ABSTRACT

KUMAR KODAKKAT, NIKIL. A Heuristic Approach to Maximize Workload Smoothness for an Apparel Production Line with Work Sharing and Worker Movement. (Under the direction of Dr. Jeff A. Joines).

A variation of the Simple Assembly Line Balancing Problem (SALBP) is examined. We consider an apparel production line where workers serve as the main constraint, with each worker potentially cross trained to do one or more tasks with a predefined efficiency. The primary objective is to obtain the optimal assignment of workers to tasks to maximize workload smoothness for the line which is measured by Mean Absolute Deviation (MAD) of workload and satisfy demand/production for a given time period. We provide a comparative assessment of scenarios where work sharing is used and not used. We also study the effects of allowing workers to be idle in these scenarios (i.e., workers would be able to perform other duties that are not directly related to production of the product). Smaller instances of the problem can be solved directly using CPLEX whereas larger versions cannot be solved optimally within reasonable computation times. Therefore, a heuristic utilizing a Genetic Algorithm is developed to find good solutions. A real industrial case is also solved as an example.

The worker to task assignment generated by solving these static models is then used to build simulation models in order to account for stochastic processing times. The results of the deterministic system are compared to the stochastic systems with ±10%, ±20% and ±30% variation in the Standard Operation Time based on a PERT distribution. We also determine when a worker is required to move to another station based on the number waiting in queue (termed as buffer size) at each station. This is done by comparing the effect of two types of policies – no buffers and a common buffer size for all stations. We measure three responses - MAD, Actual Production and Total Worker Movement, to determine the “optimal” buffer size.

A Heuristic Approach to Maximize Workload Smoothness for an Apparel Production Line with Work Sharing and Worker Movement

by
Nikil Kumar Kodakkat

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Master of Science

Operations Research

Raleigh, North Carolina

2013

APPROVED BY:

Dr. Jeffrey A. Joines
Committee Chair

Dr. Russell E. King

Dr. Kevin Gross

DEDICATION

To my parents, who have always encouraged me and given me the freedom to pursue anything I wanted.

BIOGRAPHY

Nikil Kumar Kodakkat completed his undergraduate degree in Industrial Engineering and Management at Sir M Visweswaraya Institute of Technology, Bangalore, India in July 2011. During his final semester he completed an internship as a Production Engineer at L&T – Komatsu. He joined the M.S. Operations Research program at North Carolina State University in Fall 2011. He began working on his Masters thesis under the guidance of Dr. Jeff Joines in Summer 2012. He enjoys playing his bass guitar as well as composing and recording music in his free time.

ACKNOWLEDGMENTS

I would like to thank Dr. Joines for his continuous support and guidance. Learning and working with him has been inspiring and rewarding. His advice, encouragement, patience and cheerful disposition over the past year and half have made the entire experience very educational and enjoyable.

I would like to thank Dr. King for his valuable insight with respect to the CPLEX and GA methodology, and for his comments and suggestions that have made this thesis a better final product. I am grateful to Dr. Gross for his advice and help with the simulation methodology and the notes he provided for his statistics course which was useful throughout my graduate studies.

I am thankful to my parents, grandparents and my brother for their support. They have been a constant source of love and encouragement all these years.

I would like to thank Priyanka Sheshadri for the entertainment, care and encouragement over the past two years, and Maruthi Chandrasekaran and Ashirward Barbora for the musically related distractions they provided whenever I needed a break.

TABLE OF CONTENTS

LIST OF TABLES viii
LIST OF FIGURES xi

Chapter 1 Introduction 1
1.1 History of the Assembly Line: 1
1.2 The Assembly Line Balancing Problem 2
1.3 Types of Assembly Lines 3
1.4 Terminology Associated with Assembly Lines: 4
1.5 Work Sharing and Cross Training 6
1.6 Assembly Lines in the Apparel Industry: 6
1.7 Motivation 7
1.8 Thesis Summary 8

Chapter 2 Literature Review 9
2.1 Classification of ALBPs: 9
2.2 Techniques used to solve ALBPs 10
2.2.1 Tabu Search 12
2.2.2 Simulated Annealing 12
2.2.3 Ant Colony Optimization 13
2.2.4 Particle Swarm Optimization 14
2.2.5 Genetic Algorithms 15
2.2.6 Hybrid Algorithms 15
2.2.7 Simulation 16
2.3 The Assembly Line Balancing Problem in Apparel Manufacturing 17

Chapter 3 Model Formulation 20
3.1 Methodology 20
3.2 Assumptions 23
3.3 Nomenclature 23
3.4 LP Formulation 25
3.5 Optimization using CPLEX 27

3.5.1 Industrial Example 27
3.5.2 Computational Study...30

Chapter 4 Genetic Algorithms 35
4.1 GA Specification 38
4.1.1 Chromosome Representation...38
4.1.2 Generation of Initial Population 39
4.1.3 Selection Scheme 40
4.1.4 Genetic Operators 41
4.1.5 Termination 43
4.1.6 Fitness Function...44
4.1.7 GA Parameters..53
4.2 Results..53
4.2.1 Industrial Example 54
4.2.2 Computational Study..55
4.2.3 Effect of Reducing the Number of Machines..63
4.3 Comparison between CPLEX and GA results 67
4.3.1 Industrial Example 67
4.3.2 Computational Study...68

Chapter 5 Simulation...82
5.1 Simulation Methodology..83
5.2 Linear Model with 6 Tasks 85
5.2.1 No Buffer Policy...87
5.2.2 Common Buffer Size 87
5.3 Branched Model with 6 Tasks...92
5.3.1 No Buffer Policy...94
5.3.2 Common Buffer Size 95

Chapter 6 Conclusions and Future Work 100
6.1 Conclusion..100
6.2 Future Work 102

References 103

Appendices 110
Appendix A 111
Data for Problem Sets 111
A.1 Data for the Linear 6 Task and Branched 6 Task Problems 111
A.2 Data for the Linear 12 Task and Branched 12 Task Problems 114
A.3 Data for the Linear 24 Task and Branched 24 Task Problems 116

LIST OF TABLES

Table 2 - 1: Summary of Techniques used to solve ALBPs 11

Table 3 - 1: Summary of Factors 22
Table 3 - 2: Apparel Line Constants 27
Table 3 - 3: Standard Processing Time 28
Table 3 - 4: Worker Skills 28
Table 3 - 5: Movement Time Matrix (in minutes) 29
Table 3 - 6: Results from the Industrial Case 29
Table 3 - 7: Legend for Results 30
Table 3 - 8: Results from the Linear Model 31
Table 3 - 9: Results from the Branched Model 33

Table 4 - 1: Worker Skills (Example) 38
Table 4 - 2: Types of Crossover 42
Table 4 - 3: Types of Mutation 43
Table 4 - 4: GA Parameters 53
Table 4 - 5: Legend for GA results 53
Table 4 - 6: GA Results for the Industrial Model 54
Table 4 - 7: Summary of GA results for the Linear Model 57
Table 4 - 8: Summary of GA results for the Branched Model 60
Table 4 - 9: Solutions for Multi-Objective Optimization 64
Table 4 - 10: Effect on the Average no. of Workers 66
Table 4 - 11: Legend for the comparison between CPLEX and GA results 67
Table 4 - 12: Comparison between CPLEX and GA solutions for the Industrial Model 67
Table 4 - 13: Comparison between CPLEX and GA computation times for the Industrial Model 68
Table 4 - 14: Comparison between CPLEX and GA Solutions for the Linear Model 70
Table 4 - 15: Comparison between CPLEX and GA Computation Times for the Linear Model 73
Table 4 - 16: Comparison between CPLEX and GA Solutions for the Branched Model 76
Table 4 - 17: Comparison between CPLEX and GA Computation Times for the Branched Model 79

Table 5 - 1: Variation in processing time for the four systems 85
Table 5 - 2: Static Assignments for the Linear 6 Tasks Model 85
Table 5 - 3: MAD and Actual Production (AP) with warm up 86
Table 5 - 4: MAD, Actual Production and Total Movement for policy A 87
Table 5 - 5: MAD, Actual Production and Total Movement after applying policy B to the Linear 6 Task Deterministic System 88

Table 5 - 6: Results from Tukey's test after applying policy B to the Linear 6 Task Deterministic System 89
Table 5 - 7: MAD, Actual Production and Total Movement after applying policy B to the Linear 6 Task Stochastic ±10% System 89
Table 5 - 8: Results from Tukey's test after applying policy B to the Linear 6 Task Stochastic ±10% System 90
Table 5 - 9: MAD, Actual Production and Total Movement after applying policy B to the Linear 6 Task Stochastic ±20% System 90
Table 5 - 10: Results from Tukey's test after applying policy B to the Linear 6 Task Stochastic ±20% System 91
Table 5 - 11: MAD, Actual Production and Total Movement after applying policy B to the Linear 6 Task Stochastic ±30% System 91
Table 5 - 12: Results from Tukey's test after applying policy B to the Linear 6 Task Stochastic ±30% System 92
Table 5 - 13: Static Assignments for the Branched 6 Tasks Model 93
Table 5 - 14: Change in MAD and Actual Production with warm up 93
Table 5 - 15: MAD, Actual Production and Total Movement for policy A 94
Table 5 - 16: MAD, Actual Production and Total Movement after applying policy B to the Branched 6 Task Deterministic System 95
Table 5 - 17: Results from Tukey's Test after applying policy B to the Branched 6 Task Deterministic System 96
Table 5 - 18: MAD, Actual Production and Total Movement after applying policy B to the Branched 6 Task Stochastic ±10% System 96
Table 5 - 19: Results from Tukey's Test after applying policy B to the Branched 6 Task Stochastic ±10% System 97
Table 5 - 20: MAD, Actual Production and Total Movement after applying policy B to the Branched 6 Task Stochastic ±20% System 97
Table 5 - 21: Results from Tukey's Test after applying policy B to the Branched 6 Task Stochastic ±20% System 98
Table 5 - 22: MAD, Actual Production and Total Movement after applying policy B to the Branched 6 Task Stochastic ±30% System 98
Table 5 - 23: Results from Tukey's Test after applying policy B to the Branched 6 Task Stochastic ±30% System 99

Table A - 1: Production Line Constants for the Linear and Branched 6 Task Sets 112
Table A - 2: Standard Processing Times for the Linear 6 Task Sets 112
Table A - 3: Standard Processing Times for the Branched 6 Task Sets 112
Table A - 4: Worker Skills for the Linear and Branched 6 Task Sets 113
Table A - 5: Movement Time Matrix for the Linear and Branched 6 Task Sets (in minutes) 113
Table A - 6: Production Line Constants for the Linear and Branched 12 Task Sets 114
Table A - 7: Standard Processing Times for the Linear 12 Task Sets 114
Table A - 8: Standard Processing Times for the Branched 12 Task Sets 115

Table A - 9: Worker Skills for the Linear and Branched 12 Task Sets................................115
Table A - 10: Movement Time Matrix for the Linear and Branched 12 Task Sets (in minutes) 116
Table A - 11: Production Line Constants for the Linear and Branched 24 Task Sets 117
Table A - 12: Standard Processing Times for the Linear 24 Task Sets................................117
Table A - 13: Standard Processing Times for the Branched 24 Task Sets...........................118
Table A - 14: Worker Skills for the Linear and Branched 24 Task Sets..............................119
Table A - 15: Movement Time Matrix for Linear and Branched 24 Task Sets (in minutes) 121

LIST OF FIGURES

Figure 1 - 1: Henry Ford's Highland Park Plant (Taken from [2]) 2
Figure 1 - 2: Types of Assembly Lines (Taken from Becker *et al.* [6])..................................3
Figure 1 - 3: Precedence Graph..5
Figure 1 - 4: Visual Representation of Terms used in Line Balancing (Adapted from De Jong [8])..5

Figure 3 - 1: Linear Structure...21
Figure 3 - 2: Branched Structure...21

Figure 4 - 1: A Simple Genetic Algorithm 36
Figure 4 - 2: Structure of a GA 37
Figure 4 - 3: General Chromosome Representation..39
Figure 4 - 4: Simple Crossover 41
Figure 4 - 5: Mutation Mechanism...42
Figure 4 - 6: Fitness Function 45
Figure 4 - 7: Chromosome violating the utilization constraint 47
Figure 4 - 8: Utilization Repaired Chromosome...47
Figure 4 - 9: Chromosome violating the maximum number of machines constraint..............48
Figure 4 - 10: Repair Procedure for Exceeding the Maximum Number of Machines 48
Figure 4 - 11: Maximum Number of Machines Repaired Chromosome 49
Figure 4 - 12: Chromosome violating the $N_{MaxTasks}$ constraint...49
Figure 4 - 13: Repair Procedure for Exceeding the Maximum Number of Tasks 50
Figure 4 - 14: 'Max Tasks' Repaired Chromosome..51
Figure 4 - 15: GA Convergence 55
Figure 4 - 16: Effect on the Average no. of Workers..66

Figure 5 - 1: MAD vs. Warm up 86
Figure 5 - 2: Actual Production vs. Warm up...86
Figure 5 - 3: MAD vs. Warm up 93
Figure 5 - 4: Actual Production vs. Warm up...94

Figure A - 1: Precedence graph for Branched 6 Task Problem 111
Figure A - 2: Precedence graph for Branched 12 Task Problem 114
Figure A - 3: Precedence graph for Branched 24 Task Problem 116

Chapter 1

Introduction

1.1 History of the Assembly Line:

The invention of the assembly line was an important landmark in the history of industrialization. It revolutionized the traditional approach to manufacturing and marked a transition from craftsmanship to mass production. Its origin is fairly deep rooted in history and examples can be traced from the system of bucket elevators in Assyrian times to meat packing factories and textile mills of the 19^{th} century. It is a concept that has evolved and improved through the ages and was popularized by Henry Ford. The credit for the modern version of the invention however, belongs to Ransom Eli Olds of the Olds Motor Vehicle Company who used it to mass produce the Oldsmobile Curved Dash in 1901. Henry Ford further improved his concept by implementing a conveyor system to mass produce the Ford Model T at his Highland Park Plant in 1913, shown in Figure 1 - 1.

When manufacturers first implemented the assembly line, they were able to achieve a considerable gain in productivity and a reduction in manufacturing costs, making products much more affordable for the consumer. However, it began to affect the nature of work in a factory. The need for skilled labor gradually reduced, since tasks were now broken down and simplified so that could be done by unskilled or semi-skilled laborers. Manufacturers soon realized that a more analytical approach was required with respect to task assignments and sequencing [1].

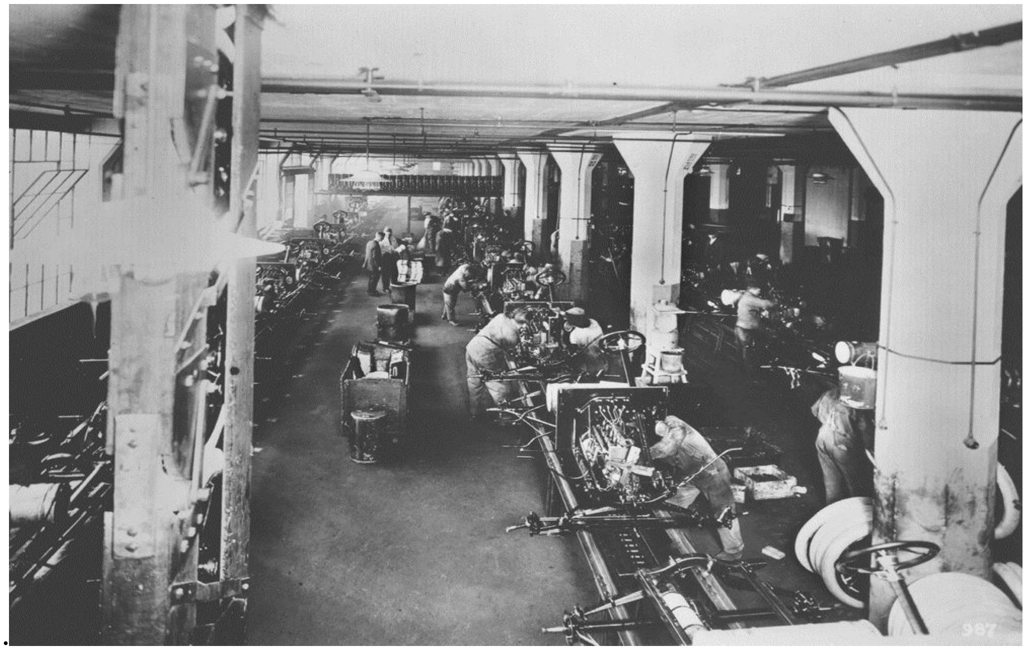

Figure 1 - 1: Henry Ford's Highland Park Plant (Taken from [2])

1.2 The Assembly Line Balancing Problem

The Assembly Line Balancing Problem (ALBP) can be described as follows. Given a set of tasks of various lengths subject to precedence constraints (i.e. some tasks must be completed before others can begin) and a time constant called cycle time, the tasks should be distributed over workstations along a production (assembly) line such that no workstation takes longer than the cycle time to complete all the tasks assigned to it and all the precedence constraints are satisfied [3]. A perfectly balanced line runs on a constant beat where every task is completed with the same constant time.

Some of the common goals of the assembly line balancing problem are:

1. To minimize the number of workstations for a given cycle time,
2. To minimize the cycle time for a given number of workstations or corollary maximize the production capability,

3. To maximize the workload smoothness so that work is distributed equally for a given number of workstations. The mean absolute deviation (MAD) of workload (i.e., average absolute deviation of a task time to the average task time) is typically used as a measure of workload smoothness [5,6], thus, the objective is to minimize the MAD of the workload of the line,
4. To minimize idle time for a given number of workstations, and/or
5. To minimize cost for the system.

1.3 Types of Assembly Lines

Assembly lines can be broadly categorized into three types: single model line, multi-model line, and mixed-model line (see Figure 1 - 2). A single –model line consists of a line that produces only one type of product while a multi-model line is used to produce a variety of products. Finally, the mixed-model line produces a variety of products with intermediate setups between the different product types.

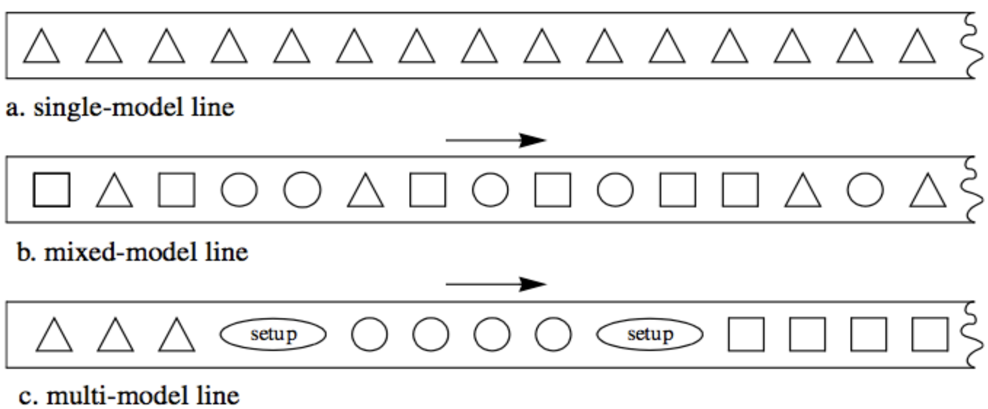

Figure 1 - 2: Types of Assembly Lines (Taken from Becker *et al.* [6])

1.4 Terminology Associated with Assembly Lines:

Rao [7] defined a set of terms associated with Assembly Lines which will be used in this work:

1. A **Work Element** is a single task that cannot be subdivided. The number of work elements for a given product assembly can be indexed on i where $1 \leq i \leq n$, where n indicates the total number of work elements required to complete a product assembly. "Task" and "operation" are also sometimes used in place of "work element".
2. **Work Element Time** is defined as the amount of time required to complete each work element 'i' and is measured in terms of a performance time 't_i'. The sum of these performance times associated with all work elements is defined as the total work content for each product being assembled.
3. A **Job** is defined as a combination of work elements.
4. A **Workstation** is a location on the assembly line where one or more jobs are performed by one or more operators.
5. **Cycle Time** is the amount of time elapsed between successive units as they move down the line when it is at 100 percent efficiency. For mixed-model scheduling the time that separates the launching of two consecutive units (either of the same or of different models) on the main conveyor line is termed as production cycle time.
6. **Workstation Time** is the total time required to complete the total amount of work assigned to a specific station on the assembly line is termed as work station time. If t_i is the unit time of the element 'i' and T_k is the k^{th} station time, then $t_i \leq T_k \leq c$ where c is the cycle time.
7. **Station Idle Time** can be calculated as the difference between the cycle time and workstation time for a particular station.
8. **Throughput time** is defined as the amount of time required for raw material to be processed into a finished product.
9. A **Precedence Graph** is one that represents the sequence of operations to be carried out on an assembly line. It contains a node for each task, arcs for the precedence constraints, and arc weights for the task times. Figure 1 - 3 shows a precedence graph with $n = 6$

tasks. It can be seen that starting task five requires that tasks three and four (direct predecessors) and one and two (indirect predecessor) be completed.

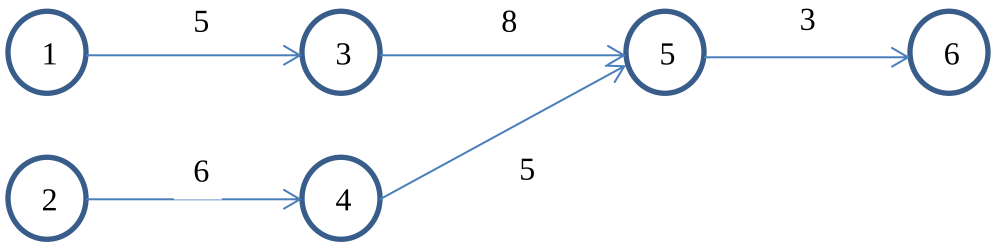

Figure 1 - 3: Precedence Graph

The terms that have been defined so far can be visually represented as shown in Figure 1 - 4.

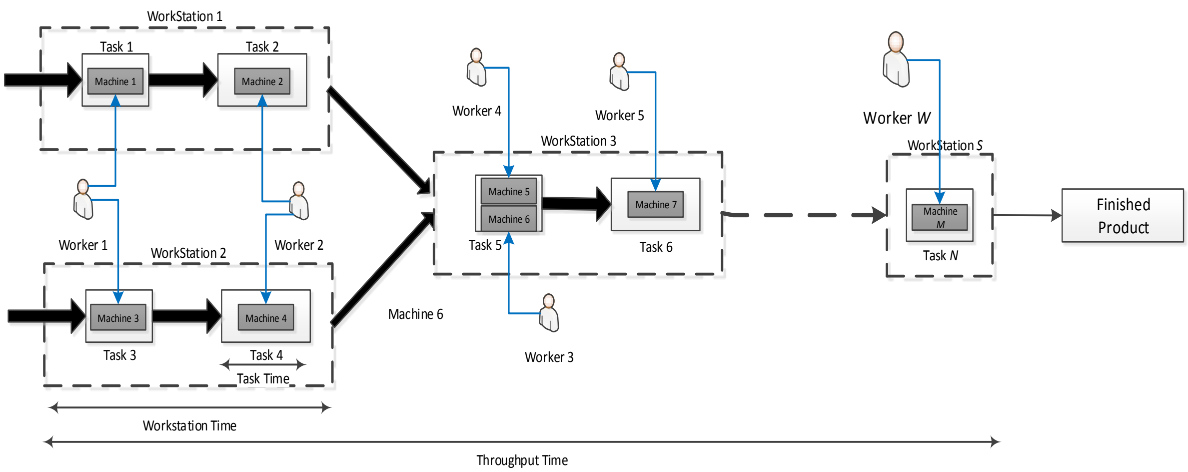

Figure 1 - 4: Visual Representation of Terms used in Line Balancing (Adapted from De Jong [8])

1.5 Work Sharing and Cross Training

Due to the competitive nature of the manufacturing industry, assembly lines are required to maintain high levels of productivity. This often results in an unequal distribution of workload among workers and hence an unbalanced line. In order to improve flexibility, resource utilization and throughput, many industries apply work sharing concepts [9]. Thus Workers are often cross trained so that they can perform more than one task, hence allowing work to be shared among workers on the line. This offers benefits such as reduced idle time, reduced buffer sizes, reduced risk of repetitive worker injuries, job enrichment and flexibility [10].

Hopp and Van Oyen [11] point out that cross trained workers represent flexible capacity and can be dynamically shifted wherever they are needed. They also describe how cross training helps to reduce labor cost by increasing labor productivity, reduce lead times, support better internal quality and increase the production flexibility of an organization.

1.6 Assembly Lines in the Apparel Industry:

Apparel manufacturing has evolved from the traditional system of hand-sewn garments to an industry which is constantly updating itself to keep up with the ever changing concepts of fashion. The sewing process, which is analogous to the assembly process of a traditional manufacturing line, is the most important process in this industry due to the high material and labor cost involved [12]. Therefore, the objective of line balancing in apparel manufacturing is to reduce cycle time or minimize idle time or minimize the number of workstations or maximize workload smoothness of the sewing process, thus minimizing cost and increasing the efficiency of production.

There are two types of imbalances that can occur. Static imbalance occurs due to a difference in capacity over a period of time. It may result in underutilization of workstations or machines or workers. Dynamic imbalance occurs over a short period of time and is caused by changes in product mix. Imbalance may also occur due to variations in operator efficiency and some unexpected factors such as absenteeism and machine breakdown. Thus, line balancing control is required to smooth the bottlenecks and improve the efficiency of the line [13].

1.7 Motivation

This research attempts to solve a variation of the traditional Assembly Line Balancing Problem that is described in Section 1.2. We consider a system where sewing machines can be reconfigured for every task (i.e., the fixtures on the sewing machines can be changed to perform every task). Therefore, workers serve as the main constraint, with each worker cross trained to do one or more tasks with a predefined efficiency. Only one task is assigned to a station, implying that the terms 'task' and 'station' are interchangeable. Hence, all machines assigned to a station perform the same task. The primary objective in our case is to obtain the optimal assignment of workers to tasks to maximize workload smoothness (by minimizing the mean absolute deviation of workload) of the line and satisfy demand/production for a given time period.

The aim of this research is to provide a comparative assessment of scenarios where work sharing is used and not used. We also study the effects of allowing workers to be idle in these scenarios (i.e., workers would have time to perform other duties that are not directly related to production of the product). We also compare the effect of using different levels of demand.

We present a pilot methodology for using simulation to convert a static system to a dynamic one in order to account for stochastic processing times. The results of the deterministic system are compared to the stochastic systems with ±10%, ±20% and ±30% variation in the Standard Operation Time based on a PERT distribution. We determine when a worker is required to move to another station based on the number waiting in queue (termed as buffer size) at each station. This is done by comparing the effect of two types of policies – no buffers and a common buffer size for all stations. We measure three responses - MAD, Actual Production and Total Worker Movement, to determine the "optimal" buffer size.

1.8 Thesis Summary

Chapter 2 describes some of the literature and techniques that have been used to solve ALBPs for a variety of industries. The model formulation is presented in Chapter 3. Smaller instances of the problem can be solved directly using CPLEX whereas larger versions cannot be solved optimally within reasonable computation times as seen in Chapter 3. Because large-scale problems cannot be solved optimally, a heuristic utilizing a Genetic Algorithm is developed to create good solutions in Chapter 4. Also, in Chapter 4, a real industrial case is solved. In Chapter 5, we use the results from these static models as input parameters for a simulation model that takes the stochastic nature of the system into account. We also compare the results of the static model with the simulation model and apply certain policies that dictate worker movement. Finally, Chapter 6 describes the conclusions and future work.

Chapter 2

Literature Review

The Assembly Line Balancing Problem (ALBP) and its variations have been extensively studied in the literature. In this chapter, a brief description of the literature and techniques that have been used to solve ALBPs is presented. Emphasis is given to research which is pertinent to the apparel industry.

2.1 Classification of ALBPs:

The first mathematical formulation of the ALBP was given by Salveson [14]. It focused primarily on the core problem of the configuration, which is the assignment of tasks to stations. Baybars [15] labels this type of basic formulation as the Simple Assembly Line Balancing Problem (SALBP) due to its simplicity and underlying assumptions.

Further research in this field provided a more realistic view of the assembly line and led to the inclusion of other factors such as parallel stations, U-shaped lines, processing alternatives, etc. which come under the category of Generalized Assembly Line Balancing Problems (GALBP) [4]. Ghosh and Gagnon [11] and Erel and Serin [12] provide early classifications of GALBPs.

The most comprehensive classification of ALBPs in the literature is by Boysen *et al.* [16] which follows a 3-tuple notation '[α |β |γ]. The α parameter is used to describe the precedence graph characteristics such as the structure (which may be acyclic or unique for a given problem). It also identifies sequence dependent tasks, assignment restrictions and processing alternatives. They also include processing times (which may be stochastic or deterministic) and the type of product (which may be single, multi or mix). The β parameter represents the station and line characteristics such as movement of work pieces, layout of the line, parallelization (if any), resource assignment, station dependent time increments (if any) and some additional configuration aspects such as buffers and feeder lines. The γ parameter specifies the objective function of the problem which may be minimizing the number of stations or minimizing cycle time or workload smoothing or cost minimization etc.

Boysen *et al.* [17] provides a detailed classification specifically for mixed model assembly lines. They review and discuss the three major approaches in mixed model assembly line balancing which are mixed model sequencing, car sequencing and level scheduling. They also include additional characteristics for stations and lines such as station boundaries, concurrent work, setup, etc.

2.2 Techniques used to solve ALBPs

Due to its computational complexity, the assembly line balancing problem is considered to be NP-hard [15]. Early researchers used Mathematical Programming and Branch and Bound techniques to solve the ALBP. However, large instances of the problem cannot be solved within reasonable computation times by using exact methods. For m tasks and r precedence constraints there are $m!/2r$ possible task sequences, resulting in a very large problem space [15].

More recent research shows the increasing popularity of meta-heuristics such as Tabu Search, Simulated Annealing, Genetic Algorithms, Particle Swarm optimization, Ant Colony optimization as well as hybrid methods.

Table 2 - 1 provides a brief overview of some the techniques that have been used to solve ALBPs. For a more detailed classification, please refer Boysen *et al.* [16].

Table 2 - 1: Summary of Techniques used to solve ALBPs

Technique	References
Mathematical Programming	
• Linear and Integer Programming	Bowman [18], Gökcen and Erel [19], Lamar [20], Peeters and Degraeve [21], Urban and Chiang [22]
• Dynamic Programming	Carraway [23], Held et. al [24], Henig [25], Lee and Uzsoy [26], Schrage and Baker [27]
Tree Search Algorithms	
• Branch and Bound	Berger *et al.* [28], Bukchin and Rabinowitch [29], Liu *et al.* [30], Miralles *et al.* [31], Tozkapan *et al.* [32]
Metaheuristics	
• Tabu Search	Chiang [33], Lapierre *et al.* [34], McMullen [35], Pastor *et al.* [36], Scholl and Voß [37]
• Simulated Annealing	Erel *et al.* [38], McMullen and Frazier [39], Mendes *et al.* [40], Suresh and Sahu [41], Vilarinho and Simaria [40]
• Ant Colony Optimization	Baykasoglu and Türkay [42], McMullen [43], McMullen and Tarasewich [44], Vilarinho and Simaria [45], Zheng *et al.* [46]
• Particle Swarm Optimization	Chutima and Chimklai [47], Dou and Zhao [48], Hamta *et al.* [49], Nearchou [50], Petropoulos and Nearchou [51]
• Genetic Algorithms	Ayazi *et al.* [52], Chan *et al.* [53], Chen *et al.* [4], Falkenauer and Delchambre [3], Guo *et al.* [54], Kim *et al.* [5], Kim & Kim [55], Lee [56], Moreira *et al.* [57], Haq *et al.* [58], Razali and Geraghty [59], Wong *et al.* [13], Yolmeh and Kianfar [60], Zhang & Gen [61]
Hybrid Algorithms and Models	Hamta *et al.*[49], Mendes *et al.* [40], Haq *et al.* [58] Tonyhsieh [62], Yolmeh and Kianfar [60]
Simulation Modeling	Bahadir [63], Guner and Unal [64], Kursun and Kalaoglu [12], Lee et. al. [56], Unal *et al.* [65]

It can be seen that there are a variety of methods that have been used solve the line balancing problem. The next subsections will describe the various techniques.

2.2.1 Tabu Search

Tabu Search is a local search algorithm that was developed by Glover [66]. The basic principle of the algorithm is to mark certain search paths as 'tabu' or 'forbidden' if they have been recently visited or violate some predefined conditions. These search paths are stored using memory structures to prevent the algorithm from visiting these paths repeatedly and converging to a local optimum. They often employ neighborhood structures and choose the best next neighbor that is not tabu. Therefore, the heuristic does not repeat paths or solutions that have been recently visited.

McMullen [35] presents a Tabu Search heuristic for sequencing mixed-model production schedules for a JIT production line with setup requirements. The technique seeks to minimize both material usage rates and setups and presents three different types of objective functions depending on the importance given to these two objectives. Pastor *et al.* [36] solved a Type 2 General Assembly Line Balancing Problem (GALBP-2) with ten workstations using Tabu Search. The primary objective of the type 2 problem is to maximize production rate for a given number of workstations. They also consider additional objectives such as minimizing the dispersion of worker tasks, obtaining an equal cycle time of all the models and an equal workload of the different workstations.

2.2.2 Simulated Annealing

Simulated Annealing is a probabilistic method that is used to find the global minimum of a cost function that may have several local minima. It was independently developed by Kirkpatrick *et al.* [67] and Černý [68]. Its name comes from the technique of annealing in metallurgy, which is a process by which a material is heated to a temperature above its critical temperature, maintained at that temperature and then slowly cooled. With respect to the algorithm, the slow cooling represents the gradual decrease of the probability of accepting worse solutions as the solution space is explored.

Suresh and Sahu [41] use Simulated Annealing to solve a set of problems where the objective is to minimize the smoothness index and another set of problems where the objective is to minimize the probability of line stopping. Smoothness Index (SI) is another performance measure for the relative smoothness of the assembly line. It is defined by the equation shown below. In this research, we use mean absolute deviation as a measure of workload smoothness.

$$SI = \sqrt{\sum_{k=1}^{n} (S_{max} - S_k)^2}$$

where n is the total number of workstations

S_{max} is the cycle time

S_k is the workstation time of station k.

Erel *et al.* [38] use Simulated Annealing to balance a U-line assembly system (UALBP). The performance of the algorithm is compared to Dynamic Programming and Integer Programming based algorithms and heuristic procedures and is found to be more efficient.

2.2.3 Ant Colony Optimization

Ant Colony Optimization (ACO) was proposed by Dorigo [69] and falls under the broader category of 'swarm intelligence' techniques that are used for optimization. It models the behavior of ants that move from their colony to a source of food and upon finding the source, they return to the colony leaving behind pheromone trails so that other ants can follow the same path and reinforce the trail. These trails evaporate over time and hence trails that are longer tend to evaporate quicker. In this way, the shortest trail is reinforced and indicated to be the optimal solution.

Baykasoglu and Dereli [42] examine a two-sided assembly line balancing problem with and without zoning constraints. These constraints force some tasks to be restricted to certain facilities of the line due to specific requirements such as size of the part, type of operation, etc. The authors apply an Ant Colony based algorithm with the objective to minimize the number of workstations and maximize the work relatedness where possible. Work relatedness is a performance indicator that measures the degree to which interrelated tasks are assigned to the same workstation. Vilarinho and Simaria [45] use Ant Colony Optimization to balance a mixed model assembly line of type I and type II with parallel stations and zoning constraints. The type I problem considers a dual objective of minimizing the number of workstations for a given cycle time and balancing the workload across workstations. They use idle time as one of the criteria to measure workload smoothness. The objective for the type II problem is to maximize the production rate of the line for a predetermined number of operators.

2.2.4 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a population-based stochastic optimization technique developed by Eberhart and Kennedy [70]. It is inspired by social behavior of bird flocking or fish schooling and is also a swarm intelligence technique. This method is similar to genetic algorithms in the sense that potential solutions are encoded as particles (instead of chromosomes). The position of each particle represents a candidate solution to the optimization problem. Particles move through the problem space to find better solutions by changing their “velocity”, which represents a magnitude and direction toward a new, presumably better, solution/position. This is dictated by their best known position in the search-space as well as the entire swarm's best known position.

Nearchou [50] applies a PSO approach to the simple assembly line balancing problem with two optimization criteria: minimizing the cycle time and maximizing workload smoothness among workstations. They also use MAD as a measure of workload smoothness. Chutima and Chimklai [47] use a variation of the traditional PSO algorithm by incorporating a concept known as negative knowledge which increases the probability of selecting better solutions. They consider a multi-objective problem where the primary objective is to minimize the line length, the secondary objective is to minimize the number of workstations and the tertiary objective consists of two conflicting sub objectives – maximizing work relatedness and maximizing workload smoothness. They use idle time as one of the criteria for workload smoothness.

2.2.5 Genetic Algorithms

A Genetic Algorithm (GA) is a computational technique that is inspired by evolutionary biology and the principle of 'survival of the fittest'. It was proposed by Holland [71] in 1975. The algorithm evaluates solutions (encoded as 'chromosomes') based on a 'fitness function', which is a measure of the quality of the solution, i.e., objective function value. These 'fit solutions' are then allowed to 'crossover', to produce offspring solutions, which are again evaluated by the 'fitness function'. Solutions that show no improvement compared to the parent solutions are eliminated. 'Mutations' and 'inversions' are also used to modify solutions in order to perform a more effective global search. In this way, the algorithm narrows down the search space to find an optimal solution.

Ayazi *et al.* [52] apply a GA to the simple assembly line balancing problem with multiple objectives – minimizing cycle time, minimizing the number of workstations and minimizing the workload deviation (which is measured in terms of idle time). Zhang and Gen [61] apply a generalized Pareto-based scale-independent fitness function genetic algorithm (gp-siffGA) to the mixed model assembly line balancing problem to minimize the cycle time, the variation of workload (which is measured by the smoothness index) and the total cost. They propose a new approach to calculate the cycle time which is based on the demand ratio of each model and another one to calculate the human resource cost.

Section 2.3 provides additional literature in which GAs have been applied to ALBPs in the apparel industry. A more detailed explanation of GAs is provided in Section Four.

2.2.6 Hybrid Algorithms

A Hybrid Algorithm combines two different techniques and tries to maintain the advantages of both. Hamta *et al.* [49] use a hybrid PSO algorithm with a local search to model the SALBP with flexible operation times, sequence-dependent setup times and worker learning.

They consider three objectives - minimizing the cycle time, minimizing the total equipment cost, and minimizing the smoothness index.

Yolmeh and Kianfar [60] apply a GA that is hybridized with a dynamic programming[60] procedure to the setup assembly line balancing and scheduling problem (SUALBSP) with the objective of minimizing the cycle time for a given number of stations. They assume forward and backward setup times between tasks. Other GA hybrids include GA and a tabu search heuristic [72] and a GA with a modified ranked positional weight method [58].

2.2.7 Simulation

Computer Simulation is a powerful tool that can be used to model and analyze various systems. With sufficient data and a suitable number of replications, a fairly accurate simulation model of a given system can give very realistic results. Simulation models can be developed using programming languages such as C, C++, Java, CPLEX etc. or using software packages such as ARENA, SIMIO or Witness etc. The basic procedure in most cases is to conduct time studies to gather sufficient input data and then to fit this data to probability distributions that can be used as inputs for the simulation model. The advantage of using simulation is the ability to generate values from these probability distributions, thus allowing for a stochastic formulation of the ALBP rather than a deterministic one.

Ünal *et al.* [65] provide a heuristic algorithm for line balancing and evaluate its effectiveness under both straight line and U-type line configurations using simulation. They first develop a simulation model of the existing system, followed by verification and validation, and then the implementation of the proposed heuristic to generate new line configurations. The performances of these new configurations are then evaluated using simulation. They conclude that U-type line configuration is more advantageous compared to straight line configuration according to both mean throughput per worker and also mean workstation utilization values. In this research, we only consider a straight line configuration.

2.3 The Assembly Line Balancing Problem in Apparel Manufacturing

Recent literature shows the increasing popularity of heuristic approaches to obtain good solutions to the line balancing problem in apparel manufacturing. One of the common approaches is to utilize a Genetic Algorithm (GA). Chan *et al.* [53] used a GA to minimize the total slack time (i.e., the sum of the idle times of all the stations) along the line. They consider a fully cross trained workforce with efficiencies ranging from 0 to 1.5, but workers are restricted to performing only one task (i.e., work sharing is not used). They compare the efficiency of the GA with a greedy algorithm using a 41 task example and show that the GA performs better than the greedy algorithm. The authors conclude that a GA is an effective tool to solve the assembly line balancing problem, especially in a dynamic environment like the apparel industry.

Wong *et al.* [13] use a GA to determine the optimal operative assignments of workers to tasks in order to minimize the makespan which is achieved by minimizing operator idle time. They use a six task example and compare the impact of a partially cross trained and fully cross trained workforce on the assembly makespan to determine the optimal number of task skills an operator should possess in the apparel assembly process. They allow multiple workers to be assigned to the same task, but restrict workers from being assigned to multiple tasks, whereas our formulation allows for both scenarios.

Guo *et al.* [54] investigate an example of the flexible assembly line balancing problem with work sharing and workstation revisiting. They use a bi-level genetic algorithm with multi-parent crossover to determine the assignment of tasks to workstations and the task proportion of each shared operation being processed on different workstations. A heuristic operation routing rule is used to route the shared operation of each product to an appropriate workstation when it should be processed. The GA uses a two-fold objective. The first one is used to satisfy the desired cycle time of each order while the second one minimizes the total idle time in each cycle. In their formulation, there are two types of machines which are allotted to a certain number of workstations. Certain tasks can be performed on type one machines and others on type two machines and the efficiencies associated with each task depend on the workstation it is assigned to. Each task can be split and assigned to one or more machines while also allowing for one task to be assigned to multiple machines. Our research follows a slightly different approach by focusing on how workers that are cross trained are split between tasks based on the time available and also examines scenarios where workers are allowed to be idle. The efficiency associated with a given task depends on the worker that has been assigned to it. The objective in our case is to maximize workload smoothing and satisfy demand for the given time period.

Chen *et al.* [4] also focus on maximizing workload smoothing based on Mean Absolute Deviation. They utilize a Grouping Genetic Algorithm (GGA) and use real data from a garment industry to evaluate its performance. The initial population of the GGA is generated by a heuristic that arranges the workstations in ascending order of available processing times. In their formulation, they group tasks into task sets and workers into labor types. Each labor type has an efficiency associated with it and can only operate a specific type of machine, which in turn can only be used for specific tasks. They use worker skill levels to indicate the number of tasks that can be done by a particular worker. They solve a simple problem where 34 tasks are assigned to 20 workstations and a complex problem where 176 tasks are assigned to 54 workstations. ANOVA is used to determine the effect of factors such as number of operators in workstations, operator skill level, and operator efficiency. They conclude that assigning more operators to workstation and training operators with multiple skills and high efficiency can result in low MAD, short cycle time, and high throughput, and assigning more operators to workstation and training operators with multiple skills can also increase operator utilization.

Askin and Chen [10] consider a case where workers are partially cross-trained and certain tasks can alternate between stations. Their objective was to define the tradeoff between the cost of work-in-process inventory and the cost of cross-training. They provide a simulation model and various heuristic rules that attempt to approximate the optimal threshold for WIP buffer sizes. The key difference in their model is that tasks are shared between stations;

whereas this research assumes that workers are trained to do one or more tasks, and multiple workers can execute multiple tasks provided there are enough machines in the system.

As can be seen, the ALBP has been extensively studied in a variety of industries as well as the apparel industry. However, they make several assumptions about the problem which are not present in our case and will be discussed in the next chapter.

Chapter 3

Model Formulation

In this chapter we formulate a mathematical model of the problem and demonstrate the efficiency of CPLEX by solving an industrial example as well as small and large instances of randomly generated problems.

Previous research has focused on the assignment of tasks to workstations to minimize cycle time or the assignment of tasks to workstations to minimize the number of workstations. In this formulation, we consider an objective that has been comparatively neglected in literature [73] which is maximizing the workload smoothness of the line. The primary constraint in our case is the assignment of skilled workers to tasks based on whether the worker is trained to do that particular task. We also use a demand based formulation of the problem as opposed to a cycle time based formulation to determine if the assignment of workers to tasks is able to meet the demand for a given time period as well as maximize the workload smoothness of the line. Note that a cycle time based formulation can be constructed by dividing the total time available by the demand required.

3.1 Methodology

We examine two types of precedence graphs, and for each type we examine different scenarios with respect to worker utilization, maximum number of task assignments and demand.

The two types we examine are:

a. *Linear:* With the exception of the first task, each task is preceded by only one task as shown in Figure 3 - 1.

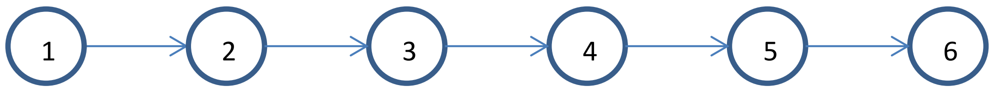

Figure 3 - 1: Linear Structure

b. *Branched:* With the exception of the first task(s), each task may be preceded by one or more tasks as shown in Figure 3 - 2.

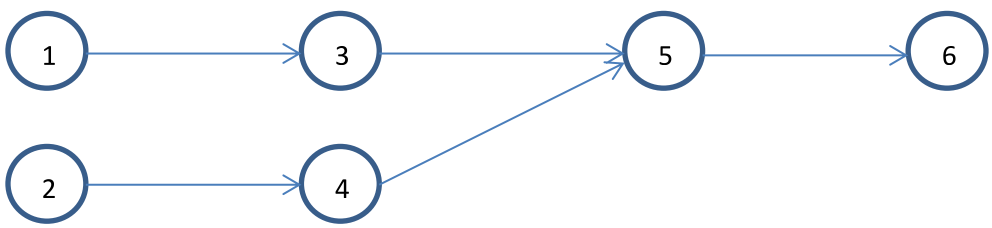

Figure 3 - 2: Branched Structure

Note that for a given number of tasks, multiple branched structures can be constructed, whereas only one type of linear structure can exist. This makes it hard to directly compare the linear model structure with the set of all possible branched model structures. In this research we only compare the branched structures that we have defined by the precedence graphs shown in Appendix A.

We consider the following factors in our model:

1. Worker Utilization:

a. *100%*: Workers are not allowed to be idle. In scenarios where the demand is low and the time available for workers is high, forcing utilization to be 100% may result in producing more units than required. From a lean point, this waste is associated with overproducing.

b. *<100%*: Workers are allowed to be idle a portion of the time. Therefore, they are allowed to work on other assignments for this idle time (e.g., machine maintenance, restocking of materials, etc.)

2. Maximum Number of Tasks:

a. *N* Tasks: Work sharing is allowed (i.e., one worker can be assigned to more than one task). In our real case, workers are allowed to perform a maximum of two tasks.

b. Only One Task: Workers follow a fixed assignment (i.e., each worker while cross trained can be assigned to only one task).

3. Demand:

Two levels of demand are considered for the specified time period. For the industrial case we use 9,500 and 11,000 for a time period of 11 hours and for the computational study we use 180 and 220 for a time period of 24 hours. These levels of demand were chosen based on the task times specified in the industrial example and computational study.

A summary of these factors is shown in Table 3 - 1.

Table 3 - 1: Summary of Factors

Factor	**Levels**	
Utilization	100%	<100%
Assignment	2 Tasks	1 Task
Demand	9500 for Industrial Example 180 for Computational Study	11000 for Industrial Example 220 for Computational Study

3.2 Assumptions

The following are the assumptions of our model with any limitations explained.

1. Only one task is assigned to a station. Hence, all machines assigned to a station are configured to perform the same task. This is not limiting in our apparel case as the work element and task are the same. For example, a collaring operation cannot be performed on a machine setup for sleeve production.
2. Worker movement is dictated by the layout of the stations. Tasks have to be defined and the distances between stations/tasks are known ahead of time. Again, the tasks have already been broken down into their simplest elements and the goal is to assign workers to those tasks so as to balance the line.
3. Sewing machines are identical and can be reconfigured for every task (i.e., the fixtures on the sewing define the operation and task).
4. Operation times are deterministic and inclusive of any setup time.
5. Efficiency and skill of workers for each task is known and constant (i.e., no learning curves are utilized).
6. Machine breakdown or operator absenteeism is not considered.
7. There is a minimum percentage of time a worker can be assigned to a station.

3.3 Nomenclature

Indices

1. Stations (tasks) indexed from $s = 1$ to S
2. Machines indexed from $m = 1$ to M
3. Workers indexed from $w = 1$ to W

Constants

N = Total number of tasks

M = Total number of machines available for distribution

S = Total number of stations

W = Total number of workers

$Pred\ \{i\}$ = Set of predecessor tasks for task i

O_i = Operation time of task i

R_{min} = Threshold minimum percentage of time that a worker that can be assigned to a task

E_{wi} = efficiency of worker w performing task i

T_{system} = Total time available in system for a given time period

$T_{move(sk)}$ = Time taken to move from station s to station k

C_{ws}= Capability of worker w at station s= $\begin{cases} 1 \text{ if worker w can be assigned to station s} \\ 0 \text{ otherwise} \end{cases}$

D = Demand required to be produced over a given time period (i.e., T_{system})

$N_{MaxTasks}$ = Maximum number of allowable assignments for all workers

Variables

1. $x_{ws}= \begin{cases} 1 \text{ if worker } w \text{ is assigned to station } s \\ 0 \text{ otherwise} \end{cases}$
2. $y_{ms}= \begin{cases} 1 \text{ if machine } m \text{ is assigned to station } s \\ 0 \text{ otherwise} \end{cases}$
3. R_{ws} = Percentage of time that worker w is assigned to station s (ranges from R_{min} to 1)
4. $V_{wsk}= \begin{cases} 1 \text{ if worker } w \text{ moves from station } s \text{ to station } k \\ 0 \text{ otherwise} \end{cases}$
5. T_w = Total time available for worker w and is defined by the following constraint.

$$T_w = T_{system} - \sum_{s}^{S\text{-}1} \sum_{k=s+1}^{S} V_{wsk} T_{move(sk)}$$

6. P_s^T = Theoretical production capability of station s

$$P_s^T = \sum_{w}^{W} x_{ws} R_{ws} \left[\frac{T_w}{O_i/E_{wi}}\right] \qquad \forall s$$

7. P_s = Actual production capability of station s

$$P_s = \min\left(P_s^{\mathrm{T}}, \{P_s\}_{pred}\right)$$

where $\{P_s\}_{\mathrm{pred}}$ = Actual Production of set of preceding stations

8. P_{Last} = Actual Production of last station

9. $\overline{P}$ = Mean Production of all stations

$$\overline{P} = \frac{1}{s}\sum_{s=1}^{S} P_s$$

10. m_s = Total number of machines assigned to station s

$$m_s = \sum_{s=1}^{S} y_{ms} \qquad \forall s$$

3.4 LP Formulation

$$minimize\ MAD = \frac{1}{s}\sum_{s=1}^{S} |P_s - \bar{P}| \tag{1}$$

Subject to the following,

$$P_s \leq P_s^{\mathrm{T}} \qquad \forall s \tag{2}$$

$$P_{\mathrm{Last}} \geq D \tag{3}$$

$$\sum_{s=1}^{S} m_s \leq M \qquad \forall s \tag{4}$$

$$\sum_{w=1}^{W} x_{ws} \leq m_s \qquad \forall s \tag{5}$$

$$\sum_{s=1}^{S} R_{ws} x_{ws} \leq 1 \qquad \forall w \tag{6}$$

$$R_{ws} x_{ws} \geq R_{min} \qquad \forall s \tag{7}$$

$$R_{ws} \leq C_{ws} \tag{8}$$

$$x_{ws} \geq R_{ws} \tag{9}$$

$$V_{wsk} \geq x_{ws} + x_{wk} - 1 \qquad \forall w, s \; from \; s = 1 \; to \; S - 1 \; and \; k \; from \; s + 1 \; to \; S \tag{10a}$$

$$V_{wsk} \leq x_{ws} \qquad \forall w, s \; from \; s = 1 \; to \; S - 1 \; and \; k \; from \; s + 1 \; to \; S \tag{10b}$$

$$V_{wsk} \leq x_{wk} \qquad \forall w, s \; from \; s = 1 \; to \; S - 1 \; and \; k \; from \; s + 1 \; to \; S \tag{10c}$$

$$\sum_{s=1}^{S} x_{ws} \leq N_{MaxTasks} \quad \forall w \tag{11}$$

Eq. (1) is the objective function, given by the Mean Absolute Deviation (MAD) which is typically used as a measure of workload smoothness [5,6]. Eq. (2) constrains the actual production to be less than or equal to its theoretical maximum production. Eq. (3) ensures that the production of the last station is able to satisfy demand for the given time period. Eq. (4) constrains the total number of assigned machines assigned to be less than or equal to the total number of machines that exist in the system. Eq. (5) guarantees that workers can be assigned to a station only if enough machines have been assigned to that station. Eq. (6) constrains the total utilization of a worker to be less than or equal to 100%. If we force 100% utilization, then this would be an equality constraint. Eq. (7) ensures that the percentage of time that a worker is assigned to a station is always greater than or equal to the threshold minimum. Eq. (8) allows a worker to be assigned to a station only if the worker is capable of performing the task at that station. Eq. (9) considers a worker to be assigned to a station if that worker is partially assigned to that station. Eqs. (10a), (10b) and (10c) ensures that a worker can move from station *s* to station *k* only if he has been assigned to both station *s* and station *k*. Eq. (11) provides an upper bound on the maximum number of task assignments for all workers. If the maximum number of tasks is greater than one, workers are allowed to share their time between multiple tasks thus cutting down on idleness.

3.5 Optimization using CPLEX

We first examine a real world case as an example and then provide a computational study that examines the efficiency of CPLEX by solving small and large instances of randomly generated problems.

We use IBM ILOG CPLEX Optimization Studio v 12.4 on a PC with an Intel i7 Quad Core Processor; 3.40 GHz CPU and 8 GB RAM to solve these cases.

3.5.1 Industrial Example

As an example, we consider a real apparel line with constants shown in Table 3 - 2. Each task has a standard deterministic processing time (as shown in Table 3 - 3). Each worker is trained to perform 1 or 2 of the 11 tasks with varying efficiencies as shown in Table 3 - 4. The model assumes that only one worker can work on a given task on a given machine at any point in time. Multiple workers can work on the same task provided that there are enough machines in the system. The time taken to move between stations is shown in Table 3 - 5.

Table 3 - 2: Apparel Line Constants

Constant	Description	Value
N	Total number of tasks	11
W	Total number of workers	39
M	Total number of available machines	100
R_{min}	Threshold minimum for partial assignment	25%
T_{system}	Time available in system for given period	660 min

Table 3 - 3: Standard Processing Time

Tasks	Standard Processing Time (s)		Tasks	Standard Processing Time (s)
T1	3.48		T7	15.84
T2	12.6		T8	7.26
T3	9.72		T9	9.72
T4	4.14		T10	26.88
T5	12.66		T11	11.7
T6	7.92			

Table 3 - 4: Worker Skills

Worker	Task	Efficiency		Worker	Task	Efficiency		Worker	Task	Efficiency
W1	T1	1		W19	T7	1		W29	T10	1
W2	T2	1		W20	T7	1		W30	T8	1
W3	T2	1		W21	T7	1		W30	T9	1
W4	T2	1		W22	T7	1		W30	T10	1
W5	T2	1		W23	T8	1		W31	T8	1
W6	T3	1		W23	T9	1		W31	T9	1
W7	T3	1		W23	T10	1		W31	T10	1
W8	T3	1		W24	T8	1		W32	T8	1
W9	T4	1		W24	T9	1		W32	T9	1
W9	T5	1		W24	T10	1		W32	T10	1
W10	T4	1		W25	T8	1		W33	T8	1
W10	T5	1		W25	T9	1		W33	T9	1
W11	T4	1		W25	T10	1		W33	T10	1
W11	T5	1		W26	T8	1		W34	T8	1
W12	T4	1		W26	T9	1		W34	T9	1
W12	T5	1		W26	T10	1		W34	T10	1
W13	T4	1		W27	T8	1		W35	T8	1
W13	T5	1		W27	T9	1		W35	T9	1
W14	T4	1		W27	T10	1		W35	T10	1
W14	T5	1		W28	T8	1		W36	T11	1
W15	T6	1		W28	T9	1		W37	T11	1
W16	T6	1		W28	T10	1		W38	T11	1
W17	T6	1		W29	T8	1		W39	T11	1
W18	T7	1		W29	T9	1				

Table 3 - 5: Movement Time Matrix (in minutes)

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11
T1	0	4	8	12	16	20	24	28	32	36	40
T2	4	0	4	8	12	16	20	24	28	32	36
T3	8	4	0	4	8	12	16	20	24	28	32
T4	12	8	4	0	4	8	12	16	20	24	28
T5	16	12	8	4	0	4	8	12	16	20	24
T6	20	16	12	8	4	0	4	8	12	16	20
T7	24	20	16	12	8	4	0	4	8	12	16
T8	28	24	20	16	12	8	4	0	4	8	12
T9	32	28	24	20	16	12	8	4	0	4	8
T10	36	32	28	24	20	16	12	8	4	0	4
T11	40	36	32	28	24	20	16	12	8	4	0

The results obtained are shown in Table 3 - 6. Solutions marked in green indicate optimal solutions. It can be seen that CPLEX is able to find an optimal solution to all cases of this problem. CPLEX is also able to indicate within 0.02 seconds that no feasible solution exists for two of these cases. These cases are infeasible as the demand is too high if a single task assignment is forced.

Table 3 - 6: Results from the Industrial Case

Utilization	Assignment	Demand	MAD	Computation Time (s)
100%	1 Task	9500	217.5207	15.5
100%	1 Task	11000	No Feasible Solution Exists	0.01
100%	2 Tasks	9500	0	653.3
100%	2 Tasks	11000	0	54.21
<100%	1 Task	9500	0	7.59
<100%	1 Task	11000	No Feasible Solution Exists	0.02
<100%	2 Tasks	9500	0	18.62
<100%	2 Tasks	11000	0	22.1

3.5.2 Computational Study

The previous section looked at one simple industrial case where CPLEX was able to solve the problem. In this section, we test its efficiency on five sets of randomly generated problems of different sizes. We use the factors specified in Table 3 - 1.

The precedence graphs, standard processing times, worker skills and movement times for this computational study are given in Appendix A. 'Set0' is the base problem set for a given problem. The standard processing times of the other sets were randomly generated using a uniform distribution with the lower and upper bound specified by the smallest and largest processing times of Set0 respectively. Five different sets (Set0 through Set4) of processing times were created for each problem to make sure that the efficacy of CPLEX is independent of the processing times. It should be noted that a time limit of three hours (10,800 seconds) was used to terminate the algorithm if it was still branching.

The results of the computational study for the linear and branched models are shown in Table 3 - 8 and Table 3 - 9 respectively. The legend for the results is shown in Table 3 - 7.

Table 3 - 7: Legend for Results

Color	**Description**
	Optimal Solution
NFSE	No Feasible Solution Exists
	Best Solution found after 3 hours (10,800 seconds)
NFSF	No Feasible Solution Found after 3 hours (10,800 seconds)

It can be seen that CPLEX is efficient at solving the smaller six task instances for both linear and branched cases. It also works well for the 12 task problems and for most branched 24 task problems. However, it is unable to find a feasible solution for the linear 24 task problems.

Table 3 - 8: Results from the Linear Model

Number of Tasks	Utilization	Assignment	Demand	Set 0		Set 1		Set 2	
				Objective Function	**Computation Time (s)**	**Objective Function**	**Computation Time (s)**	**Objective Function**	**Computation Time (s)**
6	100%	1 Task	180	2.5	3.12	45.556	2.42	0	0.77
	100%	1 Task	220	2.5	3.44	NFSE	0.88	0	0.73
	100%	2 Tasks	180	0	0.8	0	0.86	0	1.4
	100%	2 Tasks	220	0	2.09	0	0.81	0	0.92
	<100%	1 Task	180	0	0.81	0	0.71	0	0.77
	<100%	1 Task	220	0	0.7	NFSE	0.88	0	0.76
	<100%	2 Tasks	180	0	0.76	0	0.89	0	0.76
	<100%	2 Tasks	220	0	0.8	0	0.72	0	0.78
12	100%	1 Task	180	NFSE	14.36	NFSE	16.84	10.625	66.64
	100%	1 Task	220	NFSE	0.04	NFSE	18.06	NFSE	0.05
	100%	2 Tasks	180	0	24.52	0	20.69	0	15.71
	100%	2 Tasks	220	NFSE	20.05	NFSF	10800	0	19.69
	<100%	1 Task	180	NFSE	14.97	NFSE	17.37	0	11.16
	<100%	1 Task	220	NFSE	0.06	NFSE	19	0	0.07
	<100%	2 Tasks	180	0	11.84	0	11.35	0	11.48
	<100%	2 Tasks	220	NFSE	20.75	NFSF	10800	0	25.31
24	100%	1 Task	180	NFSF	10800	NFSF	10800	NFSF	10800
	100%	1 Task	220	NFSF	10800	NFSF	10800	NFSF	10800
	100%	2 Tasks	180	NFSF	10800	NFSF	10800	NFSF	10800
	100%	2 Tasks	220	NFSF	10800	NFSF	10800	NFSF	10800
	<100%	1 Task	180	NFSF	10800	NFSF	10800	NFSF	10800
	<100%	1 Task	220	NFSF	10800	NFSF	10800	NFSF	10800
	<100%	2 Tasks	180	NFSF	10800	NFSF	10800	NFSF	10800
	<100%	2 Tasks	220	NFSF	10800	NFSF	10800	NFSF	10800

Table 3 – 8 continued

Number of Tasks	Utilization	Assignment	Demand	Set 3		Set 4	
				Objective Function	Computation Time (s)	Objective Function	Computation Time (s)
6	100%	1 Task	180	14.222	2.8	10.5	3.79
	100%	1 Task	220	14.222	2.75	10.5	4.28
	100%	2 Tasks	180	0	0.8	0	0.92
	100%	2 Tasks	220	0	0.78	0	0.83
	<100%	1 Task	180	0	0.79	0	0.76
	<100%	1 Task	220	0	0.89	0	0.79
	<100%	2 Tasks	180	0	0.78	0	0.75
	<100%	2 Tasks	220	0	0.78	0	0.76
12	100%	1 Task	180	NFSE	14.51	0	10.73
	100%	1 Task	220	NFSE	26.8	NFSE	32.81
	100%	2 Tasks	180	0	13.78	0	13.95
	100%	2 Tasks	220	NFSF	10800	0	13.05
	<100%	1 Task	180	NFSE	14.41	0	10.77
	<100%	1 Task	220	NFSE	26.98	NFSE	36.76
	<100%	2 Tasks	180	0	10.72	0	12.91
	<100%	2 Tasks	220	NFSF	10800	0	14.38
24	100%	1 Task	180	NFSF	10800	NFSF	10800
	100%	1 Task	220	NFSF	10800	NFSF	10800
	100%	2 Tasks	180	NFSF	10800	NFSF	10800
	100%	2 Tasks	220	NFSF	10800	NFSF	10800
	<100%	1 Task	180	NFSF	10800	NFSF	10800
	<100%	1 Task	220	NFSF	10800	NFSF	10800
	<100%	2 Tasks	180	NFSF	10800	NFSF	10800
	<100%	2 Tasks	220	NFSF	10800	NFSF	10800

Table 3 - 9: Results from the Branched Model

Number of Tasks	Utilization	Assignment	Demand	Set 0		Set 1		Set 2	
				Objective Function	Computation Time (s)	Objective Function	Computation Time (s)	Objective Function	Computation Time (s)
6	100%	1 Task	180	16.889	2.85	0.889	2.77	43.333	3.6
	100%	1 Task	220	35.722	3.73	75	0.72	43.333	5.63
	100%	2 Tasks	180	0	3.55	0	5.74	0	6.81
	100%	2 Tasks	220	0	2.53	0	2.4	0	3.58
	<100%	1 Task	180	0	0.64	0	0.64	0	0.58
	<100%	1 Task	220	0	0.6	0	0.61	0	0.58
	<100%	2 Tasks	180	0	0.72	0	0.64	0	0.73
	<100%	2 Tasks	220	0	0.67	0	0.66	0	0.71
12	100%	1 Task	180	NFSE	17.63	39.111	10800	32	10800
	100%	1 Task	220	NFSE	0.03	NFSE	14.61	NFSE	0.05
	100%	2 Tasks	180	4.305	10800	10.972	10800	0	4329.22
	100%	2 Tasks	220	NFSE	14.32	8.833	10800	NFSE	112.71
	<100%	1 Task	180	NFSE	21.25	0	6.56	0	6.53
	<100%	1 Task	220	NFSE	0.02	NFSE	16.43	NFSE	0.08
	<100%	2 Tasks	180	0	6.75	0	6.68	0	6.51
	<100%	2 Tasks	220	NFSE	19.06	0	6.45	NFSE	113.5
24	100%	1 Task	180	59.955	10800	39.5	10800	37.916	10800
	100%	1 Task	220	NFSF	10800	NFSF	10800	62	10800
	100%	2 Tasks	180	43.618	10800	23.472	10800	60.215	10800
	100%	2 Tasks	220	40.482	10800	NFSF	10800	37.656	10800
	<100%	1 Task	180	0	9741.09	0	7461.22	0	10800
	<100%	1 Task	220	NFSF	10800	NFSF	10800	0	10364.79
	<100%	2 Tasks	180	0.319	10800	0	6241.35	13.569	10800
	<100%	2 Tasks	220	0	10800	0	2304.52	0	10800

Table 3 – 9 continued

Number of Tasks	Utilization	Assignment	Demand	Set 3		Set 4	
				Objective Function	Computation Time (s)	Objective Function	Computation Time (s)
6	100%	1 Task	180	58.222	3.21	20.333	2.66
	100%	1 Task	220	NFSE	3.98	20.333	2.02
	100%	2 Tasks	180	0	1.53	0	2.96
	100%	2 Tasks	220	0	1072.29	0	3.77
	<100%	1 Task	180	0	0.68	0	0.67
	<100%	1 Task	220	NFSE	4.25	0	0.68
	<100%	2 Tasks	180	0	0.64	0	0.66
	<100%	2 Tasks	220	0	0.68	0	0.66
12	100%	1 Task	180	NFSE	8.22	33.028	10800
	100%	1 Task	220	NFSE	9.19	33.028	10800
	100%	2 Tasks	180	9.708	10800	12.5	10800
	100%	2 Tasks	220	NFSE	28.01	8.889	10800
	<100%	1 Task	180	NFSE	8.36	0	6.6
	<100%	1 Task	220	NFSE	9.24	0	6.27
	<100%	2 Tasks	180	0	6.48	0	6.54
	<100%	2 Tasks	220	NFSE	27.67	0	13.71
24	100%	1 Task	180	67.291	10800	55.417	10800
	100%	1 Task	220	74.104	10800	55.222	10800
	100%	2 Tasks	180	71.489	10800	14.042	10800
	100%	2 Tasks	220	13.763	10800	24.34	10800
	<100%	1 Task	180	0	7250.94	0	4977.3
	<100%	1 Task	220	0	10800	0	9389.36
	<100%	2 Tasks	180	11.763	10800	1.458	10800
	<100%	2 Tasks	220	4.632	10800	0	9174.62

Chapter 4

Genetic Algorithms

The results from the previous section demonstrate that CPLEX is unable to find a good solution within a reasonable amount of time for larger instances of the problem. This research focuses on solving problems for a large apparel company for which the problem size ranges from 11 to 24 tasks. Thus, a heuristic technique was developed to find a good solution for these instances.

Recent literature has shown that one of the fairly common and powerful heuristics for a wide variety of applications is a Genetic Algorithm (GA) [74]. It is a computational technique that is inspired by evolutionary biology and the principle of ‘survival of the fittest’. Section 4.1 provides a detailed explanation of GAs. Figure 4 - 1 shows a simple GA and Figure 4 - 2 shows the basic structure of a GA.

The algorithm evaluates solutions (encoded as ‘chromosomes’) based on a ‘fitness function’, which represents the objective function of the given optimization problem. A portion of these solutions are ‘selected’ and allowed to ‘crossover’ and undergo ‘mutation’ to produce offspring solutions which are again evaluated by the ‘fitness function’. These new solutions are then used for the next iteration (termed ‘generation’) of the algorithm. This procedure is repeated for a certain number of generations or till an optimal solution is found.

Most algorithms tend to get confined to local optimums when applied to problems with a complex fitness landscape. GA’s on the other hand are efficient at solving problems with a large solution space and are able to traverse the landscape of the function with a different approach. The GA initially creates a population that represents various points on the landscape. Through crossover, mutation and selection, the algorithm is able to find completely new solutions and solutions within the vicinity of the initial population. Although there is no way to determine if the solution to a problem is a global optimum or a very good local optimum, in most cases, the GA returns very good solutions.

1. Set generation counter $i \leftarrow 0$.
2. Create the initial population, P_0, by randomly generating N individuals.
3. Determine the fitness of each individual in the population by applying the objective function to the individual and recording the value found.
4. Increment to the next generation, $i \leftarrow i + 1$.
5. Create the new population, P_i, by selecting N individuals stochastically based on the fitness from the previous population, P_i-1.
    a. Randomly select R parents from the new population to form the new children by application of the genetic operators (Crossover, Mutation, and Inversion).
    b. Evaluate the fitness of the newly formed children by applying the objective function.
6. If i < the maximum number of generations to be considered, go to Step 4.
7. Output the best solution found.

Figure 4 - 1: A Simple Genetic Algorithm

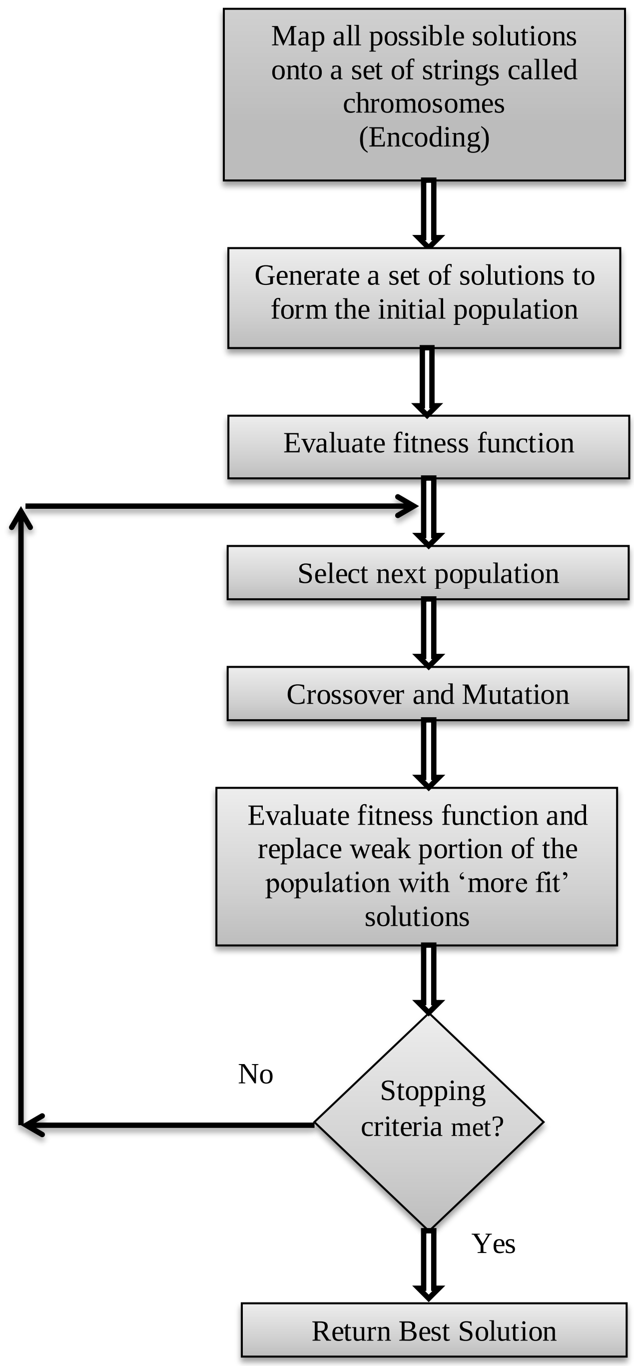

Figure 4 - 2: Structure of a GA

4.1 GA Specification

To develop a GA solution to any problem, one needs to define seven fundamental properties: chromosome representation, initial population, selection method, genetic operators, termination criteria, fitness function, and the GA Parameters.

4.1.1 Chromosome Representation

For any GA, a chromosome representation is needed to describe each individual in the population. Chromosome representation determines how the problem is structured in the GA, as well as the genetic operators that can be used. The problem has to be encoded and the type of encoding depends on the type of problem to be solved. Some of the common types of encoding are binary, floating point (real) and tree encoding [75].

For the variation of the ALBP problem, the GA in this research uses a real encoding scheme which is represented as

$$\left[\{X_i\} \,\middle|\, \{X_{i+1}\} \,\middle|\, \ldots. \,\middle|\, \{X_{N\text{-}1}\} \,\middle|\, \{X_N\}\right]$$

where $\{X_i\}$ represents the set of workers that are trained to do task i and N represents the number of tasks. Each set $\{X_i\}$ consists of a variable x_{ij} which ranges from zero to one and represents the percentage of time task i has been assigned to worker j. Consider an example of a manufacturing system with four tasks and four workers where each worker is capable of performing at two different tasks as shown in Table 4 - 1.

Table 4 - 1: Worker Skills (Example)

Worker	**Task**
1	A, C
2	B, C
3	A, D
4	B, D

Using this information, the chromosome is represented as seen in Figure 4 - 3.

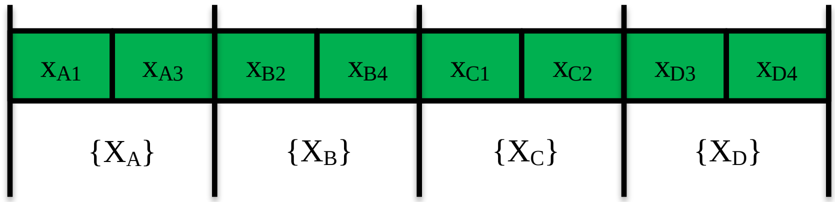

Figure 4 - 3: General Chromosome Representation

If we consider an example where workers are allowed to be idle, we could have a chromosome as shown below:

$$\langle 0.25, 0.75 | 0.5, 0.3 | 0.5, 0.4 | \, 0, 0.7 \, \rangle$$

Task A has been assigned to worker one 25% of the time and worker three 75% of the time. Task B has been assigned to worker two 50% of the time and worker four 30% of the time. Task C has been assigned to worker one 50% of the time and worker two 40% of the time. Task D has been assigned to worker four 70% of the time. Conversely, worker one has been assigned to Task A 25%, Task C 50% of the time and will be idle 25% of the time while worker two has been assigned 50% and 40% of the time to Task B and C, respectively. Worker three is assigned to Task A 75% of the time only while Worker four is assigned to Task B 30% of the time and Task D 70% resulting in a utilization of 100%.

4.1.2 Generation of Initial Population

The initial population may be seeded with solutions from other algorithms or existing solutions to the problem. The most common method is to generate the initial population by complete random seeding, which is employed in this research by randomly generating numbers between zero and one which are the lower and upper bounds respectively. This allows for a better coverage of the solution space.

4.1.3 Selection Scheme

Similar to nature's concept of 'survival of the fittest', chromosomes are chosen for reproduction by a defined set of selection rules based on their fitness. Chromosomes that are more fit have a higher chance of being selected than others, but even the poorer chromosomes have some non-zero probability of being selected. There are several types of selection schemes that determine and assign selection probabilities. Roulette wheel selection, tournament selection and ranking selection are a few examples. For a description of these schemes and others, refer to Michalewicz [75].

The GA in this research uses normalized geometric ranking as the selection mechanism. It ranks chromosomes from best to worst depending on their fitness value and then assigns a probability of selection based on the normalized geometric distribution [76].

$$\text{P(Selecting } i^{th} \text{ individual} = q'(1-q)^{i-1}$$

where,

$$q' = \frac{q}{(1-(1-q)^N)}$$

q is the probability of selecting the best individual,

i is the rank of the individual where one is the best

N is the size of the population

In our experiments q is set to 0.08 which is a standard probability of selecting the best individual [77].

4.1.4 Genetic Operators

Genetic operators are used to carry out the process of reproduction. The type of operator that can be applied depends on the encoding of the chromosome. There are three types of genetic operators – crossover, mutation and inversion. In this research, the crossover and mutation operators are utilized as inversion is typically used in order based representations.

Crossover:

The crossover operator produces child chromosomes by combining information from two parent chromosomes. For example, the simple crossover is performed by choosing a random crossover site for the parent chromosomes. The chromosome is then 'split' at that point and the two offspring is obtained by joining together non-corresponding parts of each parent.

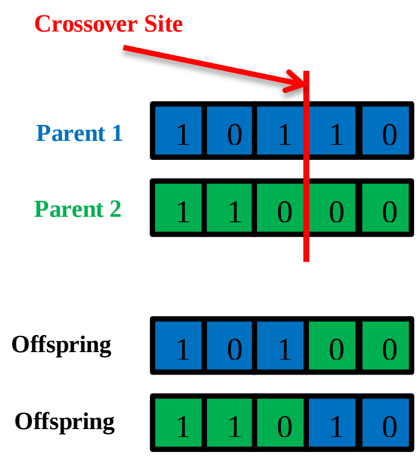

Figure 4 - 4: Simple Crossover

The GA in this research incorporates three types of crossovers which are described in Table 4 - 2 [76]. It is possible to generate an offspring that is not feasible. The mechanism to handle this is explained later in section 4.2.

Table 4 - 2: Types of Crossover

Crossover Type	**Description**
Simple Crossover	Randomly splits two parent chromosomes at a point and forms two new child chromosomes by attaching the 1st segment of the 1st parent to the 2nd segment of the 2nd parent and the 2nd segment of the 1st parent to the 1st segment of the 2nd parent. (Shown in Figure 4 - 4)
Arithmetic Crossover	Uses random proportions of parent chromosomes to produce a complimentary pair of linear combinations that become child chromosomes. Consider the chromosomes $C1 = aP1 + bP2,$ $C2 = bP1 + aP2,$ where $P1$ and $P2$ are the parent chromosomes, and $0 \leq a \leq 1$ and $b = 1 - a$
Heuristic Crossover	Produces a child chromosome that is a linear extrapolation away from the better parent along the direction of the vector joining the two parents. This type of crossover helps the GA to move into a new part of the solution space, though in some cases, it may generate an infeasible solution.

Mutation:

A mutation can be defined as a random modification of a chromosome. The mutation operator is very useful because it allows the GA to explore a part of the search space that is not part of the current population, thus helping to not get stuck at a local optimum. In Figure 4 -5, a bit in the chromosome is randomly selected and set to another value.

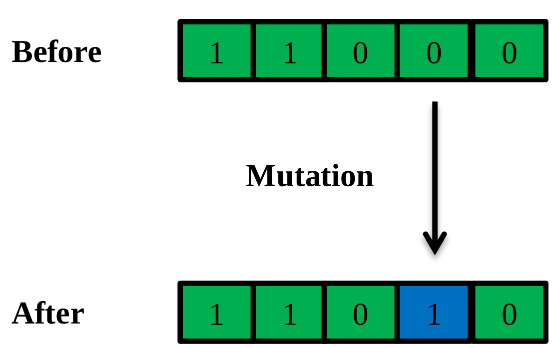

Figure 4 - 5: Mutation Mechanism

The GA incorporates four types of mutations which are described in Table 4 - 3 [76]. As with crossover operators, it is possible to generate an offspring that is not feasible. The mechanism to handle this is explained later in section 4.6.

Table 4 - 3: Types of Mutation

Mutation Type	**Description**
Boundary Mutation	Randomly selects one of the variables and sets its equal to its lower or upper bound, in this case, 0 or 1.
Uniform Mutation	Randomly selects one of the variables and sets it equal to a number uniformly distributed between the lower bound and upper bound, in this case, 0 and 1.
Non-uniform Mutation	Randomly selects one of the variables and sets it equal to a random number from a non-uniform distribution. This operator starts as a uniform distribution and then becomes increasingly narrow based on the current generation number to help explore in closer neighborhoods to the current variable.
Multi Non-uniform Mutation	Applies the Non-Uniform mutation operator to all variables.

4.1.5 Termination

The termination condition determines if the GA should stop or continue searching for a solution. Some of the common termination criteria used are upper bound on the number of generations, number of generations since the last improvement and the total amount of time expired, which is often based on the number of evaluations if local search is combined with the GA [78]. In this research, we use maximum number of generations as the termination criterion which is a surrogate for running the algorithm for a certain amount of time.

4.1.6 Fitness Function

Genetic algorithms rely on the simple premise of using natural selection as a means of solution elimination, with the objective function acting as the driving force behind the GA search. The fitness function in this research (shown in Figure 4 - 6) first repairs infeasible chromosomes and then evaluates their fitness. Once the chromosomes have been repaired, the fitness function is used to determine the fitness of individuals in the population. In this research we use the mean absolute deviation (MAD) to determine the fitness of each chromosome.

$$MAD = \frac{1}{s}\sum_{s=1}^{S} |P_s - \bar{P}|$$

where S is the total number of stations,

P_s is the actual production of station s,

$\bar{P}$ is the mean production of all stations.

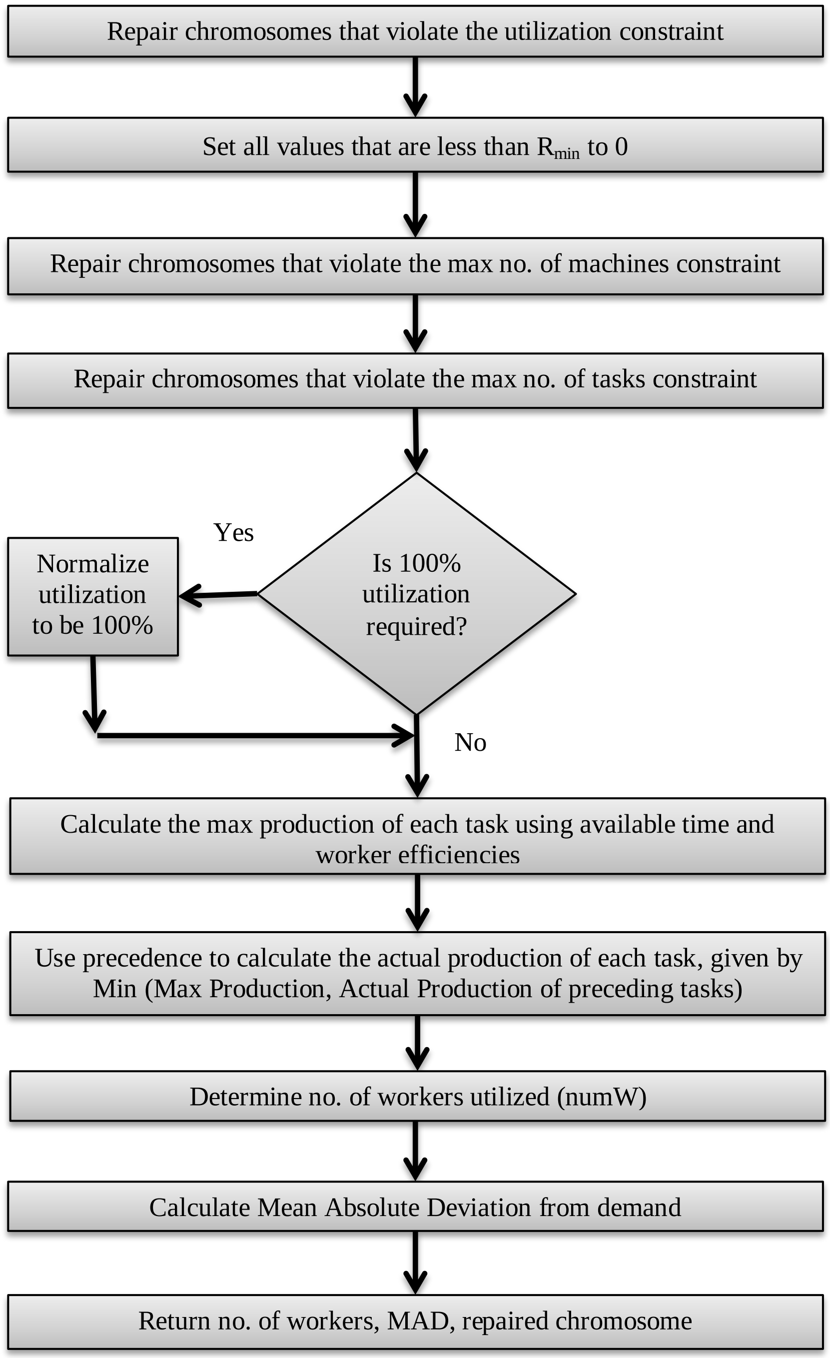

Figure 4 - 6: Fitness Function

4.1.6.1 Constraint Handling and Chromosome Repair

One of the main issues that arises when applying GAs to a particular problem is constraint handling [79]. Michalewicz and Janikow [79] discuss three constraint handling approaches that have been used in the past.

1. **Penalty functions:** The principle behind this approach is to incorporate Lagrangian relaxation and penalize chromosomes that violate constraints by decreasing the "goodness" of the fitness function. $G(x) = F(x) + \lambda C(x)$ where $C(x)$ represents sum of the constraint violations.
2. **Decoders and Repair Algorithms:** Special chromosome mapping can be used to increase the probability of generating feasible solutions. If infeasible solutions are generated, they can be corrected by using a repair algorithm. This method is highly problem specific and might not be suitable for some types of problems.
3. **Specialized data structures and genetic operators:** In some cases, the genetic operators may be able to incorporate the constraints that are present in the problem directly into the encoding of the problem. This approach is also highly problem specific but very effective.

They also propose a more general approach to handle constraints. It involves the elimination of equalities and design of genetic operators such that they are able to keep all the chromosomes in the solution space. Refer to Michaelewicz [75] and Michalewicz and Janikow [79] for details of this method and past approaches. In this research, we use a combination of a repair method and a penalty function that is specific to the problem.

Owing to the nature of our problem, the chromosomes only determine the percentage of time one worker is assigned to a particular task. In some cases, the GA may generate a solution where the chromosomes are infeasible. There are three constraints that have to be handled to ensure that chromosomes are feasible.

1. The utilization constraint:

The GA may generate a chromosome where the utilization of a worker exceeds 100%. In the chromosome shown in Figure 4 - 7 which represents the four task four worker problem of Table 4 - 1 and Figure 4 - 3, worker three has been assigned two tasks with a total utilization of 125%.

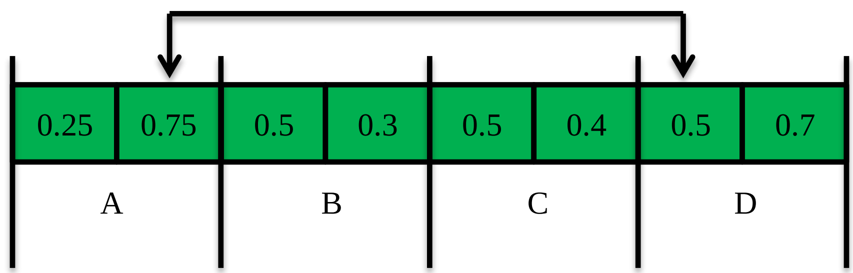

Figure 4 - 7: Chromosome violating the utilization constraint

The genes corresponding to worker three can be repaired by normalizing the assignments to be less than or equal to 100%. In this case the two values are set to 0.75/1.25 (i.e., 0.6) and 0.5/1.25 (i.e., 0.4) thus only utilizing 100%.

Thus the chromosome is repaired as shown in Figure 4 - 8.

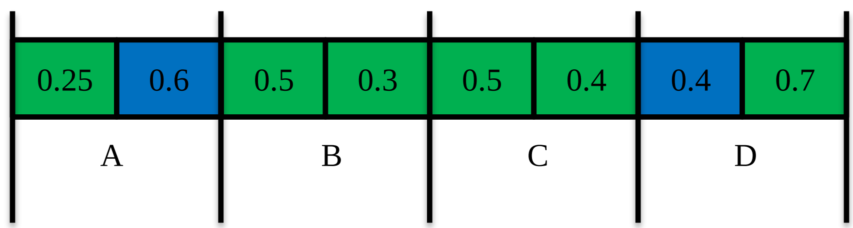

Figure 4 - 8: Utilization Repaired Chromosome

2. The 'maximum number of machines' constraint:

In some instances, the total number of assigned machines may exceed the total number of machines in the system.

Consider an example where the total number of machines (M) in the system is six. In Figure 4 - 9, the total number of assigned machines, $\sum_{s=1}^{S} m_s = 8$. Since we assume that if a worker has been assigned to a task, there is an available machine.

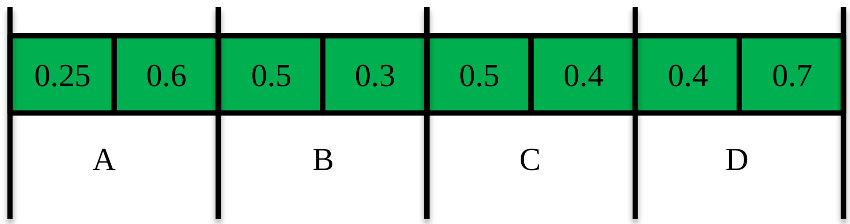

Figure 4 - 9: Chromosome violating the maximum number of machines constraint

The repair mechanism can be described by the flowchart shown in Figure 4 - 10.

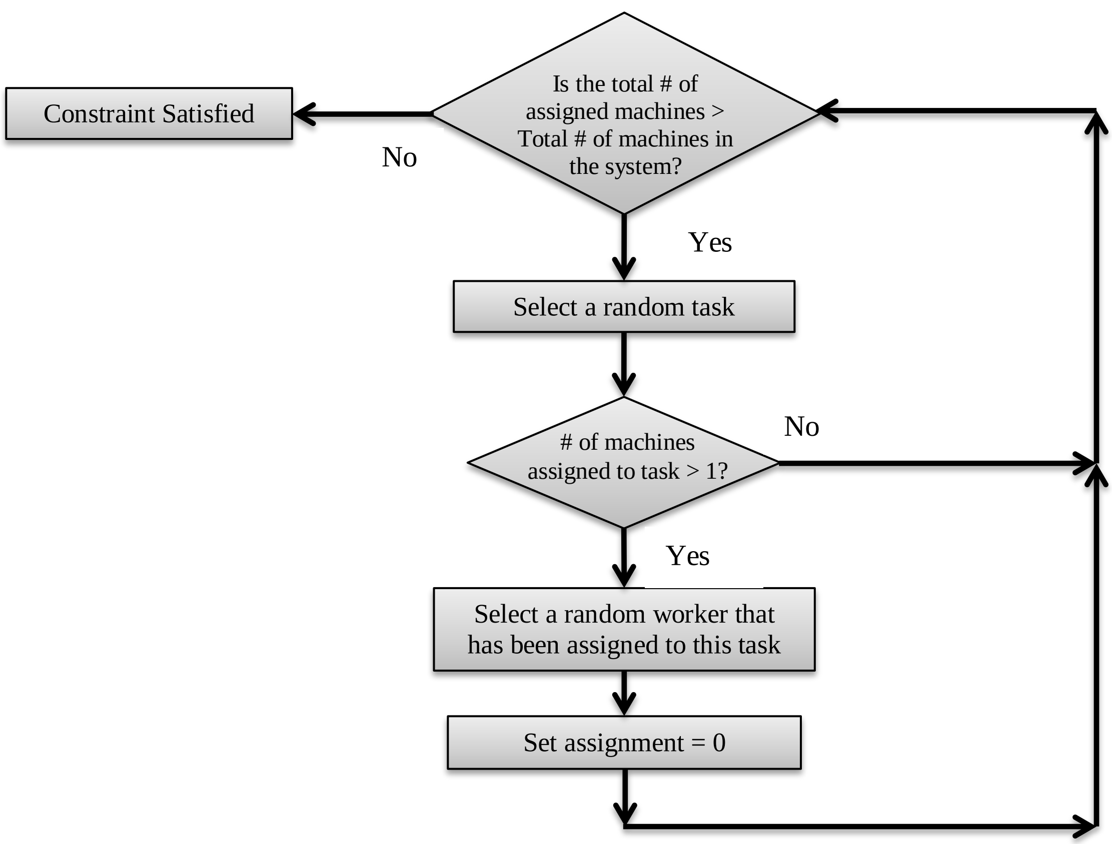

Figure 4 - 10: Repair Procedure for Exceeding the Maximum Number of Machines

The repair procedure only selects tasks that have more than one worker assigned to it to ensure each task has been assigned at least one worker. If the repair mechanism had chosen task A and task C and set the 1^{st} and 6^{th} gene to 0, the resulting chromosome can be seen in Figure 4 - 11.

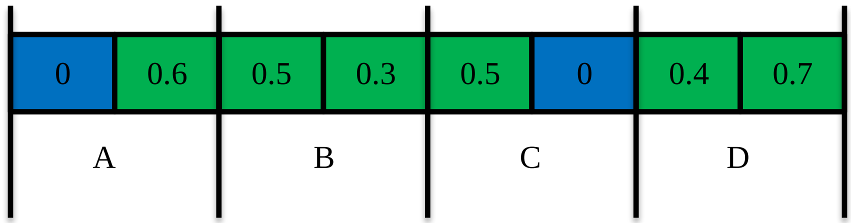

Figure 4 - 11: Maximum Number of Machines Repaired Chromosome

3. The 'maximum number of tasks' constraint:

The GA may generate a chromosome where the number of tasks assigned to a worker may exceed '$N_{MaxTasks}$' which is the maximum number of tasks that can be assigned to a worker, as shown in Figure 4 - 12.

Consider an example where partial assignment is not allowed (i.e., $N_{MaxTasks}$ = 1). The chromosome in the figure below shows that worker three has been assigned to task A and D, which violates this constraint. The chromosome must be repaired so that the constraint is satisfied and all tasks have been assigned.

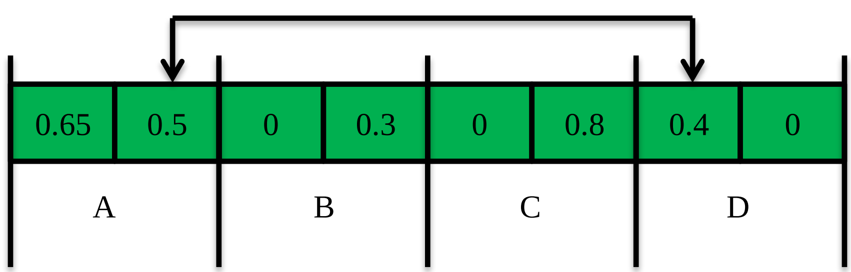

Figure 4 - 12: Chromosome violating the $N_{MaxTasks}$ constraint

The repair mechanism can be described by the flowchart shown in Figure 4 - 13.

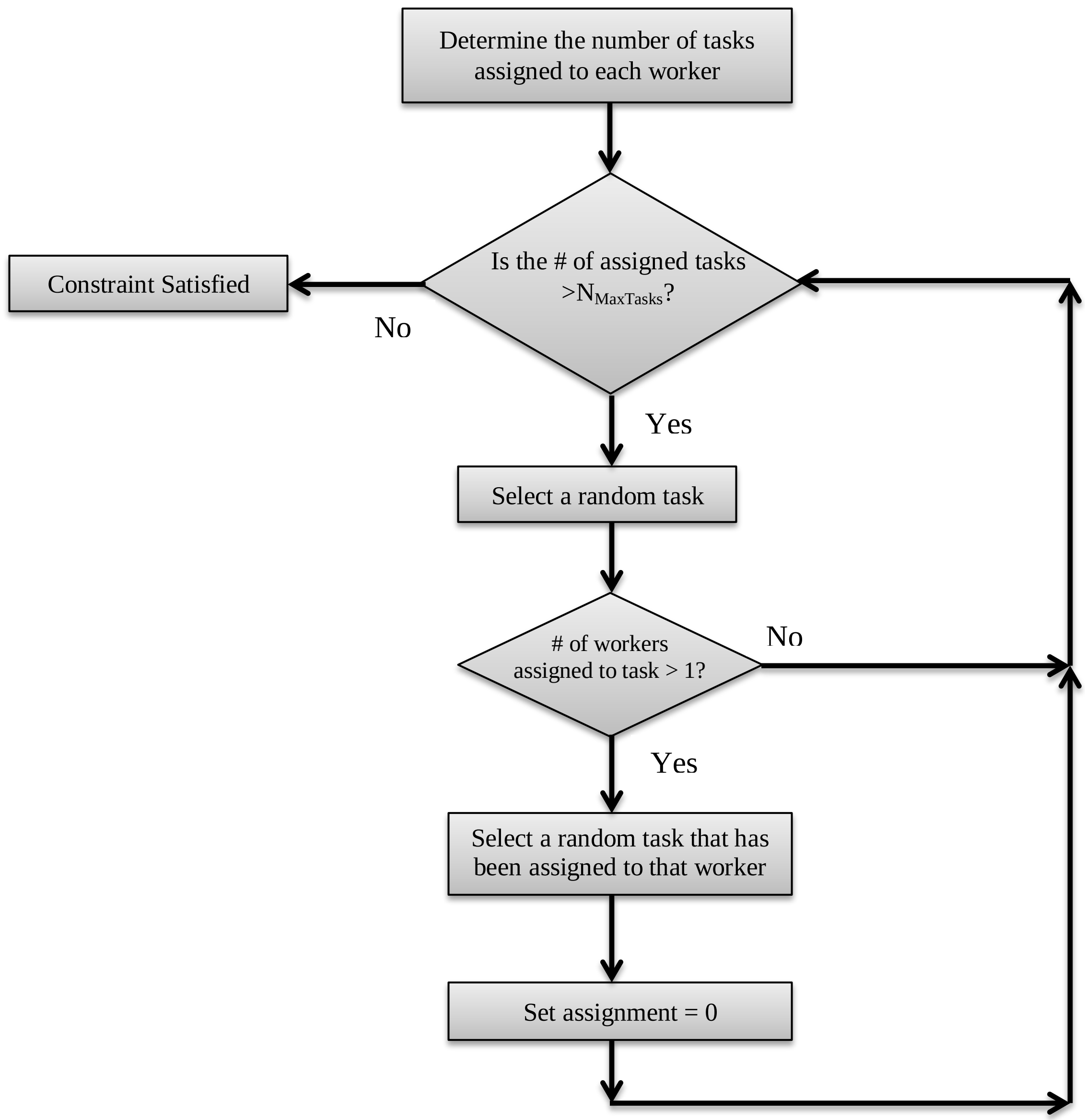

Figure 4 - 13: Repair Procedure for Exceeding the Maximum Number of Tasks

The repaired chromosome would be as shown in Figure 4 - 14.

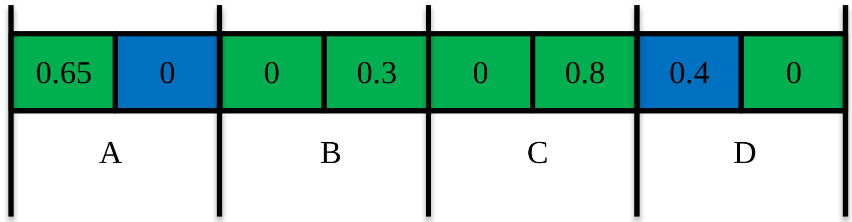

Figure 4 - 14: 'Max Tasks' Repaired Chromosome

Note that it is possible to have a chromosome (solution) where multiple workers are assigned to the same task and a particular task may not have a worker assigned to it. In such a case, the actual production for that particular task would evaluate to zero. This would result in a very high value for MAD and hence a very low fitness value for the chromosome thus automatically penalizing it. Such a chromosome will have a very low probability of being selected for the next generation.

4.1.6.2 Search Strategy with respect to Inheritable Characteristics

There are two types of strategies based on whether the chromosome should reflect the result of the local improvement procedure or in this case the repair procedure.

1. **Search strategy based on Lamarckian Evolution:**

   In the early 1800s, French biologist Jean-Baptiste Lamarck put forward the idea that that an organism can pass on characteristics that it has acquired during its lifetime to its offspring. This idea (known as Lamarckian Evolution) can be applied to GAs in a similar context by updating the chromosome to reflect the result of the local improvement or repair procedure.

2. **Search strategy based on the Baldwin Effect:**

   This strategy is based on a theory of a possible evolutionary process that was published in 1896 by American psychologist James Mark Baldwin. Contradictory to Lamarckian evolution, it stated that selected offspring would be genetically predisposed to learning new skills rather than being genetically coded. In the context of GAs, the chromosome is not re-coded to reflect the result of the local improvement or repair procedure. This strategy was first proposed by Hinton and Nolan [80], using a flat landscape with a single well representing the optimal solution. They showed that the GA was able to find the optimum only when it was hybridized with a random search.

One of the main problems faced when using the Baldwinian strategy is the mapping of multiple genotypes to the same phenotype. In the context of GAs, a genotype refers to the variables that are associated with a chromosome and phenotype refers to the solution that is constructed from a chromosome.

Whitley *et al.* [81] provide a comparison between Baldwinian and Lamarckian learning. They show that both strategies work better than a GA without a local improvement procedure. They concluded that the Baldwinian strategy is slower and has a lesser chance of converging to a local optimum when compared to Lamarckian learning. They also state that the test functions may have been too easy to provide more conclusive evidence as to the effectiveness of the Baldwinian strategy, which should be evaluated in more complex domains.

Houck *et al.* [82] apply both strategies to a wide variety of functions, as well as the location – allocation problem and cell formation problem. One of the key differences is that though Lamarckian learning disrupts the schema processing capability of the GA and may lead to premature convergence, the need to have a one-to-one genotype to phenotype mapping is significantly reduced. Baldwinian learning on the other hand, does not affect the schema processing capability of the GA and results in large number of genotypes mapping to the same phenotype. The authors conclude that neither pure Baldwinian nor pure Lamarckian strategies prove to be consistently effective and hence suggest a partial Lamarckian strategy.

In this research we implement a Lamarckian learning strategy as we need to update the chromosomes to reflect the result of the repair procedure.

4.1.7 GA Parameters

The parameters for the GA are given in Table 4 - 4.

Table 4 - 4: GA Parameters

Parameter	**Value**
No. of boundary mutation operators	0.08
No. of uniform mutation operators	0.04
No. of non-uniform mutation operators	0.06
No. of multi non-uniform mutation operators	0.06
No. of simple crossover operators	0.02
No. of arithmetic crossover operators	0.02
No. of heuristic crossover operators	0.02
Probability of selecting the best individual	0.08
Size of initial population	300
Maximum number of generations	4000
Number of replications	20

4.2 Results

We test the GA on the industrial example data as well as the randomly generated problems that were used for CPLEX, described in Chapter 3. The legend for the results is shown in Table 4 - 5.

Table 4 - 5: Legend for GA results

Color	**Description**
	Optimal Solution
	Best Feasible Solution Found
	Best Infeasible Solution Found

There are two ways we can determine if a solution is optimal - either the solution found by the GA is zero, or the solution found by the GA is equal to a CPLEX optimal solution. If the mean of a solution is marked green, it indicates that the GA was able to find an optimal solution on all 20 replications.

Solutions marked in yellow indicate the best feasible solution that was found by the GA after 4000 generations. In some cases, these solutions may be optimal, but we cannot verify this as the GA is a heuristic algorithm.

Solutions marked in red are infeasible as they are unable to meet the demand for the given time period. One of the advantages of using a GA is that it is always able to generate a solution. In this case, an infeasible solution would give an idea of the maximum output of the system for the given constraints.

4.2.1 Industrial Example

The results from the industrial model are shown in Table 4 - 6. It can be seen that the GA is able to solve the problem efficiently. We know that no feasible solution exists for the solutions marked in red from the CPLEX results in Section 3.5.1. This implies that the GA was able to find an optimal solution for all cases of this problem. We will compare its performance with CPLEX in the next section.

Table 4 - 6: GA Results for the Industrial Model

Utiliz-ation	Assign-ment	Demand	MAD			Actual Production		Computation Time (s)	
			Best	Mean	StdDev	Mean	StdDev	Mean	StdDev
100%	1 Task	9500	217.62	217.62	0.00	10909.09	0.00	276.98	35.11
100%	1 Task	11000	217.62	217.62	0.00	10909.09	0.00	411.38	40.30
100%	2 Tasks	9500	0.00	0.00	0.00	11379.31	0.00	0.87	0.18
100%	2 Tasks	11000	0.00	0.00	0.00	11379.31	0.00	0.84	0.22
<100%	1 Task	9500	0.00	0.00	0.00	9581.05	95.08	3.96	1.37
<100%	1 Task	11000	41.64	42.22	0.42	10909.09	0.00	230.65	38.12
<100%	2 Tasks	9500	0.00	0.00	0.00	9745.02	254.25	4.31	1.17
<100%	2 Tasks	11000	0.00	0.00	0.00	11072.29	90.98	19.35	36.87

4.2.2 Computational Study

As mentioned in Section 3.5.2, the precedence graphs, standard processing times, worker skills and movement times for this computational study are given in Appendix A. 'Set0' is the base problem set for a given problem. The standard processing times of the other sets were randomly generated using a uniform distribution with the lower and upper bound specified by the smallest and largest processing times of Set0 respectively. Five different sets (Set0 through Set4) of processing times were utilized for each problem to make sure that the efficacy of the GA is independent of processing times.

Figure 4 - 15 shows the convergence of the GA for one iteration of Set4 of the linear 12 task problem for the case with <100% utilization, 2 task assignment and 220 demand. It can be seen that the GA converges fairly quickly. In most cases, the GA converges within 4000 generations.

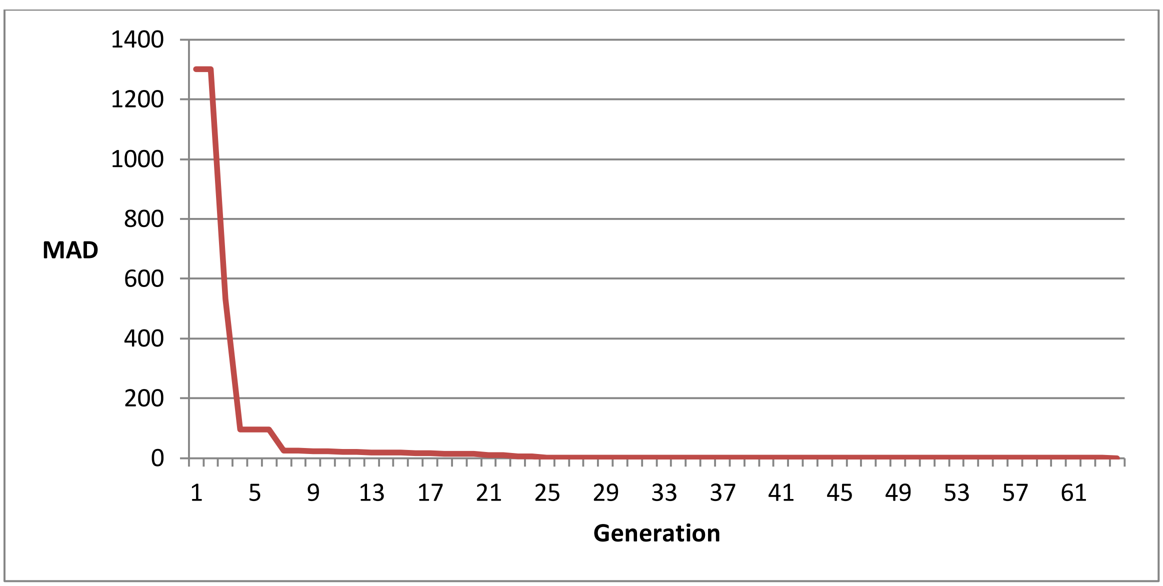

Figure 4 - 15: GA Convergence

The GA results for the Linear and Branched Models are shown in Table 4 - 7 and Table 4 - 8 respectively. The results show that the GA is very effective at solving the linear model. It is able to find an optimal solution in most cases of the 24 task instance of the problem. Recall that these could not be solved by CPLEX. In the case of larger instances of the branched model, the GA is able to find optimal solutions in cases where the utilization is constrained to be less than 100%, and good solutions in cases where the utilization is constrained to be equal to 100%. In the next section we provide a comparison between the GA and CPLEX results.

Table 4 - 7: Summary of GA results for the Linear Model

# of Tasks	Utiliz-ation	Assig-nment	Dem-and	Set0					Set1				
				MAD			Actual Production		MAD			Actual Production	
				Best	Mean	StdDev	Mean	StdDev	Best	Mean	StdDev	Mean	StdDev
6	100%	1 Task	180	2.88	3.74	1.35	245.09	9.48	45.54	49.37	7.92	211.88	15.98
	100%	1 Task	220	2.88	5.90	4.30	247.25	10.03	45.54	46.67	5.05	217.97	7.47
	100%	2 Tasks	180	0.00	0.00	0.00	218.88	27.40	0.00	0.00	0.00	206.39	24.02
	100%	2 Tasks	220	0.00	0.00	0.00	245.90	15.84	0.00	0.00	0.00	239.39	8.39
	<100%	1 Task	180	0.00	0.00	0.00	189.93	5.84	0.00	0.02	0.10	181.72	2.75
	<100%	1 Task	220	0.00	0.68	1.66	219.70	7.52	0.23	10.93	9.24	196.87	19.54
	<100%	2 Tasks	180	0.00	0.00	0.00	205.81	18.97	0.00	0.00	0.00	197.84	15.59
	<100%	2 Tasks	220	0.00	0.00	0.00	236.62	11.67	0.00	0.00	0.00	227.24	6.58
12	100%	1 Task	180	2.75	2.75	0.00	154.80	0.00	13.77	14.73	2.35	144.88	2.49
	100%	1 Task	220	20.13	23.22	3.86	153.00	5.54	17.66	18.97	3.20	144.88	2.49
	100%	2 Tasks	180	0.00	0.00	0.00	183.29	2.52	0.00	0.00	0.00	185.98	4.28
	100%	2 Tasks	220	7.29	7.81	0.65	191.87	2.33	3.05	6.23	1.00	197.05	2.43
	<100%	1 Task	180	3.85	8.69	3.99	137.24	7.02	9.47	11.82	2.95	143.18	3.41
	<100%	1 Task	220	11.49	18.24	6.09	137.70	9.09	16.11	24.13	5.86	142.18	4.05
	<100%	2 Tasks	180	0.00	0.00	0.00	181.01	1.63	0.00	0.00	0.00	181.34	1.44
	<100%	2 Tasks	220	7.58	8.90	0.73	187.77	2.40	6.41	6.75	0.40	195.83	1.06
24	100%	1 Task	180	12.96	16.01	2.96	187.58	10.09	14.73	24.63	7.10	203.31	16.68
	100%	1 Task	220	11.84	20.75	7.87	197.59	6.85	12.91	26.27	5.26	218.91	17.05
	100%	2 Tasks	180	0.00	0.00	0.00	198.54	15.30	0.00	0.00	0.00	192.95	10.01
	100%	2 Tasks	220	0.00	0.00	0.00	234.18	13.29	0.00	0.00	0.00	231.63	9.22
	<100%	1 Task	180	0.00	0.44	1.36	179.51	2.62	0.00	0.00	0.00	180.25	0.39
	<100%	1 Task	220	3.63	10.09	4.21	193.08	9.48	0.00	4.09	4.66	207.67	11.54
	<100%	2 Tasks	180	0.00	0.00	0.00	186.23	9.14	0.00	0.14	0.62	187.72	7.73
	<100%	2 Tasks	220	0.00	0.00	0.00	222.60	3.72	0.00	0.00	0.00	220.58	1.06

Table 4 – 7 Continued

# of Tasks	Utiliz-ation	Assig-nment	Dem-and	Set2					Set3				
				MAD			Actual Production		MAD			Actual Production	
				Best	Mean	StdDev	Mean	StdDev	Best	Mean	StdDev	Mean	StdDev
6	100%	1 Task	180	0.00	0.00	0.00	239.47	6.12	0.00	0.00	0.00	177.30	0.00
	100%	1 Task	220	0.00	0.00	0.00	238.22	5.56	14.09	14.09	0.00	288.86	0.00
	100%	2 Tasks	180	0.00	0.00	0.00	210.90	19.64	0.00	0.00	0.00	231.68	30.70
	100%	2 Tasks	220	0.00	0.00	0.00	241.38	13.17	0.00	0.00	0.00	255.67	24.08
	<100%	1 Task	180	0.00	0.00	0.00	190.03	12.33	0.00	0.00	0.00	189.05	8.70
	<100%	1 Task	220	0.00	0.00	0.00	228.13	9.50	0.00	0.00	0.00	225.14	7.47
	<100%	2 Tasks	180	0.00	0.00	0.00	209.84	22.97	0.00	0.00	0.00	209.54	25.98
	<100%	2 Tasks	220	0.00	0.00	0.00	234.50	10.20	0.00	0.00	0.00	242.50	15.30
12	100%	1 Task	180	10.65	11.48	1.17	184.43	3.78	4.52	4.61	0.05	156.37	5.29
	100%	1 Task	220	11.31	11.59	0.88	184.43	3.78	12.52	16.58	3.74	153.79	6.48
	100%	2 Tasks	180	0.00	0.00	0.00	189.14	7.44	0.00	0.00	0.00	188.06	4.97
	100%	2 Tasks	220	0.00	0.19	0.50	219.69	2.07	2.31	3.04	0.70	208.99	2.47
	<100%	1 Task	180	0.00	2.59	2.15	172.65	6.72	5.19	5.65	0.31	155.08	6.06
	<100%	1 Task	220	9.44	15.71	3.07	171.13	5.88	11.14	16.18	2.76	155.72	5.73
	<100%	2 Tasks	180	0.00	0.00	0.00	183.21	3.43	0.00	0.00	0.00	185.87	5.44
	<100%	2 Tasks	220	0.00	0.20	0.41	219.40	1.56	2.32	3.77	0.60	206.43	2.17
24	100%	1 Task	180	0.00	0.81	2.52	186.76	14.83	0.00	0.71	3.16	203.82	13.71
	100%	1 Task	220	5.04	11.21	3.63	215.80	9.52	0.00	7.96	7.68	222.74	9.47
	100%	2 Tasks	180	0.00	0.00	0.00	190.33	9.89	0.00	0.00	0.00	191.48	11.98
	100%	2 Tasks	220	0.00	0.00	0.00	226.76	7.61	0.00	0.00	0.00	225.69	4.62
	<100%	1 Task	180	0.00	0.04	0.17	180.43	0.93	0.00	0.57	1.40	177.65	6.41
	<100%	1 Task	220	0.00	7.20	8.83	205.32	17.89	0.00	1.53	2.18	210.79	8.50
	<100%	2 Tasks	180	0.00	0.00	0.00	181.62	1.28	0.00	0.00	0.00	182.04	2.85
	<100%	2 Tasks	220	0.00	0.00	0.00	220.78	1.26	0.00	0.00	0.00	220.60	0.81

Table 4 – 7 Continued

# of Tasks	Utilization	Assignment	Demand	Set4				
				MAD			Actual Production	
				Best	Mean	StdDev	Mean	StdDev
6	100%	1 Task	180	10.58	12.15	3.86	280.29	7.71
	100%	1 Task	220	10.58	12.59	4.16	280.61	6.92
	100%	2 Tasks	180	0.00	0.00	0.00	213.51	23.18
	100%	2 Tasks	220	0.00	0.00	0.00	243.69	15.18
	<100%	1 Task	180	0.00	0.00	0.00	187.46	10.00
	<100%	1 Task	220	0.00	0.04	0.14	221.68	3.99
	<100%	2 Tasks	180	0.00	0.00	0.00	204.45	17.83
	<100%	2 Tasks	220	0.00	0.00	0.00	238.78	16.36
12	100%	1 Task	180	0.00	0.00	0.00	187.10	1.79
	100%	1 Task	220	0.00	0.00	0.00	192.35	0.00
	100%	2 Tasks	180	0.00	0.00	0.00	193.96	8.99
	100%	2 Tasks	220	0.00	0.00	0.00	223.24	4.02
	<100%	1 Task	180	0.00	0.61	2.06	178.30	14.50
	<100%	1 Task	220	0.00	1.19	3.00	184.96	10.27
	<100%	2 Tasks	180	0.00	0.00	0.00	184.56	5.08
	<100%	2 Tasks	220	0.00	0.00	0.00	221.75	2.03
24	100%	1 Task	180	1.67	9.44	5.44	255.76	24.34
	100%	1 Task	220	3.21	7.59	3.52	262.17	12.79
	100%	2 Tasks	180	0.00	0.00	0.00	192.71	11.09
	100%	2 Tasks	220	0.00	0.00	0.00	231.09	10.75
	<100%	1 Task	180	0.00	0.23	1.03	180.75	5.03
	<100%	1 Task	220	0.00	2.59	5.71	210.92	19.15
	<100%	2 Tasks	180	0.00	0.00	0.00	189.49	9.77
	<100%	2 Tasks	220	0.00	0.00	0.00	223.49	3.51

Table 4 - 8: Summary of GA results for the Branched Model

# of Tasks	Utiliz-ation	Assig-nment	Dem-and	Set0					Set1				
				MAD			Actual Production		MAD			Actual Production	
				Best	Mean	StdDev	Mean	StdDev	Best	Mean	StdDev	Mean	StdDev
6	100%	1 Task	180	17.07	23.98	5.79	206.40	0.00	0.92	0.92	0.00	209.66	0.00
	100%	1 Task	220	35.79	37.59	3.00	253.58	8.84	75.11	75.11	0.00	221.65	0.00
	100%	2 Tasks	180	0.00	0.00	0.00	254.65	30.17	0.00	0.11	0.23	230.77	31.65
	100%	2 Tasks	220	0.00	0.00	0.00	259.91	23.07	0.00	0.00	0.00	260.17	14.78
	<100%	1 Task	180	0.00	0.00	0.00	191.48	10.27	0.00	0.00	0.00	189.93	10.21
	<100%	1 Task	220	0.00	0.99	2.54	224.43	13.48	0.00	1.92	2.75	215.97	6.43
	<100%	2 Tasks	180	0.00	0.00	0.00	203.31	19.67	0.00	0.00	0.00	212.66	22.58
	<100%	2 Tasks	220	0.00	0.00	0.00	238.77	14.32	0.00	0.00	0.00	234.46	8.98
12	100%	1 Task	180	15.70	16.49	3.55	154.80	0.00	38.95	39.67	1.38	201.75	0.00
	100%	1 Task	220	27.55	29.65	1.91	154.80	0.00	42.81	45.31	2.37	201.75	0.00
	100%	2 Tasks	180	11.75	14.60	3.83	192.88	1.79	7.99	16.87	10.08	255.37	24.23
	100%	2 Tasks	220	13.37	15.44	1.56	191.49	2.68	7.98	16.68	10.09	265.22	6.31
	<100%	1 Task	180	3.85	7.53	3.26	139.91	8.98	0.00	3.10	7.54	177.21	19.68
	<100%	1 Task	220	12.14	19.16	5.16	140.04	10.18	7.36	14.09	12.46	188.13	27.94
	<100%	2 Tasks	180	0.00	0.03	0.05	183.16	3.49	0.00	0.01	0.03	220.11	20.38
	<100%	2 Tasks	220	8.25	9.01	0.79	187.31	2.73	0.00	0.00	0.00	230.31	7.96
24	100%	1 Task	180	31.55	37.98	4.18	192.15	5.24	27.73	34.33	4.06	188.67	10.56
	100%	1 Task	220	31.67	41.98	5.39	194.62	7.16	29.27	36.63	4.55	192.30	17.13
	100%	2 Tasks	180	0.84	4.15	3.23	278.19	15.78	0.00	1.14	1.36	260.66	7.79
	100%	2 Tasks	220	0.45	6.59	5.50	278.82	15.82	0.00	1.69	2.14	257.00	17.47
	<100%	1 Task	180	0.00	2.18	3.49	174.75	8.79	0.00	8.42	7.55	159.49	18.39
	<100%	1 Task	220	4.71	12.65	5.00	186.80	12.68	3.76	18.41	10.28	174.98	25.43
	<100%	2 Tasks	180	0.00	1.41	1.29	184.82	7.97	0.00	0.07	0.18	180.98	2.13
	<100%	2 Tasks	220	0.00	0.21	0.41	220.14	0.51	0.00	0.01	0.03	219.99	0.25

Table 4 – 8 Continued

# of Tasks	Utiliz-ation	Assig-nment	Dem-and	Set2					Set3				
				MAD			Actual Production		MAD			Actual Production	
				Best	Mean	StdDev	Mean	StdDev	Best	Mean	StdDev	Mean	StdDev
6	100%	1 Task	180	43.37	49.04	5.83	266.96	17.70	2.39	33.03	28.44	179.09	7.53
	100%	1 Task	220	43.37	49.13	5.94	265.22	17.43	2.39	33.03	28.44	179.09	7.53
	100%	2 Tasks	180	0.00	0.00	0.00	274.04	26.47	0.00	0.00	0.00	216.40	17.51
	100%	2 Tasks	220	0.00	0.00	0.00	265.82	18.97	0.00	0.00	0.00	244.25	13.88
	<100%	1 Task	180	0.00	0.03	0.11	191.33	14.68	0.00	4.40	3.29	171.63	9.21
	<100%	1 Task	220	0.00	0.00	0.00	228.42	11.48	21.79	25.64	4.53	162.32	10.19
	<100%	2 Tasks	180	0.00	0.00	0.00	205.41	24.57	0.00	0.00	0.00	194.76	10.67
	<100%	2 Tasks	220	0.00	0.00	0.00	242.21	19.44	0.00	0.00	0.00	226.56	7.67
12	100%	1 Task	180	32.20	32.69	1.19	203.79	3.31	50.21	54.79	5.61	155.76	0.00
	100%	1 Task	220	32.20	33.41	1.00	204.44	2.38	49.83	51.87	3.86	155.76	0.00
	100%	2 Tasks	180	1.26	6.64	7.09	203.75	6.83	7.85	15.89	7.32	204.21	3.56
	100%	2 Tasks	220	17.75	27.24	6.80	217.10	2.68	8.46	15.06	6.95	205.32	0.77
	<100%	1 Task	180	0.00	0.87	3.88	178.90	9.24	9.09	9.98	1.24	153.39	3.31
	<100%	1 Task	220	6.61	11.10	8.18	194.61	18.98	23.05	24.84	1.49	152.71	3.46
	<100%	2 Tasks	180	0.00	0.00	0.00	192.89	7.50	0.00	0.00	0.00	190.87	6.20
	<100%	2 Tasks	220	0.62	5.58	2.41	208.47	4.97	6.65	9.09	1.04	199.58	2.55
24	100%	1 Task	180	16.26	27.65	5.88	227.17	27.62	36.26	42.33	3.99	207.79	21.61
	100%	1 Task	220	15.49	28.67	7.96	241.14	14.64	32.03	44.65	6.89	233.28	13.56
	100%	2 Tasks	180	1.31	7.05	4.81	310.37	10.11	0.57	6.06	5.49	285.56	33.05
	100%	2 Tasks	220	1.35	7.30	4.48	303.76	22.22	0.65	5.00	2.44	300.39	19.70
	<100%	1 Task	180	0.00	0.03	0.06	180.49	1.51	0.00	0.15	0.54	179.78	1.55
	<100%	1 Task	220	0.00	2.33	3.83	214.16	9.54	0.00	4.28	6.60	209.38	16.08
	<100%	2 Tasks	180	0.00	0.26	0.55	187.95	10.73	0.00	0.80	1.41	185.95	9.37
	<100%	2 Tasks	220	0.00	0.20	0.72	221.17	4.56	0.00	0.55	0.99	221.54	3.63

Table 4 – 8 Continued

Number of Tasks	Utilization	Assignment	Demand	Set4				
				MAD			Actual Production	
				Best	Mean	StdDev	Mean	StdDev
6	100%	1 Task	180	20.21	22.13	0.66	234.51	0.00
	100%	1 Task	220	20.21	22.24	0.48	234.51	0.00
	100%	2 Tasks	180	0.00	0.00	0.00	249.25	23.85
	100%	2 Tasks	220	0.00	0.00	0.00	255.98	15.07
	<100%	1 Task	180	0.00	0.00	0.00	188.64	8.69
	<100%	1 Task	220	0.00	1.09	1.71	218.76	8.73
	<100%	2 Tasks	180	0.00	0.00	0.00	208.41	19.52
	<100%	2 Tasks	220	0.00	0.00	0.00	239.18	14.96
12	100%	1 Task	180	33.11	36.66	4.91	237.80	17.88
	100%	1 Task	220	33.11	35.83	3.68	242.56	8.75
	100%	2 Tasks	180	6.82	12.30	3.30	263.92	5.30
	100%	2 Tasks	220	7.46	12.90	3.35	266.09	0.00
	<100%	1 Task	180	0.00	0.14	0.61	187.93	10.59
	<100%	1 Task	220	0.00	1.68	4.10	217.95	10.30
	<100%	2 Tasks	180	0.00	0.00	0.00	219.79	17.27
	<100%	2 Tasks	220	0.00	0.00	0.00	239.32	12.67
24	100%	1 Task	180	26.18	34.38	5.97	205.89	27.33
	100%	1 Task	220	28.44	38.54	6.96	220.39	19.34
	100%	2 Tasks	180	1.56	8.69	5.16	292.85	7.46
	100%	2 Tasks	220	0.33	8.18	4.38	288.85	20.18
	<100%	1 Task	180	0.00	0.44	1.42	179.37	3.69
	<100%	1 Task	220	0.00	8.69	7.83	197.93	19.94
	<100%	2 Tasks	180	0.00	0.13	0.30	191.63	11.98
	<100%	2 Tasks	220	0.00	0.11	0.24	220.61	1.52

4.2.3 Effect of Reducing the Number of Machines

Since we assume that a machine is available for each worker to task assignment, we can control the number of tasks assigned to each worker by controlling the number of machines that exist in the system. If we consider cases where workers are allowed to be assigned to two tasks, a gradual reduction in the number of machines would force some workers to be assigned to just one task.

If we consider the MAD as a primary objective and the number of workers as a secondary objective, we can use the GA to perform multi-objective optimization by gradually reducing the number of machines which will in turn reduce the number of workers. We can reduce the number of machines till we obtain a Pareto optimal solution for the MAD. This can be achieved fairly quickly due to small of amount of computation time required to run the GA. Note that we cannot directly reduce the number of workers as each worker is trained to perform different tasks. Eliminating random workers may result in a scenario where a given task cannot be performed due to the absence of workers trained to perform that task.

We use Set4 of the linear 12 task example with to demonstrate how the GA can be used for multi-objective optimization. From the solutions in Table 4 - 9 we can see that as we decrease the number of machines, the solutions with higher demand become infeasible Again, green indicates an optimal solution and red indicates an infeasible solution as the system is unable to meet the demand for the given time period.

Table 4 - 10 and Figure 4 - 16 shows the effect of reducing the number of machines on the average number of workers. Cases 2 and 6 are infeasible even with 30 machines and 18 workers. Cases 1, 3, 5 and 7 are able to meet the demand with a MAD of zero by utilizing just 16 machines and 16 workers or less. For Cases 4 and 8, the number of machines does not become a binding constraint till it is reduced from 30 to 18, after which solutions become infeasible.

Table 4 - 9: Solutions for Multi-Objective Optimization

Utilization	Assignment	Demand	30 Machines					27 Machines				
			MAD			Actual Production		MAD			Actual Production	
			Best	Mean	StdDev	Mean	StdDev	Best	Mean	StdDev	Mean	StdDev
100%	1 Task	180	0.00	0.00	0.00	187.10	1.79	0.00	0.00	0.00	189.44	2.99
100%	1 Task	220	0.00	0.00	0.00	192.35	0.00	0.00	0.00	0.00	192.35	0.00
100%	2 Tasks	180	0.00	0.00	0.00	193.96	8.99	0.00	0.00	0.00	190.88	8.52
100%	2 Tasks	220	0.00	0.00	0.00	223.24	4.02	0.00	0.00	0.00	224.14	4.82
<100%	1 Task	180	0.00	0.61	2.06	178.30	14.50	0.00	0.00	0.00	182.60	3.58
<100%	1 Task	220	0.00	1.19	3.00	184.96	10.27	0.00	1.10	2.35	187.16	8.06
<100%	2 Tasks	180	0.00	0.00	0.00	184.56	5.08	0.00	0.00	0.00	182.79	2.38
<100%	2 Tasks	220	0.00	0.00	0.00	221.75	2.03	0.00	0.00	0.00	220.86	2.17

Utilization	Assignment	Demand	24 Machines					21 Machines				
			MAD			Actual Production		MAD			Actual Production	
			Best	Mean	StdDev	Mean	StdDev	Best	Mean	StdDev	Mean	StdDev
100%	1 Task	180	0.00	0.00	0.00	188.56	2.85	0.00	0.00	0.00	188.56	2.85
100%	1 Task	220	0.00	0.00	0.00	192.35	0.00	0.00	0.00	0.00	192.06	1.30
100%	2 Tasks	180	0.00	0.00	0.00	190.92	10.93	0.00	0.00	0.00	187.15	3.93
100%	2 Tasks	220	0.00	0.00	0.00	227.08	6.22	0.00	0.00	0.00	225.32	3.93
<100%	1 Task	180	0.00	0.00	0.00	182.93	3.36	0.00	0.00	0.00	182.53	2.37
<100%	1 Task	220	0.00	0.32	0.89	188.57	3.23	0.00	0.72	1.28	189.26	3.56
<100%	2 Tasks	180	0.00	0.00	0.00	181.52	1.82	0.00	0.00	0.00	181.71	3.16
<100%	2 Tasks	220	0.00	0.00	0.00	220.94	1.55	0.00	0.00	0.00	220.37	1.40

Table 4 – 9 continued

Utilization	Assignment	Demand	18 Machines					17 Machines				
			MAD			Actual Production		MAD			Actual Production	
			Best	Mean	StdDev	Mean	StdDev	Best	Mean	StdDev	Mean	StdDev
100%	1 Task	180	0.00	0.00	0.00	188.85	2.93	0.00	0.00	0.00	188.27	2.74
100%	1 Task	220	0.00	0.00	0.00	192.06	1.30	0.00	0.14	0.65	191.67	2.10
100%	2 Tasks	180	0.00	0.00	0.00	188.54	4.31	0.00	0.00	0.00	189.44	2.99
100%	2 Tasks	220	0.00	0.40	1.26	223.10	8.98	3.27	5.87	3.36	201.18	7.12
<100%	1 Task	180	0.00	1.16	2.30	174.67	16.52	0.00	0.76	2.11	177.48	15.06
<100%	1 Task	220	0.00	0.84	2.27	187.24	7.96	0.00	1.78	3.44	184.76	12.08
<100%	2 Tasks	180	0.00	0.00	0.00	180.73	1.16	0.00	0.00	0.00	181.20	2.73
<100%	2 Tasks	220	0.00	1.43	2.77	214.19	10.69	3.27	8.72	4.37	193.96	9.55

Table 4 – 9 continued

Utilization	Assignment	Demand	16 Machines				
			MAD			Actual Production	
			Best	Mean	StdDev	Mean	StdDev
100%	1 Task	180	0.00	0.00	0.00	189.14	2.98
100%	1 Task	220	0.00	0.14	0.65	191.09	2.61
100%	2 Tasks	180	0.00	0.23	0.47	187.42	3.03
100%	2 Tasks	220	0.00	0.54	1.04	188.88	3.63
<100%	1 Task	180	0.00	0.61	2.06	178.39	14.68
<100%	1 Task	220	0.00	2.96	4.50	179.74	17.56
<100%	2 Tasks	180	0.00	1.07	3.40	177.53	10.38
<100%	2 Tasks	220	0.00	9.85	14.67	175.48	19.61

Table 4 - 10: Effect on the Average no. of Workers

Case	Utiliz-ation	Assign-ment	Dem-and	Average no. of Workers						
				30 Machines	27 Machines	24 Machines	21 Machines	18 Machines	17 Machines	16 Machines
1	100%	1 Task	180	17.90	17.80	17.95	17.65	17.55	16.70	16.00
2	100%	1 Task	220	18.00	17.95	17.95	17.95	17.85	16.90	16.00
3	100%	2 Tasks	180	17.85	17.90	17.70	17.30	16.70	16.30	15.70
4	100%	2 Tasks	220	18.00	17.90	17.85	17.65	16.80	16.00	16.00
5	<100%	1 Task	180	17.60	17.75	17.85	17.60	17.35	17.00	15.95
6	<100%	1 Task	220	17.40	17.55	17.70	17.45	17.35	16.90	16.00
7	<100%	2 Tasks	180	17.75	17.40	17.10	16.40	16.10	15.80	15.50
8	<100%	2 Tasks	220	17.75	17.30	17.20	17.05	16.65	16.00	15.65

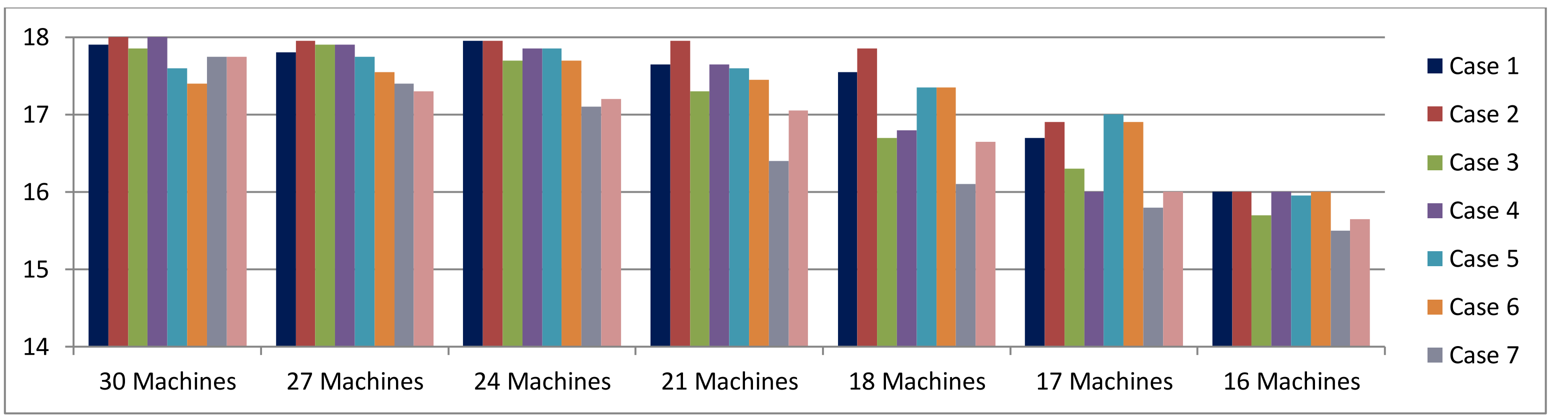

Figure 4 - 16: Effect on the Average no. of Workers

4.3 Comparison between CPLEX and GA results

The legend for the comparison is shown in Table 4 - 11. We compare the solutions as well as the computation times for all instances of the linear and branched models. The computation times are also color coded to show the type of solution that was obtained.

Table 4 - 11: Legend for the comparison between CPLEX and GA results

Color	Description
	Optimal Solution
	Best Feasible Solution Found by the GA and CPLEX
	Better Feasible Solution Found either by the GA or CPLEX
NFSE	No Feasible Solution Exists
	Best Infeasible Solution Found by GA
NFSF	No Feasible Solution Found by CPLEX

4.3.1 Industrial Example

The comparisons between CPLEX and GA solutions and computation times are shown in Table 4 - 12 and Table 4 - 13 respectively. It can be seen that both methods do equally well at finding an optimal solution. In terms of computation time, the GA is able to find solutions much faster for all cases except the one with 100% utilization and one task assignment.

Table 4 - 12: Comparison between CPLEX and GA solutions for the Industrial Model

Utilization	Assignment	Demand	CPLEX Solution	GA Solution		
				Best	Mean	StdDev
100%	1 Task	9500	217.52	217.52	217.52	0.00
100%	1 Task	1100	NFSE	217.52	217.52	0.00
100%	2 Tasks	9500	0	0.00	0.00	0.00
100%	2 Tasks	1100	0	0.00	0.00	0.00
<100%	1 Task	9500	0	0.00	0.00	0.00
<100%	1 Task	1100	NFSE	41.64	42.22	0.42
<100%	2 Tasks	9500	0	0.00	0.00	0.00
<100%	2 Tasks	1100	0	0.00	0.00	0.00

Table 4 - 13: Comparison between CPLEX and GA computation times for the Industrial Model

Utilization	Assignment	Demand	CPLEX Computation Time (s)	GA Computation Time (s)	
				Mean	StdDev
100%	1 Task	180	15.5	276.98	35.11
100%	1 Task	220	0.01	411.38	40.30
100%	2 Tasks	180	653.3	0.87	0.18
100%	2 Tasks	220	54.21	0.84	0.22
<100%	1 Task	180	7.59	3.96	1.37
<100%	1 Task	220	0.02	230.65	38.12
<100%	2 Tasks	180	18.62	4.31	1.17
<100%	2 Tasks	220	22.1	19.35	36.87

4.3.2 Computational Study

The results in Table 4 - 14 shows that the GA performs better than CPLEX when applied to larger instances of the linear model. For the six task problem, the GA is able to solve all cases easily except for the cases with 100% utilization and one Task assignment. For the 12 task problem it does as well as CPLEX and is able to find optimal solutions for all cases where a feasible solution exists. For the 24 task problem, it is able to find an optimal solution in most cases and a good solution in others. CPLEX on the other hand was unable to find any feasible solution for the 24 task problem.

The comparison between computation times for the linear model is shown in Table 4 - 15. It can be seen that CPLEX is able to determine infeasibility fairly quickly for the six and 12 task problems. A possible strategy would be to use CPLEX till a feasible solution is found, after which the GA could be used. The results also show that the GA takes longer than CPLEX in cases where the optimal solution isn't zero. This is because the GA is a heuristic and cannot determine if non-zero solutions are optimal and will only terminate only after 4000 generations.

Table 4 - 16 shows a comparison between the GA and CPLEX solutions for the branched model. For the six task problem, the GA performs as well as CPLEX and is able to solve all

cases optimally. For the 12 task problem, CPLEX and the GA are able to find optimal solutions for all cases except those with 100% utilization and one task assignment. For the 24 task problem, the GA performs better than CPLEX and is able to find optimal solutions in all cases where the utilization is less than 100%. For cases where utilization is constrained to be equal to 100%, the GA is able to find better solutions than CPLEX.

The comparison between computation times for the branched model is shown in Table 4 - 17. Again, it can be seen that CPLEX is able to determine infeasibility fairly quickly for the six and 12 task problems. The results also show that the computation times for the GA are greater for the branched model as compared to the linear model. The average computation time for the 24 task branched problems is about 450 seconds per replication, and it takes up to 900 seconds per replication for cases where the utilization is constrained to be less than 100%. In general, the computation time per replication for the GA is small enough to allow for more than 20 replications per case, thus allowing the algorithm to yield better results. We can then choose the best one found.

Table 4 - 14: Comparison between CPLEX and GA Solutions for the Linear Model

Number of Tasks	Utilization	Assignment	Demand	Set0				Set1			
				CPLEX Solution	GA Solution			CPLEX Solution	GA Solution		
					Best	Mean	StdDev		Best	Mean	StdDev
6	100%	1 Task	180	2.50	2.88	3.74	1.35	45.56	45.54	49.37	7.92
	100%	1 Task	220	2.50	2.88	5.90	4.30	NFSE	45.54	46.67	5.05
	100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	100%	2 Tasks	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	1 Task	180	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.10
	<100%	1 Task	220	0.00	0.00	0.68	1.66	NFSE	0.23	10.93	9.24
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	100%	1 Task	180	NFSE	2.75	2.75	0.00	NFSE	13.77	14.73	2.35
	100%	1 Task	220	NFSE	20.13	23.22	3.86	NFSE	17.66	18.97	3.20
	100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	100%	2 Tasks	220	NFSE	7.29	7.81	0.65	NFSF	3.05	6.23	1.00
	<100%	1 Task	180	NFSE	3.85	8.69	3.99	NFSE	9.47	11.82	2.95
	<100%	1 Task	220	NFSE	11.49	18.24	6.09	NFSE	16.11	24.13	5.86
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	NFSE	7.58	8.90	0.73	NFSF	6.41	6.75	0.40
24	100%	1 Task	180	NFSF	12.96	16.01	2.96	NFSF	14.73	24.63	7.10
	100%	1 Task	220	NFSF	11.84	20.75	7.87	NFSF	12.91	26.27	5.26
	100%	2 Tasks	180	NFSF	0.00	0.00	0.00	NFSF	0.00	0.00	0.00
	100%	2 Tasks	220	NFSF	0.00	0.00	0.00	NFSF	0.00	0.00	0.00
	<100%	1 Task	180	NFSF	0.00	0.44	1.36	NFSF	0.00	0.00	0.00
	<100%	1 Task	220	NFSF	3.63	10.09	4.21	NFSF	0.00	4.09	4.66
	<100%	2 Tasks	180	NFSF	0.00	0.00	0.00	NFSF	0.00	0.14	0.62
	<100%	2 Tasks	220	NFSF	0.00	0.00	0.00	NFSF	0.00	0.00	0.00

Table 4 – 14 Continued

Number of Tasks	Utilization	Assignment	Demand	Set2				Set3			
				CPLEX Solution	GA Solution			CPLEX Solution	GA Solution		
					Best	Mean	StdDev		Best	Mean	StdDev
6	100%	1 Task	180	0.00	0.00	0.00	0.00	14.22	0.00	0.00	0.00
	100%	1 Task	220	0.00	0.00	0.00	0.00	14.22	14.09	14.09	0.00
	100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	100%	2 Tasks	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	1 Task	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	1 Task	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	100%	1 Task	180	10.63	10.65	11.48	1.17	NFSE	4.52	4.61	0.05
	100%	1 Task	220	NFSE	11.31	11.59	0.88	NFSE	12.52	16.58	3.74
	100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	100%	2 Tasks	220	0.00	0.00	0.19	0.50	NFSF	2.31	3.04	0.70
	<100%	1 Task	180	0.00	0.00	2.59	2.15	NFSE	5.19	5.65	0.31
	<100%	1 Task	220	0.00	9.44	15.71	3.07	NFSE	11.14	16.18	2.76
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.20	0.41	NFSF	2.32	3.77	0.60
24	100%	1 Task	180	NFSF	0.00	0.81	2.52	NFSF	0.00	0.71	3.16
	100%	1 Task	220	NFSF	5.04	11.21	3.63	NFSF	0.00	7.96	7.68
	100%	2 Tasks	180	NFSF	0.00	0.00	0.00	NFSF	0.00	0.00	0.00
	100%	2 Tasks	220	NFSF	0.00	0.00	0.00	NFSF	0.00	0.00	0.00
	<100%	1 Task	180	NFSF	0.00	0.04	0.17	NFSF	0.00	0.57	1.40
	<100%	1 Task	220	NFSF	0.00	7.20	8.83	NFSF	0.00	1.53	2.18
	<100%	2 Tasks	180	NFSF	0.00	0.00	0.00	NFSF	0.00	0.00	0.00
	<100%	2 Tasks	220	NFSF	0.00	0.00	0.00	NFSF	0.00	0.00	0.00

Table 4 – 14 Continued

Number of Tasks	Utilization	Assignment	Demand	Set4			
				CPLEX Solution	GA Solution		
					Best	Mean	StdDev
6	100%	1 Task	180	10.50	10.58	12.15	3.86
	100%	1 Task	220	10.50	10.58	12.59	4.16
	100%	2 Tasks	180	0.00	0.00	0.00	0.00
	100%	2 Tasks	220	0.00	0.00	0.00	0.00
	<100%	1 Task	180	0.00	0.00	0.00	0.00
	<100%	1 Task	220	0.00	0.00	0.04	0.14
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.00	0.00
12	100%	1 Task	180	0.00	0.00	0.00	10.45
	100%	1 Task	220	NFSE	0.00	0.00	10.00
	100%	2 Tasks	180	0.00	0.00	0.00	0.00
	100%	2 Tasks	220	0.00	0.00	0.00	0.00
	<100%	1 Task	180	0.00	0.00	0.61	5.14
	<100%	1 Task	220	NFSE	0.00	1.19	8.23
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.00	0.00
24	100%	1 Task	180	NFSF	1.67	9.44	5.44
	100%	1 Task	220	NFSF	3.21	7.59	3.52
	100%	2 Tasks	180	NFSF	0.00	0.00	0.00
	100%	2 Tasks	220	NFSF	0.00	0.00	0.00
	<100%	1 Task	180	NFSF	0.00	0.23	1.03
	<100%	1 Task	220	NFSF	0.00	2.59	5.71
	<100%	2 Tasks	180	NFSF	0.00	0.00	0.00
	<100%	2 Tasks	220	NFSF	0.00	0.00	0.00

Table 4 - 15: Comparison between CPLEX and GA Computation Times for the Linear Model

Number of Tasks	Utilization	Assignment	Demand	Set0			Set1		
				CPLEX Computation Time (s)	GA Computation Time (s)		CPLEX Computation Time (s)	GA Computation Time (s)	
					Mean	StdDev		Mean	StdDev
6	100%	1 Task	180	3.12	151.25	34.40	2.42	152.01	33.84
	100%	1 Task	220	3.44	283.98	41.20	0.88	283.85	41.45
	100%	2 Tasks	180	0.80	0.21	0.00	0.86	0.21	0.00
	100%	2 Tasks	220	2.09	0.21	0.00	0.81	0.22	0.02
	<100%	1 Task	180	0.81	1.07	1.30	0.71	42.83	86.20
	<100%	1 Task	220	0.70	85.41	131.91	0.88	426.57	39.52
	<100%	2 Tasks	180	0.76	0.19	0.01	0.89	0.32	0.13
	<100%	2 Tasks	220	0.80	0.24	0.08	0.72	0.73	0.27
12	100%	1 Task	180	14.36	197.97	34.25	16.84	196.46	33.99
	100%	1 Task	220	0.04	326.89	39.55	18.06	329.81	40.25
	100%	2 Tasks	180	24.52	0.87	0.44	20.69	1.86	5.24
	100%	2 Tasks	220	20.05	501.89	38.64	10800.00	503.99	42.48
	<100%	1 Task	180	14.97	566.24	38.48	17.37	173.32	34.33
	<100%	1 Task	220	0.06	690.99	36.18	19.00	306.15	39.76
	<100%	2 Tasks	180	11.84	10.50	24.20	11.35	2.68	1.51
	<100%	2 Tasks	220	20.75	841.00	37.09	10800.00	463.88	40.39
24	100%	1 Task	180	10800.00	333.96	33.61	10800.00	352.98	35.38
	100%	1 Task	220	10800.00	465.92	39.94	10800.00	485.27	39.05
	100%	2 Tasks	180	10800.00	1.23	0.49	10800.00	1.60	0.87
	100%	2 Tasks	220	10800.00	3.86	2.16	10800.00	4.11	1.97
	<100%	1 Task	180	10800.00	159.04	183.44	10800.00	82.34	114.68
	<100%	1 Task	220	10800.00	578.86	37.74	10800.00	475.81	159.98
	<100%	2 Tasks	180	10800.00	36.97	94.24	10800.00	84.49	175.51
	<100%	2 Tasks	220	10800.00	35.59	76.68	10800.00	15.90	3.49

Table 4 -15 Continued

Number of Tasks	Utilization	Assignment	Demand	Set2			Set3		
				CPLEX Computation Time (s)	GA Computation Time (s)		CPLEX Computation Time (s)	GA Computation Time (s)	
					Mean	StdDev		Mean	StdDev
6	100%	1 Task	180	0.77	0.19	0.09	2.80	152.20	34.35
	100%	1 Task	220	0.73	0.17	0.00	2.75	283.92	41.27
	100%	2 Tasks	180	1.40	0.15	0.00	0.80	0.21	0.00
	100%	2 Tasks	220	0.92	0.15	0.00	0.78	0.21	0.00
	<100%	1 Task	180	0.77	0.25	0.06	0.79	0.97	1.52
	<100%	1 Task	220	0.76	1.83	4.39	0.89	33.61	82.77
	<100%	2 Tasks	180	0.76	0.12	0.00	0.78	0.18	0.00
	<100%	2 Tasks	220	0.78	0.13	0.02	0.78	0.18	0.00
12	100%	1 Task	180	66.64	194.70	34.23	14.51	196.92	34.48
	100%	1 Task	220	0.05	326.39	39.98	26.80	326.77	40.07
	100%	2 Tasks	180	15.71	0.48	0.14	13.78	0.53	0.20
	100%	2 Tasks	220	19.69	100.15	177.00	10800.00	502.52	39.34
	<100%	1 Task	180	11.16	304.01	206.56	14.41	174.83	34.33
	<100%	1 Task	220	0.07	553.81	36.40	26.98	305.37	41.12
	<100%	2 Tasks	180	11.48	2.52	0.95	10.72	1.21	0.78
	<100%	2 Tasks	220	25.31	144.49	260.90	10800.00	461.82	40.12
24	100%	1 Task	180	10800.00	254.58	154.63	10800.00	33.89	75.32
	100%	1 Task	220	10800.00	459.31	42.60	10800.00	292.90	127.66
	100%	2 Tasks	180	10800.00	1.23	0.52	10800.00	1.50	0.88
	100%	2 Tasks	220	10800.00	1.85	1.26	10800.00	2.75	1.22
	<100%	1 Task	180	10800.00	77.26	124.79	10800.00	127.10	140.15
	<100%	1 Task	220	10800.00	394.62	210.48	10800.00	328.24	171.36
	<100%	2 Tasks	180	10800.00	7.65	5.22	10800.00	6.64	2.85
	<100%	2 Tasks	220	10800.00	11.29	3.81	10800.00	11.54	2.70

Table 4 -15 Continued

Number of Tasks	Utilization	Assignment	Demand	Set4		
				CPLEX Computation Time (s)	GA Computation Time (s)	
					Mean	StdDev
6	100%	1 Task	180	3.79	148.61	34.29
	100%	1 Task	220	4.28	280.47	40.97
	100%	2 Tasks	180	0.92	0.21	0.00
	100%	2 Tasks	220	0.83	0.21	0.00
	<100%	1 Task	180	0.76	8.65	19.79
	<100%	1 Task	220	0.79	66.92	123.55
	<100%	2 Tasks	180	0.75	0.18	0.00
	<100%	2 Tasks	220	0.76	0.18	0.00
12	100%	1 Task	180	10.73	0.34	0.20
	100%	1 Task	220	32.81	192.65	34.59
	100%	2 Tasks	180	13.95	0.27	0.07
	100%	2 Tasks	220	13.05	0.72	0.35
	<100%	1 Task	180	10.77	45.86	83.38
	<100%	1 Task	220	36.76	333.05	39.38
	<100%	2 Tasks	180	12.91	1.06	0.58
	<100%	2 Tasks	220	14.38	31.81	96.17
24	100%	1 Task	180	10800.00	383.29	38.73
	100%	1 Task	220	10800.00	517.92	39.28
	100%	2 Tasks	180	10800.00	1.00	0.46
	100%	2 Tasks	220	10800.00	3.20	1.73
	<100%	1 Task	180	10800.00	65.01	125.90
	<100%	1 Task	220	10800.00	157.74	204.72
	<100%	2 Tasks	180	10800.00	66.30	152.87
	<100%	2 Tasks	220	10800.00	11.60	3.13

Table 4 - 16: Comparison between CPLEX and GA Solutions for the Branched Model

Number of Tasks	Utilization	Assignment	Demand	Set0				Set1			
				CPLEX Solution	GA Solution			CPLEX Solution	GA Solution		
					Best	Mean	StdDev		Best	Mean	StdDev
6	100%	1 Task	180	16.89	17.07	23.98	5.79	0.89	0.92	0.92	0.00
	100%	1 Task	220	35.72	35.79	37.59	3.00	75.00	75.11	75.11	0.00
	100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.11	0.23
	100%	2 Tasks	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	1 Task	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	1 Task	220	0.00	0.00	0.99	2.54	0.00	0.00	1.92	2.75
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	100%	1 Task	180	NFSE	15.70	16.49	3.55	39.11	38.95	39.67	1.38
	100%	1 Task	220	NFSE	27.55	29.65	1.91	NFSE	42.81	45.31	2.37
	100%	2 Tasks	180	3.46	11.75	14.60	3.83	9.33	7.99	16.87	10.08
	100%	2 Tasks	220	NFSE	13.37	15.44	1.56	8.83	7.98	16.68	10.09
	<100%	1 Task	180	NFSE	3.85	7.53	3.26	0.00	0.00	3.10	7.54
	<100%	1 Task	220	NFSE	12.14	19.16	5.16	NFSE	7.36	14.09	12.46
	<100%	2 Tasks	180	0.00	0.00	0.03	0.05	0.00	0.00	0.01	0.03
	<100%	2 Tasks	220	NFSE	8.25	9.01	0.79	0.00	0.00	0.00	0.00
24	100%	1 Task	180	59.96	31.55	37.98	4.18	39.50	27.73	34.33	4.06
	100%	1 Task	220	NFSF	31.67	41.98	5.39	NFSF	29.27	36.63	4.55
	100%	2 Tasks	180	43.62	0.84	4.15	3.23	23.47	0.00	1.14	1.36
	100%	2 Tasks	220	40.48	0.45	6.59	5.50	NFSF	0.00	1.69	2.14
	<100%	1 Task	180	0.00	0.00	2.18	3.49	0.00	0.00	8.42	7.55
	<100%	1 Task	220	NFSF	4.71	12.65	5.00	NFSF	3.76	18.41	10.28
	<100%	2 Tasks	180	0.32	0.00	1.41	1.29	0.00	0.00	0.07	0.18
	<100%	2 Tasks	220	0.00	0.00	0.21	0.41	0.00	0.00	0.01	0.03

Table 4 -16 Continued

Number of Tasks	Utilization	Assignment	Demand	Set2				Set3			
				CPLEX Solution	GA Solution			CPLEX Solution	GA Solution		
					Best	Mean	StdDev		Best	Mean	StdDev
6	100%	1 Task	180	43.33	43.37	49.04	5.83	58.22	58.22	33.03	28.44
	100%	1 Task	220	43.33	43.37	49.13	5.94	NFSE	2.39	33.03	28.44
	100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	100%	2 Tasks	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	1 Task	180	0.00	0.00	0.03	0.11	0.00	0.00	4.40	3.29
	<100%	1 Task	220	0.00	0.00	0.00	0.00	NFSE	21.79	25.64	4.53
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	100%	1 Task	180	32.00	32.20	32.69	1.19	NFSE	50.21	54.79	5.61
	100%	1 Task	220	NFSE	32.20	33.41	1.00	NFSE	49.83	51.87	3.86
	100%	2 Tasks	180	0.00	1.26	6.64	7.09	8.30	7.85	15.89	7.32
	100%	2 Tasks	220	NFSE	17.75	27.24	6.80	NFSE	8.46	15.06	6.95
	<100%	1 Task	180	0.00	0.00	0.87	3.88	NFSE	9.09	9.98	1.24
	<100%	1 Task	220	NFSE	6.61	11.10	8.18	NFSE	23.05	24.84	1.49
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	NFSE	0.62	5.58	2.41	NFSE	6.65	9.09	1.04
24	100%	1 Task	180	37.92	16.26	27.65	5.88	67.29	36.26	42.33	3.99
	100%	1 Task	220	62.00	15.49	28.67	7.96	74.10	32.03	44.65	6.89
	100%	2 Tasks	180	60.22	1.31	7.05	4.81	71.49	0.57	6.06	5.49
	100%	2 Tasks	220	37.66	1.35	7.30	4.48	13.76	0.65	5.00	2.44
	<100%	1 Task	180	0.00	0.00	0.03	0.06	0.00	0.00	0.15	0.54
	<100%	1 Task	220	0.00	0.00	2.33	3.83	0.00	0.00	4.28	6.60
	<100%	2 Tasks	180	13.57	0.00	0.26	0.55	11.76	0.00	0.80	1.41
	<100%	2 Tasks	220	0.00	0.00	0.20	0.72	4.63	0.00	0.55	0.99

Table 4 -16 Continued

Number of Tasks	Utilization	Assignment	Demand	Set4			
				CPLEX Solution	GA Solution		
					Best	Mean	StdDev
6	100%	1 Task	180	20.33	20.21	22.13	0.66
	100%	1 Task	220	20.33	20.21	22.24	0.48
	100%	2 Tasks	180	0.00	0.00	0.00	0.00
	100%	2 Tasks	220	0.00	0.00	0.00	0.00
	<100%	1 Task	180	0.00	0.00	0.00	0.00
	<100%	1 Task	220	0.00	0.00	1.09	1.71
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.00	0.00
12	100%	1 Task	180	33.03	33.11	36.66	4.91
	100%	1 Task	220	33.03	33.11	35.83	3.68
	100%	2 Tasks	180	12.50	6.82	12.30	3.30
	100%	2 Tasks	220	8.89	7.46	12.90	3.35
	<100%	1 Task	180	0.00	0.00	0.14	0.61
	<100%	1 Task	220	0.00	0.00	1.68	4.10
	<100%	2 Tasks	180	0.00	0.00	0.00	0.00
	<100%	2 Tasks	220	0.00	0.00	0.00	0.00
24	100%	1 Task	180	55.42	26.18	34.38	5.97
	100%	1 Task	220	55.22	28.44	38.54	6.96
	100%	2 Tasks	180	14.04	1.56	8.69	5.16
	100%	2 Tasks	220	24.34	0.33	8.18	4.38
	<100%	1 Task	180	0.00	0.00	0.44	1.42
	<100%	1 Task	220	0.00	0.00	8.69	7.83
	<100%	2 Tasks	180	0.00	0.00	0.13	0.30
	<100%	2 Tasks	220	0.00	0.00	0.11	0.24

Table 4 - 17: Comparison between CPLEX and GA Computation Times for the Branched Model

Number of Tasks	Utilization	Assignment	Demand	Set0			Set1		
				CPLEX Computation Time (s)	GA Computation Time (s)		CPLEX Computation Time (s)	GA Computation Time (s)	
					Mean	StdDev		Mean	StdDev
6	100%	1 Task	180	2.85	147.70	33.72	2.77	150.11	33.20
	100%	1 Task	220	3.73	278.56	40.96	0.72	282.51	40.00
	100%	2 Tasks	180	3.55	1.56	0.70	5.74	76.42	153.84
	100%	2 Tasks	220	2.53	1.39	0.75	2.40	1.35	0.47
	<100%	1 Task	180	0.64	4.73	4.99	0.64	2.96	1.48
	<100%	1 Task	220	0.60	78.43	129.64	0.61	326.93	142.45
	<100%	2 Tasks	180	0.72	1.66	0.58	0.64	2.21	1.21
	<100%	2 Tasks	220	0.67	1.90	0.91	0.66	1.91	0.79
12	100%	1 Task	180	17.63	195.69	34.09	10800.00	191.21	34.45
	100%	1 Task	220	0.03	330.08	38.73	14.61	322.95	39.91
	100%	2 Tasks	180	10800.00	520.20	39.96	10800.00	508.04	37.71
	100%	2 Tasks	220	14.32	648.83	37.82	10800.00	629.98	39.81
	<100%	1 Task	180	21.25	181.46	34.57	6.56	171.23	256.18
	<100%	1 Task	220	0.02	314.57	41.21	16.43	730.77	37.15
	<100%	2 Tasks	180	6.75	160.95	183.39	6.68	120.43	205.05
	<100%	2 Tasks	220	19.06	512.36	35.71	6.45	80.27	147.61
24	100%	1 Task	180	10800.00	330.27	34.29	10800.00	369.90	37.91
	100%	1 Task	220	10800.00	462.71	40.27	10800.00	504.23	39.81
	100%	2 Tasks	180	10800.00	693.74	37.32	10800.00	653.11	263.82
	100%	2 Tasks	220	10800.00	820.03	34.73	10800.00	824.75	186.07
	<100%	1 Task	180	9741.09	240.11	89.73	7461.22	295.28	37.86
	<100%	1 Task	220	10800.00	402.62	41.52	10800.00	428.72	46.16
	<100%	2 Tasks	180	10800.00	496.63	144.96	6241.35	475.23	163.71
	<100%	2 Tasks	220	10800.00	532.08	215.90	2304.52	464.53	216.13

Table 4 -17 Continued

Number of Tasks	Utilization	Assignment	Demand	Set2			Set3		
				CPLEX Computation Time (s)	GA Computation Time (s)		CPLEX Computation Time (s)	GA Computation Time (s)	
					Mean	StdDev		Mean	StdDev
6	100%	1 Task	180	3.60	152.57	33.74	3.21	150.25	34.17
	100%	1 Task	220	5.63	284.80	40.87	3.98	281.95	40.94
	100%	2 Tasks	180	6.81	2.37	1.37	1.53	1.44	0.73
	100%	2 Tasks	220	3.58	2.92	1.81	1072.29	1.60	0.88
	<100%	1 Task	180	0.58	57.05	103.86	0.68	323.03	146.23
	<100%	1 Task	220	0.58	6.78	8.90	4.25	513.16	36.21
	<100%	2 Tasks	180	0.73	2.25	1.34	0.64	2.49	1.48
	<100%	2 Tasks	220	0.71	2.50	2.37	0.68	3.34	1.37
12	100%	1 Task	180	10800.00	192.43	32.83	8.22	194.76	34.00
	100%	1 Task	220	0.05	321.61	39.94	9.19	326.24	40.05
	100%	2 Tasks	180	4329.22	503.06	40.67	10800.00	512.81	39.65
	100%	2 Tasks	220	112.71	634.61	37.81	28.01	634.83	38.63
	<100%	1 Task	180	6.53	17.15	28.81	8.36	184.85	34.65
	<100%	1 Task	220	0.08	195.66	34.45	9.24	317.18	41.74
	<100%	2 Tasks	180	6.51	12.07	8.26	6.48	36.27	68.56
	<100%	2 Tasks	220	113.50	348.81	39.81	27.67	477.64	38.45
24	100%	1 Task	180	10800.00	372.32	37.95	10800.00	324.81	34.64
	100%	1 Task	220	10800.00	507.10	40.27	10800.00	455.97	39.03
	100%	2 Tasks	180	10800.00	773.58	39.18	10800.00	679.46	33.84
	100%	2 Tasks	220	10800.00	891.97	35.74	10800.00	804.59	40.71
	<100%	1 Task	180	10800.00	195.91	110.42	7250.94	219.18	105.98
	<100%	1 Task	220	10364.79	352.33	76.81	10800.00	338.75	117.57
	<100%	2 Tasks	180	10800.00	448.74	145.55	10800.00	505.83	63.85
	<100%	2 Tasks	220	10800.00	525.42	180.23	10800.00	616.62	93.71

Table 4 -17 Continued

Number of Tasks	Utilization	Assignment	Demand	Set4 CPLEX Computation Time (s)	Set4 GA Computation Time (s) Mean	Set4 GA Computation Time (s) StdDev
6	100%	1 Task	180	2.66	149.24	34.66
	100%	1 Task	220	2.02	282.30	41.42
	100%	2 Tasks	180	2.96	1.82	0.88
	100%	2 Tasks	220	3.77	1.36	0.49
	<100%	1 Task	180	0.67	4.65	4.18
	<100%	1 Task	220	0.68	143.09	167.45
	<100%	2 Tasks	180	0.66	1.55	0.51
	<100%	2 Tasks	220	0.66	1.82	1.06
12	100%	1 Task	180	10800.00	188.70	34.16
	100%	1 Task	220	10800.00	320.17	39.85
	100%	2 Tasks	180	10800.00	498.02	41.43
	100%	2 Tasks	220	10800.00	620.82	41.79
	<100%	1 Task	180	6.60	87.20	192.38
	<100%	1 Task	220	6.27	129.59	231.26
	<100%	2 Tasks	180	6.54	28.71	5.35
	<100%	2 Tasks	220	13.71	55.97	60.50
24	100%	1 Task	180	10800.00	341.06	35.13
	100%	1 Task	220	10800.00	474.41	39.86
	100%	2 Tasks	180	10800.00	705.21	37.96
	100%	2 Tasks	220	10800.00	828.21	38.17
	<100%	1 Task	180	4977.30	236.45	85.58
	<100%	1 Task	220	9389.36	396.14	44.17
	<100%	2 Tasks	180	10800.00	390.50	210.97
	<100%	2 Tasks	220	9174.62	551.89	189.47

Chapter 5

Simulation

In the previous two chapters, the apparel assembly line balancing problem was solved utilizing static processing times. Also, the static model assumed that a partial assignment could potentially really happen at the same time (i.e., worker assigned to two tasks would contribute to the processing capability of each task independently). In this chapter we develop simulation models using SIMIO to account for the stochastic nature of the system as well as the modeling of the partial assignment. We consider two simple cases – Set0 of the branched six task example and Set0 of the linear six task example and describe the methodology involved to convert a static model to a dynamic one. We use the task assignments obtained by solving these static models. We compare the results of the deterministic system and stochastic systems with ±10%, ±20% and ±30% variation in the Standard Operation Time based on a PERT distribution. We also determine when a worker is required to move to another station based on the number waiting in queue (termed as buffer size) at each station by comparing the effects of the following policies:

1. **Policy A: No buffers**

   Workers move whenever an entity is waiting to be processed. This is a worst case scenario where there are no buffer size restrictions and workers essentially move when there is a request. These are handled in a FIFO order which could result in doing one garment and then moving to another station to do another garment etc.

2. **Policy B: Common maximum and minimum buffer size for each station**

   Workers move to another station when the number of entities waiting at that stations queue exceeds the common maximum buffer size for the system. Workers move back to their original station when the number of entities in the current stations queue falls below the common minimum buffer size for the system.

Buffer sizes are represented as *'MaxAMinB'* where '*A'* and '*B'* are integers that range from 0 to 10 and *A* > *B*. For example, 'Max5Min2' would represent a maximum buffer size of five and a minimum buffer size of two. If a worker was assigned to Station A and the number of entities waiting at Station B exceeded five, the worked would shift to station B. If the number of entities waiting at Station B was less than or equal to two, the worker would then move back to Station A.

5.1 Simulation Methodology

To convert the system from a static to a dynamic model, the following methodology is adopted in the experimentation.

1. Examine the worker to task assignments generated by the static model.
2. For workers with utilization lower than 75%, we assign a work schedule that forces the worker to be idle for a period of time that approximates the actual utilization.
3. The inter-arrival time of entities is determined by:

$$\max_{i=1\ to\ N} O_i$$

   Where O_i is the Operation time of task *i*.
4. Simulate the system using three arbitrary buffer sizes to determine the warm up period to reach steady state. The experiment is conducted with 500 replications.
5. Use this warm up period to compare the above policies.
   A. In the case of no buffers, we determine the values of MAD, actual production and total worker movement for the deterministic and stochastic systems. The experiment is conducted with 500 replications.
   B. In the case of common maximum and minimum buffer sizes, we use OptQuest to determine the common "optimal" maximum and minimum buffer size(s) that are able to meet demand as well as minimize MAD for the deterministic and stochastic systems. The term "optimal" here refers to finding the best

solution possible. There is no guarantee that it is optimal as OptQuest utilizes a combination of linear programming, Scatter and Tabu search coupled with a Neural Network regression limiting model. The experiment is conducted with 500 replications. We constrain the maximum size of a buffer to be no more than ten.

6. For policy B the correlation between the standard deviation of the MAD and standard deviation of Actual Production is computed. A value closer to one would indicate that scenarios with a low standard deviation in MAD would also have a low standard deviation in actual production and vice versa. Thus, if a scenario is chosen where the actual production meets the demand with a low standard deviation, it would imply that the variation in MAD for that particular scenario would also be low.
7. Also, for policy B we filter buffer sizes based on MAD, actual production and total movement to reduce the number of multiple means comparisons. Tukey's test was used to compare these filtered results in terms of MAD to determine buffer sizes that are statistically different from others. While testing for normality, we found that the data was right skewed, which violates the normality assumption for Tukey's test. We applied a square root transformation to the MAD to account for this. The square root transformation in this case does not affect the interpretation of MAD by a significant amount, as the value for MAD is always less than 1.5.

Note that for the deterministic case, the half width and standard deviation are not always zero because workers are selected randomly and assigned to a task based on their skill, and because each workers efficiency is different, the value of the responses may change for certain replications.

The variation in processing time for the four systems is shown in Table 5 - 1. A Pert distribution was chosen to model the variability about the standard time that is normally used in the industry as well as the static models of the two previous chapters.

Table 5 - 1: Variation in processing time for the four systems

Task	Standard Processing Time (in min)	Stochastic with ±10% (in min)	Stochastic with ±20% (in min)	Stochastic with ±30% (in min)
1	5	PERT(4.5,5,5.5)	PERT(4,5,6)	PERT(3.5,5,6.5)
2	6	PERT(5.4,6,6.6)	PERT(4.8,6,7.2)	PERT(4.2,6,7.8)
3	8	PERT(7.2,8,8.8)	PERT(6.4,8,9.6)	PERT(5.6,8,10.4)
4	5	PERT(4.5,5,5.5)	PERT(4,5,6)	PERT(3.5,5,6.5)
5	3	PERT(2.7,3,3.3)	PERT(2.4,3,3.6)	PERT(2.1,3,3.9)
6	7	PERT(6.3,7,7.7)	PERT(5.6,7,8.4)	PERT(4.9,7,9.1)

5.2 Linear Model with 6 Tasks

The static worker to task assignments given by CPLEX is shown in Table 5 - 2. We simulate the system using three arbitrary buffer sizes to determine the warm up period to reach steady state. Table 5 - 3 shows the change in MAD and Actual Production as the warm up period increases. Figure 5 - 1 and Figure 5 - 2 shows that the system fills up and reaches steady state with a warm up period of ten hours.

Table 5 - 2: Static Assignments for the Linear 6 Tasks Model

	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Total Utilization
Worker 1							0.00
Worker 2			33.45			63.34	96.79
Worker 3				30.93	54.25		85.18
Worker 4	74.70		25.24				99.94
Worker 5							0.00
Worker 6				46.49		29.09	75.58
Worker 7		89.20					89.20
Worker 8			77.47				77.47
Worker 9							0.00

Table 5 - 3: MAD and Actual Production (AP) with warm up

	Deterministic		**Stochastic with ±10%**		**Stochastic with ±20%**		**Stochastic with ±30%**	
Warm up (in hours)	**MAD**	**Actual Production**	**MAD**	**Actual Production**	**MAD**	**Actual Production**	**MAD**	**Actual Production**
0	4.7778	164	4.2022	165.04	4.0822	165	4.3511	164.68
5	2	173	1.3822	175.16	1.1422	175.92	1.2	176.16
10	1.5556	177	1.1378	179.32	0.9267	179.44	1.0844	178.48
15	1.2222	178	1.2044	179.12	1.0933	179.64	1.0911	179.36
20	0.8889	179	1.1422	180.72	1.0378	179.8	1.16	179.92

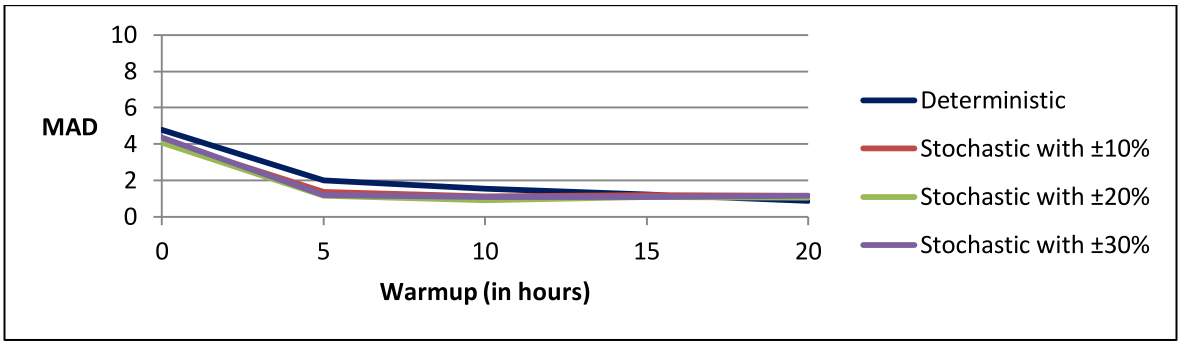

Figure 5 - 1: MAD vs. Warm up

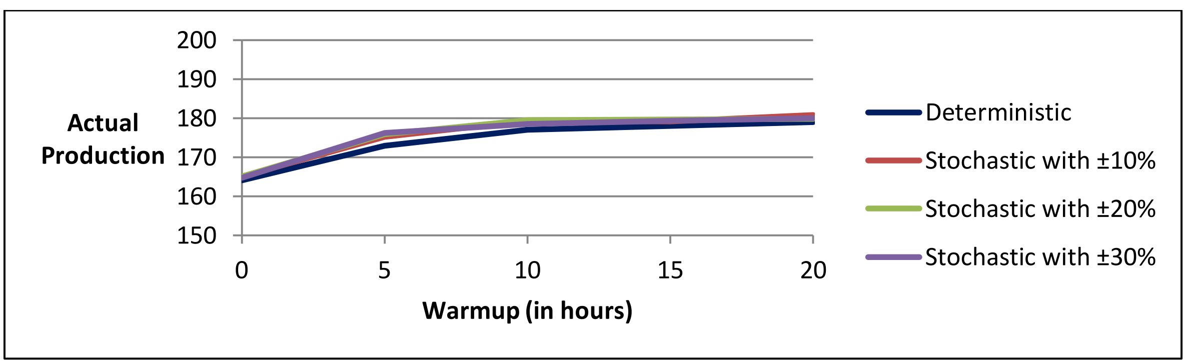

Figure 5 - 2: Actual Production vs. Warm up

5.2.1 No Buffer Policy

Using a warm-up period of 10 hours, we apply policy A. The results are shown in Table 5 - 4.

Table 5 - 4: MAD, Actual Production and Total Movement for policy A

Type of System	MAD			Actual Production			Total Movement		
	Average	HW	SD	Average	HW	SD	Average	HW	SD
Deterministic	0.496	0.092	0.222	180.04	0.30	0.74	561.96	4.94	11.98
Stochastic with ±10%	0.487	0.069	0.166	179.8	0.4	0.96	580.96	4.93	11.96
Stochastic with ±20%	0.584	0.081	0.196	180.08	0.4	0.96	590.68	6.13	14.85
Stochastic with ±30%	0.551	0.073	0.178	180	0.6	1.45	591.36	4.26	10.31

The results show that policy A is able to meet the daily demand of 180 units for the deterministic as well as the stochastic cases. The MAD is low, indicating that the line is balanced. But the total movement is very high and can be reduced if we apply a different policy.

5.2.2 Common Buffer Size

Using a warm-up period of ten hours, policy B is applied to all four systems. To reduce the number of Tukey comparisons, we filter the data to include buffer sizes with a MAD no more than 1.5, actual production no less than 179 and total movement no more than 60. We use an actual production of 179 in this case because most of the policies are unable to meet an average actual production of 180. The mean, 95% confidence limits and standard deviation of MAD, Actual Production and Total Movement for the filtered buffer sizes for each system, as well as the Tukey's test results is shown in Table 5 - 5 through Table 5 - 12.

Table 5 - 5: MAD, Actual Production and Total Movement after applying policy B to the Linear 6 Task Deterministic System

Buffer Size	Variable	Mean	Upper 95% CL for Mean	Lower 95% CL for Mean	Std Dev
Max10Min4	MAD	1.278	1.278	1.278	0
	Actual Production	184	.	.	0
	Total Movement	36	.	.	0
Max10Min6	MAD	1.444	1.444	1.444	0
	Actual Production	181	.	.	0
	Total Movement	31	.	.	0
Max10Min7	MAD	0.278	.	.	0
	Actual Production	180	.	.	0
	Total Movement	44	.	.	0
Max5Min1	MAD	1.111	.	.	0
	Actual Production	183	.	.	0
	Total Movement	55	.	.	0
Max5Min2	MAD	0.278	.	.	0
	Actual Production	181	.	.	0
	Total Movement	60	.	.	0
Max6Min3	MAD	0.278	.	.	0
	Actual Production	182	.	.	0
	Total Movement	60	.	.	0
Max7Min4	MAD	0.556	.	.	0
	Actual Production	180	.	.	0
	Total Movement	60	.	.	0
Max8Min4	MAD	1	.	.	0
	Actual Production	181	.	.	0
	Total Movement	36	.	.	0
Max9Min3	MAD	1	.	.	0
	Actual Production	180	.	.	0
	Total Movement	32	.	.	0
Max9Min5	MAD	1	.	.	0
	Actual Production	181	.	.	0
	Total Movement	34	.	.	0

Table 5 - 6: Results from Tukey's test after applying policy B to the Linear 6 Task Deterministic System

Alpha	0.05	
Error Degrees of Freedom	4990	
Error Mean Square	0	
Critical Value of Studentized Range	4.4761	
Minimum Significant Difference	0	
Tukey Grouping	**Mean of Square Root of MAD**	**Buffer Size**
A	1.202	Max10Min6
B	1.13	Max10Min4
C	1.054	Max5Min1
D	1	Max9Min3
D	1	Max9Min5
D	1	Max8Min4
E	0.745	Max7Min4
F	0.527	Max10Min7
F	0.527	Max5Min2
F	0.527	Max6Min3

Table 5 - 7: MAD, Actual Production and Total Movement after applying policy B to the Linear 6 Task Stochastic ±10% System

Buffer Size	**Variable**	**Mean**	**Upper 95% CL for Mean**	**Lower 95% CL for Mean**	**Std Dev**
Max5Min2	MAD	1.288	1.343	1.233	0.626
	Actual Production	179.314	179.506	179.122	2.183
	Total Movement	57.936	58.628	57.244	7.879
Max6Min3	MAD	1.311	1.365	1.258	0.608
	Actual Production	179.248	179.464	179.032	2.464
	Total Movement	52.556	53.132	51.98	6.55
Max7Min4	MAD	1.395	1.45	1.341	0.624
	Actual Production	179.208	179.441	178.975	2.655
	Total Movement	49.974	50.527	49.421	6.292

Table 5 - 8: Results from Tukey's test after applying policy B to the Linear 6 Task Stochastic ±10% System

Alpha		0.05	
Error Degrees of Freedom		1497	
Error Mean Square		0.077473	
Critical Value of Studentized Range		3.3178	
Minimum Significant Difference		0.0413	
Tukey Grouping		**Mean of Square Root of MAD**	**Buffer Size**
	A	1.14994	Max7Min4
B	A	1.1134	Max6Min3
B		1.09596	Max5Min2

Table 5 - 9: MAD, Actual Production and Total Movement after applying policy B to the Linear 6 Task Stochastic ±20% System

Buffer Size	**Variable**	**Mean**	**Upper 95% CL for Mean**	**Lower 95% CL for Mean**	**Std Dev**
Max6Min3	MAD	1.29	1.342	1.239	0.585
	Actual Production	179.374	179.594	179.154	2.501
	Total Movement	54.214	54.797	53.631	6.631
Max7Min4	MAD	1.286	1.341	1.232	0.62
	Actual Production	179.486	179.702	179.27	2.456
	Total Movement	51.052	51.591	50.513	6.14

Table 5 - 10: Results from Tukey's test after applying policy B to the Linear 6 Task Stochastic ±20% System

Alpha		0.05
Error Degrees of Freedom		998
Error Mean Square		0.073198
Critical Value of Studentized Range		2.77517
Minimum Significant Difference		0.0336
Tukey Grouping	**Mean of Square Root of MAD**	**Buffer Size**
A	1.10546	Max6Min3
A	1.09918	Max7Min4

Table 5 - 11: MAD, Actual Production and Total Movement after applying policy B to the Linear 6 Task Stochastic ±30% System

Buffer Size	**Variable**	**Mean**	**Upper 95% CL for Mean**	**Lower 95% CL for Mean**	**Std Dev**
Max6Min3	MAD	1.319	1.375	1.263	0.636
	Actual Production	179.28	179.509	179.059	2.556
	Total Movement	55.08	55.659	54.501	6.589
Max7Min4	MAD	1.3	1.353	1.247	0.601
	Actual Production	179.24	179.462	179.018	2.532
	Total Movement	52.228	52.759	51.697	6.04

Table 5 - 12: Results from Tukey's test after applying policy B to the Linear 6 Task Stochastic ±30% System

Alpha	0.05	
Error Degrees of Freedom	998	
Error Mean Square	0.0754	
Critical Value of Studentized Range	2.7752	
Minimum Significant Difference	0.0341	
Tukey Grouping	**Mean of Square Root of MAD**	**Buffer Size**
A	1.1139	Max6Min3
A	1.1083	Max7Min4

Using the filtered data and the results from Tukey's test, we can conclude that the best general policies for the linear six task example are 'Max6Min3' and 'Max7Min4', as they are able to meet the filter criteria for all four systems and there is no statistically significant difference between these two buffer sizes.

5.3 Branched Model with 6 Tasks

The static worker to task assignments given by CPLEX is shown in Table 5 - 13. Note that Worker 1's utilization is less than 75%. We implement a work schedule to approximate a utilization of 43.89%.

We simulate the system using three arbitrary buffer sizes to determine the warm up period to reach steady state. Table 5 - 14 shows the change in MAD and Actual Production as the warm up period increases. Figure 5 - 3 and Figure 5 - 4 shows that the system fills up and reaches steady state with a warm up period of ten hours.

Table 5 - 13: Static Assignments for the Branched 6 Tasks Model

	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Total
Worker 1	43.89						43.89
Worker 2						89.29	89.29
Worker 3				48.23	42.64		90.86
Worker 4	25		56.13				81.13
Worker 5							0.00
Worker 6							0.00
Worker 7		87.21					87.21
Worker 8			75	25			100.00
Worker 9							0.00

Table 5 - 14: Change in MAD and Actual Production with warm up

	Deterministic		**Stochastic with ±10%**		**Stochastic with ±20%**		**Stochastic with ±30%**	
Warm up (in hours)	**MAD**	**Actual Production**	**MAD**	**Actual Production**	**MAD**	**Actual Production**	**MAD**	**Actual Production**
0	6.907	164.56	6.847	163.84	6.613	164.28	6.913	163.6
5	2.040	183.56	2.773	180.68	2.618	181.28	2.442	181.2
10	3.727	187.56	3.800	183.52	3.760	184.68	3.720	184.92
15	3.460	188	3.780	184.76	3.896	186.2	3.622	186.44
20	3.813	187.56	4.033	187.08	3.864	186.84	3.662	187.2

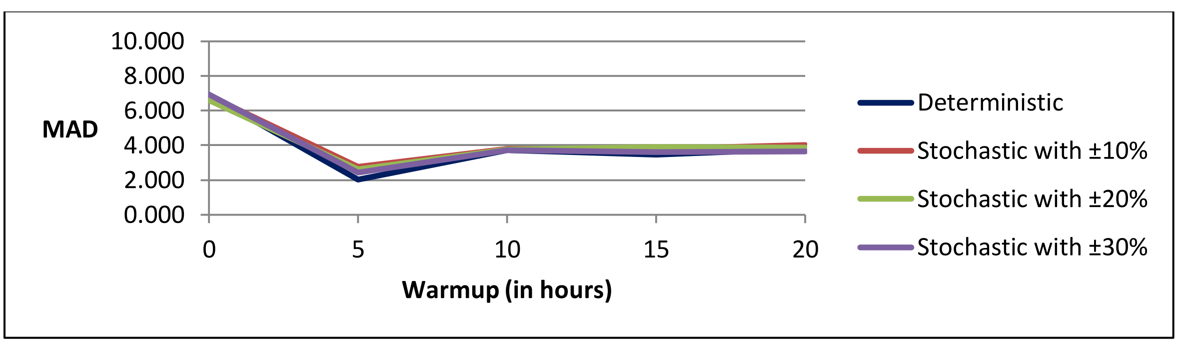

Figure 5 - 3: MAD vs. Warm up

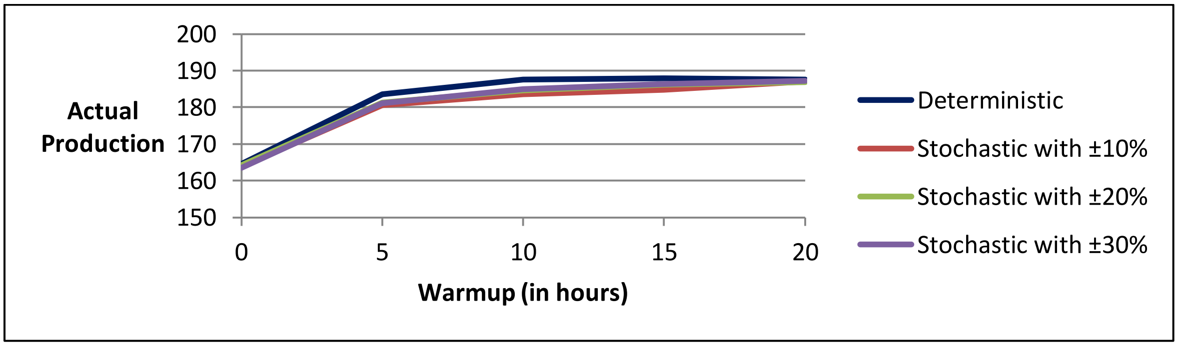

Figure 5 - 4: Actual Production vs. Warm up

5.3.1 No Buffer Policy

Using a warm-up period of ten hours, we apply policy A. The results are shown in Table 5 - 15.

Table 5 - 15: MAD, Actual Production and Total Movement for policy A

Type of System	MAD			Actual Production			Total Movement		
	Average	HW	SD	Average	HW	SD	Average	HW	SD
Deterministic	1.384	0.248	0.601	178.080	1.045	2.532	545.800	3.830	9.278
Stochastic with ±10%	2.922	0.670	1.624	173.880	1.847	4.475	546.800	2.764	6.696
Stochastic with ±20%	2.502	0.618	1.498	174.520	1.450	3.513	541.880	4.163	10.084
Stochastic with ±30%	3.156	0.871	2.109	172.800	2.050	4.967	541.440	4.928	11.938

The results show that policy A is unable to meet the daily demand of 180 units for the deterministic and stochastic cases. The MAD is also high, indicating that the line is not balanced. This shows that the Worker - Task assignments from the static model cannot be used without constructing a simulation model to determine the optimal buffer sizes for each station.

5.3.2 Common Buffer Size

Using a warm-up period of 10 hours, policy B is applied to all four systems. To reduce the number of Tukey comparisons, we filter the data to include buffer sizes with a MAD no more than 1.5, actual production no less than 180 and total movement no more than 100. The mean, confidence limits and standard deviation of MAD, Actual Production and Total Movement for the filtered buffer sizes for each system, as well as the Tukey's test results is shown in Table 5 - 16 through Table 5 - 23.

Table 5 - 16: MAD, Actual Production and Total Movement after applying policy B to the Branched 6 Task Deterministic System

Buffer Size	Variable	Mean	Upper 95% CL for Mean	Lower 95% CL for Mean	Std Dev
Max10Min5	MAD	1.158	1.19	1.126	0.362
	Actual Production	180.402	180.542	180.262	1.594
	Total Movement	87.35	87.479	87.221	1.471
Max10Min6	MAD	1.203	1.256	1.15	0.601
	Actual Production	181.71	181.79	181.63	0.91
	Total Movement	106.54	106.777	106.303	2.693
Max4Min0	MAD	0.854	0.869	0.84	0.167
	Actual Production	180.746	180.817	180.675	0.809
	Total Movement	117.788	117.902	117.674	1.296
Max5Min0	MAD	0.961	0.985	0.937	0.275
	Actual Production	180.74	180.779	180.701	0.439
	Total Movement	92.464	92.561	92.367	1.11
Max7Min4	MAD	0.957	0.993	0.922	0.403
	Actual Production	181.442	181.589	181.295	1.676
	Total Movement	136.23	136.555	135.905	3.703
Max8Min2	MAD	1.443	1.484	1.402	0.465
	Actual Production	180.616	180.864	180.368	2.819
	Total Movement	77.904	77.986	77.822	0.936
Max8Min5	MAD	1.228	1.258	1.199	0.336
	Actual Production	182	.	.	0
	Total Movement	137.566	137.663	137.469	1.103
Max9Min0	MAD	1.365	1.405	1.324	0.458
	Actual Production	180.116	180.169	180.063	0.599
	Total Movement	60.622	60.665	60.579	0.485

Table 5 - 17: Results from Tukey's Test after applying policy B to the Branched 6 Task Deterministic System

Alpha	0.05	
Error Degrees of Freedom	3992	
Error Mean Square	0.035107	
Critical Value of Studentized Range	4.2886	
Minimum Significant Difference	0.0359	
Tukey Grouping	**Mean of Square Root of MAD**	**Buffer Size**
A	1.19007	Max8Min2
B	1.14231	Max9Min0
C	1.09776	Max8Min5
C	1.06507	Max10Min6
C	1.06223	Max10Min5
D	0.96937	Max5Min0
D	0.95609	Max7Min4
E	0.91995	Max4Min0

Table 5 - 18: MAD, Actual Production and Total Movement after applying policy B to the Branched 6 Task Stochastic ±10% System

Buffer Size	**Variable**	**Mean**	**Upper 95% CL for Mean**	**Lower 95% CL for Mean**	**Std Dev**
Max10Min5	MAD	1.188	1.234	1.143	0.516
	Actual Production	180.852	181.001	180.703	1.697
	Total Movement	87.802	87.937	87.667	1.536
Max5Min0	MAD	1.247	1.339	1.154	1.049
	Actual Production	181.068	181.288	180.848	2.501
	Total Movement	93.29	93.464	93.116	1.975
Max6Min0	MAD	1.283	1.333	1.232	0.572
	Actual Production	181.71	181.838	181.582	1.451
	Total Movement	81.156	81.283	81.029	1.444
Max7Min1	MAD	1.457	1.517	1.397	0.684
	Actual Production	180.284	180.458	180.11	1.98
	Total Movement	77.482	77.623	77.341	1.603
Max7Min4	MAD	1.384	1.446	1.321	0.71
	Actual Production	180.726	180.944	180.508	2.479
	Total Movement	136.446	136.795	136.097	3.971
Max9Min6	MAD	1.391	1.458	1.323	0.771
	Actual Production	181.046	181.285	180.807	2.721
	Total Movement	133.718	134.07	133.366	4.002

Table 5 - 19: Results from Tukey's Test after applying policy B to the Branched 6 Task Stochastic ±10% System

Alpha		0.05	
Error Degrees of Freedom		2994	
Error Mean Square		0.089995	
Critical Value of Studentized Range		4.03271	
Minimum Significant Difference		0.0541	
Tukey Grouping		**Mean of Square Root of MAD**	**BufferSize**
	A	1.1723	Max7Min1
B	A	1.13795	Max9Min6
B	A	1.13605	Max7Min4
B	C	1.10639	Max6Min0
D	C	1.06448	Max10Min5
D		1.0455	Max5Min0

Table 5 - 20: MAD, Actual Production and Total Movement after applying policy B to the Branched 6 Task Stochastic ±20% System

Buffer Size	Variable	Mean	Upper 95% CL for Mean	Lower 95% CL for Mean	Std Dev
Max10Min5	MAD	1.296	1.354	1.238	0.661
	Actual Production	180.91	181.092	180.736	2.023
	Total Movement	87.818	87.968	87.668	1.707
Max5Min0	MAD	1.454	1.567	1.341	1.289
	Actual Production	181.52	181.797	181.247	3.124
	Total Movement	93.604	93.789	93.419	2.11
Max6Min0	MAD	1.388	1.448	1.329	0.675
	Actual Production	181.7	181.863	181.545	1.811
	Total Movement	81.178	81.313	81.043	1.533
Max7Min4	MAD	1.354	1.419	1.29	0.733
	Actual Production	180.15	180.368	179.924	2.525
	Total Movement	135.89	136.256	135.532	4.115
Max9Min6	MAD	1.446	1.519	1.372	0.836
	Actual Production	180.5	180.751	180.249	2.851
	Total Movement	134.06	134.404	133.708	3.964

Table 5 - 21: Results from Tukey's Test after applying policy B to the Branched 6 Task Stochastic ±20% System

Alpha	0.05	
Error Degrees of Freedom	2495	
Error Mean Square	0.11205	
Critical Value of Studentized Range	3.86048	
Minimum Significant Difference	0.0578	
Tukey Grouping	**Mean of Square Root of MAD**	**Buffer Size**
A	1.15668	Max9Min6
A	1.14633	Max6Min0
A	1.12252	Max7Min4
A	1.11815	Max5Min0
A	1.10324	Max10Min5

Table 5 - 22: MAD, Actual Production and Total Movement after applying policy B to the Branched 6 Task Stochastic ±30% System

Buffer Size	Variable	Mean	Upper 95% CL for Mean	Lower 95% CL for Mean	Std Dev
Max10Min5	MAD	1.4	1.464	1.336	0.729
	Actual Production	180.96	181.154	180.762	2.228
	Total Movement	87.854	88.026	87.682	1.961
Max7Min3	MAD	1.259	1.316	1.203	0.641
	Actual Production	180.17	180.357	179.983	2.124
	Total Movement	107.34	107.56	107.128	2.455

Table 5 - 23: Results from Tukey's Test after applying policy B to the Branched 6 Task Stochastic ±30% System

Alpha		0.05
Error Degrees of Freedom		998
Error Mean Square		0.08231
Critical Value of Studentized Range		2.77517
Minimum Significant Difference		0.0356
Tukey Grouping	**Mean of Square Root of MAD**	**Buffer Size**
A	1.14711	Max10Min5
B	1.08596	Max7Min3

Using the filtered data and the results from Tukey's test, we can conclude that the best general policy for the branched six task example is 'Max10Min5', as it is able to meet the filter criteria for all four systems and is statistically better than other buffer sizes for some systems in terms of MAD.

Chapter 6

Conclusions and Future Work

6.1 Conclusion

In this research we have presented a variation a variation of the Simple Assembly Line Balancing Problem (SALBP) for an apparel production line to include worker cross training and movement. Our primary objective was to obtain the optimal assignment of workers to tasks to maximize workload smoothness for the line and satisfy demand/production for a given time period. We have compared scenarios where work sharing is used and not used and found that implementing work sharing greatly improves the workload smoothness of the line. Also, allowing workers to be idle (i.e., allowing them to perform other duties that are not directly related to production of the product) improves the workload smoothness as well.

We have also provided a comparison of the solutions generated by CPLEX and the Genetic Algorithm using a randomly generated problems as well as real industrial case. The results from Table 4 - 14 shows that the GA performs better than CPLEX when applied to larger instances of the linear model. For the linear six task problem, it is able to solve all cases easily except for the cases with 100% utilization and one Task assignment. For the linear 12 task problem it does as well as CPLEX and is able to find optimal solutions for all cases where a feasible solution exists. For the linear 24 task problem, it is able to find an optimal solution in most cases and good solutions in others. CPLEX on the other hand was unable to find any feasible solution for the linear 24 task problem in the three hour time limit. The comparison between computation times for the linear model is shown that the GA takes longer than CPLEX in cases where the optimal solution isn't zero.

For the branched six task problem, the GA performs as well as CPLEX and is able to solve all cases optimally. For the branched 12 task problem, CPLEX and the GA are able to find optimal solutions for all cases except those with 100% utilization and one task assignment. For the branched 24 task problem, the GA performs better than CPLEX and is able to find optimal solutions in all cases where the utilization is less than 100%. For cases where utilization is constrained to be equal to 100%, the GA is able to find better solutions than CPLEX. The comparison between computation times for the branched model shows that the computation times for the GA are greater for the branched model as compared to the linear model. The average computation time for the 24 task branched problems is about 450 seconds per replication, and it takes up to 900 seconds per replication for cases where the utilization is less than 100%. But in general, the computation time per replication for the GA is small enough to allow for more than 20 replications per case, which could yield better results. We recommend running the GA for 20 replications then choosing the best one.

We used the worker to task assignment generated by solving two of these static models which are based on standard times to build simulation models that account for ±10%, ±20% and ±30% variation in the Standard Operation Time based on a PERT distribution. We compare the effect of MAD, actual production and total worker movement by applying two types of policies on these systems – a policy with no buffers and a policy with common buffer sizes. We filter results to reduce the number of Tukey comparisons.

We conclude that applying a ‘no buffer’ policy to the linear model yields a low MAD and an actual production that is able to satisfy a demand of 180, but with a large value for total worker movement. To reduce this we applied a common buffer size policy. The optimal buffer size was found to be ‘Max6Min3’ and ‘Max7Min4’, as they are able to meet the filter criteria for all four systems and there is no statistical difference between these two buffer sizes.

Applying a ‘no buffer’ policy to the branched model does not work as well. It yields a higher MAD, an actual production that is less than 180 and a large value for total worker movement.

We apply a common buffer size policy works and the optimal buffer size for the branched 6 task example was found to be 'Max10Min5' as it is able to meet the filter criteria for all four systems.

6.2 Future Work

Future research would focus on developing a local search heuristic to hybridize the GA and improve its ability to find solutions for the cases with 100% utilization and one task assignment.

Another area of research would be to develop a statistical estimate for the lower bound of the objective function. A $100(1 - \alpha)$, $0 < \alpha < 1$, confidence interval for the optimal objective-function value can be calculated using a set of randomly sampled objective-function values corresponding to solutions of the problem, according to a probability distribution defined on the problems feasible region [83].

Simulation models could be built for larger instances of the problem to verify the methodology.

References

[1] "Assembly Line - History." [Online]. Available: http://science.jrank.org/pages/558/Assembly-Line-History.html. [Accessed: 10-Jun-2013].

[2] L. Armbruster, "Five Ways Ford Motor Company Changed the World," *Ford Magazine - June Issue*, 2011. [Online]. Available: http://www.at.ford.com/news/cn/Pages/Five Ways Ford Motor Company Changed the World.aspx. [Accessed: 01-Aug-2013].

[3] E. Falkenauer and A. Delchambre, "A genetic algorithm for bin packing and line balancing," *Robot. Autom. 1992. …*, 1992.

[4] J. C. Chen, C.-C. Chen, L.-H. Su, H.-B. Wu, and C.-J. Sun, "Assembly line balancing in garment industry," *Expert Syst. Appl.*, vol. 39, no. 11, pp. 10073–10081, Sep. 2012.

[5] Y. Kim, Y. Kim, and Y. Cho, "A heuristic-based genetic algorithm for workload smoothing in assembly lines," *Comput. Oper. Res.*, vol. 25, no. 2, pp. 99–111, 1998.

[6] C. Becker and A. Scholl, "A survey on problems and methods in generalized assembly line balancing," *Eur. J. Oper. Res.*, vol. 168, no. 3, pp. 694–715, Feb. 2006.

[7] D. N. Rao, "Single and Mixed-Model Assembly Line Balancing Methods for both Deterministic and Normally Distributed Work Element Times," Oregon State University, 1971.

[8] R. De Jong, "Designing Mixed - Model Assembly Lines : A literature Review," 2009.

[9] N. Vidic, "Developing Methods To Solve The Workforce Assignment Problem Considering Worker Heterogeneity And Learning And Forgetting," University of Pittsburgh, 2008.

[10] R. G. Askin and J. Chen, "Dynamic task assignment for throughput maximization with worksharing," *Eur. J. Oper. Res.*, vol. 168, no. 3, pp. 853–869, Feb. 2006.

[11] W. J. Hopp and M. P. Van Oyen, "Agile workforce evaluation: A framework for cross-training and coordination," *IIE Trans.*, 2004.

[12] S. Kursun and F. Kalaoglu, "Simulation of production line balancing in apparel manufacturing," *FIBRES Text. East. Eur.*, vol. 17, no. 4, pp. 68–71, 2009.

[13] W. K. Wong, P. Y. Mok, and S. Y. S. Leung, "Developing a genetic optimisation approach to balance an apparel assembly line," *Int. J. Adv. Manuf. Technol.*, vol. 28, no. 3–4, pp. 387–394, Apr. 2005.

[14] M. Salveson, "The assembly line balancing problem," *J. Ind. Eng.*, vol. 6, no. 3, pp. 18–25, 1955.

[15] I. Baybars, "A survey of exact algorithms for the simple assembly line balancing problem," *Manage. Sci.*, vol. 32, no. 8, pp. 909–932, 1986.

[16] N. Boysen, M. Fliedner, and A. Scholl, "A classification of assembly line balancing problems," *Eur. J. Oper. …*, vol. 183, no. 2, pp. 674–693, Dec. 2007.

[17] N. Boysen, M. Fliedner, and A. Scholl, "Sequencing mixed-model assembly lines: Survey, classification and model critique," *Eur. J. Oper. …*, vol. 192, no. 2, pp. 349–373, Jan. 2009.

[18] E. Bowman, "Assembly-line balancing by linear programming," *Oper. Res.*, vol. 8, no. 3, pp. 385–389, 1960.

[19] H. Gökcen and E. Erel, "Binary integer formulation for mixed-model assembly line balancing problem," *Comput. Ind. Eng.*, vol. 34, no. 2, 1998.

[20] S. Graves and B. Lamar, "An integer programming procedure for assembly system design problems," *Oper. Res.*, vol. 31, no. 3, pp. 522–545, 1983.

[21] M. Peeters and Z. Degraeve, "An linear programming based lower bound for the simple assembly line balancing problem," *Eur. J. Oper. Res.*, vol. 168, no. 3, pp. 716–731, Feb. 2006.

[22] T. L. Urban and W.-C. Chiang, "An optimal piecewise-linear program for the U-line balancing problem with stochastic task times," *Eur. J. Oper. Res.*, vol. 168, no. 3, pp. 771–782, Feb. 2006.

[23] R. Carraway, "A dynamic programming approach to stochastic assembly line balancing," *Manage. Sci.*, vol. 35, no. 4, pp. 459–471, 1989.

[24] M. Held, R. M. Karp, and R. Shareshian, "Assembly-Line Balancing-Dynamic Programming with Precedence Constraints," *Oper. Res.*, vol. 11, no. 3, pp. 442–459, 1963.

[25] M. I. Henig, "Extensions Of The Dynamic Programming Method in the Deterministic Assembly-Line Balancing Problems," *Comput. Oper. Res.*, vol. 13, no. 4, pp. 443–449, 1986.

[26] C.-Y. Lee and R. Uzsoy, "A new dynamic programming algorithm for the parallel machines total weighted completion time problem," *Oper. Res. Lett.*, vol. 11, no. 2, pp. 73–75, Mar. 1992.

[27] L. Schrage and K. R. Baker, "Dynamic Programming Solution of Sequencing Problems with Precedence Constraints," *Oper. Res.*, vol. 26, no. 3, pp. 444–449, 1978.

[28] I. Berger, J. Bourjolly, and G. Laporte, "Branch-and-bound algorithms for the multi-product assembly line balancing problem," *Eur. J. Oper. …*, vol. 58, pp. 215–222, 1992.

[29] Y. Bukchin and I. Rabinowitch, "A branch-and-bound based solution approach for the mixed-model assembly line-balancing problem for minimizing stations and task duplication costs," *Eur. J. Oper. Res.*, vol. 174, no. 1, pp. 492–508, Oct. 2006.

[30] S. B. Liu, K. M. Ng, and H. L. Ong, "Branch-and-bound algorithms for simple assembly line balancing problem," *Int. J. Adv. Manuf. Technol.*, vol. 36, no. 1–2, pp. 169–177, Nov. 2006.

[31] C. Miralles, J. P. García-Sabater, C. Andrés, and M. Cardós, "Branch and bound procedures for solving the Assembly Line Worker Assignment and Balancing Problem: Application to Sheltered Work centres for Disabled," *Discret. Appl. Math.*, vol. 156, no. 3, pp. 352–367, Feb. 2008.

[32] A. Tozkapan, Ö. Kırca, and C. Chung, "A branch and bound algorithm to minimize the total weighted flowtime for the two-stage assembly scheduling problem," *Comput. Oper. Res.*, vol. 30, pp. 309–320, 2003.

[33] W. Chiang, "The application of a tabu search metaheuristic to the assembly line balancing problem," *Ann. Oper. Res.*, vol. 77, pp. 209–227, 1998.

[34] S. D. Lapierre, A. Ruiz, and P. Soriano, "Balancing assembly lines with tabu search," *Eur. J. Oper. Res.*, vol. 168, no. 3, pp. 826–837, Feb. 2006.

[35] P. R. McMullen, "JIT sequencing for mixed-model assembly lines with setups using Tabu Search," *Prod. Plan. Control*, vol. 9, no. 5, pp. 504–510, Jan. 1998.

[36] R. Pastor, C. Andrés, a Duran, and M. Pérez, "Tabu search algorithms for an industrial multi-product and multi-objective assembly line balancing problem, with reduction of the task dispersion," *J. Oper. Res. Soc.*, vol. 53, no. 12, pp. 1317–1323, Dec. 2002.

[37] A. Scholl and S. Voß, "Simple assembly line balancing—Heuristic approaches," *J. Heuristics*, vol. 244, pp. 217–244, 1997.

[38] E. Erel, I. Sabuncuoglu, and B. a. Aksu, "Balancing of U-type assembly systems using simulated annealing," *Int. J. Prod. Res.*, vol. 39, no. 13, pp. 3003–3015, Jan. 2001.

[39] P. McMULLEN and G. Frazier, "A simulated annealing approach to mixed-model sequencing with multiple objectives on a just-in-time line," *IIE Trans.*, 2000.

[40] A. R. Mendes, A. L. Ramos, A. S. Simaria, and P. M. Vilarinho, "Combining heuristic procedures and simulation models for balancing a PC camera assembly line," *Comput. Ind. Eng.*, vol. 49, no. 3, pp. 413–431, Nov. 2005.

[41] G. Suresh and S. Sahu, "Stochastic assembly line balancing using simulated annealing," *Int. J. Prod. Res.*, no. August 2013, pp. 37–41, 1994.

[42] A. Baykasoglu and T. Dereli, "Two-sided assembly line balancing using an ant-colony-based heuristic," *Int. J. Adv. Manuf. Technol.*, vol. 36, no. 5–6, pp. 582–588, Nov. 2006.

[43] P. R. McMullen, "An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives," *Artif. Intell. Eng.*, vol. 15, no. 3, pp. 309–317, Jul. 2001.

[44] P. R. McMullen and P. Tarasewich, "Multi-objective assembly line balancing via a modified ant colony optimization technique," *Int. J. Prod. Res.*, vol. 44, no. 1, pp. 27–42, Jan. 2006.

[45] P. M. Vilarinho and A. S. Simaria, "ANTBAL: an ant colony optimization algorithm for balancing mixed-model assembly lines with parallel workstations," *Int. J. Prod. Res.*, vol. 44, no. 2, pp. 291–303, Jan. 2006.

[46] Q. Zheng, M. Li, Y. Li, and Q. Tang, "Station ant colony optimization for the type 2 assembly line balancing problem," *Int. J. Adv. Manuf. Technol.*, vol. 66, no. 9–12, pp. 1859–1870, Aug. 2012.

[47] P. Chutima and P. Chimklai, "Multi-objective two-sided mixed-model assembly line balancing using particle swarm optimisation with negative knowledge," *Comput. Ind. Eng.*, vol. 62, no. 1, pp. 39–55, Feb. 2012.

[48] J. Dou and X. Zhao, “A particle swarm optimization for assembly line design with station paralleling and equipment selection,” *2012 IEEE Int. Conf. Comput. Sci. Autom. Eng.*, pp. 468–472, May 2012.

[49] N. Hamta, S. M. T. Fatemi Ghomi, F. Jolai, and M. Akbarpour Shirazi, “A hybrid PSO algorithm for a multi-objective assembly line balancing problem with flexible operation times, sequence-dependent setup times and learning effect,” *Int. J. Prod. Econ.*, vol. 141, no. 1, pp. 99–111, Jan. 2013.

[50] A. Nearchou, “Maximizing production rate and workload smoothing in assembly lines using particle swarm optimization,” *Int. J. Prod. Econ.*, vol. 129, no. 2, pp. 242–250, Feb. 2011.

[51] D. I. Petropoulos and A. C. Nearchou, “A particle swarm optimization algorithm for balancing assembly lines,” *Assem. Autom.*, vol. 31, no. 2, pp. 118–129, 2011.

[52] S. Ayazi, A. N. Hajizadeh, M. E. Nooshabadi, H. reza Jalaie, and Y. M. Moradi, “Multi-objective assembly line balancing using genetic algorithm,” *... J. Ind. ...*, vol. 2, no. 4, pp. 863–872, Oct. 2011.

[53] K. C. C. Chan, P. C. L. Hui, K. W. Yeung, and F. S. F. Ng, “Handling the assembly line balancing problem in the clothing industry using a genetic algorithm,” *Int. J. Cloth. Sci. Technol.*, vol. 10, no. 1, pp. 21 – 37, 1998.

[54] Z. X. Guo, W. K. Wong, and S. Y. S. Leung, “A genetic-algorithm-based optimization model for solving the flexible assembly line balancing problem with work sharing and workstation revisiting,” *Syst. Man, ...*, vol. 38, no. 2, pp. 218–228, 2008.

[55] Y. Kim and Y. Kim, “Genetic algorithms for assembly line balancing with various objectives,” *Comput. Ind. Eng.*, 1996.

[56] S. Lee, L. Khoo, and X. Yin, “Optimising an assembly line through simulation augmented by genetic algorithms,” *Int. J. Adv. ...*, vol. 16, no. 3, pp. 220–228, 2000.

[57] M. Moreira, M. Ritt, A. Costa, and A. Chaves, “Simple heuristics for the assembly line worker assignment and balancing problem,” *J. heuristics*, vol. 18, no. 3, p. 21, Mar. 2012.

[58] A. Noorul Haq, K. Rengarajan, and J. Jayaprakash, “A hybrid genetic algorithm approach to mixed-model assembly line balancing,” *Int. J. Adv. Manuf. Technol.*, vol. 28, no. 3–4, pp. 337–341, Apr. 2005.

[59] N. M. Razali and J. Geraghty, “Biologically inspired genetic algorithm to minimize idle time of the assembly line balancing,” *2011 Third World Congr. Nat. Biol. Inspired Comput.*, pp. 105–110, 2011.

[60] A. Yolmeh and F. Kianfar, “An efficient hybrid genetic algorithm to solve assembly line balancing problem with sequence-dependent setup times,” *Comput. Ind. Eng.*, vol. 62, no. 4, pp. 936–945, May 2012.

[61] W. Zhang and M. Gen, “An efficient multiobjective genetic algorithm for mixed-model assembly line balancing problem considering demand ratio-based cycle time,” *J. Intell. Manuf.*, vol. 22, no. 3, pp. 367–378, Jul. 2011.

[62] S. Tonyhsieh, “Hybrid analytic and simulation models for assembly line design and production planning,” *Simul. Model. Pract. Theory*, vol. 10, no. 1–2, pp. 87–108, 2002.

[63] S. Bahadir, “Assembly Line Balancing in Garment Production by Simulation,” 2011.

[64] M. G. Guner and C. Unal, “Line Balancing in the Apparel Industry Using Simulation Techniques,” *Fibres Text. East. Eur.*, vol. 16, no. 2, pp. 75–78, 2008.

[65] C. Unal, S. Tunali, and M. Guner, “Evaluation of Alternative Line Configurations in Apparel Industry using Simulation,” *Text. Res. J.*, vol. 79, no. 10, pp. 908–916, Jun. 2009.

[66] F. Glover, “Future paths for integer programming and links to artificial intelligence,” *Comput. Oper. Res.*, vol. 13, no. 5, pp. 533–549, 1986.

[67] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, “Optimization by Simulated Annealing,” *Sci. New Ser.*, vol. 220, no. 4598, pp. 671–680, 1983.

[68] V. Černý, “Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm,” *J. Optim. Theory Appl.*, vol. 45, no. l, pp. 41–51, 1985.

[69] M. Dorigo, “Optimization, Learning and Natural Algorithms,” Politecnico di Milano, Milan, Italy, 1992.

[70] J. Kennedy and R. Eberhart, “Particle swarm optimization,” *Proc. ICNN’95 - Int. Conf. Neural Networks*, vol. 4, pp. 1942–1948, 1995.

[71] J. H. Holland, *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. Oxford, England: U Michigan Press, 1975, p. 183.

[72] F. Glover, J. Kelly, and M. Laguna, “Genetic algorithms and tabu search: hybrids for optimization,” *Comput. Oper. Res.*, vol. 22, no. 1, pp. 111–134, 1995.

[73] S. O. Tasan and S. Tunali, “A review of the current applications of genetic algorithms in assembly line balancing,” *J. Intell. Manuf.*, vol. 19, no. 1, pp. 49–69, Jul. 2007.

[74] S. Chaudhry* and W. Luo, “Application of genetic algorithms in production and operations management: a review,” *Int. J. Prod. …*, no. September 2013, pp. 37–41, 2005.

[75] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Second, Ex. Springer-Verlag, 1992, p. 340.

[76] C. Houck, J. A. Joines, and M. Kay, “Comparison of genetic algorithms, random restart and two-opt switching for solving large location-allocation problems,” *Comput. Oper. Res.*, vol. 23, no. 6, pp. 587–596, 1996.

[77] J. A. Joines, C. T. Culbreth, and R. E. King, “Manufacturing cell design : an integer programming model employing genetic algorithms,” no. November 2012, pp. 37–41, 2007.

[78] M. Safe, J. Carballido, I. Ponzoni, and N. Brignole, “On stopping criteria for genetic algorithms,” *Adv. Artif. …*, pp. 405–413, 2004.

[79] Z. Michalewicz and C. Janikow, “Handling Constraints in Genetic Algorithms.,” *ICGA*, 1991.

[80] G. Hinton and S. Nowlan, “How learning can guide evolution,” *Complex Syst.*, vol. 1, pp. 495–502, 1987.

[81] D. Whitley, V. Gordon, and K. Mathias, “Lamarckian evolution, the Baldwin effect and function optimization,” *... Probl. Solving from Nature—PPSN III*, 1994.

[82] C. Houck, J. A. Joines, M. Kay, and J. Wilson, “Empirical investigation of the benefits of partial lamarckianism,” *Evol. Comput.*, pp. 1–40, 1997.

[83] A. D. Wilson, R. E. King, and J. R. Wilson, “Case study on statistically estimating minimum makespan for flow line scheduling problems,” *Eur. J. Oper. Res.*, vol. 155, no. 2, pp. 439–454, Jun. 2004.

Appendices

Appendix A

Data for Problem Sets

This section provides the data for all the problem sets that were solved using CPLEX and the GA. Set 0 is the base problem set for a given problem. The standard processing times of other sets are randomly generated by using a uniform distribution with the smallest value and largest value of processing times of Set 0 as a lower bound and upper bound respectively. Linear precedence graphs have not been provided as they can be easily constructed for a given number of tasks.

A.1 Data for the Linear 6 Task and Branched 6 Task Problems

The data pertaining to precedence graphs, worker skills and movement time can be found in Section 3.4.

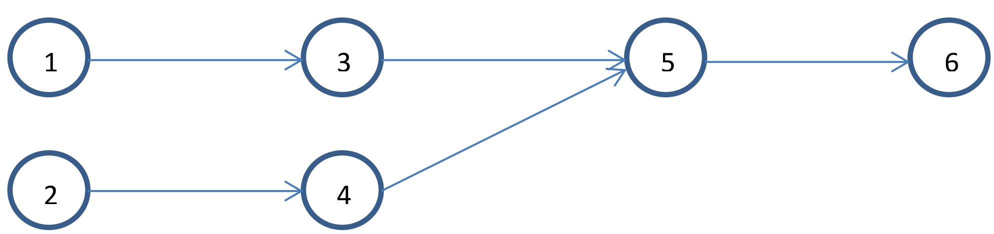

Figure A - 1: Precedence graph for Branched 6 Task Problem

Table A - 1: Production Line Constants for the Linear and Branched 6 Task Sets

Constant	Description	Value
N	Total number of tasks	6
W	Total number of workers	9
M	Total number of available machines	16
R_{min}	Threshold minimum for partial assignment	25%
T_{system}	Time available in system for given period	1440 min

Table A - 2: Standard Processing Times for the Linear 6 Task Sets

Task	Standard Processing Time				
	Set 0	Set 1	Set 2	Set 3	Set 4
1	5	3.736103	5.757647	8.040671	8.462388
2	6	7.687207	2.984148	6.221257	3.282523
3	8	6.031772	6.306232	4.586358	2.673315
4	5	7.02945	4.96022	3.395474	4.57235
5	3	8.067299	8.262537	5.37987	5.786011
6	7	4.486101	3.18774	3.816455	8.037727

Table A - 3: Standard Processing Times for the Branched 6 Task Sets

Task	Standard Processing Time				
	Set 0	Set 1	Set 2	Set 3	Set 4
1	5	6.733285	2.584106	7.675716	8.19734
2	6	5.700544	7.064342	7.24288	4.684112
3	8	5.977112	7.38694	4.741217	5.649211
4	5	4.237375	6.754227	8.271719	7.10267
5	3	4.311688	2.772117	7.728675	2.821027
6	7	7.148441	4.984196	2.837421	6.054749

Table A - 4: Worker Skills for the Linear and Branched 6 Task Sets

Worker	Task	Efficiency		Worker	Task	Efficiency
W1	T1	0.94		W5	T5	0.66
W1	T2	0.83		W6	T6	0.99
W2	T3	0.92		W6	T4	0.84
W2	T6	0.98		W7	T1	0.97
W3	T4	0.83		W7	T2	1
W3	T5	0.88		W8	T3	0.89
W4	T1	0.85		W8	T4	0.69
W4	T3	0.78		W9	T5	0.78
W5	T2	0.83		W9	T6	0.63

Table A - 5: Movement Time Matrix for the Linear and Branched 6 Task Sets (in minutes)

	T1	T2	T3	T4	T5	T6
T1	0	0.2	0.4	0.6	0.8	1
T2	0.2	0	0.2	0.4	0.6	0.8
T3	0.4	0.2	0	0.2	0.4	0.6
T4	0.6	0.4	0.2	0	0.2	0.4
T5	0.8	0.6	0.4	0.2	0	0.2
T6	1	0.8	0.6	0.4	0.2	0

A.2 Data for the Linear 12 Task and Branched 12 Task Problems

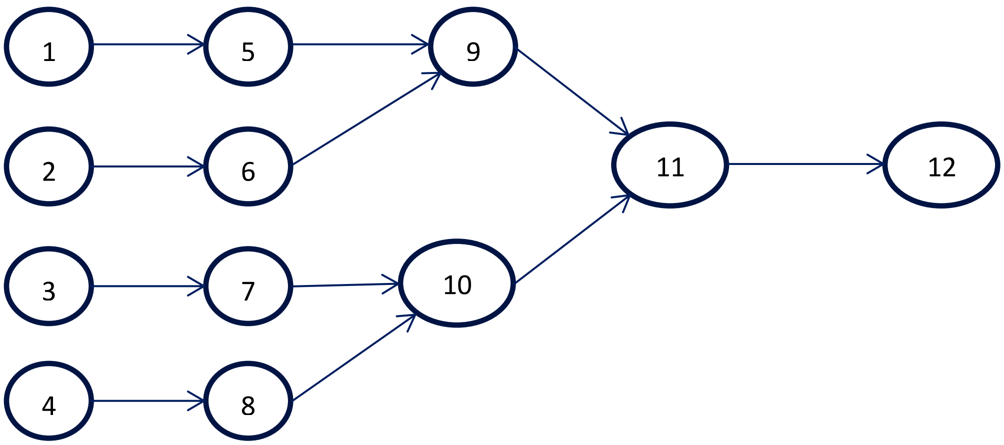

Figure A - 2: Precedence graph for Branched 12 Task Problem

Table A - 6: Production Line Constants for the Linear and Branched 12 Task Sets

Constant	**Description**	**Value**
N	Total number of tasks	12
W	Total number of workers	18
M	Total number of available machines	30
R_{min}	Threshold minimum for partial assignment	0.25
T_{system}	Time available in system for given period	1440 min

Table A - 7: Standard Processing Times for the Linear 12 Task Sets

Task	**Standard Processing Time**				
	Set 0	**Set 1**	**Set 2**	**Set 3**	**Set 4**
1	5	6.805971	5.883346	4.919981	7.411479
2	6	5.75137	6.845416	4.842935	8.75711
3	8	9.277964	9.017312	3.70249	3.79395
4	5	9.109804	6.827309	6.738569	8.962238
5	3	7.879421	5.920267	4.286094	4.120946
6	7	8.275161	6.572545	7.176234	10.1541
7	7	10.47417	4.322253	5.40767	3.026808
8	9	5.218449	7.453501	9.507797	2.992178
9	3	6.46169	6.418643	6.324113	8.845581
10	4	5.803742	7.946555	4.025067	5.536838
11	10	8.062254	9.592801	7.972498	6.208672
12	8	3.932714	5.464123	8.479326	3.456352

Table A - 8: Standard Processing Times for the Branched 12 Task Sets

Task	Standard Processing Time				
	Set 0	Set 1	Set 2	Set 3	Set 4
1	5	3.091627	6.133663	5.137887	4.553031
2	6	3.343621	5.356185	3.263433	3.219172
3	8	5.153555	3.699849	7.21834	2.747605
4	5	3.525999	8.135166	3.858988	5.081741
5	3	2.50193	9.930288	9.920932	8.821031
6	7	6.794353	6.7417	3.283439	4.878064
7	7	7.75644	3.217131	6.0509	4.382258
8	9	6.852112	8.561834	4.683575	6.343798
9	3	9.119295	5.714736	9.480375	4.536815
10	4	3.155149	6.194995	8.505502	5.224853
11	10	4.03538	6.437325	4.683539	2.859471
12	8	7.931307	4.161018	7.889174	6.359425

Table A - 9: Worker Skills for the Linear and Branched 12 Task Sets

Worker	Task	Efficiency		Worker	Task	Efficiency
W1	T1	0.6		W10	T7	1
W1	T2	0.98		W10	T8	0.62
W2	T3	0.75		W11	T9	0.76
W2	T4	0.79		W11	T10	0.71
W3	T5	0.85		W12	T11	0.6
W3	T6	0.91		W12	T12	0.73
W4	T7	0.72		W13	T1	0.96
W4	T8	0.96		W13	T2	0.95
W5	T9	0.9		W14	T3	0.94
W5	T10	0.88		W14	T4	0.88
W6	T11	0.88		W15	T5	0.65
W6	T12	0.86		W15	T6	0.96
W7	T1	0.99		W16	T7	0.9
W7	T2	0.75		W16	T8	0.63
W8	T3	0.76		W17	T9	0.69
W8	T4	0.84		W17	T10	0.92
W9	T5	0.78		W18	T11	0.82
W9	T6	0.66		W18	T12	0.76

Table A - 10: Movement Time Matrix for the Linear and Branched 12 Task Sets (in minutes)

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
T1	0	3	3	3	2	2	3	4	3	4	3	3
T2	3	0	2	4	4	3	3	4	3	4	2	3
T3	3	2	0	3	2	4	2	4	4	4	2	3
T4	3	4	3	0	4	4	3	4	2	2	4	2
T5	2	4	2	4	0	2	2	4	3	3	3	4
T6	2	3	4	4	2	0	2	2	2	4	2	3
T7	3	3	2	3	2	2	0	4	2	3	4	3
T8	4	4	4	4	4	2	4	0	2	3	4	2
T9	3	3	4	2	3	2	2	2	0	4	2	2
T10	4	4	4	2	3	4	3	3	4	0	4	4
T11	3	2	2	4	3	2	4	4	2	4	0	4
T12	3	3	3	2	4	3	3	2	2	4	4	0

A.3 Data for the Linear 24 Task and Branched 24 Task Problems

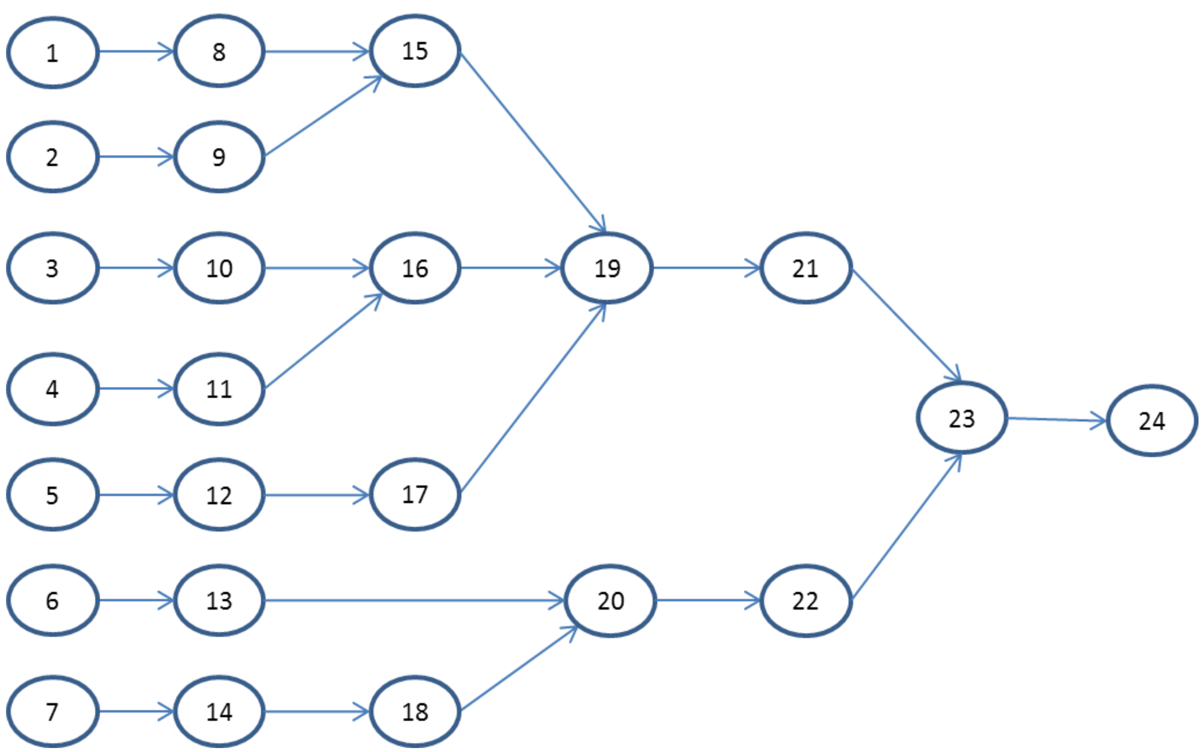

Figure A - 3: Precedence graph for Branched 24 Task Problem

Table A - 11: Production Line Constants for the Linear and Branched 24 Task Sets

Constant	Description	Value
N	Total number of tasks	24
W	Total number of workers	42
M	Total number of available machines	200
R_{min}	Threshold minimum for partial assignment	0.25
T_{system}	Time available in system for given period	1440 min

Table A - 12: Standard Processing Times for the Linear 24 Task Sets

Task	Standard Processing Time				
	Set 0	Set 1	Set 2	Set 3	Set 4
1	3	3.308291	7.641417	6.402507	3.36158
2	6	3.716414	4.440179	8.864429	9.179106
3	8	2.836832	4.996104	7.100795	8.081081
4	8	7.503714	5.5342	5.587803	7.373995
5	6	6.231156	9.114797	7.350962	5.311232
6	6	6.427065	3.35085	2.951438	2.614084
7	7	4.017135	7.021659	7.792648	3.67434
8	9	5.77604	4.934829	7.4036	3.649639
9	5	7.724483	3.231026	5.979497	6.06804
10	7	7.766176	3.797818	3.58805	5.342439
11	4	5.292526	3.043793	4.066163	3.242936
12	6	8.821692	5.529317	4.783008	4.432863
13	7	7.722048	9.214896	7.99142	7.001328
14	8	3.119917	6.294205	2.85184	8.4437
15	7	6.941998	5.960637	6.126111	5.985784
16	7	7.491196	9.310767	7.799207	3.814426
17	3	2.609777	4.024599	8.104689	8.776316
18	6	5.51801	5.152939	4.776438	5.1097
19	7	5.313878	5.27091	9.309103	4.764884
20	3	4.427068	4.470524	8.12987	7.895829
21	4	9.397617	6.023598	7.223783	4.026002
22	8	8.118112	3.471126	8.835905	5.628953
23	7	7.372705	6.120791	8.630458	4.152194
24	4	5.423194	9.257551	5.416539	8.648692

Table A - 13: Standard Processing Times for the Branched 24 Task Sets

Task	Standard Processing Time				
	Set 0	Set 1	Set 2	Set 3	Set 4
1	3	8.308611	9.358806	5.499195	3.589642
2	6	8.272215	5.30962	3.185817	5.821214
3	8	6.624141	4.44796	6.427281	4.300874
4	8	9.402652	3.623091	7.361397	6.901263
5	6	8.87675	3.639751	8.896023	6.294491
6	6	4.088062	7.02611	8.34372	3.594115
7	7	7.365809	5.370513	2.658404	9.069816
8	9	9.360023	5.389368	6.303524	7.081496
9	5	4.207519	7.489113	8.913148	6.042612
10	7	6.237112	4.783443	5.511828	5.2333
11	4	3.244588	6.932252	7.245634	3.251627
12	6	9.495902	3.952928	6.017177	7.987967
13	7	7.233231	3.802095	6.096162	5.717486
14	8	2.609927	6.583513	5.74086	7.775817
15	7	6.526287	3.065003	4.974309	6.672662
16	7	3.200366	5.7058	5.333839	8.329111
17	3	3.221158	8.840109	4.388121	2.631308
18	6	8.092191	4.329578	2.889154	3.97258
19	7	4.491362	7.996486	4.206916	3.017673
20	3	2.819544	5.152318	9.353546	3.238169
21	4	4.57041	4.527655	2.926414	4.821861
22	8	5.174075	8.93564	5.23204	3.397749
23	7	4.606793	6.922197	5.054968	2.501689
24	4	9.139998	6.893494	5.929263	6.257558

Table A - 14: Worker Skills for the Linear and Branched 24 Task Sets

Worker	Task	Efficiency		Worker	Task	Efficiency
W1	T1	0.92		W22	T13	0.73
W1	T2	0.77		W22	T14	0.85
W1	T3	0.82		W22	T15	0.71
W1	T4	0.91		W22	T16	0.74
W2	T5	0.72		W23	T17	0.91
W2	T6	0.74		W23	T18	0.86
W2	T7	0.72		W23	T19	0.95
W2	T8	0.87		W23	T20	0.9
W3	T9	0.71		W24	T21	0.92
W3	T10	0.83		W24	T22	0.82
W3	T11	0.71		W24	T23	0.93
W3	T12	0.99		W24	T24	0.84
W4	T13	0.94		W25	T1	0.82
W4	T14	0.88		W25	T2	0.84
W4	T15	0.88		W25	T3	0.85
W4	T16	0.88		W25	T4	0.92
W5	T17	0.95		W26	T5	0.84
W5	T18	0.76		W26	T6	0.81
W5	T19	0.99		W26	T7	0.76
W5	T20	0.89		W26	T8	0.99
W6	T21	0.99		W27	T9	0.72
W6	T22	0.98		W27	T10	0.98
W6	T23	0.78		W27	T11	0.93
W6	T24	0.88		W27	T12	0.77
W7	T1	0.73		W28	T13	0.87
W7	T2	0.73		W28	T14	0.73
W7	T3	0.75		W28	T15	0.96
W7	T4	0.79		W28	T16	0.9
W8	T5	0.86		W29	T17	0.91
W8	T6	1		W29	T18	0.95
W8	T7	0.99		W29	T19	0.73
W8	T8	0.74		W29	T20	0.86
W9	T9	0.91		W30	T21	0.83
W9	T10	1		W30	T22	0.91
W9	T11	0.77		W30	T23	0.95
W9	T12	0.72		W30	T24	0.93
W10	T13	0.93		W31	T1	0.76
W10	T14	0.78		W31	T2	0.91
W10	T15	0.76		W31	T3	0.71
W10	T16	0.97		W31	T4	0.83
W11	T17	0.92		W32	T5	0.87

Table A – 14 continued

W11	T18	0.96		W32	T6	0.94
W11	T19	0.98		W32	T7	0.93
W11	T20	0.75		W32	T8	0.86
W12	T21	0.73		W33	T9	0.82
W12	T22	0.89		W33	T10	0.98
W12	T23	0.98		W33	T11	0.95
W12	T24	0.85		W33	T12	0.88
W13	T1	1		W34	T13	0.76
W13	T2	0.74		W34	T14	0.73
W13	T3	0.97		W34	T15	0.9
W13	T4	0.91		W34	T16	0.83
W14	T5	0.72		W35	T17	0.92
W14	T6	0.71		W35	T18	1
W14	T7	0.77		W35	T19	0.97
W14	T8	0.76		W35	T20	0.78
W15	T9	0.89		W36	T21	0.81
W15	T10	0.94		W36	T22	0.8
W15	T11	0.73		W36	T23	0.93
W15	T12	0.71		W36	T24	0.76
W16	T13	1		W37	T1	0.95
W16	T14	0.98		W37	T2	0.95
W16	T15	0.8		W37	T3	0.96
W16	T16	0.72		W37	T4	0.97
W17	T17	0.7		W38	T5	0.87
W17	T18	0.77		W38	T6	0.86
W17	T19	0.7		W38	T7	0.78
W17	T20	0.76		W38	T8	0.79
W18	T21	0.91		W39	T9	0.9
W18	T22	0.8		W39	T10	0.74
W18	T23	0.94		W39	T11	0.93
W18	T24	0.81		W39	T12	0.81
W19	T1	0.95		W40	T13	0.82
W19	T2	1		W40	T14	0.76
W19	T3	0.83		W40	T15	0.86
W19	T4	0.79		W40	T16	0.91
W20	T5	0.79		W41	T17	0.9
W20	T6	0.88		W41	T18	0.96
W20	T7	0.93		W41	T19	0.88
W20	T8	0.92		W41	T20	0.78
W21	T9	0.93		W42	T21	0.7
W21	T10	0.7		W42	T22	0.84
W21	T11	0.85		W42	T23	0.8
W21	T12	0.87		W42	T24	0.85

Table A - 15: Movement Time Matrix for Linear and Branched 24 Task Sets (in minutes)

	T1	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**	**T11**	**T12**
T1	0	5	5	5	3	3	4	5	4	4	4	4
T2	5	0	2	5	3	3	5	4	5	2	5	5
T3	5	2	0	4	4	2	3	2	5	4	2	4
T4	5	5	4	0	2	4	3	5	3	5	3	2
T5	3	3	4	2	0	4	4	4	2	5	4	2
T6	3	3	2	4	4	0	2	4	4	5	4	2
T7	4	5	3	3	4	2	0	5	4	4	4	3
T8	5	4	2	5	4	4	5	0	3	5	4	2
T9	4	5	5	3	2	4	4	3	0	4	4	3
T10	4	2	4	5	5	5	4	5	4	0	4	2
T11	4	5	2	3	4	4	4	4	4	4	0	3
T12	4	5	4	2	2	2	3	2	3	2	3	0
T13	3	3	2	4	3	4	5	3	2	3	3	5
T14	2	5	2	2	2	3	2	2	2	5	2	5
T15	2	3	2	5	5	4	4	4	2	2	5	3
T16	3	3	4	4	2	4	5	3	3	5	4	5
T17	3	2	5	4	2	2	3	3	4	5	5	5
T18	4	2	5	3	2	2	3	2	4	2	4	3
T19	2	4	4	3	4	2	5	5	3	2	3	4
T20	5	5	5	4	2	3	2	2	2	5	5	5
T21	2	5	3	3	3	3	4	5	3	3	4	4
T22	5	5	3	5	2	5	5	4	3	4	2	2
T23	3	3	3	3	4	3	5	2	3	2	3	4
T24	5	5	3	4	5	3	4	2	3	3	4	5

Table A – 15 Continued

	T13	**T14**	**T15**	**T16**	**T17**	**T18**	**T19**	**T20**	**T21**	**T22**	**T23**	**T24**
T1	3	2	2	3	3	4	2	5	2	5	3	5
T2	3	5	3	3	2	2	4	5	5	5	3	5
T3	2	2	2	4	5	5	4	5	3	3	3	3
T4	4	2	5	4	4	3	3	4	3	5	3	4
T5	3	2	5	2	2	2	4	2	3	2	4	5
T6	4	3	4	4	2	2	2	3	3	5	3	3
T7	5	2	4	5	3	3	5	2	4	5	5	4
T8	3	2	4	3	3	2	5	2	5	4	2	2
T9	2	2	2	3	4	4	3	2	3	3	3	3
T10	3	5	2	5	5	2	2	5	3	4	2	3
T11	3	2	5	4	5	4	3	5	4	2	3	4
T12	5	5	3	5	5	3	4	5	4	2	4	5
T13	0	4	5	4	2	2	3	4	5	5	3	2
T14	4	0	5	2	4	2	5	4	3	2	3	3
T15	5	5	0	2	3	5	4	5	5	5	3	4
T16	4	2	2	0	4	3	2	4	2	2	3	4
T17	2	4	3	4	0	4	3	2	5	5	4	3
T18	2	2	5	3	4	0	3	4	5	5	4	4
T19	3	5	4	2	3	3	0	5	2	4	2	2
T20	4	4	5	4	2	4	5	0	3	2	2	4
T21	5	3	5	2	5	5	2	3	0	5	3	2
T22	5	2	5	2	5	5	4	2	5	0	5	3
T23	3	3	3	3	4	4	2	2	3	5	0	4
T24	2	3	4	4	3	4	2	4	2	3	4	0