production systems, which have traditionally been identified by rigid automation. In fact, there are an input buffer (I), machines $M_1, M_2, ..., M_m$ and an output buffer (O), in a robotic cell served by a single central robot. Initially, the robot picks up each part at I, moves it to the first machine and loads it on that machine for processing. On the completion of that process, the robot unloads the part and moves it to the next machine for processing, and so on. After finishing all of operations, the robot moves the completed part to O and drops it there. I and O have unlimited storage capacity.

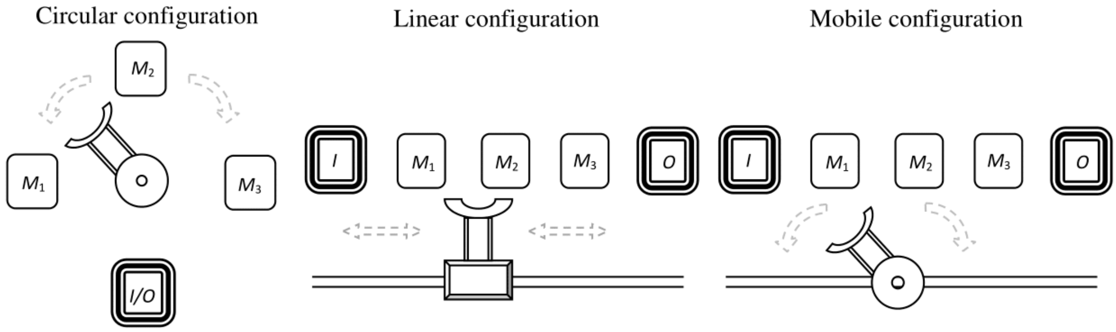


Fig. 1.2. The linearly, circularly and mobile-configured robotic cells.

It should be emphasized that the robot is able to handle only one part at a specific time. Therefore, the fundamental feasibility of robot movement assumptions are as follows: (1) the robot cannot unload an empty production machine; (2) the robot cannot load a busy production machine (Crama et al., 2000). According to these feasibility assumptions, three different robotic cell layouts are researched in the literature: circularly-configured robotic cells where the the stationary base robot rotates on its axis, linearly-configured robotic cells where a moveable robot moves in a rail network and finally mobile-configured robotic cells which are combination of circularly- and linearly-configured cells (Akturk et al., 2005).

Figure 1.2 illustrates these cells configurations with three production machines. On the one side, it reveals a circularly-configured robotic cell is more efficient in term of travel times between adjacent or non-adjacent machines. The intuition behind this is that there are two alternatives, clockwise and counter clockwise, in order to transfer a part between two machines in a circularly-configured robotic cell, whereas there is one way for that in both linearly-configured and mobile-configured cells. For instance, the time for counter clockwise travelling between M_1 and O under the circular configuration is one third of the time for the corresponding robot movement under both linear and mobile configurations.

On the other side, the efficiency in travel in a circular configuration is expensive due to the complexity of the robots hardware as well as its control mechanism. As a consequence, it is needed to weigh up the costs and benefits of the circular configuration. The robot namely Cartesian robot follows straight-line motions in the linear configuration, while a human-like robotic arm is suitable for rotational motions in the circular configuration. The robotic arm consists of both an elbow and a shoulder joint along with a wrist to plan a motion with a certain degrees of freedom. For example, assume that elbow motion takes place as pitch (up and down), shoulder motion occurs as pitch or yaw (left and right), and finally wrist motion occurs as pitch or yaw. Rotation motion can be performed by both wrist and shoulder. This means that the corresponding robotic arm has five to seven degrees of freedom and need a precise control mechanism to transfer parts between machines. It should be emphasized that increasing the degrees of freedom even increases the probability of the robots breakdown, and consequently robotic arms need more corrective and preventive maintenances (Rajapakshe et al., 2011).

As a direct result of this discussion, it is obvious that all circular, linear and mobile configurations can be efficiently employed in practice. Accordingly, a bi-objective trade-off between efficiently of the robot and complexity of the motion planning is necessary to determine how they are changing as employed robot and correspondingly the configuration of the cell are changed. In this thesis, we are interested in a detailed analysis of all configurations of robotic cells.

1.3 Cyclic Scheduling

A key factor in the competitive world of industry is time. Similarly, the time schedule plays an active role in improving the performance in robotic cells. It is, however, recommended that the time schedule be divided into non-cyclic and cyclic based on the parameter settings and the characteristics of the cell. The non-cyclic scheduling is preferable when there is a limited demand of parts, and therefore the robotic cell must operate in intervals between receiving orders to have high-efficiency. Nonetheless, generating a set of cyclic schedules is much more appropriate than the other way if the demand is assumed to be unlimited. More precisely, cyclic scheduling of robot movement occurs if a set of parts is to be implemented an infinite number of times. Although the robotic scheduling problem is reduced to finding the optimal strategy for the robot movement to obtain the maximum throughput (called production rate through this thesis), it is NP-hard, especially for multiple parts production (Brauner et al., 2003).

Due to cyclic behaviour of robotic cells, the long-run average throughput should be maximized to have a steady-state system. Whenever a robotic cell runs at steady state, it means that the recently observed behaviour of it will certainly continue toward infinity. Even this concept can be extended to stochastic robotic cells where the probabilities of repeating various states are constant values. Due to the fact that the steady-state of robotic cells is not reached until a period of time after the cell is started running, we can also call it the transient state (or start-up period). Although steady-state analysis is a crucial component of the robotic cell scheduling process, it is often a short period of time in comparison with the cycle time and can be ignored. Therefore, for sake of convenience, we limit this thesis to the robotic cells running at the steady state. Let the function $F(e_m, j)$ be defined for a cell being composed of m machines to represent the completion time of the jth implementation of event m. This even can be the completion time of the part on the last machine, M_m, and dropping off this completed part at output buffer. As a result of this notation, the long-run average throughput of the cell namely μ is (Crama and van de Klundert, 1997):

$$\mu = \lim_{j \to \infty} \frac{j}{F(e_m, j)} \tag{1.1}$$

Minimizing the long-run average unit cycle time results in maximizing the long-run average throughput, and actually there is a reverse relationship between them. For sake of simplicity, the long-run average unit cycle time is called cycle time through this thesis. One of the distinguishing characteristics of cyclic production is that it has the best performance for mass production. Among different production planning strategies, the most popular class is cyclic production. Therefore, this study is motivated by a robotic cell which cyclically produces parts.

The rest of this chapter provides a detailed classification scheme related to the recent studies in the field of robotic cell scheduling. This classification tree is based on the machine environment, processing restrictions, and objective function in order to cover different configurations of cells and the robot types. Actually, the classification tree is a modified version of classical classification tree, and the distinguishing characteristic of it is that it put emphasis on the stochastic cells.

1.4 A Classification Scheme

In this section, we present a classification tree for robotic cell scheduling problems. As shown in Figure 1.3, this classification tree is based on machine environment (α), processing restrictions (β), and objective function (γ). Due to huge number of recent studies in robotic cell scheduling, it is vital to provide a detailed classification scheme that covers general configurations of cells and the robot types.

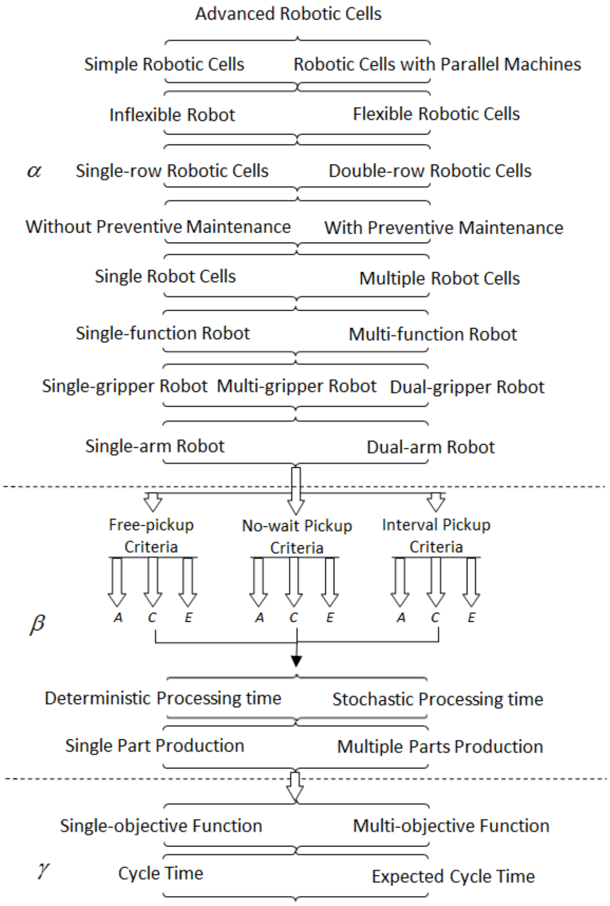


Fig. 1.3. A classification scheme for advanced robotic cells.

It should be noted that each one of these three characteristics can be broken into several subsets . Considering Dawande et al. (2005), we describe these characteristics as a the following form of $\alpha|\beta|\gamma$.

- $\alpha = RRF^{i,j,k}_{l,m,n}(m_1, ..., m_l)$ where the first R, the second R and F show the number of production rows, level of robotics, and the flowshop production environment, respectively. Superscripts are related to robots , whereas subscripts are related to machines. Accordingly, we list them as follows:
  - i: the number of robots
  - j: the number of arms for robots
  - k: the number of grippers for robots
  - l: the number of Workstations
  - m: the number of functions in presence of multi-functionality
  - n: the number of flexible stages

  Note that the vector $(m_1, ..., m_l)$ shows the number of parallel machines at each Workstation. We do not mention this vector when all its elements are 1 and the cell has no stage with parallel machines.
- β = *(pickup, travel metric, uncertainty, part type, production strategy)* where
  - *pickup* can be either $free, no-wait$, or $interval$.
  - *travel metric* can be either $additive, constant$, or $Euclidean$.
  - *uncertainly level* can be either *diterminstic* or *stochastic.*
  - *part type* can be either *single* or *multiple*
  - *production strategy* can be either *cyclic* or *noncyclic.*
- γ is the objective function that can be single- or multi-objective function.

Let us consider two examples to shed light on application of this form.

Example 1: $SRF^{1,2,1}_{3,1,1}$|*free, additive, deterministic, identical, cyclic*|T: α represents a single-row robotic flowshop (SRF) with three machines (l=3) and without multi-functionality (m=1) and flexibility (n=1). One robot (i=1) which is dual-arm (j=2) and has a single gripper (k=1) is applied here. β shows a free pickup, additive travel-time, deterministic data, single part production, and cyclic. γ shows the single-objective function is minimization of the cycle time (T).

Example 2: $DRF^{3,2,4}_{3,2,2}(2,3,1)|$*interval, constant, stochastic, identical, cyclic*$|T,C$: This example shows a more complex cell in which α represents a double-row robotic flowshop (DRF) with six workstations (l=3), two robotic functions (m=2) and flexibility (n=2). Three robots (i=3) which are dual-arm (j=2) and each of them has four grippers (k=4) are employed in here. β shows a interval pickup, constant travel-time, stochastic data, single part production, and cyclic behaviour. γ shows the multi-objective function is minimization of the cycle time (T) and cost (C).

1.4.1 Machine and Robot Environment

There are at least two differences between robots and machines. The first difference is that robots are able to sense and react to the environment to carry out complicated tasks autonomously. The second difference is that robots often perform tasks that a human might do. However, for the sake of simplicity, we analyse physical features of robots and machines in this section, simultaneously.

1.4.1.1 Number of Machines in each Workstation

The robotic cell is named a simple robotic cell when any one of workstations has only one machine. Contrary to this simple case, at least one workstation of robotic cells with parallel machines is over the size of one machine. Cells with parallel machines are more applicable because, in practice, they may improve cycle time by adding an exactly alike machine to a specific workstation which is bottleneck. Such an added machine and existing machine of the workstation should be employed in the cell parallelly (Geismar et al., 2008). Since a part is processed at each workstation according to the predefined processing route, a cell with parallel machines is comparable with a flexible flowshop where there are m workstations. The operation of a part can be completed at workstation i by any one of the m_i similar machines at that workstation. Note that m_i independent machines at workstation i are represented by $M_{ij}, j \in \{0,1,2,3,...,m_i\}$, and have processing time p_i. A cell with $M_{11}, M_{21}, M_{22}, M_{23}, M_{31}, M_{32}$, is in Figure 1.4.

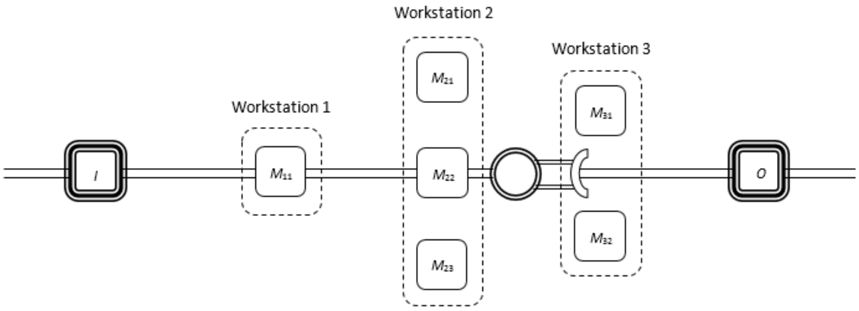


Fig. 1.4. A robotic cell with one, three, and two machines in station 1, 2, and 3.

Using parallel machines will considerably reduce a workstation's effect on the lower bound of cycle time if this workstation contains a small number of production machines whose operation times are meaningfully more than those of the other production machines. In fact, applying m_i parallel machines for these kinds of workstations is the best solution in order to overcome slow production lines. As a result, it is vital to find out how much machines should be added to each workstation in order to have the desired robotic cell.

1.4.1.2 Flexibility of Machines

Inflexible robotic cells are one of the application areas of flowshops where the number of operations is equal to the number of machines, and hence each machine can only perform one of these operations. However, on the one side, we can assume that the robotic cell consists of CNC (or multi-purpose) machines in flexible robotic cells. A CNC machine can handle a mixture of operations. That is, if the part requires a variety of operations (e.g. drilling, milling and broaching), the CNC machine is flexible enough to perform all of them. This ability is obtained from the tool magazine of the machine that rapidly changes tools. On the other side, flexible robotic cells may also have operational flexibility. Operational flexibility means the operations constituting the part can be processed in any arbitrary order. Therefore, it converts the flexible robotic cell into an openshop.

1.4.1.3 Number of Production Lines

The robotic cell can be either single-row or double-row. In the single-row layout, all machines are placed on one side of a central corridor, whereas they are placed on both sides of the central corridor in the double-row layout (please see Figure 1.5). The double-row robotic cell is common in industry and has many applications for production and service facilities although it is more complicated.

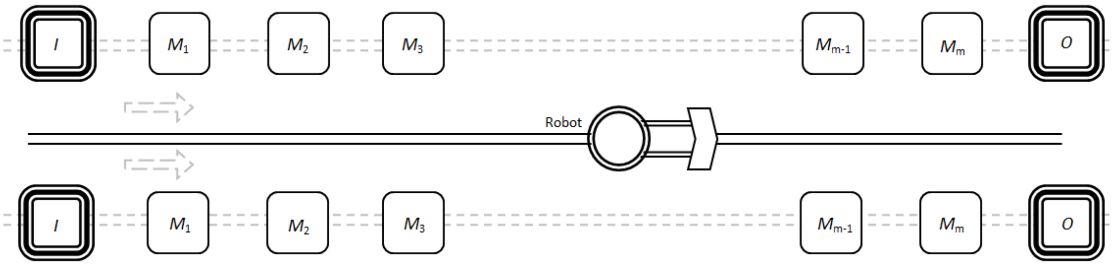


Fig. 1.5. The independent double-row robotic cell.

1.4.1.4 Machine Availability

In addition, we can consider that machines in a robotic cell may be unavailable during some periods which may either be known in advance or not. The known unavailable period means that the machine needs a constant time to maintain after completing a fixed number of parts. The preventive maintenance constraints are applicable for the scheduling problem if unavailable periods are known.

1.4.1.5 Number of Robots in the Cell

Supplemental robots in a cell may significantly improve production rate by increasing the volume which material-handling system can transfer at any given moment. Regarding the layout of the robotic cell, additional robots provides flexibility, especially for cells with large number of workstations. Generally, robotic cells with more than one robot are named multiple robot cells. A multiple robot cell consisting of 16 machines and four robots is shown in Figure 1.6.

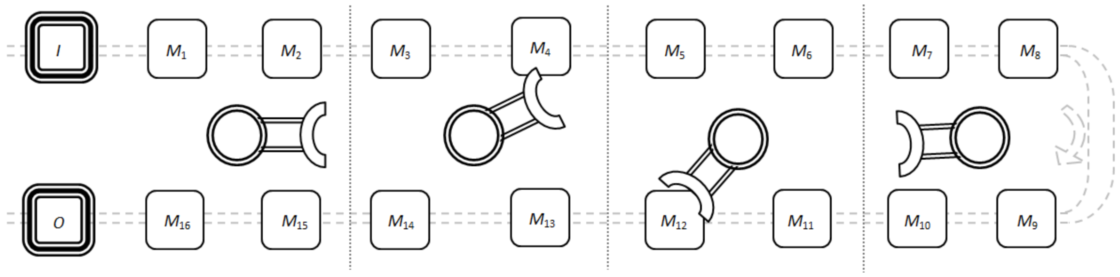


Fig. 1.6. A multiple robotic cell with 16 machines and four robots.

Studies related to multiple robotic cells concentrating on cells in which the robots move to different workstations to perform a variety of assembly tasks by using different tools (Che et al., 2011a). The main purposes of scheduling these multiple robotic cells is to find a sequence of robots movements that minimize the risk of robots colliding with each other when they move parts simultaneously.

1.4.1.6 The Functionality of the Robot

Industrial robots predominantly perform a variety of functions such as welding, assembly, painting, testing and inspection in production environments. For instance, a spot-welding robot can be used for assembling sheet-metals in automotive industry. A painting robot can be applied to minimize the thickness variation of the paint spraying. An inspection robot may reduce testing time and storage requirements, and finally the robots can act as material handling devices. Due to the importance of the last mentioned function of industrial robot, the main focus of this thesis is on the material handing robots.

1.4.1.7 The Robot Capacity for Handling Parts

A robotic arm is classified as single-, dual- or multi-gripper robot which transfer one, two or multiple parts simultaneously. In cells equipped by single-gripper robots, none of the robot and machines is able to process more than one part at any given moment, whereas the robot can handle two parts concurrently in the cells served by dual-gripper robots (Geismar et al., 2008). The robot is also called multi-gripper robot if it is furnished by more than two grippers in order to be capable of handling several parts concurrently. For instance, a three-gripper robot can be instructed to assemble the bottom caps screwing on the back of a flashlight so that the cantered gripper handled lamp holders and springs, and two other grippers handled bails (Quinn et al., 1997). Figure 1.7 shows an example of multi-gripper robots with four grippers which is used to unload parts from I.

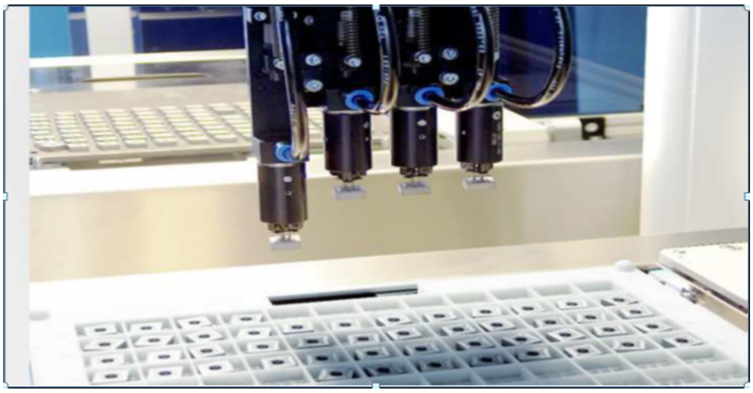


Fig. 1.7. A four-gripper robot which simultaneously handles four parts.

Although multi-gripper end-effectors are more costly to install and complicated by increasing the number of feasible permutations, they can help to meet a specified production rate and improve the flexibility of manufacturing cells. This is due to the fact that the multi-gripper robot can move to an occupied machines in a workstation, uses grippers that containing no part to unload the completed parts, and then load parts which are holding in other grippers. This kind closed-loop activities are rapid enough due to the fact that grippers switching time is negligible in comparison with the required time for inter-machine movement. Therefore, multi-gripper robots potentially maximize production rate in robotized shops that are constrained by the robot's speed.

1.4.1.8 Number of Robotic Arms

The robot can be a single-arm or dual-arm robot. Single-arm robots improve manufacturing cost, and make scheduling tasks simple, whereas the throughput rate of dual-armed robots are much higher than single-arm ones. Dual-arm robots have the ability to load two adjacent machines simultaneously. A simple dual-arm robot loading and unloading parts on machines is shown in Figure 1.8.

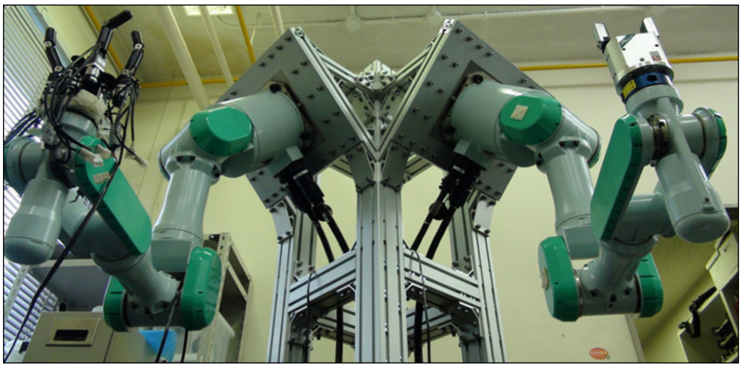


Fig. 1.8. A dual-arm robot using for load and unload operations.

The conclusion from our study in Subsections 1.6.1.1 - 1.6.1.8 supports the notion of a relationship between production rate and manufacturing cost. These subsections make it clear that both production rate and manufacturing cost are strictly increasing function of the complexity of the robot and machine environment. In other words, an advanced manufacturing cell increases the production rate, whereas it leads to higher manufacturing cost which is undesirable. As a consequence, a trade-off between the production rate and manufacturing cost is necessary to determine how they are change as the robot and machines change. Simply increasing the production may not always be feasible or desirable, and therefore we should also make choices regarding the desired level of the production rate.

1.4.2 Processing Restrictions

Another important factor in robotic cells classification is related to the way that parts are processed by machines (or robot). We can structure the processing restrictions as follows:

1.4.2.1 Pickup Criterion

Three major processing scenarios mentioned in Figure 1.3 are free, no-wait, and interval pickup criteria. It should be emphasized that there is no residency constraint on the amount of time a part can stay on a machine after finishing corresponding operation on this machine under the first scenario. The second and third scenarios are the extensions of the free pickup criterion in order to cover all possible conditions. In a robotic cell with no-wait pickup criterion, parts are processed from input buffer to output buffer, with no interruption between machines in the cell. On a more detailed level, the robot unloads the part from a machine without delay, and transfers this part to the next machine as soon as the operation of this part on the machine is finished (Paul et al., 2007). For a robotic cell with interval pickup criterion, each machine has a processing time window for which a part can be processed. In other words, feasible processing time of each operation i is partitioned into $[a_i, b_i]$ for this scenario. There are a huge number of real-world industrial cases that should be classified as a robotic cell with no-wait or interval pickup criteria. They are especially prevalent among: 1) Steel manufacturing where semi-completed parts defect if their temperature drops to a certain temperature, 2) chemical processing tank lines in which the printed circuit boards have to wait on tanks upon finishing of corresponding operation may result in damaged boards.

1.4.2.2 The Distance between Machines

The robot end-effector generally follows a complicated travel path in three dimensions to travel between two points and minimize the cycle time. In addition to selecting the shorten path of the robot, its motion is collision-free and visit multiple task-points considering the inverse kinematics and the obstacle avoidance.

The most addressed travel time metrics are Additive (A), Constant (C), and Euclidean (E). For the additive distance, the robot has to pass through all intermediate machines with a constant speed to move between two machines. Let us assume that the required time for the robot to travel from machine i to machine $i + 1$ is δ. Hence, the time required to perform a direct move between two non-adjacent machines i and j is $|i - j|\delta$ under additive distance assumption. Additive is the most prevailing travel time metric because it is appropriate even when the cell is dense (Brauner, 2008).

If the robot's acceleration and deceleration vary based on the distance between departed and destination machines, the robot travel times between adjacent or non-adjacent machines equals the constant δ. This means that physical distance between all machines is negligible in compact robotic cells fitted into a small space. This case is usually named constant distance. It is also possible to represent the entire robot environment in Euclidean space satisfying the triangle inequalities (Steiner and Xue, 2005). This inequalities is $\delta_{ij} + \delta_{jk} > \delta_{ik}$ if δ_{ij} represents travel time between machine i and j. In fact, Euclidean travel time means that the robot have time to speed up between distant machines so that it is faster than additive one. However, it is not as fast as constant one.

1.4.2.3 The Distribution of Processing Time

In general, processing time is the estimated or actual duration to complete a task. Deterministic processing times indicate that the duration is fixed and that it is known with certainty. In contrast, it is possible that at least one process in a robotic cell has a stochastic processing time. This is very common in practice. Giving an example, in the expose process of microlithography, the desired circuit's image is projected from the photomask onto the wafer. Since the wafers vary in their deviations from perfect flatness, the processing time varies from wafer to wafer (Geismar and Pinedo, 2010). Note that the distribution of the stochastic processing time is an important characteristic of it. Therefore, we need to find it before determining the expected cycle time. Although it is more realistic to analyse a robotic cell with stochastic processing times, it considerably increases the complexity of the scheduling problem.

1.4.2.4 Part Variety

Contrary to single part-type production environments which produce identical parts, multiple part-type cells process lots including a variety of products. As a consequence, anyone of these various parts has its own processing times for a particular production machine. The single part-type production strategy is popular in mass production. However, the multiple part-type production strategy due to the increasing usage of it is small-scale robotized shops (Sriskandarajah et al., 1998b; Hall et al., 1998; Kamoun et al., 1999; Sriskandarajah et al., 1998a; Abdulkader et al., 2013).

The complexity of the scheduling problem considerably increases when a multiple part-type is assumed. The reason behind this complexity is that the robot scheduling problem and part sequencing problem must be jointly solved under this condition. To meet Just In Time (JIT) objectives and reduce in-process inventory and associated carrying costs, the percentage of each part type in batch must have the similar percentage of it in the total demand. Thus, the total demand can be broken into a number of the minimal part set (MPS) (Zarandi et al., 2013). Giving an example of press transfer line equipped by robots, assume the demand of the related automobile company for four different kinds of vehicle doors A, B, C, D be 65%, 15%, 10%, 10%, and the MPS contains 200 doors. Consequently, the number of each one of doors A, B, C, D in MPS is 130, 30, 20, and 20, respectively.

1.4.3 Objective Function

The objective of a linear programming problem can be to maximization or minimization of some numerical values (e.g. makespan, cycle time, and total manufacturing cost). Most academic works on robotic cell scheduling consider cyclic scheduling. Cyclic scheduling has merits such as reduced scheduling complexity, predictable behaviour, improved throughput rate, steady or periodical timing patterns, regulated or bounded task delays and work-in-progress, and reduced variation of flow times (Lee et al., 2007; Lee, 2008). Therefore, the objective of robotic cell scheduling problem is predominantly maximizing the average number of finished parts produced per unit, which is called throughput rate.

1.4.3.1 Multi-Objective Optimization

Robotic cell scheduling problems may be defined as minimizing or maximizing one or more objectives. The first alternative results in a single-objective optimization problem, whereas the second alternative is called multi-objective optimization. The single-objective optimization methodology cannot provide a set of alternative solutions that properly trade multiple objectives against each other. Therefore, there is a basic need to analyse the interaction of multiple objectives if the problem considered multi-objective.

This long-term strategic perspective can help to achieve competitive standing in global markets. For instance, the problem of finding an optimal cycle time is an integral part of production planning, and the problem of finding an optimal quality cost is more important from the quality control point of view (Kumar et al., 2014) . Any one of these problems separately results in a single-objective problem. However, this type of optimization methodology cannot provide a set of alternative solutions that properly trade two objectives against each other. This situation arises, for example, in industries where the demand of the customers is limited and there is no obligation to process parts as fast as possible. Hence, we can accomplish the maximum benefit by considering the optimization problem as a multi-objective one to generate a group of alternative solutions.

1.4.3.2 Expected Cycle Time

The stochastic robotic cell scheduling problems assume that the processing time on M_i is a random variable X_i with the distribution function $F_i, i \in \{1, 2, 3, ..., m\}$. Therefore, the objective of the problem is often minimization of expected cycle time, and the optimal robot move cycle depends on the characteristics of F_i (e.g. its mean and standard deviation).

1.5 Research Goals

As mentioned in Section 1.2, the field of robotic cell scheduling has grown steadily over the past decade. As a direct result, a variety of issues in this field was given saturation coverage by researchers. They provided a wide perspective on some issues related to deterministic robotic cell scheduling paradigm, where the scheduling of the cell is off-line and all data are assumed to be known with certainty in advance. Some common examples of these issues are cell formation, optimal sequence of robot moves, and cost analysis of the cell. Additionally, manufacturers have employed robotic cells in greater numbers and varieties to offer high standard products. As a consequence, researchers have to develop advanced techniques to reduce the cycle time and keep manufacturer's products intensely competitive. As out best knowledge, some issues related to stochastic robotic cell scheduling problem continues to attract the attention of more researchers who are interested in on-line scheduling. Particularly, the problem of a robotic cell with stochastic processing times as well as stochastic processing route is not enough studied in the literature. This problem has often been overlooked in research into robotic cells since it needs more sophisticated studies on stochastic data.

The goal in this thesis is to achieve the minimum cycle time for the scheduling problem of a robotic cell with a stochastic operation like inspection between any two consecutive machines. We carry out the corresponding analysis from two main points of view. The first goal of the study is merely theoretical and aims at relaxing the stochastic problem. In more detail, under this condition, the stochastic data are only recorded by the robot in an independent computer. This means that the result of inspection of the part has no impact of the processing route, and its use is only for recording. Although this relaxation can decrease the hardness of the problem, it is not applicable for many real-life examples where the result of inspection has influence on the processing route, and even cycle time. Subsequently, the second goal of this thesis is to evaluate how the inspection result of a robotic cell can effect on the performance of the cell. Notice that the size of the cell is also considered as an important factor in this study. We conclude that the following problems are extracted from the main goals and resolved in the thesis using some operation research techniques:

- Small-scale deterministic robotic cells with inspection processes
- Large-scale deterministic robotic cells with inspection processes
- Small-scale stochastic robotic cells with inspection processes
- Large-scale stochastic robotic cells with inspection processes

Under the condition when the processing sequence of non-identical parts must and the robot move sequence must jointly be determined, the outcome of this research can still be used after obtaining the optimal parts permutation. Only, it is needed to develop a search method for arranging parts in an optimal (or near-optimal) permutation. Nonetheless, this case is ignored here.

1.6 Thesis Outline

As the final chapter of Part I of this thesis, Chapter 2 briefly reviews the practice-oriented literature concerning robotic cells. We introduce the major research directions in the area of robotic cell scheduling (e.g. steady-state analysis and sensitivity analysis). Following that, we also discuss the complexity of any one of these directions as a prerequisite for the motivation behind this study. The outcome of this chapter is a list of the assumptions made in robotic cell scheduling problems.

Part II includes a description of cells with the inspection processes where the stochastic data are only recorded by the robot in an independent computer. We start this part with a discussion on this topic in Chapter 3 and show how a classical robotic cell can be converted onto a robotic cell with the inspection process. Afterwards, in Chapters 4 and 5, we take into account the problem for two subcases: small-scale and large-scale. The expressions for cycle times are provided in a graphical way to find the efficient cycle under any conditions, and also the conditions in which reduction in cycle time is achieved by the new-developed cell. Finally, operational flexibility of the robot is taken into consideration in Chapter 6.

Part III is devoted to the stochastic scheduling problem without any sort of relaxations for both small- and large-scale problems. It will be discussed how an inspection process should be performed by a real robotic cell, and how much this increase the hardness of the scheduling problem. We even go further toward three different inspection scenarios in Chapters 7, 8 and 9 to make the problem closer to the reality. These inspection scenarios are called post-process, in-process and in-line inspection scenarios depending on the robotic cell where they are employed. We develop an algorithm to solve the problem and find the optimal expected cycle time in this chapter. In order to make the algorithm more effective, we take into consideration different pickup criteria for any one of inspections. Part IV, which has a single chapter, concludes the thesis with some recommendation for future study.

Chapter 2

Backgrounds

In this chapter, we give a brief summary of key research literature findings, and then we highlight some critical issues that should be investigated more in order to develop an approach for increasing the performance of robotic cells. More detailed literature review of each one of these issues appear in later chapters. Therefore, we limit the literature review here and refer the reader to the literature review part of the corresponding published paper given in Chapters 3 to 9.

This chapter places emphasis on the research papers which are relevant to domain of this thesis as described in Section 2.1. The literature review of Section 2.2 is enough for the case where the processing route in this case is fixed although the robot performs the inspection process of the part in the cell. This is due to the fact that all parameters are deterministic under this condition. However, we need additional literature review related to the case in which there is a randomness about the processing route. Consequently, Section 2.3 is assigned to literature review of stochastic robotic cells. The section present a comprehensive and rigorous discussion of computational complexity of stochastic robotic cells scheduling problems which strongly influence on this thesis. Finally, the main motivation behind this research, the linkage of published papers, the problem under study, and solution methods are presented in Sections 2.4, 2.5, 2.6 and 2.7, respectively. This manner of literature review will ensure that the outcome of this chapter can provide a comprehensive coverage of the subject studying in this thesis.

2.1 Problem Domain

There has been extensive research on different issues concerning robotic cells. Quite generally, these research point to the conclusion robotic cells benefit by a careful design, an effective control policy, and a flexible scheduling. Therefore, the domain of the problem in the literature dealing with the robotic cells is:

- The design stage

- The control stage
- The scheduling stage

Firstly, an approach used for new-established robotic cell to design the location of machines and applied operating system. Following that, secondly, it is vital to form computerized control logic for coordinating all movements of robots, machines and parts. This control policy is especially important for the robot since its control mechanism is complex to program and must predict all forward and backward movements between machines. Any fails in the policy may cause the phenomenon of deadlock (or collusion) in the cell. We will specify this phenomenon later on. However, we should employ appropriate tools to capture the detail of the complexity of such a robotic cell which makes us able to determine the cell performance for different configurations and control policies. Note that it is not also straightforward to transit the operating logic contained within the simulation to the actual control policy. As a consequence, policies with a more flexible and easy to use logics are more demanding. Finally, in the last stage, a schedule of the robot (and also parts) involving timings as well as costs of all policies is needed from optimization point of view. If the robotic cell be appropriately fitted with the schedule of the robot movements and parts sequences, we can give a guarantee that the integrated robotic cell system can meet all needs of the customers.

A simple robotic cell, hereinafter called a Single-Function Robotic Cell (SFRC), consists $I, M_1, M_2, ..., M_m, O$ and a Single-Function Robot (SFR) as material handling device. In such a robotic cell, raw materials are fed to M_1, get processed serially through $M_1, M_2, ..., M_m$, and finally come out as completed parts from M_m. Note that SFR operate with a programmable PC-based control platform which allows manufacturers to profit from all the benefits of PC technology. It is vital to distinguish between a SFRC and a Multi-Function Robotic Cell (MFRC) here, and also describe how MFRCs are incorporated into the design of production environments. We impose an operation-oriented extension of SFRCs to introduce MFRCs. Let us assume that the cell is equipped with a Multi-Function Robot (MFR) processing the part in transit between any two consecutive machines. Accordingly, a group of $m + 1$ processes is performed by a MFR in addition to m operations that must be performed by m tandem machines in a MFRC.

From the practical point of view, the use of an advanced robot offers the attractive prospect of an increase in the cell productivity, but there are only a few papers addressing this topic in MFRCs. The purpose of this study is to extend the existing conceptual framework to cutting-edge robotic cells, and to provide some insight and a more efficient use of the scheduling task. Specifically, in this study we introduce a robot with an advanced gripper, called Grip-Gage-Go. The gripper integrates measurement systems into parts handling devices, and makes the robot capable of measuring parts in transit. Grip-Gage-Go grippers are widely employed in the inspection of automotive products including crankshaft, gears, engine valves and lifters in transit. It is the objective of interest to determine the optimal cycle of the corresponding robot assuming two following cases.

Firstly, we limit our study to a robot which only measures the thickness of the part and records results in an independent computer. The processing route in this case is fixed although the robot performs the inspection process of the part in the cell. In contract, secondly, we consider that the user interface computer can be used to modify each the processing route of the part based on the inspection result in order to make this model better represent reality in a stochastic environment.

2.2 Scheduling in Deterministic Robotic Cells

Most researchers in robotic cells have chosen to allocate priority studying deterministic robotic cells so that the analysis of this class of robotic cell forms the main portion of the literature review. There are some literature reviews of deterministic robotic cells, Dawande et al. (2005) and Brauner (2008), but the point is that many new versions of deterministic robotic cells have become apparent recently. Consequently, it is essential to restructure the corresponding literature review in this section for readers who want to update their theoretical knowledge.

It is interesting to mention that most studies focused on only one function of robot, which is material handling. Nonetheless, industrial robots can perform a variety of functions such as welding, assembly, painting, and testing. Specifically, inspection in a cell is one of the important issues in the field of robotic cell scheduling which reflects many real-life cases. Since this thesis concentrating on a kind of robot that can perform inspection and material handling simultaneously, this section mainly addresses issues that arise when considering inspection, laying some important analytical foundations for the under-studied problem.

Let us initially give a brief literature review on robots that act as material handling devices. A few attempts have been made to develop a mathematical model to increase cell performance although many models have been developed to schedule the robot movement (Olabi et al., 2010; Xie et al., 2012; Machmudah et al., 2013). Sethi et al. (1992) developed the theoretical framework for robotic cell scheduling problems. This framework provided a basis for finding the optimal cycle for two-machine cells. Similarly, Crama and de Klundert (1999) obtained a similar result for three-machine cells. However, these results were disproved by Brauner and Finke (2001) for four-machine cells. Furthermore, they performed an analysis of three cell configurations: general cells, regular cells and regular balanced cells. In another work, Levner et al. (1997) extended this structure for the problem where the number of machines is arbitrary and the part finished processing by a machine must be handled to the next machine without delay.

Concerning scheduling of multiple parts production in two-machine dual-gripper robotic cells, an algorithm with the performance ratio of 3/2 was introduced by Drobouchevitch et al. (2004). Dawande et al. (2009) also found a pareto-optimal set for two prevalent robotic cells in practice: a single-gripper cell with a unit-capacity output buffer at each production machine, and a bufferless dual-gripper cell. The problem of scheduling operations in interval robotic cells with dual-gripper robots was studied by Dawande et al. (2010). The robot is called multi-gripper robot if it is furnished by more than two grippers in order to be capable of handling several parts concurrently. Quinn et al. (1997) presented an agile design for a real-life robotic cell with multi-gripper robots. In their study, the whole assembly process was mounted on a three-gripper robot. Levner et al. (2007) studied a special case of the cyclic jobshop model with a multi-gripper robot. Considering a jobshop network, precedence relations between the processing steps for different part types were classified under the form of PERT graphs. They were succeeded in minimizing the cycle time subject to the multiple-part production, the multi-gripper robot traveling time constraint, and time window constraints on the processing times.

Geismar et al. (2012) described all benefits of equipping cells with a dual-arm robot. Then, they determined optimal sequence of cells with two, three, and m production machines. For each one of these cases the productivity of single-arm and dual-arm cells was also compared. Optimal transient periods of dual-armed robotic cells with part delay regulation were determined by Kim et al. (2012). They considered an especial kind of robotic cell namely cluster tool, which is common in semiconductor manufacturing process where the part delays must not surpass a particular upper bound. If the robot unloads this part from machine after the upper bound, the part has poor quality because of residual gases and heat within the workstation. In contrast with other related studies which only assumed the steady state situations, they maximized throughput during transient periods. These periods are time spans in which an unoccupied cell starts to process new parts or completes all operations and becomes empty. They were generally called start-up and close-down periods, respectively.

The literature dealing with the machine features can be generally broken into following different categories: machines with operational flexibility, machines availability, double-row layout of production machines, and part sequence depending machines setup times. One of the first analytic studies on scheduling robotic cell with operational flexibility was performed by Geismar et al. (2005). They restricted themselves to a flexible robotic cell where number of operation was equal to the number of machines and demonstrated that there are $m!$ feasible processing orders for the operations of a part. They found that two permutations have same throughput rate in the two-machine case. In another study, Gultekin et al. (2009) evaluated the effect of the pure cycle on the cycle time. They defined this kind of cycles as modern cycles when operational flexibility is possible. The improved pure cycles, as superior cycles, were introduced by Foumani and Jenab (2013a). They followed flexibility concepts to extend the results for an m-machine flexible robotic cell in a linear and circular configuration. Finally, Jolai et al. (2012) developed an approach based on a new lower bound and a specific pure cycle in order to minimize the cycle time in a circularly configured flexible robotic cell with swap ability. Actually, a robot with swap ability is a robot with a temporary buffer.

The double-row layout robotic cell is common in industry and has many applications for production and service facilities. As a layout problem, Chung and Tanchoco (2010) found out how to place machines on both sides of a central corridor to minimize total material handling cost. They initially developed a mixed integer linear programming model for the double-row layout problem which is more complicated than the single-row layout one. Then, this problem was solved by providing five heuristics, and the performance of these heuristics was compared with each other. Zhang and Murray (2012) highlighted the errors in the formulation proposed by Chung and Tanchoco (2010), and consequently established a modified mixed integer model for the double-row layout problem. They also provided an analytical validation of the corrections. Afterwards, machine sequence, cell formation, and cell layout were jointly taken into account by Chang et al. (2013). A two-phase model was established to combine these three assumptions with regard to production volume, operation sequences, and process routes. One of the direct results of solving this problem was that the double-row layout performance is better than the single-row one which usually used in manufacturing cells.

The separated maintenance constraint may be applicable for flowshop scheduling problem. This means that the machine may be unavailable during the scheduling period, and need a constant time to maintain after completing a fixed number of parts. Regardless of material handling robots, a non-cyclic two-machine scheduling problem with an availability constraint on first production machine was studied by Allaoui et al. (2006) in order to find optimal part sequencing. Also, a similar problem with an availability constraint on both production machines was studied by Yang et al. (2008). The concentration was in a workcell where each one of two machines is unavailable in a predefined interval in order to minimize makespan.

It should be also emphasized that the objective function of scheduling problems can be separately made up of makespan, cycle time, and total manufacturing cost (Che and Chu, 2009) . However, multi-objective function consisting the cycle time and total manufacturing cost is studied in Gultekin et al. (2010); Yildiz et al. (2011). They determined the non-dominated regions for two particular cycles, and then they provided the worst case performance of these cycles for the rest of regions. Robotic cells may have intermediate buffers between machines where these buffers can be modelled as additional machines with zero processing times as show in Drobouchevitch et al. (2010); Geismar et al. (2011). Considering the cycle time, they proved that there is no benefit from providing both machine input and output buffers. An additional reason behind this intuition was that the cost of providing two one-unit buffers at each particular machine is considerably high in comparison with bufferless robotic cells.

The parts are sometimes required to enter a specific machine several times before they are completed, which make the scheduling problem more complex. in a relevant study, Foumani and Jenab (2012) analysed the cycle time of reentrant robotic cells with two machines. They determine both essential and sufficient optimality conditions of each cycle for two different cases: 1) each part only reenters M_1 two times. 2) each part reenters both M_1 and M_2 two times. They also performed sensitivity analysis of parameter to show the influence of each one of them in the regions of optimality for cycles. an interesting result extracted from Foumani and Jenab (2012) is that the reentrant robotic cell is almost theoretically equivalent to a robotic cell with the inspection process. More precisely, if we consider that M_2 is an inspection machine in the reentrant robotic cells with two machines and it always fail the part in the first inspection and accept it in the second inspection, then a few results of the reentrant robotic cells is applicable for robotic cells with inspection machines. However, in the real-life examples, we need to analyse the robotic cells with inspection process separately since there is no guarantee that always the part pass the inspection machine in the second inspection.

As mentioned earlier, robots can perform a variety of functions. There are many studies considering these robot functions separately. For spot welding applications, Zacharia and Aspragathos (2005) solved the problem of determining the optimum route of a spot-welding end-effector visiting a number of task points as a variation of Traveling Salesman Problem (TSP) using Genetic Algorithm (GA). They have implemented the inverse kinematics of a robot as the model for calculating the cycle time. Following this study, Givehchi et al. (2011) applied a new assembly planning method to the sequence optimization of a real-life robotic spot-welding problem for a sheet-metal assembly. This case study involved the optimization of sequences for assembling a partial cabin of a vehicle consisting of 9 sheet-metal parts connecting by 134 spot-welding points. For painting applications, Potkonjak et al. (2000) studied dynamic optimization of a painting robot motion. They focused on the minimization of the manufacturing cost subject to a constraint on painting quality. In their research, it was assumed that a robotized painting shop needs: (*i*) the simulation of the painting operation; (*ii*) modelling of the quality level; (*iii*) definition of cost function; (*iv*) kinematic and technological parameters of the painting task. Afterwards, Diao et al. (2009) developed an optimal motion planning for a painting robot to optimize the sequences of spray-painting end-effector on a free surface. In this study, nonlinear programming techniques took into account position and orientation of the spray-painting device to minimize cycle time. They also minimized the thickness variation of the paint spraying of a specified spatial path. For inspection applications, Wang and Cannon (1996) explored an inspection system interfaced with a robot to reduce testing time and storage requirements. Also, Edinbarough et al. (2005) presented an on-line robotic inspection for identifying ICs lead defect on printed circuit boards.

There are several works in the literature dealing with robotic cell scheduling problems considering two functions. A model for determining the cost of replacing a semi-automated single-function robotic cell with a fully-automated multi-function robotic cell was developed by Geren and Redford (1999). A robot with multi-functional capabilities to assemble and inspect printed circuit boards was used in this fully-automated cell. Bernd et al. (2006) automatized a rework multi-function robotic cell where a high-glossed fitting was inspected and polished by a special multi-function robot. A new approach namely Compound Fabrication was presented by Keating and Oxman (2013) for robotic manufacturing of large printed foam structures. This approach made real both multi-functional and multi-material processes by applying a multi-function robot in cell. The multi-function robot performed both printing and milling, and also was able to shift between manufacturing ways using various grippers.

2.3 Scheduling in Stochastic Robotic Cells

A narrow number of robotic cell studies focused on stochastic robotic cells although robotic cells are subject to a wide range of stochastic issues in a practical sense. Some of these studies are Savsar and Aldaihani (2008); Tysz and Kahraman (2010); Shafiei-Monfared et al. (2009); Geismar and Pinedo (2010). These studies refer to the analysis of machine breakdown, and stochastic processing times, respectively.

Machine reliability plays an important role in the performance of robotic cells. In fact, machine breakdown rate which is a stochastic factor is usually generated from by Weibull or exponential distributions. In this regard, a Markovian stochastic approach was established by Savsar and Aldaihani (2008) in order to evaluate utilization and throughput rate of flexible robotic cells with stochastic occurrence of production machine failures and repairs. In a similar study, sophisticated fuzzy robotic cells with machine reliability consideration were analysed in two phases by Tysz and Kahraman (2010). They obtained a Stochastic Petri Net (SPN) for these cells and then calculated associated fuzzy steady-state probabilities for the reachable markings. Note that a SPN which is also called a Stochastic place/transition net is a mathematical programming language of Stochastic distributed networks.

It is a collection of directed arcs connecting places and transitions. In a SPN, the arcs show which places are pre- and/or post-conditions of each transition.

The cycle time in a virtually arranged robotic cell with stochastic processing time elements was analysed by Shafiei-Monfared et al. (2009). This cell was composed of three machines with increasing (or decreasing) arrangement of stochastic processing times. They assumed that the robotic cell acts like a closed-loop structure, and then replaced this closed-loop structure with a dummy transmittance that links I to O. Finally, they calculated the time of the dummy transmittance by the moment generating function (MGF) of the normal distribution, and proved that it equals the expected cycle time. Furthermore, Geismar and Pinedo (2010) provided an on-line scheduling scheme to determine the expected cycle time of a cell where only one machine has a stochastic processing time with normal distribution. They proved the proximity of the stochastic process to the bottleneck process may have influence on the cycle time, which makes the problem more complicated.

2.4 Motivation

Against the background, the object in this thesis is to not only consider randomness of processing times but also consider randomness of the processing route in MFRCs. Multi-functionality of robots is almost a new objective of interest, both theoretically and in practice. On the one side, the use of an advanced robot which performs multiple functions offers the attractive prospect of an increase in the cell productivity. On the other side, combining multiple functions may decrease manufacturing costs. Accordingly, the purpose of this study is to extend the existing conceptual framework to cutting-edge robotic cells, and to provide some insight and a more efficient use of the scheduling task. Specifically, a narrow number of robotic cell studies focused on both multi-function and stochastic robotic cells, as evidenced by the literature review in the previous section. Therefore, in this study, we introduce a notational and modelling framework for cyclic production in this version of manufacturing cells. Concentrating on the robot cycles, we discuss the issues of feasibility and explore the combinatorial aspects of the problem. The main focus of our study is finding optimal robot move cycle.

2.5 Linkage of Scientific Papers

Table 2.1 shows a chain of papers that follow a long-term trend toward modelling complex MFRCs. All these publications are the first accurate analysis of MFRCs with different categories.

Table 2.1: List of publications produced during probationary candidature

No.	Title	Journal/Conference	Status	Chapter
1*	Scheduling dual-gripper robotic cells with a hub machine	22th IEEE International Symposium on Industrial Electronics	published	3
2	Cyclic scheduling in small-scale robotic cells served by a multi-function robot	39th Annual Conference of IEEE Industrial Electronics Society	published	None
3*	Notes on optimality conditions of small-scale multi-function robotic cell scheduling problems with pickup restrictions	IEEE Transactions on Industrial Informatics	published	4
4	Quantifying the impact of using multi-function robots on productivity of rotationally arranged robotic cells	IEEE International Conference on Industrial Engineering and Engineering Management	published	None
5*	Scheduling rotationally arranged robotic cells served by a multi-function robot	International Journal of Production Research	published	5
6	Cyclic production for robotic cells served by multi-function robots with resumable processing regime	IEEE International Conference on Industrial Engineering and Engineering Management	published	None
7*	Increasing throughput for a class of two-machine cell served by a multi-function robot	IEEE Transactions on Automation Science and Engineering	published	6
8	Stochastic Scheduling of a Two-Machine Robotic Cell with In-Process Inspection	the 45th Conference on Computers and Industrial Engineering	published	None
9*	Resolution of deadlocks in a robotic cell scheduling problem with post-process inspection: avoidance and recovery scenarios	IEEE International Conference on Industrial Engineering and Engineering Management	published	7
10*	Stochastic Scheduling of a Two-Machine Robotic Cell with In-Process Inspection	Journal of Computers and Industrial Engineering	Submitted	8
11*	Two-machine robotic rework cells: in-process, post-process and in-line inspections	Omega, The International Journal of Management Science	Submitted	9

The table starts from modelling a SFRC with a multi-purpose machine acting similar to a MFRC, and considering it as origin of MFRCs in the first paper. Then, a set of four studies on deterministic MFRCs with small and large scales are presented. The study on deterministic MFRCs ends up taking into account MFRCs with flexibility in the following two papers. The last four papers are related to stochastic MFRCs with different inspection scenarios. Also, the linkage of paper mentioned in Table 2.1 is graphically shown in Figure 2.1.

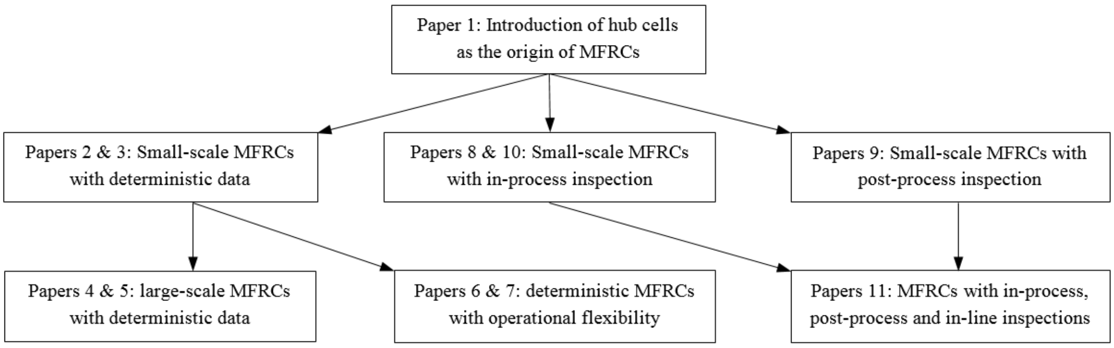

Paper 1: Introduction of hub cells as the origin of MFRCs
Papers 2 & 3: Small-scale MFRCs with deterministic data
Papers 8 & 10: Small-scale MFRCs with in-process inspection
Papers 9: Small-scale MFRCs with post-process inspection
Papers 4 & 5: large-scale MFRCs with deterministic data
Papers 6 & 7: deterministic MFRCs with operational flexibility
Papers 11: MFRCs with in-process, post-process and in-line inspections

Fig. 2.1. The linkage of paper mentioned in Table 2.1.

For simplicity, key papers shown by * will detail in the following chapters.

2.6 The Scheduling Problem under Study

Initial results in previous section are based on deterministic data because it is always realistic to find a deterministic model for a typical robotic cell. Nonetheless, modern robotic cells may have technical issues like inspection processes. In other words, considering inspection stages in a robotic cell is one of the important issues in the field of scheduling which reflects real-life cases, and therefore the final results of previous section are related to inspection process in cells. Three classifications of inspection, according to where and when they are made, are:

- Post-process: placing the part in an independent inspection machine
- In-process: inside the machine during the machining process
- In-line: inspect the part on a MFR as soon as a part comes off the machine

The *post-process inspection* measures the part outside the production machine. Normally, it includes inspection on an independent inspection machine. *In-process inspection* includes production machines supplied with sensors. These sensors measure the part while it is being processed. Normally, the inspecting sensor has no connection to the production mechanism. Finally, *in-line inspection* measures the parts immediately after a process is completed. For instance, it is possible to measure the diameter of a part by a MFR as soon as it comes off a grinder.

Having post-process inspection scenario, a m-machine rework SFRC is commonly captured by the succeeding framework: this cell is made up of a sub-production line of m production machines $(M_1, M_2, ..., M_m)$, a sub-inspection line of $m + 1$ measurement machines $(N_0, N_1, N_2, ..., N_m)$, a gantry robot that serves the entire production line, an input buffer I and an output buffer O with unlimited storage capacity. This makes it clear that SFRCs (see Section 2.1) are a subdivision of these cells where there is no inspection machine in the cell.

Deterministic modelling of a rework SFRC is sometimes unrealistic because it is possible that some parts unloading from I need to be reprocessed before reaching O. In fact, there is an inspection machine just after each production machine in real-production situations. SFR unloads the processed part from the production machine, and then loads it on the next corresponding inspection machine. The part on the inspection machine takes a certain amount of time to be tested for defects. If it has no defect, SFR transfers it to the next production machine. However, it is possible that the part needs reprocessing and SFR transfers it to the previous production machine for reprocessing under this condition. Figure 2.2 shows a SFRC with two production machines and three inspection ones. The first inspection machine N_0 is carried out to check the raw material for its acceptance or error based on acceptance criterion. The second inspection machine N_1 guarantees that the work in process between two production machines satisfies the certain acceptance criteria and finally the third inspection machine N_2 controls the quality of finished part to be delivered to the customer.

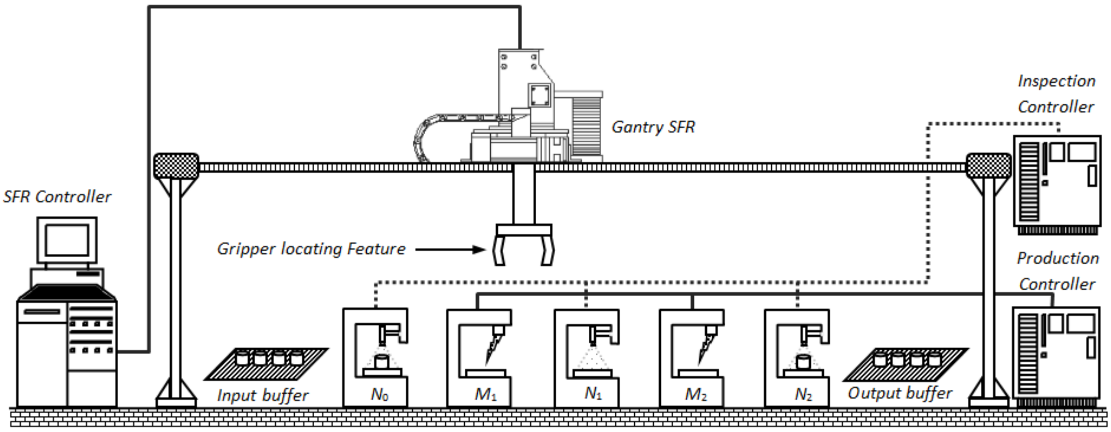


Fig. 2.2. A SFRC with machines M_1, M_2 and inspection machines N_0, N_1, N_2.

Having in-process inspection scenario, the cell is made up of a sub-production line of m production machines $(M_1, M_2, ..., M_m)$ and have no inspection machine because any one of machines has a set of independent sensors for inspection during the process. As a consequence, the part on the particular production machine M_i takes $max\{P_i, T_i\}$ to be processed and inspected (P_i is time taken to process part on M_i, and T_i is time taken by sensors to do inspection in process). At this point of time, SFR transfers this part to the next production machine if it has no defect. However, it is possible that the part needs reprocessing which means SFR cannot transfer the part to the next machine at least for one more period of $max\{P_i, T_i\}$.

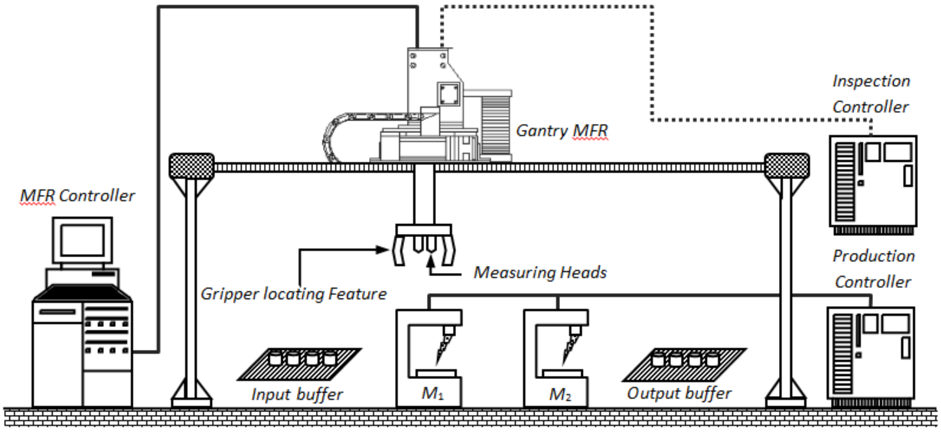


Fig. 2.3. A MFRC with production machines M_1, M_2 and in-line inspection system.

For in-line inspection scenario, there are MFRCs in addition to SFRCs. As mentioned before, the distinguished feature of a MFRC is a MFR which combines inspection processes into material handling system. The cell shown in Figure 2.3, is an example of using this kind robot instead of the SFR in the Figure 2.2. There is no need for the MFRC to have any inspection machines, and the measuring head of MFR acts as an inspection device during transition of parts between machines. Note that no-wait and interval pickup make some MFR move infeasible to execute, and hence both feasibility and optimality conditions of the cell are studied here.

2.7 Solution Method

The scheduling of robotic systems is complex both computationally and analytically due to the combinatorial explosion in the number of robot movements. Regarding computational methods, branch and bound algorithms are applied for solving the problem in addition to some evolutionary algorithms such as the genetic algorithm and the simulated annealing algorithm. The point about evolutionary algorithms is that they are problem-independent methods. As such, they do not take advantage of any specificity of the problem and this may be the cause of weaker results (Zarandi et al., 2013). Also, although they can find a good cycle in a reasonable amount of time, there is no guarantee that they can find an optimal cycle. Likewise, there are computationally effective heuristics such as Heuristic MCell (Dawande et al., 2005) to solve industry-sized problems, but they only have an approximation guarantee on the obtained solutions. In this thesis, we focus on analytical methods to find exact algorithms for robotic cell scheduling problems. The advantage of exact algorithms is that they are designed in a way that will certainly find the optimal solution in a finite amount of time. However, for difficult problems (e.g. NP-hard) this finite amount of time may increase exponentially with respect to the dimensions of the problem. Consequently, we also establish some heuristics under these conditions to find near-optimal solutions.

Since the part processing routes in large-scale MFRCs is complicated, one of the most economic strategies in order to deals with this complexity is breaking these MFRCs into small-scale clusters. Then, a MFR serves within each one of newly designed clusters which generally consist of one, two or three machines. This gives us an indirect method to optimize a particular large-scale MFRC. Although this method predominantly works for large-scale MFRCs, it is better to find a method to directly analyse them. As a result, we also derived upper and lower bounds of cycle times, and then developed an exact method that divides all solution into two groups: feasible and infeasible. These solutions are on top and down of the constant value of lower bound. If a solution was infeasible, it is ignored. If not, the nearest one to the lower bound is the optimal cycle, as shown in Figure 2.4.

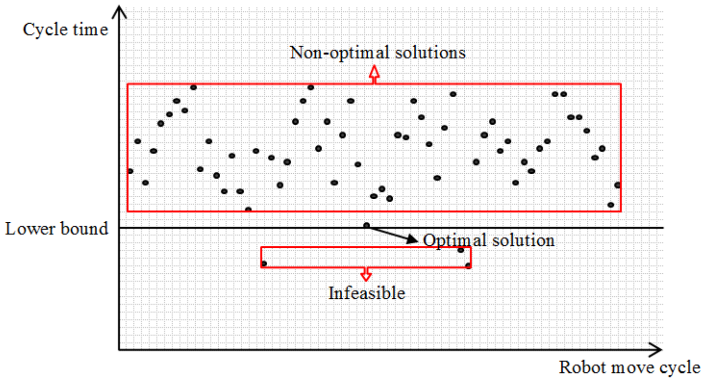


Fig. 2.4. A graphical scheme of lower bound and all solutions.

At the end, the large-scale MFRCs are modelled as a TSP. Actually, the motivation behind modelling MFRC scheduling problem as a TSP is that modelling MFRC scheduling problems as a TSP gives computational benefits due to the existing solution methods. Also, as a design problem, a preliminary analysis given in this thesis identifies the regions where the productivity gain of a regular MFRC is more than that of the corresponding SFRC for both small- and large-scale cells. This strong tool helps robotic cell designers to make the final decision as to whether they should invest money in purchasing a MFRC or not. It should be stressed that models adopted in this research are predominantly related to linearly-configured cells. However, the analysis can be easily extended to other cell layouts. The extension of results to a circularly-configured (or mobile-configured) robotic cell is as trivial as adding a linear unit of time to the robot travel time.

With regard to stochastic MFRCs, three classifications of inspection scenarios face stochastic environments and this makes it vital to use stochastic models. To the best of our knowledge, the set of techniques employed to find a solution for similar robotics scheduling problems has so far included heuristics, Petri Nets, critical path method, formulating as a multi-path TSP with stochastic travel costs, simulation, dynamic programming. The description of them is as follows: 1) Heuristics applied in robotic cells are methods that do not guaranteed optimality, but are suitable computationally when determining an optimal cycle is impractical. 2) A Petri Net is a collection of directed arcs connecting places and transitions. Places (machines or robots) may hold tokens (parts). The state or marking of a net is its assignment of tokens to places. 3) Critical path method is based on the length of the robotic networks longest path, named the critical path. 4) For TSP-based approaches, each part is a city, and the distance between a pair of cities is the duration of the interval between the corresponding consecutive occurrences of the robot activities. 5) Simulation is the imitation of the operation of a real-world robotic cells over time. 6) Dynamic programming in robotic cells employ dispatching rules to determine the next action of the robot based on the current state of the cell. The computation of the partial cycle time via the above approaches, which can be divided into graphical and algebraic approaches, are often high as reviewed in Dawande et al. (2005). Thus, there is a need for some stronger approaches to be developed in this thesis. From our point of view, the following strategies are essential for tackling the stochastic version of robotic cells:

- The deadlock problem is potentially one of the critical issues in designing robotic systems. Under this assumption, we discuss the problem of testing whether a cell can deadlock with the given inspection system. So, under ideal condition, we find a strategy such as giving constraints to prevent deadlock for any sequence of robot movements and make it deadlock-free.
- For some cells, it is possible to resolve deadlocks considering a penalty (e.g. time gap or cost) with associated probability of occurrence. Penalty of deadlock may be high or low, and also deadlock can occurs frequently or rarely. Accordingly, penalty and risk of deadlock are two indexes belong to the level of deadlock. We assume different level of deadlock based on these indexes.
- Performance measures are essential to be added to analyse of cell before any quantitative evaluation. A performance measure from robotic cell designer's perspective is an expression that can contain fundamental measures such as $E\{C_t\}$ which mentions the expected value of cycle time. Accordingly, we assume a minimization problem of $E\{C_t\}$.
- The dynamic behaviour of cell is determined by the activation and execution of the robot actions, which in turn change the robotic cell state at any given moment. So, it will be a noteworthy achievement if we establish simple heuristics for dynamic scheduling, and then determine the worst-case performance of heuristics, relative to the optimal solution for inspection scenarios.

The methodology developed in this study is based on defining three kinds of stochastic order relations, sorting them through strangeness, and then finding the dominancy regions of feasible cycles by an analytical method. We consider all issues above in our methodology, and therefore, the conclusions of solution methods can be helpful for both managers of companies which use modern robots in automated manufacturing system and international manufacturers of industrial robots who generally try to design agile robotized shops. To our best knowledge, none of the literature produced has provided an answer to the question that how the operations must be allocated to the machines and the corresponding robot must be instructed to minimize the cycle time under aforementioned assumptions. Therefore, the novelty of research is that it tries to answer to this question and enhances performance of MFRCs.

Part II

Deterministic Modelling

Chapter 3 is based on the published article Foumani, M., Ibrahim, M.Y., Gunawan, I., 2013. Scheduling dual gripper robotic cells with a hub machine. 2013 IEEE International Symposium on Industrial Electronics (ISIE), On pages: 1 - 6.

Abstract *This paper introduces a new methodology to optimize the cycle time of dual-gripper robotic workcells. The workcell under study is composed of a group of* m *production machines. In order to produce a completed part, a chain of* $m-1$ *secondary operations are performed by* $m-1$ *different machines, and a hub machine is alternately visited for* m *primary operations. Indeed, parts must reenter the hub machine after any one of secondary operations. Those types of robotics workcells are used for high capacity production such as in photolithography manufacturing, These cells are cluster tools for semiconductor manufacturing where a wafer visits a processing stage several times for the atomic layer deposition (ALD) processes. For electroplating lines, these cells are also common in practice where there is a multi-function production stage that is visited by parts over once. This optimization methodology is limited to the dual-gripper robotic cells, where identical parts are produced and these parts completely follow a similar sequence. The lower bound of cycle time for such dual-gripper robotic cells is obtained by finding the cycle time of all related robot move cycles, and subsequently optimal robot task sequence, which is a two-unit cycle, is determined.*

Note to Practitioners *Considering real-life applications, this study in this paper provides a framework for converting a special case of production system in semiconductor manufacturing facilities with hub machine into a robotic system with higher performance. Applications of such cells can be found in several production environments, particularly in advanced technologies, such as in the Atomic Layer Deposition (ALD) process where film deposition is manufactured with the accuracy of mono-atomic layers.*

Keywords *Reentrant robotic cell, Cyclic production, Dual-gripper robot, Hub machine, Scheduling*

Classification $SRF^{1,1,2}_{l,1,1}|\textit{free, additive , deterministic , identical, cyclic}|T$ with hub

Note References are considered at the end of the thesis.

Chapter 3

Dual Gripper Robotic Cells with a Hub Machine

3.1 Introduction

The main attainments in using industrial robots include high-quality products, lower manufacturing cost, safety of workers, and satisfying flexibility (Gao et al., 2009). A significant application of robots in industries is typically their use as material handling devices in robotic work cells. Figure 3.1 depicts the movement of a part in such a robotic cell, where parts are identical and the input/output buffer (I/O) holds both raw materials and finished parts. Both artificial machine M_0 and M_{m+1} can also denote I/O. Each part being processed passes sequentially from I/O to production machines $M_1, M_2, ..., M_m$ and eventually to I/O.

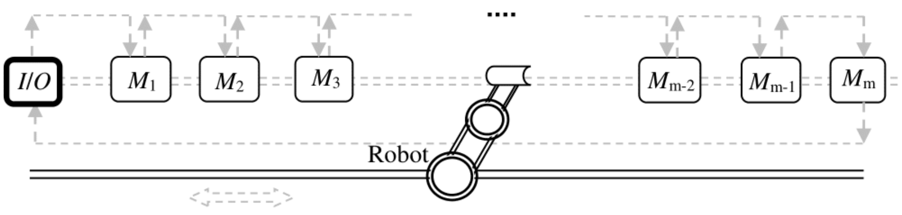


Fig. 3.1. A classical robotic cell with m machines

The hub reentrant robotic cell, which is a practical case of classical robotic cell, is considered in this paper. Each part enters alternatively a particular hub machine to be considered as a completed part in this kind of robotic cell, which is shown in Figure 3.2. The hub reentrant robotic cells are totally different from the classical robotic cells. The most important difference between these cells is that a series of sequential operations denoted by $O_1, O_2, O_3, , O_m$ must be performed by

m machines in the classical robotic cells, whereas a set of processes denoted by $G_1, O_2, G_2, O_3, G_3, O_4, , G_{m-1}, O_m, G_m$ must be performed by the hub machine M_1 and $m-1$ machines $M_2, M_3, ..., M_m$ in the hub reentrant robotic cells to produce a completed part. $O_i, i = 2, , m$, is the index of the secondary operation on M_i. Also, $G_h, h = 1, , m$, is the index of h^{th} primary operation on M_1. Because of entering alternately the hub machine M_1, it plays a more active role in the hub reentrant robotic cells in comparison with the classical ones, and performs m primary operations instead of only one operation. Consequently, the numbers of necessary operations for producing one part in the classical and hub reentrant robotic cells are given by m and $2m - 1$, respectively.

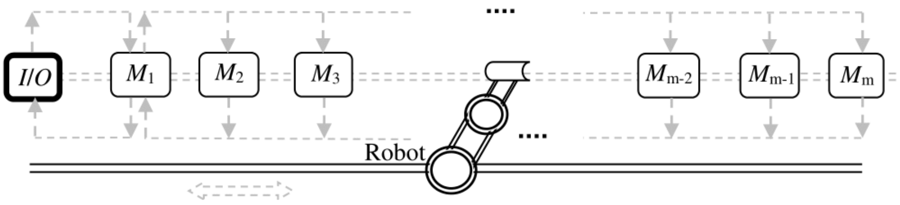


Fig. 3.2. A hub reentrant robotic cell with m machines

A great deal of exact methods has been suggested for solving classical robotic cell scheduling problem. Some researchers have applied meta-heuristic algorithms to solve this problem (e.g. Li et al., 2010; Yan et al., 2012). Although the majority of researches on robotic cell focus on determining optimal robot moves for classical robotic cells, no study was found where the robot performs material handling of a hub reentrant cell. Applications of such cells can be found in several production environments, particularly in advanced technologies, such as in the Atomic Layer Deposition (ALD) process where film deposition is manufactured with the accuracy of mono-atomic layers. This technology improves the quality of wafers by controlling their thickness, and consequently redoing ALD processes several times. If these ALD processes are performed at a single hub machine, it will certainly increase the wafer quality. This is due to the existence of the same processing conditions for all repeated processes and the more economical application of a single hub machine (Lee and Lee, 2006). These cells are also used in some electroplating processes, as reported by Liu et al. (2002). Due to the inherent nature of the printed circuit boards (PCBs) electroplating, there is predominantly a hub production stage that is visited by parts several times. Other applications of the hub reentrant robotic cells are in painting shops and Very Large Scale Integrated (VLSI) circuit wafer manufacturing (Kubiak et al., 1996). Consequently, there is a wide gap between theoretical and practical studies on the hub reentrant robotic cell, which has been used in different industries.

Two main types of robots end-effectors, generally seen in manufacturing cells, are single-gripper and dual-gripper. A single-gripper robot manipulator only has one gripper, which is able to hold at most one part at any particular time. However, a dual-gripper robot manipulator has two grippers, either of which is able to transfer a part at any given moment. As a result, a dual-gripper robot has a capacity for carrying two parts. This difference between these two kinds of robots makes it clear that the dual-gripper robots, which are the focus of this paper, increase productivity and decrease the cycle time in comparison with the single-gripper robots (Drobouchevitch et al., 2006).

The plan of the paper is as follows: A chronological literature review pertaining to different kinds of robotic cells is presented in Section 3.2. The associated definitions and notation of cyclic production issues are described in Section 3.3. In Section 3.4, the lower bounds of robot move cycles are determined for a robotic cell consisting of a dual-gripper robot, $m-1$ machines, and a hub machine. Additionally, optimal robot task sequence is determined in this section. The conclusion and future research directions are generalized in Section 3.5.

3.2 Related Research

Many researchers concentrated on the dual-gripper robotic cell scheduling problem. Su and Chen (1996) were the first to consider a simple two-machine robotic cell with a dual-gripper robot. Later, Sethi et al. (2001) presented a more extended analysis regarding number of possible robot move cycles for a dual-gripper robotic cell composed of two machines. The problem of cycle time minimization in a multi-robot serial system was addressed by Galante and Passannanti (2006).

Recent studies have concentrated on dual-gripper robotic cells with different characteristics. Dawande et al. (2009) found a group of dominant solutions for two prevalent robotic cells in practice: a single-gripper cell with a unit-capacity output hopper at each production machine, and a bufferless dual-gripper cell. The problem of scheduling operations in interval robotic cells with dual-gripper robots was studied by Dawande et al. (2010). Productivity improvement from using machine buffers in dual-gripper robotic cells has recently been an interesting topic. For instance, in Geismar et al. (2011), the quantity of improvement in productivity is expressed when unit-capacity input and output buffers at each one of machines is assumed.

It should be also emphasized that most of the pertaining literature to the single-gripper robotic cell scheduling problem were based on the assumption that a busy robot cannot load an occupied machine. However, some researchers recently assumed that both robot and machine can be busy to load or unload a part onto a production machine, which is called swap ability. As a matter of fact, the robot can be used as a temporary stock with one unit capacity in the manufacturing cells. For the first time, a Genetic Algorithm (GA) was presented in Soukhal and Martineau (2005) to minimize the makespan of a robotic cell with swap ability. Jolai et al. (2012) also developed a new approach based on a particular kind of robot move cycle to obtain minimum cycle time of flexible robotic cells with swap ability. Eventually, these results were extended in Foumani and Jenab (2013a). They introduced a better-quality robot move cycle, improved pure cycle, and proved that it always decrease the cycle time and number of production machines, simultaneously.

The problem of minimizing mean flow time in a hub reentrant cell was studied in Kubiak et al. (1996). In contrast with the framework of this study, they assumed that there is not any material handling devices in a non-cyclic production system. Furthermore, the objective of their problem was to minimize the total flow time, not cycle time. The reentrant cell scheduling problem was first studied in Steiner and Xue (2005) in the context of cyclic production. The optimal schedule of a two-machine reentrant robotic cell was found in Che and Chu (2009) to minimize the makespan. Lately, Foumani and Jenab (2012) analyzed a two-machine reentrant cell under consideration of swap ability.

3.3 Problem Notation and Definitions

In addition to basic definitions and notation for robotic cell scheduling problems, we describe the related concepts used throughout the paper to give a conventional description of the reentrant cell under investigation. The following definitions, which are relevant to this study, are developed:

Definition 1. Considering a reentrant robotic cell made up by a hub machine and $m-1$ machines, an n-unit robot task sequence is a steady state cycle where n parts are completed. For an n-unit cycle, each one of $m-1$ machines is loaded and unloaded exactly n times, and the hub machine is loaded and unloaded exactly $n \times m$ times.

The state of the robotic cell is defined by the position of the dual-gripper robot and whether the machines and this robot are occupied or empty at any given time. In term of robot motions, it should be emphasized that, after loading a part onto a machine, the robot can wait at the machine to complete its processing on the part or move to another machine.

Definition 2. For a reentrant robotic cell with a dual-gripper robot, a hub machine, M_1, and $m-1$ machines, the state transition in a cycle is shown with

$E = \{e_1, e_2, e_3, , e_m|e\}$ where $e_j = \{\Omega_j, \phi_j\}$ and $e = \{A_{i+}, A_{i-}, B_{h+}, B_{h-}\}$. Ω_j and ϕ_j indicate that $M_j (j = 1, ..., m)$ is either occupied or empty. A_{i+} and A_{i-} indicate that the robot either unloads or loads $M_i (i = 0, 2, 3, ..., m)$, respectively, and finally B_{h+} and B_{h-} mean that the robot either unloads or loads M_1 for h times $(h = 1, 2, ..., m)$.

Figure 3.3 shows an example of a three-unit cycles execution. In this example, there are a hub machine which performs the operations G_1, G_2 and G_3, and M_2 and M_3 which perform the secondary operations O_2 and O_3. Hence, the cell follows the sequence of operations G_1, O_2, G_2, O_3, G_3. Note that the arcs $1, 2, ..., 18$ show the sequence of the robot movements. It is clear that each one of machines M_2 and M_3 is loaded and unloaded exactly three times, and the hub machine is loaded and unloaded exactly nine times. The number of completed parts is equal to three since the robot drops off three parts at I/O, respectively, after movements 10, 14, 18.

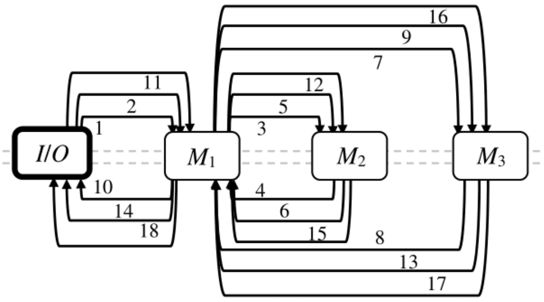


Fig. 3.3. The state transition of a three-unit cycle for a three-machine cell

The corresponding state transition in this cycle is:

$$
\begin{aligned}
&\phi_1, \phi_2, \phi_3|A_{0+} \rightarrow \Omega_1, \phi_2, \phi_3|B_{1-} \rightarrow \Omega_1, \phi_2, \phi_3|A_{0+} \rightarrow \phi_1, \phi_2, \phi_3|B_{1+} \rightarrow\\
&\Omega_1, \phi_2, \phi_3|B_{1-} \rightarrow \Omega_1, \Omega_2, \phi_3|A_{2-} \rightarrow \Omega_1, \phi_2, \phi_3|A_{2+} \rightarrow \phi_1, \phi_2, \phi_3|B_{1+} \rightarrow\\
&\Omega_1, \phi_2, \phi_3|B_{2-} \rightarrow \Omega_1, \Omega_2, \phi_3|A_{2-} \rightarrow \Omega_1, \phi_2, \phi_3|A_{2+} \rightarrow \phi_1, \phi_2, \phi_3|B_{2+} \rightarrow\\
&\Omega_1, \phi_2, \phi_3|B_{2-} \rightarrow \Omega_1, \phi_2, \Omega_3|A_{3-} \rightarrow \Omega_1, \phi_2, \phi_3|A_{3+} \rightarrow \phi_1, \phi_2, \phi_3|B_{2+} \rightarrow\\
&\Omega_1, \phi_2, \phi_3|B_{3-} \rightarrow \Omega_1, \phi_2, \Omega_3|A_{3-} \rightarrow \phi_1, \phi_2, \Omega_3|B_{3+} \rightarrow \phi_1, \phi_2, \Omega_3|A_{0-} \rightarrow\\
&\phi_1, \phi_2, \Omega_3|A_{0+} \rightarrow \Omega_1, \phi_2, \Omega_3|B_{1-} \rightarrow \phi_1, \phi_2, \Omega_3|B_{1+} \rightarrow \phi_1, \Omega_2, \Omega_3|A_{2-} \rightarrow\\
&\phi_1, \Omega_2, \phi_3|A_{3+} \rightarrow \Omega_1, \Omega_2, \phi_3|B_{3-} \rightarrow \phi_1, \Omega_2, \phi_3|B_{3+} \rightarrow \phi_1, \Omega_2, \phi_3|A_{0-} \rightarrow\\
&\phi_1, \phi_2, \phi_3|A_{2+} \rightarrow \Omega_1, \phi_2, \phi_3|B_{2-} \rightarrow \phi_1, \phi_2, \phi_3|B_{2+} \rightarrow \phi_1, \phi_2, \Omega_3|A_{3-} \rightarrow\\
&\phi_1, \phi_2, \phi_3|A_{3+} \rightarrow \Omega_1, \phi_2, \phi_3|B_{3-} \rightarrow \phi_1, \phi_2, \phi_3|B_{3+} \rightarrow \phi_1, \phi_2, \phi_3|A_{0-}
\end{aligned}
$$

Due to the fact that potential deadlock exists, the robot is not able to perform all sequences. Drobouchevitch et al. (2006) provided the following assumptions to insure the feasibility of a robot move cycle: 1) an empty machine cannot be unloaded, 2) a loaded machine cannot be loaded again, 3) the dual-gripper robot cannot hold more than two parts at any given moment, or equally, the robot must at least has one empty gripper if unloading a new part is its next motion. To give an example, $\phi_1, \Omega_2, \phi_3, ..., \Omega_{m-1}, \Omega_m | B_{2+} \rightarrow \phi_1, \Omega_2, \Omega_3, ..., \Omega_{m-1}, \Omega_m | A_{3-} \rightarrow \phi_1, \Omega_2, \Omega_3, ..., \Omega_{m-1}, \Omega_m | B_{4+}$ is impracticable since the robot unloads a part from the hub machine (B_{2+}) and transfers and loads it to M_3 (A_{3-}). Thus, it is impossible to unload a part from the empty hub machine (B_{4+}). The following notations were made for analysis of the reentrant cell:

ε The load (or unload) time of machines or I/O.

δ Time taken by the dual-gripper robot to move between two sequential machines.

d_{ij} Time for transferring a part between two non-adjacent machines i and j.

θ The time taken for switching the robot grippers position with each other.

p_i The processing time of parts on $M_i (i = 2, , m)$.

H_h The processing time of parts on the hub for the h^{th} time, $(h = 1, ..., m)$.

T_j The cycle time of robot move cycle j to complete a part.

$\underline{T}_d$ The lower bound for the robot move cycle to complete a part.

In this study, the robotic cell scheduling problem is also solved under the following assumptions:

- Presuming that the robots travel time is additive, d_{ij} is always equal to $|i - j|\delta$.
- It is assumed that the switching of robots gripper is performed as fast as possible ($\theta \leqslant \delta$ and $\theta \leqslant P_i$), which is predominantly true in real-life application of robotic cells (Drobouchevitch et al., 2006; Sethi et al., 2001; Geismar et al., 2011; Jenab et al., 2012).
- The analysis is limited to the situation where all $m - 1$ secondary operations are negligible in comparison with all m primary operations. This assumption, which is occasionally called bottleneck assumption, is common among the pertaining literature of scheduling for many industrial processes such as painting and VLSI wafer fabrication (Foumani and Jenab, 2013a). The bottleneck assumption is shown as below:

  $max_{2 \leqslant i \leqslant m} P_i \leqslant min_{1 \leqslant h \leqslant m} H_h$

Note that $\theta \leqslant H_h$ since $\theta \leqslant P_i$ and $max_{2\leqslant i\leqslant m}P_i \leqslant min_{1\leqslant h\leqslant m}H_h$.

3.4 Lower Bound and Optimal Cycle for Hub Reentrant Robotic Cells

The scheduling problem data is composed of the gripper's rotating time, the load (or unload) time of machines, the travel time for the dual-gripper robot, the number of machines and processing times on different machines. Depending on the data, the lower bound must be estimated in order to find optimal cycle. If a particular solution attains this lower bound, it is certainly the optimal robot task sequence. Thus, we limit the problem to finding the optimal robot task sequence in order to fulfill the lower bound for cycle time. This lower bound is calculated by the following theorem:

Theorem 1. For a hub reentrant cell consisting of a dual-gripper robot, a chain of $m-1$ machines, and a hub machine which follows the sequence of operations $G_1, O_2, G_2, O_3, G_3, O_4, ..., G_{m-1}, O_m, G_m$, the lower bound of the per unit cycle time is:

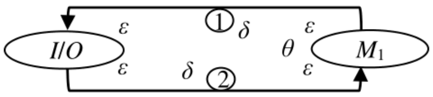
$$\begin{aligned}\underline{T}_d = &\frac{1}{2}(max\{4\varepsilon+2\delta+\theta, H_1+2\varepsilon+\theta\}) + max\{4\varepsilon+2\delta+\theta, H_1+2\varepsilon+\theta\}\\ &+\textstyle\sum_{h=2}^{m-1}(max\{4\varepsilon+2(h-1)\delta+2\theta, H_h+2\varepsilon+\theta\} + max\{4\varepsilon+2h\delta+2\theta, H_h+2\varepsilon+\theta\})\\ &+max\{4\varepsilon+2(m-1)\delta+2\theta, H_m+2\varepsilon+\theta\}+max\{4\varepsilon+2\delta+\theta, H_m+2\varepsilon+\theta\} \qquad (3.1)\end{aligned}$$

Proof: The proof of this theorem is given in three sections.

1. The first portion of the equation.

   1.1. Loading M_1 before starting G_1: Considering the first argument, under best condition, the robot must unload M_1 switch grippers, load M_1 and moves to I/O for dropping a completed part $(2\varepsilon+\delta+\theta)$. After dropping the completed part, the robot unloads a new part from I/O and moves to M_1, $(2\varepsilon+\delta)$. This closed loop shows the minimum essential time that robot takes to load M_1 for its first operation after unloading it. If H_1 is negligible in comparison with δ, this closed loop takes $4\varepsilon+2\delta+\theta$ because the robot unloads the completed part from M_1 without any waiting time and exactly when it finished traveling from I/O to M_1. Otherwise, this closed loop takes $H_1+2\varepsilon+\theta$ if H_1 is meaningful in comparison with δ. The reasoning behind this taken time is that the minimum time taken by the robot from the moment that moves from M_1 to returns to M_1 is less than H_1 and the robot necessarily waits in front of M_1 to finish the operation of the part and, after finishing this operation, the dual-gripper robot unloads finished part from M_1 and loads the new part to it by using a switching operation ($H_1 + 2\varepsilon + \theta$). See Figure 3.4a.

1.2. Unloading M_1 after finishing G_1: This argument results from the succeeding robot movements: the robot must unload M_1, rotate the grippers at the end of its arm, load M_1 and move to M_2 ($2\varepsilon + \delta + \theta$). After unloading a part from M_2 and switching grippers, the robot loads the part that finished G_1 to M_2 and moves to M_1 ($2\varepsilon + \delta + \theta$). If H_2 is insignificant in comparison with δ, this loop takes $4\varepsilon + 2\delta + 2\theta$. Otherwise, it takes $H_1 + 2\varepsilon + \theta$ similar to the previous case. See Figure 3.4b.



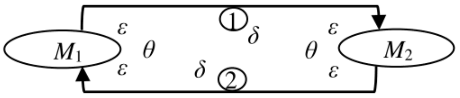
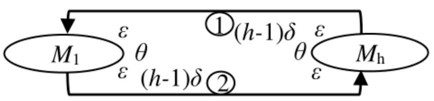


Fig. 3.4a (left). The first closed loop of performing operation H_1. Fig. 3.4b (right). The second closed loop of performing operation H_1

2. The second portion of the equation.

2.1. Loading M_1 before starting $G_h, h = 2, 3, ..., m-1$: In a similar manner, the robot is instructed to unload M_1, switch its grippers, load M_1, transfer the part to M_h ($2\varepsilon + (h-1)\delta + \theta$). In the second robot activity unloads M_h without waiting, completes its pivot and positions its second gripper in front of M_h, loads this machine, transfers the part which completed Oh and is ready to start G_h in M_1 ($2\varepsilon + (h-1)\delta + \theta$). Thus, the length of time for the resulting closed loop is $4\varepsilon + 2(h-1)\delta + 2\theta$ or $H_h + 2\varepsilon + \theta$ depending on H_h and δ. See Figure 3.5a.

2.2. Unloading M_1 after finishing $G_h, h = 2, 3, ..., m-1$: The structure of this closed loop looks quite similar to previous case. Initially the robot is placed in front of M_1 that finished G_h. Then, it has a forward movement from M_1 to M_{h+1} and a backward movement from M_{h+1} to M_1 ($2h\delta$). It unloads and loads each one of these machines and switch grippers ($4\varepsilon + 2\theta$); hence, the length of time for this closed loop is equal to $4\varepsilon + 2h\delta + 2\theta$ or $H_h + 2\varepsilon + \theta$ depending on H_h. See Figure 3.5b.



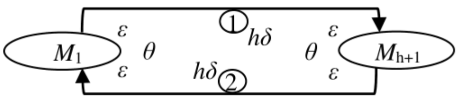
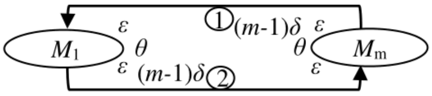


Fig. 3.5a (left). The first closed loop of performing operation H_h. Fig. 3.5b (right). The second closed loop of performing operation H_h

3. The third portion of the equation.

3.1. Loading M_1 before starting G_m: Clearly, the robot must visit each machine M_1 and $M_m(2(m-1)\delta)$, and change its gripper positions for unloading and loading of each one of these machines once $(4\varepsilon + 2\theta)$. It seems reasonable to conclude that the robot spends $4\varepsilon + 2(m-1)\delta + 2\theta$ or $H_m + 2\varepsilon + \theta$ time units depending on H_m. See Figure 3.6a.

3.2. Unloading M_1 after finishing G_m: Due to the fact that the part is complete after finishing G_m, it must be dropped at I/O. Therefore, in order to execute essential activities, the robot has a robot motion from M_1 to I/O, and then it return to M_1. Note that the robot loads and unloads both M_1 and I/O, while it only switches to other gripper in front of M_1. Thus, the robot spends $4\varepsilon + 2\delta + \theta$ or $H_m + 2\varepsilon\theta$ time units. See Figure 3.6b.



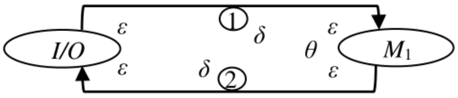


Fig. 3.6a (left). The first closed loop of performing operation H_m. Fig. 3.6b (right). The second closed loop of performing operation H_m

The lower bound for the per unit cycle time is simply found by summing up these values and dividing by 2 in that a dual-gripper robot is applied in this cell $\square$.

The objective is to determine a sequence of robot activities in such a way that the sum of all loading, unloading, traveling and waiting times is minimized. In this regard, we propose a robot move cycle namely dominant cycle and examine the performance of this proposed cycle, which is practical and can be put to use without difficulty. The dominant cycle is defined to be a robot move cycle in which, starting with an initial state, all machines are busy and the robot, which unloaded a part from I/O. Note that M_1 is also occupied and a part is loaded on it for G_1. The state transition in this cycle is shown below:

$1 - \Omega_1, ..., \Omega_m|B_{1+} \rightarrow \Omega_1, ..., \Omega_m|B_{1-}$
$2.1 - \Omega_1, ..., \Omega_m|A_{2+} \rightarrow \Omega_1, ..., \Omega_m|A_{2-} \rightarrow \Omega_1, ..., \Omega_m|B_{1+} \rightarrow \Omega_1, ..., \Omega_m|B_{2-}$
$\rightarrow \Omega_1, ..., \Omega_m|A_{2+} \rightarrow \Omega_1, ..., \Omega_m|A_{2-} \rightarrow \Omega_1, ..., \Omega_m|B_{2+} \rightarrow \Omega_1, ..., \Omega_m|B_{2-}$

$2.2 - \Omega_1, ..., \Omega_m|A_{3+} \rightarrow \Omega_1, ..., \Omega_m|A_{3-} \rightarrow \Omega_1, ..., \Omega_m|B_{2+} \rightarrow \Omega_1, ..., \Omega_m|B_{3-}$
$\rightarrow \Omega_1, ..., \Omega_m|A_{3+} \rightarrow \Omega_1, ..., \Omega_m|A_{3-} \rightarrow \Omega_1, ..., \Omega_m|B_{3+} \rightarrow \Omega_1, ..., \Omega_m|B_{3-}$

$2.3 - \Omega_1, ..., \Omega_m|A_{4+} \rightarrow \Omega_1, ..., \Omega_m|A_{4-} \rightarrow \Omega_1, ..., \Omega_m|B_{3+} \rightarrow \Omega_1, ..., \Omega_m|B_{4-}$
$\rightarrow \Omega_1, ..., \Omega_m|A_{4+} \rightarrow \Omega_1, ..., \Omega_m|A_{4-} \rightarrow \Omega_1, ..., \Omega_m|B_{4+} \rightarrow \Omega_1, ..., \Omega_m|B_{4-}$
...
$2.m{-}1{-}\Omega_1, ..., \Omega_m|A_{m+} \rightarrow \Omega_1, ..., \Omega_m|A_{m-} \rightarrow \Omega_1, ..., \Omega_m|B_{(m-1)+} \rightarrow \Omega_1, ..., \Omega_m|B_{m-}$
$\rightarrow \Omega_1, ..., \Omega_m|A_{m+} \rightarrow \Omega_1, ..., \Omega_m|A_{m-} \rightarrow \Omega_1, ..., \Omega_m|B_{m+} \rightarrow \Omega_1, ..., \Omega_m|B_{m-}$

$3 - \Omega_1, ..., \Omega_m|A_{0-} \rightarrow \Omega_1, ..., \Omega_m|A_{0+} \rightarrow \Omega_1, ..., \Omega_m|B_{m+} \rightarrow \Omega_1, ..., \Omega_m|B_{1-}$
$\rightarrow \Omega_1, ..., \Omega_m|A_{0-} \rightarrow \Omega_1, ..., \Omega_m|A_{0+}$

In Figure 3.7, a simple example consisting of three production machines, a hub machine and a dual gripper robot is illustrated to help clarify the sequence of the performed activities for the dominant cycle. In this example, the dominant cycle can be explained by the following sequences:

1. As the starting point of the dominant cycle, the robot moves from I/O to M_1 and, after unloading a part which has completed, rotates grippers and loads M_1 with another part in order to complete G_1.

2. Begin phase 2

  2.1. The robot travels between two adjacent machines M_1 and M_2, unloads M_2, switches grippers and finally loads M_2. Then, it moves backward from M_2 to M_1 and, after unloading the part which finished G_1, rotates grippers and loads M_1 with Finished part of M_2 to complete G_2. Once again, the robot moves to M_2, unloads it, exchanges grippers, loads it, returns to M_1, unloads the part which completed G_2, switches grippers, and loads M_1 with another part to complete G_2.

  2.2. The robot moves from M_1 to M_3 and, after unloading, rotates grippers and loads M_3. Thus, it returns to M_1 and, after unloading the part which finished G_2, changes the grippers position and loads M_1 with the part to complete G_3. Once more, the robot travels to M_3, unloads it, switches to the other gripper, loads it, moves to M_1, unloads the part which finished G_3, rotates grippers, and loads M_1 with another part to perform G_3.

  2.3. The robot travels between two nonadjacent machines M_1 and M_4 and, after unloading, switches grippers and loads M_4. Afterwards, it travels back to M_1 and, unloads the part which finished G_3, switches grippers

and loads M_1 with another part to perform G_4. Another time, the robot moves to M_4, unloads it, switches grippers, loads it, returns to M_1, unloads the part which completed G_4, exchanges grippers, and then loads M_1 with another part to complete G_4.
End phase 2

3. The dual-gripper robot travels to I/O, drops off the first finished part at I/O, and picks up a new part. It returns to M_1 and, after unloading the part which completed the last operation G_4, changes the grippers position and loads M_1 with the new part for performing G_1. The robot travels to I/O, drops off the second finished part at I/O, and picks up a new part again. Therefore, the cell ends up with the same state of the robotic cell as the initial state.

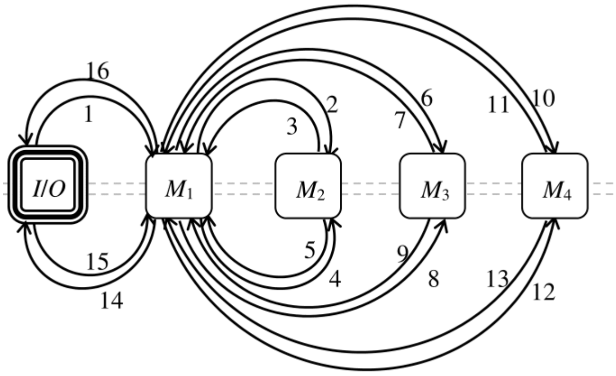


Fig. 3.7. The state transition of the dominant cycle for a hub reentrant robotic cell with four machines

Clearly, the phases 2.1, 2.2 and 2.3 are repetitive. Also, the dominant cycle produces exactly two parts, which shows it is a two-unit cycle. For examining the performance of the dominant cycle, initially, we present its cycle time. As a result, the cycle time of this cycle is derived in the following lemma:

Lemma 1. The following is the per unit cycle time of a dominant cycle for a reentrant cell consisting of a dual-gripper robot, a chain of $m-1$ machines, and a hub machine which follows $G_1, O_2, G_2, O_3, G_3, O_4, ..., G_{m-1}, O_m, G_m$:

$$\begin{aligned}\underline{T}_d &= 4m\varepsilon + ((m-1)m+2)\delta + (2m-1)\theta \\ &+\frac{1}{2}(\textstyle\sum_{h=2}^{m} max\{0, H_h-(2\varepsilon+2(h-1)\delta+\theta)\}+\sum_{h=1}^{m-1} max\{0, H_h-(2\varepsilon+2h\delta+\theta)\} \\ &+max\{0, H_1-(2\varepsilon+2\delta)\}+max\{0, H_m-(2\varepsilon+2\delta)\})\end{aligned} \tag{3.2}$$

Proof: Due to the fact that the sequence of $2m-1$ operations for this two-unit cycle is $G_1, O_2, G_2, O_3, G_3, O_4, ..., G_{m-1}, O_m, G_m$ and each one of these operations needs a loading and unloading operation before starting and after finishing it, there are $8m-4$ loading/unloading operations on machines. In addition to these $8m-4$ loading/unloading operations, the robot has to perform four loading/unloading operations on I/O; hence, there are totally $8m$ loading/unloading operations for producing two parts. In connection with travel time, the robot moves forward from M_1 to each one of $m-1$ machines exactly twice, which takes $(m-1)m\delta$. Likewise, it moves backwards from each one of $m-1$ machines to M_1 twice, which takes $(m-1)m\delta$ again. Also, the robot travels between I/O and M_1 four times to produce two parts. It can be deduced that the total travel time for producing two parts is $(2(m-1)m{+}4)\delta$ for producing two parts. For this two-unit cycle, we have $4m-2$ gripper switching because $4m-2$ operations must be performed by $m-1$ machines and the hub machine, and consequently one gripper switching is necessary for each one of these operations. The robots waiting time at any one of $m-1$ machine, which perform secondary operations, is zero. The reasoning behind this is that the waiting time is equal to subtracting P_i from taken time between leaving M_i and returning to $M_i, i = 2, , m$, which requires a minimum time of H_{i-1}.

In order to estimate the robot waiting time at the hub machine when it performed G_h $(h = 1, 2, ..., m)$, generally, H_h must be subtracted from the time required for the robot to leave M_1 after starting G_h and return to M_1 to unload it after finishing G_h. The sequence of this cycles activities is in a way that, after loading a part on M_1 to perform G_h, the robot moves to M_h or M_{h+1}, and then return to M_1 to unload the part that its h^{th} operation is performed. As a result of these kinds closed loops, $max\{0, H_1 - (2\varepsilon + 2\delta)\}$ and $max\{0, H_1 - (2\varepsilon + 2\delta + \theta)\}$ are the robots waiting time performing G_1, $max\{0, H_h - (2\varepsilon + 2(h-1)\delta + \theta)\}$ and $max\{0, H_h - (2\varepsilon + 2h\delta + \theta)\}$ are its waiting time for completing G_h, $h = 2, , m-1$, and finally $max\{0, H_m - (2\varepsilon + 2(m-1)\delta + \theta)\}$ and $max\{0, H_m - (2\varepsilon + 2\delta)\}$ are also its waiting time for completing G_m. The sum of all aforementioned waiting time is equal to:

$$\begin{aligned}&\textstyle\sum_{h=2}^{m} max\{0, H_h - (2\varepsilon + 2(h-1)\delta + \theta)\}\\ &+ \textstyle\sum_{h=1}^{m-1} max\{0, H_h - (2\varepsilon + 2h\delta + \theta)\}\\ &+ max\{0, H_1 - (2\varepsilon + 2\delta)\} + max\{0, H_m - (2\varepsilon + 2\delta)\})\end{aligned} \tag{3.3}$$

We determined the sum of all loading, unloading, traveling, gripper switching and waiting times for this two-unit cycle, so per unit cycle time is half of Equation (3.3), which yields the cycle time in Equation (3.2). This completes the proof.

If we prove the per unit cycle time of the dominant cycle time is always equal to the lower bound, we can conclude that the dominant cycle is the optimal cycle. For discussing the performance of this cycle, Theorems 2 is established below:

Theorem 2. For a reentrant cell consisting of a dual-gripper robot, a chain of $m-1$ machines, and a hub machine which follows $G_1, O_2, G_2, O_3, G_3, O_4, ..., G_{m-1}, O_m, G_m$, the optimal robot cycle is a two-unit cycle namely the dominant cycle.

Proof: Initially, we divide Equation (3.1) into three parts, A, B and C and rewrite them as follows:

$$A = \frac{1}{2} max\{4\varepsilon{+}2\delta{+}\theta, H_1{+}2\varepsilon{+}\theta\} \tag{3.4}$$

$$\begin{aligned} B = & \frac{1}{2}(\textstyle\sum_{h=2}^{m} max\{4\varepsilon + 2(h-1)\delta + 2\theta, H_h + 2\varepsilon + \theta\} \\ & + \textstyle\sum_{h=1}^{m-1} max\{4\varepsilon{+}2h\delta{+}2\theta, H_h{+}2\varepsilon{+}\theta\}) \end{aligned} \tag{3.5}$$

$$C = \frac{1}{2} max\{4\varepsilon + 2\delta + \theta, H_m + 2\varepsilon + \theta\} \tag{3.6}$$

Also, Equation (3.2) is divided into three following parts:

$$A' = 2\varepsilon{+}\delta{+}\frac{1}{2}\theta{+}\frac{1}{2} max\{0, H_1{-}(2\varepsilon{+}2\delta\} \tag{3.7}$$

$$\begin{aligned} B' = & 4(m-1)\varepsilon + ((m-1)m)\delta + 2(m-1)\theta \\ & + \frac{1}{2}(\textstyle\sum_{h=2}^{m} max\{0, H_h - (2\varepsilon + 2(h-1)\delta + \theta)\} \\ & + \textstyle\sum_{h=1}^{m-1} max\{0, H_h{-}(2\varepsilon{+}2h\delta{+}\theta)\}) \end{aligned} \tag{3.8}$$

$$C' = 2\varepsilon{+}\delta{+}\frac{1}{2}\theta{+}\frac{1}{2} max\{0, H_m{-}(2\varepsilon{+}2\delta\} \tag{3.9}$$

After adding left side to right side, Equation (3.7) and Equation (3.9) are rewritten as follows:

$$A' = \frac{1}{2} max\{4\varepsilon + 2\delta + \theta, H_1 + 2\varepsilon + \theta\} \tag{3.10}$$

$$C' = \frac{1}{2} max\{4\varepsilon + 2\delta + \theta, H_m + 2\varepsilon + \theta\} \tag{3.11}$$

Regarding Equation (3.8), we have:

$$B' = \frac{1}{2}(4(m-1)\varepsilon + ((m-1)m)\delta + 2(m-1)\theta$$

$$
\begin{aligned}
&+\textstyle\sum_{h=2}^{m} max\{0, H_h-(2\varepsilon+2(h-1)\delta+\theta)\}\\
&+4(m-1)\varepsilon+((m-1)m)\delta+2(m-1)\theta\\
&+\textstyle\sum_{h=1}^{m-1} max\{0, H_h-(2\varepsilon+2h\delta+\theta)\})\\
&=\frac{1}{2}(\textstyle\sum_{h=2}^{m} max\{4\varepsilon+2(h-1)\delta+2\theta, H_h+2\varepsilon+\theta\}\\
&+\textstyle\sum_{h=1}^{m-1} max\{4\varepsilon+2h\delta+2\theta, H_h+2\varepsilon+\theta\}) \qquad (3.12)
\end{aligned}
$$

$A = A', B = B', C = C'$, with the result that $\underline{T}_d = T_d$. This completes the proof.

We have discussed a proposed two-unit cycle and performed a detailed investigation into its productivity. We conclude that dominant cycle has satisfactory performance in comparison with other cycles in most applications.

3.5 Concluding Remarks

A new optimization methodology for advanced robotic hub reentrant workcell was introduced in this paper. We provided not only the lower bound of its cycle time but also the cycle time of a proposed cycle, namely the dominant cycle. We demonstrated some outcomes about optimality for this cycle and proved that it is an appropriate option for the hub reentrant robotic cell. The result of this paper is advantageous to many industries such as wafer fabrication, painting and electroplating lines. Hence, this study helps robotic cell manufacturers to be as competitive as possible. Considering cyclic production, there are some areas for development on the subject of the hub reentrant cells because the first study of a hub reentrant robotic cell is presented in this research to the best of our knowledge. To give an example, scheduling the hub reentrant cells with multiple robots is an appropriate track for future research. Considering the multiple robots increase the complexity of problem since the robots must not collide at their shared stages. The reentrant cells occasionally involve a stochastic process time is also an interesting topic for research. Finally, computational study of robotic hub reentrant workcell is also an interesting problem. Some open problems related to robotic hub reentrant workcell are: 1) What is the complexity of finding an optimal one-unit cycle in this scheduling problem? 2) What is the productivity improvements that may be obtained by using optimal one-unit cycle instead of two-unit cycles in this scheduling problem? 3) What is the productivity improvements that may be obtained by using dual gripper robot instead of single gripper robot in this scheduling problem?

Chapter 4 is based on the published article Foumani, M., Gunawan, I., Smith-Miles, K., Ibrahim, M.Y., 2015. Notes on Feasibility and Optimality Conditions of Small-Scale Multi-Function Robotic Cell Scheduling Problems with Pickup Restrictions. IEEE Transactions on Industrial Informatics, 11(3), 821 - 829.

Abstract *Optimization of robotic workcells is a growing concern in automated manufacturing systems. This study develops a methodology to maximize the production rate of a multi-function robot (MFR) operating within a rotationally arranged robotic cell. A MFR is able to perform additional special operations while in transit between transferring parts from adjacent processing stages. Considering the free pickup criterion, the cycle time formulas are initially developed for small-scale cells where a MFR interacts with either two or three machines. A methodology for finding the optimality regions of all possible permutations is presented. The results are then extended to the no-wait pickup criterion in which all parts must be processed from the input hopper to the output hopper, without any interruption either on or between machines. This analysis enables insightful evaluation of the productivity improvements of MFRs in real-life robotized workcells.*

Note to Practitioners *One of the most important traits of a robot is its work envelope, the space in which the robot can position its end effector. A robotic manipulator with a smaller work envelope can cover at most three machines, but still practitioners of robotic cell manufacturing prefer to use kinds of manipulators due to their speed and precision. KR 16-2, KR 40-PA and KR 60-3 are three types of robots produced by KUKA Robotics that have such a small work envelope. In addition, semiconductor cluster tools, which process semiconductor wafers for the fabrication of microelectronic components, typically consist a small number of wafer processing stations placed around a central, automated handling unit. Therefore, the study in this paper offer valuable help to practitioners who are considering the use of compact robots cells and also cluster tools with multi-functionality.*

Keywords *Automated manufacturing systems, Cyclic scheduling, Robotic cells, multi-function, No-wait*

Classification $SRF^{1,1,1}_{2,2,1}|$*free, additive , deterministic , identical, cyclic*$|T$
$SRF^{1,1,1}_{3,2,1}|$*free, additive , deterministic , identical, cyclic*$|T$
$SRF^{1,1,1}_{2,2,1}|$*no-wait, additive , deterministic , identical, cyclic*$|T$
$SRF^{1,1,1}_{3,2,1}|$*no-wait, additive , deterministic , identical, cyclic*$|T$

Note References are considered at the end of the thesis.

Chapter 4

Notes on Feasibility and Optimality Conditions of Small-scale Multi-function Robotic Cell Scheduling Problems with Pickup Restrictions

4.1 Introduction

Today's automated systems predominantly incorporate material handling robots interacting well with other equipment such as Computer Numerical Control (CNC) machines, and automated storage and retrieval systems in the production line (Ferrolho and Crisostomo, 2007). Any savings in robot movement time enhances the competitiveness of world class companies. Two classes of problem are Single-Function Robotic Cell (SFRC) and Multi-Function Robotic Cell (MFRC) scheduling problems, where determining a cyclic robot move sequence which yields the highest throughput gain is critical to success.

The first problem, which addresses a manufacturing cell equipped with a pick-and-place robot to perform a single task, is common in practice (Ferrolho and Crisostomo, 2011). This kind of transporting robot is usually called a Single-Function Robot (SFR). For the second problem, the cell is served by a Multi-Function Robot (MFR), which concurrently performs an arbitrary task in addition to part transportation tasks. One of the most recent industrial developments is the use of these MFRs in manufacturing cells. As an instance of MFRs, the application of Grip-Gage-Go (GGG) grippers performing in-process control as its additional task has become popular in manufacturing cells recently. The grippers, installed at the end of a MFR arm, perform quality control (e.g. accurately measure diameters) while carrying a part to the next machine. Figure 4.1 shows an example of these grippers used for measuring the diameter of a crankshaft. The measuring heads are integrated into the automation by adding gages and crankshaft locating features to MFRs (Foumani et al., 2013a). Here, we present a detailed study regarding GGG grippers.

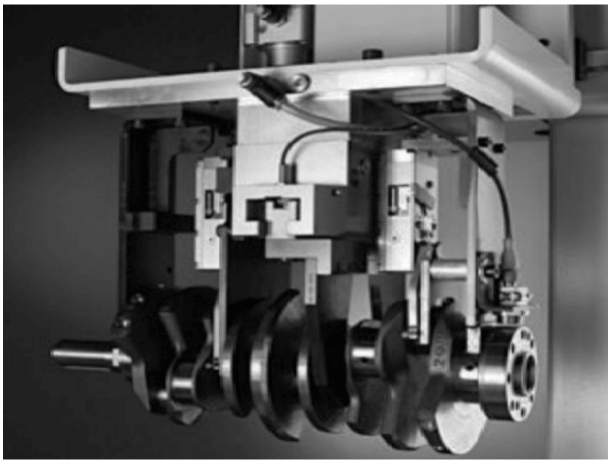


Fig. 4.1. Measurement of crankshaft diameters in transit

Because a gripper is an independent tool at the end of a robot's mechanical arm which can adapt to various production environments, the GGG gripper can be attached to a wide range of robots. A simple example of this is depicted in Figure 4.2 where a GGG gripper is added to the arm of Fanuc M-710iB/45 Robot. Hence, the Fanuc M-710iB/45 Robot can measure the thickness of shaft in transit between machines.

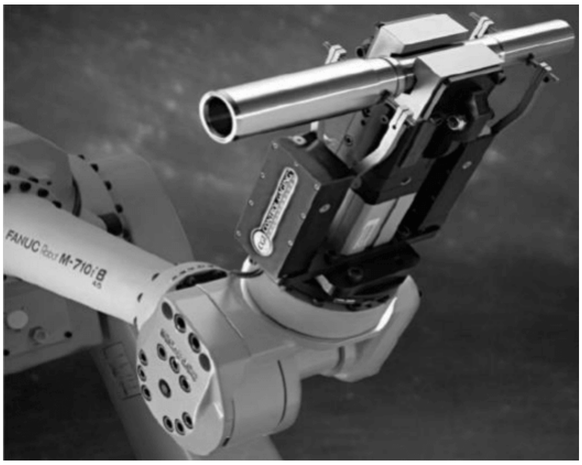
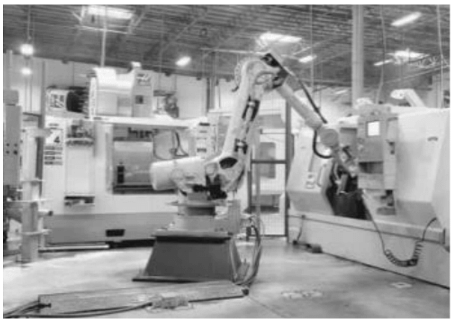


Fig. 4.2. The arm of Fanuc M-710iB/45 Robot equipped by a GGG gripper

A SFRC is generally composed of two machines M_1 and M_2 or three machines M_1, M_2, and M_3. A stationary base SFR rotating on its axis is used in this robotized shop to transfer parts from each machine to the next, and between machines and a joint input/output hopper I/O. Any arbitrary machine M_j placed in the cell performs operation O_j with the corresponding processing time P_j (Foumani and Jenab, 2013a).



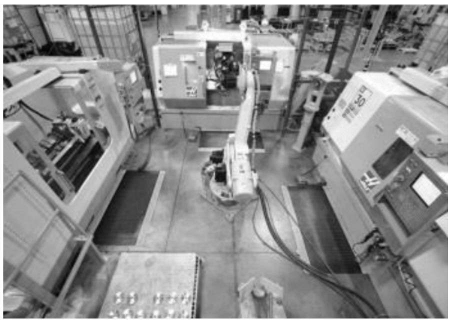
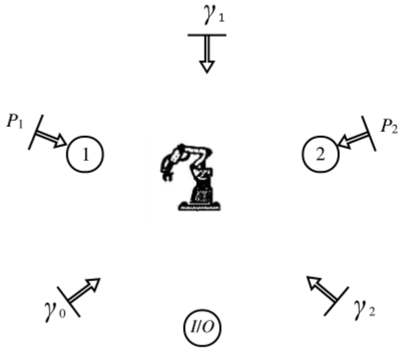


Fig. 4.3a (left). Two-machine SFRCs with rotational layout. Fig. 4.3b (right). Three-machine SFRCs with rotational layout

Figures 4.3a and 4.3b show real-life applications of two- and three-machine SFRCs at Haas Automation Incorporation. Physically, one SFR is assigned to each cell to avoid collisions. In these manufacturing cells, a SFR is in charge of picking up a part from I/O, loading it on CNC machine M_1 to be processed, transferring it through other machines and eventually dropping off this part at I/O where both the raw material and completed parts are stored. Two scenarios for unloading the part can be considered as soon as the parts operation on a machine is completed. Under the free pickup criterion, which is the predominant type in real-world cells, the part can stay indefinitely on the machine waiting for the SFR. However, under the no-wait pickup criterion, which is stricter, the part must be unloaded from the machine without delay and then carried to the down-stream machine. Consequently, the SFR must reach the machine on time.



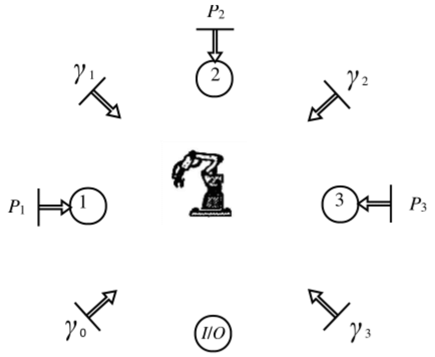


Fig. 4.4a (left). Two-machine MFRCs with rotational layout. Fig. 4.4b (right). Three-machine MFRCs with rotational layout

Figures 4.4a and 4.4b show two- and three-machine rotationally-arranged MFRCs in which Γ_j represents the robots operation while in transit between transferring parts from M_j to M_{j+1}. Also, γ_j denotes the processing time required by the robot to perform Γ_j. In Figures 4.4a and 4.4b, a single MFR is in charge of moving the parts through $\Gamma_0 \longrightarrow O_1 \longrightarrow \Gamma_1 \longrightarrow O_2 \longrightarrow \Gamma_2$ and $\Gamma_0 \longrightarrow O_1 \longrightarrow \Gamma_1 \longrightarrow O_2 \longrightarrow \Gamma_2 \longrightarrow O_3 \longrightarrow \Gamma_3$, respectively. In fact, the MFR is also responsible for performing processes $\{\Gamma_0, \Gamma_1, \Gamma_2\}$ and $\{\Gamma_0, \Gamma_1, \Gamma_2, \Gamma_3\}$ in transit, respectively. The time taken to perform these operations can be shown as $\{\gamma_0, \gamma_1, \gamma_2\}$ and $\{\gamma_0, \gamma_1, \gamma_2, \gamma_3\}$. The goal of this paper is to find a periodic MFRs task set that satisfies both the timing and other constraints (Quan and Chaturvedi, 2010). Thus, the rest of the paper is organized as follows. After presenting a brief literature review in Section 4.2, the authors describe the problem definitions and notation in Section 4.3. Section 4.4 is dedicated to find the optimal permutation in MFRCs with the free pickup criterion. Similar analysis for a MFRC with the no-wait pickup criterion is conducted in Section 4.5 to find an optimal permutation if residency time is restricted. Section 4.6 is devoted to the conclusions and discussion of future work.

4.2 Related Research

Considering the free pickup criterion, Sethi et al. (1992) presented a case study of two- and three SFRCs which performed drilling and boring operations on twenty pound castings to be used in truck differential assemblies. They succeeded in optimizing the production lines adapted from PRAB Robotic Company. Shortly afterwards, Sethi et al. (2001) focused on analyzing a class of two- and three-machine SFRCs served by two-unit SFRs. Other studies have also addressed multiple part-types, for example, scheduling multiple part-types in a dual-gripper robot cell was addressed in Drobouchevitch et al. (2004). The developed algorithm in Drobouchevitch et al. (2004) was only able to achieve a near optimal permutation with the worst-case performance ratio of 3/2. Note that a linear programming approach was employed in their research to compute the performance ratio without finding a lower bound.

Considering a case study in metal cutting industries, Geismar et al. (2005) established a unified notational and modelling structure to optimize two- and three-machine flexible SFRCs. They defined a flexible SFRC as the combination of a Flexible Manufacturing System (FMS) with a flow shop. Then, they derived the highest performance which could be obtained by changing the assignment of operations to production machines. Furthermore, an enumerative technique was applied for finding the worst-case performance ratio similar to Drobouchevitch et al. (2004). This worst case performance was 14 2/7% for the three-machine case, which means the maximum productivity increase of using a flexible SFRC instead of inflexible was 14 2/7%.

Also, Nambiar and Judd (2011) used max-plus algebra as a tool to develop a mathematical model for cyclic production lines. The newly-modeled max-plus formulation was able to facilitate the calculation of cycle time. In fact, it was used as the underlying mechanism to calculate cycle time precisely when an improvement heuristic algorithm such as Tabu Search (TS) or Genetic Algorithm (GA) was used to search for the optimal (or near-optimal) permutation. Subsequently, a reentrant SFRC that combined two machines with a SFR in a closed environment was optimized in Foumani and Jenab (2012). The employed SFR with temporary buffer had the ability to swap a part on a busy machine with a part on a busy SFR. The regions of optimality of all permutations were presented in Foumani and Jenab (2012) after performing a comparative analysis.

The no-wait pickup criterion is more suitable for real-life scheduling problems than other simplified scenarios. In this regard, Agnetis (2000) established polynomial algorithms for scheduling of two- and three-machine SFRCs. Also, Paul et al. (2007) developed a heuristic for a scheduling problem of a SFR used by an aircraft manufacturer with the surface treatment of component parts attached to both wings of transport aircrafts. Afterwards, Alcaide et al. (2007) took into account a scheduling problem appearing in the electroplating line, and established a graph model of operations for this small-scale SFRC with no-wait scenario. The SFR used in this automated cell was a part of the computer-integrated manufacturing system CIM-2000 Mechatronics manufactured by DEGEM Systems Company. A real-life radar scheduling problem, which is equivalent to single machine SFRC with no-wait pickup criterion, was studied in Brauner et al. (2009). She proved a radar system can be simulated by a no-wait SFRC due to the fact that the first task is a wave transmission and the second task is reflected wave receiving without delay.

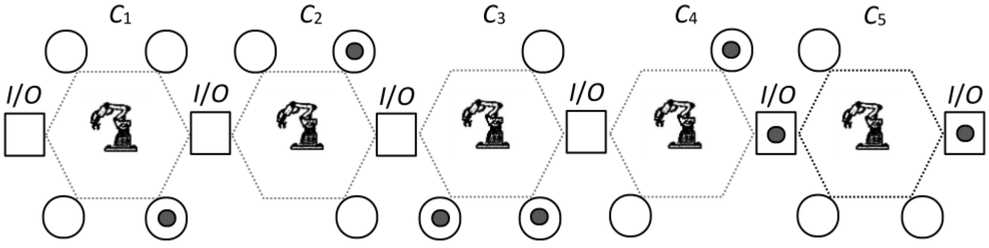


Fig. 4.5. A clustering system for connection between five small-scale MFRCs

A few recent papers are closely related to MFRCs with the free pickup criterion. For MFRCs where the MFR performs both printing and milling operations to supply large printed foam structures, an optimal schedule is generated in Keating and Oxman (2013). A MFR transferring the part between two adjacent processing stages and simultaneously performing an inspection operation in this transit was introduced for the first time in Foumani and Jenab (2013b). They considered the restricted model of the linearly-configured MFRCs producing identical parts, and only compared the performance of these MFRCs with SFRCs. The proposed approach for this MFRC involved deriving the lower bound of cycle time, and then finding some permutations with the cycle time as close as possible to this lower bound.

It is known that the number of feasible permutations for a MFRC with k machines is $k!$, whereas the research by Foumani and Jenab (2013b) was only restricted to studying two permutations. As a consequence, the results from Foumani and Jenab (2013b) could not be fully beneficial to MFRCs throughput analysis. Following that, Foumani et al. (2014) considered rotational MFRCs instead of in-line ones and discussed some results for replacing related MFRCs with SFRCs. Similar to Foumani and Jenab (2013b), the parameter values for which only two special permutations are optimal were determined. As a consequence, once again, the analysis was not complete and the impact for the remaining feasible region was not analyzed. Therefore, it is vital to develop a detailed analysis that fully covers all feasible regions, especially for two- and three-machine MFRCs.

The approach proposed in this paper determines the regions of optimality of all permutations and performs a comparative analysis after computing their cycle time. When the part processing routes in MFRCs are complicated, one of the most economic strategies is breaking these MFRCs into small-scale clusters. A MFR serves within one cluster consisting of two or three machines (Chan et al., 2008). Fig. 4.5 provides an example of converting a 15-machine semiconductor production line into five MFRCs. From the left side to the right side, we have four, three, three, two, and three-machine MFRCs. At first, parts must enter to the system from left-side I/O and then pass through cells C_1, C_2, C_3, C_4, and C_5. Finally, the part is stored at the right-side I/O. This paper also extends the results to the no-wait pickup criterion to consider more realistic conditions.

The most important contribution of this paper is to provide managerial insights into the advantages that can be achieved by applying MFRs for small-scale cells. In more detail, the novelty of this study is developing a methodology to maximize the production rate of MFRCs under both the free and no-wait pickup criteria. For all possible combinations of parameters, the feasibility and optimality regions of all permutations are presented. This research will provide a bridge between academic research on MFRCs and relevant real-world problems.

4.3 Problem Notation and Definitions

Compact SFRCs generally restrict intermediate hoppers, and consequently blocking or delay may happen. Scheduling MFR movement is also not deadlock-free and this results in the following operational restrictions: The receiving device (MFR or anyone of the machines) and sending device (MFR or anyone of the machines) must be empty and loaded before the load/unload process, respectively (Foumani and Jenab, 2013b). When the pickup criterion is no-wait, there is also an additional feasibility constraint: unloading the machine by the MFR with delay is not permitted. MFR is subjected to two types of waits, namely full and partial waits, in keeping with these constraints. In fact, after loading a part on a machine, MFR either stays on this machine until the end of the operation or moves to the next production machine to remove a part (Yan et al., 2008). The MFRC scheduling is expressed extending the notations and definitions below from (Foumani et al., 2013a):

ε The load (or unload) time of machines by MFR

δ The time taken by empty MFR to travel from M_i to M_{i+1}

S^i_{jmf} The j^{th} permutation of a MFRCs in which i, m and f denote the number of machines, multi-functionality and free pickup scenario, respectively

$T_{S^i_{jmf}}$ The cycle time of S^i_{jmf}

S^i_{jmw} The j^{th} permutation in which i, m and w denote the number of machines, multi-functionality and no-wait pickup

$T_{S^i_{jmw}}$ The cycle time of S^i_{jmw}

P_l The processing time of M_l dominating all machines as $\beta_{l-1} + P_l + \beta_l \geqslant \beta_{i-1} + P_i + \beta_i$

w_i MFRs waiting time at M_i for free- and no-wait scenarios.

Definition 1. Having a MFR, $2\varepsilon + max\{\gamma_i, \delta\}$ is the time elapsed of an activity $A_i, \forall i \in \{0, 1, 2, 3\}$, with sequence: 1) Empty MFR unloads a part from busy M_i. 2) MFR carries this part to M_{i+1}. 3) Finally, the busy MFR loads this part onto empty M_{i+1}.

We know two cases may occur when MFR performs activity A_i: 1) $\gamma_i \leq \delta$: this means MFR finishes the operation before arriving at M_{i+1}. Therefore, it loads the part to M_{i+1} as soon as the transfer of the part is finished, which totally takes δ. 2) $\gamma_i > \delta$: in contrast to previous case, MFR finishes the operation after arriving at M_{i+1}: thus, MFR stops in front of M_{i+1} to finish the operation and then loads the part to the machine. This takes γ_i time unit. Hence, as mentioned in Definition 1, the time taken by busy MFR to perform activity A_i is a couple of load/unload operations plus the *max* term of these two values: $2\varepsilon + max\{\gamma_i, \delta\}$. Note SFRC is a simplified subdivision of MFRC if $\gamma_i = 0, \forall i \in \{0, 1, 2, 3\}$. For simplicity, hereinafter β_i is used instead of $max\{\gamma_i, \delta\}$. The definition below deriving from Sethi et al. (1992) is applicable to MFRCs as well.

Definition 2. Having a MFR in the cell, a permutation of all activities in which one finished parts are dropped at I/O in each implementation is called a one-unit permutation.

These permutations are referred to as one-unit since each A_i occurs once. Note one-unit permutations are actually the easiest to understand, implement and also control in comparison to other permutations (Sethi et al., 2001). Also, focusing on one-unit permutations gives us insight into the behavior of complex permutations (Drobouchevitch et al., 2004). Hence, this study is restricted to one-unit permutations. It is also assumed that the empty and occupied machines of each permutation are specified in advance since this permutation must meet the steady state cyclic requirement following from Geismar et al. (2005).

Definition 3. Having a one-unit permutation starting with A_0, activity A_i is a *pushed (pulled)* activity if A_{i-1} is completed before (after) it. The *pushed (pulled)* activity A_i implies that M_i is empty (occupied) at the starting stage of the one-unit permutation.

It should be noted A_{i-1} is A_2 and A_3 when i=0 for two- and three-machine MFRCs. For example, A_1, A_2 are pushed activities and A_3 is pulled activity for permutation A_0, A_3, A_1, A_2 of three-machine case. This means M_1, M_2 are empty and M_3 is busy before starting it.

4.4 Free pickup Criterion

In essence, the robot with multi-functionality never results in increasing the number of permutations. Actually, $S^2_{1mf} = A_0, A_1, A_2$ and $S^2_{2mf} = A_0, A_2, A_1$ represent permutations which can be occur for MFRCs with two production machines. Note the regions of optimality for both S^2_{1mf} and S^2_{2mf} should be obtained later than reformulating the cycle time of these permutations. S^2_{1mf} only has pushed activities resulting in full stop on M_1 and M_2. This means that $T_{S^2_{1mf}}$ is made up the following independent portions: six load/unload operations, three dextrorotary and occupied MFR rotations, and two full waiting. Subsequently, we have $T_{S^2_{1mf}} = 6\varepsilon + \sum_{i=0}^{2} \beta_i + P_1 + P_2$.

Regarding S^2_{2mf}, MFR picks up an unprocessed part from I/O and loads it to $M_1(2\varepsilon+\beta_0)$. Then, based on the activitys route described above, MFR removes the previous part from M_2 and drops it at I/O after an empty rotation from M_1 to M_2 and a partial stop on $M_2(\delta + w_2 + 2\varepsilon + \beta_2)$. Likewise, the empty MFR comes back M_1, waits on M_1, unloads the part, and loads it on $M_2(\delta+w_1+2\varepsilon+\beta_1)$, and returns to $I/O(\delta)$. So, $T_{S^2_{2mf}}$ consists of six load/unload, three empty MFR rotations, three busy MFR rotations, and two partial stops: $w_1 = max\{0, P_1 - (2\varepsilon + 2\delta + \beta_2 + w_2)\}$ and $w_2 = max\{0, P_2 - (2\varepsilon + 2\delta + \beta_0)\}$. Because the summation of stops is equal to $max\{0, P_1 - (2\varepsilon + 2\delta + \beta_2), P_2 - (2\varepsilon + 2\delta + \beta_0)\}$, we can conclude that $T_{S^2_{2mf}} = 6\varepsilon + 3\delta + \sum_{i=0}^{2} \beta_i + max\{0, P_1 - (2\varepsilon + 2\delta + \beta_2), P_2 - (2\varepsilon + 2\delta + \beta_0)\}$.

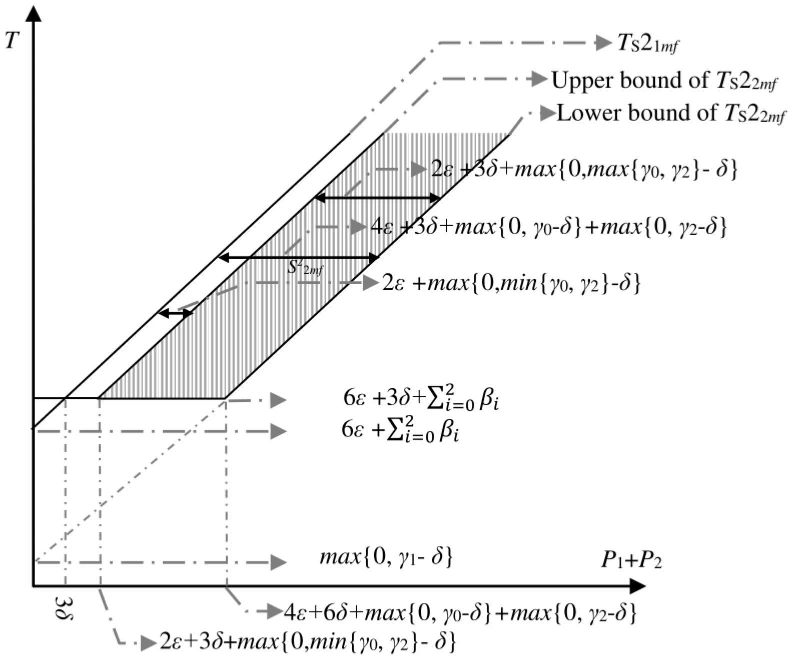


Fig. 4.6. The lower and upper bound of $T_{S^2_{1mf}} \leq T_{S^2_{2mf}}$

Figure 4.6 above illustrates the optimality regions of S^2_{1mf} and S^2_{2mf} by applying their cycle times. Obviously, S^2_{1mf} is optimal when $P_1 + P_2 \leq 3\delta$, and S^2_{2mf} is optimal in the rest of feasible area. Optimizing three-machine MFRCs are complicated in comparison with two-machine ones because the number of permutations grows from two to six permutations below:

$S^3_{1mf} = A_0, A_1, A_2, A_3$
$S^3_{2mf} = A_0, A_2, A_1, A_3$
$S^3_{3mf} = A_0, A_1, A_3, A_2$
$S^3_{4mf} = A_0, A_3, A_1, A_2$
$S^3_{5mf} = A_0, A_2, A_3, A_1$
$S^3_{6mf} = A_0, A_3, A_2, A_1$

We name S^3_{1mf} and S^3_{6mf} uphill and downhill permutations, and the rest of permutations rolling hill permutations. For the sake of simplicity, the cycle time of two the most complex ones, S^3_{2mf} and S^3_{6mf}, are calculated, and then the rest of cycle times are shown in this section. Since $S^3_{2mf} = A_0, A_2, A_1, A_3$, total load/unload time, empty MFR rotation, busy MFR rotation, and partial waiting times is $8\varepsilon+4\delta+\sum_{i=0}^{3}\beta_i+\sum_{i=0}^{3}w_i$. Clearly, $8\varepsilon+4\delta+\sum_{i=0}^{3}\beta_i$ is a constant value, whereas w_1, w_2, w_3 are variable values below: $w_1 = max\{0, P_1-(2\varepsilon+3\delta+\beta_2+w_2)\}, w_2 = max\{0, P_2-(4\varepsilon+2\delta+\beta_0+\beta_3+w_3)\}, w_3 = max\{0, P_3-(2\varepsilon+3\delta+\beta_1+w_1)\}$. Each one of waiting times w_1, w_2, w_3 can be zero or nonzero meaning there are eight subdivisions as follows:

- $w_1 = 0, w_2 = 0, w_3 = 0 \longrightarrow \sum_{i=0}^{3} w_i = 0$
- $w_1 = 0 \longrightarrow w_3 = max\{0, P_3-(2\varepsilon+3\delta+\beta_1)\}$
  $\sum_{i=0}^{3} w_i = max\{0, P_2-(4\varepsilon+2\delta+\beta_0+\beta_3), P_3-(2\varepsilon+3\delta+\beta_1)\}$
- $w_2 = 0 \longrightarrow w_1 = max\{0, P_1-(2\varepsilon+3\delta+\beta_2)\}$
  $\sum_{i=0}^{3} w_i = max\{0, P_1-(2\varepsilon+3\delta+\beta_2), P_3-(2\varepsilon+3\delta+\beta_1)\}$
- $w_3 = 0 \longrightarrow w_2 = max\{0, P_2-(4\varepsilon+2\delta+\beta_0+\beta_3)\}$
  $\sum_{i=0}^{3} w_i = max\{0, P_1-(2\varepsilon+3\delta+\beta_2), P_2-(4\varepsilon+2\delta+\beta_0+\beta_3)\}$
- $w_1 = 0, w_2 = 0 \longrightarrow \sum_{i=0}^{3} w_i = w_3 = max\{0, P_3-(2\varepsilon+3\delta+\beta_1)\}$
- $w_1 = 0, w_3 = 0 \longrightarrow \sum_{i=0}^{3} w_i = w_2 = max\{0, P_2-(4\varepsilon+2\delta+\beta_0+\beta_3)\}$
- $w_2 = 0, w_3 = 0 \longrightarrow \sum_{i=0}^{3} w_i = w_1 = max\{0, P_1-(2\varepsilon+3\delta+\beta_2)\}$
- $w_2 \neq 0, w_3 \neq 0, w_1 \neq 0$

It is easy to calculate all combinations, excluding the last one. The simplex method is applied for computation of the last $\sum_{i=0}^{3} w_i$. Assuming $A =$

$P_1 - (2\varepsilon + 3\delta + \beta_2), B = P_2 - (4\varepsilon + 2\delta + \beta_0 + \beta_3)$, and $C = P_3 - (2\varepsilon + 3\delta + \beta_1)$, we rewrite w_1, w_2, and w_3 as:

$$\begin{array}{l} w_1 \geq 0, w_1 \geq P_1 - (2\varepsilon + 3\delta + \beta_2 + w_2) \longrightarrow w_1 + w_2 \geq A \\ w_2 \geq 0, w_2 \geq P_2 - (4\varepsilon + 2\delta + \beta_0 + \beta_3 + w_3) \longrightarrow w_2 + w_3 \geq B \\ w_3 \geq 0, w_3 \geq P_3 - (2\varepsilon + 3\delta + \beta_1 + w_1) \longrightarrow w_1 + w_3 \geq C \end{array}$$

If s_1, s_2, s_3 were slack variables of these three inequalities, the execution of this algorithm is as Table 4.1. The algorithm deals with the maximization problem, whereas our goal is minimizing $\sum_{i=0}^{3} w_i$. Thus, $\sum_{i=0}^{3} w_i = A + \dfrac{C - A + B}{2} = \dfrac{P_1 + P_2 + P_3}{2} - (4\varepsilon + 4\delta + \dfrac{1}{2}\sum_{i=0}^{3} \beta_i) = max\{0, P_1 - (2\varepsilon + 3\delta + \beta_2), P_2 - (4\varepsilon + 2\delta + \beta_0 + \beta_3), P_3 - (2\varepsilon + 3\delta + \beta_1), \dfrac{P_1 + P_2 + P_3}{2} - (4\varepsilon + 4\delta + \dfrac{1}{2}\sum_{i=0}^{3} \beta_i)\}$.

Table 4.1: Implementation of the Simplex Algorithm for S_{2mf}^3

	w_1	w_2	w_3	s_1	s_2	s_3	Z
	1	1	1	0	0	0	0
s_1	1	1	0	-1	0	0	A
s_2	0	1	1	0	-1	0	B
s_3	1	0	1	0	0	-1	C
	0	0	1	1	0	0	$-A$
w_1	1	1	0	-1	0	0	A
s_2	0	1	1	0	-1	0	B
s_3	0	-1	1	1	0	-1	$C - A$
	0	0	1	1	0	0	$-A$
w_1	1	0	-1	-1	1	0	$A - B$
w_2	0	1	1	0	-1	0	B
s_3	0	0	2	1	-1	-1	$C - A + B$
	0	0	0	1/2	1/2	1/2	$-A - (C - A + B)/2$
w_1	1	0	0	-1/2	1/2	-1/2	$A - B + C - A + B)/2$
w_2	0	1	0	-1/2	-1/2	1/2	$B - C - A + B)/2$
w_3	0	0	1	1/2	-1/2	-1/2	$(C - A + B)/2$

Also, S_{6mf}^3 is made up four closed loops. Note the corresponding machine is located in the center of each one of them (See Figure 4.7), and the required time is $4\varepsilon + 2\delta + \beta_{i-1} + \beta_i + w_{i-1} + w_i$. Due to overlap between closed-loops, $T_{S_{6mf}^3} = 8\varepsilon + 8\delta + \sum_{i=0}^{3} \beta_i + w_1 + w_2 + w_3$ where $w_1 = max\{0, P_1 - (4\varepsilon + 6\delta + \beta_2 + \beta_3 + w_2 + w_3)\}, w_2 = max\{0, P_2 - (4\varepsilon + 6\delta + \beta_0 + \beta_3 + w_3)\}, w_3 = max\{0, P_3 - (4\varepsilon + 6\delta + \beta_0 + \beta_1 + w_1)\}$.

This means that $w_1 + w_2 + w_3 = max\{0, P_1 - (4\varepsilon + 6\delta + \beta_2 + \beta_3), P_2 - (4\varepsilon + 6\delta + \beta_0 + \beta_3), P_3 - (4\varepsilon + 6\delta + \beta_0 + \beta_1)\}$.

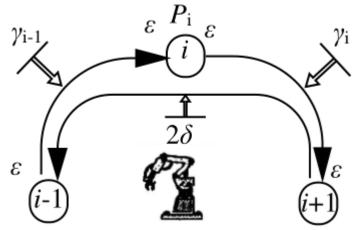


Fig. 4.7. The closed-loop i of three-machine MFRCs

We can conclude that the cycle times of six permutations are:

$$T_{S^3_{1mf}} = 8\varepsilon + \textstyle\sum_{i=0}^{3} \beta_i + P_1 + P_2 + P_3 \tag{4.1}$$

$$T_{S^3_{2mf}} = 8\varepsilon + 4\delta + \textstyle\sum_{i=0}^{3} \beta_i + max\{0, P_1 - (2\varepsilon + 3\delta + \beta_2), P_2 - (4\varepsilon + 2\delta + \beta_0 + \beta_3), P_3 - (2\varepsilon + 3\delta + \beta_1), \frac{P_1 + P_2 + P_3}{2} - (4\varepsilon + 4\delta + \frac{1}{2} \textstyle\sum_{i=0}^{3} \beta_i)\} \tag{4.2}$$

$$T_{S^3_{3mf}} = 8\varepsilon + 4\delta + \textstyle\sum_{i=0}^{3} \beta_i + P_1 + max\{0, P_2 - (2\varepsilon + 3\delta + \beta_3), P_3 - (4\varepsilon + 2\delta + \beta_0 + \beta_1 + P_1)\} \tag{4.3}$$

$$T_{S^3_{4mf}} = 8\varepsilon + 4\delta + \textstyle\sum_{i=0}^{3} \beta_i + P_2 + max\{0, P_1 - (2\varepsilon + 3\delta + \beta_3), P_3 - (2\varepsilon + 3\delta + \beta_0)\} \tag{4.4}$$

$$T_{S^3_{5mf}} = 8\varepsilon + 4\delta + \textstyle\sum_{i=0}^{3} \beta_i + P_3 + max\{0, P_1 - (4\varepsilon + 2\delta + \beta_2 + \beta_3 + P_3, P_2 - (2\varepsilon + 3\delta + \beta_0)\} \tag{4.5}$$

$$T_{S^3_{6mf}} = 8\varepsilon + 8\delta + \textstyle\sum_{i=0}^{3} \beta_i + max\{0, P_1 - (4\varepsilon + 6\delta + \beta_2 + \beta_3), P_2 - (4\varepsilon + 6\delta + \beta_0 + \beta_3), P_3 - (4\varepsilon + 6\delta + \beta_0 + \beta_1)\} \tag{4.6}$$

Table. 4.2: Optimality region of cycles of three-machine MFRCs with free pickup

$S^3{}_{1mf}$		$S^3{}_{2mf}$		$S^3{}_{3mf}$	
Compared permutations	Dominant Conditions	Compared permutations	Dominant Conditions	Compared permutations	Dominant Conditions
1&2	$P_1+P_2+P_3\leq 4\delta$	2&1	$P_1+P_2+P_3\geq 4\delta$	3&1	$P_2+P_3\geq 4\delta$
1&3	$P_2+P_3\leq 4\delta$	2&3	$P_3-P_1\leq 2\varepsilon+3\delta+\beta_1$ or $P_3-P_1-P_2\leq\beta_1-\beta_3$	3&2	$P_3-P_1\geq 2\varepsilon+3\delta+\beta_1$ and $P_3-P_1-P_2\geq\beta_1-\beta_3$
1&4	$P_1+P_3\leq 4\delta$	2&4	$\left\{\begin{array}{ll}P_2\geq\beta_3-\beta_2 & \text{or}\\ P_1-P_2\leq 2\varepsilon+3\delta+\beta_2 & \text{or}\\ P_1-P_2-P_3\leq\beta_2+\beta_0 & \text{and}\end{array}\right.$ $\left\{\begin{array}{ll}P_2\geq\beta_0-\beta_1 & \text{or}\\ P_3-P_2\leq 2\varepsilon+3\delta+\beta_1 & \text{or}\\ P_3-P_1-P_2\leq\beta_1+\beta_3 & \end{array}\right.$	3&4	$P_1\leq P_2$ or $P_2+P_3-P_1\geq 2\varepsilon+3\delta+\beta_0$
1&5	$P_1+P_2\leq 4\delta$	2&5	$P_1-P_3\leq 2\varepsilon+3\delta+\beta_2$ or $P_1-P_2-P_3\leq\beta_2+\beta_0$	3&5	$P_1\leq P_3$ or $P_2+P_3-P_1\geq 2\varepsilon+3\delta+\beta_0$ and $P_1+P_3-P_3\leq 2\varepsilon+3\delta+\beta_3$ or $P_1-P_3\leq\beta_3+\beta_0$
1&6	$P_1+P_2+P_3\leq 8\delta$	2&6	$P_1\leq 2\varepsilon+7\delta+\beta_2$ and $P_2\leq 4\varepsilon+6\delta+\beta_0+\beta_3$ and $P_3\leq 2\varepsilon+7\delta+\beta_1$ and $P_1+P_2+P_3\leq 8\varepsilon+16\delta+\sum_{i=0}^{3}\beta_i$	3&6	$P_1\leq 4\delta$ and $P_1+P_2\leq 2\varepsilon+7\delta+\beta_3$ and $P_3\leq 4\varepsilon+6\delta+\beta_0+\beta_1$
$S^3{}_{4mf}$		$S^3{}_{5mf}$		$S^3{}_{6mf}$	
Compared permutations	Dominant Conditions	Compared permutations	Dominant Conditions	Compared permutations	Dominant Conditions
4&1	$P_1+P_3\geq 4\delta$	5&1	$P_1+P_2\geq 4\delta$	6&1	$P_1+P_2+P_3\geq 8\delta$ or $P_1\geq 4\varepsilon+6\delta+\beta_2+\beta_3$ or $P_2\geq 4\varepsilon+6\delta+\beta_0+\beta_3$ or $P_3\geq 4\varepsilon+6\delta+\beta_0+\beta_1$
4&2	$\left\{\begin{array}{ll}P_1-P_2\geq 2\varepsilon+3\delta+\beta_2 & \text{or}\\ P_3-P_2\geq 2\varepsilon+3\delta+\beta_1 & \text{and}\end{array}\right.$ $\left\{\begin{array}{ll}P_2\leq\beta_3+\beta_2 & \text{or}\\ P_3-P_1-P_2\geq\beta_1+\beta_3 & \text{and}\end{array}\right.$ $\left\{\begin{array}{ll}P_1-P_2-P_3\geq\beta_2+\beta_0 & \text{or}\\ P_2\leq\beta_0+\beta_1 & \end{array}\right.$	5&2	$P_1-P_3\geq 2\varepsilon+3\delta+\beta_2$ and $P_1-P_2-P_3\geq\beta_2-\beta_0$	6&2	$P_1\geq 2\varepsilon+7\delta+\beta_2$ or $P_2\geq 4\varepsilon+6\delta+\beta_0+\beta_3$ or $P_3\geq 2\varepsilon+7\delta+\beta_1$ or $P_1+P_2+P_3\geq 8\varepsilon+16\delta+\sum_{i=0}^{3}\beta_i$
4&3	$P_2\leq P_1$ and $P_2+P_3-P_1\leq 2\varepsilon+3\delta+\beta_0$	5&3	$\left\{\begin{array}{ll}P_3\leq P_1 & \text{or}\\ P_1+P_2-P_3\geq 2\varepsilon+3\delta+\beta_3 & \text{and}\end{array}\right.$ $\left\{\begin{array}{ll}P_2+P_3-P_1\leq 2\varepsilon+3\delta+\beta_0 & \text{or}\\ P_1-P_3\geq\beta_3-\beta_0 & \end{array}\right.$	6&3	$P_1\geq 4\delta$ or $P_1+P_2\geq 2\varepsilon+7\delta+\beta_3$ or $P_3\geq 4\varepsilon+6\delta+\beta_0+\beta_1$
4&5	$P_2\leq P_3$ and $P_1+P_2-P_3\leq 2\varepsilon+3\delta+\beta_3$	5&4	$P_3\leq P_2$ or $P_1+P_2-P_3\geq 2\varepsilon+3\delta+\beta_3$	6&4	$P_2\geq 4\delta$ or $P_1+P_2\geq 2\varepsilon+7\delta+\beta_3$ or $P_2+P_3\geq 2\varepsilon+7\delta+\beta_0$
4&6	$P_2\leq 4\delta$ and $P_1+P_2\leq 2\varepsilon+7\delta+\beta_3$ and $P_2+P_3\leq 2\varepsilon+7\delta+\beta_0$	5&6	$P_3\leq 4\delta$ and $P_1\leq 4\varepsilon+6\delta+\beta_2+\beta_3$ and $P_2+P_3\leq 2\varepsilon+7\delta+\beta_0$	6&5	$P_3\geq 4\delta$ or $P_1\geq 4\varepsilon+6\delta+\beta_2+\beta_3$ or $P_2+P_3\geq 2\varepsilon+7\delta+\beta_0$

The results about the regions of optimality for six possible permutations are depicted in Table 4.2. For example, the common region where S^3_{1mf} dominates all permutations must be obtained to introduce S^3_{1mf} as the optimal permutation. This common region is the intersection of all possible dominant conditions. Therefore, S^3_{6mf} is optimal if $P_1 + P_2 + P_3 \leq 4\delta$ as can be seen from Table 4.2. Giving other example, S^3_{6mf} is optimal if $\beta_{l-1} + P_l + \beta_l \geq 4\varepsilon + 6\delta + \sum_{i=0}^{3} \beta_i$ or $P_1 + P_2 + P_3 \geq 8\varepsilon + 16\delta + \sum_{i=0}^{3} \beta_i$. The reason behind this is that the last part of Table 4.2 lists the conditions in which S^3_{6mf} dominates any one of another permutations, and the intersection of them equals $\beta_{l-1} + P_l + \beta_l \geq 4\varepsilon + 6\delta + \sum_{i=0}^{3} \beta_i$ or $P_1 + P_2 + P_3 \geq 8\varepsilon + 16\delta + \sum_{i=0}^{3} \beta_i$. This table gives a practical framework to use the robots permutation with maximum production rate for two- and three machine MFRCs with free pickup criterion. This framework makes a meaningful contribution to industrial automation, and assists industry in designing and developing appropriate MFRCs.

4.5 No-Wait Pickup Criterion

There is no study which concentrated on MFRCs with no-wait pick up scenario arising when the part must be immediately unload from the machine when its process is finished by the machine. This kind MFRC where machines cannot act as intermediate hoppers is generally called the no-wait MFRC. Since MFR also does operation on the part in transit, the no-wait restriction is not applicable about MFRs operation. The reason behind this is that MFR does secondary operations such as inspection, not primary operations. All secondary operations have same nature and do not respect to no-wait restriction (Foumani et al., 2014). Finding an optimal permutation for a MFRC with no-wait pickup criterion is a two-phase problem where all feasible permutations are determined in the first phase, and then optimal one is found in the second phase. To make the feasibility condition more clearly, let us present the following counterexample: For $\varepsilon = 0.5, \delta = 1, P_1 = 5, P_2 = 3, \beta_0 = 2, \beta_1 = 1, \beta_2 = 2$, the cycle S^2_{2mw} is infeasible because MFR cannot unload a part from M_2 as soon as it is processed by M_2. In fact, the time taken for MFR returns to M_2 is 5, whereas P_2=3.

S^2_{1mw} has no partial waiting; thus, it is always feasible and its cycle time is $T_{S^2_{1mw}} = 6\varepsilon + \sum_{i=0}^{2} \beta_i + P_1 + P_2$ regardless of the values of different parameters. However, S_2 has two partial stops on M_1 and M_2 which maybe cause of infeasibility. So, S^2_{2mw} is called feasible when both these partial stops satisfy. In fact, MFR must arrive at M_1 and M_2 not later than finishing the part processing. This means $P_1 \geq 2\varepsilon + 2\delta + \beta_2$ and $P_2 \geq 2\varepsilon + 2\delta + \beta_0$ are feasibility conditions of S^2_{2mw}.

As mentioned before, I/O is similar to auxiliary machine which should not meet the no-wait restriction. So, a strategy for making S^2_{2mw} feasible is that the part enters the MFRC with a time delay. This release time is indicated by R to calculate $T_{S^2_{2mw}}$. Clearly, the cycle time is $T_{S^2_{2mw}} = R + 6\varepsilon + 3\delta + \sum_{i=0}^{2} \beta_i + w_1 + w_2$ where R and $w_1 + w_2$ are not constant parts. We have:

$$w_1 = P_1 - (2\varepsilon + 2\delta + \beta_2 + w_2) \quad and \quad w_2 = P_2 - (2\varepsilon + 2\delta + \beta_0 + R) \tag{4.7}$$

$$\Leftrightarrow w_1 + w_2 = P_1 - (2\varepsilon + 2\delta + \beta_2) \tag{4.8}$$

$$\Leftrightarrow R = max\{0, P_2 + \beta_2 - (P_1 + \beta_0)\} \tag{4.9}$$

So, $T_{S^2_{2mw}}$ is shown by the double-sided function $max\{4\varepsilon + \delta + \beta_0 + P_1 + \beta_1, 4\varepsilon + \delta + \beta_1 + P_2 + \beta_2\}$. After derivation of $T_{S^2_{1mw}}$ and $T_{S^2_{2mw}}$, the performance of S^2_{1mw} and S^2_{2mw} should be compared to optimize two-machine MFRCs with no-wait scenario. Since $T_{S^2_{1mw}} > T_{S^2_{2mw}}$, we can conclude that S^2_{2mw} *is certainly optimal if it be feasible. It is only enough to check* S^2_{2mw} *meets the feasibility conditions* *(*$P_1 2\varepsilon + 2\delta + \beta_2$ *and* $P_2 2\varepsilon + 2\delta + \beta_0$*).*

Optimizing three-machine MFRCs are complicated in comparison with two-machine ones. This is even more difficult when pickup criterion is no-wait. In

Table 4.3: Cycle time and feasibility region of cycles with no-wait pickup scenario

Cycle	Feasibility conditions	Release time (RT) and cycle time (CT)

S^3_{1mw}	Always	RT=0
		CT=$8\varepsilon + \sum_{i=0}^{3} \beta_i + P_1 + P_2 + P_3$

S^3_{2mw}	$P_1 \geq 2\varepsilon + 3\delta + \beta_2$
	$P_2 \geq 4\varepsilon + 2\delta + \beta_0 + \beta_3$
	$P_3 \geq 2\varepsilon + 3\delta + \beta_1$
	$B + C \geq A, A + B \geq C$
		RT=$max\{0, P_2 + 4\delta + \beta_1 + \beta_2 - (P_1 + P_3 + \beta_0 + \beta_3)\}$
		CT=$max\{4\varepsilon + \frac{\sum_{i=0}^{3} \beta_i + P_1 + P_2 + P_3}{2}, 4\varepsilon + 2\delta + \beta_1 + \beta_2 + P_2\}$

S^3_{3mw}	$P_2 \geq 2\varepsilon + 3\delta + \beta_3$
	$P_3 \geq 4\varepsilon + 2\delta + P_1 + \beta_0 + \beta_1$
		RT=$max\{0, P_3 + delta + \beta_3 - (P_1 + P_2 + 2\varepsilon + \beta_0 + \beta_1)\}$
		CT=$max\{6\varepsilon + \delta + \beta_0 + \beta_1 + \beta_2 + P_1 + P_2, 4\varepsilon + 2\delta + \beta_2 + \beta_3 + P_3\}$

S^3_{4mw}	$P_1 \geq 2\varepsilon + 3\delta + \beta_3$
	$P_3 \geq 2\varepsilon + 3\delta + \beta_0$
		RT=$max\{0, P_3 + \beta_3 - (P_1 + \beta_0)\}$
		CT=$max\{6\varepsilon + \delta + \beta_0 + \beta_1 + \beta_2 + P_1 + P_2, 6\varepsilon + \delta + \beta_1 + \beta_2 + \beta_3 + P_2 + P_3\}$

S^3_{5mw}	$P_1 - P_3 \geq 4\varepsilon + 2\delta + \beta_2 + \beta_3$
	$P_2 \geq 2\varepsilon + 3\delta + \beta_0$
		RT=$max\{0, P_2 + P_3 + 2\varepsilon + \beta_2 + \beta_3 - (P_1 + P_2 + \delta + \beta_0)\}$
		CT=$max\{4\varepsilon + 2\delta + \beta_0 + \beta_1 + P_1, 6\varepsilon + \delta + \beta_1 + \beta_2 + \beta_3 + P_2\}$

S^3_{6mw}	$P_1 \geq 4\varepsilon + 6\delta + \beta_2 + \beta_3$
	$P_2 \geq 4\varepsilon + 6\delta + \beta_0 + \beta_3$
	$P_3 \geq 4\varepsilon + 6\delta + \beta_0 + \beta_1$
		RT=$max\{0, P_2 + \beta_2 - (P_1 + \beta_0), P_3 + \beta_2 + B_3 - (P_1 + \beta_0 + \beta_1)\}$
		CT=$max\{4\varepsilon + 2\delta + \beta_0 + \beta_1 + P_1, 4\varepsilon + 2\delta + \beta_1 + \beta_2 + P_2, 4\varepsilon + 2\delta + \beta_2 + \beta_3 + P_3\}$

fact, it is possible that no overlap exist between three machine operations and four MFR operations. In other words, every one of machines and MFR is potentially critical equipment if it shortly processes the part. Initially, we should take problem feasibility into consideration to better formulate no-wait restriction and estimate the gain of productivity. The cycle time and feasibility region of six permutations are listed in Table 4.3. This table indicates that the scheduling problem is never infeasible because S^3_{1mw} always gives a guarantee of feasibility. For the sake of simplicity, we present the process of cycle time calculation for S^3_{3mw} and S^3_{4mw} here, to show how we obtained the rest of cycle times and feasibility conditions except for S^3_{2mw} in Table 4.3. At first glance in S^3_{3mw}, there are two partial waits on M_2 and M_3. Only these two critical points may make S^3_{3mw} infeasible. Indeed, P_2 and P_3 must not be smaller than the time elapses between when the corresponding machine was loaded and when MFR come back to remove it. Two inequalities $P_2 \geq 2\varepsilon + 3\delta + \beta_3$ and $P_3 4\varepsilon + 2\delta + P_1 + \beta_0 + \beta_1$ cover the state space of S^3_{3mw} in keep with A_0, A_1, A_3, A_2. Also, $T_{S^3_{3mw}} = R + 8\varepsilon + 4\delta + \sum_{i=0}^{3} MRP_i + P_1 + w_2 + w_3$ where w_1 and w_2 are:

$$w_2 = P_2 - (2\varepsilon + 3\delta + \beta_3 + w_3) \; and \; w_3 = P_3 - (4\varepsilon + 2\delta + \beta_0 + \beta_1 + P_1 + R) \tag{4.10}$$

$$\Leftrightarrow w_2 + w_3 = P_2 - (2\varepsilon + 3\delta + \beta_3) \tag{4.11}$$

Therefore, $R = max\{0, P_3 + \delta + \beta_3 - (P_1 + P_2 + 2\varepsilon + \beta_0 + \beta_1)\}$ and $T_{S^3_{3mw}} = max\{6\varepsilon + \delta + \beta_0 + \beta_1 + \beta_2 + P_1 + P_2, 4\varepsilon + 2\delta + \beta_2 + \beta_3 + P_3\}$. Also, $S^3_{4mw} = A_0, A_1, A_3, A_2$ has two partial waits $P_1 - 2\varepsilon + 3\delta + \beta_3$ and $P_3 - 2\varepsilon + 3\delta + \beta_0 + R$ on M_1 and M_3, respectively. Since both of these partial waiting must be positive; the intersection of $P_1 \geq 2\varepsilon + 3\delta + \beta_3$ and $P_3 2\varepsilon + 3\delta + \beta_0$ shows feasible state space of S^3_{4mw}. Also, the cycle time of S^3_{4mw} is $T_{S^3_{4mw}} = R + 8\varepsilon + 4\delta + \sum_{i=0}^{3} \beta_i + P_1 + w_1 + w_3$ where

$w_1 + w_3 = P_1 - (2\varepsilon + 3\delta + \beta_3)$. This result:

$$R = max\{0, P_3{+}\beta_3{-}(P_1{+}\beta_0)\} \tag{4.12}$$

$$T_{S^3_{4mw}} = max\{6\varepsilon + \delta + \beta_0 + \beta_1 + \beta_2 + P_1 + P_2, 6\varepsilon + \delta + \beta_1 + \beta_2 + \beta_3 + P_2 + P_3\} \tag{4.13}$$

S^3_{2mw} is a tough permutation to deal with. Indeed, MFR has three partial stops in addition to artificial stop R at I/O during execution of this permutation. Note R can be called w_0 or w_4. The constant portion of S^3_{2mw} is $8\varepsilon + 4\delta + \sum_{i=0}^{3} \beta_i$, whereas $w_1 + w_2 + w_3 + R$ is the variable portion of it that should be minimized. $w_1 + w_2 + w_3 + R$ is built up four sub portions $w_1 = P_1 - (2\varepsilon + 3\delta + \beta_2 + w_2) \geq 0, w_2 = P_2 - (4\varepsilon + 2\delta + \beta_0 + \beta_3 + w_3 + R) \geq 0, w_3 = P_3 - (2\varepsilon + 3\delta + \beta_1 + w_1) \geq 0$, and $w_4 = R \geq 0$. We rewrite this minimization problem as the following formulation reassuming $A = P_1 - (2\varepsilon + 3\delta + \beta_2), B = P_2 - (4\varepsilon + 2\delta + \beta_0 + \beta_3)$, and $C = P_3 - (2\varepsilon + 3\delta + \beta_1)$:

$$\text{Minimize } x_1{+}x_2{+}x_3 \tag{4.14}$$

Subject to

$$x_1{+}x_2 \leq 10 \tag{4.15}$$

$$x_2{+}x_3 \leq 8 \tag{4.16}$$

$$x_1{+}x_3 \leq 5 \tag{4.17}$$

$Z = w_1 + B$ is an indirect result from (4.16). Thus, it is enough to find minimum amount of w_1 which is presented in four sub-cases representing the corner points the feasibility region:

1. $w_1 = 0$
   $w_2 = A \geq 0$
   $w_3 = C \geq 0$
   $w_4 = B - (A + C) \geq 0 \longrightarrow B \geq A + C$
2. $w_1 = A \geq 0$
   $w_2 = 0$
   $w_3 = C - A \geq 0 \longrightarrow C \geq A$
   $w_4 = B - (C - A) \geq 0 \longrightarrow A + B \geq C$
3. $w_1 = C \geq 0$
   $w_2 = A - C \geq 0 \longrightarrow A \geq C$
   $w_3 = 0$
   $w_4 = B - (A - C) \geq 0 \longrightarrow B + C \geq A$

4. $w_1 = (A + C - B)/2 \longrightarrow A + C \geq B$
   $w_2 = (B + A - C)/2 \longrightarrow A + B \geq C$
   $w_3 = (B + C - A)/2B + C \geq A$
   $w_4 = 0$

Let us assume $B = A + C$ is the breakpoint dividing the feasible regions of corner points 1 and 4. The corner points 1 is feasible for the left side of this breakpoint ($B \geq A + C$), and the amount of $w_1 = 0$ for this corner point is smaller than the amount for second and third corner points ($w_1 = A \geq 0$ and $w_1 = C \geq 0$). On the other hand, the corner points 4 is feasible for the right side of ($B < A + C$). Then, the amount of w_1 of the corner point 4 is $0 \leq w_1 \leq A$ and $0 \leq w_1 \leq C$ if $A \leq C$ and $C \leq A$. This prove that the amount w_1 of the corner point 4 is smaller than both 2 and 3 which are $w_1 = A$ and $w_1 = C$. So, the corner points 2 and 3 should be omitted from the formulation of $T_{S^3_{2mw}}$ in that one of the corner points 1 or 4 always dominates both of them and has smaller w_1. Note it is impossible to execute S^3_{2mw} if $B + C < A$ or $A + B < C$. In fact, $B + C = w_1 + w_2 + 2w_3 + w_4 \geq A = w_1 + w_2$ and $A + B = w_1 + 2w_2 + w_3 + w_4 \geq C = w_1 + w_3$ with respect to (4.15)-(4.17). We calculate two possible subcases of R using the original value of A, B, and C. Then, $T_{S^3_{2mw}}$ is obtained from the summation of the constant potion $8\varepsilon + 4\delta + \sum_{i=0}^{3} \beta_i$ and the variable portion $Z = w_1 + w_2 + w_3 + w_4 = w_1 + B$. This result:

$$R = \begin{cases} P_2 + 4\delta + \beta_1 + \beta_2 - (P_1 + P_3 + \beta_0 + \beta_3) & \text{if } B \geq A + C{=}0 \\ 0 & \text{if } B < A + C{=}0 \end{cases}$$

$$T_{S^3_{2mw}} = \begin{cases} 4\varepsilon + 2\delta + \beta_1 + \beta_2 + P_2 & \text{if } B \geq A + C{=}0 \\ 4\varepsilon + \dfrac{\sum_{i=0}^{3} \beta_i + P_1 + P_2 + P_3}{2} & \text{if } B < A + C{=}0 \end{cases}$$

Considering feasibility condition of S^3_{2mw}, (4.18) and (4.19) are rewritten by two max terms in the second row of Table 4.3. Now, we need an algorithm to reach the optimal permutation using the outcome of the Table 4.3. This algorithm is:

Search Algorithm: Finding the feasible and optimal permutation.

```
Input: State information (Machines and MFRs process times, empty MFR travel time, load/unload time).
for $j$=1 to 6
    if $S^3_{\mathrm{j}mw}$ is feasible according to conditions in Table 4.3 then
        $S \longleftarrow S + S^3_{\mathrm{j}mw}$
    end
Initialization of $T^* = \infty$
for x=1 to s
    if $T_{S^3_{\mathrm{x}mw}} \leq T^*$ then
        $S^* \longleftarrow S^3_{\mathrm{x}mw}$
        $T^* \longleftarrow T_{S^3_{\mathrm{x}mw}}$
    else
        $S^* \longleftarrow S^*$
        $T^* \longleftarrow T^*$
    end
Output: The optimal permutation $S^*$ and its cycle time $T^*$
```

As shown above, Search Algorithm is constructed from Table 4.3. The mechanism to reach the optimal permutation in trivial time is defining the set of feasible permutation $s \in S$, and then finding the optimal permutation S^* and its cycle time T^* using two For Loops. Anyone of permutations is stopped when an infeasible activity occurs in its activity route. In brief, it is expected the outcome of this algorithm be a practical help for robotic cell manufacturers who face difficult task of forming and scheduling a no-wait MFRC.

4.6 Concluding Remarks

An effective methodology was developed in this study for addressing the issue of industrial robots' functionality within a cellular production system. Two and six feasible permutations are developed for two- and three-machine MFRCs with the free pickup criterion, and the optimality regions of these permutations and their formulas are determined. Then, the results are extended to the no-wait pickup criterion. Through this research it was found there is no unique optimal permutation for MFR movement between different stations with different parameter inputs. To state the matter differently, it should be noted any one of the permutations has the chance of obtaining optimality considering different values of $\varepsilon, \delta, P_1, P_2, P_3, \gamma_0, \gamma_1, \gamma_2, \gamma_3$. It is enough to check whether it meets the optimality conditions or not. The scheduling method developed in this research can be broadened for multi-unit permutations in future research directions. In addition, some mathematical formalism such as max-plus algebra can be an important tool for research in this area to simplify the procedure for determination of cycle times. In fact, the analysis of all partial waits can be eliminated using max-plus algebra since synchronization is an inherent property of max-plus algebra systems. Lastly, reentrant MFRCs where a part visits a machine more than once in its processing route can be taken into account in future work. In this regard, considering the feasibility study in addition to the optimality study, we even can consider interval pickup scenario.

Chapter 5 is based on the published article Foumani, M., Gunawan, I., Ibrahim, M.Y., 2014. Scheduling Rotationally Arranged Robotic Cells Served by a Multi-Function Robot. International Journal of Production Research, 52(13), 4037 - 4058.

Abstract *Automated material handling systems are usually characterized by robotic cells that result in improvement of the production rate. The main purpose of this research is to study the scheduling of a rotationally arranged robotic cell with the multi-function robot (MFR). This special class of industrial robot is able not only to transfer the part between two adjacent processing stages but also to perform a special operation in transit. Considering MFR for material handling and operation, the objective function of the research here is the maximization of production rate, or equivalently the minimization of the steady-state cycle time for identical part production. This problem is modeled as a Travelling Salesman Problem (TSP) to give computational benefits with respect to the existing solution methods. Then, the lower bound for the cycle time is deduced in order to measure the productivity gain of two practical production permutations namely uphill and downhill permutations. As a design problem, a preliminary analysis initially identifies the regions where the productivity gain of a regular Multi-Function Robotic Cell (MFRC) is more than that of the corresponding Single-Function Robotic Cell (SFRC) for both small- and large-scale cells. The conclusion shows the suggested topics for future research.*

Note to Practitioners *This work targeted the design of a robot cell in terms of two parameters: the degree of multi-functionality and the size of the system. This study only consider that the stochastic data are only recorded by the robot in an independent computer which simplifies that problem. However, it extend the existing conceptual framework to real-life robotic cells. The framework can cover a range of real-world applications, such as complex automobile assembly lines.*

Keywords *Cyclic scheduling, Robotic cell, Multi-function, Rotational arrangement*

Classification $SRF^{1,1,1}_{l,2,1}|$*free, additive , deterministic , identical, cyclic*$|T$

Note References are considered at the end of the thesis.

Chapter 5

Scheduling Rotationally Arranged Robotic Cells Served by a Multi-Function Robot

5.1 Introduction

Cellular Manufacturing Systems (CMSs) are probably the most popular alternative for mass production environments. Actually, CMSs are applied with the purpose of maximizing the production rate in the factory. As a powerful tool for material handling, industrial robots are being progressively employed in CMSs. A consequential problem is to find a cyclic robot move which increases the production rate as much as possible and gives the maximum cell output. This problem is predominantly named Robotic Cell Scheduling Problem (RCSP). Also, it should be emphasized that one of the most important features of an industrial robot is its gripper. In addition to material handling, it makes the industrial robot appropriate for a set of tasks such as spot welding, spray painting and inspection. Based on these operational orientations, RCSP is divided into Single-Function Robotic Cell (SFRC) and Multi-Function Robotic Cell (MFRC) problems.

Literature concerning with former case assumed that the industrial robot is only able to perform one of above tasks. More precisely, these studies take into account the robot has the ability to act as a spot welding-gun, spray painting-gun, gaging device, and material handling device (MHD) separately. This kind of robot is usually named Single-Function Robot (SFR). There are a number of studies considering SFR tasks separately. For spot welding applications, Zacharia and Aspragathos (2005) solved the problem of determining the optimum route of a spot-welding end-effector visiting a number of task points as a variation of Traveling Salesman Problem (TSP) using Genetic Algorithm (GA). They have implemented the inverse kinematics of a SFR as the model for calculating the cycle time. Following this study, Givehchi et al. (2011) applied a new assembly planning method to the sequence optimization in minimizing the cycle time of a real-life robotic spot-welding problem for a sheet-metal assembly found in automotive industry. This case study involved the optimization of sequences for assembling a partial cabin of a vehicle consisting of 9 sheet-metal parts connecting by 134 spot-welding points. For painting applications, Potkonjak et al. (2000) studied dynamic optimization of a painting SFR motion. They focused on the minimization of the manufacturing cost subject to a constraint on painting quality. In their research, it was assumed that a robotized painting shop needs: (i) the simulation of the painting operation; (ii) modeling of the quality level; (iii) definition of cost function; (iv) kinematic and technological parameters of the painting task. Afterwards, Diao et al. (2009) developed an optimal motion planning for a painting SFR to optimize the sequences of spray-painting end-effector on a free surface. In this study, nonlinear programming techniques took into account position and orientation of the spray-painting device to minimize cycle time. They also minimized the thickness variation of the paint spraying of a specified spatial path. For inspection applications, Wang and Cannon (1996) explored an automated inspection system interfaced with a SFR to reduce testing time and storage requirements. Also, Edinbarough et al. (2005) presented an on-line robotic inspection and monitoring system for identifying ICs lead defect on PCBs.

There are several works in the literature dealing with MFRC scheduling problems. A model for determining the cost of replacing a semi-automated SFRC with a fully-automated MFRC was developed by Geren and Redford (1999). A MFR with multi-functional capabilities to assemble and inspect PCBs was used in this fully-automated cell. Bernd et al. (2006) automatized a rework MFRC where a high-glossed fitting was inspected and polished by a special MFR. A new approach namely Compound Fabrication was presented by Keating and Oxman (2013) for robotic manufacturing of large printed foam structures. This approach made real both multi-functional and multi-material processes by applying a MFR in cell. MFR performed both printing and milling, and also was able to shift between manufacturing ways using various grippers. To our best knowledge, it is vital to develop an effective method to schedule the sequence of a Multi-Function Robot (MFR) performing both material handling and an arbitrary operation concurrently in a rotationally-arranged MFRC. It is obvious that the industrial robot with this modern ability, which is used in the latter case, may dramatically raise production rate. Here, the material handling is considered as a fixed duty of MFR due to the fact that the common application of the industrial robots, which more than other application investigated extensively in the literature, includes only transferring of uncompleted parts in CMSs. Actually, this assumption helps the scheduling problem to represent realistic conditions.

It is vital to describe how MFRs are incorporated into the design of the real production environments. First of all, a special kind of gripper namely Grip-Gage-Go is introduced. Due to the fact that the robot arm with this gripper accurately measures diameters while part is carried to next machine. It supports the purpose of reducing floor space requirements, waiting time at a particular machine, loading/unloading time and finally robot motions in the manufacturing cell. Actually, this modern gripper integrates measurement systems into parts handling device, and makes MFR competent to measure the diameter of the part in transit between any two consecutive machines. More precisely, MFRs are widely employed in the inspection of automotive products including crankshaft, gears, engine valves and lifters in transit. Giving an example of the crankshaft production lines, the measuring heads are integrated into the automation by adding gages and crankshaft locating features to MFR using in these lines. Figure 5.1 shows an example of this kind of robot gripper used for measuring the diameter of crankshaft. Obviously, measuring the diameter of crankshaft is a difficult task on account of crankshaft's peculiar geometry and orientation.

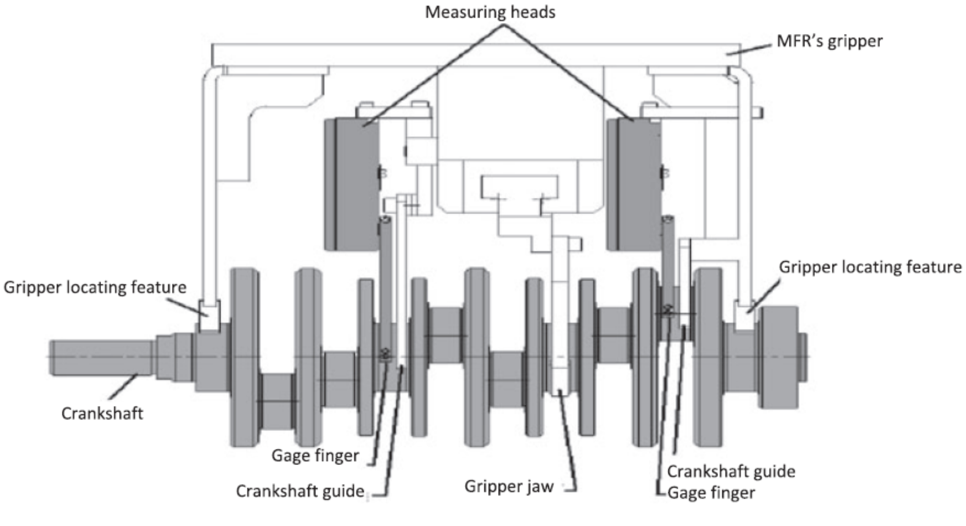


Fig. 5.1. The measurement of crankshaft diameters in transit

Secondly, there is a special kind of robot, namely SDA10, which is suitable for assembly and part transfer in production lines, simultaneously. When fixturing is costly, this MFR is an economical robotic solution since it can performs an operation while the part is carried to the next production machine. The above real-life evidences show the increasing use of MFRCs in the market, particularly for assembly lines. Subsequently, it is crucial to undertake a comprehensive research onto effect of this kind cell on production rate.

A SFRC is normally described by the subsequent framework: this cell consists a SFR and a group of k machines, namely $M_1, M_2, ..., M_k$. A group of k corresponding operations, namely $O_1, O_2, ..., O_k$, is performed by these k machines. The processing time of $O_i, i = 1, ..., k$, equals $P_i, i = 1, ..., k$, and $M_i, i = 1, ..., k$, processes at most one part at any given moment. Since all parts are processed in the same non-decreasing order of operations, SFRC has the form of a flowshop. SFR operate with a programmable PC-based control platform which allows manufacturers to profit from all the benefits of PC technology. The unlimited space for the raw material and completed parts are separately available at input hopper (I) and output hopper (O), which are also named auxiliary machines M_0 and M_{k+1}, respectively. These two hoppers can be also combined into a joint input/output hopper (I/O) where both the raw material and completed parts are stored. From technical point of view, once again, I/O can be called both auxiliary machines M_0 and M_{k+1} with the processing time equal to zero. I/O acts like the auxiliary machine M_0 when SFR unloads the part from this hopper, and it acts like M_{k+1} when SFR loads the part to this hopper.

There is a significant interaction between cell layout and optimum robot move sequences (Gultekin et al., 2008). Based on hoppers layout, the in-line and rotational arrangements are two prevailing robotic cell types which are applied in industry. These two arrangements offer more travel economy for the robot and are easy to be under control. It should be emphasized that a moveable robot moving on a rail network is employed for the in-line arrangement, while a stationary base robot rotating on its axis is used in the rotational arrangement (Foumani and Jenab, 2013a). To examine these layouts in detail, let us consider that I and O can be separated or jointed, respectively. For the former, the in-line case, SFR picks up the raw materials from I, loads them on M_1 to be processed, transfers them through k machines in a similar way and finally drop off the completed parts at O. However, for the later which is the rotational case, SFR picks up the raw materials from I/O and again drop off the completed parts at I/O after transferring them through all k machines. Let us assume that the required time for SFR to travel from M_i to M_{i+1} is δ. Hence, the time required to perform a direct move from M_i to M_j is $|i-j|\delta$ and $min\{|i-j|, k+1-|i-j|\}\delta$ for in-line and rotational arrangements, respectively. Since always $|i-j|\delta \geq min\{|i-j|, k+1-|i-j|\}\delta$, the travel time between any arbitrary machines in the in-line arrangement never is less than this travel time in the rotational arrangement. As a matter of fact, the moveable robot has only one option to transfer the part from M_i to M_j in an in-line arrangement, whilst there are two options available for the stationary base robot to travel between these two machines in a rotational arrangement and these two options can potentially reduce the average travel-time.

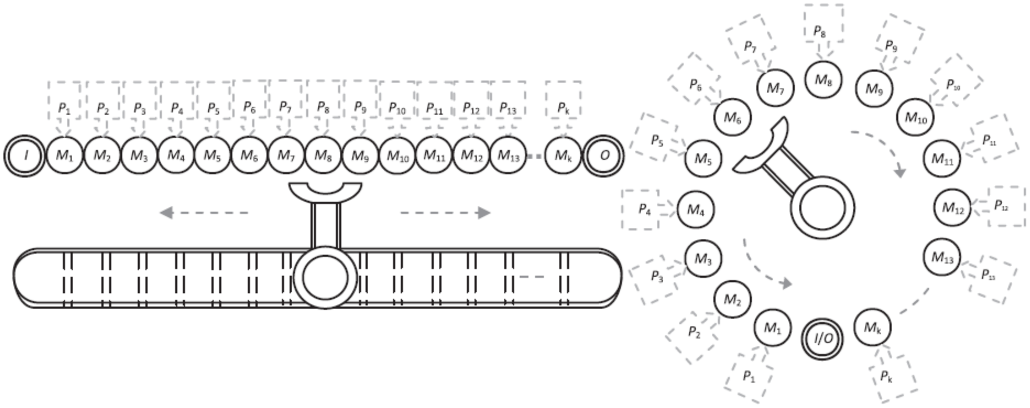


Fig. 5.2. SFRCs with in-line and rotational arrangements

In addition to saving a considerable amount of travel-time, increasing the productivity and space efficiency of the cell, reducing the installation cost of the robot, increasing the robot programmability are the main reason that manufacturers prefer to use rotationally arranged robotic cells instead of linearly arranged robotic cells (Yildiz et al., 2012). In this paper particular emphasis is placed on the rotational arrangement because of the remarkable advantages in terms of technology and its common application in many high-tech industries like semiconductor manufacturing. Figure 5.2 shows SFRCs with in-line and rotational arrangements.

Since the rotational arrangement as layout and MFR as operating device are considered in this study, the manufacturing cell is both layout- and operation-oriented. As already mentioned above, only performing operations on machines is considered in the context of scheduling, whereas we impose an operation-oriented extension, and consider MFR processes the part in transit between two consecutive machines. A group of $k+1$ processes, namely $RO_0, RO_1, RO_2, , RO_k$, is performed by MFR. The processing time of $RO_i, i = 1, ..., k$, equals $RP_i, i = 0, 1, ..., k$. As a result, a series of $2k+1$ operations, namely $RO_0, O_1, RO_1, O_2, RO_2, O_3, ..., RO_k$, indicates the order of $2k+1$ necessary operations that are generally configured to be performed by MFR and k machines. Figure 5.3 shows such a rotationally-arranged MFRC.

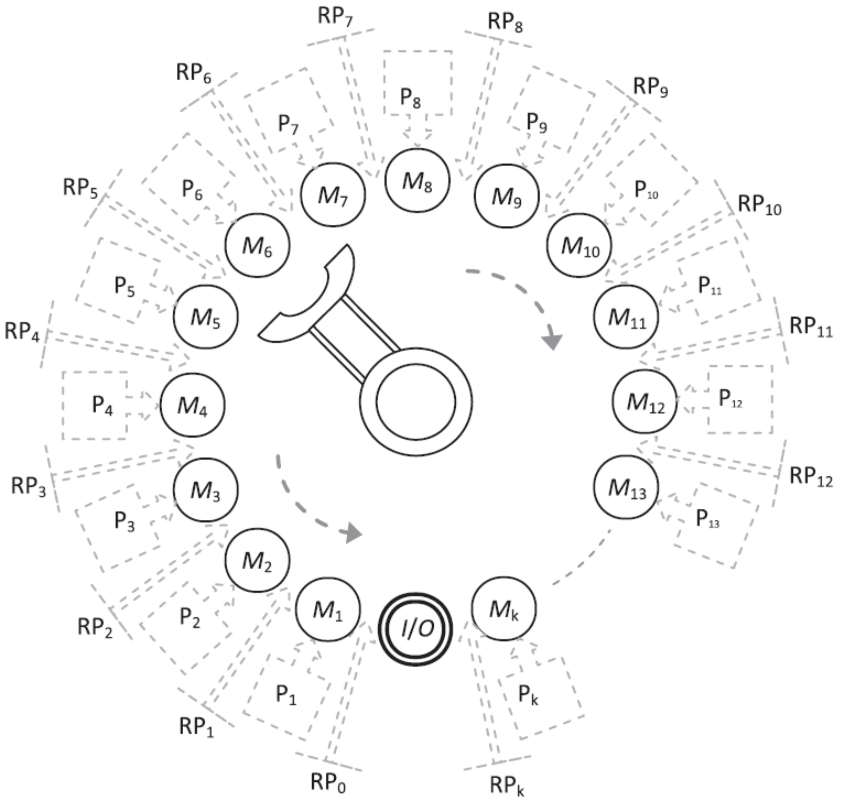


Fig. 5.3. The order of operations for a rotationally-arranged MFRC

It is clear that, similar to SFRC, MFRC operates as a flow shop manufacturing system in that all parts are processed on operating devices, MFR and machines, in the same order.

5.2 Related Research

Literature witnesses valuable results related to SFRC problems in a cyclic sense. The first purely analytic research on SFRC scheduling problem were properly performed by Sethi et al. (1992). The novelty of this study was the introduction of the optimal one-unit permutation as the overall optimal permutation in the two-machine SFRCs. For three-machine SFRCs, similar results were also proved by Crama and de Klundert (1999). However, these results were disproved by Brauner and Finke (2001) for the four-machine SFRCs. In another work, Levner et al. (1997) extended this structure for the problem where the number of machines is arbitrary and the part finished processing by a machine must be handled to the next machine without delay. Classical SFRC scheduling problems are widely addressed in Dawande et al. (2005); Brauner (2008).

Recently, a great deal of attention has been paid to SFRC scheduling problem. They have predominantly concentrated on the feasibility of following assumptions: SFRC with CNC machines (Gultekin et al., 2009), SFR with two grippers (Geismar et al., 2008; Dawande et al., 2010), the stochastic objective function (Shafiei-Monfared et al., 2009), the bicriteria objective function (Gultekin et al., 2010), the machine with one-unit input and output hoppers (Drobouchevitch et al., 2010), multiple SFRs in the manufacturing cell (Che and Chu, 2009; Che et al., 2011a), SFRC with bounded work-in-process (WIP) level (Che et al., 2011b), SFRC with time window constrains (Che et al., 2011c), and SFR with temporary buffer (Jolai et al., 2012; Foumani and Jenab, 2012). Although many exact methods has been suggested for solving SFRC scheduling problem, the pertaining researches witness an acute shortage of attention focused on the effects of MFRs on the production rate. Therefore, it is crucial to understand how MFRs with the capabilities of performing both material handling and an arbitrary operation can improve the throughput rate in the context of cyclic production.

The remainder of this research is organized as follows: after giving a general overview of the robotic cell, the notation and definitions of a cyclic production are described in Section 5.3. Then, a TSP based structure for MFRC scheduling problem is developed in Section 5.4. The motivation behind modeling MFRC scheduling problem as a TSP is that modeling MFRC scheduling problems as a TSP gives computational benefits due to the existing solution methods (Bagchi et al., 2006). The lower bound of MFRC and the associated optimal one-unit permutations are determined in Section 5.5. For the sake of completeness, Section 5.6 is devoted to a comparison between the performance of MFRCs and SFRCs when the constraint that MFR performs odd-numbered operations is taken into account. It is necessary to mention that Sections 5.3 and 5.5 are precedent for section 5.6. In fact, the motivation behind these sections is that the optimal one-unit permutation, which equals lower bound, must be initially determined based on predefined notation and definitions, and then obtained results are used for a comparison between the performance of MFRCs and SFRCs. Eventually, Section 5.7 is dedicated to conclusion and future research direction.

5.3 Problem Notation and Definitions

In MFRC scheduling problem, the major problem is how to determine the sequence of MFR moves to maximize the production rate. The reason behind this statement is that MFRC scheduling problem is part of RCSP where determining optimal sequence of robot movements is the focus of the majority of research (Dawande et al., 2005). In order to tackle this problem, some important definitions are given. These definitions assist us in describing MFRCs as much detail as possible and establishing the solving approach.

For getting the maximum benefit from MFR under consideration, the general definitions such as the full waiting and partial waiting are implied. If MFR wholly waits at $M_i, \forall i \in \{1, 2, ..., k\}$, to finish the processing after loading a part onto this machine, M_i is said to have full waiting equals P_i. An alternative way is that, without stopping at M_i, the empty MFR immediately moves to another occupied machine after loading a part onto M_i. This kind of waiting is topically known as partial waiting. In contrary to full waiting, the calculation of partial waiting is complex. It at most equals P_i minus the time that elapses between the point when the robot has completed loading M_i and is about to start unloading M_i (Dawande et al., 2002).

Owing to significant difference between SFRCs and MFRCs, updating the definitions belong to pertaining literature is essential. Notice that scheduling the sequence of MFR movement is not trivial in that potential deadlock exists. Accordingly, there are many impossible sequences due to the deadlock-related constraints. The feasibility constraints of MFR movement are as follow: 1- loading busy machine by MFR is impossible, 2- loading the machine by empty MFR is impossible 3- unloading empty machine by MFR is wrong 4- unloading the machine by busy MFR is wrong. In a nutshell, these four constraints mean that receiving device (MFR or anyone of machines) and sending device (MFR or anyone of machines) must be empty and loaded before loading/unloading process, respectively, in order to prevent potential deadlocks existence. In keeping with these constrains, a repeatable sequence of robot movement, namely *Activity*, is established here. Actually, each permutation is a different combination of MFR activities which repeats infinitely. Considering ε as the load (or unload) time of machines by MFR, the definition of a particular MFR activity is presented as follows:

Definition 1. Under a MFRC being composed of k machines and a MFR performing both material handling and operation in transit, an activity $A_i, \forall i \in \{0, 1, 2, ..., k\}$, is:

1. Empty MFR unloads a part from busy M_i, taking ε.
2. MFR carries this part to M_{i+1}, taking $max\{RP_i, \delta\}$.
3. Busy MFR loads this part onto empty M_{i+1}, taking ε.

Obviously, Definition 1 implies that the time elapsed during the execution of activity A_i is the constant value $2\varepsilon + max\{RP_i, \delta\}$. It should be emphasized that SFRC is a subdivision of MFRC when $RP_i = 0, \forall i \in \{0, 1, 2, ..., k\}$.

Accordingly, it looks fair to say that the total time taken by MFRC to produce a finished part is certainly more than that of SFRC in that the overall elapsed time of $A_i, , \forall i \in \{0, 1, 2, ..., k\}$, into MFRC is bigger than it for SFRC, $2\varepsilon + max\{RP_i, \delta\} \geq 2\varepsilon + \delta$. However, MFRC has a great opportunity to improve the cycle time in comparison with SFRC due to the fact that the number of operational machines is declined if SFRC is swapped with MFRC. For simplicity, we use MRP_i instead of $max\{RP_i, \delta\}$ here. A detailed analysis of this challenging replacement will be done in Section 5.6 for both small- and large-scale cells. The following definition which is derived from Crama and de Klundert (1999) is applicable to MFRC as well.

Definition 2. Having a MFRC being composed of a MFR and k production machines, a permutation of $n(k + 1)$ activities in which n finished parts are dropped at I/O is called an n-unit MFR permutations.

As a matter of fact, n-unit permutations are so named since each activity A_i occurs n times, and consequently MFR loads and unloads each one of k machines n times in each implementation. It should be noted that one-unit permutations are the most important subdivision of n-unit permutations. Therefore, the necessary framework for RCSP is developed here based on one-unit permutation consideration. For this permutation, each production machine is loaded and unloaded by MFR once, and one part is completed after each implementation.

It is known that the number of possible one-unit permutation in a simple SFRC equals $k!$ if all one-unit permutations start with A_0 (Sethi et al., 1992). However, Akturk et al. (2005) showed that this result may no longer be valid when there is an operational flexibility and new robot move sequences should be developed to capture the operational flexibility. Therefore, it is needed to find out whether the number of one-unit permutation in a MFRC also equals $k!$ or whether we need to find new sequences for a MFRC, as shown in Akturk et al. (2005). Clearly, similar to Sethi et al. (1992), there are totally $k!$ possible one-unit permutations in a MFRC since no machine or robot is added to this cell. In fact, the possible one-unit permutations of SFRCs and SFRCs have one-to-one correspondence relationship with each other. The question naturally arises from this result is that: Is it possible to use the earlier result on SFRCs to determine the optimal one-unit permutation of MFRCs? Although both of these cells have same number of possible one-unit permutations, we cannot apply the obtained result of SFRCs analysis for MFRCs. The reason behind it is that SFRCs are an especial subcase of MFRCs where $RP_i = 0, \forall i \in \{0, 1, 2, ..., k\}$: therefore, MFRC scheduling problem is much more complex than SFRC scheduling problem. As a matter of fact, potentially, the cycle time of every one of $k!$ permutations of a MFRC may change by adding at least one arbitrary RP_i bigger than δ. To make it more clear, the following counterexample is illustrated:

Example 1. Considering a ten-machine SFRC with in-line arrangement and following data: $\varepsilon = 0.5, \delta = 1, P_i = 0.5 (\forall i \in \{1, 2, ..., 9\})$ and finally $P_{10} = 50$. For k-machine SFRC with in-line arrangement, Crama and van de Klundert (1997) addressed the lower bound as $max\{2(k+1)(\varepsilon+\delta)+\sum_{i=1}^{k} min\{P_i, \delta\}, 4(\varepsilon+\delta)+P_{l(s)}\}$. In this lower bound $P_{l(s)}$ represents the processing time of M_l which is biggest processing time in a SFRC. Thus, lower bound of cycle time of aforementioned ten-machine SFRC, which is balanced for identical products, is 56 time units. On the other hand, the cycle time of $A_0, A_{10}, A_9, A_8, A_7, A_6, A_5, A_4, A_3, A_2, A_1$ equals $max\{2(k+1)\varepsilon + 4k\delta, 4(\varepsilon + \delta) + P_{l(s)}\}$ time units meaning this permutation is optimal for this SFRC. Now, assume the scenario when the raw materials require an arbitrary operation RO_0 before loading them on first machine, and the robot is able to do this operation during handling materials between I and M_1. The processing time of this operation, which equals $RP_0 = 40$ time units, is larger than δ and results in converting SFRC into MFRC. Surprisingly, the cycle time of aforementioned permutation for this new developed MFRC is $max\{2(k + 1)\varepsilon + (4k - 1)\delta + RP_0, 4(\varepsilon + \delta) + P_{l(s)}\} = 90$, whereas there is the cycle $A_0, A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_{10}, A_9$ with smaller cycle time, $max\{2(k+1)\varepsilon + 2(k + 3)\delta + RP_0 + \sum_{i=1}^{k-2} P_i, 4(\varepsilon + \delta) + P_{l(s)}\} = 78$. This means the optimal permutation of the first scenario (SFRC) and the second scenario (MFRC) are different, and consequently the result of SFRC scheduling problems is not applicable for MFRC scheduling problems. This can be easily extended for rotational arrangement.

Example 1 clarifies the statement that the result of SFRC problems is not applicable to MFRC ones, and need more analysis. Also, as mentioned before, four feasibility constraints must be satisfied to prevent occurrence of impossible permutations. This means the empty and occupied machines relating to each permutation should be specified in advance. Thus, in keeping with Definition 1 and 2, the definition below is stated to assign the parts to the machines:

Definition 3. Having a MFRC being composed of a MFR and k production machines, a permutation of $n(k + 1)$ activities in which n finished parts are dropped at I/O is called an n-unit MFR permutations.

Having a one-unit permutation consisting $k+1$ activities, $A_i, \forall i \in \{0, 1, 2, ..., k\}$, is called a pushed activity (pulled activity) if A_{i-1} is completed before (after) this activity. The pushed activity (pulled activity) A_i implies M_i is empty (occupied) at the starting stage of the one-unit permutation.

It should be emphasized that $A_{i-1} = A_m$ when i=0. Giving an example, the type of activities for the one-unit permutation $A_0, A_3, A_2, A_6, A_7, A_9, A_8, A_1, A_5, A_4$ is presented as follows: the activity sets A_1, A_4, A_7, A_8 and A_2, A_3, A_5, A_6, A_9 indicate pushed and pulled activities, respectively. This means that machine sets M_1, M_4, M_7, M_8 and M_2, M_3, M_5, M_6, M_9 are empty and busy before starting this cycle. Figure 5.4 shows pushed and pulled activities, and consequently empty and loaded machines.

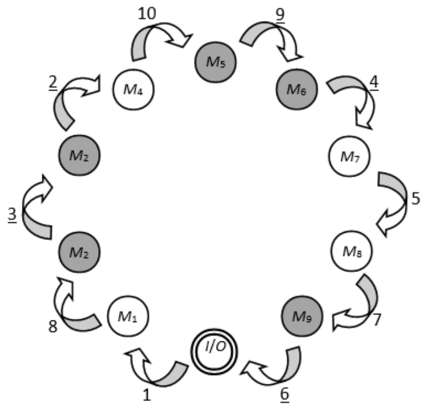


Fig. 5.4. The empty and busy machines in the beginning stage of permutation $A_0, A_3, A_2, A_6, A_7, A_9, A_8, A_1, A_5, A_4$

In Figure 5.4, the sequences orders of pulled activities are underlined, but the rest of activates, pushed activities, have no underline. Note that white-colored machines are empty, and gray-colored machines are loaded at the beginning phase. Now, Let us use the following notation throughout the text:

$s_{j(s)}$ The j^{th} SFR's permutation after reaching steady state from an empty cell.

$s_{j(m)}$ The j^th MFR's permutation after reaching steady state from an empty cell.

$T_{s_{j(s)}}$ The per unit cycle time of $s_{j(s)}$.

$T_{s_{j(m)}}$ The per unit cycle time of $s_{j(m)}$.

$P_{l(s)}$ The processing time of M_l which is biggest processing time in a SFRC, $P_{l(s)} \geq P_i, \forall l, i \in \{1, 2, ..., k\}$.

$P_{l(m)}$ The processing time of M_l which dominates the processing times of all the rest machines in a MFRC as $MRP_{l(m)-1} + P_{l(m)} + MRP_{l(m)} \geq MRP_{i-1} + P_i + MRP_i$.

$\underline{T}_s$ The lower bound of per unit cycle time for k-machine SFRCs.

$\underline{T}_m$ The lower bound of per unit cycle time for k-machine MFRCs.

$T_{s\rightarrow m}$ The lower bound of per unit cycle time for swapping k-machine SFRCs with $\lceil\frac{k-1}{2}\rceil$-machine MFRCs.

PO_k The operating efficiency of using $\lceil\frac{k-1}{2}\rceil$-machine MFRCs in place of k-machine SFRCs.

w_i MFR's waiting time at M_i (I/O performs the dummy operations with $w_0 = 0$ and $w_{k+1} = 0$)

The explanation of why a k-machine SFRC can be only replaced with $\lceil\frac{k-1}{2}\rceil$-machine MFRC is given in more detail in Section 5.6. In a nutshell, the reason behind this replacement is that the odd-numbered operations performing by MFR have same nature, whereas the even-numbered operations performing by $\lceil\frac{k-1}{2}\rceil$ machines are completely different in nature.

A classification scheme based on the standard scheme of Hall et al. (1997) is also extended here. The standard scheme is generally divided into three descriptors $E_1|E_2|E_3$, where E_1 represents the robot and production machines environment, E_2 explains processing characteristics, and E_3 implies the objective function. Hence, the extended scheme is employed to specify the MFRC scheduling problem under discussion as $R_{mf}C_k^O|\delta_i = max\{RP_i, \delta\}, \varepsilon_i = \varepsilon, cyclic - 1|C_t$. In the first descriptor, the robotic cell with multi-function robot (R_{mf}) and k rotationally arranged machine (C_k^O) are characterized as $R_{mf}C_k^O$. The second descriptor implies that the travel time between any two consecutive machines M_i and M_{i+1} is independent of their physical positions meaning that the distance between any successive machines is equal to the constant value δ. However, it is dependent on the type of operation which is performed by machine M_i since MFR performs a related operation RO_i on part after this operation and during transferring the part from M_i to $M_{i+1}(\delta_i = max\{RP_i, \delta\})$. In addition, the loading/unloading time is not machine dependent ($\varepsilon_i = \varepsilon$), and the permutation only produces a single identical part ($cyclic - 1$). The last descriptor also describes that the objective function is minimization of cycle time.

Before proposing a lower bound for a MFRC's cycle time, the complexity of the problem is discussed in the following section when the objective is maximization of production rate.

5.4 A Modified TSP-Based Problem Formulation

Since MFRC under consideration in this study produces the identical parts. Therefore, the study of multiple part-types is outside the scope of this research. First of all, a brief description of the currently established approaches of analyzing the complexity of SFRC scheduling problems is presented. Then, a mixed integer linear programing (MILP) model is developed for MFRC scheduling problems.

An $O(m3)$ dynamic programming model was developed by Crama and van de Klundert (1997) in order to minimize the one-unit cycle time of a SFRC producing single part-type. Following interval SFRC structure, Levner and Kats (1998) found the optimal permutation in $O(m3)$ time. Similarly, Ioachim et al. (2001) suggest a polynomial-time algorithm with a time complexity of $O(qm3)$ to find the optimal q-unit permutations. Afterwards, Brauner et al. (2003) demonstrated that SFRC scheduling problem will be strongly NP-hard if it has symmetrical travel-times and also the triangle inequality holds. Rajapakshe et al. (2011) proved that finding an optimal one-unit cycle in a SFRC with a circular layout and an additive travel-time metric is NP-hard. They develop a polynomial algorithm with a 5/3-approximation to an optimal one-unit cycle. Using a mixed integer linear programming model, Gultekin et al. (2009) formulate the flexible SFRC scheduling problem as a modified TSP which was much more complicated than that classical TSP.

Topically, $G(V, D)$ is a graph where a TSP of size k cities is presented. $V = \{0, 1, 2, ..., k\}$ and $D = [d_{ij}]_{V \times V}$ indicate the group of nodes and edges of this graph, respectively. Also, the travel cost of each edge (i, j) is denoted by d_{ij} in this $V \times V$ travel cost matrix. The main aim of this journey is to obtain a Hamiltonian tour of this graph with the lowest cost. The constraints are that travelling salesman must pass all nodes and finally returns to the starting city. It is not also allowed to have any unvisited cities.

It is known that modeling MFRC scheduling problems as a TSP gives computational benefits due to the existing solution methods. Accordingly, a TSP based structure for MFRC scheduling problem is developed here. Each of the MFR activities $A_0, A_1, A_2, ..., A_k$ is symbolized by a node $V = \{0, 1, 2, ..., k\}$ to be passed. Generally, a sequence of two consecutive activities $..., Ai, Aj, ...$, where $\forall i, j \in \{0, 1, 2, ..., k\}$, has one-to-one correspondence relationship with d_{ij}. The time elapsed between these two activities, namely e_{ij}, equals the overall loading, unloading and travel time which is elapsed after finishing A_i and before finishing A_j. Accordingly, there are only two cases of permutation of any two activities A_i and A_j:

1. $j = i + 1$: This means that MFR has a full waiting equal to P_j on M_j. Under

this condition, MFR wholly waits at M_{i+1} to finish the processing, unloads the part, performs RO_{i+1} on the part and carries out it to M_{i+2} concurrently, and finally loads the part on $M_{i+2}(P_{i+1}+\varepsilon+MRP_i+1+\varepsilon)$. Therefore, we have:

$$e_{ij} = 2\varepsilon + MRP_j \tag{5.1}$$
$$d_{ij} = e_{ij} + P_j \tag{5.2}$$

2. $j \neq i+1$: This means that MFR has a partial waiting equal to w_j on M_j; therefore, the empty MFR immediately moves from M_{i+1} to M_j, unloads the part from M_j after a partial waiting, performs RO_j on the part and carries out it to M_{j+1}, and then loads the part on M_{j+1}. The elapsed time is:
$$e_{ij} = 2\varepsilon + min\{|i+1-j|, k+1-|i+1-j|\}\delta + MRP_j \tag{5.3}$$
$$d_{ij} = e_{ij} + w_j \tag{5.4}$$

Based on these two feasible cases, a simplified example of d_{ij} for four-machine MFRC is as follows:

$$d_{ij} = \begin{bmatrix} \infty & 2\varepsilon+MRP_1+P_1 & 2\varepsilon+\delta+MRP_2+w_2 & 2\varepsilon+2\delta+MRP_3+w_3 & 2\varepsilon+\delta+MRP_4+w_4 \\ 2\varepsilon+2\delta+MRP_0 & \infty & 2\varepsilon+MRP_2+P_2 & 2\varepsilon+\delta+MRP_3+w_3 & 2\varepsilon+2\delta+MRP_4+w_4 \\ 2\varepsilon+2\delta+MRP_0 & 2\varepsilon+2\delta+MRP_1+w_1 & \infty & 2\varepsilon+MRP_3+P_3 & 2\varepsilon+\delta+MRP_4+w_4 \\ 2\varepsilon+\delta+MRP_0 & 2\varepsilon+2\delta+MRP_1+w_1 & 2\varepsilon+2\delta+MRP_2+w_2 & \infty & 2\varepsilon+MRP_4+P_4 \\ 2\varepsilon+MRP_0 & 2\varepsilon+\delta+MRP_1+w_1 & 2\varepsilon+2\delta+MRP_2+w_2 & 2\varepsilon+2\delta+MRP_3+w_3 & \infty \end{bmatrix}$$

The travel cost matrix d_{ij} is made up fixed parameters $\varepsilon, \delta, P_i, RP_i$ and variable parameter w_i which make the model more complex than classical TSP. Assuming c_i and c_j as the completion times of A_i and A_j, MFRC can be modeled as the following TSP based formulation:

$$T^* = MinT \tag{5.5}$$

S.t.

$$c_j \geq c_i + (e_{ij}+P_j)x_{ij} - (1-x_{ij})M \qquad \forall i \in \{0,1,2,...,k\}, j = i+1 \tag{5.6}$$

$$c_j \geq c_i + e_{ij}x_{ij} + y_{ij} - (1-x_{ij})M \qquad \forall i,j \in \{0,1,2,...,k\}, i \neq j, j = i+1 \tag{5.7}$$

$$T \geq c_i + z_i x_{i0} - (1-x_{i0})M \qquad \forall i \in \{1,2,...,k\} \tag{5.8}$$

$$y_{ij} \geq w_j - (1-x_{ij})M \qquad \forall i,j \in \{0,1,2,...,k\}, i \neq j, j = i+1 \tag{5.9}$$

$$y_{ij} \leq w_j + (1-x_{ij})M \qquad \forall i,j \in \{0,1,2,...,k\}, i \neq j, j = i+1 \tag{5.10}$$

$$z_i \leq (i{+}1)\delta \qquad \forall i \in \{1, 2, ..., k\} \tag{5.11}$$

$$z_i \leq (k{-}i)\delta \qquad \forall i \in \{1, 2, ..., k\} \tag{5.12}$$

$$w_j \geq P_j{-}(2(k{-}1)\varepsilon{+}2k\delta{+}\textstyle\sum_{f=0,f\neq j-1,f\neq j}^{k} MRP_f) \qquad \forall i \in \{0, 1, 2, ..., k\} \tag{5.13}$$

$$y_{ij} \leq Mx_{ij} \qquad \forall i, j \in \{0, 1, 2, ..., k\}, i \neq j, j = i{+}1 \tag{5.14}$$

$$\textstyle\sum_{i=1}^{k} x_{ij} = 1 \qquad \forall j \in \{0, 1, 2, ..., k\} \tag{5.15}$$

$$\textstyle\sum_{i=1}^{k} x_{ij} = 1 \qquad \forall i \in \{0, 1, 2, ..., k\} \tag{5.16}$$

$$y_{ij} \geq 0 \qquad \forall i, j \in \{0, 1, 2, ..., k\}, i \neq j, j = i + 1 \tag{5.17}$$

$$w_j \geq 0 \qquad \forall j \in \{0, 1, 2, ..., k\} \tag{5.18}$$

$$x_{ij} = \{0, 1\} \qquad \forall i, j \in \{0, 1, 2, ..., k\}, i \neq j \tag{5.19}$$

This MILP model concentrates on minimizing the cycle time (T) so that it meets Constraints (5.6) to (5.19). The state space in Inequality (5.6) is the same as described in case *a*, and consequently results in the full waiting. In fact, this inequality assure us that the completion time of $A_j, j = i + 1$, is never less than the summation of the completion time of $A_i(c_i)$, the processing time of $M_j(P_j)$, the unloading time of M_j, the busy MFR travel time between two consecutive machines M_j and M_{j+1}, and loading time of $M_{j+1}(2\varepsilon + MRP_j)$. Note that M is an infinitive number making this inequality redundant if $j \neq i + 1$. Likewise, Inequality (5.7) covers the state space of case *b*, and gives a guarantee that the completion time of $A_j, j \neq i + 1$, is certainly bigger than the summation of the completion time of $A_i(c_i)$, the empty MFR movement from M_{i+1} to $M_j(min\{|i{+}1{-}j|, k{+}1{-}|i{+}1{-}j|\}\delta)$, the partial waiting on $M_j(y_{ij})$, the unloading time of M_j, the busy MFR travel time between two consecutive machines M_j and M_{j+1}, and loading time of $M_{j+1}(2\varepsilon + MRP_j)$. y_{ij} is specified by Inequalities (5.9), (5.10), (5.14), and (5.17). This variable is defined in order to show that there is a waiting time before starting A_j with respect to the previous activity A_i. The first couple of inequalities are applicable and imply that $y_{ij} = w_j$ if x_{ij}=1. This means that there is a waiting time equals $y_{ij} = w_j$ before starting A_j when A_i is the preceding activity. On the other hand, the last couple are applicable and states that y_{ij}=0 if x_{ij}=0. In fact, when A_i does not immediately precede $A_j, y_{ij} = 0$ indicates that the waiting time before starting A_j is not dependent on A_i. On the contrary to y_{ij} which is activity-dependent, the waiting time of M_j which is $w_j = max\{0, P_j - (2(k-1)\varepsilon + 2k\delta + \sum_{f=0,f\neq j-1,f\neq j}^{k} MRP_f)\}$ is indicated by Inequalities (5.13) and (5.18). Note that w_j only represents the waiting time of M_j regardless previous activity. This means that w_j is independent of the preceding activity. Due to the fact that the first activity of each one of feasible permutations is A_0, Inequality (5.8) implies that each obtained cycle time cannot be less than the starting time of the next implementation of this permutation. This time is totally the summation of the completion time of the final activity (c_i), the elapsed time of the empty MFR movement from M_{i+1} to $I/O(z_i)$. According to Inequality (5.11) and (5.12), $z_i = min\{|i+1-0|, k+1-|i+1-0|\}\delta = min\{i+1, k-i\}\delta$. Equation (5.15) ensures that each activity j must succeed one activity i. Likewise, Equation (5.16) implies that each activity i must precede one activity j. Equation (5.19) characterizes x_{ij} as the binary variable meaning whether A_j succeeds A_i without delay or not. This can be described as follows:

$$x_{ij} = \begin{cases} 1 & \text{if } A_j \text{ immediately succeeds } A_i \\ 0 & \text{otherwise} \end{cases}$$

The above model is much more complex than the classical TSP. As a result, the concentration in this paper is on the analysis of two very productive permutations namely uphill and downhill.

5.5 Lower Bound and Optimal MFR's Permutation for MFRCs

In mathematical literature, a lower bound of a set is the element which is less than or equal to every element of this set. In other words, a set with a lower bound is occasionally said to be bounded from bellow by a bound if have the lower bound. Because of the fact that MFRC scheduling problem arises in mass production environments, this problem is exploit MFR permutations in an optimal way. If the lower bound of MFR cycle time can be obtained in the first step, it is considerably easier than to find the optimal permutation in the next step. Consequently, the theorem below initially represents the lower bounding procedure for MFRC:

Theorem 1. The lower bound of cycle time for $R_{mf}C_k^O|\delta_i = \delta, \varepsilon_i = \varepsilon, cyclic-1|C_t$ with the part processing route $RO_0, O_1, RO_1, O_2, RO_2, O_3, ..., RO_k$ is:

$$\underline{T}_m = max\{2(k+1)\varepsilon + \sum_{i=0}^{k} MRP_i \\ + \sum_{i=1}^{k} min\{P_i, \delta\}, 4\varepsilon + 2\delta + MRP_{l(m)-1} + P_{l(m)} + MRP_{l(m)}\} \tag{5.20}$$

Proof: A one-unit permutation completes one part with the processing route $RO_0, O_1, RO_1, O_2, RO_2, O_3, ..., RO_k$. Hence, MFR unloads the part from I/O and k machines once $((k+1)\varepsilon)$, and loads it onto k machines and I/O once $((k+1)\varepsilon)$, meaning that overall loading/unloading time of the one-unit permutation is at least $(2(k+1)\varepsilon)$. The busy MFR has at least a series of $k+1$ clockwise transposition. Since travel time between adjacent location pairs (M_i, M_{i+1}) is MRP_i, elapsed time by occupied MFR to perform these $k+1$ clockwise transposition is $\sum_{i=0}^{m} MRP_i$. Recall that MFR must start unloading a part of either M_i or another machine after it finished loading a part on M_i. For the first and second cases, full waiting time at M_i and partial waiting time at another machine are at least P_i and δ, respectively. The minimum value of these waiting times is the overall waiting time of M_i, which for all k machines this sum up $\sum_{i=1}^{k} min\{P_i, \delta\}$.

The last part of Equation (5.20) implies that the cycle time of any permutations is at least as big as the time between two successive unloading of each one of machines. As shown in Figure 5.5, the activities between two successive unloading of M_i and its time are at least: unloading a part from $M_i(\varepsilon)$, carrying the part to $M_{i+1}(MRP_i)$, loading this part at $M_{i+1}(\varepsilon)$, transferring the empty MFR to $M_{i-1}(2\delta)$, unloading another part from $M_{i-1}(\varepsilon)$, carrying the part to $M_i(MRP_{i-1})$, loading this part at $M_i(\varepsilon)$, and finally full waiting at $M_i(P_i)$.

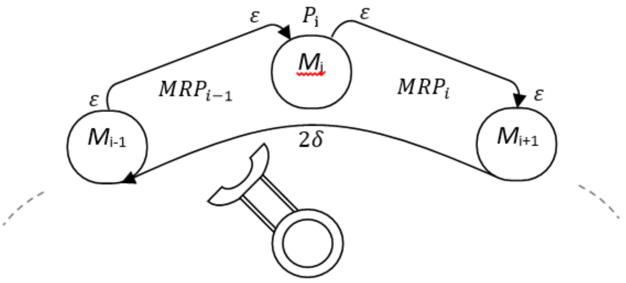


Fig. 5.5. The graph of two successive unloading of M_i

It should be noted that the time elapsed during two successive unloading process of the machine with dominant processing time must be considered. Subsequently, the last part of Equation (5.20) is $4\varepsilon + 2\delta + MRP_{l(m)-1} + P_{l(m)} + MRP_{l(m)}$. This completes the proof.

The performance improvement of two MFR permutations, namely uphill and downhill permutations, is determined here. These two permutations, which jointly include a remarkable number of feasible solutions, employed in SFRC-related researches due to programmability and practicality. For a MFRC, the sequences of activities for uphill and downhill permutations are $_{V(m)} = (A_0, A_1, A_2, ..., A_{k-1}, A_k)$

and $_{D(m)} = (A_0, A_k, A_{k-1}, ..., A_2, A_1)$, respectively. For analyzing the performance of $_{V(m)}$ and $_{D(m)}$, initially, their cycle time are given:

Lemma 1. The cycle time of the uphill permutation $_{V(m)}$ for a rotationally arranged MFRC is represented as below:

$$T_{\Pi_{v(m)}} = 2(k+1)\varepsilon + \textstyle\sum_{i=0}^{k} MRP_i + \sum_{i=1}^{k} P_i \tag{5.21}$$

Proof: The activity's route of this permutation only consist uphill activities. Thus, the cycle time is composed of three components: $2(k+1)$ loading/unloading operations $(2(k+1)\varepsilon)$, k+1 clockwise busy MFR transpositions $(\sum_{i=0}^{k} MRP_i)$, and eventually k full waiting on k corresponding machines $(\sum_{i=1}^{k} P_i)$. It should be emphasized that the empty MFR movement is zero in that there is not any partial waiting. This completes the proof.

Lemma 2. For a rotationally arranged MFRC, the cycle time of the downhill permutation $_{D(m)}$ where $MRP_{l(m)-1} + P_{l(m)} + MRP_{l(m)} \geq MRP_{i-1} + P_i + MRP_i, \forall l, i \in \{1, 2, ..., k\}$ is expressed by

$$\begin{aligned} T_{\Pi_{D(m)}} &= 2(k+1)(\varepsilon + \delta) + \textstyle\sum_{i=0}^{k} MRP_i \\ &\quad + max\{0, P_{l(m)} - 2(k-1)\varepsilon - 2k\delta - \textstyle\sum_{i=0, i \neq l-1, i \neq l}^{k} MRP_i\} \end{aligned} \tag{5.22}$$

Proof: On the contrary with the uphill permutation, the activity's route of this permutation only consist downhill activities. The cycle time of downhill permutation is composed of four components. The first component is $2(k+1)$ loading/unloading operations in a similar manner taking $2(k+1)\varepsilon$. The second component is $2(k+1)$ empty movement in that MFR performs $2k$ empty movement from M_{i+1} to $M_{i-1}, \forall l, i \in \{1, 2, ..., k\}$ under activity route A_i, A_{i-1}, and two empty movement from M_1 to M_k under activity route A_0, A_k. The third component is $\sum_{i=0}^{k} MRP_i$ since downhill permutation is a one-unit permutation with $k+1$ busy MFR movement. Until this stage, it is apparent that $T_{\Pi_{D(m)}} \geq 2(k+1)(\varepsilon + \delta) + \sum_{i=0}^{k} MRP_i$. The last component is the overall partial waiting time. It is enough to subtract $P_j, \forall j \in \{1, 2, ..., k\}$, from entire time except time M_j is unoccupied to calculate partial waiting time on this arbitrary machine. M_j is empty for $4\varepsilon + 2\delta + MRP_{j-1} + MRP_j + w_{j-1} + w_j$, which means:

$$w_j = max\{0, P_j - 2(k-1)\varepsilon - 2k\delta - \textstyle\sum_{i=0, i \neq j-1, i \neq j}^{k} MRP_i - \sum_{i=1, i \neq j-1, i \neq j}^{k} w_i\} \tag{5.23}$$

It is known that M_l is machine with dominant processing time. Therefore, $w_l \geq w_i, \forall i \in \{1, 2, ..., k\}$, which means that it is enough to only consider the waiting time of M_l.

$$w_l = max\{0, P_{l(m)} - 2(k-1)\varepsilon - 2k\delta - \textstyle\sum_{i=0,i\neq l-1,i\neq l}^{k} MRP_i - \sum_{i=1,i\neq l-1,i\neq l}^{k} w_i\} \tag{5.24}$$

$$\rightarrow \textstyle\sum_{i=1}^{k} w_i = max\{0, P_{l(m)} - 2(k-1)\varepsilon - 2k\delta - \sum_{i=0,i\neq l-1,i\neq l}^{k} MRP_i\} \tag{5.25}$$

This completes the proof.

Now, it is possible to show when uphill and downhill permutations can be separately optimal permutation:

Theorem 2. If $P_i \leq \delta, \forall i \in \{1,2,...,k\}$ in a rotationally arranged MFRC, the optimal permutation is $_{v(m)}$ with uphill activity route.

Proof: If $P_i \leq \delta, \forall i \in \{1,2,...,k\}$ in Equation (5.20), then $4\varepsilon + 2\delta + MRP_{l(m)-1} + P_{l(m)} + MRP_{l(m)} \leq 4\varepsilon + 3\delta + MRP_{l(m)-1} + MRP_{l(m)}$. Also, it is known that $4\varepsilon + 3\delta + MRP_{l(m)-1} + MRP_{l(m)} \leq 2(k+1)\varepsilon + \sum_{i=0}^{k} MRP_i + \sum_{i=1}^{k} min\{P_i, \delta\}$, and $\sum_{i=1}^{k} min\{P_i, \delta\} = \sum_{i=1}^{k} P_i$. As a result, $\underline{T}_m = 2(k+1)\varepsilon + \sum_{i=0}^{k} MRP_i + \sum_{i=1}^{k} P_i = T_{\Pi_{V(m)}}$, meaning $T_{\Pi_{V(m)}}$ is the optimal permutation when $P_i \leq \delta, \forall i \in \{1,2,...,k\}$ in MFRC. This completes the proof.

Theorem 3. If $P_{l(m)} \geq 2(k-1)\varepsilon + 2k\delta + \sum_{i=0,i\neq l-1,i\neq l}^{k} MRP_i$ in a rotationally arranged MFRC, then the optimal permutation is $\Pi_{D(m)}$ with downhill activity route.

Proof: For simplicity, considering $2(k-1)\varepsilon + 2k\delta + \sum_{i=0,i\neq l-1,i\neq l}^{k} MRP_i = \zeta$ and $2(k-1)\varepsilon + 2k\delta + \sum_{i=0,i\neq l-1,i\neq l}^{k} MRP_i = \eta$ Equation (5.22) is rewritten as Equation (5.26).

$$T_{\Pi_{D(m)}} = \begin{cases} 2(k+1)(\varepsilon+\delta) + \sum_{i=0}^{m} MRP_i & \text{if } P_{l(m)} \leq \zeta \\ 4\varepsilon + 2\delta + MRP_{l(m)-1} + P_{l(m)} + MRP_{l(m)} & \text{if } P_{l(m)} \geq \eta \end{cases} \tag{5.26}$$

If $P_{l(m)} \geq 2(k-1)\varepsilon + 2k\delta + \sum_{i=0,i\neq l-1,i\neq l}^{k} MRP_i$ in Equation (5.20), one then easily obtains $2(k+1)\varepsilon + \sum_{i=0}^{k} MRP_i + \sum_{i=1}^{k} min\{P_i, \delta\} \leq 4\varepsilon + 2\delta + MRP_{l(m)-1} + P_{l(m)} + MRP_{l(m)}$. As a consequence, $\underline{T}_m = 4\varepsilon + 2\delta + MRP_{l(m)-1} + P_{l(m)} + MRP_{l(m)} = T_{\Pi_{D(m)}}$, meaning $T_{\Pi_{D(m)}}$ is the optimal permutation under this condition. This completes the proof.

5.6 The Comparison of SFRCs and MFRCs

In this section, the performance gaps between different cases of rotationally arranged SFRCs and MFRCs are generally determined, and consequently a preliminary analysis identifies the regions where the productivity gain of a regular MFRC is more than that of the corresponding SFRC. In a general manner, a regular MFRC originates from a SFRC with the processing route $O_1, O_2, O_3, O_4, ..., O_k$. More precisely, The odd-numbered operations, $O_1, O_3, ..., O_k$, and even-numbered operations are performed by MFR and $\lceil (k-1)/2 \rceil$ machines in the regular MFRCs, respectively. In the first glance at regular MFRCs, the following questions arise: Why only the odd-numbered operations must be assigned to MFR? Is not it possible to assign any operations to MFR? In answer to these questions, it must be mentioned that the odd-numbered operations have same nature, whereas the even-numbered operations are completely different in nature. For instance, all odd-numbered operations can be quality control operation, but even-numbered operations can be marking, cutting, welding and grinding operations. Therefore, there is an inspection machine just before and after a processing on one of the marking, cutting, welding and grinding machines. There are two options to perform quality control operations and complete a part under this condition: either by MFR or by original inspection machines. First case leads to a regular MFRC where there are fewer machines in manufacturing cell, while the latter one leads to a SFRC. The number of machine in a regular MFRC is $\lceil (k-1)/2 \rceil$ in that one inspection machine is removed from between any two machines with different nature which existed in the original manufacturing cell, SFRC. Using a combination of these two alternatives is not economical. That is, either all of odd-numbered operations having same nature are performed by MFR or by machines. It is not possible to have a system where some of odd-numbered operations are performed by MFR while some others are performed by machines. A regular MFRC given in Figure 5.6 originating from SFRC presented in Figure 5.2. Despite the fact that these figures completely look similar, Figure 5.6 represents a MFR performing processes of $M_1, M_3, M_5, ..., M_{2\lceil (k-1)/2 \rceil -1}, M_{2\lceil (k+1)/2 \rceil -1}$. More precisely, same again, it is vital to complete k operations to finish a part in that these two kinds manufacturing cells are designed to produce identical parts. The only difference is that $M_2, M_4, M_6, ..., M_{2\lceil (k-3)/2 \rceil}, M_{2\lceil (k-1)/2 \rceil}$ are designed to perform $\lceil (k-1)/2 \rceil$ even-numbered operations and MFR is designed to perform $\lceil (k+1)/2 \rceil$ remaining odd-numbered operations in MFRCs. It is apparent that $P_{2\lceil (k+1)/2 \rceil -1}$=0 if O_k is an even-numbered operation. The reason is that the processing route is in a manner which $M_{2\lceil (k-1)/2 \rceil}$ performs the last operation and MFR only transfers finished part from $M_{2\lceil (k-1)/2 \rceil}$ to I/O.

The focus lies here mainly on improvement of real-life manufacturing systems; thus, the developed framework is based on the problem of scheduling a common

SFRC, namely the hub SFRC. Similarly, Kubiak et al. (1996); Foumani et al. (2013c) considered the processing route of each part is a chain of k operations. The industrial limitations require that the part visit alternate machines with same nature between $M_2, M_4, M_6, ..., M_{2\lceil (k-3)/2\rceil}, M_{2\lceil (k-1)/2\rceil}$. Accordingly, machines with same nature are substituted with a MFR here. Also, the applications of the regular MFRCs exist in many high-tech industries.

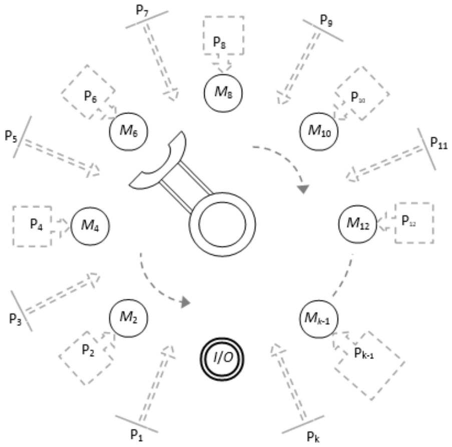


Fig. 5.6. MFRC with $\lceil (k-1)2 \rceil$ machines

If ideal conditions hold, a simple three-machine SFRC can be swapped with a single-machine MFRC. Two following examples distinguish between two different styles of replacements.

Example 2. A small scale example, a three-machine SFRC, is given here. In this manufacturing cell, it is possible to replace M_1 and M_3 with a MFR. Assume, for example, that three operations O_1, O_2 and O_3 with processing times P_1=5, P_2=3 and P_3=4 are performed to produce a single outgoing part. Also, Assume that ε=1 and δ=2. Since this rotationally arranged SFRC contains three machines, there are 3! possible permutations below:

$$S_{1(s)} = A_0, A_1, A_2, A_3$$
$$S_{2(s)} = A_0, A_2, A_1, A_3$$
$$S_{3(s)} = A_0, A_1, A_3, A_2$$
$$S_{4(s)} = A_0, A_3, A_1, A_2$$
$$S_{5(s)} = A_0, A_2, A_3, A_1$$
$$S_{6(s)} = A_0, A_3, A_2, A_1$$

The cycle time of these one-unit permutations are (Sethi et al., 2001).

$T_{S_{1s}} = 8\varepsilon + 4\delta + P_1 + P_2 + P_3$ (5.27)

$T_{S_{2s}} = max\{8\varepsilon+8\delta, 6\varepsilon+4\delta+P_1, 4\varepsilon+4\delta+P_2, 6\varepsilon+4\delta+P_3, 4\varepsilon+2\delta+\dfrac{P_1+P_2+P_3}{2}\} =$
24 (5.28)

$T_{S_{3s}} = max\{8\varepsilon + 8\delta + P_1, 4\varepsilon + 4\delta + P_3, 6\varepsilon + 4\delta + P_1 + P_2\} = 29$ (5.29)

$T_{S_{4s}} = max\{8\varepsilon + 8\delta + P_2, 6\varepsilon + 4\delta + P_2 + P_3, 6\varepsilon + 4\delta + P_1 + P_2\} = 27$ (5.30)

$T_{S_{5s}} = max\{4\varepsilon + 4\delta + P_1, 8\varepsilon + 8\delta + P_3, 6\varepsilon + 4\delta + P_2 + P_3\} = 28$ (5.31)

$T_{S_{6s}} = max\{8\varepsilon + 8\delta, 8\varepsilon + 12\delta, 4\varepsilon + 4\delta + P_1, 4\varepsilon + 4\delta + P_2, 4\varepsilon + 4\delta + P_3\} = 32$

(5.32)

Employing a MFR instead of M_1 and M_3, the three-machine SFRC is converted into a single-machine MFRC with the part processing route RO_0, O_1, RO_1. Notice that processing times of operations RO_0, O_1, RO_1 are $RP_0 = 5, P_1 = 3$ and RP_1=4, respectively. Clearly, the single-machine MFRC has a permutation $S_{1(m)} = (A_0, A_1)$. Hence, this unique permutation consists the sequence below: unloading a part from I/O and loading it to the single machine in MFRC $(\varepsilon + MRP_0 + \varepsilon)$, waiting for competition of the operation (P_1), and eventually unloading the part from machine and carrying it to $I/O(\varepsilon + MRP_1 + \varepsilon)$. The cycle time of this permutation is $T_{S_{1(m)}}$=16. By decreasing cycle time about eight units in rotationally arranged MFRC, the productivity gain of MFRC shows 33.3% improvement, which means using a MFRC instead of a SFRC is economical in this example.

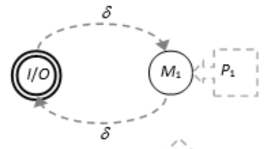
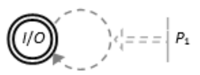
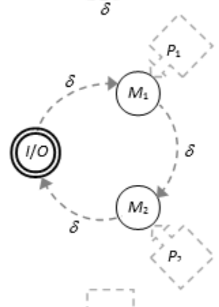
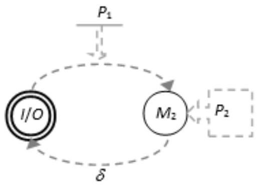
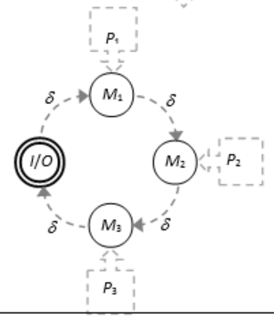
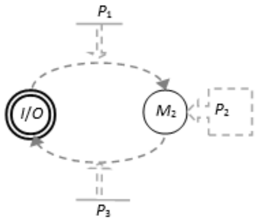
Example 3. Similar to Example 2, a three-machine SFRC is given. Likewise, the processing time of O_1 and O_3 are P_1=5 and P_3=4. However, the processing time of O_2 is P_2=23. Due to 20 unit increase in P_2, the cycle time of six possible one-unit permutations are: $T_{S_{1(s)}}$=48, $T_{S_{2(s)}}$=35, $T_{S_{3(s)}}$=42, $T_{S_{4(s)}}$=47, $T_{S_{5(s)}}$=41 and $T_{S_{6(s)}}$=35. For replaced MFRC, the cycle time of the unique permutation increases to $T_{S_{1(m)}}$=36. As a result of Example 3, the three-machine SFRC is 0.3% more productivity than the single-machine MFRC since $T_{S_{2(s)}} = T_{S_{6(s)}} = 35 < T_{S_{1(m)}} = 36$, meaning that applying MFRC for this example is not economical.

The numerical examples 2 and 3 jointly clarify the statement that replacing a rotationally arranged SFRC with a rotationally arranged MFRC is not always an attractive production strategy, even for a three-machine SFRC. Consequently, the productivity gain of using a rotationally arranged MFRC instead of a rotationally arranged SFRC must be precisely measured. Initially, this analysis is carried out for small-scale MFRCs due to their efficaciously and extendibility, and then it is finalized for large-scale ones.

5.6.1 The Comparison of Small-Scale SFRCs with MFRCs

Many studies concentrated on small-scale SFRCs, especially SFRCs with one, two and three machines. Even recently, small-scale cases have drawn much more consideration because they are efficacious, easy to control, and extendable to complicated cells (Geismar et al., 2012; Jolai et al., 2012; Zarandi et al., 2013). Furthermore, when the part processing routes in large-scale SFRCs is complicated, one of the most economic strategies in order to deals with this complexity is breaking these SFRCs into small-scale clusters. A SFR serves within each one of newly designed clusters which generally consist of one, two or three machines (Chan et al., 2008).

Table 5.1: SFRCs with one, two, three machines and extracted MFRCs from them

Number of machines	Configuration of SFRC	Configuration of extracted MFRC
Single machine		
Two machines		
Three machines		

SFRCs with one, two, three machines and extracted MFRCs from anyone of them are shown in Table 5.1. For a single machine SFRC which is the simplest possible SFRC, $S_{1(s)} = (A_0, A_1)$ is the unique permutation. Also, the extracted MFRC does not have any machines and only consists of a MFR that performs O_1 in front of I/O and without any travel. Therefore, we have:

$$T_{S_{1s}} = 4\varepsilon + 2\delta + P_1 \tag{5.33}$$
$$T_{S_{1m}} = 2\varepsilon + P_1 \tag{5.34}$$

This means that the use of MFRCs instead of single-machine SFRCs always reduce cycle time, especially for single-machine SFRCs that have big SFR travel-times. There are two permutations $S_{1(s)} = (A_0, A_1, A_2)$ and $S_{2(s)} = (A_0, A_2, A_1)$ for a two-machine rotationally arranged SFRC, and the unique permutation $S_{1(m)} = (A_0, A_1)$ for the extracted MFRC. As a result, the cycle time of these permutations are:

$$T_{S_{1s}} = 6\varepsilon + 3\delta + P_1 + P_2 \tag{5.35}$$
$$T_{S_{2s}} = 6\varepsilon + 6\delta + max\{0, P_1 - (2\varepsilon + 3\delta), P_2 - (2\varepsilon + 3\delta)\} \tag{5.36}$$
$$T_{S_{1m}} = 4\varepsilon + \delta + max\{\delta, P_1\} + P_2 \tag{5.37}$$

Equations (5.35) and (5.37) implies that $T_{S_{1(m)}} < T_{S_{1(s)}}$. However, $T_{S_{1(m)}} > T_{S_{2(s)}}$ if $P_1 > 2\delta$, $P_2 > 2\delta$ and $P_1 + P_2 > 2\varepsilon + 5\delta$. Therefore, the result of this comparison shows the assumption that the use of MFRCs instead of two-machine SFRCs improves cycle time is only wrong when $P_1 > 2\delta, P_2 > 2\delta$ and $P_1 + P_2 > 2\varepsilon + 5\delta$. A SFRC with three machines has six possible permutations as mentioned earlier. The cycle time of them are also shown in Equations (5.27) to (5.32). It is essential to determine the regions of optimality for anyone of them in Theorem 4 before comparing these permutations with the permutation of extracted MFRC.

Theorem 4. For a three-machine rotationally arranged SFRC, the regions of optimality of permutations are as follows:

$$S^* = S_{1(s)} \leftrightarrow P_1 + P_2 + P_3 \leq 4\delta \tag{5.38}$$
$$S^* = S_{2(s)} \leftrightarrow P_1 \leq 2\varepsilon + 8\delta, P_3 \leq 2\varepsilon + 8\delta, |P_1 - P_3| \leq max\{2\varepsilon + 4\delta, P_2\}, 4\delta < P_1 + P_2 + P_3 \leq 8\varepsilon + 20\delta \tag{5.39}$$
$$S^* = S_{3(s)} \leftrightarrow P_1 \leq 4\delta \leq P_2 + P_3, P_1 - P_3 < -max\{2\varepsilon + 4\delta, P_2\}, P_1 + P_2 \leq 2\varepsilon + 8\delta \tag{5.40}$$
$$S^* = S_{5(s)} \leftrightarrow P_3 \leq 4\delta \leq P_1 + P_2, P_1 - P_3 > max\{2\varepsilon + 4\delta, P_2\}, P_2 + P_3 \leq 2\varepsilon + 8\delta \tag{5.41}$$
$$S^* = S_{6(s)} \leftrightarrow P_1 > 4\delta, P_3 > 4\delta \tag{5.42}$$

Proof: Due to the fact that the method of determining the regions of optimality is similar in each of six permutations, this method is only presented for $S_{1(s)}$ here.

For simplicity, let max_{ij} be the $j^{th}max$ term in the i^{th} permutation where the max terms are arranged as in the formulas given by Equations (5.27) to (5.32). Giving two simple examples, max_{11} and max_{53} are $8\varepsilon + 4\delta + P_1 + P_2 + P_3$ and $6\varepsilon + 4\delta + P_2 + P_3$, respectively. Therefore,

$$\begin{array}{ll} max_{11} \le max_{21} \leftrightarrow P_1 + P_2 + P_3 \le 4\delta & \text{possible} \\ max_{11} \le max_{22} \leftrightarrow P_2 + P_3 \le -2\varepsilon & \text{impossible} \\ max_{11} \le max_{23} \leftrightarrow P_1 + P_3 \le -4\varepsilon & \text{impossible} \\ max_{11} \le max_{24} \leftrightarrow P_1 + P_2 \le -2\varepsilon & \text{impossible} \\ \underbrace{max_{11} \le max_{25} \leftrightarrow P_1 + P_2 + P_3 \le -(8\varepsilon + 4\delta)}_{S_{1(s)} \le S_{2s)} \leftrightarrow P_1+P_2+P_3 \le 4\delta} & \text{impossible} \end{array} \tag{5.43}$$

$$\begin{array}{ll} max_{11} \le max_{31} \leftrightarrow P_2 + P_3 \le 4\delta & \text{possible} \\ max_{11} \le max_{32} \leftrightarrow P_1 + P_2 \le -4\varepsilon & \text{impossible} \\ \underbrace{max_{11} \le max_{33} \leftrightarrow P_3 \le -2\varepsilon}_{S_{1(s)} \le S_{3(s)} \leftrightarrow P_2+P_3 \le 4\delta} & \text{impossible} \end{array} \tag{5.44}$$

$$\begin{array}{ll} max_{11} \le max_{41} \leftrightarrow P_1 + P_3 \le 4\delta & \text{possible} \\ max_{11} \le max_{42} \leftrightarrow P_1 \le -2\varepsilon & \text{impossible} \\ \underbrace{max_{11} \le max_{43} \leftrightarrow P_3 \le -2\varepsilon}_{S_{1(s)} \le S_{4(s)} \leftrightarrow P_1+P_3 \le 4\delta} & \text{impossible} \end{array} \tag{5.45}$$

$$\begin{array}{ll} max_{11} \le max_{51} \leftrightarrow P_2 + P_3 \le -4\varepsilon & \text{possible} \\ max_{11} \le max_{52} \leftrightarrow P_1 + P_2 \le 4\delta & \text{impossible} \\ \underbrace{max_{11} \le max_{53} \leftrightarrow P_1 \le -2\varepsilon}_{S_{1(s)} \le S_{5(s)} \leftrightarrow P_1+P_2 \le 4\delta} & \text{impossible} \end{array} \tag{5.46}$$

$$\begin{array}{ll} max_{11} \le max_{61} \leftrightarrow P_1 + P_2 + P_3 \le 8\delta & \text{possible} \\ max_{11} \le max_{62} \leftrightarrow P_2 + P_3 \le -4\varepsilon & \text{impossible} \\ max_{11} \le max_{63} \leftrightarrow P_1 + P_3 \le -4\varepsilon & \text{impossible} \\ \underbrace{max_{11} \le max_{64} \leftrightarrow P_1 + P_2 \le -4\varepsilon}_{S_{1(s)} \le S_{6s)} \leftrightarrow P_1+P_2+P_3 \le 8\delta} & \text{impossible} \end{array} \tag{5.47}$$

The common region where $S_{1(s)}$ dominates all permutations should be obtained in order to present $S_{1(s)}$ as the optimal permutation. This region is the intersection of Inequalities (5.43)-(5.47), which is $P_1 + P_2 + P_3 \le 4\delta$. This completes the proof.

The above theorem implies hereinafter that $S_{4(s)}$ should be eliminated from analysis since it has no chance of optimality. Also, it is only possible that $S_{2(s)}$

and $S_{6(s)}$ be optimal at the same time. The intuition for the optimality of $S_{1(m}$ in comparison with the rest of SFRCs permutations which have the chance of optimality is given in Theorem 5.

Theorem 5. The use of MFRCs is better than three-machine SFRCs if one of conditions below hold:

1. $P_1 + P_2 + P_3 \leq 4\delta$
2. $P_1 \leq 2\varepsilon + 8\delta$
   $P_3 \leq 2\varepsilon + 8\delta$
   $|P_1 - P_3| \leq max\{2\varepsilon + 4\delta, P_2\}$
   $4\delta < P_1 + P_2 + P_3 \leq 8\varepsilon + 20\delta$
   $max\{\delta, P_1\} + P_2 + max\{\delta, P_3\} \leq max\{4\varepsilon + 8\delta, 2\varepsilon + 4\delta + P_1, 4\delta + P_2, 2\varepsilon + 4\delta + P_3, \dfrac{P_1 + P_2 + P_3 + \delta}{2}\}$
3. $P_1 \leq 4\delta \leq P_2 + P_3$
   $P_1 - P_3 < -max\{2\varepsilon + 4\delta, P_2\}$
   $P_1 + P_2 \leq 2\varepsilon + 8\delta$
   $max\{\delta, P_1\} + P_2 + P_3 \leq max\{4\varepsilon + 8\delta + P_1, 2\varepsilon + 4\delta + P_1 + P_2, 4\delta + P_3\}$
4. $P_3 \leq 4\delta \leq P_1 + P_2$
   $P_1 - P_3 > max\{2\varepsilon + 4\delta, P_2\}$
   $P_2 + P_3 \leq 2\varepsilon + 8\delta$
   $P_1 + P_2 + max\{\delta, P_3\} \leq max\{4\varepsilon + 8\delta + P_3, 2\varepsilon + 4\delta + P_2 + P_3, 4\delta + P_1\}$

Proof: With Theorem 4, it is proved that only five permutations $S_{1(s)}, S_{2(s)}, S_{3(s)}$, $S_{5(s)}$, and $S_{6(s)}$ can be optimal. Hence, for anyone of these permutations, it is only necessary to find the intersection of the regions of optimality and the region where it is dominated by $S_{1(m)}$ (See the following table):

Table 5.2: Intersection of optimality regions of cycles & domination region of $S_{1(m)}$

Cycle	Optimality	Dominance Relation of $S_{1(m)}$ resulting from pairwise comparisons	Intersect
$S_{1(s)}$	Eq.(5.38)	Always	Cond.1
$S_{2(s)}$	Eq.(5.39)	$max\{\delta, P_1\}+P_2+max\{\delta, P_3\} \leq max\{4\varepsilon+8\delta, 2\varepsilon+4\delta+P_1, 4\delta+P_2, 2\varepsilon+4\delta+P_3, \dfrac{P_1 + P_2 + P_3 + \delta}{2}\}$	Cond.2
$S_{3(s)}$	Eq.(5.40)	$max\{\delta, P_1\} + P_2 + P_3 \leq max\{4\varepsilon + 8\delta + P_1, 2\varepsilon + 4\delta + P_1 + P_2, 4\delta + P_3\}$	Cond.3
$S_{5(s)}$	Eq.(5.41)	$P_1 + P_2 + max\{\delta, P_3\} \leq max\{4\varepsilon + 8\delta + P_3, 2\varepsilon + 4\delta + P_2 + P_3, 4\delta + P_1\}$	Cond.4
$S_{6(s)}$	Eq.(5.42)	Never	No

This completes the proof.

Our task now is to show that when complicated MFRCs can have better performance in comparison to SFRCs. In the following, we present our intuition for the optimality of the large-scale MFRCs.

5.6.2 The Comparison of Large-Scale SFRCs with MFRCs

The analysis is carried out in Theorems 7 and 8 where dominant permutations are considered to be $_{v(m)}$ and $_{D(m)}$, respectively. Prior to that, Theorems 6 which shows the lower bound of a cycle time when the rotationally arranged SFRC is substituted with the rotationally arranged MFRC is presented.

Theorem 6. If a rotationally arranged MFRC with $\lceil(k-1)/2\rceil$ machines is used instead of a rotationally arranged SFRC with k machines, the lower bound of the cycle time for the problem $R_{mf}C^{O}_{\lceil(k-1)/2\rceil}|\delta_i = max\{RP_i, \delta\}, \varepsilon_i = \varepsilon, cyclic-1|C_t$ is:

$$T_{s\to m} = max\{2\lceil\frac{k+1}{2}\rceil\varepsilon + \sum_{i=1}^{\lceil\frac{k+1}{2}\rceil} max\{P_{2i-1}, \delta\} + \sum_{i=1}^{\lceil\frac{k-1}{2}\rceil} min\{P_{2i}, \delta\}, 4\varepsilon + 2\delta + max_{1\leq i\leq\lceil\frac{k-1}{2}\rceil}\{max\{P_{2i-1}, \delta\} + P_{2i} + max\{P_{2i+1}, \delta\}\}\} \tag{5.48}$$

Proof: MFR unloads a part from $I/O, M_2, M_4, M_6, ..., M_{2\lceil(k-3)/2\rceil}$ and $M_{2\lceil(k-1)/2\rceil}$ and loads this part on $M_2, M_4, M_6, ..., M_{2\lceil(k-3)/2\rceil}, M_{2\lceil(k-1)/2\rceil}$ and I/O once, taking time $\lceil(k+1)/2\rceil\varepsilon$ and $\lceil(k+1)/2\rceil\varepsilon$, respectively. This means that the overall loading/unloading execution time is $2\lceil(k+1)/2\rceil\varepsilon$. The number of occupied MFR transpositions between adjacent even-numbered machines in original system, denoting by adjacent location pairs $(M_{2i}, M_{2(i+1)})$, i=1, $\lceil(k-3)/2\rceil$, is equal to $\lceil(k-3)/2\rceil$ by decreasing the number of machines to $\lceil(k-1)/2\rceil$ in MFRC. In fact, $\lceil(k-3)/2\rceil$ indicates the number of intermediate odd-numbered processes $O_3, O_5, ..., O_{2\lceil(k-3)/2\rceil-1}, O_{2\lceil(k-1)/2\rceil-1}$ between these machines in the original system. Furthermore, there are two additional occupied MFR transpositions denoting by adjacent location pairs $(I/O, M_2)$ and $(M_{2\lceil(k-1)/2\rceil}, I/O)$ for operations O_1 and $O_{2\lceil(k+1)/2\rceil-1}$. Apparently, the overall occupied MFR travel time is $\sum_{i=1}^{\lceil(k+1)/2\rceil} max\{P_{2i-1}, \delta\}$. For the first portion of Equation (48), finally, $\sum_{i=1}^{\lceil(k-1)/2\rceil} min\{P_{2i}, \delta\}$ implies the least full/partial waiting time of all $\lceil(k-1)/2\rceil$ even-numbered production machines. For the last portion, the idea of the proof of the last portion of Equation (5.20) is generalized to $\lceil(k-1)/2\rceil$ even-numbered machines in MFRC. Subsequently, the last portion of Equation (5.48) is $4\varepsilon + 2\delta + max_{1\leq i\leq\lceil(k-1)/2\rceil}\{max\{P_{2i-1}, \delta\} + P_{2i} + max\{P_{2i+1}, \delta\}\}$ by decreasing the feasible area to these $\lceil(k-1))/2\rceil$ even-numbered machines. This completes the proof.

Theorem 7. A decrease in cycle time results from using a rotationally arranged

MFRC with $\lceil(k-1)/2\rceil$ machines instead of a rotationally arranged SFRC with k machines if $P_i \leq \delta, \forall i \in \{1, 2, ..., k\}$.

Proof: For a rotational arranged SFRC with k machines, it should be recalled that the uphill permutation $_{v(s)}$ is the combination of $k+1$ activity with the following route: $A_0, A_1, A_2, ..., A_{k-1}, A_k$. For a particular $A_i, \forall i \in \{0, 1, 2, ..., k\}$, the time elapsed during the execution of this activity is the constant value $2\varepsilon + \delta$, which resulted in the overall execution time $2(k+1)\varepsilon + (k+1)\delta$ for $k+1$ activities. In addition, the route of activities is composed of $k+1$ pairs $(A_i, A_{i+1}), \forall i \in \{0, 1, 2, ..., k\}$, which means $k+1$ full waiting on k machines and I/O. The overall waiting time is $\sum_{i=1}^{k} P_i$ since the processing time of I/O is zero. SFR is at the starting position when it finishes A_k, resulting in no empty SFR movement. Therefore,

$$T_{\Pi_{V(s)}} = 2(k+1)\varepsilon + (k+1)\delta + \sum_{i=1}^{k} P_i \tag{5.49}$$

If all processing times are considerably shorter than for the cell under consideration, $P_i \leq \delta, \forall i \in \{1, 2, ..., k\}$, then $\Pi_{V(s)}$ has the least cycle time between all one-unit permutations of the rotational arranged SFRC with k machines in that SFRC is a subdivision of MFRC when $RP_i = 0, \forall i \in \{0, 1, 2, ..., k\}$. Also, it is known that the optimal permutation of the rotational arranged MFRC with $\lceil(k-1)/2\rceil$ machines is uphill permutation (See Theorem 2). It is enough to compare the cycle time of $\Pi_{V(s)}$ and $\Pi_{V(m)}$ for determining productivity gain of replacement $\lceil(k-1)/2\rceil$-machine MFRCs for k-machine SFRCs. Under Equation (5.21), when there are $\lceil(k-1)/2\rceil$ machines in the rotational arranged MFRC, the cycle time of the uphill permutation is:

$$T_{\Pi_{V(m)}} = 2\lceil\frac{k+1}{2}\rceil\varepsilon + \sum_{i=1}^{\lceil\frac{k+1}{2}\rceil} max\{P_{2i-1}, \delta\} + \sum_{i=1}^{\lceil\frac{k-1}{2}\rceil} P_{2i} \tag{5.50}$$

Recall that the processing time of all odd-numbered operations are considerably shorter than δ in the original cell, $P_i \leq \delta, \forall i \in \{1, 3, ..., 2\lceil(k+1)/2\rceil - 1\}$, and consequently $\sum_{i=1}^{\lceil(k+1)/2\rceil} max\{P_{2i-1}, \delta\} = \lceil(k+1)/2\rceil\delta$. This means that an alternative equation for Equation (5.50) is:

$$T_{\Pi_{V(m)}} = 2\lceil\frac{k+1}{2}\rceil\varepsilon + \lceil\frac{k+1}{2}\rceil\delta + \sum_{i=1}^{\lceil\frac{k-1}{2}\rceil} P_{2i} \tag{5.51}$$

Similar to Equation (5.49), Equation (5.51) is composed of three components: loading/unloading times, travel times and full waiting times. It is easy to prove that each one of these three components in Equation (5.49) are larger in contrast with them in Equation (5.51), which means Equation (5.49) is bigger than Equation (5.51). As a direct result of this proof, when $P_i \leq \delta, \forall i \in \{1, 2, ..., k\}$, the productivity benefits of the rotational arranged MFRC with $\lceil (k-1)/2 \rceil$ machines is substantial in comparison with the rotational arranged SFRC with k machines. This completes the proof.

Theorem 7 leads to the following question: How much would this swap increase productivity? A straight answer to this question raises the possibility of reaching full utilization and maximum ratio of production output from replaced manufacturing cell. This analysis is suitable especially when the goal is to determine an optimal permutation so that other management objectives such as the total manufacturing cost and the number of Work In Process (WIP), etc. are jointly taken into account.

Lemma 3. Assume that $P_i \leq \delta, \forall i \in \{1, 2, ..., k\}$. Then, the lower and upper bounds of productivity of employing a rotationally arranged MFRC with $\lceil (k-1)/2 \rceil$ machines in place of a rotationally arranged SFRC with k machines can be written as:

$$1 + \frac{2\lceil \frac{k}{2} \rceil \varepsilon + \lceil \frac{k}{2} \rceil \delta}{2\lceil \frac{k+1}{2} \rceil \varepsilon + (\lceil \frac{k+1}{2} \rceil + \lceil \frac{k-1}{2} \rceil)\delta} \leq PO_k \leq 1 + \frac{2\lceil \frac{k}{2} \rceil \varepsilon + \lceil \frac{k}{2} \rceil \delta}{2\lceil \frac{k+1}{2} \rceil \varepsilon + \lceil \frac{k+1}{2} \rceil \delta} \tag{5.52}$$

Proof: As a direct outcome of Theorem 7, PO_k is $T_{\Pi_{V(s)}}/T_{\Pi_{V(m)}}$ when $P_i \leq \delta$. Considering Equations (5.49) and (5.50), the operating efficiency of the replacement is:

$$\begin{aligned}
\frac{T_{\Pi_{V(s)}}}{T_{\Pi_{V(m)}}} &= \frac{2(k+1)\varepsilon + (k+1)\delta + \sum_{i=1}^{k} P_i}{2\lceil \frac{k+1}{2} \rceil \varepsilon + \sum_{i=1}^{\lceil \frac{k+1}{2} \rceil} max\{P_{2i-1}, \delta\} + \sum_{i=1}^{\lceil \frac{k-1}{2} \rceil} P_{2i}} \\
&= \frac{2(k+1)\varepsilon + (k+1)\delta + \sum_{i=1}^{k} P_i}{2\lceil \frac{k+1}{2} \rceil \varepsilon + \lceil \frac{k+1}{2} \rceil \delta + \sum_{i=1}^{\lceil \frac{k-1}{2} \rceil} P_{2i}}
\end{aligned} \tag{5.53}$$

For simplicity, Equation (5.53) is revised as:

$$\frac{T_{\Pi_{V(s)}}}{T_{\Pi_{V(m)}}} = \frac{2(\lceil \frac{k}{2} \rceil + \lceil \frac{k+1}{2} \rceil)\varepsilon + (\lceil \frac{k}{2} \rceil + \lceil \frac{k+1}{2} \rceil)\delta + (\sum_{i=1}^{\lceil \frac{k+1}{2} \rceil} P_{2i-1} + \sum_{i=1}^{\lceil \frac{k-1}{2} \rceil} P_{2i})}{2\lceil \frac{k+1}{2} \rceil \varepsilon + \lceil \frac{k+1}{2} \rceil \delta + \sum_{i=1}^{\lceil \frac{k-1}{2} \rceil} P_{2i}}$$

$$= 1 + \frac{2\lceil\frac{k}{2}\rceil\varepsilon + \lceil\frac{k}{2}\rceil\delta + \sum_{i=1}^{\lceil\frac{k+1}{2}\rceil} P_{2i-1}}{2\lceil\frac{k+1}{2}\rceil\varepsilon + \lceil\frac{k+1}{2}\rceil\delta + \sum_{i=1}^{\lceil\frac{k-1}{2}\rceil} P_{2i}} \tag{5.54}$$

Based on Equation (5.54), the least value of productivity, $min(T_{\Pi_{V(s)}}/T_{\Pi_{V(m)}})$, and the most value of productivity, $max(T_{\Pi_{V(s)}}/T_{\Pi_{V(m)}})$, are obtained to extract bounds on the amount of performance improvement of employing a regular MFRC in place of a SFRC when rotational arrangement is taken into consideration. Apparently, Equation (5.54) takes its least value if $P_{2i-1} = 0, \forall i \in \{1, 2, ..., \lceil(k+1)/2\rceil\}$ and $P_{2i} = \delta, \forall i \in \{1, 2, ..., \lceil(k-1)/2\rceil\}$:

$$min(\frac{T_{\Pi_{V(s)}}}{T_{\Pi_{V(m)}}}) = 1 + \frac{2\lceil\frac{k}{2}\rceil\varepsilon + \lceil\frac{k}{2}\rceil\delta}{2\lceil\frac{k+1}{2}\rceil\varepsilon + (\lceil\frac{k+1}{2}\rceil + \lceil\frac{k-1}{2}\rceil)\delta} \tag{5.55}$$

Once again, considering Equation (5.54), $max(T_{\Pi_{V(s)}}/T_{\Pi_{V(m)}})$ is obtained by $P_{2i-1} = \delta, \forall i \in \{1, 2, ..., \lceil k/2\rceil\}$ and $P_{2i} = 0, \forall i \in \{1, 2, ..., \lceil(k-1)/2\rceil\}$:

$$max(\frac{T_{\Pi_{V(s)}}}{T_{\Pi_{V(m)}}}) = 1 + \frac{2\lceil\frac{k}{2}\rceil\varepsilon + \lceil\frac{k}{2}\rceil\delta}{2\lceil\frac{k+1}{2}\rceil\varepsilon + \lceil\frac{k+1}{2}\rceil\delta} \tag{5.56}$$

This completes the proof.

Theorem 8. An increase in cycle time directly results from applying a rotationally arranged MFRC with $\lceil(k-1)/2\rceil$ machines instead of a rotationally arranged SFRC with k machines if $P_{l(s)} \geq 2(k-1)\varepsilon + (3k-1)\delta$.

Proof: This theorem is proved in two phases. In the first phase the optimal permutation for a SFRC with k machines is obtained when $P_{l(s)} \geq 2(k-1)\varepsilon + (3k-1)\delta$. Recall that, for a rotational arranged SFRC with k machines, the downhill permutation $\Pi_{D(s)}$ is the combination of $k+1$ pairs $(A_i, A_{i-1}), \forall i \in \{0, 1, 2, ..., k\}$ which result in the following route: $A_0, A_k, A_{k-1}, ..., A_2, A_1$. Note that $i - 1 = k$ when i=0. Because all of these $k+1$ pairs are similar, it is enough to analyze one of them and extend the result to all of them. Before performing an arbitrary activity $A_{i-1}, \forall i \in \{0, 1, 2, ..., k\}$, the empty SFR, which already finished A_i returns from M_{i+1} to $M_{i-1}(2\delta)$. Then, SFR performs $A_{i-1}(2\varepsilon + \delta)$ after a partial waiting on $M_{i-1}(w_{i-1})$. Therefore,

$$T_{\Pi_{D(s)}} = \begin{cases} 2(k+1)\varepsilon + 3(k+1)\delta & \text{if } P_{l(s)} \leq 2(k-1)\varepsilon + (3k-1)\delta \\ P_{l(s)} + 4\varepsilon + 4\delta & \text{if } P_{l(s)} \geq 2(k-1)\varepsilon + (3k-1)\delta \end{cases} \tag{5.57}$$

This means that the cycle time is $2(k+1)\varepsilon + 3(k+1)\delta$ when $\sum_{i=1}^{k} w_i = 0$. Nevertheless, when $\sum_{i=1}^{k} w_i \neq 0$, it equals $P_{l(s)} + 4\varepsilon + 4\delta$ in that the partial waiting certainly happens on M_l and covers all other component of cycle time. Also if $P_{l(s)} \geq 2(k-1)\varepsilon + (3k-1)\delta$, then $\Pi_{D(s)}$ has the least cycle time between all one-unit permutations, $P_{l(s)} + 4\varepsilon + 4\delta$, since SFRC is generally a subdivision of MFRC when $RP_i = 0, \forall i \in \{0, 1, ..., k\}$.

After finding optimal permutation for SFRC, in the second phase, it is proved that applying a rotationally arranged MFRC with $\lceil (k-1)/2 \rceil$ machines instead of aforementioned SFRC with k machines is not a good idea when $P_{l(s)} \geq 2(k-1)\varepsilon + (3k-1)\delta$. Obviously, for this SFRC, the downhill permutation $\Pi_{D(s)}$ dominates all possible permutations of MFRC if $T_{\Pi_{D(s)}} \leq T_{s\to m}$. For making it easier to prove this inequality, it should be emphasized that an arbitrary value c is less than double-sided function $f(x) = max(A, B)$, if and only if $C \leq A$ or $C \leq B$. In other words, it is not necessary to prove C is smaller than both arbitrary values A and B in order to show $C \leq max(A, B)$. Following this mathematical concept and considering any given $\varepsilon, \delta, P_1, P_2, ,$ and P_k, it can be conclude that $T_{\Pi_{D(s)}} \leq T_{s\to m}$ when $T_{\Pi_{D(s)}}$ is less than the left or right side of $T_{s\to m}$. On the other hand, the cycle time of the optimal permutation $\Pi_{D(s)}$ is smaller than $T_{s\to m}$ when $P_{l(s)} + 4\varepsilon + 4\delta \leq 2\lceil (k+1)/2 \rceil \varepsilon + \sum_{i=1}^{\lceil (k+1)/2 \rceil} max\{P_{2i-1}, \delta\} + \sum_{i=1}^{\lceil (k-1)/2 \rceil} min\{P_{2i}, \delta\}$ or $P_{l(s)} + 4\varepsilon + 4\delta \leq 4\varepsilon + 2\delta + max_{1 \leq i \leq \lceil (k-1)/2 \rceil}\{max\{P_{2i-1}, \delta\} + P_{2i} + max\{P_{2i+1}, \delta\}\}$. For simplicity, the left and right sides of $T_{s\to m}$ and $T_{\Pi_{D(s)}}$ are represented as below:

$$A = 2\lceil \frac{k+1}{2} \rceil \varepsilon + \sum_{i=1}^{\lceil \frac{k+1}{2} \rceil} max\{P_{2i-1}, \delta\} + \sum_{i=1}^{\lceil \frac{k-1}{2} \rceil} min\{P_{2i}, \delta\}$$
$$B = 4\varepsilon + 2\delta + \max_{1 \leq i \leq \lceil \frac{k-1}{2} \rceil} \{max\{P_{2i-1}, \delta\} + P_{2i} + max\{P_{2i+1}, \delta\}\}$$
$$C = P_{l(s)} + 4\varepsilon + 4\delta$$

Notice that $P_{l(s)}$ can be the processing time of odd-numbered or even-numbered machine in SFRC. This difference is especially important become important when SFRC is converted to MFRC. For the former case, MFR performs this operation instead of this machine, whereas this machine remains in MFRC for the latter case. Therefore, this phase of proof is divided into two sub-cases below:

1. M_l is odd-numbered machine in SFRC: Under this condition, $4\varepsilon \leq 2\lceil (k+1)/2 \rceil \varepsilon$ and $P_{l(s)} + 4\delta \leq \sum_{i=1}^{\lceil (k+1)/2 \rceil} max\{P_{2i-1}, \delta\}$ in that k is not a small

number. In fact, $\sum_{i=1}^{\lceil (k+1)/2 \rceil} max\{P_{2i-1}, \delta\}$ is composed of $\lceil (k+1)/2 \rceil max$ terms. We know that M_l is odd-numbered machine; hence, one of these max terms is $max\{P_{l(s)}, \delta\} = P_{l(s)}$, meaning that $P_{l(s)} + 4\delta \leq P_{l(s)} + \lceil (k-1)/2 \rceil \delta \leq \sum_{i=1}^{\lceil (k+1)/2 \rceil} max\{P_{2i-1}, \delta\}$. Therefore, $C \leq A$.

2. M_l is even-numbered machine in SFRC: Under this condition, $P_{l(s)} + 2\delta \leq max_{1 \leq i \leq \lceil (k-1)/2 \rceil}\{max\{P_{2i-1}, \delta\} + P_{2i} + max\{P_{2i+1}, \delta\}\}$ in that one of $\lceil (k-1)/2 \rceil$ right-sided component of this inequality is $max\{P_{l(s)-1}, \delta\} + P_{l(s)} + max\{P_{l(s)+1}, \delta\}$. Subsequently, we have $P_{l(s)} + 2\delta \leq max\{P_{l(s)-1}, \delta\} + P_{l(s)} + max\{P_{l(s)+1}, \delta\} \leq max_{1 \leq i \leq \lceil (k-1)/2 \rceil}\{max\{P_{2i-1}, \delta\} + P_{2i} + max\{P_{2i+1}, \delta\}\}$. This means that $C \leq B$. Due to the fact that always $C \leq A$ or $C \leq B$, using a MFRC with $\lceil (k-1)/2 \rceil$ machines instead of a SFRC with k machines is not good idea when $P_{l(s)} 2(k-1)\varepsilon + (3k-1)\delta$.

This completes the proof.

Lemma 4. Considering $P_{l(s)} \geq 2(k-1)\varepsilon + (3k-1)\delta$, the lower and upper bounds of productivity of employing a rotationally arranged MFRC with $\lceil (k-1)/2 \rceil$ machines in place of a classical one with k machines is:

$$\frac{P_{l(s)} + 4\varepsilon + 4\delta}{2\lceil \frac{k+1}{2} \rceil \varepsilon + \lceil \frac{k+1}{2} \rceil P_{l(s)} + \lceil \frac{k-1}{2} \rceil \delta} \leq PO_k \leq 1 \tag{5.58}$$

Proof: It is known that $\Pi_{D(s)}$ is optimal permutation if $P_{l(s)} \geq 2(k-1)\varepsilon + (3k-1)\delta$ in SFRC. Also, the optimal cycle time for MFRC can be equal to Equation (5.48) under the best condition. Therefore, considering Equations (5.48) and (5.57), the operating efficiency of the replacement $(T_{\Pi_{D(s)}} / \underline{T}_{s \to m})$ is:

$$\frac{P_{l(s)} + 4\varepsilon + 4\delta}{max\{2\lceil \frac{k+1}{2} \rceil \varepsilon + \sum_{i=1}^{\lceil \frac{k+1}{2} \rceil} max\{P_{2i-1}, \delta\} + \sum_{i=1}^{\lceil \frac{k-1}{2} \rceil} min\{P_{2i}, \delta\}, 4\varepsilon + 2\delta + max\{P_{2i-1}, \delta\} + P_{2i} + max\{P_{2i+1}, \delta\}\}} \tag{5.59}$$

Based on Equation (5.59), the least value of productivity and the most value of productivity can be obtained. Equation (5.59) takes its least value if $P_i = P_{l(s)}, \forall i \in \{1, 2, ..., k\}$:

$$\begin{aligned} min(\frac{T_{\Pi_{D(s)}}}{\underline{T}_{s \to m}}) &= \frac{P_{l(s)} + 4\varepsilon + 4\delta}{max\{2\lceil \frac{k+1}{2} \rceil \varepsilon + \sum_{i=1}^{\lceil \frac{k+1}{2} \rceil} P_{l(s)} + \sum_{i=1}^{\lceil \frac{k-1}{2} \rceil} \delta, 4\varepsilon + 2\delta + 3P_{l(s)}\}} \\ &= \frac{P_{l(s)} + 4\varepsilon + 4\delta}{2\lceil \frac{k+1}{2} \rceil \varepsilon + \lceil \frac{k+1}{2} \rceil P_{l(s)} + \lceil \frac{k-1}{2} \rceil \delta} \end{aligned} \tag{5.60}$$

Also, considering Equation (5.59), $max(T_{\Pi_{D(s)}}/\underline{T}_{s\to m})$ is obtained by $0 \le P_{2i-1} \le \delta$ and $P_{2i} = 0, \forall i \in \{1, 2, ..., k\}$, excluding $P_{l(s)}$. Note that $\underline{T}_{s\to m}$ is $P_{l(s)}$ dependent under this condition. In other words, $\underline{T}_{s\to m}$ is different for odd-numbered $P_{l(s)}$ and even-numbered $P_{l(s)}$ as follows:

$$T_{\Pi_{D(s)}} = \begin{cases} 2\lceil\frac{k+1}{2}\rceil\varepsilon + \lceil\frac{k-1}{2}\rceil\delta + P_{l(s)} & \text{if } l(s) \text{ is odd} \\ P_{l(s)} + 4\varepsilon + 4\delta & \text{if } l(s) \text{ is even} \end{cases} \tag{5.61}$$

Therefore, under the best condition, the denominator of $max(T_{\Pi_{D(s)}}/\underline{T}_{s\to m})$ is $min\{2\lceil(k+1)/2\rceil\varepsilon + \lceil(k-1)/2\rceil\delta + P_{l(s)}, P_{l(s)} + 4\varepsilon + 4\delta\} = P_{l(s)} + 4\varepsilon + 4\delta$ which means that $max(T_{\Pi_{D(s)}}/\underline{T}_{s\to m})$=1. This completes the proof.

Now, there is an appropriate framework for replacing the different kinds of SFRCs, small-scale and large-scale, with MFRCs. This framework helps the companies which are enthusiastic about using MFRs in fully automated manufacturing systems. In other words, using this framework before employing MFRs in the production line, they can find out whether this option can increase the productivity or not? This would assist manufacturers deciding on which type of robotic cell is better for any one of the part processing routes.

5.7 Concluding Remarks

In this paper, a framework for scheduling an advanced robotic cell namely rotationally arranged MFRC has been developed for the first time. Initially, a TSP-based model of this problem has been formulated to give computational benefits due to the existing solution methods of TSP. Then, the lower bound of cycle time and the cycle time of two practical permutations $\Pi_{V(m)}$ and $\Pi_{D(m)}$ have been established, and some results about optimality for these permutations for a MFRC have been presented. Due to the fact that small-scale SFRCs play an important role in productivity improvement and it is possible to break a complicated SFRC into small-scale SFRCs, a proper evaluation with respect to cycle time between small-scale SFRCs and MFRCs has been performed. This evaluation has made it clear how to effectively change one- two- or three-machine SFRCs into a MFRC. In addition to small-scale SFRCs, the evaluation of large-scale SFRCs have been performed and resulted that a MFRC improves production rate when permutation $\Pi_{V(s)}$ is optimal. However, the use of MFRCs instead of SFRCs is a wasted expense when permutation $\Pi_{D(m)}$ is optimal. Furthermore, the productivity of using MFRC against that of SFRC expressed by PO_k for permutations Π_V and Π_D in order to establish a practical framework for solving multi-objective scheduling problems where other objectives such as cost and idle time are also considered. The expected future research in this topic is research about scheduling in MFRCs with features like double-arm MFRs. Also, in place of identical parts, one can take into account the multiple parts. Finally, computational study of the problem understudy is also interesting. Some open problems related to the complexity of it are: 1) What is the complexity of the problem of finding an optimal one-unit cycle under linear layout with additive travel-times in MFRCs? 2) What is the complexity of the problem of finding an optimal one-unit cycle under circular layout with additive travel-times in MFRCs? 3) What is the complexity of the problem of finding an optimal one-unit cycle with constant travel-times in MFRCs?

Chapter 6 is based on the published article Foumani, M., Gunawan, I., Smith-Miles, K., 2015. Increasing Throughput for a Class of Two-Machine Robotic Cells Served by a Multi-Function Robot. IEEE Transactions on Automation Science and Engineering, PP(99), 1 - 10. DOI: 10.1109/TASE.2015.2504478

Abstract *The multi-function robotic cell scheduling problem has been recently studied in the literature. The main assumption in the pertaining literature is that a multi-function robot performs an operation on the part during any loaded move between two adjacent processing stages. For a two-machine cell, these stages are the input hopper, the first machine, the second machine and the output hopper. Consequently, the multi-function robot performs three operations with fixed processing times. In contrast, we assume a class of two-machine cells where none of the processing times of three operations are fixed. However, their summation is fixed and equivalent to the processing time of the unique operation. The processing mode of the unique operation performed by the multi-function robot is stop resume. Thus, regardless of the gap interrupts during operations by two machines, the robot continues performing the unique operation of the part when it is reloaded to the robot without any loss of time. The focus lies on n-unit cycles due to their popularity. It is proven one-unit cycles have better performance for the problem under study. The cycle time of one-unit cycles are obtained and optimality conditions are determined for different pickup criteria: free, interval, and no-wait.*

Note to Practitioners *Multifunction robotic cells are extensively used in the inspection of many automotive products including crankshaft, gears and lifters in transit. Here, some practice scheduling problems addressing multifunction robots with stop resume processing modes are identified, and then the optimality conditions are determined for these problems. Using the results in this paper, production managers are able to develop a methodology for scheduling of multifunction robotic cells bought by their companies.*

Classification $SRF^{1,1,1}_{2,2,2}|$*free, additive , deterministic , identical, cyclic*$|T$
$SRF^{1,1,1}_{2,2,2}|$*interval, additive , deterministic , identical, cyclic*$|T$
$SRF^{1,1,1}_{2,2,2}|$*no-wait, additive , deterministic , identical, cyclic*$|T$

Note References are considered at the end of the thesis.

Chapter 6

Increasing Throughput for a Class of Two-machine Robotic Cells Served by a Multi-Function Robot

6.1 Introduction

High volume production environments are always controlled by cyclic production. Due to an urgent need for increasing automation in such production environments, the manufacturer's preferred option is applying material handling robots of different sizes for almost every industrial process, like semiconductor manufacturing, automotive production and the aircraft industry (Sun and Wu, 2011; Basile et al., 2014). Some issues should be taken into account for this option (e.g., part processing route, production line design, and robot movement schedule) to increase efficiency of the production environment. Most commonly, there is no more than one robot in a production line in the standard setting to avoid collisions. However, this often causes the robot to be the bottleneck of the production line, especially if its operations are not scheduled efficiently. As a consequence, the main focus of this study is on optimizing the sequence of robot activities to increase efficiency of the automated production line.

A Single-Function Robotic Cell (referred to as SFRC in short) with two tandem production machines is composed of an input hopper I, a production machine M_1, a production machine M_2, an output hopper O, and a Single-Function Robot (SFR). The robot is called SFR since it only acts as a material handling device in charge of transferring parts between machines. In such a robotic cell, raw materials are fed to M_1, are processed serially from M_1 to M_2, and finally come out as completed parts from M_2. Thus, a SFRC is, in essence, a flow shop with a fixed number of servers (Geismar et al., 2008). SFRCs are also referred to as cluster tools and extensively used in semiconductor wafer fabrication (Geismar et al., 2011). Two dedicated machines which are defined as unit-capacity machines perform operations O_1 and O_2 with the associated processing time P_1 and P_2. Normally, I and O are called auxiliary machines (M_0 and M_3) and have zero processing times. Note that two-machine SFRCs are very flexible and practical due to the fact that they are easily able to divide a complex multi-process robotic cell into a number of two-machine robotic cells (Foumani and Jenab, 2013a).

Most studies in the literature analyzed SFRCs based on three types of pickup criteria: free pickup (the completed part can remain on the machine indefinitely), interval pickup (the maximum wait that the part can have on the machine is bounded within an interval), and no-wait pickup (the part waiting on the machine is not allowed) (Che et al., 2002). The comprehensive reviews of the literature related to the scheduling of SFRCs under different pickup criteria can be found in Dawande et al. (2005); Brauner (2008). Also, the extension of studies under these pickup criteria covers other problems (e.g., radar scheduling (Brauner et al., 2009), crane scheduling (Zhou and Li, 2012), task allocation of straddle carriers at container terminals (Cai et al., 2014) and nozzle selection and component allocation in surface mounting devices (Torabi et al., 2013)).

For any type of aforementioned pickup criterion, we briefly refer to some interesting SFRC scheduling problems addressed in the recent literature. Regarding the free pickup criterion, the method in Geismar et al. (2012) considered both design and scheduling problems for cells with dual-arm SFRs. A comprehensive approach for two-machine reentrant SFRCs consisting of a SFR with one-unit temporary hopper on the end-effector was presented in Foumani and Jenab (2012). Zarandi et al. (2013) studied the problem where part processing sequence and SFR movement schedule depend on setup times. For the interval pickup criterion, a mixed integer programming model of the SFRC scheduling problem based on n-unit cycles was presented by Zhou et al. (2012). Then, a branch and bound algorithm in Chui (2014) was proposed for scheduling of SFRCs with multiple part types in which the processing times of parts vary within their given time windows. Finally, for the no-wait pickup criterion, a SFRC scheduling problem with multiple robots was studied in Shabtay et al. (2014) to minimize the makespan and SFR selection cost.

Recently, specific robot grippers have been developed to improve the performance of SFRs. These grippers can perform an additional operation needed before loading a part to a production machine. An example of these robotic grippers is a material handling gripper which is equipped with a spot-welding gun. Such a gripper is added to the end-effector of the robot to perform this activity during transferring the part between two machines, avoiding the time wastage if performed at a particular machine. Also, the robotic gripper is able to inspect the part while carrying a part to the downstream machine. The robot with this state-of-the-art gripper integrates the inspection function into the parts transferring function. Thus, it is known as a Multi-Function Robot (MFR) which can be modelled as a portable tester machine with unit capacity. We represent the robotic arm of a MFR inspecting a crankshaft as in Figure 6.1. In this figure, the measuring heads are integrated into the automation by adding gages and crankshaft locating features to the robotic arm. This can decrease the extra loading/unloading motions and wait time required by an independent post-process measuring machine (in other words, this eliminates the free-standing measuring machine and the floor space which it occupies).

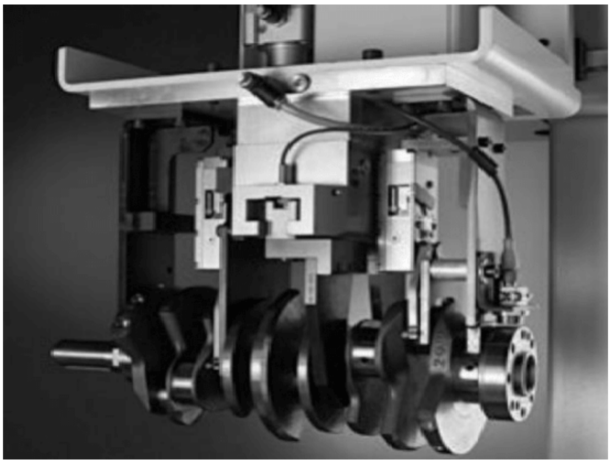


Fig. 6.1. Measurement of crankshaft diameters in transit

A substantial amount of research into the SFRC scheduling problem has been directed towards scheduling of the two-machine cells fitted with SFRs. To our best knowledge, the majority of research in SFRC scheduling is devoted to the two-machine cases because they are easy to control, and can be extended to complicated cells as identified by (Lim et al., 2006; Yi et al., 2008). Thus, we focus on the scheduling of Multi-Function Robotic Cells (referred to as MFRCs in short) with two machines due to their controllability and effectiveness. The formation of a MFRC with two tandem machines is quite similar to a SFRC. The only clear distinction is that MFRCs are always furnished with a MFR, rather than a SFR. The operational flexibility of a MFR means that it must be in charge of an exclusive and breakable operation Γ with the associated processing time γ during all inter-machine transfers of work in-process from I to O as shown in Figure 6.2. In more detail about any part j, MFR performs the first portion of Γ during the loaded move between I and M_1. Then, it performs the second portion of during the loaded move between M_1 and M_2, and finally it completes the rest of Γ during the loaded move from M_2 to O.

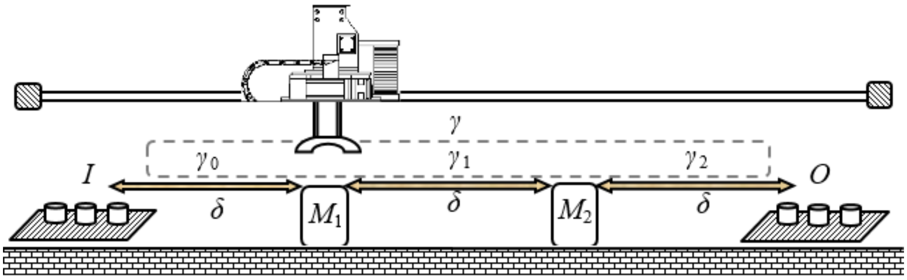


Fig. 6.2. A two-machine multi-function robotic cell

The processing mode of Γ is stop resume, which is especially prevalent among high-tech industries. This processing mode means that two gaps interrupt the processing Γ because it is obligatory for any particular part j to be processed on both M_1 and M_2. The MFR continues processing Γ on part j as soon as it is reloaded to MFR without any penalty for resumption or loss in time. Ideally, this resumable process can be divided into three subsets $\Gamma_{0j}, \Gamma_{1j}, \Gamma_{2j}$ with the associated processing times $\gamma_{0j}, \gamma_{1j}, \gamma_{2j}$ for any part j (j is an element of the set of natural numbers, $\forall j \in N$, because of a cyclic behavior). Operation $\Gamma_{ij}, \forall i \in \{0, 1, 2\}$, is processed during traverse of part j between M_i and M_{i+1}. Also, a constant value δ shows time taken by MFR to travel between these consecutive machines, which can be axillary or production machines. This means the time elapsed during transportation of part j between machines M_i and M_{i+1} is the varying value $max\{\gamma_{ij}, \delta\}$. We refer reader to Xie and Wang (2005); Allaoui et al. (2006) for a more detail description of the stop resume, or equivalently resumable processing.

Owing to each one of γ_{ij} may vary from zero to γ, it can be demonstrated that MFR considered here has process flexibility as the ability to handle a mixture of continuous operations. In fact, the related scheduling problem is composed of determining the best proportions of Γ to be allocated between I, M_1, M_2 and O as well as corresponding MFR movement sequence which gives us the optimal cycle time. In contrast, earlier studies of the MFRC scheduling problem neglected this assumption and considered Γ as an inflexible operation which was divided into three fixed subsets (or secondary operations). Let us consider an example to shed light on the process flexibility of the resumable process Γ.

Example. Similar to the current literature related to MFRCs, as the first alternative, assume that we have two inflexible operations to be completed by M_1 and M_2,

and three inflexible operations to be completed by the MFR. For all parts j, the associated processing times of these operations are predefined as P_1=10, P_2=12, γ_{0j}=7, γ_{1j}=6, γ_{2j}=1 and therefore γ=7+6+1=14. Also, assume loading/unloading time of the MFR and its empty travel time between two consecutive machines are ε=1 and δ=5, respectively. As a result, the time taken to complete a part under the closed loop $I \longrightarrow M_1 \longrightarrow M_2 \longrightarrow O \longrightarrow I$ is:
$6\varepsilon + 3\delta + P_1 + P_2 + max\{\gamma_{0j}, \delta\} + max\{\gamma_{1j}, \delta\} + max\{\gamma_{2j}, \delta\} = 61$
where $max\{\gamma_{0j}, \delta\} = max\{7, 5\}, max\{\gamma_{1j}, \delta\} = max\{6, 5\}$ and $max\{\gamma_{2j}, \delta\} = max\{1, 5\}$. For instance, $max\{7, 5\}$ is the time taken for the loaded move of MFR between I and M_1. Although MFR takes 5 time unit to move from I to M_1, it must wait for 2 time units in front of M_1 to finish Γ_{0j}. Now, as the second alternative, let us consider that Γ is a resumable process which can be broken into three flexible subsets with the associated processing times $\gamma_{0j}, \gamma_{1j}, \gamma_{2j}$ where $\gamma_{0j} + \gamma_{1j} + \gamma_{2j} = 14$. Subsequently, the allocation of γ_{0j}, γ_{1j}, and γ_{0j} should not necessarily be predefined. Obviously, in comparison with the inflexible case, the optimal allocation of γ_{0j}, γ_{1j}, and γ_{2j} is obtained if γ_{0j} and γ_{1j} are decreased by 2 and 1 time units, respectively, and also γ_{2j} is increased by 3 time units. So, considering the new allocation of γ_{0j}, γ_{1j}, and γ_{0j}, the time taken to complete a part under the closed loop for this case is:
$6\varepsilon + 3\delta + P_1 + P_2 + max\{\gamma_{0j}, \delta\} + max\{\gamma_{1j}, \delta\} + max\{\gamma_{2j}, \delta\} = 58$
where $max\{\gamma_{0j}, \delta\} = max\{5, 5\}, max\{\gamma_{1j}, \delta\} = max\{5, 5\}$ and $max\{\gamma_{2j}, \delta\} = max\{4, 5\}$.This means the time taken to complete the closed loop is decreased by 3 time units, or equivalently the throughput is increased by $(3/61) \times 100 = 4.92\%$. This shows that fixing the processing times on the MFR is a considerable hindrance restricting the better of the alternatives.

As mentioned before, all studies related to the MFRC scheduling problem follow the first alternative of the aforementioned example. Therefore, the first (and also the main) contribution of this study is that it assumes that the resumable process Γ can be broken into three varying proportions in order to increase the throughput of the MFRC. Regardless this assumption on the flexibility of the resumable process, there are some studies associated with the literature on MFRC scheduling problems under the first alternative. Foumani et al. (2013a) studied the problem for small-scale MFRCs under the free pickup criterion where there were no need to consider varying proportions of Γ. Also, Foumani et al. (2015a) provided a set of guideline notes on feasibility and optimality conditions of the problem with the no-wait criterion in which proportions of Γ were not varying. Subsequently, the scheduling problems studied by Foumani et al. (2013a, 2015a) were only simplified to find the optimal MFR movement sequence, not the best proportions of Γ. Foumani and Jenab (2013b) extended the results to m-machine MFRCs. They considered scheduling of the linearly configured MFRCs with m machines. The suggested method for the MFRC involved deriving the lower bound of cycle times, and listing the optimality condition of two specific cycles based on this lower bound. In an analogous work, Foumani et al. (2014) focused on the scheduling of rotationally configured MFRCs with m machines, and determined the parameter values for which two specific cycles are optimal. It should be noted that the number of feasible cycles for a MFRC with m machines is $m!$, while researches by Foumani and Jenab (2013b); Foumani et al. (2014) are not only limited to the resumable process with fixed subsets but also are restricted to two cycles. In other words, the impact for the remaining m-2 cycles is analyzed in none of Foumani and Jenab (2013b); Foumani et al. (2014), and these studies only give some notes regarding large-scale MFRCs. Thus, it is essential to provide a comprehensive analysis covering all cycles, especially for MFRCs with two tandem machines. This results in the second contribution of this study which is to provide a detail analysis of all feasible cycles.

MFRC with varying proportions of Γ was only presented by Foumani et al. (2013b). The study was limited to finding the optimal one-unit cycle of MFRC under the free pickup criterion. In other words, it does not enable full evaluation of the performance for other categories of MFRs with varying proportions of Γ. Hence, as the third contribution of this paper, it is necessary to not only give a framework that analyzes n-unit cycle's performance for the free pickup criterion, but also extend the results to two other main pickup criteria; interval and no-wait.

The bulk of this work deals with developing a stepwise procedure to calculate the cycle time, analyze it, and schedule MFR movements. The essential definitions are briefly provided in Section 6.2 of this article. In Section 6.3, the cycle time of two attained one-unit cycles are calculated precisely for cells in which a MFR interacts with two machines. As a byproduct, it is proved the optimal cycle is one-unit, and the optimality region of two one-unit cycles is determined in conformity with the free pickup criterion. Section 6.4 is devoted to interval and no-wait pickup criteria. The paper is concluded and implications provided in Section 6.5.

6.2 Cyclic Production

For complex systems executing in a cyclic behaviour, the balanced sequence of MFR movements is referred to as a cyclic schedule and its length of time is typically known as the cycle time. In each cycle namely n-unit cycle, n parts enter the two-machine MFRC and n parts leave from MFRC after following a series of five configured operations $\Gamma_{0j}, O_1, \Gamma_{1j}, O_2, \Gamma_{2j}$ for the part $j, \forall j \in \{0, 1, 2, ..., n\}$. The simplest case of n-unit cycles is one-unit cycle where exactly one part such as $j+1^{\text{th}}$ part enters MFRC and j^{th} part leaves from MFRC after the execution of the cycle (Yan et al., 2012). The Π^{th} MFRs one-unit cycle and its cycle time are characterized by S_Π and T_{S_Π}, respectively. Note that the throughput rate of the cell is also the multiplicative inverse of the cycle time. It equals $1/T_{S_\Pi}$, and shows the long-term average number of parts dropped at O per unit time. However, the use of cycle time is more common in practice since it is easier to analyze. In this stage, it is essential to modify the definitions described in Crama and de Klundert (1999) to gain the full benefit from cyclic solutions. Recalling δ and ε from the example illustrated in the previous section, the smallest subdivision of MFR movement, Activity, is defined as follows:

Definition 1. Having a cell in which a MFR interacts with two production machines, an activity $A_{ij}, \forall i \in \{0, 1, 2\}$ and $\forall j \in N$, is:

1. Empty MFR is instructed to unload part j from busy M_i.
2. MFR carries part j to M_{i+1}.
3. Busy MFR is finally instructed to load part j onto empty M_{i+1}.

This definition indirectly tells us the total length of time elapsed during the execution of A_{ij} is $2\varepsilon + max\{\delta, \gamma_{ij}\}$. For the sake of simplicity, $max\{\delta, \gamma_{ij}\}$ is labelled by β_{ij} throughout the remainder of the paper. Also, as stated by Che et al. (2014), the following constraints are necessary in order to avoid the collision.

Definition 2. A permutation of MFR activities is named a deadlock-free n-unit cycle if and only if:

1. It is not needed to unload a part from an idle M_1 or M_2 (between occurrence of $A_{i(j-1)}$ and A_{ij} must exactly have one occurrence of $A_{(i-1)j}, \forall i \in \{1, 2\}$ and $\forall j \in \{0, 1, 2, ..., n\}$).
2. It is not needed to load a part into a busy M_1 or M_2 (between occurrence of $A_{(i-1)(j-1)}$ and $A_{(i-1)j}$ must exactly have one occurrence of $A_{i(j-1)}, \forall i \in \{1, 2\}$ and $\forall j \in \{0, 1, 2, ..., n\}$).

In order to implement this definition in an n-unit cycle, the number of activities related to unloading each particular machine is exactly n (in other words, $\sum_{j=1}^{n} numA_{ij} = n, \forall i \in \{0, 1, 2\}$ when $numA_{ij}$ dictates the number of A_{ij} occurrences).

Regarding the one-unit cycle, it is supposed that the empty and busy machines associated with any particular cycle be clarified at the starting point of the cycle.

It therefore suffices to set activities into two different groups. When $A_{(i-1)j}$ is implemented before (after) A_{ij}, it tells us that M_i is empty (busy) at the initial point of the one-unit cycle (only, it should be noticed that $A_{(i-1)j} = A_{2j}$ if $i = 0$). We activate all cycles with the activity A_{0j} to achieve a homogeneous procedure. The definition below is extracted from Yan et al. (2010).

Definition 3. It is supposed that a MFR has full waiting in front of M_i for an arbitrary part j if MFR entirely waits at M_i to complete the processing after loading the part j onto it. Also if the empty MFR without delay moves to another occupied machine after loading part j onto M_i, the type of its waiting is referred to as partial waiting.

Without consideration the type of the waiting at M_i, we show its length of time by w_{ij} for j^{th} part fed to the manufacturing cell.

6.3 Optimal Cycle of Two-Machine MFRCs with Free Pickup Criterion

We initially study one-unit cycles in this section, and then end up with similar results for n-unit cases. It is apparent that two one-unit sequences of robot moves $S_1 = A_{0j}, A_{1j}, A_{2j}$ and $S_2 = A_{0j}, A_{2(j-1)}, A_{1j}$ can be executed for SFRC because of the fact that $i \in \{0, 1, 2\}$ and it has $(3-1)!$ permutations of three activities which start with A_{0j} (Foumani and Jenab, 2013a). Note this number is equal to the number of ways to arrange three distinct objects around a circle by fixing one of the objects. Therefore, these two cycles can be rewritten as $S_1 = A_{1j}, A_{2j}, A_{0(j+1)}$ and $S_2 = A_{1j}, A_{0(j+1)}, A_{2j}$ when their permutations start with A_{1j}.

Similarly, S_1 and S_2 are extracted from running a cell in which a MFR interacts with two tandem machines. The intuition behind it is that we can distinguish two cases SFRC and MFRC base on their characteristics. A SFRC has a strong orientation toward layout, whereas operational orientation of MFRC's is more remarkable. For this obvious reason, swapping a SFR with a MFR is only operation-directed, and has no influence on given part processing route priority and the cell layout. So, considering a two-machine MFRC, the number of cycles and MFR's path for them are completely similar to them for a SFRC. Nevertheless, the cycle time of both cyclic solutions S_1 and S_2 are potentially possible to be changed since MFR is operation-directed and it must perform a breakable process Γ (with not restart processing mode) when passes through I, intervening machines, and O.

This segment determines each of the cycle times in conformity with a stepwise procedure for MFRCs. obviously, it is known $\delta \leq \beta_{ij} \leq \gamma$ despite the fact that

$0 \le \gamma_{ij} \le \gamma$. Let us assume that β is the sum of β_{0j}, β_{1j} and β_{2j}. The clear meaning behind this statement is that β is the total length of time elapsed for performing operation Γ in transit of part j from I to O. Thus, β is never less than γ which is the time of performing operation Γ (excluding travel times). Furthermore, γ and β are data fed into the computer and information sent out from the computer for the determination of the cycle time. At the beginning, we graphically clarify how β is affected by γ. The details based on two breakpoints δ and 3δ in Figure 6.3 are given below:

1. $\gamma \le 3\delta$: The output of this case is one of the following two subdivisions:
    1.1. $\gamma \le \delta$: It follows immediately from $\gamma_{0j} \le \delta, \gamma_{1j} \le \delta$ and $\gamma_{2j} \le \delta$ that the best of the consuming time by MFR is $\beta_{0j} \le \delta, \beta_{1j} \le \delta$ and $\beta_{2j} \le \delta$. Accordingly, $\beta = 3\delta$ under both best and worst conditions named the lower bound of $\beta(\underline{\beta})$ and the upper bound of $\beta(\overline{\beta})$, respectively.
    1.2. $\delta \le \gamma \le 3\delta$: The best of sub processes of part j are $\gamma_{0j} \le \delta, \gamma_{1j} \le \delta$ and $\gamma_{1j} \le \delta$ which mean $\underline{\beta} = 3\delta$. Moreover, $\overline{\beta} = \gamma + 2\delta$ is the worst of β when an arbitrary γ_{ij} equals γ and both other γ_{ij} equal zero.
2. $3\delta \le \gamma$: β cannot be better than $\underline{\beta} = \gamma$ that occurs if everyone of γ_{ij} is bigger than δ. Once again, $\overline{\beta} = \gamma + 2\delta$ is the worst of β if an arbitrary γ_{ij} equals γ and both other γ_{ij} equal zero.

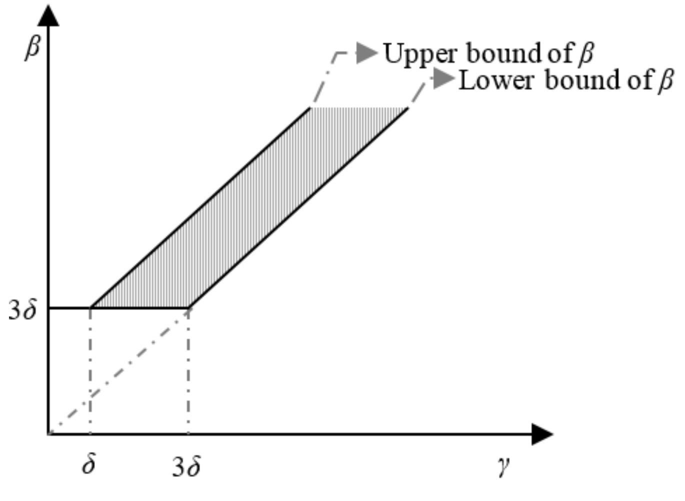


Fig. 6.3. Lower and upper bounds for β

The following questions arise in this stage of the work: why $\delta$and 3δ are two breakpoints of β? In answer to this question, it should be noted that $\underline{\beta}$ is a constant value when $\gamma \le \delta$, whereas it is an increasing function with respect to γ

when $\gamma > \delta$. In an analogous manner, $\overline{\beta}$ is a constant value when $\gamma \leq 3\delta$, while it is an increasing function when $\gamma > 3\delta$. In other words, the gap between the lower and upper bounds of β only increases when $\delta < \gamma \leq 3\delta$. Recalling Figure 6.3, we should proceed to determine the cycle time of cyclic solutions S_1 and S_2 in lemmas below.

Lemma 1. Having the free pickup criterion, the time required by MFR, which is also responsible for a unique operation Γ, to ready itself to perform exactly once S_1 is:

$$6\varepsilon+3\delta+max\{3\delta,\gamma\}+P_1+P_2 \tag{6.1}$$

Proof: A_{0j}, A_{1j}, A_{2j} tells us that MFR consumes times $2\varepsilon+\beta_{0j}, 2\varepsilon+\beta_{1j}$, and $2\varepsilon+\beta_{2j}$ in the execution of A_{0j}, A_{1j}, and A_{2j}. $\beta_{0j}+\beta_{1j}+\beta_{2j}$ depend directly on γ. It follows from the best condition in Figure 6.3 that $\beta_{0j}+\beta_{1j}+\beta_{2j}=\underline{\beta}=3\delta$ if $\gamma \leq 3\delta$. Making an additional point to that, $\beta_{0j}+\beta_{1j}+\beta_{2j}=\underline{\beta}=\gamma$ when $\gamma \geq 3\delta$. We also know that MFR returns to I for starting the next repetition of S_1 immediately after completing A_{2j}, and MFR consumes 3δ for this action. Finally, the increasing nature of the activities' permutation results in two full waits $w_{1l}=P_1$ and $w_{2l}=P_2$ at M_1 and M_2, respectively, and at most one part is processing in the cell from star to end state (S_1 is independent of part allocation type). Accordingly, Equation(6.1) is proved.

Lemma 2. T_{S_2} for a two-machine MFRC dealing with a unique operation Γ on MFR is expressed as below:

$$T_{S_2}=4\varepsilon+4\delta+max\{2\varepsilon+4\delta, 2\varepsilon+\delta+\gamma, P_1, P_2, \frac{P_1+P_2+\gamma-3\delta}{2}\} \tag{6.2}$$

Proof: The following are steps of calculating T_{S_2}: The cycle is initialized with a state in which M_2 is loaded, M_1 is empty, and the clock is set to zero. Note that T_{S_2} depends solely on part allocation type. The intuition behind it is that a part is loaded on M_2 when MFR picks up a part from I. Following that, the cycle time for an arbitrary part l is determined, and then this structure is extended for achieving T_{S_2}. For part l, the starting state of $A_{0l}, A_{2(l-1)}$, A_{1l} tells us MFR is removing the part l from I, machine M_1 is empty, and M_2 is processing the part $l-1$.

MFR consumes times $2\varepsilon+\beta_{0l}, 2\varepsilon+\beta_{2(l-1)}$, and $2\varepsilon+\beta_{1l}$ in the implementation of $A_{0l}, A_{2(l-1)}$, and A_{1l}, respectively. Also, we add empty MFR movements and waiting times to them to calculate T_{S_2}. It is known that MFR moves with holding no part only before executing A_{0l} (from M_2 to I with elapsed time of 2δ), $A_{2(l-1)}$ (from

M_1 to M_2 with elapsed time of δ), and A_{1l} (from O to M_1 with elapsed time of 2δ). Till now, the sum of elapsed times is $6\varepsilon+5\delta+\beta_{0l}+\beta_{2(l-1)}+\beta_{1l}$. Assuming N as an unlimited number of finished parts, the long-run value of this sum equals $6N\varepsilon+5N\delta+\sum_{j=1}^{N}(\beta_{0j}+\beta_{2(j-1)}+\beta_{1j})$. The sum is for cyclic behavior of N finished parts; Thus, the per-unit time is $\lim_{N\to\infty}\dfrac{6N\varepsilon+5N\delta+\sum_{j=1}^{N}(\beta_{0j}+\beta_{2(j-1)}+\beta_{1j})}{N}=6\varepsilon+5\delta+\beta$. We know that MFR is instructed to have two partial waiting depending exclusively on the assigned part. The first one is MFR waiting time at the top of M_2 for the part $l-1$, and the second one is MFR waiting time at M_1 for part l. So, $w_{1l}, w_{2(l-1)}$ are:

$$w_{1l}=max\{0,P_1-(2\varepsilon+3\delta+\beta_{2(l-1)}+w_{2(l-1)})\}$$
$$w_{2(l-1)}=max\{0,P_2-(2\varepsilon+3\delta+\beta_{0l})\}$$
$$w_{1l}+w_{2(l-1)}=max\{0,P_1-(2\varepsilon+3\delta+\beta_{2(l-1)}),P_2-(2\varepsilon+3\delta+\beta_{0l})\}$$

It follows from the sum of waits that $\sum_{j=1}^{N}\sum_{i=1}^{2}w_{ij}=\sum_{j=1}^{N}max\{0,P_1-(2\varepsilon+3\delta+\beta_{2(j-1)}),P_2-(2\varepsilon+3\delta+\beta_{0j})\}$ for N iteration of S_2. This value is the waiting time required for processing N parts; therefore, the per-unit waiting time is $\dfrac{\sum_{j=1}^{N}\sum_{i=1}^{2}w_{ij}}{N}$. Apparently, adding $6\varepsilon+5\delta+\beta$ to this value leads to:

$$T_{S_2}=6\varepsilon+5\delta+\beta+\frac{1}{N}\sum_{j=1}^{N}\sum_{i=1}^{2}w_{ij} \tag{6.3}$$

Equation (6.3) is broken onto the following stepwise analysis:

1. $P_1\le 2\varepsilon+4\delta, P_2\le 2\varepsilon+4\delta$: The output of this subcase is $\frac{1}{N}\sum_{j=1}^{N}\sum_{i=1}^{2}w_{ij}=0$ and consequently $T_{S_2}=6\varepsilon+5\delta+max\{3\delta,\gamma\}$ by adapting the proof of Lemma 1 and Figure 6.3.

2. $P_1>2\varepsilon+4\delta, P_2\le 2\varepsilon+4\delta$: It is supposed that the impact of γ on T_{S_2} be taken into account; thus, the analysis for this subcase brunched into three sub-cases below:

  2.1. $\gamma\le 3\delta$: the best proportions of secondary MFR processes are $\gamma_{0j}\le\delta,\gamma_{1j}\le\delta,\gamma_{2j}\le\delta$. So, $\beta=\underline{\beta}=3\delta$, and we can conclude $T_{S_2}=6\varepsilon+8\delta+\frac{1}{N}\sum_{j=1}^{N}max\{0,P_1-(2\varepsilon+4\delta)\}=4\varepsilon+4\delta+P_1$.

  2.2. $3\delta\le\gamma\le P_1-2\varepsilon-\delta$: It can be stated that the best shares of secondary MFR processes are $\gamma_{0j}=\delta,\gamma_{1j}=\delta,\gamma_{2j}=\gamma-2\delta$ to meet $\underline{\beta}$. Thus,

$\beta = \underline{\beta} = \gamma$ and $T_{S_2} = 6\varepsilon + 5\delta + \gamma + \frac{1}{N}\sum_{j=1}^{N} max\{0, P_1 - (2\varepsilon + 3\delta + \gamma - 2\delta)\} = 4\varepsilon + 4\delta + P_1$.

2.3. $P_1 - 2\varepsilon - \delta \leq \gamma$: the best of β is accomplished by $\gamma_{0j} = \delta, \gamma_{1j} = \delta, \gamma_{2j} = \gamma - 2\delta$. Furthermore, it is evident $T_{S_2} = 6\varepsilon + 5\delta + \gamma + \frac{1}{N}\sum_{j=1}^{N} max\{0, P_1 - (2\varepsilon + 3\delta + \gamma - 2\delta)\} = 6\varepsilon + 5\delta + \gamma$ because $P_1 \leq 2\varepsilon + \delta + \gamma$.

3. $P_1 \leq 2\varepsilon + 4\delta, P_2 \geq 2\varepsilon + 4\delta$: the result of case 2 and 3 are similar if we interchange P_1 and P_2.

4. $P_1 > 2\varepsilon + 4\delta, P_2 > 2\varepsilon + 4\delta$: This is the most complex step of the analysis where P_1 and P_2 are both bigger than $2\varepsilon + 4\delta$.

4.1. $\gamma \leq 3\delta$: apparently, $T_{S_2} = 6\varepsilon + 8\delta + \frac{1}{N}\sum_{j=1}^{N} max\{0, P_1 - (2\varepsilon + 4\delta), P_2 - (2\varepsilon + 4\delta)\} = 4\varepsilon + 4\delta + max\{P_1, P_2\}$ for this subcase.

4.2. $\gamma > 3\delta$ and $P_1 - P_2 > \gamma - 3\delta$: A new technique is devised to carry out this sub case. In this technique, the analysis is based on $P_1 - P_2$ instead of P_1 and P_2 separately in order to overcome limitations. $\gamma_{0j} = \delta, \gamma_{1j} = \delta, \gamma_{2j} = \gamma - 2\delta$ are extracted for secondary operations to meet $\beta = \underline{\beta} = \gamma$ and then $T_{S_2} = 6\varepsilon + 5\delta + \gamma + \frac{1}{N}\sum_{j=1}^{N} max\{0, P_1 - (2\varepsilon + 3\delta + \gamma - 2\delta), P_2 - (2\varepsilon + 4\delta)\} = 4\varepsilon + 4\delta + P_1$.

4.3. $\gamma > 3\delta$ and $P_1 - P_2 < -(\gamma - 3\delta)$: the solution of this situation is same as 4.2. Actually, it suffices to interchange P_1 and P_2 to have $T_{S_2} = 4\varepsilon + 4\delta + P_2$.

4.4. $\gamma > 3\delta$ and $-(\gamma - 3\delta) \leq P_1 - P_2 \leq \gamma - 3\delta$: First of all, we must mention that γ_{1j} at most equals δ. The reason behind it is that γ_{1j} does not include in neither $P_1 - (2\varepsilon + 3\delta + \beta_{2(j-1)})$ nor $P_2 - (2\varepsilon + 3\delta + \beta_{0j})$. Notice these values show both sides of waiting time of part j in Equation (6.3). Nevertheless, γ_{0j} and γ_{2j} are bigger than δ. It is essential to keep a balance between γ_{0j} and γ_{2j} based on $P_1 - P_2$. Thus,

$\gamma_{0j} = \delta + \dfrac{\gamma - 3\delta}{2} - \dfrac{P_1 - P_2}{2}, \gamma_{2j} = \delta + \dfrac{\gamma - 3\delta}{2} + \dfrac{P_1 - P_2}{2}$, and also $T_{S_2} = 6\varepsilon + 5\delta + \gamma + \dfrac{1}{N}\sum_{j=1}^{N} max\{0, P_1 - (2\varepsilon + 4\delta + \dfrac{\gamma - 3\delta + P_1 - P_2}{2}), P_2 - (2\varepsilon + 4\delta + \dfrac{\gamma - 3\delta - P_1 + P_2}{2})\} = 4\varepsilon + \dfrac{5\delta + \gamma + P_1 + P_2}{2}$.

It follows immediately that this stepwise analysis can be summed up in Equation (6.4) where $2\varepsilon + 4\delta = \omega$ for simplicity:

$$T_{S_2} = \begin{cases} 6\varepsilon + 5\delta + max\{3\delta, \gamma\} & \text{if } P_1 \leq \omega, P_2 \leq \omega \\ 4\varepsilon + \delta + max\{\omega + \gamma, P_1\} & \text{if } P_1 > \omega, P_2 \leq \omega \\ 4\varepsilon + \delta + max\{\omega + \gamma, P_2\} & \text{if } P_1 \leq \omega, P_2 > \omega \\ 4\varepsilon + 4\delta + max\{P_1, P_2\} & \text{if } P_1, P_2 > \omega, |P_1 - P_2| > \gamma - 3\delta \\ 4\varepsilon + \dfrac{5\delta + \gamma + P_1 + P_2}{2} & \text{if } P_1, P_2 > \omega, |P_1 - P_2| \leq \gamma - 3\delta \end{cases} \tag{6.4}$$

The united formulation of Equation (6.4) represents Equation (6.2).

Let us now deviate from the cycle time calculations towards the impact of input data on the regions of optimality. It also follows that the region of optimality for S_1 and S_2 should be determined independently because either one of them have a chance to dominate the other one according to the input data. To explain this more apparently, let us give an example of a MFRC with input data: $P_1 = 5$, $P_2 = 10$, $\varepsilon = 1$, $\delta = 4$, *and* $\gamma = 15$. The Gantt chart in Figure 6.4a shows the sequence of activities when γ is equally shared between γ_{0j}, γ_{1j} and γ_{2j} (thus, $T_{S_1} > T_{S_2}$).

Further, the Gantt chart in Figure 6.4b conveniently characterizes the sequence of activities when P_1 and P_2 change into 2 and 4 (thus, $T_{S_1} < T_{S_2}$). This example yields the following question: which one of S_1 and S_2 has better performance for any given parameters? A correct answer to this question raises the possibility of gaining full productivity.

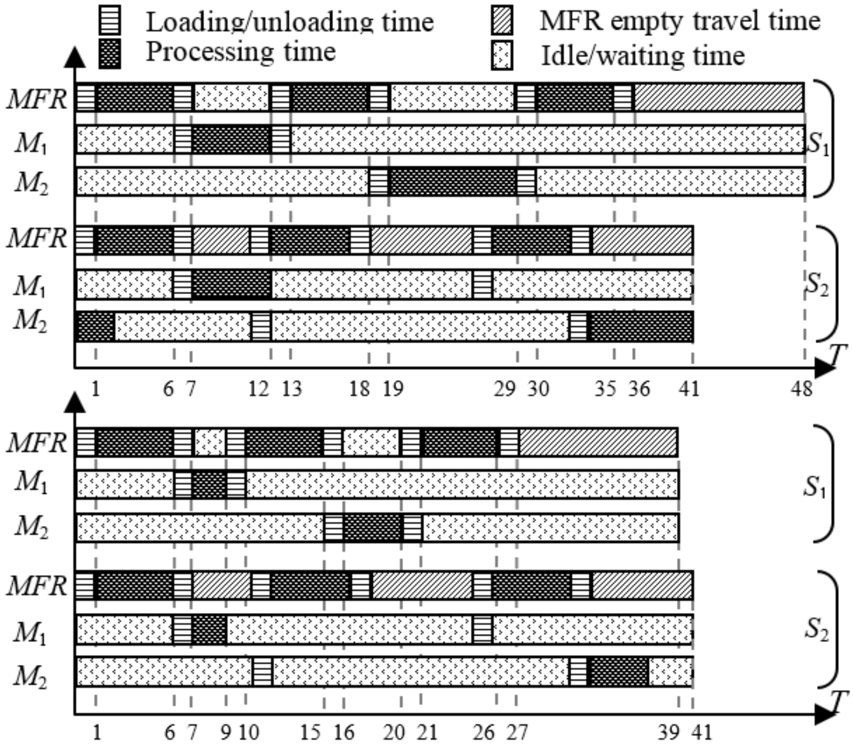


Fig. 6.4a (Top). Gantt chart of T_{S_1} and T_{S_2} when $\varepsilon = 1, \delta = 4, P_1 = 5, P_2 = 10, \gamma = 15$.
Figure 6.4b (Down). Gantt chart of T_{S_1} and T_{S_2} when
$\varepsilon = 1, \delta = 4, P_1 = 2, P_2 = 4, \gamma = 15$

Theorem 1. A MFRC with two dedicated machines as well as free pickup criterion reaches full utilization by implementation of S_1 only if $P_1 + P_2 \leq 2\delta$. Further, S_2 is the best between all one-unit cycles of MFRC under the assumption $P_1 + P_2 > 2\delta$.
Proof: We can rewrite Equation (6.1) as:

$$T_{S_1} = 4\varepsilon + 4\delta + max\{2\varepsilon + 2\delta + P_1 + P_2, 2\varepsilon - \delta + \gamma + P_1 + P_2\} \tag{6.5}$$

It is supposed that $max_{\Pi,j}$ characterizes the j^{th} max term in the Π^{th} cycle where the max terms are arranged as in the formulae for T_{S_Π} given by Equations (6.2) and (6.5). As instance of this kind formulating, $max_{2,5}$ is $(P_1 + P_2 + \gamma - 3\delta)/2$. Furthermore, the maximum max term between all max terms of the cyclic solution S_Π is characterized by max^*_Π. The cycle S_1 have no chance of being optimal when max^*_2 equals any one of $max_{2,3}, max_{2,4}$, or $max_{2,5}$ by reason of $max_{1,1} > max_{2,3}, max_{1,1} > max_{2,4}$ and $max_{1,2} > max_{2,5}$. Nevertheless, S_1 may be optimal when max^*_2 equals any one of $max_{2,1}$ or $max_{2,2}$.
When $max_{2,1}$ is the maximum max term of S_2, it essentially guarantees the correctness of $max^*_2 = max_{2,1}$. This condition is $\gamma \leq 3\delta, P_1 \leq 2\varepsilon + 4\delta$, and $P_2 \leq 2\varepsilon + 4\delta$.

It is also known $max^*_1 = max_{1,1}$ under this condition, and $max^*_1 \leq max^*_2$ if and only if $P_1 + P_2 \leq 2\delta$. It is supposed that $max^*_2 = max_{2,2}$ occurs only if $3\delta < \gamma, P_1 < 2\varepsilon + \delta + \gamma, P_2 < 2\varepsilon + \delta + \gamma$, as well as $P_1 + P_2 < 4\varepsilon + 5\delta + \gamma$. Finally, it is apparent that $max^*_1 = max_{1,2}$ under this condition, and $max^*_1 \leq max^*_2$ when $P_1 + P_2 \leq 2\delta$. Accordingly, the common region is the intersection of all conditions $(P_1 + P_2 \leq 2\delta)$.

Corollary 1. It is instructive to mention that M_1 and M_2 act exactly like two intermediate hoppers if there is not any operations for both machines ($P_1 = 0$ and $P_2 = 0$). In other words, a MFRC with two tandem production machines is converted into a MFRC with two intervening hoppers and no production machines. The optimality of the cycle S_1 in all regions results from Theorem 1 for this case. It therefore follows that MFR must perform Γ without interruption during transit of the part from I to O.

Corollary 2. When the layout of the cell is in a way that the distance between dedicated machines and hoppers is negligible ($\delta \approx 0$), MFR acts exactly like a single setup server which setups at most one machine at a given moment. These setup operations are performed exactly before starting an operation on the production machine and without delay. Furthermore, all of them are separable from the operation of machines, and are processed by a MFR. The optimality of S_2 in all regions results from Theorem 1 for this case. Note these kind robotized shops are common in industries such as CD production lines (Lim et al., 2006).

Theorem 1 tells us that it suffices to take into consideration the sum of processing time on machines to optimize the productivity of a MFRC with two tandem machines. The intuitive description behind this is also remarkable: MFR must fully wait in front of both machines and do not leave them before completing their processes when both processing times can be disregarded in comparison with δ (this is correct for S_1). Nevertheless, MFR is kept in standby in front of every one of machines if their processes take a long time and S_1 is selected for production planning. This means it is better to shift from S_1 to S_2 to have a better performance. Another secondary result of this theorem is that MFR's operation has no influence on the optimal cycle selection. Actually, Theorem 1 indirectly states the optimal one-unit cyclic solution of a MFRC scheduling problem depends solely on P_1, P_2, δ, and proves the outcomes of SFRCs are valid for MFRCs. Now, we analyze n-unit MFR move cycles for a cell with resumable processing regime and free pickup criterion.

Theorem 2. The cycle time of no n-unit MFR move cycle is less than the cycle time of both S_1 and S_2.

Proof: We initially show that any n-unit cycle is a convex combination of S_1 and S_2 in order to be deadlock-free. It is known that an n-unit MFR move cycle is the steady state cycle in which exactly n parts are completed and each activity exactly n times is implemented. Giving an example, M_2 should be loaded n times considering activities $A_{1j}, \forall j \in \{1, 2, ..., n\}$. Considering feasibility constraints, Definition 2, $A_{1j}, \forall j \in \{1, 2, ..., n\}$ are only activities that a choice between having a full waiting on M_2 or moving to I for a partial waiting exists after their implementations. The former case is followed by the activities A_{2j} and $A_{0(j+1)}$ and results in S_1, whereas the latter one is followed by $A_{0(j+1)}$ and A_{2j} and results in S_2 as shown in Figure 6.5.
Let us assume T_{S_n} is the total cycle time of an arbitrary n-unit MFR cycle to complete n parts. As seen in Figure 6.5, this cycle is a combination of k repetitions of S_1 and $n-k$ repetitions of S_2, and its cycle time is $T_{S_n} = k(T_{S_1}) + (n-k)(T_{S_2})$. As the average (per unit) cycle time to complete one part with this MFR cycle is T_{S_n}/n, the per-unit cycle time of S_n cannot be less than $min\{T_{S_1}, T_{S_2}\}$.

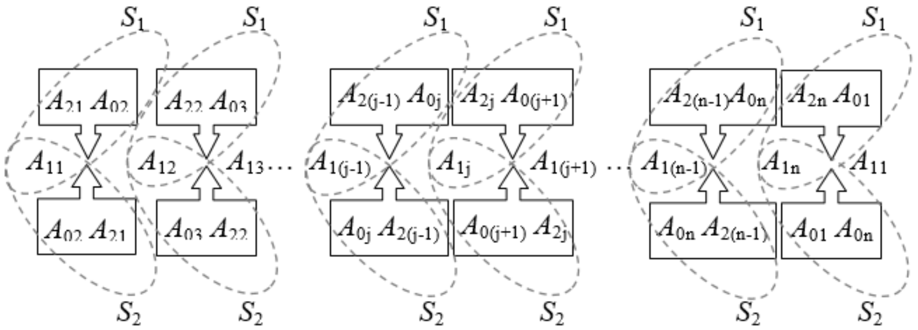


Fig. 6.5. Transition graph of 2^n possible cases to construct an n-unit cycle.

One of the pioneering studies initiating the two-machine SFRC in scheduling is by Sethi et al. (1992). Theorem 2 actually proves that the conjecture of Sethi et al. (1992), the optimal cycle of a SFRC is certainly one-unit, holds even if the cell is equipped with a MFR. In other words, an n-unit MFR move cycle must be decomposed into n sequences of S_1 and S_2 and its cycle time is the sum of the cycle times of the sub-cycles composing it. As a result of this theorem, the problem of finding the optimal n-unit MFR move cycle is simply reducible to the problem of finding the optimal one-unit MFR move cycle when the pickup criterion is free.

6.4 Analysis of the Interval and No-Wait Pickup Criteria

We start this section with reducing our MFRC scheduling problem to a problem where pickup criterion is interval and then propose a theorem for demonstrating the optimality region of all cycles. Before proceeding with this theorem, let us present a comprehensive definition of the interval pickup criterion. Due to the nature of production lines in some industries like steel, chemical and plastic manufacturing where the part should keep a constant temperature after finishing its operation on a particular machine M_i, the processing time at M_i is predominantly bounded within a pre-defined time interval (Paul et al., 2007). In other words, the maximum wait that any part j can have on M_i is $\overline{P}_i$, and the part will be scrapped if its waiting time on M_i, which is w_{ij}, be bigger than $\overline{P}_i$. Due to the fact that parts are usually expensive, it is more economical to find a cycle that imposes all limitations rather than scrap them.

Theorem 3. Whenever $P_1 + P_2 > 2\delta, P_1 + \overline{P}_1 \geq 2\varepsilon + 3\delta + \beta_{2j}$ and $P_2 + \overline{P}_2 \geq 2\varepsilon+3\delta+\beta_{0(j+1)}$, the two-machine MFRC under the interval pickup criterion reaches full utilization by execution of S_2. Otherwise, S_1 is optimal among all cyclic solutions of MFRC assuming anyone of $P_1 + P_2 \leq 2\delta, P_1 + \overline{P}_1 < 2\varepsilon + 3\delta + \beta_{2j}$ or $P_2 + \overline{P}_2 < 2\varepsilon + 3\delta + \beta_{0(j+1)}$.

Proof: we should emphasize that the free pickup criterion is actually a more general case of the interval one where $\overline{P}_1 = \overline{P}_1 = \infty$ is permited. Therefore, MFRCs under the interval pickup criterion also follow the optimality rule presented in Theorem 1 ($T_{S_1} < T_{S_2}$ if $P_1 + P_2 \leq 2\delta$, and also $T_{S_1} \geq T_{S_2}$ if $P_1 + P_2 > 2\delta$). However, implementation of cycles may be infeasible depending to the parameters $\overline{P}_1$ and $\overline{P}_2$. We know that a cycle which only has full waiting is certainly feasible, and only cycles with partial waiting may be infeasible. Thus, S_1 is always feasible and we only should find feasibility constraints for S_2. In fact, cycle S_1 guarantees that this scheduling problem always has a feasible solution. Regarding cycle S_2, considering the proof of Lemma 2, $max\{0, (2\varepsilon + 3\delta + \beta_{2j}) - P_1\}$ and $max\{0, (2\varepsilon + 3\delta + \beta_{0(j+1)}) - P_2\}$ are the time that MFR take to come back M_1 and M_2 after loading parts $j + 1$ and j and leaving them, respectively. Therefore:

$$\overline{P}_1 \geq max\{0, (2\varepsilon+3\delta+\beta_{2j})-P_1\} \longrightarrow P_1+\overline{P}_1 \geq 2\varepsilon+3\delta+\beta_{2j} \tag{6.6}$$

$$\overline{P}_2 \geq max\{0, (2\varepsilon+3\delta+\beta_{0(j+1)})-P_2\} \longrightarrow P_2+\overline{P}_2 \geq 2\varepsilon+3\delta+\beta_{0(j+1)} \tag{6.7}$$

where both constraints must be satisfied to prevent the occurrence of infeasible S_2. Even if one of them was not satisfied, S_2 is infeasible and consequently S_1 is optimal.

The MFRC scheduling problems under the interval pickup criterion can be divided into more applicable sub problems for scheduling based on pickup criterion. One of these sub problems that originates from the interval pickup criterion is MFRC scheduling problem under the no-wait pickup criterion. The no-wait here means the part waiting on none of machines is allowed, and the part must be unloaded as soon as its operation on any one of machines is finished(Shabtay et al., 2014; Che and Chu, 2005). Obviously, it is a subcase of the interval pickup criterion when $\overline{P}_1 = 0$ and $\overline{P}_1 = 0$.

Before proceeding with the next theorem, It should be noted that the result of Theorem 2 hold for interval and no-wait pickup criteria due to the fact that any one of them is a special case of the free pickup criterion where $\overline{P}_1 \neq \infty$ or $\overline{P}_2 \neq \infty$. In more detail, there are two possible cases for MFRCs with any one of interval and no-wait pickup criteria: 1) S_2 is a feasible cycle: Then, the n-unit cycle is a combination of k repetitions of S_1 and $n-k$ repetitions of S_2, and its per unit cycle time is $\frac{T_{S_n}}{n} = \frac{k(T_{S_1}) + (n-k)(T_{S_2})}{n}$ which cannot be less than $min\{T_{S_1}, T_{S_2}\}$. 2) S_2 is an infeasible cycle: Then, there is only one feasible one unit cycle under this condition, and this means that any n-unit cycle is actually n repetitions of S_1. Subsequently, $\frac{T_{S_n}}{n} = T_{S_1}$ and there is no need to take into consideration any $n > 1$.

Theorem 4. S_2 with the cycle time $T_{S_2} = 4\varepsilon + 4\delta + max\{P_1, P_2, \frac{P_1 + P_2 + \gamma + 3\delta}{2}\}$ is the optimal move cycle if:

1. $\begin{cases} P_1 \geq 2\varepsilon + 4\delta \\ P_2 \geq 2\varepsilon + 4\delta \\ \gamma \leq 3\delta \end{cases}$ or
2. $\begin{cases} P_1 - P_2 > \gamma - 3\delta \\ \gamma > 3\delta \end{cases}$ or
3. $\begin{cases} P_2 - P_1 > \gamma - 3\delta \\ \gamma > 3\delta \end{cases}$ or
4. $\begin{cases} |P_1 - P_2| \leq \gamma - 3\delta \\ P_1 + P_2 \geq 4\varepsilon + 5\delta + \gamma \\ \gamma > 3\delta \end{cases}$

Otherwise, S_1 is the optimal cycle of a two-machine MFRC under the no-wait pickup criterion.

Proof: In the proof of Lemma 2, we derived a four-phase analysis to calculate MFR waiting times for the free pickup criterion. Following this analysis and Equations (6.6) and (6.7), we present the cycle time and feasibility region of S_2 for the no-wait pickup criterion. For sub cases 1-3, at least one of P_1 or P_2 is less than $2\varepsilon + 4\delta$. Thus, implementation of S_2 is impossible when the values of input parameters are based on these three sub cases. Owing to both P_1 and P_2 are more than $2\varepsilon + 4\delta$ having sub cases 4, a link between cases 1, 2, 3, 4 here and 4.1, 4.2, 4.3, 4.4 in Lemma 2 can be demonstrated. This results in the following cycle times for cases 1, 2, 3, 4: $4\varepsilon + 4\delta + max\{P_1, P_2\}, 4\varepsilon + 4\delta + P_1, 4\varepsilon + 4\delta + P_2, 4\varepsilon + 4\delta + \frac{P_1 + P_2 + \gamma - 3\delta}{2}$. There is a Or operator between these cycle times which means the cycle time equals the maximum of them, $T_{S_2} = 4\varepsilon + 4\delta + max\{P_1, P_2, \frac{P_1 + P_2 + \gamma - 3\delta}{2}\}$.

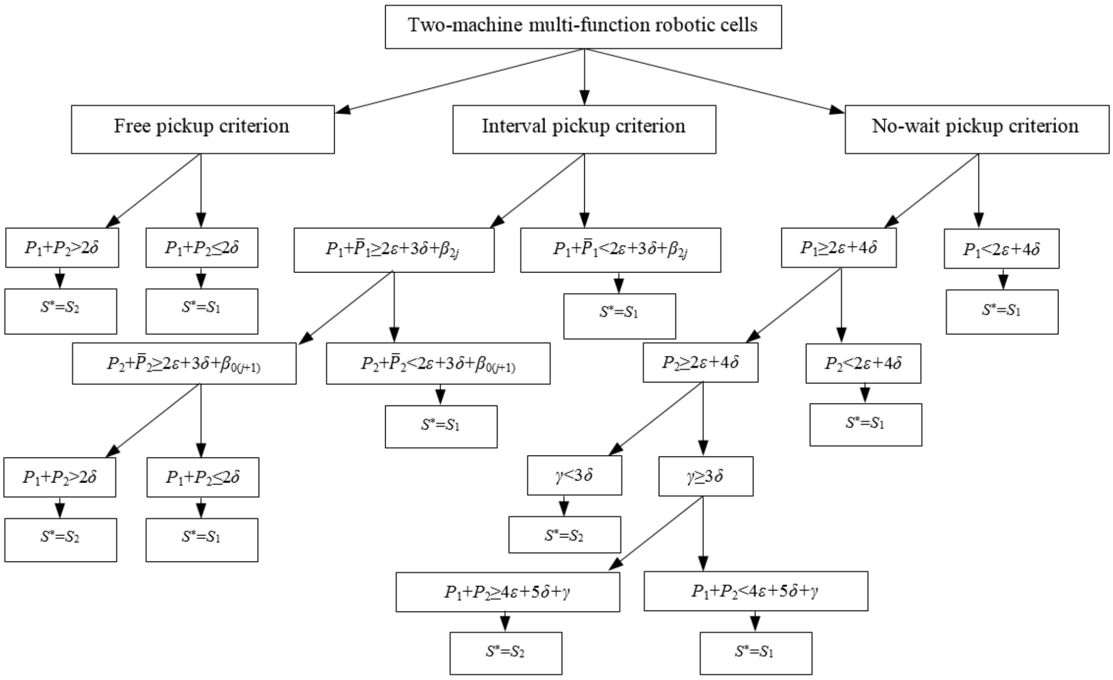

Two-machine multi-function robotic cells
Free pickup criterion
Interval pickup criterion
No-wait pickup criterion
$P_1+P_2>2\delta$
$P_1+P_2\leq2\delta$
$S^*=S_2$
$S^*=S_1$
$P_1+\bar{P}_1\geq2\varepsilon+3\delta+\beta_{2j}$
$P_1+\bar{P}_1<2\varepsilon+3\delta+\beta_{2j}$
$S^*=S_1$
$P_2+\bar{P}_2\geq2\varepsilon+3\delta+\beta_{0(j+1)}$
$P_2+\bar{P}_2<2\varepsilon+3\delta+\beta_{0(j+1)}$
$S^*=S_1$
$P_1+P_2>2\delta$
$P_1+P_2\leq2\delta$
$S^*=S_2$
$S^*=S_1$
$P_1\geq2\varepsilon+4\delta$
$P_1<2\varepsilon+4\delta$
$S^*=S_1$
$P_2\geq2\varepsilon+4\delta$
$P_2<2\varepsilon+4\delta$
$S^*=S_1$
$\gamma<3\delta$
$\gamma\geq3\delta$
$S^*=S_2$
$P_1+P_2\geq4\varepsilon+5\delta+\gamma$
$P_1+P_2<4\varepsilon+5\delta+\gamma$
$S^*=S_2$
$S^*=S_1$

Fig. 6.6. A summary of the results of robotic cell scheduling problems for different pickup criteria

Figure 6.6 summarizes the feasibility and optimality results concerning free, interval and no-wait cells established in this paper. Considering this figure, it can easily be verified that the cycle times obtained in Section 6.3 for the free pickup criterion will hold for the interval and no-wait pickup criteria. The only difference is that S_2 may sometimes be infeasible based on input data. This issue makes analyze of the problem a bit more complex because both the feasibility and optimality condition should be jointly considered. Theorem 3 shows that there is no guarantee that S_2 (in comparison with S_1) will be optimal even if it be feasible for a MFRC under the interval pickup criterion. As an extreme case of the interval pickup criterion, the no-wait pickup one studied in Theorem 4, and this resulted in clarifying the optimality region under this condition.

6.5 Concluding Remarks

We have addressed the result of research aimed at assessing the productivity gain accomplished by using a robot with a hybrid gripper. The gripper is able to perform a breakable operation on a part in transit from an input hopper to an output hopper of a production line dealing with two tandem machines. Assuming *stop resume* processing mode for the robot, it continues processing of the part when it is reloaded to the robot with no loss in time. At the starting point, the best proportions of the unique operation of the robot to be done between I, M_1, M_2 and O is determined graphically. Then, the cycle time of two one-unit cyclic solution have been obtained using this graphical representation of the operation on the robot, and following that the optimality region of each one of them has been determined when dealing with the free pickup criterion . This line of thought brings us to the result that S_1 is more productive for cells with short processing time on machines, and S_2 is more productive for cells with time-consuming processing time on machines. It is proved that any n-unit MFR cycle is a combination of k repetitions of S_1 and $n-k$ repetitions of S_2, and its cycle time is bigger than $min\{T_{S_1}, T_{S_2}\}$ and also less than $max\{T_{S_1}, T_{S_2}\}$. Subsequently, the optimal one-unit is the global optimal cycle for the MFRC. Furthermore, the analysis showed that there is no guarantee that S_2 (in comparison with S_1) will be optimal even if it be feasible for a MFRC under the interval pickup criterion. However, only if S_2 be feasible, we can conclude that it is the optimal cycle for a MFRC under the no-wait pickup criterion. A key topic of interest for future work on MFRCs is the extension of multiple-part MFRCs. This subject is challenging due to the NP-hardness of the problem when different kinds of parts should be produced. In fact, the problem is composed of determining the proportions of Γ, appropriate part sequencing, and finally MFR move sequence which jointly give us the optimal cycle time. In addition, the MFR studied here has a single arm. So, the future work can extend the results to MFRs with multiple arms.

Part III

Stochastic Modelling

Chapter 7 is based on the published article Foumani, M., Gunawan, I., Smith-Miles, K., 2015. Resolution of Deadlocks in a Robotic Cell Scheduling Problem with Post-process Inspection System: Avoidance and Recovery Scenarios. IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore, on pages: 1107-1111.

Abstract *The phenomenon of deadlock in robotic cells has been long ignored by most scheduling literature. A deadlock situation arises if a part cannot change its current state indefinitely since the destination machine is occupied by another part. The probability of the deadlock occurrence is likely to be large when the processing route cannot be predicted with certainty due to inspection processes. Our focus here is on a specific robotic cell with a post-process inspection system where the inspection is performed on an independent inspection machine. Avoidance and recovery policies are applied to overcome deadlocks originated from this cell. We develop these policies to prevent deadlock or alternatively resolve it during the online implementation of cycles. The former policy minimizes the storage cost, whereas the later policy minimizes the expected cycle time. An analysis of the scheduling problem that involves timings and costs is also carried out for comparing policies.*

Note to Practitioners *It has been long recognized that traditional deterministic modelling is not suitable for capturing truly dynamic behaviour of most real-world applications and certainly scheduling is one of them. Since the inspection process can be cause of uncertainty in the real world applications, the main focus of this study is on a class of robotic cells that offers inspection of products in a self-loop. Scheduling of cells with this class of robot is an important issue for practitioners in the area of robotics who seek techniques to improve the productivity of their companies. The outcome of this research helps production engineers to develop a time and cost effective methodology for stochastic scheduling of real-world cells employed in their own companies.*

Keywords *Scheduling, Cyclic production, Post-process inspection*

Classification $SRF^{1,1,1}_{2,2,1}|$*free, additive , stochastic , identical, cyclic*$|T$ with post-process inspection

Note References are considered at the end of the thesis.

Chapter 7

Resolution of Deadlocks in a Robotic Cell Scheduling Problem with Post-process Inspection System: Avoidance and Recovery Scenarios

7.1 Introduction

A robotic cell is a flow shop composed of a robotic arm, a number of production machines with single-unit capability, and a computerized control logic coordinating all movements of the robot and parts. Robotic cells are basically classified into robotic cells without inspection process and robotic cells with inspection process. It is always realistic to find a deterministic model for the robotic cells without rework assumption. Following that, there are many works in the literature dealing with the scheduling of the robot activities, as widely addressed in Dawande et al. (2005) for robotic cells without inspection process.

Basically there are two major methods of inspection called in-process and post-process inspections. For the in-process inspection, the measurement is performed by a set of sensors integrated into the production machine. Alternatively, the measurement is performed by an independent inspection machine located after the production machine if we follow the post-process inspection method. There are at least two reasons for the post-process inspection method. The inspection process predominantly needs a specific condition which cannot be satisfied by the production machine. This specific condition can be include an exact temperature or pressure. Thus, it is more demanding to perform the inspection process by an extra inspection machine. Additionally, performing both production and inspection steps by a production machine equipped with sensors may make this machine the bottleneck of the cell. Following that, it is often impossible or at least time-consuming to perform both production and inspection steps by the same machine. Hereinafter, the robotic cell with the inspection process means the robotic cell with the post-process inspection (RCPI) for the sake of simplicity.

The smallest possible RCPI, two-machine RCPI, is commonly captured by the succeeding framework: the cell is made up of a production machine M_1, an inspection machine M_2, a gantry robot that serves the entire production line, an input conveyor (I or the axillary machine M_0) and an output conveyor (O or the axillary machine M_3) with unlimited storage capacity. This framework makes it clear that typical robotic cells are a subdivision of RCPIs where all produced parts are failure-free. It should be emphasized that typical robotic cells can not give a guarantee that all finished parts have high quality if these robotic cells are not failure-free. Thus, using an appropriate RCPI is the best strategy to satisfy the customer's requirements. An example of two-machine RCPIs is illustrated in Figure 7.1 for the crankshaft production lines. This figure shows the robots control mechanism is complex to program in the two-machine RCPI due to the fact that all failed crankshafts must have backward movement from the machine M_2 to M_1 and this backward movement certainly causes a stochastic processing route.

Although a survey of different kinds of two-machine cells without an inspection process was developed by Dawande et al. (2005), there is no thorough study of robotic cells to determine a guideline for post-process inspections. Only some similar studies of robotic cells focused on the stochastic processing time due to machine failures (Savsar and Aldaihani, 2008; Tysz and Kahraman, 2010) and the speed of machines (Shafiei-Monfared et al., 2009; Geismar and Pinedo, 2010) to evaluate utilization of cells. In more detail, scheduling problems attempted in Savsar and Aldaihani (2008); Tysz and Kahraman (2010); Shafiei-Monfared et al. (2009); Geismar and Pinedo (2010) only concentrated on the case where the processing time of the part is stochastic, but the processing route of the part is fixed and not allowed to be changed. In contrast, here, we deal with a stochastic processing route results from the inspection process which increases the complexity of the problem.

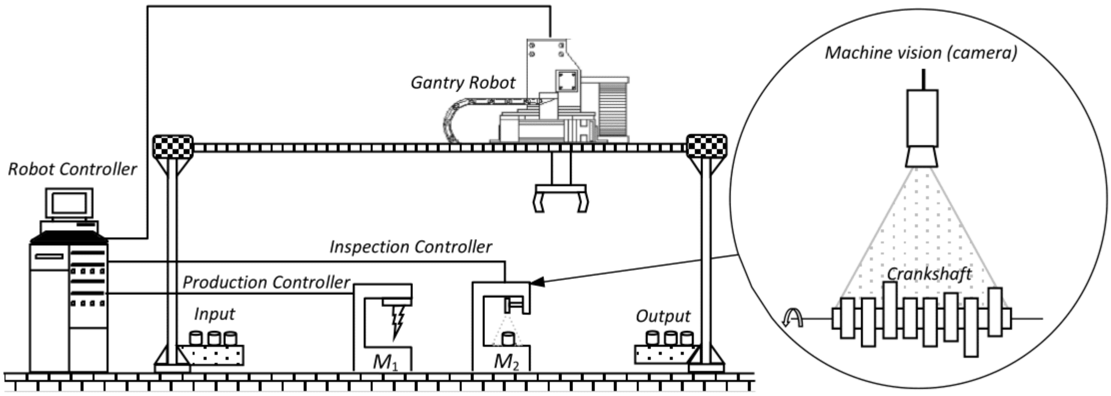


Fig. 7.1. A two-machine RCPI for the crankshaft production Ayub et al. (2014)

The outline of this study is as follows: Section 7.2 contains fundamental concepts related to RCPIs. Section 7.3 is devoted to avoidance and recovery policies for deadlock resolution. We initially show how the deadlock can occur in a two-machine RCPI, and then present possible control policies to prevent deadlock or alternatively resolve it. Following that, Section 7.4 is dedicated to the analysis of the problem involving timings as well as costs of both policies. Finally, Section 7.5 concludes the paper with perspectives.

7.2 Problem Notation and Definitions

Here, we define the problem of maximizing expected throughput and cost for a RCPI, and then summarize required notations. First, let us explain why an analytic study of two-machine RCPIs is vital. Actually, an efficient way for scheduling of a complex RCPI is to consider it as a combination of sub-RCPIs (Chan et al., 2008). As seen from Figure 7.2, a network of a six-machine printed circuit board assembly line with multiple robots can be replaced with three integrated two-machine RCPIs namely C_1, C_2 and C_3. Also, a case study of press machines for draw-forming of automobile body panels was undertaken by Osakada et al. (2011). The press line consisted of integrated two-machine cells which can be extended for two-machine RCPIs.

We elaborate on cycles and define terms that are applied for classifying them. Much of the researches to date in the field of robotic cell scheduling have concentrated on considering a cycle as a permutation of a number of the robot activities. Subsequently, the *Robot activity* defined as follows is a basic concept that plays a key role in this field. Notice that Definition 1 is borrowed from Crama and van de

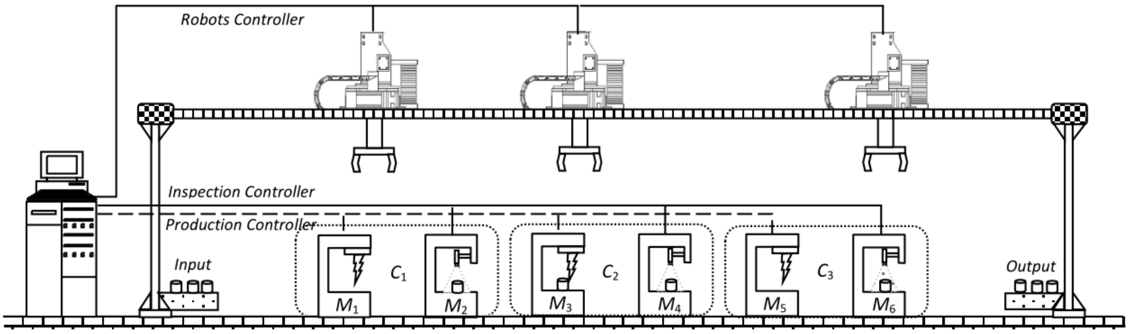


Fig. 7.2. A clustering scheme for a set of three integrated two-machine RCPIs

Klundert (1997).

Definition 1. The forward activity namely F_i, $\forall i \in \{0, 1, 2\}$, is implemented with respect to a two-phase sequence of the robot actions for a two-machine RCPI:

1. The robot takes a part from M_i if the part is not identified by tester as faulty (damaged, incorrect and so on).
2. The robot carries the part to M_{i+1}, and loads it onto this machine.

Definition 2. The backward activity called B_i, $\forall i \in \{1, 2, 3\}$, is implemented with respect to an inverse sequence of the robot actions for a two-machine RCPI:

1. The robot takes a part from M_i when it is recognized as a defective part.
2. The robot carries the part back to M_{i-1}, and finally loads it onto M_{i-1} for a rework process.

Obviously, the starting point of backward activities must be an inspection machine, and this means there is only one possible backward activity, B_2, for a two-machine RCPI. However, any one of I, M_1, or M_2 can act as the starting point of the corresponding forward activity. Therefore, we characterise a particular n-unit cycle as follows:

Definition 3. An n-unit cycle is a permutation of forward and backward activities in which any forward activities F_0 and F_2 is repeated exactly n times.

It is clear that an n-unit cycle produces n final products. We are also able to say that the lower bounds of repetitions of B_2 and F_1 are zero and n, respectively, due to the fact that failure occurrence for any particular part follows a stochastic nature. Actually, B_2 and F_1 together build up the stochastic closed-loop event (B_2, F_1). This stochastic closed-loop event is associated with the probabilities of the rework being needed after inspection of the part on M_2. We make the following assumptions regarding (B_2, F_1):

1. Elements of (B_2, F_1) have deterministic occurrence time, and
2. The number of switching into (B_2, F_1) for the kth part is stochastic.

The most popular case of an n-unit cycle in industry is the *one-unit cycle* which produces exactly one part in each iteration of it. Let us assume that the first occurrence of F_1 is placed in the last position of each cycle, so that the sequence of other activities is either $(B_2, F_1)^k, F_2, F_0$ or $F_0, (B_2, F_1)^k, F_2$ where $k \in N$. This results in two cycles called $S_1 = (B_2, F_1)^k, F_2, F_0, F_1$ and $S_2 = F_0, (B_2, F_1)^k, F_2, F_1$. Both cycles are finished with the first occurrence of F_1 for the next part in the cell because the only moment when the robot can switch from S_1 to S_2 (or S_2 to S_1) is after performing F_1. The reason that one-unit cycles are popular is that the partial cycle time of any n-unit cycle is a convex combination of the expected partial cycle time of S_1 and S_2 as two given corner points. Thus, we should only analyse one-unit cycles (Sethi et al., 1992). We recall a definition related to stochastic dominancy due to stochastic nature of RCPIs Ross (1996).

Definition 4. For two random variables μ and λ, we say μ is second-order smaller than λ only if $E(\mu) \leq E(\lambda)$.

Let us use the following notation predominately derived from Geismar and Pinedo (2010) throughout the text:

ε The load/unload time of machines by the robot

δ The required time for traveling between adjacent location pairs

a The processing time of M_1

b The inspection time of M_2

p The probability of identifying no defect in each time that M_2 inspects the part

q The probability of detecting the parts failure in each time that M_2 inspects the part

c_1 The capital cost of the internal buffer

c_2 The storage cost of the internal buffer

$c^k_{S_j}$ The overall cost of the kth implementation of $S_j, \forall j \in \{1, 2\}$, under recovery policy

w^k_{il} The robots waiting time at $M_i, \forall i \in \{1, 2\}$ before lth times of unloading the part k

$T^k_{S_j}$ The partial cycle time of $S_j, \forall j \in \{1, 2\}$, for the part k fed to the two-machine RCPI

It is assumed that ε for all machines and buffers is constant, and δ is both symmetric and additive.

7.3 Avoidance and Recovery Policies

The main aim of this section is to propose two on-line policies for avoiding and resolving deadlocks in RCPIs. Before proceeding with avoidance and recovery policies, let us present preventive constraints which are enough to keep typical robotic cells away from deadlock (Dawande et al., 2005):

1. The robot must not load a part onto a busy machine.
2. The robot must not unload a part from an idle machine.

Aforementioned constraints are only able to avert deadlock in the robotic cells without inspection process, which follow Off-Line Programing (OLP). However, a counterexample for these constraints is depicted in Figure 7.3 to show that they are necessary but not sufficient for deadlock prevention. In this figure, the part on M_1 is instructed to visit M_2 for inspection, whereas the part on M_2 may be defective and consequently needs to revisit M_1. This closed chain, namely circular deadlocking (Venkatesh and Smith, 2003), increases the risk of deadlocking even though both constraints are satisfied. Thus, a set of additional constraints is demanded to make RCPIs deadlock-free or resolve the circular deadlock.

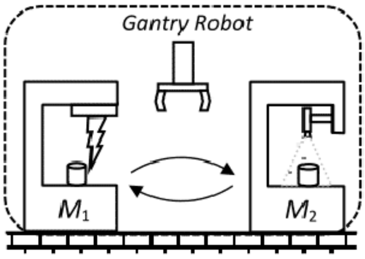


Fig. 7.3. Counterexample of a two-machine RCPIs deadlock

A variety of on-line control policies can be adopted to overcome circular deadlocks in RCPIs, and each one may cause an increase in manufacturing costs or the complexity of control (Venkatesh and Smith, 2003). Thus, they are divided into two main groups for the sake of simplicity and effectiveness:

- *Avoidance Policy*: As the name of this policy shows, it works by avoiding any kind of the circular deadlocks. Actually, the avoidance policy allows the robot to load the part on a machine if the machine is a deadlock-free resource. On the contrary, loading the part on a machine is prohibited if the machine is a deadlock-risk resource, which can lead to a circular deadlock.

- *Recovery Policy*: This policy allows the occurrence of the circular deadlock in the first stage, and then recovers it using a correction method in the second stage. The correction method can be an internal buffer with one-unit storage capacity (or a local conveyor) attached to the machine where deadlock occurs.

Clearly, anyone of these on-line policies has its own advantages and disadvantages. On the one side, avoidance policy minimizes the cost of reworking per unit since there is no need for internal buffer installation. On the other side, the expected throughput of the two-machine RCPI will increase if we accept storage cost per unit and the constant installation costs of the internal buffer. As a consequence, it is essential to provide a mathematical evaluation of the tradeoffs between expected throughput and cost. The main aim of the next section is to make this evaluation in keeping with on-line control policies.

7.4 Solution for the Problem Involving Cycle Time and Storage Cost

A description of the recently established approaches of analyzing the bicriteria robotic cell scheduling problem is initially presented here. Research on the bicriteria robotic cells fall into two streams. One stream is dedicated to bicriteria robotic cells without an inspection process.Following that, Gultekin et al. (2008) looked into the throughput rate as well as machine operating costs, and then determined the set of nondominated solutions for two-machine robotic cells. The novelty of their study was the introduction of each operation time as a nonlinear function of the machine specifications. Also, Gultekin et al. (2010) extended this approach for flexible robotic cells without inspection process. Another stream of research concentrates on bicriteria RCPIs. To the best of our knowledge, there is no research in this area.

As mentioned before, M_2 identifies no defect in each time inspection of the part k with probability p. This means the inspection result of the part is a Bernoulli variable Y_k with parameter p. However, it is essential to estimate the number of the part failure to pass M_2 before the first success. This number is denoted by a geometric random variable X_k with success parameter p in which the time elapsed between two successive inspections equals the required time for performing the closed-loop event $(B_2, F_1)^k$. The reason for this intuition is that the geometric distribution is defined as a discrete and memoryless distribution counting the number of Bernoulli trials until the first success. This leads to following results.

Lemma 1. The partial cycle time for kth implementation of S_1 is the random variable $T^k_{S_1}$ given by:

$$T^k_{S_1} = 6\varepsilon + 6\delta + a + b + (4\varepsilon + 2\delta + a + b)X_k \tag{7.1}$$

Proof: The sequence of activities of S_1 is $(B_2, F_1)^k, F_2, F_0,$ F_1. Subsequently, S_1 starts with the stochastic closed-loop event and repeat it X_k times. Each iteration of the loop includes the robot tasks below: unloading the part k from M_2, carrying the part from M_2 to M_1, loading the part onto M_1, a full waiting at M_1 to finish the processing of the part, unloading the part from M_1, carrying the part from M_1 to M_2, loading the part onto M_2, and finally a full waiting at M_2 $(4\varepsilon + 2\delta + a + b)$. At this phase, the part k met with the first success. Therefore, the robot drops off it at O instead of M_1, moves to I, picks up the $(k{+}1)$th part, carries the part to M_1, loads the part on M_1, has a full waiting at M_1, unloads the part from M_1, carries the part to M_2, loads the part on M_2, and finally has a full waiting at M_2 before starting the $(k{+}1)$th stochastic closed-loop event $(6\varepsilon + 6\delta + a + b)$. This completes the proof.

It is noteworthy that S_1 is always deadlock-free and has no need for internal buffer installation on M_1. This assumption is not applicable for S_2. Therefore, we have two alternatives: 1. RCPI strictly adopts an avoidance policy and consequently only implements S_1. 2. RCPI adopts a more flexible control policy (recovery policy) and also implements S_2. Note the second alternative increases the per unit storage cost which will be specified later on. Let us initially proceed to find $T^k_{S_2}$ in Lemma 2, and then prove Theorems 1 and 2 together to find the optimal partial cycles without cost assumptions.

Lemma 2. The partial cycle time for kth implementation of S_2 is the random variable $T^k_{S_2}$ given by:

$$T^k_{S_2} = \eta + \begin{cases} \alpha & \text{if } X_k = 0 \\ (4\varepsilon + 2\delta)X_k + \beta + \gamma(X_k - 1) + \zeta & \text{if } X_k \geq 1 \end{cases} \tag{7.2}$$

Where

$6\varepsilon + 8\delta = \eta$
$max\{0, a - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\} = \alpha$
$max\{0, a - (\varepsilon + 2\delta), b - (2\varepsilon + 4\delta)\} = \beta$
$max\{a - (3\varepsilon + 2\delta), b\} = \gamma,\ max\{a - (2\varepsilon + 4\delta), b\} = \zeta$

Proof: If X_k=0, there is no difference between the RCPI and a typical robotic cell. We refer to Sethi et al. (1992) for this case where the *max*term equals $w^k_{11} + w^k_{21}$. We have two cases if $X_k \geq 1$:

1) For $X_k = 1$, the first repetition of $(B_2, F_1)^k$ is generated, and therefore we consider the following occurrence time of the event instead of the *max*term generated for $X_k = 0$: The robot initially waits at M_2 to receive the inspection result, unloads the failed part k from M_2, moves backward to M_1, waits at M_1 until this machine finishes the processing of the part $k + 1$, loads the part k on M_1 after transferring the part $k + 1$ from M_1 to its internal buffer, unloads the part $k + 1$ from the internal buffer, moves forward to M_2, loads the part $k + 1$ on M_2. The time taken by the robot to perform the closed-loop event is $(4\varepsilon+2\delta)+w^{k+1}_{11}+w^k_{21} = (4\varepsilon+2\delta)+max\{0, a-(\varepsilon+2\delta), b-(2\varepsilon+4\delta)\}$. Following this event, for the finally portion of the cycle, the robot transfers the part $k+1$ from M_2 to O, drops it off at O, returns to M_1 to unload the part k after a partial waiting, and loads it on M_2. This causes a waiting time $w^k_{12} = max\{0, a - (2\varepsilon + 4\delta), b\}$ for the part k in comparison with the previous case.

2) For $X_k = j$, $\forall j \in \{2, 3, ...\}$, we should only add elapsed time of the current closed-loop event to this time for the current closed-loop event $j - 1$. Considering an approach similar to the Case 1, this cumulative time equals $(4\varepsilon + 2\delta) + max\{0, a - (3\varepsilon + 2\delta), b\}$. This completes the proof.

Corollary 1. The expected values of two cycle time are:

$$E(T^k_{S_1}) = 6\varepsilon + 6\delta + a + b + (4\varepsilon + 2\delta + a + b)\tfrac{q}{p} \tag{7.3}$$

$$E(T^k_{S_2}) = 6\varepsilon+8\delta+\alpha p+((4\varepsilon+2\delta)\tfrac{1}{p}+\beta+\gamma(\tfrac{q}{p})+\zeta)q \tag{7.4}$$

Corollary 2. On the contrary to all earlier research on the robotic cell, the two-machine RCPI does not necessarily act like a First In First Out (FIFO) system since it is possible that any one of parts $k+1, k+2, ...$ can be picked from I after the part k, but be dropped into O before the part k.

Theorem 1. No dynamic state transition from S_1 to S_2 (or from S_2 to S_1) exists for a two-machine RCPI.

Proof: The start state of both cycles is the instant of time when the robot loaded a part on M_2 for the first time and stops at M_2 for the upcoming order. If M_1 was occupied at this instant of time, the stochastic programming was unavoidable since the extent to which any particular part was processed on M_1 at the initial state of the corresponding partial cycle does not have steady-state behaviour. Nevertheless, M_1 is unoccupied at the start and end states of both cycles, and no dynamic transition from S_1 to S_2 (or from S_2 to S_1) exists. This completes the proof.

Theorem 2. Comparing the two-machine RCPI with the same cell without the post-process inspection, the chance of optimality for S_2 increases whereas it decreases for S_1.

Proof: For any arbitrary$X_k = j, \forall j \in \{1, 2, 3, ...\}$, we have $T^k_{S_1} \leq T^k_{S_2}$ if $6\varepsilon + 6\delta + a+b+(4\varepsilon+2\delta+a+b)X_k \leq 6\varepsilon+8\delta+(4\varepsilon+2\delta)X_k+\beta+\gamma(X_{k-1})+\zeta \implies (a+b)(j+1) \leq 2\delta + \beta + \gamma(j-1) + \zeta$. This inequality is only feasible when $\beta = 0, \gamma = b, \zeta = b$, and finally $a(j+1) + b \leq 2\delta$. It is also know that S_1 is optimal cycle for a typical robotic cell when $a + b \leq 2\delta$ (Sethi et al., 1992). Thus, we conclude that there is a reverse relationship between j and the chance of optimality for S_1. This completes the proof.

The above theorem shows that execution of S_2 for a two-machine RCPI is a good idea. However, there are two types of costs for the implementation of S_2 which make it challenging: capital and storage costs. On the one side, capital cost is fixed, time-independent and one-time expense incurred on the purchase of the one-unit internal buffer of M_1, whereas storage cost is variable, time-dependent and imposed by storing the part having finished processing on M_1 until M_2 be available for the inspection process. Here, we consider the storage cost per unit time for each part. The reason behind this assumption is that it reflects a commonly observed situation in the food industry and plastic molding in which parts are stored at a constant temperature. The overall cost of the kth implementation of S_2 under recovery policy is:

$$c_{S_2}^k = c_1 u + c_2 T_{S_2}^k \tag{7.5}$$

, where u is the constant internal buffer usage rate. The structure of Equation (7.5) shows that the first portion of this equation is constant, but the last portion of it is a stochastic function of the cycle time. Note $c_{S_1}^k$ always equals zero. Now, we come to the objective function of the S_2 which should be compared with the partial cycle time of S_1. This function, $T_{S_2}^k + \varphi c_{S_2}^k$, is chosen as a linear combination of the partial cycle time of S_2 and the overall cost with relative weight φ. when the overall cost is a critical criterion and the company suffers financially, φ should be big enough. Also, $\varphi = 0$ if there is no financial limitation for adding an internal buffer to the cell.

Theorem 3. The implementation of avoidance policy has a better performance if $T_{S_1}^k$ is second-order smaller than $T_{S_2}^k$.

Proof: As mentioned before, the avoidance policy has no cost in comparison with the recovery policy. Also, it has a better performance in term of the cycle time if $E(T_{S_1}^k) \leq E(T_{S_2}^k)$ for a RCPI. Thus, there is no reason for using of the recovery policy in such a RCPI. This completes the proof.

Theorem 4. The implementation of recovery policy is more effective if $E(T_{S_2}^k) \leq \frac{\frac{a+b}{p} - 2\delta - \alpha p - (\beta + \gamma \frac{q}{p} + \zeta) q - \varphi c_1 u}{\varphi c_2}$.

Proof: We omit the proof which is very similar to that of Theorem 3. Only, it should be mentioned that the result directly follows from the fact that the recovery policy is more effective when $E(T_{S_2}^k + \varphi c_{S_2}^k) \leq E(T_{S_1}^k)$. This completes the proof.

It is worth noting that the results of this section along with the previous section create a framework for time and cost analyzing of control policies. This helps the companies which are enthusiastic about using post-process inspections in their automated systems to satisfy customer needs.

7.5 Concluding Remarks

An analytical method for minimizing the partial cycle time and cost of cells with the post-process inspection has been developed in this study. We have shown that not only it is possible to avoid deadlock, but also resolve it during the online implementation of the robot move cycles using avoidance and recovery control policies. The avoidance policy minimizes the cost of reworking per unit while the recovery policy decreases the expected cycle time. After reaching the the steady state of the cell, we have evaluated the tradeoffs between these two criteria. Comparing cells with the post-process inspection with cells without this additional step has made it clear that the performance of the partial cycle S_2 is improved due to the fact that the average time of producing a part is definitely increased. Further work must be done to substitute a post-process inspection system with a robotic inspection system. In this system, the inspection process is performed on a multi-function robot which is able not only to transfer the part between two machines but also to inspect it in transit.

Chapter 8 is based on the published article Foumani, M., Smith-Miles, K., Gunawan, I., Moeini, A., 2016. Stochastic Scheduling of an Automated Two-machine Robotic Cell with In-process Inspection System. Computers and Industrial Engineering, Submitted in March 2016.

Abstract *This study is focused on the domain of a two-machine robotic cell scheduling problem for three various kinds of pickup criteria: free, interval, and no-wait pickup criteria. We propose an analytical method for minimizing the partial cycle time of such a cell with a PC-based automatic inspection system. It is assumed parts are inspected in one of the production machines, and this may result in a rework process. The stochastic nature of the rework process prevents us from applying deterministic solution methods for the problem. This study aims to develop an in-line inspection of identical parts using multiple sensors. Initially, we present a heuristic method that converts a multi-sensor inspection system into a single-sensor inspection system. Then, the expected sequence times of two different cycles are derived based on a geometric distribution, and finally the maximum expected throughput is pursued for each individual case with free pickup criterion. Results are also extended for the interval and no-wait pick up scenarios as two well-solved classes of the scheduling problem. The waiting time of the part at each machine after finishing its operation is bounded within a fixed time interval in cells with interval pickup criterion, whereas the part is processed from the input to the output without any interruption on machines in cells with no-wait pickup criterion.*

Note to Practitioners *These cells are widely employed in the inspection of automotive products such as vehicle doors. In detail, optimization of robotic spot-welding and press for a sheet-metal in automotive industry is common in practice. An application of a robotic rework cells is given in Osakada et al. (2011). It is related to press lines for draw-forming of automobile body panels for Honda (with Aida Engineering), Toyota (with Komatsu) and BMW (with Schuler), where the space between press machines were compact enough only for accommodating robots.*

Keywords *Scheduling, Rework, Robotic cell, Performance*

Classification

$SRF^{1,1,1}_{2,2,1}|$*free, additive , stochastic , identical, cyclic*$|T$ with in-process test
$SRF^{1,1,1}_{2,2,1}|$*interval, additive , stochastic , identical, cyclic*$|T$ with in-process test
$SRF^{1,1,1}_{2,2,1}|$*no-wait, additive , stochastic , identical, cyclic*$|T$ with in-process test

Note References are considered at the end of the thesis.

Chapter 8

Stochastic Scheduling of an Automated Two-Machine Robotic Cell with In-process Inspection System

8.1 Introduction

Robotic cells are one of the complicated application areas of flow-shops that have received a considerable amount of attention in robot move sequencing (Brauner, 2008). They are basically classified into two categories: the robotic cells without rework assumption and robotic rework cells. The term "rework" here means that a processed part may need reprocessing. Therefore, it is cycled between test and processing stations until deemed acceptable. It is straightforward to find a deterministic model for the robotic cells without rework assumption. Following that, there are many studies in the literature dealing with the scheduling of the robot activities, as widely addressed in Dawande et al. (2005) for two-machine cells. Nonetheless, inspection and rework stages in a robotic cell is one of the important issues in the field of robotic cell scheduling which reflects most real-life cases. This paper addresses the stochastic issues that arise when considering inspection and rework stages, laying some important analytical foundations for this under-studied problem.

A robotic cell with an additional inspection process in one of the rework stages is called a robotic rework cell. A two-machine robotic rework cell which is the smallest possible robotic rework cell is commonly captured by the following framework: the cell is made up of two production machines M_1 and M_2, multiple contact/non-contact sensors installed into M_1 or M_2, a gantry robot that serves the entire production line, an input conveyor (I or the axillary machine M_0) and an output conveyor (O or the axillary machine M_3) with unlimited storage capacity. This framework makes it clear that typical robotic cells are a special case of robotic rework cells where there is no inspection sensor on production machines, and all produced parts are failure-free.

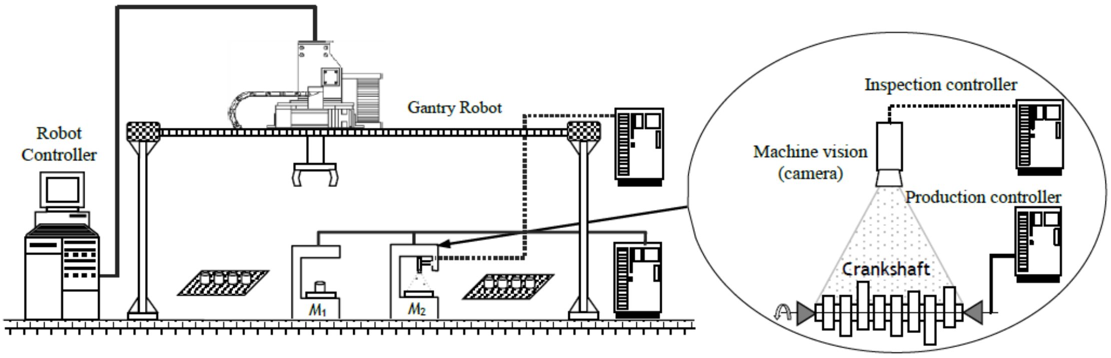


Fig. 8.1. A two-machine robotic rework cell with end of line inspection

Two-machine robotic rework cells are classified into two groups: two-machine robotic rework cells with start of line inspection (RRCSI) and two-machine robotic rework cells with end of line inspection (RRCEI). They are also called "Start of Line" testing and "End of Line" testing, respectively. An example of two-machine robotic rework cells with end of line inspection is shown in Figure 8.1 for the crankshaft production lines (Ayub et al., 2014). A particular crankshaft being processed goes through I, the lathe machine M_1, the lathe machine M_2 and O under this part processing route. After loading the crankshaft to any one of the lathe machines, the robot either waits for the crankshaft to finish its operation or immediately moves to another occupied lathe machine or I for unloading a new crankshaft. The difference between two machines is that the crankshaft is failure-free when it is processing on M_1, whereas the crankshaft may fail and need rework when it is processing on M_2.

Another example of robotic rework cells with inspection is extracted from cluster tools which are employed in processes such as deposition and inspection. The reason why we give this example is that cluster tools actually act as closed mini-environment robotic cells (Dawande et al., 2005). For fabrication of wafer in cluster tools, Atomic Layer Deposition (ALD) is a process that controls the wafer thick-ness by repeating the deposition processes with mono-atomic layer precision as needed. The quality of the wafer is often inspected by Spectroscopic Ellipsometry (SE) inspection method in order to check whether a conformal layer is fabricated. Therefore, the thickness of the wafer is inspected by SE method during madding the depositions of each layer on wafer (Langereis et al., 2009).

We should mention that a wide variety of real-life studies of production environments have been conducted on noncyclic production aiming at minimization of the maximum completion time (also referred to as makespan) (Chu et al., 2010). However, this study is limited to cyclic scheduling of the robotic rework cell due to its popularity in mass production environments in which the robot is applied for material handling. A cyclic schedule is based on a repeating pattern of part processing (Kayaligil and Ozlu, 2002). Robotic cells under consideration in this study can do rework processes, and consequently the stochastic nature of the rework process prevents us from applying existing deterministic solution methods for the cyclic scheduling problem. Therefore, the study of robotic cells without rework assumption, which has a deterministic processing route, will not be reviewed in this paper. We refer readers to the rigorous analysis of robotic cells with deterministic data elaborated in the book by (Dawande et al., 2007).

The analysis of stochastic robotized cells has a fragmented history of development in the literature. Some studies of stochastic robotic cells concentrated on two-machine cells which operate under a production system with machine failures and repairs (Tysz and Kahraman, 2010; Savsar and Aldaihani, 2008), whereas some others touched on the robotic cells with stochastic processing times (Shafiei-Monfared et al., 2009; Geismar and Pinedo, 2010). To our best knowledge, there is a lack of research on the stochastic part processing route although it is applicable to many practice-oriented scenarios. Note that the stochastic part processing route is actually possible when a set of sensors is integrated into at least one of the machines. We will provide more details of the concept of robotic rework cells later. Let us now summarize the main contributions of our study as follows:

- We present a proof of dynamicity of the problem of determining the optimal one-unit cycle for robotic rework cells with more than two machines. In contrast, we also prove that the problem is not dynamic for two-machine robotic rework cells, and the pickup criterion has no impact on the result of this theorem.
- We define three kinds of stochastic order relations and sort them through strangeness. Based on these relations, we find the dominancy regions of two feasible one-unit cycles by an analytical method. Then, we extend the result to a mass production environment under free pickup criterion.
- We extend results for the interval and no-wait pick up scenarios as two well-solved classes of this scheduling problem. For the sake of generalization, we graphically represent the linkage between optimality regions of feasible cycles of different pickup criteria to provide an integrated framework.

The overall structure of this paper is as follows. Section 8.2 contains fundamental concepts related to stochastic robotic cells. Sections 8.3 and 8.4 are dedicated to RRCSI and RRCEI scheduling problems to cover "Start of Line" and "End of Line" testing approaches for a cell with free pickup criterion, respectively. Following that, results are extended for interval and no-wait pickup scenarios in Section 8.5. Finally, Section 8.6 concludes the paper with perspectives.

8.2 Problem Notation and Definitions

We begin this section by first giving a precise definition of each problem class, and then summarizing the notations used in robotic rework cells to find the expected partial cycle times. The concept of the robot activity is one of the best tools to consider a cyclic formulation of the robot movement. In a two-machine robotic rework cell, the robot activity noted $A_i, \forall i \in \{0, 1, 2\}$, corresponds to the following sequence of actions: 1) The robot takes a part from M_i if the inspection process discovers no error. 2) The robot immediately carries the part to M_{i+1}. 3) The robot immediately completes the activity A_i by loading the part onto M_{i+1}.

Consistent with Brauner (2008), a particular n-unit cycle can be characterised by a permutation of activities in which any activity is repeated exactly n times. Since A_2 is one of the activities with n repetition, we can conclude that an n-unit cycle is able to produce n final products. Therefore, the cycle time is per unit cycle time or the time to complete one part in a cyclic behavior. An important point about the permutation is that the rework cell must return to the initial state at the starting of the permutation after completing it. In addition, scheduling of cells must be constrained to the deadlock-free subregion of permutations. The deadlock-free schedule can be guaranteed only if the following activities are executed:

- The receiving server and sending server must be empty and loaded before the load/unload process, respectively (Foumani et al., 2014).
- The robot can load a part to M_i only if the last process of the part was performed by $M_{i-1}, \forall i \in \{1, 2, 3\}$ (Foumani et al., 2014).
- Since the part's failure occurrence is possible, additionally, it is forbidden to unload the machine when its current part needs rework.

It should be noted that the number of n-unit cycles of a particular cell depends on the first two deadlock prevention instructions, whereas the last instruction only increases the cycle time. The simplest case of n-unit cycles is one-unit cycles where exactly one completed part leaves from the cell after the cycle's execution. Let us fix A_1 in the last position, so the order of other activities can alternatively be A_2, A_0 or A_0, A_2. This results in two cycles, namely $S_1 = A_2, A_0, A_1$ and $S_2 = A_0, A_2, A_1$. It is necessary to mention that we choose A_1 as the last activity of both cycles since it is the only activity that is able to act as switching point from one cycle to another. In other words, the only common state between S_1 and S_2 is exactly the moment when the robot loaded a part on M_2 and waits at M_2 until it receives the next order by the robot controller. The key note here is that any n-unit cycle is actually a combined sequence repeating S_1 exactly q times and also S_2 exactly $n - q$ times in each iteration of the cycle. Therefore, it is expected that per unit partial cycle time is a convex combination of the expected partial cycle time of S_1 and S_2 as two given corner points. This means that the expected partial cycle time of one of cycles S_1 or S_2 always dominates the expected per unit partial cycle time of any given n-unit cycle. As a result, it is enough to limit our search to S_1 and S_2 (Sethi et al., 1992). Now, let us use the following notation derived from Geismar and Pinedo (2010) throughout the text:

ε The load/unload time of machines (or buffers) by the robot.

δ The required time for traveling between adjacent location pairs (I, M_1), (M_1, M_2), and (M_2, O).

m_i The number of sensors installed into $M_i, \forall i \in \{1, 2\}$

a The processing time of the first machine.

b The processing time of the second machine.

$T^k_{S_j}$ The partial cycle time of S_j for the kth part fed to the rework cell, $\forall j \in \{1, 2\}$ and $\forall k \in \mathbb{N}$

w^k_i The waiting time of the robot at M_i for the kth part fed to the rework cell, $\forall i \in \{1, 2\}$ and $\forall k \in \mathbb{N}$

The notation w^k_i is also useful for definition of three different pickup criteria. This waiting time means that the robot has to wait for machine i to finish processing if it reaches the machine before finishing the operation. However, instead, it is possible that the robot reaches the machine i after the machine finishes the part processing. Under this condition, the part has to wait for the robot to reach and remove it. This waiting time of the part can provide a framework for classification of pickup criteria as follows:

• *The waiting time of the part on the machine is unbounded*: this case is named free pickup criterion in which the completed part can stay indefinitely on the machine before removing by the robot. Hence, the waiting time of the part can be shown by interval $[0, +\infty), \forall i \in \{1, 2\}$.

• *The waiting time of the part on the machine is bounded*: this case is named interval pickup criterion in which the waiting time of the part at $M_i, \forall i \in \{1, 2\}$, is bounded within a pre-defined time interval. The interval is $[0, \overline{a})$ for M_1 and $[0, \overline{b})$ for M_2. Therefore, the maximum wait times of the part on M_1 and M_2 are $\overline{a}$ and $\overline{b}$, respectively.

• *The waiting time of the part on the machine is zero*: this case is no-wait pickup criterion in which the part must be removed from $M_i, \forall i \in \{1, 2\}$, without delay and carried to the next machine, which is M_{i+1}.

It is noticeable that different pickup criteria do not affect the number of n-unit cycles although they may affect the cycle times. More precisely, the waiting times of cycles change according to the above description of pickup criteria and this may also change the total cycle times. We can simply extend this result for one-unit cycles S_1 and S_2 in robotic rework cells with two machines, and say that the permutations of S_1 and S_2 are independent from pickup criteria.

At the end of this section, we should emphasized that we cannot be assured that the robotic rework cell operates in steady state as a result of its dynamic nature. In this regard, we define the variable $C(A_1, k)$ as the completion time of the kth implementation of A_1 to explain why there is no guarantee of cyclic behaviour of the cell (Crama and van de Klundert, 1997). Obviously, $C(A_1, k) - C(A_1, k-1)$ does not remain constant for $\forall k \in \mathbb{N}$, and it is vital to carry out a separate analysis for any $C(A_1, k) - C(A_1, k-1)$, which is named the partial cycle time $T^k_{S_j}$ for the kth implementation of S_j.

8.3 Scheduling of RRCSIs with Free Pickup Criterion

We start this section by reducing the scheduling problem to a problem where pickup criterion is always free. Under this condition for the RRCSI case, a part processing on M_1 is monitored to detect the presence of different types of defects using a multi-sensor system. Each sensor $i, \forall i \in \{1, 2, ..., m_1\}$, detects the part's failure with specified probability q_{1i} each time the part is processing on M_1, and therefore this individual sensor identifies no defect in each time inspection with probability $p_{1i} = 1 - q_{1i}$. This means that the inspection result of each sensor i after each time processing of the part k on M_1 is a random variable Y_{1i}^k which follows a Bernoulli probability distribution of parameter p_{1i}. Needless to say that here we have a sub-inspection set of m_1 different sensors, and the sequence of random variables $[Y_{11}^k, Y_{12}^k, ..., Y_{1m1}^k]$ that represent the inspection results of m_1 sensors is independent and identically distributed (i.i.d) since all of them follow Bernoulli probability distribution and there is no dependency between them. Therefore, the part cannot pass the multi-sensor inspection process even if one of the sensors detects the part's failure. Following that, the generalized Bernoulli distributed variable Y_1^k supporting the success of the multi-sensor inspection system of the part k is:

$$Y_1^k = \begin{cases} 1 & \text{if } \sum_{i=1}^{m_1} Y_{1i} = m_1 \\ 0 & \text{if } \sum_{i=1}^{m_1} Y_{1i} < m_1 \end{cases}$$

$$P(Y_1^k = 1) = \prod_{i=1}^{m_1} p_{1i}$$
$$P(Y_1^k = 0) = 1 - \prod_{i=1}^{m_1} p_{1i} \tag{8.1}$$

The analysis of M_1 with multiple sensors integrated into it can be converted to the analysis of M_1 with a single sensor using the above equations. The important random variable for us here to determine is the number of inspections performed by the newly established single-sensor inspection system before passing all sub-inspections. Clearly, for the kth parts interred to the cell, this number can be represented by a random variable X_1^k which is associated with a geometric distribution with parameter $p_1 = \prod_{i=1}^{m_1} p_{1i}$ and the time between two inspections equals a. The reason for this intuition is that the geometric distribution is defined as a discrete distribution counting the number of Bernoulli trials until the first success. The memoryless property implies that the current sub-inspection process is performed independent of the number of completed sub-inspections before.

Lemma 1. Having a RRCSI with free pickup criterion, the partial cycle time $T_{S_1}^k$ for kth implementation of S_1 is:

$$T_{S_1}^k = 6\varepsilon + 6\delta + aX_1^k + b \tag{8.2}$$

Proof: The kth implementation of S_1 includes the tasks below: the robot unloads $(k-1)$th part from M_2 after a full waiting at this machine to finish the processing of the $(k-1)$th part $(b + \varepsilon)$. The robot carries this part from M_2 to O, and then moves backward to I to pick up the kth part and load it on $M_1(3\varepsilon + 5\delta)$. In this stage, the robot undergoes a random waiting depending on the number of rework processes performed by M_1, and eventually it picks up the part k and loads it on M_2 $(aX_1^k + 2\varepsilon + \delta)$. This completes the proof.

Lemma 2. Having a RRCSI with free pickup criterion, the partial cycle time for kth implementation of S_2 is the random variable $T_{S_2}^k$ as:

$$T_{S_2}^k = 6\varepsilon + 8\delta + max\{0, aX_1^k - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\} \tag{8.3}$$

Proof: Here, the robot visits I, M_2, and M_1 in the kth implementation of S_2, respectively. Our goal is to find the time required for all intervening activities which are implemented between two consecutive loadings of M_2. The robot initially moves backward to I to unload the kth part and load it on $M_1(2\varepsilon + 3\delta)$. In the second phase, it returns to M_2 to unload the $(k-1)$th part and transfer it to O after a partial waiting at $M_2(2\varepsilon + 2\delta + w_2^{k-1})$. Likewise, the empty robot returns to M_1 to unload the kth part and transfer it to M_2 after a partial waiting at $M_1(2\varepsilon + 3\delta + w_1^k)$. Thus, $T_{S_2}^k = 6\varepsilon + 8\delta + w_2^{k-1} + w_1^k$, and we have:

$$w_2^{k-1} = max\{0, b - (2\varepsilon + 4\delta)\}$$
$$w_1^k = max\{0, aX_1^k - (2\varepsilon + 4\delta) - w_2^{k-1}\}$$
$$w_2^{k-1} + w_1^k = max\{0, aX_1^k - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\}$$

Recall that w_2^{k-1} and w_1^k are waiting times of the robot, not parts $k-1$ and k. Accordingly, these two variables can be equal to zero since the pickup criterion is free and the completed part can stay indefinitely on the machine to be removed by the robot as soon as it reaches the machine. This completes the proof.

Our task now is to find an optimal long-term production strategy considering Lemmas 1 and 2. Before proceeding with the following theorem, let us notice that it is pickup criterion-independent, and therefore it is applicable for free, interval, and no-wait pickup criteria. The reason behind this intuition is that the dynamic nature of a cell only can change cycle times, not the number of cycles.

Theorem 1. For two-machine robotic rework cells, there is no dynamic state change from S_1 to S_2, and vice versa. In contrast, robotic rework cells with over two machines always have dynamic behaviour.

Proof: In regard with the first segment of this theorem, as mentioned before, the starting point of each one of S_1 and S_2 is the moment when the robot loaded a part on M_2 and waits at M_2 until it receives the next order by the robot controller.

If M_1 was loaded at this given moment, the stochastic modelling was essential due to the fact that the extent to which each part has been processed on M_1 at the starting point of each partial cycle is random. Fortunately, M_1 which has a random number of rework on the part is empty at this common state. Therefore, there is no relationship between the optimal partial cycle in the current state and the number of rework performed on the part in the previous partial cycle.

Regarding the second segment of this theorem, let us initially consider three-machine case. For this case, there in an extra machine M_3 is comparison with two-machine case. Therefore, activity A_3 should be added to the list of activities so that the following tree scheme shows all possible cycles.

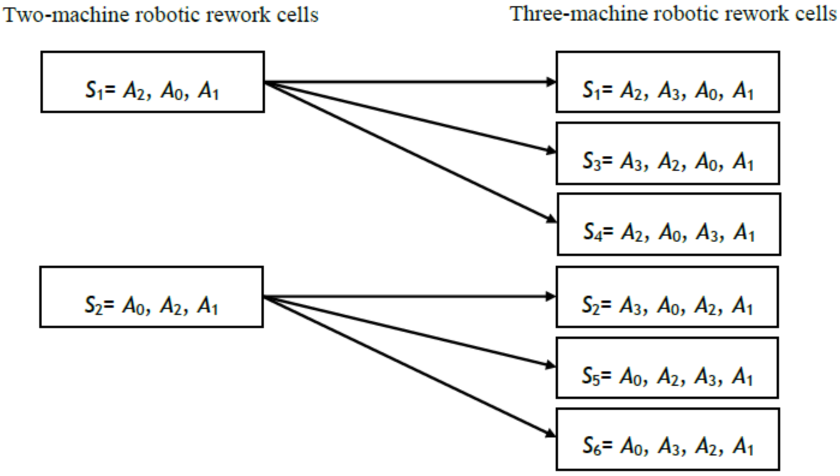


Fig. 8.2. A tree scheme for cycle generation of three-machine robotic rework cells

In Figure 8.2, it is clear that six possible cycles are originated from S_1 and S_2. More specifically cycles S_1, S_3 and S_4 rise from S_1 in two-machine case. Also, cycles S_2, S_5 and S_6 rise from S_2 in two-machine case. It should be emphasized that all S_1, S_3, S_4 and S_5 can reach a state where the robot has just completed loading a part on a machine and the rest of machines are unoccupied. This means that these cycles get a chance to exhibit non-dynamic behaviour. The machine is either M_1, M_2 or M_3 for S_1; the machine is M_3 for S_3; the machine is either M_2 or M_3 for S_4; and finally the machine is M_2 for S_5. However, S_2 and S_6 always have dynamic behaviour since their permutations shows that cell never reach an state in which there is only one part is processing. Therefore, three-machine robotic rework cells always have dynamic behaviour.

An indirect result from the aforementioned tree scheme is that possible cycles of an m-machine robotic rework cell originate from cycles of a rework robotic cell with $m-1$ machines. We only need to take into account activity A_m. Therefore, the

m-machine robotic rework cell will certainly has dynamic behaviour if the robotic rework cell with $m-1$ machines has it. This means that all robotic rework cell with over three machine also have dynamic behaviour. This completes the proof.

This results in two cycles, namely $S_1 = A_2, A_0, A_1$ and $S_2 = A_0, A_2, A_1$. It is necessary to mention that we choose A_1 as the last activity of both cycles since it is the only activity that is able to act as switching point from one cycle to another.

This theorem in particular implies that all k, where $k \to +\infty$, have a similar production behaviour. As a direct result of this, it can be concluded that it is enough to compare a particular implementation of S_1 and S_2, for example the kth implementation of both of them, and then extend the result to a mass production environment with an infinite period of time. To point out this subject more clearly, let us find the dominancy regions of any one of S_1 and S_2 for kth implementation. We must initially recall some concepts related to stochastic dominancy. There are three kinds of stochastic order relations that are sorted through strangeness (Ross, 1996):

1. Absolute Dominance (AD): we say that $T^k_{S_1}$ is absolute dominant over $T^k_{S_2}$, written $T^k_{S_1} \geq T^k_{S_2}$, only if $P(T^k_{S_1} \geq T^k_{S_2}) = 1$ and $T^k_{S_1} > T^k_{S_2}$ be satisfied for at least one $T^k_{S_1}$.
2. First-order Stochastic Dominance (FSD): we say that $T^k_{S_1}$ is first-order dominant over $T^k_{S_2}$, written $T^k_{S_1} \geq_{st} T^k_{S_2}$, only if $P(T^k_{S_2} > \theta) \geq P(T^k_{S_1} > \theta)$ for all θ.
3. Second-order Stochastic Dominance (SSD): if $T^k_{S_1}$ is second-order dominant over $T^k_{S_2}$, then $E(T^k_{S_1})E(T^k_{S_2})$.

Since AD has priority over other dominance relations, we determine the regions of AD of $T^k_{S_1}$ and $T^k_{S_2}$ in the first stage.

Theorem 2. Under a RRCSI, it is absolutely better to instruct the robot for implementation of S_2 in an infinite period of time if $\lceil (2-b)/a \rceil \leq 1$.

Proof: we start with considering $P(T^k_{S_1} \geq T^k_{S_2}) = P(6\varepsilon + 6\delta + aX^k_1 + b \geq 6\varepsilon + 8\delta + max\{0, aX^k_1 - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\}) = P(aX^k_1 + b \geq 2\delta + max\{0, aX^k_1 - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\}) = P(aX^k_1 + b \geq 2\delta) = P(X^k_1 \geq \lceil (2-b)/a \rceil)$. It is necessary to recall that the discrete variable X^k_1 is associated with a geometric distribution and at least equals 1. Therefore, we have $P(T^k_{S_1} \geq T^k_{S_2}) = P(X^k_1 \geq \lceil (2-b)/a \rceil) = 1$ when $\lceil (2-b)/a \rceil \leq 1$. This completes the proof.

Due to the fact that FSD and SSD are our second and third priorities, we determine the regions of the FSD and SSD of $T_{S_1}^k$ and $T_{S_2}^k$, respectively, in the Theorems 3 and 4.

Theorem 3. There is no FSD relationship between $T_{S_1}^k$ and $T_{S_2}^k$ executed for a RRCSI.

Proof: It is enough to prove that the inequality $P(T_{S_1}^k > \theta) \geq P(T_{S_2}^k > \theta)$ is not satisfied for all θ. Obviously, the intersection of $T_{S_1}^k$ and $T_{S_2}^k$ is $6\varepsilon + 6\delta$. Therefore, θ can be replaced with $\lambda = \theta - (6\varepsilon + 6\delta)$, and this yields the following results where the cumulative distribution function $F_{X_1^k}$ of the geometric variable X_1^k is equal to $1 - (1 - p_1)^{x_1^k}$ and also $\alpha = max\{0, b - (2\varepsilon + 4\delta)\}$ for the sake of simplicity.

$$P(T_{S_1}^k > \theta) = P(aX_1^k + b > \lambda) = P(X_1^k > \tfrac{\lambda - b}{a} = 1 - F(\lfloor \tfrac{\lambda - b}{a} \rfloor) = (1 - p_1)^{\lfloor \frac{\lambda - b}{a} \rfloor}$$

$$\begin{aligned}P(T_{S_2}^k > \theta) &= P(2\delta + max\{0, aX_1^k - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\} > \lambda)\\ &= P(\alpha > \lambda - 2\delta | X_1^k \leq \frac{\alpha + 2\varepsilon + 4\delta}{a})P(X_1^k \leq \frac{\alpha + 2\varepsilon + 4\delta}{a})\\ &+P(X_1^k > \frac{\lambda + 2\varepsilon + 2\delta}{a} | X_1^k > \frac{\alpha + 2\varepsilon + 4\delta}{a})P(X_1^k > \frac{\alpha + 2\varepsilon + 4\delta}{a})\end{aligned}$$

None of α, λ, and δ is a random variable. So, any one of two following cases may occur for $P(T_{S_2}^k > \theta)$:

1. $\alpha > \lambda - 2\delta$

$$\rightarrow P(T_{S_2}^k > \theta) = P(X_1^k \leq \frac{\alpha + 2\varepsilon + 4\delta}{a}) + P(X_1^k > \frac{\alpha + 2\varepsilon + 4\delta}{a}) = 1$$

2. $\alpha \leq \lambda - 2\delta$

$$\begin{aligned}\rightarrow P(T_{S_2}^k > \theta) &= P(X_1^k > \frac{\lambda + 2\varepsilon + 2\delta}{a} | X_1^k > \frac{\alpha + 2\varepsilon + 4\delta}{a})P(X_1^k > \frac{\alpha + 2\varepsilon + 4\delta}{a})\\ &= P(X_1^k > \frac{\lambda + 2\varepsilon + 2\delta}{a} \cap X_1^k > \frac{\alpha + 2\varepsilon + 4\delta}{a})\\ &= P(X_1^k > \frac{\lambda + 2\varepsilon + 2\delta}{a}) = 1 - F(\lfloor \frac{\lambda + 2\varepsilon + 2\delta}{a} \rfloor)\\ &= (1 - p_1)^{\lfloor \frac{\lambda + 2\varepsilon + 2\delta}{a} \rfloor}\end{aligned}$$

If the case 1 is taken into consideration, $P(T_{S_1}^k > \theta) \leq P(T_{S_2}^k > \theta)$, whereas $P(T_{S_1}^k > \theta) \geq P(T_{S_2}^k > \theta)$ for the second case. Thus, it is impossible to satisfy inequality $P(T_{S_1}^k > \theta) \geq P(T_{S_2}^k > \theta)$ (or inequality $P(T_{S_1}^k > \theta) \leq P(T_{S_2}^k > \theta)$) for any θ, and this means that there is no FSD relationship between $T_{S_1}^k$ and $T_{S_2}^k$. This completes the proof.

As mentioned before, SSD order relations get through the expected values of $T^k_{S_1}$ and $T^k_{S_2}$. As a result of that, before proceeding with the next theorem, we must calculate $E(T^k_{S_1})$ and $E(T^k_{S_2})$.

Lemma 3. For the case of a RRCSI, the expected values of cycle time of S_1 and S_2 are given by:

$$E(T^k_{S_1}) = 6\varepsilon + 6\delta + \frac{a}{p_1} + b \tag{8.4}$$

$$\begin{aligned} E(T^k_{S_2}) &= max\{6\varepsilon + 8\delta, b + 4\varepsilon + 4\delta\} \\ &+(a\lfloor\tfrac{max\{2\varepsilon+4\delta,b\}}{a}\rfloor + \frac{a}{p_1} - max\{2\varepsilon+4\delta, b\})(1-p_1)^{\lfloor\frac{max\{2\varepsilon+4\delta,b\}}{a}\rfloor} \end{aligned} \tag{8.5}$$

Proof: The fact that X^k_1 is associated with a geometric distribution with parameter $p_1 = \prod_{i=1}^{m_1} p_{1i}$ implies that $(T^k_{S_1}) = E(6\varepsilon + 6\delta + aX^k_1 + b) = E(6\varepsilon + 6\delta + b) + aE(X^k_1) = 6\varepsilon + 6\delta + b + \frac{a}{p_1}$. Regarding $E(T^k_{S_2})$, it contains a triple-sided max term where 0 and $b - (2\varepsilon + 4\delta)$ are fixed and $aX^k_1 - (2\varepsilon + 4\delta)$ is variable. For simplicity, hereinafter $\beta = 4\varepsilon + 4\delta$ and $\gamma = max\{2\varepsilon + 4\delta, b\}$, and therefore we have the calculation of the conditional expected value below:

$$\begin{aligned} E(T^k_{S_2}) &= E(6\varepsilon + 8\delta + max\{0, aX^k_1 - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\}) \\ &= \beta + E(max\{\gamma, aX^k_1\}) \\ &= \beta + E(max\{\gamma, aX^k_1\}|X^k_1 \leq \lfloor\frac{\gamma}{a}\rfloor)P(X^k_1 \leq \lfloor\frac{\gamma}{a}\rfloor) \\ &+E(max\{\gamma, aX^k_1\}|X^k_1 > \lfloor\frac{\gamma}{a}\rfloor)P(X^k_1 > \lfloor\frac{\gamma}{a}\rfloor) \\ &= \beta + \gamma F(\lfloor\frac{\gamma}{a}\rfloor) + a(\lfloor\frac{\gamma}{a}\rfloor + \frac{1}{p_1})(1 - F(\lfloor\frac{\gamma}{a}\rfloor)) \\ &= \beta + \gamma(1 - (1 - p_1)^{\lfloor\frac{\gamma}{a}\rfloor}) + a(\lfloor\frac{\gamma}{a}\rfloor + \frac{1}{p_1})(1 - p_1)^{\lfloor\frac{\gamma}{a}\rfloor}) \\ &= \beta + \gamma + (a\lfloor\frac{\gamma}{a}\rfloor + \frac{a}{p_1} - \gamma)(1 - p_1)^{\lfloor\frac{\gamma}{a}\rfloor} \\ &\leftrightarrow E(T^k_{S_2}) = max\{6\varepsilon + 8\delta, b + 4\varepsilon + 4\delta\} + \\ &(a\lfloor\frac{max\{2\varepsilon + 4\delta, b\}}{a}\rfloor + \frac{a}{p_1} - max\{2\varepsilon + 4\delta, b\})(1 - p_1)^{\lfloor\frac{max\{2\varepsilon+4\delta,b\}}{a}\rfloor} \end{aligned}$$

This completes the proof.

Theorem 4. If $b + \frac{a}{p_1}(1 - (1 - p_1)^{\lfloor\gamma/a\rfloor}) \leq 2\delta$ in a RRCSI, then $T^k_{S_2}$ is second-order larger than $T^k_{S_1}$; else if $b + \frac{a}{p_1}(1 - (1 - p_1)^{\lfloor\gamma/a\rfloor})) > 2\delta$, then $T^k_{S_2}$ is second-order smaller than $T^k_{S_1}$.

Proof: The proof will be presented in a structure similar to that of Theorem 3: we use Equations (8.4) and (8.5) and initially consider the case in which $E(T^k_{S_2}) \geq E(T^k_{S_1})$, and then extend the proof for the case in which $E(T^k_{S_2}) < E(T^k_{S_1})$. If we relax the first bracket in $E(T^k_{S_2})$, then we have:

1. $E(T^k_{S_1}) \leq E(T^k_{S_2})$

$$\rightarrow 2\varepsilon + 2\delta + b + \frac{a}{p_1} \leq max\{2\varepsilon + 4\delta, b\} + \frac{a}{p_1}(1-p_1)^{\lfloor \frac{\gamma}{a} \rfloor}$$

$$\rightarrow b + \frac{a}{p_1}(1-(1-p_1)^{\lfloor \frac{\gamma}{a} \rfloor}) \leq 2\delta$$

2. $E(T^k_{S_1}) > E(T^k_{S_2})$

$$\rightarrow 2\varepsilon + 2\delta + b + \frac{a}{p_1} > max\{2\varepsilon + 4\delta, b\} + \frac{a}{p_1}(1-p_1)^{\lfloor \frac{\gamma}{a} \rfloor}$$

$$\rightarrow b + \frac{a}{p_1}(1-(1-p_1)^{\lfloor \frac{\gamma}{a} \rfloor}) > 2\delta$$

This completes the proof.

At the end of this section, it should be emphasized that Theorems 1 to 4 give an appropriate structure to select the robot's partial cycle with the maximum expected throughput of RRCSIs, and Lemma 3 helps us to calculate this maximum expected throughput. Clearly, this structure assists industry in both designing and developing basic rework cells.

8.4 Scheduling of RRCEIs with Free Pickup Criterion

For the RRCEI case, the final parts processed on M_2 are monitored to detect the presence of different types of defects before delivering them to the customers. Similar to RRCSIs, there is no difficulty with converting the multi-sensor system into a single-sensor system in RRCEIs. Each sensor $j, \forall j \in \{1, 2, ..., m_2\}$, identifies no defect in each time inspection with probability p_{2j}. This builds up a sub-inspection set of m_2 different sensors, and the sequence of random variables $[Y^k_{21}, Y^k_{22}, , Y^k_{2m_2}]$ that represent the inspection results of m_2 sensors follow Bernoulli probability distribution. Following that, the generalized Bernoulli distributed variable Y^k_2 supporting the success of the multi-sensor inspection system of the part

k is expressed as:

$$Y_2^k = \begin{cases} 1 & \text{if } \sum_{j=1}^{m_2} Y_{2j} = m_2 \\ 0 & \text{if } \sum_{j=1}^{m_2} Y_{2j} < m_2 \end{cases}$$

$$P(Y_2^k = 1) = \prod_{j=1}^{m_2} p_{2j}$$
$$P(Y_2^k = 0) = 1 - \prod_{j=1}^{m_2} p_{2j} \tag{8.6}$$

According to Equation (8.6), the number of inspection of the kth parts interred to the rework cell performed by the single-sensor inspection system is the random variable X_2^k which is associated with a geometric distribution with success parameter $p_2 = \prod_{j=1}^{m_2} p_{2j}$ and the time between two inspections equals b. The reason for this intuition is that the geometric distribution is defined as a discrete distribution counting the number of Bernoulli trials until the first success. The partial cycle times for kth implementation of S_1 and S_2 are presented in the following lemma, respectively.

Lemma 4. Having a RRCEI, the partial cycle times for kth implementations of S_1 and S_2 are:

$$T_{S_1}^k = 6\varepsilon + 6\delta + a + bX_2^k \tag{8.7}$$
$$T_{S_2}^k = 6\varepsilon + 8\delta + max\{0, a - (2\varepsilon + 4\delta), bX_2^k - (2\varepsilon + 4\delta)\} \tag{8.8}$$

Proof: If we follow the order of tasks performed in Lemmas 1 and 2, once again, we achieve the desired results. The only difference here is that the processing time of M_2 is the random variable, not M_1. The rest of this proof is easy and therefore omitted. This completes the proof.

Our task now is to find an optimal long-term production strategy considering Lemmas 4.

Corollary 1. Theorem 1 to 4 are also correct for RRCEIs if a and b be swapped with each other in all associated inequalities.

Proof: Easy and omitted.

Corollary 2. For the case of a RRCEI, the expected values of partial cycle time of S_1 and S_2 are given by:

$$E(T_{S_1}^k) = 6\varepsilon + 6\delta + a + \frac{b}{p_2} \tag{8.9}$$

$$\begin{aligned}E(T^k_{S_2}) &= max\{6\varepsilon + 8\delta, a + 4\varepsilon + 4\delta\}\\ &+(b\lfloor \tfrac{max\{2\varepsilon+4\delta,a\}}{b} \rfloor + \frac{b}{p_2} - max\{2\varepsilon+4\delta, a\})(1-p_2)^{\lfloor \frac{max\{2\varepsilon+4\delta,a\}}{b} \rfloor}\end{aligned} \tag{8.10}$$

Proof: Easy and omitted.

It is worth noting that the result of this section along with the previous section create a parallel mechanism for analysing both RRCSIs and RRCEIs with free pickup criteria. Actually, we must make it clear that the main purpose of inspection is to control the quality of parts at the early stage of production to decrease production complexity, or control the quality of parts through a final inspection at the last stage. According to the first or the second priorities, a RRCSI or RRCEI with free pickup criteria can be designed, respectively, and the result of Sections 8.3 and 8.4 can be applied for optimizing the performance. However, we still need to extend results of these sections for cases in which the pickup criterion is either interval or no-wait.

8.5 Analysis of Interval and No-Wait Pickup criteria

We begin with generalizing results to the case of interval pickup criterion and then provide a set of guidance notes to determine the optimality region of S_1 and S_2 for rework cells with no-wait pickup criterion. Let us first provide a more precise definition of the interval pickup criterion. A wide variety of robotic rework cells, in particular those used in steel, chemical and plastic industries, work under interval pickup criterion. For these cells, we keep the temperature of the part within a fixed range after completing any particular process, and hence, the waiting time of the part at the machine must be bounded (Paul et al., 2007). Otherwise, the part will be certainly scrapped because it overwaits for the robot to unload it. Under this condition, it seems more economical to impose waiting time's restrictions in order to have no scrap in the production line, especially for expensive parts.

Theorem 5. Theorems 2, 3, and 4 (Corollary 1) hold(s) for a RRCSI (a RRCEI) with interval pickup criterion if and only if $a + \overline{a} \geq 2\varepsilon + 4\delta$ and $b + \overline{b} \geq 2\varepsilon + 4\delta$. Otherwise, S_1 is the optimal cycle of rework cells under interval pickup criterion.

Proof: The definition of interval pickup criterion makes it clear that it is a special case of the free pickup criterion in which neither M_1 nor M_2 can keep the part for an infinite time ($\overline{a} \neq \infty$ and $\overline{b} \neq \infty$). This means that converting the pickup criterion into interval has no impact on the optimality region of S_1 and

S_2. Nonetheless, we should check whether these cycles are always feasible. For S_1, the robot loads the part to the machine (M_1 or M_2) and waits in front of it throughout that the part processing. Therefore, this cycle is always feasible (and also the optimal cycle if S_2 be infeasible). In contrast, it is possible that S_2 be an infeasible cycle if $a + \overline{a} < 2\varepsilon + 4\delta$ or $b + \overline{b} < 2\varepsilon + 4\delta$. More particularly, the proof of Equations (8.3) shows that the total waiting time of the robot in a RRCSI is $max\{0, aX_1^k - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\}$. In this max term, $max\{0, (2\varepsilon + 4\delta) - aX_1^k\}$ and $max\{0, (2\varepsilon + 4\delta) - b\}$ are waiting times of the part on M_1 and M_2 since they are the reciprocal of the waiting time of the robot on machines. This yields the result:

$$\overline{a} \geq max\{0, (2\varepsilon + 4\delta) - aX_1^k\} \rightarrow aX_1^k + \overline{a} \geq 2\varepsilon + 4\delta \rightarrow a + \overline{a} \geq 2\varepsilon + 4\delta$$
$$\overline{b} \geq max\{0, (2\varepsilon + 4\delta) - b\} \rightarrow b + \overline{b} \geq 2\varepsilon + 4\delta$$

Also, considering the proof of Equations (8.8) gives us same results for a RRCEI. Accordingly, Theorems 2, 3, 4 and Corollary 1, which are related to the optimality conditions of S_1 and S_2 hold. This completes the proof.

The results of analysing the rework cell with interval pickup criterion is even extendable for more specific pickup criteria. One of these well-solved pickup criteria originating from interval pickup criterion is no-wait pickup criterion. Within a manufacturing context, the no-wait means that the robot is instructed to unload the part immediately after completing its operation on the machine (Che and Chu, 2005). Subsequently, we can consider it as a special case of the interval pickup criterion in which $\overline{a} = 0$ and $\overline{b} = 0$.

Corollary 3. The cycle times of S_2 for RRCSIs and RRCEIs with no-wait pickup criterion are $4\varepsilon + 4\delta + max\{aX_1^k, b\}$ and $4\varepsilon + 4\delta + max\{a, bX_2^k\}$, respectively.

Proof: Typically, the issues involved in no-wait pick up is that the waiting time of the robot in front of that machine is not zero, or equivalently the waiting time of the part at any particular machine is zero. This implies that $aX_1^k \geq (2\varepsilon + 4\delta)$ and $b \geq (2\varepsilon + 4\delta)$ for Equations (8.3), and therefore $max\{0, a - (2\varepsilon + 4\delta), bX_2^k - (2\varepsilon + 4\delta)\} = max\{a - (2\varepsilon + 4\delta), bX_2^k - (2\varepsilon + 4\delta)\}$. This is enough to prove the first part of this corollary. The proof of the second part of this corollary is completely analogous to above derivation and therefore it is omitted. This completes the proof.

Theorem 6. S_2 is the optimal cycle for both two-machine RRCSIs and RRCEIs if it be feasible. Otherwise, S_1 is the optimal cycle of rework cells under no-wait pickup criterion.

Proof: It is enough to prove that the inequality $T^k_{S_1} \geq T^k_{S_2}$ is always satisfied if S_2 is feasible. Therefore, considering Corollary 3, we have:

1. For RRCSIs:
   $6\varepsilon + 6\delta + aX^k_1 + b \geq 4\varepsilon + 4\delta + max\{aX^k_1, b\} \rightarrow 2\varepsilon + 2\delta + aX^k_1 + b \geq max\{aX^k_1, b\}$
2. For RRCEIs:
   $6\varepsilon + 6\delta + a + bX^k_2 \geq 4\varepsilon + 4\delta + max\{a, bX^k_2\} \rightarrow 2\varepsilon + 2\delta + a + bX^k_2 \geq max\{a, bX^k_2\}$

, which are always satisfied. This completes the proof.

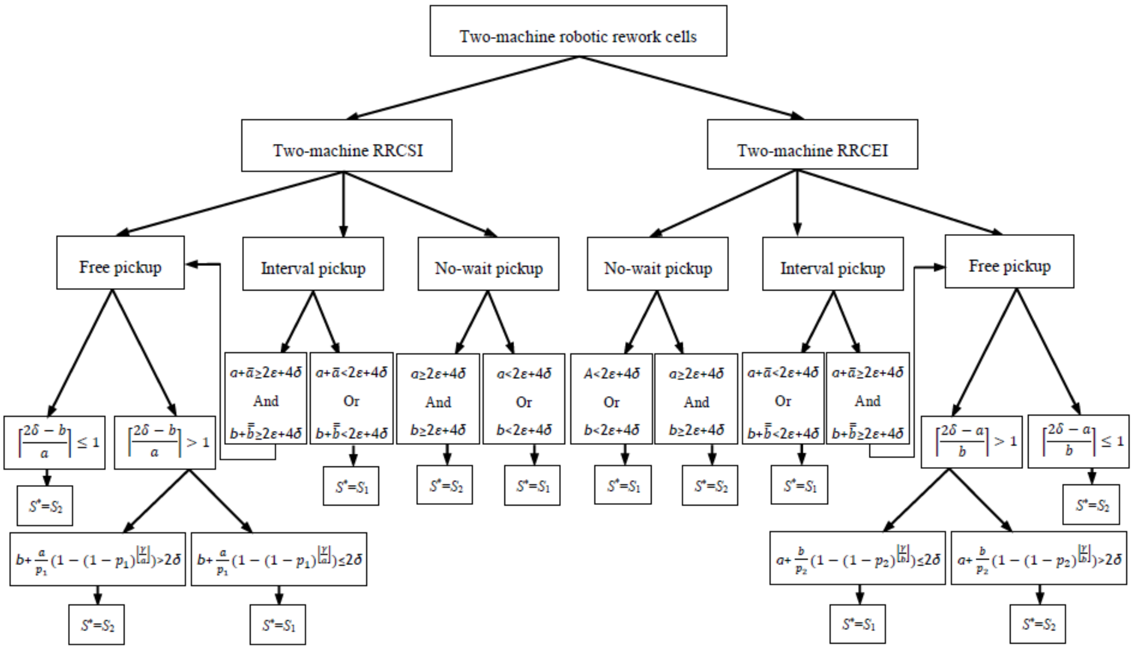

Two-machine robotic rework cells
Two-machine RRCSI
Two-machine RRCEI
Free pickup
Interval pickup
No-wait pickup
No-wait pickup
Interval pickup
Free pickup
$\left\lceil \frac{2\delta - b}{a} \right\rceil \leq 1$
$\left\lceil \frac{2\delta - b}{a} \right\rceil > 1$
$a+\bar{a} \geq 2\varepsilon+4\delta$ And $b+\bar{\bar{b}} \geq 2\varepsilon+4\delta$
$a+\bar{a} < 2\varepsilon+4\delta$ Or $b+\bar{\bar{b}} < 2\varepsilon+4\delta$
$a \geq 2\varepsilon+4\delta$ And $b \geq 2\varepsilon+4\delta$
$a < 2\varepsilon+4\delta$ Or $b < 2\varepsilon+4\delta$
$A < 2\varepsilon+4\delta$ Or $b < 2\varepsilon+4\delta$
$a \geq 2\varepsilon+4\delta$ And $b \geq 2\varepsilon+4\delta$
$a+\bar{a} < 2\varepsilon+4\delta$ Or $b+\bar{\bar{b}} < 2\varepsilon+4\delta$
$a+\bar{a} \geq 2\varepsilon+4\delta$ And $b+\bar{\bar{b}} \geq 2\varepsilon+4\delta$
$\left\lceil \frac{2\delta - a}{b} \right\rceil > 1$
$\left\lceil \frac{2\delta - a}{b} \right\rceil \leq 1$
$S^*=S_2$
$S^*=S_1$
$S^*=S_2$
$S^*=S_1$
$S^*=S_1$
$S^*=S_2$
$S^*=S_1$
$S^*=S_2$
$b+\frac{a}{p_1}(1-(1-p_1)^{\lfloor \frac{\gamma}{a} \rfloor}) > 2\delta$
$b+\frac{a}{p_1}(1-(1-p_1)^{\lfloor \frac{\gamma}{a} \rfloor}) \leq 2\delta$
$a+\frac{b}{p_2}(1-(1-p_2)^{\lfloor \frac{\gamma}{b} \rfloor}) \leq 2\delta$
$a+\frac{b}{p_2}(1-(1-p_2)^{\lfloor \frac{\gamma}{b} \rfloor}) > 2\delta$
$S^*=S_2$
$S^*=S_1$
$S^*=S_1$
$S^*=S_2$

Fig. 8.3. A summary of the results of robotic rework cell scheduling problems

Figure 8.3 also summarizes the feasibility and optimality results obtained in this research. This figure shows an integrated framework for both start of line and end of line inspections in small-scale robotic rework cells with three common pickup criteria. This framework makes a significant contribution to real-life applications of industrial automation, and it assists manufacturers in designing automated inspection cells. In more detail, it helps companies to remain competitive since suggested cycles in this framework minimize the cycle time for the first time in the literature related to robotic cells with inspection process.

8.6 Concluding Remarks

Some of the small-scale robotic cells, especially two-machine ones, are still used in automated manufacturing systems. The analysis of these cells is not an easy task if stochastic variables such as inspection processes through a multiple-sensor inspection system are taken into account. Therefore, an analytical method for minimizing the partial cycle time of such cells has been developed in this study. Regardless of the pickup criterion, we have proven that it is possible to reach a steady state of these cells which have a dynamic behaviour, and then maximize the expected throughput of associated cells. Comparing the two-machine robotic rework cell under free pickup criterion with the same robotic cell without a rework assumption, it has been realized that the performance of the partial cycle S_2 is improved due to the fact that the average time of producing a part is definitely increased. With regard to the interval pickup criterion, we conclude there is no guarantee that S_2 (in comparison with S_1) will be an optimal cycle when it is feasible. Nonetheless, it is enough to check whether S_2 satisfies feasibility conditions for rework cells with interval and no-wait pickup criteria to conclude that it is the optimal cycle. Further work should be done to consider the limited number of permitted rework processes for a particular part. If the part could not pass the inspection process even after this number of rework processes, it must be considered as scrap, not as a final product. Clearly, this assumption makes the analysis of the system more complex.

Chapter 9 is based on the published article Foumani, M., Smith-Miles, K., Gunawan, I., 2016. Scheduling of Two-machine Robotic Rework Cells: In-process, Post-process and In-line Inspection Scenarios. Omega, Submitted in April 2016.

Abstract *This study focuses on the domain of a two-machine robotic cell scheduling problem under three inspection scenarios. We propose a method for minimizing the partial cycle time of cells with in-process and post-process inspection scenarios, and then we convert this cell into a multi-function cell with in-line inspection scenario. For the first scenario, parts are inspected in one of the production machines using multiple sensors, while the inspection process is performed by an independent inspection machine for the second scenario. Alternatively, the inspection can be performed by a multi-function robot for the third scenario. A distinguishing characteristic of this robot is that it can perform inspection of the part in transit. However, the robot cannot complete the part transition and load it on the next destination machine if it identifies a fault in the part. The stochastic nature of the process prevents us from applying deterministic methods for corresponding problems. In the first stage, we present a method that converts a multi-sensor system into a single-sensor system. The expected cycle times of two different cycles are derived based on a geometric distribution, and then the maximum expected throughput is pursued for in-process and post-process inspection sensors, respectively. In the second stage, we develop the inspection system into an in-line system using a multi-function robot. Finally, we determine if it is technically profitable to replace the inspection scenario with the in-line inspection scenario.*

Note to Practitioners *Stochastic modelling of robotic cells are essential for robotic manufacturers who seek methods to increase the productivity of robotized systems with all different types of measuring systems. This study provides support for the use of every inspection system by robotic cell designers. A real-life example of this environment is a robotic arm equipped with a Grip-Gage-Go gripper to measure the thickness of the shaft in transit. The measuring heads are integrated into the automation by adding gages and crankshaft locating features to the arm.*

Keywords *Scheduling, Inspection, Multi-function, Robotic cell*

Classification

$SRF_{2,2,1}^{1,1,1}|$*free, additive , stochastic , identical, cyclic*$|T$ with post-process test
$SRF_{2,2,1}^{1,1,1}|$*free, additive , stochastic , identical, cyclic*$|T$ with in-process test
$SRF_{2,2,1}^{1,1,1}|$*free, additive , stochastic , identical, cyclic*$|T$ with in-line test

Note References are considered at the end of the thesis.

Chapter 9

Scheduling of Two-machine Robotic Rework Cells: In-process, Post-process and In-line Inspection Scenarios

9.1 Introduction

The Cellular Manufacturing System (CMS) has been identified as a critical part of the manufacturing and service industries. CMS considerably decreases cycle time, work in process, and manufacturing cost. The manufacturing cell is the smallest unit of a CMS where the level of automation of the material handling system has a direct relationship with the level of automation of the cell (Safaei and Tavakkoli-Moghaddam, 2009). For instance, the material handling system must be robotized and processing time kept deterministic for the cell to be fully automated. This kind of cell is generally called a robotic cell because a robot is specifically responsible for transporting parts. Robotic cells are one of the complicated application areas of flow-shops that have received a considerable amount of attention in robot move sequencing (Brauner, 2008). The main reason for this is that the CMS requires a huge expenditure on material handling devices such as industrial robots (Pinedo, 2009).

Robotic cells are basically classified into two categories: the robotic cells without rework assumption and robotic rework cells. The term "rework" here means that a processed part may need reprocessing. Therefore, it is cycled between test and processing stations until deemed acceptable (Geren and Redford, 1999). These two cases can also be defined as "deterministic cells" and "stochastic cells" respectively. It is straightforward to find a deterministic model for the robotic cells without rework assumption. Following that, there are many studies in the literature dealing with the scheduling of the robot activities, as widely addressed in (Agnetis, 2000; Dawande et al., 2005; Batur et al., 2012; Zarandi et al., 2013) for different small-scale cells. Nonetheless, inspection and rework stages in a robotic cell is one of the important issues in the field of robotic cell scheduling which reflects most real-life cases. This paper addresses the stochastic issues that arise when considering inspection and rework stages, laying some important analytical foundations for this under-studied problem.

A robotic cell with an additional inspection process in one of the rework stages is called a robotic rework cell. A two-machine robotic rework cell which is the smallest possible robotic rework cell is commonly captured by the following framework: the cell is made up of two production machines M_1 and M_2, a gantry robot that serves the entire production line, an input conveyor (I or dummy machine M_0) and an output conveyor (O or dummy machine M_3) with unlimited storage capacity. The characteristic of the robotic rework cell which distinguishes it from other types of robotic cells is the mechanism used to perform the inspection of a part: either by a group of contact/non-contact sensors installed into a production machine; an independent inspection machine; or via robot grippers. Considering these three alternatives, the robotic rework cell can follow any one of the following inspection scenarios:

- *In-process inspection scenario* where multiple contact/non-contact sensors are installed into M_1 or M_2.
- *Post-process inspection scenario* where multiple contact/non-contact sensors are installed into an independent inspection machine located after M_1 or M_2.
- *In-line inspection scenario* where a couple of measuring heads are integrated into the automation by adding gages and part locating features to the robot end effector.

These scenarios make it clear that typical robotic cells are a special case of the robotic rework cells in which neither machines nor the robot gripper perform inspection, and therefore all produced parts must be failure-free. As shown in Figure 9.1, two-machine robotic rework cells are classified into two groups based on the part inspection route: two-machine robotic rework cells with start of line inspection (2RRCSI) and two-machine robotic rework cells with end of line inspection (2RRCEI) (Colledani et al., 2014). Following the 2RRCSI, it is possible that the raw parts have poor quality. As a consequence, the following solutions give a guarantee that all finished parts have high quality: 1) installing multiple contact/non-contact inspection sensors into M_1, 2) installing multiple contact/non-contact inspection sensors into an independent inspection machine after M_1, 3) installing multiple contact/non-contact inspection sensors into a multi-function robot performing the inspection of the part during its transition between M_1 and M_2. In contrast with this plan of quality control which is the so-called "Start of Line" testing, there is a plan namely "End of Line" testing that only concentrates on final products and therefore is applicable for 2RRCEIs. This testing plan is executed when all manufacturing stages have been completed and the part only needs a final approval by: 1) sensors installed into M_2, 2) the inspection machine after M_2, or 3) the multi-function robot during the part transition between M_2 and O. Then, it can be packed for the customers.

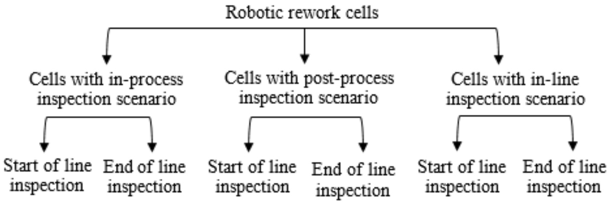


Fig. 9.1. The classification scheme of robotic rework cells

An example of 2RRCEIs with in-process inspection scenario is shown in Figure 9.2 for the crankshaft production lines (Ayub et al., 2014). A particular crankshaft being processed goes through I, the lathe machine M_1, the lathe machine M_2 and O under this part processing route. After loading the crankshaft to any one of the lathe machines, the robot either waits for the crankshaft to finish its operation or immediately moves to another occupied lathe machine or I for unloading a new crankshaft. The significant difference between two lathe machines is that the crankshaft is failure-free when it is processing on M_1, whereas the crankshaft may fail and need rework when it is processing on M_2. As a consequence, there is an urgent need for in-process inspection of the part on M_2.

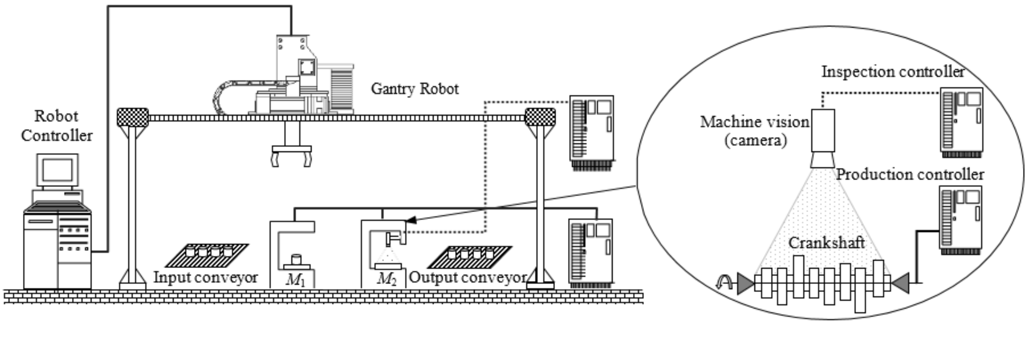


Fig. 9.2. A two-machine robotic rework cell with end of line testing plan

There are a number of issues such as kinematics of the robot, process and system design, inspection types, and part processing routes that must be addressed in order to optimize a two-machine robotic rework cell. Specifically, the stochastic analysis of cyclic schedules for two-machine robotic rework cells is a key issue considering in this study. The analysis of stochastic robotized cells has a fragmented history of development in the literature. Some studies of stochastic robotic cells concentrated on two-machine cells which operate under a production system with machine failures and repairs, whereas some others touched on the robotic cells with stochastic processing times. For the former, a Markovian stochastic approach was established by Savsar and Aldaihani (2008) in order to evaluate the utilization and throughput rate of two-machine robotic cells with stochastic occurrence of production machine failures and repairs. In a similar study, two-machine robotic cells with machine failure considerations were analyzed in two phases by Tysz and Kahraman (2010). They obtained a Stochastic Petri Net for these cells and then calculated corresponding steady-state probabilities for the reachable markings. Also, for the latter, Aldaihani and Savsar (2005) were among the first to assume the machine's processing time as a random quantity following an exponential distribution. They presented exact numerical solutions to minimize total manufacturing cost for a two-machine robotic cell. Following that, the cycle time in a virtually arranged two-machine robotic cell with normally distributed processing time elements was analyzed by Shafiei-Monfared et al. (2009). Finally, Geismar and Pinedo (2010) provided an on-line scheduling scheme to determine the expected partial cycle time of a two-machine robotic cell where only one of the machines has a stochastic processing time with normal distribution. To our best knowledge, there is a lack of research on the stochastic part processing route although it is applicable to many practice-oriented scenarios. Note that the stochastic part processing route is actually possible when a set of sensors is integrated into at least one of the machines or the attached to the robot to find defects in the manufactured parts.

The overall structure of this paper is as follows. Section 9.2 contains fundamental concepts related to robotic rework cells. Section 9.3 is devoted to the scheduling problem of two-machine robotic rework cells with in-process inspection scenario. We initially show how the analysis of a machine with multiple sensors integrated into it can be converted to the analysis of the machine with a single sensor. Following that, all expected completion times under "Start of Line" and "End of Line" testing plans are developed for various feasible solutions, and then the expected throughput is maximized for any one of scenarios. Likewise, the post-process inspection scenario is discussed in Section 9.4. Furthermore, a policy to prevent deadlock problems originating from the inspection scenario is presented in this section. Section 9.5 is dedicated to the scheduling problem of two-machine robotic rework cells with in-line inspection scenario to cover both "Start of Line" and "End of Line" testing plans. Section 9.6 is the heart of this study in which the obtained results of Sections 9.3, 9.4 and 9.5 are used for a comparison between performances of three feasible inspection scenarios. Finally, Section 9.7 concludes the paper with some perspectives and discussion of further research.

9.2 Related Research

We begin this section by first giving a precise definition of each problem class, and then summarizing the notations used in robotic rework cells to find the expected partial cycle times. An effective strategy for scheduling of a robotic cell with a complex part processing route is to consider it as a collection of smaller clusters (Chan et al., 2008). It should be emphasized that it is even possible for a complex robotic rework cell to be converted into a chain of two-machine robotic rework cells. Figure 9.3 shows an example of converting a 10-machine production line into five independent two-machine rework cells after increasing the number of intermediate conveyors and robots.

Another real-world example of converting a complex robotic rework cell into a group of integrated two-machine robotic rework cells is given in Osakada et al. (2011). This example is related to press lines for draw-forming of automobile body panels for Honda (with Aida Engineering), Toyota (with Komatsu) and BMW (with Schuler), where the space between press machines were compact enough only for accommodating material handling robots. The lines include a set of press machines for automobile panels, intermediate buffers, multiple computer-controlled robots, an input conveyor and an output conveyor.

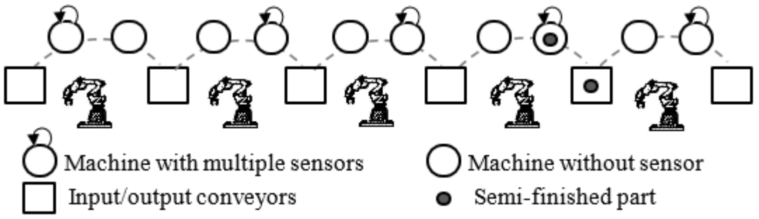


Fig. 9.3. A clustering scheme for five independent two-machine cells

The concept of the robot activity is one of the most important tools to consider a cyclic formulation of the robot movement. In a two-machine robotic rework cell with the inspection scenario $i, \forall i \in \{1, 2, 3\}$, the robot activity denoted as A_{ij} corresponds to the robot starting this activity from machine $j, \forall j \in \{0, 1, 2\}$, with the following three-phase sequence of actions:

1. The robot takes a part from the production machine M_j.

2. The robot immediately carries the part to the production machine M_{j+1}.
3. The robot immediately completes the activity A_{ij} by loading the part onto the production machine M_{j+1}.

A_{ij} is also subject to fluctuation based on the inspection scenarios as follows:

(a) For the in-process inspection scenario, i=1, the first phase contains an additional segment: if a group of sensors is installed into M_j, then the robot takes the part from the production machine M_j after identifying no fault in the part.

(b) For the post-process inspection scenario, i=2, the additional segment of the second phase is: if an independent inspection machine is located after M_j, then the robot initially carries the part to this inspection machine. The part is cycled between the inspection machine and M_j until deemed acceptable and finally carried to the production machine M_{j+1}.

(c) For the in-line inspection scenario, i=3, the second phase contains an additional segment: Assume a measuring head is integrated with the robot arm and the part needs inspection after unloading it from M_j. Then, the second phase is finished not only when the robot carries the part to M_{j+1} but also when the measuring head identifies no fault in the part.

Consistent with Brauner (2008), a particular n-unit cycle can be characterised by a permutation of activities in which any activity is repeated exactly n times. Since A_{i2} is one of the activities with n repetitions, we can conclude that an n-unit cycle is able to produce n final products. Therefore, the cycle time is per unit cycle time or the time to complete one part in a cyclic behavior. An important point about the permutation is that the rework cell must return to the initial state at the starting of the permutation after completing it. In addition, the scheduling of two-machine robotic rework cells must be constrained to the deadlock-free subregion of permutations. The deadlock-free schedule can be guaranteed only if the following activities are executed: The receiving server (the robot, M_1 or M_2) and sending server (the robot, M_1 or M_2) must be empty and loaded before the load/unload process, respectively (Crama et al., 2000). Note that some additional constraints may be considered for any one of the three inspection scenarios. For example, it is forbidden to unload the machine when its current part needs rework for an in-process inspection scenario.

The simplest case of n-unit cycles is one-unit cycles where exactly one completed part leaves from the two-machine robotic rework cell after the cycle's execution. The sequence of performing activities has two alternatives. Let us fix A_{i1} in the last position of any particular cycle of inspection scenario i, so the order of other activities can be either A_{i2}, A_{i0} or A_{i0}, A_{i2}. This results in two cycles, namely the forward cycle $S_{i1}^{hk} = A_{i2}, A_{i0}, A_{i1}$ and the backward cycle $S_{i2}^{hk} = A_{i0}, A_{i2}, A_{i1}, \forall i \in \{1,2,3\}, \forall h \in \{1,2\}$ and $\forall k \in \{0,1,2\}$. Note that i represents three inspection scenarios again, and h represents "Start of Line" and "End of Line" testing plans. k is zero for both in-process and post-process inspection scenarios. It is only applicable for the in-line inspection scenario and consequently it will be specified later on. It is also necessary to mention that we choose A_{i1} as the last activity of both cycles since it is the only activity that is able to act as switching point from one cycle to another. In other words, for any particular i, h, and k, the only common state between S_{i1}^{hk} and S_{i2}^{hk} is exactly the moment when the robot has loaded a part on M_2 and waits at M_2 until it receives the next order by the robot controller. The key note here is that any n-unit cycle is actually a combined sequence repeating S_{i1}^{hk} exactly q times and also S_{i2}^{hk} exactly $n-q$ times in each iteration of the cycle. Therefore, it is expected that per unit partial cycle time is a convex combination of the expected partial cycle time of S_{i1}^{hk} and S_{i2}^{hk} as two given corner points. This means that the expected partial cycle time of one of cycles S_{i1}^{hk} or S_{i2}^{hk} always dominates the expected per unit partial cycle time of any given n-unit cycle. As a result of this, it is enough to limit our search to one-unit cycles, S_{i1}^{hk} and S_{i2}^{hk} for any particular i, h, and k (Sethi et al., 1992). Now, let us extend the notations from Foumani et al. (2015b) for the problem in this paper:

γ_1 The inspection time of the part on M_1, the inspection machine located after M_1, or the robot gripper in transit between M_1 and M_2, depending on the inspection scenario used for

γ_2 The inspection time of the part on M_2, the inspection machine located after M_2, or the robot gripper in transit between M_2 and O

ε The load/unload time of machines (or conveyors) by the robot

δ The required time for traveling between adjacent location pairs

m_1 The number of sensors needed to inspect the part processed on the first machine

m_2 The number of sensors needed to inspect the part processed on the second machine

a The processing time of the first machine

b The processing time of the second machine

$T^l_{S^{hk}_{ij}}$ The partial cycle time of S^{hk}_{ij} for the lth identical part fed to the rework cell, where $l \in \mathbb{N}$

T^*_{ihk} The optimal partial cycle time considering the inspection scenario i, testing plan h, and inspection strategy k

w^l_1 The robot's waiting time at M_1 for the lth identical part fed to the rework cell, where $l \in \mathbb{N}$

w^l_2 The robot's waiting time at M_2 for the lth identical part fed to the rework cell, where $l \in \mathbb{N}$

As a result of the dynamic nature of the robotic rework cell, we cannot be assured that it operates in steady state. The variable $C(A_{i1}, l)$ is defined as the completion time of the lth implementation of A_{i1} to explain why there is no guarantee of cyclic behaviour of the robotic rework cell (Crama and van de Klundert, 1997). Obviously, $C(A_{i1}, l) - C(A_{i1}, l-1)$ does not remain a constant value for $\forall l \in \mathbb{N}$. Therefore, it is vital to carry out a separate analysis for any $C(A_{i1}, l) - C(A_{i1}, l-1)$, which is named the partial cycle time $T^l_{S^{hk}_{ij}}$ for the lth implementation of an arbitrary robot move sequence S^{hk}_{ij}. The main aim of the following section is to obtain the most appropriate robot move sequence that gives us the minimum expected partial cycle time or in other words, the maximum expected partial throughput for a robotic rework cell subject to aforementioned deadlock-related constraints.

9.3 Sequencing of Activities under In-Process Inspection Scenario

In-process inspection refers to the various types of inspection scenarios in which a part processing on the machine is monitored by a multi-sensor system to detect the presence of different types of defects. Any one of these sensors is typically classified under either contact or non-contact. A real-life instance of a contact sensor is the CNC touch probe contacting a workpiece to simplify precision measurement. Also, non-contact sensors can be video and vision systems that are used to capture an image of a workpiece by an optical base and then analyze it. Finally, a variety of laser sensors comprising interferometric, triangulation, and dispersive capabilities can be applied for detection of defects (Vacharanukul and Mekid, 2005). Hence, the use of these multi-sensor systems requires the conversion of them into single-sensor systems in order to make monitoring of the part's processing easier.

9.3.1 Scheduling of 2RRCSIs: In-Process Inspection Scenario

We assume that each sensor f, where $f \in \{1, 2, ..., m_1\}$, detects the part's failure with specified probability q_{1f} each time the part is processing on M_1, and therefore this individual sensor identifies no defect in each time inspection with probability $p_{1f} = 1 - q_{1f}$. This means that the inspection result of each sensor f after each time processing of the part l on M_1 is a random variable Y_{1f}^l which follows a Bernoulli probability distribution of parameter p_{1f}. Needless to say that here we have a sub-inspection set of m_1 different sensors, and the sequence of random variables $[Y_{11}^l, Y_{12}^l, ..., Y_{1m_1}^l]$ that represent the inspection results of m_1 sensors is independent and identically distributed (i.i.d) since all of them follow Bernoulli probability distribution and there is no dependency between them (Sambola et al., 2011). Therefore, the part cannot pass the multi-sensor inspection process even if one of the sensors detects the part's failure. Following that, the generalized Bernoulli distributed variable Y_1^l supporting the success of the multi-sensor inspection system of the part k is expressed as:

$$Y_1^l = \begin{cases} 1 & \text{if } \sum_{f=1}^{m_1} Y_{1f} = m_1 \\ 0 & \text{if } \sum_{f=1}^{m_1} Y_{1f} < m_1 \end{cases}$$
$$P(Y_1^l = 1) = \prod_{f=1}^{m_1} p_{1f}$$
$$P(Y_1^l = 0) = 1 - \prod_{f=1}^{m_1} p_{1f} \tag{9.1}$$

The analysis of M_1 with multiple sensors integrated into it can be converted to the analysis of M_1 with a single sensor using the above equations. Following that, the important random variable for us here to determine is the number of inspections performed by the newly established single-sensor inspection system before passing all sub-inspections. Clearly, for the lth part interred to the rework cell, this number can be represented by a random variable X_1^l which is associated with a geometric distribution with success parameter $p_1 = \prod_{f=1}^{m_1} p_{1f}$ and the time between two inspections equals $(a + \gamma_1)$. The reason for this intuition is that the geometric distribution is defined as a discrete distribution counting the number of Bernoulli trials until the first success. It is then necessary to mention that all sub-inspections of an iteration of the rework process must be restarted because of the memoryless property of the geometric distribution. This property implies that the current sub-inspection process can be performed independent of the number of completed sub-inspections before. The partial cycle times for lth implementation of S_{11}^{10} and S_{12}^{10} are presented in the following lemmas, respectively.

Lemma 1. Having a 2RRCSI, the partial cycle time for lth implementation of S_{11}^{10} is the random variable $T_{S_{11}^{10}}^{l}$ as:

$$T_{S_{11}^{10}}^{l} = 6\varepsilon + 6\delta + (a+\gamma_1)X_1^l + b \tag{9.2}$$

Proof: We know that the lth implementation of S_{11}^{10} includes the tasks below: the robot unloads $(l-1)$th part from M_2 after a full waiting at this machine to finish the processing of the $(l-1)$th part $(b+\varepsilon)$. The robot carries the part from M_2 to O, and then moves backward to I to pick up the lth part and load it on $M_1(3\varepsilon + 5\delta)$. In this stage, the robot undergoes a random waiting depending on the number of rework processes performed by M_1, and eventually it picks up the part l and loads it on M_2. Note that the time taken for an iteration of the rework process equals the summation of the processing and inspection times of the first machine. Therefore, we have $((a+\gamma_1)X_1^l + 2\varepsilon + \delta)$. This completes the proof.

Lemma 2. Having a 2RRCSI, the partial cycle time for lth implementation of S_{12}^{10} is the random variable $T_{S_{12}^{10}}^{l}$ as:

$$T_{S_{12}^{10}}^{l} = 6\varepsilon + 8\delta + max\{0, (a+\gamma_1)X_1^l - (2\varepsilon+4\delta), b-(2\varepsilon+4\delta)\} \tag{9.3}$$

Proof: Here, the robot visits I, M_2, and M_1 in the lth implementation of S_{12}^{10}, respectively. Our goal is to find the time required for all intervening activities which are implemented between two consecutive loadings of M_2. The robot initially moves backward to I to unload the lth part and load it on $M_1(2\varepsilon + 3\delta)$. In the second phase, it returns to M_2 to unload the $(l-1)$th part and transfer it to O after a partial waiting at $M_2(2\varepsilon + 2\delta + w_2^{l-1})$. Likewise, the empty robot returns to M_1 to unload the lth part and transfer it to M_2 after a partial waiting at $M_1(2\varepsilon + 3\delta + w_1^l)$. Therefore, the partial cycle time for lth implementation of S_{12}^{10} is:

$$T_{S_{12}^{10}}^{l} = 6\varepsilon + 8\delta + w_2^{l-1} + w_1^l$$

Where

$$w_2^{l-1} = max\{0, b - (2\varepsilon + 4\delta)\}$$

$$w_1^l = max\{0, (a+\gamma_1)X_1^l - (2\varepsilon + 4\delta) - w_2^{l-1}\}$$

$$w_2^{l-1} + w_1^l = max\{0, (a+\gamma_1)X_1^l - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\}$$

This completes the proof.

Our task now is to find an optimal long-term production strategy considering Lemmas 1 and 2. We present the following theorem which is applicable for all inspection scenarios.

Theorem 1. For two-machine robotic rework cells, there is no dynamic state change from S_{i1}^{hk} to S_{i2}^{hk} (and vice versa) for any particular i, h, and k.

Proof: As mentioned before, the starting point of each one of S_{i1}^{hk} and S_{i2}^{hk} is the moment when the robot loaded a part on M_2 and therefore it waits at M_2 until it receives the next order by the robot controller. If M_1 was loaded at this given moment, the stochastic modelling was essential due to the fact that the extent to which each part has been processed on M_1 at the starting point of each partial cycle is random. Fortunately, M_1 which performs a random number of reworks on the part is empty at this common state. Therefore, there is no relationship between the optimal partial cycle in the current state and the number of reworks performed on the part in the previous partial cycle. This completes the proof.

This theorem in particular implies that all l, where $l \to +\infty$, have a similar production behaviour. As a direct result of this, it can be concluded that it is enough to compare a particular implementation of S_{i1}^{hk} and S_{i2}^{hk}, for example the lth implementation of both of them, and then extend the result to a mass production environment with an infinite period of time. To point out this subject more clearly, let us find the dominancy regions of any one of S_{i1}^{hk} and S_{i1}^{hk} for lth implementation of any particular i, h, and k. We must initially recall some concepts related to stochastic dominancy. There are three kinds of stochastic order relations that are sorted through strangeness (Ross, 1996):

- Absolute Dominance (AD): we say that $T^{l}_{S_{i1}^{hk}}$ is absolutely dominant over $T^l_{S_{i2}^{hk}}$, written $T^l_{S_{i1}^{hk}} \geq_{AD} T^l_{S_{i2}^{hk}}$, only if $P(T^l_{S_{i1}^{hk}} T^l_{S_{i2}^{hk}}) = 1$ and $T^l_{S_{i1}^{hk}} > T^l_{S_{i2}^{hk}}$ is satisfied for at least one $T^l_{S_{i1}^{hk}}$.

- First-order Stochastic Dominance (FSD): we say that $T^{l}_{S_{i1}^{hk}}$ is first-order dominant over $T^l_{S_{i2}^{hk}}$, written $T^l_{S_{i1}^{hk}} \geq_{FSD} T^l_{S_{i2}^{hk}}$, only if $P(T^l_{S_{i1}^{hk}} > \theta) \geq P(T^l_{S_{i2}^{hk}} > \theta)$ for all θ.

- Second-order Stochastic Dominance (SSD): if $T^{l}_{S_{i1}^{hk}}$ is second-order dominant over $T^l_{S^{hk}_{i1}}$, written $T^l_{S^{hk}_{i1}} \geq_{SSD} T^l_{S^{hk}_{i2}}$, then $E(T^l_{S^{hk}_{i1}}) \geq E(T^l_{S^{hk}_{i2}})$.

Since AD has priority over other dominance relations, we determine the regions of AD of $T^l_{S^{10}_{11}}$ and $T^l_{S^{10}_{12}}$ in the first stage.

Theorem 2. Under a 2RRCSI with in-process inspection scenario, it is absolutely better to instruct the robot for implementation of S^{10}_{12} in an infinite period of time if $\lceil \frac{2\delta - b}{a+\gamma_1} \rceil \leq 1$.

Proof: we start with considering $P(T^l_{S^{10}_{11}} \geq T^l_{S^{10}_{12}}) = P(6\varepsilon + 6\delta + (a+\gamma_1)X^l_1 + b \geq 6\varepsilon + 8\delta + max\{0, (a+\gamma_1)X^l_1 - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\}) = P((a+\gamma_1)X^l_1 + b \geq 2\delta + max\{0, (a+\gamma_1)X^l_1 - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\}) = P((a+\gamma_1)X^l_1 + b \geq 2\delta) = P(X^l_1 \geq \lceil \frac{2\delta - b}{a+\gamma_1} \rceil)$. It is necessary to recall that the discrete variable X^l_1 is associated with a geometric distribution and at least equals 1. Therefore, we have $P(T^l_{S^{10}_{11}} \geq T^l_{S^{10}_{12}}) = P(X^l_1 \geq \lceil \frac{2\delta - b}{a+\gamma_1} \rceil) = 1$ when $\lceil \frac{2\delta - b}{a+\gamma_1} \rceil \leq 1$. This is enough to say $T^*_{110} = T^l_{S^{10}_{11}}$ is absolutely dominant over $T^l_{S^{10}_{12}}$ in an infinite period of time. This completes the proof.

Corollary 1. Comparing the two-machine robotic rework cell with the same robotic cell without rework assumption, the chance of optimality for S^{10}_{12} increases whereas this chance decreases for S^{10}_{11}.

Proof: Theorem 2 makes it clear that robotic cells without a rework assumption are a special case of robotic rework cells where X^l_1=1. Nevertheless, completed parts by typical robotic rework cells are not failure-free, and consequently $X^l_1 \geq 1$. It is known that the probability that S^{10}_{12} is the optimal partial cycle of the lth execution is $P(T^l_{S^{10}_{11}} \geq T^l_{S^{10}_{12}}) = P(X^l_1 \geq \lceil \frac{2\delta - b}{a+\gamma_1} \rceil)$. The lower bound of this probability is obtained when $X^l_1 = 1$. Therefore, as shown in Figure 9.4, an increase in the optimality chance of S^{10}_{12} results from an increase in the feasible domain of X^l_1. Regarding Figure 9.4, it only should be noted that S^{10}_{11} and S^{10}_{12} are labelled as S_1 and S_2, respectively, in robotic cells without a rework assumption. This completes the proof.

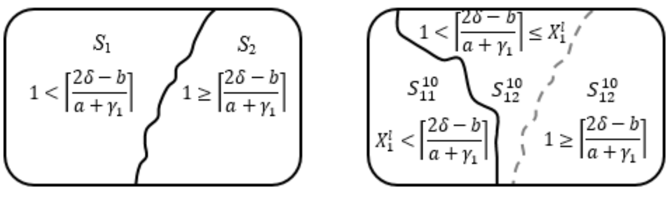

Robotic cells without rework
S_1
$1 < \lceil \frac{2\delta - b}{a+\gamma_1} \rceil$
S_2
$1 \geq \lceil \frac{2\delta - b}{a+\gamma_1} \rceil$
Robotic rework cells
$1 < \lceil \frac{2\delta - b}{a+\gamma_1} \rceil \leq X^l_1$
S^{10}_{11}
S^{10}_{12}
S^{10}_{12}
$X^l_1 < \lceil \frac{2\delta - b}{a+\gamma_1} \rceil$
$1 \geq \lceil \frac{2\delta - b}{a+\gamma_1} \rceil$

Fig. 9.4. The graphical representation of the regions of optimality

Due to the fact that FSD and SSD are our second and third priorities, we determine the regions of the FSD and SSD of $T^l_{S^{10}_{11}}$ and $T^l_{S^{10}_{12}}$, respectively, in Theorems 3 and 4.

Theorem 3. There is no FSD relationship between $T^l_{S^{10}_{11}}$ and $T^l_{S^{10}_{12}}$ executed for a 2RRCSI with an in-process inspection scenario.

Proof: It is enough to prove that the inequality $P(T^l_{S^{10}_{11}} > \theta) \geq P(T^l_{S^{10}_{12}} > \theta)$ is not satisfied for all θ. Obviously, the intersection of $T^l_{S^{10}_{11}}$ and $T^l_{S^{10}_{12}}$ is $6\varepsilon + 6\delta$. Therefore, θ can be replaced with $\lambda = \theta - (6\varepsilon + 6\delta)$, and this yields the following results where the cumulative distribution function $F_{X^l_1}$ of the geometric variable X^l_1 is equal to $1-(1-p_1)^{x^l_1}$ and also $\alpha = max\{0, b-(2\varepsilon+4\delta)\}$ for the sake of simplicity.

$$\begin{aligned}
P(T^l_{S^{10}_{11}} > \theta) &= P((a+\gamma_1)X^l_1 + b > \lambda) = P(X^l_1 > \frac{\lambda - b}{a+\gamma_1}) \\
&= 1 - F(\lfloor \frac{\lambda - b}{a+\gamma_1} \rfloor) = (1-p_1)^{\lfloor \frac{\lambda - b}{a+\gamma_1} \rfloor}
\end{aligned}$$

$$\begin{aligned}
P(T^l_{S^{10}_{12}} > \theta) &= P(2\delta + max\{0, (a+\gamma_1)X^l_1 - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\} > \lambda) \\
&= P(\alpha > \lambda - 2\delta | X^l_1 \leq \frac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1}) P(X^l_1 \leq \frac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1}) \\
&+ P(X^l_1 > \frac{\lambda + 2\varepsilon + 2\delta}{a+\gamma_1} | X^l_1 > \frac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1}) P(X^l_1 > \frac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1})
\end{aligned}$$

None of α, λ, and δ is a random variable. As a consequence, any one of the two following cases may occur for $P(T^l_{S^{10}_{12}} > \theta)$:

1. $\alpha > \lambda - 2\delta$
$$\to P(T^l_{S^{10}_{12}} > \theta) = P(X^l_1 \leq \frac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1}) + P(X^l_1 > \frac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1}) = 1$$
2. $\alpha \leq \lambda - 2\delta$
$$\begin{aligned}
&\to P(T^l_{S^{10}_{12}} > \theta) = P(X^l_1 > \frac{\lambda + 2\varepsilon + 2\delta}{a+\gamma_1} | X^l_1 > \frac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1}) P(X^l_1 > \tfrac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1}) \\
&= P(X^l_1 > \frac{\lambda + 2\varepsilon + 2\delta}{a+\gamma_1} \cap X^l_1 > \frac{\alpha + 2\varepsilon + 4\delta}{a+\gamma_1}) = P(X^l_1 > \frac{\lambda + 2\varepsilon + 2\delta}{a+\gamma_1}) \\
&= 1 - F(\lfloor \frac{\lambda + 2\varepsilon + 2\delta}{a+\gamma_1} \rfloor) = (1-p_1)^{\lfloor \frac{\lambda + 2\varepsilon + 2\delta}{a+\gamma_1} \rfloor}
\end{aligned}$$

If the case 1 is taken into consideration, $P(T^l_{S^{10}_{11}} > \theta) \leq P(T^l_{S^{10}_{12}} > \theta)$, whereas $P(T^l_{S^{10}_{11}} > \theta) \geq P(T^l_{S^{10}_{12}} > \theta)$ for the second case. Thus, it is impossible to satisfy inequality $P(T^l_{S^{10}_{11}} > \theta) \geq P(T^l_{S^{10}_{12}} > \theta)$ (or inequality $P(T^l_{S^{10}_{11}} > \theta) \leq P(T^l_{S^{10}_{12}} > \theta)$ for all θ, and this means that there is no FSD relationship between $T^l_{S^{10}_{11}}$ and $T^l_{S^{10}_{12}}$. This completes the proof.

As mentioned before, SSD order relations are obtained through the expected values of $T^l_{S^{10}_{11}}$ and $T^l_{S^{10}_{12}}$. As a result of that, before proceeding with the next theorem, we must calculate $E(T^l_{S^{10}_{11}})$ and $E(T^l_{S^{10}_{12}})$.

Lemma 3. For the case of a 2RRCSI, the expected values of two partial cycle times $T^l_{S^{10}_{11}}$ and $T^l_{S^{10}_{12}}$ are given by:

$$E(T^l_{S^{10}_{11}}) = 6\varepsilon + 6\delta + \frac{a+\gamma_1}{p_1} + b \tag{9.4}$$

$$\begin{aligned} E(T^l_{S^{10}_{12}}) &= max\{6\varepsilon + 8\delta, b + 4\varepsilon + 4\delta\} \\ &\quad + ((a+\gamma_1)\lfloor \tfrac{max\{2\varepsilon+4\delta, b\}}{a+\gamma_1} \rfloor + \frac{a+\gamma_1}{p_1} - max\{2\varepsilon+4\delta, b\})(1-p_1)^{\lfloor \frac{max\{2\varepsilon+4\delta,b\}}{a+\gamma_1} \rfloor} \end{aligned} \tag{9.5}$$

Proof: The fact that X^l_1 is associated with a geometric distribution with success parameter $p_1 = \prod_{f=1}^{m_1} p_{1f}$ implies:

$$\begin{aligned} E(T^l_{S^{10}_{11}}) &= E(6\varepsilon + 6\delta + (a+\gamma_1)X^l_1 + b) \\ &= E(6\varepsilon + 6\delta + b) + (a+\gamma_1)E(X^l_1) = 6\varepsilon + 6\delta + b + \frac{a+\gamma_1}{p_1} \end{aligned}$$

Regarding $E(T^l_{S^{10}_{12}})$, it contains a triple-sided max term where 0 and $b-(2\varepsilon+4\delta)$ are fixed and $(a+\gamma_1)X^l_1 - (2\varepsilon+4\delta)$ is variable. For simplicity, hereinafter $\beta = 4\varepsilon + 4\delta$ and $\gamma = max\{2\varepsilon + 4\delta, b\}$, and therefore the calculation of the conditional expected value is as follows:

$$\begin{aligned} E(T^l_{S^{10}_{12}}) &= E(6\varepsilon + 8\delta + max\{0, (a+\gamma_1)X^l_1 - (2\varepsilon+4\delta), b-(2\varepsilon+4\delta)\}) \\ &= \beta + E(max\{\gamma, (a+\gamma_1)X^l_1\}) \\ &= \beta + E(max\{\gamma, (a+\gamma_1)X^l_1\} | X^l_1 \leq \lfloor \frac{\gamma}{a+\gamma_1} \rfloor) P(X^l_1 \leq \lfloor \frac{\gamma}{a+\gamma_1} \rfloor) \\ &\quad + E(max\{\gamma, (a+\gamma_1)X^l_1\} | X^l_1 > \lfloor \frac{\gamma}{a+\gamma_1} \rfloor) P(X^l_1 > \lfloor \frac{\gamma}{a+\gamma_1} \rfloor) \end{aligned}$$

Clearly, the $E(max\{\gamma, (a+\gamma_1)X^l_1\})$ equals γ if $X^l_1 \leq \lfloor \gamma/(a+\gamma_1) \rfloor$. Otherwise, this expected value can be reduced to $(a+\gamma_1)\lfloor \gamma/(a+\gamma_1) \rfloor + E(max\{(a+\gamma_1)X^l_1\}$ when $X^l_1 > \lfloor \gamma/(a+\gamma_1) \rfloor$. It is necessary to recall the memoryless property of the geometric distribution for $X^l_1 > \lfloor \gamma/(a+\gamma_1) \rfloor$. This property implies that the current sub-inspection process can be performed independent of the number of completed sub-inspections before, and therefore $E(max\{(a+\gamma_1)X_1^l\} = (a+\gamma_1))/p_1$. We can conclude that:

$$\begin{aligned}
E(T^l_{S^{10}_{12}}) &= \beta + \gamma F(\lfloor \frac{\gamma}{a+\gamma_1} \rfloor) + (a+\gamma_1)(\lfloor \frac{\gamma}{a+\gamma_1} \rfloor + \frac{1}{p_1})(1 - F(\lfloor \frac{\gamma}{a+\gamma_1} \rfloor)) \\
&= \beta + \gamma(1 - (1-p_1)^{\lfloor \frac{\gamma}{a+\gamma_1} \rfloor}) + (a+\gamma_1)(\lfloor \frac{\gamma}{a+\gamma_1} \rfloor + \frac{1}{p_1})(1-p_1)^{\lfloor \frac{\gamma}{a+\gamma_1} \rfloor}) \\
&= \beta + \gamma + ((a+\gamma_1)\lfloor \frac{\gamma}{a+\gamma_1} \rfloor + \frac{a+\gamma_1}{p_1} - \gamma)(1-p_1)^{\lfloor \frac{\gamma}{a+\gamma_1} \rfloor}
\end{aligned}$$

,where F is the cumulative distribution function (CDF) of X_1^l. This equation can be rewritten as Equations (9.5) considering the real amounts of β and γ. This completes the proof.

Theorem 4. If $b + \frac{a+\gamma_1}{p_1}(1 - (1-p_1)^{\lfloor \frac{max\{2\varepsilon+4\delta,b\}}{a+\gamma_1} \rfloor}) < 2\delta$ in a 2RRCSI with in-process inspection scenario, then $T^l_{S^{10}_{12}}$ is second-order larger than $T^l_{S^{10}_{11}}$; else if $b + \frac{a+\gamma_1}{p_1}(1 - (1-p_1)^{\lfloor \frac{max\{2\varepsilon+4\delta,b\}}{a+\gamma_1} \rfloor}) > 2\delta$, then $T^l_{S^{10}_{12}}$ is second-order smaller than $T^l_{S^{10}_{11}}$.

Proof: The proof will be presented in a structure similar to that of Theorem 3: we use Equations (9.4) and (9.5) to consider the case in which $E(T^l_{S^{10}_{11}}) \leq E(T^l_{S^{10}_{12}})$, and then extend the proof for the case in which $E(T^l_{S^{10}_{11}}) > E(T^l_{S^{10}_{12}})$. If we relax the first bracket in $E(T^l_{S^{10}_{12}})$ and recall γ from Lemma 3, then we have:

1. $E(T^l_{S^{10}_{11}}) < E(T^l_{S^{10}_{12}})$

$$\begin{aligned}
&\rightarrow 2\varepsilon + 2\delta + b + \frac{a+\gamma_1}{p_1} < max\{2\varepsilon + 4\delta, b\} + \frac{a+\gamma_1}{p_1}(1-p_1)^{\lfloor \frac{\gamma}{a+\gamma_1} \rfloor} \\
&\rightarrow b + \frac{a+\gamma_1}{p_1}(1 - (1-p_1)^{\lfloor \frac{\gamma}{a+\gamma_1} \rfloor}) \leq 2\delta
\end{aligned}$$

2. $E(T^l_{S^{10}_{11}}) > E(T^l_{S^{10}_{12}})$

$$\begin{aligned}
&\rightarrow 2\varepsilon + 2\delta + b + \frac{a+\gamma_1}{p_1} > max\{2\varepsilon + 4\delta, b\} + \frac{a+\gamma_1}{p_1}(1-p_1)^{\lfloor \frac{\gamma}{a+\gamma_1} \rfloor} \\
&\rightarrow b + \frac{a+\gamma_1}{p_1}(1 - (1-p_1)^{\lfloor \frac{\gamma}{a+\gamma_1} \rfloor}) > 2\delta
\end{aligned}$$

Note that $T^l_{S^{10}_{11}} =_{SSD} T^l_{S^{10}_{12}}$ if $b + \frac{a+\gamma_1}{p_1}(1 - (1-p_1)^{\lfloor \frac{max\{2\varepsilon+4\delta,b\}}{a+\gamma_1} \rfloor}) = 2\delta$. This can be considered as the supplementary part of this theorem. This completes the proof.

At the end of this section, it should be emphasized that Theorems 1 to 4 give an appropriate structure to select the robot's partial cycle with the maximum expected throughput of 2RRCSIs with an in-process inspection scenario, and Lemma 3 helps us to calculate this maximum expected throughput. Clearly, this structure assists industry in both designing and developing basic rework cells.

9.3.2 Scheduling of 2RRCEIs: In-Process Inspection Scenario

For the 2RRCEI case with an in-process inspection scenario, the final parts processed on M_2 are monitored to detect the presence of different types of defects before delivering them to the customers. Similar to 2RRCSIs, there is no difficulty with converting the multi-sensor system into a single-sensor system in 2RRCEIs with an in-process inspection scenario. Each sensor g, where $g \in \{1, 2, ..., m_2\}$, identifies no defect in each time inspection with probability p_{2g}. This builds up a sub-inspection set of m_2 different sensors, and the sequence of random variables $[Y^l_{21}, Y^l_{22}, ..., Y^l_{2m_2}]$ that represent the inspection results of m_2 sensors follow Bernoulli probability distribution. Following that, the generalized Bernoulli distributed variable Y^l_2 supporting the success of the multi-sensor inspection system of the part l is expressed as:

$$Y^l_2 = \begin{cases} 1 & \text{if } \sum_{g=1}^{m_2} Y_{2g} = m_2 \\ 0 & \text{if } \sum_{g=1}^{m_2} Y_{2g} < m_2 \end{cases}$$

$$P(Y^l_2 = 1) = \prod_{g=1}^{m_2} p_{2g}$$
$$P(Y^l_2 = 0) = 1 - \prod_{g=1}^{m_2} p_{2g} \tag{9.6}$$

According to Equation (9.6), the number of inspection of the lth part interred to the rework cell performed by the single-sensor inspection system is the random variable X^l_2 which is associated with a geometric distribution with success parameter $p_2 = \prod_{g=1}^{m_2} p_{2g}$ and the time between two inspections equals $(b + \gamma_2)$. As mentioned earlier, the reason for this intuition is that the geometric distribution is defined as a discrete distribution counting the number of Bernoulli trials until the first success. The partial cycle times for lth implementation of S^{20}_{11} and S^{20}_{12} are presented in the following lemma, respectively.

Lemma 4. Having a 2RRCEI with an in-process inspection scenario, the partial cycle times for lth implementations of S^{20}_{11} and S^{20}_{12} are the random variables $T^l_{S^{20}_{11}}$ and $T^l_{S^{20}_{12}}$ as:

$$T^l_{S^{20}_{11}} = 6\varepsilon{+}6\delta{+}a{+}(b{+}\gamma_2)X^l_2 \tag{9.7}$$

$$T^l_{S^{20}_{12}} = 6\varepsilon{+}8\delta{+}max\{0, a{-}(2\varepsilon{+}4\delta), (b{+}\gamma_2)X^l_2{-}(2\varepsilon{+}4\delta)\} \tag{9.8}$$

Proof: If we follow the order of tasks performed in Lemmas 1 and 2, once again, we achieve the desired results. The only difference here is that the processing time of M_2 is the random variable due to sensors installed into it, not M_1. The rest of this proof is easy and therefore omitted.

Our task now is to find an optimal long-term production strategy considering Lemmas 4.

Corollary 2. Theorem 1 to 4 are also correct for 2RRCEIs if a and b be swapped with each other in all associated inequalities.

Proof: Easy and omitted.

Corollary 3. For the case of a 2RRCEI with an in-process inspection scenario, the expected values of partial cycle time of S^{20}_{11} and S^{20}_{12} are given by:

$$E(T^l_{S^{20}_{11}}) = 6\varepsilon + 6\delta + a + \frac{b+\gamma_2}{p_2} \tag{9.9}$$

$$\begin{aligned} E(T^l_{S^{20}_{12}}) &= max\{6\varepsilon + 8\delta, a + 4\varepsilon + 4\delta\} \\ &\quad +((b+\gamma_2)\lfloor\tfrac{max\{2\varepsilon+4\delta,a\}}{b+\gamma_2}\rfloor + \frac{b+\gamma_2}{p_2} - max\{2\varepsilon + 4\delta, a\})(1-p_2)^{\lfloor\frac{max\{2\varepsilon+4\delta,a\}}{b+\gamma_2}\rfloor} \end{aligned} \tag{9.10}$$

Proof: Easy and omitted.

It is worth noting that the result of this subsection along with the previous subsection create a parallel mechanism for analysing both 2RRCSIs and 2RRCEIs with an in-process inspection scenario. Actually, we must make it clear that the main purpose of inspection is to control the quality of parts at the early stage of production to decrease production complexity, or control the quality of parts through a final inspection at the last stage to satisfy the customers' requirements. According to either of these priorities, a 2RRCSI or 2RRCEI can be designed for a robotic rework cell with an in-process inspection scenario, respectively, and then the result of Section 9.3 can be applied for optimizing the performance. cells.

9.4 Sequencing of Activities under Post-Process Inspection Scenario

This section is dedicated to the scheduling problem of two-machine robotic rework cells to cover both "Start of Line" and "End of Line" testing plans for a post-process inspection scenario. On the contrary to in-process inspection scenario, the measurement is performed by an independent inspection machine located after the production machine if we follow the post-process inspection scenario. There are at least two motivations for a post-process inspection scenario. Firstly, the inspection process may predominantly need a specific condition which cannot be satisfied by the production machine. This specific condition can be an exact temperature, pressure, etc. Thus, it is necessary to perform the inspection process by an extra inspection machine. Secondly, performing both production and inspection steps by a production machine equipped by sensors may make this machine a bottleneck of the rework cell. Following that, it is often impossible or at least time-consuming to perform both production and inspection steps by the same machine.

9.4.1 Scheduling of 2RRCSIs: Post-Process Inspection Scenario

The performance of post-process inspection must be tested for 2RRCSIs. Consequently, we provide the best design of them at first. Consider a 2RRCSI in which the inspection process is performed by an independent machine, namely N, between M_1 and M_2. This means that the set of m_1 sensors is installed into N instead of M_1. Accordingly, the stochastic part processing route is $I \rightarrow M_1 \leftrightarrows N \rightarrow M_2 \rightarrow O$. In this route, the rework machine M_1 and the inspection machine N together build up the stochastic closed-loop event $M_1 \leftrightarrows N$. This stochastic closed-loop event is associated with the probability of the rework being needed after inspection of the part on N. We make the following assumptions regarding $M_1 \leftrightarrows N$: 1) Elements of $M_1 \leftrightarrows N$ have deterministic occurrence time. 2) The number of switching into $M_1 \leftrightarrows N$ for the lth part is the stochastic variable X_1^l.

Due to the fact that the design of the rework cell is changed in comparison with the rework cell under in-process inspection scenario, it is essential to check whether the cell is still deadlock-free or not. As mentioned before, the receiving server and sending server must be empty and loaded before the load and unload processes, respectively, in regard to the in-process inspection scenario. However, considering the post-process inspection scenario, a counterexample that shows these constraints are necessary but not sufficient for deadlock prevention is as follows:

Assume that M_1 is empty and N is loaded by the part $l-1$ in a 2RRCSI. The robot loads the part l on M_1 as receiving server, and then move to N in order to unload the part $l-1$. The part on N may be defective and consequently needs to revisit M_1. This means that the robot is not able to reload part $l-1$ on the busy machine M_1, and also it is not able to load part l on the busy machine N for inspection. A circular deadlock occurs under this condition even though all constraints were satisfied for the previous robot's movements (Wysk et al., 1994; Venkatesh and Smith, 2003; Odrey and Meja, 2005). Therefore, an additional constraint is demanded to make 2RRCSI deadlock-free. This constraint works by avoiding any kinds of circular deadlocks. It does not allow the robot to load the part on a machine if the machine is a deadlock-risk resource, which potentially can lead to a circular deadlock. Hence, M_1 can be loaded only if N is empty considering a 2RRCSI. The following lemmas are obtained if we assume that the required time for traveling between M_1 and N equals δ.

Lemma 5. Having a 2RRCSI, the partial cycle time for lth implementation of S_{21}^{10} is the random variable $T_{S_{21}^{10}}^{l}$ as:

$$T_{S_{21}^{10}}^{l} = 4\varepsilon+6\delta+(a+\gamma_1+4\varepsilon+2\delta)X_1^l+b \tag{9.11}$$

Proof: All tasks that the robot must perform are similar to S_{11}^{10}. The only difference here is that $M_1 \leftrightarrows N$ occurs X_1^l times for any particular part l. The stochastic time taken for this number of iterations of $M_1 \leftrightarrows N$ is $(a+\gamma_1+4\varepsilon+2\delta)X_1^l$, whilst this time is $(a+\gamma_1)X_1^l+2\varepsilon$ for S_{11}^{10}. This completes the proof.

Lemma 6. Having a 2RRCSI, the partial cycle time for lth implementation of S_{22}^{10} is the random variable $T_{S_{22}^{10}}^{l}$ as:

$$T_{S_{22}^{10}}^{l} = 8\varepsilon+12\delta+\gamma_1+(a+\gamma_1+4\varepsilon+2\delta)(X_1^l-1)+max\{0, a-(2\varepsilon+6\delta), b-(2\varepsilon+6\delta)\} \tag{9.12}$$

Proof: Starting from M_2, the robot visits I, M_2, M_1 and N in the lth implementation of S_{22}^{10}, respectively. It is remarkable that the robot does not stop the implementation of $M_1 \leftrightarrows N$ until it has the first success at N. Let us now calculate $T_{S_{22}^{10}}^{l}$ as follows: When the robot does not execute the stochastic closed-loop event $M_1 \leftrightarrows N$, the total time required for loading, unloading, and traveling is constant and equal to $4\varepsilon+10\delta$. The stochastic time taken for the first iterations of $M_1 \leftrightarrows N$ is $w_2^{l-1}+w_1^l+\gamma_1+4\varepsilon+2\delta$ where:

$$w_1^l = max\{0, a-(2\varepsilon+6\delta)-w_2^{l-1}\}$$

$$w_2^{l-1} = max\{0, b-(2\varepsilon+6\delta)\}$$

$w_2^{l-1} + w_1^l = max\{0, a - (2\varepsilon + 6\delta), b - (2\varepsilon + 6\delta)\}$

And the stochastic time taken for any one of X_1^{l-1} remaining closed-loop events is equal to $a + \gamma_1 + 4\varepsilon + 2\delta$. This completes the proof.

An indirect result from these lemmas is that the expected values of $T^l_{S^{10}_{21}}$ and $T^l_{S^{10}_{22}}$ are $E(T^l_{S^{10}_{21}}) = 4\varepsilon + 6\delta + \frac{a+\gamma_1+4\varepsilon+2\delta}{p_1} + b$ and $E(T_(S_22^10)^l) = 8\varepsilon + 12\delta + \gamma_1 + \frac{(a+\gamma_1+4\varepsilon+2\delta)(1-p_1)}{p_1} + max\{0, a - (2\varepsilon + 6\delta), b - (2\varepsilon + 6\delta)\}$. Since AD has priority over other dominance relations, we determine the regions of AD of $T^l_{S^{10}_{21}}$ and $T^l_{S^{10}_{22}}$ in the following Theorem. Note that there is no need to evaluate FSD and SSD relationship between these partial cycles if the regions of AD relationship cover entire state space.

Theorem 5. Under a 2RRCSI with post-process inspection scenario, the robot movements must be instructed according to Table 9.1:

Table 9.1: The optimality region of post-process inspection scenario for 2RRCSIs

Operational Parameter	$a + b < 4\delta$	$a + b = 4\delta$	$a + b > 4\delta$
Stochastic Order Relations	$T^l_{S^{10}_{21}} \leq_{AD} T^l_{S^{10}_{22}}$	$T^l_{S^{10}_{21}} =_{AD} T^l_{S^{10}_{22}}$	$T^l_{S^{10}_{21}} \geq_{AD} T^l_{S^{10}_{22}}$
Optimality Relationships	$T^*_{210} = T^l_{S^{10}_{21}}$	$T^*_{210} = T^l_{S^{10}_{21}} \& T^l_{S^{10}_{22}}$	$T^*_{210} = T^l_{S^{10}_{22}}$

Proof: The first column of Table 9.1 results from the following observation:

$P(T^l_{S^{10}_{11}} < T^l_{S^{10}_{12}}) = P(4\varepsilon + 6\delta + (a + \gamma_1 + 4\varepsilon + 2\delta)X_1^l + b < 8\varepsilon + 12\delta + \gamma_1 + (a + \gamma_1 + 4\varepsilon + 2\delta)(X_1^l - 1) + max\{0, a - (2\varepsilon + 6\delta), b - (2\varepsilon + 6\delta)\}) = P(a + b < 4\delta + max\{0, a - (2\varepsilon + 6\delta), b - (2\varepsilon + 6\delta)\}) = P(a + b < 4\delta)$

Clearly, both $T^l_{S^{10}_{11}}$ and $T^l_{S^{10}_{12}}$ have stochastic nature due to the input parameter X_1^l, whereas none of $a + b$ and 4δ yields a stochastic value. Accordingly, $a + b$ is less than 4δ or not, depending on deterministic parameters a, b and δ. We can prove the second and the third columns in the same way. This completes the proof.

9.4.2 Scheduling of 2RRCEIs: Post-Process Inspection Scenario

Let us represent the stochastic part processing route of 2RRCEI as $I \rightarrow M_1 \rightarrow M_2 \leftrightarrows N \rightarrow O$ here. Similar to the previous case, there is a stochastic closed-loop event $M_2 \leftrightarrows N$ which occurs X_2^l times for any particular part l. It is also known that M_2 can be loaded only if N be empty because it is a deadlock-risk resource. The performance of post-process inspection for 2RRCEIs is tested in the forthcoming lemma and theorem:

Lemma 7. For a 2RRCEI with a post-process inspection scenario, $T^l_{S^{20}_{21}}$ and $T^l_{S^{20}_{22}}$ are:

$$T^l_{S^{20}_{21}} = 4\varepsilon+6\delta+a+(b+\gamma_2+4\varepsilon+2\delta)X_2^l \tag{9.13}$$

$$\begin{aligned} T^l_{S^{20}_{22}} &= 8\varepsilon + 10\delta + \gamma_2 + (b + \gamma_2 + 4\varepsilon + 2\delta)(X_2^l - 1) \\ &+max\{0, a - (4\varepsilon + 6\delta + \gamma_2 + (b + \gamma_2 + 4\varepsilon + 2\delta)(X_2^l - 1)), b - (2\varepsilon + 4\delta)\} \end{aligned} \tag{9.14}$$

Proof: Regarding $T^l_{S^{20}_{21}}$, the only difference in comparison with the corresponding cycle in 2RRCSI is that completed parts on M_1 are failure-free, not on M_2. Thus, we must swap a and γ_1 with b and γ_2, respectively. The proof of $T^l_{S^{20}_{22}}$ is a bit more complex. We not only must swap a and γ_1 with b and γ_2, but also decrease total traveling time to 10δ. Therefore, w_1^l and w_2^{l-1} are changed to:

$$w_1^l = max\{0, a - (4\varepsilon + 6\delta + \gamma_2 + (b + \gamma_2 + 4\varepsilon + 2\delta)(X_2^l - 1) + w_2^{l-1})\}$$

$$w_2^{l-1} = max\{0, b - (2\varepsilon + 4\delta)\}$$

$$w_2^{l-1} + w_1^l = max\{0, a - (4\varepsilon + 6\delta + \gamma_2 + (b + \gamma_2 + 4\varepsilon + 2\delta)(X_2^l - 1)), b - (2\varepsilon + 4\delta)\}$$

This completes the proof.

Theorem 6. Under a 2RRCEI with post-process inspection scenario, the robot movements must be instructed according to Table 9.2:

Table 9.2: The optimality region of post-process inspection scenario for 2RRCEIs

Operational Parameter	$a + b < 2\delta$	$a + b = 2\delta$	$a + b > 2\delta$
Stochastic Order Relations	$T^l_{S^{20}_{21}} \leq_{AD} T^l_{S^{20}_{22}}$	$T^l_{S^{20}_{21}} =_{AD} T^l_{S^{20}_{22}}$	$T^l_{S^{20}_{21}} \geq_{AD} T^l_{S^{20}_{22}}$
Optimality Relationships	$T^*_{220} = T^l_{S^{20}_{21}}$	$T^*_{220} = T^l_{S^{20}_{21}} \& T^l_{S^{20}_{22}}$	$T^*_{220} = T^l_{S^{20}_{22}}$

Proof: Obviously, $P(T^l_{S^{20}_{21}} < T^l_{S^{20}_{22}}) = P(a + b < 2\delta + max\{0, a - (4\varepsilon + 6\delta + \gamma_2 + (b + \gamma_2 + 4\varepsilon + 2\delta)(X_2^l - 1)), b - (2\varepsilon + 4\delta)\})$ in which there is no stochastic parameter and also the *max* term is nonnegative. Consequently, $T^l_{S^{20}_{21}} \leq_{AD} T^l_{S^{20}_{22}}$ only if $P(a+b < 2\delta)$. The rest of this proof is easy and therefore omitted. This completes the proof.

It is worth noting that the result of Section 9.4 creates a mechanism for analysing cells with a post-process inspection scenario. It is assumed that there is an obligation to apply this scenario due to a need of specific conditions such as an exact temperature or pressure. Thus, it is more appropriate to perform the inspection process by an extra inspection machine. Nonetheless, relaxing this assumption, we describe a newly-developed scenario in the forthcoming section to improve performance of cells.

9.5 Sequencing of Activities under In-Line Inspection Scenario

This section is dedicated to the scheduling problem of two-machine robotic rework cells to cover both "Start of Line" and "End of Line" testing plans for an in-line inspection scenario. For such a scenario, the inspection is performed by a multi-function robot which is able not only to transfer the part between two adjacent machines but also to inspect the part in transit.

Although there are a number of studies related to multi-function robots, all of them assumed that the robot acts as a spot-welding gun, a spray-painting gun, or an assembly device in addition to the material handling device (Keating and Oxman, 2013; Foumani et al., 2014). All these types of operations have the same nature, whilst the result of inspection is not clear in advance. This characteristic of the inspection process distinguishes the analysis of it from other types of operations.

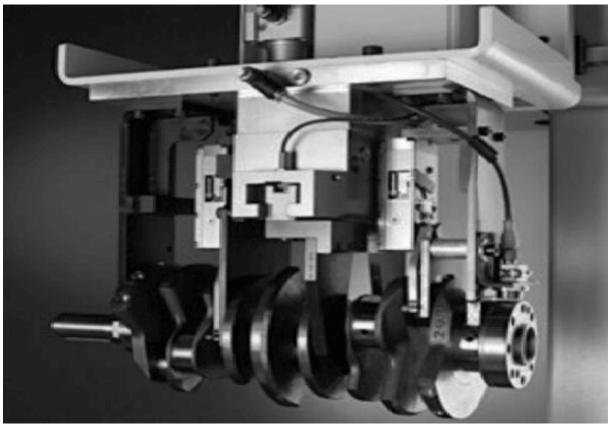


Fig. 9.5. Measurement of crankshaft diameters in transit

Under an in-line inspection scenario, the inspection is predominantly performed by a Grip-Gage-Go (GGG) gripper attached to the robot arm. Figure 9.5 depicts an example of this gripper used for measuring the diameter of a crankshaft. In this figure, the measuring heads are integrated into the automation by adding gages and also crankshaft locating features to the robotic arm.

9.5.1 Sequencing of Multi-Function Robot in 2RRCSIs

The multi-function robot is responsible for inspection of the part here. However, this inspection is not necessarily performed in transit, which is named transition inspection strategy. Alternatively, the robot can unload the part from M_1 and then stop in front of this machine to finish the inspection process. This alternative strategy is called stop inspection strategy. Clearly, the stop inspection strategy leads to one of the following cases: 1) it increases the partial cycle time, for at least γ_1, if the inspection process shows that the part must pass M_1, 2) it decreases the partial cycle time, for at least $min\{\delta, \gamma_1\}$, if the inspection process shows that the part must reload to M_1 for a rework process. It should be emphasized that γ_1 it the time that the robot could save during the forward movement from M_1 to M_2 and performing inspection simultaneously, while $min\{\delta, \gamma_1\}$ is the time that the robot saved due to omitting the backward movement to M_1 after identifying the part failure. In order to determine the partial cycle times, we label them by $T^l_{S^{11}_{31}}, T^l_{S^{11}_{32}}, T^l_{S^{12}_{31}}$ and $T^l_{S^{12}_{32}}$, respectively. The number of cycles is increased from two to four since k=1 under the stop inspection strategy and k=2 under the transition inspection strategy.

Lemma 8. Having a 2RRCSI with in-line inspection scenario, the partial cycle times of $S^{11}_{31}, S^{11}_{32}, S^{12}_{31}$ and S^{12}_{32} are:

$$T^l_{S^{11}_{31}} = 4\varepsilon+6\delta+(a+\gamma_1+2\varepsilon)X^l_1+b \tag{9.15}$$

$$T^l_{S^{11}_{32}} = 6\varepsilon + 8\delta + \gamma_1 + (a + \gamma_1 + 2\varepsilon)(X^l_1 - 1) + max\{0, a - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\} \tag{9.16}$$

$$T^l_{S^{12}_{31}} = 6\varepsilon+5\delta+a+max\{\delta, \gamma_1\}+(a+\gamma_1+min\{\delta, \gamma_1\}+2\varepsilon)(X^l_1-1)+b \tag{9.17}$$

$$T^l_{S^{12}_{32}} = 6\varepsilon + 7\delta + max\{\delta, \gamma_1\} + (a + \gamma_1 + min\{\delta, \gamma_1\} + 2\varepsilon)(X^l_1 - 1) + max\{0, a - (2\varepsilon+4\delta), b-(2\varepsilon+4\delta)\} \tag{9.18}$$

Proof: The partial cycle time calculations are almost the same in each of four formulas, and therefore we do not need to prove all of them. Let us only obtain $T^l_{S_{31}^{11}}$ and $T^l_{S_{31}^{12}}$:

- $T^l_{S_{31}^{11}}$ is equal to Equation (9.2) plus $2\varepsilon(X_1^l - 1)$. Obviously, this additional time which follows a geometric distribution is because of the fact that the robot must unload the part from M_1 for an individual inspection and then reload it at M_1 if the outcome of the inspection process does not be satisfying.
- The additional time for $T^l_{S_{31}^{12}}$ is $-\delta - \gamma_1 + max\{\delta, \gamma_1\} + (min\{\delta, \gamma_1\} + 2\varepsilon)(X_1^l - 1)$. In more detail, this is the elapsed inspection time until the first success. It contains $min\{\delta, \gamma_1\}$ and 2ε for backward movement and unload/reload operations of $X_1^l - 1$ fails, respectively. Also, $\delta + \gamma_1$ in Equation (9.2) is replaced with $max\{\delta, \gamma_1\}$ which is time taken by robot for movement from M_1 to M_2.

Likewise, we can prove $T^l_{S_{32}^{11}}$ and $T^l_{S_{32}^{12}}$. This completes the proof.

The following question arises in this stage of the work: which one of stop and transition inspection strategies should be implemented? In answer to this question, we summarize briefly the linkages between these strategies regardless of "Start of Line" and "End of Line" testing plans.

Theorem 7. *In a rework robotic cell with an in-line inspection scenario, the stop inspection strategy is more effective if the success probability is less than $\frac{1}{2}$, whereas the transition inspection strategy is more effective if the success probability is greater than $\frac{1}{2}$.*

Proof: Taking into account three kinds of stochastic order relationships, the proof will be presented in a combined structure similar to that of Theorems 2, 3 and 4:

- Absolute dominance: it is enough to prove that both $P(T^l_{S_{31}^{11}} \geq T^l_{S_{31}^{12}}) = 1$ and $P(T^l_{S_{31}^{11}} \leq T^l_{S_{31}^{12}}) = 1$ are not satisfied to conclude that there is no AD relationship between $T^l_{S_{31}^{11}}$ and $T^l_{S_{31}^{12}}$. We know that $P(T^l_{S_{31}^{11}} \geq T^l_{S_{31}^{12}}) = P(max\{\delta, \gamma_1\} + min\{\delta, \gamma_1\}(X_1^l - 1) \leq \delta + \gamma_1) = P(X_1^l \leq (\delta + \gamma_1 - max\{\delta, \gamma_1\} + min\{\delta, \gamma_1\})/(min\{\delta, \gamma_1\})) = P(X_1^l \leq 2) = 1 - (1 - p_1)^2 \neq 1$. Also, similarly, $P(T^l_{S_{31}^{11}} \leq T^l_{S_{31}^{12}}) = 1 - p_1 \neq 1$.

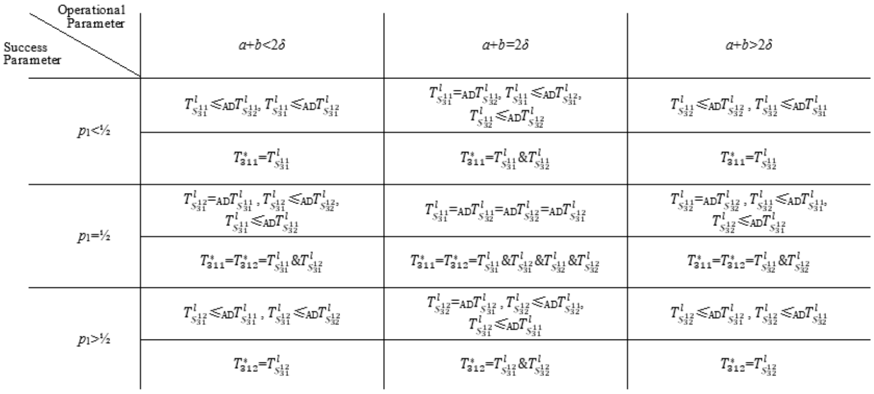
- First-order stochastic dominance: For this case, we have $P(T^{l}_{S^{11}_{31}} \geq \theta) = P(X^{l}_{1} \geq (\theta - (4\varepsilon + 6\delta + b))/(a + \gamma_1 + 2\varepsilon))$ and $P(T^{l}_{S^{12}_{31}} \geq \theta) = P(X^{l}_{1} \geq (\theta + \gamma_1 + min\{\delta, \gamma_1\} - (4\varepsilon + 5\delta + max\{\delta, \gamma_1\} + b))/(a + \gamma_1 + min\{\delta, \gamma_1\} + 2\varepsilon))$. It is easy to prove that there is no FSD relationship between them.

- Second-order stochastic dominance: we should perform a comparison between $E(T^{l}_{S^{11}_{31}}) = 4\varepsilon + 6\delta + (a + \gamma_1 + 2\varepsilon)/p_1 + b$ and $E(T^{l}_{S^{12}_{31}}) = 4\varepsilon + 5\delta + max\{\delta, \gamma_1\} - \gamma_1 - min\{\delta, \gamma_1\} + (a + \gamma_1 + min\{\delta, \gamma_1\} + 2\varepsilon)/p_1 + b$. Since $E(T^{l}_{S^{11}_{31}}) - E(T^{l}_{S^{12}_{31}}) = 2min\{\delta, \gamma_1\} - (min\{\delta, \gamma_1\})/p_1$, we can conclude that $\frac{1}{2}$ is the breakpoint with regards to SSD.

Likewise, we can prove these stochastic order relationships for $T^{l}_{S^{11}_{32}}$ and $T^{l}_{S^{12}_{32}}$. This completes the proof.

Theorem 8. As shown in Table 9.3, the optimal partial cycle for a 2RRCSI can be obtained through a two-phase procedure for the in-line inspection scenario.

Table 9.3: The optimality region under in-line inspection scenario for 2RRCSIs



Operational Parameter / Success Parameter	$a+b<2\delta$	$a+b=2\delta$	$a+b>2\delta$
$p_1<½$	$T^{l}_{S^{11}_{31}} \leqslant_{AD} T^{l}_{S^{11}_{32}}, T^{l}_{S^{11}_{31}} \leqslant_{AD} T^{l}_{S^{12}_{31}}$	$T^{l}_{S^{11}_{31}} =_{AD} T^{l}_{S^{11}_{32}}, T^{l}_{S^{11}_{31}} \leqslant_{AD} T^{l}_{S^{12}_{31}},$ $T^{l}_{S^{11}_{32}} \leqslant_{AD} T^{l}_{S^{12}_{32}}$	$T^{l}_{S^{11}_{32}} \leqslant_{AD} T^{l}_{S^{12}_{32}} , T^{l}_{S^{11}_{32}} \leqslant_{AD} T^{l}_{S^{11}_{31}}$
	$T^{*}_{311}=T^{l}_{S^{11}_{31}}$	$T^{*}_{311}=T^{l}_{S^{11}_{31}} \& T^{l}_{S^{11}_{32}}$	$T^{*}_{311}=T^{l}_{S^{11}_{32}}$
$p_1=½$	$T^{l}_{S^{12}_{31}} =_{AD} T^{l}_{S^{11}_{31}} , T^{l}_{S^{12}_{31}} \leqslant_{AD} T^{l}_{S^{12}_{32}},$ $T^{l}_{S^{11}_{31}} \leqslant_{AD} T^{l}_{S^{11}_{32}}$	$T^{l}_{S^{11}_{31}} =_{AD} T^{l}_{S^{11}_{32}} =_{AD} T^{l}_{S^{12}_{32}} =_{AD} T^{l}_{S^{12}_{31}}$	$T^{l}_{S^{11}_{32}} =_{AD} T^{l}_{S^{12}_{32}} , T^{l}_{S^{11}_{32}} \leqslant_{AD} T^{l}_{S^{11}_{31}},$ $T^{l}_{S^{12}_{32}} \leqslant_{AD} T^{l}_{S^{12}_{31}}$
	$T^{*}_{311}=T^{*}_{312}=T^{l}_{S^{11}_{31}} \& T^{l}_{S^{12}_{31}}$	$T^{*}_{311}=T^{*}_{312}=T^{l}_{S^{11}_{31}} \& T^{l}_{S^{12}_{31}} \& T^{l}_{S^{11}_{32}} \& T^{l}_{S^{12}_{32}}$	$T^{*}_{311}=T^{*}_{312}=T^{l}_{S^{11}_{32}} \& T^{l}_{S^{12}_{32}}$
$p_1>½$	$T^{l}_{S^{12}_{31}} \leqslant_{AD} T^{l}_{S^{11}_{31}} , T^{l}_{S^{12}_{31}} \leqslant_{AD} T^{l}_{S^{12}_{32}}$	$T^{l}_{S^{12}_{32}} =_{AD} T^{l}_{S^{12}_{31}} , T^{l}_{S^{12}_{32}} \leqslant_{AD} T^{l}_{S^{11}_{32}},$ $T^{l}_{S^{12}_{31}} \leqslant_{AD} T^{l}_{S^{11}_{31}}$	$T^{l}_{S^{12}_{32}} \leqslant_{AD} T^{l}_{S^{12}_{31}} , T^{l}_{S^{12}_{32}} \leqslant_{AD} T^{l}_{S^{11}_{32}}$
	$T^{*}_{312}=T^{l}_{S^{12}_{31}}$	$T^{*}_{312}=T^{l}_{S^{12}_{31}} \& T^{l}_{S^{12}_{32}}$	$T^{*}_{312}=T^{l}_{S^{12}_{32}}$

Proof: Relationships related to p_1 follow from Theorem 7. So, we limit this proof to the columns of Table 9.3. The first column results from the following observation:

- $p_1 < \frac{1}{2}$ and $a + b < 2\delta$: This means that the state of this case is restricted to $T^{l}_{S^{11}_{31}}$ and $T^{l}_{S^{11}_{32}}$. Clearly, $T^{l}_{S^{11}_{31}} - T^{l}_{S^{11}_{32}} = a + b - (2\delta + max\{0, a - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\})$. Therefore, $T^{l}_{S^{11}_{31}} < T^{l}_{S^{11}_{32}}$ if $a + b < 2\delta$.

- $p_1 = \frac{1}{2}$ and $a+b < 2\delta$: This means that there is no restriction and any one of $T^l_{S^{11}_{31}}, T^l_{S^{11}_{32}}, T^l_{S^{12}_{31}}$ and $T^l_{S^{12}_{32}}$ has a chance of optimality. $T^l_{S^{11}_{31}} < T^l_{S^{11}_{32}}$ and $T^l_{S^{12}_{31}} < T^l_{S^{12}_{32}}$ when $a+b < 2\delta$. Also, following form Theorem 7, $T^l_{S^{11}_{31}} = T^l_{S^{12}_{31}}$ when $p_1 = \frac{1}{2}$. Accordingly, $T^*_{311} = T^*_{312} = T^l_{S^{11}_{31}} \& T^l_{S^{12}_{31}}$.
- $p_1 > \frac{1}{2}$ and $a+b < 2\delta$: This means that the state of this case is restricted to $T^l_{S^{12}_{31}}$ and $T^l_{S^{12}_{32}}$. It is obvious that, $T^l_{S^{12}_{31}} - T^l_{S^{12}_{32}} = a + b - (2\delta + max\{0, a - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\})$. Therefore, $T^l_{S^{12}_{31}} < T^l_{S^{12}_{32}}$ if $a+b < 2\delta$.

Similarly, we can prove results in the second and third columns. This completes the proof.

9.5.2 Sequencing of Multi-Function Robot in 2RRCEIs

Multi-function robots are able to perform different types of inspections. For instance, such flexibility in crankshaft's inspection can be achieved by considering alternative types of measuring heads for inspections between M_1 and M_2 and loading it to the tool magazines of the robot. Therefore, it is an easy task to convert a multi-function robot used in a 2RRCSI into an appropriate multi-function robot for the 2RRCEI. Only, the measuring head must be exchanged.

Lemma 9. Having a 2RRCEI with in-line inspection scenario, the partial cycle times of $S^{21}_{31}, S^{21}_{32}, S^{22}_{31}$ and S^{22}_{32} are:

$$T^l_{S^{21}_{31}} = 4\varepsilon+6\delta+a+(b+\gamma_2+2\varepsilon)X^l_2 \tag{9.19}$$

$$T^l_{S^{21}_{32}} = 6\varepsilon + 8\delta + \gamma_2 + (b + \gamma_2 + 2\varepsilon)(X^l_2 - 1) + max\{0, a - (2\varepsilon + 4\delta + \gamma_2 + (b + \gamma_2 + 2\varepsilon)(X^l_2 - 1)), b - (2\varepsilon + 4\delta)\} \tag{9.20}$$

$$T^l_{S^{22}_{31}} = 6\varepsilon+5\delta+b+max\{\delta,\gamma_2\}+(b+\gamma_2+min\{\delta,\gamma_2\}+2\varepsilon)(X^l_2-1)+a \tag{9.21}$$

$$T^l_{S^{22}_{32}} = 6\varepsilon+7\delta+max\{\delta,\gamma_2\}+(b+\gamma_2+min\{\delta,\gamma_2\}+2\varepsilon)(X^l_2-1)+max\{0, a-(2\varepsilon+3\delta+max\{\delta,\gamma_2\}+(b+\gamma_2+min\{\delta,\gamma_2\}+2\varepsilon)(X^l_2-1)), b-(2\varepsilon+4\delta)\} \tag{9.22}$$

Proof: Follows from Lemma 8.

Corollary 4. Even if a 2RRCSI be swapped with a 2RRCEI, Theorems 7 and 8 hold.

Proof: Regarding Theorem 7, it should be emphasized that both stop and transition inspection strategies are operation-oriented, not layout-oriented. Therefore, it is not a matter where we want execute these strategies, either between M_1 and M_2 or between M_2 and O. Additionally, regarding Theorem 8, it should be noted that the breakpoint $a+b=2\delta$ is not changed for this case although all cycle times are decreased. Hence, there is no loss of generality by considering results of Table 9.3 for a 2RRCEI with the in-line inspection scenario. Only, for this case, h=2 should be considered instead of h=1. This completes the proof.

9.6 The Comparison of In-Process and Post-Process Inspection Scenarios with an In-Line Inspection Scenario

This section is done to evaluate a basic two-machine robotic rework cell and determine if it is technically profitable to replace an in-process (or post-process) inspection scenario with an in-line inspection scenario. It is assumed that all inspection scenarios are cost-free. Also, there is no obligation to apply only a predefined inspection scenario, and it is technically feasible to choose any one of three inspection scenarios which has higher performance. We develop a comparative analysis for this purpose due to the fact that it is an effective tool for evaluating the performance of a new production system design before any actual implementation. The obtained results of Sections 9.3, 9.4 and 9.5 together are used for the comparison between the performance of in-process, post-process and in-line inspection scenarios here. Let us initially compare the performance of in-process and post-process scenarios in Theorem 9.

Theorem 9. Regardless of the applied testing plan, any particular partial cycle of a rework cell with post-process inspection scenario is dominated by at least one of the partial cycles of the corresponding rework cell with in-process inspection scenario.

Proof: let us start the proof with 2RRCEIs case. For a 2RRCEI with post-process scenario, S_{21}^{10} or S_{22}^{10} is the optimal cycle. So, regardless of the optimal cycle of the post-process inspection scenario, it suffices to show that always there is a partial cycle for in-process inspection scenario which its cycle time is less than T_{210}^{*}. For the first case, we assume that $T_{210}^{*} = T_{S_{21}^{10}}^{l}$. Clearly, $T_{S_{11}^{10}}^{l} \leq_{AD} T_{S_{21}^{10}}^{l}$ since $T_{S_{21}^{10}}^{l} = T_{S_{11}^{10}}^{l} + (4\varepsilon + 2\delta)X_1^l - 2\varepsilon$. Now, let us assume that $T_{210}^{*} = T_{S_{22}^{10}}^{l}$ for the second case where again $T_{S_{11}^{10}}^{l} \leq_{AD} T_{S_{21}^{10}}^{l}$. Therefore, T_{110}^{*} always dominates T_{210}^{*}. A similar argument can be made for 2RRCEIs with the post-process inspection scenario.

Since all partial cycles of the cell with post-process inspection scenario are disqualified, we should only compare two scenarios: in-process and in-line. In this regard, emphasis is initially put on implementation of stop inspection strategy in 2RRCSIs and 2RRCEIs with in-line inspection scenario. Assume without loss of generality that $p_1 \leq \frac{1}{2}$ and $p_2 \leq \frac{1}{2}$ for 2RRCSIs and 2RRCEIs with in-line inspection scenario, respectively. Then, the robot must be instructed to perform either of S_{31}^{11} or S_{32}^{11} depending on a, b, and δ.

Theorem 7. Whenever a stop inspection strategy yields an optimal cycle, the inspection scenario must not be shifted from in-process to in-line.

Proof: To prove this theorem, it is enough to show that at least one of $T^l_{S_{11}^{10}}$ or $T^l_{S_{12}^{10}}$ is smaller than both $T^l_{S_{31}^{11}}$ and $T^l_{S_{32}^{11}}$. Then, we can conclude that neither of S_{31}^{11} or S_{32}^{11} seems to reduce the partial cycle time of a rework cell formed by a multi-function robot. Observing Equation (9.2) and (9.15), we can state that $T^l_{S_{31}^{11}} = T^l_{S_{11}^{10}} + 2\varepsilon(X_1^l - 1)$ for 2RRCSIs. This means that $T^l_{S_{31}^{11}}$ is bigger than $T^l_{S_{11}^{10}}$. Also, recalling γ and β from Lemma 3, we can rewrite Equations (9.3) and (9.16) as:

$$T^l_{S_{12}^{10}} = \begin{cases} \beta + \gamma & \text{if } (a+\gamma_1)X_1^l \leq \gamma \\ \beta + (a+\gamma_1)X_1^l & \text{if } (a+\gamma_1)X_1^l > \gamma \end{cases} \tag{9.23}$$

$$T^l_{S_{32}^{11}} = \begin{cases} \beta + \gamma + \gamma_1 + (a+\gamma_1+2\varepsilon)(X_1^l - 1) & \text{if } a \leq \gamma \\ \beta + a + \gamma_1 + (a+\gamma_1+2\varepsilon)(X_1^l - 1) & \text{if } a > \gamma \end{cases} \tag{9.24}$$

Clearly, Equation (9.23) and (9.24) resulted from the second argument of the *max* function of original equations. Since $X_1^l \geq 1$ and all other input data are nonnegative, $T^l_{S_{32}^{11}}$ is absolutely bigger than $T^l_{S_{12}^{10}}$ for 2RRCSIs. Note that the proof for 2RRCEIs is same. This completes the proof.

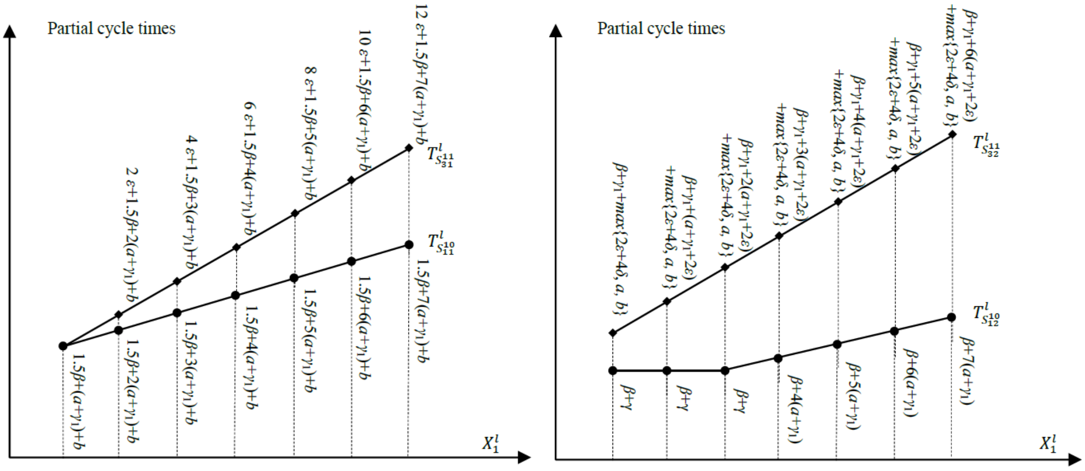


Fig. 9.6. Comparison of in-process and in-line inspection scenarios with respect to stop inspection strategy

The scheme of four feasible partial cycles is shown in Figure 9.6 in order to demonstrate their relationship in a graphical way. The figure shows that in-process inspection scenario is even more preferable as the number of rework processes increases towards infinity. Let us now extend the analysis to the case when $p_1 > \frac{1}{2}$. The other case, $p_2 > \frac{1}{2}$, can be treated in a similar fashion.

Theorem 8. If a transition inspection strategy leads to the optimal cycle under any one of the following conditions, it must be permitted to replace a cell with in-process inspection scenario with a cell with in-line inspection scenario:

1. $a + b < 2\delta$
   (a) $p_1 > \dfrac{min\{\delta, \gamma_1\} + 2\varepsilon}{2min\{\delta, \gamma_1\} + 2\varepsilon}$
   (b) $a + b + max\{\delta, \gamma_1\} + (a + \gamma_1 + min\{\delta, \gamma_1\} + 2\varepsilon)(\dfrac{1 - p_1}{p_1}) < 3\delta + ((a + \gamma_1)(\dfrac{1}{p_1} + \lfloor \dfrac{2\varepsilon + 4\delta}{a + \gamma_1} \rfloor) - (2\varepsilon + 4\delta))(1 - p_1)^{\lfloor \frac{2\varepsilon + 4\delta}{a + \gamma_1} \rfloor}$
2. $a + b > 2\delta$
   (a) $p_1 > (a + \gamma_1 + b + min\{\delta, \gamma_1\} + 2\varepsilon - 2\delta)(min\{\delta, \gamma_1\} + 2\varepsilon)$
   (b) $max\{\delta, \gamma_1\} + (a + \gamma_1 + min\{\delta, \gamma_1\} + 2\varepsilon)(\dfrac{1 - p_1}{p_1}) + max\{0, a - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\} < \delta + max\{0, b - (2\varepsilon + 4\delta)\} + ((a + \gamma_1)\lfloor \dfrac{max\{2\varepsilon + 4\delta, b\}}{a + \gamma_1} \rfloor +$

$$\frac{a+\gamma_1}{p_1} - max\{2\varepsilon+4\delta,b\})(1-p_1)^{\lfloor\frac{max\{2\varepsilon+4\delta,b\}}{a+\gamma_1}\rfloor}$$

Proof: We know that two cases may occur. In the first case, $T^*_{312} = T^l_{S^{12}_{31}}$ since $a+b < 2\delta$. Subcases 1.a and 1.b are also extracted from simplifying inequities $E(T^l_{S^{12}_{31}}) < E(T^l_{S^{10}_{11}})$ and $E(T^l_{S^{12}_{31}}) < E(T^l_{S^{10}_{12}})$, respectively. Note that $p_1 > \frac{min\{\delta,\gamma_1\}+2\varepsilon}{2min\{\delta,\gamma_1\}+2\varepsilon} > \frac{1}{2}$, and therefore the optimality of the transition inspection strategy does not give a guarantee that T^*_{312} is certainly less than T^*_{310}. The proof of the second case is similar. $T^*_{312} = T^l_{S^{12}_{32}}$ if $a+b < 2\delta$. Also, $E(T^l_{S^{12}_{32}}) < E(T^l_{S^{10}_{11}})$ and $E(T^l_{S^{12}_{32}}) < E(T^l_{S^{10}_{12}})$, result in subcases 2.a and 2.b. This completes the proof.

Now, there is an appropriate framework which helps us to find the best inspection scenario for each designed robotic rework cell. As a direct result from Theorem 7 and 8 together, bottleneck identification is also an important objective in order to compare different scenarios. This means that we should not apply a multi-function robot for a rework cell if it acts like a sensor installed into the machine and never perform inspection in transit. In more detail, the robot has to repeat inspection several times under this condition which often makes it the bottleneck of the rework cell.

9.7 Concluding Remarks

An analytical method for minimizing the partial cycle time of such cells has been developed for three different inspection scenarios: in-process, post-process and in-line. We have proven that it is possible to reach a steady state of the small-scale cells, which have a dynamic behaviour, and then maximize the expected throughput of associated cells. Comparing the two-machine robotic rework cell with the same robotic cell without a rework assumption, it has been realized that the performance of the backward cycle $S^{hk}_{i2}, i \in \{1,2,3\}, h \in \{1,2\}$ and $k \in \{0,1,2\}$ is improved due to the fact that the average time of producing a part is definitely increased. It has been proven that there is second-order stochastic dominancy relationship between feasible partial cycles of the in-process scenario, whilst it is possible to find absolute dominancy relationship between all feasible partial cycles of post-process (and also in-line) scenario. Furthermore, we have considered the problem of converting rework cells with in-process and post-process inspection scenarios into the rework cell with the in-line inspection scenario where a multi-function robot is responsible for inspection of the part in transit. Since the optimal cycle under in-process scenario always dominates the corresponding optimal cycle under post-process scenario, the problem has been reduced to only comparing in-process and in-line scenarios. The comparison has revealed that the inspection scenario must not be shifted from in-process to in-line if the multi-function robot first performs the inspection of the part and then transfers it to the next machine. The reason for this intuition is that the robot only acts like an inspection sensor under this condition. Therefore, it is not only the bottleneck of the cell, but also decreases partial throughput. Finally, we have proven that it is technically profitable to replace the in-process inspection scenario with the in-line inspection scenario if the robot performs inspection of the part in transit. Nonetheless, the performance improvement depends on the probability that the part needs rework after each time processing on the production machine. There are a couple of further research topics which will be studied in the future works. Firstly, we can consider the non-identical parts case in which some of the parts need ”Start of Line” testing plan, whereas the rest need ”End of Line” testing plan. Also, a secondary problem for this case is to determine the sequence of non-identical parts to be completed. Secondly, further work should be done to consider the limited number of permitted rework processes for a particular part. If the part could not pass inspection process even after this number of rework processes, it must be considered as scrap, not as a final product.

Part IV

Conclusions

Chapter 10

10.1 Conclusions

As mentioned in the Part I of this thesis, robotic cells are one of the sophisticated application areas of flowshops that have gained increasingly importance in production engineering. They are classified into two categories from a quality control point of view: (*i*) robotic cells without an inspection process and (*ii*) robotic cells with an inspection process. In regard with the second category, it should be mentioned that there exist advanced robots that are able not only to act as a material handling device but also to inspect the part in transit between machines. Such a kind of robots and cells in which these robots are applied are called Multi-Function Robots (MFRs) and Multi-Function Robotic Cells(MFRCs), respectively. A real-life example of this environment is a robotic arm which is equipped with a Grip-Gage-Go gripper. The robotic arm, or MFR, can measure the thickness of the shaft in transit between machines. Part II of the thesis included a description of cells with the inspection processes where stochastic data are only recorded by the robot in an independent computer, whereas Part III is devoted to stochastic scheduling problems without any sort of relaxations.

This thesis is one of the pioneering studies initiating MFRC in in the general area of scheduling and performance evaluation. Consequently, before proceeding with different types of MFRCs in Part II, it is needed to determine their robotic cell in origin. Chapter 3 is related to robotic cells with a hub machine and shows that these types of robotic cells are one of the origins of MFRCs. This chapter introduces a hub reentrant robotic cell consisting of a group of m production machines. In order to produce a completed part in a hub reentrant robotic cell, a chain of $m-1$ secondary operations are performed by $m-1$ different machines, and a hub machine is alternately visited for m primary operation so that parts must reenter the hub machine after any one of secondary operations. Although the robot employed in a hub reentrant robotic cell is only a material handling device, the processing route of the part in the cell is similar to the processing route of parts in MFRCs. The reason behind this intuition is that secondary operations are completely different in nature, whereas primary operations often have same nature. For example, it can be assumed that all primary operations in the hub machine are inspection processes. This operational assumption reveals similarities between a hub reentrant robotic cell and a MFRC. More precisely, if we remove the hub machine in the hub reentrant robotic cell and instead replace the robot with a MFR, then the cell will be converted to a MFRC. An optimization methodology for hub reentrant robotic cells is introduced in Chapter 3. We have provided not only the lower bound of the cycle time but also the cycle time of a proposed cycle, namely the dominant cycle. We have demonstrated some outcomes about optimality for this cycle and proven that it is an appropriate option for the hub reentrant robotic cell. The result of this chapter is advantageous to many industries such as wafer fabrication, painting and electroplating lines.

The analysis in Chapter 4 enables insightful evaluation of the productivity improvements of MFRs in small-scale MFRCs with deterministic data. We limited our study in this chapter to a MFR which only measures the thickness of the part and records results in an independent computer. Accordingly, the processing route is fixed although the MFR performs the inspection process of the part. Under this condition, we presented a methodology to maximize the production rate of a MFR operating within a rotationally arranged robotic cell. Considering the free pickup criterion, the cycle time formulas are initially developed for cells where a MFR interacts with either two or three machines. Two and six feasible cycles have been developed for two- and three-machine MFRCs with the free pickup criterion, and the optimality regions of these cycles and their formulas have been determined. In brief, through this chapter it has been found there is no unique optimal cycle for MFR movement when we change given parameters. To state the matter differently, it should be noted any one of the cycles has the chance of obtaining optimality considering different values of given data $\varepsilon, \delta, P_1, P_2, P_3, \gamma_0, \gamma_1, \gamma_2, \gamma_3$. Hence, it is enough to check whether the cycle meets the optimality conditions or not. Note that we also extended results to the no-wait pick up scenario in which all parts must be processed from the input buffer to the output buffer, without any interruption either on or between machines. A Search Algorithm is constructed for this type of pickup criterion. The mechanism of the algorithm to reach the optimal cycle in trivial time is to define the set of feasible cycles, and then find the optimal cycle using two For Loops. Anyone of cycles is stopped when an infeasible activity occurs in its activity route.

Chapter 5 presents a generalization of the results in Chapter 4 so that it is associated with large-scale MFRCs with deterministic data. Likewise, the processing route is fixed although the MFR performs the inspection process of the part. Only, the size of the problem is increased and this makes it more complex to analyse this case. The problem is modelled as a Travelling Salesman Problem to give computational benefits with respect to the existing solution methods. Then, the lower bound for the cycle time is deduced in order to measure the productivity gain of two practical production permutations, namely uphill cycle $\Pi_{V_{(m)}}$ and downhill cycle $\Pi_{D_{(m)}}$. As a design problem, a preliminary analysis initially identifies the regions where the productivity gain of a MFRC is more than that of the corresponding SFRC. The comparison of SFRCs and MFRCs has given the result that MFRCs improve throughput rate when $\Pi_{V_{(m)}}$ is optimal. However, the use of MFRCs instead of SFRCs is a wasted expense when $\Pi_{D_{(m)}}$ is optimal. Furthermore, the productivity of using MFRC against that of SFRC expressed by PO_k for cycles Π_V and Π_D in order to establish a practical framework for solving multi-objective scheduling problems.

Chapter 6, which is the last chapter of Part II, is a supplementary chapter which discussed about operational flexibility in MFRCs. The main assumption in Chapter 6 in that the inspection time of MFR is flexible. It should be stressed that neither the inspection time nor the processing route is a stochastic variable. We assumed a class of grippers which is able to perform a breakable operation on a part in transit from I to O of a MFRC dealing with two tandem machines. Assuming "stop resume" processing mode for the MFR, it continues processing of the part when it is reloaded to the robot with no loss in time. At the starting point, the best proportions of the unique operation of the MFR to be done between I, M_1, M_2 and O is determined graphically. Then, the cycle time of two one-unit cyclic solution have been obtained using this graphical representation of the operation on the MFR, and following that the optimality region of each one of them has been determined when dealing with the free pickup criterion . This line of thought brings us to the result that the first one-unit cycle is more productive for cells with short processing time on machines, and the second one-unit cycle is more productive for cells with time-consuming processing time on machines. It has been proven that the cycle time of any n-unit cycle is a convex combination of cycle times of two one-unit cycle, and hence the optimal one-unit is the global optimal cycle for the MFRC. Finally, Chapter 6 provides a comprehensive discussion of feasibility regions of two one-unit cycles under interval and no-wait pickup criteria.

Part III of this thesis is deviated from deterministic given parameters towards the impact of stochastic given parameters on the regions of optimality in MFRCs. Accordingly, previous three chapters are precedents for this part of the thesis where the processing route of each part can be modified based on its inspection results. Part III includes Chapters 7, 8 and 9 that highlight the idea of applying three different types of inspection in robotic cells: post-process, in-process and in-line.

The probability of the deadlock occurrence is likely to be large, especially for robotic cells with a post-process inspection scenario. This motivated us to focus on resolution of deadlock in Chapter 7. Avoidance and recovery policies have been developed to overcome deadlocks originated from a robotic cell with a post-process inspection scenario. Also, an analytical method for minimizing the partial cycle time and cost of cells with the post-process inspection is developed in this chapter. The novelty of this study is that we have shown there are two control policies in term of deadlock: 1) it is possible to avoid the occurrence of deadlock using an avoidance policy. This policy prevents existence of potential deadlocks. 2) we can allow a deadlock to occur, and then resolve it during the online implementation of the robot move cycles using a recovery policy. Considering these two control policies, we have given a mathematical proof that the avoidance policy minimize the cost of reworking per-unit while the recovery policy decreases the expected cycle time. Also, we have compared cells with the post-process inspection with cells without this additional step. This has made it clear that the performance of the second partial cycle is improved due to the fact that the average time of producing a part is definitely increased in this case.

Chapter 8 is related to robotic cells with an in-process inspection. Under this condition, the inspection of the part processed by a machine is performed by a multiple-sensor inspection system installed into this machine. The overall structure of this chapter is as follows: 1) we initially presented a heuristic method that converts a multiple-sensor inspection system into a single-sensor inspection system. 2) we presented a proof of dynamicity of the problem of determining the optimal one-unit cycle for cells with more than two machines. In contrast, we have proven that the problem is not dynamic for two-machine robotic cells, and the pickup criterion has no impact on the result of the theorem. 3) Based on three kinds of stochastic order relations, we developed an analytical method to find the dominancy regions of two feasible one-unit cycles, and then we extend the result to interval and no-wait pick up scenarios as two well-solved classes. With regard to the interval pickup criterion, we concluded that there is no guarantee that the second cycle (in comparison with the first cycle) will be an optimal cycle when it is feasible. Nonetheless, it is enough to check whether the second cycle satisfies feasibility conditions for the cell with no-wait pickup criterion to conclude that it is the optimal cycle.

Chapter 9 covers a wide range of inspection scenarios. It starts with results in post-process and in-process inspection scenarios and extend this results to an in-line inspection scenario. An important point about the in-line inspection scenario is that a MFR is in charge of performing inspection process. Therefore, similar to Part II, we can assume that cells with either post-process or in-process inspection scenarios are also the origin of MFRCs. We have considered the problem of converting cells with in-process and post-process inspection scenarios into a MFRC in which a MFR is responsible for inspection of the part in transit (in-line inspection). Since the optimal cycle under in-process scenario always dominates the corresponding optimal cycle under post-process scenario, the problem has been reduced to only comparing a cell with an in-process and a MFRC. The comparison has revealed that the inspection scenario must not be shifted from in-process to in-line if the MFR first performs the inspection of the part and then transfers it to the next machine. The reason for this intuition is that the MFR only acts like an inspection sensor under this condition. Therefore, it is not only the bottleneck of the cell, but also decreases partial throughput. Finally, we have proven that it is technically profitable to replace a cell with an in-process inspection scenario with a MFRC if the robot performs inspection of the part in transit.

One of the paramount contributions of this dissertation is the establishment an appropriate framework for different types of MFRCs. This framework can help companies which are enthusiastic about using MFRs in fully automated manufacturing systems. In other words, using this framework before employing MFRs in the production line, they can find out whether this option can increase the productivity or not. This would assist manufacturers in deciding which type of robotic cell is better for any one of the part processing routes.

10.2 Future Work

There is a number of intriguing research points which may be pursued later. Accordingly, the objective of this section is to mention some of them:

(i) An interesting future research direction is to study the problem of sequencing non-identical parts in MFRCs. A problem for this case is to schedule the robot move cycle and sequence the parts processing order concurrently. This small difference in the problem appears to have major impact on the solution approach due to the problem complexity. It is easy to prove that associated problem is NP-hard in the strong sense (Brauner et al., 2003). Therefore, we recommend developing a mixed-integer linear programming (MILP) model. This MILP model can be enhanced using a set of dominance constraints. In addition, a Cross Entropy (CE) algorithm can be developed to find near-optimal solutions of large- scale scheduling problems in a short running time.

(ii) We have concentrated our attention on simplest cases of post-process, in-process and in-line inspection scenarios which can be classified as a closed-loop inspection. Under this condition, parts are cycled between successive a processing machine and test device (this device can be an independent machine, a set of sensors or a robot) until deemed acceptable. However, the inspection pattern may be more complicated since there is a great number of inspection patterns when there are more than two machines. Let us give some examples of complex inspection patterns. Assume that M_3 is an solo inspection machine for a four-machine robotic cell. The inspection result of M_3 follows this distribution: (a) the part is fully-failed and needs reprocessing on both M_1 and M_2 with the associated probability 0.1, (b) the part is semi-failed and only needs reprocessing on M_2 with the associated probability 0.2, (c) the part moves forward to M_4 with the associated probability 0.6, (d) the part does not even need processing on M_4 and it should be dropped off directly at O with the associated probability 0.1.

(iii) With more precise inspection sensors and computers available, further work can be done to consider a limited number of permitted rework processes for any particular part. Consequently, the part must be considered as scrap if it can not pass inspection process after this number of rework processes. Clearly, this will probably make it more complicated to find the optimal cycle. The reason behind this intuition is that the status of the part under this condition is increased from 2 into 3: Accepted, needs rework, scrap. It is necessary to mention that this future research direction has some similarities with the previous one. In both cases, the inspection results are a variable with more than two possible outcomes. In other words, the distribution of this variable is equivalent to the Bernoulli distribution for $k > 2$ possible disjunctive outcomes. This generalization of the Bernoulli distribution is named *Bernoulli Scheme* or *Bernoulli Shift* in mathematics, and it is defined as a discrete-time stochastic process in which the random variable takes on one of k distinct possible values, with the outcome i occurring with corresponding probability $p_i, \forall i \in \{1, ..., k\}$.

(iv) Finally, future research direction can extend the work of this thesis by generalizing the results to m-machine MFRCs with post-process, in-process, and in-line inspection scenarios as a dynamic system. Since the dynamic behaviours of such a system is particularly the most important and interesting future research direction, we give a separate indication of how appropriate is a dynamic programming for this case. The rest of this section provides a two-phase dynamic programming model, which needs completion in the future, to justify the importance of this situation.

Now, our goal is to detail the progress we have made over the last year to build a stochastic dynamic programming model of large-scale robotic cells. This goal is due a dynamicity in the scheduling problem for large-scale robotic cells with an inspection process. More precisely, as mentioned in Theorem 1 of Chapter 8, all robotic rework cells with over two machines have dynamic behaviour that makes it more complex to analyse them. Downhill cycles can be used as an example that disproves the statements that any large-scale robotic cell with an inspection process can reach a steady-state from an empty cell. For example, we have provided this counterexample for an m-machine robotic cell with an inspection process regardless of the type of its inspection scenario. The downhill cycle for this case is $A_0 A_m, A_{m-1}, ..., A_2, A_1$, and this permutation shows that the cell never reaches an state in which there is only one part is processing there. Therefore, at least for this cycle, we cannot claim that there is no relationship between the optimal partial cycle in the current state and the number of rework performed on the part in the previous partial cycle. This counterexample is enough to limit our analysis to solution techniques for problems with dynamic behaviour.

It should be stressed that we need to design a method for sequentially choosing the next activity of the robot until one robot move cycle be completed. Therefore, we have two alternatives: 1. A *non-adaptive* solution method: It specifies an entire solution in advance so that we cannot make any further changes as the cycle implementation is started. In more detail, a non-adaptive solution method is only an unchangeable permutation of activates to be executed. Clearly, such a method that have an unchangeable structure may have some drawbacks. 2. an *adaptive* solution method: It makes decisions in a dynamic fashion in reaction to the state of machines already occupied by parts. It is more complex to implement an adaptive solution method, but its final solution is predominantly better performance than a non-adaptive one (Dean et al., 2008).

Dynamic programming is one of adoptive solution methods often used for solving complex problems with dynamic behaviour. This stepwise method considers a related version of the scheduling problem by dividing it into a set of sub-problems, and then determining the optimal sub-solution of these sub-problems. In each stage, the optimal sub-solution of the sub-problem is created from the solution of the previous sub-problem, and finally sub-problems should be nested recursively inside the original problem as applicable. Dynamic programming algorithm has a polynomial complexity that guarantees a reasonable running time. More specifically, the dynamic programming can solve a robotic cell scheduling problem with an inspection process in $O(m^3)$ time.

We need to clearly state the two-phase approach for stochastic dynamic programming of robotic cells with an inspection process. In the first phase, we must use a control policy to guarantee that control rules meet deadlock-related constraints. Then, in the second phase, we must add performance rules based on the dynamic programming algorithm to minimize the partial cycle time. Let us start with the first phase. We know that control rules are changed if we have robotic cells with different inspection scenarios. However, regardless of the type of the inspection, the robot always acts as server and other components of the cell always need its service as shown in Figure 10.1. Note that components of the cell vary based on the applied inspection scenario as follows: components of the cell with the post-process inspection scenario are buffers, production machines, and inspection machines, whereas these components of both cells with in-process and in-line inspection scenarios are buffers, production machines, and sensors.

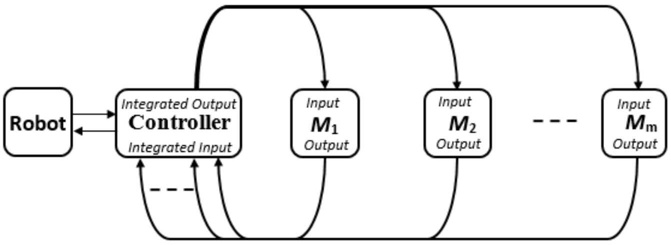


Fig. 10.1. The overall control policy of a robotic cell with any type of inspection process scenario

In Figure 10.1, each component is represented by a node. An important characteristic of the node is that it can receive control messages as an input and send central controller information related to its current state as an output. The central controller receive information related to all components and the robot before each time that it is instructed to send out a new control message to components and the robot. This makes it easy for the central controller to choose which robot activity between requested activities are feasible. In other words, based on the control rules, it provides a list of forbidden activities before determining the next activity of the robot. This can be considered as the result of the first phase of our approach, and also as an input for the second phase of the approach in which the controller should select the optimal activity along all feasible activities to be implemented.

It is noteworthy that Figure 10.1 only shows the first layer of the control policy which can be detailed even more. The second layer of the control policy can show specific control policies for each machine $j, \forall j \in \{1, 2, ..., m\}$ considering a set of inner and outer transitions. We say that the type of transition is inner if changing current state of the machine (or the robot) is independent from the central controller. An example of inner transition is the situation in which a production machine j is finished the part processing so that its state is changed from *Processing* (P_j) to *Completed* (C_j) and this can be shown with Ω. Another example of inner transition is the final result of the inspection sensor attached to the production machine in an in-process inspection scenario. If the result shows that the part does not need reprocessing then the state is changed from P to C. We graphically show an outer transition with a solid line. In contrast with the outer transition, an outer transition is not independent from outer inputs received from the central controller. Giving an example, the state of a busy machine j is changed from F to Unoccupied (U_j) if and only if the robot be instructed to remove the part from it. We can graphically show an inner transition with a dashed line. We also can define a set A of robot activities as $M_j - M_k, \forall j \& k \in \{0, 1, 2, , m+1\}$. For example, $M_1 - M_2$ refer to an outer transition in which the robot unload M_1 and load the part on M_2. Recall that M_0 and M_{m+1} can be named I and O, respectively. *Deadlock* of M_j (referred to D_j) is taken into account here.

Before proceeding with the post-process inspection scenario let us first analyse simpler cases. Considering aforementioned notations, an examples of the second layer of state transition for a cell with in-process inspection scenario is depicted in Figure 10.2. This example is related to machine $j, \forall j \in \{1, 2, ..., m\}$ which is equipped with a set of sensors, and therefore it is able to process and inspect the part simultaneously. The diamond in this figure indicates a question (or a decision) which is used when there are two options (Yes/No) for the inspection result. "No" is associated with the probability of the rework being needed after inspection of the part on M_j and this is an inner transition. "Yes" is associated with the probability of the transformation of the part to M_{j+1} if it is deemed acceptable and this is an outer transition.

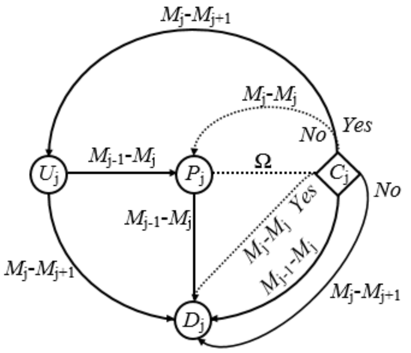


Fig. 10.2. The second layer of state transition for M_j with in-process inspection

It is vital to mention that two following types of error may occur when sensors perform inspection process. More precisely, the result of no inspection process is 100% certain since the inspection result follows a Bernoulli probability distribution and therefore there is always a chance of drawing an incorrect result.

1. *Type one error*: Sensors make a type one error when the part should be accepted but the result of inspection shows it needs reprocessing on M_j.

2. *Type two error*: Sensors make this error when the part needs reprocessing on M_j but sensors fail to reject it and consequently they deems part acceptable.

Likewise, an example of the second layer of state transition for a cell with in-line inspection scenario is depicted in Figure 10.3. The main difference between this figure and the previous one is that the *Robot* (R) is in charge of inspection process and therefore it is shown by a two-option diamond. "No" is associated with the probability of the rework on M_j being needed after inspection of the part on R. "Yes" is associated with the probability that R completes the transformation of the part to M_{j+1} if the part is deemed acceptable. Note that, once again, we have two similar types of error for this case.

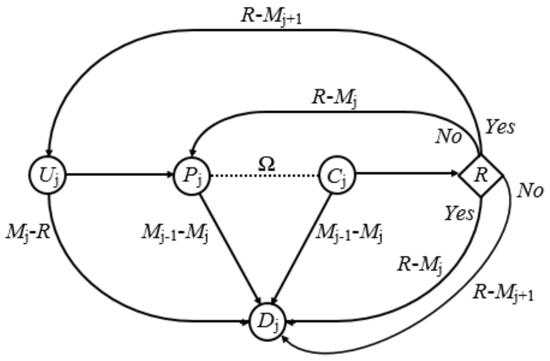


Fig. 10.3. The second layer of state transition for M_j with in-line inspection scenario

The state transition is more complex for post-process inspection scenario as shown in Figure 10.4. The reason behind this intuition is that we should show this state for a couple of machines: a production machine and the next machine in the production line which is certainly an inspection machine. In this figure, C_j connects the network of the first machine with the network of the second machine and therefore we can remove U_{j+1}.

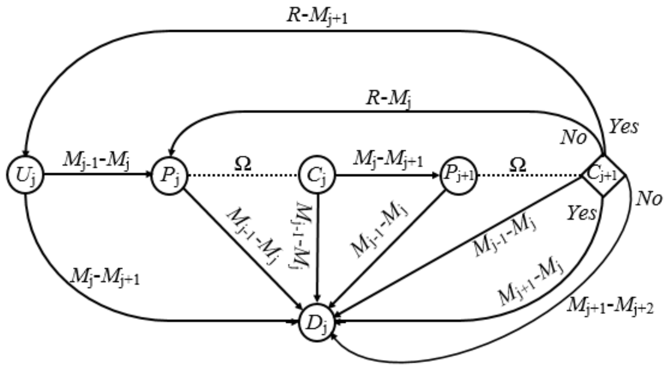


Fig. 10.4. The second layer of transition for M_j and M_{j+1} with post-process inspection

The second phase of this algorithm is part of future research work. In this phase, we are planing to add performance rules, which are based on a dynamic programming algorithm, to minimize the partial cycle time.

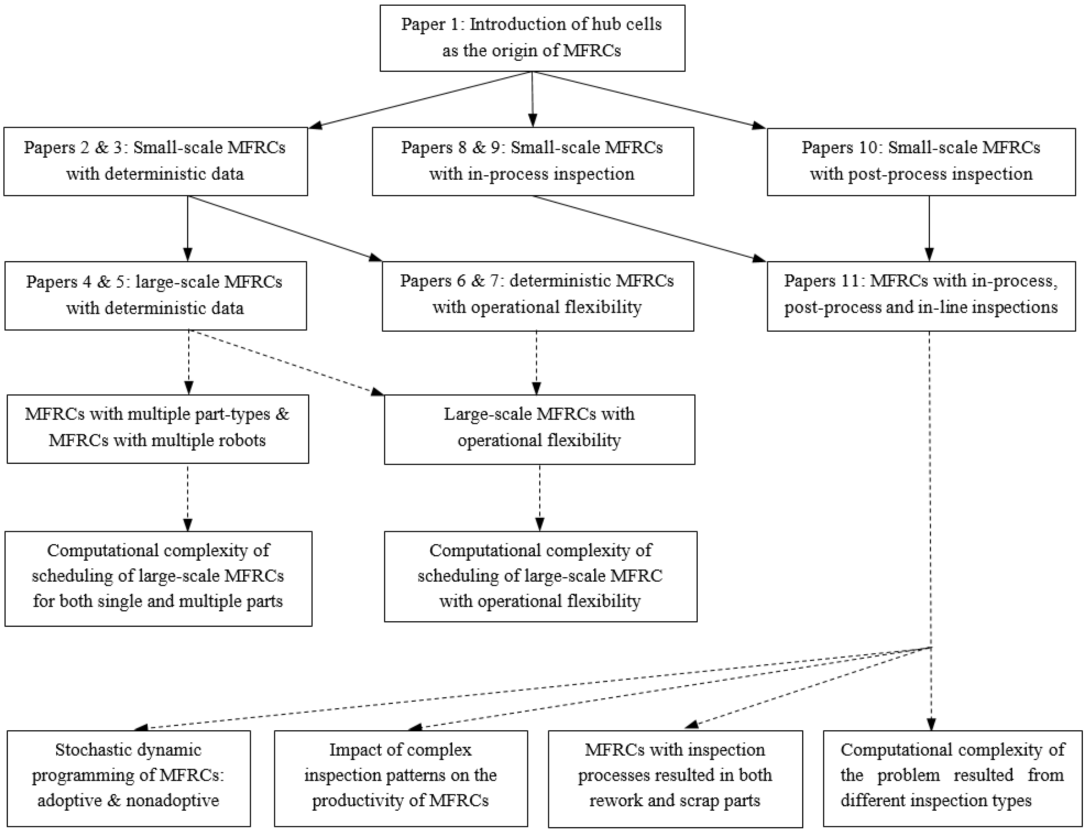

Paper 1: Introduction of hub cells as the origin of MFRCs
Papers 2 & 3: Small-scale MFRCs with deterministic data
Papers 8 & 9: Small-scale MFRCs with in-process inspection
Papers 10: Small-scale MFRCs with post-process inspection
Papers 4 & 5: large-scale MFRCs with deterministic data
Papers 6 & 7: deterministic MFRCs with operational flexibility
Papers 11: MFRCs with in-process, post-process and in-line inspections
MFRCs with multiple part-types & MFRCs with multiple robots
Large-scale MFRCs with operational flexibility
Computational complexity of scheduling of large-scale MFRCs for both single and multiple parts
Computational complexity of scheduling of large-scale MFRC with operational flexibility
Stochastic dynamic programming of MFRCs: adoptive & nonadoptive
Impact of complex inspection patterns on the productivity of MFRCs
MFRCs with inspection processes resulted in both rework and scrap parts
Computational complexity of the problem resulted from different inspection types

Fig. 10.5. A hierarchy between problems considered in this thesis and open problems

At the end of this section, a hierarchy is presented in Figure 10.5 to help readers who are interested in extending results of the thesis. Solid lines in this figure recall outcomes of the thesis, as mentioned earlier in Figure 2.1. However, Figure 10.5 shows that there are many rooms for improvement of results of the thesis. Dashed line in this figure presents some ideas for extension of the results of this thesis in the future.

Bibliography

M.M.S. Abdulkader, M.M. ElBeheiry, N.H. Afia, and A.K. El-Kharbotly. Scheduling and sequencing in four machines robotic cell: Application of genetic algorithm and enumeration techniques. *Ain Shams Engineering Journal*, 4(3):465 – 474, 2013.

A. Agnetis. Scheduling no-wait robotic cells with two and three machines. *European Journal of Operational Research*, 123(2):303 – 314, 2000.

M.S. Akturk, H. Gultekin, and O.E. Karasan. Robotic cell scheduling with operational flexibility. *Discrete Applied Mathematics*, 145(3):334 – 348, 2005.

D. Alcaide, C. Chu, V. Kats, E. Levner, and G. Sierksma. Cyclic multiple-robot scheduling with time-window constraints using a critical path approach. *European Journal of Operational Research*, 177(1):147 – 162, 2007.

M.M. Aldaihani and M. Savsar. A stochastic model for the analysis of a two-machine flexible manufacturing cell. *Computers and Industrial Engineering*, 49 (4):600 – 610, 2005.

H. Allaoui, A. Artiba, S.E. Elmaghraby, and F. Riane. Scheduling of a two-machine flowshop with availability constraints on the first machine. *International Journal of Production Economics*, 99(12):16 – 27, 2006.

M.A. Ayub, A.B. Mohamed, and A.H. Esa. In-line inspection of roundness using machine vision. *Procedia Technology*, 15:807 – 816, 2014.

T.P. Bagchi, J.N.D. Gupta, and C. Sriskandarajah. A review of tsp based approaches for flowshop scheduling. *European Journal of Operational Research*, 169(3):816 – 854, 2006.

F. Basile, F. Caccavale, P. Chiacchio, J. Coppola, A. Marino, and D. Gerbasio. Automated synthesis of hybrid petri net models for robotic cells in the aircraft industry. *Control Engineering Practice*, 31:35 – 49, 2014.

G.D. Batur, O.E. Karasan, and M.S. Akturk. Multiple part-type scheduling in flexible robotic cells. *International Journal of Production Economics*, 135(2): 726 – 740, 2012.

K. Bernd, K. Carsten, and S. Thorsten. Adaptive robot based reworking system. In Low Kin Huat, editor, *In Industrial Robotics: Programming, Simulation and Applications*, chapter 17, pages 341–348. InTech Education and Publishing, Vienna, Austria, 2006.

N. Brauner. Identical part production in cyclic robotic cells: Concepts, overview and open questions. *Discrete Applied Mathematics*, 156(13):2480 – 2492, 2008.

N. Brauner and G. Finke. Optimal moves of the material handling system in a robotic cell. *International Journal of Production Economics*, 74(13):269 – 277, 2001.

N. Brauner, G. Finke, and W. Kubiak. Complexity of one-cycle robotic flow-shops. *Journal of Scheduling*, 6(4):355–372, 2003.

N. Brauner, G. Finke, V. Lehoux-Lebacque, C. Potts, and J. Whitehead. Scheduling of coupled tasks and one-machine no-wait robotic cells. *Computer and Operations Research*, 36(1):301 – 307, 2009.

B. Cai, S. Huang, D. Liu, and G. Dissanayake. Rescheduling policies for large-scale task allocation of autonomous straddle carriers under uncertainty at automated container terminals. *Robotics and Autonomous Systems*, 62(4):506 – 514, 2014.

W-K. Chan, J. Yi, S. Ding, and D. Song. Optimal scheduling of k-unit production of cluster tools with single-blade robots. In *IEEE International Conference on Automation Science and Engineering*, pages 335–340, Aug 2008.

C-C. Chang, T-H. Wu, and C-W. Wu. An efficient approach to determine cell formation, cell layout and intracellular machine sequence in cellular manufacturing systems. *Computers and Industrial Engineering*, 66(2):438 – 450, 2013.

A. Che and C. Chu. Multi-degree cyclic scheduling of two robots in a no-wait flowshop. *IEEE Transactions on Automation Science and Engineering*, 2(2): 173–183, 2005.

A. Che and C. Chu. Multi-degree cyclic scheduling of a no-wait robotic cell with multiple robots. *European Journal of Operational Research*, 199(1):77 – 88, 2009.

A. Che, C. Chu, and F. Chu. Multicyclic hoist scheduling with constant processing times. *IEEE Transactions on Robotics and Automation*, 18(1):69–80, 2002.

M. Dawande, H.N. Geismar, S.P. Sethi, and C. Sriskandarajah. Sequencing and scheduling in robotic cells: Recent developments. *Journal of Scheduling*, 8(5): 387–426, 2005.

M. Dawande, H.N. Geismar, S.P. Sethi, and C. Sriskandarajah. *Throughput Maximization in Robotic Cells.* Springer series in International Series in Operations Research and Management Science. Springer, 2007.

M. Dawande, M. Pinedo, and C. Sriskandarajah. Multiple part-type production in robotic cells: Equivalence of two real-world models. *Manufacturing and Service Operations Management*, 11(2):210–228, 2009.

M. Dawande, H.N. Geismar, M. Pinedo, and C. Sriskandarajah. Throughput optimization in dual-gripper interval robotic cells. *IIE Transactions*, 42(1):1–15, 2010.

B.C. Dean, M.X. Goemans, and J. Vondrk. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Mathematics of Operations Research*, 33(4): 945–964, 2008.

X.D. Diao, S.X. Zeng, and V.W.Y. Tam. Development of an optimal trajectory model for spray painting on a free surface. *Computers and Industrial Engineering*, 57(1):209 – 216, 2009.

I.G. Drobouchevitch, S.P. Sethi, J.B. Sidney, and C. Sriskandarajah. A note on scheduling multiple parts in two-machine dual-gripper robot cells: heuristic algorithm and performance guarantee. *International Journal of Quantity Management*, 10(1):297–314, 2004.

I.G. Drobouchevitch, S.P. Sethi, and C. Sriskandarajah. Scheduling dual gripper robotic cell: One-unit cycles. *European Journal of Operational Research*, 171 (2):598 – 631, 2006.

I.G. Drobouchevitch, H.N. Geismar, and C. Sriskandarajah. Throughput optimization in robotic cells with input and output machine buffers: A comparative study of two key models. *European Journal of Operational Research*, 206(3):623 – 633, 2010.

I. Edinbarough, R. Balderas, and S. Bose. A vision and robot based on-line inspection monitoring system for electronic manufacturing. *Computers in Industry*, 56 (89):986 – 996, 2005.

A. Ferrolho and M. Crisostomo. Intelligent control and integration software for flexible manufacturing cells. *IEEE Transactions on Industrial Informatics*, 3(1): 3–11, 2007.

A. Ferrolho and M. Crisostomo. Resource allocation in free-choice multiple reentrant manufacturing systems based on machine-job incidence matrix. *IEEE Transactions on Industrial Informatics*, 7(1):105–114, 2011.

M. Foumani and K. Jenab. Cycle time analysis in reentrant robotic cells with swap ability. *International Journal of Production Research*, 50(22):6372–6387, 2012.

M. Foumani and K. Jenab. Analysis of flexible robotic cells with improved pure cycle. *International Journal of Computer Integrated Manufacturing*, 26(3):201–215, 2013a.

M. Foumani and K. Jenab. An operation-oriented analysis of hybrid robotic cells. *International Journal of Robotics and Automation*, 28(2):123–128, 2013b.

M. Foumani, M.Y. Ibrahim, and I. Gunawan. Cyclic scheduling in small-scale robotic cells served by a multi-function robot. In *39th IEEE Annual Conference of Industrial Electronics Society (IECON)*, pages 4362–4367, Nov 2013a.

M. Foumani, M.Y. Ibrahim, and I. Gunawan. Cyclic production for robotic cells served by multi-function robots with resumable processing regime. In *IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 551–555, Dec 2013b.

M. Foumani, M.Y. Ibrahim, and I. Gunawan. Scheduling dual gripper robotic cells with a hub machine. In *IEEE International Symposium on Industrial Electronics (ISIE)*, pages 1–6, 2013c.

M. Foumani, I. Gunawan, and M.Y. Ibrahim. Scheduling rotationally arranged robotic cells served by a multi-function robot. *International Journal of Production Research*, 52(13):4037–4058, 2014.

M. Foumani, I. Gunawan, K. Smith-Miles, and M.Y. Ibrahim. Notes on feasibility and optimality conditions of small-scale multifunction robotic cell scheduling problems with pickup restrictions. *IEEE Transactions on Industrial Informatics*, 11(3):821–829, 2015a.

M. Foumani, K. Smith-Miles, I. Gunawan, and A. Moeini. Stochastic scheduling of an automated two-machine robotic cell with in-process inspection system. In *45th International Conference on Computers and Industrial Engineering*, pages 1–8, Oct 2015b.

G. Galante and G. Passannanti. Minimizing the cycle time in serial manufacturing systems with multiple dual-gripper robots. *International Journal of Production Research*, 44(4):639–652, 2006.

J. Gao, L. Sun, L. Wang, and M. Gen. An efficient approach for type {II} robotic assembly line balancing problems. *Computers and Industrial Engineering*, 56 (3):1065 – 1080, 2009.

H.N. Geismar and M. Pinedo. Robotic cells with stochastic processing times. *IIE Transactions*, 42(12):897–914, 2010.

H.N. Geismar, S.P. Sethi, J.B. Sidney, and C. Sriskandarajah. A note on productivity gains in flexible robotic cells. *International Journal of Flexible Manufacturing Systems*, 17(1):5–21, 2005.

H.N. Geismar, M. Pinedo, and C. Sriskandarajah. Robotic cells with parallel machines and multiple dual gripper robots: a comparative overview. *IIE Transactions*, 40(12):1211–1227, 2008.

H.N. Geismar, M. Dawande, and C. Sriskandarajah. Productivity improvement from using machine buffers in dual-gripper cluster tools. *IEEE Transactions on Automation Science and Engineering*, 8(1):29–41, 2011.

N. Geismar, U.V. Manoj, A. Sethi, and C. Sriskandarajah. Scheduling robotic cells served by a dual-arm robot. *IIE Transactions*, 44(3):230–248, 2012.

N. Geren and A. Redford. Cost and performance analysis of a robotic rework cell. *International Journal of Production Economics*, 58(2):159 – 172, 1999.

M. Givehchi, A. Ng, and L. Wang. Evolutionary optimization of robotic assembly operation sequencing with collision-free paths. *Journal of Manufacturing Systems*, 30(4):196 – 203, 2011.

H. Gultekin, M.S. Akturk, and O.E. Karasan. Scheduling in robotic cells: process flexibility and cell layout. *International Journal of Production Research*, 46(8): 2105–2121, 2008.

H. Gultekin, O.E. Karasan, and M.S. Akturk. Pure cycles in flexible robotic cells. *Computers and Operations Research*, 36(2):329 – 343, 2009.

H. Gultekin, M.S. Akturk, and O.E. Karasan. Bicriteria robotic operation allocation in a flexible manufacturing cell. *Computers and Operations Research*, 37 (4):779 – 789, 2010.

N.G. Hall, H. Kamoun, and C. Sriskandarajah. Scheduling in robotic cells: Classification, two and three machine cells. *Operations Research*, 45(3):421–439, 1997.

Nicholas G. Hall, Hichem Kamoun, and Chelliah Sriskandarajah. Scheduling in robotic cells: Complexity and steady state analysis. *European Journal of Operational Research*, 109(1):43 – 65, 1998.

I. Ioachim, E. Sanlaville, and M. Lefebvre. The basic cyclic scheduling model for robotic flow shops. *INFOR*, 39(3):257–277, 2001.

K. Jenab, M. Foumani, A.Sarfaraz, M. Rajai, and P. Weinsierm. Operation and configuration-based analysis of dual-gripper robotic cells. *International Journal of Agile Manufacturing*, 12(1):1–12, 2012.

F. Jolai, M. Foumani, R. Tavakoli-Moghadam, and P. Fattahi. Cyclic scheduling of a robotic flexible cell with load lock and swap. *Journal of Intelligent Manufacturing*, 23(5):1885–1891, 2012. ISSN 0956-5515.

H. Kamoun, N.g. Hall, and C. Sriskandarajah. Scheduling in robotic cells: heuristics and cell design. *Operations Research*, 47(2):821–835, 1999.

S. Kayaligil and S. Ozlu. Loading of pallets on identical {CNC} machines with cyclic schedules. *Computers and Industrial Engineering*, 42(24):221 – 230, 2002.

S. Keating and N. Oxman. Compound fabrication: A multi-functional robotic platform for digital design and fabrication. *Robotics and Computer-Integrated Manufacturing*, 29(6):439 – 448, 2013.

T-K Kim, C. Jung, and T-E Lee. Scheduling start-up and close-down periods of dual-armed cluster tools with wafer delay regulation. *International Journal of Production Research*, 50(10):2785–2795, 2012.

W. Kubiak, X.C.L Sheldon, and Y. Wang. Mean flow time minimization in reentrant job shops with a hub. *Operations Research*, 44(5):764–776, 1996.

S.V. Kumar, V.G.S. Mani, and N. Devraj. Production planning and process improvement in an impeller manufacturing using scheduling and {OEE} techniques. *Procedia Materials Science*, 5:1710 – 1715, 2014.

E. Langereis, S.B.S. Heil, H.C.M. Knoops, W. Keuning, M.C.M. van de Sanden, and W.M.M. Kessels. In situ spectroscopic ellipsometry as a versatile tool for studying atomic layer deposition. *Journal of Physics D: Applied Physics*, 42(7): 1 – 19, 2009.

H-Y. Lee and T-E. Lee. Scheduling single-armed cluster tools with reentrant wafer flows. *IEEE Transactions on Semiconductor Manufacturing*, 19(2):226–240, 2006.

T-E Lee. A review of scheduling theory and methods for semiconductor manufacturing cluster tools. In *Simulation Conference, 2008. WSC 2008. Winter*, pages 2127–2135, Dec 2008.

T-E Lee, H-Y Lee, and S-J Lee. Scheduling a wet station for wafer cleaning with multiple job flows and multiple wafer-handling robots. *International Journal of Production Research*, 45(3):487–507, 2007.

E. Levner and V. Kats. A parametric critical path problem and an application for cyclic scheduling. *Discrete Applied Mathematics*, 87(13):149 – 158, 1998.

E. Levner, V. Kats, and V.E. Levit. An improved algorithm for cyclic flowshop scheduling in a robotic cell. *European Journal of Operational Research*, 97(3): 500 – 508, 1997.

E. Levner, V. Kats, and D.A. Lopez De Pablo. Cyclic scheduling in robotic cells: An extension of basic models in machine scheduling theory. In Eugene Levner, editor, *Multiprocessor Scheduling, Theory and Applications*, chapter 1, pages 1–20. InTech Education and Publishing, Vienna, Austria, 2007.

J. Li, L. Zhang, C. ShangGuan, and H. Kise. A ga-based heuristic algorithm for non-permutation two-machine robotic flow-shop scheduling problem of minimizing total weighted completion time. In *IEEE International Conference on Industrial Engineering and Engineering Management*, pages 1281–1285, 2010.

A. Lim, B. Rodrigues, and C. Wang. Two-machine flow shop problems with a single server. *Journal of Scheduling*, 9(6):515–543, 2006.

V. Lippiello, L. Villani, and B. Siciliano. An open architecture for sensory feedback control of a dualarm industrial robotic cell. *Industrial Robot: An International Journal*, 34(1):46–53, 2007.

J. Liu, Y. Jiang, and Z. Zhou. Cyclic scheduling of a single hoist in extended electroplating lines: a comprehensive integer programming solution. *IIE Transactions*, 34(10):905–914, 2002.

A. Machmudah, S. Parman, A. Zainuddin, and S. Chacko. Polynomial joint angle arm robot motion planning in complex geometrical obstacles. *Applied Soft Computing*, 13(2):1099 – 1109, 2013.

A. Nambiar and R. Judd. Max-plus based mathematical formulation for cyclic permutation flow-shops. *International Journal of Mathematical Modeling and Numerical Optimization*, 2(1):85–971, 2011.

N.G. Odrey and G. Meja. An augmented petri net approach for error recovery in manufacturing systems control. *Robotics and Computer-Integrated Manufacturing*, 21(45):346 – 354, 2005.

A. Olabi, R. Bare, O. Gibaru, and M. Damak. Feedrate planning for machining with industrial six-axis robots. *Control Engineering Practice*, 18(5):471 – 482, 2010.

K. Osakada, K. Mori, T. Altan, and P. Groche. Mechanical servo press technology for metal forming. *{CIRP} Annals - Manufacturing Technology*, 60(2):651 – 672, 2011.

H.J. Paul, C. Bierwirth, and H. Kopfer. A heuristic scheduling procedure for multi-item hoist production lines. *International Journal of Production Economics*, 105 (1):54 – 69, 2007.

M. Pinedo. *Planning and Scheduling in Manufacturing and Services*. 2nd Edition. Springer, 2009.

V. Potkonjak, G.S. Dordevic, D. Kostic, and M. Rasic. Dynamics of anthropomorphic painting robot: Quality analysis and cost reduction evolutionary. *Robotics and Autonomous Systems*, 32(1):17 – 38, 2000.

G. Quan and V. Chaturvedi. Feasibility analysis for temperature-constraint hard real-time periodic tasks. *IEEE Transactions on Industrial Informatics*, 6(3): 329–339, 2010.

R.D. Quinn, G.C. Causey, and F.L. Merat. An agile manufacturing workcell design. *IIE Transactions*, 29(10):901 – 909, 1997.

T. Rajapakshe, M. Dawande, and C. Sriskandarajah. Quantifying the impact of layout on productivity: An analysis from robotic-cell manufacturing. *Operations Research*, 59(2):440–454, 2011.

S.M. Ross. *Stochastic processes*. Wiley series in probability and statistics: Probability and statistics. Wiley, 1996.

N. Safaei and R. Tavakkoli-Moghaddam. Integrated multi-period cell formation and subcontracting production planning in dynamic cellular manufacturing systems. *International Journal of Production Economics*, 120(2):301 – 314, 2009.

M.A. Sambola, E. Fernndez, and F. Saldanha da Gama. The facility location problem with bernoulli demands. *Omega*, 39(3):335 – 345, 2011.

M. Savsar and M. Aldaihani. Modeling of machine failures in a flexible manufacturing cell with two machines served by a robot. *Reliability Engineering and System Safety*, 93(10):1551 – 1562, 2008.

S.P. Sethi, C. Sriskandarajah, G. Sorger, J. Blazewicz, and W. Kubiak. Sequencing of parts and robot moves in a robotic cell. *International Journal of Flexible Manufacturing Systems*, 4(3-4):331–358, 1992.

S.P. Sethi, J.B. Sidney, and C. Sriskandarajah. Scheduling in dual gripper robotic cells for productivity gains. *IEEE Transactions on Robotics and Automation*, 17(3):324–341, 2001.

D. Shabtay, K. Arviv, H. Stern, and Y. Edan. A combined robot selection and scheduling problem for flow-shops with no-wait restrictions. *Omega*, 43:96 – 107, 2014.

S. Shafiei-Monfared, K. Salehi-Gilani, and K. Jenab. Productivity analysis in a robotic cell. *International Journal of Production Research*, 47(23):6651–6662, 2009.

A. Soukhal and P. Martineau. Resolution of a scheduling problem in a flowshop robotic cell. *European Journal of Operational Research*, 161(1):62 – 72, 2005.

Chelliah Sriskandarajah, I. Drobouchevitch, S.P. Sethi, and R Chandrasekaran. Scheduling multiple parts in a robotic cell served by a dual gripper robot. *Operations Research*, 52(1):65–82, 1998a.

Chelliah Sriskandarajah, Nicholas G. Hall, and Hichem Kamoun. Scheduling large robotic cells without buffers. *Annals of Operations Research*, 76(0):287–321, 1998b.

G. Steiner and Z. Xue. Scheduling in reentrant robotic cells: Algorithms and complexity. *Journal of Scheduling*, 8(1):25–48, 2005.

Q. Su and F.F. Chen. Optimal sequencing of double-gripper gantry robot moves in tightly-coupled serial production systems. *IEEE Transactions on Robotics and Automation*, 12(1):22–30, 1996.

Y-X. Sun and N-Q. Wu. Cycle time analysis for wafer revisiting process in scheduling of single-arm cluster tools. *International Journal of Automation and Computing*, 8(4):437–444, 2011.

S.A. Torabi, M. Hamedi, and J. Ashayeri. A new optimization approach for nozzle selection and component allocation in multi-head beam-type {SMD} placement machines. *Journal of Manufacturing Systems*, 32(4):700 – 714, 2013.

F. Tysz and C. Kahraman. Modeling a flexible manufacturing cell using stochastic petri nets with fuzzy parameters. *Expert Systems with Applications*, 37(5):3910 – 3920, 2010.

K. Vacharanukul and S. Mekid. In-process dimensional inspection sensors. *Measurement*, 38(3):204 – 218, 2005.

S. Venkatesh and J.S. Smith. A graph-theoretic, linear-time scheme to detect and resolve deadlocks in flexible manufacturing cells. *Journal of Manufacturing Systems*, 22(3):220 – 238, 2003.

C. Wang and D.J. Cannon. Virtual-reality-based point-and-direct robotic inspection in manufacturing. *IEEE Transactions on Robotics and Automation*, 12(4): 516–531, 1996.

R.A. Wysk, N-S. Yang, and S. Joshi. Resolution of deadlocks in flexible manufacturing systems: avoidance and recovery approaches. *Journal of Manufacturing Systems*, 13(2):128 – 138, 1994.

B. Xie, J. Zhao, and Y. Liu. Fault tolerant motion planning of robotic manipulators based on a nested rrt algorithm. *Industrial Robot: An International Journal*, 39 (1):40–46, 2012.

J. Xie and X. Wang. Complexity and algorithms for two-stage flexible flowshop scheduling with availability constraints. *Computers and Mathematics with Applications*, 50(1012):1629 – 1638, 2005.

P. Yan, C. Chu, A. Che, and N. Yang. An algorithm for optimal cyclic scheduling in a robotic cell with flexible processing times. In *IEEE International Conference on Industrial Engineering and Engineering Management*, pages 153–157, Dec 2008.

P. Yan, C. Chu, N. Yang, and A. Che. A branch and bound algorithm for optimal cyclic scheduling in a robotic cell with processing time windows. *International Journal of Production Research*, 48(21):6461–6480, 2010.

P. Yan, A. Che, X. Tang, and C. Chu. Cyclic robotic cells scheduling using tabu search algorithm. In *9th IEEE International Conference on Networking, Sensing and Control (ICNSC)*, pages 58–62, 2012.

D-Li. Yang, C-J. Hsu, and W-H. Kuo. A two-machine flowshop scheduling problem with a separated maintenance constraint. *Computers and Operations Research*, 35(3):876 – 883, 2008.

J. Yi, S. Ding, D. Song, and M.T. Zhang. Steady-state throughput and scheduling analysis of multicluster tools: A decomposition approach. *IEEE Transactions on Automation Science and Engineering*, 5(2):321–336, 2008.

S. Yildiz, M.S. Akturk, and O.E. Karasan. Bicriteria robotic cell scheduling with controllable processing times. *International Journal of Production Research*, 49 (2):569–583, 2011.

S. Yildiz, O.E. Karasan, and M.S. Akturk. An analysis of cyclic scheduling problems in robot centered cells. *Computers and Operations Research*, 39(6):1290 – 1299, 2012.

P.T. Zacharia and N.A. Aspragathos. Optimal robot task scheduling based on genetic algorithms. *Robotics and Computer-Integrated Manufacturing*, 21(1):67 – 79, 2005.

M.H. Fazel Zarandi, H. Mosadegh, and M. Fattahi. Two-machine robotic cell scheduling problem with sequence-dependent setup times. *Computers and Operations Research*, 40(5):1420 – 1434, 2013.

Z. Zhang and C.C. Murray. A corrected formulation for the double row layout problem. *International Journal of Production Research*, 50(15):4220–4223, 2012.

Z. Zhou and L. Li. Optimal cyclic single crane scheduling for two parallel train oilcan repairing lines. *Computers and Operations Research*, 39(8):1850–1856, 2012.

Z. Zhou, A. Che, and P. Yan. A mixed integer programming approach for multi-cyclic robotic flowshop scheduling with time window constraints. *Applied Mathematical Modelling*, 36(8):3621 – 3629, 2012.