YAŞAR UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

MASTER THESIS

AN ENERGY-EFFICIENT PERMUTATION FLOWSHOP SCHEDULING PROBLEM

Fatma Talya TEMİZCERİ

Thesis Advisor: Prof. Dr. M. Arslan ÖRNEK
Co-Advisor: Prof. Dr. M. Fatih TAŞGETIREN

Department Of Industrial Engineering

Presentation Date: 15.08.2018

BORNOVA / İZMİR
JULY 2018

We certify that, as the jury, we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Jury Members: **Signature:**

Prof. Dr. Mustafa Arslan ÖRNEK
Yaşar University
....

Asst. Prof. Dr. Adalet ÖNER
Yaşar University
....

Prof. Dr. Şeyda TOPALOĞLU YILDIZ
Dokuz Eylül University
....

Prof. Dr. Cüneyt GÜZELİŞ
Director of the Graduate School

ABSTRACT

AN ENERGY-EFFICIENT PERMUTATION FLOWSHOP SCHEDULING

Temizceri, Fatma Talya
Msc, Industrial Engineering
Advisor: Prof. Dr. M. Arslan ÖRNEK
Co-Advisor: Prof. Dr. M. Fatih TAŞGETİREN
July 2018, 57 pages

In this thesis, to solve permutation flowshop scheduling problem (PFSP), a bi-objective mixed integer linear programming model with the objectives of minimizing the total energy consumption (TEC) and makespan is proposed in order to see the trade-off between them. Heuristic algorithms; iterated greedy (IG_{ALL}) algorithm, which is recently adapted in literature, and variable block insertion heuristic (VBIH) are presented. To test the performance of the algorithms, extensive experimental evaluations are carried out on the well-known benchmark suite of Taillard (Taillard, 1993).

Permutation flowshop scheduling problem is a well-known problem in literature. The permutation flowshop represents a particular case of the flowshop-scheduling problem, having as goal of an optimal schedule out of the *n!* possible sequences for *n* jobs on *m* machines on which these *n* jobs are to be processed. Thus, it is classified as complex combinatorial optimization problem. Energy consumption consideration in role of scheduling can be very seldom seen in the literature, even though many service-oriented scheduling articles and studies for PFSP have been adapted. Mostly, maximum completion time is considered as an only criterion. There is a considerable gap between makespan and energy consumption criteria. An effective way to improve energy efficiency in a production plant should address to design scheduling strategies, which aims to reduce the energy consumption of the process. Since there is a multi-objective decision model in this thesis, there is no single optimal solution, which simultaneously optimizes all the objectives. The effort of this thesis is to

effectively implement the ε–constraint method for generating the Pareto optimal solutions and the aim of the thesis is to show the trade-off between minimizing makespan and total energy consumption while providing a managerial sense where energy saving may result in reduced service level and vice versa.

The augmented-epsilon constraint method is employed for generating the Pareto optimal solution sets for small sized instances. For larger instances, the augmented epsilon-constraint method with a time limit is used on CPLEX for approximating the Pareto solution sets. As the heuristic methods, a very recent iterated greedy algorithm (IG_{ALL}) and an energy-efficient variable block insertion heuristic (VBIH) algorithm are proposed with employing the speed scaling strategy similar to those proposed in (Ding et al., 2016) and (Mansorui et al., 2016) from the literature. First, the performance of VBIH and IG_{ALL} algorithms on small sized problems are given, then, it is shown that the VBIH and IG_{ALL} algorithms are extremely effective for solving larger instances when compared to the time-limited CPLEX.

Key Words: permutation flowshop scheduling, makespan, energy efficient scheduling, multi-objective optimization, heuristic optimization, iterated greedy algorithm, variable block insertion algorithm.

ÖZ

ENERJİ ETKİN PERMÜTASYON AKIŞ TİPİ ÇİZELGELEME

Temizceri, Talya
Yüksek Lisans Tezi, Endüstri Mühendisliği Bölümü
Danışman: Prof. Dr. M. Arslan ÖRNEK
Yardımcı Danışman: Prof. Dr. M. Fatih TAŞGETİREN
Temmuz 2018, 57 sayfa

Bu çalışmada, iki amaçlı bir permütasyon akış tipi çizelgeleme problemi (PATÇP) ele alınmış ve bu iki amaç arasındaki değişimin görülebilmesi için toplam enerji tüketimini ve maksimum tamamlanma zamanını en aza indirecek hedefler için iki amaçlı karışık tamsayılı doğrusal programlama modeli önerilmiştir. Literatürde çok sayıda çok amaçlı PATÇP'nin sunulmasına rağmen, bu problemin enerji tüketimi açısından değerlendirilmesi çok nadirdir. Enerji-etkin akış tipi çizelgeleme probleminde, küçük boyutlu problemler üretilmiştir ve Pareto optimal çözüm setlerini üretmek için epsilon kısıtlama yöntemi (AUGMECON) kullanılmıştır. Daha büyük boyutlu problemler için ise, CPLEX üzerinde belirlenmiş zaman sınırı ile epsilon-kısıtlama yöntemi kullanılarak Pareto çözüm setlerine yaklaşılmıştır. Çözüm yöntemi olarak, Iteratif açgözlü algoritması (IG_{ALL}) ve değişken blok yerleştirme (VBIH) algoritması kullanılmıştır. IG_{ALL} ve VBIH algoritmalarının performansları ilk olarak küçük boyutlu, daha sonra, büyük boyutlu problemler üzerinde denenmiştir. IG_{ALL} ve VBIH algoritması küçük boyutlu problemleri kolayca çözebilmektedir. Bu iki algoritmanın, büyük boyutlu problemleri çözmek için, zaman-sınırlı CPLEX ile karşılaştırıldığında son derece etkili olduğu gösterilmiştir.

Anahtar Kelimeler: permütasyon akış tipi çizelgeleme, toplam üretim süresi enerji etkin çizelgeleme, çok amaçlı eniyileme, sezgisel eniyileme, döngülü (iteratif) açgözlü algoritması, değişken blok yerleştirme algoritması

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Mustafa Arslan Örnek for the continuous support of my MSc study, for his guidance, encouragement, and immense knowledge.

I would also like to thank my thesis co-advisor Prof. Mehmet Fatih Taşgetiren for my thesis and related researches, for his patience, motivation, and support. The door to Prof. Taşgetiren office was always open whenever I had a question about my research.

Besides my advisors, I would like to thank the rest of my thesis committee: Prof. Şeyda Topaloğlu Yıldız and Asst. Prof. Adalet Öner, for their insightful comments and encouragement, but also my sincere thanks goes to Dr. Efthymia Staiou for her professional, precious and also mother-like support; and to Asst. Prof. Mahmut Ali Gökçe for his valuable comments and recommendations.

I want to express the deepest appreciation to my soulmate Ozan Sudabaş for his patience and encouragement, to my sisters, Şeyda Özcan and Gizem Sağım, also to my very close friends and officemates, Gülce Haner and Syed Shah Sultan Mohiuddin Qadri for their invaluable suggestions, and being so continuously supportive.

I specially acknowledge my colleague, Hande Öztop for her support and , stimulating discussions.

Last but not the least; I must express my very profound gratitude and enduring love to my parents for providing me with continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them and their precious support. Thank you.

Fatma Talya TEMİZCERİ

İzmir, 2018

TEXT OF OATH

I declare and honestly confirm that my study, titled "AN ENERGY-EFFICIENT PERMUTATION FLOWSHOP SCHEDULING" and presented as a Master's Thesis, has been written without applying to any assistance inconsistent with scientific ethics and traditions. I declare, to the best of my knowledge and belief, that all content and ideas drawn directly or indirectly from external sources are indicated in the text and listed in the list of references.

Fatma Talya TEMİZCERİ

………………………………..

September 6, 2018

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INDEX OF SYMBOLS AND ABBREVIATIONS

Symbols	**Explanations**
N	Set of jobs ($i, k \in N = \{1, \ldots, \|N\|\}$)
M	Set of machines ($j \in M = \{1, \ldots, \|M\|\}$)
L	Set of speed levels ($l \in L = \{1, \ldots, \|L\|\}$)
D	A very large number
p_{ij}	Processing time of job i on machine j
v_l	Speed factor of speed l
λ_l	Processing conversion factor for speed l
φ_j	Conversion factor for idle time on machine j
τ_j	Power of machine j
y_{ijl}	$\begin{cases} 1 \text{ if job i is processed at speed l on machine j} \\ 0 \text{ otherwise} \end{cases}$
x_{ik}	$\begin{cases} 1 \text{ if job i precedes job k} \\ 0 \text{ otherwise } (i < k) \end{cases}$
C_{ij}	Completion time of job i on machine j
θ_j	Idle time on machine j
π, π_i	Job permutation, i^{th} job of permutation
$C(i,k)$	Completion time of π_i on machine k

C_{max}	Makespan (Maximum Completion Time of Jobs)
P	Set of problem instances
s_i	Individual solution
$s(\pi, v)$ or $s(\pi_i, v_i)$	Solution of job permutation with speed levels
$s^D(\pi_i, v_i)$	Partial solution
$s^R(\pi_i, v_i)$	Destruction sequence
$f(\pi)$	Objective function value of the permutation
Ω	Archive set
x	Archive size
r	Uniform random number in $U(0, 1)$
CR	Crossover probability
MR	Mutation probability
b	Block size
$bMove()$	Block insertion move procedure
τP	Temperature for acceptance criterion
$s^b(\pi_i, v_i)$	Block to be removed
$s^p(\pi_i, v_i)$	Partial solution after removal
R_p	Ratio of the Pareto-optimal solutions
IGD	Inverted generational distance
I	Non-dominated solution set
$d(v, I)$	Minimum Euclidean distance between v and I

DS	Distribution spacing
d_i	Minimum Euclidean distance between solution i and its closest neighbour in I
C	Coverage of two sets
NP	Population size
CD	Crowding distance

Abbreviations:

FSP	Flowshop scheduling problem
PFSP	Permutation Flowshop Scheduling Problem
TEC	Total Energy Consumption
MILP	Mixed Integer Linear Programming
IG_{ALL}	Iterated Greedy Algorithm
DC	Destruction and construction procedure
VBIH	Variable Block Insertion Heuristic
EE_VBIH	Energy-efficient VBIH algorithm
AUGMECON	Augmented ε-constraint method
EE_PFSP	Energy efficient permutation flowshop scheduling problem
TLM_CMAX	Time Limited Solutions of Cmax on CPLEX
TLM_TEC	Time Limited Solutions of TEC on CPLEX

CHAPTER 1
INTRODUCTION

One of the important decision-making processes in both manufacturing and service industries is production scheduling. Considerable improvements in productivity, reduction in cost and time can be achieved by efficient production schedules.

In the classical flowshop scheduling problem (FSP), we are given a set of n jobs that are to be processed on m machines. Each job consists of m operations, and each operation is to be performed by one of the m machines. In the FSP, all the jobs follow the same route, meaning that the operations that comprise a job are always performed in the same order. In its simplest form, it is usually assumed that all the jobs are ready at time zero, and that preemptions are not allowed, which means that if the process of a job has started in a given machine, it cannot be interrupted. While all these restrictions also hold for the PFSP, in the latter it is also assumed that the queues in front of the machines operate under a FIFO discipline, and therefore, the order in which the jobs are to be processed is the same for each machine. Under this simplifying assumption, solving the PFSP consists of finding a solution $\pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$ representing the solution in which the jobs will be processed by the system so that a given performance measure is optimized. The flow shop scheduling problem is normally classified as a complex combinatorial optimization problem due to the nature of the system. Generally, the aim is to minimize the makespan, mean flow time, earliness, tardiness, idle time etc.

The processing times of jobs on the machines are assumed to be deterministic and non-negative. Johnson (1954) indicated that when the number of machines is two, the optimal solution for most common criteria, which is makespan (total completion time), can be determined in FSP.

The permutation flowshop scheduling problem (PFSP) has been a widely studied research problem, mainly because of the extensive set of applications found in real industrial settings. According to Garey et al. (1976), the form of this kind of problems is known to be NP-Complete when the number of machines is greater than or equal to three and the schedules to be considered increases to $(n!)^m$. In case where the same sequence, or permutation, of jobs, is maintained throughout the production process, in other words 'only' $n!$ schedules have to be considered, flow shop classification turns into permutation flow shop (PFSP). Sayadi et al. (2010) explained the permutation flowshop as an operating discipline in flowshops where sequence changes among machines are not allowed.

More formally, given an arbitrary solution π, job π_i is the job at the i^{th} position of solution π. Let $C(i,k)$ be the completion time of job π_i on machine k at position i. Following this notation, completion times of jobs at each machine are computed as in equations (1) to (4), where $p_{\pi_{i,k}}$ be the processing time of job π_i at the k^{th} machine in the system. The makespan for the solution π, denoted as $C_{max}(\pi)$, is the completion time of the last job in the solution (i.e., n) on the last machine (i.e., m). It is simply denoted as $C(n,m)$ and computed as follows:

$$C(1,1) = p_{\pi_{1,1}} \tag{1}$$

$$C(i,1) = C(i-1,1) + p_{\pi_{i,1}} \quad \forall i = 2,\dots,n \tag{2}$$

$$C(1,k) = C(1,k-1) + p_{\pi_{1,k}} \quad \forall k = 2,\dots,m \tag{3}$$

$$C(i,k) = \max\{C(i-1,k), C(i,k-1)\} + p_{\pi_{i,k}} \quad \forall i = 2,\dots,n; \forall k = 2,\dots,m \tag{4}$$

For a better understanding, a Gantt chart is presented in Fig.1.1 for a simple PFSP example which has 5 jobs and 2 machines. The processing times of jobs are $p_{ij} = \begin{bmatrix} 4 & 1 & 5 & 2 & 5 \\ 3 & 2 & 4 & 3 & 6 \end{bmatrix}$ where p_{ij} is the processing time of job i on machine j.

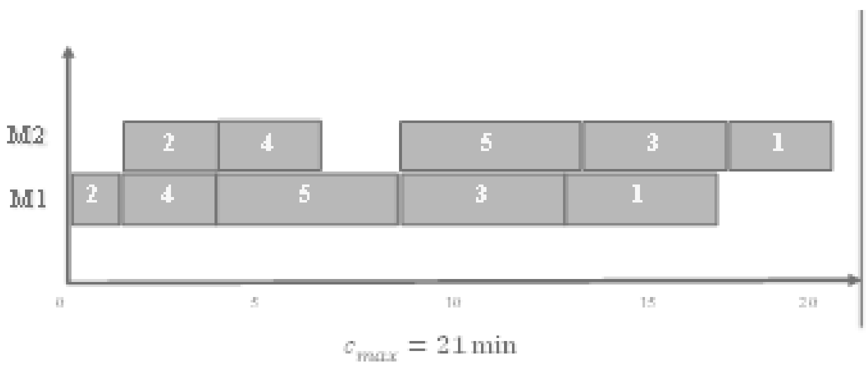

Figure 1.1. A Gantt Chart for a PFSP Example

The objective of minimizing the makespan is a general aim whilst in real life many scheduling problems are multi-objective. A massive amount of energy usage and worldwide greenhouse gas emission, manufacturers have started to investigate solutions in order to reduce energy consumption and carbon footprints in the production process. Turning on and off have been intended by manufacturers. Although it can be a solution to reduce energy consumption, this strategy may not be applicable to all manufacturing processes. With the growing attention to global warming; sustainability, green scheduling, and reduction in energy consumption have become popular concepts. Nevertheless, researches and studies on these concepts have rarely been adopted in literature. In order to fill the gap in the literature, we consider the problem of minimizing makespan and total energy consumption in m machines. Since the PFSP is an NP-Hard problem when $m \geq 3$, and solving the problem with multiple objectives require much more computational effort and time. It will be impractical to solve the problem optimally with MILP for large instances. Heuristic or meta-heuristic algorithms can provide a better solution with acceptable time consumption for computation. Well-known instances of Taillard (1993) are used for computations and analysis. We argue that a trade-off will occur between optimizing makespan and energy consumption by obtaining the Pareto optimal set between two objectives. Thus, analysing and evaluating the trade-off in an efficient way will support the decision makers when scheduling the operations in the manufacturing process.

There have been many studies in programming formulations for flowshop problems in literature. Tseng et al. (2004) concluded that Manne's model provided dichotomous constraint approach, which was superior to other assignment problem approaches of Wagner's all-integer, and Wilson's mixed integer programming (MILP). (Manne, 1960; Pan, 1997; Wagner, 1959; Wilson, 1989). Manne's model is used for problem-solving method (see Manne, 1960).

1.1. Research Goal

In this thesis, the aim is to minimize the makespan, as a means to maximize the utilization rate of the machines, which maximizes the throughput of the system, and at the same time total energy consumption (TEC). As per the makespan criterion, the PFSP has been proven in different studies to be NP-hard in the strong sense for several objective functions by Johnson (1954), Pan (1997) and Sayadi et al. (2010) respectively, and it is still very difficult to solve with exact methods. Therefore, heuristic algorithms should be employed to obtain near-optimal solutions. A bi-objective mixed integer linear programming model for the objectives of minimizing the total energy consumption and the makespan in order to see the trade-off between them is proposed.

1.2. Methodology

In accordance with the research goal, this thesis presents a multi-objective mixed integer-programming model (MIP) which is developed for the PFSP using speed-scaling strategy. The speed-scaling strategy can be described as a job-based strategy which is the same for all machines and for each machine a different speed scaling is employed. As the PFSP is NP-hard, some toy instances are developed from literature by reducing their sizes. Then, these toy instances are solved with CPLEX to find Pareto-optimal solution sets. For larger instances, the time-limited CPLEX method is employed to obtain non-dominated solutions as a heuristic method. On the other hand, as heuristic algorithms, a multi-objective energy efficient iterated greedy (IG_{ALL}) algorithm, which is proposed recently as a variant of traditional IG algorithms by Stützle et al. (2017), and a very recent variable block insertion heuristic (VBIH) algorithm by Tasgetiren et al. (2017) are proposed. Because we thought IG_{ALL} algorithm could be inadequate, we searched for a new solution. VBIH algorithm was used to verify the results that we found by IG_{ALL}. The performances of VBIH and IG_{ALL} with CPLEX are compared.

This thesis is organized as follows: Literature reviews on permutation flowshop scheduling, energy consumption in scheduling and multi-objective optimization, along with the scope of the thesis are provided in Chapter 2. Analysis of the mathematical model for permutation flow shop scheduling problem and details of heuristic algorithms are given in Chapter 3, and Chapter 4, respectively. Computational results are presented in Chapter 5. Lastly, Conclusion and Future Research are summarized in Chapter 6.

CHAPTER 2
AN ENERGY-EFFICIENT PERMUTATION FLOWSHOP SCHEDULING PROBLEM

According to Pinedo (2008), a flowshop can be explained if there are *n* jobs and *m* machines in series such that all jobs have to be processed on each machine and to follow the same route. Every job queues up for the next machine after completion on one machine. Permutation flowshop is referred if the First in First out (FIFO), which is one of the operating disciplines, is in effect and if flow shops that do not allow sequence changes between machines where that sequence or permutation of jobs is maintained throughout. On the other hand, Aghezzaf and Landeghem (2002) indicated that finding a sequence for given *n* jobs with the same orders at *m* machines in accordance with a certain performance measure is defined as the flowshop-scheduling problem.

In the production scheduling literature, tardiness and flow time-based performance measures have been generally discussed in order to measure production efficiency and customer satisfaction. However, energy efficiency in production scheduling has been rarely considered. Recently, the energy consumption has become a key concern for manufacturing sector because of the negative environmental impacts such as gas emissions (CO2) and global warming. As the manufacturing facilities consume high energy, they are forced to reduce their energy consumption by developing more energy efficient scheduling systems (Fang et al., 2011).

Gahm et al. (2016) outlined an energy efficient scheduling framework for manufacturing companies by classifying three dimensions of energy efficient scheduling approaches such as energetic coverage, energy supply and energy demand. As a pioneering study, it was concluded that once machines are turned off at idle times, a considerable amount of energy can be saved (Mouzon et al., 2007). Later on, this turn off strategy was employed in single machine scheduling problem that minimizes total energy consumption and total tardiness in (Mouzon and Yildirim, 2008). Similarly, turn off strategy was employed for the flexible flowshop problem in (Dai et al, 2013). Even though the turn off strategy is good at providing energy savings, it may not be suitable for some shop floors.

Dai et al. (2011) first developed a speed scaling strategy for the energy-efficient FSP due to the inefficiency of the turn off strategy. They considered operation speed as an independent variable that can be adjusted to improve energy efficiency. Later, a MIP formulation was given in (Dai et al., 2013), for the PFSP considering makespan as an objective and peak power consumption as a constraint. In addition, Ding et al. (2016) used the speed scaling strategy for the PFSP that minimizes the total carbon emissions and the makespan.

Recently, a multiobjective genetic algorithm was presented by Zhang and Chiong (2016) in order to minimize the total weighted tardiness and total energy consumption in a job shop scheduling. Mansouri et al. (2016) studied variable speed levels for the two machines PFSP with sequence-dependent setup times and proposed some lower bounds as well as a heuristic. In addition, an energy efficient permutation flowshop scheduling using backtracking algorithm was developed by Lu et al. (2017) whereas in (Yin et al., 2016), an energy efficient evolutionary algorithm for single machine scheduling with sequence-dependent setup times.

This thesis presents a multi-objective IG_{ALL} and VBIH algorithm for the energy-efficient PFSP considering speed scaling strategy. Note that similar speed scaling strategy was used in (Ding et al., 2016) and (Mansouri et al., 2016). In fact, from these two notable papers was an inspiration. In these two works, they employed a matrix representation for speed scaling strategy. In other words, for each machine, a different speed scaling strategy is employed. However, this thesis employs a simple job-based speed scaling strategy where the same speed strategy is used on all machines.

CHAPTER 3
BI-OBJECTIVE MATHEMATICAL MODEL FOR ENERGY EFFICIENT PERMUTATION FLOWSHOP

The preceding chapter provides insights into mixed integer linear programs as mathematical modelling for flowshops and extension for energy efficiency. As mentioned before, a regular permutation flowshop scheduling problem has such two major components as a set of *N* jobs and set of *M* machines. Accordingly the objective is to obtain a schedule that minimizes a specific performance criterion, e.g., makespan. Tseng et al. (2004) conducted an empirical analysis for four different integer-programming models to evaluate their relative effectiveness of the regular permutation flowshop problems. These competing models were compared according to their computer solution times based on the complexity of the problem.

3.1. Terminology

As mentioned in the previous chapters, the problem is a multi-objective problem, since there are two objectives which are conflicting with each other. For that reason, dominance relation concepts indicated by Deb (2001) are presented when solving the PFSP.

Dominance relation. In multi-objective minimization problems, a feasible solution π_i dominates another solution π_j if the two following conditions are satisfied (denoted as $\pi_i \succ \pi_j$):

- $\forall p \in 1,..,P; f_p(\pi_i) \leq f_p(\pi_j)$
- $\exists p \in 1,..,P; f_p(\pi_i) < f_p(\pi_j)$

A solution π_i weakly dominates another solution π_j (denoted as $\pi_i \preccurlyeq \pi_j$) if :

- $\forall p \in 1,..,P; f_p(\pi_i) \leq f_p(\pi_j)$

A solution π_i is indifferent to another solution π_i (denoted as $\pi_i \sim \pi_j$) if :

• $\forall p \in 1,..,P; f_p(\pi_i) \nleq f_p(\pi_j) \wedge f_p(\pi_j) \nleq f_p(\pi_i)$

Non-dominated set. Amongst a set of solutions S, the non-dominated set of solutions are the elements of the set S^* non-dominated by any element of the set S.

Pareto-optimal set. The non-dominated set of the entire feasible search space I is called the Pareto-optimal solutions.

In this chapter, we formulate the problem as a bi-objective PFSP examining a trade-off between C_{max} and TEC. The notation is given in Table 3.1.

Table 3.1. Notation

Indexes	
l	Index of speed levels $(l \in L = \{1, \dots, \lvert L\rvert\})$
j	Index for machines $(j \in M = \{1, \dots, \lvert M\rvert\})$
i and k	Index for jobs $(i, k \in N = \{1, \dots, \lvert N\rvert\})$
Parameters	
p_{ij}	Processing time of job i on machine j
v_l	Speed factor of speed l
λ_l	Processing conversion factor for speed l
φ_j	Conversion factor for idle time on machine j
τ_j	Power of machine j (kW)
D	A very large number
Decision Variables	
y_{ijl}	1 if job i is processed at speed l on machine j; 0,otherwise
x_{ik}	1 if job i precedes job k; 0 otherwise $(i < k)$
C_{ij}	Completion time of job i on machine j
θ_j	Idle time on machine j
C_{max}	Maximum completion time (makespan)
TEC	Total energy consumption (kWh)

The bi-objective MILP formulation is given below:

$$Min\ C_{max}, TEC \quad (1)$$

s.t.

$$C_{i1} \geq \sum_{l \in L} \frac{p_{i1} y_{i1l}}{v_l} \quad (1 \leq i \leq N) \quad (2)$$

$$C_{ij} - C_{i,j-1} \geq \sum_{l \in L} \frac{p_{ij} y_{ijl}}{v_l} \quad (2 \leq j \leq M; 1 \leq i \leq N) \quad (3)$$

$$C_{ij} - C_{kj} + D x_{i,k} \geq \sum_{l \in L} \frac{p_{ij} y_{ijl}}{v_l} \quad (1 \leq j \leq M; 1 \leq i < k \leq N) \quad (4)$$

$$C_{ij} - C_{kj} + D x_{ik} \leq D - \sum_{l \in L} \frac{p_{kj} y_{kjl}}{v_l} \quad (1 \leq j \leq M; 1 \leq i < k \leq N) \quad (5)$$

$$C_{max} \geq C_{iM} \qquad (1 \leq i \leq N) \qquad (6)$$

$$\sum_{l \in L} y_{ijl} = 1; \qquad (1 \leq i \leq N; 1 \leq j \leq M) \qquad (7)$$

$$y_{ijl} = y_{i,j+1,l} \qquad (1 \leq i \leq N; 1 \leq j < M; 1 \leq l \leq L) \qquad (8)$$

$$\theta_j = C_{max} - \sum_{i=1}^{N} \sum_{l \in L} \frac{p_{ij} y_{ijl}}{v_l} \qquad (1 \leq j \leq M) \qquad (9)$$

$$TEC = \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{l \in L} \frac{p_{ij} \tau_j \lambda_l}{60 v_l} y_{ijl} + \sum_{j=1}^{M} \frac{\varphi_j \theta_j \tau_j}{60} \qquad (10)$$

The objective function (1) minimizes C_{max} and TEC. Constraint (2) ensures that the completion time of each job on machine 1 is greater than or equal to its processing time on machine 1. Constraint (3) states that a job can start only after its preceding operation has been completed. Constraints (4) and (5) guarantee that job i either precedes job k or vice versa in the sequence, but not both. Constraint (6) computes the maximum completion time of all jobs on the last machine, in other words, computes the makespan. Constraint (7) and (8) ensure that exactly one speed level is selected for each job and the same speed level is employed on every machine. Constraint (9) calculates the idle time of each machine and constraint (10) computes the total energy consumption as proposed in (Mansouri et al., 2016).

There are common solution methods for multi-objective problems such as sequential optimization, goal programming, weighting method and ε-constraint method. In this thesis, augmented ε-constraint method was preferred to use, as it generates only Pareto-optimal solutions (Mavrotas, 2009). In the augmented ε-constraint method, one of the objective functions is optimized, while other objective functions are defined by constraints.

Dissimilar to the standard ε-constraint method, slack/surplus variables are included in these objective function constraints by converting them to equalities. These variables are also defined as the second term in the objective function to ensure the Pareto-optimality. In Pareto-optimal solutions, any objective function cannot be improved without worsening another objective function.

Trade-off between $Cmax$ and TEC in CPLEX results for the first instances of small-sized problems (5x5, 5x10, 5x20) given in Fig. 3.1, Fig. 3.2 and Fig. 3.3, respectively.

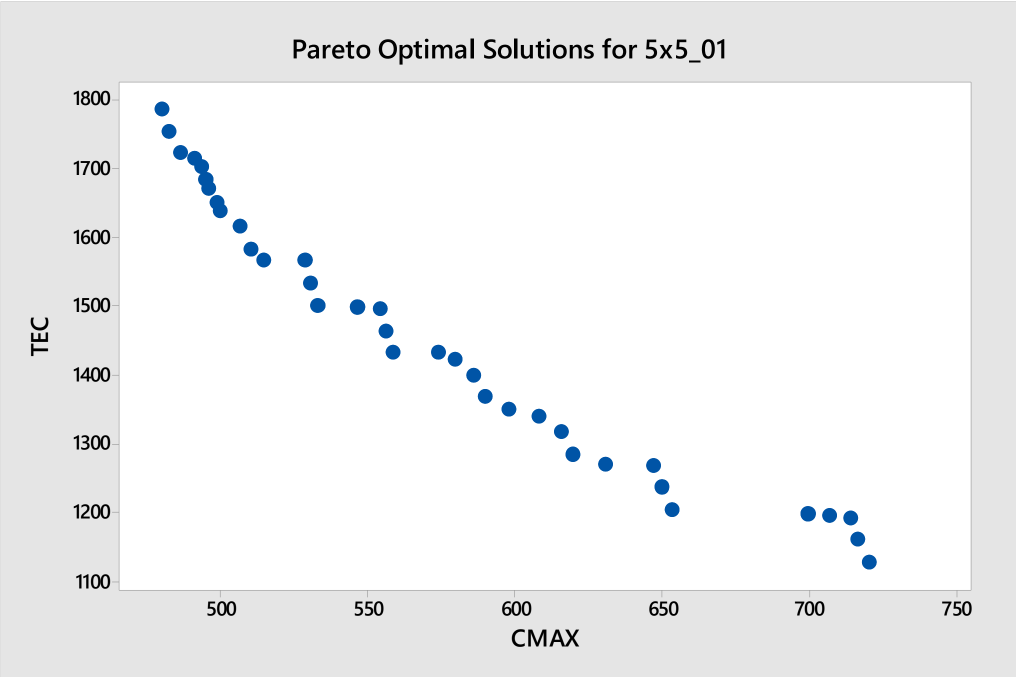

Figure 3.2. Pareto-Optimal Solutions for the instance 5x5_01

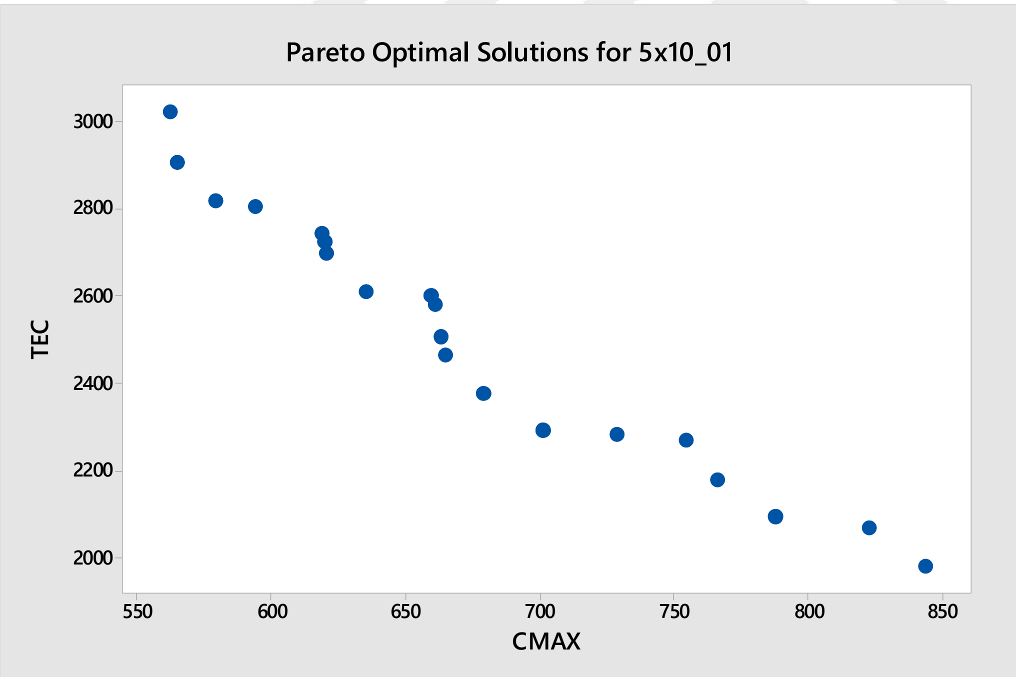

Figure 3.3. Pareto-Optimal Solutions for the instance 5x10_01

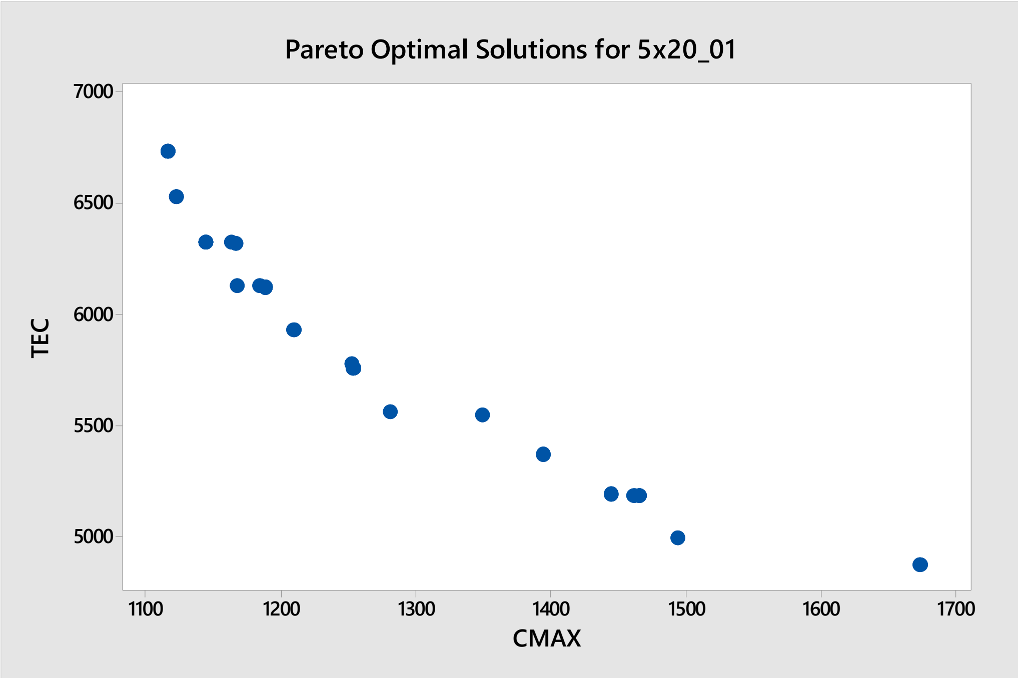

Figure 3.4. Pareto-Optimal Solutions for the instance 5x20_01

Because of the exponentially increasing solution times, non-dominated solution sets are found for larger instances using a relatively higher ε level, which is calculated by dividing the range of TEC objective function to 20 equal grids. 3 minutes time limit is set for these large instances (20x5, 20x10, 20x20, 50x5, 50x10 and 50x20) in each iteration, because the problem has the NP-hard nature.

As an expected consequence of time-limit, non-dominated solution set is smaller than true Pareto-optimal solutions. Non-dominated solutions of the first instances of 20x5 and 50x5 are presented as an example in Fig.3.4 and Fig.3.5 respectively.

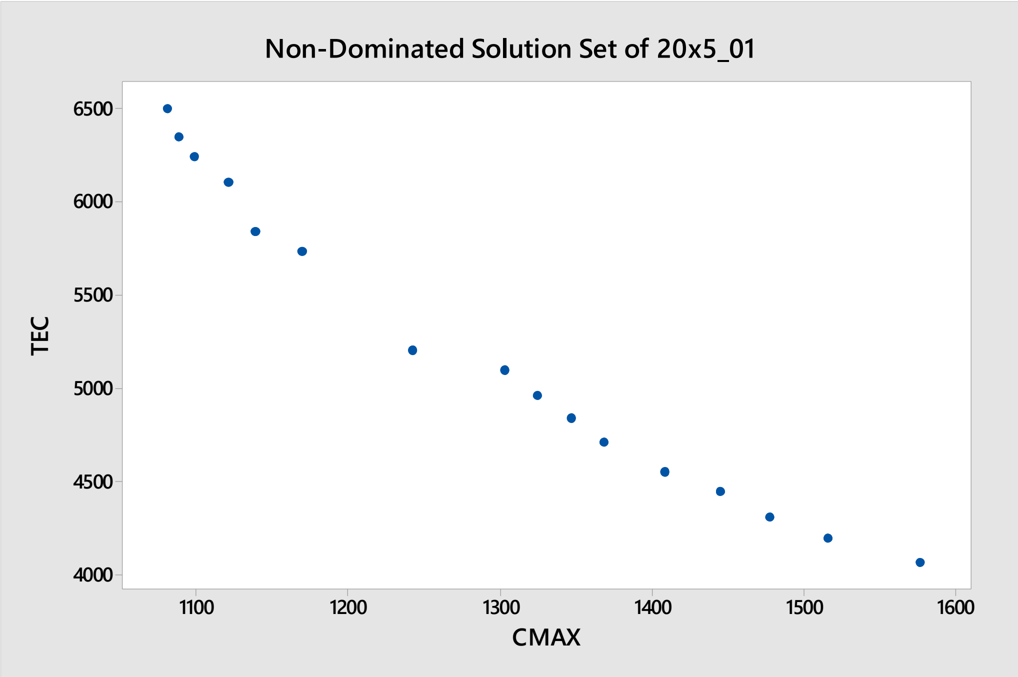

Figure 3.5. Non-Dominated Solution Set of instance 20x5_01

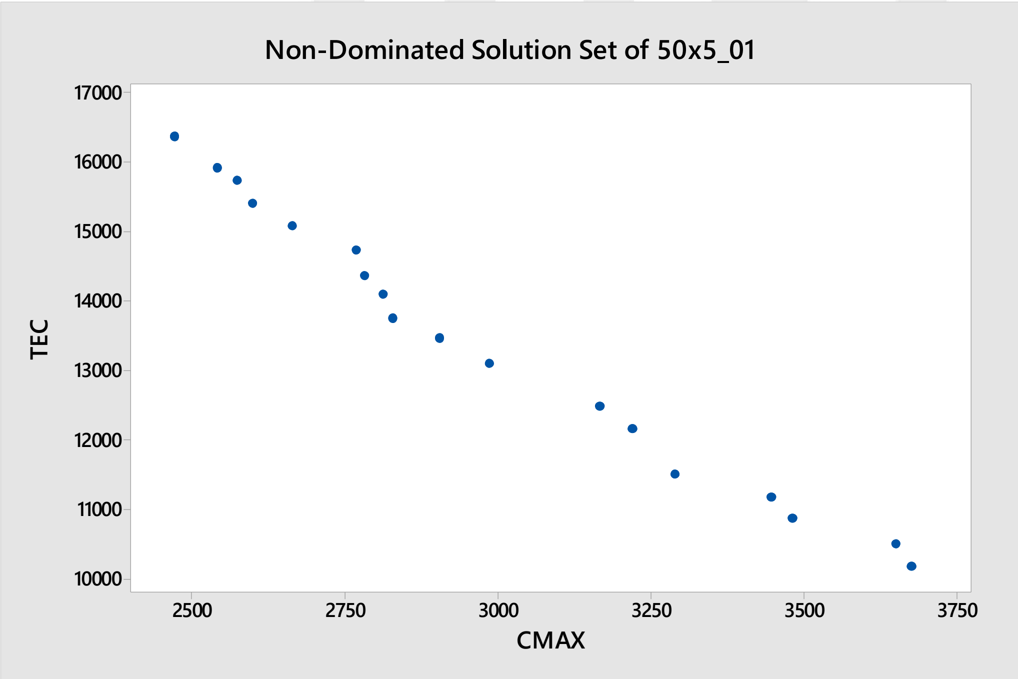

Figure 3.6. Non-Dominated Solution Set of instance 50x5_01

CHAPTER 4
HEURISTIC ALGORITHMS

4.1. Energy Efficient IG_{ALL} Algorithm

The IG algorithm is presented in Ruiz and Stützle (2007). In the traditional IG algorithm, the NEH insertion heuristic is employed to obtain the initial solution (Nawaz et al., 1983). Then, destruction and construction (DC) procedure is used to generate offspring solutions in a way that a number d of jobs is randomly removed from the current solution to be re-inserted into partial solution, sequentially. Afterwards, an insertion based local search is applied to the solution after the DC procedure. An acceptance criterion is used to accept the new solution after a local search. These simple steps are repeated until a stopping criterion is satisfied.

Recently, an IG_{ALL} algorithm is presented for the PFSP with makespan minimization in the literature. Unlike the traditional IG algorithm, the IG_{ALL} algorithm applies an additional local search to partial solutions after destruction, which substantially improves the solution quality. The pseudo-code of the IG_{ALL} algorithm is given in Fig. 4.1. and details can be found in (Dubois-Lacoste et al., 2017). In the following subsections, essential components of the proposed energy-efficient IG_{ALL} algorithm are outlined.

Procedure IG_{ALL}
π_0 = GenerateInitial Solution
π = LocalSearch (π_0)
do
  π' = Destruction (π, d)
  π' = ApplyLocalSearchToPartialSolution (π')
  π' = Construction (π')
  π'' = ApplyLocalSearchToCompleteSolution (π')
  π = AcceptionCriterion (π'', π)
while (Termination criterion is met)
endprocedure

Figure 4.1. IG_{ALL} Algorithm

4.1.1. Solution Representation

As mentioned before, a job-based speed scaling strategy is proposed for the energy-efficient IG_{ALL} algorithm. To handle speed scaling strategy, a multi-chromosome structure is used and it is composed of a permutation of n jobs and a speed vector with three levels. There exist three speed levels that correspond to fast, normal and slow speed levels, respectively. The solution representation is given in Fig. 4.2.

$s(\pi_i, v_i)$							
π	5	2	1	4	3	…	n
v	3	1	2	1	2	…	3

Figure 4.2. Solution Representation for IG_{ALL} Algorithm

In Fig.4.2, a solution/individual $s(\pi_i, v_i)$ indicates that the first job $\pi_1 = 5$ has a slow speed level, $v_1 = 3$; second job $\pi_2 = 2$ has a fast speed level, $v_2 = 1$; and so on. It is worth stating that the same speed vector is used on all machines for a specific job in PFSP.

4.1.2. Initial Population

For the initial the population with size NP, the following procedure is used: A solution is constructed by the NEH heuristic (see Nawaz et al., 1983). The NEH heuristic can be outlined in Fig. 4.3.

Step1. s = DecreasingOrder ($\sum_{j=1}^{m} p_{ij}$) and $\pi_1 = s_1$
Step2. For ($i = 2\ to\ n$) do
Remove job π_i from α
Test it in all positions in α
Insert π_i in s with the lowest makespan
Step3. EndFor
Step4. Return π
endprocedure

Figure 4.3. NEH Constructive Heuristic

Now, the procedure for constructing initial population can be summed up as follows:

- Use the NEH solution as an initial solution for the IG_{ALL} algorithm with the makespan minimization only.

- Devote 10% of the total CPU time budget to IG_{ALL} algorithm to find the best solution π_{best}.

- Keep π_{best} and assign fast, normal and slow speed levels to each job in π_{best} and construct the first three individuals in the population.

- For the rest of population, keep π_{best} and assign random speed levels between 1 and 3 to each job in π_{best}.

- Update the archive set Ω.

4.1.3. Destruction and Construction Procedure

The procedure is summarized as follows:

- Remove d jobs with their speed levels randomly from the solution $s(\pi, v)$ without repetition.

- Apply insertion local search to the partial solution as shown in Fig. 4.4.

- Assign random speed levels $v_i = rand()\%l$, $\forall i \in 1,..,NP$ and $j \in 1,..,d$ to these d jobs.

- Reinsert these d jobs with their new speed levels into the partial solution sequentially until a complete solution of n jobs with their speed levels is obtained.

- In each step, while inserting the jobs, the number of places to be inserted should be increased by 1. Use dominance rule ($\succ$) when two solutions and/or partial solutions are evaluated.

In order to clarify the DC procedure, the following example with 5 jobs and 5 speed levels is given below:

$s(\pi_i, v_i) = \{(5,1),(2,1),(1,2),(4,3),(3,2)\}$.

Suppose that we remove job 2 and job 4 with their speed levels. Then, we have two partial solutions:

$s^R(\pi_i, v_i) = \{(2,1),(4,3)\}$ and $s^D(\pi_i, v_i) = \{(5,1),(1,2),(3,2)\}$.

The DC procedure randomly changes the speed levels of each job,

$s^R(\pi_i, v_i) = \{(2,3),(4,2)\}$ and $s^D(\pi_i, v_i) = \{(5,1),(1,2),(3,2)\}$.

Now, the DC procedure in IG_{ALL} algorithm applies an insertion local search to the partial solution $s^D(\pi_i, v_i)$ first. Suppose that we have the following solution after the local search as follows:

$s^D(\pi_i, v_i) = \{(1,2),(5,1),(3,2)\}$.

Now, the DC procedure inserts job and speed $(2,3)$ into four positions in $s^D(\pi_i, v_i)$ and four partial solutions are obtained as follows:

$s^D(\pi_i, v_i) = \{(2,3),(1,2),(5,1),(3,2)\}$,

$s^D(\pi_i, v_i) = \{(1,2),(2,3),(5,1),(3,2)\}$,

$s^D(\pi_i, v_i) = \{(1,2),(5,1),(2,3),(3,2)\}$, and

$s^D(\pi_i, v_i) = \{(1,2),(5,1),(3,2),(2,3)\}$.

Suppose that the partial non-dominated solution is the last one amongst these four solutions. Then, again, the DC procedure inserts job and speed (4,2) into five positions in s^D and five complete solutions are obtained as follows:

$s^D(\pi_i, v_i) = \{(4,2),(1,2),(5,1),(3,2),(2,3)\}$,

$s^D(\pi_i, v_i) = \{(1,2),(4,2),(5,1),(3,2),(2,3)\}$,

$s^D(\pi_i, v_i) = \{(1,2),(5,1),(4,2),(3,2),(2,3)\}$,

$s^D(\pi_i, v_i) = \{(1,2),(5,1),(3,2),(4,2),(2,3)\}$, and

$s^D(\pi_i, v_i) = \{(1,2),(5,1),(3,2),(2,3),(4,2)\}$.

Suppose that the last complete solution is the non-dominated solution and it is chosen by the DC procedure for the individual s_i in the population.

4.1.4. Makespan Minimization

For the makespan minimization, we employ a very effective first improvement insertion local search given in Fig. 4.4. The local search is carried out for each individual in the population and it can be summarized as follows:

- Remove job and speed (π^*, v^*) from position j of the solution $s(\pi_i, v_i)$.
- Assign a new speed level for job by $v^* = rand()\%3$

- Insert removed job π^* and speed v^* into all possible positions of the incumbent solution.

- Find the best insertion position, which dominates the incumbent solution.

- Insert job π^* and speed v^* into that position and update the archive set Ω.

- Repeat them for all the job and speed pairs. If any non-dominated solution is found, invoke the local search again until no non-dominated solution is obtained.

Note that local search is similar to the example given in previous subsection. Instead of removing two jobs with their speeds, the local search removes only one job with its speed.

```
for j = 1 to n do
      Remove π* and v*. Assign v* = rand()%3
      s*(π*, v*) = InsertInBestPosition(s_i, (π*_i, v*_i))
       if (f(s*(π*, v*)) ≻ f(s_i(π_i, v_i))) then do
            s_i(π_i, v_i) = s*(π*, v*)
            Update archive set Ω with s*(π*, v*)
        end if
  end for
```

Figure 4.4. Insertion Local Search

4.1.5. Energy Minimization

The energy-efficient IG_{ALL} algorithm given above is extremely effective for makespan minimization. However, energy-efficient schedules should be obtained too. For this reason, a local search algorithm is proposed based on uniform crossover operator for speed levels only. In other words, after applying energy-efficient IG_{ALL} algorithm to each individual in the population, the same solution/permutation is kept for each individual in the population and make a uniform crossover on speed levels as follows:

- For each individual s_i in the population, select another individual from population randomly, say s_k,

- Generate offspring by making a uniform crossover as follows:

$$s_{new}(\pi_i, v_i) = \begin{cases} \pi_i(v_i) & if\ r \leq CR[i] \\ \pi_k(v_k) & otherwise \end{cases}$$

where r is a uniform random number in $U(0,1)$ and $CR[i]$ is the crossover probability, which is drawn from unit normal distribution $N(0.5,0.1)$. If s_{new} dominates s_i (i.e., $s_{new} \succ s_i$), s_i is replaced by s_{new}. Then, the archive set Ω is updated. This is repeated for all individuals in the population.

After crossover local search, the speed levels of jobs are mutated with a small mutation probability as follows:

$$s_i(\pi_i, v_i) = \begin{cases} s_i\left(v_{ij} = rand()\%3\right) & if\ r \leq MR[i] \\ s_i(v_i) & otherwise \end{cases}$$

where r is a uniform random number in $U(0,1)$ and $MR[i]$ is the mutation probability, which is drawn from unit normal distribution $N(0.05,0.01)$ for each individual s_i in the population.

In order to clarify the crossover an example is provided. The uniform crossover operator is given with individual s_i ,

$s_i(\pi, v) = \{(4,3),(1,2),(5,1),(3,2),(2,1)\}$, and s_k which is randomly chosen from the population, $s_k(\pi, v) = \{\ (1,2),(4,3),(5,1),(3,2),(2,1)\}$. Note that we keep the permutation of individual s_i in the offspring solution s_{new} and make the crossover on the speed levels as follows:

$$s_{new}(\pi, v) = \{\ (4,3),(1,3),(5,1),(3,2),(2,1)\}.$$

4.2. Energy-Efficient VBIH Algorithm

Recently, block move-based search algorithms were presented for scheduling problems in literature (Subramanian et al., 2014; Xu et al., 2014; González et al., 2017; Tasgetiren et al., 2016; Tasgetiren et al., 2016; Tasgetiren et al., 2017). The VBIH algorithm simply removes a block b of jobs from the current solution; then it makes a number $n - b + 1$ of block insertion moves on the partial solution denoted as $bMove()$ procedure. Then, the best one from the $bMove()$ procedure is retained in order to undergo a local search procedure. If the new solution obtained after the local search is better than the current solution, it replaces the current solution. Otherwise, a simple simulated annealing-type of acceptance criterion is used with a constant temperature, which is suggested in (Osman and Potts, 1989), as follows:

$$T = \frac{\sum_{i=1}^{n}\sum_{k=1}^{m} p_{ik}}{10nm} \times \tau P,$$

where τP is a parameter to be adjusted and r is a uniform random number in $U(0,1)$. Initially, the block size is fixed to $b = 1$. As long as it improves, it retains the same block size ($i.e., b = b$). Otherwise, it is increased by one ($i.e., b = b + 1$). The $bMove()$ procedure is carried out until the block size reaches at the maximum block size (i.e., $b \leq b_{max}$). The outline of the VBIH algorithm for a minimization problem is given in Fig. 4.5.

```
π = NEH
π_best = π
while (NotTermination) do
   b = 1
   do{
        π_1 = bMove(π,)
        π_1 = LocalSearch(π_1)
        if (f(π_1) ≤ f(π)) then do{
               π = π_1
                if f(π_1) < f(π_best)  then do{
                     π_best = π_1
                   b = b
        else{
              b = b + 1
               if (r < exp{−(f(π) − f(π_1))/T})
                      π = π_1
       }endif
   }while(b ≤ b_max)
}endwhile
return π_best and f(π_best)
endprocedure
```

Figure 4.5. Variable Block Insertion Heuristic

In this thesis, a job-based speed scaling strategy is also proposed for the energy-efficient VBIH algorithm. To handle this speed-scaling strategy, a multi-chromosome structure is used. It is composed of a permutation of n jobs (π) and a speed vector of three levels (v) corresponding to fast, normal and slow speed levels. The solution representation is given in Fig. 4.6.

$s(\pi, v)$	π	6	3	2	1	4	5	…	n
	v	1	2	1	2	1	2	…	3

Figure 4.6. Solution Representation for VBIH Algorithm

In Fig.4.6, a solution/individual $s(\pi, v)$ indicates that job $\pi_1 = 6$ has a fast speed level,($i.e.,\ v_1 = 1$), job $\pi_2 = 3$ has a normal speed level, ($i.e., v_2 = 2$); and so on.

4.2.1. Initial Population

For the initial population with size NP, the following procedure is used: A solution is constructed by the NEH heuristic. This solution is taken as an initial solution for the VBIH algorithm with makespan minimization only. Ten percent of the total CPU time budget is devoted to the VBIH algorithm in order to obtain a good starting point for the Energy Efficient Variable Block Insertion (EE_VBIH) algorithm. Once the best solution s_{best} is found by the VBIH algorithm, the first three solutions in population are obtained by assigning fast, normal or slow speed levels to each job in the best solution s_{best}. The rest of the population is obtained by assigning random speed levels between 1 and 3 to each job in the best solution s_{best}. The archive set Ω is initially empty and it is updated.

4.2.2. Block Insertion Procedure

The $bMove()$ procedure is a core function in the EE_VBIH algorithm. The procedure randomly removes a block b of jobs with their speed from the current solution. Then, block is denoted by s^b whereas the partial solution after removal will be denoted by $s^p(\pi_i, v_i) = \left(N - s^b(\pi_i, v_i)\right)$. First, speed levels in $s^b(\pi_i, v_i)$ are randomly changed between 1and 3. Then; similar to the one presented in (Dubois-Lacoste et al, 2017), the EE_VBIH algorithm applies an additional local search to partial solution s^p before carrying out a block insertion. Then; the $bMove()$ procedure carries out $n - b + 1$ block insertion moves. In other words, block s^b is inserted in all possible positions in the partial solution s^p. It should be noted that dominance rule ($\succ$) explained before is used when two solutions and/or partial solutions are compared.

In order to explain the $bMove()$ procedure, following example would be useful. Suppose that we have a current solution $s(\pi_i, v_i) = \{(3,2), (1,1), (4,3), (2,1), (5,2)\}$ with block size $b = 2$. A block is removed and two partial solutions are obtained as follows:

$s^b(\pi_i, v_i) = \{(1,1),(4,3)\}$ and $s^p(\pi_i, v_i) = \{(3,2),(2,1),(5,2)\}$.

First, speed levels of $s^b(\pi_i, v_i)$ are randomly changed to, say,

$s^b(\pi_i, v_i) = \{(1,3),(4,2)\}$. Then; an insertion local search is applied to the partial solution $s^p(\pi_i, v_i)$ in a way that each job and speed pair is removed from $s^p(\pi_i, v_i)$ and inserted into all positions without the position it is removed. The best non-dominated partial solution is retained.

Suppose that the best one is $s^p(\pi_i, v_i) = \{(5,2),(2,1),(3,2)\}$. Finally, the block $s^b(\pi_i, v_i)$ is inserted into all positions in $s^p(\pi_i, v_i)$ as follows:

$s(\pi_i, v_i) = \{(\mathbf{1},\mathbf{3}),(\mathbf{4},\mathbf{2}),(5,2),(2,1),(3,2)\}$,

$s(\pi_i, v_i) = \{(5,2),(\mathbf{1},\mathbf{3}),(\mathbf{4},\mathbf{2}),(2,1),(3,2)\}$,

$s(\pi_i, v_i) = \{(5,2),(2,1),(\mathbf{1},\mathbf{3}),(\mathbf{4},\mathbf{2}),(3,2)\}$, and

$s(\pi_i, v_i) = \{(5,2),(2,1),(3,2),(\mathbf{1},\mathbf{3}),(\mathbf{4},\mathbf{2})\}$.

Among these four solutions, the non-dominated one is selected with respect to min C_{max} and the archive set Ω is updated.

4.2.3. Energy Efficient Insertion Local Search

Regarding the local search algorithm, a very effective first-improvement insertion neighbourhood structure is employed for each individual s_i in the population. Similar to the $bMove()$ procedure, each job and speed level is removed from the current solution and inserted into all positions of the current solution. The non-dominated solution is retained and the archive set Ω is updated. The insertion local search has a size of $(n-1)^2$.

As an example, we consider the solution in previous subsection $s(\pi_i, v_i) = \{(3,2),(1,1),(4,3),(2,1),(5,2)\}$. The first job and its speed level, $(3,2)$ are removed from the current solution $s(\pi_i, v_i)$. Its speed level is randomly changed to another speed level, say, $(3,1)$. Then; it is inserted into all positions in the solution $s(\pi_i, v_i)$ as follows:

$s(\pi_i, v_i) = \{(3,1),(1,1),(4,3),(2,1),(5,2)\}$,

$s(\pi_i, v_i) = \{(1,1),(3,1),(4,3),(2,1),(5,2)\}$,

$s(\pi_i, v_i) = \{(1,1),(4,3),(3,1),(2,1),(5,2)\}$,

$s(\pi_i, v_i) = \{(1,1),(4,3),(2,1),(3,1),(5,2)\},$

$s(\pi_i, v_i) = \{(1,1),(4,3),(2,1),(5,2),(3,1)\}.$

Among these five solutions, the non-dominated one is selected with respect to min C_{max} and the archive set Ω is updated. This is repeated for the next pair of job and its speed level until the last job and its speed level are inserted into all positions.

4.2.4. Energy-Efficient Uniform Crossover and Mutation

In order to obtain more energy-efficient schedules, a local search algorithm based on uniform crossover operator by considering only speed levels is again proposed for the EE_VBIH algorithm, too. Note that with the same permutation, any change in speed levels leads to a different solution in terms of C_{max} $and\ TEC$. For this reason, after having applied the VBIH algorithm to each individual in the population, the same permutation is kept for each individual in the population and a uniform crossover on speed levels is carried out as follows. The similar procedure with the previous subsection is followed.

For each individual s_i in the population, we select another individual from population randomly, say s_k, a new solution is obtained in a way that taking the speed level is either taken from s_i or s_k with a crossover probability $CR[i]$. The uniform crossover is carried out as follows:

$$s_{new}(\pi_i, v_i) = \begin{cases} s_i(v_i) & if\ r \leq CR[i] \\ s_k(v_k) & otherwise \end{cases} \qquad \forall j \in 1,..,n$$

where r is a uniform random number in $U(0,1)$ and $CR[i]$ is the crossover probability, which is drawn from unit normal distribution $N(0.5,0.1)$ for each individual s_i in the population. If s_{new} dominates s_i (i.e., $s_{new} \succ s$), s_i is replaced by s_{new} and the archive set Ω is updated. This is repeated for all individuals in the population.

After having carried out uniform crossover for all individuals in the population, we mutate the population by lowering the speed levels with a small mutation probability as follows:

$$s_i(\pi_i, v_i) = \begin{cases} s_i(v_{ij} = 1 + rand()\%2) & if\ r \leq MR[i] \\ s_i(v_{ij}) & otherwise \end{cases} \forall j \in 1,..,n;\ \forall i \in 1,..,NP$$

where r is a uniform random number in $U(0,1)$ and $MR[i]$ is the mutation probability, which is drawn from unit normal distribution $N(0.05,0.01)$ for each individual s_i in the population.

4.2.5. The Archive Set

An archive set Ω is used to store the non-dominated solutions during the optimization process. This archive set should be updated with non-dominated solutions in order to approximate the Pareto-optimal solutions. When a new non-dominated solution is obtained, it should be added to the archive set Ω and any member dominated by the new non-dominated solution should be removed.

4.2.5.1. Update Archive Set

In order to update the archive set Ω, Pan et.al. (2009) proposed an effective method for updating the archive set as follows: The non-dominated solutions in Ω are stored in increasing order of their first objective function values. Then, their second objective values will be in decreasing order. The procedure for updating the archive set Ω can be summarized as follows:

Step 1. Archive size is $x = |\Omega|$ and $\Omega = \{a_1, a_2, .., a_x\}$. Initially, Ω is empty and the first non-dominated solution s will be added to the first position in Ω. Let $j = k = 1$.

Step 2. Find a most suitable position pos for the next individual s in the archive set Ω by the following procedure:

$do\{$

$j = \lfloor (k + x)/2 \rfloor$

$if\ \left(f_1(s) = f_1(a_j)\right)\ then\ j = j$

$elseif\ \left(f_1(s) < f_1(a_j)\right)\ then\ x = j - 1$

$else\ \ k = j + 1$

$while(k \leq x)$

Step 3. When comparing $f_1(s)$ with $f_1(a_j)$, following cases may occur:

$Case$ 1. $if\ \left(f_1(s) = f_1(a_j)\right)\ and\ \ if\ \left(f_2(s) < f_2(a_j)\right) then\ pos = j$

$Case$ 2. $if\ \left(f_1(s) < f_1(a_j)\right)$

$if\ j = 1\ then\ pos = j\ and\ x = x + 1$

$if\ j > 1\ and\ \left(f_2(s) < f_2(a_{j-1})\right)\ then\ pos = j\ and\ x = x + 1$

$Case$ 3. $if \left(f_1(s) > f_1(a_j)\right)\ and\ if \left(f_2(s) < f_2(a_j)\right) then\ pos = j\ and\ x =$ $x + 1$

If any of cases above is satisfied, solution s is added to position pos, but all solutions dominated by s in Ω should be removed. The following procedure removes the dominated solutions from Ω:

Step 1. $If\ (pos = x)\ then\ go\ to\ Step\ 4$

Step 2. $Let\ pos = pos + 1.$

$If\ f_2(a_{pos}) \geq f_2(s)\ then\ remove\ a_{pos}; otherwise\ go\ to\ Step\ 4$

Step 3. $if (pos < x)\ then\ go\ to\ Step\ 2$

Step 4. $\Omega = non - dominated\ solutions$

4.2.5.2. Crowding Distance

For a solution in Ω, the crowding distance is the sum of the normalized distance between its previous and next neighbors for each objective function value. The extreme solutions have the crowding distance set to infinity. It is clear that the larger the crowding distance, the sparser the nearby solutions. Based on the storage structure of Ω, the crowding distance of a non-dominated solution a_j is given as follows:

$$CD_j = \begin{cases} \infty & if\ (j = 1\ or\ j = s) \\ \dfrac{f_1(a_{j+1}) + f_1(a_{j-1})}{f_1(a_s) - f_1(a_1)} + \dfrac{f_2(a_{j-1}) + f_2(a_{j+1})}{f_2(a_1) - f_1(a_s)} & otherwise \end{cases}$$

CHAPTER 5

COMPUTATIONAL RESULTS

In this thesis, a novel mathematical model and two different algorithms are proposed to solve energy efficient permutation flow shop scheduling problem. IG_{ALL} and VBIH algorithm, as mentioned in Chapter 4, are applied to energy efficient permutation flowshop scheduling problem from the literature. All instances for the mathematical model are solved with the augmented ε-constraint method using IBM ILOG CPLEX 12.6.3 on a Core i7, 2.60 GHz, 8 GB RAM computer. The IG_{ALL} and VBIH algorithm are coded in C++ programming language on Microsoft Visual Studio 2013 and all instances are solved on a Core i5, 3.20 GHz, 8 GB RAM computer. In order to test the performance of the IG_{ALL} and VBIH algorithms, extensive experimental evaluations are carried out on the well-known benchmark suite of Taillard (1993). The benchmark set is composed of 12 groups of the given problems with the size ranging from 20 jobs and 5 machines to 500 jobs and 20 machines, and each group consists of ten instances. Only the first 60 instances from 20 jobs and 5 machines to 50 jobs and 20 machines were employed (20x5, 20x10, 20x20, 50x5, 50x10 and 50x20).

Initially, the ranges of each objective are obtained from payoff tables using lexicographic optimization, i.e. first C_{max} is minimized, then TEC is minimized. In addition, due to the computational difficulty of the bi-objective problem, we generate 30 small-sized instances with 5 jobs and 5 machines, 5 jobs and 10 machines, 5 jobs and 20 machines by truncating 20x5, 20x10 and 20x20 problems. Population size is taken as NP=100. C_{max} is minimized subject to TEC. Afterwards, the single-objective model is solved repetitively by reducing the constraint on TEC with a specific ε level.

The speed and conversion parameters of (Mansouri et al., 2016) are used in TEC computation. There are three processing speed factors $v_l = \{1.2, 1, 0.8\}$ and 3 conversion factors $\lambda_l = \{0.6, 1, 1.5\}$ corresponding to 3 speed levels (slow, normal and fast, respectively). The power of machines are the same (60 kW) and the conversion factor for idle time is 0.05.

For each instances thirty replications with IG_{ALL} and five replications VBIH algorithms are made. In each replication, both IG_{ALL} and VBIH algorithms are run for $10|N||M|$ milliseconds for small-sized instances and $30|N||M|$ milliseconds for larger instances, where $|N|$ is the number of jobs and $|M|$ is the number of machines.

For VBIH algorithm, it is important to note that initially the archive size is set to $x = 5 \times NP$ in each replication. After five replications, only non-dominated solutions in Ω are kept because a solution in a replication can dominate a solution in another replication. Due to the real values of objective functions, as many as non-dominated solutions are generated after five replications. However, the crowding distances of all these solutions are computed and only the most crowded solutions are reported up to $s = 100$.

Very close approximations is obtained for the Pareto-optimal frontiers of instances with 5 jobs (5x5, 5x10 and 5x20) choosing an ε level as 10^{-3}. These finite numbers of Pareto-optimal solutions are named as Pareto-optimal solution set (P). Appendix 1 gives the mathematical model solutions of the small instances explained above while Appendix 2 and Appendix 3 present the IG_{ALL} and VBIH results, respectively.

Since very close approximations to Pareto-optimal frontiers for instances with 5 jobs, below performance measures are used to evaluate the solution quality of the IG_{ALL} and VBIH algorithms. I refers to the non-dominated solution set of the IG_{ALL} and VBIH algorithms.

- Ratio of the Pareto-optimal solutions found

$R_p = |I \cap P| / |P|$

- Inverted Generational Distance (Coello et al., 2002)

$IGD = \sum_{v \in P} d(v, I) / |P|$, where $d(v, I)$ indicates the minimum Euclidean distance between v and the solution in I. The low IGD value means that set I is very close to set P.

• Distribution Spacing (Tan et al, 2006)

$$\text{DS} = \frac{\left[\frac{1}{|I|}\sum_{i\in I}(d_i - \bar{d})^2\right]^{1/2}}{\bar{d}} \quad \text{where } \bar{d} = \frac{\sum_{i\in I} d_i}{|I|}$$

d_i is the minimum Euclidean distance between solution i and its closest neighbour in I. Low spacing value shows that the solutions in I are uniformly distributed.

Due to the exponentially increasing solution times, non-dominated solution sets are found for larger instances using a relatively higher ε level, which is calculated by dividing the range of TEC objective function to 20 equal grids. Due to the NP-hard nature of the problem, 3 minutes time limit is set in each iteration for these large instances (20x5, 20x10, 20x20, 50x5, 50x10 and 50x20). Appendix 4 and Appendix 7 give the mathematical model results as an example for the large instances (first instance of 20 and 50 jobs), respectively. Additionally, IG_{ALL} results for the first instance of 20 and 50 jobs are given in Appendix 5 and Appendix 8, while Appendix 6 and Appendix 9 provide VBIH results for the first instance of 20 and 50 jobs, respectively.

For large instances, non-dominated solution sets of IG_{ALL} and VBIH algorithms (*I*) and time-limited CPLEX (*T*) are compared with each other in terms of the below performance metrics and the aforementioned DS metric.

• Cardinality: the number of non-dominated solutions found.

• Coverage of Two Sets for a minimization problem (*C*) (Zitzler, 1999)

$C_{IT} = |t \in T;\ \exists i \in I: i \preccurlyeq t\}|/|T|$ where C_{IT} equals 1 if some solutions of I weakly dominate all solutions of T.

5.1. Computational Results of IG_{ALL} Algorithm

For the IG_{ALL} with makespan minimization only, destruction size and temperature for acceptance criterion are taken as $d = 4$ and $\tau = 0.4$.

For small-sized problems we generated, Pareto optimal solution set of mathematical model and IG_{ALL} algorithm are given for the first instance of 5 jobs 5 machines in Fig. 5.1 as an example. As seen in Fig. 5.1, IG_{ALL} is able to find all Pareto optimal solutions.

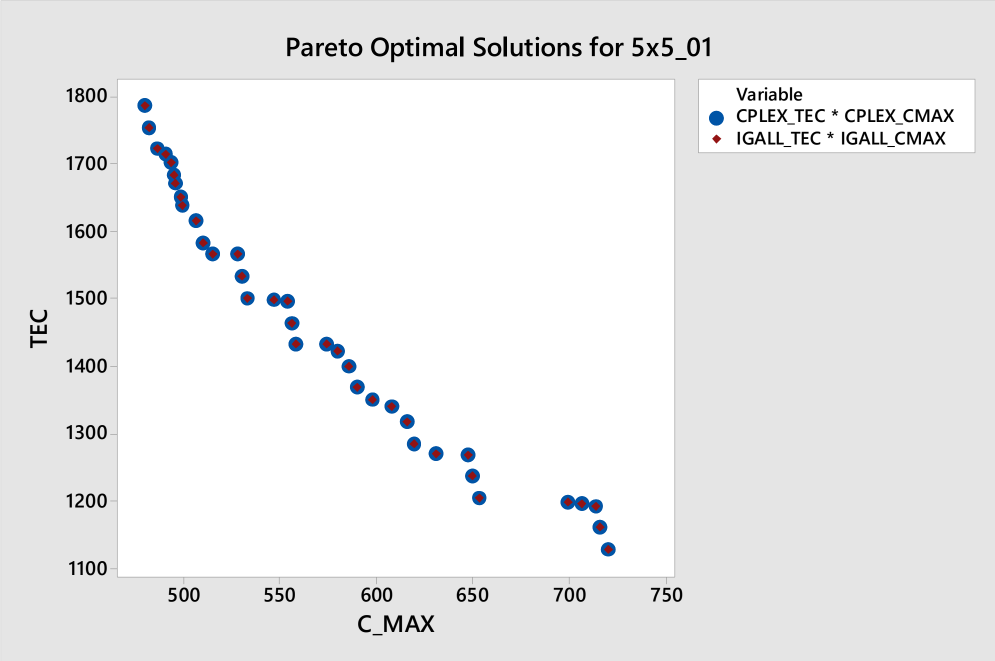

Figure 5.1. Pareto Optimal Solution Set of 5x5_01 with Mathematical Model vs IG_{ALL}

Table 5.1 reports the comparison of R_p, IGD and DS measures for IG_{ALL} algorithm and mathematical model results on small-sized instances. As shown in the table, IG_{ALL} algorithm finds approximately 82% of the Pareto-optimal solutions. Especially, for eight instances, all Pareto-optimal solutions are found by IG_{ALL} algorithm. Furthermore, average IGD value of IG_{ALL} algorithm is very low (0.0003), which indicates that the IG_{ALL} provides very close approximations to the Pareto-optimal solution sets. In terms of distribution spacing, we can say that solutions in I are evenly distributed due to the low DS value.

Table 5.1. Comparison of IG_{ALL} and Mathematical Model on Small Sized Instances

				CPU Time (sec)	
Instance Set	**Rp**	**IGD**	**DS**	**CPLEX**	**IG_{ALL}**
5x5	0.9820	0.00003	0.7000	4.04	2.52
5x10	0.8170	0.00023	0.8346	3.96	5.01
5x20	0.6360	0.00055	0.8902	6.06	10.02
Average	**0.8117**	**0.00027**	**0.8083**	**4.68**	**5.85**

As an example for large instances, the non-dominated solutions of 20x5_01 for IG_{ALL} and time limited CPLEX are given in Fig. 5.2 IG_{ALL} algorithm was superior to the time limited CPLEX and many new non-dominated solution are found.

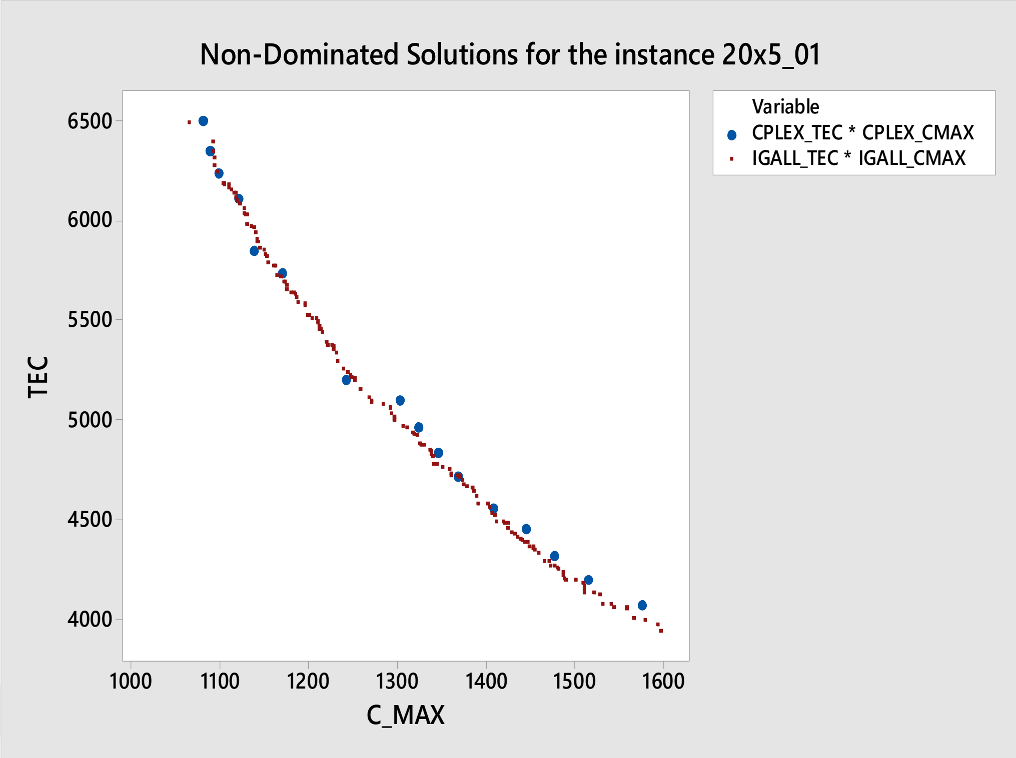

Figure 5.2. Non-dominated Solution Set of 20x5_01 with Mathematical Model vs IG_{ALL}

Table 5.2 reports the average results for T and I on large instances. As shown in the table, IG_{ALL} generates approximately eight times as many non-dominated solutions in very reasonable computation times. Furthermore, IG_{ALL} performs much better than the time-limited CPLEX in terms of coverage metric, since 78% of the solutions of T are weakly dominated by some solutions of I. Particularly, some solutions of I weakly dominate all solutions of the T , in 22 of the instances. In terms of distribution spacing, solutions in T are distributed more uniformly than the solutions in I, as a fixed ε level is employed through the augmented ε-constraint method in time-limited CPLEX.

Table 5.2. Comparison of IG_{ALL} and Mathematical Model on Large Instances

							CPU Time (sec)	
Instance Set	**|T|**	**|I|**	**C_{TI}**	**C_{IT}**	**DS_T**	**DS_I**	**CPLEX**	**IG_{ALL}**
20x5	18.300	175.100	0.039	0.826	0.182	1.541	3600	3.45
20x10	19.000	145.100	0.900	0.089	0.162	2.035	3600	6.11
20x20	16.090	110.900	0.007	0.896	0.299	1.831	3600	12.27
50x5	14.400	149.800	0.000	0.897	0.416	4.904	3600	8.02
50x10	9.000	104.400	0.001	0.981	0.737	4.546	3600	16.98
50x20	6.900	86.800	0.000	1.000	0.940	4.366	3600	40.81
Average	**13.948**	**128.683**	**0.158**	**0.782**	**0.456**	**3.204**	**3600**	**14.61**

5.2. Computational Results of VBIH Algorithm

For the VBIH with makespan minimization only, the maximum block size is taken as $b_{max} = 5$; and temperature for acceptance criterion are taken as $\tau P = 0.4$.

Pareto optimal solution set of mathematical model and VBIH algorithm is given as an example for the first instance of 5 jobs 5 machines for small sized problems; and VBIH algorithm was able to find all Pareto optimal solutions as shown in Fig. 5.3.

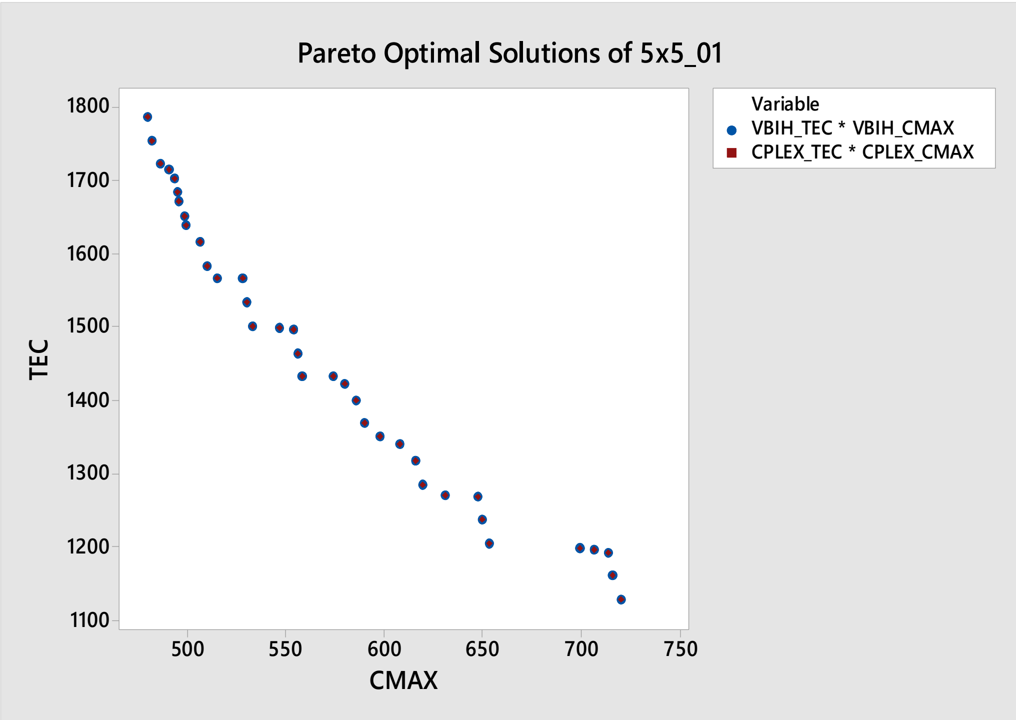

Figure 5.3. Pareto Optimal Solution Set of 5x5_01 with Mathematical Model vs VBIH

Table 5.3 presents the average results of R_p, IGD and DS measures for each small-sized instance set, where there are 10 instances in each set. As shown in the table, the EE_VBIH algorithm finds approximately 82% of the Pareto-optimal solutions. Especially, for eight instances, all Pareto-optimal solutions are found by EE_VBIH algorithm. Furthermore, average IGD value of EE_VBIH algorithm is very low (0.00027), which indicates that the EE_VBIH provides very close approximations to the Pareto-optimal solution sets. In terms of distribution spacing, we can say that solutions in I are evenly distributed due to the low DS value.

Table 5.3. Comparison of VBIH and Mathematical Model on Small Sized Instances

				CPU Time (sec)	
Instance Set	**Rp**	**IGD**	**DS**	**CPLEX**	**VBIH**
5x5	0.9820	0.00003	0.7000	4.04	0.31
5x10	0.8170	0.00023	0.8346	3.96	0.55
5x20	0.6360	0.00055	0.8902	6.06	1.05
Average	**0.8117**	**0.00027**	**0.8083**	**4.68**	**0.64**

The non-dominated solutions of 20x5_01 for VBIH and time limited CPLEX instances are given as an example for large instance in Fig. 5.4. It is seen on Fig. 5.4, VBIH algorithm is superior to the time limited CPLEX and many new non-dominated solution are found.

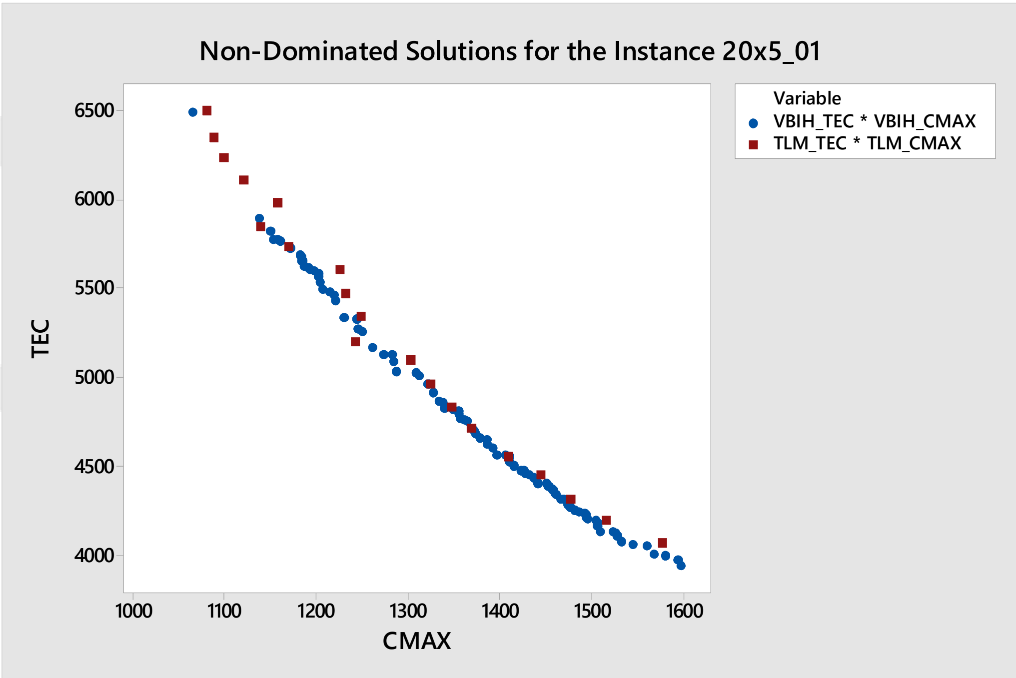

Figure 5.4. Non-dominated Solution Set of 20x5_01 with Mathematical Model vs VBIH

Table 5.4 presents the average results for *T* and *I* for each large instance set, where there are 10 instances in each set. As shown in the table, EE_VBIH generates approximately seven times as many non-dominated solutions in very reasonable computation times. Furthermore, EE_VBIH performs much better than the time-limited CPLEX in terms of coverage metric, since 85% of the solutions of *T* are weakly dominated by some solutions of *I*. Particularly, some solutions of *I* weakly dominate all solutions of the *T* , in 17 of the instances. In terms of distribution spacing, solutions in *T* are distributed more uniformly than the solutions in *I*, as a fixed ε level is employed through the augmented ε-constraint method in time-limited CPLEX.

Table 5.4. Comparison of VBIH and Mathematical Model on Large Instances

Instance Set	|T|	|I|	C_{TI}	C_{IT}	DS_T	DS_I	CPU Time (sec) CPLEX	CPU Time (sec) VBIH
20x5	18.300	100.000	0.057	0.651	0.181	2.019	3600	3.19
20x10	16.300	93.800	0.013	0.786	0.212	2.481	3600	6.59
20x20	16.900	83.700	0.020	0.808	0.301	2.289	3600	12.72
50x5	14.400	100.000	0.000	0.866	0.415	4.428	3600	9.86
50x10	9.000	80.700	0.001	0.957	0.737	4.271	3600	20.8
50x20	6.900	67.100	0.000	1.000	0.941	4.115	3600	45.44
Average	**13.633**	**87.550**	**0.015**	**0.845**	**0.464**	**3.267**	**3600**	**16.43**

Besides comparing mathematical model with the aforementioned algorithms, following part of this chapter makes pairwise comparison of the performances of these algorithms. Machine based average comparisons of IG_{ALL} and VBIH algorithms on both small sized and large instances are presented in Table 5.5 and Table 5.6, respectively. As shown in Table 5.5, both algorithms can find exactly the same solutions with the mathematical model on small sized instances, that is, both algorithms are verified. However, VBIH is quicker than IG_{ALL} in terms of finding optimal solutions.

Table 5.5. Comparison of IG_{ALL} and VBIH on Small Instances

Algorithm	Instance Set	Rp	IGD	DS	CPU Time (sec)
IG_{ALL}	**5x5**	0.9820	0.00003	0.7000	2.52
	5x10	0.8170	0.00023	0.8346	5.01
	5x20	0.6360	0.00055	0.8902	10.02
	Average	**0.8117**	**0.00027**	**0.8083**	**5.85**
VBIH	**5x5**	0.9820	0.00003	0.7000	0.31
	5x10	0.8170	0.00023	0.8346	0.55
	5x20	0.6360	0.00055	0.8902	1.05
	Average	**0.8117**	**0.00027**	**0.8083**	**0.64**

According to performance metrics presented in Table 5.6, we can conclude that IG_{ALL} algorithm is superior to VBIH in terms of finding non-dominated solutions while coverage of VBIH is higher than IG_{ALL}. In terms of the average distribution spacing, IG_{ALL} solutions distributed more uniformly than VBIH. For large instances, IG_{ALL} performs quicker than VBIH. When efficiencies of the algorithms are compared with these performance metrics, IG_{ALL} algorithm is slightly better than VBIH algorithm.

Table 5.6. Comparison of IG_{ALL} and VBIH on Large Instances

Algorithm	|M|	|I|	C_{TI}	C_{IT}	DS_I	CPU Time (sec)
IG_{ALL}	**5**	162	0.019	0.861	2.148	5.73
	10	125	0.450	0.535	3.290	14.62
	20	96	0.003	0.948	3.098	26.54
	Average	**128**	**0.158**	**0.782**	**3.204**	**15.63**
VBIH	**5**	100	0.028	0.758	3.223	6.525
	10	87	0.007	0.871	3.376	13.69
	20	75	0.010	0.904	3.202	29.08
	Average	**87**	**0.015**	**0.845**	**3.267**	**16.43**

CHAPTER 6
CONCLUSIONS AND FUTURE RESEARCH

This thesis presents an energy-efficient PFSP which minimizes makespan and total energy consumption. A simple job-based speed scaling strategy, which is the same for all machines in the problem, is proposed. A multi-objective MILP model and two heuristic algorithms, IG_{ALL} and VBIH, are developed. Taillard's benchmarks (1993) are used to generate Pareto frontiers. This benchmark set is composed of 12 groups of the given problems with the size ranging from 20 jobs and 5 machines to 500 jobs and 20 machines, and each group consists of ten instances. Only the first 60 instances from 20 jobs and 5 machines to 50 jobs and 20 machines were employed (20x5, 20x10, 20x20, 50x5, 50x10 and 50x20). Small-sized instances were generated from to find Pareto optimal solution sets by truncating the instances of 20x5, 20x10 and 20x20.

First, the MILP model is run for these toy instances and Pareto optimal solution sets were obtained. For the larger instances, time-limited CPLEX is employed to find 20 solutions for each instance. For small sized instances, proposed algorithms, were able to find approximately 82% of the Pareto-optimal solutions. Especially, for eight instances, all Pareto-optimal solutions are found by EE_VBIH and IG_{ALL} algorithm. For larger instances, EE_VBIH generates approximately eight times as many non-dominated solutions in very reasonable computation times. Furthermore, IG_{ALL} performs much better than the time-limited CPLEX in terms of coverage metric, since the IG_{ALL} algorithm dominates 78% of the solutions of time-limited CPLEX. EE_VBIH performs much better than the time-limited CPLEX in terms of coverage metric, since the EE_VBIH algorithm dominates 85% of the solutions of time-limited CPLEX.

In particular, IG_{ALL} dominates all solutions of time-limited CPLEX in 22 out of 60 instances and EE_VBIH weakly dominates all solutions of time-limited CPLEX in 17 out of 60 instances.

For further research, the matrix representation for speed scaling strategy can be easily adapted by modifying the MILP model, IG_{ALL} and EE_VBIH algorithm. Machine-based speed scaling strategy can be used instead of job-based speed scaling strategy. If the products to be produced are in lots, lot streaming can be applied. Some multi-objective metaheuristic algorithms can be employed and different performance measures such as weighted total tardiness and total flow time criteria can be another research direction.

REFERENCES

Coello, C. A., Veldhuizen, D. A., & Lamont, G. B. (2002). MOEA Testing and Analysis. *Genetic Algorithms and Evolutionary Computation Evolutionary Algorithms for Solving Multi-Objective Problems,242*, 141-178. doi:10.1007/978-1-4757-5184-0_4

Dai, M., Tang, D., Giret, A., Salido, M. A., & Li, W. (2013). Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. *Robotics and Computer-Integrated Manufacturing,29*(5), 418-429. doi:10.1016/j.rcim.2013.04.001

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*(Vol. 16). John Wiley & Sons.

Ding, J., Song, S., & Wu, C. (2016). Carbon-efficient scheduling of flow shops by multi-objective optimization. *European Journal of Operational Research,248*(3), 758-771. doi:10.1016/j.ejor.2015.05.019

Dubois-Lacoste, J., Pagnozzi, F., & Stützle, T. (2017). An iterated greedy algorithm with optimization of partial solutions for the makespan permutation flowshop problem. *Computers & Operations Research,81*, 160-166. doi:10.1016/j.cor.2016.12.021

Fang, K., Uhan, N., Zhao, F., & Sutherland, J. W. (2011). A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. *Journal of Manufacturing Systems,30*(4), 234-240. doi:10.1016/j.jmsy.2011.08.004

Fang, K., Uhan, N. A., Zhao, F., & Sutherland, J. W. (2013). Flow shop scheduling with peak power consumption constraints. *Annals of Operations Research,206*(1), 115-145. doi:10.1007/s10479-012-1294-z

Gahm, C., Denz, F., Dirr, M., & Tuma, A. (2016). Energy-efficient scheduling in manufacturing companies: A review and research framework. *European Journal of Operational Research,248*(3), 744-757. doi:10.1016/j.ejor.2015.07.017

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The Complexity of Flowshop and Jobshop Scheduling. *Mathematics of Operations Research,1*(2), 117-129. doi:10.1287/moor.1.2.117

González, M. A., Palacios, J. J., Vela, C. R., & Hernández-Arauzo, A. (2017). Scatter search for minimizing weighted tardiness in a single machine scheduling with setups. *Journal of Heuristics,23*(2-3), 81-110. doi:10.1007/s10732-017-9325-1

Johnson, S. M. (1954). Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly,1*(1), 61-68. doi:10.1002/nav.3800010110

Lu, C., Gao, L., Li, X., Pan, Q., & Wang, Q. (2017). Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm. *Journal of Cleaner Production,144*, 228-238. doi:10.1016/j.jclepro.2017.01.011

Manne, A. S. (1960). On the Job-Shop Scheduling Problem. *Operations Research,8*(2), 219-223. doi:10.1287/opre.8.2.219

Mansouri, S. A., Aktas, E., & Besikci, U. (2016). Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption. *European Journal of Operational Research,248*(3), 772-788. doi:10.1016/j.ejor.2015.08.064

Mavrotas, G. (2009). Effective implementation of the ε-constraint method in Multi-Objective Mathematical Programming problems. *Applied Mathematics and Computation,213*(2), 455-465. doi:10.1016/j.amc.2009.03.037

Mouzon, G., Yildirim, M. B., & Twomey, J. (2007). Operational methods for minimization of energy consumption of manufacturing equipment. *International Journal of Production Research,45*(18-19), 4247-4271. doi:10.1080/00207540701450013

Mouzon, G., & Yildirim, M. B. (2008). A framework to minimise total energy consumption and total tardiness on a single machine. *International Journal of Sustainable Engineering,1*(2), 105-116. doi:10.1080/19397030802257236

Nawaz, M., Enscore, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega,11*(1), 91-95. doi:10.1016/0305-0483(83)90088-9

Osman, I., & Potts, C. (1989). Simulated annealing for permutation flow-shop scheduling. *Omega,17*(6), 551-557. doi:10.1016/0305-0483(89)90059-5

Pan, C. (1997). A study of integer programming formulations for scheduling problems. *International Journal of Systems Science,28*(1), 33-41. doi:10.1080/00207729708929360

Pan, Q., Wang, L., & Qian, B. (2009). A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems. *Computers & Operations Research,36*(8), 2498-2511. doi:10.1016/j.cor.2008.10.008

Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research,177*(3), 2033-2049. doi:10.1016/j.ejor.2005.12.009

Sayadi, M. K., Ramezanian, R., & Ghaffari-Nasab, N. (2010). A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. *International Journal of Industrial Engineering Computations,1*(1), 1-10. doi:10.5267/j.ijiec.2010.01.001

Subramanian, A., Battarra, M., & Potts, C. N. (2014). An Iterated Local Search heuristic for the single machine total weighted tardiness scheduling problem with sequence-dependent setup times. *International Journal of Production Research,52*(9), 2729-2742. doi:10.1080/00207543.2014.883472

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research,64*(2), 278-285. doi:10.1016/0377-2217(93)90182-m

Tan, K., Goh, C., Yang, Y., & Lee, T. (2006). Evolving better population distribution and exploration in evolutionary multi-objective optimization. *European Journal of Operational Research,171*(2), 463-495. doi:10.1016/j.ejor.2004.08.038

Tasgetiren, M. F., Pan, Q., Ozturkoglu, Y., & Chen, A. H. (2016). A memetic algorithm with a variable block insertion heuristic for single machine total weighted tardiness problem with sequence dependent setup times. *2016 IEEE Congress on Evolutionary Computation (CEC)*. doi:10.1109/cec.2016.7744157

Tasgetiren, M. F., Pan, Q., Kizilay, D., & Velez-Gallego, M. C. (2017). A variable block insertion heuristic for permutation flowshops with makespan criterion. *2017 IEEE Congress on Evolutionary Computation (CEC)*. doi:10.1109/cec.2017.7969382

Tasgetiren, M., Pan, Q., Kizilay, D., & Gao, K. (2016). A Variable Block Insertion Heuristic for the Blocking Flowshop Scheduling Problem with Total Flowtime Criterion. *Algorithms,9*(4), 71. doi:10.3390/a9040071

Tseng, F. T., Stafford, E. F., & Gupta, J. N. (2004). An empirical analysis of integer programming formulations for the permutation flowshop. *Omega - The International Journal of Management Science,32*(4), 285-293. doi:10.1016/j.omega.2003.12.001

Xu, H., Lü, Z., & Cheng, T. C. (2013). Iterated Local Search for single-machine scheduling with sequence-dependent setup times to minimize total weighted tardiness. *Journal of Scheduling,17*(3), 271-287.doi:10.1007/s10951-013-0351-z

Yin, L., Li, X., Lu, C., & Gao, L. (2016). Energy-Efficient Scheduling Problem Using an Effective Hybrid Multi-Objective Evolutionary Algorithm. *Sustainability,8*(12), 1268. doi:10.3390/su8121268

Zhang, R., & Chiong, R. (2016). Solving the energy-efficient job shop scheduling problem: A multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. *Journal of Cleaner Production,112*, 3361-3375. doi:10.1016/j.jclepro.2015.09.097

Zitzler, E. (1999). *Evolutionary algorithms for multiobjective optimization: Methods and applications*(Vol. 63).Ithaca: Shaker.

APPENDIX 1 – Mathematical Model Results by CPLEX for Small Sized Instances

MATHEMATICAL MODEL					
5x5_01		5x10_01		5x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
1786.2917	480.0000	3021.7500	562.5000	3486.1250	633.3333
1754.3667	482.5000	2906.7500	565.0000	3381.6500	645.0000
1722.2292	486.2500	2819.2000	579.0000	3355.8750	651.8333
1714.3833	491.0000	2806.3750	594.1667	3252.4000	665.5000
1702.3500	493.5000	2743.0000	618.7500	3151.8125	688.5000
1682.8750	495.1667	2724.6583	619.8333	3129.9000	709.8333
1670.4250	496.0000	2698.3000	620.3333	3022.6875	719.5833
1650.7375	498.9167	2611.0833	635.0000	3007.9500	726.3333
1638.2875	499.7500	2599.7583	659.5000	3003.5167	744.3333
1615.1833	506.5000	2579.6500	661.0000	2979.1583	754.0000
1583.0458	510.2500	2506.9583	663.0000	2912.2375	759.0833
1567.4208	514.9167	2463.4000	664.7500	2871.8500	760.0000
1565.9167	528.5000	2375.8500	678.7500	2769.2625	779.0000
1533.8250	530.3333	2291.0250	701.2500	2746.4750	792.7500
1501.4375	533.0833	2283.9125	728.5000	2642.8875	809.7500
1499.4958	546.6667	2268.4750	754.7500	2642.6750	860.2500
1496.4500	554.0833	2179.5500	766.0000	2639.7250	873.2500
1464.3583	555.9167	2094.4750	788.0000	2534.0125	886.0000
1431.9708	558.6667	2066.7000	822.7500	2519.3000	897.0000
1431.9083	573.9167	1981.1250	843.7500	2412.0875	906.7500
1421.9125	579.7500	3021.7500	562.5000	2408.8125	927.5000
1400.2167	586.0000	2906.7500	565.0000	2385.9000	941.0000
1368.0792	589.7500	2819.2000	579.0000	2278.3125	950.0000
1350.5500	597.7500	2806.3750	594.1667	3486.1250	633.3333
1340.3125	608.2500			3381.6500	645.0000
1316.4625	615.7500			3355.8750	651.8333
1284.3250	619.5000			3252.4000	665.5000
1270.0750	630.7500			3151.8125	688.5000
1268.1625	647.2500			3129.9000	709.8333
1236.2375	649.7500			3022.6875	719.5833
1204.1000	653.5000				
1198.7375	699.2500				
1196.3500	706.5000				
1192.1250	713.7500				
1160.2000	716.2500				
1128.0625	720.0000				

APPENDIX 2 – IG_{ALL} Results for Small Sized Instances

IG_{ALL} ALGORTIHM					
5x5_01		5x10_01		5x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
1786.2917	480.0000	3021.7500	562.5000	3486.1250	633.3333
1754.3667	482.5000	2906.7500	565.0000	3381.6500	645.0000
1722.2292	486.2500	2819.2000	579.0000	3355.8750	651.8333
1714.3833	491.0000	2806.3750	594.1667	3252.4000	665.5000
1702.3500	493.5000	2743.0000	618.7500	3151.8125	688.5000
1682.8750	495.1667	2724.6583	619.8333	3129.9000	709.8333
1670.4250	496.0000	2698.3000	620.3333	3022.6875	719.5833
1650.7375	498.9167	2611.0833	635.0000	3007.9500	726.3333
1638.2875	499.7500	2599.7583	659.5000	3003.5167	744.3333
1615.1833	506.5000	2579.6500	661.0000	2979.1583	754.0000
1583.0458	510.2500	2506.9583	663.0000	2912.2375	759.0833
1567.4208	514.9167	2463.4000	664.7500	2871.8500	760.0000
1565.9167	528.5000	2375.8500	678.7500	2769.2625	779.0000
1533.8250	530.3333	2291.0250	701.2500	2746.4750	792.7500
1501.4375	533.0833	2283.9125	728.5000	2642.8875	809.7500
1499.4958	546.6667	2268.4750	754.7500	2642.6750	860.2500
1496.4500	554.0833	2179.5500	766.0000	2639.7250	873.2500
1464.3583	555.9167	2094.4750	788.0000	2534.0125	886.0000
1431.9708	558.6667	2066.7000	822.7500	2519.3000	897.0000
1431.9083	573.9167	1981.1250	843.7500	2412.0875	906.7500
1421.9125	579.7500	3021.7500	562.5000	2408.8125	927.5000
1400.2167	586.0000	2906.7500	565.0000	2385.9000	941.0000
1368.0792	589.7500	2819.2000	579.0000	2278.3125	950.0000
1350.5500	597.7500	2806.3750	594.1667	3486.1250	633.3333
1340.3125	608.2500			3381.6500	645.0000
1316.4625	615.7500			3355.8750	651.8333
1284.3250	619.5000			3252.4000	665.5000
1270.0750	630.7500			3151.8125	688.5000
1268.1625	647.2500			3129.9000	709.8333
1236.2375	649.7500			3022.6875	719.5833
1204.1000	653.5000				
1198.7375	699.2500				
1196.3500	706.5000				
1192.1250	713.7500				
1160.2000	716.2500				
1128.0625	720.0000				

APPENDIX 3 – VBIH Results for Small Sized Instances

VBIH ALGORTIHM					
5x5_01		5x10_01		5x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
1786.2917	480.0000	3021.7500	562.5000	3486.1250	633.3333
1754.3667	482.5000	2906.7500	565.0000	3381.6500	645.0000
1722.2292	486.2500	2819.2000	579.0000	3355.8750	651.8333
1714.3833	491.0000	2806.3750	594.1667	3252.4000	665.5000
1702.3500	493.5000	2743.0000	618.7500	3151.8125	688.5000
1682.8750	495.1667	2724.6583	619.8333	3129.9000	709.8333
1670.4250	496.0000	2698.3000	620.3333	3022.6875	719.5833
1650.7375	498.9167	2611.0833	635.0000	3007.9500	726.3333
1638.2875	499.7500	2599.7583	659.5000	3003.5167	744.3333
1615.1833	506.5000	2579.6500	661.0000	2979.1583	754.0000
1583.0458	510.2500	2506.9583	663.0000	2912.2375	759.0833
1567.4208	514.9167	2463.4000	664.7500	2871.8500	760.0000
1565.9167	528.5000	2375.8500	678.7500	2769.2625	779.0000
1533.8250	530.3333	2291.0250	701.2500	2746.4750	792.7500
1501.4375	533.0833	2283.9125	728.5000	2642.8875	809.7500
1499.4958	546.6667	2268.4750	754.7500	2642.6750	860.2500
1496.4500	554.0833	2179.5500	766.0000	2639.7250	873.2500
1464.3583	555.9167	2094.4750	788.0000	2534.0125	886.0000
1431.9708	558.6667	2066.7000	822.7500	2519.3000	897.0000
1431.9083	573.9167	1981.1250	843.7500	2412.0875	906.7500
1421.9125	579.7500	3021.7500	562.5000	2408.8125	927.5000
1400.2167	586.0000	2906.7500	565.0000	2385.9000	941.0000
1368.0792	589.7500	2819.2000	579.0000	2278.3125	950.0000
1350.5500	597.7500	2806.3750	594.1667	3486.1250	633.3333
1340.3125	608.2500			3381.6500	645.0000
1316.4625	615.7500			3355.8750	651.8333
1284.3250	619.5000			3252.4000	665.5000
1270.0750	630.7500			3151.8125	688.5000
1268.1625	647.2500			3129.9000	709.8333
1236.2375	649.7500			3022.6875	719.5833
1204.1000	653.5000				
1198.7375	699.2500				
1196.3500	706.5000				
1192.1250	713.7500				
1160.2000	716.2500				
1128.0625	720.0000				

APPENDIX 4 – Mathematical Model Results for Large Instances (20jobs)

MATHEMATICAL MODEL					
20x5_01		20x10_01		20x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
6496.75	1080.83	12851.20	1361.67	26337.60	2039.50
6348.89	1088.33	12641.00	1404.83	25885.70	2058.00
6237.62	1098.50	12118.80	1480.42	25452.90	2078.33
6106.00	1121.08	11848.20	1515.00	24879.40	2097.67
5841.77	1138.42	11131.20	1552.08	23911.20	2155.33
5731.18	1169.75	10851.40	1598.08	23564.50	2215.08
5201.55	1242.42	10373.50	1685.42	23074.10	2224.50
5094.15	1302.67	10038.00	1702.58	22566.70	2322.42
4957.88	1324.33	9830.30	1747.50	22008.10	2368.08
4836.10	1347.00	9582.00	1779.50	21110.90	2406.92
4712.55	1368.50	9362.20	1803.50	20589.50	2460.58
4553.63	1408.25	9112.80	1906.83	20139.30	2580.75
4449.63	1444.67	8857.80	1914.00	19639.70	2651.00
4311.61	1477.00	8587.20	1963.50	19192.70	2695.42
4195.74	1516.00			18684.10	2733.00
4069.05	1576.25			17684.30	2840.50
				17219.70	2957.25

APPENDIX 5 – IG_{ALL} Results for Large Instances (20 jobs)

IG_{ALL} ALGORITHM					
20x5_01		20x10_01		20x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
6492.792	1065	13140.46	1319.167	26429.04	1932.5
6394.925	1091	12646.26	1407.5	26036.63	2038.667
6343.254	1091.167	12489.65	1412.917	25784.18	2040.167
6311.392	1093.917	12415.78	1415.667	25774.09	2045.833
6278.65	1094	12223.9	1423.917	25617.43	2049.333
6246.725	1096.5	12107.7	1431.75	24787.28	2055.667
6191.217	1102.833	12077.68	1434.75	24542.23	2109
6187.142	1104.917	11972.35	1440.583	24302.31	2122.5
6177.117	1105.333	11944.81	1455.25	24127.83	2131.5
6175.896	1108.75	11883.88	1458.417	23987.43	2142.167
6162.492	1108.833	11878.25	1460.083	23910.11	2149.833
6156.308	1112	11876.07	1461.917	23882.1	2150.5
6143.558	1115.083	11863.4	1462.417	23787.98	2150.667
6135.967	1116.75	11803.03	1463.083	23508.19	2157.833
6118.117	1117.333	11791.7	1465.75	23505.98	2175.833
6096.438	1118.417	11746.14	1466.583	23302.96	2179.333
6082.825	1121.5	11601.23	1466.583	23134.98	2181.333
6062.883	1126.833	11590.53	1482	23121.77	2198.5
6037.325	1127.25	11507.24	1483.75	23064.54	2199.5
6027.325	1128.583	11485.25	1485.25	23031.11	2199.75
6026.517	1130.167	11457.32	1487.25	23008.74	2204.667
5983.933	1130.333	11437.07	1488.083	22874.08	2215.083
5972.738	1135.5	11378.19	1497.417	22861.7	2217.167
5968.242	1137.833	11347.61	1497.75	22851.07	2221.833
5943.613	1139.167	11336.48	1499.25	22756.35	2233.083
5941.567	1140.25	11317.12	1503.417	22660.48	2237.667
5906.888	1141.417	11306.69	1504.25	22598.07	2241.833
5896.829	1142.167	11305.07	1508.25	22585.53	2245
5895.188	1142.833	11299.99	1508.417	22568.86	2257.833
5891.263	1143.667	11286.01	1509.917	22517.37	2260.333
5860.575	1145.083	11268.18	1510.417	22367.79	2262.583
5849.688	1149.25	11198.51	1510.583	22321.84	2263.167
5826.892	1150.333	11175.72	1514.083	22313.17	2274.167
5822.058	1153	11156.33	1515.083	22298.6	2275
5790.188	1154.083	11152.17	1516.583	22286.47	2279.5
5775.383	1160	11149.35	1520.25	22278.99	2280.75
5773.15	1162.917	11144.61	1520.583	22095.48	2280.917
5721.933	1163.167	11120.29	1523.083	21869.8	2283.083
5721.017	1168.25	11083.76	1523.917	21859.41	2295.5
5696.879	1171.667	11015.35	1526.583	21824.71	2303.667

5693.096	1173.417	11010.77	1527.25	21802.75	2307.75
5675.167	1174.75	11007.36	1530.25	21609.95	2313.417
5652.888	1175.75	11003.21	1531.25	21578.35	2319
5637.588	1179.75	10983.04	1532.75	21538.73	2330.083
5635.021	1183.083	10957	1537.667	21420.7	2336
5626.438	1184.75	10947.05	1537.75	21105.8	2349.333
5614.354	1186.167	10939.29	1538.25	21097.4	2366.667
5587.088	1187.167	10865.96	1540.917	21037.87	2380.667
5584.583	1195	10842.4	1549.083	20913.86	2381.833
5574.629	1195.083	10818.69	1551.25	20897.11	2387.25
5523.875	1198.917	10799.27	1553.75	20892.29	2400.167
5523.225	1201.167	10797.16	1556.25	20689.18	2403.75
5513.788	1204.25	10796.6	1558	20688.52	2405.083
5512.667	1208.75	10780.78	1559.417	20684.08	2406
5492.288	1209.75	10764.12	1559.917	20681.65	2418.167
5487.042	1210.25	10753.52	1560.417	20675.71	2432.167
5474	1211.25	10737.56	1562.083	20605.92	2434.417
5456.538	1211.667	10736.58	1568.167	20526.58	2448.333
5452.967	1213.917	10671.14	1569.25	20508.38	2450
5439.308	1215.083	10662.81	1572.75	20451.14	2455.5
5392.6	1219.167	10648.95	1572.917	20421.75	2466.5
5378.504	1221.917	10648.76	1573.583	20419.98	2472.667
5374.05	1226.25	10619.77	1577.083	20296.99	2483.167
5366.621	1227.083	10604.31	1579.75	19976.42	2483.583
5351.933	1227.417	10604.05	1583	19741.76	2483.75
5335.85	1230.25	10597.68	1583.833	19631.7	2543.75
5297.696	1231.917	10594.82	1585.667	19516.41	2555.25
5255.625	1238.5	10513.01	1586.25	19492.88	2574.5
5237.533	1244.083	10490.93	1593	19487.75	2577.25
5221.45	1246.917	10437.03	1603.333	19409.96	2581.833
5213.05	1248.667	10405.39	1607.25	19408.02	2593.167
5205.525	1251.417	10400.58	1612.333	19342.76	2594.75
5204.013	1252.417	10389.68	1613	19199.85	2596.833
5202.479	1252.583	10335.61	1614.5	18998.52	2610.75
5153.654	1257.833	10309.84	1620.083	18933.49	2621.75
5115.088	1267	10299	1620.75	18915.11	2624.417
5096.517	1271.083	10295.83	1621.75	18726.55	2639.25
5084.096	1271.5	10269.89	1627	18539.3	2667.25
5079.021	1283	10261.68	1628	18524.58	2670.417
5064.663	1291	10223.94	1630.667	18462.39	2677.75
5054.146	1292	10221.5	1632.5	18456.85	2680.917
5032.738	1293.5	10180.46	1637.25	18391.1	2701.75
5015.371	1296.5	10164.25	1642	18372.7	2704.917
5000.6	1297.25	10156.66	1644.25	18356.45	2707
4968.629	1305.667	10153.84	1647	18348.76	2713.75
4961.163	1311.667	10145.29	1649.75	18325.38	2714.25
4935.763	1317.25	10076.7	1656	18266.54	2717.667
4925.692	1318.583	10069.25	1660	18254.48	2719.25

4918.271	1322.417	10056.36	1661	18110.19	2723.5
4882.192	1324.667	10041.35	1663.75	18061.9	2744.25
4876.475	1326.25	10039.85	1665.25	17991.63	2745.417
4869.529	1331	10030.68	1666.75	17907.79	2763.333
4848.304	1336.417	10010.13	1667.75	17903.1	2765
4838.629	1337.75	10001.69	1671.25	17815.5	2770
4822.217	1338.25	9989.925	1672.5	17537.37	2774.417
4818.654	1339.75	9943.075	1677.5	17537.3	2798.5
4780.317	1340.75	9912.85	1679	17520.35	2804.917
4778.754	1345	9893.05	1680.25	17511.7	2809.917
4760.738	1350.5	9865.7	1687.5	17310.6	2810.417
4753.546	1358.417	9864.925	1695.5	17305.2	2863.25
4726.396	1359.833	9864.438	1699.25	17087.94	2866.75
4724.588	1360.25	9845.225	1701.25	17083.88	2874.5
4722.2	1367.5	9835.238	1701.75	17037.54	2898.25
4717.871	1370.417	9750.213	1702.75	16836.44	2898.75
4714.15	1371	9663.413	1718.25		
4698.279	1372.917	9621.4	1734.5		
4669.738	1375	9607.975	1741.25		
4663.479	1378.667	9598.125	1741.5		
4658.779	1383.833	9580.35	1743.75		
4643.917	1385.417	9562.913	1748.25		
4614.688	1389.167	9555.971	1751.083		
4578.342	1391.75	9548	1751.5		
4573.913	1402.25	9534.825	1753.5		
4559.629	1403.167	9532.588	1759		
4531.071	1407.5	9520.638	1759.25		
4528.488	1407.667	9463.138	1759.75		
4522.488	1410.75	9449.7	1767		
4490.017	1412.25	9437.088	1767.417		
4489.321	1420.583	9411.45	1769.25		
4484.813	1421.75	9360.1	1775.75		
4484.063	1424	9353.675	1780.75		
4457.183	1424.333	9329.342	1782.917		
4434.638	1430	9323.263	1784.5		
4425.517	1433.25	9315.1	1785.5		
4409.675	1436.417	9311.263	1786.75		
4400.525	1439.5	9304.546	1789.5		
4392.563	1441.25	9294.775	1790		
4387.179	1444.167	9282.525	1791.75		
4384.213	1447.75	9224.15	1795.75		
4365.046	1448.917	9189.463	1797.25		
4358.838	1452.917	9183.713	1801.5		
4354.842	1454	9170.425	1802.75		
4347.963	1455	9161.488	1803.25		
4327.813	1460.5	9152.888	1806		
4287.638	1466.75	9137.563	1810		
4287.325	1470.75	9061.163	1810.5		

4266.375	1472	9044.5	1815.5
4263.363	1476.75	9011.475	1824
4263	1480.25	8990.538	1832
4253.375	1483	8973.35	1837
4237.375	1487.25	8955.725	1838.5
4216.963	1487.5	8929.95	1840.5
4203.038	1488.5	8925.4	1844
4195.075	1490.25	8900.425	1845.5
4192.238	1502	8899.988	1852.5
4182.688	1510	8841.125	1855
4161.225	1510.25	8820.2	1857.25
4147.3	1511.25	8811.725	1860.25
4132.988	1511.75	8794.388	1860.75
4132.263	1522.5	8782.875	1864.5
4127.463	1528.5	8764.813	1867.75
4076.625	1532	8736.275	1870
4074.613	1540.75	8678.413	1874.5
4061.238	1545	8657.488	1876.75
4061.138	1558.25	8642.9	1882.75
4053.638	1559.75	8620.163	1884
4005.5	1567.75	8619.688	1892.5
3998.988	1579.5	8538.688	1893.25
3974.2	1593.75	8480.563	1903
3942.063	1597.5	8441.113	1916.5
		8395.113	1924.25
		8393.325	1927.5
		8377.4	1929.25
		8371.763	1934.25
		8355.163	1936.75
		8339.35	1939.25
		8334.175	1941.5
		8306.763	1951.25
		8247.75	1954.5
		8228.538	1956.5
		8228.088	1964
		8197.2	1965.75
		8185.5	1977
		8182.85	1978
		8090.563	1978.75

APPENDIX 6 – VBIH Results for Large Instances (20 Jobs)

VBIH ALGORITHM					
20x5_01		20x10_01		20x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
6492.79	1065	13140.5	1319.17	26419	1922.5
3942.06	1597.5	11929.4	1452.75	25483.6	2039.5
5893.44	1136.92	11824.7	1474.25	25281.1	2048.83
3974.2	1593.75	11703.7	1480.92	25237.3	2051.83
3998.99	1579.5	11668.4	1484.25	25235.4	2089.17
4005.5	1567.75	11581.3	1485.75	25074.5	2090.5
4053.64	1559.75	11523.2	1489.92	25040.6	2093
4061.24	1545	11516.6	1493.08	25026.3	2102.5
4076.63	1532	11482.6	1497.08	25022.5	2104.33
4132.36	1509.25	11468.3	1499.42	24665.1	2114.33
4108.76	1528.25	11465.4	1508.75	24431.5	2115.5
4437.18	1437	11387.8	1509.92	24414.5	2127
4405.16	1441.25	11324.8	1516.75	24359.9	2128.67
4131.73	1523.5	11281.7	1517.92	24175	2155
4624.35	1386.33	11234	1521.83	23896.3	2167.5
4125.99	1525.75	11217.9	1522.92	23725.8	2184.42
4561.22	1396.33	11209.8	1526.67	23692.4	2197
4164.69	1506.25	11187.1	1529.08	23343.1	2205.25
4195.16	1504.25	11180.3	1531.08	23193.5	2210
4213.66	1494.25	11161.7	1532.58	22975.8	2216.75
4234.16	1492.25	11071.4	1533.75	22724.6	2228.75
4202.03	1496	11055.7	1538.08	22693.6	2232.42
4183.2	1505.75	11048.3	1541.42	22635.7	2258.5
4559.6	1405.58	11026.4	1542.92	22559.7	2266
4684.28	1374.08	10953	1543.17	22555.8	2269.75
4250.71	1480.75	10904.2	1548.58	22546.1	2270
4244.34	1486.75	10903.8	1551.92	22341.6	2271.33
4266.78	1476.75	10899.1	1552.67	22283.2	2276.83
4304.5	1472.25	10888.6	1555.25	22243	2286.5
4229.23	1493.5	10851.1	1558.92	22030.1	2305
4400.6	1450	10810.9	1560.92	21566.4	2307
4264.74	1477	10783.4	1567.42	21421	2335.25
4339.97	1461.58	10760.2	1568.33	21379.3	2366.5
4387.1	1453	10758.7	1572.5	21354.4	2372.25
4372.23	1456.5	10673.1	1573.08	21243.3	2373.75
4818.23	1349	10650.7	1573.92	21242.3	2382.08
4822.08	1338.75	10619.8	1577.08	21177.4	2384.92

4600.62	1392.67	10604.1	1583	20937.1	2387.08
4455.04	1427.92	10513	1586.25	20914.4	2411.75
4387.79	1452.17	10405.4	1607.25	20873.1	2417.08
4285.25	1474	10384.9	1614.5	20710.6	2432.33
5478.86	1214.42	10371.3	1621	20674.7	2451.75
5771.13	1156.92	10330.1	1624.33	20462.9	2467
4646.18	1385.83	10319.1	1626.75	20462.3	2474.25
5656.09	1185.5	10261.7	1628	20274.2	2479.92
4770.09	1357	10254.5	1645	20245	2495.5
5686.49	1182.42	10182.5	1649.25	20230.7	2502.25
4509.4	1414.67	10133.4	1658	20198.4	2503.08
4349.03	1459.25	10061.7	1663.75	19995.2	2508.5
4962.5	1320.42	10046.3	1668.25	19963.6	2523.83
5656.3	1184.42	10023	1674.83	19960	2530
5169.4	1260.75	9988.2	1680	19959.1	2542.25
5823.78	1150.17	9903.69	1688.5	19747	2547.5
5768.52	1161.17	9873.3	1696.42	19742.3	2549.5
4658.44	1377.67	9860.93	1703.25	19738.8	2565.67
5257.76	1249.42	9783.73	1704.25	19723.7	2569.42
4316.59	1466	9771.35	1716.25	19589.8	2573.92
4806.25	1354.58	9701.6	1723.75	19401.3	2580.17
5035.22	1287.33	9664.91	1731.75	19284.8	2593
5086.05	1283.83	9657.09	1740.25	19265.8	2598.67
4471.99	1425.67	9630.5	1743.25	19246.2	2613.75
4706.86	1369.67	9568.45	1747.25	19239.6	2614
5325.08	1244.42	9515.54	1753.25	19060.2	2617
5130.1	1281.92	9417.25	1763	19000.7	2621.17
4754.55	1364.08	9376.56	1773.5	18797.4	2626.92
5460.35	1219.67	9323.51	1785	18711.7	2667.75
5603.87	1193.58	9303.34	1788.75	18670	2684.25
5428.87	1220.25	9290.29	1791	18639.6	2688.5
4857.94	1337.75	9268.93	1795	18628.6	2692
4524.53	1410.5	9262.53	1799	18245	2694.58
4863.33	1334	9183.04	1802.25	18212.9	2722.08
5267.55	1246.08	9175.45	1806.5	18010.7	2735.08
4312.28	1469.75	9149.75	1810.75	17780.5	2773.42
4684.98	1373.5	9118.68	1820	17764.9	2794.58
4957.84	1323	9105.36	1822.25	17659.3	2815.5
4365.24	1459	9089.75	1827.25	17527.6	2825.25
4475.31	1423.33	9051.25	1829	17510.6	2849.75
5675.56	1184.33	8996.68	1838.5	17502.6	2852.75
4694.92	1371.5	8972.9	1839.25	17358.7	2853
4760.25	1360.83	8932.19	1850.75	17305.5	2856.5
5027.33	1309.17	8858.35	1853.75	17299.9	2875.25

5724.17	1171.92	8794.11	1861.25	17259.3	2879.25
4914.65	1327.83	8750.96	1865.25	16821.4	2883.75
5130.52	1272.58	8724.98	1886.25		
4553.17	1410.33	8628.28	1890.25		
5566.84	1201.58	8604.25	1901		
4704.84	1369.83	8556.34	1906.5		
5335.33	1229.92	8460.58	1920.25		
5623.5	1186.42	8451.15	1929.75		
4496.58	1415.42	8339.35	1944.5		
4451.81	1431.5	8304.15	1966.5		
5599.03	1197.83	8090.56	1978.75		
4791.66	1355.17				
5496.78	1207.58				
5609.96	1191.67				
5688.23	1181.75				
5579.85	1201.5				
5534.92	1204.08				
5773.47	1152.5				
5004.61	1312.17				

APPENDIX 7 – Mathematical Model Results for Large Instances (50 Jobs)

MATHEMATICAL MODEL					
50x5_01		50x10_01		50x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
15189.7	2386.67	14827.8	2684.58	64490.3	4193.33
14884.5	2522.92	14189.8	2800.83	61917.9	4219.83
14582.4	2588.5	13235.3	2832.75	49702.8	5838.83
14289.5	2632.25	12588	2904.75	59517.7	4644.17
13990.3	2661.33	12264.4	3141	49453.4	5908.33
13697.6	2693.92	11011.1	3363.08	56740.5	4987.52
13377.3	2707.67	10353.1	3621.67	49453.4	5908.33
12485.5	2795.58	10077.3	3790.83	51788.7	5775.33
12169.6	2888.25	31707.5	2978.83		
11869.4	3109.75	29285.6	3059.75		
11590.2	3208.5	25828.8	3514.67		
11283.6	3223.25	24346.9	4012.75		
10686.9	3357	22361.7	4163.42		
9466.5	3609.5	21198.5	4392.25		
		19903.3	4620.5		

APPENDIX 8 – IG_{ALL} Results for Large Instances (50 jobs)

IG_{ALL} ALGORITHM					
50x5_01		50x10_01		50x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
15161.6	2274.17	31605.4	2552.5	65998.3	3272.5
13640.3	2510.83	27334.3	2910.75	58736.9	3766.75
13460.6	2520	27289.2	2925.83	58476.8	3782.83
13451.5	2526.92	27188	2933.75	58385.1	3795.92
13435.9	2527.42	27166.8	2935	58217.6	3816.75
13434.9	2529.75	27158.1	2936.42	58109.5	3819.25
13423.8	2533.08	27158	2940.5	58033.2	3823.75
13388	2533.42	27112.6	2941	58002.6	3835.58
13270.9	2537.17	27034	2946.67	57938.3	3840.58
13251.2	2546.33	26915.8	2951.92	57871.3	3841
13246.5	2558.83	26908.9	2956.83	57230.1	3841.5
13097.1	2559.42	26876.7	2957.08	56803.2	3842.08
13096.6	2585.08	26869	2958.92	56671.6	3862.92
12994.5	2586	26812.7	2967.5	56440.1	3873.75
12944.8	2588.42	26765	2979.5	56428.4	3874.42
12944.3	2596.67	26755.2	2982.83	56389.8	3875.33
12943	2600.33	26726.5	2986.83	56367.4	3883.42
12841.3	2600.33	26725.5	3002.5	55671	3887.33
12793.6	2614.25	26652.5	3022	53242.5	3927
12759.9	2618.25	26537	3030.58	53220.8	4089
12758.6	2620.17	26318.2	3037.92	52998.1	4097.08
12752	2624.67	26302	3041.67	52968	4097.67
12746	2624.83	26296.5	3043.25	52843.7	4105.33
12731.9	2625.33	26292.8	3045	52721	4116.33
12728.1	2633.67	26266.1	3051.92	52656.2	4122.5
12687.1	2633.83	26180.4	3056.83	52361.5	4126
12636.1	2636.5	26168.3	3057.67	52144.3	4151.17
12627.9	2637.67	25376.5	3063	51898.4	4151.42
12618.7	2641.33	25326.1	3161.83	51758.2	4170.75
12615.4	2641.5	25212.4	3163.25	51660.9	4194.83
12602.1	2643	25209	3164.58	51606.6	4195.17
12584.9	2646.67	25199.1	3174.83	51586.7	4224.25
12505.8	2657.42	25190.5	3176.08	51502.1	4232.42
12499.8	2662.42	25189.6	3180.5	51462.7	4243
12497.2	2663.25	25187.2	3180.92	51415.2	4251.17
12486.8	2666.08	25147.2	3181.5	51357.2	4251.58
12446.9	2672.08	25050.9	3183.08	51124	4256.42
12365.8	2680.75	24894.7	3197.67	50986.8	4273.25
12356.9	2685.08	24880.6	3207.67	50939.7	4278.42

12333.9	2692.92	24834.6	3209.75	50932.6	4284.67
12305.4	2693.5	24833.8	3210.75	50873.9	4285.75
12298.8	2702	24693.6	3214.5	50740.1	4298.33
12271.5	2705.58	24677	3229.75	50659	4305.33
12256.5	2718.5	24661.7	3232.25	50415.2	4311.83
12231.2	2722.67	24501.2	3244.08	50405.8	4314.33
12221.6	2724.58	24419.5	3248.25	50286.9	4314.83
12171.7	2726.25	24338.2	3248.67	50231	4317.33
12155.4	2729	24338	3251.33	50105.7	4320.67
12106	2740.17	24306.2	3252.92	49965.9	4322.17
12081.2	2745.33	24303.7	3254.17	49964.8	4324.17
12057.5	2749.92	24294.1	3255	49905.7	4326.83
12010	2755.17	24270.1	3260	49892	4334.25
11990.3	2756.58	24263.7	3265.5	49708.4	4338.92
11973	2757.75	24178.3	3271.17	49421.9	4350.08
11970.6	2764.17	24064.2	3276.58	49405.8	4366.33
11949.9	2765.92	24056.7	3280.08	49138.9	4369.42
11948.9	2766.25	24035.4	3284.5	49002.5	4420.17
11917.5	2770.08	23925.7	3294	48791.8	4426
11894.7	2775.5	23783.3	3310.25	48774.3	4432.5
11863.8	2780.25	23769.2	3332.33	48481.2	4437.42
11857.1	2782.33	23688.6	3334.92	48373.9	4450.25
11847.5	2783.42	23686.3	3346.5	48028.3	4490.33
11847.3	2787.42	23511.9	3347.25	47802	4499.5
11823.5	2790.42	23462.7	3355.83	47473.9	4505.33
11811.6	2792	23460.1	3364.33	47421.4	4559.67
11806.4	2801.67	23448.2	3369.5	47242.5	4571.17
11805.5	2804.33	23364.4	3370.83	47198.9	4593.42
11775.6	2804.83	23263.5	3387.92	47121.7	4623.33
11771.7	2805.5	23222.3	3388.42	46923.7	4641.75
11752.3	2810.08	23203.5	3398.67	46843	4647.33
11734.8	2816.33	23138.8	3406.08	46837.1	4652.42
11714.9	2816.42	23127.8	3410.08	46691.7	4658.42
11704.8	2817.42	23019.6	3415.92	46648.7	4671.17
11702.9	2818.92	22759.3	3428.08	46631.2	4677.08
11680.1	2819.92	22736.2	3466.75	46504.5	4710.33
11648.5	2821.17	22600	3473.25	45955.4	4806.67
11640.4	2829.92	22543.2	3491.33	40597.6	4908.75
11634.1	2833.17	22532.2	3500.08		
11631.7	2835.75	22526.8	3520.75		
11610.6	2837.33	22507.5	3525		
11587.1	2839.83	22442.3	3525.25		
11587.1	2842.83	22251.3	3526.25		
11572.3	2844.5	22179	3532.67		

11531.9	2846	21796.6	3570.08
11500.6	2849.75	21781.6	3574.75
11492.7	2850.25	21745.9	3580
11481.1	2852.42	21735.1	3634.25
11470	2859	21321.6	3643.58
11465.1	2859.42	19170.6	3828.75
11429.6	2865.25		
11368.6	2871.08		
11360.2	2877.08		
11359.4	2883.58		
11312.9	2884		
11259.7	2889.92		
11203.5	2918.92		
11197.5	2927.58		
11180.7	2932.58		
11166.7	2934.83		
11160.2	2938.08		
11155	2938.33		
11124.7	2942.08		
11121.7	2951.08		
11094.9	2952.83		
11086.6	2955.58		
11045.8	2956.92		
11017.9	2970.67		
10999	2973.92		
10977.8	2974.33		
10923.1	2986.83		
10895.6	2987.58		
10843.1	3002.67		
10802.2	3022		
10774	3033.5		
10767.6	3035.67		
10743.5	3037.33		
10663.6	3044.42		
10576.1	3059.58		
10546.5	3079.5		
10542.5	3082.67		
10534.2	3084.42		
10456.9	3094.17		
10441.4	3138.33		
10421	3149.17		
10377.1	3177.75		
9155.75	3411.25		

APPENDIX 9 – VBIH Results for Large Instances (50 jobs)

IG$_{ALL}$ ALGORITHM					
50x5_01		50x10_01		50x20_01	
TEC	C_{MAX}	TEC	C_{MAX}	TEC	C_{MAX}
15160.5	2270	31602.9	2547.5	65995.8	3270
9154.19	3405	28111.4	2937	58783.5	3801.75
13278.6	2557.25	28065.4	2944.75	58505.5	3806.5
10503	3253	28058.3	2947.5	58190.5	3833.17
10596.6	3180.5	27949.5	2955.58	57993.2	3851.5
10541.3	3224	27833.3	2959.83	57843.3	3866.17
10594.7	3220	27831.8	2964	57763.3	3872.17
10595.6	3181.58	27826	2966.33	57714.1	3875.17
10617.5	3119.33	27822.4	2966.5	57572.7	3875.75
10645.4	3030.92	27820.7	2966.75	57237.9	3889.92
11161.9	2953.67	27807.9	2966.83	56863.9	3891
10717.5	3024.75	27797.8	2967.83	56859	3899.5
10773.8	3007.5	27794.1	2969.75	56366.8	3908.67
10728.7	3017.58	27784.1	2974.42	53239.5	3924
10916.7	2983.5	27688.8	2977.92	53228	4111.42
10560.8	3223.42	27664.1	2978.5	53010.4	4116.25
10856.6	2994.25	27059.7	2979.58	52888.2	4119
10814.4	3002.75	26894.3	3009.75	52840.7	4128.25
10572.3	3220.67	26848.8	3037.5	52822.1	4138.33
11098	2967.92	26840.9	3043.25	52641.6	4147.42
10868.1	2987.5	26617	3044.83	52632	4160.17
11085.5	2979	26596.5	3045.5	52501.5	4165.33
11077.4	2982.75	26285.2	3052	52365.4	4168.58
13178.8	2578	25373.5	3057	52272	4170.25
11223.8	2940.67	25368.3	3151	52267	4177.08
11524.7	2872.92	25365.2	3155.67	52007.7	4181.25
10858.8	2992.5	25318.8	3156.33	51996.3	4192.67
11346.4	2904.75	25299.3	3165.92	51758.8	4193.67
11992.4	2778.08	25241.6	3170.5	51681.4	4205.58
11452.8	2884.17	25048.4	3173.5	51656.2	4216.33
11342.6	2913.5	25005.5	3204.33	51587.1	4216.42
11585.3	2861.25	24960.6	3204.75	51539.4	4216.58
12335.4	2718.17	24945.6	3211.08	51424.6	4216.75
11238.5	2934.25	24907	3214.83	51204.5	4228.92
11576.8	2870.17	24897.9	3217.75	51120.4	4257.83
11901.5	2796.42	24862.1	3218.17	50934.6	4259.08
10756	3014	24843.9	3230.42	50833.7	4261.33
11268	2925.75	24823.5	3235.33	50611.1	4276.83
11412.3	2899.92	24800.2	3236.25	50362.3	4286.08

11320.1	2922.42	24745.6	3238.25	50352.9	4290.67
11407.9	2900.75	24707.4	3238.33	50345.5	4297.75
11230.3	2935.75	24640.8	3239	50308.3	4298.75
11490.1	2878.33	24559.7	3242.25	50102	4306.75
11398	2903.25	24549.1	3245.33	50042.4	4308.08
12033.1	2775	24547.4	3247.5	49863	4347.17
11253	2932.92	24439.9	3248.75	49815.2	4383.67
12960	2616.17	24429.5	3255.92	49780	4394.08
11695.1	2836.42	24422.7	3256.5	49179.2	4397.67
11480.4	2882.67	24310.5	3257.83	49156.9	4438.33
12759	2667.08	24299.5	3263.83	49053.7	4443.5
11801.8	2814.33	24291.3	3265.08	49017.7	4448.5
12252.9	2723.5	24288.9	3266.67	48818.4	4452.58
11323.8	2920.42	24233.7	3269.58	48563.1	4475.58
12542.4	2681.33	24170.4	3272	48485.1	4485.75
11629	2855.33	24155	3277	48365.7	4493.67
11480.9	2878.67	24153.2	3280	48165.8	4502.75
12111.3	2756.58	24149.6	3280.08	47967.7	4526.5
12154.2	2724	24066	3288.5	47676.6	4537.75
12440.1	2700.58	24055.7	3298.25	47400.8	4562.17
11277.1	2925.42	23986.7	3303.83	47288.9	4634
11775.4	2825.08	23918.6	3306.42	47266.8	4660.75
11734.7	2828.67	23853	3316.92	46984.2	4670.5
12121.8	2748.92	23816	3333.75	40593.8	4905
11262.4	2927.08	23792.7	3337.08		
12475.6	2695.92	23752.5	3345.25		
11443.6	2893.83	23673.1	3345.58		
12899.7	2640.5	23645.1	3348.33		
11733.8	2832.25	23556.2	3353.67		
12341	2712.92	23553.2	3368.75		
11578.4	2866.17	23547.5	3371.42		
12389	2702.75	23519.4	3374.17		
12482.4	2694.92	23441.6	3378.5		
11760.5	2825.67	23418.9	3382.08		
12639.8	2680.75	23370.5	3394.67		
11883.7	2802.92	23334.2	3408.42		
12358.9	2707	23260.7	3416.75		
11781.2	2820.17	23082	3418.92		
12140	2745.25	23021	3433		
12769.8	2662.75	23015.2	3445.5		
12381.1	2705.25	22993.5	3450.92		
12120.4	2752.58	22881.1	3459.92		
11786.4	2817.17	22839.3	3486.42		
11863.2	2809.17	22637.6	3490.92		

11426.3	2899.75	22627	3509.08
12653.5	2674	22615.6	3516.33
13114.1	2587.17	22545.4	3554.5
12891.4	2642.5	22520	3565.75
11793	2816.5	22509	3566.25
11740.6	2828.08	22502	3566.33
12849.9	2643.17	22132.7	3567.92
11436.8	2895.92	22025.6	3571.67
12399.1	2700.75	21856.7	3588.08
13104.3	2589.25	21854.8	3653.75
11727.4	2835.92	21784.7	3678.25
11429.4	2897.92	21710.7	3700
11626.8	2858.17	19166.9	3821.25
11878.2	2805.08		
12091.8	2762.08		
12896.9	2642		
12254.4	2721		