Heuristic Approaches for the Lot Streaming Problem in Multi-Product Flow Shops

by

Emin Rodoslu

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science
in
Industrial Engineering

Koc University

26.07.2013

Koc University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Emin Rodoslu

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Prof. Dr. Ceyda Oguz (Advisor)

_____

Assoc. Prof. Dr. Emre Alper Yildirim

_____

Asst. Prof. Dr. Rahime Sancar Edis

Date: 26.07.2013

ABSTRACT

In thesis research, we consider the multi-product lot streaming (MPLS) problem with equal and consistent sublots in multi-machine flow shops (MMFS) with objective of minimizing the makespan. We firstly introduce various types of scheduling problems and provide detailed background information and literature review on lot streaming problems with equal and consistent sublots. Then we develop two heuristic procedures for equal and consistent sublot sized MPLS problems respectively.

First heuristic approach that we develop for MPLS problem with equal sublots in MMFS (heuristic RO) is a constructive procedure, which has many distinguishing characteristics. Its fast and easy construction method of initial lot sequence lets tabu search algorithm start with a better initial solution in contrast with random initial solutions. Moreover, we utilize the concept of "Interior Lots" in order to restrict the insertion of a given lot into first position. We also provide a proof of the claim, which supports the use of "Interior Lot" concept in both of the heuristics. Second heuristic approach (heuristic RO-C) deals with the MPLS problem with consistent sublots in MMFS. In compliance with the different characteristics of the problem we develop an additional tabu search algorithm in order to generate better initial sublot size matrix.

Finally we present comparative results of experimental studies for both heuristics. We show that solution qualities of both heuristics that we develop are better than or equal to those obtained by the heuristic and exact methods that we choose to compare with.

Proposed heuristics have considerable contributions to MPLS literature due to their unique ordering and tie breaking rules for sorting bottleneck dominant and reversely bottleneck dominant lots, utilization of interior lot concept in several steps of heuristics and solution quality with respect to similar studies from the literature.

Keywords: Lot Streaming, Equal Sublot, Consistent Sublot, Bottleneck Dominance, Interior Lots, Tabu Search, Run-in Time, Run-out Time

ÖZET

Bu proje kapsamında, çok makineli akış tipi üretim sistemleri için çok ürünlü kafile bölme ve kaydırma problemlerine (ÇÜ-KBK problemleri) sezgisel çözüm yaklaşımları geliştirilmiştir. Amaç fonksiyonu olarak tüm ürünlerin bitiş zamanını en küçükleme fonksiyonu kullanılmıştır. İlk olarak, literatürde ÇÜ-KBK problemlerinin eşit ve tutarlı alt kafile problemleri üzerine yazılmış makaleler incelenmiş ve ÇÜ-KBK problemlerinin daha anlaşılır olması için gerekli arka plan bilgileri sağlanmıştır. Daha sonra, proje kapsamında eşit ve tutarlı alt kafileli ÇÜ-KBK problemlerine çözüm yaklaşımı getiren iki sezgisel (Sezgisel RO ve Sezgisel RO - Tutarlı) detaylı olarak anlatılmıştır. Aynı problem türlerine çözüm getiren literatürden seçilmiş farklı çözüm yaklaşımlarıyla karşılaştırmalı sonuç analizlerine yer verilmiştir. Sonuç analizleri göstermektedir ki, proje kapsamında geliştirilen sezgisel yaklaşımlar karşılaştırıldıkları çözüm yöntemlerine yakın ya da daha iyi sonuçlar verebilmektedirler. Ayrıca, sezgisel yaklaşımların oluşturulmasında kullanılan "İç kafile" kavramını destekleyici olarak sunulan bir savın ispatı da araştırma içerisinde yer almaktadır.

Geliştirilen sezgiseller, darboğaz ve tersten darboğaz egemen kafileleri sıralamada ve eşitlik bozmadaki farklı yaklaşımları, iç kafile kavramını sezgisellerin ve tabu arama metodunun farklı yerlerinde kullanmaları ve karşılaştırıldıkları çözüm yöntemlerine yakın ya da daha iyi sonuçlar vermeleriyle ÇÜ-KBK literatürüne katkı sağlamaktadırlar.

Anahtar Kelimeler: Kafile Bölme ve Kaydırma, Eşit Alt Kafile, Tutarlı Alt Kafile, Darboğaz Egemenliği, İç Kafile, Tabu Arama, Giriş Zamanı, Çıkış Zamanı

ACKNOWLEDGEMENTS

I would like to express my special thanks of gratitude to my thesis advisor Prof. Ceyda Oğuz, whose contribution in stimulating suggestions and encouragement helped me to complete my thesis research. She has been very understanding, helpful and guiding throughout the research. Without her guidance and persistent help this thesis would not have been possible. Moreover, I would also like to express my sincere gratitude to Asst. Prof. Rahime Sancar Edis for her continuous assistance in every aspect of this research.

Secondly I would like to thank my beloved family and my cousins who helped me to gather necessary strength and motivation whenever needed. I would also like to thank my closest friends for their intimacy and support and my office mates for creating positive atmosphere in the faculty that led me to finalize this research within the limited time frame.

Finally, I would like to thank TUBITAK for their financial support, since this study was supported partially by TUBITAK (110M636).

TABLE OF CONTENTS

List of Tables viii

List of Figures ix

Nomenclature x

Chapter 1: Introduction 1

1.1 Multi Stage Scheduling Problems 1

1.2 Objectives of Scheduling Problems 2

1.3 The Concept of Lot Streaming 3

1.4 Advantages of Lot Streaming 5

1.5 Contributions and Summary of Remaining Chapters 6

Chapter 2: Literature Review 8

2.1 Overview 8

2.2. Lot Streaming Problems with Single-Product – 2-3 Machines 10

2.3. Lot Streaming Problems with Single-Product – Multi-machines 11

2.4. Lot Streaming Problems with Multi-product – 2-3 Machines 12

2.5. Lot Streaming Problems with Multi-product – Multi-machines 14

2.6. Lot Streaming Problems with Consistent Sublots 16

2.7. Other Types of Scheduling Problems Related to Lot Streaming 19

Chapter 3: Heuristic Approach for the MPLS Problem with Equal Sublots in MMFS 21

3.1 Introduction 21

3.2. Terminology 21

3.2.1. Equal Sublots 21

3.2.2. Consistent and Variable Sublots 23

3.2.3. Bottleneck Dominance Theorem and BMI Heuristic by Kalir and Sarin (2001) 24

3.2.4. Run-in and Run-out Times for Lot *j* 25
3.2.5. Intermingled versus Non-Intermingled Schedules 26
3.2.6. Sublot Transfer Times/Costs and Setup Times 26
3.3. Problem Definition 26
3.4. General Assumptions 27
3.5. Heuristic Algorithm for the LS Problem with Multi-Product Flow Shops with Equal Sublots (Heuristic RO) 27
3.6. Tabu Search Algorithm for Generating Better Sequences 33
3.6.1. Steps of Tabu Search Algorithm 33
3.6.2. Time Complexity of RO and TSA 36
3.7. Significance of Using an Interior Lot in the Heuristic Method 36
Chapter 4: Heuristic Approach for MPLS Problem with Consistent Sublots in MMFS 43
4.1. Introduction 43
4.2. Problem Definition of LS with Consistent Sublots 43
4.3. Heuristic Approach for MPLS problems with Consistent Sublots in MMFS (Heuristic RO - Consistent) 44
Chapter 5: Computational Experiments 49
5.1. Introduction 49
5.2. Random Start vs. Heuristic RO Start 49
5.3. Computational Results on LS Problems with Equal Sublot Size 53
5.4. Computational Results on LS Problems with Consistent Sublot Size 58
5.4.1. Revision of IP Model of Feldmann and Biskup (2008) 58
5.4.2. Computational Experiments for LS Problems with Consistent Sublots 60
Chapter 6: Conclusions 64
Bibliography 66

LIST OF TABLES

Table 1 Different Components of Lot Streaming Problem.....22

LIST OF FIGURES

Figure 1 Processing without lot streaming concept 4
Figure 2 Processing with lot streaming concept 4
Figure 3 Characteristics of Lot Streaming Problems in the Literature 9
Figure 4 Lot streaming with consistent sublot sizes of 32, 16, and 16 items 23
Figure 5 Lot streaming with variable sublot sizes of 32, 16, and 16 items in machine 1 and 16, 32, and 16 items in machine 2.....23
Figure 6 Run-in time for lot j when $p_1 > p_2$ 25
Figure 7 Before the Insertion of Lot I^* into First Position 38
Figure 8 After the Insertion of Lot I^* into First Position 38
Figure 9 Number of Better Solutions Found by Random Start and Heuristic RO 51
Figure 10 Percentage Deviation of Makespan of RO with respect to Random Start 52
Figure 11 Percentage Deviation of the Solution Values from the Optimal Solution.....55
Figure 12 Percentage Deviation of the Solution Values from the Best Solution Found 56
Figure 13 Performance Profile on [1, 1.07] 57
Figure 14 Average CPU Times (in seconds) 58
Figure 15 Percentage Deviation from the Best Solution (%) 61
Figure 16 Performance Profile on [1, 1.042] 62
Figure 17 Average CPU Time (in seconds) 63

NOMENCLATURE

FSLS	Flow Shop Lot Streaming
GA	Genetic Algorithm
GTSP	Generalized Traveling Salesman Problem
LS	Lot Streaming
LSSP	Lot Streaming Sequencing Problem
LP	Linear Programming
MMFS	Multi-machine Flow Shops
MPLS	Multi-product Lot Streaming
No-Wait	Start the processing of each sublot on each machine immediately after it is finished on the preceding machine
SA	Simulated Annealing
TS	Tabu Search

Chapter 1

INTRODUCTION

Current global competition and industrial restructure leads to better production scheduling and faster responses for changing market needs. Thus, accelerated yet sustainable scheduling of production processes becomes a key issue for flow shop manufacturing systems. One of the many approaches to deal with this challenging matter is the concept of lot streaming.

1.1. Multi Stage Scheduling Problems

Multi stage scheduling problems can be studied in three general environments: Job shop, permutation flow shop and hybrid flow shop scheduling.

For the job shop scheduling problems, any processing order of the jobs on the machines is allowed. Moreover, the operations must be processed in a given order on the machines for each job; however this order may be different depending on the jobs. For instance, a job may require multiple processing on a particular machine. In the case of open shop problem which is a type of job shop scheduling when the processing order is arbitrary, every operation is assigned to a given machine; however they have the freedom to select the order of the operations of each job on the given machine. The solution to this type of scheduling problem comprises total order of operations of a job as well as total order of operations on a machine.

For the permutation flow shop problems, the processing of every job must be completed on machines in a fixed order. Moreover, the processing order of the jobs on the machines is the same for every machine. Therefore, the number of operations of a job is equal to the number of machines for the permutation flow shop problems. Thus, the main problem in permutation flow shop is to generate a permutation of all jobs that minimizes the makespan. This scheduling problem is NP-hard and complexity increases with the number of machines and the number of jobs.

Hybrid flow shop scheduling is the generalization of flow shop scheduling problems where a stage may have multiple processing machines. Flexible flow shop (FFS) scheduling, which has also been referred to as hybrid flow shops and as multi-processor flow shops, is a type of hybrid flow shop scheduling where jobs may skip some stages. (Pinedo (2008))

1.2. Objectives of Scheduling Problems

Among many different kinds of scheduling objectives three of them prevail among the researchers throughout the decades. These objectives are minimizing makespan, minimizing mean weighted tardiness and minimizing total flow time.

The objective of minimizing makespan, which amounts to minimizing the amount of time required for all the jobs to complete processing on certain sets of machines, is utilized when the main concern is to increase the production rate. By minimizing makespan in scheduling problems the idle time on machines is eliminated and jobs may be produced in a shorter time period.

Minimizing the mean weighted (by job priority) tardiness is one of the typical objectives in both single and multi-job scheduling problems. Tardiness is defined as the difference between order due date and completion date of a job if its completion date is greater than its order due date. Thus, the objective of minimizing mean weighted tardiness is utilized when main concern is to meet the customer demand before the order due date.

Considering penalty (weights) on late delivery, this objective also minimizes cost of late delivery and maximizes customer satisfaction.

Flow time of a job is defined as the time elapsed between the completion time of the job and the time the job was first available for processing. Minimizing total flow time, which is often referred to as the sum of the completion times of jobs when all jobs are available at time zero, measures the responsiveness of the production system. In many production systems which involve queues (for instance, in networks) the flow time of a job consists of both the waiting time in the queue and the processing time on machines so that minimization of flow time leads to improvements in service quality. (Leonardi and Raz (1997))

1.3. The Concept of Lot Streaming

In multi-machine ***flow shop manufacturing systems***, a scenario where different products (called lots) which have several discrete items (called sublots) are to be processed on each machine is commonly used. One of the many techniques to deal with the processing of the entire lot set is to transfer them in smaller batches instead of waiting for the whole batch to be processed on each machine. This concept of splitting a production batch (called lot) into smaller portions (called sublots); to be transferred and scheduled in an overlapping fashion on downstream machines is called ***lot streaming***.

In contrast with the concept of lot sizing, which is a medium term planning approach and aims to determine production lot size and inventory levels in order to minimize setup and inventory holding costs and meet a given customer demand, the concept of lot streaming is a short term planning approach which is mainly utilized after orders are released to shop floors with given lot sizes found in lot sizing stage in order to find the optimal sublot sizes and production sequences.

To illustrate this concept, examine the following example where the lot consisting of 64 identical items is to be processed on two-machines. The unit processing times of the lot are 2 and 7 for machine 1 (M1) and machine 2 (M2) respectively. In a traditional two-machine flow shop environment (under which no lot streaming concept is utilized), it is straightforward to find the makespan of the lot over two-machines (In our case below makespan is 576 time units). The resultant chart without lot streaming concept utilized is shown in Figure 1:

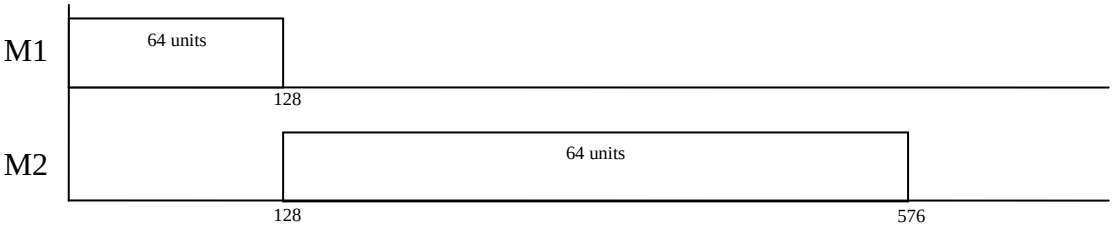

Figure 1. Processing without lot streaming concept

On the other hand, when we split the entire lot into four equal sublots and permit processing of sublots in overlapping fashion over two-machines we obtain the following chart in Figure 2:

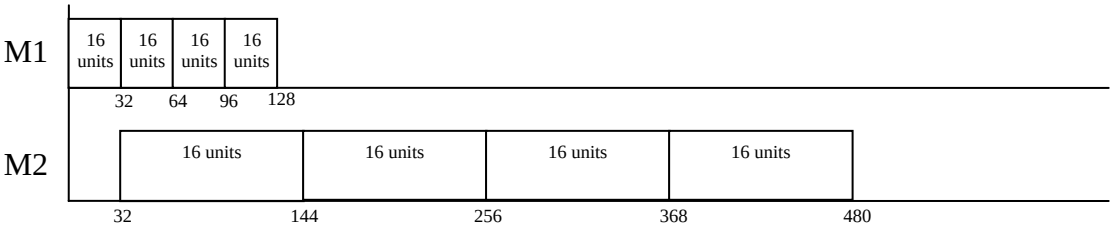

Figure 2. Processing with lot streaming concept

One of the many advantages of lot streaming is the reduction in the makespan value, which is depicted in the Figures 1 and 2 (576 time-units in Figure 1 against 480 time-units in Figure 2 which leads to 16.6% decrease in the makespan).

1.4. Advantages of Lot Streaming

While the main benefit of lot streaming is to reduce the makespan of the resulting schedule, it also reduces work-in process (WIP) inventory levels and in turn improves the overall performance of the production system. As supported by Kalir and Sarin (2000), adopting lot streaming concepts in flow shop manufacturing environment is more beneficial in terms of "three commonly used performance measures, namely, the makespan, the mean flow time, and the average WIP levels."

Reducing WIP and mean flow time of the production batches are the core concepts of ***lean manufacturing***, which mainly targets any possible elimination of waste created by the manufacturing system. Toyota, as one of the pioneers of lean manufacturing, fully deploys the concept and pursues elimination of "***The Seven Wastes***" in their entire manufacturing steps. The seven wastes are identified as follows: Transport, inventory, motion, waiting, overproduction, over processing and defects.

Reductions in WIP, makespan and mean flow time of products have strong connection with inventory, waiting and overproduction wastes in the system. Keeping unnecessary inventory, which could be specifically thought of in the form of WIP in our case, causes a capital expense that has not generated any income yet. Therefore, this ***waste of capital*** is reduced to a large extent by employing the concept of lot streaming. Moreover, ***waste of time***, which generally emanates from waiting for the next production step, is also decreased by the lot streaming concept, since the main drive to apply lot streaming is to lower the makespan and mean flow time of products. Finally, overproduction of semi and final products is induced when the demand of customer is unknown (to keep safety stock) or

total production time of all products takes a very long time, which is the case of production of large batches. Overproduction has the utmost importance since it triggers off all the other wastes. Utilization of lot streaming concept in manufacturing facilities yields elimination of the aforementioned consequences of overproduction to a considerable extent.

1.5. Contributions and Summary of Remaining Chapters

In this research, we propose two heuristic approaches for the lot streaming problem in multi-product flow shop environments. We consider the multi-product lot streaming (MPLS) problem with equal and consistent sublots in multi-machine flow shops (MMFS) with objective of minimizing the makespan. We present comparative results of experimental studies for both heuristics, which show that solution qualities of both heuristics that we developed are better than or equal to those obtained by the heuristic and exact methods from the literature. Solution qualities of both heuristics can be attributed to construction of better initial solution by using the several properties of bottleneck dominant, reversely bottleneck dominant and interior lots. Moreover, utilization of the interior lot concept, which can be regarded as the main contribution of the research, leads us to generate better neighborhoods at each iteration of the specially structured tabu search algorithm for LS problems.

Chapter 2 provides detailed background information and literature review on four distinctive kinds of lot streaming problems, namely single-product – 2-3 machines, single-product – multi-machine, multi-product – 2-3 machines and multi-product – multi-machine lot streaming problems. This chapter also gives a brief review on lot streaming problems with consistent sublots.

Chapter 3 explains the core concepts in lot streaming terminology which we use while constituting the heuristic method. Moreover, thorough explanation of the heuristic

algorithm (heuristic RO) is presented and specially structured tabu search method is also introduced. Finally, a claim is proven in order to demonstrate the significance of using an "interior lot" in the heuristic method.

Chapter 4 deals with the LS problems with consistent sublot size. Proposed heuristic approach for MPLS problems with consistent sublots in MMFS (heuristic RO - Consistent), which utilizes heuristic RO for generating better initial lot sequences and tabu search algorithm for improving the quality of sublot size matrix, is provided in this chapter.

Chapter 5 provides computational experiments of proposed heuristics comparing them with similar studies in the literature to demonstrate the performance and the solution quality of the proposed heuristic algorithms.

Chapter 6 briefly summarizes and concludes the study while explaining differentiating parts of developed heuristics.

Chapter 2

LITERATURE REVIEW

2.1. Overview

Lot streaming (LS) problems in the literature differ from each other depending on several components. Overview of lot streaming problems can be generalized in Figure 3. Moreover, Figure 3 also demonstrates the ground of this study in the literature (blue shadowed items).

LS problems in flow shop production systems with time-based objective function prevail among the researchers. Furthermore, the most common objective is makespan minimization since it provides more realistic scenarios for the production facilities. Articles with makespan minimization objective in LS literature can be generalized in four distinctive categories, which are single-product – 2-3 machines, single-product – multi-machine, multi-product – 2-3 machines and multi-product – multi-machine LS problems. Therefore, literature survey will essentially mention the articles falling into these four categories. Moreover, comprehensive review of articles dealing with LS problems with consistent sublots will be presented, and finally the survey will be concluded with the articles on other types of scheduling problems which are related to lot streaming.

The literature mainly focuses on simpler LS problems because the complexity of LS problems increases rapidly for large scale problems. Main explanations for this simplification are the complexity of product sequencing problem, which is classified as

NP-hard, and the existence of an additional sublot allocation problem. Researchers generally partition the entire multi-product lot streaming (MPLS) problems into a series of sub-problems and deal with those particular cases individually. To solve these sub-problems, they have proposed different solution techniques most of which include heuristic approaches.

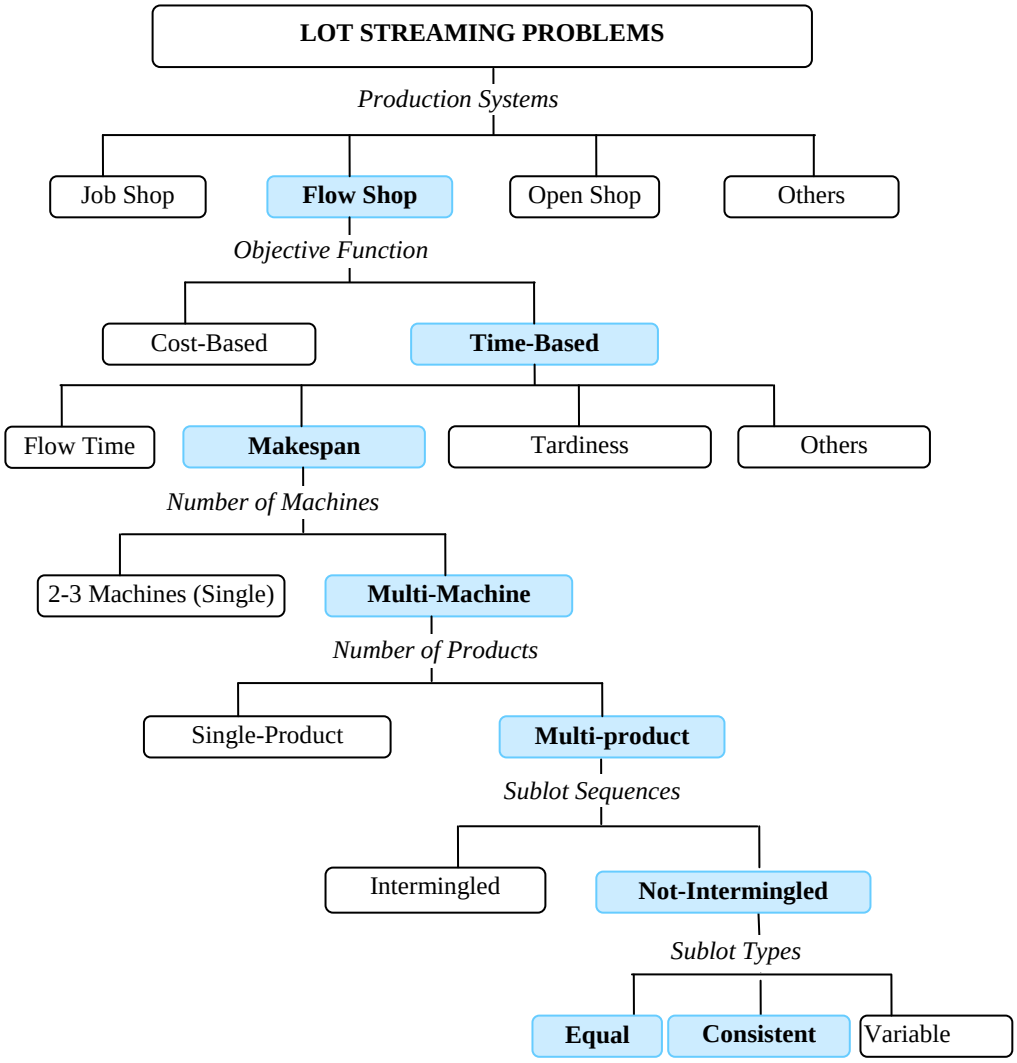

Figure 3. Characteristics of Lot Streaming Problems in the Literature

2.2. Lot Streaming Problems with Single-Product – 2-3 Machines

Potts and Baker (1989) model the single-product lot streaming problem up to three-machines for the flow shop environments allocating work across sublots with objective of minimizing the makespan. They highlight two important cases: Consistent sublots with the same allocation of work across all machines and equal sublots under which work is allocated equally among sublots on all machines. When number of machines is less than or equal to three, they show that they can always find the optimal scheduling policy with consistent sublots.

Baker and Jia (1993) deal with LS problems with single-product and three-machines. They endeavor to find special constraints imposed on the schedule. They research the impacts of different constraints on the makespan value, such as no-idling, equal and consistent sublot sizes. Their experimental study shows that different combination of those constraints may eventually increase the makespan value of optimal schedule. On the other hand, they have statistical analysis on the impact of dominant machine over other machines. They stress that no-idling constraint has no effect on the problem if second machine is dominant over other machines.

Chen and Steiner (1998) study the single-product LS problem with attached setup times in a multi-machine flow shop. They extend the results of Glass et. al (1994) with no-setup case to the case of attached setups. For the optimal sublot sizes, they split the problem into three cases and deal with each case individually proving the optimality conditions for each of the cases. They also conclude that in some certain situations no-wait schedules are more desirable, since it can be seen from the structure of the optimal solutions that no-wait requirement can be satisfied without increasing the length of the optimal schedule.

Sen et al (1998) present theoretical results on single-product lot streaming problem with two-machines. They also perform analysis on equal, consistent and variable sublot typed lot streaming problems. They conclude that even when variable sublot sizes are allowed,

the schedule with consistent sublots becomes optimal. Finally, their findings justify the use of equal sublots in practice, since equal sublots give quite effective results in most cases of their analysis.

Sriskandarajah and Wagneur (1999) deal with the scheduling of single-product in two-machine no-wait flow shops. When sublot sizes are allowed to take continuous values, they show that their schedule is always optimal. They also provide a polynomial heuristic procedure to solve the integer sublot sized version.

2.3. Lot Streaming Problems with Single-Product – Multi-machines

Baker and Pyke (1990) consider the scheduling of a single-product in multi-machine flow shop environment with the objective of "cycle-time minimization". They present a solution algorithm for this lot streaming problem with only two sublots. Moreover, newer concept of **bottleneck machine**, which is based on a critical path analogy, is developed and utilized throughout their algorithms. On the other hand, they provide a series of heuristic procedures for the multi-product case owing to their analysis on two sublots.

Glass and Potts (1998) study the lot streaming of a single job in a flow shop, where the objective is to minimize makespan. They only deal with the continuous sublot sized LS problems and keep them equal on each machine. They restrict their results to three-machine case and provide a characterization of the structure of an optimal solution. This characterization enables them to extend the range of problems for which they can provide a direct solution. Analysis of the problem is carried out in two-stages: First, they derive a relaxation algorithm, by which they can reduce the number of machines that they consider. Secondly, they characterize the critical path structure of the optimal solution.

Kumar et al. (2000) extend the approach of Sriskandarajah and Wagneur (1999) for the case of multiple machines. They obtain continuous-sized sublots optimally using linear programming approach when the flow shop produces only one type of products.

Kalir and Sarin (2001) address the LS problem with single-product in a flow shop environment. They consider different setup time criteria and develop a solution procedure on how to split a lot into optimal sublots that can optimize several different performance measures. Their main procedure enables them to optimize the objective function of makespan, however their procedure is said to be adaptable to other objective functions, such as mean flow time and work-in-process inventory. In order to address the multi-product version of their problem, they point out the importance of the concept of bottleneck machine.

Chen and Steiner (2003) present a new LP model for the single-product LS problem in no-wait flow shops. They also develop a polynomial time solution for the discrete LS problem with two sublots. Moreover, they have also shown that the minimum makespan achievability does not change by the addition of the no-wait constraints in a regular flow shop when s = 2. On the other hand, for the general case, they have found good quality approximations for the optimal solution.

Kalir and Sarin (2003) address the flow shop lot streaming (FSLS) problem with sublot attached setups. They develop an optimal solution algorithm for the single-batch problem and guarantee finding the optimal solution in a fast and efficient manner with low polynomial order.

Edis and Ornek (2009) develop a heuristic procedure for lot streaming problems in stochastic flow shops. Their proposed heuristic combines simulation and tabu search to minimize the makespan for a single-product multi-stage ***stochastic*** flow shop problem with consistent sublot types and discrete sublot sizes.

2.4. Lot Streaming Problems with Multi-product – 2-3 Machines

Vickson and Alfredsson (1992) consider MPLS problems with two and three-machines. In their theoretical study, they focus on “unit and equal” sublot sizes and present an

optimality condition when setup and transfer times are neglected. Moreover, they show that among many optimal solutions there exists an optimal solution where sublots of the same products are processed continuously on each machine (so called ***non-intermingled*** solution).

Cetinkaya and Kayaligil (1992) extend the study of Vickson and Alfredsson (1992) on MPLS problems with two and three-machines by considering the case of detached setups. They suggest a procedure of finding optimal solution for two-stage flow shops with unit sized transfer batches. Their procedure is similar to the one of Johnson's rule (1954). The procedure includes run-in and run-out times of lots between two machines and comparison is made accordingly. Moreover, Cetinkaya (1994) and Vickson (1995) showed that sequencing of the products optimally and splitting of each product into optimal sublots could be done disjunctively. The optimal sequence was obtained using Johnson's rule (1954) as stated above.

Sriskandarajah and Wagneur (1999) deal with the lot streaming and scheduling of multiple products in a two-machine no-wait flow shop and present efficient algorithms for solving the problem simultaneously. They also show that sequencing and lot streaming of multiple products with continuous sized-sublots can be done in polynomial time. Based on their findings and results, they present an effective heuristic procedure to solve sequencing and lot streaming of multiple products with ***integer-sized sublots***.

Bukchin et al. (2002) present a solution procedure (SMB – Single machine bottleneck) for the two-machine flow shop lot streaming problem with sublot-attached setup times to minimize flow-time. Although their SMB property does not always guarantee the optimality of their solutions, they have an empirical analysis which shows that their solutions are close to the optimal. Finally, they prove that the SMB property is satisfied in all optimal solutions for some specific cases.

Ganapathy et al. (2004) develop two heuristic algorithms (tabu search and simulated annealing) for LS problems in two-machine flow shop environment to minimize makespan and total flow time. Performance comparison between two heuristics is carried out and it is observed that when the objective is to minimize the makespan, tabu search (TS) and simulated annealing (SA) perform almost similar. On the other hand, if the objective is to minimize total flow time, TS outperforms SA regardless of the neighborhood creation schemes.

Marimuthu and Ponnambalam (2005) evaluate three heuristic algorithms, namely genetic algorithm (GA), SA and Baker's algorithm (1995), for lot streaming in a two-machine flow shop to minimize makespan. They propose the GA and SA algorithm in order to compare it with Baker's algorithm (1995). According to their results, GA performs better over others. As one might expect, the CPU time required for the simulated annealing algorithm is less compared with the time for the genetic algorithm. However, they conclude that despite the requirement of higher CPU time, overall performance of genetic algorithm is better for lot streaming and scheduling in a flow shop.

2.5. Lot Streaming Problems with Multi-product – Multi-machines

Kropp and Smunt (1990) investigate optimal lot streaming policies in multi-machine environment and present optimal sublot size policies and two heuristic methods for flow time minimization in a flow shop setting with no additional constraints. Their objective is either minimizing the makespan or mean flow time. For the problem with objective of minimizing the mean flow time, they use quadratic programming approach and determine the optimal way of splitting a job into smaller sublots under various setup times to run time ratios, number of machines in the flow shop, and number of allowed sublots.

Kumar et al. (2000) also extend their study for single-batch multi-machine LS problems to multi-batch problems. For the multi-product and continuous-sized sublots case they

develop a traveling salesman problem (TSP) formulation and determine an optimal sequence that minimizes the makespan. Finally, they devise a different heuristic to obtain an optimal sequence for integer-sized sublots.

Kalir and Sarin (2001) propose a heuristic for the lot streaming problem with *m* machines, which minimizes the idle time between sublots of each product (bottleneck minimal idleness). They find near-optimal sequences for the lot streaming sequencing problem (LSSP) in a multi-batch multi-machine flow shop. Their efficient heuristic procedure is called the bottleneck minimal idleness (BMI) heuristic. The BMI heuristic constructs a schedule attempting to minimize the idle time at the bottleneck machine. Moreover, "bottleneck dominant lots" term is introduced, which is built on a "bottleneck dominance theorem". Bottleneck dominancy and bottleneck dominant lots play a key role in their constructive heuristic. Laha and Sarin (2009) and Glass and Possani (2011) also mention the bottleneck dominance theorem introduced by Kalir and Sarin (2001) in their articles.

With an objective of minimizing the mean weighted absolute deviation from due dates, Yoon and Ventura (2002a) develop an LP model to find the optimal sublot allocation for a given sequence. They also consider the no-wait lot streaming flow shop. They employ sixteen pair-wise interchange methods in order to build the best sequences. They derive these sixteen pair-wise exchange methods from combining four rules used to generate the initial sequences with four neighborhood search mechanisms. In principle, they divide the problem into two independent sub-problems and solve them separately.

Yoon and Ventura (2002b) also develop a hybrid genetic algorithm (HGA) to solve the lot streaming flow shop scheduling problem for the case where buffers between successive machines have infinite capacities and sublots have equal size. Their hybrid approach incorporates LP and non-adjacent pair-wise interchange (NAPI) methods. They also recommend that their hybrid method can be applied to other lot streaming flow shop

scheduling problems where sublots are consistent and buffers between successive machines have finite capacities.

Hall et al. (2003) develop a useful heuristic for minimizing the makespan in no-wait flow shops. They show that it is possible to formulate this problem as generalized traveling salesman problem (GTSP). They also provide an effective and customized heuristic for the multiple product lot streaming problems.

Kalir and Sarin (2003) also deal with the multiple-batches in flow shop lot streaming (FSLS) problem with sublot attached setups. For the multiple-batch problem they propose several algorithms, some of which obtain a near optimal solution procedure for the determination of the number of sublots as well as the sequence in an FSLS problem with sublot-attached setups. Moreover, they develop a heuristic procedure in order to substantially reduce the complexity of the optimal solution algorithm.

Zhang et al. (2005) study the MPLS problem in *m-1* two-stage hybrid flow shops and develop two heuristic algorithms to minimize the mean completion time of the jobs. Both heuristic algorithms are based on the similar strategy of first sequencing the jobs and then lot streaming each job. They utilize an LP model to decide the sublot sizes, after solving the lot streaming problem of each job in the sequence.

Pan and Ruiz (2012) utilize the heuristic of Nawaz, Enscore and Ham (NEH) (1983) to develop the initial solution of their improvement type algorithms (EDA) for MPLS problem in MMFS with setup times under both idling and no-idling case. They claim to outperform all the other algorithms they considered in their paper for the lot streaming flow shop problem with setup times to minimize makespan.

2.6. Lot Streaming Problems with Consistent Sublots

In order to demonstrate the impact of equal and consistent sublots on system performance, Baker and Jia (1993) give comparative results using over 6000 test problems

in a three-stage and one-product environment. They find that when the number of sublots increases, improvements in makespan reduction gradually decline. For each solution procedure they considered, they found that over 80% of the potential makespan reductions from employing ten sublots are already obtained using just three sublots. On the other hand, their performance comparisons indicate that the ratio of makespan values calculated using equal and consistent sublots approaches one, which states that it is less appealing to use consistent sublots instead of equal sublots.

Vickson (1995) considers non-intermingled, discrete and consistent sublot sized LS problems on two-stages. He also considers detached and attached setup cases respectively. For discrete sublot case he presents a polynomially bounded search algorithm, and for the continuous sublot case he finds a closed form solutions.

Sriskandarajah and Wagneur (1999) deal with LS problems with consistent sublot sizes, discrete (as well as continuous) lot sizes and detached setups. They propose a two-staged solution procedure to solve the problem with those settings. Their two-staged approach is then generalized to a three-staged solution procedure by Kumar et al. (2000). In a multi-product, multi-stage, no-wait flow shop environment with non-intermingled and discrete sublots, Kumar et al. (2000) propose a solution procedure composed of three main steps. Initially, they calculate the optimal continuous and consistent sublots individually for each lot by utilizing linear programming. Then, they round the sublots to meet the requirement of discrete sublot case. Finally, they formulate the remaining sequencing problem again as a TSP and solve it using heuristics.

Hall et al. (2003) study the problem of Sriskandarajah and Wagneur (1999) with attached setups and provide an efficient heuristic to solve the multi-stage no-wait lot streaming problem with multiple products, if consistent non-intermingling but integer sublot sizes are assumed. Later, Hall et al. (2005) modify their procedures given in Hall et al. (2003) to minimize the makespan in no-wait two-machine open shops with consistent

and non-intermingling sublots. Since their problem additionally requires a machine sequence for each product, they limit the study to two-stage settings. They utilize a dynamic programming algorithm in order to create dominant schedule profiles for each product, which are required to formulate the open shop problem as a generalized TSP. Finally, they provide an efficient heuristic which shows that good solutions can quickly be found for two-machine open shops with up to 50 products.

Zhang et al. (2005) study the multi-job lot streaming problem in two-stage hybrid flow shops with *m* identical machines at the first stage and a single machine at the second stage. They keep sublot sizes consistent at two stages. They propose two heuristic algorithms to solve the problem. Both heuristics first sequence the jobs and then schedules them one at a time. Only difference between the two heuristics arises when the jobs are sequenced, since they use two distinctive methods to sequence the jobs. Finally, they utilize mixed integer linear programming (MILP) formulation in order to calculate the lower bounds, which are used to compare the results of heuristic solutions and MILP.

Martin (2009) presents a hybrid genetic algorithm/mathematical programming heuristic for the *n*-job, *m*-machine flow shop problems with lot streaming. He restricts the study to consistent sublot sizes. He generates the number of sublots for each job and the size of sublots by the heuristic and also addresses a new aspect of the problem which is the interleaving of sublots from different jobs in the processing sequence.

Buscher and Shen (2011) consider the lot streaming problem with consistent sublot sizes in a job shop environment. They propose a three-phase algorithm which includes the phases of the predetermination of sublot size, determination of tabu-search based schedules and the sublot size variation. In their tabu search algorithm they connect three distinct neighborhood functions through a constructive multi-level neighborhood method. Their test results suggest that all tested instances rapidly converge to their lower bounds.

2.7. Other Types of Scheduling Problems Related to Lot Streaming

Studies regarding multi-product multi-machine sequencing problems date back to 1970's. Campbell, Dudek and Smith (1970) develop a simpler heuristic approach for *n*-product *m*-machine sequencing problems, which is called "***Campbell-Dudek algorithm***" in the literature. They provide a constructive heuristic, in which they generate different lot sequences dependent on the number of machines (at most *m*-1) and print the lot sequence with minimum total processing time as the solution of their constructive heuristic algorithm. Although their algorithm is not classified in lot streaming literature, their study can be regarded as an important contribution to the LS literature since they acknowledge the importance of total processing time of products on machines and include it in their study.

For job shop production environments, the ***shifting bottleneck procedure*** developed by Adams et al. (1988) is one of the pioneers in job shop literature. In their study they describe an approximation method that solves the minimum makespan problem in job shop scheduling. Briefly, they successively manage to sequence the machines taking each time the machine with the highest processing time (the one which is identified as bottleneck machine) among the machines not yet sequenced. Their straightforward method is generally utilized to obtain near optimal sequences (See Dauzere-Peres and Lasserre (1997)).

In a job shop production environment, Dauzere-Peres and Lasserre (1997) propose an iterative procedure to obtain the sublot sizes and the schedule, respectively. Their procedure alternates between solving for the optimal sublot sizes for a fixed number of sublots and a fixed sequence of these sublots on the machines; and the "optimal" schedule, for a fixed number of sublots and fixed sublot sizes. The latter task, which is finding optimal schedule for a job shop environment, is proven to be NP-hard, even for fixed sublot sizes. Thus, they utilize the heuristic, called "***the shifting bottleneck heuristic***", developed

by Adams et al. (1988) in order to obtain schedules that are optimal or near optimal in some cases.

In flow shop scheduling literature, the heuristic proposed by Nawaz, Enscore and Ham (NEH) (1983) is regarded as the best performing heuristic in flow shop environment with the makespan minimization criterion (Turner and Booth (1987), Taillard (1990)). Superiority of NEH comes from its ease of implementation and solution quality particularly on large scale problems. Brief explanation of the NEH procedure can be given as follows: They initially arrange all jobs in decreasing order according to sums of their total processing times. Then, they select first two jobs, and find their order that gives shorter makespan value. Remaining jobs are sequenced according to the similar criterion above: Selected job is inserted into a place in the current subsequence on hand, where it has the shortest makespan. Many articles have compared their results with NEH, some of which claimed to outperform it on different comparison criteria (See Kalir and Sarin (2001) and Rad, Ruiz and Boroojerdian (2009)).

Chapter 3

HEURISTIC APPROACH FOR THE MPLS PROBLEM WITH EQUAL SUBLOTS IN MMFS

3.1. Introduction

This chapter explains the core concepts of lot streaming which will be used in developing the heuristic method. Detailed analysis and explanation of the heuristic algorithm we developed for MPLS problems with equal sublots in MMFS, namely heuristic RO is presented. Moreover, specially structured tabu search method is also introduced. Finally, a claim is proven in order to demonstrate the significance of using an "interior lot" in the heuristic method.

3.2. Terminology

The structure and solution procedure of any given lot streaming problem is generally dependent upon the combination of different components specified in Table 1.

To capture the structure of any given lot streaming problem completely, some of the components supplied in Table 1 will be explained in more detail in the following subsections.

3.2.1. Equal Sublots

When all the sublots of a given lot have the same size, this specific case is called "lot streaming problem with equal sublots". The more restricted version of the case of equal sublots is ***unit sublots***. In the unit sublot case, the lot size of a given product (all items that are to be processed for a given lot) is equal to the number of sublots of the product. Figure

2 is an example of equal sublots where the lot consisting of 64 identical items is split into four equal sublots of 16 items each.

Table 1. Different Components of Lot Streaming Problem

Components	**Level**
Product (Lot) Types	Single-Product Multi-product
Machine Types	Single Machine Multi-machine
Production Types	Flow shop Job shop Open-shop
Sublot Types	Equal Consistent Variable
Number of Sublots	Upper-bounded Fixed
Sublot Sequence	Intermingled Not-Intermingled
Performance Measures	Makespan Mean Flow Time Average Tardiness Number of Tardy jobs Total Tardiness Total cost

3.2.2. Consistent and Variable Sublots

When sublot sizes of the lot are not restricted to be the same, we have a general case of the problem. If invariable (identical in size) sublot sizes are used for transferring a lot from one machine to another, we use the term ***consistent sublot***. On the other hand, if sublot sizes that are used for transferring a lot from one machine to another may vary, we have the most general case of the lot streaming problem, which is called lot streaming problem with ***variable sublots***.

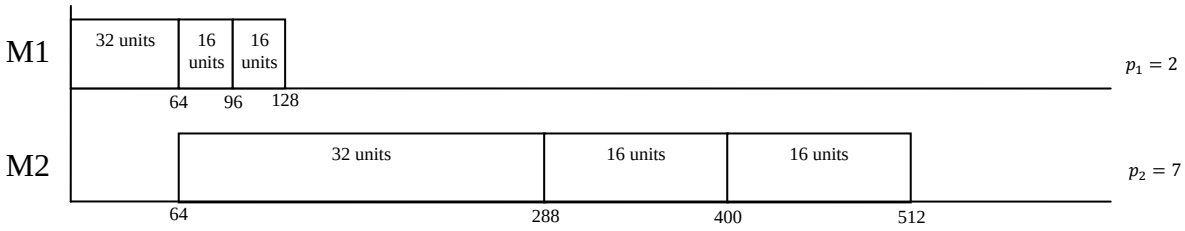

Figure 4. Lot streaming with consistent sublot sizes of 32, 16, and 16 items

Figure 4 illustrates an example of consistent sublots, where sublot sizes are selected as 32, 16 and 16 respectively. As indicated in the definition of consistent sublot, sublot sizes are kept the same in between machines 1 and 2.

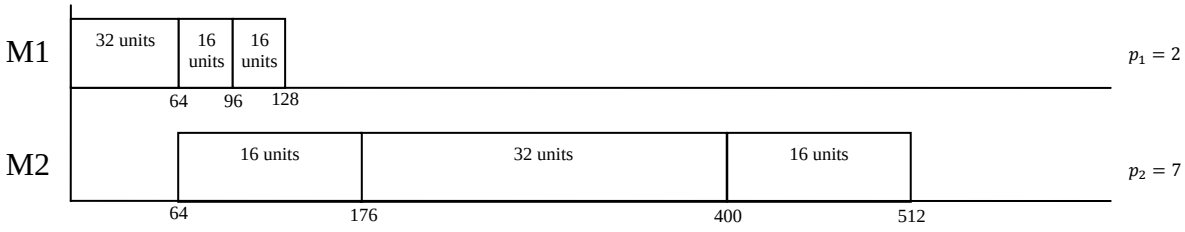

Figure 5. Lot streaming with variable sublot sizes of 32, 16, and 16 items in machine 1 and 16, 32, and 16 items in machine 2

Figure 5 provides an example of variable sublots, where sublot sizes are selected as 32, 16 and 16 items respectively for machine 1 and 16, 32, and 16 items respectively for machine 2. As indicated in the definition of the variable sublot, sublot sizes may vary in between machines 1 and 2. Even though the resulting makespan values for problems with consistent and variable sublots are the same, depending on the processing times and different sublot sizes they may result in different makespan values.

3.2.3. Bottleneck Dominance Theorem and BMI Heuristic by Kalir and Sarin (2001)

Kalir and Sarin (2001) propose a heuristic for the lot streaming problem with *m* machines, which aims to minimize the idle time between sublots of each product. In their heuristic procedure called the bottleneck minimal idleness (BMI) heuristic, they provide a "bottleneck dominance theorem" and introduce the term "bottleneck dominant lots", which is also utilized in the heuristic procedures we developed. The bottleneck dominance theorem states that if for some lot *i*, the difference:

$$p_{i,BN} - \max_{1 \le j < BN} \{p_{i,j}\} \geq 0$$

Then, under lot streaming, there would be no idle time created between the sublots of lot *i* on the bottleneck machine (Kalir and Sarin (2001)). They define a lot *i* as "bottleneck dominant" if it satisfies the bottleneck dominance property given above. The BMI heuristic considers only two sets, which are the set of bottleneck dominant lots (S_1) and the set of bottleneck dominated lots (S_2) and orders them by arranging the lots in the decreasing order of the closeness of the secondary bottleneck machine to BN. On the other hand, the heuristic we developed (Heuristic RO) considers three distinct sets, namely the sets of bottleneck dominant lots, reversely bottleneck lots and interior lots. We also have entirely different ordering and tie breaking rules, unique sequence construction steps and additional tabu search algorithm for generating better final sequences, which will be covered in upcoming sections.

3.2.4. Run-in and Run-out Times for Lot *j*

Run-in and run-out times are used for lot streaming problems with two-machines in particular. In order to gain more knowledge about how sublots of a lot can spread over two-machines without increasing the makespan value, we use these terms. ***Run-in time*** for lot *j* is defined as time elapsed after its first sublot begins processing on machine 1 till its latest possible start on machine 2 without increasing the makespan.

Consider the lot streaming problem depicted in Figure 4 in which processing times are given as 2 (p_1) and 7 (p_2) for machine 1 and 2 respectively. Run-in time for the lot given in Figure 4 is 64. In general, for the case of two-machines if $p_1 \leq p_2$ holds, run-in time of the lot will be the ***total processing time*** of the ***first*** sublot on the first machine (machine 1). On the other hand, if $p_1 > p_2$ holds (as in Figure 6), sublots of the lot can be right-shifted on the second machine (machine 2) to eliminate possible idleness created between its sublots on machine 2 without increasing the makespan value. In Figure 6, the run-in time for the lot will be 352 (not 224) due to the properties given above.

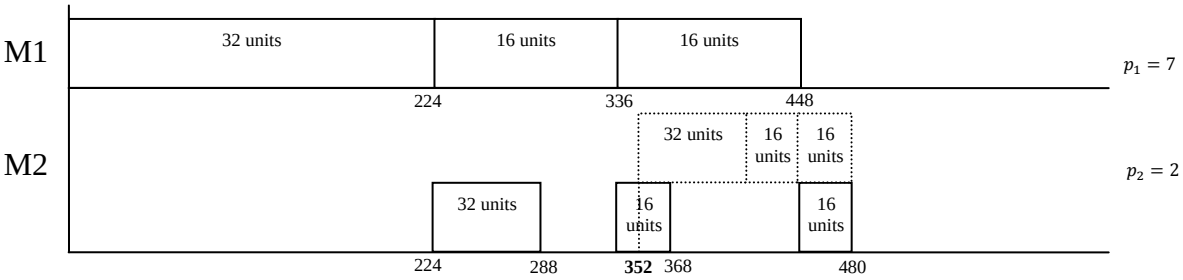

Figure 6. Run-in time for lot *j* when $p_1 > p_2$

Run-out time for lot *j* is defined as the time elapsed between the end of its processing on the first machine and that on second machine. Contrary to the run-in time calculation given above, if $p_1 \leq p_2$ holds (as in Figure 4), run-out time of the lot will be 384 (512 - 128) according to the procedure specified above. On the other hand, if $p_1 > p_2$ holds (as in Figure 6), run-out time of the lot will be equal to the ***total processing time*** of the ***last*** sublot

on the second machine (machine 2). For the lot given in Figure 6, its run-out time is 32 (16*2, also 480 – 448).

3.2.5. Intermingled versus Non-Intermingled Schedules

Intermingled schedules allow sublot(s) of a different lot to intervene between successive sublots of another lot on the same machine. The problem structure and solution methodology completely changes when intermingling is allowed. In our study, we do not allow intermingling of different sublots, so we only deal with **non-intermingled schedules**.

3.2.6. Sublot Transfer Times/Costs and Setup Times

When a sublot finishes its operation on a machine and needs to be conveyed to the downstream machine (the machine which comes later in the fixed machine sequence) ***transfer times*** and (or) costs may occur. If transfer times (and/or costs) constitute a major part of the objective function, they need to be considered in the problem structure. Furthermore, in some of the flow shop environments ***setup times*** of machines between the processing of sublots of different lots may take longer than expected. In this case, the setup time parameter needs to be added to the problem definition. Moreover, sublot-attached and sublot-detached setups are two of the common setup types that have distinct characteristics. In the ***sublot-attached*** version, setup of a lot on a given machine cannot be started unless its first sublot arrives at that machine. On the other hand, in ***sublot-detached*** case we are able to perform the setup of the lot on a given machine if its first sublot is being processed on a previous machine.

3.3. Problem Definition

In this study, we propose a heuristic (***heuristic RO***) for multi-product lot streaming (MPLS) in multi-machine flow shop (MMFS) environments, where sublots of all lots are equal or consistent, no idling is permitted in between sublots and each sublot consists of integer number of items. The motivation behind this study is to characterize certain

properties of the problem so that the heuristic can be extended to more general versions in future studies.

3.4. General Assumptions

As briefly explained in earlier sections, we have some general assumptions that most of the studies in the literature exploit and also some particular ones that only we employ for our study. Here is the list of assumptions which are used throughout the study:

1. All jobs (lots) to be processed are available at time zero.
2. Intermingling of sublots belonging to different lots is not allowed.
3. Preemption of any sublot is not allowed; that is, processing of a sublot on any machine cannot be interrupted.
4. Sublot transfer times between machines are assumed to be zero (negligible).
5. Setup times of machines between the processing of sublots of different lots are neglected.
6. Sublot sizes of each lot are assumed to be ***equal.***

3.5. Heuristic Algorithm for the LS Problem with Multi-Product Flow Shops with Equal Sublots (Heuristic RO)

Below we present the heuristic algorithm (heuristic RO) developed for MPLS in MMFS with equal sublots. The following notation is used throughout the section:

Parameters:

m	Number of machines
n	Number of jobs (lots)
L_i	Lot size for lot i
$p_{i,j}$	Unit processing time of lot i on machine j
ss_i	Size of sublots of lot i
ns_i	Number of sublots of lot i

Indices**:**

j	Machine index, $j = 1..m$

i	Lot index, *i = 1..n*

Sets**:**

M	Machine set
N	Job (lot) set
S	Bottleneck dominant lot set
R	Reversely bottleneck dominant lot set
I	Interior lot set
Bestfinalseq	Final set of lot sequences

Heuristic RO:

Set loop = 0;

While loop $\leq \mathbf{1}$, do the followings:

if loop $== 0$;

Keep processing time matrix (p matrix) as it is. (So that ordinary pass will be executed.)

elseif loop $== 1$;

Reverse processing time matrix. (So that backward pass will be executed.)

End

Step 1:

Identify primary and secondary Bottleneck Machines (BN) by calculating total time spent on each machine, that is:

$$T_j = \sum_{i=1}^{n} L_i * p_{ij}\,, \qquad j \in M.$$

The machine with the highest T_j value will be the primary bottleneck machine (BN1) and the machine with the second highest T_j value will be the secondary bottleneck machine (BN2).

Step 2:

Identify bottleneck dominant lots, the concept which is introduced by Kalir and Sarin (2001), using BN1 machine and put them in set *S*. That is: If for some lot *i*,

$$p_{i,BN1} - \max_{1 \le j < BN1}\{p_{i,j}\} \ge 0 \text{ then } Lot\ i \in S.$$

Sort lots in set S in ***ascending order*** of their first arrival times to BN1 machine. That is:

$$FTbn_i = \sum_{j=1}^{BN1-1} ss_i * p_{i,j}\ , \qquad i \in S\ .$$

If there is a tie, break it by favoring the lot with ***smaller*** run-out time between BN1 and BN1+1 machine (Sarin and Jaiprakash (2007)). That is, for lot *i*:

If $p_{i,BN1} \ge p_{i,BN1+1}$, run − out time for lot *i* is: $\boldsymbol{ss_i * p_{i,BN1+1}}$,

If $p_{i,BN1} < p_{i,BN1+1}$, run − out time for lot *i* is: $\boldsymbol{p_{i,BN1+1} * L_i - p_{i,BN1} * ss_i * (ns_i - 1)}$

Call this final ordered set: $\boldsymbol{S_{sorted}}$

Note: If bottleneck machine is the last machine in the fixed machine sequence, run-out time criterion will not apply. Therefore, new tie breaking rule for this specific case will be: Favoring the lot with the ***smaller processing time*** on bottleneck machine.

Step 3:

Identify ***reversely*** bottleneck dominant lots using BN1 machine and put them in set *R*. That is: If for some lot *i*,

$$p_{i,BN1} - \max_{BN1 < j \le m}\{p_{i,j}\} \ge 0 : Lot\ i \in R.$$

Sort lots in set R in ***ascending order*** of their reversely first arrival times to BN1 machine. That is:

$$RFTbn_i = \sum_{j=BN1+1}^{m} ss_i * p_{i,j}\ , \qquad i \in R\ .$$

If there is a tie, break it by favoring the lot with ***smaller*** run-in time between BN1-1 and BN1 machine. That is, for lot *i*:

If $p_{i,BN1-1} \leq p_{i,BN1}$, run − in time for lot i is: $\boldsymbol{ss_i * p_{i,BN1-1}}$,

If $p_{i,BN1-1} > p_{i,BN1}$, run − in time for lot i is: $\boldsymbol{p_{i,BN1-1} * LS_i - p_{i,BN1} * ss_i * (ns_i - 1)}$

Call this final ordered set: $\boldsymbol{R_{sorted}}$

Note: If bottleneck machine is the first machine in the fixed machine sequence, run-in time criterion will not apply. Therefore, new tie breaking rule for this specific case will be: Favoring the lot with the ***smaller processing time*** on bottleneck machine.

Step 4:

Keep the ***interiors lots*** in set *I.* Interior lots are the lots that satisfy the following:

$$I \in N - (S \cup R)$$

Step 5: (Sequence Construction Step)

If $\boldsymbol{S_{sorted}}$ and $\boldsymbol{R_{sorted}}$ are not empty sets, apply the following procedure:

5. A. Select the first lot in $\boldsymbol{S_{sorted}}$ as ***very first lot in the final sequence***. Remove that lot from $\boldsymbol{S_{sorted}}$ and $\boldsymbol{R_{sorted}}$ (if it is also a member of that set). Keep the sorted order of $\boldsymbol{S_{sorted}}$ and $\boldsymbol{R_{sorted}}$ after the deletion of the selected lot from both lists.

5. B. Select the first lot in remaining $\boldsymbol{S_{sorted}}$ set. Check whether the selected lot *i* is the first lot in $\boldsymbol{R_{sorted}}$ set. If the lot is not the first lot in $\boldsymbol{R_{sorted}}$ set, select this lot as the first lot in the remaining sequence. If it is indeed the first lot in $\boldsymbol{R_{sorted}}$ set, check if $FTbn_i \leq RFTbn_i$. If it is true, then we can again select this lot as the first lot in the remaining sequence. If not, skip this lot and remove it from $\boldsymbol{S_{sorted}}$ and $\boldsymbol{R_{sorted}}$ **(**if it is also a member of that set). Apply this step until all members of $\boldsymbol{S_{sorted}}$ are either selected or skipped.

5. C. Remove lots that are included in the final sequence from the initial $\boldsymbol{R_{sorted}}$ set generated in step 3. If this set is not empty, set the very first lot of this set as ***last lot in the final sequence.***

5. D. If there remains any lot that is not sequenced yet (including interior lots), use the following procedure:

5D.1. Let set *P* collect lots that are not sequenced yet. Sort lots in *P* in ***descending order*** of their total processing times on all machines (P_{val}). That is:

$$Pval_i = \sum_{j=1}^{m} p_{i,j} * L_i \,, \qquad i \in P \,.$$

5D.2. For each lot in *P*, calculate total run-in (TRI) and total run-out (TRO) times between (BN1-1)-BN1 and BN1-(BN1+1) machines. TRI and TRO times are calculated as follows:

Total Run-in (TRI) Time: As explained in step 3, TRI is calculated between (BN-1) - BN and BN - (BN+1) machines.

Total Run-out (TRO) Time: As explained in step 2, TRO is calculated between (BN-1) - BN and BN - (BN+1) machines.

Note: If BN machine is either the first or the last machine, TRI time should be calculated between primary bottleneck machine (BN) and secondary bottleneck machine (BN2). Similarly, if BN machine is either the first or the last machine, TRO time should be calculated between primary bottleneck machine (BN) and secondary bottleneck machine (BN2).

5D.3. Select the first lot in sorted set *P* and locate it in the final sequence according to following criteria:

- If for lot i $TRI_i > TRO_i$ is true, sequence lot i to the top of the remaining sequence.
- If for lot i $TRI_i \leq TRO_i$ is true, sequence lot i to the end of the remaining sequence.
- Remove lot i from set *P*.

Repeat step 5D.3 until each of the member of set *P* is sequenced.

<u>Step 6</u>:

Add the resulting final sequence that is generated in steps 1-5 to the ***bestfinalseq*** list.

Note: If loop number is 1, ***reverse the resulting final sequence*** and add it accordingly.

Step 7: **Generation of revised lot sequences**

Apply the following procedures to the final sequence generated in step 5, and add the resulting sequence(s) (If there exists any change in the schedule) to the ***bestfinalseq*** list:

A. For each lot $i \in S$ in the final sequence, if the processing time of lot *i* on BN1 machine is greater than the processing time of that lot on BN1+1, insert lot *i* to place where it comes after ***all of the interior lots*** in the final sequence. For instance, consider that we have an LS problem with 5 products and final sequence is generated as 2-1-4-3-5 throughout steps 1-6. Moreover, assume that the interior lot is found to be lot 3 and bottleneck dominant lots are 1 and 2. If lot 1 satisfies the property given above, revised lot sequence will be: 2-4-3-1-5.

B. For each lot $i \in R$ in the final sequence, if the processing time of lot *i* on BN1 machine is less than the processing time of that lot on BN1-1, insert lot *i* to place where it comes before ***all of the interior lots*** in the final sequence. For instance, consider that we have an LS problem with 5 products and final sequence is generated as 2-1-3-4-5 throughout steps 1-6. Moreover, assume that the interior lot is found to be lot 3 and reversely bottleneck dominant lots are 4 and 5. If lot 4 satisfies the property given above, revised lot sequence will be: 2-1-4-3-5.

Step 8:

Reverse the processing time matrix (matrix p) and carry out steps 1-8 according to reversed p matrix. Increase the loop number by 1. ($Loop = Loop + 1$)

Note: Reverse of the final lot sequence generated using the reversed *p* matrix will provide an appropriate sequence for the problem with original processing time matrix. Thus, if the loop number is equal to 1 (that is, if processing time matrix is reversed) resulting final sequence must be reversed in order to get an appropriate solution.

Step 9: Generation of better lot sequences by tabu search (TSA)

Steps 1-8 generate at most 10 different lot sequences and we keep them in ***bestfinalseq*** list, which records all the final lot sequences. In this step, we narrow the number of lot sequences in ***bestfinalseq*** list down to at most five elements. Selection rule while narrowing the list down to five elements is to simply favor the lot sequences with lower makespan value. Then we apply a specially structured tabu search algorithm (TSA) for lot streaming problem, using those five lot sequences as initial solutions for TSA. TSA will be explained in Section 3.6.

Step 10:

Select the final lot sequence(s), which are the output of TSA, with minimum makespan value as the output of heuristic RO.

3.6. Tabu Search Algorithm for Generating Better Sequences

Tabu search algorithm designed is examined in detail in the following subsection. While the TSA is in general generic, there are some features, which make it unique for the lot streaming problem.

3.6.1. Steps of Tabu Search Algorithm

Tabu search algorithm is comprised of four main steps, namely initialization of parameters, best neighborhood generation, construction of tabu list and termination:

Step 1: Initialization of Parameters

Essential parameters of the algorithm are initialized in this step. There are mainly three components to be initialized: Maximum iteration number, tabu tenure and initial solution of the algorithm.

Among those components, ***maximum iteration number*** determines after how many iterations the algorithm is going to terminate. Moreover, a solution which is previously visited is kept in a tabu list for certain number of iterations (called tabu tenure). ***Tabu***

tenure is the number that specifies how many of those previous solutions to be stored. On the other hand, ***initial solution*** determines the starting point of the algorithm. Finding a better initial solution seems to be crucial for TSA, since it controls where the algorithm starts generating new solutions by using its neighborhood function and generally selection of well-structured solutions leads to better final solutions (in terms of the objective function) in fewer number of iterations. Below we present initial values for those three components:

Maximum Iteration Number: Number of products

Tabu Tenure: Number of products / 2

Initial Solution: As explained previously, five final lot sequences obtained by using heuristic RO are utilized as initial solutions respectively.

Step 2: **Best Neighborhood Generating Function**

This step differentiates our TSA from other studies in the literature. To begin with, a lot sequence that is obtained by heuristic RO is used as an input for the "***Best Neighborhood Generating Function***". Then, the function produces new lot sequences (as many as the number of products) using specially structured ***insertion method*** and selects the lot sequence with the minimum makespan value as an output of the function.

Insertion Method:

Insertion method generates totally a "product size" of new lot sequences from the current available sequence on hand. For instance, if we have 5-product LS problem with current available sequence of 2-1-3-4-5 as an input for the best neighborhood function and if the lot to be inserted is selected as lot 2, we will generate five new sequences, such as: 2-1-3-4-5, 1-2-3-4-5, 1-3-2-4-5, 1-3-4-2-5 and 1-3-4-5-2 from the current sequence using insertion method.

***Differentiating Part of the Insertion Method*:**

Insertion method is based on the following priority rule: ***Interior lots*** are privileged to be selected as a candidate for insertion. Other lots are considered after all the interior lots are inserted in their initial positions in the lot sequence. On the other hand, if the first member of the output of the best neighborhood generating function is an interior lot, it is immediately disregarded and kept in the tabu list. Main reason behind it is given in upcoming sections (Section 3.7).

Step 3: Construction of Tabu List

First of all, tabu tenure is set to be the half of the number of products as stated above (For a 10-product LS problem, tabu tenure will be 5).

Construction of tabu list is comprised of two major parts: Firstly, lot sequences which have been visited in the recent past (less than *n* iterations ago, where *n* is the number of previous solutions to be stored, namely the tabu tenure) is kept in the tabu list. This prevents the algorithm from visiting the same sequences repeatedly and so allows a diversification of generated lot sequences.

Secondly, if the first member of the lot sequence is among interior lot list, this lot sequence is directly added to tabu list (as described previously).

Aspiration Criterion:

Commonly used aspiration criterion is also utilized here: We allow lot sequences which are better than the currently-known best sequence with minimum makespan value. Specifically before directly adding the lot sequence with its first member being a lot among interior lots, this aspiration criterion prevents us from disregarding the solution if it satisfies the criterion.

Step 4: Termination Criterion of Tabu Search

Single termination criterion is utilized for our TSA, which is checking whether pre-specified number of iterations is passed or not. As stated above, specified number of iterations (maximum number of iterations) is set to be the "number products".

3.6.2. Time Complexity of RO and TSA

Steps 1 to 8 of heuristic RO run in $O(n^2 + nm)$ mainly due to step 5 of the algorithm (sequence construction step). On the other hand, examining the tabu search algorithm in detail, step 2 (best neighborhood generating function) is found to be the one which adds more complexity than other steps. Considering the insertion method in step 2 with the time complexity of $O(n)$ together with the maximum iteration number of n (the number of products), we culminate with the total complexity of $O(n^2)$. Thus, heuristic RO still runs in $O(n^2 + nm)$.

3.7. Significance of Using an Interior Lot in the Heuristic Method

To better observe the importance of the utilization of interior lots repeatedly in steps of heuristic RO as well as in steps of tabu search algorithm, a claim is made and proven. Before the details of the proof, preliminary information about the proof is as follows:

Preliminary Information:

1. Claim deals with multi-product lot streaming (MPLS) problem in 3-machine flow shop ($m = 3$) with unit and equal sublot sizes ($ss_i = 1, i \in N$). The claim also applies for MPLS problem in 3-machine flow shop ($m = 3$) with equal sublot sizes if unit processing time matrix ($p_{i,j}$) is multiplied by the size of sublots (ss_i) for each product. ($ss_i * p_{i,j}$)
2. Claim applies when bottleneck machine (BN1) is the second machine in fixed machine sequence. (BN1 = 2)
3. Claim applies for the case where there is ***no idle time*** created on BN1 machine before the processing of the selected interior lot.

Following notation will be used in the rest of the proof:

Notation:

I^*B_j : Time elapsed from the start of first sublot of all the lots before lot I^* to the completion of last sublot of all the lots before lot I^* on machine j, $j \in 1..3$

I^*_j : Time elapsed from the start of first sublot of lot I^* to the completion of last sublot of lot I^* on machine j, $j \in 1..3$

I^*A_j : Time elapsed from the start of first sublot of all the lots after lot I^* to the completion of last sublot of all the lots after lot I^* on machine j, $j \in 1..3$

D_j : Deadline on machine j found by calculating the completion time of last sublot of all the lots after lot I^* on machine j, $j \in 1..3$

T_j : Completion time of last sublot of lot I^* on machine j, $j \in 1..3$

α, β : Length of the block $(I^*)_1$ and $(I^*)_2$ respectively

Claim:

For any given sequence if lot $i \in I$ (the set of interior lots) is inserted into **first** position and the remaining lots are shifted accordingly, minimum makespan value that particular sequence can have **will not decrease**.

Proof:

Suppose that there exists a lot satisfying the property given above (call this lot: Lot I^*) and also suppose that we have sequence on hand where lot I^* is not the first lot of that sequence. Given the preliminary information, a typical sequence where lot I^* is not the first lot is depicted in Figure 7:

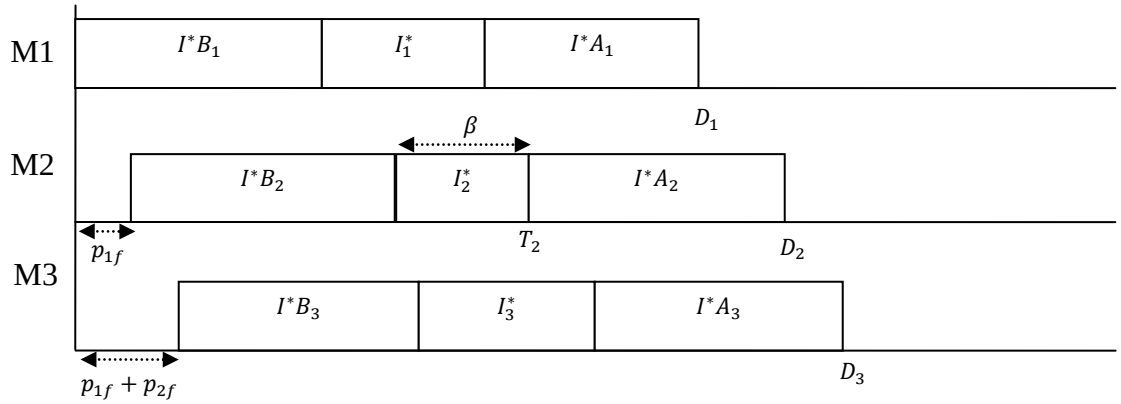

Figure 7. Before the Insertion of Lot I^* into First Position

In Figure 7, I^*B_j values represent the total amount of time that all the lots before lot I^* spend on each machine *j*. On the other hand, I^*A_j values represent the total amount of time that all the lots after lot I^* spend on each machine *j*. After inserting lot I^* into first position, we obtain the sequence depicted in Figure 8:

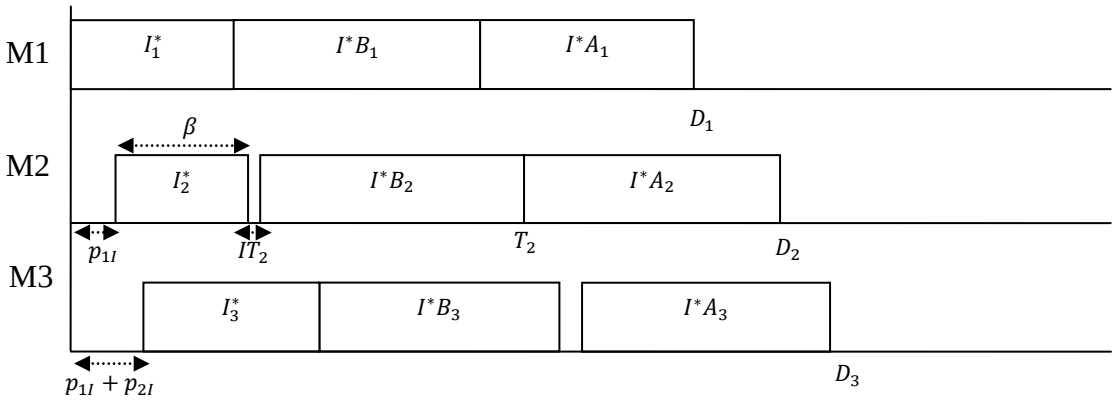

Figure 8. After the Insertion of Lot I^* into First Position

Let f denote the first sublot of block ($\boldsymbol{I^*B}$), g denote the last sublot of block ($\boldsymbol{I^*A}$), $p_{j,i}$ denote the processing time of lot *i* (*i* = 1...*N*) on machine *j* (*j*=1...3), ns_{I^*} denote number of sublot of lot $\boldsymbol{I^*}$ and IT_2^{after} denote the possible idle time created on machine 2 between

blocks $(\boldsymbol{I^*})_{\boldsymbol{2}}$ and $(\boldsymbol{I^*B})_{\boldsymbol{2}}$ after the insertion**.** When we examine the completion time of last sublot of lot I^* on machine 2 (T_2) before and after the insertion, we obtain the following:

Before Insertion:

$$T_2^{before} = p_{1,f} + I^*B_2^{before} + \beta^{before} + IT_2^{before}$$
$$(i) \qquad (ii) \qquad (iii) \qquad (iv)$$

After Insertion:

$$T_2^{after} = p_{1,I^*} + I^*B_2^{after} + \beta^{after} + IT_2^{after}$$

We observe that parts (*i*), (*ii*), (*iii*) and (*iv*) are the main components that constitute T_2 before and after the insertion. Now we examine the changes in the values of each component case by case:

Case 1: If $p_{1,f} \geq p_{1,I^*}$;

(i) From the assumption above $p_{1,f} \geq p_{1,I^*}$

(ii) $I^*B_2^{after} = I^*B_2^{before}$ since $\boldsymbol{p_{1,f} \geq p_{1,I^*} > p_{2,I^*}}$ causes IT_2^{after} to be greater than zero, so block of $(\boldsymbol{I^*B})_{\boldsymbol{2}}$ will not be compressed and remain the same.

(iii) $\beta^{after} = (ns_{I^*} - 1) * p_{1,I^*} + p_{2,I^*} = ns_{I^*} * p_{1,I^*} - (p_{1,I^*} - p_{2,I^*})$

$= \alpha - (p_{1,I^*} - p_{2,I^*})$

$\boldsymbol{\beta^{after} < \alpha}$

Observe that value of β^{after} has the maximum value it can have, since in the case of β^{before}, β^{before} may be compressed by block of $(\boldsymbol{I^*B})_{\boldsymbol{2}}$ and its value may decrease, thus: $\boldsymbol{\beta^{before} \leq \beta^{after}}$

(iv) It is given that $IT_2^{before} = 0$ and for IT_2^{after}, since $\boldsymbol{p_{1,f} \geq p_{1,I^*} > p_{2,I^*}}$ IT_2^{after} will be greater than zero. Also its exact value will be, $IT_2^{after} = p_{1,f} - p_{2,I^*}$

Checking $T_2^{after} - T_2^{before}$:

$= p_{1,I^*} + I^*B_2^{after} + \beta^{after} + IT_2^{after} - p_{1,f} - I^*B_2^{before} - \beta^{before} - IT_2^{before}$

$= p_{1,I^*} - p_{1,f} + p_{1,f} - p_{2,I^*} + \beta^{after} - \beta^{before}$

$= p_{1,I^*} - p_{2,I^*} + \beta^{after} - \beta^{before} > 0 \rightarrow \boldsymbol{T_2^{after} - T_2^{before} > 0}$

Case 2: If $p_{1,f} < p_{1,I^*}$;

(i) From the assumption above $p_{1,f} < p_{1,I^*}$

(ii) $I^*B_2^{after} \leq I^*B_2^{before}$ holds because:

If $p_{1,f} \leq p_{2,I^*} < p_{1,I^*}$ it causes IT_2^{after} to be zero, so block of $(\boldsymbol{I^*B})_2$ may be compressed and its value after insertion may decrease. The exact decrease in its value will be: $p_{2,I^*} - p_{1,f}$

If $p_{2,I^*} \leq p_{1,f} < p_{1,I^*}$ it causes IT_2^{after} to be greater than zero, so block of $(\boldsymbol{I^*B})_2$ will not be compressed and its value after insertion will remain the same. So, both cases indicate that $I^*B_2^{after} \leq I^*B_2^{before}$

(iii) $\beta^{after} = (ns_{I^*} - 1) * p_{1,I^*} + p_{2,I^*} = ns_{I^*} * p_{1,I^*} - (p_{1,I^*} - p_{2,I^*})$

$= \alpha - (p_{1,I^*} - p_{2,I^*})$

$\boldsymbol{\beta^{after} < \alpha}$

Observe that value of β^{after} is the maximum value it can have, since in the case of β^{before}, β^{before} may be compressed by block of $(\boldsymbol{I^*B})_2$ and its value may decrease, thus: $\beta^{before} \leq \beta^{after}$

(iv) It is given that $IT_2^{before} = 0$ and for IT_2^{after}, (ii) states that value of IT_2^{after} will be either zero or $p_{1,f} - p_{2,I^*}$.

Checking $T_2^{after} - T_2^{before}$:

If $p_{1,f} \leq p_{2,I^*} < p_{1,I^*}$;

$= p_{1,I^*} + I^*B_2^{after} + \beta^{after} + IT_2^{after} - p_{1,f} - I^*B_2^{before} - \beta^{before} - IT_2^{before}$

$= p_{1,I^*} - p_{1,f} - p_{2,I^*} + p_{1,f} + \beta^{after} - \beta^{before}$

$= p_{1,I^*} - p_{2,I^*} + \beta^{after} - \beta^{before} > 0 \rightarrow \boldsymbol{T_2^{after}} - \boldsymbol{T_2^{before}} > 0$

If $p_{2,I^*} \leq p_{1,f} < p_{1,I^*}$;

$= p_{1,I^*} + I^*B_2^{after} + \beta^{after} + IT_2^{after} - p_{1,f} - I^*B_2^{before} - \beta^{before} - IT_2^{before}$

$= p_{1,I^*} - p_{1,f} + p_{1,f} - p_{2,I^*} + \beta^{after} - \beta^{before}$

$= p_{1,I^*} - p_{2,I^*} + \beta^{after} - \beta^{before} > 0 \rightarrow \boldsymbol{T_2^{after}} - \boldsymbol{T_2^{before}} > 0$

We observe that for this particular 3-machine case of multi-product lot streaming problem, makespan value may increase when the value of $\boldsymbol{T_2^{after}}$ increases. The reasoning is given as follows: Exact value of deadline on machine 2 after the insertion can be found as: $D_2^{after} = \boldsymbol{T_2^{after}} + (\boldsymbol{I^*A})_2^{\boldsymbol{after}}$. Since the increase in $\boldsymbol{T_2^{after}}$ will either compress the possible idleness in the block $(I^*A)_2$ or have no effect on the size of that block, it is certain that D_2^{after} will **not decrease**.

Moreover, it is easy to verify that any feasible sequence of the problem will satisfy the following inequality which sets a lower bound on the deadline on machine 3 (which is also a lower bound on the makespan value of that particular lot sequence):

$D_3^{after} \geq \boldsymbol{D_2^{after}} + p_{3,g}$

It is also clear that if D_2^{after} increases, smallest value that D_3^{after} can have will also increase. In our case, since D_2^{after} will not decrease after the insertion, we are sure that the smallest value that D_3^{after} can have will not decrease either.

Conclusion:

It is shown that in each case $\boldsymbol{T_2^{after}}$ will be greater than $\boldsymbol{T_2^{before}}$ which signifies that D_3^{after} will not decrease. Thus, we can conclude that for any given sequence if lot $i \in I$ is

inserted into **first** position and the remaining lots are shifted accordingly, minimum makespan value that particular sequence can have **will not decrease**.

Heuristic RO extends this inference for MPLS problems in MMFS and gives more importance on interior lots in sequence construction steps of the heuristic as well as in the tabu search algorithm. As explained in more detail in previous sections, heuristic RO and tabu search algorithm restrict the insertion of an interior lot into first position by:

- Giving priority to bottleneck and reversely bottleneck dominant lots whilst the selection of the first (and the last) member of the lot sequence,
- Immediately disregarding and keeping the lot in the tabu list if the first member of the output of the best neighborhood generating function is an interior lot.

Chapter 4

HEURISTIC APPROACH FOR MPLS PROBLEM WITH CONSISTENT SUBLOTS IN MMFS

4.1. Introduction

The chapter examines LS problems with consistent sublot size. To provide a better understanding of LS problems with consistent sublot sizes, problem definition is given. Details of the heuristic approach for MPLS problems with consistent sublots in MMFS (heuristic RO - Consistent), which utilizes heuristic RO for generating better initial lot sequences and tabu search algorithm for improving the quality of sublot size matrix, is provided in this chapter.

4.2. Problem Definition of LS with Consistent Sublots

When sublot sizes of the lot are not restricted to be the same (as in the case of equal sublots), we have a general case of the problem. If invariable (identical in size) sublot sizes are used for transferring a lot among each pair of machines, we use the term ***consistent sublot***.

It is apparent that when the sublot sizes are changed from equal to consistent for a given problem instance, makespan value will decrease. (Or at least remain the same) Main reasoning is as follows: Utilization of consistent sublot sizes gives us more flexibility to generate final schedules with less idle times between sublots of each lot. On the other hand, as we use consistent sublot sizes, problem structures and solution methods may vary. For

instance, ***determination of lot sequences*** is the main goal of the LS problems with equal sublots, since sublot sizes (so the number of sublots for each lot) are known beforehand. On the other hand, LS problems with consistent sublots have multi objectives, namely ***determination of number of sublots*** (hence sublot sizes) and also ***determination of lot sequences***.

4.3. Heuristic Approach for MPLS problems with Consistent Sublots in MMFS (Heuristic RO - Consistent)

Following notation will be used in the rest of the subsection:

Notation:

Parameters:

S'_j	Maximum number of sublots for lot *j*
M	Number of machines
J	Number of products (lots)
L_j	Lot size for lot *j*
t_{jm}	Unit processing time of lot *j* on machine *m*

Indices:

s,t	Sublot indices, *s,t* = 1..S'_j
m	Machine index, *m* = 1..*M*
j	Lot index, *j* = 1..*J*

Decision Variables:

SS_{js}	Size of sublot *s* of lot *j*
p_{jsm}	Processing time of sublot *s* of lot *j* on machine *m*
b_{jsm}	Starting time of sublot *s* of lot *j* on machine *m*

Using the notation given above, we can use the following heuristic to generate coherent solution for MPLS problems with Consistent Sublots in MMFS (heuristic RO - Consistent):

Steps of Heuristic RO – Consistent:

Step 1: Generate initial matrix SS for heuristic RO by simply applying the following steps:

- Create $S_j^{'}$ number of sublots for lot *j*
- For each sublot of lot *j*, set the initial sublot size as $\left\lfloor \frac{L_j}{S_j^{'}} \right\rfloor$
- Starting from the first sublot of lot *j* (till the last sublot of lot *j* if applicable) increase the size of each sublot of lot *j* by 1 to accommodate each of the remaining products.
- Apply these steps for all products to create the final matrix SS

For instance, when the lot size of lot *j* (L_1) is 15 and the maximum number of sublots for lot *j* ($S_1^{'}$) is 6, then applying the procedure given above we will find matrix SS_1 as:

$SS_1 = [3,3,3,2,2,2]$

Step 2: Find a lot sequence by using heuristic RO with initial sublot size matrix generated in step 1 (SS).

Step 3: Generate new matrix SS using ***tabu search algorithm*** with starting solution found in step 2.

Step 4: Using the lot sequence (found in Step 2) and new matrix SS (found in step 3), recalculate the makespan value and print the resulting makespan as the output of heuristic RO – Consistent.

This four-step heuristic method aims to create initial lot sequence by using heuristic RO with given sublot size matrix SS (found in step 1). In order to generate better sublot size matrices, tabu search algorithm is applied in step 3 of the heuristic. Thus, the makespan value of this particular lot sequence is decreased and the resulting makespan is printed as an output of heuristic RO – Consistent in step 4. It is also necessary to give the details of tabu search algorithm applied in step 3.

Tabu Search Algorithm to Generate Better Sublot Size Matrix

Step 1: **Initialization of Parameters**

This step sets the initial values for the following parameters: Tabu tenure, maximum iteration number and starting sublot size matrix. Among those parameters, tabu tenure specifies how many iteration is needed for a lot to be excluded from the list. Maximum iteration number designates the stopping condition for the tabu search algorithm. Finally, starting sublot size matrix is found by the Step 1 of the heuristic RO – Consistent and used here as an initial sublot size matrix. So, initial values for each parameter are as follows:

- Tabu Tenure: Number of products / 2
- Maximum iteration number: Number of products * 2
- Starting sublot size matrix: Found in step 1 of the heuristic RO – Consistent

Step 2: **Best Neighborhood Function**

In this step, we utilize the starting sublot size matrix (SS) found in step 1 of heuristic RO – Consistent as an input for the best neighborhood function of the tabu search algorithm.

In order to find the best neighborhood(s) of the sublot size matrix on hand, following method is used:

- According to current loop number, respective row of matrix SS is chosen. (As an instance, if the loop number is 2 second row of the matrix is chosen.) On the other hand, if the current loop number exceeds the number of rows of the matrix, we go back to the very first row and start over.

  Note: If makespan value is decreased in previous loop, we continue to select the same row of the matrix, until no improvement is possible. Thus, we can apply a deeper search on each row.
- Neighborhoods of the selected array are found by increasing the selected column number by 1 and decreasing the succeeding column number by 1. For instance, applying the procedure given above, neighborhoods of $SS_1 = [3,3,3,2,2,2]$ will be as follows:

  $[4,2,3,2,2,2]$, $[3,4,2,2,2,2]$, $[3,3,4,1,2,2]$, $[3,3,3,3,1,2]$, $[3,3,3,2,3,1]$

  Note: If the succeeding column number of the selected column is 0, it is skipped since no reduction is possible on this neighborhood.
- Replace the rows found above with the selected row of matrix SS and recalculate the makespan value of the same lot sequence based on those candidate matrices.
- Select one of the candidate matrices found above with the minimum makespan value and print the resulting matrix as the output of the neighborhood function.

Step 3: **Construction of Tabu List**

First of all, as specified above, tabu tenure is selected as the half of the number of products. So, for a 10 product LS problem, tabu tenure will be specified as 5. Moreover, members of the tabu list are the rows that are changed in previous iterations. Therefore, reselection of previously visited solution is prohibited so that differentiation of solutions at each iteration is achieved.

Aspiration Criterion: If a row that is candidate to be in tabu list is in the matrix SS which has generated minimum makespan value so far, this row is not put into the tabu list.

Step 4: **Termination of Tabu Search Algorithm**

As explained above, only termination criterion is selected as reaching the predetermined loop number (maximum iteration number), which is specified as twice of the number of products.

After reaching the maximum iteration number, final matrix SS and the lot sequence found in step 2 of heuristic RO – Consistent are used to find the final makespan value.

Chapter 5

COMPUTATIONAL EXPERIMENTS

5.1. Introduction

The chapter begins with the comparison between random start and heuristic RO start for tabu search algorithm. Then comparative computational analysis on MPLS problems with equal sublot sizes is provided. We then switch our focus on MPLS problems with consistent sublot size. Finally, we provide the comparative computational results of our heuristic for MPLS problems with consistent sublots.

5.2. Random Start vs. Heuristic RO Start

In step 1 of the tabu search algorithm, we set the initial values for three parameters of the TSA, namely maximum iteration number, tabu tenure and initial solution of the algorithm. Among three of them, foremost important step is the initialization of the "***starting solution***". The reasoning can be given as follows: In contrast with other parameters, initial solution mostly shapes the computational behavior of the TSA. That is, if the initial solution is already close to one of the optimal solutions, performance of the neighborhood search increases and the number of iterations that TSA spends to converge to optimum can gradually decrease. On the other hand, if the initial solution is close to one of the local optimal solutions, TSA bears the risk of getting stuck at those local optima, printing the same local optimum regardless of the number of iterations that it performs.

This section is devoted to the analysis of performance of our TSA under different initial solutions. For that purpose, 10 instances for 10, 20, and 30 products with 5 and 10 machines are created for LS problem with ***equal sublot size***. Sublot sizes and processing times are randomly generated from a uniform distribution U (*1, 10*). Numbers of sublots are selected as 5 and 10. Thus, a total number of 120 randomly generated problem instances are used to measure the performance of the TSA under two distinct scenarios: The one with the random start (five initial solutions for the TSA are randomly generated) and the one with the heuristic RO start (five initial solutions are obtained via heuristic RO).

For the first scenario, five solutions are generated randomly and used as an initial solution for the TSA. Five solutions are used in order to be consistent with heuristic RO, since it also produces five lot sequences as initial solutions for TSA. The output of the random start is the final lot sequence(s) with minimum makespan value. Moreover, the test is conducted ten times in order to reduce the effect of total randomness.

The second scenario is basically applying heuristic RO on those 120 randomly generated problem instances to obtain the initial solution for TSA.

The performance comparison criterion is selected as the total number of better solutions found by both scenarios. That is, we compare the makespan values of the best sequences obtained in both scenarios and conclude that the solution is "***better***" if the makespan of this solution is less than or equal to the makespan value of the solution obtained by other scenario. Note that whenever both scenarios end up with the solutions that have the same makespan value, we increase the number of better solutions found for both of them.

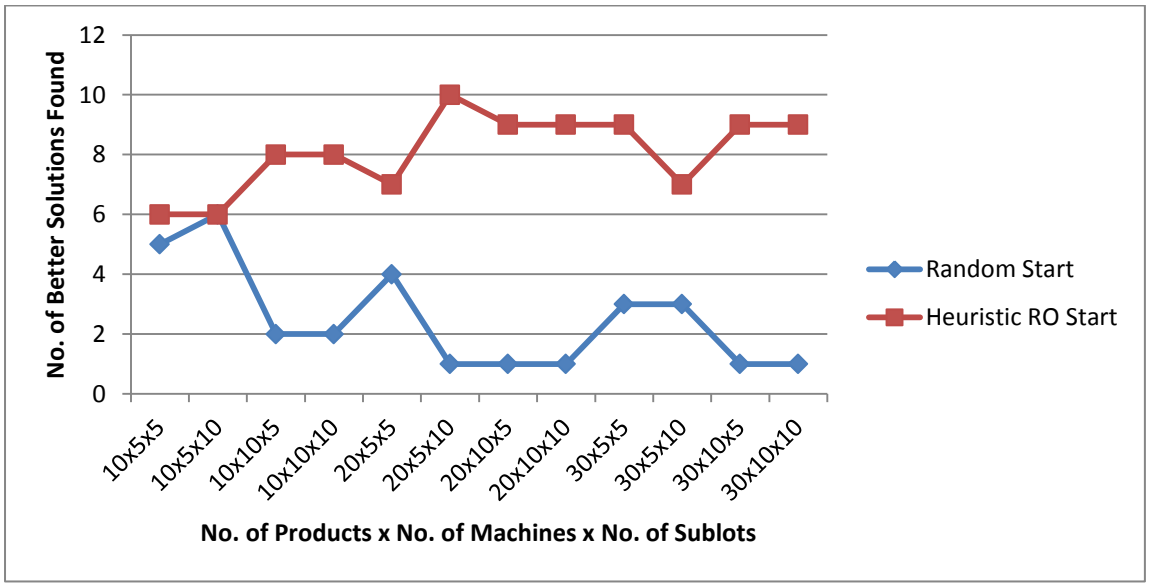

Figure 9. Number of Better Solutions Found by Random Start and Heuristic RO

Figure 9 indicates that using the five final lot sequences obtained via heuristic RO as initial solutions for TSA performs better than using randomly generated initial solutions. One can also deduce that the performance of heuristic RO on larger problem instances is much better than that of random initial solutions. The reason for this can be given as follows: The solution space grows rapidly as the number of products increases; however the growth in the neighborhood space for TSA is gradual. For instance, the solution space for the 10-product problem instances is 10! (3628800), yet the neighborhood size for the neighborhood space function of TSA is only 10, which amounts to saying that we inspect only 10 of 3628800 available sequences at each iteration of TSA. On the other hand, the solution space for the 20-product problem instances is 20! ($2.43290201 \times 10^{18}$) and the neighborhood size for the neighborhood space function of TSA is only 20. Thus, the importance of starting off the tabu search algorithm with a good initial solution becomes more apparent for larger problem instances. So, we can observe the effect of a better initial solution comparing the number of better solutions found by random start and Heuristic RO.

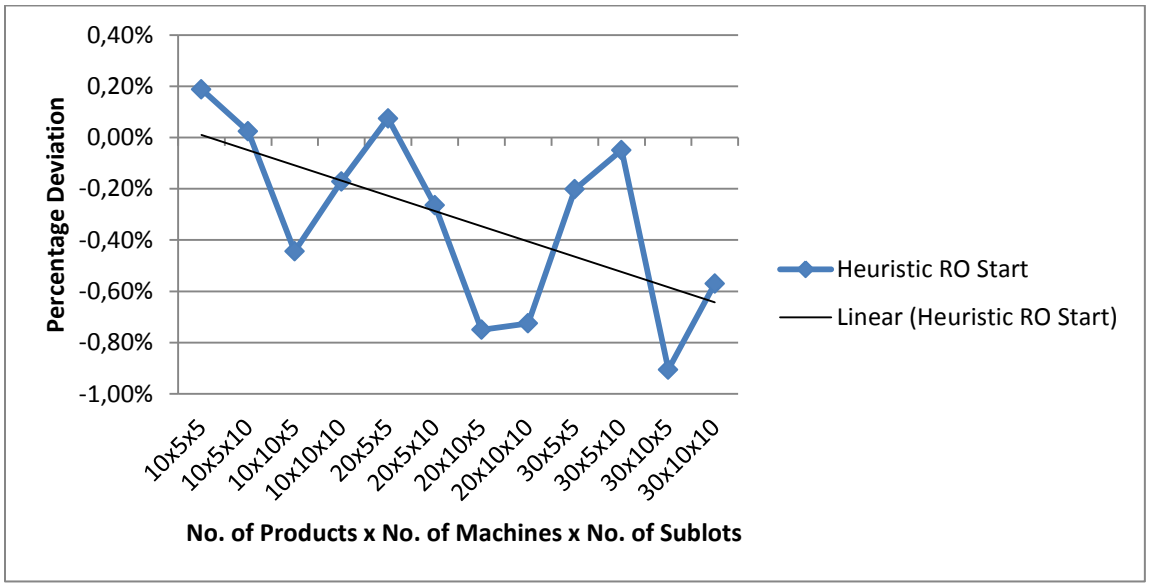

Figure 10. Percentage Deviation of Makespan of RO with respect to Random Start

It is also important to observe the percentage deviation of solutions found by using heuristic RO start from the solutions obtained by using random start. Figure 10 demonstrates the correlation between the total number of better solutions found and the percentage deviation of solutions found by both scenarios.

We can deduce from the Figure 10 that there exists a decreasing trend (from 0.20% down to -0.90%) in percentage deviation of makespan values of heuristic RO with respect to random start as the number of products increases. (Best linear line fit (linear (heuristic RO Start)) for the observations in Figure 10 also shows a decreasing trend.) Moreover, effect of total randomness is also more apparent in this figure. That is, despite the existence of the decreasing trend in percentage deviation of makespan values of heuristic RO with respect to random start, we sometimes observe the uptrend in the figure (specifically in 30x5x5 and 30x5x10 problem instances) which can be attributed to random starting solutions. This

allows us to restate that heuristic RO indeed generates better starting solutions for the initialization step of TSA particularly for the larger problem instances.

To conclude, computational results show the importance of starting with a better initial solution in TSA particularly for larger problem instances. Moreover, five lot sequences obtained from heuristic RO comply with the necessity of starting with better initial solutions for the TSA. Therefore, heuristic RO should be utilized for obtaining good initial solutions for the TSA.

5.3. Computational Results on LS Problems with Equal Sublot Size

Heuristic RO, as thoroughly explained in previous sections, is coded in Matlab programming language. For the tabu search algorithm, C++ programming language is preferred since it has better performance on iterative algorithms.

Same problem instances are utilized for the purpose of comparative analysis between heuristic RO and other heuristics as well as exact methods. To remind the problem instances that we work on, 10 instances for 10, 20, and 30 products with 5 and 10 machines each are created for LS problems with ***equal sublot size***. Sublot sizes and processing times are randomly generated from a uniform distribution U (*1, 10*). Numbers of sublots are selected as 5 and 10. Thus, a total number of 120 randomly generated problem instances are used for the computational analysis.

In order to test the quality of solutions that we obtain via heuristic RO on those problem instances, we selected one heuristic method and one exact method from the literature. The heuristic method that we chose to compare our heuristic RO is the revised version of the heuristic proposed by Nawaz, Enscore and Ham (NEH) (1983), which is regarded as the best performing heuristic in flow shop environment with the makespan minimization criterion. Note that, time complexity of revised NEH is $O(n^2m)$ which is greater than the time complexity of heuristic RO ($O(n^2 + nm)$). Revised NEH heuristic was coded in IBM

ILOG CPLEX 12.1 Development Studio. Moreover, the exact method for solving multi-product lot streaming problems in multi-machine flow shop environment is selected as the integer programming (IP) model of Biskup and Feldmann (2005). Note that, IP model proposed by Biskup and Feldmann solves the MPLS problem with the inclusion of intermingling. Therefore, we adapted their model to our case to solve MPLS problem with no-intermingling allowed. IP model was coded in ILOG CPLEX 12.1 and maximum run time for IP model in CPLEX was selected as 3600 seconds. Therefore, it is important to stress that while comparing those three methods (heuristic RO, revised NEH and IP model) we use the final results of heuristic RO and revised NEH; however some results of the IP model are the best solutions obtained after 3600 seconds.

Primary performance evaluation criterion was selected as the "percentage deviation of the solution values from the best solution found". It is important to point out that IP model managed to terminate before 3600 seconds of maximum run time for the 10-product lot streaming problem instances, so "the best solutions obtained" for the 10-product MPLS problem instances are the ***optimal*** solutions found by IP model. On the other hand, IP model could not terminate in 3600 seconds for all 20 and 30-product LS problem instances. Thus, "the best solutions obtained" in those cases are the ones which have the minimum makespan value among the solutions found by heuristic RO, revised NEH and IP model. Therefore, we separate the analysis of 10-product case from the others.

Figure 11 shows that heuristic RO is capable of finding solutions that have lower percentage deviation than solutions found by revised NEH heuristic with respect to optimal solutions found by IP model for 10-product problem instances.

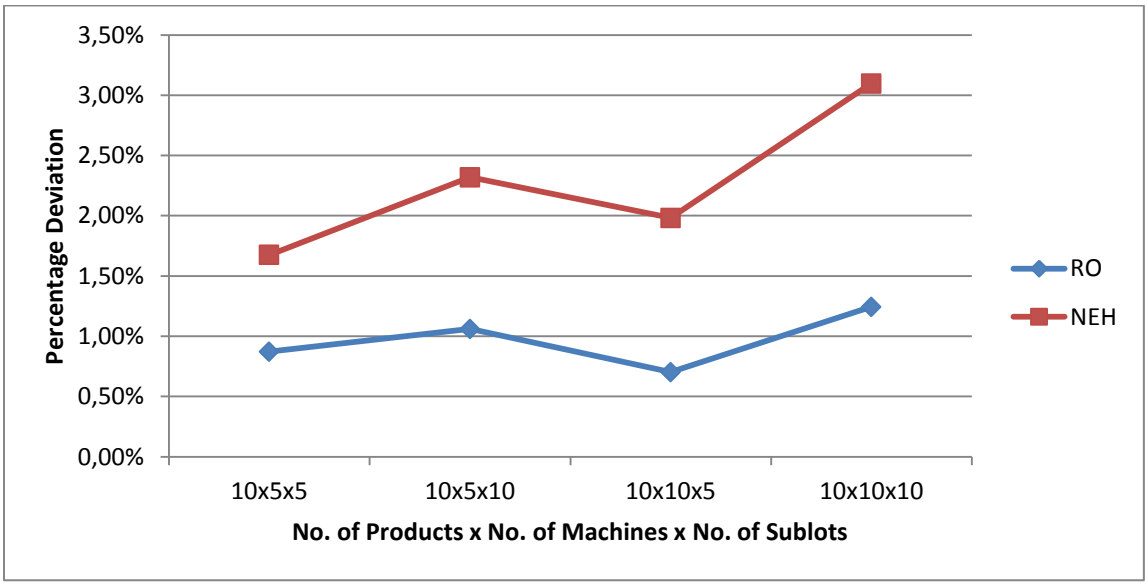

Figure 11. Percentage Deviation of the Solution Values from the Optimal Solution

Moreover, for the analysis of the problems instances with 20 or more products, Figure 12 clearly demonstrates the quality of the final lot sequence generated by heuristic RO compared to both revised NEH and IP formulation in terms of percentage deviation from the best available solution. Heuristic RO mostly finds the best available solution among all different settings of the problem. On the other hand, heuristic RO is more favorable with regards to stability and standard deviation of the solutions found by three methods. Max percentage deviation of heuristic RO is only 1.24% for 10Px10Mx10S problem setting, while it is 3.10% for revised NEH on the same setting and 3.52% for IP formulation on 30Px10Mx10S problem setting, which is also depicted in Figures 11 and 12.

It is also important to point out that both heuristic methods show decreasing trend as the number of products increases, however heuristic RO has less fluctuation than revised NEH heuristic, which again demonstrates the stability and quality of the solutions found by heuristic RO.

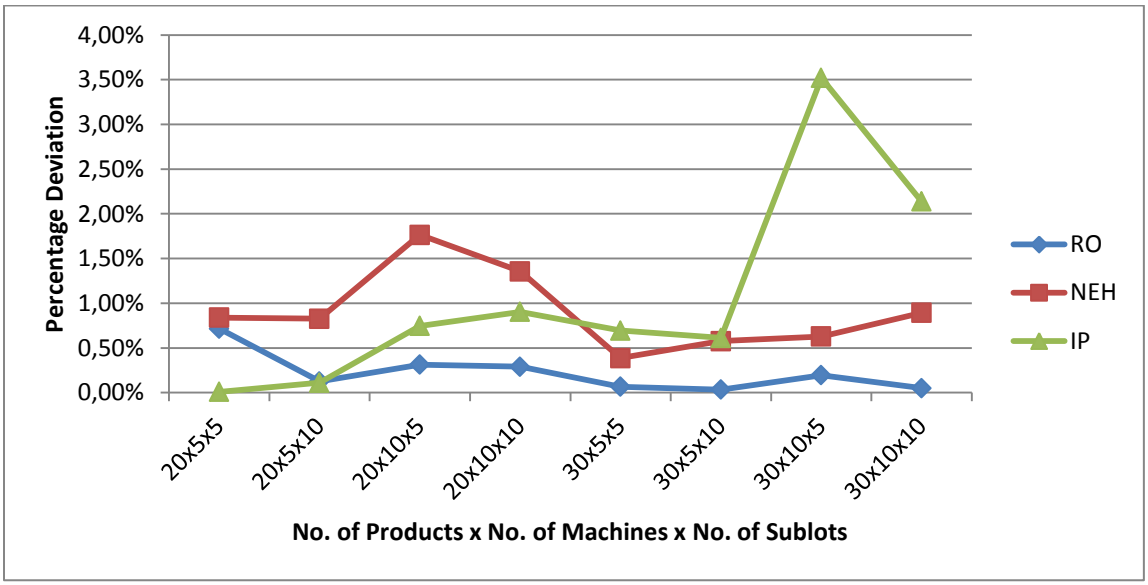

Figure 12. Percentage Deviation of the Solution Values from the Best Solution Found

We also conducted performance profile analysis introduced by Dolan and More (2002) to compare the performances of these three methods on the problem instances with 20 or more products. Using the notation given by Dolan and More (2002), we have 3 methods (solvers) ($n_s = 3$) and 80 randomly generated problem instances ($n_p = 80$). Performance measure is selected as the makespan value found by the solver on a given problem instance. Using set of solvers S and test set P we define $t_{p,s}$ as the makespan value found by solver $s \in S$ on problem instance $p \in P$ for each solver s and problem p. In order to select a baseline for comparisons, we also compare the makespan value of solver s on problem p with the best makespan value by any solver on this problem, so we define our *performance ratio* as: $r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s}: s \in S\}}$ In order to obtain overall assessment of the performance of the solvers $\rho_s(\tau)$, which is the probability for $s \in S$ that the performance ratio $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio, is selected as: $\rho_s(\tau) = \frac{1}{n_p} \text{size}\{p \in P: r_{p,s} \leq \tau\}$. So, the function ρ_s is the cumulative distribution function for the performance ratio of solver s.

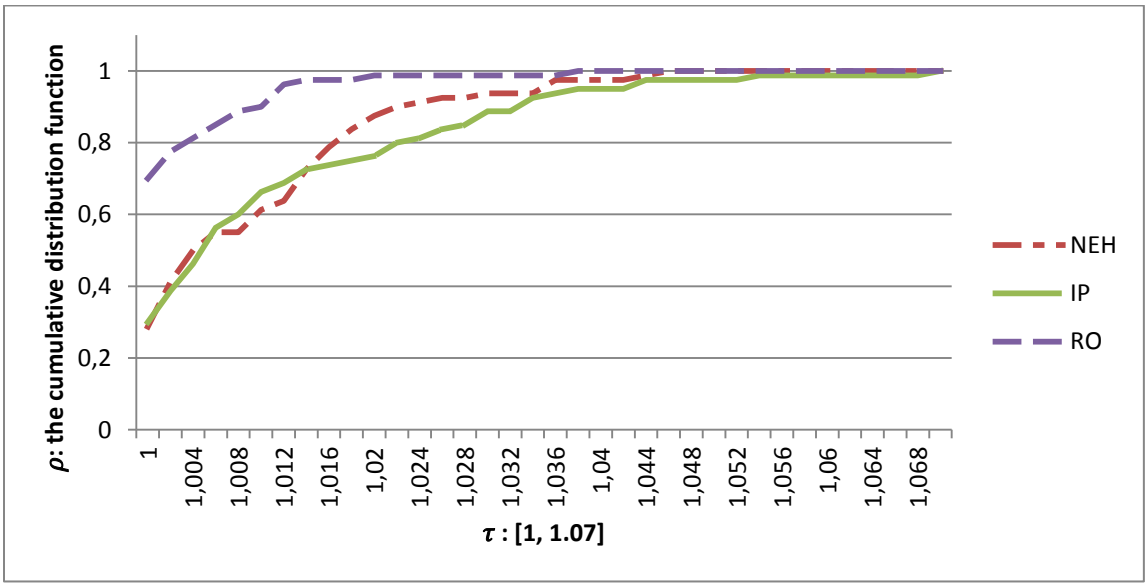

Figure 13. Performance Profile on [1, 1.07]

From Figure 13 we deduce that among all solvers the probability that heuristic RO will find the best solution on a given problem instance is 0.70, while it is only 0.28 for NEH heuristic and 0.30 for IP model. Moreover, the cumulative distribution function of heuristic RO reaches its maximum value of 1 when the factor $\tau = 1.038$ which amounts to stating that it is certain that heuristic RO will find makespan values that are at most 3.8% above the best solution. On the other hand, the factor τ values are 1.046 and 1.068 for NEH heuristic and IP model, respectively. Thus, performance profile analysis also enables us to conclude that heuristic RO performs better on large scale problems compared to NEH and IP model.

"Average CPU Time" should also be considered as the one of the performance indicators and should be used when comparing the performances of both heuristics RO and revised NEH. Figure 14 shows the average computational time (in seconds) results of each method for all different problem settings. Figure 14 also depicts that heuristic RO and

revised NEH spend similar amount of CPU time to generate their solutions. On the other hand, as the IP model spends at least 3600 seconds (predetermined maximum run time of CPLEX model) to find the optimal lot sequence for 20 and 30 product LS problem instances, we can disregard the average CPU time results for IP model.

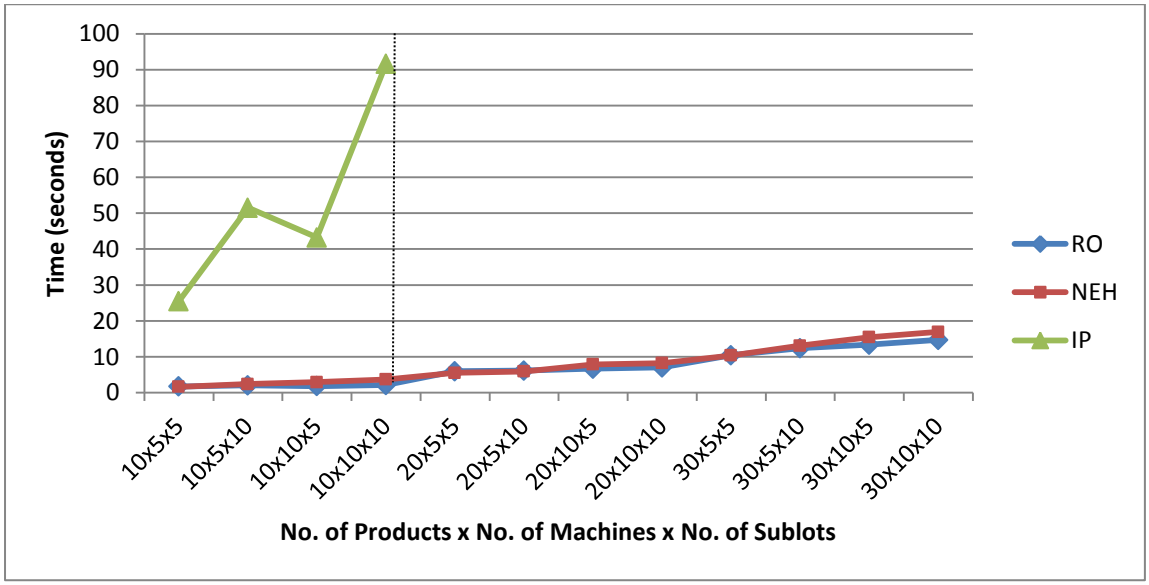

Figure 14. Average CPU Times (in seconds)

5.4. Computational Results on LS Problems with Consistent Sublot Size

Although Heuristic RO is designed specifically for multi-product lot streaming problems with equal sublots on multi-machine flow shops, it is straightforward to adapt the heuristic for LS problems with consistent sublots. Before the detailed explanation on how to adapt heuristic RO for LS problems with consistent sublots, it is necessary to restate the definition of LS problems with consistent sublots.

5.4.1. Revision of IP Model of Feldmann and Biskup (2008)

In order to compare the performance of Heuristic RO – Consistent on selected problem instances, IP model of Feldmann and Biskup (2008) for multi-product lot streaming

problems with consistent and intermingled sublots is adapted to our case, which is MPLS with consistent and not-intermingled sublots. In order to fully adapt the IP model of Feldmann and Biskup (2008) to the consistent and not-intermingled sublot case, following modifications are made using the notation given in section 4.3:

Decision Variables:

New decision variables below are added to the existing IP model and some of the existing decision variables are replaced by those new decision variables:

y_{jk} : Binary (0-1) variable, 1 if lot j is processed before lot k, 0 otherwise, j,k = 1..J

(Replaces x_{jskt} decision variable in original IP model)

u_{js} : Amount processed in sublot s of lot j, j = 1..J, s = 1.. $S_j^{'}$

w_{js} : Binary (0-1) variable, 1 if corresponding $u_{js} > 0$, 0 otherwise, j = 1..J, s = 1.. $S_j^{'}$

Constraints for Not-Intermingling Case:

$y_{jk} = 0$, when j,k = 1..J and $j = k$

$y_{jk} + y_{kj} = 1$, when j,k = 1..J and $j < k$

Addition of each constraint to the existing IP model allows us to prevent intermingling of sublots of different lots. (If lot j is processed before lot k no sublot of lot k can intermingle with the sublots of lot j.)

Introduction of Parameter $S_j^{'}$ to the Model:

$w_{js} * R \geq u_{js}$, j = 1..J, s = 1.. $S_j^{'}$ and R is a sufficiently large integer

$w_{js} \leq u_{js}$, j = 1..J, s = 1.. $S_j^{'}$

$\sum_{s=1}^{S} w_{js} \leq S_j^{'}$, j = 1..J

Addition of all of the constraints above allows us to set different maximum number of sublot levels for each lot, which makes the problem more flexible.

5.4.2. Computational Experiments for LS Problems with Consistent Sublots

Heuristic RO – Consistent is coded in Matlab, whereas additional tabu search algorithm is coded in C++, since C++ can perform better on cyclical operations. Adapted IP model of Biskup and Feldmann (2008) is coded in IBM ILOG CPLEX 12.1 programming language.

In order to compare the performance of both models, 10 instances for 5 and 10 products with 5 and 10 machines each are created for LS problem with ***consistent sublot size***. Processing times are randomly generated from a uniform distribution U (*1, 10*). Also, maximum number of sublots for each lot (array S') are generated as 5 + U (*-1, 1*) and 10 + U (*-2, 2*). Thus, a total number of 80 randomly generated problem instances are used to compare the performance of heuristic RO – Consistent (RO-C) with adapted IP model (IPM) of Biskup and Feldmann (2008).

We again select the primary performance evaluation criterion as "percentage deviation of the solution values from the best solution found" for convenience. "Best Solution" (if available) is regarded as the makespan values found by IPM ***before*** the predetermined termination time. If IPM cannot terminate before the predetermined termination time (which is selected as 3600 seconds), the best solution is the minimum of the solution found by RO-C and IPM.

It is important to point out that IP model manages to terminate before 3600 seconds of maximum run time for the 5-product x 5 machine lot streaming problem instances, so "the best solutions obtained" for those MPLS problem instances are the ***optimal*** solutions found by IP model. Figure 15 shows the results found by both approaches on the same problem instances:

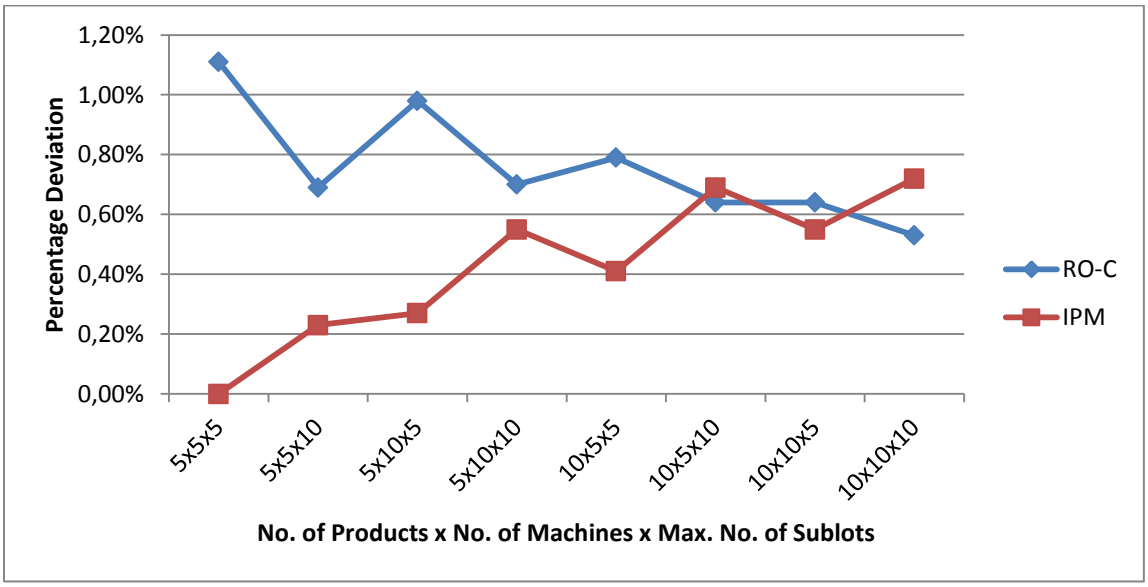

Figure 15. Percentage Deviation from the Best Solution (%)

We can deduce from Figure 15 that RO-C can also find good solutions for MPLS problems with consistent sublots, although it is originally designed for MPLS problems with equal sublots. Using the "percentage deviation from the best solution found" as the primary performance evaluation criterion, for S' matrix with 5±1 RO-C can find solutions that are 0.88% above from the best solution in general. On the other hand, adapted IPM can find solutions that are 0.31% above from the best solution in general. Moreover, for S' matrix with 10±2 when RO-C can find solutions that are 0.64% above from the best solution in general, adapted IPM can find solutions that are 0.65% above from the best solution.

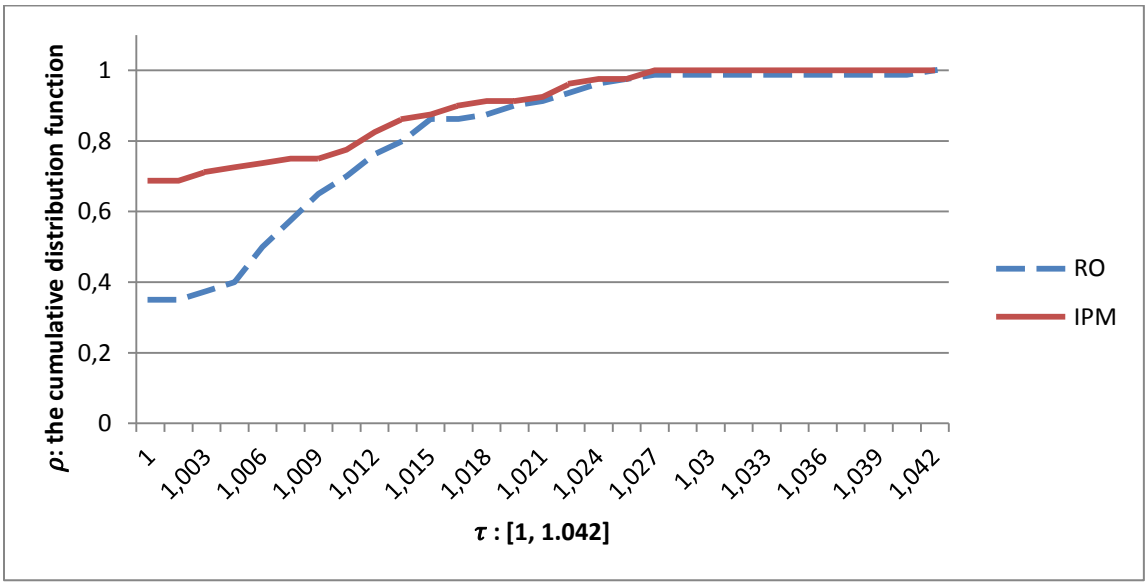

Figure 16. Performance Profile on [1, 1.042]

Conducting performance profile analysis for LS problems with consistent sublots given in Figure 16, we observe that the probability that heuristic RO will find the best solution on a given problem instance is 0.35, while it is 0.68 for IPM. However, the slope of the cumulative distribution function for heuristic RO is greater than that of IPM. Thus, we see that the cumulative distribution function of heuristic RO reaches its maximum value of 1 when the factor $\tau = 1.042$ which amounts to saying that it is certain that heuristic RO will find makespan values that are at most 4.2 % above the best solution. (The factor τ value for IPM is 1.027) To conclude, percentage deviation and performance profile analysis show that RO-C has stability as the number of products as well as maximum number of sublots increase.

It is important to observe how fast our heuristic can find solution in similar quality with IPM, so we choose “average CPU time” as the last performance evaluation criterion. Figure 17 shows the average CPU times of both methods for all problem settings:

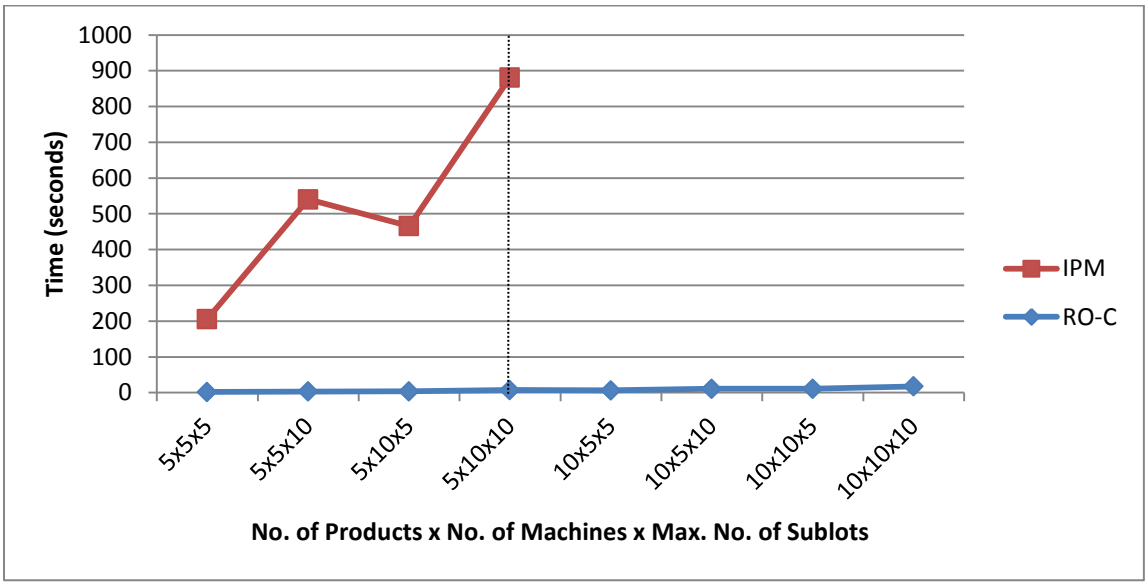

Figure 17. Average CPU Times (in seconds)

Average CPU time results shown in Figure 17 suggest us to conclude that heuristic RO-C spends considerably less amount of time to generate final lot sequences with similar makespan values. It is also important to note that average CPU time values of IPM on the entire 10-product problem are not shown in the figure, because all the averages are equal to the predetermined termination time of 3600 seconds.

Chapter 6

CONCLUSIONS

This chapter presents a summary of the research conducted in this thesis. Several concluding remarks are also presented based on the heuristic method and results analysis.

This research provided heuristic approaches for the lot streaming problem in multi-product flow shop environments. We considered the multi-product lot streaming (MPLS) problem with equal and consistent sublots in multi-machine flow shops (MMFS) with objective of minimizing the makespan.

Firstly, we provided detailed background information and literature review on various kinds of lot streaming problems with equal and consistent sublots. Then we developed two heuristic procedures for equal and consistent sublot sized MPLS problems respectively (heuristic RO and heuristic RO-C). We then presented comparative results of experimental studies for both heuristics. It is shown that solution qualities of both heuristics that we developed are better than or equal to those obtained by the heuristic and exact methods from the literature. There are several justifications for the superiority of the heuristics that we proposed. Heuristic RO uses several properties of the problem to obtain a better solution. Firstly, ordering and tie breaking rules for sorting bottleneck dominant and reversely bottleneck dominant lots are selected according to making bottleneck machine available as soon as possible. Moreover, backward pass utilizes the fact that makespan values of the problems with ordinary processing time matrix and reverse processing time matrix are equivalent. Thus, finding a sequence for the problem with reverse processing

time matrix and reversing that sequence will give a good sequence for the original problem. Furthermore, the final sequences generated by ordinary and backward passes are improved with specially structured tabu search algorithm to obtain local optimum. The results indicate that in most cases this local optimum coincides with the global optimum. Lastly, in sequence construction step (steps 5.D.1, 5.D.2 and 5.D.3), remaining lots are sequenced according to series of criteria, which leads to better sequences than using a single criterion.

Utilization of the interior lot concept led us to generate better neighborhoods at each iteration of tabu search method. The proof of the claim, which supports the use of interior lot concept in both of the heuristics allowed us to give more importance on interior lots in heuristic RO as well as in the tabu search algorithm. Heuristic RO and tabu search algorithm restricted the insertion of an interior lot into first position by giving priority to bottleneck and reversely bottleneck dominant lots while selecting the first (and the last) member of the lot sequence and immediately disregarding and keeping the lot in the tabu list if the first member of the output of the best neighborhood generating function is an interior lot.

Finally, comparative results of experimental studies suggest that heuristic RO and heuristic RO-C find solutions that are better than or equal to those obtained by the revised NEH heuristic IP model formulation. In most of the cases, both heuristics manage to outperform their competitors on several performance evaluation criteria which are percentage deviation from the best solution found, performance profile analysis and average CPU time.

BIBLIOGRAPHY

Adams, J, Balas, E, Zawack, D., The shifting bottleneck procedure for job shop scheduling. Management Science, 34, 391–401, (1988).

Baker, K.R., Lot streaming in the two-machine flow shop with setup times. Annals of Operations Research, 57, 1–11, (1995).

Baker, K.R., Jia, D., A comparative study of lot streaming procedures. OMEGA International Journal of Management Science, 21(5), 561–566, (1993).

Baker, K.R., Pyke, D.K., Solution procedures for the lot streaming problem. Decision Sciences, 21, 475-491, (1990).

Biskup D., Feldmann M., Lot streaming with variable sublots: an integer programming formulation. Journal of the Operational Research Society, Vol 57, 296–303, (2005).

Bukchin J., Tzur M., Jaffe M., Lot splitting to minimize average flow-time in a two-machine flow shop. IIE Transactions, 34, 953-970, (2002).

Buscher, U., Shen, L., An Integer Programming Formulation for the Lot Streaming Problem in a Job Shop Environment with Setups. International Multi-Conference of Engineers and Computer Scientists (IMECS 2011), 1343-1348, (2011).

Campbell, H.G., Dudek, R.A., Smith, M.L., A heuristic algorithm for the n job, m machine sequencing problem. Management Science, 16:B630–7, (1970).

Cetinkaya, F.C., Lot streaming in a two-stage flow shop with set-up, processing and removal times separated. Journal of Operational Research Society, 45 (12), 1445–1455, (1994).

Cetinkaya, F.C., Kayaligil, M.S., Unit sized transfer batch scheduling with setup times. Computers & Industrial Engineering, 22 (2), 177–183, (1992).

Chen, J., Steiner, G., Lot streaming with attached setups in three-machine flow shops. IIE Transactions, 30, 1075–1084, (1998).

Chen, J., Steiner, G., On discrete lot streaming in no-wait flow shops. IIE Transactions, 35, 91-101, (2003).

Dauzere-Peres, S., Lasserre, J.B., Lot streaming in job shop scheduling. Operations Research 45 (4), 584–595, (1997).

Dolan, E.D., More, J.J., Benchmarking optimization software with performance profiles. Math. Program., Ser. A 91: 201–213, (2002).

Edis, R.S., Ornek, M.A., A tabu search-based heuristic for single-product lot streaming problems in flow shops. The International Journal of Advanced Manufacturing Technology, 43 (11), 1202 – 1213, (2009).

Feldmann, M., Biskup, D., Lot streaming in a multiple product permutation flow shop with intermingling, International Journal of Production Research, 46 (1), 197–216, (2008).

Ganapathy, V., Marimuthu, S., Ponnambalam, S.G., Tabu Search And Simulated Annealing Algorithms for Lot-Streaming in Two-Machine Flowshop. IEEE International Conference on Systems, Man and Cybernetics, 4221-4225, (2004).

Glass, C. A., Gupta, J.N.D., Potts, C.N., Lot streaming in three-stage production processes. European Journal of Operational Research, 75, 378–394, (1994).

Glass, C.A. Possani, E., Lot streaming multiple jobs in a flow shop. International Journal of Production Research, 49:9, 2669-2681, (2011).

Glass, C.A., Potts, C.N., Structural Properties of Lot Streaming in a Flow Shop. Mathematics of Operations Research, 23,624-639, (1998).

Hall, N.G., Laporte, G., Selvarajah, E., Sriskandarajah, C., Scheduling and lot streaming in flowshops with no-wait in process. Journal of Scheduling, Vol. 6, 339-354, (2003).

Hall, N.G., Laporte, G., Selvarajah, E., Sriskandarajah, C., Scheduling and lot streaming in two-machine open shops with no-wait in process. Naval Research Logistics, Vol. 52, 261-275, (2005).

Johnson, S., Optimal two- and three-stage production schedules with setup times included. Naval Research Logistics Quarterly, 1, 61-68, (1954).

Kalir A.A., Sarin S.C., Evaluation of the potential benefits of lot streaming in flow shop systems. International Journal of Production Economics, vol. 66, 131–142, (2000).

Kalir, A.A., Sarin, S.C., A near-optimal heuristic for the sequencing problem in multiple-batch flow shops with small equal sublots. Omega, 29, 577–584, (2001).

Kalir, A.A., Sarin, S.C., Constructing Near Optimal Schedules for the Flow shop Lot Streaming Problem with Sublot-Attached Setups. Journal of Combinatorial Optimization, 7, 23–44, (2003).

Kropp, D.H., Smunt, T.L., Optimal and heuristic models for lot splitting in a flow shop. Decision Sciences, 21 (4), 691–709, (1990).

Kumar, S., Bagchi, T.P., Sriskandarajah, C., Lot streaming and scheduling heuristics for m-machine no-wait flowshops. Computers & Industrial Engineering, 38, 149–172, (2000).

Laha, D., Sarin, S.C., A heuristic to minimize total flow time in permutation flow shop. Omega, 37 (3), 734-739, (2009).

Leonardi, S., Raz, D., Approximating total flow time on parallel machines. ACM Symposium on Theory of Computing, 29, 110-119, (1997).

Marimuthu, S., Ponnambalam, S.G., Heuristic search algorithms for lot streaming in a two-machine flowshop. International Journal of Advanced Manufacturing Technology, 27,174-180, (2005).

Martin, C., A hybrid genetic algorithm/mathematical programming approach to the multi-family flowshop scheduling problem with lot streaming. Omega, 37, 126- 137, (2009).

Nawaz, M., Encore, E.E., Ham, I., A heuristic algorithm for the m-machine, n-job flowshop sequencing problem. Omega, 11, 91-95, (1983).

Pan, Q.K., Ruiz, R., An estimation of distribution algorithm for lot streaming flow shop problems with setup times. Omega, Vol 40, pp. 166-180, (2012).

Pinedo, M.L., Scheduling Theory, Algorithms, and Systems Third Edition. Springer, 15-16, (2008).

Potts, C.N., Baker, K.R., Flow shop scheduling with lot streaming. Operations Research Letters, Vol. 8, 297-303, (1989).

Rad, S.F., Ruiz, R., Boroojerdian, N., New high performing heuristics for minimizing makespan in permutation flowshops. Omega, 37 (2), 331-345, (2009).

Sarin, S.C., Jaiprakash, P., Flow shop lot streaming. New York: Springer, 5-6, (2007).

Sen, A., Topaloglu, E., Benli, O.S., Optimal streaming of a single job in a two-stage flow shop. European Journal of Operational Research, 110, 42–62, (1998).

Sriskandarajah, C., Wagneur, E., Lot streaming and scheduling multiple products in two-machine no-wait flowshops. IIE Transactions, 31, 695–707, (1999).

Taillard, E., Some efficient heuristic methods for the flow shop sequencing problem. European Journal of Operational Research, 47, 67–74, (1990).

Turner, S., Booth, D., Comparison of heuristics for flow shop sequencing. Omega, 15 (1), 75–78, (1987).

Vickson, R.G., Optimal lot streaming for multiple products in a two-machine flow shop. European Journal of Operational Research, 85, 556–575, (1995).

Vickson, R.G., Alfredsson, B.E., Two- and three-machine flow shop scheduling problems with equal sized transfer batches. International Journal of Production Research, 30 (7), 1551–1574, (1992).

Yoon, S.H., Ventura, J.A., Minimizing the mean weighted absolute deviation from due dates in lot streaming flow shop scheduling. Computers & Operations Research, 29, 1301–1315, (2002a).

Yoon, S.H., Ventura, J.A., An application of genetic algorithms to lot streaming flow shop scheduling. IIE Transactions, 34, 779–787, (2002b).

Zhang, W., Yin, C, Liu J., Linn, R.J., Multi-job lot streaming to minimize the mean completion time in m-1 hybrid flowshops. International Journal of Production Economics, 96, 189-200, (2005).