



Tribhuvan University
Institute of Science and Technology

Bottleneck Just-in-Time Sequencing for Mixed-Model Production Systems

Thesis
Submitted to

Central Department of Mathematics
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Mathematics

by
Chudamani Poudyal

December 2008



**Tribhuvan University
Institute of Science and Technology**

Bottleneck Just-in-Time Sequencing for Mixed-Model Production Systems

Thesis
Submitted to

Central Department of Mathematics
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for the Master's Degree in Mathematics

by

Chudamani Poudyal

December 2008

Supervisor

Dr. Tanka Nath Dhamala

Co-Supervisor

Shree Ram Khadka



**Tribhuvan University
Institute of Science and Technology
Central Department of Mathematics**

Student's Declaration

I hereby declare that I am the only author of this work and that no sources other than that listed here have been used in this work.

Chudamani.

Chudamani Poudyal

Date: *...1.. December 2008*

Supervisor's Recommendation

I hereby recommend that this thesis prepared under my supervision by **Mr. Chudamani Poudyal** entitled **Bottleneck Just-in-Time Sequencing for Mixed-Model Production Systems** in partial fulfillment of the requirements for the degree of M. A./M. Sc. in Mathematics be processed for the evaluation.

Tanka Nath Dhamala

Dr. Tanka Nath Dhamala

Asst. Prof. Central Department of Mathematics

Presently, **Head**, Central Department of Computer Science and Information Technology

Institute of Science and Technology

Tribhuvan University, Nepal


Date: *..Dec..2.. 2008*




**Tribhuvan University
Institute of Science and Technology
Central Department of Mathematics**

We certify that this research work done by *Mr. Chudamani Poudyal* entitled **Bottleneck Just-in-Time Sequencing for Mixed-Model Production Systems** is satisfactory in the scope and generality as a thesis in the partial fulfillment for the requirement of M. A./M. Sc. degree in Mathematics.

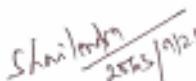
Evaluation Committee


.....
Prof. Yadav Prasad Koirala
Head, Central Department of Mathematics
Chairman, Thesis Evaluation Committee
Institute of Science and Technology
Tribhuvan University, Nepal




.....
Dr. Tanka Nath Dhamala
2075.9.21

Asst. Prof. Central Department of Mathematics
Presently, Head, Central Department of Computer Science and Information Technology
Institute of Science and Technology
Tribhuvan University, Nepal
(Supervisor)


.....
(External Examiner)

ACKNOWLEDGMENTS

To form the existence of this thesis has been a rewarding experience for me. This research work would not have been completed without the careful guidance and encouragement that I received from the individuals to whom I am to owe my enormous debt.

I owe a debt of profound gratitude to my respected supervisor *Dr. Tanka Nath Dhamala*, Asst. Professor, Central Department of Mathematics, at present, Head, Central Department of Computer Science and Information Technology, Tribhuvan University for his invaluable and untiring guidance, encouragement and suggestions from the very beginning to the end of the writing of this thesis.

My sincere gratitude goes to my respected co-supervisor, *Shree Ram Khadka*, Asst. Professor, Central Department of Mathematics, Tribhuvan University for his genuine constructive and perennial inspiration to come up with such materials.

I am sincerely grateful to *Professor Yadav Prasad Koirala*, Head, Central Department of Mathematics, Tribhuvan University for his continuous encouragement and inspiration.

I am also grateful to the research committee members of Central Department of Mathematics, Tribhuvan University who conclude a positive decision for providing this golden opportunity. I owe very much to all my respected teachers of the Central Department of Mathematics, Tribhuvan University who have always been my source of legitimate suggestion and inspiration. I would like to express my grateful sincere thanks to *Gyan Bahadur Thapa*, Asst. Professor, Pulchowk Campus, Institute of Engineering, Tribhuvan University, Nepal.

I would like to express my deep respect to *Dr. George Steiner*, Professor, McMaster University, Hamilton, Ontario, Canada for providing me some resourceful materials.

I am very much indebted to my family members especially to my mother for her transcendent and lucid inspiration and suggestion which are indeterminate. My special thanks go to all my colleagues, friends and relatives for their generative suggestions, kind help and motivation and others for wishing me all the best.

Finally, I am immensely thankful to my friends *Prabin Rai* and *Prem Bhatta* for their patient computer work and attractive type setting. I am also especially indebted to *Bedu Dahal* and *Chuda Bohara* for their generous grammatical correction with creating academic environment in my room during the preparation of this thesis.

December 2008
Kathmandu, Nepal

Chudamani Poudyal
Central Department of Mathematics

ABSTRACT

Bottleneck Just-in-Time Sequencing for Mixed-Model Production Systems

Due to today's competitive automotive industrial challenges of providing a variety of products at a very low cost by smoothing productions on a flexible transfer line, one of the most important and fertile research topic in industrial mathematics is to penalize jobs both for being early and for being tardy. A problem is to determine a production sequence for producing different types of products on the line. Just-in-Time (JIT) mixed-model production system is used to address this problem, which involves producing only the right products of different models of a common base product in evenly balanced sequences in the exact quantities, at the right times, at the right place. Sequencing JIT production system can be formulated as a challenging nonlinear integer programming problem. The goal of such system is to balance the rate of production of products. Minimization of the variation in demand rates for outputs of supplying processes is the output rate variation problem (ORVP) and minimization of the variation in the rate at which different products are produced on the line is the product rate variation problem (PRVP).

The problem for minimizing of deviations between actual and desired production for PRVP can be solved efficiently in pseudo-polynomial time complexity. However, the ORVPs for two or more levels are strongly *NP*-hard. Heuristic algorithms and dynamic programming to solve such *NP*-hard problems are summarized. But ORVPs with pegging assumption are solvable by reducing them to the corresponding weighted PRVPs. The cyclic sequences are optimal for both sum and max deviation PRVPs.

For the bottleneck PRVP, a binary search technique is used to test the existence of a perfect matching and thereby to get optimal sequence. A feasible sequence always exists such that, at all times, the deviation of actual production from the desired level of production for every product is never more than one unit for the max-absolute and max-squared PRVPs. An elegant algebraic concept of balanced words is used to deal the bottleneck PRVP. The max-absolute PRVP is shown to be *Co-NP* with leaving its general complexity open.

In this thesis, we study several interesting algebraic structures, properties, existence of cyclic solutions and two applications of bottleneck PRVP. An optimal sequence for an instance of max-absolute PRVP is obtained. With considering two min-sum and max-absolute objectives, a bicriterion objective for balancing the sequence is analyzed. A comparative study of different objectives is also summarized. Moreover, several directions for further research are also explored including some emerged conjectures.

CONTENTS

ACKNOWLEDGMENTS	i
ABSTRACT	ii
1 INTRODUCTION	1
2 MATHEMATICAL BACKGROUND	4
2.1 Functions.....	4
2.2 Sequences.....	5
2.3 Algorithms, Complexity and Heuristics.....	6
2.3.1 Algorithms.....	6
2.3.2 Computational Complexity.....	7
2.3.3 Heuristics.....	8
2.4 Combinatorial Optimization Problems.....	8
2.4.1 Integer Programming.....	9
2.4.2 Bipartite Matching Problems.....	9
2.4.3 The Assignment Problems.....	11
2.5 Dynamic Programming.....	13
2.6 Scheduling Problems.....	14
3 MATHEMATICAL FORMULATIONS OF MIXED-MODEL JIT PRODUCTION SYSTEMS	16
3.1 Multi-Level Problem Formulation.....	16
3.2 Single-Level Problem Formulation.....	19
4 THE SUM-DEVIATION MIXED-MODEL JIT PRODUCTION SYSTEMS	22
4.1 Toyota's Goal Chasing Method to Solve Sum-Deviation ORVP.....	22
4.2 The Sum-Deviation Mixed-Model PRV-JIT Production Systems.....	23
4.2.1 Ideal Corners and Positions.....	23
4.2.2 Reduction of F_{sum} to Assignment Problem.....	25
4.2.3 Reduction of F_{sum} to Integer Linear Programming.....	29
4.2.4 The Cyclic Sequences to F_{sum}	30
4.2.5 Dynamic Programming for F_{sum}	32

5	BOTTLENECK MIXED-MODEL JIT PRODUCTION SYSTEMS	34
5.1	Dynamic Programming for \tilde{F}_{\max}^{wa}	34
5.2	Reduction to Release Date/Due Date Decision Problem.....	36
5.3	Perfect Matching Representation	41
5.4	Bounds on the Threshold Value	53
5.4.1	Lower Bound.....	53
5.4.2	Upper Bound.....	57
5.5	The Binary Search for Optimality.....	60
5.5.1	Binary Search for Absolute Deviation.....	60
5.5.2	Binary Search for Squared Deviation.....	64
5.6	The Cyclic Sequences.....	64
5.7	Small Deviations.....	66
5.7.1	Small Deviations for F_{\max}^a	66
5.7.2	Small Deviations for F_{\max}^s	71
5.8	Two Product Problem.....	73
5.9	Bottleneck Assignment for F_{\max}	74
5.10	Complexity Status of F_{\max}^a	75
5.11	Application of F_{\max}^a Solution.....	76
6	BICRITERION ANALYSIS	78
6.1	B-Bounded Problem F_{sum}	78
6.2	Simultaneous Optimization of F_{\max} and F_{sum}	80
7	CONCLUSIONS	84
	REFERENCES	86
	MATHEMATICAL NOTATIONS	90

LIST OF FIGURES

2.1 Gantt Charts.....	14
4.1 Determination of Ideal Corner and Position.....	24
5.1 Level Curves for the Deviation of Ideal Production from Each Copy over all Time Periods.....	37
5.2 Deviation “Attributable” to Each Copy of a Product.....	38
5.3 Bipartite Graph of Feasible Producing Times for Four Products Induced by a Threshold Value of $B = 0.75$ for F_{\max}^a	42
5.4 Convex Bipartite Graph for an Instance $\left(3;3,3,1;\frac{4}{7}\right)$ of F_{\max}^a	46
5.5 Two Distinct Intervals $[k_i \dots k_i']$ and $[k_j \dots k_j']$ with $i < j$ Containing the Neighbors of Vertex k	47
5.6 Bipartite Graph of Feasible Producing Times for Five Products with Threshold Value $B = 13/20$ for F_{\max}^a	54
5.7 Bipartite Graphs for Different Threshold Value for F_{\max}^a	62
5.8 A Conjectured Topography of Balanced Words and the Solutions of JITSP F_{\max}^a ...	72
6.1 Representation of the Sets of Instances Corresponding to the Values of the Quintuplet (AM, SM, AM1, SM1, AS).....	82
6.2 Example of Instances with Sequences Optimizing Several Criteria.....	82

LIST OF ABBREVIATIONS

AP	Assignment Problem
DP	Dynamic Programming
EDD	Earliest-Due-Date
JIT	Just-in-Time
JITSP	Just-in-Time Sequencing Problem
LBAP	Linear Bottleneck Assignment Problem
MDJIT	Maximum Deviation Just-in-Time
MMJIT	Mixed-Model Just-in-Time
MMJITSP	Mixed-Model Just-in-Time Sequencing Problem
ORVP	Output Rate Variation Problem
PRV-JIT	Product Rate Variation Just-in-Time
PRV-MDJIT	Product Rate Variation Maximum Deviation Just-in-Time
PRV-MMJIT	Product Rate Variation Mixed-Model Just-in-Time
PRV-MMJITSP	Product Rate Variation MMJITSP
PRVP	Product Rate Variation Problem
SDJIT	Sum Deviation Just-in-Time

CHAPTER 1

INTRODUCTION

Mixed-model or flexible manufacturing systems with negligible change-over costs between the different products make it possible for diversified small-lot instead of large-lot production, e. g., mixing models (products) on a car assembly line. Just-in-Time (JIT) production system is designed to control such mixed-model production systems. The fundamental and underlying ideas of JIT have their origins at least at the beginning of the twentieth century (see [55]). The philosophy of JIT production system is all about having “the right product, at the right time, at the right place, and in the exact quantity”. The goal of such system is to meet customer’s demands for a variety of products without holding large inventories and incurring large shortages, provided that there is sufficient production capacity available.

Because of modern competitive industrial challenges of providing a variety of products at a very low cost by smoothing products (e. g. Toyota production system) and increasing computer applications (e. g. hard real time), schedulers are motivated in the years 1980’s on the concept of flexible manufacturing JIT systems to penalize jobs both for being early and tardy. JIT makes production operations more efficient, effective in cost and responsive to the customers. Overall, JIT lowers the cost and inventory, reduces waste, and thereby raising the quality of products and profit.

A multi-level JIT production system consists of hierarchy of finite and distinct levels such as products, sub-assemblies, components, raw materials, etc. It is a pull system where the sub-assemblies, components, raw materials, etc. are pulled forward as they are needed. The multi-level JIT production system is more challenging problem in sequencing theory than the single-level JIT production system because the components require for different models may or may not be distinct in multi-level but the different models require the same number and mix of components in single-level. Production is initiated by higher level’s requirement for another level’s output. As a result, the final assembly line is the focus for control. Its sequence determines the production sequence at all other levels.

An important and obvious optimization problem associated with JIT production system is that of determining the sequence for manufacturing different types of products on the final assembly line that minimizes deviation throughout the line, between the actual and the ideal (desired) production. The production sequence depends upon the goals of the manufacturing facility. Typically, there are two goals; *usage and loading*. A production sequence with usage goal is referred to as *leveling* or *balancing* the sequence and one of the foremost proponents of JIT system, *Monden* [41] describes it as the cornerstone of the JIT production system by the management of Toyota and how it is handled at Toyota, the first company to use JIT production system.

The seminal work of *Miltenburg* [37] is the formulation of the balanced sequence problem for single-level mixed-model JIT production system as a nonlinear integer programming, under the assumption that the products require approximately the same number and mix of parts. As an extension, *Miltenburg and Sinnamon* [39] and *Miltenburg and Goldstein* [38] formulate the multi-level JIT production systems. *Miltenburg et al.*

[40] propose a dynamic programming algorithm for optimizing the single-level sum-squared problem.

Kubiak [29] provides a comprehensive review of the analytical literature until 1993. He also refers to the single-level problem as the Product Rate Variation Problem (PRVP) and the multi-level problem as the Output Rate Variation Problem (ORVP). We call min-max for the bottleneck objective and min-sum in the case of sum objective. *Inman and Bulfin* [24] develop an efficient earliest due date (EDD) first algorithm for the PRVP with the objective function that is mathematically different but intuitively similar to *Miltenburg* [40], and *Kubiak and Sethi* [32, 33].

Steiner and Yeomans [49] investigate the bottleneck max-absolute PRVP introducing a new non-convex objective function to be minimized. The authors reduce the problem into a single machine scheduling problem with release dates and due dates and develop an efficient graph theoretic optimization pseudo-polynomial time algorithm for optimality. The seminal aspect of this research is that an optimal sequence always exists with the deviation for every product is never more than one unit and the lower bound for such problem is also established. They [50] also give an algorithm for max-absolute ORVJIT assembly systems under the pegging assumption. They [50] show that the problem with pegging assumption is equivalent to a weighted PRVP which then can be solved by modified algorithm for un-weighted PRVP. They [47] further investigate both the min-sum and max-absolute objectives simultaneously for the first time. Subsequently, *Labacque et al.* [35] compare PRVP with different objective functions and give structures in which some sequences optimize several objective functions simultaneously with many conjectures. Further, *Steiner and Yeomans* [50] establish the existence of optimal cyclic sequences for the bottleneck PRVP. Similarly, *Kubiak* [27] shows that the cyclic sequences are also optimal for the sum PRVP.

Kubiak and Sethi [32, 33] reduce the min-sum PRVP with the objective function the sum of unimodal, symmetric, nonnegative and convex functions having 0 at 0 into an assignment problem and thereby present an efficient optimization algorithm with pseudo-polynomial time complexity determining ideal position of the product and penalize equally for both early or tardy production, the cost for the corresponding assignment problem.

Kubiak [29] and *Kubiak et al.* [34] respectively prove that even the special cases of the min-sum and min-max ORVPs are *NP*-hard. However, as a solution procedure, *Kubiak et al.* [34] give the dynamic programming approach to solve both the problems. A number of heuristics are proposed in [34, 38, 39]. Several researchers have implemented meta-heuristic methods including genetic algorithms, beam search, ant colony optimization, and a multi-agent method for ORVP to obtain a better solution (see [57]).

Brauner and Crama [7] study structural properties of the max-absolute PRVP and give a set of algebraic necessary and sufficient conditions for the existence of a solution for a given objective value. They show that the problem is *Co-NP* and polynomially solvable when the number of products is fixed, but its general version is still open whether it is *Co-NP*-complete or polynomially solvable [19, 20]. They further improved the upper bound for max-absolute and proved it is strictly less than one. This bound is further improved in [28]. They formulate the small deviation conjecture. *Brauner et al.* [8] exploit the concept of algebraic balanced words to give the proof of the conjecture. The

max-absolute problem has been studied by reducing it into other scheduling problem such as the apportionment problem (see [3, 4]) and chairman assignment problem (see [17]). The applications of min-max optimal JIT sequences have been extended to the other areas, e. g. in resource allocation problems, multiprocessing, service industry, hard real time, internet and network services (see [17, 28]). *Aigbdo* [1] describes the structural properties of min-sum PRVP and thereby establishes the lower bound for the min-squared PRVP. *Corominas and Moreno* [12, 13] investigate relationships between the solution spaces of different objective functions.

Dhamala [15] presents an efficient min-max-absolute-chain-algorithm to obtain an optimal solution for max-absolute objective of PRVJITSP. *Dhamala et al.* [18, 19] modify various algebraic structures and results established in [7] and [49] for max-absolute PRVP to max-squared PRVP.

The JIT production system with special attention to bottleneck objectives with matching problem is the main content of this thesis.

In Chapter 2, a synopsis introduction of the pivotal mathematical terminologies is given relating to the subsequent study of this work.

The mathematical formulation of the JIT production systems only under the usage goal is the content of Chapter 3. The first section describes the formulation of ORVP both for weighted and un-weighted cases. The final section is for the PRVP formulation.

In Chapter 4, the sum deviation JIT production system is investigated in brief. A heuristic for the ORVP is given. By introducing ideal corners and positions, the PRVP is reduced to the assignment problem to solve for optimality. The cyclic solutions are optimal for PRVP. Since the PRVP can be solved in pseudo-polynomial time, an efficient dynamic programming approach has been given for practical sized problem.

In Chapter 5, we study the sequencing procedures for the bottleneck JIT production systems with its algebraic structures, important properties including its two applications. A dynamic programming algorithm has been used to obtain the optimal solutions for the *NP*-hard ORVP. The bottleneck JIT problem can be considered as a perfect matching problem in a convex bipartite graph with the reduction to release date/due date decision problem (cf. [49]). An optimal sequence can be obtained under the binary search algorithm within the interval given by the bounds of the problem. The optimal sequence is also cyclic for this problem (cf. [50]).

The Chapter 6 is devoted to study the min-sum and max-absolute objectives to obtain all the Pareto optimal solutions. It also gives the comparison results of simultaneous optimization of different bottleneck and sum objectives.

As a conclusion, Chapter 7 is a synopsis of this work together with concluding remarks, the main results of this work, and with some open problems for further research.

In this research, we not only study the sequencing procedure both for ORV and PRV JITSP but also we explore the structural properties, existence of cyclic sequences, complexity status and two applications of the max-absolute PRVJITSP. Moreover, an optimal sequence for an instance of max-absolute PRVP is obtained under binary search.

CHAPTER 2

MATHEMATICAL BACKGROUND

2.1 Functions

Let A and B be sets (not necessarily finite). A *function* f from A to B is an assignment of exactly one element of B to each element of A . We write $f(a) = b$ if b is the unique element of B assigned by the function f to the element a of A . If f is a function from A to B , then we write $f : A \rightarrow B$. The sets A and B are called *domain* and *co-domain* of f respectively. The *range* of f is the set of all images of elements of A , i. e. the set $\{y \mid y \in B, \exists x \in A, f(x) = y\}$ is the range of f . The *graph* of the function $f : A \rightarrow B$ is the set of all ordered pairs $\{(x, y) \mid x \in A, y \in B \text{ and } f(x) = y\}$.

A function f which values are in the set of real numbers R is called a *real-valued* function and is *nonnegative* if $f \geq 0$. Since we shall be mostly interested in real valued function of real variable throughout this thesis, we write only "function" to mean the real-valued function of real variable unless otherwise specified. The function f is said to be *monotonically increasing* if $f(x) \leq f(y)$ whenever $x \leq y$. Similarly, f is called *monotonically decreasing* if $f(x) \geq f(y)$ whenever $x \leq y$.

A function f defined over a set $A \subseteq R$ is said to take on its *maximum* and *minimum* over A at the points x^* and x' respectively if $f(x') \leq f(x) \leq f(x^*)$ for all $x \in A$.

The function f is said to be *unimodal* if for some value a (the mode) such that either (i) or (ii) holds:

- (i) f is monotonically increasing for $x \leq a$ and monotonically decreasing for $x \geq a$. In that case, the maximum value of f is $f(a)$ and there are no other local maxima.
- (ii) f is monotonically decreasing for $x \leq a$ and monotonically increasing for $x \geq a$. In that case, the minimum value of f is $f(a)$ and there are no other local minima.

A set $A \subseteq R$ is said to be *convex* if, whenever it contains x and y , it also contains elements of the form $\alpha x + (1 - \alpha)y$, for all α , $0 \leq \alpha \leq 1$. And a function f is said to be *convex* over a convex set $A \subseteq R$ if for any two points $x, y \in A$ and for all α , $0 \leq \alpha \leq 1$, $f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$.

The *floor function* (it is often also called greatest integer function) denoted by $\lfloor x \rfloor$ assigns to the real number x the largest integer that is less than or equal to x . The *ceiling function* denoted by $\lceil x \rceil$ assigns to the real number x the smallest integer that is greater than or equal to x . The *rounding function* denoted by $[x]$ assigns to the real number x to the closest integer and if the fractional part of x is equal to $\frac{1}{2}$, it rounds downward. The following properties are obvious from definition: $\lfloor -x \rfloor = -\lceil x \rceil$ and $\lceil -x \rceil = -\lfloor x \rfloor$.

2.2 Sequences

By a *finite sequence* of n terms we understand a function whose domain is the first n natural numbers and we denote the finite sequence of n terms by $s = (s_1, \dots, s_n)$.

Definition 2.2.1 [51, 52, 54] *An alphabet (or vocabulary) A is a finite non-empty set of elements called symbols (letters). A word (or infinite sequence) over A is defined as a subjective function $\mathbb{N} \rightarrow A$, where \mathbb{N} is the set of all natural numbers. Sometimes, it is also called bi-infinite word or \mathbb{N} -word and is denoted by $s = (s_1, s_2, s_3, \dots)$.*

A function $f : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{R}$ is called a *double sequence* (see [2]). Here we state an interesting theorem arising in JIT assembly systems concerning to double sequence.

Theorem 2.2.1 [53] *Let λ_{ij} ($1 \leq i \leq n; j \in \mathbb{N}$) be a double sequence of nonnegative numbers with $n \geq 2$ such that $\sum_{i=1}^n \lambda_{ij} = 1$ for all $j \in \mathbb{N}$. For an infinite sequence s in an alphabet $\{1, \dots, n\}$, let x_{ik} be the number of i 's in the k -prefix of s . Then here exists a sequence s in $\{1, \dots, n\}$ such that*

$$\sup_{i,k} \left| \sum_{j=1}^k \lambda_{ij} - x_{ik} \right| \leq 1 - \frac{1}{2(n-1)}. \quad \square$$

We construct an infinite word s on an alphabet $A = \{1, \dots, n\}$ as $s = (s_1, s_2, \dots)$ such that $s_k \in A$ is the k^{th} letter of s , for all $k \in \mathbb{N}$. A factor of s is a finite word $S = (s_k, s_{k+1}, \dots, s_j)$ for some j with $k \leq j$ and the length of S is $|S| = j - k + 1$. The empty word is the word of length 0. We say that the index k is the position of the letter $i \in A$ in the word s . The rate (density) r_i of the letter $i \in A$ in finite word, S is defined as the fraction $r_i = \frac{|S|_i}{|S|}$, where $|S|_i$ denotes the number of occurrences of the index i in the word S .

Definition 2.2.2 *A δ -balanced word on alphabet $A = \{1, \dots, n\}$ is an infinite sequence $s = (s_1, s_2, \dots)$ such that*

- i) $s_j \in A$ for each $j \in \mathbb{N}$, and
- ii) if S and R are two subsequences consisting of t consecutive elements of s ($t \in \mathbb{N}$), then the number of occurrences of i in S and R differs by at most δ , i. e. $\left| |S|_i - |R|_i \right| \leq \delta$ for each $i \in A$.

For a finite word S , let $S^* = (SSS\dots)$ be the repetition of S . An infinite word s is called *periodic* if $s = S^*$ for some finite word S . An infinite word s is called *periodic, δ -balanced word* if s is δ -balanced and $s = S^*$ for some finite word S . A finite word S is called *symmetric* if $S = S^R$, where S^R is a mirror reflection of S , then, S is clearly a palindrome. An infinite word s is called *periodic, symmetric and δ -balanced word* if s is δ -balanced and $s = S^*$ for some finite symmetric word S . One of the main problems of balanced words in practice is to construct an infinite periodic sequence over a finite set of letters whose each letter is distributed as "evenly" through the sequence as possible

and each letter occurs with a given rate. Unfortunately, the existence of balanced sequences for most rates is unlikely (see [20]).

Brauner et al. [8] prove the following theorem.

Theorem 2.2.2 *There exists a periodic, symmetric and 1-balanced word on $n \geq 3$ letters with rates $r_1 < r_2 < \dots < r_n$, if and only if the rates verify $r_i = \frac{2^{i-1}}{2^n - 1}$. \square*

2.3 Algorithms, Complexity and Heuristics

2.3.1 Algorithms

A *computational problem* is a mathematical object representing a general question that might want to solve and is independent of its specific input. A problem with a specific set of inputs is called an *instance*. Hence, a computational problem is a function $\Pi : \mathcal{Z} \rightarrow Y$, where \mathcal{Z} is the set of all problems instances I and Y is the set of solutions. Naturally, one seeks an algorithm for solving a computational problem. An algorithm, in short *Alg*, is any well-defined sequence of mathematical procedure for solving a prescribed problem Π . We say that an algorithm solves problem Π , if it computes $\Pi(I)$ for each instance $I \in \mathcal{Z}$. Let us denote $\mathcal{A}(\Pi) = \{Alg \mid Alg \text{ solves } \Pi\}$.

Here, a searching algorithm to find an integer from the list of integers a_1, \dots, a_n where $a_1 < a_2 < \dots < a_n$ is described. To search for the integer x in the list, we begin by asking the questions of the form “Is $x > a_i$ for some $a_i, i = 1, \dots, n-1$?”. This can be done by first asking whether x is in the upper or lower half of the interval $[a_1 \dots a_n]$ which is the set of all integers between a_1 and a_n including both, then asking whether x is in the upper or lower half of the new (smaller by a factor of 2) interval, and so on, repeat this process until a list with one term is obtained. The full algorithm is as follows:

Algorithm 2.3.1 [45] The Binary Search

Input: A list of n integers $a_1 < a_2 < \dots < a_n$.

Output: Position of an integer x .

set: $i = 1$ (i is the left end point of search interval)

$j = n$ (j is the right end point of search interval)

while $i < j$, **do**

begin

$$m := \left\lfloor \frac{i+j}{2} \right\rfloor;$$

if $x > a_m$ **then** $i = m + 1$;

else $j = m$;

end

if $x = a_i$ **then** position: $= i$;

else position: $= 0$.

(Position is subscript of the term equal to x , or 0 if x is not found)

2.3.2 Computational Complexity

Definition 2.3.1 [6, 11, 43, 45] Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $O(g(x))$ if there are constants $c > 0$ and $k \in \mathcal{N}$ such that $|f(x)| \leq c|g(x)|$ whenever $x > k$. (This is read as “ $f(x)$ is big-Oh of $g(x)$.”)

One of the major goals of computational complexity theory and algorithm analysis is to measure the performance of algorithms with respect to their computational time. The *running time* of an algorithm is said to be $O(h(k))$ if for some constant $c > 0$ there exists an implementation that terminates after at most $ch(k)$ elementary steps of computer for all instances of size $k \geq k_0$. The smallest function such that the algorithm has running time $O(h(k))$ is called the *time complexity* of the algorithm (see [9, 16]). The time complexity $T(k)$ of a problem Π is the minimal time complexity of all $\text{Alg} \in \mathcal{A}(\Pi)$ so that for some constant $c > 0$ and $k_0 \in \mathcal{N}$ it holds $T(k) \leq ch(k)$ for all $k \geq k_0$. The existence of this minimality in general is not guaranteed and it is in fact one of the focal points of research in the computational complexity theory. Obtaining the lower bounds for the complexity of a problem are harder, however upper bounds are usually obtainable.

A *polynomial time* (polynomial) algorithm is the one whose time complexity function $T(k) \in O(h(k))$, where h is some polynomial and k is the input length of an instance I . If time complexity function cannot be bounded by the polynomial function, it is called *exponential time* algorithm (see [6, 16]). A computational problem Π is called *polynomially solvable* if there exists a polynomial time algorithm to solve it. A problem Π is called *pseudo-polynomially solvable* if the time complexity function $T(k)$ is polynomial with respect to $|I|$ and $\max(I)$, where $|I|$ and $\max(I)$ respectively denote the input length and the largest integer appearing in the instance $I \in \Pi$. Hence, the notion of pseudo-polynomially solvable depends on the magnitude of the largest input data involved.

Given any problem instance $I \in \mathcal{Z}$ of an optimization problem to minimize a certain objective function γ with respect to constraint set X , the optimal solution is given by $\gamma(x_0) = \min\{\gamma(x) \mid x \in X\}$, therefore, $\Pi(I) = \gamma(x_0)$. However, the range Y must contain elements to represent “unbounded” and “infeasibility”, too, in general. A problem Π is called *decision problem* if $Y = \{\text{yes}, \text{no}\}$. Each optimization problem has its decision counterpart which is associated by defining an additional threshold value y for the corresponding objective function γ . For example, given an additional threshold value y for the objective function γ we ask: does there exist a feasible solution $x \in X$ such that $\gamma(x) \leq y$?

In complexity classes, the set of all decision problems which are polynomially solvable is denoted by P . The class of all decision problems whose all yes instances can be

checked for validity in polynomial time, given some additional information called certificate, is denoted by NP (Non-deterministic Polynomial Time).

Similarly, the class of all problems that are the complements of the problems in NP , i. e. for every no instance I there exists a concise certificate for I , which can be checked for validity in polynomial time, is denoted by $Co-NP$. The set of complements of NP -complete problems is called $Co-NP$ -complete.

We say that a decision problem Π_2 reduces to another decision problem Π_1 , denoted by $\Pi_2 \propto \Pi_1$, if there exists a polynomial time transformation function $h: \mathcal{Z}_2 \rightarrow \mathcal{Z}_1$ such that $\Pi_2(I) = \text{yes}$ for $I \in \mathcal{Z}_2$ if and only if $\Pi_1(h(I)) = \text{yes}$ for $h(I) \in \mathcal{Z}_1$. A decision problem Π_1 is called NP -complete if $\Pi_1 \in NP$ and for any other known decision problem $\Pi_2 \in NP$ we have $\Pi_2 \propto \Pi_1$. Since it follows from $\Pi_2 \propto \Pi_1$ that the problem Π_1 is at least as NP -hard as the problem Π_2 , it is sufficient to consider any known NP -complete problem Π_2 in the complexity hierarchy. The “problem reducibility” relation is a transitive relation on the class of decision problems. A decision problem in NP is called NP -complete in strong sense if it can be solved pseudo-polynomially only if $P = NP$, which is one of the major open problems in modern mathematics. An optimization problem is called NP -hard if the corresponding decision problem is NP -complete.

Example 2.3.1 [6, 43] *THREE PARTITION*

Given $3n$ integers $\{c_1, \dots, c_{3n}\}$, is there a partition of these integers into n triples T_1, \dots, T_n such that

$$\sum_{c_j \in T_i} c_j = \sum_{c_j \in T_k} c_j, \text{ for all } i, k?$$

This problem is strongly NP -complete. □

2.3.3 Heuristics

Any approach without formal guarantee of performance can be considered a “*heuristic*”. Such approach is useful in practical situations when no better method is available (see [9]). A heuristic does not always guarantee to solve the problem, but often solves it well enough for most uses, and often does so more quickly than a more complete solution would. Heuristic is the art and science of discovery and invention. The word comes from the Greek root as “eureka”, means “to find”. A heuristic for a given problem is a way of direction towards a solution (see [46]). It is different from an algorithm in that it merely serves as a rule of thumb or guideline, as opposed to an invariant procedure. Heuristics may not achieve the desired outcome, but can be extremely valuable to problem-solving (see [55]). Good heuristics can dramatically reduce the time required to solve a problem by eliminating the need to consider unlikely possibilities or irrelevant states.

2.4 Combinatorial Optimization Problems

Combinatorial optimization (it is also often called “discrete optimization”) is a branch of applied mathematics and theoretical computer science, related to algorithm theory, computational theory etc. A *combinatorial optimization* is a function $\Pi: \mathcal{Z} \rightarrow Y$, in which Π seeks to pick out the best feasible solution from Y which is in this case finite

or possibly countably infinite. Hence, the domain \mathcal{Z} of combinatorial optimization problem Π is a set of optimization problem instances I having finite or countably infinite number of feasible solutions under some given constraints. Some combinatorial optimization problems are described here.

2.4.1 Integer Programming

An optimization problem in the form

$$\begin{aligned}
 \text{[P2.1]} \quad & \text{minimize } f(x) \\
 & \text{subject to the constraints} \\
 & Ax = b \\
 & x \in W^n
 \end{aligned} \tag{2.1}$$

where $f: \mathcal{R}^n \rightarrow \mathcal{R}$, $A = (a_{ij})_{m \times n}$ integer matrix, b an n -dimensional integer column vector, and W is the set of whole numbers, is called the *integer programming*. The problem [P2.1] is said to be *linear integer programming* (ILP) if in addition f is linear having integer coefficients (see [43]). And the problem [P2.1] is said to be *nonlinear integer programming* (NLIP) if the function f is non linear. A vector $x = (x_1, \dots, x_n)$ satisfying the constraints (2.1) is called *feasible solution* to the problem [P2.1]. If $x \in \{0,1\}^n$, then the problem [P2.1] is said to be *binary linear program* or *zero-one linear program* (ZOLP) (see [43]).

Definition 2.4.1 A square, integer matrix B is called *unimodular (UM)* if its determinant $\det(B) = \pm 1$. An integer matrix A is called *totally unimodular (TUM)* if every square, non-singular sub-matrix of A is UM.

The following theorem is implicit in [43].

Theorem 2.4.1 If the coefficient matrix A of ILP is TUM, then the LP relaxation of ILP gives the optimal ILP solution. \square

2.4.2 Bipartite Matching Problems

A graph as a mathematical structure is a pair $G = (V, E)$ where $V = \{v_1, \dots, v_n\}$ is a non-empty finite set of *vertices* (or nodes), and E has as elements subsets of V of cardinality two called *edges*. An edge between two vertices v_i and v_j for $i \neq j$ is denoted by $[v_i, v_j]$. A *directed graph*, or *digraph* is a pair $G = (V, A)$ where V is again a set of vertices and A is a set of ordered pairs of vertices called arcs; that is, $A \subseteq V \times V$. An arc directed from vertex v_i to v_j is denoted by (v_i, v_j) . If $(v_i, v_j) \in A$, then v_i is the *predecessor* of v_j and v_j is the *successor* of v_i . For the sake of simplicity, we write $G = (V, E)$ for undirected graph and $G = (V, A)$ for digraph by ignoring the words undirected and directed throughout this study.

Two vertices v_i and v_j in V are adjacent if there exists an edge $e \in E$ having v_i and v_j as its end-vertices in a graph $G = (V, E)$; we say e is *incident* with v_i and v_j . A graph

$G = (V, E)$ with $V = \{v_1, \dots, v_n\}$, $E = \{e_1, \dots, e_m\}$ can be represented by its *node-edge incidence* $n \times m$ matrix $M = (m_{ij})$, where

$$m_{ij} = \begin{cases} 1 & \text{if } e_j \text{ is incident with } v_i \\ 0 & \text{otherwise} \end{cases}$$

The graph $G = (V, E)$ together with a function $w: E \rightarrow \mathbb{R}^+$ is called the *edge weighted graph* and together with a function $w': V \rightarrow \mathbb{R}^+$ is called *vertex weighted graph*, where \mathbb{R}^+ is the set of all nonnegative real numbers. Suppose that $G = (W, E)$ is a graph that has the following property. The set of vertices W can be partitioned into two sets, V and U , and each edge in E has one vertex in V and one vertex in U , i. e. any $[v, u] \in E$ implies that $v \in V$ and $u \in U$. Then G is called a *bipartite graph* and is usually denoted by $G = (V \cup U, E)$. A bipartite graph $G = (V \cup U, E)$ is said to be *complete* if each vertex of V is connected to each vertex of U . The bipartite graph $G = (V \cup U, E)$ is *V-convex* if there is an ordering on V such that $[v_i, u_k] \in E$ and $[v_j, u_k] \in E$ with $v_i, v_j \in V$, $v_i < v_j$ implies that $[v_p, u_k] \in E$ for $v_i \leq v_p \leq v_j$ (see [48]).

A *matching* M of a graph $G = (V, E)$ is a subset of the edges with the property that no two edges of M share the same node. Given a graph $G = (V, E)$, the matching problem is to find a *maximum matching* M of G (see [43]). When the cardinality of a matching is $\left\lfloor \frac{|V|}{2} \right\rfloor$, the largest possible in a graph with $|V|$ nodes, we say that the matching is *complete*, or *perfect* and the problem of finding a perfect matching M of G is called the *perfect matching problem* (see [43]).

Let us consider a graph $G = (V, E)$ together with a fixed matching M of G . Edges in M are called a *matched edges*; the other edges are *free*. If $[u, v]$ is a matched edge, then u and v are *mate* to each other. Nodes that are not incident upon any matched edges are called *exposed*; the remaining nodes are matched.

Now, consider a bipartite graph $G = (V \cup U, E)$ with $n = |V| \leq |U| = m$. For any subset X of vertices, denote by $N(X)$ the *neighborhood* of X , i. e. the set of all vertices adjacent to at least one vertex in X . Clearly, n is an upper bound for the perfect matching in G . The following theorem due to *Hall* [1935] (see [7, 9]) gives necessary and sufficient conditions for the existence of a matching with cardinality n .

Theorem 2.4.2 [7, 9] *Let $G = (V \cup U, E)$ be a bipartite graph with $n = |V| \leq |U| = m$. Then there exists in G a matching with cardinality n if and only if*

$$|N(X)| \geq |X| \quad \text{for all } X \subseteq V. \quad \square$$

A maximum matching M in a bipartite graph $G = (V \cup U, E)$ can be calculated in $O(\min(|V|, |U|) \cdot |E|)$ time (see [43]). But for a convex bipartite graph *Steiner and Yeomans* [48] prove the following.

Theorem 2.4.3 [48] *The maximum matching problem in the V -convex bipartite graph $G = (V \cup U, E)$ with $|U| = n$ can be solved in $O(n)$ time. \square*

2.4.3 The Assignment Problems

The *assignment problem* is one of the fundamental combinatorial optimization problems in the branch of optimization or operations research in mathematics. It consists of finding a maximum weighted matching in a weighted bipartite graph. Let us consider the complete bipartite graph $G = (V \cup U, V \times U)$ with $n = |V| \leq |U| = m$. Define a cost function $c : E \rightarrow R^+$ such that $c([v_i, u_j]) = c_{ij}$ where c_{ij} is the cost of assigning the vertex $v_i \in V$ to the vertex $u_j \in U$. An assignment is given by a one-to-one mapping $\varphi : V \rightarrow U$. The assignment problem (AP) [P2.2] is to find an assignment φ such that

$$\sum_{i=1}^n c_{ij}, \quad j = \varphi(i)$$

is minimized (see [9]). The corresponding *Linear Bottleneck Assignment Problem* (LBAP) [P2.3] is to find an assignment φ such that

$$\max c_{ij}, \quad j = \varphi(i)$$

is minimized (see [56]).

The assignment problem is the weighted complete bipartite matching problem where the weights w_{ij} are considered to the cost c_{ij} (see [43]). A weighted matching problem is to find a maximum matching in the weighted graph $G = (V, E)$ with the smallest possible sum of the corresponding weights (see [43]).

Here we summarize the *Hungarian Method* for the assignment problem strictly based on the reference [43]. To develop the Hungarian Method, without loss of generality, we may assume that the bipartite graph $G = (V \cup U, E)$ is complete with $V = \{v_1, \dots, v_n\}$, $U = \{u_1, \dots, u_n\}$, i. e. $|V| = |U| = n$. Let $x_{ij} \in \{0, 1\}$ for $i = 1, \dots, n$; $j = 1, \dots, n$, where $x_{ij} = 1$ if the edge $[v_i, u_j]$ is incident in the matching and $x_{ij} = 0$ if otherwise. Therefore, we must have

$$\begin{aligned} \sum_{i=1}^n x_{ij} &= 1 & j = 1, \dots, n \\ \sum_{j=1}^n x_{ij} &= 1, & i = 1, \dots, n \\ x_{ij} &\in \{0, 1\} & i, j = 1, \dots, n \end{aligned} \tag{2.2}$$

and our goal is to minimize $\sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij}$.

To describe the Hungarian Method, we use the following terminology and notations: The *labels* of vertices in a graph $G = (V, A)$ form an array with $|V|$ entries representing the predecessor vertex of all vertices. The label of a vertex $v \in V$ is denoted by $label[v]$. To represent the current matching in the complete bipartite graph $G = (V \cup U, E)$ we use

the array *mate* having $2n$ entries where $mate[w]$ for any vertex $w \in V \cup U$ denotes the vertex w' which is the mate of w . For any $v \in V$ *exposed*[v] is a node of U that is exposed and is adjacent to v ; if no such node exists, $exposed[v] = 0$. Now, for $j = 1, \dots, n$, *slack*[u_j] is the minimum of $(c_{ij} - \alpha_i - \beta_j)$ over all labeled vertices v_i of V and *nhbor*[u_j] is the particular labeled vertex v_i with which *slack*[v_j] is achieved.

Algorithm 2.4.1 [43] The Hungarian Method

Input: An $n \times n$ matrix $[c_{ij}]$ of nonnegative integers.

Output: An optimal perfect matching (given in terms of the array *mate*) of the complete bipartite graph $G = (V \cup U, E)$ with $|V| = |U| = n$ under the cost c_{ij} .

begin

for all $v_i \in V$ **do** $mate[v_i] := 0, \alpha_i := 0$;

for all $u_j \in U$ **do** $mate[u_j] := 0, \beta_j := \min_i \{c_{ij}\}$;

(comment: initialize)

for $i = 1, \dots, n$ **do** (comment : repeat for n stages)

begin

$A := \emptyset$;

for all $v \in V$ **do** $exposed[v] := 0$;

for all $u \in U$ **do** $slack[u] := \infty$;

for all v_i, u_j with $v_i \in V, u_j \in U$, and $\alpha_i + \beta_j = c_{ij}$ **do**

if $mate[u_j] = 0$ **then** $exposed[v_i] := u_j$

else $A := A \cup \{(v_i, mate[u_j])\}$;

(comment : construct the auxiliary graph)

$Q := \emptyset$;

for all $v_i \in V$ **do**

if $mate[v_i] = 0$ **then**

begin

if $exposed[v_i] \neq 0$ **then** augment (v_i) , **go to** endstage;

$Q := Q \cup \{v_i\}$;

$label[v_i] := 0$;

for all $u_k \in U$ **do**

if $0 < c_{ik} - \alpha_i - \beta_k < slack[u_k]$ **then** $slack[u_k] := c_{ik} - \alpha_i - \beta_k, nhbor[u_k] := v_i$;

end;

search: **while** $Q \neq \emptyset$ **do**

begin

let v_i be any node in Q ;

remove v_i from Q ;

for all unlabeled $v_j \in V$ with $(v_i, v_j) \in A$ **do**

begin

$label[v_j] := v_i$;

$Q := Q \cup \{v_j\}$;

```

        if exposed[vj] ≠ 0 then augment (vj), go to endstage;
        for all uk ∈ U do
            if 0 < cjk - αj - βk < slack[uk] then slake[uk] := cjk - αj - βk, nhbor[uk] := vj;
        end
    end;
    modify;
    go to search
endstage: end
end

```

procedure modify

(comment : it calculate θ_1 , updates the α 's and β 's, and activates new nodes to continue the search)

begin

$\theta_1 := \frac{1}{2} \min_{u \in U} \{slack[u] > 0\};$

for all v_i ∈ V **do**

if v_i is labeled then α_i := α_i + θ₁ else α_i := α_i - θ₁;

for all u_j ∈ U **do**

if slack [u_j] = 0 then β_j := β_j - θ₁ else β_j := β_j + θ₁;

for all u ∈ U with slack[u] > 0 **do**

begin

slack[u] := slack[u] - 2θ₁;

if slack[u] = 0 then (comment: new admissible edge)

if mate[u] = 0 then exposed[nhbor[u]] := u, augment(nhbor[u]), go to endstage;

else (comment: mate[u] ≠ 0)

label[mate[u]] := nhbor[u], Q := Q ∪ {mate[u]}, A := A ∪ {(nhbor[u], mate[u])};

end

end

procedure argument (v)

if label [v] = 0 then mate[v] := exposed [v],
mate [exposed[v]] := v;

else **begin**

exposed[label [v]] := mate[v];

mate[v] := exposed [v];

mate[exposed[v]] := v;

augment(label[v])

end

Theorem 2.4.4 [43] *The Algorithm 2.4.1 correctly solves the assignment problem for a complete bipartite graph with 2n nodes in $O(n^3)$ time.* □

2.5 Dynamic Programming

Dynamic programming (DP) refers to a computational technique rather than to a particular type of optimization problem and “programming” in this context refers to a tabular method (see [11, 23]). DP is an implicit enumerative method as it enumerates in

an intelligent way all possible solutions indirectly (see [9]). A DP algorithm solves every sub-problem just once and then saves its answer in a table, thereby avoiding the work of recomputing the answer every time the sub-problem is encountered. The idea is to work backwards from the last decisions to the earlier ones.

If a DP is applied to a combinatorial problem, then in order to calculate the optimal solutions for any sub-problem of size k , we first have to know the optimal value for each sub-problem of size $k-1$. Thus, if the problem is characterized by a set of n elements, the number of subsets considered is 2^n . It means that usually DP algorithms are of exponential computational complexity. However, for problems which are *NP*-hard it is often possible to construct pseudo-polynomial DP algorithms which are of practical value for reasonable instance sizes (see [6]).

DP can only be applied when the problem under concern has optimal sub-structure. Optimal sub-structure means that the optimal solutions of local problems can lead to the optimal solution of the global problem. That is, the problem can be solved by breaking it down, and solving the simpler problems.

2.6 Scheduling Problems

Scheduling is a decision making process that plays an important role in manufacturing and service industries. Suppose there are n -jobs J_i ($1 \leq i \leq n$) to be processed on m -machines M_j ($1 \leq j \leq m$). A schedule is for each job an allocation of one or more time intervals on one or more than one machines. Schedules may be represented by Gantt-Charts and is either machine oriented Figure 2.1 (a) or job oriented Figure 2.1 (b).

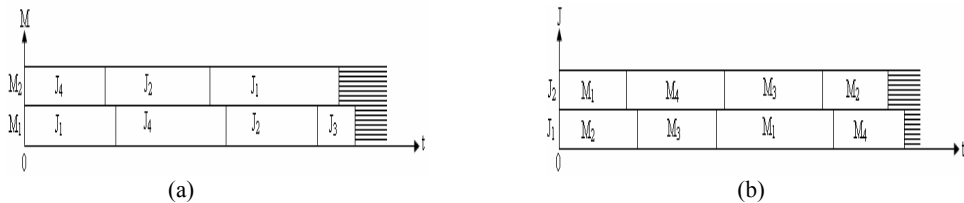


Figure 2.1: Gantt Charts

The processing of a job on a machine is called an *operation*. A schedule is *feasible* if no two time intervals on the same machine overlap, if no two time intervals allocated to the same job overlap, and, if in addition, it meets a number of problem specific characteristics (see [9]).

We denote a scheduling problem by a three field notation $\alpha|\beta|\gamma$ (cf. [22], see also [6, 9]). The first field $\alpha = \alpha_1\alpha_2$ describes the machine environment, where α_1 denotes the machine characteristic and α_2 denotes the number of machines used. The single machine environment is described by $\alpha_1 = \circ$, and $\alpha_2 = 1$, i. e. $\alpha = 1$, where \circ denotes the empty symbol. The second field β describes the job characteristic. If we denote preemption, precedence relation, release date (r_i), processing time (p_i) and due date (d_i), respectively, by $\beta_1, \beta_2, \beta_3, \beta_4$, and β_5 , then $\beta \in \{\beta_1, \beta_2, \beta_3, \beta_4, \beta_5\}$. The third field, γ , denotes the objective function. Corresponding to the completion time C_i of job i , the associate cost is denoted by the function $f_i(C_i)$. The usual bottleneck and sum objective

functions are, respectively, denoted by $f_{\max}(C) = \max_{i=1}^n \{f_i(C_i)\}$ and $f_{\text{sum}}(C) = \sum_{i=1}^n f_i(C_i)$.

Furthermore, given d_i as the fixed parameters for all jobs lateness, tardiness and earliness defined by $L_i = C_i - d_i$, $T_i = \max\{C_i - d_i, 0\}$, $E_i = \max\{0, d_i - C_i\}$, respectively, are usual objective functions depending upon due date parameters. With each of these functions h_i , we get different objectives

$$\gamma \in \left\{ \max_{i=1}^n (w_i |h_i|), \max_{i=1}^n (w_i h_i^2), \max_{i=1}^n (w_i h_i), \sum_{i=1}^n w_i |h_i|, \sum_{i=1}^n w_i h_i^2, \sum_{i=1}^n w_i h_i \right\}.$$

The *Scheduling Around the Shortest Job* (SASJ) problem is defined as (see [29]):

“ Given a set of n independent jobs (i. e. jobs without β_2) J_1, \dots, J_n , with processing times $p_1 \leq p_2 \leq \dots \leq p_n$ to be scheduled on a single machine. Find a nonpreemptive schedule of the jobs that minimizes $\sum_{i=1}^n (C_i - C_1)^2$ ”.

Theorem 2.6.1 [29] *The problem (SASJ), i. e. the problem*

$$1|p_1 \leq p_2 \leq \dots \leq p_n|\sum_{i=1}^n (C_i - C_1)^2$$

is NP-hard. □

Earliest-Due-Date First (EDD) Algorithm [6, 9]

Whenever a machine is freed, the job with the earliest due is selected to be processed next. This rule is to minimize the maximum lateness among the jobs waiting for processing. Actually, in a single machine setting, with n -jobs available at time 0, the EDD rule does minimize the maximum lateness.

Example 2.6.1: $1|r_i; d_i; pmtn||L_i|_{\max}$ is the problem of finding a preemptive schedule on one machine for a set of n -jobs with given release times $r_i \neq 0$ and due dates $d_i \neq 0$ such that the objective function $|L_i|_{\max}$ is minimized. □

CHAPTER 3

MATHEMATICAL FORMULATIONS OF MIXED-MODEL JIT PRODUCTION SYSTEMS

Mathematical models of a system grow out of equations that determine how the system changes from one state to the next and or how one variable depends on the value or state of other variable [36]. By an ideal mathematical formulation of any industrial problem the way of solution will be simpler and accurate. That is, mathematical modeling is a key to solve the complex industrial problems specially faced by automobile companies.

Just-in-Time Production System often uses mixed-model assembly lines. The effective utilization of these lines requires determining the sequence schedule for producing different products on the line. The sequence depends upon the goals of the manufacturing facility. There are two goals (cf. [41], see also [25, 39]).

1. USAGE Goal: Keeping a constant rate of usage of all parts in the facility.
2. LOADING Goal: Smooth the work load on the final assembly process to reduce the chance of production delays and stoppage.

A joint usage and loading problem is developed in [38], and is called the *joint problem*. Although both goals are important and need to be considered for all mixed-model assembly lines, the usage goal is considered to be more important goal for JIT production system (see [41]). That is, the quantity of each part used by the mixed-model assembly line per-unit time should be kept as constant as possible. The constant demand rates for the products yields a constant rate of parts usage. However, the integral nature of production creates variability between the actual and ideal production. Thus there should be very little variability in the usage of each part from one time period to the next. This is called *leveling or balancing* the sequence (see [37]).

The problems are formulated under the assumption that the systems have sufficient capacity with negligible switch over costs from one product to another and each product is produced in a unit time.

The single level problem of JIT production systems is formulated by *Miltenburg* [37] in 1989 and is in the form of nonlinear integer program. In his research different types of objective functions are considered and sequencing algorithms and heuristics are presented in an efficient way. As an extension of that work *Miltenburg and Sinnamon* [39] formulate the mixed-model, multi-level JIT assembly systems. But till now, there are very few research results which are carried out in this area such as [20, 29, 34, 39, 50] etc. The single level and multi level problems are respectively known as *Product Rate Variation Problem* (PRVP) and *Output Rate Variation Problem* (ORVP) (cf. [29]).

3.1 Multi-Level Problem Formulation

A mixed-model multi level assembly consists of a hierarchy of L distinct production levels $l, l=1,2,\dots,L$ with highest product level 1, where multiple copies of various products are made. The lower production levels subassemblies, component parts and raw materials are either fabricated or purchased for use in the products. Let n_l be the number of different part types of level l and d_{il} be the demand for part type i of level $l, i=1,2,\dots,n_l$. Let t_{ip} represents the number of total units of part type i at level l

required to produce one unit of product p , $p = 1, 2, \dots, n_1$, then $d_{il} = \sum_{p=1}^{n_1} t_{ilp} d_{p1}$ is the dependent demand for part type i of level l determined by the final product demands d_{p1} , $p = 1, 2, \dots, n_1$. Clearly, $t_{ilp} = 1$ for $i = p$, and 0, otherwise. Let $D_l = \sum_{i=1}^{n_l} d_{il}$ be the total part demand of level l . So the demand ratio for part type i at level l is $r_{il} = \frac{d_{il}}{D_l}$

with $\sum_{i=1}^{n_l} r_{il} = 1$ for $l = 1, 2, \dots, L$. The preemption is not allowed. Hence, one production on a product at level 1 has commenced, it must be completed before the production of another unit can start. This introduces the concept of a stage or cycle. One is said to be stage k (or in cycle k) if k units of product have been produced at level 1. There will be k complete units of the various product p at level 1 produced during these first k -stages and the time horizon will consist of D_1 stages in total. Let the quantity of part i at level l produced during stage 1 through k is denoted by x_{ilk} and y_{lk} be the total quantity of parts produced by level l during stages 1 through k . So clearly,

$y_{lk} = \sum_{i=1}^{n_l} x_{ilk}$. Thus, the cumulative production at level 1 through the first k stages is

$y_{1k} = \sum_{p=1}^{n_1} x_{p1k} = k$. By the pull nature of JIT systems and from the fact that lower level parts (from level 2, 3, ..., L) are drawn as needed by the final assembly process, the particular combination of the highest level products produced during these k stages (the x_{p1k} values) determines the necessary cumulative production at every other level.

Hence, for level $l \geq 2$ the required cumulative production through k stages for output i will be $x_{ilk} = \sum_{p=1}^{n_1} t_{ilp} x_{p1k}$. Let $w_{il} \geq 0$ be a weighting factor which reflects the relative

importance of balancing the sequence for part i at level l . Now consider f_{il} unimodal symmetric convex function with minimum 0 at 0, $i = 1, 2, \dots, n_l$; $l = 1, 2, \dots, L$. Then the mixed-model multi-level JIT production problem [P3.1] is to extract the matrix $\tilde{X} = (x_{p1k})_{n_1 \times D_1}$ that minimizes the following objective function(s) (cf. [34, 39], see also [19, 38]).

$$\tilde{F}_{\max}^w(\tilde{X}) = \max_{i,l,k} w_{il} f_{il}(x_{ilk} - y_{lk} r_{il}), \quad (3.1)$$

and
$$\tilde{F}_{\text{sum}}^w(\tilde{X}) = \sum_{k=1}^{D_1} \sum_{l=1}^L \sum_{i=1}^{n_l} w_{il} f_{il}(x_{ilk} - y_{lk} r_{il}) \quad (3.2)$$

subject to the constraints

$$x_{ilk} = \sum_{p=1}^{n_1} t_{ilp} x_{p1k}, \quad i = 1, \dots, n_l; l = 1, \dots, L; k = 1, \dots, D_1 \quad (3.3)$$

$$y_{lk} = \sum_{i=1}^{n_l} x_{ilk}, \quad l = 2, \dots, L; k = 1, \dots, D_1 \quad (3.4)$$

$$y_{1k} = \sum_{p=1}^{n_1} x_{p1k} = k, \quad k = 1, \dots, D_1 \quad (3.5)$$

$$x_{p1k} \geq x_{p1(k-1)}, \quad p = 1, \dots, n_1; k = 1, \dots, D_1 \quad (3.6)$$

$$x_{p1D_1} = d_{p1}, \quad x_{p10} = 0, \quad p = 1, \dots, n_1 \quad (3.7)$$

$$x_{ilk} \geq 0, \text{ integer}, \quad i = 1, \dots, n_l; l = 1, \dots, L; k = 1, \dots, D_l \quad (3.8)$$

Constraint (3.3) ensures that the necessary cumulative production of part i of level l by the end of time unit k is determined explicitly by the quantity of products produced at level 1. Constraints (3.4) and (3.5) show the total cumulative production of level l and level 1, respectively, during the time units 1 through k . Constraint (3.6) ensures that the total production of every product over k time units is a non-decreasing function of k . Constraint (3.7) guarantees that the demands for each product are met exactly. Constraints (3.5), (3.6), (3.8) ensure that exactly one unit of a product is sequenced during one time unit in the product level. The selection of weights will be based on the total production at various levels. Weights can be used to smooth the variability and to prevent lower level parts to be dominated over higher level parts in the measures at different levels and sometimes level wise weights w_l can also be considered (see [39, 50]). If $\tilde{\mathcal{X}} = \{\tilde{X} \mid \tilde{X} = (x_{p1k})_{n_1 \times D_1}\}$ be the set of all feasible solutions, then the multi level JIT sequencing problem is equivalent to the optimization problem (see [20]):

$$\min \{ \tilde{F}(\tilde{X}) \mid \tilde{X} \in \tilde{\mathcal{X}} \}, \text{ where } \tilde{F} \in \{ \tilde{F}_{\max}^w, \tilde{F}_{\text{sum}}^w \}.$$

The objectives of types $\tilde{F}_{\max}^w(\tilde{X})$ and $\tilde{F}_{\text{sum}}^w(\tilde{X})$ are respectively known as bottleneck and sum deviation objectives.

We assume $w_{il} = 1, \forall i = 1, \dots, n_l; l = 1, \dots, L$ for un-weighted case and the superfluous subscript w will be dropped out. So for un-weighted case, the objective function (3.1) and (3.2) respectively become:

$$\tilde{F}_{\max}(\tilde{X}) = \max_{i,l,k} f_{il}(x_{ilk} - y_{lk}r_{il}) \quad (3.9)$$

$$\tilde{F}_{\text{sum}}(\tilde{X}) = \sum_{k=1}^{D_1} \sum_{l=1}^L \sum_{i=1}^{n_l} f_{il}(x_{ilk} - y_{lk}r_{il}). \quad (3.10)$$

Now for absolute deviation $f_{il}(x) = |x|$, the objectives (3.1) and (3.2) respectively take the form

$$\tilde{F}_{\max}^{wa}(\tilde{X}) = \max_{i,l,k} w_{il} |x_{ilk} - y_{lk}r_{il}| \quad (3.11)$$

and
$$\tilde{F}_{\text{sum}}^{wa}(\tilde{X}) = \sum_{k=1}^{D_1} \sum_{l=1}^L \sum_{i=1}^{n_l} w_{il} |x_{ilk} - y_{lk}r_{il}|. \quad (3.12)$$

Similarly for squared deviation $f_{il}(x) = x^2$, (3.1) and (3.2) respectively take the form

$$\tilde{F}_{\max}^{ws}(\tilde{X}) = \max_{i,l,k} w_{il} (x_{ilk} - y_{lk}r_{il})^2 \quad (3.13)$$

and
$$\tilde{F}_{\text{sum}}^{ws}(\tilde{X}) = \sum_{k=1}^{D_1} \sum_{l=1}^L \sum_{i=1}^{n_l} w_{il} (x_{ilk} - y_{lk}r_{il})^2. \quad (3.14)$$

The absolute and squared objective functions corresponding to (3.11), (3.12), (3.13) and (3.14) without weighting factor w_{il} can be obtained from these objectives by removing w_{il} respectively. For convenience, by $\widetilde{F}_{\max}^{wa}$, for example, we mean the problem [P3.1] with the objective function $\widetilde{F}_{\max}^{wa}(\widetilde{X})$ and the feasible solution set $\widetilde{\chi}$.

3.2 Single-Level Problem Formulation

The notation of Section 3.1 for the highest product level 1 is used here by dropping out the superfluous subscript 1. Let us assume that the flexible transfer line produces n ($\in \mathbb{N}$) different products (models) with demands $d_i \in \mathbb{N}$ for products $i = 1, \dots, n$, totaling $D = \sum_{i=1}^n d_i$ units are to be produced during a specified time horizon, which is partitioned into D equal time units of which 1 time unit is required for a unit of a product to be produced. Let $r_i = \frac{d_i}{D}$ be the relative demand also known as ideal production rate for product i with $\sum_{i=1}^n r_i = 1$. A JIT system tries to keep the effective production rate as close as possible to r_i and a production sequence is called *uniformly leveled* if, at each time period k , $k = 1, \dots, D$, the line has assembled kr_i copies of product i (see [35]).

A production sequence $s = (s_1, s_2, \dots, s_D)$ is a finite sequence of integers satisfying $1 \leq s_k \leq n$ for $k = 1, \dots, D$. If $s_k = i$ then a copy of product i is produced in period k . A sequence s is feasible if the number of times $s_k = i$ is precisely d_i for every i , $i = 1, \dots, n$. Let x_{ik} ($1 \leq i \leq n; 1 \leq k \leq D$) be the cumulative production of product i through period k , then x_{ik} is the number of times $s_j = i$ for $1 \leq j \leq k$, and the equation $\sum_{i=1}^n x_{ik} = k$ must hold for each k . Thus, for any feasible sequence s it is possible to obtain a corresponding matrix $X = (x_{ik})_{n \times D}$ of the cumulative productions of each product at each period. Consider f_i ($1 \leq i \leq n$) be unimodal symmetric convex nonnegative discrepancy functions between the actual and ideal production with minimum $f_i(0) = 0$ and $f_i(y) > 0$ for $y \neq 0$. Then the mathematical model for the single level mixed-model JIT production system is the following optimization problem [P3.2]: Find a production sequence $s = (s_1, s_2, \dots, s_D)$, where product i occurs exactly d_i times that minimizes the following objective function(s) (cf. [37, 49], see also [14]):

$$F_{\max}(s) = \max_{i,k} f_i(x_{ik} - kr_i) \quad (3.15)$$

$$F_{\text{sum}}(s) = \sum_{k=1}^D \sum_{i=1}^n f_i(x_{ik} - kr_i) \quad (3.16)$$

subject to the constraints

$$\sum_{k=1}^n x_{ik} = k, \quad k = 1, \dots, D \quad (3.17)$$

$$x_{i(k-1)} \leq x_{ik}, \quad i = 1, \dots, n; k = 2, \dots, D \quad (3.18)$$

$$x_{iD} = d_i; \quad x_{i0} = 0, \quad i = 1, \dots, n \quad (3.19)$$

$$x_{ik} \geq 0, \quad \text{integer.} \quad (3.20)$$

Each of the constraints can be described similarly as in the case of multi level. If $\mathcal{X} = \{X | X = (x_{ik})_{n \times D}\}$ be the set of all feasible solutions, then the single level JIT sequencing problem is equivalent to solve the optimization problem (see [14]):

$$\min\{F(s) | X \in \mathcal{X}\}, \text{ where } F \in \{F_{\max}, F_{\text{sum}}\}.$$

If we introduce the weighting factor $w_i \geq 0$ for product $i, i = 1, \dots, n$, it reflects the priority of product i relative to the other products in the system. Then the weighted single level objective functions can be considered as:

$$F_{\max}^w(s) = \max_{i,k} w_i f_i(x_{ik} - kr_i) \quad (3.21)$$

and
$$F_{\text{sum}}^w(s) = \sum_{k=1}^D \sum_{i=1}^n w_i f_i(x_{ik} - kr_i) \quad (3.22)$$

For absolute deviation $f_i(x) = |x|$, the objectives (3.15) and (3.16) respectively become:

$$F_{\max}^a(s) = \max_{i,k} |x_{ik} - kr_i| \quad (3.23)$$

and
$$F_{\text{sum}}^a(s) = \sum_{k=1}^D \sum_{i=1}^n |x_{ik} - kr_i|. \quad (3.24)$$

Similarly for squared deviation $f_i(x) = x^2$, (3.15) and (3.16) respectively become:

$$F_{\max}^s(s) = \max_{i,k} (x_{ik} - kr_i)^2 \quad (3.25)$$

and
$$F_{\text{sum}}^s(s) = \sum_{k=1}^D \sum_{i=1}^n (x_{ik} - kr_i)^2. \quad (3.26)$$

Which are widely studied measure of deviations in the literature. Again as in the case of multi-level, by F_{\max}^a , for example, we mean the problem [P3.2] with the objective function $F_{\max}^a(s)$ and the set \mathcal{X} . Let $B > 0$ be a constant, then the decision version of the problem F_{\max}^a is to decide whether there exists a solution $s = (s_1, s_2, \dots, s_D)$ such that $\max_{i,k} f_i(x_{ik} - kr_i) \leq B$ holds for the matrix $X = (x_{ik})_{n \times D}$ derived from s and satisfied the constraints (3.17)-(3.20), and such a solution is known as B-bounded (B-feasible).

The mixed-model maximum deviation and sum deviation JIT sequencing problems are respectively denoted by MDJIT and SDJIT problems (see [7, 20]). Similarly the abbreviated form MMJIT refers to mixed-model Just-in-Time and MMJITSP refers to MMJIT sequencing problem. The bottleneck objective functions seek to find smooth sequences at each stage and as a result it precludes the possibility of relatively large deviations in every time period. In contrast the min-sum objective functions are concerned for finding the smooth sequences on the average which may result in relatively large deviations in certain time periods.

Under the pegging assumption, parts of output i at production levels which fed the level 1 are dedicated or pegged to the specific final product into which they will be assembled. This assumption decomposes the lower level parts that will be assembled into different level 1 products into disjoint sets. As a result, a distinction is made between t_{ilh} and t_{ilp} , $h \neq p$ for each part i at level l . With this assumption the multi level min-sum JIT

sequencing problem can be reduced to a weighted single level problem (see [19, 20]). Similarly with the same assumption the objective of the bottleneck problem \tilde{F}_{\max}^{wa} can be formulated as (see [20, 50]).

$$\begin{aligned}\tilde{F}_{\max}^{wa\ peg}(\tilde{X}) &= \max_{p,i,l,k} \left\{ w_{p1} |x_{p1k} - kr_{p1}|, w_{il} |x_{p1k} t_{ilp} - kt_{ilp} r_{p1}| \right\} \\ &= \max_{p,i,l,k} \left\{ w_{il} t_{ilp} |x_{p1k} - kr_{p1}| \right\}\end{aligned}$$

$p = 1, \dots, n_1; i = 1, \dots, n_l; k = 1, \dots, D_1; l = 1, \dots, L$. Now by letting $\bar{w}_{p1} = \max_{i,l} \{w_{il} t_{ilp}\}$, the objective function reduces to $\tilde{F}_{\max}^{wa\ peg}(\tilde{X}) = \max_{p,i,l,k} \left\{ \bar{w}_{p1} |x_{p1k} - kr_{p1}| \right\}$. Now by dropping out the superfluous subscript 1 the problem is reduced to the weighted PRVP

$$\min \left[\tilde{F}_{\max}^{wa\ peg}(\tilde{X}) = F_{\max}^{wa}(s) = \max_{i,k} \left\{ \bar{w}_i |x_{ik} - kr_i| \right\} \right],$$

$i = 1, \dots, n; k = 1, \dots, D$ and the set χ . Similarly the problem $\tilde{F}_{\max}^{ws\ peg}$ can be reduced to F_{\max}^{ws} (see [18]).

CHAPTER 4

THE SUM-DEVIATION MIXED-MODEL JIT PRODUCTION SYSTEMS

In this chapter, we summarize the solution procedures for sum deviation MMJITSP. The ORVP \tilde{F}_{sum}^s has been shown to be strongly *NP*-hard by reducing the already known *NP*-hard scheduling problem “*Around the Shortest Job*” to an instance of the ORVP \tilde{F}_{sum}^s (cf. [29]).

The heuristic approach to solve sum deviation ORVP for joint usage and loading problems is described in [38] and for only usage problems is described in [39]. Toyota used a heuristic known as Goal Chasing Method (GCM) to solve sum deviation ORVP with parts usage goal (cf. [41]).

Miltenburg [37] formulates three algorithms and two heuristics to solve F_{sum}^s . *Kubiak and Sethi* [33] reduce the problem F_{sum}^s to the polynomially solvable assignment problem. The existence of optimal cyclic solution for the problem F_{sum}^s with $n = 3$, and for demand vector $(600, 600, 100)$ is presented in [37] for the first time and is proved analytically in [27].

In Section 4.1 we describe Toyota Goal Chasing Method to solve Sum-Deviation ORVP. The main strategy of Section 4.2 is to bring the solution approaches for the problem F_{sum}^s .

4.1 Toyota’s Goal Chasing Method to Solve Sum-Deviation ORVP

Among the major car manufactures, Toyota has always been an innovator in the areas of manufacturing and assembly. Toyota operates according to the JIT principle. Toyota’s most important goal in the operation of its mixed-model production system is to keep the rates of consumption of all parts constant. For sequencing mixed model multi-level JIT production system, Toyota developed and used an algorithm known as the *Goal Chasing Method* (GCM) to sequence automobile final assembly lines, (cf. [41], see also [25, 39]). The GCM suggests sequencing the product (model) p among n_1 products at stage k ($1 \leq k \leq D$) with the lowest

$$GCM_{pk} = \sqrt{\sum_{i=1}^{n_2} \left(\frac{kd_{i2}}{D_1} - x_{i2(k-1)} - t_{i2p} \right)^2}. \quad (4.1)$$

To minimize this objective the GCM algorithm is described as:

Algorithm 4.2.1 [44] Goal Chasing Method

Step 1.

Set $x_{i20} = 0$, $S_0 = \{1, 2, \dots, n_1\}$ and $k = 1$.

Step 2.

Select for the k^{th} position in the sequence model p^* that minimizes the measure.

$$GCM_{p^*k} = \min_{p \in S_{k-1}} \left\{ \sqrt{\sum_{i=1}^{n_2} \left(\frac{kd_{i2}}{D_1} - x_{i2(k-1)} - t_{i2p} \right)^2} \right\}.$$

Step 3.

If more copies of model p^* remain to be sequenced, set $S_k = S_{k-1}$. If all copies of model p^* now already have been sequenced set $S_k = S_{k-1} - \{p^*\}$.

Step 4.

If $S_k = \emptyset$, then stop.

If $S_k \neq \emptyset$, set $x_{i2k} = x_{i2(k-1)} + t_{i2p^*}$, $i = 1, \dots, n_2$.

Set $k = k+1$ and go to step 2.

But in practice it is difficult to apply the GCM to all parts as the total number of parts required for a car is around 20,000. Therefore, the parts are represented only by their respective subassemblies. The number of subassemblies is around 20 and Toyota gives the important subassemblies additional weights. The sub assemblies include the following items (see [44]):

Engines,	Bumpers,
Transmissions,	Steering assemblies,
Frames,	Wheels,
Front axles,	Doors,
Rear axles,	Air conditioners.

Since the GCM is developed only for two levels, it considers only the variability at the subassembly and the variability at final assembly is ignored. The GCM can also be extended for all levels and is known as *Extended Goal Chasing Method* (EGCM) (cf. [39]) in which, for example with L levels and for the objective

$\sqrt{\sum_{k=1}^{D_1} \sum_{l=1}^L \sum_{i=1}^{n_i} w_l \left(\frac{kd_{il}}{D_1} - x_{ilk} \right)^2}$ sequence the product p at stage k with the lowest

$$EGCM_{pk} = \sqrt{\sum_{l=1}^L \sum_{i=1}^{n_i} \left(\frac{kd_{il}}{D_1} - x_{il(k-1)} - t_{ilp} \right)^2} \quad (4.2)$$

and to minimize this objective, a similar algorithm as that of 4.2.1 can be developed.

4.2.1 The Sum-Deviation Mixed-Model PRV-JIT Production Systems

4.2.2 Ideal Corners and Positions

Consider the PRV-JIT problem [P3.2]. Following *Kubiak* [29, 30], the unique crossing point k_{ij} satisfying

$$f_i(j - k_{ij}r_i) = f_i(j - 1 - k_{ij}r_i)$$

and the quantity

$$Z_{ij}^* = \lceil k_{ij} \rceil$$

are referred to as the *ideal corner* and *position* respectively for the j^{th} copy of product i , denoted by (i, j) , $i = 1, 2, \dots, n$; $j = 1, 2, \dots, d_i$. For example, if $f_i(x) = |x|$, x^2 , $\forall i$, then we get

$$k_{ij} = \frac{2j-1}{2r_i} \quad \text{and} \quad Z_{ij}^* = \left\lceil \frac{2j-1}{2r_i} \right\rceil.$$

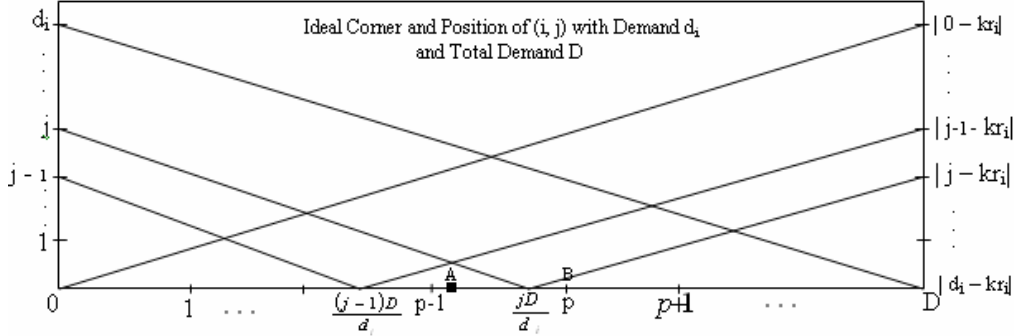


Figure 4.1: Determination of Ideal Corner and Position

In the Figure 4.1 the point A denotes the ideal corner of (i, j) and the point B placed horizontally along x-axis at a distance p from the origin O is the ideal position of (i, j) for the function $f_i(x) = |x|$.

Definition 4.2.1 An instance of PRV-JIT problem is said to be standard if $0 < d_1 \leq d_2 \leq \dots \leq d_n$, $n > 1$ and the greatest common divisor of d_1, \dots, d_n, D is 1, i.e., $\gcd(d_1, \dots, d_n, D) = 1$.

Lemma 4.2.1 [30] Ideal corners of any product i are equally spaced in $[0, D]$ with the first corner at $\frac{D}{2d_i}$, each next at a distance $\frac{D}{d_i}$ from its immediate predecessor and the last one at $\frac{D}{2d_i}$ from D . □

Lemma 4.2.3 [30] For any standard instance $0 < d_1 \leq d_2 \leq \dots \leq d_n$, $n > 2$, if at least one ideal corner is integer, then some of n sequences of product ideal positions overlap. □

The concept of ideal position for (i, j) to be produced is introduced by Inman and Bulfin [24]. They define the ideal position for (i, j) as $k_{ij} = \frac{2j-1}{2r_i}$. They consider a measure of deviation that is mathematically different, but intuitively similar to the one in PRVP. Let Z_{ij} denote the time at which the copy (i, j) is actually produced and so it incurs penalty $(Z_{ij} - k_{ij})^2$, this leads to the following problem:

$$[\text{P4.1}] \quad \text{minimize} \sum_{i=1}^n \sum_{j=1}^{d_i} (Z_{ij} - k_{ij})^2$$

Subject to

$$Z_{ij} \leq Z_{i(j+1)}, \quad i = 1, \dots, n; \quad j = 1, \dots, d_i - 1 \quad (4.3)$$

$$1 \leq Z_{ij} \leq D, \quad i = 1, \dots, n; \quad j = 1, \dots, d_i \quad (4.4)$$

$$Z_{ij} \neq Z_{i'j'}, \quad (i, j) \neq (i', j') \quad (4.5)$$

$$Z_{ij} \in \mathbb{W}, \quad i = 1, \dots, d_i \quad (4.6)$$

Constraint (4.3) ensures that the production time of each copy of a product type i is a strictly increasing function of each copy j . Constraint (4.4) guarantees that the production time of any copy of any product lies in the interval $[1..D]$. Constraint (4.5) is the only linking constraint and is not in the standard integer programming format and it specifies that only one copy of any product type can be produced in each period. By defining k_{ij} as the due-date of copy (i, j) where each copy of product is treated as a separate job, *Inman and Bulfin* [24] observe that [P4.1] may be interpreted as a single machine scheduling problem

$$|p_{(i,j)} = 1| \sum_{(i,j) \in I} (E_{(i,j)} + T_{(i,j)}), \quad (4.7)$$

where $p_{(i,j)}$, $E_{(i,j)}$ and $T_{(i,j)}$ respectively represents the processing time, earliness and tardiness of copy (i, j) and $I = \{(i, j) | i = 1, \dots, n; j = 1, \dots, d_i\}$. And in conclusion they suggest the following.

Theorem 4.2.1 *The optimal sequence for [P4.1] is to order the copies following the EDD rule for the problem (4.7). \square*

The EDD procedure can run in $O(nD)$ time and the EDD rule also gives an optimal sequence for the sum of absolute deviations as well (see [24, 29]).

4.2.2 Reduction of F_{sum} to Assignment Problem

Consider the problem [P3.2] with objective function $F_{sum}(s)$, where each f_i is assumed to be a unimodal, nonnegative convex function, $i = 1, 2, \dots, n$. Let Z_{ij} denote the period in which the item (i, j) is actually produced. Then the problem F_{sum} can be written equivalently in the following form [P4.2] which facilitates decomposition into single part type problems with an appropriate linking constraint.

$$[\text{P4.2}] \quad \text{minimize} \left[F_{sum}(s) = \sum_{i=1}^n \left\{ \sum_{k=0}^{Z_{i1}-1} f_i(0 - kr_i) + \sum_{k=Z_{i1}}^{Z_{i2}-1} f_i(1 - kr_i) + \dots + \sum_{k=Z_{id_i}}^D f_i(d_i - kr_i) \right\} \right]$$

subject to the constraints (4.3), (4.4), (4.5), (4.6).

If we remove the constraint (4.5) from problem [P4.2] then the problem [P4.2] could be partitioned into n independent sub-problems. Moreover, obtaining an optimal solution for each product in isolation is straight forward. For that, let us define

$$f_j^i(k) = f_i(j - kr_i), \quad j = 0, 1, 2, \dots, d_i; \quad k = 0, 1, \dots, D, \quad (4.8)$$

which represents the inventory or shortage cost in period k if the item (i, j) and only j -copies of the i^{th} product have been produced by period k . Then the i^{th} product type sub-problem [P4.2i] can be started as below

$$[P4.2i] \quad \text{minimize} \left[F_{sum}^i(s) = \sum_{j=0}^{d_i} \sum_{k=Z_{ij}}^{Z_{i(j+1)}-1} f_j^i(k) \right]$$

subject to the constraints (4.3), (4.4), (4.6), where $Z_{i0} = 0, Z_{i(d_i+1)} - 1 = D$ and $f_0^i(0) = f_{d_i}^i(D) = 0$. Now, we have

Theorem 4.2.2 [33] *An Optimal Solution to [P4.2i] is given by*

$$Z_{ij} = Z_{ij}^* = \left\lceil k_{ij} \right\rceil, j = 1, \dots, d_i$$

with optimal objective value

$$F_{sum}^i(s) = \sum_{k=1}^D f^i(k)$$

where

$$f^i(k) = \inf_j f_j^i(k), k = 1, \dots, D. \quad (4.9)$$

Moreover, if f_i is symmetric, then

$$Z_{ij} = Z_{ij}^* = \left\lceil \frac{2j-1}{2r_i} \right\rceil. \quad \square$$

Now, if all of the copies can be placed in their ideal positions, i.e. if $Z_{ij}^*, (i, j) \in I$, satisfy the linking constraint (4.3), then we shall, obviously, have an optimal solution to the sequencing problem [P4.2], in which the product i will contribute the cost $f^i(k)$ to the total cost of the solution. This will unfortunately not be the case in general and we must somehow come to terms with the resolution of conflicts between various copies for their ideal positions in a way that preserves the ordinary relations defined by inequalities in (4.8).

The idea for the resolution of conflict between various copies is simple. It makes sense to reduce the problem F_{sum} to an assignment problem. To formulate the assignment problem, let $\mathfrak{C} = \{(i, j, k) \mid i = 1, \dots, n; j = 1, \dots, d_i; k = 1, \dots, D\}$. Following Kubiak and Sethi [32, 33] for any $((i, j), k) \in \mathfrak{C}$, denoted by C_{ijk} the cost of assigning item (i, j) to the k^{th} period is defined by

$$C_{ijk} = \begin{cases} \sum_{l=k}^{Z_{ij}^*-1} \psi_{ijl} & \text{if } k < Z_{ij}^* \\ 0 & \text{if } k = Z_{ij}^* \\ \sum_{l=Z_{ij}^*}^{k-1} \psi_{ijl} & \text{if } k > Z_{ij}^* \end{cases} \quad (4.10)$$

where

$$\psi_{ijl} = \left| f_j^i(l) - f_{j-1}^i(l) \right| = \begin{cases} f_j^i(l) - f_{j-1}^i(l) & \text{for } l < Z_{ij}^* \\ f_{j-1}^i(l) - f_j^i(l) & \text{for } l \geq Z_{ij}^* \end{cases} \quad (4.11)$$

which represents the excess cost of having j -copies of product i produced by period over l having $(j-l)$ copies of the same product produced by period l .

Now, to analyze the definition of C_{ijk} in (4.10), if $k = Z_{ij}^*$, then the j^{th} copy of product i has its ideal position and $C_{ijk} = 0$. If the item (i, j) is produced too soon, i.e. $k < Z_{ij}^*$, then excess inventory costs ψ_{ijl} are incurred in periods from $l = k$ to $l = Z_{ij}^* - 1$. On the other hand if $k > Z_{ij}^*$, i.e. the item (i, j) is produced so late, then the excess shortage costs ψ_{ijl} are incurred on account of not having it in periods $l = Z_{ij}^*$ to $l = k - 1$. And with some manipulations (4.10) can be written as

$$C_{ijk} = \sum_{l=\min(k, Z_{ij}^*)}^{\max(k, Z_{ij}^*)-1} \psi_{ijl}, \quad (4.12)$$

where we use the convention that $\sum_{l=k}^k a_k = 0$ for $k' < k$. Now, the assignment variables for any $((i, j), k) \in \mathfrak{C}$ can be defined as

$$y_{ijk} = \begin{cases} 1 & \text{if } (i, j) \text{ is produced in time period } k \\ 0 & \text{otherwise} \end{cases}$$

Then the assignment problem corresponding to the problem F_{sum} is

$$[\text{P4.3}] \quad \text{minimize} \left[F(s) = \sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk} y_{ijk} \right]$$

subject to

$$\sum_{i=1}^n \sum_{j=1}^{d_i} y_{ijk} = 1, \quad k = 1, 2, \dots, D \quad (4.13)$$

$$\sum_{k=1}^D y_{ijk} = 1, \quad i = 1, 2, \dots, n; j = 1, 2, \dots, d_i. \quad (4.14)$$

In order to find the optimal solution for the assignment problem [P4.3], we consider it in terms of graph. The objective is to find a smallest perfect matching in the weighted complete bipartite graph $G_w = (V_1 \cup V_2, E)$, where the vertex set $V_1 = [1 \dots D]$ represents the production time periods, V_2 is the set of all parts (i, j) , and an edge $e = [k, (i, j)] \in E$ has the weights $w_e = C_{ijk}$. Since there are $2D$ nodes in this assignment problem [P4.3], it can be solved for optimality in $O(D^3)$ time by Algorithm 2.4.1.

Let $S \subseteq \mathfrak{C}$, and define $c(S) = \sum_{((i,j),k) \in S} C_{ijk}$ and call the set S is feasible for the problem F_{sum}

if it satisfies the following three constraints:

- c₁: For each $k, k = 1, 2, \dots, D$ there is exactly one $(i, j), i = 1, 2, \dots, n; j = 1, 2, \dots, d_i$, such that $((i, j), k) \in S$, i.e., exactly one copy is produced at one time unit.
- c₂: For each $(i, j), i = 1, 2, \dots, n; j = 1, 2, \dots, d_i$, there is exactly one $k, k = 1, 2, \dots, D$, such that $((i, j), k) \in S$, i.e., each copy is produced exactly once.
- c₃: If $((i, j), k), ((i, j'), k') \in S$ and $k < k'$ then $j < j'$, i.e. lower indices copies are produced earlier.

Constraints c_1 and c_2 are related to the assignment problem. Constraint c_3 imposes an order on copies of a product.

Consider any feasible set S of D triples $((i, j), k)$ and define the sequence $s = (s_1, \dots, s_D)$ with $s_k = i$ if $((i, j), k) \in S$ for some $j = 1, \dots, d_i$ corresponding to the set S , and

$$y_{ijk} = \begin{cases} 1 & \text{if } ((i, j), k) \in S \\ 0 & \text{otherwise} \end{cases}$$

Then the sequence s is feasible for any given instance with demand vector (d_1, \dots, d_n) . Now, the implication of reducing the problem F_{sum} to the assignment problem [P4.3] is the following:

Theorem 4.2.3 [33] *If $S \subseteq \mathcal{K}$ satisfies c_1 and c_2 , then a feasible S^* such that $c(S) \geq c(S^*)$ can be determined in $O(D)$ time. Moreover, each copy in the sequence s^* from S^* preserves the order that it has in the sequence s from S . Furthermore, for any sequence s constructed from the feasible S*

$$y_{ijk} = 0, \quad j \geq k + 1; i = 1, \dots, n; j = 2, \dots, d_i \quad (4.15)$$

and $y_{ij(D-k)} = 0, \quad j \leq d_i - k - 1; i = 1, \dots, n; j = 1, \dots, d_{i-1} \quad (4.16)$

□

Clearly, pre-assigning zero values to a number of variables would reduce the computational effort for solving the problem [P4.3]. It is possible to count the number of variables y_{ijk} that can be assigned zero value each a priori. The number of variables both in (4.15) and (4.16) is $\frac{d_i(d_i - 1)}{2}$. But some variables occur in both (4.15) and (4.16) and if D is sufficiently large and d_i is sufficiently small, then $2(d_i - 1) \leq D$ and there is no repetition of variables in (4.15) and (4.16). Thus, the number of variables that can be pre-assigned a zero value is $\sum_{i=1}^n d_i(d_i - 1)$. Now, for any feasible $S \subseteq \mathcal{K}$, let $F_i(s)$ denote the parts of the objective function of the assignment problem [P4.3], attributed to product type i corresponding to sequence s derived from S , then we have the optimality theorem.

Theorem 4.2.4 [33] *For any feasible $S \subseteq \mathcal{K}$ and s derived from S ,*

$$F_{sum}^i(s) = F_i(s) + \sum_{k=1}^D f^i(k) \quad (4.17)$$

and then

$$F_{sum}(s) = c(S) + \sum_{k=1}^D \sum_{i=1}^n f^i(k) \quad (4.18)$$

□

As the term $\sum_{k=1}^D \sum_{i=1}^n f^i(k)$ is independent of the set S , that is constant, an optimal solution to the problem F_{sum} would be an immediate consequence if an optimal feasible subset S is obtained.

A direct consequence of Theorem 4.2.3 and Theorem 4.2.4 is that the optimal set S for [P4.3] that is feasible is also optimal for the problem F_{sum} . Moreover, any optimal set S for [P4.3] can be converted into an optimal set S^* for F_{sum} (cf. [33]) and thus, an account of Theorem 4.2.4, S^* provides the optimal sequence s^* for PRV-MMJITSP F_{sum} .

Again, from Theorem 4.2.4, we can construct an optimal feasible set S^* for F_{sum} from any optimal set S for [P4.3] which is feasible or not, and thereby solving the problem F_{sum} . The explicit procedure is stated in the following theorem.

Theorem 4.2.5 [33] *Let S be an optimal set to [P4.3] with the corresponding sequence $y_{ijk} \{i = 1, \dots, n; j = 1, \dots, d_i; k = 1, \dots, D\}$. Then*

$$x_{ik} = \sum_{l=1}^k \sum_{j=1}^{d_i} y_{ijl}, \quad i = 1, \dots, n; k = 1, \dots, D$$

is an optimal solution to F_{sum} . Furthermore, for $i = 1, \dots, n$

$$Z_{ij} = 1; x_{il} = 1$$

$$Z_{ij} = k, x_{i(k-1)} = j - 1$$

and $x_{ik} = j, j = 1, \dots, d_i; k = 2, \dots, D$ is an optimal solution to [P4.2]. □

4.2.3 Reduction of F_{sum} to Integer Linear Programming

Since in a sum-deviation MMJITSP, a part $(i, j), i = 1, 2, \dots, n; j = 1, 2, \dots, d_i$, can be produced at any time period $k, k = 1, 2, \dots, D$, then we can define a complete weighted bipartite graph $G_w = (V_1 \cup V_2, E)$, where $V_1 = [1 \dots D]$ represents the production time periods, V_2 is the set of all parts (i, j) and for any edge $e = [k, (i, j)] \in E$, we assign the edge weight $w_e = C_{ijk}$. Let $A = (a_{ij})_{2D \times D^2}$ be the vertex edge incidence matrix of the graph G_w . Let $w \in R^{D^2}$ be the vector of the weights and $y \in \{0, 1\}^{D^2}$ be the column vector of the corresponding assignment variables. Then the problem F_{sum} equivalently can be written in the form of ILP as

$$\begin{aligned} \text{[P4.4]} \quad & \text{minimize } w^T y \\ & \text{subject to} \\ & Ay = 1 \\ & y \in \{0, 1\}^{D^2}. \end{aligned} \tag{4.19}$$

Clearly the matrix A is totally unimodular and as a result due to Theorem 2.4.1, we can solve ILP (4.19) by relating into LP for optimality.

The dual of the linear relation of (4.19) is

$$\begin{aligned} \text{[P4.5]} \quad & \text{minimize } 1^T z \\ & \text{subject to} \\ & z^T A \leq w^T \\ & z - \text{free} \end{aligned} \tag{4.20}$$

The problem (4.20) can be interpreted as finding vertex weights $z \in R^{2D}$ such that the sum of z_l for all $l \in V_1 \cup V_2$, is maximal and such that for all edges $[k, (i, j)] \in E$, we have $z_k + z_{ij} \leq C_{ijk}$. Therefore, we have

Theorem 4.2.6 [35] *A sequence $s = (s_1, \dots, s_D)$ with objective value s^* for the problem F_{sum} is optimal if and only if there is a minimum weighted perfect matching M with a weight function $z : V_1 \cup V_2 \rightarrow R$ such that*

$$\begin{aligned} z_k + z_{ij} &\leq C_{ijk} \quad \forall [k, (i, j)] \in E \\ \sum_{k \in V_1} z_k + \sum_{(i, j) \in V_2} z_{ij} &= \sum_{[k, (i, j)] \in M} C_{ijk} = s^*. \end{aligned} \quad \square$$

4.2.4 The Cyclic Sequences to F_{sum}

By a *cyclic solution* for MMJITSP, we mean, if $s = (s_1, \dots, s_D)$ is an optimal sequence for an instance of MMJITSP with demand vector (d_1, \dots, d_n) then s^m for any $m \geq 1$ is an optimal sequence for the instance with demand vector (md_1, \dots, md_n) , where s^m is a concatenation of m -copies of s . *Miltenburg* [37] and *Miltenburg and Sinnamon* [39] observe the existence of the cyclic sequences for the problem F_{sum} . Such a cyclic sequence can be found under the assumption that $f_i = f, \forall i, i = 1, \dots, n$ where f is unimodal symmetric convex nonnegative with minimum $f(0) = 0$ and $f(y) > 0$ for $y \neq 0$ (cf. [27], see also [19]). The result is also true if all f_i are convex, symmetric and equal in the interval $(0, 1)$ but not true even if a single f_i is asymmetric (cf. [26]).

An instance for the MMJITSP is said to be *even* if the demand vector is of the form $(2d_1, \dots, 2d_n)$ for some vector (d_1, \dots, d_n) of positive integers, and feasible sequence have

length $2D$, where $D = \sum_{i=1}^n d_i$. Then we have

Theorem 4.2.7 [27] *Let*

$$s = (s_1, \dots, s_D, s_{D+1}, \dots, s_{2D})$$

be a feasible sequence for the problem F_{sum} with demand vector $(2d_1, \dots, 2d_n)$. Then, a feasible sequence

$$s^* = (s_1^*, \dots, s_D^*, s_{D+1}^*, \dots, s_{2D}^*)$$

where i occurs d_i times in the first half s_1^, \dots, s_D^* and d_i times in the second half $s_{D+1}^*, \dots, s_{2D}^*$ can be constructed such that*

$$F_{sum}(s^*) \leq F_{sum}(s). \quad \square$$

Now, with the help of this Theorem, we are ready to state the main result of this section as

Theorem 4.2.8 [27] *Let s be an optimal sequence for the problem F_{sum} with the demand vector (d_1, \dots, d_n) . Then the concatenation s^m for any $m (\geq 1)$ copies of s , is optimal sequence for the problem F_{sum} with demand vector (md_1, \dots, md_n) .*

Proof: The underlying idea to prove this theorem is strictly based on the reference [27] and is proved under the mathematical induction on m . The theorem obviously holds for $m = 1$. Now, it remains to prove that the theorem holds for $m + 1$ under the assumption that the theorem holds for any $m \geq 1$. For that, consider an optimal sequence $s' = (s'_1, \dots, s'_{(m+1)D})$ for the demand vector $((m+1)d_1, \dots, (m+1)d_n)$. Now, if m is odd. Then by Theorem 4.2.7, this sequence can be transformed without increasing the cost into a sequence $s^* = (s_1^*, \dots, s_{\frac{m+1}{2}D}^*, s_{\frac{m+1}{2}D+1}^*, \dots, s_{(m+1)D}^*)$, where i occurs $\frac{m+1}{2}d_i$ times in each of the two halves of s^* . Thus, each half must be optimal for F_{sum} with demand vector $(\frac{m+1}{2}d_1, \dots, \frac{m+1}{2}d_n)$. Therefore, by the induction hypothesis, each half is the concatenation of $\frac{m+1}{2}$ copies of s , and the theorem holds for $m + 1$ if it holds for any $m \geq 1$. Now, if m is even, then consider a sequence ss' for F_{sum} with demand vector $((m+2)d_1, \dots, (m+2)d_n)$. We have $F_{sum}(ss') = F_{sum}(s) + F_{sum}(s')$ (see [37]). By Theorem 4.2.7, ss' can be transformed without increasing the cost into a sequence $s^{**} = (s_1^{**}, \dots, s_{\frac{m+2}{2}D}^{**}, s_{\frac{m+2}{2}D+1}^{**}, \dots, s_{(m+2)D}^{**})$, where i occurs $\frac{m+2}{2}d_i$ times in each of the two halves of s^{**} . Thus, each half must be optimal for F_{sum} with demand vector $(\frac{m+2}{2}d_1, \dots, \frac{m+2}{2}d_n)$. Therefore, by the induction hypothesis, each half is the concatenation of $\frac{m+2}{2}$ copies of s and

$$\begin{aligned} F_{sum}(ss') &= F_{sum}(s) + F_{sum}(s') \geq (m+2)F_{sum}(s) \\ \Rightarrow F_{sum}(s') &\geq (m+1)F_{sum}(s) \end{aligned}$$

which proves the theorem holds for $m + 1$, if it holds for any even $m \geq 1$. This proves the theorem. \square

Therefore, the optimal MMJIT sequences for F_{sum} are cyclic. This result provides an important theoretical support to the usual for Just-in-Time systems practice of repeating relatively short sequence to build a sequence for a longer time horizon, *Monden* [41] and *Miltenburg* [37]. It has also important consequences for the computational time complexity of all existing algorithms for PRVP. All these time complexities depend on the magnitude of demands d_1, \dots, d_n and consequently on the magnitude of number D . The only known polynomial time, with respect to D and n , optimization algorithm for MMJITSP has time complexity $O(D^3)$, and is Hungarian Method for Assignment Problem as presented in Section 2.4.3 (see [33]). Theorem 4.2.8 makes it possible to reduce each of these demands by the factor of m , where m is the greatest common

divisor of numbers d_1, \dots, d_n , in the computations of optimal MMJIT sequences for F_{sum} . The Euclidean Algorithm can find m in $O(n \log D)$ steps (see [27]).

4.2.5 Dynamic Programming for F_{sum}

The computational time complexity of solving the assignment problem [P4.3] grows exponentially, if the demands for the problem F_{sum} are in exponential form. Then to determine an optimal sequence for the problem F_{sum} becomes difficult for other than extremely small problems. The dynamic programming (DP) algorithm proposed by *Miltenburg et al.* [40] determines the optimal sequence, for the first time, of the problem F_{sum} with large input size. Here we describe a DP algorithm to solve F_{sum} based on *Miltenburg et al.* [40], where the DP algorithm is developed to solve the sum-squared joint usage and loading problem.

Let $d = (d_1, \dots, d_n)$ be the demand vector of F_{sum} . Define states in a sequence as $X = (x_1, \dots, x_n)$, where x_i is a nonnegative integer representing the production of exactly x_i units of product i , $x_i \leq d_i, \forall i$. Let e_i be the usual i^{th} unit vector with n entries, all of which are zero except a single 1 in the i^{th} place. A state X can be sequenced in the first k stages if

$$k = |X| = \sum_{i=1}^n x_i.$$

Now, let

$$\phi(X) = \min \sum_{j=1}^k \sum_{i=1}^n f_i(x_i - jr_i)$$

and

$$g(X) = \sum_{i=1}^n f_i(x_i - kr_i).$$

Then the following DP recursion holds for $\phi(X)$:

$$\begin{aligned} \phi(X) &= \phi(x_1, \dots, x_n) \\ &= \min \{ \phi(X - e_i) + g(X) \mid i = 1, \dots, n; x_i \geq 1 \} \\ \phi(\emptyset) &= \phi(X \mid \forall x_i = 0) = \phi(0, \dots, 0) = 0. \end{aligned} \tag{4.20}$$

Clearly, $\phi(X) \geq 0$, and from the definition of r_i 's

$$g(X \mid \forall x_i = d_i) = 0.$$

Now, we have

Theorem 4.2.9 [40] *The DP recursion (4.20) solves the MMJIT sequencing problem F_{sum} in $O\left(n \prod_{i=1}^n (d_i + 1)\right)$ time.*

Proof [40]: Clearly, $g(X)$ represents the ‘‘contribution’’ of each product to the objective function in stage k ($1 \leq k \leq D$). The minimization in (4.20) determines the product to be

sequenced in stage k. Now, since $x_i = 0, 1, \dots, d_i, \forall i = 1, \dots, n$, then the cardinality of the set containing all states X, in the DP recursion is given by $\prod_{i=1}^n (d_i + 1)$. To calculate $\phi(X)$ for a state X from recursion (4.20), we must calculate at most n values $\phi(X - e_i)$, whose calculation requires $O(n)$ time. Hence, the calculation of $\phi(X)$ can be done in $O(n)$ steps for each state X. Therefore, the time complexity for the entire problem is

$$O\left(n \prod_{i=1}^n (d_i + 1)\right). \quad \square$$

The total number of feasible sequences for F_{sum} is $\frac{D!}{d_1! \dots d_n!}$ which is considerably larger than the number of sets in the DP recursion. Moreover,

$$\prod_{i=1}^n (d_i + 1) \leq \left(\frac{d_1 + d_2 + \dots + d_n + n}{n}\right)^n = \left(\frac{D + n}{n}\right)^n.$$

Hence, although the number of sets may grow at exponential rate with n, its growth rate is polynomial in D, and so the DP procedure is effective for small n even with large D.

CHAPTER 5

BOTTLENECK MIXED-MODEL JIT PRODUCTION SYSTEMS

In this chapter, we study the bottleneck objective of MMJITSP, which seeks to minimize the max-deviation between ideal production and actual production for each product at each stage and as a result a smooth sequence in every time period can be obtained (see [50]). *Kubiak et al.* [34] prove that the problem \tilde{F}_{\max}^{wa} with only two levels is strongly *NP*-hard by transforming an instance of already known strongly *NP*-hard 3-PARTITION problem into an instance of the ORVP \tilde{F}_{\max}^{wa} with only two levels in pseudo-polynomial time. Hence, as the ORVP \tilde{F}_{\max}^{wa} is *NP*-hard, an efficient algorithm for the optimality is unlikely to exist. However, by transforming the problems \tilde{F}_{\max}^{wa} and \tilde{F}_{\max}^{ws} concisely into the matrix representation, *Kubiak et al.* [34] provide a dynamic programming (DP) for optimal sequence and is the topic of Section 5.1.

We study the ORVP \tilde{F}_{\max}^{wa} in Section 5.1 and all the remaining sections of this chapter are concerned to the study of the bottleneck PRV problem F_{\max} and its structural properties.

5.1 Dynamic Programming for \tilde{F}_{\max}^{wa}

Since the problem \tilde{F}_{\max}^{wa} is strongly *NP*-hard, polynomially bounded procedures for it are extremely unlikely to exist. Here we describe an implicit enumeration dynamic programming (DP) procedure which can optimize the problem \tilde{F}_{\max}^{wa} . By definition, we have

$$\begin{aligned} x_{ilk} - y_{lk}r_{il} &= \sum_{p=1}^{n_l} t_{ilp}x_{p1k} - r_{il} \sum_{p=1}^{n_l} \sum_{i=1}^{n_l} t_{ilp}x_{p1k} \\ &= \sum_{p=1}^{n_l} \left(t_{ilp} - r_{il} \sum_{i=1}^{n_l} t_{ilp} \right) x_{p1k} \\ &= \sum_{p=1}^{n_l} \delta_{ilp} x_{p1k}, \end{aligned}$$

where

$$\delta_{ilp} = t_{ilp} - r_{il} \sum_{i=1}^{n_l} t_{ilp}.$$

Since $w_{il} \geq 0$, $x_{ilk} \geq 0$ and $r_{il} > 0$, then the deviation for part i of level l at stage k for \tilde{F}_{\max}^{wa} would be

$$w_{il} |x_{ilk} - y_{lk}r_{il}| = \left| w_{il} \left(\sum_{p=1}^{n_l} \delta_{ilp} x_{p1k} \right) \right| = \left| \sum_{p=1}^{n_l} \gamma_{ilp} x_{p1k} \right|,$$

where $\gamma_{ilp} = w_{il}\delta_{ilp}$ is the measure of the weighted deviation in the usage of part i in level l from the proportional usage per unit of product p . Let $\Gamma = (\gamma_{ilp})_{n \times n_1}$ be the matrix where $n = \sum_{l=1}^L n_l$ is the total number of different parts and products. Each row of Γ corresponds to either a product or a part at the corresponding levels. The value γ_{ilp} will be the element appearing in the $\left(\sum_{m=1}^{l-1} n_m + i\right)$ th row and the p^{th} column of the matrix Γ . The maximum norm of a vector $a = (a_1, \dots, a_n)$ is defined to be $\|a\|_1 = \max_{1 \leq i \leq n} \{a_i\}$. Then the objective function $\tilde{F}_{\max}^{wa}(\tilde{X})$ can be written as $\tilde{F}_{\max}^{wa}(\tilde{X}) = \max_k \|\Gamma X_k\|_1$, where $X_k = (x_{11k}, \dots, x_{n_1k})$ is the cumulative, level 1 production vector through the first k stages. Let the demand vector at level 1 be $d = (d_{11}, \dots, d_{n_11}) = (d_1, \dots, d_{n_1})$ and the states in a sequence be $X = (x_1, \dots, x_{n_1})$ with $|X| = \sum_{i=1}^{n_1} x_i$, $i = 1, \dots, n_1$ where x_i is the cumulative production of product i , $x_i \leq d_i$. Let $e_i = (0, \dots, 1, \dots, 0)$ be the unit vector with n_1 entries all of which are zero except for a single 1 in the i^{th} row. Let $\phi(X)$ be the minimum value of the maximum deviation for all parts and products over all partial sequences which lead to state X . The norm $\|\Gamma X\|_1$ represents the maximum deviation of actual production from desired one over all products and parts in state X at stage $k = |X|$. The following DP recursion holds for $\phi(X)$ (cf. [34], see also [19]):

$$\begin{aligned} \phi(\emptyset) &= \phi(X : X = 0) = 0 \\ \phi(X) &= \phi(x_1, \dots, x_{n_1}) = \min_i \left\{ \max \left\{ \phi(X - e_i), \|\Gamma X\|_1 \right\} : i = 1, \dots, n_1; x_i \geq 1 \right\} \end{aligned} \quad (5.1)$$

It can be observed that $\phi(X) \geq 0$ and $\|\Gamma(X : X = d)\|_1 = 0$ for any state X .

Theorem 5.1.1 [34] The DP recursion (5.1) solves the MMJITSP \tilde{F}_{\max}^{wa} in

$$O\left(n_1 n \prod_{i=1}^{n_1} (d_i + 1)\right)$$

time.

Proof: As $x_i = 0, 1, \dots, d_i$, $\forall i = 1, \dots, n_1$, the cardinality of the set containing all states X , in the DP recursion is

$$\prod_{i=1}^{n_1} (d_i + 1).$$

On the other hand, any state X can be generated from at most n_1 states of the form $X - e_i$. The computation time for $\|\Gamma X\|_1$ is $O(n_1 n)$. Thus, the total computation time for the entire problem is

$$O\left(n_1 n \prod_{i=1}^{n_1} (d_i + 1)\right). \quad \square$$

The number of feasible sequences for any instance of the problem \tilde{F}_{\max}^{wa} is $\frac{D_1!}{d_1! \dots d_{n_1}!}$ which is considerably larger than the number of states in the DP recursion. The relation $\prod_{i=1}^{n_1} (d_i + 1) \leq \left(\frac{D_1 + n_1}{n_1} \right)^{n_1}$ shows that the DP algorithm is effective for small number of products even with large number of copies. During the enumeration process, an excessive amount of time is reduced by using some fast heuristic as a filter which eliminates any state from DP's state that would lead to no optimality. *Kubiak et al.* [34] present two myopic heuristics for generating the filter. If the heuristics yield near-optimal sequences, then the number of state could be reduced. The DP algorithm progresses through the state in the forward direction of increasing the cardinality as the procedure generates all states X with $|X| = k$ before $|X| = k + 1$, $\forall k = 1, \dots, D_1$.

The DP for \tilde{F}_{\max}^{wa} can also be modified for the problem \tilde{F}_{sum}^{ws} (cf. [34], see also [20]):

The objective function $\tilde{F}_{sum}^{ws}(\tilde{X})$ can be written as

$$\text{minimize } \sum_{k=1}^{D_1} (\|\Omega X_k\|_2)^2$$

where Ω is the deviation matrix $(\sqrt{w_{il}} \delta_{ilp})_{n \times n_1}$ and $\|a\|_2 = \sqrt{\sum_{i=1}^n (a_i)^2}$ is the Euclidean norm of the vector $a = (a_1, \dots, a_n)$. Define $\Phi(X)$ to be the minimum total squared deviation for all parts and products over all partial sequences of X . Let $\theta(X) = (\|\Omega X\|_2)^2$ be the squared sum of the deviations of actual production from the desired one for all parts and products when X is the amount of product produced. Then the following DP recursion holds for $\Phi(X)$ (cf. [34], see also [20]):

$$\begin{aligned} \Phi(\emptyset) &= \Phi(X : X \equiv 0) = 0 \\ \Phi(X) &= \min\{\Phi(X - e_i) + \theta(X) : i = 1, \dots, n_1; x_i \geq 1\} \end{aligned} \quad (5.2)$$

Thus, it is always the case that $\Phi(X) \geq 0$ and that $\theta(X : X = d) = 0$.

5.2 Reduction to Release Date/ Due – Date Decision Problem

As the general solution techniques do not exist to solve the integer programming formulation of the problem F_{\max}^a , a special solution procedure is developed for the specific problem under consideration. If (i, j) is produced in period k , $k = 1, 2, \dots, D$, then $x_{ik} = j$ and the penalty associated with (i, j) under F_{\max}^a in period k is

$$f_j^i(k) = \left. \begin{aligned} &|j - kr_i|, \\ & \left. \begin{aligned} i &= 1, 2, \dots, n \\ j &= 0, 1, \dots, d_i \\ k &= 1, 2, \dots, D \end{aligned} \right\} \end{aligned} \right\} \quad (5.3)$$

where $j = 0$ has been introduced to account for the periods k in which $x_{ik} = 0$. All together there are $D + n$ such individual penalty functions $f_j^i(k)$ depending upon the values of $i = 1, \dots, n$ and $j = 0, 1, \dots, d_i$ under the consideration of k is continuous time variable running from 0 to D , i. e. $k \in [0, D]$. There are $d_i + 1$ individual penalty

functions of a single variable $k \in [0, D]$ for each $i = 1, \dots, n$, associated with individual copies of product i as shown in Figure 5.1 for $d_i = 3$ and $D = 14$.

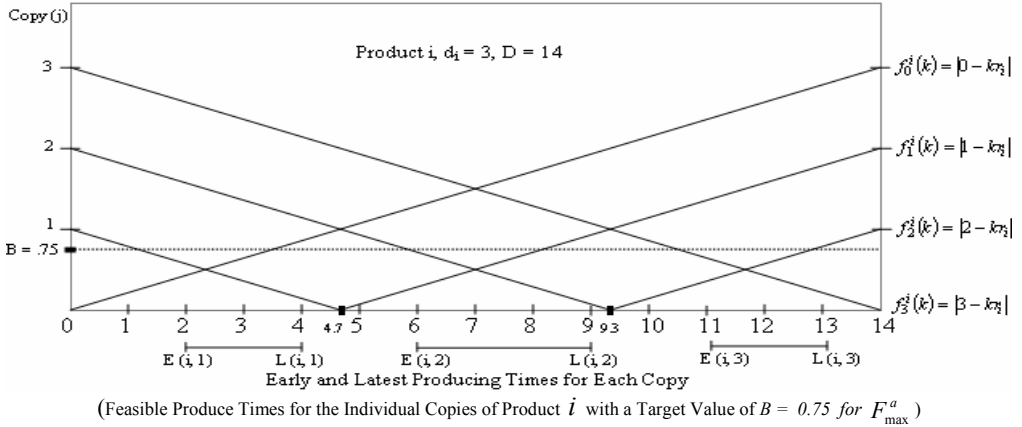


Figure 5.1: Level Curves for the Deviation of Ideal Production from Each Copy over all Time Periods

Since Z_{ij} be the completion time of (i, j) then the penalties 'attributable' to product i , for a particular product sequence are given by

$$g_j^i(k) = \begin{cases} f_j^i(k) & \text{for } Z_{ij} \leq k < Z_{i(j+1)} \\ 0 & \text{otherwise} \end{cases} \quad (5.4)$$

Now, if

$$f^i(k) = \max_{0 \leq j \leq d_i} g_j^i(k)$$

then $f^i(k)$ is the envelope of the $f_j^i(k)$ functions and has a saw tooth shape, as shown in Figure 5.1 (e). Clearly the envelope is non-convex function for each i , $i = 1, \dots, n$. Now, minimizing the objective function of F_{\max}^a is equivalent to finding a sequence which minimizes:

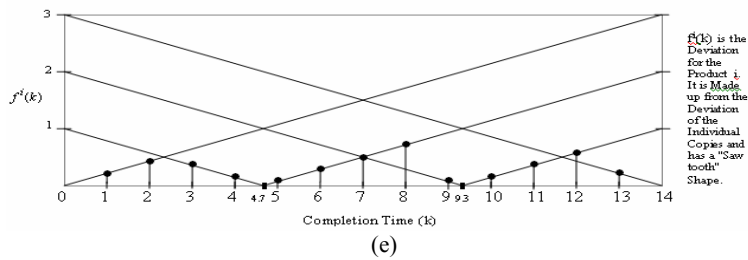
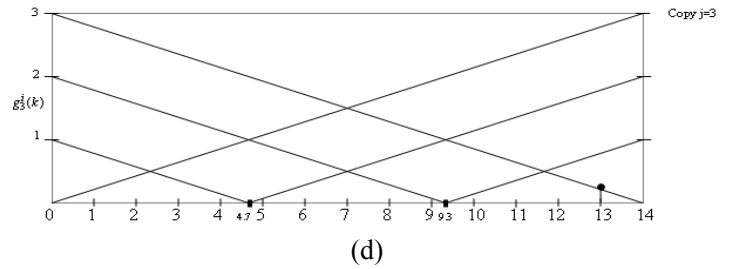
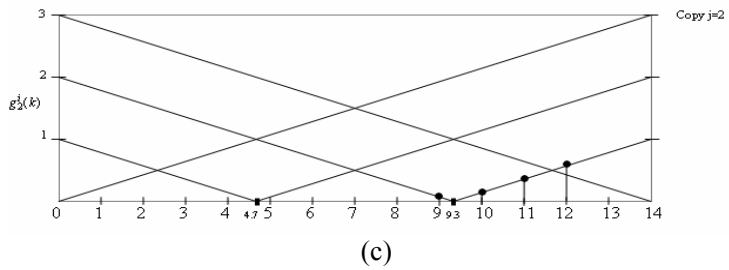
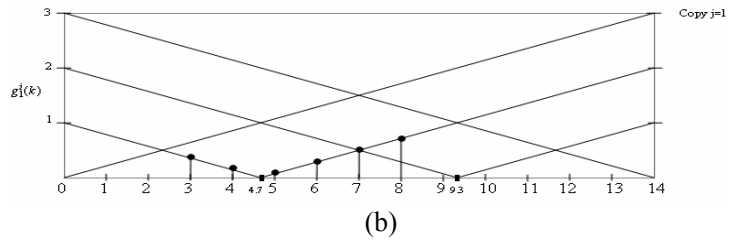
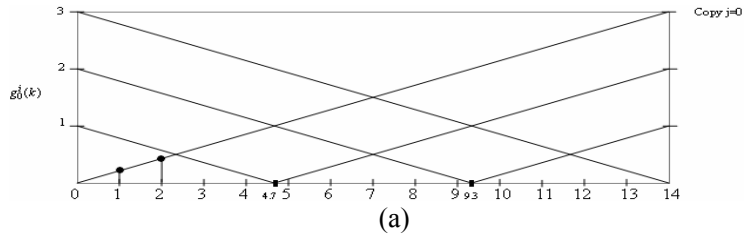
$$Z = \max_{\substack{1 \leq i \leq n \\ 1 \leq k \leq D}} \{f^i(k) \mid i = 1, \dots, n; k = 1, \dots, D\}.$$

Hence to minimize the objective functions of F_{\max}^a are made, those functions specifically created by (5.4) will be under examination. Denote a threshold value for the objective function of the problem F_{\max}^a by the variable $B \in Q$, where Q is the set of rational numbers. Then the assembly goal is to determine the smallest possible B for which a sequence $s = (s_1, s_2, \dots, s_D)$ can be constructed in which each copy (i, j) has a completion time Z_{ij} such that $g_j^i(k) \leq B$ for $k \in [Z_{ij}, (Z_{i(j+1)} - 1)]$. For the threshold value B , a copy (i, j) cannot produce before $k \geq 2$ if $f_j^i(k-1) = j - (k-1)r_i > B$, and can produce at time period $k \geq 1$ if $f_j^i(k) = j - kr_i \leq B$.

Thus, any fixed threshold (target) value B allows the calculation of a release date and due date for a specific copy of a product. By letting $E(i, j)$ be the *earliest feasible producing time* for copy (i, j) then it must have:

$$j - (E(i, j) - 1)r_i > B, \text{ and} \quad (5.5)$$

Given a Sequence for Product i , where $Z_{i1} = 3, Z_{i2} = 9, Z_{i3} = 13$, the Graphs Show the Deviation for the Individual Copies. The symbol "●" Show that the Deviation is Measure only at Integral Times



$f_i^j(k)$ is the Deviation for the Product i . It is Made up from the Deviation of the Individual Copies and has a "Saw tooth" Shape.

Figure 5.2: Deviation "Attributable" to Each Copy of a Product

$$j - E(i, j)r_i \leq B. \quad (5.6)$$

So, the earliest feasible producing time will be the unique integer satisfying

$$\left\lceil \frac{1}{r_i} \right\rceil [j - B] \leq E(i, j) < \left\lceil \frac{1}{r_i} \right\rceil [j - B] + 1 \quad (5.7)$$

That is,

$$E(i, j) = \left\lceil \frac{j - B}{r_i} \right\rceil. \quad (5.8)$$

It is a general convention that when $E(i, j)$ will be calculated to be a nonpositive integer, then we set $E(i, j) = 1$. Similarly, by letting $L(i, j)$ to be the *latest feasible producing time* for (i, j) then the latest time $L(i, j) \leq D$ at which (i, j) can produce and will satisfy the threshold value must satisfy:

$$(L(i, j) - 1)r_i - (j - 1) \leq B \quad (5.9)$$

$$\text{and } L(i, j)r_i - (j - 1) > B. \quad (5.10)$$

Thus the latest feasible producing time will be the unique integer satisfying

$$\left\lceil \frac{1}{r_i} \right\rceil [(j - 1) + B] < L(i, j) \leq \left\lceil \frac{1}{r_i} \right\rceil [(j - 1) + B] + 1. \quad (5.11)$$

That is,

$$L(i, j) = \left\lceil \frac{j - 1 + B}{r_i} + 1 \right\rceil. \quad (5.12)$$

We assume that $L(i, j) = D, \forall L(i, j) \geq D + 1$.

Hence, the release date and due date for each copy (i, j) under F_{\max}^a are considered to be

$$E(i, j) = \left\lceil \frac{j - B}{r_i} \right\rceil \text{ and } L(i, j) = \left\lceil \frac{j - 1 + B}{r_i} + 1 \right\rceil, \quad (5.13)$$

respectively.

The similar results for the problem F_{\max}^s are (cf. [18]):

$$E(i, j) = \left\lceil \frac{j - \sqrt{B}}{r_i} \right\rceil \text{ and } L(i, j) = \left\lceil \frac{j - 1 + \sqrt{B}}{r_i} + 1 \right\rceil. \quad (5.14)$$

The earliest and latest producing times for each copy (i, j) under weighted case can be calculated in the similar fashion as in the un-weighted case. The effect of the weighting factors is to shift the slopes of the corresponding unweighted problem. For the problem F_{\max}^{wa} , the results are (cf. [50]):

$$E(i, j) = \left\lceil \frac{jw_i - B}{r_i w_i} \right\rceil \text{ and } L(i, j) = \left\lceil \frac{(j - 1)w_i + B}{r_i w_i} + 1 \right\rceil. \quad (5.15)$$

Similarly for the problem F_{\max}^{ws} (cf. [18]):

$$E(i, j) = \left\lceil \frac{j - \sqrt{\frac{B}{w_i}}}{r_i} \right\rceil \text{ and } L(i, j) = \left\lceil \frac{(j - 1) + \sqrt{\frac{B}{w_i}}}{r_i} + 1 \right\rceil. \quad (5.16)$$

For a given threshold value B , early and late producing dates can be calculated for each copy of each product in a one pass procedure, and hence, can be constructed in $O(D)$ time. Thus, the decision problem (Is $Z \leq B$? for F_{\max}^a) of PRV-MDJIT is equivalent to the problem of determining whether there is a feasible schedule of D unit-time jobs on a single machine with release dates and due dates for each job. That is, the problem of solving PRV-MDJIT problem is equivalent to solve the scheduling problem, $|E(i, j); L(i, j); D; p_{(i, j)} = 1| -$, where $p_{(i, j)}$ is the processing time of copy (i, j) .

Theorem 5.2.1 [7] Consider an instance $(d_1, \dots, d_n; B)$ of the problem F_{\max}^a . A sequence $s = (s_1, s_2, \dots, s_D)$ for F_{\max}^a is B -feasible if and only if, for all $i = 1, \dots, n$ and $j = 1, \dots, d_i$, this sequence assigns copy (i, j) to the interval $[E(i, j) \dots L(i, j)]$ where $E(i, j)$ and $L(i, j)$ are given by (5.13).

Proof [7]: Necessity: Consider any feasible sequence with threshold value $B > 0$. Now, on the contrary assume that the copy (i, j) is produced at time $k < \frac{j-B}{r_i}$ on this feasible sequence. Then $x_{ik} = j$ and we must have

$$|x_{ik} - kr_i| \geq x_{ik} - kr_i > j - j + B = B,$$

which is absurd, contradicting the supposition. Therefore, copy (i, j) cannot produce before time $\left(\frac{j-B}{r_i}\right)$ and hence before time $\left\lceil \frac{j-B}{r_i} \right\rceil$.

Similarly, assume that copy (i, j) is produced at time $k > \frac{j-1+B}{r_i} + 1$. Then at time $k-1$, there holds $x_{i(k-1)} = j-1$, and we must have

$$\begin{aligned} |x_{i(k-1)} - (k-1)r_i| &\geq (k-1)r_i - x_{i(k-1)} \\ &> j-1 + B - j + 1 \\ &= B \end{aligned}$$

a contradiction. By which we conclude that copy (i, j) cannot produce after time,

$$L(i, j) = \left\lceil \frac{j-1+B}{r_i} + 1 \right\rceil$$

which establish the necessary condition.

Sufficiency: Suppose that each copy (i, j) is assigned to some time period in $[E(i, j) \dots L(i, j)] \cap [1 \dots D]$ and all copies are assigned to different time periods. Consider a fixed copy of product i and a time period k (i.e. $[k-1, k]$). Let $j \in \{1, 2, \dots, d_i\}$ be the number of copies of product i which have been produced up to (and including) time period k , i.e. $x_{ik} = j$.

Now, we must show that

$$|j - kr_i| \leq B.$$

We have

$$k \geq E(i, j)$$

So that,

$$j - kr_i \leq j - E(i, j)r_i \leq j - \left\lfloor \frac{j - B}{r_i} \right\rfloor r_i = B. \quad (5.17)$$

If $j = d_i$, then $j - kr_i = d_i - kr_i \geq 0$.

Hence, (5.17) implies that $|j - kr_i| \leq B$ as required.

Else $j < d_i$ and since the $(j+1)$ st copy of product i is produced after k , we have

$$k < L(i, j+1).$$

Then,

$$\begin{aligned} kr_i - j &\leq (L(i, j+1) - 1)r_i - j = \left\lfloor \frac{j + B}{r_i} \right\rfloor r_i - j \\ &\leq j + B - j \\ &= B. \end{aligned}$$

Therefore, $kr_i - j \leq B. \quad (5.18)$

From (5.17) and (5.18), we have

$$|j - kr_i| \leq B,$$

which completes the proof. \square

5.3 Perfect Matching Representation

The decision version of the PRV-MDJIT problem can be formulated as a perfect matching problem in the bipartite graph $G(B) = (V_1 \cup V_2, E)$ as follows (cf. [49], see also [18]): Let the vertex set $V_1 = \{1, 2, \dots, D\}$ represents the production time periods and let V_2 correspond to the set of all copies (i, j) of each product. We construct an edge between $k \in V_1$ and $(i, j) \in V_2$ if (i, j) may produce at time period k . That is, $[k, (i, j)] \in E$ if and only if $k \in [E(i, j) \dots L(i, j)]$ (see Figure 5.3 for F_{\max}^a). For any subset X of vertices, denotes by $N(X)$ the neighborhood of X , i.e., the set of all vertices adjacent to at least one vertex in X . Then, clearly the neighborhood of every vertex $(i, j) \in V_2$ is an interval $[E(i, j) \dots L(i, j)]$. Therefore, by definition the bipartite graph $G(B)$ is V_1 -convex. Now, to find a feasible sequence in the release date/due date MJIT decision problem is equivalent to finding a perfect matching in the bipartite graph $G(B)$ with the additional property that lower numbered copies of a product are always matched to earlier producing times than higher numbered copies. Such a matching will be referred to as *order preserving*.

As suggested by *Steiner and Yeomans* [49] a modified version of *Glover's* [21] $O(|E|)$ running time algorithm for finding a perfect matching in a V_1 -convex bipartite graph $G = (V_1 \cup V_2, E)$ can be used to find a perfect matching in $G(B)$, and the modified version is called the *Glover's* Earliest Due Date (EDD) algorithm (cf. [49], see also [18], see [7]). The algorithm runs through the time period $k = 1, \dots, D$, in order and assigns to k the copy (i, j) with smallest value of $L(i, j)$ along all the available copy such that $[k, (i, j)] \in E$. The algorithm can stop at $k < D$ for one of the following two reasons, as

two separate cases. Suppose $N(i, k, k')$ denotes the number of copies of product i which can be matched to some time period $t \in [k \dots k']$.

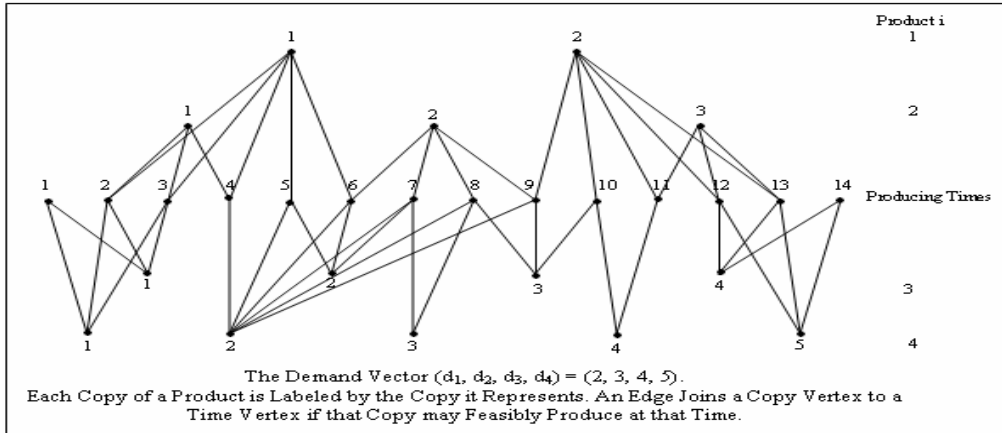


Figure 5.3: Bipartite Graph of Feasible Producing Times for Four Products Induced by a Threshold Value of $B = 0.75$ for F_{\max}^a

Stopping Case 1 Too few Products for the Available Time Periods.

The first case of stopping occurs when there are less than p copies available to sequence, intotal, in the first p time periods. That is, $\sum_{i=1}^n N(i, 1, p) < p$.

Stopping Case 2 Too many Products for the Available Time Periods.

The second case of stopping occurs when, although $\sum_{i=1}^n N(i, 1, p) \geq p$ is satisfied for $1 \leq p \leq D$, the algorithm stops at a $k < D$, because it cannot find a matching product for k , i.e., $\sum_{i=1}^n N(i, k, D) < D - k + 1$. This would be caused by more than $k-1$ products having to produce at $t \in [1 \dots k - 1]$.

Now, if we follow the rule “ For each $k \in V_1$ in descending order find the previously unmatched item (i, j) with $[k, (i, j)] \in E$ and match to k the (i, j) with the largest $L(i, j)$ value”, then showing that the stopping case 2 dose not hold for a particular problem is equivalent to show that the stopping case 1 does not hold.

The implementation of the EDD algorithm for the decision version of the MDJIT problem gives an answer in $O(D)$ time, i. e. the existence of a perfect matching in $G(B)$ for a given B can be checked in $O(D)$ time (cf. [49], see also [47]).

Lemma 5.3.1 [49] For the problem F_{\max}^a

- (a) Only the first copy of a product may produce at time $k = 1$, if $1 - r_{\max} \leq B \leq 1$.
- (b) Only the last copy of a product may produce at time $k = D$, with $1 - r_{\max} \leq B \leq 1$.

Proof: (a) Clearly copy (i, j) may produce at time period $k = 1$, if

$$\begin{aligned} j - r_i &\leq B \\ \Rightarrow j &\leq B + r_i \\ \Rightarrow j &\leq 1 + r_i < 2 \end{aligned}$$

since $0 < r_i < 1$.

Therefore, $j = 1$ is the only copy that could produce at time $t = 1$.

(b) Similarly, copy (i, j) may produce at time period $k = D$ if

$$\begin{aligned} (k-1)r_i - (j-1) &\leq B \\ \Rightarrow d_i - r_i - j + 1 &\leq B \\ \Rightarrow d_i - j &\leq r_i < 1. \end{aligned}$$

Hence, $j = d_i$, since $j \leq d_i$. □

Lemma 5.3.2 [49] *For any threshold value $B > 0$ for F_{\max}^a*

- (a) $E(i, j) < E(i, j+1)$
- (b) $L(i, j) < L(i, j+1)$

Proof: (a) From (5.7), we have

$$\begin{aligned} E(i, j) &< \frac{j-B}{r_i} + 1 \\ E(i, j+1) &\geq \frac{j+1-B}{r_i} = \frac{j}{r_i} + \frac{1}{r_i} - \frac{B}{r_i}. \end{aligned}$$

Now, since $0 < r_i < 1$, then we get

$$E(i, j) < \frac{j}{r_i} - \frac{B}{r_i} + 1 = \frac{j}{r_i} + 1 - \frac{B}{r_i} < \frac{j}{r_i} + \frac{1}{r_i} - \frac{B}{r_i} \leq E(i, j+1).$$

Therefore, $E(i, j) < E(j, j+1)$.

(b) From (5.11), we have

$$L(i, j) \leq \frac{j-1+B}{r_i} + 1 \quad \text{and} \quad L(i, j+1) > \frac{j+B}{r_i}.$$

Since $0 < r_i < 1$, then

$$\begin{aligned} L(i, j) &\leq \frac{j}{r_i} - \frac{1}{r_i} + \frac{B}{r_i} + 1 \\ &= \frac{j}{r_i} + \frac{B}{r_i} - \frac{1}{r_i} + 1 \\ &< \frac{j}{r_i} + \frac{B}{r_i} < L(i, j+1). \end{aligned}$$

Therefore, $L(i, j) < L(i, j+1)$. □

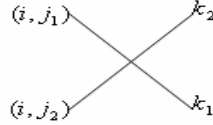
The Lemmas 5.3.1 and 5.3.2 also hold for the problem F_{\max}^s with $(1 - r_{\max})^2 \leq B \leq 1$ (cf. [18]).

These lemmas suggest that earlier copies of a product become available for sequencing before later copies and the algorithm chooses products in the order of their due dates. Therefore, *Brauner and Crema* [7] prove the following which is implicit in [49].

Proposition 5.3.1 *The PRV-MDJIT decision problem has a feasible solution if and only if the graph $G(B)$ has a perfect matching $M(B)$.*

Proof [7]: *Necessity*: Assume that, for a prespecified threshold value B , there is a feasible solution for MDJIT Problem. Clearly for such a solution each copy (i, j) of each product corresponds to exactly one time period in $[1 \dots D]$ and hence this feasible solution defines a perfect matching $M(B)$ in $G(B)$. Moreover, any feasible solution to MDJIT corresponds to an order preserving matching, i.e. a perfect matching $M(B)$ such that, when $j_1 < j_2$, copy (i, j_1) is matched to an earlier time period than copy (i, j_2) .

Sufficiency: Assume that the graph $G(B)$ has a perfect matching $M(B)$. If $M(B)$ is an order preserving one then there is nothing to prove. Otherwise, if (i, j_1) is matched to k_1 and (i, j_2) is matched to k_2 in $M(B)$, where $k_1 > k_2$ for $j_1 < j_2$ as shown below:



Then due to the convexity of $G(B)$ and Lemma 5.3.2, we may match (i, j_1) to k_2 and (i, j_2) to k_1 without destroying the feasibility. \square

Now, depending up on the *Hall's* Theorem 2.4.2, we proceed to develop the necessary and sufficient conditions for the existence of a perfect matching $M(B)$ in $G(B)$ for a given threshold value B and such a perfect matching $M(B)$ on the graph $G(B)$ can be found by applying the EDD algorithm described above.

Definition 5.3.1 Put $\mathcal{I} = \{I \mid I \text{ is an interval of } V_1\}$.

Now, for each $I \in \mathcal{I}$, let $U(I)$ be the largest subset of V_2 whose neighborhood is completely contained in I . That is, $U(I) = \{v \in V_2 \mid N(v) \subseteq I\}$.

Finally define, $\mathcal{I}_2 = \{U(I) \mid I \in \mathcal{I}\}$.

Theorem 5.3.1 [7] *For a given threshold value B for the problem F_{\max}^a , the V_1 -convex bipartite graph $G(B) = (V_1 \cup V_2, E)$ has a perfect matching if and only if for all*

$$X \in \mathcal{I}_1 \cup \mathcal{I}_2, |N(X)| \geq |X|. \quad (5.19)$$

Proof: *Necessity*: Since $|V_1| = |V_2|$, then clearly, by *Hall's* Theorem 2.4.2, condition (5.19) holds.

Sufficiency [7]: Assume that the condition (5.19) holds. Now, suppose on the contrary, there exist $X_1 \subseteq V_1$ such that $|N(X_1)| < |X_1|$. But the set X_I is a union of disjoint intervals

$X_1 = I_1 \cup \dots \cup I_p$ with $I_i = [k_i \dots k'_i]$ and $k'_i < k_{i+1} - 1$ for $i = 1, \dots, p-1$. Assume that X_1 is chosen so that p is minimal. Then, it arises the following two cases;

Case (1): Assume that for all $i, j \in \{1, \dots, p\}$ with $i \neq j$, implies $N(I_i) \cap N(I_j) = \emptyset$.

$$\text{Hence, } |N(X_1)| = \sum_{i=1}^p |N(I_i)|.$$

But each I_i is an interval and so by condition (5.19), we have

$$|N(I_i)| \geq |I_i| \text{ for all } i \in \{1, \dots, p\}.$$

$$\text{Therefore, } |N(X_1)| = \sum_{i=1}^p |N(I_i)| \geq \sum_{i=1}^p |I_i| = |X_1|,$$

which is a contradiction to the hypothesis on X_1 .

Case (2): There exist i and j with $i < j$ such that

$$N(I_i) \cap N(I_j) \neq \emptyset.$$

So, let $u \in N(I_i) \cap N(I_j)$.

Now, due to V_1 -convexity of $G(B)$, we have, $u \in N(I_i) \cap N(I_{i+1})$.

Therefore, with out loss of generality, we may assume that $j = i+1$

Put $I = [k'_i + 1 \dots k_{i+1} - 1]$.

Then clearly, $X_1 \cap I = \emptyset$.

Since $U = U(I) \in \mathcal{A}$, then condition (5.19) implies that, $|U| \leq |N(U)|$

Moreover, by definition of U , we have, $|N(U)| \leq |I|$.

Consequently $|U| \leq |I|$.

Also, we have

$$U \cap N(X_1) = \emptyset.$$

Indeed,

$$x \in U \cap N(X_1)$$

$\Rightarrow \exists y \in I$ such that $y \in X_1$

$\Rightarrow X_1 \cap I \neq \emptyset$,

which is a contradiction to $X_1 \cap I = \emptyset$. Again, due to the convexity of the graph $G(B)$,

we have, $N(I) \subseteq U \cup N(X_1)$. Therefore,

$$|N(I) \cup N(X_1)| \leq |U \cup N(X_1)| = |U| + |N(X_1)| < |I| + |X_1| = |I \cup X_1|.$$

Hence,

$$|N(X_1 \cup I)| < |X_1 \cup I|.$$

Let $Y = X_1 \cup I$. The set Y is a union of $p-1$ disjoint intervals of V_1 and satisfies $|N(Y)| < |Y|$. This is the contradiction to the minimality of p . Therefore, condition (5.19) implies the *Hall's* condition of Theorem 2.4.2, and thus (5.19) implies the existence of a perfect matching in G . \square

In the above Theorem 5.3.1 the condition on \mathcal{A} are not superfluous. That is the *Hall's* conditions on intervals of V_1 would not sufficient, by themselves, to ensure the existence of a perfect matching $M(B)$ in a V_1 -convex bipartite graph $G(B)$. For the support of this discussion *Brauner and Crama* [7] present the following counter example:

Example 5.3.1 The following convex bipartite graph $G(B)$ is associated with an instance of the MDJIT problem F_{\max}^a where $(n; d_1, \dots, d_n; B) = (3; 3, 3, 1; \frac{4}{7})$.

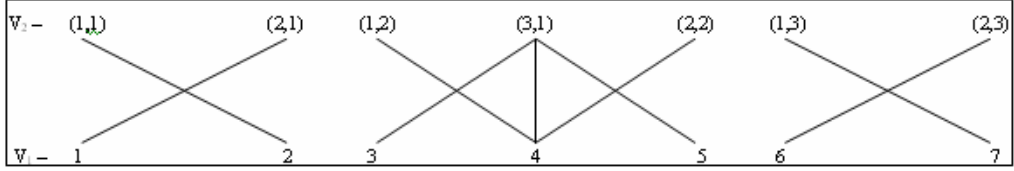


Figure 5.4: Convex Bipartite Graph for an instance $(3; 3, 3, 1; \frac{4}{7})$ of F_{\max}^a

Clearly, this convex bipartite graph $G(\frac{4}{7})$ satisfies the conditions, for all $X \in \mathcal{F}_1$, $|N(X)| \geq |X|$, but unfortunately, does not have a perfect matching.

Now, to strengthen the condition (5.19) by restricting them to those sets of \mathcal{F}_2 whose neighborhood is an interval. For that we go with a

Definition 5.3.2 $\mathcal{F}'_2 = \{U \in \mathcal{F}_2 \mid N(U) \in \mathcal{F}_1\}$

Consequently, Brauner and Crema [7] give

Proposition 5.3.2 [7] Conditions (5.20) and (5.21) are equivalent:

$$\forall X \in \mathcal{F}_2, |N(X)| \geq |X| \quad (5.20)$$

$$\forall X \in \mathcal{F}'_2, |N(X)| \geq |X| \quad (5.21)$$

Proof: (5.20) \Rightarrow (5.21): $\mathcal{F}'_2 \subseteq \mathcal{F}_2$ and hence (5.20) \Rightarrow (5.21).

Now, (5.21) \Rightarrow (5.20):

Assume that (5.21) holds and consider $U \in \mathcal{F}_2$. If $N(U)$ is an interval, then $U \in \mathcal{F}'_2$ and immediately (5.21) \Rightarrow (5.20). So, suppose that $N(U)$ is a union of disjoint intervals:

$$N(U) = [k_1 \dots k'_1] \cup \dots \cup [k_p \dots k'_p].$$

with $k'_i < k_{i+1} - 1$ for $i = 1, \dots, p - 1$.

Define

$$Y_i = \{v \in V_2 \mid N(v) \subseteq [k_1 \dots k_2]\}.$$

Then, by construction

$$\bigcup_{i=1}^p Y_i \subseteq U.$$

And actually

$$\bigcup_{i=1}^p Y_i = U$$

For that let $k \in U$ and $k \notin \bigcup_{i=1}^p Y_i$. This means that k has neighbors in at least two distinct intervals $[k_i \dots k_i']$ and $[k_j \dots k_j']$, with $i < j$. Since the graph is V_1 -convex, this would imply that all vertices between k_i' and k_j are in $N(U)$ as shown in the Figure 5.5. This is absurd. Therefore, we must have $\bigcup_{i=1}^p Y_i = U$.

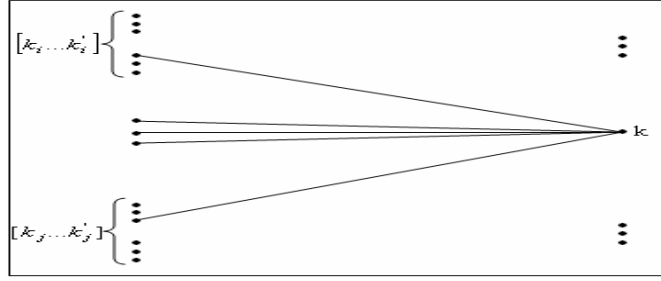


Figure 5.5: Two Distinct Intervals $[k_i \dots k_i']$ and $[k_j \dots k_j']$ with $i < j$ Containing the Neighbors of Vertex k

Since every element of $[k_i \dots k_i']$ is the neighbor of some element of U , there follows immediately that, $N(Y_i) = [k_i \dots k_i']$ and hence $Y_i \in \mathcal{G}'_2$, for $i = 1, \dots, p$.

Therefore, $|N(U)| = \sum_{i=1}^p |N(Y_i)| \geq \sum_{i=1}^p |Y_i| = |U|$, i. e. $|N(U)| \geq |U|$.

Thus (5.21) \Rightarrow (5.20). This completes the proof. \square

From Proposition 5.3.2 it can be concluded that, in the case of convex graphs, *Hall's* conditions need only to be applied to those sets X such that X is either an interval in V_I or the neighborhood of an interval in V_I .

Theorem 5.3.2 [7] *For a prespecified threshold value $B > 0$, the MDJIT problem F_{\max}^a has a feasible solution if and only if for all k_1, k_2 in $[1 \dots D]$ with $k_1 \leq k_2$, the following inequalities are both valid:*

$$\sum_{i=1}^n \max(o, \lfloor k_2 r_i + B \rfloor - \lceil (k_1 - 1) r_i - B \rceil) \geq k_2 - k_1 + 1; \text{ and} \quad (5.22)$$

$$\sum_{i=1}^n \max(o, \lceil k_2 r_i - B \rceil - \lfloor (k_1 - 1) r_i + B \rfloor) \leq k_2 - k_1 + 1 \quad (5.23)$$

Proof [7]: *Necessity:* Suppose the MDJIT problem F_{\max}^a has a feasible solution with threshold value $B > 0$. Now, since F_{\max}^a has a feasible solution then $G(B)$ has a perfect matching $M(B)$. This is equivalent to

$$\forall X \in \mathcal{F}_1 \cup \mathcal{G}'_2, |N(X)| \geq |X|. \quad (5.24)$$

Now, let $X = [k_1 \dots k_2]$ be an interval of V_I . Then copy (i, j) is in the neighborhood of X ,

$$\begin{aligned}
&\Leftrightarrow [k_1 \dots k_2] \subseteq [E(i, j) \dots L(i, j)] \\
&\Leftrightarrow E(i, j) \leq k_1 \text{ and } L(i, j) \geq k_2 \\
&\Leftrightarrow E(i, j) \leq k_1 \leq k_2 \leq L(i, j) \\
&\Leftrightarrow E(i, j) \leq k_2 \text{ and } L(i, j) \geq k_1 \\
&\Leftrightarrow \left\lfloor \frac{j-B}{r_i} \right\rfloor \leq k_2 \text{ and } \left\lceil \frac{j-1+B}{r_i} + 1 \right\rceil \geq k_1 \\
&\Leftrightarrow \frac{j-B}{r_i} \leq k_2 \text{ and } \frac{j-1+B}{r_i} + 1 \geq k_1 \\
&\Leftrightarrow (k_1 - 1)r_i + 1 - B \leq j \leq k_2 r_i + B. \tag{5.25}
\end{aligned}$$

For a given product $i \in \{1, \dots, n\}$, the number of copies j which satisfy the inequality (5.25) is (i.e. the total number of natural numbers between $(k_1 - r_i)r_i + 1 - B$ and $k_2 r_i + B$ including both if they are natural).

$$\max(0, \lfloor k_2 r_i + B \rfloor - \lceil (k_1 - 1)r_i + 1 - B \rceil + 1) = \max(0, \lfloor k_2 r_i + B \rfloor - \lceil (k_1 - 1)r_i - B \rceil).$$

Therefore, for $X = [k_1 \dots k_2] \in \mathcal{F}_1$, $|N(X)| \geq |X|$, if and only if

$$\sum_{i=1}^n \max(0, \lfloor k_2 r_i + B \rfloor - \lceil (k_1 - 1)r_i - B \rceil) \geq k_2 - k_1 + 1,$$

which is (5.22).

Now, for given k_1, k_2 in V_1 , with $k_1 \leq k_2$, and given $(i, j) \in V_2$, we have

$$\begin{aligned}
&[E(i, j) \dots L(i, j)] \subseteq [k_1 \dots k_2] \\
&\Leftrightarrow E(i, j) \geq k_1 \text{ and } L(i, j) \leq k_2 \\
&\Leftrightarrow E(i, j) > k_1 - 1 \text{ and } L(i, j) < k_2 + 1 \\
&\Leftrightarrow \frac{j-B}{r_i} > k_1 - 1 \text{ and } \frac{j-1+B}{r_i} + 1 < k_2 + 1 \\
&\Leftrightarrow (k_1 - 1)r_i + B < j < k_2 r_i + 1 - B \\
&\Leftrightarrow \lfloor (k_1 - 1)r_i + B + 1 \rfloor \leq j \leq \lceil k_2 r_i - B \rceil. \tag{5.26}
\end{aligned}$$

For a given product i , the number of j which satisfy the inequality (5.26) is

$$\max(0, \lceil k_2 r_i - B \rceil - \lfloor (k_1 - 1)r_i + B \rfloor).$$

Thus the cardinality of

$$U([k_1 \dots k_2]) = \{(i, j) \in V_2 \mid [E(i, j) \dots L(i, j)] \subseteq [k_1 \dots k_2]\}$$

can be computed as

$$|U([k_1 \dots k_2])| = \sum_{i=1}^n \max(0, \lceil k_2 r_i - B \rceil - \lfloor (k_1 - 1)r_i + B \rfloor). \tag{5.27}$$

Now, assume that the inequalities (5.23) hold for all values of $k_1 \leq k_2$ and consider $X \in \mathcal{F}'_2$. By definition, $N(X)$ is an interval of V_1 , say $N(X) = [k_1 \dots k_2]$. Hence, $X \subseteq U([k_1 \dots k_2])$. Thus, from (5.23) and (5.27), we have

$$|X| \leq |U([k_1 \dots k_2])| \leq k_2 - k_1 + 1 = |N(X)|,$$

which is required by (5.24).

Sufficiency: Suppose both conditions (5.22) and (5.23) hold. Now, we will show that the corresponding sequence is feasible for the MDJIT problem F_{\max}^a .

Condition (5.22) implies that for any $X = [k_1 \dots k_2] \in \mathcal{F}_1$, with $k_1 \leq k_2$,

$$|X| \leq |N(X)|. \quad (5.28)$$

Now, consider $X \in \mathcal{F}'_2$. By definition $N(X)$ is an interval of V_1 , say $N(X) = [k_1 \dots k_2]$.

So that, $X \subseteq U([k_1 \dots k_2])$.

Then $|X| \leq |U([k_1 \dots k_2])|$,

or $|X| \leq k_2 - k_1 + 1$ (By 5.23 and 5.27)

or $|X| \leq |N(X)|$.

Therefore, $|X| \leq |N(X)|$ for all $X \in \mathcal{F}'_2$.

Hence, for all $X \in \mathcal{F}_1 \cup \mathcal{F}'_2$, $|X| \leq |N(X)|$.

Therefore, by Theorem 5.3.1 and Proposition 5.3.2, the graph $G(B)$ has a perfect matching and consequently due to Proposition 5.3.1, the MDJIT problem F_{\max}^a has a feasible solution. \square

A similar result for the MDJIT problem F_{\max}^s is obtained by *Dhamala et al.* [18] as

Theorem 5.3.3 [18] *For a given threshold value $B > 0$, the graph $G(B) = (V_1 \cup V_2, E)$ corresponding to the problem F_{\max}^s has a perfect matching if and only if for all $k_1, k_2 \in V_1$, $k_1 \leq k_2$ and $[E(i, j) \dots L(i, j)] \cap [k_1 \dots k_2] \neq \emptyset$, it holds*

$$\begin{aligned} \sum_{i=1}^n \left(\lfloor k_2 r_i + \sqrt{B} \rfloor - \lceil (k_1 - 1) r_i - \sqrt{B} \rceil \right) &\geq k_2 - k_1 + 1; \quad \text{and} \\ \sum_{i=1}^n \left(\lfloor k_2 r_i - \sqrt{B} \rfloor - \lceil (k_1 - 1) r_i + \sqrt{B} \rceil \right) &\leq k_2 - k_1 + 1. \end{aligned} \quad (5.29)$$

Proof [18]: Let $X = [k_1 \dots k_2] \subseteq V_1$. Then $(i, j) \in N(X)$

$$\Leftrightarrow [E(i, j) \dots L(i, j)] \cap X \neq \emptyset$$

$$\Leftrightarrow E(i, j) \leq k_2 \quad \text{and} \quad L(i, j) \geq k_1$$

$$\Leftrightarrow \frac{j - \sqrt{B}}{r_i} \leq k_2 \quad \text{and} \quad \frac{j - 1 + \sqrt{B}}{r_i} + 1 \geq k_1$$

$$\Leftrightarrow \lceil (k_1 - 1) r_i + 1 - \sqrt{B} \rceil \leq j \leq \lfloor k_2 r_i + \sqrt{B} \rfloor$$

Therefore, for $X \subseteq V_1$, $|N(X)| \geq |X|$ if and only if

$$\sum_{i=1}^n \left(\lfloor k_2 r_i + \sqrt{B} \rfloor - \lceil (k_1 - 1) r_i - \sqrt{B} \rceil \right) \geq k_2 - k_1 + 1.$$

If X is the neighborhood of an interval $[k_1 \dots k_2]$ in V_1 , i.e., $N(X) = [k_1 \dots k_2] \subseteq V_1$. Then $(i, j) \in X \subseteq V_2$

$$\begin{aligned}
&\Leftrightarrow [E(i, j) \dots L(i, j)] \subseteq [k_1 \dots k_2] \subseteq V_1 \\
&\Leftrightarrow k_1 \leq E(i, j) \text{ and } L(i, j) \leq k_2 \\
&\Leftrightarrow k_1 \leq \frac{j - \sqrt{B}}{r_i} \text{ and } \frac{j - 1 + \sqrt{B}}{r_i} + 1 \leq k_2 \\
&\Leftrightarrow (k_1 - 1)r_i + \sqrt{B} < j < k_2 r_i + 1 - \sqrt{B} \\
&\Leftrightarrow \lfloor (k_1 - 1)r_i + 1 + \sqrt{B} \rfloor \leq j \leq \lceil k_2 r_i - \sqrt{B} \rceil.
\end{aligned}$$

Thus, for X with $N(X) = [k_1 \dots k_2] \subseteq V_1$, $|N(X)| \geq |X|$ if and only if

$$\sum_{i=1}^n \left(\lceil k_2 r_i - \sqrt{B} \rceil - \lfloor (k_1 - 1)r_i + \sqrt{B} \rfloor \right) \leq k_2 - k_1 + 1. \quad \square$$

A slight modification of Theorem 5.3.2 to state a simpler form needs the following proposition.

Proposition 5.3.3 [7] *When a prespecified threshold value B is strictly less than 1 for F_{\max}^a , then the following statements (a)-(d) are equivalent.*

(a) for all $k_1, k_2 \in [1 \dots D]$ with $k_1 \leq k_2$,

$$\sum_{i=1}^n \left(\lfloor k_2 r_i + B \rfloor - \lceil (k_1 - 1)r_i - B \rceil \right) \geq k_2 - k_1 + 1, \quad (5.30)$$

(b) for all k_1, k_2 in $[1 \dots D]$ with $k_1 \leq k_2$,

$$\sum_{i=1}^n \max(0, \lfloor k_2 r_i + B \rfloor - \lceil (k_1 - 1)r_i - B \rceil) \geq k_2 - k_1 + 1, \quad (5.31)$$

(c) for all k in $[1 \dots D]$,

$$\sum_{i=1}^n \lfloor k r_i + B \rfloor \geq k, \quad (5.32)$$

(d) for all k in $[1 \dots D]$,

$$\sum_{i=1}^n \lfloor k r_i + B \rfloor \geq k, \quad (5.33)$$

$$\sum_{i=1}^n \lceil k r_i - B \rceil \leq k. \quad (5.34)$$

Proof:

1. (a) \Rightarrow (b): Here the right-side term of (5.30) is always positive integer and as a result all terms of the sum in left-hand-side of (5.30) may not be non-positive. Now, replace the non-positive term in the sum by 0 and positive terms remain same. Then, consequently, we get (b). Therefore, (a) \Rightarrow (b).

2. (b) \Rightarrow (c): Suppose (b) holds. Now, set $k_1 = 1$ in (5.31) to get

$$\sum_{i=1}^n \max(0, \lfloor k_2 r_i + B \rfloor - \lceil -B \rceil) \geq k_2. \quad (5.35)$$

But $k > 0$, $r_i > 0$ and $0 \leq B < 1$, implies that

$$\lfloor k_2 r_i + B \rfloor \geq 0 \text{ and } \lceil -B \rceil = 0.$$

Thus, (5.35) becomes

$$\sum_{i=1}^n \lfloor k_2 r_i + B \rfloor \geq k_2 .$$

Now, replace the dummy variable k_2 by the general variable k to get

$$\sum_{i=1}^n \lfloor k r_i + B \rfloor \geq k , \text{ for all } k \text{ in } [1 \dots D].$$

Therefore, (b) \Rightarrow (c).

3. (c) \Rightarrow (d): Here we need to show that (5.33) holds for all k in $[1 \dots D]$ if and only if (5.34) holds for all k in $[1 \dots D]$.

For that, (5.33) \Rightarrow (5.34).

Write $k' = D - k$ in place of k in (5.33), to get

$$\begin{aligned} & \sum_{i=1}^n \lfloor k' r_i + B \rfloor \geq k' \\ \Rightarrow & \sum_{i=1}^n \lfloor D r_i - k r_i + B \rfloor \geq D - k \\ \Rightarrow & \sum_{i=1}^n \{d_i + \lfloor B - k r_i \rfloor\} \geq D - k \\ \Rightarrow & D + \sum_{i=1}^n \lfloor B - k r_i \rfloor \geq D - k \\ \Rightarrow & \sum_{i=1}^n \lfloor B - k r_i \rfloor \geq -k \\ \Rightarrow & - \sum_{i=1}^n \lfloor B - k r_i \rfloor \leq k \\ \Rightarrow & \sum_{i=1}^n \lceil k r_i - B \rceil \leq k . \end{aligned}$$

Since $k = D$, then $k' = 0$.

So that,

$$\begin{aligned} & \sum_{i=1}^n \lfloor k' r_i + B \rfloor \geq k' \\ \Rightarrow & \sum_{i=1}^n \lfloor B \rfloor \geq 0 \\ \Rightarrow & 0 \geq 0, \text{ which is true.} \end{aligned}$$

Hence, (5.33) \Rightarrow (5.34).

Now, it remains to show (5.34) \Rightarrow (5.33)

For that, write $k' = D - k$ in place of k in relation (5.34), to get

$$\sum_{i=1}^n \lceil k' r_i - B \rceil \leq k'$$

i.e.

$$\sum_{i=1}^n \lceil D r_i - k r_i - B \rceil \leq D - k$$

or
$$\sum_{i=1}^n [d_i - kr_i - B] \leq D - k$$

or
$$\sum_{i=1}^n \{d_i + [-(kr_i + B)]\} \leq D - k$$

or
$$D + \sum_{i=1}^n [-(kr_i + B)] \leq D - k$$

or
$$\sum_{i=1}^n [-(kr_i + B)] \leq -k.$$

But, we know
$$[-x] = -[x].$$

Therefore,
$$\sum_{i=1}^n (-[kr_i + B]) \leq -k$$

$$\Rightarrow -\sum_{i=1}^n [kr_i + B] \leq -k$$

$$\Rightarrow \sum_{i=1}^n [kr_i + B] \geq k.$$

Since $k = D$, then $k' = 0$.

So that,
$$\sum_{i=1}^n [k'r_i - B] \leq k'$$

$$\Rightarrow \sum_{i=1}^n [-B] \leq 0$$

$$\Rightarrow 0 \leq 0, \text{ which is true.}$$

Hence, (5.34) \Rightarrow (5.33). Therefore, (c) \Rightarrow (d).

Finally it remains to show that

4. (d) \Rightarrow (a)

In (5.33) replace k by k_2 , to get

$$\sum_{i=1}^n [k_2 r_i + B] \geq k_2. \quad (5.36)$$

Again, in (5.34) replace k by $k_1 - 1$, to get

$$\begin{aligned} & \sum_{i=1}^n [(k_1 - 1)r_i - B] \leq k_1 - 1 \\ \Rightarrow & -\sum_{i=1}^n [(k_1 - 1)r_i - B] \geq 1 - k_1. \end{aligned} \quad (5.37)$$

Add (5.36) and (5.37), we have

$$\sum_{i=1}^n [k_2 r_i + B] - \sum_{i=1}^n [(k_1 - 1)r_i - B] \geq k_2 - k_1 + 1$$

i.e.
$$\sum_{i=1}^n ([k_2 r_i + B] - [(k_1 - 1)r_i - B]) \geq k_2 - k_1 + 1,$$

which is (5.30). Therefore, (d) \Rightarrow (a).

Thus, in the conclusion we have (a) \Rightarrow (b) \Rightarrow (c) \Rightarrow (d) \Rightarrow (a) and hence proof is completed. \square

Corollary 5.3.1 [7] *For the given threshold value $B < 1$, the MDJIT problem F_{\max}^a has a feasible solution if and only if, for all k_1, k_2 in $[1 \dots D]$ with $k_1 \leq k_2$, the following inequalities are both valid:*

$$\sum_{i=1}^n \lfloor k_1 r_i + B \rfloor \geq k_1 \quad (5.38)$$

$$\sum_{i=1}^n \max(0, \lceil k_2 r_i - B \rceil - \lfloor (k_1 - 1) r_i + B \rfloor) \leq k_2 - k_1 + 1. \quad (5.39)$$

Proof: This corollary can be proved with the direct application of Theorem 5.3.2 and Proposition 5.3.3. \square

The condition (5.39) is not redundant one. For that *Brauner and Crama* [7] demonstrate a counter example with $(n; d_1, \dots, d_n; B) = (3; 3, 3, 1; \frac{4}{7})$ of MDJIT problem F_{\max}^a . Clearly, the conditions $\sum_{i=1}^n \lfloor k r_i + B \rfloor \geq k$ for all k in $[1 \dots D]$. But the instance does not have a feasible solution for the given $B (= \frac{4}{7})$ since the graph in Figure 5.4 has no perfect matching. That is, the inequality (5.39) fails for $k_1 = k_2 = 4$.

Now before closing this section, we summarize the main result of this section as

Proposition 5.3.4 [18, 35] *The optimal objective values of the MDJIT problems F_{\max}^a and F_{\max}^s are respectively the smallest B_a and B_s such that the corresponding graphs $G(B_a)$ and $G(B_s)$ have perfect matchings $M(B_a)$ and $M(B_s)$.* \square

5.4 Bounds on the Threshold Value

In this section the bounds on the optimal PRV-MDJIT problems are analyzed.

5.4.1 Lower Bound

From Section 5.3, if the first copy of product i is sequenced to be produced at $k = 1$, then by definition

$$f^i(1) = \max_{1 \leq j \leq d_i} \{g_j^i(1)\} = 1 - r_i.$$

By which we conclude that, the threshold value B for F_{\max}^a can be feasible if

$$\begin{aligned} & \min_i f^i(1) \leq B \\ \Rightarrow & \min_i \{1 - r_i\} \leq B \\ \Rightarrow & 1 - \max_i r_i \leq B \\ \Rightarrow & 1 - r_{\max} \leq B. \end{aligned}$$

Therefore, for being a feasible solution to PRV-MDJIT problem F_{\max}^a , the threshold value B must be bounded from below by the quantity $1 - r_{\max}$ (cf. [49]). For the problem F_{\max}^s , a copy of some product i must be assigned to the time unit $k = 1$. Then it must hold

$$\begin{aligned} \min_i (1 - r_i)^2 &\leq B \\ \Rightarrow (1 - r_{\max})^2 &\leq B \end{aligned}$$

for a given threshold value B to be feasible sequence. Thus the lower bound for the problem F_{\max}^s is $(1 - r_{\max})^2$ (cf. [18]). By the similar argument, we can calculate the lower bounds for the optimal value of the problems F_{\max}^{wa} and F_{\max}^{ws} as $w_{\min}(1 - r_{\max})$ (cf. [50]) and $w_{\min}(1 - r_{\max})^2$ (cf. [18]) respectively, where $w_{\min} = \min_i w_i$.

Example 5.4.1 *The lower bound of the threshold value B for the MDJIT problem F_{\max}^a is tight.*

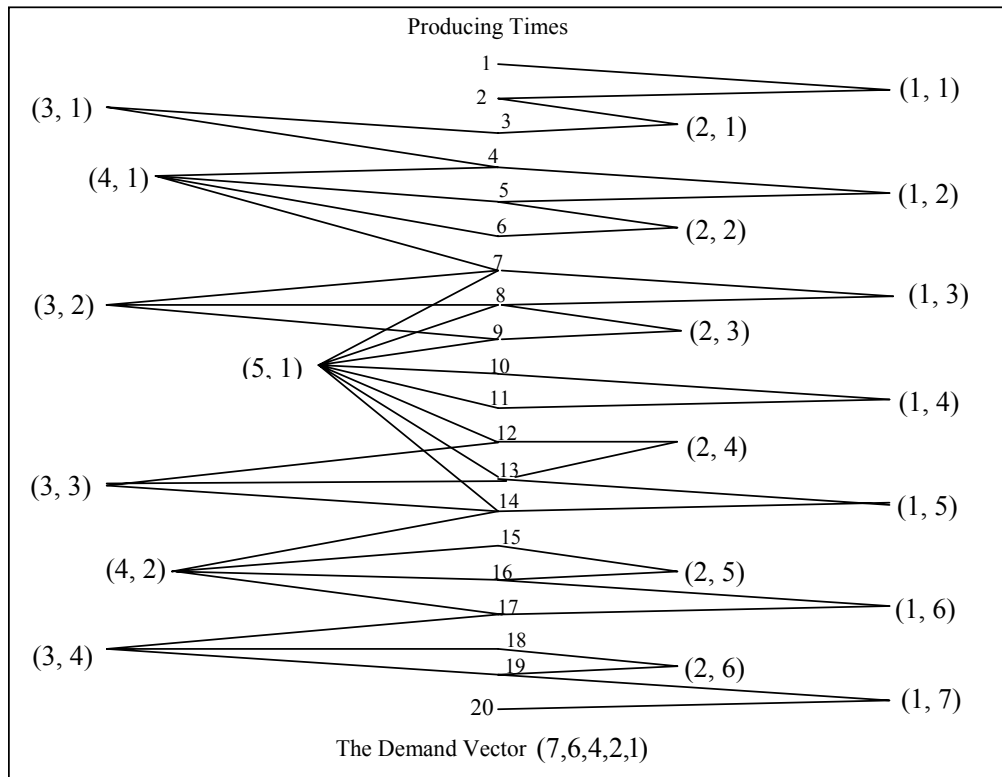


Figure 5.6: Bipartite Graph of Feasible Producing Times for Five Products with Threshold Value $B = 13/20$ for F_{\max}^a

For that consider an instance

$$(n; d_1, \dots, d_n; B) = (5; 7, 6, 4, 2, 1; \frac{13}{20})$$

of the MDJIT problem F_{\max}^a where $B = 1-r_{\max}$. The corresponding V_1 -convex bipartite graph $G(\frac{13}{20})$ is given by Figure 5.6. Now, by applying the EDD algorithm, we easily

obtain the following perfect order preserving perfect matching $M(\frac{13}{20})$. That is

$$M(\frac{13}{20}) = \{[1,(1,1)], [2,(2,1)], [3,(3,1)], [4,(1,2)], [5,(2,2)], [6,(4,1)], [7,(1,3)], [8,(2,3)], [9,(3,2)], [10,(1,4)], [11,(5,1)], [12,(2,4)], [13,(1,5)], [14,(3,3)], [15,(2,5)], [16,(1,6)], [17,(4,2)], [18,(2,6)], [19,(3,4)], [20,(1,7)]\}.$$

This yields the optimal production sequence $s = (1,2,3,1,2,4,1,2,3,1,5,2,1,3,2,1,4,2,3,1)$ for the given instance. \square

Proposition 5.4.1 For $d_1 \leq d_2 \leq \dots \leq d_n$. The optimal value B^* of the MDJIT problem

(a) [7] F_{\max}^a satisfies

$$r_i \leq B^*, \quad \text{for } i = 1, \dots, n-1, \quad (5.40)$$

$$1 - r_i \leq 2B^*, \quad \text{for } i = 1, \dots, n-1, \quad (5.41)$$

$$r_n \leq 2B^*, \quad (5.42)$$

$$1 - r_n \leq B^*, \quad (5.43)$$

$$\frac{n-1}{2n-1} \leq B^*. \quad (5.44)$$

(b) F_{\max}^s satisfies

$$r_i \leq \sqrt{B^*}, \quad \text{for } i = 1, \dots, n-1,$$

$$1 - r_i \leq 2\sqrt{B^*}, \quad \text{for } i = 1, \dots, n-1,$$

$$r_n \leq 2\sqrt{B^*},$$

$$1 - r_n \leq \sqrt{B^*},$$

$$\frac{n-1}{2n-1} \leq \sqrt{B^*}.$$

Proof:

(a) [7] In any feasible sequence, some copy (i, j) must be produced in time period $k = 1$ (i.e. in period $[0, 1]$). But $r_n = \max\{r_1, \dots, r_n\}$.

Hence, to minimize $\max_{1 \leq i \leq n} |x_{i1} - r_i|$ we have to sequence product n in the first time period.

Therefore, we must have, $|1 - r_n| \leq B^* \Rightarrow 1 - r_n \leq B^*$, which is (5.43). Now, since copy $(n, 1)$ is sequenced in time period $k = 1$, then clearly, $|0 - r_i| = r_i \leq B^*, \forall i = 1, \dots, n-1$, which is condition (5.40).

Again, for a feasible solution of the MDJIT problem F_{\max}^a , it holds $E(i, j) \leq L(i, j)$ (cf. [31] for proof) for all copies (i, j) . Hence,

$$0 \leq L(i, j) - E(i, j) \leq \frac{j-1+B^*}{r_i} + 1 - \frac{j-B^*}{r_i} = \frac{2B^*+r_i-1}{r_i}$$

and the condition (5.41) follows. Moreover, $2B^* \geq 1 - r_1 = \sum_{i=2}^n r_i \geq r_n$, which is condition (5.42). Now, for the bound (5.44), we simply add the inequalities (5.42) for $i = 1, \dots, n-1$ to inequality (5.43) to get the required result

$$\frac{n-1}{2n-1} \leq B^*.$$

(b) It follows immediately from (a). \square

Theorem 5.4.1 Define $\Delta_i = \frac{D}{\gcd(d_i, D)}$. The optimal value B^* of the MDJIT problem

(a) [7] F_{\max}^a satisfies

$$B^* \geq \frac{1}{\Delta_i} \left\lfloor \frac{\Delta_i}{2} \right\rfloor \quad \forall i = 1, \dots, n. \quad (5.45)$$

(b) F_{\max}^s satisfies

$$\sqrt{B^*} \geq \frac{1}{\Delta_i} \left\lfloor \frac{\Delta_i}{2} \right\rfloor \quad \forall i = 1, \dots, n. \quad (5.46)$$

Proof:

(a) [7] The ideal production rate for any product $i = 1, \dots, n$ is $r_i = \frac{d_i}{D} = \frac{\delta_i}{\Delta_i}$ with $\gcd(\Delta_i, \delta_i) = 1$. Clearly, any feasible solution $(x_{ik})_{n \times D}$ of the MDJIT problem F_{\max}^a satisfies the inequality

$$|x_{ik} - kr_i| \geq |[kr_i] - kr_i|, \text{ for all } k \in [1 \dots D].$$

Now, it is sufficient to show that there exists at least one k in $[1 \dots D]$ such that

$$|[kr_i] - kr_i| = \frac{1}{\Delta_i} \left\lfloor \frac{\Delta_i}{2} \right\rfloor.$$

Case (I): Δ_i is even.

$$\text{Put } k = \frac{\Delta_i}{2},$$

then, $|kr_i - [kr_i]| = \left| \frac{\Delta_i}{2} \frac{\delta_i}{\Delta_i} - \left[\frac{\Delta_i}{2} \frac{\delta_i}{\Delta_i} \right] \right| = \left| \frac{\delta_i}{2} - \left[\frac{\delta_i}{2} \right] \right|.$

But $\gcd(\Delta_i, \delta_i) = 1$ and Δ_i -even, so the optimal value of the objective function is at least

$$\frac{1}{\Delta_i} \left\lfloor \frac{\Delta_i}{2} \right\rfloor = 0.5.$$

Case (II): Δ_i is odd.

Since Δ_i is odd, then we have $\left\lfloor \frac{\Delta_i}{2} \right\rfloor = \frac{\Delta_i}{2} - \frac{1}{2}.$

Again, we show that there exists a k such that

$$|[kr_i] - kr_i| = \frac{1}{\Delta_i} \left\lfloor \frac{\Delta_i}{2} \right\rfloor = \frac{\Delta_i - 1}{2\Delta_i}.$$

As the integers δ_i and Δ_i are relatively prime, then there exist two integers u and v such that $u\delta_i + v\Delta_i = 1$ (see [10]).

Which implies that
$$-|u|\Delta_i + |v|\delta_i = \pm 1 \tag{5.47}$$

Now, multiply on both sides of equality (5.47) by a constant number $\frac{\Delta_i - 1}{2\Delta_i}$ and set

$k = |v|\frac{\Delta_i - 1}{2}$ to obtain

$$k \frac{\delta_i}{\Delta_i} = |u| \frac{\Delta_i - 1}{2} \pm \frac{\Delta_i - 1}{2\Delta_i}. \tag{5.48}$$

Now, for $\frac{\Delta_i - 1}{2\Delta_i} < 0.5$ and Δ_i is odd, we have from (5.48), $\left\lceil k \left(\frac{\delta_i}{\Delta_i} \right) \right\rceil = |u| \frac{\Delta_i - 1}{2}$ and

hence the desired conclusion is obtained as $|\lceil k \left(\frac{\delta_i}{\Delta_i} \right) \rceil - k \left(\frac{\delta_i}{\Delta_i} \right)| = \frac{\Delta_i - 1}{2\Delta_i}$

i.e.,
$$|[kr_i - kr_i]| = \frac{1}{\Delta_i} \left\lfloor \frac{\Delta_i}{2} \right\rfloor.$$

(b) It directly follows from the proof of (a). □

5.4.2 Upper Bound

Interestingly, *Steiner and Yeomans* [49] prove that the optimal value of the MDJIT problem F_{\max}^a is always less or equal to 1. This means that for any number n of products and any set of demand values d_1, \dots, d_n , there always exists a sequence in which at any time $k \in [1 \dots D]$ no product is ahead or behind the ideal cumulative production kr_i for product i by more than $B = 1$. But due to *Brauner and Crama* [7] and *Tijdeman* [53], *Kubiak* [28] gives a slightly stronger version of this result for PRV-MDJIT problem F_{\max}^a as

Theorem 5.4.2 *For any instance d_1, \dots, d_n ($n > 1$) of the MDJIT problem F_{\max}^a , the optimal value*

$$B^* \leq 1 - \max \left\{ \frac{1}{D}, \frac{1}{2n - 2} \right\}.$$

Proof: It is sufficient to prove the following for an optimal solution B^* of the MDJIT problem F_{\max}^a

- (a) $B^* \leq 1 - \frac{1}{D}$
- (b) $B^* \leq 1 - \frac{1}{2(n - 1)}$

(a) [7] Due to Corollary 5.3.1, it is sufficient to show that the two inequalities (5.38) and (5.39) of the same Corollary hold for $B = 1 - \frac{1}{D} < 1$.

For that, let $k \geq 0$ be any integer, and if kr_i is also an integer, then $\lfloor kr_i + B \rfloor = kr_i$.

Otherwise $\{kr_i\} \geq \frac{1}{D}$, where $\{kr_i\}$ denotes the fractional part of kr_i . This implies

$$\{kr_i\} + B \geq \frac{1}{D} + 1 - \frac{1}{D} = 1,$$

and hence,

$$\lfloor kr_i + B \rfloor = \lfloor kr_i \rfloor + 1 > kr_i.$$

Therefore, for any integer $k \geq 0$, we have

$$\lfloor kr_i + B \rfloor \geq kr_i \quad (5.49)$$

Now, by summing (5.49) over $i = 1, \dots, n$, we have

$$\sum_{i=1}^n \lfloor kr_i + B \rfloor \geq k$$

for any integer $k \geq 0$.

And in particular $\sum_{i=1}^n \lfloor k_1 r_i + B \rfloor \geq k_1$, for any k_1 in $[1 \dots D]$. This proves inequality (5.38).

Now, to establish the inequality (5.39), fix $k_1, k_2 \in [1 \dots D]$, with $k_1 \leq k_2$, and consider the set $J \subseteq \{1, \dots, n\}$, defined by $i \in J \Leftrightarrow \lceil k_2 r_i - B \rceil - \lfloor (k_1 - 1)r_i + B \rfloor \geq 0$. Then by substituting $D - k_2$ in place of k in (5.49), we get

$$\begin{aligned} & \lfloor (D - k_2)r_i + B \rfloor \geq (D - k_2)r_i \\ \Rightarrow & \lfloor d_i - k_2 r_i + B \rfloor \geq d_i - k_2 r_i \\ \Rightarrow & d_i + \lfloor -(k_2 r_i - B) \rfloor \geq d_i - k_2 r_i \\ \Rightarrow & \lfloor -(k_2 r_i - B) \rfloor \geq -k_2 r_i \\ \Rightarrow & -\lfloor -(k_2 r_i - B) \rfloor \leq k_2 r_i. \end{aligned}$$

But

$$-\lfloor x \rfloor = \lceil -x \rceil.$$

Therefore,

$$\lceil k_2 r_i - B \rceil \leq k_2 r_i, \text{ for } i = 1, 2, 3, \dots, n.$$

Also, we have

$$\sum_{i \in J} r_i \leq 1$$

Hence, we derive successively

$$\begin{aligned} \sum_{i=1}^n \max(0, \lceil k_2 r_i - B \rceil - \lfloor (k_1 - 1)r_i + B \rfloor) &= \sum_{i \in J} (\lceil k_2 r_i - B \rceil - \lfloor (k_1 - 1)r_i + B \rfloor) \\ &\leq \sum_{i \in J} (k_2 - k_1 + 1)r_i \\ &= (k_2 - k_1 + 1) \sum_{i \in J} r_i \leq k_2 - k_1 + 1, \end{aligned}$$

which establish the required inequality (5.39), and consequently (a) is proved.

(b) The proof given by *Kubiak* [28] with the direct application of Theorem 2.2.1 is given here. For that define the double sequence λ_{ij} ($1 \leq i \leq n; j \in \mathbb{N}$) of nonnegative number

with $\lambda_{ij} = r_i = \frac{d_i}{D}$ for all $j \in \mathcal{N}$. Then clearly for all $j \in \mathcal{N}$, $\sum_{i=1}^n \lambda_{ij} = 1$. Therefore, the Theorem 2.2.1 guarantees that the existence of an infinite sequence s in the alphabet $\{1, \dots, n\}$ such that

$$\max_{i,k} \left| \sum_{j=1}^k \lambda_{ij} - x_{ik} \right| \leq 1 - \frac{1}{2n-2}.$$

That is, $\max_{i,k} |kr_i - x_{ik}| \leq 1 - \frac{1}{2n-2}$,

where x_{ik} gives the same meaning as in the Theorem 2.2.1.

We now require the number of copies of each product in first D -prefix of s . Consider the first D -prefix of s and assume that there is a product i ($1 \leq i \leq n$) with $x_{iD} > d_i$, so there must be a product $j \neq i$ ($1 \leq j \leq n$) such that $x_{jD} < d_j$. But for $x_{iD} > d_i$, we have $|Dr_i - x_{iD}| > |d_i - x_{iD} + 1|$, and for $x_{jD} < d_j$, we have $|Dr_j - x_{jD}| > |d_j - x_{jD} - 1|$. Hence, by replacing the last copy of product i in the D -prefix of s by product j does not increase the absolute maximum-deviation for the D -prefix as the new two deviations strictly decrease and other remains the same. Therefore, it is easy to obtain a D -prefix where each product i occurs exactly d_i -times and with maximum deviation not exceeding $1 - \frac{1}{2(n-1)}$. This proves relation (b). Hence, the theorem is established by combining relations (a) and (b). \square

Dhamala et al. [18] establish the upper bound for the problem F_{\max}^s as

Theorem 5.4.3 *For any instance d_1, \dots, d_n ($n > 1$) of the MDJIT problem F_{\max}^s , the optimal value*

$$B^* \leq \left(1 - \frac{1}{D}\right)^2.$$

Proof [18]: To prove this theorem, it is sufficient to establish the relations of (5.29) for $B = \left(1 - \frac{1}{D}\right)^2$. Let $B = \left(1 - \frac{1}{D}\right)^2$, then $\lfloor k_2 r_i + \sqrt{B} \rfloor = \lfloor k_2 r_i + 1 - \frac{1}{D} \rfloor$. If $k_2 r_i$ is an integer, then $\lfloor k_2 r_i + 1 - \frac{1}{D} \rfloor = k_2 r_i$, and if $k_2 r_i$ is not an integer, then $k_2 r_i = \lfloor k_2 r_i \rfloor + \{k_2 r_i\}$ where $\{x\}$ denotes the fractional part of the number x . Since $\{k_2 r_i\} \geq \frac{1}{D}$, we have

$$\left\lfloor k_2 r_i + 1 - \frac{1}{D} \right\rfloor \geq \lfloor k_2 r_i \rfloor + 1 > k_2 r_i.$$

Therefore, $\left\lfloor k_2 r_i + 1 - \frac{1}{D} \right\rfloor \geq k_2 r_i$. Thus,

$$\sum_{i=1}^n \left(\lfloor k_2 r_i + \sqrt{B} \rfloor - \lfloor (k_1 - 1)r_i - \sqrt{B} \rfloor \right) \geq \sum_{i=1}^n k_2 r_i - \sum_{i=1}^n (k_1 - 1)r_i \geq k_2 - k_1 + 1.$$

On the other hand for such B , we have $\lceil k_2 r_i - \sqrt{B} \rceil = \left\lceil k_2 r_i - 1 + \frac{1}{D} \right\rceil$. If $k_2 r_i$ is an integer, then $\left\lceil k_2 r_i - 1 + \frac{1}{D} \right\rceil = k_2 r_i$, and if $k_2 r_i$ is not an integer, then $k_2 r_i = \lfloor k_2 r_i \rfloor + \{k_2 r_i\}$ where $\{x\}$ denotes the fractional part of the number x . Since $\{k_2 r_i\} \leq 1 - \frac{1}{D}$, we have

$$\left\lceil k_2 r_i - 1 + \frac{1}{D} \right\rceil = \left\lceil \lfloor k_2 r_i \rfloor + \{k_2 r_i\} - 1 + \frac{1}{D} \right\rceil \leq \left\lceil \lfloor k_2 r_i \rfloor \right\rceil < k_2 r_i.$$

Therefore, $\left\lceil k_2 r_i - 1 + \frac{1}{D} \right\rceil \leq k_2 r_i$. Thus,

$$\sum_{i=1}^n \left(\left\lceil k_2 r_i - \sqrt{B} \right\rceil - \left\lfloor (k_1 - 1)r_i + \sqrt{B} \right\rfloor \right) \leq \sum_{i=1}^n k_2 r_i - \sum_{i=1}^n (k_1 - 1)r_i \leq k_2 - k_1 + 1.$$

Hence, the V_1 -convex bipartite graph $G = (V_1 \cup V_2, E)$ constructed by the problem yields a perfect matching within this bound. \square

As a direct consequence of the Theorems 5.4.2 and 5.4.3, we have

Corollary 5.4.1 [47] *Upper bounds on the objective functions for the problem F_{sum}^a and F_{sum}^s are respectively*

$$F_{sum}^a(s) \leq nD \quad \text{and} \quad F_{sum}^s(s) \leq nD. \quad \square$$

Surprisingly, the upper bounds for both problems F_{sum}^a and F_{sum}^s are identical. However, the bounds are not tight (see [19]). Moreover, the lower bound for the problem F_{sum}^s is

$$\sum_{i=1}^n \frac{D^2 - d_i^2}{12D} \quad (\text{cf. [1]}).$$

The upper bounds for the problems F_{max}^{wa} and F_{max}^{ws} are w_{max} (cf. [50]) and

$$w_{max} \left(1 - \frac{1}{D} \right)^2 \quad (\text{cf. [18]}) \text{ respectively, where } w_{max} = \max_i w_i.$$

5.5 The Binary Search for Optimality

5.5.1 Binary Search for Absolute Deviation

The tight lower bound for the PRV-MDJIT problem F_{max}^a is proven to be $l - r_{max}$ in Section 5.4.1. For small examples, the optimal value of F_{max}^a very often coincides with the lower bound $l - r_{max}$. Depending up on this analysis *Kovalyov et al.* [26] conjectured that the value of the optimal MDJIT problem F_{max}^a will always be at the problem's theoretical lower bound, $l - r_{max}$. But, unfortunately this conjecture is refuted several

times in their computational experiments. In fact, in their 1,00,000 trials, approximately 25% of the time, the optimal F_{\max}^a value is found not to be at its lower bound (cf. [26]). As a matter of fact, assume that an optimal sequence attains its lower bound $1-r_{\max}$ and also assume that $r_{\max} > \frac{2}{3}$. Now, by (5.42) and (5.43), we have

$$\begin{aligned} 1 - r_{\max} &= B^* \\ r_{\max} &\leq 2B^* \end{aligned}$$

Combining these two implies

$$\begin{aligned} r_{\max} &\leq 2 - 2r_{\max} \\ \Rightarrow \quad \frac{2}{3} &< r_{\max} \leq \frac{2}{3} \\ \Rightarrow \quad \frac{2}{3} &< \frac{2}{3}. \end{aligned}$$

Which is absurd and as a result the lower bound cannot be tight as soon as $r_{\max} > \frac{2}{3}$ (cf. [7]).

Now, for an instance of PRV-MDJIT problem F_{\max}^a with $r_{\max} > \frac{2}{3}$, we use the binary search technique to find the optimum B^* in the interval $[(1-r_{\max}), 1]$ for which the corresponding V_I -convex bipartite graph $G(B^*)$ has a perfect matching and can be obtained by doing $O(\log D)$ tests.

Theorem 5.5.1 [49] *An optimal sequence for the problem F_{\max}^a can be determined when a binary search algorithm is performed in the interval $[1-r_{\max}, 1]$ in exact pseudo-polynomial time complexity $O(D \log D)$.*

Proof: Let the optimal value be B^* . Then for some values of i and k , it must hold

$$\begin{aligned} B^* &= |x_{ik} - kr_i| \\ \Rightarrow \quad DB^* &= D \left| x_{ik} - k \frac{d_i}{D} \right|. \end{aligned}$$

Hence, DB^* an integer in the interval $[D - d_{\max} \dots D]$, and the problem becomes to find an optimal integer DB^* in the interval $[D - d_{\max} \dots D]$, which requires to solve $O(\log d_{\max})$ decision problems. Since $E(i, j)$ and $L(i, j)$ can be calculated in $O(D)$ time, an optimal sequence can be obtained in $O(D \log d_{\max}) \leq O(D \log D)$ time. \square

The optimal solution to the weighted case can also be determined by the

Corollary 5.5.1 [50] *An optimal sequence for the problem F_{\max}^{wa} can be determined when a binary search is performed in the interval $[w_{\min}(1-r_{\max}), w_{\max}]$ in exact pseudo-polynomial time complexity $O(D \log(D\phi w_{\max}))$, where ϕ is a positive integer constant that depends on the problem data.* \square

Example 5.5.1 Here we consider an instance $(n; d_1, \dots, d_n) = (3; 1, 2, 7)$ of the PRV-MDJIT sequencing problem F_{\max}^a in which $r_{\max} = \frac{7}{10} > \frac{2}{3}$. So that the optimal value B^* of the

instance does not attain its problem's theoretical lower-bound $1 - r_{\max} = 1 - \frac{7}{10} = \frac{3}{10}$.

Hence, we search for optimal B^* under Binary Search Algorithm 2.3.1 as follows:

Since $B^* \in [1 - \frac{7}{10}, 1]$, then $10B^* \in [10 - 7 \dots 10] = [3 \dots 10]$.

Now, first we check:

$$\text{Is } 10B^* < 6?$$

i.e., is $B^* < \frac{6}{10}$?

Now, to check this, let $B_1 = \frac{5}{10}$ be a threshold value of the instance and the

corresponding graph $G(\frac{5}{10})$ is given by Figure 5.7 (a). Obviously, the graph $G(\frac{5}{10})$ does not contain any perfect matching since the time period 2 and 9 are not assigned to any copy of vertex set V_2 .

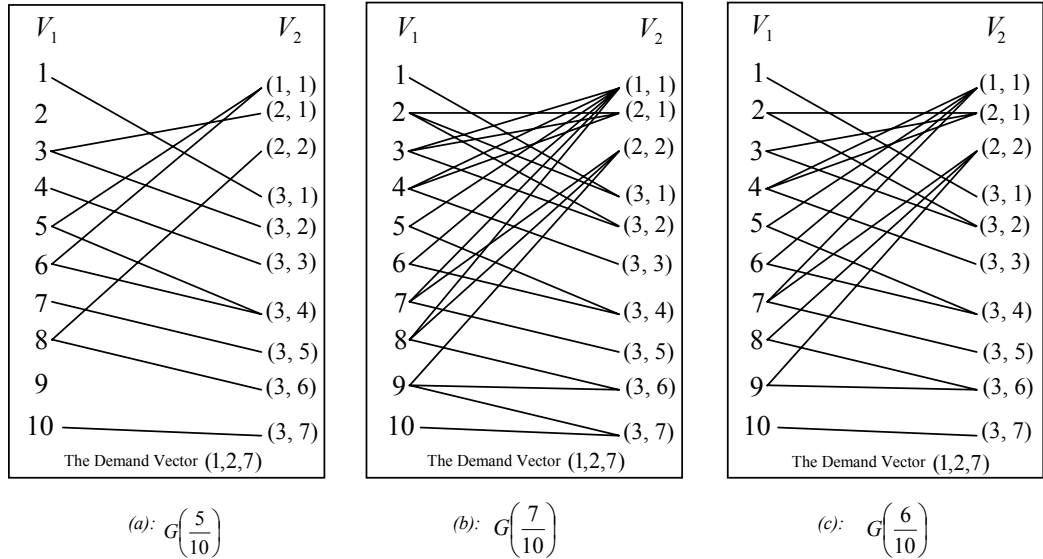


Figure 5.7: Bipartite Graphs for Different Threshold Value for F_{\max}^a

Thus, the answer of the question is no and we have $10B^* \in [6 \dots 10]$.

Again, we check:

$$\text{Is } 10B^* < 8?$$

That is, is $B^* < \frac{8}{10}$?

For that, let $B_2 = \frac{7}{10}$ be a threshold value of the instance and the corresponding graph

$G(\frac{7}{10})$ is Figure 5.7 (b).

By applying the EDD Algorithm in the graph $G(\frac{7}{10})$, we can easily find the order preserving perfect matching $M(\frac{7}{10})$ as

$$M(\frac{7}{10}) = \{[1, (3,1)], [2, (3,2)], [3, (2,1)], [4, (3,3)], [5, (3,4)], [6, (1,1)], [7, (3,5)], [8, (3,6)], [9, (2,2)], [10, (3,7)]\}.$$

Therefore, $10B^* \in \{6, 7\}$. Similarly we check;

$$\text{Is } 10B^* < 7?$$

$$\text{i.e. is } B^* < \frac{7}{10} ?$$

For that, let $B_3 = \frac{6}{10}$ be the threshold value of the instant and the corresponding graph is given by Figure 5.7 (c). Again, under the application of EDD algorithm for the graph $G(\frac{6}{10})$, we can obtain an order preserving perfect matching $M(\frac{6}{10})$ as

$$M(\frac{6}{10}) = \{ \text{As same as for threshold value } \frac{7}{10} \}.$$

Thus, we have $10B^* > 5$ and $10B^* \leq 6$. By which we must have

$$10B^* = 6.$$

This implies that

$$B^* = \frac{6}{10} = 0.6.$$

Which is the optimal solution to the instance $(n; d_1, \dots, d_n) = (3; 1, 2, 7)$ of the PRV-MDJJJ sequencing problem F_{\max}^a . The optimal sequence is $s = (3, 3, 2, 3, 3, 1, 3, 3, 2, 3)$.

By Theorem 5.4.2, the optimal value of the example 5.5.1 must satisfy

$$B^* \leq 1 - \max \left\{ \frac{1}{10}, \frac{1}{4} \right\} = 1 - \frac{1}{4} = \frac{3}{4}$$

$$\text{Therefore, } 1 - r_{\max} \leq B^* \leq \frac{3}{4}$$

$$\Rightarrow \frac{3}{10} \leq B^* \leq \frac{3}{4}.$$

But the value B^* is the integral multiple of $\frac{1}{D}$, so the optimal value B^* must be one among $\frac{4}{10}$, $\frac{5}{10}$, $\frac{6}{10}$, and $\frac{7}{10}$, since the optimal value does not attain its lower bound in this example. And clearly, the possible value $\frac{6}{10}$ is found to be optimal by binary search algorithm in the desired interval. \square

5.5.2 Binary Search for Squared Deviation

Theorem 5.5.2 [18] *An optimal sequence for the problem F_{\max}^s can be determined when a binary search algorithm is performed in the interval $\left[(1-r_{\max})^2, (1-\frac{1}{D})^2 \right]$ in exact pseudo-polynomial time complexity $O(D \log D)$. \square*

Corollary 5.5.2 [18] *An optimal sequence for the problem F_{\max}^{ws} can be determined when a binary search is performed in the interval $\left[w_{\min} (1-r_{\max})^2, w_{\max} (1-\frac{1}{D})^2 \right]$ in exact pseudo-polynomial time complexity $O(D \log(D^2 \phi w_{\max}))$, where ϕ is a positive integer constant that depends on the problem data. \square*

In Section 3.2, it has been shown that the bottleneck multi-level problems \tilde{F}_{\max}^{wa} and \tilde{F}_{\max}^{ws} with pegging assumption can be reduced to weighted single level problems F_{\max}^{wa} and F_{\max}^{ws} respectively and Corollary 5.5.1 and 5.5.2 demonstrates that the problems F_{\max}^{wa} and F_{\max}^{ws} can be solved for optimality in time which is polynomial in D and in the size of the weighting factors. Hence, optimal solutions to the balanced sequence problem for multi-level, pegging models can also be efficiently determined.

5.6 The Cyclic Sequences

For any instance d_1, \dots, d_n with $D = \sum_{i=1}^n d_i$ of the PRV-MMJIT sequencing problem F_{sum} , we have summarized the existence of cyclic optimal sequence in Section 4.2.4. On the other hand *Steiner and Yeomans* [50] establish the existence of optimal cyclic sequences for bottleneck absolute deviation both for weighted and un-weighted cases for the first time. Subsequently, *Dhamala et al.* [18] prove the existence of optimal cyclic sequences for the bottleneck squared deviation. In this section, we describe the existence of optimal cyclic sequences for any instance of the bottleneck problems F_{\max}^a and F_{\max}^s by which the computational complexity reduce significantly. Let $m = \gcd(d_1, \dots, d_n)$. Then the product requirement vector becomes $d = (d_1, \dots, d_n) = (mm_1, \dots, mm_n)$ with $d_i = mm_i, \forall i = 1, \dots, n$.

Letting $A = \sum_{i=1}^n m_i$, then $D = \sum_{i=1}^n d_i = mA$ and $r_i = \frac{d_i}{D} = \frac{mm_i}{mA} = \frac{m_i}{A}$.

Let each copy of the product be labeled as $(k-1)m_i + c$, where $k = 1, \dots, m$ and $c = 1, \dots, m_i$. Then for each fixed value of k, there will be a group of m_i copies of product i in the range

$$[(k-1)m_i + 1 \dots (k-1)m_i + m_i].$$

This range will be referred to as the k^{th} tier (cf. [50]) of copies for product i . The following lemma is implicit in [18] and [50].

Lemma 5.6.1 For a threshold value $B < 1$, we have the linear relations

$$E(i, km_i + c) = E(i, c) + kA \text{ and } L(i, km_i + c) = L(i, c) + kA.$$

Proof: We give the proof only for the case F_{\max}^a of Steiner and Yeomans [50] and the proof for the case F_{\max}^s can be obtained similarly from [18].

$$\begin{aligned} E(i, km_i + c) &= \left\lceil \frac{km_i + c - B}{r_i} \right\rceil \\ &= \left\lceil \frac{(k-1)m_i + c - B + m_i}{r_i} \right\rceil \\ &= \left\lceil \frac{(k-1)m_i + c - B}{r_i} + A \right\rceil \\ &= \left\lceil \frac{(k-1)m_i + c - B}{r_i} \right\rceil + A \\ &= E(i, (k-1)m_i + c) + A \\ &= E(i, c) + kA. \end{aligned}$$

Similarly, it can be proved that $L(i, km_i + c) = L(i, c) + kA$ □

The Lemma 5.6.1 implies that only the early and late producing times for copies $c = 1, \dots, m_i$ in the first tier need to be calculated, as the produce times for all copies in the remaining tiers are linear function of those in the first tier.

Lemma 5.6.2 For bottleneck $B < 1$, then for all $i = 1, \dots, n$ and $k = 1, \dots, m$, we have

$$(a) \quad L(i, km_i) \leq kA$$

$$(b) \quad (k-1)A < E(i, (k-1)m_i + 1)$$

□

An obvious result of Lemma 5.6.2 is

Corollary 5.6.1 For an bottleneck $B < 1$, then for all $i = 1, \dots, n$ and $k = 1, \dots, m-1$, we have

$$[E(i, (k-1)m_i + 1) \dots L(i, km_i)] \cap [E(i, km_i + 1) \dots L(i, (k+1)m_i)] = \emptyset.$$

□

Now, we state the main result from [18, 31, 50].

Theorem 5.6.1 Let $m = \gcd(d_1, \dots, d_n)$. Then the problem

(a) F_{\max}^a has an optimal sequence which consists of m repetitions of the optimal sequence to the sub-problem where $d = (m_1, \dots, m_n)$.

(b) F_{\max}^s has an optimal sequence which consists of m repetitions of the optimal sequence to the sub-problem where $d = (m_1, \dots, m_n)$.

Proof:

(a) Consider any optimal sequence $s = (s_1, \dots, s_D)$ to the problem F_{\max}^a with objective value $B^* < 1$. By Theorem 5.4.2 such a solution always exists. If s itself is an m repetitions of the optimal sequence to the sub-problem where $d = (m_1, \dots, m_n)$, then there is nothing to prove further. Else, a copy (i, j) of product i occupies a position in the interval $[E(i, j) \dots L(i, j)]$. Moreover, Lemma 5.6.1 implies that each interval $[(k-1)A+1 \dots kA]$ consists of m units of products and hence by Corollary 5.6.1 we can rearrange the products sub-sequence on each of the interval $[(k-1)A+1 \dots kA]$, for $k = 2, \dots, m$ as in the first interval $[1 \dots A]$ without destroying the B^* feasibility of the sequence.

(b) The proof directly follows from Theorem 5.4.3 and (a). □

It is also noted that the cyclic sequence analogously exists for the weighted problem with appropriate weights (see [18, 50]).

Dhamala and Kubiak [20] conjectured that the cyclic JIT sequences in multi-level problem are optimal and till now there are no further investigation on this conjecture.

5.7 Small Deviations

5.7.1 Small Deviations for F_{\max}^a

Since the optimal cyclic sequences exist, without loss of generality, in this section, we may assume that all instances are standard. It is noted that the bound $B^* \geq \frac{1}{\Delta_i} \left\lfloor \frac{\Delta_i}{2} \right\rfloor$ is

usually close to $\frac{1}{2}$, and then $B^* \geq \frac{1}{2}$ as soon as there exists some i such that Δ_i is even.

On the other hand the bound $\left(\frac{n-1}{2n-1}\right) \leq B^*$, established in Proposition 6.3.1, goes to

$\frac{1}{2}$ as $n \rightarrow \infty$. Thus, in this section we look more closely at those instances for which

the maximum deviation does not exceed the value $\frac{1}{2}$ and the ultimate goal of this

section is to identify all instances with optimal value $B^* < \frac{1}{2}$. Clearly for $B^* < \frac{1}{2}$ the

condition, $|x_{ik} - kr_i| \leq B^*$ forces x_{ik} to be equal to $\lfloor kr_i \rfloor$ and the kr_i not be equal to sum

of some integral and $\frac{1}{2}$ by which the problem becomes highly constrained.

Lemma 5.7.1 [30] *For $B^* < \frac{1}{2}$, each copy (i, j) must be sequenced in its ideal position and no ideal corner is integer.*

Proof [30]: For $\Delta \geq 1$ an integer, assume that a copy (i, j) is sequenced in position

$$\left\lceil \frac{2j-1}{2r_i} \right\rceil + \Delta. \text{ Then, } x_{i \left\lceil \frac{2j-1}{2r_i} \right\rceil} \leq j-1.$$

But, we have

$$\left\lceil \frac{2j-1}{2r_i} \right\rceil r_i \geq \frac{2j-1}{2r_i} r_i = j - \frac{1}{2}.$$

Therefore, at the time period $\left\lceil \frac{2j-1}{2r_i} \right\rceil$, we have $\left| x_{i \left\lceil \frac{2j-1}{2r_i} \right\rceil} - \left\lceil \frac{2j-1}{2r_i} \right\rceil r_i \right| \geq \frac{1}{2}$,

which contradicts to $B^* < \frac{1}{2}$.

Now, assume that copy (i, j) is sequenced in position $\left\lceil \frac{2j-1}{2r_i} \right\rceil - \Delta$, where Δ is as before. Then, $x_{i \left\lceil \frac{2j-1}{2r_i} \right\rceil - 1} \geq j$.

But $\left(\left\lceil \frac{2j-1}{2r_i} \right\rceil - 1 \right) r_i \leq j - \frac{1}{2}$.

Therefore, at time period $\left\lceil \frac{2j-1}{2r_i} \right\rceil - 1$, $\left| x_{i \left\lceil \frac{2j-1}{2r_i} \right\rceil - 1} - \left(\left\lceil \frac{2j-1}{2r_i} \right\rceil - 1 \right) r_i \right| \geq \frac{1}{2}$.

This is again a contradiction to the hypothesis $B^* < \frac{1}{2}$. Therefore, for $B^* < \frac{1}{2}$, we have to sequence each copy (i, j) in its ideal position.

Now, let the ideal corner of copy (i, j) be integer.

Then, $\left\lceil \frac{2j-1}{2r_i} \right\rceil = \frac{2i-1}{2r_i}$.

And we must have $x_{i \left\lceil \frac{2j-1}{2r_i} \right\rceil} = j$

Therefore,

$$\left| x_{i \left\lceil \frac{2j-1}{2r_i} \right\rceil} - \left\lceil \frac{2j-1}{2r_i} \right\rceil r_i \right| = \left| j - \left(j - \frac{1}{2} \right) \right| = \frac{1}{2},$$

which is a contradiction. Hence, no one ideal corner is an integer for $B^* < \frac{1}{2}$. □

Lemma 5.7.2 [30] For $B^* < \frac{1}{2}$,

- (a) exactly one ideal corner falls inside the interval $[k-1, k]$, for $k = 1, 2, \dots, D$.
- (b) $1 < \frac{D}{d_n} < 2$ and $\frac{D}{d_i} > 2$ for $i = 1, \dots, n-1$
- (c) exactly one d_i , $i = 1, \dots, n$ is odd.

Proof [30]: (a) By definition there are exactly D numbers of ideal corners and positions. Again, we can sequence exactly one copy at a time. Then, the proof immediately follows from Lemma 6.5.1.

(b) For standard instance, it is clear that $1 < \frac{D}{d_n}$. Now, we show $\frac{D}{d_n} < 2$. On the contrary assume that $\frac{D}{d_n} \geq 2$, i.e. $\frac{D}{2d_n} \geq 1$. Then, due to Lemma 5.1.1 and 6.5.1, the first and the last ideal corners of product n do not fall inside $[0,1]$ and $[D-1,D]$ respectively. Moreover, none ideal corner falls in either $[0,1]$ or $[D-1,D]$, since $\frac{D}{d_i} \geq \frac{D}{d_n}$ for $i = 1, \dots, n-1$, and standard instances, which contradicts (a) and hence we must have $\frac{D}{d_n} < 2$. Now, it remains to show that $\frac{D}{d_i} > 2$ for $i = 1, \dots, n-1$. Suppose on the contrary $\frac{D}{d_i} \leq 2$ for some $i = 1, \dots, n-1$. Then $\frac{D}{2d_i} \leq 1$ and consequently, the two ideal corners of product i shares both time intervals $[0,1]$ and $[D-1,D]$ with n , which again contradicts the part (a) and proves that $\frac{D}{d_i} > 2$ for $i = 1, \dots, n-1$.

(c) For a product i with odd d_i , we have $d_i = 2k + 1$ for some integer $k \geq 0$. Then the ideal corner of copy $(i, k+1)$ is $\frac{2(k+1)-1}{2r_i} = \frac{D}{2}$. Therefore, each product i with an odd d_i has one of its ideal corners at $\frac{D}{2}$. But by part (a), there is no more than one d_i ($1 \leq i \leq n$). Moreover, for being standard instance one d_i must be odd, otherwise the instance will not be standard since $\gcd(d_1, \dots, d_n, D) \geq 2$. \square

Proposition 5.7.1 [7, 8] *Let $d_i, i = 1, 2, \dots, n$ be an instance of the MDJIT problem F_{\max}^a for which there is an optimal sequence s with $B^* < \frac{1}{2}$. Then s generates a periodic word*

$w = (sss\dots)$ with distinct rates $\frac{d_1}{D}, \dots, \frac{d_n}{D}$ which is symmetric and 1-balanced.

Proof: Let s be the optimal sequence for the instance (d_1, \dots, d_n) with $B^* < \frac{1}{2}$ and define an infinite sequence obtained by repeating s , i.e. $w = (ss\dots)$. Now, we show that the infinite sequence w is periodic, symmetric and 1-balanced on alphabet $\{1, \dots, n\}$. The periodicity of w immediately follows from the definition.

Now, let $t \in \mathbb{N}$ and S_1, S_2 be two subsequences consisting of t consecutive elements of w . Assume that S_j ranges from time $t_j + 1$ to time $t_j + t$, for $j = 1, 2$. Fix $i \in \{1, 2, \dots, n\}$ and denote by $|I|$ the number of occurrences of i in any time interval I . Then for $j = 1, 2$,

$$|[t_j + 1 \dots t_j + t]| = |[1 \dots t_j + t]| - |[1 \dots t_j]| < \left((t_j + t)r_i + \frac{1}{2} \right) - \left(t_j r_i - \frac{1}{2} \right) = tr_i + 1$$

and similarly

$$|[t_j + 1 \dots t_j + t]| = |[1 \dots t_j + t]| - |[1 \dots t_j]| > \left((t_j + t)r_i - \frac{1}{2} \right) - \left(t_j r_i + \frac{1}{2} \right) = tr_i - 1.$$

Thus, $|[t_1 + 1 \dots t_1 + t]|$ and $|[t_2 + 1 \dots t_2 + t]|$ are two integers in the interval $(tr_i - 1, tr_i + 1)$. There follows that $|[t_1 + 1 \dots t_1 + t]|$ and $|[t_2 + 1 \dots t_2 + t]|$ differ at most by 1, and hence w is 1-balanced.

Now, the symmetricity follows as

$$\begin{aligned} x_{i(D-k)} - x_{i(D-k-1)} &= [(D-k)r_i] - [(D-k-1)r_i] \\ &= d_i - [kr_i] - d_i + [(k+1)r_i] \\ &= [(k+1)r_i] - [kr_i] \\ &= x_{i(k+1)} - x_{ik}. \end{aligned}$$

Finally all the rates are distinct. Indeed, if $r_i = r_j$ for $i \neq j$, then $x_{ik} = [kr_i] = [kr_j] = x_{jk}$ for all k , which is clearly impossible for $B^* < \frac{1}{2}$. Hence, the conclusion follows. \square

Theorem 5.7.1 [8] For $n \geq 3$ a standard instance (d_1, \dots, d_n) of the PRV-MDJIT problem F_{\max}^a has optimal value $B^* < \frac{1}{2}$ if and only if the demand vector

$$(d_1, \dots, d_i, \dots, d_n) = (1, \dots, 2^{i-1}, \dots, 2^{n-1}) \text{ and } B^* = \frac{2^{n-1} - 1}{2^n - 1}.$$

Proof: *Necessity:* Assume that the standard instance (d_1, \dots, d_n) of the PRV-MDJIT problem F_{\max}^a has an optimal finite sequence s of length $D = \sum_{i=1}^n d_i$ with $B^* < \frac{1}{2}$. Then by

Proposition 5.7.1, s generates an infinite periodic, symmetric and 1-balanced word $w = (sss\dots)$ with distinct rates (r_1, \dots, r_n) . Therefore, Theorem 2.2.2 guarantees that the rates should be

$$r_i = \frac{2^{i-1}}{2^{n-1}} \quad \text{for each } i = 1, \dots, n.$$

Hence, the demand vector is given by

$$(d_1, \dots, d_i, \dots, d_n) = (Dr_1, \dots, Dr_i, \dots, Dr_n) = (1, \dots, 2^{i-1}, \dots, 2^{n-1}).$$

Now, it remains to show that for such instance the optimal value $B^* = \frac{2^{n-1} - 1}{2^n - 1}$. For that,

Theorem 5.4.1 suggests that $B^* \geq \frac{1}{\Delta_i} \left\lfloor \frac{\Delta_i}{2} \right\rfloor$ for $i = 1, \dots, n$. Now, let

$B^* = \frac{D-1}{2D} = \frac{2^{n-1} - 1}{2^n - 1} < \frac{1}{2}$ be the threshold value of the problem. Again, by Lemma 5.7.1, for $B^* < \frac{1}{2}$, each copy of each product must be sequenced in its ideal position. Hence, to

test the feasibility of B^* , by Theorem 5.2.1, it is sufficient to show that $E(i, j) \leq Z_{ij}^* \leq L(i, j)$, where

$$Z_{ij}^* = \left\lceil \frac{2j-1}{2r_i} \right\rceil = \left\lceil \frac{2j-1}{2 \frac{2^{i-1}}{2^n-1}} \right\rceil = \left\lceil \frac{(2^n-1)(2j-1)}{2^i} \right\rceil = \left\lceil 2^{n-i}(2j-1) - \frac{2j-1}{2^i} \right\rceil = 2^{n-i}(2j-1),$$

is the ideal position of copy (i, j) .

Indeed,

$$\begin{aligned} Z_{ij}^* - \frac{j-B^*}{r_i} &= \frac{2^{n-1}(2j-1) - j(2^n-1) + 2^{n-1} - 1}{2^{i-1}} \\ &= \frac{j-1}{2^{i-1}} \geq 0, \end{aligned}$$

and

$$\frac{j-1+B^*}{r_i} + 1 - Z_{ij}^* = \frac{(j-1)(2^n-1) + 2^{n-1} - 1 + 2^{i-1} - 2^{n-1}(2j-1)}{2^{i-1}} = \frac{2^{i-1} - j}{2^{i-1}} \geq 0.$$

Now, Z_{ij}^* integer implies that

$$E(i, j) = \left\lfloor \frac{j-B^*}{r_i} \right\rfloor \leq Z_{ij}^* \leq \left\lceil \frac{j-1+B^*}{r_i} + 1 \right\rceil = L(i, j).$$

Furthermore, $Z_{ij}^* \neq Z_{i'j'}$, for $i \neq i'$ or $j \neq j'$ and clearly $1 \leq Z_{ij}^* \leq D = 2^{n-1}$.

Hence, the necessary condition.

Sufficiency: Assume that the demand vector of a standard instance of the PRV-MDJIT problem F_{\max}^a is given by

$$(d_1, \dots, d_i, \dots, d_n) = (1, \dots, 2^{i-1}, \dots, 2^{n-1}).$$

Now, as shown in the necessity condition. Such standard instance has an optimal finite sequence s of length $D = \sum_{i=1}^n d_i$ with $B^* = \frac{2^{n-1} - 1}{2^n - 1} < \frac{1}{2}$.

Which completes the proof of the Theorem. \square

The structure of instances with $B^* = \frac{1}{2}$ becomes more complex as x_{ik} may now be equal to either $\lceil kr_i \rceil$ or to $\lceil kr_i \rceil = kr_i + \frac{1}{2}$ when kr_i is half integral and hence the following conjecture is formulated by Brauner and Crama [7].

Conjecture 5.7.1 *A standard instance (d_1, \dots, d_n) of the PRV-MDJIT problem F_{\max}^a has optimal value $B^* \leq \frac{1}{2}$ if and only if it satisfies one of the following conditions:*

- (a) d_1 and d_2 are arbitrary and for $i \geq 3$, d_i is the sum of all demands with smaller index: $d_i = \sum_{j<i} d_j = 2^{i-3} (d_1 + d_2)$ for all $i \in [3..n]$,

- (b) $d_1 = 1, d_2 = 2, d_3 = 9$ and for $i \geq 4$, d_i is the sum of all demands with smaller index: $d_1 = 1; d_2 = 2; d_3 = 9; d_i = \sum_{j < i} d_j = 2^{i-4} \times 12$ for all $i \in [4 \dots n]$,
- (c) $d_1 = 2, d_2 = 3, d_3 = 7$ and for $i \geq 4$, d_i is the sum of all demands with smaller index: $d_1 = 2; d_2 = 3; d_3 = 7; d_i = \sum_{j < i} d_j = 2^{i-4} \times 12$ for all $i \in [4 \dots n]$,
- (d) There exists an index $\ell \in [3 \dots n]$ such that
 $d_i = 2^{i-1}$ for all $i \in [1 \dots \ell]$;
 $d_i = \sum_{j < i} d_j = 2^{i-\ell-1}(2^\ell - 1)$ for all $i \in [\ell + 1 \dots n]$,
- (e) There exists an index $l \in [3 \dots n]$ such that
 $d_i = 2^{i-1}$ for all $i \in [1 \dots l-1]$,
 $d_l = 2^{l-1} + 1, d_i = \sum_{j < i} d_j = 2^{i-1}$ for all $i \in [l+1 \dots n]$. \square

Proposition 5.7.2 [7] For the feasible instance $(n; d_1, \dots, d_n; B^*)$ of the PRV-MDJIT problem F_{\max}^a with $B^* \geq \frac{1}{2}$. Then the following instance, involving an additional product, is also feasible:

$$(n+1; d_1, \dots, d_n, d_{n+1} = \sum_{i=1}^n d_i; B^*). \quad \square$$

5.7.2 Small Deviations for F_{\max}^s

The Proposition 5.7.1 for the problem F_{\max}^s can be generalized as

Corollary 5.7.1 Let $d_i, i = 1, 2, \dots, n$ be an instance of the MDJIT problem F_{\max}^s for which there is an optimal sequence s with $B^* < \frac{1}{4}$. Then s generates a periodic word $w = (sss \dots)$ with distinct rates $\frac{d_1}{D}, \dots, \frac{d_n}{D}$ which is symmetric and 1- balanced. \square

Theorem 5.7.2 For $n \geq 3$ a standard instance (d_1, \dots, d_n) of the PRV-MDJIT problem F_{\max}^s has optimal value $B^* < \frac{1}{4}$ if and only if the demand vector

$$(d_1, \dots, d_i, \dots, d_n) = (1, \dots, 2^{i-1}, \dots, 2^{n-1}) \text{ and } B^* = \left(\frac{2^{n-1} - 1}{2^n - 1} \right)^2. \quad \square$$

Corollary 5.7.2 There is no instance (d_1, \dots, d_n) with $n \geq 2$ of PRV- MMJIT problem

- (a) [19] F_{\max}^a that has a feasible solution with $B < \frac{1}{3}$.
- (b) [18] F_{\max}^s that has a feasible solution with $B < \frac{1}{9}$. \square

For any finite sequences s of length $D = \sum_{i=1}^n d_i$ with maximum deviation B^* for n -product demands $d_i, i = 1, \dots, n$, any infinite periodic word w of period s is 1-Balanced, 2-Balanced, or 3-Balanced on each product i if $B^* < \frac{1}{2}$, $B^* < \frac{3}{4}$ and $B^* < 1$, respectively. But any sequence with $d_i = 1, \forall i = 1, \dots, n$, is a 1-Balanced word though its maximum deviation $B^* = 1 - \frac{1}{n} > \frac{1}{2}$ for $n \geq 3$ (see [20]). However, the maximum deviation B^* is greater than $\frac{3}{4}$ for the 2-Balanced word $w = (s_1, s_1, s_2, s_2, \dots, s_n, s_n)$ with $d_i = 2$ for each $i = 1, \dots, n$. Similarly, the maximum deviation B^* is greater than 1 for the 3-Balanced word $w = (s_1, s_1, s_1, s_2, s_2, s_2, \dots, s_n, s_n, s_n)$ with $d_i = 3$ for each $i = 1, \dots, n$, with $n \geq 3$. Hence, we have

Theorem 5.7.3 [14] *Let s be a finite sequence of length $D = \sum_{i=1}^n d_i$ with maximum deviation B^* for n -ideal rates $r = (r_1, \dots, r_n)$, and let $S^{\frac{1}{2}}, S^{\frac{3}{4}}$ and S^1 be the sets of sequences with $B^* < \frac{1}{2}, B^* < \frac{3}{4}$ or $B^* < 1$ respectively. Then $S^{\frac{1}{2}}, S^{\frac{3}{4}}$ and S^1 are properly contained in the sets of 1-Balanced, 2-Balanced and 3-Balanced words, respectively. \square*

From Theorem 5.4.2, 5.4.3, 5.7.2 we conclude that all the optimal solutions of the PRV-MDJIT problems F_{\max}^a and F_{\max}^s belongs to the set of all 3-Balanced words. But the question whether there always exists a 2-Balanced word that optimizes PRV-MDJIT problem raised by *Dhamala and Kubiak* [20] still remains open. Hence, we have the Figure 5.8 showing the conjectured topography.

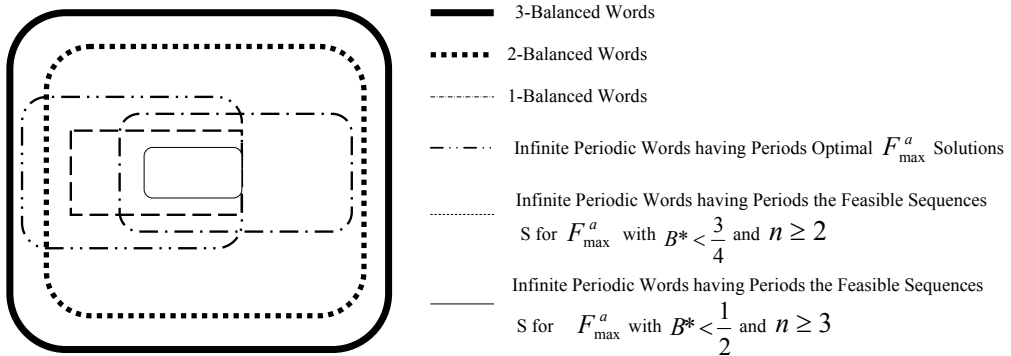


Figure 5.8: A Conjectured Topography of Balanced Words and the Solutions of JITSP F_{\max}^a

5.8 Two Product Problem

Here we study the PVR-MDJIT problem for $n = 2$ with ideal production rates r_1 and $r_2 = 1 - r_1$. We assume without loss of generality that $0 < d_1 \leq d_2$.

Theorem 5.8.1 *The matrix $X = (x_{ik})_{2 \times D}$ defined by $x_{1k} = \lceil kr_1 \rceil$, $x_{2k} = k - \lceil kr_1 \rceil$ for $k = 1, \dots, D$ is an optimal solution of the 2-product PRV-MDJIT problems F_{\max}^a and F_{\max}^s .*

Proof: Here we give the proof only for F_{\max}^a from the reference [7] and the proof for F_{\max}^s is clearly dominated by the former. For $k = 1, \dots, D$, we have

$$\begin{aligned} x_{1k} + x_{2k} &= \lceil kr_1 \rceil + k - \lceil kr_1 \rceil = k, \\ x_{10} = x_{20} &= 0, \quad \text{and} \quad x_{iD} = d_i, \quad i = 1, 2, \\ x_{1k} - x_{1(k-1)} &= \lceil kr_1 \rceil - \lceil (k-1)r_1 \rceil \geq 0, \end{aligned}$$

and $x_{2k} - x_{2(k-1)} = k - \lceil kr_1 \rceil - k + 1 + \lceil (k-1)r_1 \rceil = \lceil (k-1)r_1 \rceil - \lceil kr_1 \rceil + 1 \geq 0$.

Therefore, the matrix $X = (x_{ik})_{2 \times D}$ satisfies all the constraints (3.17) - (3.20). Now, it remains to show that $X = (x_{ik})_{2 \times D}$ is an optimal solution for the MDJIT problem F_{\max}^a with $n = 2$. Clearly,

$$|x_{1k} - kr_1| = |\lceil kr_1 \rceil - kr_1| \leq \frac{1}{2},$$

and $|x_{2k} - kr_2| = |k - \lceil kr_1 \rceil - k + kr_1| = |-\lceil kr_1 \rceil + kr_1| \leq \frac{1}{2}$.

Now, let $X' = (x'_{ik})_{2 \times D}$ be another feasible solution of the 2-product MDJIT problem F_{\max}^a . We want to show that $X' = (x'_{ik})_{2 \times D}$ has maximum deviation larger than or equal to $\frac{1}{2}$, which implies that $X' = (x'_{ik})_{2 \times D}$ is not better than $X = (x_{ik})_{2 \times D}$. Assume that X' differs from X at period l . Now, by the constraint $x_{1k} + x_{2k} = k$, $\forall k = 1, \dots, D$, it must be the case that x'_{1l} is not equal to $x_{1l} = \lceil lr_1 \rceil$. Thus, by definition of the operator $\lceil \cdot \rceil$, x'_{1l} is at distance at least $\frac{1}{2}$ from lr_1 , i. e. $|x'_{1l} - lr_1| \geq \frac{1}{2}$, which is needed. \square

The above theorem solves the two product MDJIT problems F_{\max}^a and F_{\max}^s in polynomial time, in the sense that; at every time period k , the theorem allows to determine efficiently which copy should be produced at time period k . Moreover, the optimal value of the 2-product MDJIT problems F_{\max}^a and F_{\max}^s can be computed very easily as follow:

Theorem 5.8.2 [7] *The optimal value B_a^* and B_s^* of the objective functions of the 2-product MDJIT problems F_{\max}^a and F_{\max}^s are respectively given by*

$$B_a^* = \frac{1}{\Delta} \left\lfloor \frac{\Delta}{2} \right\rfloor, \quad \text{and} \quad B_s^* = \left(\frac{1}{\Delta} \left\lfloor \frac{\Delta}{2} \right\rfloor \right)^2 \quad \text{where,} \quad \Delta = \frac{D}{\gcd(d_1, D)} = \frac{D}{\gcd(d_2, D)}. \quad \square$$

Corollary 5.8.1 *The optimal objective values $B_a^* < \frac{1}{2}$ and $B_s^* < \frac{1}{4}$ if and only if one of the demands d_1 or d_2 is odd and the next even. Moreover,*

$$B_a^* = \frac{1}{2} - \frac{1}{2D} \quad \text{and} \quad B_s^* = \left(\frac{1}{2} - \frac{1}{2D} \right)^2$$

for odd instances, and

$$B_a^* = \frac{1}{2} \quad \text{and} \quad B_s^* = \frac{1}{4}$$

for even instances.

Proof: The proof is the direct conclusion of Theorem 5.8.2. □

Hence, in conclusion for any $n \geq 3$, there is only one standard instance with maximum deviation less than $\frac{1}{2}$ for F_{\max}^a and there is only one standard instance with maximum deviation less than $\frac{1}{4}$ for F_{\max}^s , but the number of standard instances with maximum deviation less than $\frac{1}{2}$ for F_{\max}^a and less than $\frac{1}{4}$ for F_{\max}^s with $n = 2$ are infinite.

5.9 Bottleneck Assignment for F_{\max}

One means of solution lies in the reduction of the bottleneck F_{\max} to a linear bottleneck assignment problem (LBAP) such an approach is independent of discrepancy functions $f_i, \forall i$ and allows us to use either symmetric or asymmetric discrepancy functions. The idea of reducing the F_{\max} to LBAP was first mentioned by *Kubiak* [29], and is in the form.

$$[\text{P5.1}] \quad \min \max_{i,j,k} \{C_{ijk} y_{ijk}\},$$

subject to the constraints (4.13)-(4.14), where C_{ijk} and y_{ijk} are defined as in Section 4.2. Although, it has not been further developed since its formulation (see [42]). The question whether an optimal solution to the LBAP [P5.1] is optimal to the problem F_{\max}^a raised by *Kubiak* [29] still remains open (see [19]). But *Bautista et al.* [5] reduce the bottleneck problem F_{\max} to an equivalent LBAP with assignment matrix

$$\Gamma = (\lambda_{(i,j)k})_{D \times D} = (\max\{|(j-1) - (k-1)r_i|, |j - kr_i|\})_{D \times D}$$

where $\lambda_{(i,j)k}$ corresponds to the production of a copy (i,j) at time period k for $i = 1, \dots, n; j = 1, \dots, d_i; k = 1, \dots, D$. For solving the problem F_{\max} , it suffices to use the discrepancy functions $f_i(x) = |x|, \forall i = 1, \dots, n$ (cf. [42]). Therefore, for such functions, we may take the assignment matrix as

$$\tilde{\Lambda} = (D\lambda_{(i,j)k})_{D \times D} = (\tilde{\lambda}_{(i,j)k})_{D \times D}.$$

This enables to achieve more rapid performance of different algorithms. The LBAP matrix values $\tilde{\lambda}_{(i,j)k}$ grow to the right and to the left from the ideal position $\left\lceil \frac{2j-1}{2r_i} \right\rceil$.

Since the optimal solution to the problem F_{\max} is always strictly less than 1, it is possible to avoid computing a complete matrix by only computing elements for which $|x_{ik} - kr_i| < 1$.

Moreno and Corominas [42] study the problem F_{\max} using LBAP. They perform a computational experiment adopting three main approaches, viz;

- (a) Solving LBAP by means of specific LBAP algorithms.
- (b) Solving LBAP as a sequence of assignment problem using binary matrix and binary search.
- (c) Solving LBAP as a sequence of matching problem with and without binary search.

They recommend the application of third approach for the solution of bottleneck F_{\max} as this outperforms the computational results in their experiment. For, this, they start with a heuristic solution. If the heuristic solution is close to the optimal solution, the search interval can be reduced that ultimately reduces the number of iterations of the matching problem.

5.10 Complexity Status of F_{\max}^a

The input of the generic JIT sequencing problem is essentially the list of integers d_1, \dots, d_n , so that its input size is $O\left(\sum_{i=1}^n d_i\right) = O(n \log D)$. Hence, an algorithm which is polynomial in n and D is only pseudo-polynomial, but not polynomial in the size of the problem. Thus, the recognition version of the PRV-MMJIT sequencing problem F_{\max} is pseudo-polynomial in size, since it involves nD variables and $O(nD)$ constraints. So obtaining truly polynomial algorithms for JIT sequencing problems is far from trivial (if possible at all) and requires deep insight into the structural properties of the problems. *Bruener and Crama* [7] prove the following:

Theorem 5.10.1 [7] *The PRV-MDJIT SP F_{\max}^a is in Co-NP.*

Proof [7]: Let $(n; d_1, \dots, d_n; B)$ be an instance of the problem F_{\max}^a which is not feasible for F_{\max}^a . Then by Corollary 5.3.1, there exists $k_1, k_2 \in [1 \dots D], k_1 \leq k_2$ such that one of the two inequalities (5.38) or (5.39) does not hold. For given k_1 and k_2 this can be checked for validity in time $O(n \log D)$. Hence, the PRV-MDJITSP F_{\max}^a is in Co-NP class. \square

The PRV-MDJITSP F_{\max}^a is efficiently solvable for $n = 2$ (see Theorem 5.8.1). But for fixed $n > 2$, we are not aware of a direct proof of the fact that the problem F_{\max}^a is polynomially solvable. Theorem 5.10.1 also leaves open the more challenging question: whether the recognition version of F_{\max}^a is in NP or is Co-NP-complete problem?

5.11 Application of F_{\max}^a Solution

The optimal F_{\max}^a solution can be applied to resource allocation problems in diverse environment, for example, the generalized pinwheel scheduling problem and the Liu-Layland periodic scheduling in NP -hard real-time environments (see [28]). Let $(a_1, b_1), \dots, (a_n, b_n)$ be n pairs of positive integers.

Definition 5.11.1 A generalized pinwheel schedule on alphabet $\{1, 2, \dots, n\}$ is an infinite sequence $s = (s_1, s_2, \dots)$ such that

1. $s_j \in \{1, 2, \dots, n\}$ for all $j \in \mathbb{N}$, and
2. each $i \in \{1, 2, \dots, n\}$ occurs at least a_i times in any subsequence σ consisting of b_i consecutive elements of s .

The schedule for $a_1 = a_2 = \dots = a_n = 1$ refers to simply as a pinwheel schedule.

Theorem 5.11.1 [28] Let $r_i = \frac{d_i}{D} = \frac{a_i}{b_i}$, where a_i and b_i are relatively prime, be the rate for the letter i and $(s_{j+1}, \dots, s_{j+b_i})$ be any subsequence of b_i consecutive letters of $s = (SSS\dots)$, with S obtained by the min-max algorithm with $B < 1$. Then letter i occurs either $a_i - 1$ or a_i , or $a_i + 1$ times in the subsequence. Moreover, if subsequences $(s_{j+1}, \dots, s_{j+b_i})$ and $(s_{j+1+kb_i}, \dots, s_{j+(k+1)b_i})$, for some $k \geq 1$ have $a_i - 1$ letter i occurrences each, then $k \geq 2$ and there are exactly $(k-1)a_i + 1$ i 's in the subsequence $(s_{j+1+b_i}, \dots, s_{j+kb_i})$. \square

With this theorem, Kubiak [28] proves the following

Theorem 5.11.2 [28] If $\sum_{i=1}^n \left(\frac{a_i}{b_i} + \frac{1}{b_i} \right) \leq 1$, then there is a generalized pinwheel schedule for pairs $(a_1, b_1), \dots, (a_n, b_n)$. The schedule can be found by the min-max algorithm with $B < 1$.

Proof: Let $(a_1, b_1), \dots, (a_n, b_n)$ be an instance of the generalized pinwheel scheduling problem such that $\sum_{i=1}^n \left(\frac{a_i}{b_i} + \frac{1}{b_i} \right) \leq 1$. Define $d_i = \frac{L(a_i + 1)}{b_i}$, for $i = 1, \dots, n$, where $L = \text{lcm}(b_1, \dots, b_n)$. Then, $\sum_{i=1}^n d_i \leq L$ and if $\sum_{i=1}^n d_i < L$, then define $d_{n+1} = L - \sum_{i=1}^n d_i$. Theorem 5.11.1 ensures that the min-max algorithm with $B < 1$ when applied to the instance including ratios $\frac{d_i}{L} = \frac{a_i + 1}{b_i}$, for $i = 1, \dots, n$, will deliver a sequence with at least $(a_i + 1) - 1 = c_i$ occurrences of i in any sub-sequence of b_i consecutive letters, and therefore, a generalized pinwheel schedule for $(a_1, b_1), \dots, (a_n, b_n)$. \square

Definition 5.11.2 Consider n independent, preemptive, periodic jobs $1, \dots, n$ with their request periods being T_1, \dots, T_n and their run-times being C_1, \dots, C_n . The execution of the k -th request of task i , which occurs at moment $(k-1)T_i$, must finish by moment kT_i when the next request for the task is being made. Missing a deadline is fatal to the system, therefore, the deadlines $T_i, 2T_i, \dots$ are considered NP-hard for job i . All numbers are positive integers and $C_i \leq T_i$ for $i = 1, \dots, n$. We need to find an infinite sequence $s = (s_1, s_2, \dots)$ on the alphabet $\{1, 2, \dots, n\}$ such that i occurs at exactly C_i times in each sub-sequence $(s_{(k-1)T_i+1}, \dots, s_{kT_i})$ for $k = 1, \dots$ and $i = 1, \dots, n$. We call s a periodic schedule for $i = 1, \dots, n$.

The proof of the following theorem directly follows from the proof of Theorem 5.6.1 (a).

Theorem 5.11.3 [31] Any solution to the problem F_{\max}^a with ratios $r_i = \frac{C_i}{T_i}$, $i = 1, \dots, n$, $\sum_{i=1}^n r_i \leq 1$, and $B < 1$ is a periodic schedule. □

CHAPTER 6

BICRITERION ANALYSIS

Investigations of different objectives simultaneously provide us quite interesting interpretations in JITSP. In this chapter, some of the F_{sum} and F_{max} objectives are analyzed simultaneously and are compared. None of the objectives F_{sum} and F_{max} is considered to be superior over the other, and therefore, both the objectives have their own significance. Sequencing mixed-model JIT facilities with more than one objective is considered in [47] for the first time. Moreover, simultaneous optimization of different objectives is presented in [35].

In the first section of this chapter, the B -bounded problems F_{sum} is summarized and a method to find all Pareto optimal solutions to the problems F_{sum} and F_{max}^a is given. The equivalency of the two problems F_{sum}^a and F_{sum}^s under certain condition is studied and an example without oneness property for the problems F_{sum}^a and F_{sum}^s is given.

The final section is devoted to the development of different production sequences that optimizes the different F_{max}^a and F_{sum} objectives simultaneously. Some open conjectures are summarized in consequences.

6.1 B-Bounded Problem F_{sum}

For a given threshold value $B > 0$, a feasible sequence $s = (s_1, \dots, s_D)$ for the problem F_{sum} is said to be B -bounded (or s has the B -ness property) if and only if $|x_{ik} - kr_i| \leq B$ for all i and k . For such given $B > 0$, the problem F_{sum} can be represented by a weighted incomplete bipartite graph $G_w(B) = (V_1 \cup V_2, E)$ where V_1, V_2 and the weights are the same as in Section 4.2.2 and $[k, (i, j)] \in E$ if and only if $k \in [E(i, j) \dots L(i, j)]$ (cf. [47], see also [26]). An appropriate choice of B can significantly reduce the number of calculated edge weights from the total number of required in the complete graph. If an order preserving perfect matching exists in $G_w(B)$, then the minimum weight perfect matching for the corresponding weighted bipartite graph could also be determined. This matching would have the property that it provides the min sum value F_{sum} such that the maximum deviation does not exceed B . Hence, F_{sum} is a function of B and the min-sum value is denoted by $F_{sum}(B)$. Hence, such an approach would provide a solution to the bicriterion sequencing problem with a value of $(F_{sum}(B), F_{sum}(s) \leq B)$ or the objectives $F_{sum}(s)$ and $F_{max}^a(s)$.

Definition 6.1.1 [47] *A bicriterion solution to the level production sequence problem is Pareto optimal if no other production sequence exists which has an objective value that is at least as good in both of the criteria and strictly better in at least one of them.*

Lemma 6.1.1 [47] The *min-sum assignment solution for the convex bipartite graph* $G_w(B)$ with $B \leq 1$, can be determined in $O(nD^2 \log D)$ time. \square

Lemma 6.1.2 [47] A *bicriterion JIT level sequence with a solution* $(F_{sum}(1), F_{max}^a(s) \leq B = 1)$ can be determined in $O(nD^2 \log D)$ time. \square

Instead of Lemma 6.1.2, it can be observed that $F_{sum^*} \leq F_{sum}(1)$ and $B^* \leq B = 1$; where F_{sum^*} and B^* are the optimal values for the problems F_{sum} and F_{max}^a respectively. Hence, the sequence may not be the optimal with respect to either measure.

Theorem 6.1.1 [47] The *Pareto optimal solution* $(F_{sum}(B^*), B^*)$ can be determined in $O(nD^2 \log D)$ time.

Proof [47]: The optimal value $B^* < 1$ for the problem F_{max}^a can be determined in $O(D \log D)$ time as in Section 5.5. By Lemma 6.1.2 with $B = B^*$, the perfect matching which provides the objective value $F_{sum}(B^*)$ could be constructed in at most $O(nD^2 \log D)$ time. Since B^* is optimal, the constructed production sequence must be Pareto optimal. As the two operations are disjoint, the overall complexity for determining this Pareto optimal solution is $O(nD^2 \log D)$ time. \square

Since, in practice, it is generally the case that $n \ll D$, the time complexity $O(nD^2 \log D)$ is fewer than that of the time complexity $O(D^3)$. Thus, the computational complexity for determining a Pareto optimal solution is lower than that for determining an optimal solution for the problem F_{sum} alone.

The sequence created in Theorem 6.1.1 is optimal for F_{max}^a and is of comparable quality to the problem F_{sum} . However, there are also some Pareto optimal sequences which are optimal for non of the objectives. All the Pareto optimal solutions for the problems F_{sum} and F_{max}^a with $B \leq 1$ can be constructed by the following algorithm.

Algorithm 6.1.1 [47] **Pareto**

Input: An instance of JIT production system.

Output: All Pareto optimal solutions for F_{sum} and F_{max}^a with $B \leq 1$.

begin,

1. Initialize $B = 1 - r_{max}$.
2. Generate the edge set for the convex bipartite graph corresponding to B . Denote this edge set by $E(B)$.
3. Determine if a perfect matching exists in this bipartite graph. If there is no perfect matching, then let $B = B + \frac{1}{D}$ and go to 2. If a perfect matching exists, then determine a minimum weight order preserving

perfect matching in $G_w(B)$. The corresponding production sequence $s = (s_1, \dots, s_D)$ will be Pareto optimal with objective values B^* and $F_{sum}(B^*)$.

Let $E = E(B)$, $\min F_{sum} = F_{sum}(B^*)$ and go to 4.

4. Let $B = B + \frac{1}{D}$. If $B > 1$, then stop all required Pareto optimal solutions with $B \leq 1$ have been determined. Otherwise, go to 5.
5. Generate the edge set, $E(B)$, for the convex bipartite graph corresponding to B . If $E(B) = E$, then go to 4. Otherwise set $E = E(B)$ and go to 6.
6. Determine a minimum weight order preserving perfect matching in $G_w(B)$. If $F_{sum}(B) < \min F_{sum}$, then this sequence will be Pareto optimal with objectives B and $F_{sum}(B)$, set $\min F_{sum} = F_{sum}(B)$ and go to 4. Otherwise the sequence is not Pareto optimal. Go to 4.

end.

Theorem 6.1.2 [47] *Algorithm 6.1.1 determines the production sequences $s = (s_1, \dots, s_D)$ for all Pareto optimal solutions (F_{sum}, B) with $B \leq 1$, in $O(nd_{\max} D^2 \log D)$ time. \square*

The existence of a 1-bounded solution optimal for the problem F_{sum} is not always the case. Corominas and Moreno [12] establish an instance with $n=10$, $D=100$, $d_1 = d_2 = 46$, $d_3 = d_4 = \dots = d_{10} = 1$ without oneness property for the problems F_{sum}^a and F_{sum}^s .

Let χ_1 be the set of all 1-bounded feasible solutions of the PRV-JIT sequencing problem.

Theorem 6.1.3 [12, 13] *Any optimal sequence $s = (s_1, \dots, s_D)$ for the problem F_{sum}^a , such that $s \in \chi_1$, is an optimal sequence for the problem F_{sum}^s , and hence, all optimal solutions for F_{sum}^s are 1-bounded. \square*

This theorem specifies the extent of the equivalence between both problems; it can be said that the problems F_{sum}^a and F_{sum}^s are χ_1 -equivalent. Given an instance, if it can be assured that both problems have optimal solutions belonging to set χ_1 then it suffices to solve one of the problems to have an optimal solution for the other. As it is pointed out by Kovalyov et al. [26], it is preferable to solve the problem F_{sum}^a .

6.2 Simultaneous Optimization of F_{\max} and F_{sum}

In this section, we summarize the analysis of simultaneous optimization of the different objectives concerning to the problems F_{sum} and F_{\max} . Introduce five Boolean variables denoted AM, SM, SM1 and AS to describe which criteria are simultaneously optimized. In these notations, A, S, M and M1 respectively stand for the objectives F_{sum}^a , F_{sum}^s , F_{\max}^a

and $F_{\max}^a \leq 1$. Define AM is true if and only if there is a sequence optimizing both the problems F_{sum}^a and F_{\max}^a simultaneously, SM is true if and only if a sequence optimizes both F_{sum}^s and F_{\max}^a simultaneously, AM1 is true if and only if an optimal F_{sum}^a sequence has maximum deviation lower than 1, SM1 is true if and only if an optimal F_{sum}^s sequence has maximum deviation lower than 1 and AS is true if and only if a sequence optimizes both the problem F_{sum}^a and F_{sum}^s simultaneously. The Boolean value True and False are respectively denoted by T and F.

Proposition 6.2.1 [35] $SM \wedge AM1 \Rightarrow AM$.

Proof [35]: Suppose that SM and AM1 are true for an instance of the PRV-MMJITSP. Let $S^* = (S_1, \dots, S_D)$ be a sequence optimizing the problems F_{sum}^s and F_{\max}^a simultaneously and $s^* = (s_1, \dots, s_D)$ a minimal sequence of F_{sum}^a with maximum deviations less than 1. The solution S^* is optimal for F_{sum}^s . Therefore,

$$\sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk}^s \mathcal{Y}_{ijk}^* \geq \sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk}^a \mathcal{Y}_{ijk}^*,$$

where C^a and C^s are the cost matrices calculated for the problems F_{sum}^a and F_{sum}^s respectively and y_{ijk}^* and Y_{ijk}^* are the assignment variables corresponding to the sequences s^* and S^* respectively. Maximum deviation for S^* is equal to the optimum F_{\max}^a and is clearly lower than 1 by Theorem 5.4.2. Therefore, from Theorem 6.1.3, we have

$$\sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk}^s \mathcal{Y}_{ijk}^* = \sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk}^a \mathcal{Y}_{ijk}^*.$$

Likewise, since the solution s^* has maximum deviation lower than 1, we have

$$\sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk}^s \mathcal{Y}_{ijk}^* = \sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk}^a \mathcal{Y}_{ijk}^*.$$

Therefore, we obtain the inequality,

$$\sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk}^a \mathcal{Y}_{ijk}^* \geq \sum_{k=1}^D \sum_{i=1}^n \sum_{j=1}^{d_i} C_{ijk}^a \mathcal{Y}_{ijk}^*.$$

Since the sequence s^* is minimal for F_{sum}^a , S^* is also minimal for F_{sum}^a . Since S^* is optimal for both F_{\max}^a and F_{sum}^a , AM is true. \square

The set IX, $X \in \{AM, SM, AM1, SM1, AS\}$ represents the set of all instances such that X is true. Since $AM \Rightarrow SM \wedge AM1$ and $SM \wedge AM1 \Rightarrow AM$, the set IAM is equal to the intersection of the sets IAM1 and ISM. Figure 6.1 (cf. [35]) is an illustration of the possible values of the quintuplet (AM, SM, AM1, SM1, AS) and for simplicity the set IAM is not drawn there. The numbers in the sets represents the seven examples of Figure 6.2 (cf. [35]). Figure 6.2 represents the examples for all possible values of quintuplet (AM, SM, AM1, SM1, AS) except the quintuplet (F, F, F, F, F). The compact notation of the demand vector $d = (d_a^p, d_b^q)$ represents an instance with $n = p + q$ part types whose demands of the p first part types are equal to d_a and of the q following are equal to d_b .

For all instances, if one of the Boolean variables AM to AS is true, the corresponding sequence that proves it is presented. This sequence is also presented in a compact form.

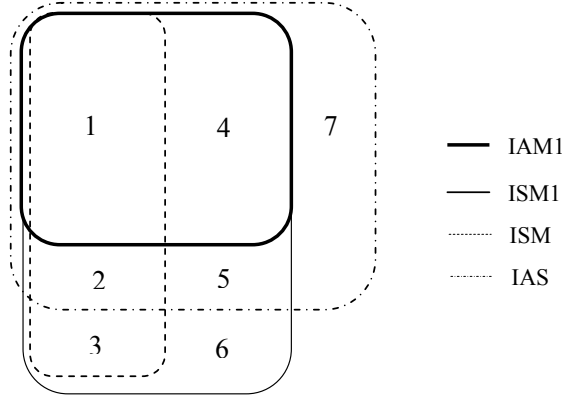


Figure 6.1: Representation of the Sets of Instances Corresponding to the Values of the Quintuplet (AM, SM, AM1, SM1, AS)

If part types i and i' are such that $d_i = d_{i'}$, we can exchange the parts (i, j) and (i', j) without modifying the sum or the maximum of deviations. Since for each example, the part types have at most two different values d_a and d_b , we denote $a_j, j = 1, \dots, d_a$ all parts (i, j) such that $d_i = d_a$ and $b_j, j = 1, \dots, d_b$ all parts (i, j) such that $d_i = d_b$.

Existence of Sequence Optimizing					Instance	Proof
AM F_{sum}^a and F_{max}^a	SM F_{sum}^s and F_{max}^a	AM1 F_{sum}^a with $F_{max}^a \leq 1$	SM1 F_{sum}^s with $F_{max}^a \leq 1$	AS F_{sum}^a and F_{sum}^s		
1	T	T	T	T	$d = (1)$	$s = (a_1)$ is optimal for all problems
2	F	T	F	T	$d = (1^9, 7^3)$	$s = (b_1^3, b_2^3, a_1^3, b_3^3, a_1, b_4^3, a_1^2, b_5^3, a_1^3, b_6^3, b_7^3)$ is optimal for F_{sum}^a and F_{sum}^s . $s' = (b_1^3, a_1, b_2^3, a_1^2, b_3^3, a_1, b_4^3, a_1^2, b_5^3, a_1^2, b_6^3, a_1, b_7^3)$ is optimal for F_{sum}^s and F_{max}^a
3	F	T	F	T	$d = (1^{12}, 10^2)$	$s = (b_1^2, a_1, b_2^2, a_1, b_3^2, a_1^2, b_4^2, a_1, b_5^2, a_1^2, b_6^2, a_1, b_7^2, a_1^2, b_8^2, a_1, b_9^2, a_1, b_{10}^2)$ is optimal for F_{sum}^s and F_{max}^a
4	F	F	T	T	$d = (1^2, 4^2)$	$s = (b_1^2, b_2^2, a_1^2, b_3^2, b_4^2)$ is optimal for F_{sum}^a with $F_{max}^a \leq 1$
5	F	F	F	T	$d = (1^{10}, 6^5)$	$s = (b_1^5, a_1, b_2^5, a_1^3, b_3^5, a_1^2, b_4^5, a_1^3, b_5^5, a_1, b_6^5)$ is optimal for F_{sum}^s with $F_{max}^a \leq 1$ and $s' = (b_1^5, b_2^5, a_1^3, b_3^5, a_1^4, b_4^5, a_1^3, b_5^5, b_6^5)$ is optimal for F_{sum}^a and F_{sum}^s
6	F	F	F	T	$d = (1^9, 6^4)$	$s = (b_1^4, a_1, b_2^4, a_1^2, b_3^4, a_1^3, b_4^4, a_1^2, b_5^4, a_1, b_6^4)$ is optimal for F_{sum}^s with $F_{max}^a \leq 1$
7	F	F	F	T	$d = (1^7, 6^4)$	$s = (b_1^4, b_2^4, a_1^2, b_3^4, a_1^3, b_4^4, a_1^2, b_5^4, b_6^4)$ is optimal for F_{sum}^a and F_{sum}^s

Figure 6.2: Example of Instances with Sequences Optimizing Several Criteria

Now, to obtain the quintuplet (T, T, T, T, T), we need a sequence $s = (s_1, \dots, s_D)$ with the desired characteristics. Theorem 4.2.6 gives the optimality of this sequence for the problem F_{sum}^a and F_{sum}^s . When AS is not already implied by AM or AM1, we can show that AS is true by giving the perfect matching representing the common optimal solution to the problems F_{sum}^a and F_{sum}^s and by proposing associated vertex weights for each complete bipartite graph as in Section 4.2.3.

To prove that optimal F_{sum} sequence is B -bounded (i. e. that the corresponding variable AM to AM1 is true), we can either compute the maximum deviation of the sequence and verify that it is indeed lower than B , or construct the graph corresponding to the B -bounded problem and verify that all the edges of the sequence belong to this restricted graph.

To prove that the Boolean variables AM to SM1 are false, we have to compare the optimal F_{sum} objective values of the bounded and unbounded problems. That is we compute optimal perfect matching and vertex weights for the unbounded and bounded problems. If the values of the two matching are different, then the corresponding variable is false.

The Boolean variable AS is false when the problems F_{sum}^a and F_{sum}^s have no common optimal solution, i.e. when the sets of all the optimal solutions of the problems F_{sum}^a and F_{sum}^s are disjoint.

The statement “For any instance there is a minimal F_{sum}^a sequence that is minimal for the problem F_{max}^a ” conjectured in [29] is refuted with providing a counter example (4) of Figure 6.2 by *Lebacque et al.* [35].

The existence of an instance with SM1 true and AM1 false proves that the converse of Theorem 6.1.3 is not true (see [35]). It also refutes the following conjecture.

Conjecture 6.2.1 [47] *For an instance, there is a sequence that is optimal for F_{sum}^a with maximum deviation lower than 1.* □

Instead of their computational testing, *Lebacque et al.* [35] conjectured the followings:

Conjecture 6.1.2 *For $n = 3$ there are instances such that the problem F_{sum}^a has no optimal solution with maximum deviation lower than 1.* □

Conjecture 6.1.3 *For any instance, if there is no optimal sequence for F_{sum}^s with maximum deviation lower than 1, then there is a sequence that optimizes the problems F_{sum}^s and F_{sum}^a simultaneously. For any instance, if there is no sequence optimizing F_{sum}^s and F_{sum}^a simultaneously, then there is a sequence optimizing F_{sum}^s with maximum deviation lower than 1.* □

CHAPTER 7

CONCLUSIONS

Determining the optimal balanced sequence in MMJITSP is considered which involves minimizing the deviation of production from demand. The mathematical formulations of the MMJITSP in sequencing theory are in the more challenging nonlinear integer programming form. It is because the linear case of the integer programming has already been *NP*-hard (see [43]). In this research, mathematical formulations of MMJITSP remain as a backbone. The main work of this research is to explore and summarize the different types of solution procedures for MMJITSP up to now. The study shows that the MMJITSP has real world existing applications as well as interesting mathematical features of theoretical value.

The min-sum PRV problems with any nonnegative symmetric convex discrepancy functions with minimum 0 at 0 are pseudo-polynomially solvable by reducing it into the assignment problem (see Section 4.2.2). Similarly, the bottleneck PRV problem has been solved by reducing it into bottleneck assignment problem (see Section 5.9).

In this study, bottleneck Just-in-Time single-level sequencing problems have been discussed in detail. Several algebraic structural properties of such problems have been explored. By reducing it into release dates, due dates decision problems and thereby shifting into perfect matching problems in bipartite graphs the optimal sequences for the bottleneck problems F_{\max}^a and F_{\max}^s are calculated in exact pseudo-polynomial time complexity $O(D \log D)$ (cf. Theorem 5.5.1 and Theorem 5.5.2) under the binary search (see Section 5.5) in which the maximum deviations are strictly less than one (cf. Theorem 5.4.2 and Theorem 5.4.3). The lower bound $1 - r_{\max}$ of the problem F_{\max}^a has been shown to be tight (cf. Example 5.4.1). This approach can also be applied to other nonnegative symmetric convex objective functions. An example (cf. Example 5.5.1) for an instance of F_{\max}^a is presented obtaining an optimal sequence.

Obtaining common solutions to different objective functions would significantly reduce the computational complexity and thus a comparative study is summarized in Section 6.2 for optimizing different objectives simultaneously. However, the 1-bounded sequence obtained in incomplete bipartite graphs does not guarantee an optimal sequence for the problem F_{sum} . But the question of determining a minimum B such that the optimal sequence for F_{sum} is B-bounded, remains open and will be a topic for further investigation.

Although most of PRV-MMJITSPs are pseudo-polynomially solvable depending upon the input size of the demands, their exact complexity status still remains open. The problem F_{\max}^a is shown to be *Co-NP* (cf. Theorem 5.10.1) but it is also open whether the problem is *Co-NP*-complete or polynomially solvable. As the input size of the general PRVP is $O(D \log D)$ and there are nD variables and $O(nD)$ constraints, existence of a polynomial time algorithm is far from trivial. Thus in spite of its apparent simplicity, the status of PRV-MMJITSP is not completely understood yet.

Throughout this research, we find that both the min-sum and min-max ORVPs even with two-levels are strongly NP -hard (cf. [29, 34]). Therefore, a heuristic procedure “Toyota’s Goal Chasing Method” (see Section 4.1) for the min-sum ORVP and an efficient, implicit enumeration dynamic programming procedure for the general ORVP which are fruitful for such NP -hard problems for optimization have been presented. However, if the products require either approximately the same number and mix of parts or the pegging assumptions are imposed then the general ORVPs are efficiently solvable by reducing these into corresponding weighted PRVPs. Thus, searching for the efficient algorithms or good approximation algorithms for ORVP might be an interesting research topic for further investigation.

In our study, we find that the optimal sequences both for the problems F_{sum} and F_{max} are cyclic (cf. Theorem 4.2.8 and Theorem 5.6.1). We give a modified proof of Theorem 5.6.1 from *Kubiak* [31] which is originally proved by *Steiner and Yeomans* [50]. The importance of the existence of the cyclic sequences is to reduce the computational complexity for obtaining the production sequences. However, the conjecture whether cyclic sequences to ORVP are optimal is still open and it may be one of the interesting foremost topic for further study.

Our study also focuses on the elegant algebraic concept of balanced words relating to the problem F_{max} (see Section 5.7). The 1-balanced words cannot be obtained for most rates, but the set of all 3-balanced words consists of optimal sequence for the problem F_{max}^a (cf. Theorem 5.7.3). Minimality of this set is unknown and enumeration of this set for optimality is expensive. It is still unsolved whether the set of all 2-balanced words is sufficient for an optimal sequence for the problem F_{max}^a . Thus, a conjectured topography (cf. Figure 5.8) waiting for a conclusion has been presented. Among the applications of optimal objective value of F_{max}^a in allocation problems, in this study, it has only been applied to pinwheel (cf. Theorem 5.11.2) and periodic scheduling problems (cf. Theorem 5.11.3).

The exact complexity status of the PRV-JITSP, relation between optimal sequences of the problems F_{max}^a and F_{max}^s and to determine the actual status of the conjectured topography of Figure 5.8 are the foremost subjects for further investigation and research to pave some new results in this field.

REFERENCES

1. *H. Aigbdo, Some Structural Properties for the JIT Level Schedule Problem*, Production Planning and Control, 11, 4 (2000), 357-362.
2. *T. M. Apostol, Mathematical Analysis* (Addison-Wesley Publishing Company, Inc., 1974).
3. *M. Balinski and N. Shahidi, A Simple Approach to the Product Rate Variation Problem (PRVP) via Axiomatics*, Operations Research Letters, 22 (1998), 129-135.
4. *J. Bautista, R. Companys and A. Corominas, A Note on the Relation Between the Product Rate Variation (PRV) Problem and the Apportionment Problem*, The Journal of the Operational Research Society, 47, 11 (1996), 1410-1414.
5. *J. Bautista, R. Companys and A. Corominas, Modeling and Solving the Production Rate Variation Problem*, TOP, 5, 2 (1997), 221-239.
6. *J. Blazewicz, K. H. Ecker, E. Pesch, G. Schmidt and J. Weglarz, Scheduling Computer and Manufacturing Processes* (Springer- Verlag, Berlin, 1996).
7. *N. Brauner and Y. Crama, The Maximum Deviation Just-in-Time Scheduling Problem*, Discrete Applied Mathematics, 134 (2004), 25-50.
8. *N. Brauner, V. Jost and W. Kubiak, On Symmetric Fraenkel's and Small Deviations Conjectures*, Les Cahiers du Laboratoire Leibniz-IMAG, 54 (2004), Grenoble, France.
9. *P. Brucker, Scheduling Algorithms*, 2nd Edition (Springer-Verlag, 1995).
10. *D. M. Burton, Elementary Number Theory*, 2nd Edition (Universal Book Stall, New Delhi, Reprint, 2003).
11. *T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms*, 2nd Edition (Prentice-Hall of India Pvt. Ltd., 2007).
12. *A. Corominas and N. Moreno, About Relations Between the Optimal Solutions for Different Types of min-sum Production Rate Variation Problem (PRVP)*, 27 Congresso de Estadística e Investigación operativa Lleida, 8-11 de abril de (2003).
13. *A. Corominas and N. Moreno, On the Relations between Optimal Solutions for Different Types of min-sum Balanced JIT Optimization Problems*, INFOR, 41, 4, (2003), 333-339.
14. *T. N. Dhamala, Balancing Just-in-Time Sequences in Mixed-Model Production Systems*, The Nepali Mathematical Sciences Report, 25, 2 (2005), 17-27.

15. *T. N. Dhamala, Just-in-Time Sequencing Algorithms for Mixed-Model Production Systems*, The Nepali Mathematical Sciences Report, 24, 1 (2005), 25-34.
16. *T. N. Dhamala, Shop Scheduling Solution-Spaces with Algebraic Characterizations* (Ph. D. Thesis Otto-von-Guericke University, Magdeburg, Germany, 2002).
17. *T. N. Dhamala and S. R. Khadka, Absolute Maximum Deviation Just-in-Time Sequencing Problem for Mixed-Model Production Systems*, Proceedings of the Seminar on Mathematical Sciences and Applications (Central Department of Mathematics, Tribhuvan University, Nepal, Sep. 20-21, 2006), 25-32.
18. *T. N. Dhamala, S. R. Khadka and M. H. Lee, Bottleneck Product Rate Variation Problem for Mixed-Model Just-in-Time Production System*, Submitted to International Journal of Operations Research, 2008.
19. *T. N. Dhamala, S. R. Khadka and M. H. Lee, On Sequencing Approaches for Mixed-Model Just-in-Time Production Systems*, Submitted to Asia-Pacific Journal of Operational Research, 2008.
20. *T. N. Dhamala and W. Kubiak, A Brief Survey of Just-in-Time Sequencing for Mixed-Model Systems*, International Journal of Operations Research, 2, 2 (2005), 38-47.
21. *F. Glover, Maximum Matching in a Convex Bipartite Graph*, Naval Research Logistics Quartile, 4, 3 (1967), 313-316.
22. *R. E. Graham, E. L. Lawer, J. K. Lenstra and A. H. G. Rinnoy Kan, Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey*, Annals of Discrete Mathematics, 5 (1979), 287-326.
23. *G. Hadley, Nonlinear and Dynamic Programming* (Addison-Wesley, 1972).
24. *R. R. Inman and R. L. Bulfin, Sequencing JIT Mixed-Model Assembly Lines*, Management Science, 37, 7 (1991), 901-904.
25. *S. Kotani, Sequencing Algorithms for a Mixed-Model Assembly Line*, Toyota Technical Engineering, 33 (1983), 31-38.
26. *M. Y. Kovalyov, W. Kubik and J. S. Yeomans, A Computational Analysis of Balanced JIT Optimization Algorithms*, Journal of Information Systems and Operational Research-INFOR, 39, 3 (Aug 2001), 299-315.
27. *W. Kubiak, Cyclic Just-in-Time Sequence are Optimal*, Journal of Global Optimization, 27 (2003), 333-347.
28. *W. Kubiak, Fair Sequences* (Hand Book of Scheduling Chapman and Hall/CRC Computer and Information Science Series, 2004).

29. *W. Kubiak, Minimizing Variation of Product Rates in Just-in-Time Systems: A Survey*, European Journal of Operational Research, 66 (1993), 259-271.
30. *W. Kubiak, On Small Deviations Conjecture*, Bulletin of the Polish Academy of Sciences, 51 (2003), 189-203.
31. *W. Kubiak, Solution of the Liu-Layland Problem via Bottleneck JIT Sequencing*, Journal of Scheduling, 8 (2005), 295-302.
32. *W. Kubiak and S. P. Sethi, A Note on Level Schedules for Mixed-Model Assembly Lines in Just-in-Time Production Systems*, Management Science, 37, 1 (1991),121-122.
33. *W. Kubiak and S. P. Sethi, Optimal Just- in-Time Schedules for Flexible Transfer Lines*, The International Journal of Flexible Manufacturing Systems, 6 (1994), 137-154.
34. *W. Kubiak, G. Steiner and J. S. Yeomans, Optimal Level Schedules for Mixed-Model, Multi-Level Just-in-Time Assembly Systems*, Annals of Operations Research, 69 (1997) , 241-259.
35. *V. Lebacque, V. Jost and N. Brauner, Simultaneous Optimization of Classical Objectives in JIT Scheduling*, European Journal of Operations Research,182 (2007), 29-39.
36. *Mathematical Models*, Science Education Resources Center, Carleton College ([http:// serc.Carleton.edu](http://serc.Carleton.edu), 2008).
37. *J. Miltenburg, Level Schedules for Mixed-Model Assembly Lines in Just-in-Time Production Systems*, Management Science, 35, 2 (1989), 192- 207.
38. *J. Miltenburg and T. Goldstein, Developing Production Schedules which Balance Part Usage and Smooth Production Loads for Just-in-Time Production Systems*, Naval Research Logistics, 38 (1991), 893-910.
39. *J. Miltenburg and G. Sinnamon, Scheduling Mixed-Model Multi-Level Just-in-Time Production Systems*, International Journal of Production Research, 27, 9 (1989), 1487-1509.
40. *J. Miltenburg, G. Steiner and J. S. Yeomans, A Dynamic Programming Algorithm For Scheduling Mixed-Model, Just-in-Time Production Systems*, Mathematical and Computer Modeling,13, 3 (1990), 57-66.
41. *Y. Monden, Toyota Production Systems (Industrial Engineering and Management Press, Norcross, GA, 1983).*
42. *N. Moreno and A. Corominas, Solving the min-max Product Rate Variation Problem (PRVP) as a Bottleneck Assignment Problem*, Computers and Operations Research, 33 (2006), 928-939.

43. *C. H. Papadimitriou and K. Steiglitz*, Combinatorial Optimization: Algorithms and Complexity (Prentice-Hall of India Pvt. Ltd., 2003).
44. *M. Pinedo and X. Chao*, Operations Scheduling with Application in Manufacturing and Services (Irwin, McGraw-Hill, 1999).
45. *K. H. Rosen*, Discrete Mathematics and its Applications (TATA McGRAW Hill Edition, New Delhi, 2003).
46. *S. J. Russel and P. Norving*, Artificial Intelligence: A Modern Approach (Prentice Hall, 1995).
47. *G. Steiner and J. S. Yeomans*, *A Bicriterion Objective for Levelling the Schedule of a Mixed-Model, JIT Assembly Process*, Mathematical and Computer Modelling, 20, 2 (1994), 123-134.
48. *G. Steiner and J. S. Yeomans*, *A Linear Time Algorithm for Maximum Matching in Convex, Bipartite Graphs*, Computers and Mathematics with Applications, 31, 12 (1996), 91-96.
49. *G. Steiner and J. S. Yeomans*, *Level Schedules for Mixed-Model Just-in-Time Processes*, Management Science, 39, 6 (1993), 728-735.
50. *G. Steiner and J. S. Yeomans*, *Optimal Level Schedules in Mixed-Model , Multi-Level JIT Assembly Systems with Pegging*, European Journal of Operational Research, 95 (1996) , 38-52.
51. *R. Tijdeman*, *Exact Covers of Balanced Sequences and Fraenkel's Conjecture*, Algebraic Number Theory and Diophantine Analysis (F. Halter-Koch and R.F. Tichy, Walterde Gruyter, Berlin, New York, 2000), 467-489.
52. *R. Tijdeman*, *Fraenkjel's Conjecture for Six-Sequences*, Discrete Mathematics, 222 (2000), 223-234.
53. *R. Tijdeman*, *The Chairman Assignment Problem*, Discrete Mathematics, 32 (1980), 323-330.
54. *L. Vuillon*, *Balanced Words*, Rapports de Recherché 2003-006. LIAFA CNRS, University Paris, 7 (2003).
55. Wikipedia, The Free Encyclopedia (<http://en.wikipedia.org>, 2008).
56. [Www.assignmentproblem.com](http://www.assignmentproblem.com), 2008.
57. *M. Yavuz and E. Akcali*, *Production Smoothing in Just-in-Time Manufacturing Systems: A Review of the Model and Solution Approaches*, International Journal of Production Research, 45, 16 (2007), 3579-3597.

MATHEMATICAL NOTATIONS

$ A $	Cardinality of a set A
$[a, b]$	Closed interval
C_{ijk}	The cost of assigning (i, j) to the k^{th} period
d_{il}	Demand for output i at level l , $i = 1, \dots, n_i$; $l = 1, \dots, L$ of ORVP
gcd	Greatest common divisor
r_{il}	Demand ratio for output i at level l , $i = 1, \dots, n_i$; $l = 1, \dots, L$ of ORVP
r_i	Demand ratio for product i to the total demand of PRVP
\emptyset	Empty (null) set
l	Level number, $l = 1, \dots, L$ of ORVP
lcm	Least common multiple
log	Logarithm to the base 2
r_{\max}	Maximum product demand ratio, $r_{\max} = \max_{i=1}^n \{r_i\}$
n_l	Number of different outputs at level l of ORVP
t_{ilp}	Number of units of output i at level l require to produce one unit of product p , $i = 1, \dots, n_i$; $l = 1, \dots, L$; $p = 1, \dots, n_1$
(a, b)	Open interval
x_{ilk}	Quantity of output i at level l produced during stages 1 through k , $k = 1, \dots, D_1$ of ORVP
x_{ik}	Quantity of product i producing during stages 1 through k of PRVP
$[a..b]$	Set of all integers between a and b including both
(i, j)	The j^{th} copy of product i , $i = 1, \dots, n$; $j = 1, \dots, d_i$
L	The number of different levels of production of ORVP
B	Threshold value for the objective function of PRVP
D_l	Total demand for production at level l , $l = 1, \dots, L$ of ORVP
D	Total demand of production of PRVP
\tilde{F}_{\max}^w	A weighted max-absolute ORVP
$\tilde{F}_{\max}^w(\tilde{X})$	A weighted max-absolute ORVP objective function
$\tilde{F}_{\max}^{wa \text{ peg}}$	A weighted max-absolute ORVP with pegging assumption
F_{\max}	Bottleneck PRVP
$F_{\max}(s)$	Bottleneck PRVP objective function
F_{\max}^a	Max-absolute PRVP
F_{\max}^s	Max-squared PRVP
F_{sum}	Min-sum PRVP
F_{sum}^a	Min-sum-absolute PRVP
F_{sum}^s	Min-sum-squared PRVP
$F_{\text{sum}}(s)$	Sum-deviation PRVP objective function
Ψ_{ijl}	Excess cost of having j -copies of product i produced by period over l

y_{lk}	Total production at level l during stages 1 through k
w_{il}	Weighting factor for the i^{th} part at level l of ORVP
w_i	Weighting factor for the i^{th} product of PRVP
$s = (s_1, \dots, s_n)$	A finite sequence
$s = (s_1, s_2, \dots)$	An infinite sequence
f_{il}	Discrepancy function between actual and ideal productions of output i at level l of ORVP
f_i	Discrepancy function between actual and ideal productions of product i of PRVP
$\ x\ $	Norm of x
Z_{ij}	The completion time of (i, j)
$E(i, j)$	The earliest feasible producing time for (i, j) in the final production sequence
k_{ij}	The ideal corner of (i, j)
Z_{ij}^*	The ideal position of (i, j)
$L(i, j)$	The latest feasible producing time for (i, j) in the final production sequence
w_{\max}	Weighting factor of largest value, $w_{\max} = \max_{i=1}^n \{w_i\}$
w_{\min}	Weighting factor of smallest value, $w_{\min} = \min_{i=1}^n \{w_i\}$
$Co-NP$	Class of complements of NP decision problems
$Co-NP$ -complete	Class of complements of NP -complete decision problems
\mathfrak{C}	Set of 3-tuples with an element $((i, j), k)$ for $i = 1, \dots, n$; $j = 1, \dots, d_i$; $k = 1, \dots, D$.
\mathbb{N}	Set of natural numbers
NP	Class of nondeterministic polynomial time solvable decision problems
NP -complete	Class of NP -complete decision problems
NP -hard	Class of NP -hard optimization problems
P	Class of polynomially solvable decision problems
\mathbb{Q}	Set of rational numbers
\mathbb{R}	Set of real numbers
\mathbb{R}^+	Set of nonnegative real numbers
\mathbb{R}^n	Set of n -tuples with entries in \mathbb{R}
\mathbb{W}	Set of whole numbers
\mathbb{W}^n	Set of n -tuples with entries in \mathbb{W}
\tilde{X}	Cumulative production matrix $(x_{plk})_{n_1 \times D_1}$ of ORVP
$\tilde{\chi}$	Set of all feasible solutions to ORVPP
χ	Set of all feasible solutions to PRVPP
\mathbb{Z}	Set of integers
$\lfloor x \rfloor$	The floor of the real number x
$\lceil x \rceil$	The ceiling of the real number x

$[x]$	The rounding of the real number x
\Rightarrow	Implies
\wedge	Conjunction (and)
\Leftrightarrow	If and only if
\forall	For all
\exists	There exist(s)
\in	Belong(s) to
\neq	Not equal to
$>$	Is greater than
$<$	Is less than
\geq	Is greater than or equal to
\leq	Is less than or equal to
\subseteq	Is contained in
\cup	Union of sets
\cap	Intersection of sets
$\sum_{i=1}^n$	Sum to n terms, one for each positive integer from 1 to n
$\prod_{i=1}^n$	Product of n terms, one for each positive integer from 1 to n
\ll	Is very much smaller
\square	The end of proof or the end of solution