DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

HYBRID META-HEURISTIC APPROACHES FOR SINGLE AND MULTI-OBJECTIVE BUFFER ALLOCATION PROBLEMS IN MANUFACTURING SYSTEMS

by
Simge YELKENCİ KÖSE

February, 2016
İZMİR

HYBRID META-HEURISTIC APPROACHES FOR SINGLE AND MULTI-OBJECTIVE BUFFER ALLOCATION PROBLEMS IN MANUFACTURING SYSTEMS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Industrial Engineering, Industrial Engineering Program

by
Simge YELKENCİ KÖSE

February, 2016
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"HYBRID META-HEURISTIC APPROACHES FOR SINGLE AND MULTI-OBJECTIVE BUFFER ALLOCATION PROBLEMS IN MANUFACTURING SYSTEMS"** completed by **SİMGE YELKENCİ KÖSE** under supervision of **ASSOC. PROF. DR. ÖZCAN KILINÇCI** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Assoc. Prof. Dr. Özcan KILINÇCI

Supervisor

Assoc. Prof. Dr. Gökalp YILDIZ

Thesis Committee Member

Asst. Prof. Dr. Umay KOÇER

Thesis Committee Member

Assoc. Prof. Dr. Deniz TÜRSEL ELİİYİ

Examining Committee Member

Prof. Dr. Osman Kulak

Examining Committee Member

Prof. Dr. Ayşe OKUR
Director
Graduate School of Natural and Applied Science

ACKNOWLEDGMENTS

First and foremost, I would like to thank my supervisor Assoc. Prof. Dr. Özcan Kılınççı, for his support, insights and encouragement. His intellectual guidance and enthusiastic dedication consistently motivated me throughout my Ph.D. study. I sincerely appreciate his valuable comments and time spent to assist and answer all my research questions.

I would like to express my sincere thanks to the other members of my dissertation committee, Assoc. Prof. Dr. Gökalp Yıldız and Assist. Prof. Dr. Umay Koçer for giving valuable suggestions and sharing their valuable knowledge throughout this Ph.D. study. I would also like to extend my gratitude to Prof. Dr. Osman Kulak and Assoc. Prof. Dr. Deniz Türsel Eliiyi for accepting to serve on my dissertation committee.

I would also like to express my deepest appreciation to Prof. Dr. Adil Baykasoğlu, for his continuous support during my doctoral education at Dokuz Eylul University.

I would also like to thank my dear friends, Dr. Leyla Demir, Dr. Hacer Güner Gören, Dr. Alper Hamzadayı and Dr. Şener Akpınar for providing continuous support whenever I need the most.

I would like to express my thanks to "The Scientific and Technological Research Council of Turkey (TUBITAK)", for providing financial suport within the scope of "National Scholarship Programme for Ph.D. Students" through my Ph.D. study.

The most, my gratefulness and hearty thanks are due to my beloved parents for their love, confidence, endless support and encouragement in my whole life. I would like to thank my mother, Kamuran Yelkenci and my father, Bülent Yelkenci very much. I feel extremely lucky to have such wonderful parents who have made many sacrifices and always give me strength to carry on to finish my thesis. I would also like to emphasize my thankfullnes to my lovely sister Fatoş Yelkenci Sert and her

husband Mehmet Sert because of their endless support and for being always with me whenever I needed. I would like to express my special thanks and deep appreciation to my husband Yasin Köse for his love, support and companion. He provided me with patience and courage and words are insufficient to express my deepest feelings of debts to him. He is not only a husband for me but also a great friend who listens all my complaints and continuously supports me.

Finally, I would like to state that this Ph.D. thesis is dedicated to my little cute son, EGE, who is the meaning of my life.

Simge YELKENCİ KÖSE

HYBRID META-HEURISTIC APPROACHES FOR SINGLE AND MULTI-OBJECTIVE BUFFER ALLOCATION PROBLEMS IN MANUFACTURING SYSTEMS

ABSTRACT

The buffer allocation problem is an NP-hard combinatorial optimization problem involving the determination of the number of buffers in buffer locations required to increase the efficiency of a production line. Researchers in this field have proposed various optimization techniques to solve the problem for different types of production system configurations. The main purpose of this Ph.D study is to introduce efficient and robust hybrid solution approaches for both single and multi-objective buffer allocation problems in serial production lines. In order to solve single objective buffer allocation problem, a hybrid algorithm combining *Genetic Algorithm* with *Simulated Annealing Algorithm* -based simulation optimization approach is proposed to allocate of a certain amount of buffers among the buffer areas of a production line so as to maximize the production rate of the system. This approach involves the use of a generative tool and an evaluative tool. The hybrid algorithm is employed as a generative tool to create candidate buffer size configurations. As a performance evaluative tool, discrete event simulation modeling is used to obtain the average production rate of the line. Prior to testing the performance of the proposed hybrid approach, an experimental design study is conducted to identify the best values/scheme for the hybrid algorithm parameters. Moreover, using these best hybrid algorithm parameters, a comprehensive experimental study along with statistical analysis is carried out to investigate the power of the hybridization for various serial line configurations. In the second stage of this Ph.D thesis, multi-objective buffer allocation problem which has two conflicting objectives, i.e. production rate maximization and total buffer size minimization is considered. For this purpose, the generative method is based on an evolutionary algorithm in which *Elitist Non-dominated Sorting Genetic* Algorithm and a special version of a *Multi-objective Simulated Annealing Algorithm* are hybridizied. Following the experimental study to identify appropriate values for the

hybrid algorithm parameters, a comparative study is carried out to present effectiveness of the proposed hybrid approach on solving buffer allocation problems for various serial line configurations.

Keywords: Buffer allocation problem, combinatorial optimization, hybrid meta-heuristics, genetic algorithm, simulated annealing algorithm, simulation, production lines.

ÜRETİM SİSTEMLERİNDE TEK VE ÇOK AMAÇLI ARA STOK YERLEŞTİRME PROBLEMLERİ İÇİN HİBRİD META-SEZGİSEL YAKLAŞIMLAR

ÖZ

Ara stok yerleştirme problemi, bir üretim hattının etkinliğini arttırmak için gereken ara stok alanlarına stok miktarını belirlemeyi içeren NP-zor kombinatoryal optimizasyon problemidir. Bu alanda çalışan araştırmacılar, farklı tip üretim sistemi konfigürasyonları için problemin çözümüne yönelik çeşitli optimizasyon teknikleri önermişlerdir. Bu doktora çalışmasının esas amacı, seri üretim hatlarında tek amaçlı ve çok amaçlı ara stok yerleştirme problemleri için etkin ve güçlü hibrid çözüm yaklaşımları ortaya koymaktır. Tek amaçlı ara stok yerleştirme problemini çözmek üzere, üretim hattında ara stok alanlarında belirli miktarda stoğun dağıtımı için *Genetik Algoritma* ve *Tavlama Benzetimi Algoritmasını* birleştiren bir hibrid meta-sezgisel algoritma tabanlı simülasyon optimizasyon yaklaşımı önerilmiş ve bu doğrultuda üretim sisteminde üretim miktarının arttırılması sağlanmıştır. Bu yaklaşım, generatif ve değerlendirici yöntemlerin birlikte ele alınmasını içermektedir. Hibrid algoritma, ara stok miktarlarını gösteren konfigürasyonları oluşturmak için generatif yöntem olarak kullanılmıştır. Sistemin performansını değerlendirmek üzere ise ortalama üretim miktarını belirlemek için kesikli sistem simülasyon modellemeye başvurulmuştur. Önerilen hibrid yaklaşımın performansını test etmeden önce algoritma parametrelerinin en iyi değerlerini belirlemek amacıyla deneysel bir tasarım çalışması yapılmıştır. Belirlenen parametre değerlerini kullanarak farklı seri üretim hatlarında hibrid yaklaşımın gücünü araştırmak amacıyla istatistiksel analizi de içeren kapsamlı bir deneysel çalışma gerçekleştirilmiştir. Bu doktora çalışmasının ikinci kısmında ise üretim miktarı maksimizasyonu ve toplam ara stok miktarı minimizasyonu gibi birbiriyle çelişkili iki amacı optimize etmeyi amaçlayan çok amaçlı ara stok yerleştirme problemi ele alınmıştır. Bu amaçla, generatif yöntem olarak *Seçkinci Baskın Olmayan Sıralamaya Dayalı Genetik Algoritma-II* ile *Çok Amaçlı Tavlama Benzetimi Algoritmasının özel bir uyarlamasının* hibridlendiği evrimsel bir algoritma kullanılmaktadır. Hibrid

algoritma için en uygun parametre değerlerini belirlemek için yapılan deneysel çalışma sonrasında önerilen hibrid yaklaşımın ara stok yerleştirme problemlerinde etkinliğini göstermek amacıyla farklı seri hat konfigürasyonlarında karşılaştırmalı bir çalışma gerçekleştirilmiştir.

Anahtar kelimeler: Ara stok yerleştirme problemi, kombinatoryal optimizasyon, hibrid meta-sezgiseller, genetik algoritma, tavlama benzetimi algoritması, simülasyon, üretim hatları.

CONTENTS

Page

Ph.D. THESIS EXAMINATION RESULT FORM ii
ACKNOWLEDGMENTS iii
ABSTRACT v
ÖZ vii
LIST OF FIGURES xiii
LIST OF TABLES xv

CHAPTER ONE - INTRODUCTION 1

1.1 The Problem and Its Significance 2
1.2 Research Objectives 4
1.3 Research Methodology 5
1.4 Outline of the Thesis 8

CHAPTER TWO - BUFFER ALLOCATION PROBLEM 10

2.1 Introduction 10
2.2 The Buffer Allocation Problem 10
2.3 Solution Approaches for the BAP 14
2.3.1 General Design Rules for Buffer Allocation 16
2.3.2 Numerical Algorithms for Buffer Allocation 18
2.4 Literature Review 19
2.4.1 Research on Single Objective BAP 20
2.4.2 Research on Multi-Objective BAP 34
2.4.3 Findings and Research Gaps 38
2.5 Chapter Summary 39

CHAPTER THREE - BACKGROUND INFORMATION FOR SOLUTION METHODS: SIMULATION OPTIMIZATION, EVOLUTIONARY ALGORITHMS, SIMULATED ANNEALING AND HYBRIDIZATION 40

3.1 Introduction 40

3.2 Simulation Optimization 40
3.3 Evolutionary Algorithms 42
3.3.1 Genetic Algorithms 43
3.3.1.1 Encoding and Initialization Scheme 45
3.3.1.2 Fitness Evaluation and Selection Scheme 45
3.3.1.3 Crossover and Mutation 46
3.3.1.4 Replacement Scheme 47
3.3.1.5 Termination Criteria 47
3.3.2 NSGA-II 48
3.3.2.1 Non-dominated Sorting Criteria 50
3.3.2.2 Crowding-Distance 52
3.3.2.3 Elitism Scheme 53
3.4 Simulated Annealing Algorithm 53
3.4.1 Neighborhood Generation Mechanism 57
3.4.2 Cooling Schedule 57
3.4.2.1 Initial Temperature 57
3.4.2.2 Cooling Function 58
3.4.2.3 Iterations at Each Temperature 59
3.5 Hybridization of Meta-heuristics 59
3.5.1 Hybridization of Genetic Algorithm with Simulated Annealing Algorithm 60
3.6 Chapter Summary 63

CHAPTER FOUR - A HYBRID APPROACH FOR SINGLE OBJECTIVE BAP IN SERIAL PRODUCTION LINES 64

4.1 Introduction 64
4.2 Problem Statement 64
4.3 Proposed Hybrid Approach 65
4.3.1 The General Specifications of the Proposed Hybrid Approach 68
4.3.1.1 Encoding and Initialization Scheme 68
4.3.1.2 Evaluation and Selection Scheme 70
4.3.1.3 Crossover and Mutation 71

4.3.1.4 SA-based Replacement Scheme 73
4.3.1.5 Termination Criterion 74
4.4 Computational Study 74
4.4.1 GAA Parameter Setup 75
4.4.2 Comparative Experiments on Benchmark Problems 81
4.4.2.1 Small Production Lines 81
4.4.2.2 Medium Production Lines 84
4.4.2.3 Large Production Lines 84
4.4.2.4 Comparative Experiments under Different Serial Line Conditions 88
4.4.2.5 Comparative Experiments on Very Large Production Lines 108
4.5 Chapter Summary 110

CHAPTER FIVE - A HYBRID EVOLUTIONARY APPROACH FOR MULTI-OBJECTIVE BAP IN SERIAL PRODUCTION LINES 112

5.1 Introduction 112
5.2 Problem Statement 112
5.3 Proposed Multi-objective Hybrid Approach 113
5.3.1 The General Specifications of Proposed Multi-objective Hybrid Approach 116
5.3.1.1 Encoding and Initialization Scheme 116
5.3.1.2 Evaluation and Selection Scheme 116
5.3.1.3 Crossover and Mutation 120
5.3.1.4 MOSA-based Replacement Scheme 123
5.3.1.5 Termination Criterion 124
5.4 Computational Experiments 125
5.4.1 MOHEA Parameter Setup 125
5.4.2 Comparative Experiments under Different Serial Line Conditions 134
5.4.2.1 Comparison Criteria 135
5.4.2.2 Test Case 1: Unbalanced Unreliable Serial Line 136
5.4.2.3 Test Case 2: Balanced Unreliable Serial Line 140
5.4.2.4 Test Case 3: Balanced Reliable Serial Line with Upper Bound for Each Buffer 142

5.4.2.5 Test Case 4: Balanced Unreliable Serial Line with Upper Bound for Each Buffer 144
5.5 Chapter Summary 146

CHAPTER SIX – CONCLUSION 148

6.1 Summary of the Thesis 148
6.2 Contributions 150
6.3 Future Research Directions 152

REFERENCES 154

APPENDICES 173

LIST OF FIGURES

Page

Figure 1.1 A serial production line with buffers. 1
Figure 1.2 Functions of a buffer. 3
Figure 1.3 Structure of the proposed optimization approach 6
Figure 2.1 Classification of solution approaches for BAP 15
Figure 3.1 General process of simulation optimization 41
Figure 3.2 Genetic algorithm cycle 44
Figure 3.3 Basic selection process 46
Figure 3.4 NSGA-II algorithm 50
Figure 3.5 Non-domination scheme 51
Figure 3.6 Illustration of fronts for NSGA-II 51
Figure 3.7 Crowding-distance interpretation 52
Figure 3.8 Elitism scheme 53
Figure 3.9 Flowchart of a standard SA algorithm 56
Figure 3.10 Hybrid representations of GA with SA. 62
Figure 4.1 The control logic of the proposed hybrid GAA. 67
Figure 4.2 Integer-valued encoding representation 68
Figure 4.3 Initialization scheme for a buffer size configuration 69
Figure 4.4 Adopted crossover procedure. 72
Figure 4.5 Adopted mutation procedure. 73
Figure 4.6 Normal probability plot. 79
Figure 4.7 Main effects for the mean of average production rate 80
Figure 5.1 The control logic of the proposed MOHEA. 115
Figure 5.2 Crowding distance 118
Figure 5.3 Binary tournament selection scheme 120
Figure 5.4 Crossover procedure for a four-buffer configuration. 121
Figure 5.5 Mutation scheme. 122
Figure 5.6 Diversity metric 130
Figure 5.7 Normal probability plot. 132
Figure 5.8 Main effects plot. 133

Figure 5.9 Interaction plots. 134
Figure 5.10 Test case 1- Extrapolated fronts for each case..... 138
Figure 5.11 Test case 2 - Extrapolated fronts for each case..... 141
Figure 5.12 Test case 3- Extrapolated fronts for each case..... 143
Figure 5.13 Test case 4 - Extrapolated fronts for each case..... 145

LIST OF TABLES

Page

Table 2.1 Overview on buffer allocation literature: Single objective BAP 22
Table 2.2 Overview on buffer allocation literature: Multi-objective BAP 36
Table 4.1 Experimental factors and levels for GAA 76
Table 4.2 24 full factorial experimental layout 77
Table 4.3 Factorial Fit: Mean versus Population size, Pc, Pm, and cooling schedule 78
Table 4.4 Dataset for a five-workstation line. 82
Table 4.5 Comparative results for a five-workstation production line. 82
Table 4.6 Comparative results for a nine-workstation production line. 83
Table 4.7 Dataset for a 10-workstation line. 83
Table 4.8 Comparative results for a 10-workstation production line. 84
Table 4.9 Comparative results for a 20-workstation production line. 86
Table 4.10 Comparative results for a 40-workstation production line. 87
Table 4.11 Summary of the production line types assumpted and parameters varied within each production line type. 89
Table 4.12 Test case 1 - Results for a five-workstation balanced reliable production line 90
Table 4.13 Test case 1 - Results for a 10-workstation balanced reliable production line 90
Table 4.14 Test case 1 - Results for a 15-workstation balanced reliable production line 91
Table 4.15 Test case 2 - Results for a 10-workstation balanced reliable production line with boundary constraint 92
Table 4.16 Test case 2 - Results for a 15-workstation balanced reliable production line with boundary constraint 93
Table 4.17 Test case 3 - Results for an eight-workstation reliable production line with a single bottleneck of T=1.1 96
Table 4.18 Test case 3 - Results for an eight-workstation reliable production line with a single bottleneck of T=1.5 96

Table 4.19 Test case 3 - Results for an eight-workstation reliable production line with a single bottleneck of T=2 97
Table 4.20 Test case 4 - Results for an eight-workstation reliable production line with two symmetrically located bottlenecks (K=8) 98
Table 4.21 Test case 5 - Results for an eight-workstation reliable production line with two asymmetric bottlenecks (K=8) 98
Table 4.22 Test case 6 - Results for five-workstation balanced unreliable production line with failures 100
Table 4.23 Test case 6 - Results for 10-workstation balanced unreliable production line with failures 101
Table 4.24 Test case 6 - Results for 15-workstation balanced unreliable production line with failures 102
Table 4.25 Test case 7 - Results for 10-workstation balanced unreliable production line with boundary constraint 103
Table 4.26 Test case 7 - Results for 15-workstation balanced unreliable production line with boundary constraint 103
Table 4.27 Test case 8 - Results for five-workstation unreliable production line with a single bottleneck 105
Table 4.28 Test case 9 - Results for six-workstation unreliable production line with two symmetric bottlenecks (K=6) 106
Table 4.29 Test case 10 - Results for six-workstation unreliable production line with two asymmetric bottlenecks (K=6) 106
Table 4.30 Wilcoxon signed-rank test results (α=0.05). 107
Table 4.31 Average CPU times for each test case. 108
Table 4.32 Computational results for very large lines. 110
Table 5.1 Experimental factors and levels for MOHEA 127
Table 5.2 2^5 full factorial experimental layout 128
Table 5.3 Dataset for seven-workstation production line 129
Table 5.4 Factorial Fit: Mean versus Pop_size, P_c, P_m, η and termination scheme . 131
Table 5.5 Dataset for Test case-1. 137
Table 5.6 Comparative results for Test case-1. 139
Table 5.7 Dataset for Test case-2. 140

Table 5.8 Comparative results for Test case-2. 142
Table 5.9 Comparative results for Test case-3. 144
Table 5.10 Comparative results for Test case-4. 146

CHAPTER ONE
INTRODUCTION

Serial production lines are archetypical production system that consists of several workstations/machines in series, possibly with buffers in between. In these lines, material flows from outside the system to the first workstation, then to the first buffer area, and so forth until it reaches last station. In Figure 1.1, an example of a material flow in a serial line in which the rectangles represent workstations and the circles represent buffers, is shown. Each material must be processed on each workstation during a fixed amount of time that is called processing time. The processing time at a machine may be assumed to be deterministic or stochastic. In stochastic case, the mean service time and its coefficient of variation can be determined from the associated processing time distribution. Often the exponential distribution is used, resulting in a coefficient of variation of 1. In practice, it has been observed that the coefficient of variation is less than 1 and thus a strict exponential distribution of processing times is inappropriate. However, phase type distributions (e.g., Coxian distribution with two phases) can be used to accommodate situations where the coefficient of variation of service time is less than 1 while retaining the analytical benefits of the exponential distribution (Papadopoulos, O'Kelly, Vidalis, & Spinellis, 2009).

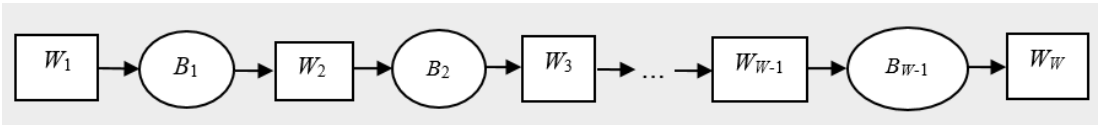


Figure 1.1 A serial production line with buffers (Vouros & Papadopoulos, 1998).

Performance of such a production line is affected by both variations in the processing times and the reliability parameters of the workstations. If the mean processing times are equal at all workstations in a production line, this line is balanced (or homogeneous) line. On the other hand, in unbalanced (non-homogeneous) lines, workstations can have different processing times, and one or more bottleneck workstations may exist, whose mean processing time is longer than other workstations. Moreover, production lines may be considered to be reliable or

unreliable lines. Reliable lines usually involve human operators who are not subject to breakdowns but whose processing times show significant variability (Harris & Powell, 1999). On the other hand, in unreliable lines, workstations operate with constant processing times and fail randomly, leading to breakdowns of significant duration. Unreliable workstations have an associated reliability or survival curve from which the mean time between failures (*MTBF*) may be determined. Failed workstations may be repaired in accordance with a repair time distribution from which the mean time to repair (*MTTR*) may be determined (Papadopoulos et al., 2009).

1.1 The Problem and Its Significance

In the design and operation of serial production lines, there are many issues affecting the capacity of lines such as policies for demand management, layout, process technology, bottleneck machines and etc. One of the main issues in the serial production lines is location and size of buffers in between workstations. Efficient management of buffer size allocation is a key concern in production systems. The basic structure of allocation involves a decision regarding how much buffer storage is available and where to place it within the production line to improve the production system. This problem is called the "*Buffer Allocation Problem* (BAP)" in the literature and is widely recognized by production systems designers. Buffers have a number of different functions in a production line as summarized in Figure 1.2. These functions are sometimes complementary to one another.

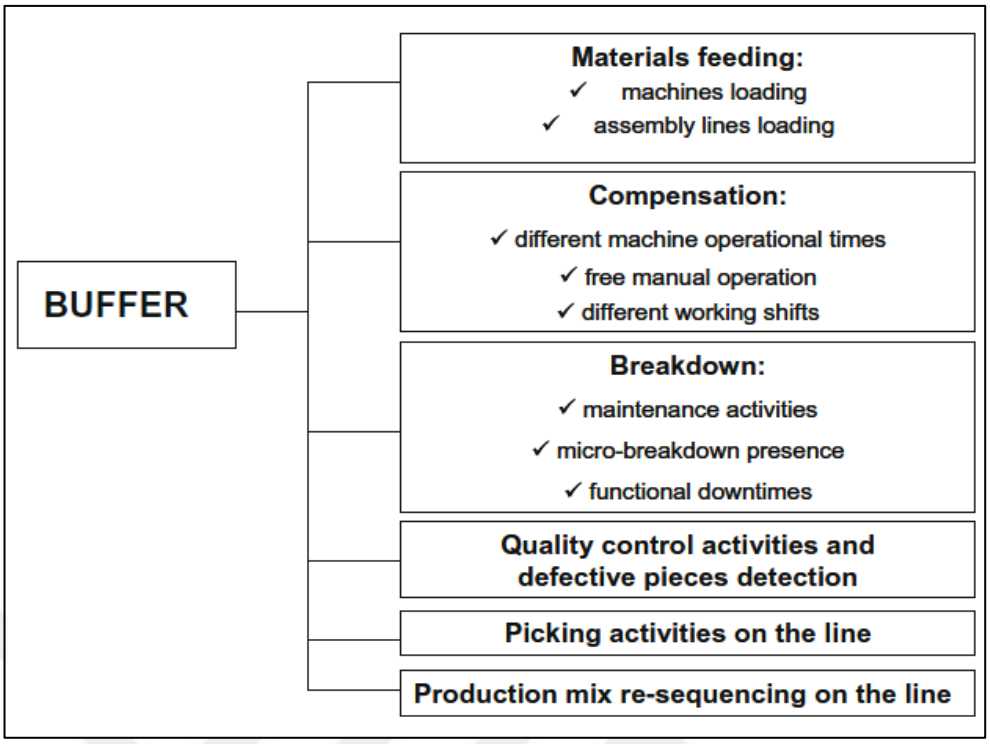


Figure 1.2 Functions of a buffer (Battini, Persona, & Regattieri, 2009).

Buffers can significantly improve the capacity of a line by reducing the effects of stochastic interference due to variations in processing times. Whenever the stations have a very specific function and a considerable time variation, each station needs to keep a greater degree of independence, so that its specific efficiency is not affected by the fluctuation in production of the precedent station (Battini et al., 2009). Lack of such independence might cause problems such as blocking and starvation in the production line. In case a station has completed its processing and the next buffer space is available, the processed part is passed on. Then, the station starts processing a new part that is taken from its input buffer. In case the buffer has no parts, the station remains empty (called that the station is starved) until a new part is placed in the buffer. Furthermore, the material flow may be disrupted by machine failures and repairs. For instance, when a machine breaks down in the production line, an up-stream machine could become blocked (when it cannot pass a processed job on to the next machine or buffer) and a down-stream machine could become starved (when no jobs are offered to this machine) (Alon, Kroese, Raviv, & Rubinstein, 2005). In this respect, using intermediate buffers help to reduce the starving and blocking time and

smooth the production flow by eliminating the disruptive effects so that the efficiency of the production line can be increased.

However, allocating buffers into a production line is costly and may be inadequate due to the floor space in the system. If the buffers are too large, both the direct costs through capital costs and indirect costs through increased work-in-process (WIP) inventory and the cycle times will outweigh the benefits of increased productivity. If the buffers are too small, the servers will be underutilized or the demand will not be met (Gershwin & Schor, 2000).

In this respect, the subject of this Ph.D. thesis is the Buffer Allocation Problem, which is concerned with the distribution of a certain amount of buffer among the intermediate buffer locations of a production line to achieve a specific objective function.

1.2 Research Objectives

Due the importance and complexity of the problem, optimal buffer allocation has been a major optimization problem faced by production systems designers. A large number of papers in literature dealing with the BAP show that this subject is still very active research area.

The purpose of this Ph.D. thesis is to introduce efficient solution approaches for BAPs to help production system designers in generating an effective buffer allocation plan.

In general, the BAP is formulated with respect to different optimality criteria. Therefore, this problem can be considered as single objective BAP in which attempts to minimize or maximize only one optimality criterion and multi-objective BAP which involves more than one objective function to be optimized simultaneously.

For single objective BAP, researchers are usually concerned with throughput (or average production rate) maximization. There are few other possible objective functions which may be of interest. These include the minimization of total buffer size, minimization of inventory holding cost, maximization of revenue, etc. In fact, there is a crucial trade-off between the objectives under consideration in the production systems. For instance, one would like to maximize average production rate and at the same time minimizing the total buffer size should be taken into consideration due to the cost factors. This necessitates a multi-objective procedure for the BAP.

Furthermore, in recent years, hybrid approaches based on meta-heuristics are proposed to achieve top performance in solving any production problem. Although many studies have focused on solving the BAP, the studies in which hybrid meta-heuristic approaches are used to solve the BAP are still very limited.

Within this perspective, in this study, initially hybrid meta-heuristics based simulation optimization approach is proposed to solve the single objective BAP and afterwards the problem is attempted to solve in a multi-objective manner with a novel hybrid evolutionary approach.

1.3 Research Methodology

Approaches to seeking the optimal buffer allocation can be based on design rules (or heuristics) or numerical algorithms for selecting optimal/near-optimal buffer allocations. In recent years, the numerical algorithms are mostly developed to optimize buffer sizes in production lines.

The main focus of this Ph.D. thesis has also been on constructing numerical algorithms which involve search (generative) methods and evaluative methods that are deployed in an iterative manner outlined of in Figure 1.3.

In this respect, search methods are employed to search for an optimal/near-optimal buffer configuration moving from an existing candidate solution. Dynamic programming, nonlinear programming, gradient-based search methods, Spendley–Hext-and Nelder–Mead-based simplex search algorithms, the degraded ceiling method, and meta-heuristics such as simulated annealing, tabu search, genetic algorithms, and ant colony optimization are examples of search methods employed for the BAP in serial production lines. Among these methods, metaheuristics are combined and referred to as hybrid meta-heuristics in this Ph.D. thesis for optimal/near-optimal buffer configuration generation.

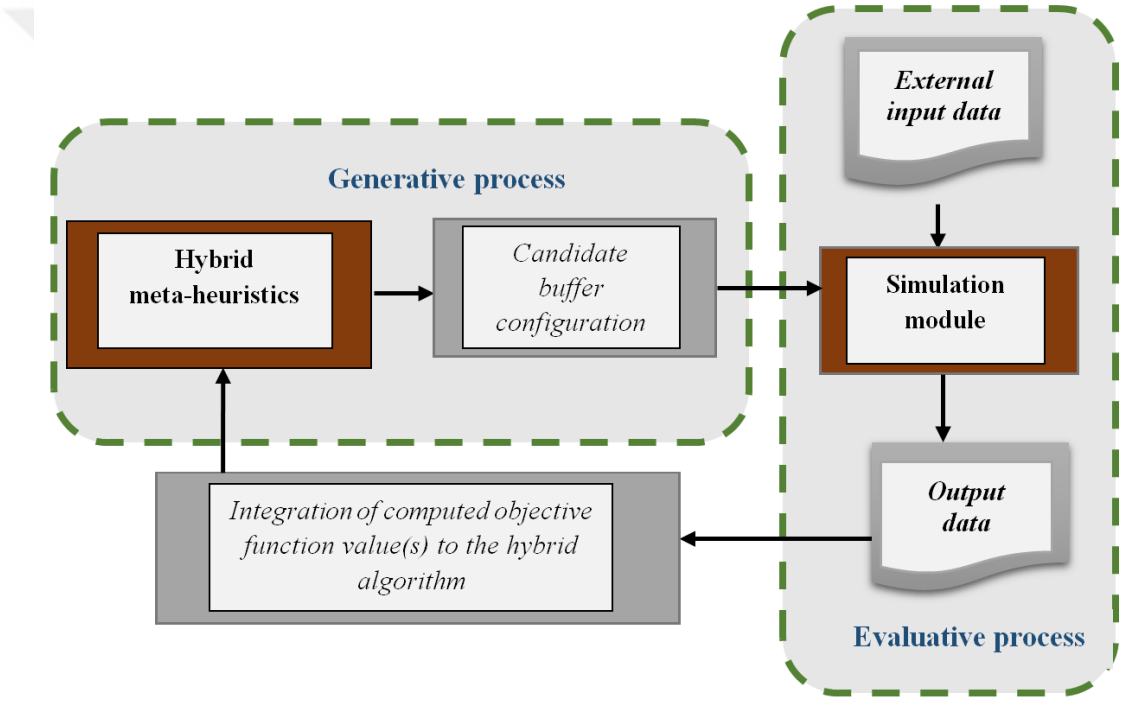


Figure 1.3 Structure of the proposed optimization approach

Through the optimization, the evaluative method is used for predicting the performance measures of a given system. In this Ph.D. thesis, a discrete event simulation modeling has been used to calculate the objective function of each candidate solution, evaluating at the same time the effect of buffer allocation decisions on the line throughput and average production rate. Hence, the candidate buffer configuration suggested by hybrid meta-heuristics after the generative process is used as an input in the simulation model as seen in Figure 1.3. The other set of

data provided as external input data to the simulation model include queue disciplines for buffers and workstations, blocking mechanism at each workstation, workstation failure (*MTBF*) and repair rates (*MTTR*), (if the production line consists of unreliable workstations), processing times at workstations, simulation time horizon and number of simulation runs. Using all these input data, the simulation model evaluates the fitness of a given buffer configuration.

It should be noted that the positions of buffers are strongly dependent on their function and production designers solve the BAP in accordance with the actual situation of buffers. For example, a buffer between two workstations might be considered to be in series or in parallel. The discipline for the serial buffer would normally be First-In, First-Out (FIFO), whereas the discipline for the parallel buffer would be Last-In, First-Out (LIFO). Where a station with parallel machines is concerned, the buffer discipline can be quite complex, in that an idle machine may not be in a position to service a waiting unit due to the materials handling protocol. So, the usual assumption of queueing theory that an idle server would immediately serve a waiting job may be violated (Papadopoulos et al., 2009). In this study, serial line designs in which the buffers are positioned in series are of interest, and hence, the FIFO discipline is implemented in all queues of buffers and workstations.

Blocking types in queueing networks with finite buffers are essential issue for BAP and it should be considered while modelling the existed production system by simulation. In the literature, Spinellis, Papadopoulos, & MacGregor Smith (2000) emphasized two blocking types that are summarized as below:

- TYPE-I (Blocking After Service (BAS)): The upstream node *i* gets blocked if the service on a customer is completed but it cannot move downstream due to the queue at the downstream node *j* being full. This blocking type is the most prevalent in manufacturing, transportation and other similar systems.

- TYPE-II (Blocking Before Service (BBS)): The upstream node is blocked when the downstream node becomes saturated and service must be suspended on the upstream customer regardless of whether service is completed or not.

In this Ph.D. thesis, the blocking mechanism is the TYPE-I (BAS) type at each workstation in all simulation models. As an output data, the computed objective function value, i.e. average production rate is obtained via the simulation module of the proposed optimization approach and after then, the output data is integrated with the hybrid algorithm for a new candidate buffer configuration generation.

1.4 Outline of the Thesis

This Ph.D. thesis which mainly focuses on optimal/near optimal buffer allocation in serial production lines using hybrid meta-heuristics-based approaches is organized as follows;

In Chapter 2, to gain a more comprehensive understanding of the problem studied in this Ph.D. thesis, the definition of BAP (i.e., characteristics, formulations and the solution methods employed in literature to solve this problem) in the serial production lines is given in detail. Additionally, to identify the current research issues, a comprehensive literature review is carried out with a structural framework based on single objective BAP and multi-objective BAP. In this respect, the research gaps on BAP are identified and the motivations are highlighted for this Ph.D. study.

In Chapter 3, background information on solution methods proposed in this Ph.D. study, namely evolutionary algorithms (EAs) and simulated annealing (SA) algorithm are explained and the hybridization concepts with a simulation optimization approach are described.

In Chapter 4, a hybrid approach by combining the key advantages of simulation, meta-heuristics, and hybridization is proposed to solve single objective BAP in serial production lines under the objective of average production rate maximization of the line. Prior to using the proposed hybrid approach, a pilot experiment is carried out to

identify the best values/scheme for the hybrid algorithm parameters. Following, using these best hybrid algorithm parameters, a comprehensive experimental study is provided including a set of benchmark problems published in the literature for small, medium and large production lines Additionally, comparative experiments under various serial line configurations such as balanced, unbalanced, reliable and unreliable lines and also on very large lines that consist of 50, 60, 80 and 100 workstations are carried out to demonstrate the efficacy of the proposed hybrid approach.

In Chapter 5, we mainly focus on solving multi-objective BAP and propose a hybrid evolutionary algorithm-based simulation optimization approach for serial production lines so as to optimize two conflicting objectives simultaneously that are average production rate maximization and total buffer size minimization. As in the case of solving single objective BAP, the best hybrid algorithm parameters are identified to improve the search ability of the proposed approach. Next, in order to investigate the performance of the proposed hybrid approach and the power of the hybridization, a comparative study for various serial line configurations is presented in this chapter.

Finally in Chapter 6, the summary and the contributions of this thesis study are discussed and the possible future research directions are given.

CHAPTER TWO
BUFFER ALLOCATION PROBLEM

2.1 Introduction

This chapter explains the buffer allocation problem (BAP) which is of particular interest to operations management for practical production line situations and provides a review of the related literature that supports the conceptual framework for this research. In this respect, the chapter is organized as follows. In the next section, the BAP in the production lines is defined in detail. Section 2.3 deals with the approaches for solving the BAP. Section 2.4 provides the studies in the BAP literature, published since 2005 under two categories: single objective BAP and multi-objective BAP. As a result of surveying the current literature, the findings and the research gaps are also stated in this section. Finally, the context of this chapter is summarized in Section 2.5.

2.2 The Buffer Allocation Problem

The BAP is mainly concerned with the allocation of a certain fixed number of buffer slots, K, among the $W-1$ buffer areas between the related workstations, W, in the production line in order to meet some specific objectives. In the line, material flows from outside the system to W_1, then to B_1, then to W_2, and so forth until it reaches W_w. The material flow may be disrupted by machine failures, repairs or variable processing times. For instance, when a machine breaks down in the production line, an up-stream machine could become blocked (when it cannot pass a processed job on to the next machine or buffer) and a down-stream machine could become starved (when no jobs are offered to this machine) (Alon et al., 2005). From this point of view, using intermediate buffers can help with smoothing the operation of the production systems by eliminating the disruptive effects so that the capacity of the production line can be improved. However, buffers are significant cost factor for an industrial company. Additionally, allocating buffers into a production line may be inadequate due to the limitations of a facility's location and there can be upper

bounds on the number of buffers allocated in the buffer locations. Thus, the BAP becomes a stochastic optimization problem with several variables involved.

The BAP is in itself a difficult NP-hard combinatorial optimization problem (Demir, Tunali, & Eliiyi, 2014; Dolgui, Eremeev, Kovalyov, & Sigaev, 2013; Huang, Chang, P-L, & Chou, 2002). There are two reasons why the BAP is a difficult optimization problem. The first reason is that there is no algebraic relation between the decision variables and the performance measures of the line; hence, it is much harder to solve the BAP. The second reason why the BAP is difficult is that finding the optimum value of the objective function, even if this function is completely known, comprises a combinatorial optimization problem over a potentially very large set with $\binom{K+W-2}{W-2}$ elements (Alon et al., 2005). Since the BAP belongs to the class of NP-hard combinatorial problems, finding optimal solutions by exact methods may require exponential computation time as the number of buffers and the number of workstations in the production system increase.

From a modeling point of view, the BAP is a stochastic, non-linear programming problem where the decision variables are integer. For the most general formulation of this problem, the decision variables may include the integer buffer sizes and the machine and conveyor speeds that result in a mixed-integer stochastic programming problem (Papadopoulos et al., 1993). In the literature, the problem related to the production line under investigation is formulated with respect to different optimality criteria. Dolgui, Eremeev, Kolokolov, & Sigaev (2002) summarized these criteria as follows:

- the average steady-state production rate, *f(B),* i.e. the average number of parts produced in time unit,
- the total buffer size, $B = B_1 + B_2 + B_3 + \ldots + B_{W-1}$,
- the average steady-state inventory cost, $Q(B)=c_1B_1+c_2B_2+\ldots+c_{W-1}B_{W-1}$, where B_i is the average steady-state number of parts in buffer i and c_i is the cost for each buffer size.
- different combinations of the above criteria.

Considering these optimality criteria, the formulation of the BAP can be expressed mathematically in different forms. The commonly used formulations for the BAP are explained in detail as below:

- **Formulation 1:** This formulation expresses the maximization of the throughput, given a certain fixed amount of buffers.

$$\text{Find } B = (B_1, B_2, B_3, \ldots, B_{W\text{-}1}) \text{ to} \tag{2.1}$$

$$\max f(B) \tag{2.2}$$

subject to

$$\sum_{i=1}^{W-1} B_i = K \tag{2.3}$$

$$Bi \text{ non-negative integers } (i = 1, 2, \ldots, W\text{-}1) \tag{2.4}$$

where W is the number of workstations in the line, K is a non-negative integer representing the total buffer size in the system allocated among the W-1 buffer locations, B represents a buffer vector, and $f(B)$ represents the average production rate of the system under a given buffer configuration.

- **Formulation 2:** This formulation expresses achieving the desired throughput rate with the minimum total buffer size.

$$\text{Find } B = (B_1, B_2, B_3, \ldots, B_{W\text{-}1}) \text{ to} \tag{2.5}$$

$$\min \sum_{i=1}^{W-1} B_i \tag{2.6}$$

subject to

$$f(B) \geq f^* \tag{2.7}$$

$$Bi \text{ non-negative integers } (i = 1, 2, \ldots, W\text{-}1) \tag{2.8}$$

where W is the number of workstations in the line, B represents the buffer vector, and $f(B)$ depicts the average production rate of the production line and f^* is the specified average production rate of the system.

- **Formulation 3:** This formulation describes the minimization of the average steady-state inventory cost subject to the total buffer size constraint.

$$\text{Find B} = (B_1, B_2, B_3, \ldots, B_{W-1}) \text{ to} \tag{2.9}$$

$$\min\ Q(B) = \sum_{i=1}^{W-1} c_i B_i \tag{2.10}$$

subject to

$$\sum_{i=1}^{W-1} B_i \leq K \tag{2.11}$$

$$Bi \text{ non-negative integers } (i = 1, 2, \ldots, W\text{-}1) \tag{2.12}$$

where B represents the buffer vector, c_i depicts the cost for each buffer location and K is a non-negative integer representing the total buffer size in the system allocated among the W-1 buffer locations so as to minimize the average steady-state inventory cost of the production system.

- **Formulation 4:** This formulation generally constructs the problem in a multi-objective manner. The objective function Z to be optimized depends on the different combinations of the optimality criteria. The general mathematical formulation of this multi-objective BAP can be basically stated as follows (Gross & Harris, 1985; MacGregor Smith & Daskalaki, 1988; Singh & MacGregor Smith, 1997; Yuzukirmizi & MacGregor Smith, 2008):

$$\text{Extremize } Z = \{f_1(\bar{x}), f_2(\bar{x}), \ldots f_p(\bar{x})\} \tag{2.13}$$

subject to

$$g_i(\bar{x}) \leq 0 \tag{2.14}$$

where

$f_1(\bar{x}) =$ Average production rate

$f_2(\bar{x}) =$ Total WIP

$f_3(\bar{x}) =$ Total cost

$f_4(\bar{x}) =$ Total revenue

$f_5(\bar{x})$ = Cycle time

$f_6(\bar{x})$ = Average queue length

. . .

$f_p(\bar{x})$ = Workstation utilization

$g_i(\bar{x})$ = Constraints on the decision variables

For a detailed analysis of mathematical models describing the effect of the buffer storage, the interested reader can refer to the following references (Buzacott & Shanthikumar, 1993; Papadopoulos, Heavey, & Browne, 1993; Papadopoulos et al., 2009). As a consequence of the specified formulations developed to solve BAPs, researchers widely adopt various solution approaches. Next section summarizes these approaches.

2.3 Solution Approaches for the BAP

The solution appraches for solving the BAP in production lines are classifed into the two main categories as indicated by Harris & Powell (1999). One of them is general design rules that can assist the decision maker in selecting good buffer configurations without undertaking complex analysis. Another approach is numerical algorithms which involve applying a generative (search) method and an evaluative method in a closed loop configuration. In this configuration, a generative method is used to search the solution space and determine optimal or near-optimal buffer sizes. To estimate the objective function value of each candidate buffer size configuration, an evaluative method is used and the objective function value is then communicated to the generative method. Clearly, this configuration basically consists of a loop process which leads to an optimal solution after a finite number of iterations.

The solution approaches are summarized in the Figure 2.1. They are discussed in the following subsections.

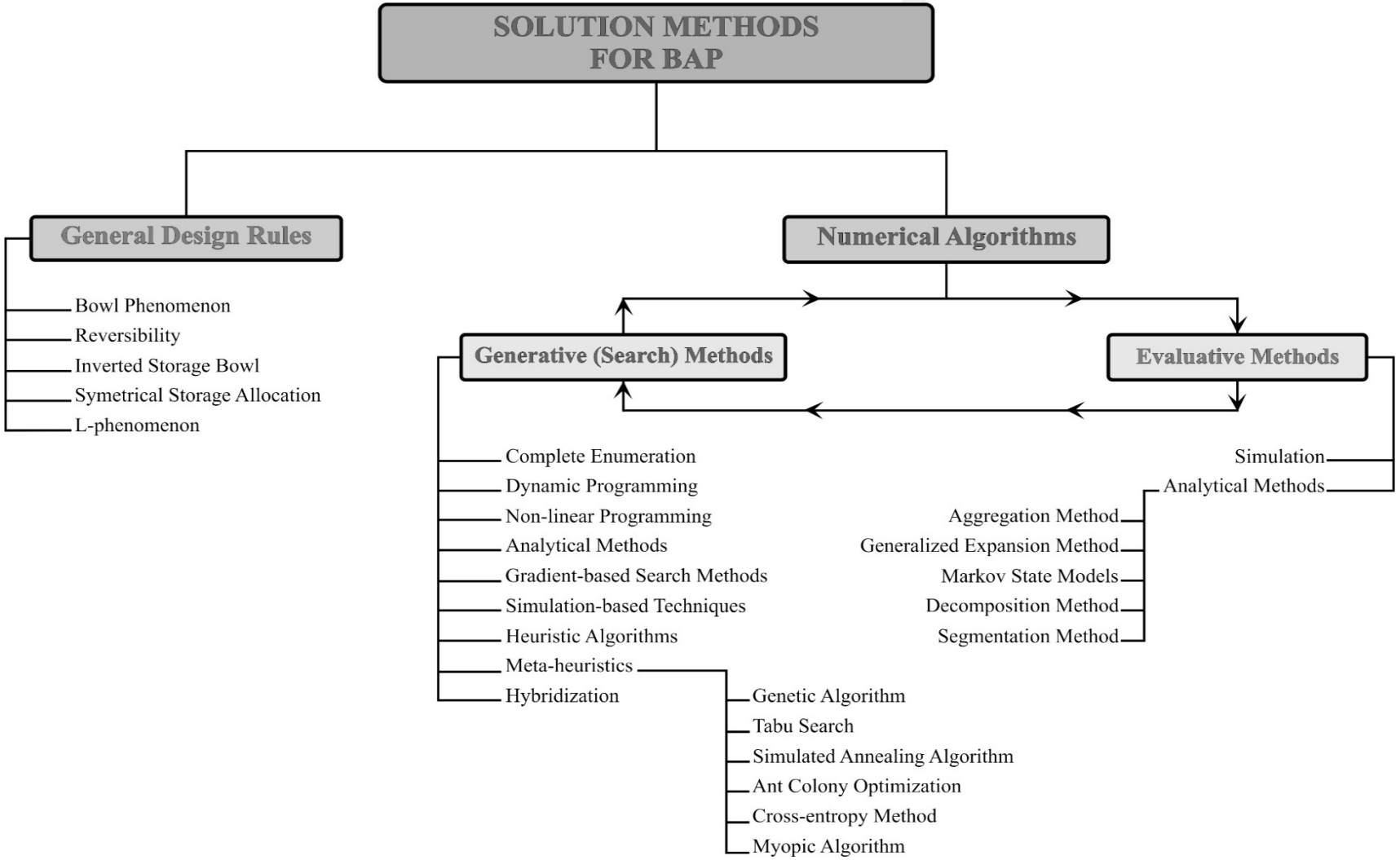


Figure 2.1 Classification of solution approaches for BAP.

2.3.1 General Design Rules for Buffer Allocation

A design rule, in general, holds true based on certain properties of the line (Staley, 2007). It has been developed after extensive computational experimentations for allocating the buffers through the line. In the relevant literature on BAP, the most prominent studies on the design rules include Freeman (1964), Conway, Maxwell, McClain, & Thomas (1988), Hillier & So (1991a, 1991b), Dallery & Gershwin (1992), Powell (1994), Powell & Pyke (1996) and Hillier (2000). In addition, Papadopoulos & Vidalis (2001), Enginarlar, Li, Meerkov, & Zhang (2002), Enginarlar, Li, & Meerkov (2005) and Staley & Kim (2012) have also made contribution to the relevant literature by identifying general buffer placement principles and design rules. Moreover, it has been observed that research related to the general design rules differs in the operating characteristics of the lines examined. General design rules that have proven out over time are explained as follows:

- ***Bowl phenomenon:*** Hillier & Boling (1966, 1977, 1979) proposed this rule for BAP. In this rule, the distributions of the completion times at the machines are identical in the production line. Assuming the shape of a bowl turned upside down; if possible, buffer capacity should be allocated evenly to all sites, with any remaining capacity allocated symmetrically from the center out. Otherwise, the first and last buffers should be relatively small and the buffers in the middle should be larger. The reason for this shape is that the machines in the middle of the line can be both blocked and starved. However, the first machines will rarely be starved and the last machines will rarely be blocked. For this reason, less buffer capacity is required at the first and the last production stages (Helber, 1999). For instance, a total of 50 buffers are allocated between the stations according to the pattern of $B = \{5, 12, 15, 13, 5\}$ for a six-workstation production line. This allocation depicts the bowl phenomenon as a design rule.

- ***Reversibility:*** Conway et al. (1988) considered a set of basic properties of lines with low to moderate coefficient of variation. They demonstrated that the same production rate is achieved with ‘‘mirror image’’ of buffer size configuration. For

example, for identical workstations only one of the configurations, B = {4, 3, 3} and B = {3, 3, 4}, needs to be considered since these configurations are mirror images (Conway et al., 1988).

- ***Storage bowl phenomenon***: This rule for allocating buffers is contributed to the literature by Conway et al., 1988; Hillier & Boling, 1966, 1967; Hillier, So, & Boling, 1993. This rule is the case for balanced lines that the production rate can often be improved by placing more buffer storage space toward the center of the line than at the end stations. For example, consider the problem of allocating a total of 50 buffers over the five buffer positions with a configuration of B = {10, 13, 14, 10, 3}. Hence, this allocation exhibits the storage bowl phenomenon as a design rule for a six-workstation production line.

- ***Symmetrical storage allocation:*** Buffer profiles that conform to symmetrical storage allocation (Conway et al., 1988) are those where the storage spaces are placed symmetrically in a line of identical machines. Conway et al., 1988 also indicated that the best buffer size configuration for a line has a slightly greater buffer capacity in the center. For instance, the buffer size configuration of B = {5, 4, 9, 4, 5} shows the symmetrical allocation pattern in six-workstation line.

- ***Freeman's downtime rule:*** Freeman (1964) studied on several design rules for the BAP literature. One of them is related with downtimes and it is a common rule for unreliable production lines. This rule indicates to allocate more buffer capacity to the station with the highest mean downtime. For instance, consider a five-workstation unreliable production line, denoted W_1-B_1-W_2-B_2-W_3-B_3-W_4-B_4-W_5. Among the workstations, fourth station has a higher mean failure rate and this results a higher mean downtime for this workstation. Regarding the Freeman's rule, allocating more buffers to the position B_3 will improve the capacity of line.

2.3.2 Numerical Algorithms for Buffer Allocation

For BAP, complete enumeration is the exact solution method for finding the optimal buffer sizes. However, the number of feasible allocations of K buffer slots among W-1 buffer locations increases dramatically with K and W. For instance, the number of possible buffer configurations for the Formulation 1 can be calculated as follows:

$$C_{K+W-2}^{W-2} = \frac{(K+1)(K+2)\ldots(K+W-2)}{(W-2)!} \tag{2.15}$$

Hence, complete enumeration is not an appropriate tool for solving a BAP in large-sized production lines. Numerical algorithms avoid complete enumeration or exhaustive search to explore the combinatorial solution space by intelligently moving from an initial buffer allocation toward the optimal solution, evaluating only potentially worthy candidate solutions (Vergara & Kim, 2009). These algorithms developed for the BAP involve generative (search) algorithms and evaluative algorithms that are deployed in an iterative manner (Papadopoulos et al., 2009; Spinellis & Papadopoulos, 2000a). In this respect, the evaluative method is used for predicting the performance measures of a given system. In the BAP literature, simulation (Altiparmak, Dengiz, & Bulgak, 2007; Battini et al., 2009; Bulgak, 2006; Can & Heavey, 2011, 2012; Harris & Powell, 1999; Jeong & Kim, 2000; Kose, 2010; Kose, Demir, Tunali, &Eliiyi, 2015; Lutz, Davis, & Sun, 1998; Sabuncuoglu, Erel, & Kok, 2002; Sabuncuoglu, Erel, & Gocgun, 2006; Vergara & Kim, 2009) and analytical methods such as the aggregation method (Diamantidis & Papadopoulos, 2004; Dolgui et al., 2002; Dolgui, Eremeev, & Sigaev, 2007), the generalized expansion method (Aksoy & Gupta, 2005; Spinellis et al., 2000), traditional Markov state models (Chararsoghi & Nahavandi, 2003), decomposition methods (Demir, Tunali, & Løkketangen, 2011; Demir, Tunali, & Eliiyi, 2012; Demir, Tunali, Eliiyi, & Løkketangen, 2013; Gershwin, 1987; Gershwin & Schor, 2000; Helber, 2001; Massim, Yalaoui, Amodeo, Chatelet, & Zeblah, 2010; Nourelfath, Nahas, & Ait-Kadi, 2005; Nahas, Ait-Kadi, & Nourelfath, 2006; Seong, Chang, & Hong, 2000; Shi & Men, 2003; Shi & Gershwin, 2009; Spinellis & Papadopoulos, 2000a, 2000b;

Tempelmeier, 2003) and the segmentation method (Shi & Gershwin, 2009) are used as evaluative tools.

As it is stated before, generative algorithms are employed to search for an optimal/near-optimal buffer configuration moving from an existing candidate solution. Dynamic programming (Diamantidis & Papadopoulos, 2004; Huang et al., 2002; Yamashita & Altiok, 1998), analytical methods (Enginarlar et al., 2005; Gershwin & Werner, 2007), nonlinear programming (Shi & Gershwin, 2009), search methods – gradient techniques (Gershwin & Schor, 2000; Helber, 2001; Seong et al., 2000), Powell's method (MacGregor Smith & Cruz, 2005; Van Woensel, Andriansyah, Cruz, MacGregor Smith, & Kerbache, 2010; Yuzukirmizi & MacGregor Smith, 2008), Spendley–Hext- and Nelder–Mead-based simplex search algorithms (Harris & Powell, 1999), the degraded ceiling method (Nahas et al., 2006), the Hooke and Jeeves search procedure (Altiok & Stidham, 1983), simulation-based techniques conjunction with perturbation or other approaches (Battini et al. 2009; Matta, Runchina, & Tolio, 2005), heuristic algorithms (Sabuncuoglu et al., 2006; Seong, Chang, & Hong, 1995), meta-heuristics such as simulated annealing (Kose et al., 2015; Spinellis & Papadopoulos, 2000a, 2000b; Spinellis et al., 2000), tabu search (Demir et al., 2011, 2012, 2013; Kose et al., 2015; Lutz et al., 1998), genetic algorithms (Dolgui et al., 2002, 2007; Kose et al., 2015; Spinellis & Papadopoulos, 2000b), ant colony optimization (Nourelfath et al., 2005), cross-entropy method (Alon et al., 2005) and myopic algorithms (Papadopoulos, O'Kelly, & Tsadiras, 2013), and hybrid approaches combining the meta-heuristics with nested partitions (Shi & Men, 2003), branch and bound methods (Dolgui et al., 2007) and line search (Amiri & Mohtashami, 2012) are examples of search methods employed for the BAP in production lines.

Regarding these methods developed for solving the BAP, the researches studied during the last 10 years, are explained in detail in the next section.

2.4 Literature Review

BAPs have been studied by many researchers since 1950s (Buzacott, 1967; Sevastyanov, 1962; Vladzievskii, 1950; 1951) because they contribute to the

improvement of the overall performance of a system. During the last 20 years, there has been even a growing interest in solving the BAP. In the literature, comprehensive survey studies are presented by Buzacott & Hanifin (1978), Dallery & Gershwin (1992), Papodopoulos & Heavey (1996), and Demir et al. (2014). In this review, we will summarize the studies dealing with BAP in the production lines published since 2005. The surveyed literature has been discussed in two sections. In Section 2.4.1, the studies considering BAP with single objective are reviewed in detail. Later, the multi-objective BAP studies are handled in Section 2.4.2.

2.4.1 Research on Single Objective BAP

There is an extensive bibliography in the area of the single objective BAP in the production lines**.** We reviewed the published literature based on;

- the topology of the production line (i.e. serial, parallel, general network, assembly, flexible/cellular manufacturing system),
- the assumption of production line (i.e. reliable, unreliable),
- the solution methodology as evaluative and generative methods,
- type of the objective function (i.e. throughput maximization, total buffer size minimization, profit maximization, service level maximization, total cost minimization, blocking type minimization, machine utilization maximization and idle time minimization),
- application environment.

Using the classification scheme, Table 2.1 chronologically lists the studies for single objective BAP published since 2005. For other studies published before 2005, the reader can refer to Gershwin & Schor (2000) and Demir et al. (2014).

A cellular manufacturing system with finite buffer capacities and unreliable servers was considered by Aksoy & Gupta (2005). In this study, the authors considered server unavailability in the remanufacturing system by means of exponential breakdown and repair rate and proposed a near optimal buffer allocation

algorithm that optimizes the buffer size so as to minimize the total cost and used expansion method to analyze the system. The performance of the algorithm was tested for both balanced and unbalanced cells in a variety of experimental conditions. Later, Aksoy & Gupta (2010) considered *N*-policy for the servers with finite buffers and the expected total cost of the remanufacturing system which is a function of the each station's throughput. Similar to their previous study, the authors employed expansion method to analyze the system and used heuristic method to find near optimal buffer allocation plans for a given number of total buffer sizes. To test the performance of the proposed algorithm, computational experiments were carried out for both balanced and unbalanced lines.

Alon et al. (2005) presented a stochastic algorithm as a generative method for solving the BAP, based on the cross-entropy method. As an evaluative method, simulation was used to estimate the throughput of the line. Another study which employed simulation as an evaluative method was presented by Matta et al. (2005). The authors presented some practical considerations to be adopted in real manufacturing flow lines.

Table 2.1 Overview on buffer allocation literature: Single objective BAP

Researcher(s)	Topology of the Production Line	Assumptions of Production Line	Solution Methodology		Objective(s)	Application Environment
			Generative	Evaluative		
Aksoy & Gupta (2005)	Cellular Manufacturing Systems	Unreliable	Heuristic algorithm	Expansion method	Min total cost	Hypothetic
Alon et al. (2005)	Serial	Unreliable	Cross-entropy method	Simulation	Max throughput	Hypothetic
Enginarlar et al. (2005)	Serial	Unreliable	Analytical method	Design rules	Min total buffer size	Hypothetic
Matta et al. (2005)	Serial	Unreliable	Design of experiments	Simulation	Max throughput	Real- Hypothetic
Nourelfath et al. (2005)	Serial	Unreliable	Ant colony optimization	Decomposition method	Max throughput	Hypothetic
Mac Gregor Smith & Cruz (2005)	General Network	Reliable	Powell's algorithm	Expansion method	Min total buffer size	Hypothetic
Bulgak (2006)	Assembly	Unreliable	Genetic algorithm	Simulation	Max throughput	Hypothetic
Hillier & Hillier (2006)	Serial	Reliable	Heuristic algorithm	Markov Chain model	Max profit	Hypothetic
Kwon (2006)	Flexible Manufacturing System	Reliable	Heuristic algorithm	Decomposition method	Max throughput	Hypothetic
Nahas et al. (2006)	Serial	Unreliable	Degraded ceiling method	Decomposition method	Max throughput	Hypothetic
Sabuncuoglu et al. (2006)	Serial	Reliable-Unreliable	Heuristic algorithm	Simulation	Max throughput	Hypothetic
Altiparmak et al. (2007)	Assembly	Unreliable	Artificial Neural Networks	Simulation	Max throughput	Hypothetic
Dolgui et al. (2007)	Serial-parallel	Unreliable	Hybrid method (Genetic algorithm – Branch and Bound)	Aggregation method	Max profit	Hypothetic
Gershwin & Werner (2007)	Serial	Unreliable	Analytical method	-	Max throughput	Hypothetic
Van Nieuwenhuyse et al. (2007)	Serial	Reliable	Mathematical model	Queueing model	Max service level	Real- Hypothetic
Qudeiri et al. (2007)	General Network	Unreliable	Genetic algorithm	Simulation	Max throughput	Hypothetic
Ribeiro et al. (2007)	Serial	Unreliable	Mixed integer linear programming	Mixed integer linear programming	Max profit	Hypothetic
Cruz et al. (2008)	General Network	Reliable	Lagrangian relaxation based algorithm	Expansion method	Min total buffer size	Hypothetic

Table 2.1 Overview on buffer allocation literature: Single objective BAP (cont).

Researcher(s)	Topology of the Production Line	Assumptions of Production Line	Solution Methodology		Objective(s)	Application Environment
			Generative	Evaluative		
Qudeiri et al. (2008)	Serial-parallel	Unreliable	Genetic algorithm	Aggregation method	Max throughput	Hypothetic
Yamamoto et al. (2008)	Flexible transfer line	Unreliable	Genetic algorithm	Simulation	Max throughput	Hypothetic
Battini et al. (2009)	Serial	Unreliable	Experimental Cross Matrix	Simulation	Min total buffer size	Hypothetic
Lee et al. (2009)	Serial	Unreliable	Genetic algorithm&Artificial intelligence based method	Simulation	Max throughput - Min total buffer size	Hypothetic
Nahas et al. (2009)	Serial-parallel	Unreliable	Ant colony optimization	Decomposition method	Max throughput	Hypothetic
Shi & Gershwin (2009)	Serial	Unreliable	Nonlinear Programming	Decomposition method	Max profit	Hypothetic
Vergara & Kim (2009)	Serial	Reliable-Unreliable	Heuristic algorithm	Simulation	Max throughput	Hypothetic
Aksoy & Gupta (2010)	Cellular Manufacturing Systems	Reliable	Heuristic algorithm	Expansion method	Min total cost	Hypothetic
Colledani et al. (2010)	Serial	Unreliable	Analytical method	Colledani et al. (2005)	Max throughput	Real
Kose (2010)	General Network	Unreliable	Genetic algorithm	Simulation	Max throughput	Real
Massim et al. (2010)	Serial	Unreliable	Artificial immune algorithm	Decomposition method	Max profit	Hypothetic
Van Woensel et al. (2010)	General Network	Reliable	Powell's Algorithm	Two-moment Aapproximation & Expansion method	Min total buffer size	Hypothetic
Can&Heavey (2011)	Serial	Reliable	Design of experiments	Genetic programming metamodels	Max throughput	Hypothetic
Demir et al. (2011)	Serial	Unreliable	Tabu search	Decomposition method	Max throughput - Min total buffer size	Hypothetic
Srinivas et al. (2011)	Flexible Manufacturing Systems	Reliable	Genetic algorithm	Simulation	Min total buffer size - Min blocking time - Max machine utilization	Hypothetic
Almomani et al. (2012)	Serial	Reliable	Heuristic algorithm	Simulation	Max throughput	Hypothetic

Table 2.1 Overview on buffer allocation literature: Single objective BAP (cont).

Researcher(s)	Topology of the Production Line	Assumptions of Production Line	Solution Methodology		Objective(s)	Application Environment
			Generative	Evaluative		
Demir et al. (2012)	Serial	Unreliable	Adaptive tabu search	Decomposition method	Max throughput	Hypothetic
Staley & Kim (2012)	Serial	Reliable-Unreliable	Design rules	Simulation	Max throughput	Hypothetic
McNamara et al. (2013)	Serial	Reliable	Design of experiments	Simulation	Max throughput - Min total buffer size - Min idle time	Hypothetic
Demir et al. (2013)	Serial	Unreliable	Integrated tabu search algorithm	Decomposition method	Min total buffer size	Hypothetic
Papadopoulos et al. (2013)	Serial	Reliable	Genetic algorithm, simulated annealing, myopic algorithms, tabu search, complete enumeration	Markovian method, Decomposition method	Max throughput	Hypothetic
Su & Xu (2014)	Hybrid manufacturing/remanufacturing system	Reliable	Heuristic algorithm	Improved decomposition-expansion algorithm	Min total cost	Hypothetic
Nahas (2014)	Serial	Unreliable	Extended great deluge algorithm	Decomposition method	Min total cost	Hypothetic
Shi & Gershwin (2014)	Serial	Unreliable	Shi & Gershwin (2009)	Segmentation method	Max profit	Hypothetic
Kose et al. (2015)	Serial - General Network	Unreliable	Genetic algorithm, simulated annealing, tabu search	Simulation	Min total buffer size	Real- Hypothetic
Tiacci (2015)	Assembly	Reliable	Genetic algorithm	Simulation	Max profit	Hypothetic
Costa et al. (2015)	Serial	Reliable	Parallel tabu search algorithm	Simulation	Min total buffer size	Hypothetic
Li et al. (2015)	Serial	Unreliable	Heuristic algorithm	Decomposition method	Max throughput	Hypothetic

A new quantitative characterization to the BAP named as *Lean Level of Buffering* was introduced by Enginarlar et al. (2005). This term, *Lean Level of Buffering*, refers the smallest, i.e. lean, normalized buffer capacity required to ensure the desired line efficiency. The authors presented several qualitative insights into the nature of lean buffering in serial production lines with identical exponential machines.

Nourelfath et al. (2005) developed an efficient heuristic approach based on ant system to solve the BAP. The objective was to maximize the efficiency of system subject to a total cost constraint. The authors aimed in this study to extend the classical BAP formulation to a more complicated problem where the decision variables are buffer and type of machine. Furthermore, the authors combined two improvement algorithms to improve the performance of the proposed algorithm. It has been noted that when combined these algorithms, proposed approach can always find near-optimal or optimal solutions quickly.

MacGregor Smith & Cruz (2005) solved the BAP for general finite buffer queueing networks in which they minimized buffer space cost under the production rate constraint. To evaluate the performance of the system, the generalized expansion method was employed and Powell's unconstrained search algorithm (Powell, 1964) was used to optimize the buffer sizes.

Hillier & Hillier (2006) investigated how the bowl phenomenon for workload allocation and the storage bowl phenomenon for buffer allocation interact when performing both allocations simultaneously. They used a basic cost-based model that includes both revenue per unit of throughput and cost per unit of buffer space in attempt to maximize the profit.

A new local search approach, namely degraded ceiling method was proposed by Nahas et al. (2006) for throughput maximization under total buffer space in unreliable production lines. An analytical decomposition-type approximation was used to estimate the production line throughput. In 2009, the same researchers studied both buffer allocation and machine selection problem in a serial-parallel

production line. Like their previous study, an analytical decomposition-type approximation was used to estimate the production line performance. The authors formulated the problem in which the decision variables were related to the buffer sizes and the number of parallel machines and made an optimization using ant colony optimization. In this study, a comparative study between the results obtained by ant colony optimization with the results obtained by simulated annealing algorithm was also presented. In a more recent study by Nahas (2014), the author proposed an extended great deluge algorithm to determine the optimal preventive maintenance policy and optimal buffer allocation that minimize the total system cost subject to a given system throughput level. Similarly to the previous study, the analytical decomposition-type approximation was used to estimate the throughput of the unreliable serial production line.

Sabuncuoglu et al. (2006) develop a heuristic procedure to allocate buffers in serial production lines to maximize throughput. The system studied in this study was unpaced and asynchronous serial production lines with single and multiple bottleneck stations under various experimental conditions.

A flexible manufacturing system with parallel workstations was considered by Kwon (2006). The author used the decomposition method to evaluate the performance of the system and proposed a heuristic algorithm to optimize the buffer sizes in attempt to maximize the throughput rate of the system. Numerical tests showed the efficiency of the proposed algorithm for small-sized problems.

Bulgak (2006) proposed simulation meta-modeling based on artificial neural networks to evaluate candidate buffer allocations in a genetic algorithm and applied to a split and merge assembly system. It has been noted that there is no statistical significant difference when compared to the genetic algorithm with simulation. But it is clear that computation times were much shorter. Altiparmak et al. (2007) further extended Bulgak's research into single close-loop asynchronous assembly system with three different machine failure occurrences: same failure rate, zero and non-zero failure rate, and low- and high failure rate, respectively. In this respect, the authors

developed an artificial neural network meta-model for the simulation model of an asynchronous unreliable assembly system. The authors stated that the artificial neural network meta-models could successfully be used for determining the most appropriate buffer combinations of the asynchronous assembly systems.

Dolgui et al. (2007) studied buffer space allocation problem of a tandem production line with unreliable machines. The authors proposed a hybrid optimization algorithm, combining the genetic algorithm and branch-and-bound approaches. For the evaluation of the fitness function which is formulated as a maximization function considering amortization time of the line, revenue for the sold production per time unit, buffers acquisition cost, Markov-model aggregation method was used.

Gershwin & Werner (2007) presented an approximate analytical decomposition method for evaluating the production rate and distribution of inventory of a closed-loop manufacturing system with unreliable machines. The authors stated that the proposed method could be applied to systems controlled by a CONWIP policy and tandem production lines with a limited number of pallets or fixtures.

In the same year, Van Nieuwenhuyse, Vandaele, Rajaram, & Karmarkar (2007) studied on the impact of management policies, such as product allocation and campaign sizing, on the required size of the finished goods inventories in a multi-product multi-reactor batch process. The authors developed a queueing model to estimate service levels as a function of the buffer size, and the allocation/campaign sizing policies.

Qudeiri, Yamamoto, Ramli, & Al-Momani (2007) proposed a production simulator based on genetic algorithm to determine optimal/near optimal buffer size for complex production systems. The authors also introduced a new encoding method, namely multi-vector encoding method for genetic algorithms. It was stated that using this new encoding method the optimal buffer sizes can be obtained after a few number of generations. In their next study (2008), the authors proposed a GA-

simulation-based method to find the nearest optimal design of a serial-parallel production lines that will maximize production efficiency. In this study, three decision variables: buffer size between each pair of work stations, machine numbers in each of the work stations, and machine types were considered for optimization. As an evaluative tool, aggregation method was used to estimate the production rate of the system. Similary to the study by Qudeiri et al. (2007), Yamamoto, Qudeiri, & Marui (2008) used the same production simulator system which consists of a genetic algorithm and a discrete simulator to decide a buffer size for flexible transfer line with bypass lines.

Ribeiro, Silveira, & Qassim (2007) considered jointly optimizing the maintenance of a capacity constrained resource, its feed machine/operation and inlet buffer size. To maximize total profit, a mixed integer linear programming model was developed in the planning horizon.

Cruz, Duarte, & Van Woensel (2008) studied the BAP in single-server general service time queueing networks. The objective of this study was to find the minimum total buffer size to achieve the desired average production rate. As a generative method, the authors proposed a Lagrangian relaxation based algorithm to optimize the buffer sizes. In order to estimate the performance of the line, they used the generalized expansion method.

Battini et al. (2009) used a simulation approach to study the relationship between the availability of machines and the size of buffer for a production system with micro-breakdowns. The authors aimed at limiting the micro-down-time problems. In turn, a new experimental cross matrix based on the guide index was developed as a generative method to determine the optimal buffer sizes. The also introduced a new parameter, guide index (G), in order to express the relationship between the maximum buffer size level and workstations availability parameters in input.

Lee, Chen, & Shunder Chang (2009) inspired by the Bulgak's research (2006) and used genetic algorithm in attempting to quickly figure out the best solutions. Through

the two-stage algorithm, the objective was to maximize the throughput of the line at first and minimize the total buffer subject to specified system throughput at second. In this reseach, artificial intelligence based method was employed to investigate the buffer allocation of unbalanced-unreliable flow type production lines.

Shi & Gershwin (2009) developed a nonlinear programming approach for transfer line profit maximization subject to a production rate constraint. In their model, both the buffer size cost and average inventory level cost were considered and the decomposition method was used to evaluate the production rate of the line. In a recent study, the same authors (2014) investigated the BAP in large production systems. They described a property which was the end effect that the optimal size of some of the buffers of a short segment of a long line were the same as those of the corresponding buffers in the long line. Using this property, a segmentation method was developed for profit rate maximization subject to a production rate constraint and like in their previous study (Shi & Gershwin, 2009), for optimization method they used nonlinear programming approach to find the optimal or near optimal buffer distribution of the line.

Vergara & Kim (2009) presented a buffer placement method for serial production line to analyze throuhput. Using simulation as an evaluative method, the throughput rate of the buffer allocations were estimated. The authors also applied a genetic algorithm for buffer placement to judge the effectiveness of their heuristic under different closed serial line conditions, *i.e.,* reliable, unreliable, balanced, unbalanced, and some bottleneck position conditions.

Colledani et al. (2010) proposed a new methodology based on analytical methods to solve BAP and repair crew optimization problem. The authors aimed at improving the performance of the real manufacturing system in Sweeden within the context of collaboration project so as to maximize throughput rate of the production system.

Another study implementing of the solution approach to a real manufacturing system was presented by Kose (2010). Specifically, the proposed approach employed

a two-phase simulation-genetic algorithm procedure. In the first phase, a detailed bottleneck analysis was carried out to identify what limited the capacity of the system by developing a discrete-event simulation model of the system. Following, the proposed approach was employed to allocate buffers to the machines so as to improve the performance of the manufacturing company founded in Turkey. Similarly to the previous study, Kose et al. (2015) aimed at achieving capacity improvements presenting a simulation optimization procedure in the same manufacturing company. The objective was obtaining the desired throughput rate with minimum total buffer size. For optimization, three metaheuristic-based search algorithms, i.e. a binary-genetic algorithm, a binary-simulated annealing algorithm and a binary-tabu search algorithm were proposed. These algorithms were integrated with the simulation model (as an evaluative method) of the production line.

More recently, Massim et al. (2010) implemented a combined artificial immune system optimization algorithm in conjunction with a decomposition method to optimally allocate buffers in transfer lines. In this study the immune decomposition algorithm was used to determine optimal buffer allocation for maximum line throughput and maximum line economic profit.

Van Woensel et al. (2010) studied both buffer and server allocation to optimize the number of buffers and servers. In this study, the Powell's algorithm (Powell, 1964) was employed as a generative method and a combination of two-moment approximation and the generalized expansion method was used as an evaluative method for the line analysis.

Can & Heavey (2011) presented an empirical analysis of experimental design approaches in simulation-based metamodelling of manufacturing systems with genetic programming. In this study, genetic programming metamodels were developed to predict throughput rates in serial production lines for a common industrial system.

Demir et al. (2011) proposed a tabu search algorithm to find near-optimal buffer allocation plans for a serial production line with unreliable machines. As an evaluative method, an analytical decomposition approximation method was used and the proposed algorithm was tested for solving BAP under the objective of maximizing throughput and minimizing total buffer size. In another study, Demir et al. (2012) developed an adaptive tabu search algorithm that included a new strategy to tune the parameters of tabu search adaptively during the search to find optimal buffer size configuration. The objective was to maximize the throughput of the line in which the capacity of each buffer space and the total buffer capacity to allocate to these spaces were as constraints. Moreover, the authors also experimented three initial solution schemes so as to decrease the search effort to obtain the best solutions. In a more recent study, Demir et al. (2013) presented an integrated approach in which nested loops employed to solve BAP for unreliable production lines to maximize the throughput rate of the line with minimum total buffer size. In the nested loops, two different search algorithms, binary search and tabu search were compared. The authors also suggested alternative neighborhood generation mechanisms to improve the efficiency of the proposed tabu search algorithm.

A flexible manufacturing system with with single and multi row was considered by Srinivas, Satyanarayana, Ramji, & Ravela (2011). The authors employed the genetic algorithm to find best buffer allocation plan and used simulaton to analyze the performance of the system considering minimizing the buffer size, minimizing the blocking and maximizing the machine utilization.

Almomani, Abdul Rahman, Baharum, & Alrefaei (2012) developed a selection approach which is a combination between cardinal and ordinal optimization to find the optimal allocation of buffers that maximizes the mean production rate in short, unbalanced and reliable production lines. As an evaluative method, simulation was used to evaluate the performance of each buffer allocation alternatives on the system.

Staley & Kim (2012) addressed buffer allocation design rules and results for closed serial production lines conducting a variety simulation experiments to better

understand the general properties of buffer allocation. The authors stated that buffer allocation decisions had more impact in closed reliable production lines than in closed unreliable production lines. Furthermore, optimal buffer allocations in closed lines were less sensitive to bottleneck severity than in open production lines.

McNamara, Shaaban, & Hudson (2013) investigated the performance of unpaced unreliable production lines considering buffer sizes by simulation method. The authors aimed at evaluating the effect that unbalancing buffers had on the efficiency of an unreliable production line which suffers from breakdown and consequent repair times. Using simulation, the specified production line was evaluated in terms of throughput, idle time, and average buffer level.

Papadopoulos et al. (2013) evaluated the effectiveness of the five search algorithms: simulated annealing algorithm, genetic algorithm, tabu search, myopic algorithm and complete enumeration if possible to solve BAP. Regarding the evaluation methods, a Markovian method and a decomposition method were used to calculate the throughput of the balance, reliable serial production lines. Moreover, the authors designed a decision support system, called *BAP DSS*, based on the comparative experiments to support the production line designer in making decisions regarding the distribution of the buffer slots associated with maximum or near maximum throughput of the algorithm.

Similarly to the study of Aksoy & Gupta (2010), another research on remanufacturing system considering *N*-policy was studied by Su and Xu (2014). The authors integrated service policy based on quality grade and priority and BAP upon the improved near optimal buffer allocation algorithm. As an evaluative tool, an improved decomposition-expansion algorithm was proposed to calculate the throughput for the system. The objective was to minimize remanufacturing cost by determining approximate optimal solution, and the total buffer capacity was set as constraint condition.

More recently, mixed model assembly line balancing problem with parallel workstations and stochastic task times was considered simultaneously with the BAP by Tiacci (2015). In an attempt to minimize total annual cost for labor, equipment and buffers costs of the line configuration, *Normalized Design Cost* function introduced by Tiacci, Saetta, & Martini (2006) was taken into consideration as an objective function. In this study, a genetic algorithm in which a parametric object oriented simulator was embedded in the genetic algorithm structure was employed to evaluate the throughput of the line.

Costa, Alfieri, Matta, & Fichera (2015) introduced a new meta-heuristic algorithm, namely parallel tabu search algorithm equipped with an adaptive neighborhood generation mechanism to solve the BAP, which consists of minimizing the total buffer capacity of a serial production system under a minimum throughput rate constraint. The proposed algorithm was compared to three real-encoding meta-heuristics such as genetic algorithm, differential evolution algorithm, particle swarm optimization derived from the relevant literature on the field of applied mathematics and computation. In this study, all the metaheuristic algorithms employ the same simulation-oriented evaluative procedure to measure the throughput rate of a given buffer configuration.

Finally, Li, Qian, Yang, & Du (2015) addressed the BAP to seek optimal buffer configurations in unreliable production lines and proposed a heuristic algorithm with the objective of maximizing production rate of the system. The main idea behind the algorithm was to decompose a long production line into a set of overlapping three-machine two-buffer systems. Moreover, the performance of the system was evaluated by the decomposition technique.

In the next section, we will be considering studies in which multi-objective procedures employed to solve BAP.

2.4.2 Research on Multi-Objective BAP

In BAP literature, it has been noted that there are few studies attempt to solve BAP in a multi-objective manner. Using the similar classification scheme for single objective BAP literature, Table 2.2 chronologically lists the studies for multi-objective BAP published since 2005. In this table, it should be pointed out that the objectives achived in these studies are the throughput maximization, the total buffer size minimization, the service rate minimization, the makespan minimization, the total production time minimization, the maintenance time maximization, and minimization of the total average waiting delays. Moreover, we reviewed the published literature based on the decision making strategy that one could use to solve the objective value for multi-objective optimization problems in general. Depending on how optimization and the decision process are combined, multi-objective optimization methods can be broadly classified into three categories (Horn, 1997; Hwang & Masud, 1979):

- **Decision making before search:** The objectives of the multi-objective procedure are aggregated into a single objective which implicitly includes preference information given by the decision maker.

- **Search before decision making:** Optimization is performed without any preference information given. The result of the search process is a set of (ideally Pareto-optimal) candidate solutions from which the final choice is made by the decision maker.

- **Decision making during search:** The decision maker can articulate preferences during the interactive optimization process. After each optimization step, a number of alternative trade-offs is presented on the basis of which the decision maker specifies further preference information, respectively guides the search.

Yuzukirmizi & MacGregor Smith (2008) presented an optimal buffer allocation procedure for closed queueing networks with finite buffers. They aimed at

maximizing the throughput rate of the system, minimizing total buffer size and also minimizing the total average waiting delay of a customer from entry to completion in the network at the same time. Using the cost values as weights, *i.e.,* profit from throughput, the cost of the cycle time, and the cost of a unit buffer due to the floor space and equipment costs as well as the WIP costs, the BAP was solved with a multi-objective procedure. As an evaluative method, expanded mean value analysis which used insights from the expansion method for modelling series, split and merge topologies was used and buffer sizes were optimized by utilizing Powell's method (Powell, 1964).

The first study employing evolutionary algorithm for solving multi-objective BAP was presented by Cruz, Van Woensel, & MacGregor Smith (2010). In this study, the authors developed a multi-objective approach for the buffer allocation and throughput trade-off problem for single server general queueing networks. The aim was both minimization of total number of buffer sizes and maximization of throughput rate. The authors used the generalized expansion method to evaluate the performance of the algorithm with a special version of multi-objective genetic algorithm, namely NSGA-II. Following, Cruz, Kendall, While, Duarte, & Brito (2012) employed the same multi-objective procedure to solve BAP considering minimization of the total number of buffer sizes, maximization of the throughput rate, and minimization of the overall server rate for general service queuing networks. Like the study of Cruz et al. (2010), the authors used the generalized expansion method to evaluate the performance of the algorithm and employed NSGA-II to produce a set of efficient buffer size configurations.

Table 2.2 Overview on buffer allocation literature: Multi-objective BAP

Researcher(s)	Topology of the Production Line	Assumptions of Production Line	Decision Making Strategy	Solution Methodology		Objective(s)	Application Environment
				Generative	Evaluative		
Yuzukirmizi & MacGregor Smith (2008)	General Network	Reliable	Decision making before search	Powell's algorithm	Expansion method	Max throughput - Min total buffer size - Min total average waiting delay	Hypothetic
Chehade et al. (2010)	Assembly	Unreliable	Search before decision making	Multi-objective ant colony optimization algorithm	Simulation	Max throughput - Min total buffer size	Hypothetic
Cruz et al. (2010)	General Network	Reliable	Search before decision making	Elitist non-dominated sorting genetic algorithm	Generalized expansion method	Max throughput - Min total buffer size	Hypothetic
Abdul-Kader et al. (2011)	Serial	Unreliable	Decision making during the search	Lexicographic goal programming	Nonlinear mathematical programming	Max throughput - Min total buffer size - Min makespan - Min total production time - Max maintenance time	Hypothetic
Amiri & Mohtashami (2012)	Serial-parallel	Unreliable	Search before decision making	Hybrid method (Genetic algorithm –Line search)	Simulation metamodelling	Max throughput - Min total buffer size	Hypothetic
Cruz et al. (2012)	General Network	Reliable	Search before decision making	Elitist non-dominated sorting genetic algorithm	Generalized expansion method	Max throughput - Min total buffer size - Min service rate	Hypothetic
Bekker (2013)	Serial, General Network	Unreliable	Search before decision making	Cross-entropy method	Simulation	Max throughput - Min total buffer size	Hypothetic

Chehade, Yalaoui, Amodeo, & Dugardin (2010) solved BAP in assembly lines with a multi-objective solution approach. For both throughput maximization and total buffer size minimization in the line, the authors proposed a multi-objective method based on a Lorenz multi-objective ant colony algorithm instead of the well-known Pareto dominance relationship. As an evaluative tool, discrete event simulation was used for the performances evaluations of the different tested buffer configurations. It has been stated stated that Lorenz dominance relationship provided a better domination area by rejecting the solutions founded on the extreme sides of the Pareto front.

Abdul-Kader, Ganjavi, & Baki (2011) addressed the problem of production output optimization in a highly automated multi-product production line. The authors explored the effects of buffer size on attenuating the impact of line blocking and starvation that can cause a reduction in the output. In this respect, an integer non-linear mathematical programming model which may have different objective functions in different scenarios including any of the five objective functions such as throughput maximization, total buffer size minimization, makespan minimization, total production time minimization, maintenance time maximization was developed to analyze the effects of buffer size on the system performance. For the multi-objective procedure, lexicographic optimization modelling was used with the priority orders. For instance, in one of the test cases of this study, the lexicographic goal program was solved in three phases. In phase 1, the throughput was maximized. In phase 2, the maximum throughput was added as a new constraint and the makespan was minimized. In phase 3, the optimal values of the throughput and makespan were added as additional constraints and the total buffer size was minimized.

Amiri & Mohtashami (2012) proposed a multi-objective formulation of the BAP in unreliable serial production lines. The authors employed a hybrid approach including genetic algorithm with line search method to determine some near optimal non-dominated solutions for buffer allocation. Considering general function distributions for all parameters of production lines and using a discrete event simulation model, a meta-model was built for estimating production rate. In a multi-

objective manner, production rate maximization and total buffer size minimization were taken into an LP metric objective function consideration.

Lastly, Bekker (2013) proposed cross-entropy method to optimize at least two conflicting objectives of the BAP, namely throughput rate and allocated buffer size. Using simulation, small to large stochastic queueing networks of unreliable resource were analyzed to evaluate the performance measures. The authors stated that the cross-entropy method as a multi-objective optimizer can minimize the total buffer size by appropriately allocating space to each buffer while maximizing throughput rate.

2.4.3 Findings and Research Gaps

Studies on single objective and multi-objective BAP in the production lines are given in previous subsections. Referring to Table 2.1 and Table 2.2 in the previous sections, we could list the findings of this survey and state the research gaps as follows:

- A great majority of studies solve the BAP with a single-objective procedure (*i.e.,* 46 out of 54 studies).
- The existing literature about the BAP has extensively dealt with serial lines with unreliable machines. About the 50% of studies analyzed the serial line configuration (*i.e.,* 30 out of 54 studies). 35 out of 54 studies consider the issue of machine failures and repairs.
- In only 5 out of 54 studies, the solution approaches to solve BAP are implemented to a real manufacturing system.
- It must be highlighted that numerical algorithms are the most preferable solution method developing for BAP.
- Only two studies emloy a hybrid approach to optimize the buffer sizes (Dolgui et al., 2007 and Amiri & Mohtashami, 2012).

- Almost half of the studies (*i.e.*, 23 out of 54) used the meta-heuristics for solving BAP. Among the meta-heuristics, genetic algorithms are the most widely used generative (search) method (*i.e.,* 14 out of 23).
- As an evaluative method, simulation and decomposition methods are most emloyed methods in BAP literature.
- Most of the studies aim at achieving throughput maximization in the production line while solving BAP (*i.e.,* 32 out of 54). In a multi-objective manner, the conflicting objectives are generally throughput maximization and total buffer size minimization in all the studies.
- It must be noted that among the studies on multi objective BAP, the strategy of search before decision making is most widely considered decision making strategy (*i.e.,* 5 out 7 studies).

2.5 Chapter Summary

In this chapter, the BAP was investigated and explained in detail. Additionally, having reviewed the relevant literature on BAP to provide a conceptual framework for this Ph.D. thesis, we stated our findings and identified the current research gaps for this study.

As a result, it can be noticed that there is lack of hybrid methods about BAP in the existing literature. On the other hand, there are many studies addressing the BAP in a single-objective manner but few studies used multi-objective approaches to solve BAP. Considering these findings and unfulfilled research potential in this area, this Ph.D. study aims at proposing hybrid methods to solve the BAP in both single and multi-objective manner.

Following chapter will focus on solution methods that are used in this study for BAP.

CHAPTER THREE
BACKGROUND INFORMATION FOR SOLUTION METHODS: SIMULATION OPTIMIZATION, EVOLUTIONARY ALGORITHMS, SIMULATED ANNEALING AND HYBRIDIZATION

3.1 Introduction

This Ph.D. study proposes hybrid meta-heuristic approaches conjunction with simulation to solve BAP. As the metaheuristics, evolutionary algorithms such as genetic algorithm (GA) and elitist non-dominated sorting genetic algorithm (NSGA-II) are employed with the adaptation of specific features of simulated annealing (SA) algorithm. The performance of these algorithms has been enhanced with hybridization and the proposed hybrid algorithms are employed with simulations in an iterative manner. To gain a more comprehensive understanding, these solution methods deployed in solving BAP are explained in detail in this chapter.

The chapter is organized as follows. Section 3.2 gives background information on simulation optimization and specifically simulation modeling which is used as an evaluative tool in this study. In Section 3.3, the basic information on evolutionary algorithms deployed for this study is presented. Section 3.4 introduces the basic principles of simulated annealing algorithm. In Section 3.5, the hybridization concepts in meta-heuristics are depicted. Finally, the context of this chapter is summarized in Section 3.6.

3.2 Simulation Optimization

The term “Simulation Optimization” can be defined as the process of finding the best values of input parameters from among all alternatives without evaluating each alternative (Yildiz, 2003). The main reason underlying the importance of simulation optimization is that most real-world problems in optimization are too complex; hence computing values of performance measures and finding optimal decision variables analytically is very hard and sometimes impossible.

In general, the simulation optimization combines the optimization techniques with simulation for the optimization problems. The objective of simulation optimization is to minimize the resources spent while maximizing the information obtained in a simulation experiment (Carson & Maria, 1997). Simulation optimization provides a structured approach to determine optimal input parameter values, where optimal is measured by a function of output variables (steady state or transient) associated with a simulation model (Swisher, Hyden, Jacobson, & Schruben, 2000). As a result, the simulation optimization consists of two main parts: generating candidate solutions and evaluating their fitness values. Figure 3.1 presents the general process of simulation optimization.

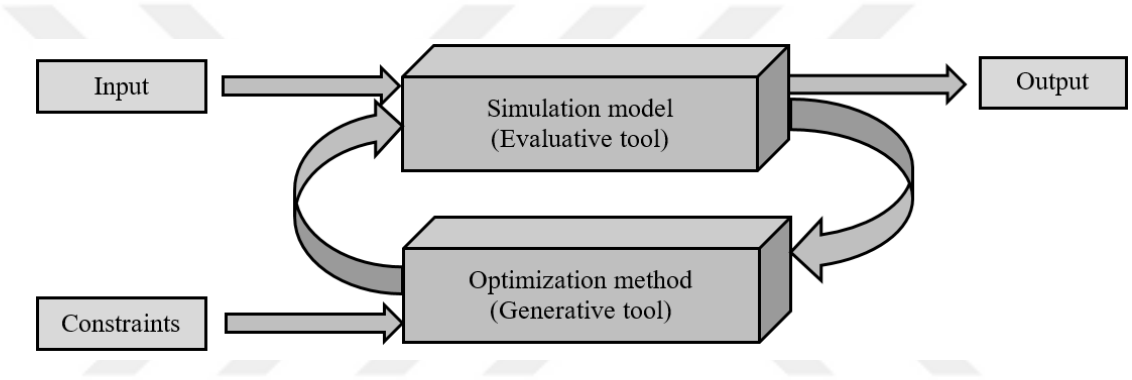


Figure 3.1 General process of simulation optimization

Simulation is one of the most commonly used evaluative tool to estimate some performance measures of complex production systems. Moreover, a simulation model is typically a descriptive model of the system under consideration, and helps decision maker to understand the dynamics and complex interactions among the elements of the system (Tekin & Sabuncuoglu, 2004). It can be also defined as the imitation of the operation of a real-world process or system over time. This method involves building a model of a system and experimenting with the model to determine how the system reacts to various conditions. One of the disadvantages of simulation is that it does not provide an optimum solution. Simulation models can be developed for a wide variety of purposes such as:

- Evaluation of system performance,
- Prediction of system behaviour in response to recent changes made in the system,

- Comparison of different system designs,
- Optimization of any system parameters with the use of other methods,
- Sensitivity analysis,
- Bottleneck analysis.

Due to the recent advancements in simulation technology and also increasing computational power with less cost, the use of simulation has evolved to the point that the decision makers do not consider the simulation models developed for design of a system as throw-away tools any more. Rather, once the system in operation, they extend the use of these models to performance evaluation and performance improvement (Kose, 2010). In this respect, the most prominent studies on the BAP such as Altiparmak et al. (2007), Battini et al. (2009), Bulgak (2006), Harris & Powell (1999), Can & Heavey (2011, 2012), Jeong & Kim (2000), Kose (2010), Kose et al. (2015), Lutz et al. (1998), Sabuncuoglu et al. (2002, 2006), Vergara & Kim (2009) used simulation as a performance evaluation tool.

As far as the optimization techniques are concerned for simulation optimization, these techniques are mainly classified in terms of local search methods and global search methods. For a more detailed survey study on these search methods and their applications, the reader can refer to Swisher et al. (2000) that surveys the literature over the year of 1988 to 2000, Tekin & Sabuncuoglu (2004) that reviews the traditional methods and global optimization methods such as evolutionary algorithms, tabu search and simulated annealing algorithm, Fu (2015) that presents an overview of the state of the art of simulation optimization, providing a survey of the most well-established approaches. In an optimization context, an overview of some global search methods relevant to developing for this study is provided in the following sections.

3.3 Evolutionary Algorithms

Evolutionary algorithms (EAs) are heuristic search methods that simulate the process of natural evolution. These algorithms work on a population of solutions in

such a way that poor solutions become extinct, whereas the good solutions evolve to reach for the optimum. The origins of EAs can be traced back to the late 1950s, and since the 1970s several evolutionary methodologies have been proposed, mainly genetic algorithms (GAs) (Goldberg, 1989), evolutionary programming (EP) (Fogel, 1992), and evolution strategies (ES) (Schwefel, 1981) (Back, Fogel, & Michalewicz, 1997). These algorithms differ in the representation of individuals, the design of variation operators, and the selection of their reproduction mechanisms (Tekin & Sabuncuoglu, 2004).

As opposed to a single solution used in traditional methods and certain meta-heuristics (e.g., tabu search, simulated annealing algorithm), EAs conduct multiple directional searches by using a population of solutions. Due to its ability to search numerous points on the problem space with numerous search directions in parallel, in this Ph.D. study, the GA and a special version of a multi-objective genetic algorithm, namely NSGA-II are used as optimization methods. The following sections present the details of these generative methods employed in this study.

3.3.1 Genetic Algorithms

Genetic Algorithms (GAs) are meta-heuristic search algorithms premised on the evolutionary ideas of natural selection and genetic. The basic concept of GAs is designed to simulate processes in natural system necessary for evolution, specifically those that follow the principles first laid down by Charles Darwin of survival of the fittest. GAs were first described by John Holland in the 1960s and further developed by Holland and his students and colleagues at the University of Michigan in the 1960s and 1970s. In 1975, Holland introduced the method with the book of *Adaptation in Natural and Artificial Systems* presenting GA as an abstraction of biological evolution.

GA is a method for moving from a constant-size population of "chromosomes" to a new population, using "selection" together with the genetics-inspired operators of crossover, mutation, and selection. The selection operator chooses those chromosomes in the population that will be allowed to reproduce, with fitter

chromosomes producing on average more offspring than less fit ones. Crossover exchanges subparts of two chromosomes, roughly mimicking biological recombination between two single-chromosome ("haploid") organisms; mutation randomly changes the allele values of some locations in the chromosome (Mitchell, 1996). The new population generated from this process is then used in the next iteration of the algorithm. The algorithm terminates when the population converges to the optimal solution. The aim during the iterative search is eventually to find solutions to a combinatorial optimization problem where the objective function value approaches the global optimum. The GA process is illustrated in Figure 3.2. For more details about GA the reader can refer to Holland (1992) and Sivanandam & Deepa (2008).

Over the last decades, GAs have found considerable interest in solving combinatorial optimization problems that arise in complex production systems. In the BAP literature, the most prominent studies employed GAs include Spinellis & Papadopoulos (2000b), Dolgui et al. (2002, 2007), Bulgak (2006), Qudeiri et al. (2007, 2008), Yamamoto et al. (2008), Lee et al. (2009), Kose (2010), Srinivas et al. (2011), Papadopoulos et al. (2013), Kose et al. (2015), and Tiacci (2015).

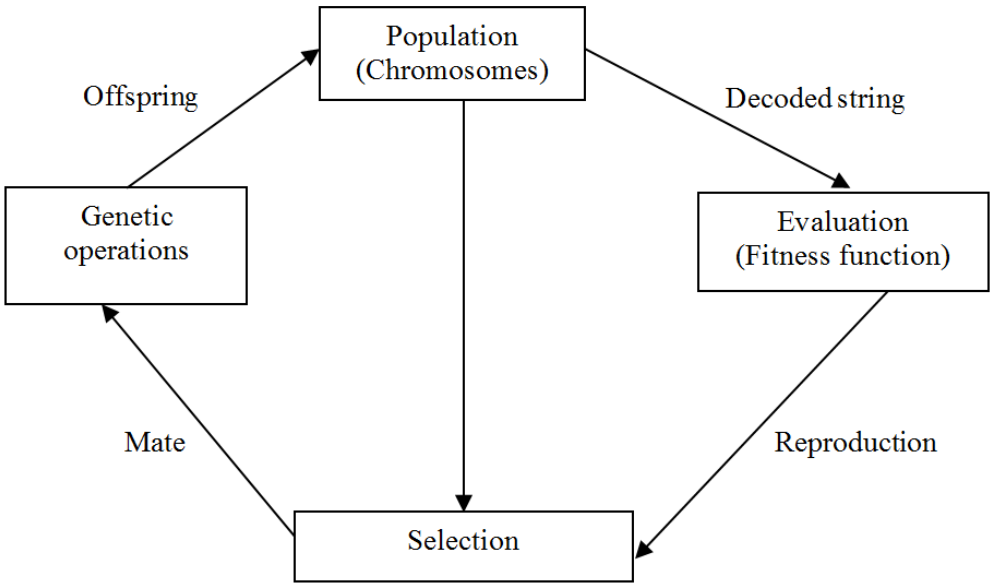


Figure 3.2 Genetic algorithm cycle (Sivanandam & Deepa, 2008)

In order to understand the philosophy of GAs, the basic terms relating to GAs must be defined. These basic components such as encoding and initialization scheme, fitness evaluation and selection scheme, genetic operators (crossover and mutation), replacement scheme and termination criteria can be described as follows.

3.3.1.1 Encoding and Initialization Scheme

In GA terminology, the chromosome represents a unique solution in the solution space. This requires a mapping mechanism between the solution space and the chromosomes. This mapping is called an encoding. In fact, GA works on the encoding of a problem, not on the problem itself (Tasan, 2007). The encoding process can be performed using various encoding schemes such as bits, numbers, trees, arrays, lists or any other objects and depends on the structure of the problem. For example, one can encode directly real or integer numbers. Hence, the way the encoding process performs differs from problem to problem.

GAs operate with a group of chromosomes, called a population. The genetic search initializes with an initial population of individuals and proceeds throughout the generations. In most of the cases, the initial population is generated randomly. But there may be instances where the initialization of population is carried out with some known good solutions. Moreover, sometimes some heuristics can be used to seed the initial population.

3.3.1.2 Fitness Evaluation and Selection Scheme

The fitness evaluation of an individual in a GA is the estimation value of an objective function for its phenotype. For calculating fitness, the chromosome has to be first decoded and the objective function has to be evaluated. The fitness not only indicates how good the solution is, but also corresponds to how close the chromosome is to the optimal one (Sivanandam & Deepa, 2008).

Selection scheme is the process that picks individuals out of the population according to their fitness function. The individuals are selected among existing

chromosomes in the population with preference towards fitness and exposed to genetic operations such as crossover and mutation. The Figure 3.3 depicts the basic selection process.

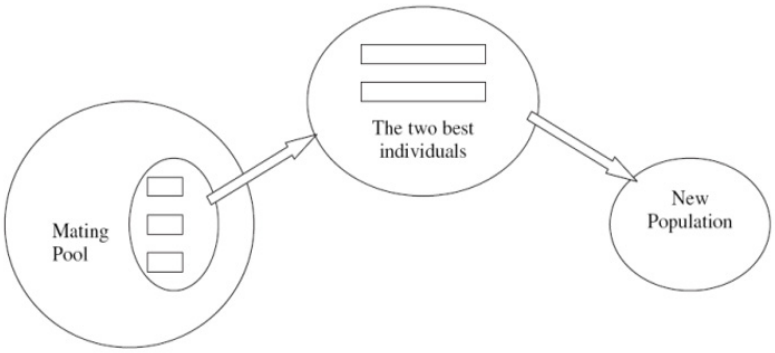


Figure 3.3 Basic selection process (Sivanandam & Deepa, 2008)

The various selection schemes are discussed in the literature. Among these schemes, two traditional selection schemes as used in this Ph.D. study, are roulette wheel selection scheme and tournament selection scheme. The roulette wheel selection, proposed by Holland (1975), is based on a linear search through a roulette wheel with the slots in the wheel weighted in proportion to the individual's fitness values. Other traditional selection scheme, tournament selection was proposed by Goldberg & Deb in 1991. In this scheme, k individuals ($k \geq 2$) are selected randomly from the larger population, and the selected individuals compete against each other. The individual with the highest fitness wins and will be included as one of the next generation population. Tournament selection also gives a chance to all individuals to be selected and thus it preserves diversity, although keeping diversity may degrade the convergence speed (Razali & Geraghty, 2011).

3.3.1.3 Crossover and Mutation

Crossover and mutation are the main genetic operators in GAs in order to generate new individuals from selected chromosomes in a population. In crossover, two chromosomes, namely parents, are selected and they are combined together to form new chromosomes, namely offspring. Crossover operator is used for the hope that it

creates a better offspring. The most common used crossover techniques are single point, two point, multi-point crossover, uniform crossover, three parent crossover, shuffle crossover and precedence preservative crossover (see Sivanandam & Deepa, 2008, for further details).

After crossover, the chromosomes are subjected to mutation. Mutation is an important part of the genetic search to introduce new genetic structures in the population by modifying one ore more gene values in a chromosome. Moreover, mutation helps to escape from local minima's trap and maintains diversity in the population. There are many different forms of mutation for the different kinds of representation, i.e., flipping, interchanging, reversing, etc.

3.3.1.4 Replacement Scheme

The key idea behind the concept of replacement scheme is to determine which individual should stay in a population and which individual should transfer to the next generation. In general, the most commonly used replacement strategy is elitism, which makes survival of some number of the best individuals at each generation; hence guaranteeing that the final population contains the best solution ever found.

3.3.1.5 Termination Criteria

It is a critic decision for decision-maker whether to continue the genetic search or stop the search. For GAs, setting a fixed number of generations is the most common used criterion. There are also various termination criteria summarized as follows (Sivanandam & Deepa, 2008):

- *Maximum generations:* The genetic algorithm stops when the specified number of generations evolves.
- *Elapsed time:* The genetic process ends when a specified time elapses. If the maximum number of generation is reached before the specified time elapses, the process ends.

- *No change in fitness:* The genetic process ends if there is no change to the best fitness of population for a specified number of generations. If the maximum number of generation is reached before the specified number of generation with no changes is reached, the process ends.
- *Stall generations:* The algorithm stops if there is no improvement in the objective function for a sequence of consecutive generations.
- *Stall time limit:* The algorithm stops if there is no improvement in the objective function during an interval of time.

3.3.2 NSGA-II

Another evolutionary algorithm proposed to solve multi-objective BAP in this study is NSGA-II. The main motivation for using EAs to solve multi-objective optimization problems is because EAs deal simultaneously with a set of possible solutions (the so-called population) which allows us to find several members of optimal solutions (largely known as Pareto-optimal solutions) concurrently in a single run of the algorithm. For the multi-objective optimization, there are various evolutionary algorithms suggested so far in the literature. For more details about the evolutionary algorithms, the interested reader can refer to Kumar, Saxena, & Kumar (2014). Among the evolutionary algorithms, it is proved that elitist non-dominated sorting genetic algorithm (NSGA-II) is the best algorithm for multi-objective optimization in view of the convergence and divergence metrics of performance (Kumar et al., 2014).

NSGA-II is a multi-objective evolutionary algorithm based on the revised version of non-dominated sorting genetic algorithm (NSGA). This algorithm was first introduced by Deb, Pratap, Agarwal, & Meyarivan (2002) with the work of solving non-convex and non-smooth single and multi-objective optimization problems. The main concept of this algorithm lies on the concept of non-domination between two solutions. Unlike the naive approach (NSGA), in NSGA-II, sharing function approach is replaced with a crowded-comparison approach in the context of non-dominated sorting scheme and elitism is used as a replacement scheme for the next

generation. On the other hand, Deb et al. (2002) has implemented Goldberg's sketch to this algorithm most directly. For more details of this algorithm, the reader can refer to Deb et al. (2002). The step-by-step procedure for NSGA-II is depicted in Figure 3.4.

At first, the algorithm initializes an *N*-sized parent population and creates an offspring population using usual operators of GA, i.e., crossover and mutation. Following, parent and offspring populations are merged into one population of size 2*N*. At this point, the new population is formed with the non-dominated sorting scheme to generate a better-fitting population. The important issues need to be considered in implementation of the NSGA-II are summarized in the following subsections.

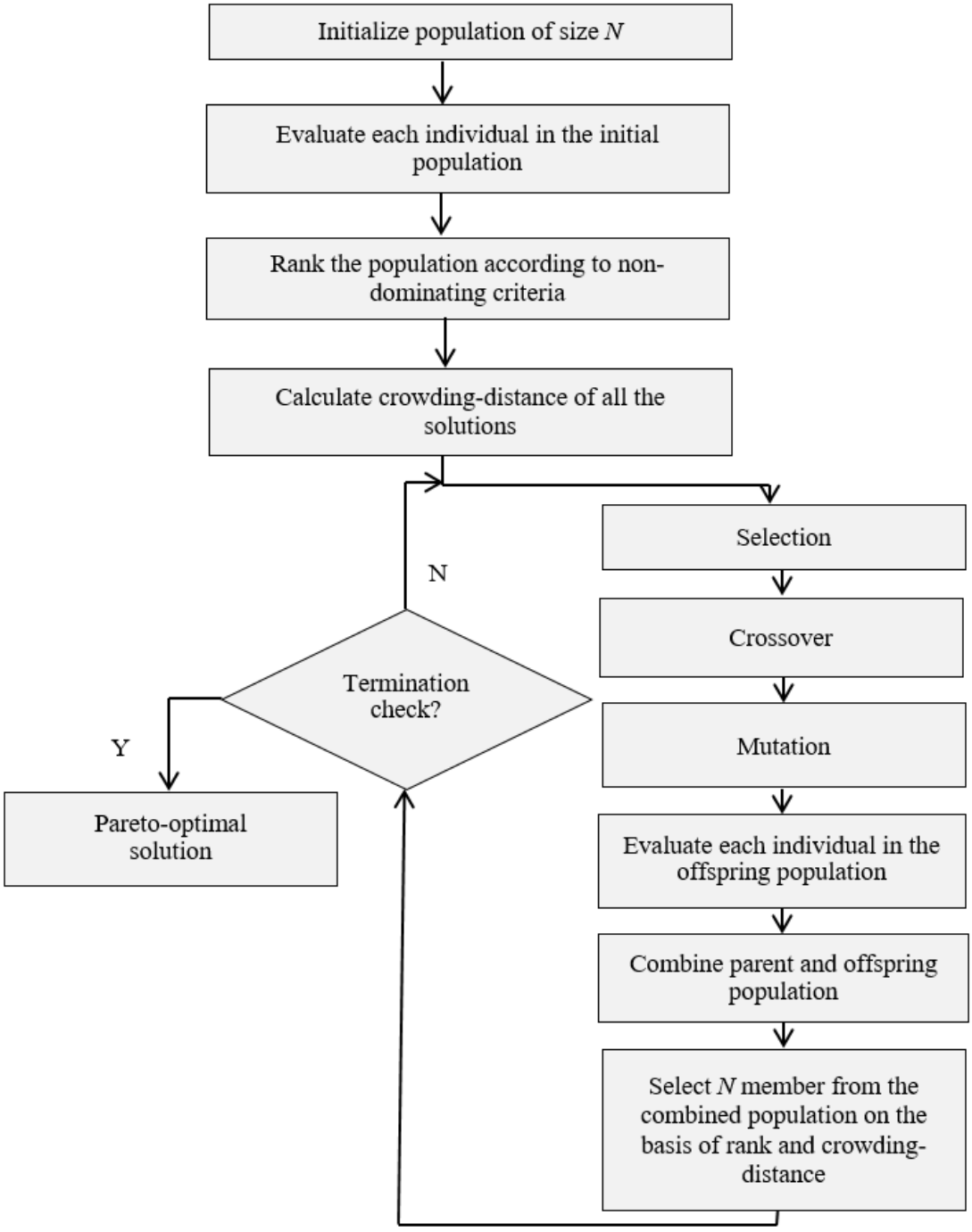


Figure 3.4 NSGA-II algorithm

3.3.2.1 Non-dominated Sorting Criteria

A solution is said to be non-dominated if none of the objective functions can be improved in value without degrading some of the other objective function values. Figure 3.5 explains non-domination for minimization function. For instance, the

points of A and D are non-dominated solutions since none of them are dominated by each point in the figure for both f_1 and f_2.

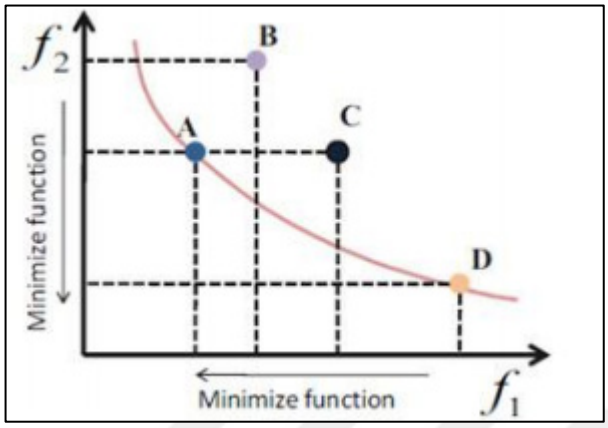


Figure 3.5 Non-domination scheme (Anwar & Manoj, 2015)

In the context of non-dominated sorting criteria, a set of non-dominated fronts are assigned using ranking mechanism. The first non-dominating front is generally assigned a rank of one. Similarly the second non-dominating front has a rank of two and so on. The solutions having lesser rank are better than those in another non-dominated front (See Figure 3.6).

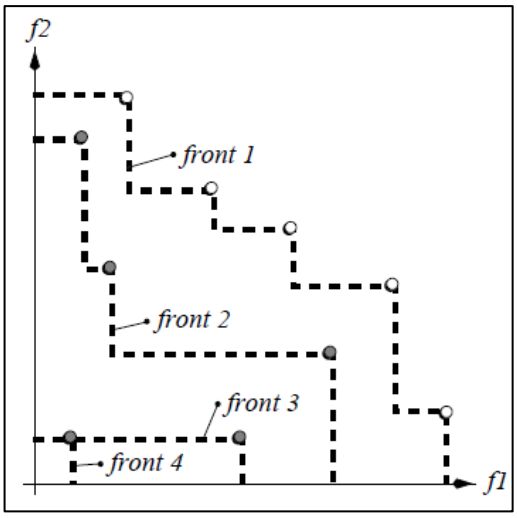


Figure 3.6 Illustration of fronts for NSGA-II (Zitzler, 1999).

3.3.2.2 Crowding-Distance

The main outcome of the NSGA-II is the assignment of a rank to each solution in the population. Two solutions of the same rank do not dominate each other. Thus, NSGA-II assigns a secondary ranking measure to each solution, namely crowding-distance. This distance implies the biggest cuboid containing the two neighboring solutions of the same non-dominating front in the objective space. The crowding-distance computation requires sorting the population according to each objective function value in ascending order of magnitude (Deb et al., 2002). Thereafter, for each objective function, the boundary solutions denoted by S_{CD} and L_{CD} (solutions with smallest and largest function values) are assigned an infinite crowding distance value. Figure 3.7 shows the crowding distance of the i^{th} solution in its front based on their m objectives as an average side length of the cuboid shown with a dashed box and the formula in which the crowding distance value for i^{th} individual (i_{CD}) is estimated with the absolute normalized difference in the function values of two adjacent solutions.

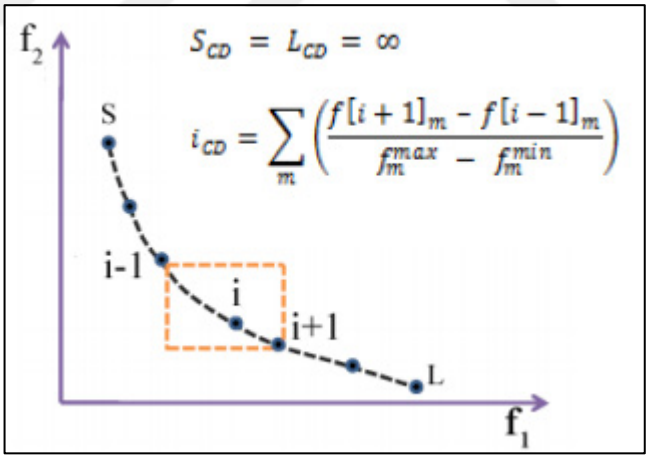


Figure 3.7 Crowding-distance interpretation (Anwar & Manoj, 2015)

Following, a random subset of k solutions from the original population is selected for crossover and then the best solution out of this subset is chosen using the crowding-distance measurement. The comparison is carried out as below:

1. if solution i has a better rank, that is $r_i < r_j$,

2. if they have the same rank but solution *i* has a better crowding distance than solution *j*, that is, $r_i = r_j$ and $d_i > d_j$.

Hence, solution *i* is selected as a candidate solution for crossover.

3.3.2.3 Elitism Scheme

NSGA-II implements elitism for multi-objective search using an elitism-preserving approach. Its elitist mechanism consists of combining the best parents with the best offspring obtained. In this scheme, the new population for the next generation is filled by each front subsequently until the population size exceeds the current population size (*N*). In other words, the filling starts with the best non-dominated front and continues with solutions of the second non-dominated front, followed by the third, and so on. If all the solutions of a front cannot be accommodated in the new population, the solutions having large crowding distance will get preference to replace the old individual. As a result, old population is replaced by the better members of the combined population (parent and offspring population). The schematic representation of elitism is shown in Figure 3.8.

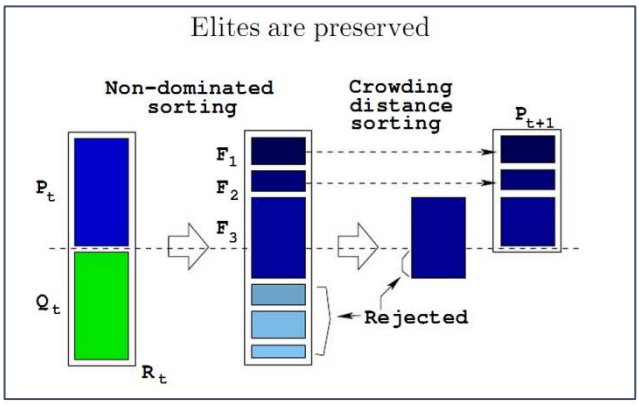


Figure 3.8 Elitism scheme (Deb et al., 2002)

3.4 Simulated Annealing Algorithm

Simulated annealing (SA) algorithm is another meta-heuristic search algorithm introduced by Kirkpatrick, Gelatt & Veechi (1983) to solve NP-hard combinatorial optimization problems. Inspired from the analogy with the simulation of the physical

annealing of solids, solutions in the combinatorial problem are equivalent to states of a physical system, and the cost of a solution is equivalent to the energy of a state (Aarts & Korst, 1989).

SA starts from an initial solution to the problem and an initial temperature value, T_0, which is a control parameter for SA is systematically decreased according to an annealing schedule. In this schedule, after producing an initial solution, the neighborhoods of the solutions are generated by using a neighborhood generation mechanism. Once a new solution is created, the corresponding change in the acceptance function is computed to decide whether the newly produced solution can be accepted as the current solution. If the change in the acceptance function (ΔE) is negative the newly produced solution is directly taken as the current solution (Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, & E. Teller, 1953). Otherwise, it may still be accepted with a Boltzman's probability based on Metropolis's criterion. This probability of accepting inferior solutions allows the SA algorithm to escape from getting trapped into local minima. For this criterion, a random number $R \in [0,1]$ is generated from a uniform distribution. If

$$R \leq \exp(-\Delta E / T) \tag{3.1}$$

then the newly produced solution is accepted as the current solution. Otherwise, the current solution is unchanged. For more detailed information about SA, the reader can refer to Kirkpatrick et al. (1983) and Dowsland & Thompson (2012). The flowchart of a standard SA algorithm is shown in Figure 3.9.

Recently, SA has been used to cope with multi-objective combinatorial optimization problems because of its simplicity and capability of producing a Pareto set of solutions in a single run with very little computational cost. Suman & Kumar (2006) provides detailed information on multi-objective simulated annealing (MOSA). The first multi-objective version of SA has been proposed by Serafini (1985, 1992). The algorithm of the method is almost the same as the algorithm of single objective SA. The method uses a modification of the acceptance function in

the standard algorithm. Moreover, various alternative approaches to the MOSA have been investigated adopted by several investigators (Czyzak & Jaszkiewicz, 1998; Hapke, Jaszkiewicz, & Slowinski, 2000; Nam & Park, 2000, Serafini, 1994; Suppapitnarm, Seffen, Parks, & Clarkson, 2000; Tuyttens, Teghem, & El-Sherbeny, 2003; Ulungu, Teghaem, Fortemps, & Tuyttens, 1999) in order to increase the probability of accepting non-dominated solutions. The basic idea behind these approaches is to combine the objectives as a weighted sum and use the composite objective as the energy to be minimized in a scalar SA optimizer. The general scheme using the combination of objectives into a single fitness function is given as follows:

$$E(x) = \sum_{i=1}^{m} w_i f_i(x) \tag{3.2}$$

where m denotes the objectives for the problem and $w_i \ldots w_m$ are the weights given to each objective function.

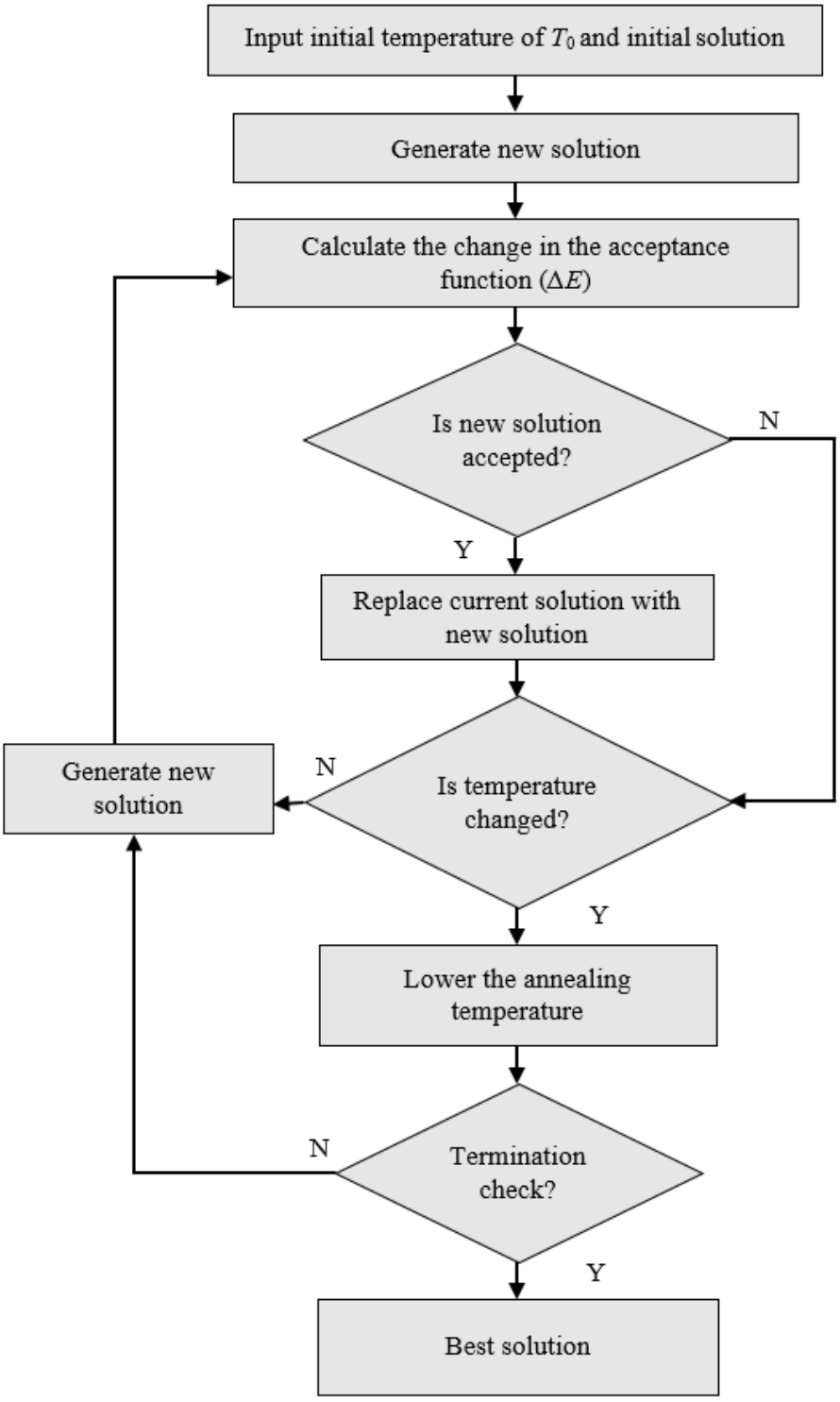


Figure 3.9 Flowchart of a standard SA algorithm

A novel enhancement in the MOSA has been proposed in the study by Sánchez & Villar (2008) in which a population based MOSA algorithm is used for obtaining transparent models of chaotic systems. In this algorithm, a number of solutions are

annealed simultaneously, and the ranking and selection are done based on Pareto dominance.

Despite the large number of studies dealing with the BAP in the literature, one can feel the lack in the works using MOSA in solving multi-objective BAP. On the other hand, the standard SA algorithm has been proposed as a solution approach for single objective BAP. The studies of Spinellis et al. (2000), Spinellis & Papadopoulos (2000a, 2000b), Papadopoulos et al. (2013) and Kose et al. (2015) can be given as applications of SA algorithm for the solution of BAP in serial production lines. Within this framework the basic elements of SA algorithm can be described in the following subsections.

3.4.1 Neighborhood Generation Mechanism

The way in generating neighborhoods that moves from one solution to another is critical dilemma faced in many algorithms. In SA algorithm, the neighborhoods are generally sampled randomly. Using different neighborhood selection strategies in a systematic way or sampling in an adaptive way can significantly improve the efficiency and effectiveness of the algorithm. Especially selecting the neighborhood in multiple ways can be a powerful tool in preventing the method from being trapped in local optima.

3.4.2 Cooling Schedule

The cooling schedule of a SA algorithm often characterized by the initial temperature (T_0), cooling function and iterations at each temperature. The following sections are devoted to the explanation of these components.

3.4.2.1 Initial Temperature

Kirkpatrick et al. (1983) remarked in their study that the value of T_0, is set large enough to make the initial probability of accepting transitions be close to 1. A suitable initial temperature, T_0 is one that results in an average increase of acceptance

probability. In other words, there is chance about an average ratio that a change which increases the objective function will be accepted. As it is stated by Reeves (1996), an acceptance probability rate of between 40% and 60% seems to give good results in many cases.

3.4.2.2 Cooling Function

In much research, the initial temperature is reduced with a so-called proportional cooling function, which was originally suggested by Kirkpatrick et al. (1983). In fact, the convergence of the algorithm is proven when the temperature decreases with a logarithmic rate. A low cooling rate may increase the running time of the algorithm while a high cooling rate may cause being trapped in a local optimum. For cooling function, there are several ways to adjust the temperature rate in which the temperature decreases may heavily affect the performance of SA. Among these ways, the simplest method used is linear cooling scheme in which the temperature is reduced in every trials. This method updates the temperature by the following formula:

$$T_{i+1} = T_i - \Delta T \tag{3.3}$$

The other simplest and most common schedule is the geometric cooling schedule. In this cooling schedule, the temperature is updated by the following formula:

$$T_{i+1} = cT_i \quad i = 0,1,... \tag{3.3}$$

where c is a cooling rate which is a constant close to 1 (typically in the range 0.80 to 0.99). Another alternative schedule to lower the temperature is exponential cooling schedule. The temperature at the i^{th} iteratation is determined with the formula as below:

$$T_i = c^i T_0 \tag{3.4}$$

Of special theoretical importance is the logarithmic cooling scheme introduced by Geman & Geman (1984),

$$T(i) = \frac{c}{\log(i+d)} \tag{3.5}$$

where *d* is usually set equal to one. As the only existence theorem, it has been proven that for *c* being greater than or equal to the largest energy barrier in the problem, this schedule will lead the system to the global minimum state in the limit of infinite time (Hajek, 1988).

3.4.2.3 Iterations at Each Temperature

The number of iterations at each iterations can be determined as proportional to the size of the problem instance. For example, in case of BAP, the number of iterations can be set to the number of stations in the production line. As another alternative method used in the SA algorithm, this value can be set to be proportional to the number of neighborhood solutions generated. In the study by Reeves (1996), the number of iterations is determined with:

$$k = \frac{logT_f - logT_0}{\log c} \tag{3.6}$$

where T_f is the final temperature and T_0 is the value for the initial temperature.

3.5 Hybridization of Meta-heuristics

To increase the performance of the aforementioned approaches based on meta-heuristics, a recent trend observed in the literature is to hybrize the meta-heuristics to solve any combinatorial optimization problem. According to Blum, Roli, & Alba (2005), the hybridization of metaheuristics is the most promising avenue for improving the quality of solutions in many real applications. Hybrid meta-heuristics can combine both single solution algorithms (such as SA, tabu search) and

population-based algorithms (such as GAs, differential evolution algorithm, particle swarm optimization algorithm, ant colony optimization algorithm).

The motivation behind hybridizations of different algorithmic concepts is usually to obtain better performing systems that exploit and unite advantages of the individual pure strategies, i.e., such hybrids are believed to benefit from synergy. The vastly increasing number of reported applications of hybrid meta-heuristics shows the popularity, success, and importance of this specific line of research. In fact, today it seems that choosing an adequate hybrid approach is determinant for achieving top performance in solving difficult problems (Raidl, 2006).

3.5.1 Hybridization of Genetic Algorithm with Simulated Annealing Algorithm

Among the modern heuristic methods, GA and SA are most popular meta-heuristics for solving combinatorial optimization problems. Indeed, to improve the effectiveness of these methods, the respective advantages of the GA and SA algorithm are combined and generally referred to GA-SA hybrid algorithm (or *Genetic Annealing Technique*). The first Genetic Annealing technique has been introduced by Price (1994) with the paper published in the October 1994 issue of Dr. Dobb's journal. The algorithm is a population-based combinatorial algorithm that realized an annealing criterion via thresholds driven by the average performance of the population (Feoktistov, 2006). Moreover, in recent years, the hybrid GA-SA algorithm has been applied to the fields of system design (Shieh & Peralta, 2005), system and network optimization (Zhao & Zheng, 2006), continuous-time production planning (Ganesh & Punniyamourthy, 2005), electrical power districting problem (Bergey, Ragsdale, & Hoskote, 2003), scheduling problems (Nearchou, 2004; Wang & Zheng, 2001; Yoo & Gen, 2007), machine loading problems (Yogeswaran, Ponnambalam, & Tiwari, 2009) and tool-path planning problems (Oysu & Bingul, 2009) by many researchers.

While hybridizing the GA and SA, one should take into both strengths and weaknesses of algorithms consideration. GA exhibits implicit parallelism and can retain useful redundant information about what is learned from previous searches by

its representation in individuals in the population (Zhang, Li, Rao, & Li, 2005). However, GA is also prone to loss of solutions due to the disruptive effects of genetic operators and suffers from poor convergence properties. By contrast to GA, SA possesses good convergence property and the ability to probabilistically escape from local optima; it accepts or rejects the newly generated candidate solution probabilistically by the Metropolis acceptance criterion, where inferior candidate can be accepted some of the time. Since SA maintains only one solution at a time, whenever it accepts a new structure, it must discard the old one, so there is no redundancy and no history of past structures (Zhang et al., 2005). However, SA costs lots of computation time in annealing process. On the other hand, there is no guarantee that the GA algorithm will succeeded in escaping from any local minima, thus it makes sense to hybridize the GA and the SA technique.

To improve premature convergence and receive more suitable objective function values, it is necessary to find some effective approach to overcome the drawbacks of GA and SA. At this point, hybridization of GA with SA algorithm is an innovative trial by applying the superior capability of SA algorithm to reach more ideal solutions, and by employing the reproduction process of GA to enhance searching process. Generally, there are two common used schemas that hybridize the GA with SA algorithm with the framework illustrated in Figure 3.10. In the first schema, SA algorithm is used as a genetic operator of the GA. In this respect, the characteristics of SA are embedded into GA to improve the quality of individual. The second hybrid schema executes a GA until the algorithm completely finishes. Once some individuals are selected from the final population, the SA algorithm is executed over them. This schema is called sequential hybridization, where one meta-heuristic is performed after the other. In doing so, it is hoped that using the final selected GA solution as the initial solution of SA might enhance the chances to improve the quality of solution.

In this Ph.D. study, we propose two hybrid methodologies integrating the characteristics of SA into GA for single objective BAP and MOSA into NSGA-II for multi-objective BAP. By exploiting the efficient parallelization of the GA and the

convergence control of SA, the Metropolis acceptance criterion is incorporated into replacement scheme.

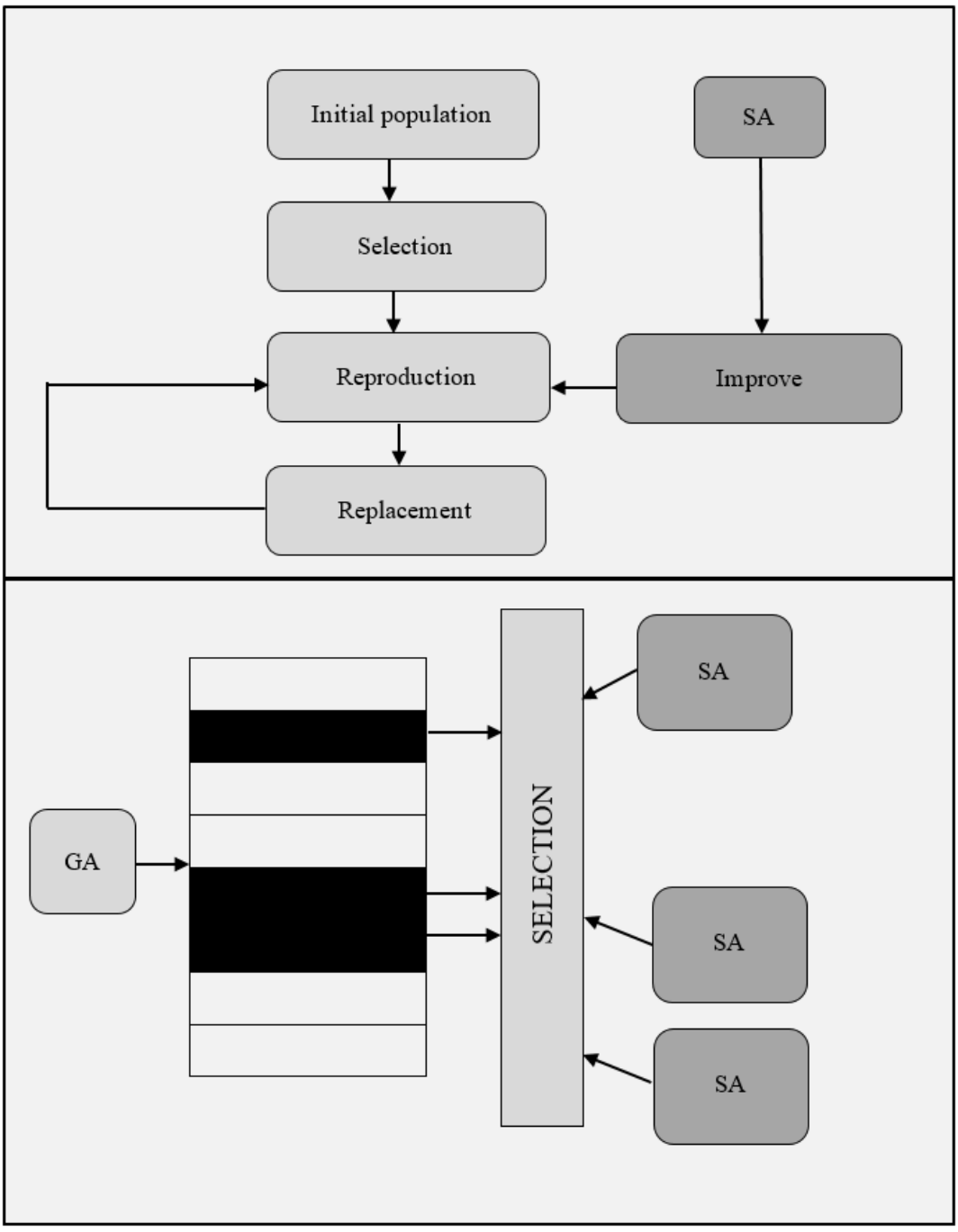


Figure 3.10 Hybrid representations of GA with SA.

3.6 Chapter Summary

In this chapter, a comprehensive survey on solution techniques used in this Ph.D. study is presented. As it is stated in the previous chapter, the studies in which hybrid meta-heuristic approaches are proposed to solve BAP are still very limited. In this respect, we propose a hybrid metaheuristics-based simulation optimization approaches to solve BAP. These approaches involve the use of a generative method and an evaluative method in an iterative manner. As a generative tool, evolutionary algorithms are hybridized with specific features of SA algorithm. For single objective BAP, GA is used with the adaptation of SA algorithm. For solving BAP in a multi-objective manner, NSGA-II is hybridized with the SA. As an evaluative tool, discrete event simulation modeling is used to estimate the performance measures of the production line.

Next chapters are devoted to the details of our proposed hybrid meta-heuristic-based simulation optimization approaches for solving single and multi-objective BAPs in serial production lines.

CHAPTER FOUR
A HYBRID APPROACH FOR SINGLE OBJECTIVE BAP IN SERIAL PRODUCTION LINES

4.1 Introduction

In this chapter, we mainly propose a hybrid approach by combining the key advantages of simulation, meta-heuristics, and hybridization for solving single objective BAP in serial production lines. The main objective is to find optimal or near-optimal buffer sizes for the production lines so as to maximize the average production rate of the line subject to a given total buffer size constraint. In our proposed approach, a hybrid algorithm (*Genetic Annealing Algorithm-GAA*) combining two meta-heuristics –Genetic Algorithm (GA) and Simulated Annealing Algorithm (SA)- is employed with simulation in an iterative manner. To ensure high performance of the proposed algorithm, the best values/scheme for the GAA control parameters are identified using design of experiments (DOEs). Following, the performance of the proposed approach and the power of the hybridization are investigated for various serial line configurations.

The remainder of this chapter is organized as follows. The next section defines the BAP along with the specific features for this study. In Section 4.3, the proposed hybrid solution approach and its implementation are explained in detail. Computational experiments are given in Section 4.4. Finally, in Section 4.5, the context of this chapter is summarized.

4.2 Problem Statement

This chapter adresses the single objective BAP which tries to maximize the average production rate subject to a given total buffer size constraint in open serial production lines. As given earlier in Chapter 2, this problem is formulated as follows:

$$\text{Find } B = (B_1, B_2, B_3, \ldots, B_{W-1}) \text{ to} \tag{4.1}$$

$$\max f(B) \tag{4.2}$$

subject to

$$\sum_{i=1}^{W-1} B_i = K \tag{4.3}$$

$$Bi \text{ non-negative integers } (i = 1, 2,\ldots, W\text{-}1) \tag{4.4}$$

where W is the number of workstations in the line, K is a non-negative integer representing the total buffer size in the system allocated among the W-1 buffer locations, B represents a buffer vector, and $f(B)$ represents the average production rate of the system under a given buffer configuration.

Since the production lines under consideration in this study are of open production lines, the first workstation on the line is never starved and the last workstation is never blocked. Additionally, each part goes through all the workstations in exactly the same order. In the following sections, we introduce the solution method to solve single objective BAP formulated above and investigate optimal buffer designs in various cases involving different system requirements for open serial production lines.

4.3 Proposed Hybrid Approach

To deal with the combinatorial nature of the BAP, meta-heuristics have recently been used as efficient search tools solve the problem as mentioned in the previous chapter. In this respect, a hybrid genetic annealing algorithm (GAA) combining meta-heuristics of GA and SA based on the simulation optimization approach has been introduced for solving the BAP. The motivation behind the hybridization is to combine the strengths of individual pure meta-heuristics and achieve top performance in solving the BAP. GA is a well-known evolutionary algorithm that mimics the process of natural selection and working with population-based solutions, using crossover and mutation operators. On the other hand, SA is a meta-heuristic algorithm that interprets slow cooling as a slow decrease in the probability of accepting worse solutions as it explores the solution space. While exploiting the power of GA to obtain candidates in different areas in the search space, SA possesses better convergence properties that can help to improve the solution. Thus, the hybrid

algorithm is enhanced by the robust characteristics of GA and SA, and it leads to better solutions relative to those achieved by the traditional GA and SA examined in this study. To the best of our knowledge, this study represents the first attempt to solve the BAP in serial lines by using the hybrid GAA.

The proposed hybrid approach to solve the BAP involves applying a search method and an evaluative method in an iterative manner. As a search tool, the GAA creates buffer size configuration searching the solution space and determines optimal or near-optimal buffer sizes. As an evaluative tool, a detailed simulation model of the production line is used by estimating the fitness value of each candidate buffer size configuration created by GAA. Although the flow of GAA and its general structure are similar to those of the traditional GA, its performance has been enhanced with the adaptation of an acceptance test in SA as the replacement test in GA. In this framework, members of a new population are replaced by new ones generated after either the crossover or the mutation operation. Hence, the GAA combines the strengths of GA such as working with population-based solutions, using crossover, mutation operators and SA incorporating acceptance probability into a single approach. The flow chart of the proposed hybrid algorithm is given in Figure 4.1. The further details about the features of the proposed hybrid approach are explained in the following subsections.

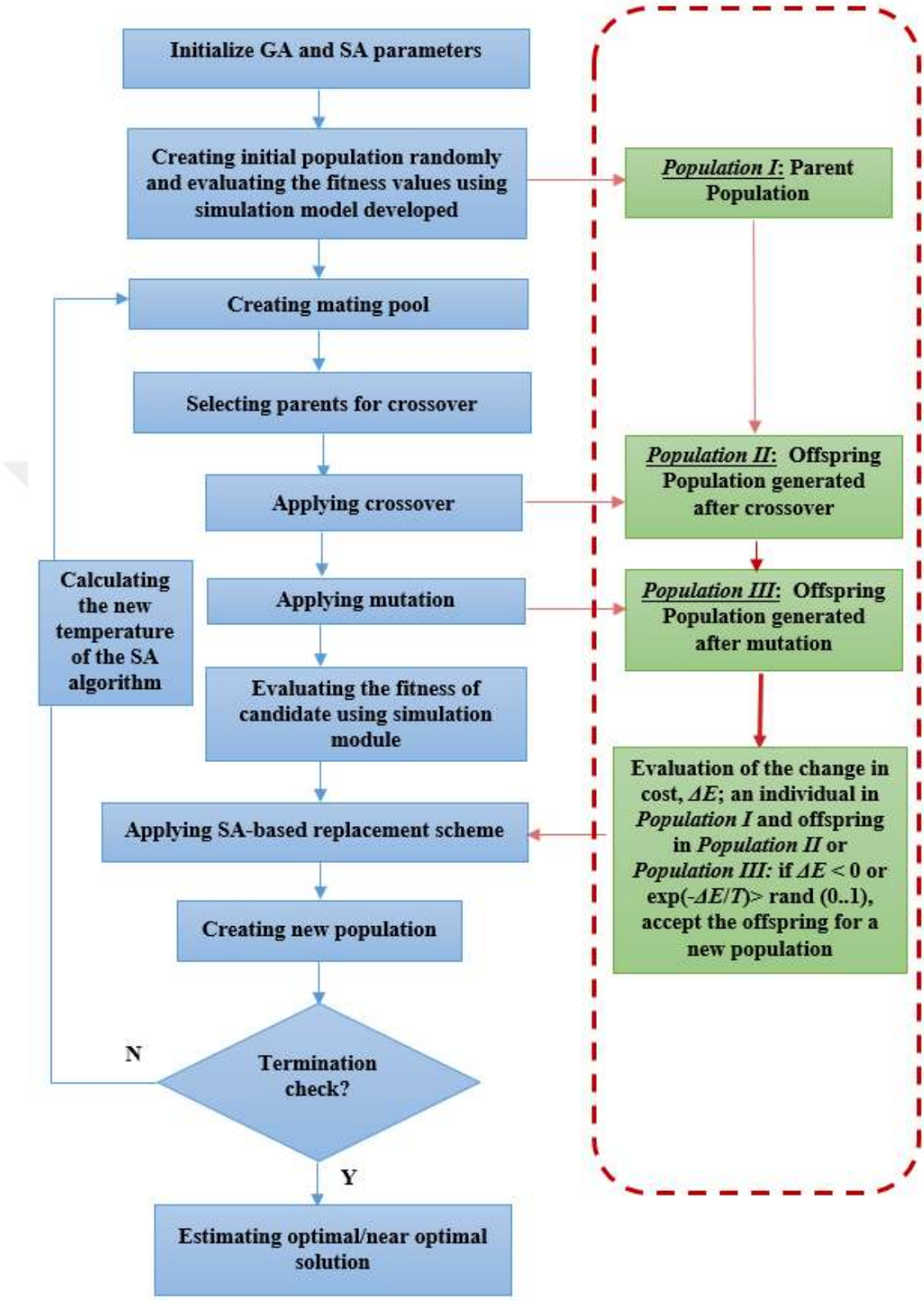


Figure 4.1 The control logic of the proposed hybrid GAA.

4.3.1 The General Specifications of the Proposed Hybrid Approach

The following subsections explain the prominent features adopted to the proposed hybrid approach, namely, the encoding and initialization scheme, fitness evaluation, selection, crossover, mutation, SA-based replacement and search termination schemes.

4.3.1.1 Encoding and Initialization Scheme

Creating an initial population is the first step of the traditional GA and proposed hybrid approach. Goldberg (1989) refers to the processing leverage of GAs as implicit parallelism; thus, GAs have the advantage that all individuals in the population are evaluated independently. In this study, the initial population consists of a sufficient number of individually feasible buffer configurations. Each buffer configuration is composed of an integer-valued coded string that represents the decision variables of the BAP ($B = \{B_1, B_2, \ldots, B_{W\text{-}1}\}$). Figure 4.2 illustrates an example of integer-valued representation of the solution for a 10-workstation and nine-buffer production line.

5	4	2	5	11	0	3	7	23

Figure 4.2 Integer-valued encoding representation

Each integer-valued coded string is generated in a random way. For environmental reasons such as layout limitations, one should consider upper and lower sizes for each buffer ($L_i \leq B_i \leq U_i$, i = 1, 2, …, W-1). Hence, L_i represents the lower limit and U_i denotes the upper limit of each buffer size. Since a pre-specified total buffer size (K) must be allocated among the W-1 buffer locations in the system in solving BAP, it should be noted that each string as a buffer configuration is created in a spesific manner as depicted step by step in Figure 4.3. In this figure, a five-workstation and four-buffer production line is considered and it can be assumed that the total buffer size (K) is set to 20. Each buffer is also bounded with the lower size (l) of 0 and according to available total buffer size, the upper value (u) is equal

to 20. In respect to these limitations, the initialization procedure can be summarized as follows:

Initilization Procedure:

for each candidate buffer configuration to generate initial population **do** (*i*=1 to population size, *pop_size*)

for each string (*j*=1 to *W*-2) **do**

Random_string (:,*j*)= randint(1,1,[*L U*])

end

sort randomly generated string in an ascending order

add lower and upper buffer sizes to the string in a ranked manner

subtract the values between two consecutive buffer sizes in the string

generate an individual based on the differences in a *Random_string*

Population(*i*,:)= *Random_string*

end

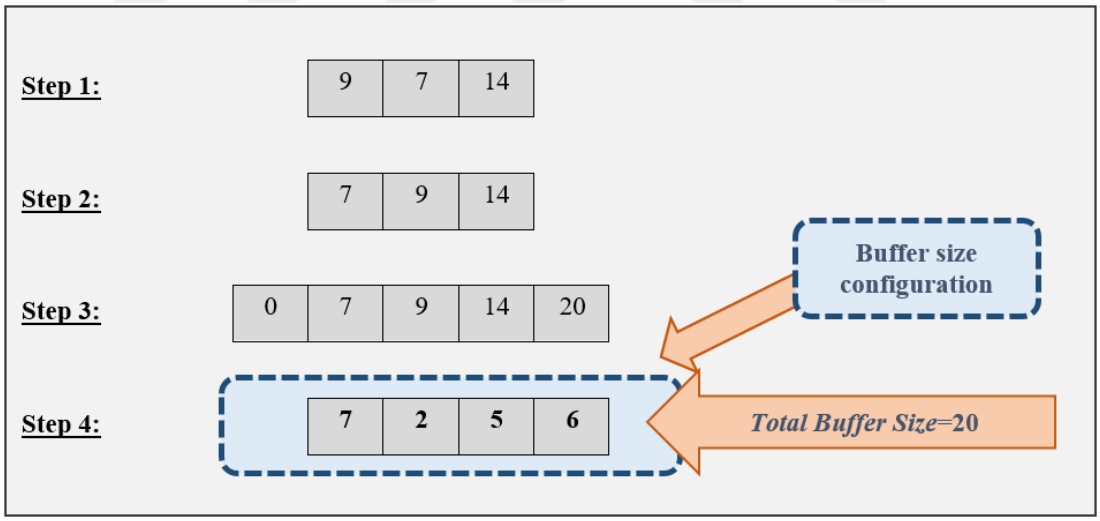


Figure 4.3 Initialization scheme for a buffer size configuration

Since random initialization may yield some infeasible individuals violating the constraint on the lower and upper limit of buffer sizes in the last step, a feasibility check is done to assure that infeasible solutions are not allowed into the initial population.

4.3.1.2 Evaluation and Selection Scheme

Having generated the initial population, each buffer configuration in the population is evaluated. The fitness value of each individual, *i.e.*, the average production rate, is estimated by using the simulation model of the system. Then, this value is communicated to the GAA in an iterative manner. The fitness evaluation procedure used in this study is given below:

Fitness Evaluation Procedure:

for each individual to obtain the fitness values **do**

write *Buffer_sizes* to an external file

execute the system file to run the simulation model

read average production rate from an external file

end

For the selection scheme, we use the roulette wheel selection, which is the most commonly used traditional GA selection scheme. The principle behind this selection scheme is a linear search through a roulette wheel with the slots in the wheel weighted in proportion to the fitness values of individuals. First, the fitness values of the individuals within the population are scaled and then cumulative survival probabilities are calculated. Following, two individuals are chosen at random from the current population for reproduction. To select an individual, first, a uniform random number within the interval (0, 1) is generated, and then the member whose cumulative survival probability is greater than the generated number has been selected as an individual (Kose, 2010). Below the algorithm of this selection sheme is given.

Selection Procedure:

for i = 1 to population size, *pop_size,* **do**

calculate *Prob_Fitness*, *i.e.,* survival probabilities for every individual in the population

calculate *Cum_Prob_Fitness, i.e.*, cumulative survival probabilities for every individual

end

loop until *offspring_candidate* = 2, *i.e.*, select two individuals as a parent;

generate a random number, *rand_num,* between 0 and 1

for *i* = 1 to population size, *pop_size,* **do**

if *rand_num* less than cumulative survival probability

offspring_candidate has been **selected**

end

end

end

4.3.1.3 Crossover and Mutation

To obtain better buffer configurations, we use crossover and mutation operators in the GAA, similarly to GA. Crossover is the main genetic operator that combines two individuals, namely, parents from the current population to produce two new offspring for the next population. Once the selection process is performed, the new population in which all of the individuals are subject to crossover with a probability of crossover (P_c) is selected. In this study, we use the same crossover procedure adopted by Vergara & Kim (2009), who also modified the procedure developed by Liu & Tu (1994) in their study. The implementation steps of the crossover operation between the parents (*a* and *b*) to generate their offspring (*a'* and *b'*) are summarized as follows (Vergara & Kim, 2009):

Step 1: $a_i' = (a_i + b_i)/2,\ b_i' = (a_i + b_i)/2,$ for all *i*;

Step 2: Find the elements that are not integers in a' and b' and pair them successively as:

(a_{i1}', a_{j1}'), …, (a_{ip}', a_{jp}') and (b_{i1}', b_{j1}'), …, (b_{ip}', b_{jp}')

Step 3: $a_{ik}' = a_{ik}' - 0.5$ and $a_{jk}' = a_{jk}' + 0.5$, for all k = 1,2,…, p;

Step 4: $b_{ik}' = b_{ik}' + 0.5$ and $b_{jk}' = b_{jk}' - 0.5$, for all k = 1,2,…, p;

Step 5: a' and b' replace *a* and *b* in the new population.

The process of the crossover operation is illustrated in Figure 4.4.

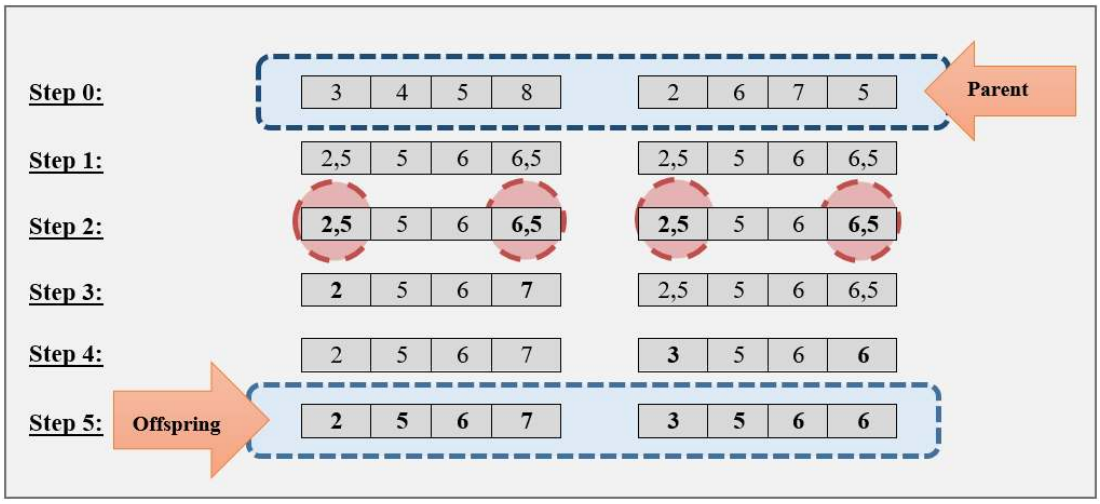


Figure 4.4 Adopted crossover procedure.

The mutation operation maintains the genetic diversity from one generation of a population to the next. In the GAA adopted, all of the individuals in the population generated after the crossover are subject to mutation with a probability of mutation (P_m). Similarly to the crossover procedure, the same mutation procedure developed by Liu & Tu (1994) and also adopted by Vergara & Kim (2009) is used in the proposed hybrid approach. The implementation of the mutation operation proceeds in three steps (Vergara & Kim, 2009):

Step 1: Select a location pair (i, j) at random, $1 \leq i \leq W, 1 < j < W, i \neq j$;

Step 2: For all $k \neq i, k \neq j \rightarrow a_k' = a_k$;

Step 3: If $a_i > 0$, then $a_i' = a_i - 1$ and $a_j' = a_j + 1$

If $a_i = 0, a_i > 0$, then $a_i' = a_i + 1$ and $a_j' = a_j - 1$

If $a_i = a_j = 0$, then $a_i' = 0$ and $a_j' = 0$.

Figure 4.5 illustrates the mutation procedure.

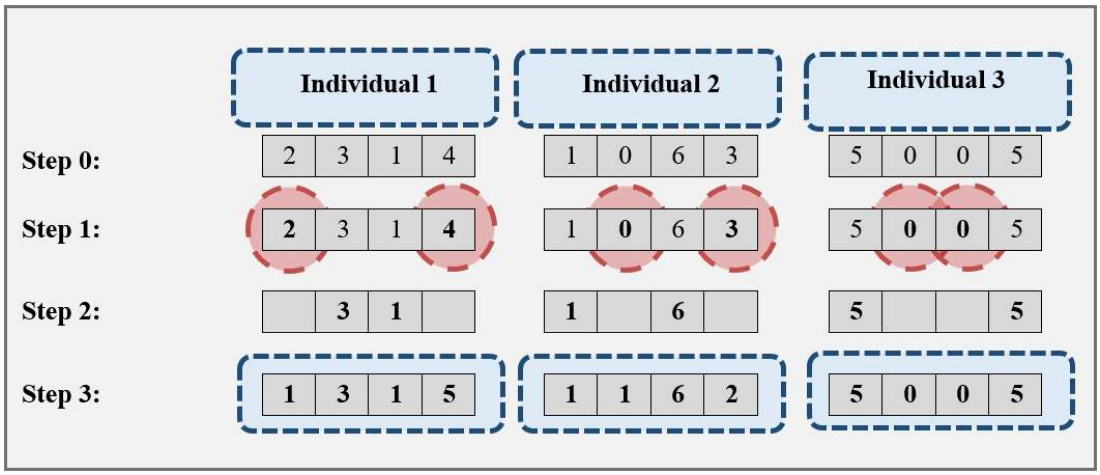


Figure 4.5 Adopted mutation procedure.

4.3.1.4 SA-based Replacement Scheme

The key idea underlying the replacement scheme is to determine which buffer configuration should stay in a population and which buffer configuration should transfer to the next generation. As the replacement scheme, the acceptance test, similar to that in the SA procedure, is adopted with the selection of individuals from the current population and the new individuals formed by crossover and mutation to the proposed hybrid approach. Thus, the best buffer configurations and the other configurations, which have acceptably average production rates, are selected for the next generation. The replacement test is designed with the change in cost (*i.e.*, average production rate) (ΔE) and the acceptance probability, which is determined by a temperature control parameter (*initial temperature*, T_0) that decreases during the SA procedure.

The replacement procedure can be summarized as follows:

Replacement Procedure:

calculate the change in cost, ΔE

loop *pop_size* is satisfied

if the change in cost less than zero (ΔE<0) then

accept the offspring to the new population

elseif exp(-$\Delta E/T$)> rand (0..1) then

accept the offspring to the new population

else

accept the individual from the *Parent Population* to the new population

end

end

4.3.1.5 Termination Criterion

In search algorithms, setting a fixed number of generations is the most commonly used termination criterion. Likewise in this study, the procedure continues for a fixed number of generations observing the general behaviour of GAA in that maximum and average fitness values keep improving in successive generations, similarly to the study by Bulgak, Diwan, & Inozu (1995). When the number of generations created is equal to the prespecified number, the relative difference between maximum fitness and average fitness is checked. If a fair amount of agreement between maximum and average fitness values is observed (the difference is less than 0.5%), the search of solution space is terminated. Otherwise, the algorithm proceeds with further generations. Finally, the best buffer configuration that leads to the best average production rate is selected to solve the BAP in the production system.

4.4 Computational Study

In this section, the experimental studies carried out to show the performance of the proposed hybrid approach are presented in four parts. In the first part, an experimental design is configured to identify the best values/scheme for the GAA control parameters to increase the search performance of the proposed approach. In the second part, a comparative study is conducted to test the performance of the proposed hybrid algorithm (GAA) against the other approaches in the literature. In the third part, an experimental study aimed at investigating the power of the hybridization in various open serial line designs is performed. In the last part, very large lines, such as 50-, 60-, 80- and 100-workstation production lines, are considered to demonstrate the efficacy of the proposed GAA. In all experiments, discrete event simulation models are developed by using the simulation language

Arena V10.0 to estimate the average production rate as an evaluative tool. Search algorithms in all test cases are implemented in Matlab V7.6. The experiments are performed on a PC with a 2-GHz Pentium (R) 4 CPU processor with 2 GB of RAM.

4.4.1 GAA Parameter Setup

Identifying the best parameter values to ensure high performance for the GAA in solving a problem is definitely necessary task throughout the search process. Since in our proposed hybrid approach, GAA combines the specific features of genetic algorithm and simulated annealing algorithm, it should be noted that the parameters used in both algorithms are taken into consideration to discover the best values of them. Although both algorithms (GA and SA) seem to be robust algorithms which contain same operators and have same algorithmic logic for different applications, in fact these algorithms are significantly different for distinct problems. The main reason is that the algorithms have several parameters and any combination of these parameters has different impacts on the performance of the algorithm. As a result, an efficient GAA parameter setup plays a crucial role on the quality of solution and convergence speed of application.

The GAA can be defined by the control parameters as:

- population size (*pop_size*),
- crossover probability rate (P_c),
- mutation probability rate (P_m),
- initial temperature (T_0),
- cooling rate (c).

As far as the GA parameters are concerned for the hybrid algorithm, the factor levels used in the BAP literature are investigated at first. Goldberg (1989) suggested that the choice of high-crossover (0.6-0.9) and low-mutation probabilities (0.01-0.1) and a moderate population size (50-100) result a good GA performance. In this respect, these control parameters (i.e., population size, crossover probability rate and mutation probability rate) are configured for the experimental design. For the

population size, it is the fact that as the population size increase, the use of simulation as the evaluative tool results more computational time. Hence, the levels are set to 50 and 80 as seen in Table 4.1. For crossover probability rate, the commonly used levels, 0.6 and 0.8 are chosen as the factor levels (See in Table 4.1). The levels of mutation probability rate are decided as 0.033 that is the commonly used value for BAP in the literature and 0.01 which is the lower bound suggested by Goldberg (1989) (See in Table 4.1).

Based on the SA parameters used in the BAP literature, since there is no significant difference between the levels, the commonly used parameter values (T_0=0.5, c=0.90) are similarly set for the hybrid algorithm. In general for SA algorithm, the key idea behind simulated annealing is to minimize the cost function using an appropriately defined annealing schedule that "cools" the system slow enough to find the optimal solution (Vigeh, 2011). Hence, the cooling schedule as a decrement function for lowering the initial temperature is an important scheme for the algorithm's performance. For this purpose, we attempt to investigate the effects of two commonly used schedules namely "*exponential schedule*" and "*geometric schedule*" in the experimental design. Regarding to these control parameters of GAA, we employ a factorial experimental design with four fixed factors having two levels. All four factors and their levels are summarized in Table 4.1.

Table 4.1 Experimental factors and levels for GAA

Factors	**Levels**	
Population size	50	80
Crossover probability rate	0.60	0.80
Mutation probability rate	0.01	0.033
Cooling schedule	Exponential ($T_k=c^k*T_0$)	Geometric ($T_{k+1}=c*T_k$)

Further, the possible intractions between factors are also of interest. To permit the detection of a possible interaction of factor effects, a 2^4 full factorial experimental layout is used to carry out the experiments (See Table 4.2).

Table 4.2 2^4 full factorial experimental layout

Experiment No	Population size	Crossover probability rate	Mutation probability rate	Cooling schedule
1	50	0.60	0.010	exponential
2	50	0.60	0.010	geometric
3	50	0.60	0.033	exponential
4	50	0.60	0.033	geometric
5	50	0.80	0.010	exponential
6	50	0.80	0.010	geometric
7	50	0.80	0.033	exponential
8	50	0.80	0.033	geometric
9	80	0.60	0.010	exponential
10	80	0.60	0.010	geometric
11	80	0.60	0.033	exponential
12	80	0.60	0.033	geometric
13	80	0.80	0.010	exponential
14	80	0.80	0.010	geometric
15	80	0.80	0.033	exponential
16	80	0.80	0.033	geometric

Test problem is chosen from the previous research by Vergara & Kim (2009) for the experiments. The manufacturing system under consideration consists of 10 workstations and nine buffers, which are located between the stations. The total buffer size (K) is set to 30. Workstations are subject to failures, and mean time between failures ($MTBF$) and mean time to repair ($MTTR$) are exponentially distributed with 20 and 2 minutes, respectively. All workstations have processing rate of 1.00 job/minute and coefficient of variations for the total time a job spends in a workstation with 0.575. The performance measure considered throughout the analysis is the average production rate.

At each parameter setting, each experiment is replicated three times to determine the variation in the results. Hence, there are a total of 48 experiments. In order to determine which control parameter effects are significant, a statistical analysis of

variance (ANOVA) is performed in MINITAB 14.0. Table 4.3 shows the estimated effects and coefficient for the average production rate.

Table 4.3 Factorial Fit: Mean versus Population size, P_c, P_m, and cooling schedule

Term	Effect	Coef	SE Coef	T	P
Constant		0.723058	0.000354	2044.62	0.000
Population size	0.000665	0.000332	0.000354	0.94	0.354
P_c	**0.001500**	**0.000750**	**0.000354**	**2.12**	**0.042**
P_m	-0.000395	-0.000198	0.000354	-0.56	0.580
Cooling schedule	-0.000320	-0.000160	0.000354	-0.45	0.654
Population size*P_c	-0.000671	-0.000336	0.000354	-0.95	0.350
Population size*P_m	0.000208	0.000104	0.000354	0.29	0.771
Population size*Cooling schedule	0.000236	0.000118	0.000354	0.33	0.740
P_c*P_m	0.000859	0.000429	0.000354	1.21	0.233
P_c*Cooling schedule	0.000502	0.000251	0.000354	0.71	0.483
P_m*Cooling schedule	-0.000226	-0.000113	0.000354	-0.32	0.751
Population size*P_c*P_m	-0.000680	-0.000340	0.000354	-0.96	0.343
Population size*P_c*Cooling schedule	-0.000592	-0.000296	0.000354	-0.84	0.409
Population size*P_m*Cooling schedule	0.000330	0.000165	0.000354	0.47	0.644
P_c*P_m*Cooling schedule	0.000602	0.000301	0.000354	0.85	0.401
Population size*P_c*P_m* Cooling schedule	-0.000492	-0.000246	0.000354	-0.70	0.492

From Table 4.3, it is clearly seen that none of the possible interactions are significant, and that only the crossover probability rate (P_c), is statistically significant; that is, its p-value is less than 0.05 (See also Figure 4.6). This suggests that in order to increase the performance of GAA in seeking the maximum fitness function, we may consider adjusting the control parameter of the crossover probability rate one at a time.

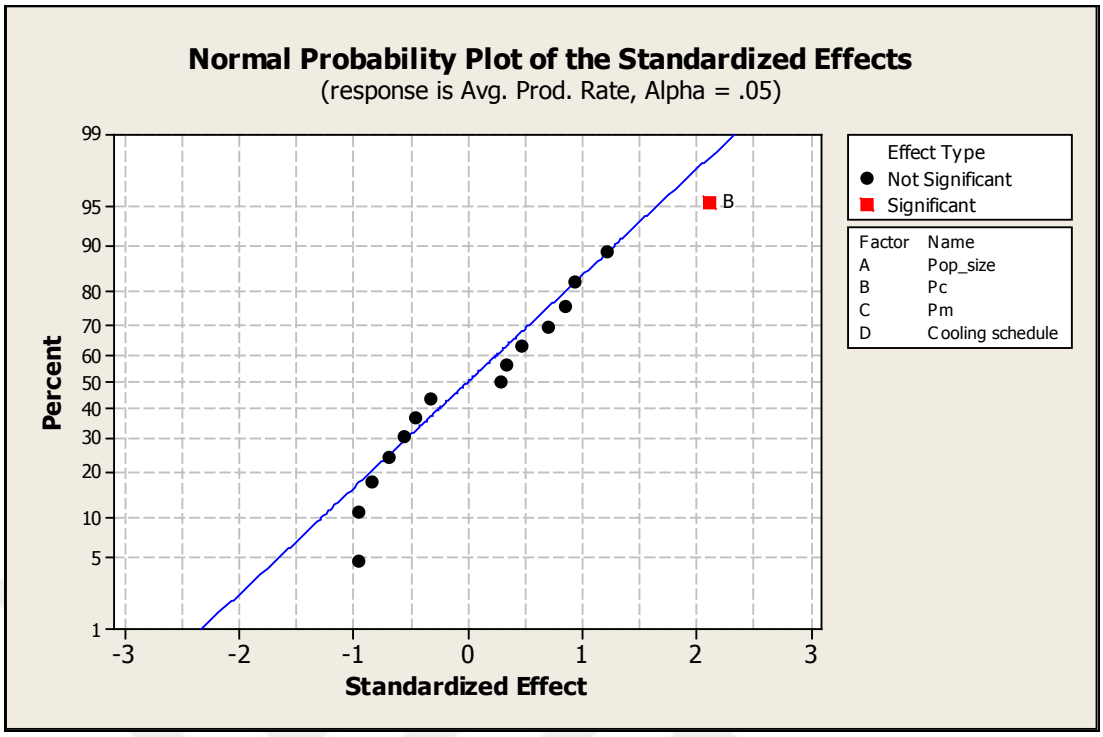


Figure 4.6 Normal probability plot.

As the results of ANOVA, we noted that understanding how the performance of GAA is influenced by the crossover probability rate is an important issue. A higher crossover rate allows exploration of more of the solution space and reduces the chances of settling for a false optimum; but if this rate is too high, it results in the wastage of a lot of computation time in exploring unpromising regions of the solution space (Gen & Cheng, 1997). As far as the main effects are concerned, Figure 4.7 depicts the plots of the four main effects to assist in the practical interpretation of this experiment. It should be noted that using the value of 0.8 for the crossover probability rate gives more average throughput rate than the value of 0.6.

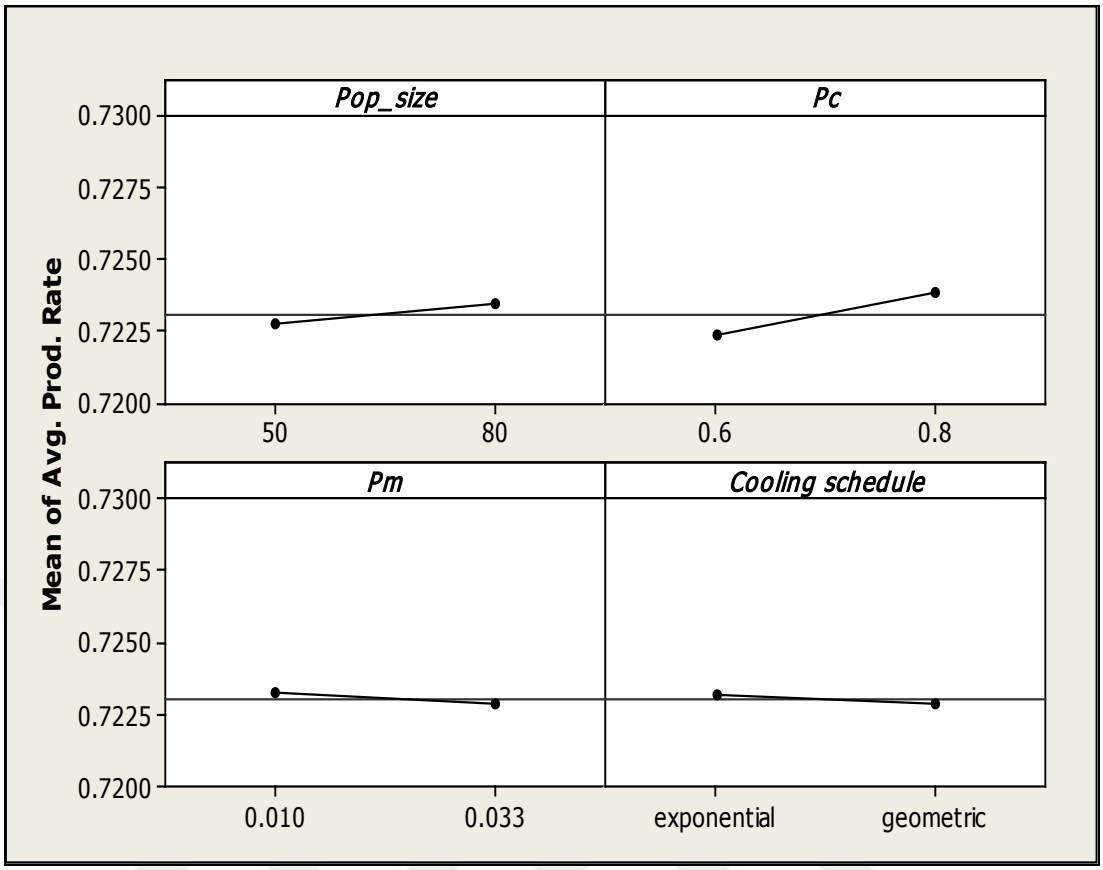


Figure 4.7 Main effects for the mean of average production rate

For the mutation probability rate, if this rate is too low, many genes that would have been useful are never tried out; but if it is too high, there will be much random perturbation, the offspring will start losing their resemblance to the parents, and the algorithm will lose the ability to learn from the history of the search (Gen & Cheng, 1997). Since the mutation probability rate is not statistically significant factor like the cooling schedule, the level of the mutation probability rate and the cooling schedule are set to 0.033 and geometric cooling schedule, respectively considering the BAP literature. Additionally, it is observed that if the population size is increased from 50 to 80, while P_c and P_m are held at their selected levels, the average throughput rate is not noticeably increased. Since large populations incur severe computational difficulties in simulation modelling, the *pop_size* is set to 50 in this study.

In summary, based on these results, it is possible to recommend the values of *pop_size*=50, P_c=0.80, P_m=0.033 and geometric cooling schedule as efficient control parameters of GAA.

4.4.2 Comparative Experiments on Benchmark Problems

In this section, the proposed hybrid GAA is compared to the approaches in the literature. Although the BAP in serial production lines has been studied to a large extent, available input data for simulation and numerical results pertinent to our comparison have rarely been reported (Demir et al., 2011; Gershwin & Schor, 2000; Ho, Eyler, & Chien, 1979; Nahas et al., 2006; Shi & Men, 2003). The experimental studies performed are classified into the three subgroups; small production lines, medium production lines, and large production lines. Throughout the experiments discussed in this section, the proposed hybrid approach and the approaches of these studies are compared to each other based on solution quality. To make a meaningful comparison, we first validate our simulation models by comparing our simulation results to those presented in the related studies. It should be noted that all simulation runs are conducted under the same experimental conditions. In all sets of problems, the processing times of the workstations are assumed to be same and constitute one time unit. Moreover, the workstations are subject to breakdown, and the failure (p_i) and repair (r_i) rates of the workstations are geometrically distributed. The main objective is to maximize the average production rate of the line with a certain fixed number of buffers.

4.4.2.1 Small Production Lines

Here, we consider small lines with W stations (W=5, 9 and 10). For a five-workstation production line, the dataset given in Table 4.4 is initially proposed by Ho et al. (1979) and used by Gershwin & Schor (2000) and Demir et al. (2011). The simulation model is run for 50 runs of 100,000 cycles each, as indicated in the study by Gershwin & Schor (2000).

Table 4.4 Dataset for a five-workstation line (Ho et al., 1979)

Workstation	**1**	**2**	**3**	**4**	**5**
$MTTR=1/r_i$	11	19	12	7	7
$MTBF=1/p_i$	20	167	22	22	26

As seen from Table 4.5, comparative results show that the GAA exhibits a slightly better production rate value than that indicated by Ho et al. (1979) and shows the same production rate value as that reported by Gershwin & Schor (2000) and Demir et al. (2011). The detailed results obtained by our approach are given in the table in Appendix A1. Ho et al. (1979) used an iterative simulation based algorithm. While Gershwin & Schor (2000) employed a gradient-based search algorithm, Demir et al. (2011) used a tabu search-based approach (DTL-TS).

Table 4.5 Comparative results for a five-workstation production line

Case	**Buffer Size Configuration**				***K***	**Average Production Rate**
Ho et al. (1979)	5	11	8	7	31	0.4914
Gershwin & Schor (2000)	7	10	10	4		0.4943
Demir et al. (2011)	7	10	10	4		0.4943
GAA	**7**	**10**	**10**	**4**		**0.4943**

For a nine-workstation production line, the dataset was also tested by Shi & Men (2003) and Demir et al. (2011). The failure (p_i) and repair (r_i) rates for all workstations are given in Table 4.6. Moreover, the total buffer size (*K)* is set to 160. The simulation model is run for 10 replications to estimate the average production rate, as indicated in the study by Shi & Men (2003). Shi & Men (2003) hybridized the tabu search with the nested partitions method for solving the BAP. Similar to previous test cases, Demir et al. (2011) used the tabu search-based approach (DTL-TS) as a search algorithm. As shown in Table 4.6, comparative results indicate that the proposed GAA generates the best buffer allocation for the maximum production rate in six of nine instances.

Table 4.6 Comparative results for a nine-workstation production line

Parameters			Average Production Rate			
p_i	r_i	K	TS (Shi & Men, 2003)	NP/TS (Shi & Men, 2003)	DTL-TS (Demir et al., 2011)	GAA
0.3	0.05	160	0.108143	0.108143	0.108147	0.107153
0.3	0.10		0.197546	0.200250	0.200255	0.199992
0.3	0.20		0.345491	0.345491	0.345495	**0.347259**
0.3	0.30		0.452074	0.452074	0.452077	**0.459709**
0.3	0.40		0.528466	0.532002	0.532006	0.531380
0.4	0.05		0.088777	0.088777	0.088782	**0.089313**
0.4	0.10		0.166232	0.166232	0.166236	**0.167459**
0.4	0.20		0.293041	0.293041	0.293046	**0.305315**
0.4	0.30		0.388517	0.390814	0.390819	**0.429878**

For a 10-workstation production line, Nahas et al. (2006) and Demir et al. (2011) used the dataset presented in Table 4.7. Moreover, the total number of buffer sizes to be allocated among production lines is 270. The simulation model is run for 10 replications.

Table 4.7 Dataset for a 10-workstation line

Workstation	1	2	3	4	5	6	7	8	9	10
$MTTR=1/r_i$	7	7	5	10	9	14	5	8	10	10
$MTBF=1/p_i$	20	30	22	22	25	40	23	30	45	20

We also provide a comparison between the performance of the GAA, degraded ceiling method employed in Nahas et al. (2006) and tabu search-based approach (DTL-TS) used in Demir et al. (2011). As seen in Table 4.8, the proposed GAA reaches the best average production rate value occurs with the buffer size configuration of B = {7, 16, 48, 61, 24, 41, 20, 34, 19} after 19 generations. While Nahas et al. (2006) observed the best average production rate value of 0.64135 with 14,000 iterations, Demir et al. (2011) reached the same results after 78 iterations. Thus, the GAA produces an approximately 1% lower average production rate over only 19 generations. All the iteration results by GAA are summarized in Appendix A2.

Table 4.8 Comparative results for a 10-workstation production line

Case	Buffer Size Configuration									*K*	Iteration	Average Production Rate
Nahas et al. (2006)	14	19	30	54	45	27	23	24	34	270	14,000	0.64135
Demir et al. (2011)	14	19	30	54	45	27	23	24	34		78	0.64135
GAA	**7**	**16**	**48**	**61**	**24**	**41**	**20**	**34**	**19**		**19**	**0.63016**

4.4.2.2 Medium Production Lines

In production systems, medium line is generally considered with 12 stations up to 30 stations (Papadopoulus et al., 2013). In this case, we consider a 20-workstation production line as a test problem used by Demir et al. (2011). As indicated in the study by Demir et al. (2011), it is assumed that the workstations have the same probability and failure rates, *i.e.* $p_i = r_i = 0.1$ up to $p_i = r_i = 0.9$ in increments of 0.1. Furthermore, the total number of buffer sizes to be allocated among production lines is set to 100. Similarly in the study by Demir et al. (2011), the simulation model of the production line is run for five iterations. Table 4.9 shows the dataset and the experimental results with respect to both the production rate and buffer size configuration, and it can be concluded that the proposed hybrid approach finds slightly better average production rates with the given buffer configurations in seven of nine instances. Demir et al. (2011) did not report the best buffer size configuration. As a result, our proposed approach has great potential for solving the BAP in medium production lines.

4.4.2.3 Large Production Lines

Large production line often consists of at least 40 workstations (Papadopoulus et al., 2013). Hence, this case involves a 40-workstation production line. Similar to the previous case, we use the same dataset implemented by Demir et al. (2011) to compare the performance of the three approaches: simple TS, DTL-TS and GAA since there is no more dataset specified for large lines to solve the BAP. Likewise, in

the study of Demir et al. (2011), the workstations have the same probability and failure rates and the total buffer size to be allocated (K) is set to 200.

Moreover, the simulation model has been run five times. Table 4.10 summarizes the comparative results with respect to the test parameters, average production rate and buffer size configuration obtained by the GAA. It can be observed from the table that in seven instances, the proposed GAA finds relatively better average production rates than those achieved using the simple TS and DTL-TS approaches. Hence, it should be noted that our approach could have a great advantage in solving the BAP in large production lines.

As a result, in all of the cases, the proposed hybrid approach is able to provide better, similar or significant average production rates that make the approach competitive. It appears that the GAA's search mechanism has some potential to solve BAPs.

Table 4.9 Comparative results for a 20-workstation production line

Parameters			Average Production Rate			Buffer Size Configuration		
p_i	r_i	K	Simple TS (Demir et al., 2011)	DTL-TS (Demir et al., 2011)	GAA	Simple TS (Demir et al., 2011)	DTL-TS (Demir et al., 2011)	GAA
0.1	0.1		0.233963	0.234191	0.226499	N.A.	N.A.	B={4,6,4,5,4,5,6,5,8,3,4,5,4,4,6,8,5,8,6}
0.2	0.2		0.296989	0.297025	**0.302448**	N.A.	N.A.	B={4,4,3,8,4,7,5,7,5,5,5,4,5,4,5,10,4,6,5}
0.3	0.3		0.334450	0.334450	**0.349517**	N.A.	N.A.	B={3,5,5,6,3,6,6,5,8,4,4,4,4,4,5,8,5,8,7}
0.4	0.4		0.359637	0.359637	0.354299	N.A.	N.A.	B={3,5,5,5,5,4,5,4,5,10,4,4,5,4,3,8,7,7,7}
0.5	0.5	100	0.377842	0.377895	**0.382401**	N.A.	N.A.	B={4,5,4,6,4,5,5,5,8,4,2,4,5,4,9,7,6,7,6}
0.6	0.6		0.391712	0.391846	**0.397120**	N.A.	N.A.	B={4,5,4,6,4,5,5,5,8,4,4,4,4,4,8,7,6,7,6}
0.7	0.7		0.402760	0.402760	**0.446904**	N.A.	N.A.	B={4,5,4,5,5,5,5,5,8,3,4,5,4,4,7,8,5,8,6}
0.8	0.8		0.411629	0.411638	**0.450032**	N.A.	N.A.	B={4,5,5,6,2,6,7,5,7,3,4,4,5,4,4,8,5,9,7}
0.9	0.9		0.419017	0.419018	**0.459111**	N.A.	N.A.	B={4,5,5,6,2,6,7,5,7,3,4,4,5,4,4,8,6,8,7}

Table 4.10 Comparative results for a 40-workstation production line

Parameters			Average Production Rate			Buffer Size Configuration		
p_i	r_i	K	Simple TS (Demir et al., 2011)	DTL-TS (Demir et al., 2011)	GAA	Simple TS (Demir et al., 2011)	DTL-TS (Demir et al., 2011)	GAA
0.1	0.1		0.219060	0.221273	0.220132	N.A.	N.A.	B={8,3,7,7,4,4,5,4,4,6,3,7,10,4,5,4,4,4,6,4,8,5,6,4,4,6,3,3,3,3,4,6,5,6,5,8,8,3,7}
0.2	0.2		0.286347	0.282169	0.270289	N.A.	N.A.	B={4,2,8,5,4,5,6,6,3,4,5,6,12,5,3,8,7,2,4,4,9,3,5,5,4,6,4,3,3,3,4,5,11,7,5,7,7,2,4}
0.3	0.3		0.325256	0.325256	**0.328518**	N.A.	N.A.	B={8,4,7,7,4,5,4,6,4,4,3,7,7,5,4,7,5,3,6,3,9,5,6,3,5,4,3,3,4,3,4,7,3,6,5,4,12,4,7}
0.4	0.4		0.351391	0.351494	**0.365301**	N.A.	N.A.	B={5,7,4,7,5,5,3,8,5,3,5,6,7,4,4,6,8,5,4,6,4,4,5,5,4,5,3,6,6,4,4,3,5,8,4,7,4,4,8}
0.5	0.5	200	0.370653	0.370666	**0.371244**	N.A.	N.A.	B={3,5,4,4,3,3,4,7,3,1,10,1,11,8,3,4,0,4,3,9,8,0,11,6,10,5,0,8,9,2,7,0,9,15,1,8,0,3,8}
0.6	0.6		0.385290	0.385328	**0.404937**	N.A.	N.A.	B={2,6,4,4,3,3,4,7,3,1,10,1,11,8,3,4,0,4,3,9,8,6,5,6,10,5,0,8,9,2,7,0,9,15,1,8,0,3,8}
0.7	0.7		0.396903	0.397255	**0.446904**	N.A.	N.A.	B={5,4,3,5,11,7,1,1,7,1,8,10,8,3,1,2,1,6,5,12,13,7,5,1,1,3,14,2,11,5,1,5,2,3,11,6,1,8,0}
0.8	0.8		0.406232	0.406559	**0.468270**	N.A.	N.A.	B={5,7,4,7,5,5,3,8,5,3,5,6,7,4,3,7,8,5,4,6,4,4,5,5,4,5,3,6,6,4,4,3,5,8,4,7,4,4,8}
0.9	0.9		0.413541	0.413990	**0.482113**	N.A.	N.A.	B={11,7,4,5,2,2,2,5,2,24,1,16,7,11,0,4,4,5,5,1,16,2,4,10,4,4,0,5,2,2,3,3,3,1,9,4,8,0,2}

4.4.2.4 Comparative Experiments under Different Serial Line Conditions

As mentioned earlier, the GAA combines the strengths of GA and SA. To demonstrate the power of the hybridization, in this section, we compare the proposed hybrid GAA to traditional GA, and traditional SA algorithm focusing on various open serial line designs to solve the BAPs. In the experiments, the flow of the traditional GA and its parameters are similar to those of the hybrid algorithm. The only difference in the GA is the replacement test, in which an elitism scheme is used. Once the best parameter values have been identified with experimental designs, the same parameter values are considered for both the GAA and the GA, *i.e., population size=50,* P_c=0.80, P_m=0.033 and *generation number*=20. Additionally, the SA algorithm developed for this study features the same parameter settings as those implemented by Spinellis & Papadopoulos (2000): initial temperature (T_0)=0.5, cooling rate (c)=0.90, maximum number of trials at a given temperature=100, and maximum number of successes at a given temperature=10. Moreover, the cooling schedule is the same as that used in the hybrid procedure, and the procedure continues until a final temperature of less than 0.001 or the maximum number of successes at a given temperature equals zero.

Test cases considered the type of workstations in the system and the characteristics of the workstations in four general types of open production lines: balanced, unbalanced, reliable and unreliable. The parameters such as the length of the line, total buffer capacity available, number of bottlenecks, position of bottleneck(s), and upper bounds for each buffer varied within each production line type as seen in Table 4.11.

Table 4.11 Summary of the production line types assumpted and parameters varied within each production line type

Type	Reliable	Unreliable
Balanced	length of the line total buffer capacity available upper limitations for each buffer size	length of the line total buffer capacity available upper limitations for each buffer size workstation variability
Unbalanced	number of bottlenecks total buffer capacity available* position of bottleneck(s) severity of bottleneck(s)	number of bottlenecks total buffer capacity available* position of bottleneck(s) severity of bottleneck(s)

*Only considered for lines with one bottleneck.

The experimental studies consist of ten test cases. The original information regarding the problem specifications for the test cases, except test cases 2 and 7, are presented by Vergara & Kim (2009). In test cases 2 and 7, an additional constraint on the upper bound for each buffer capacity is considered to allow for a more realistic environment in production systems. In all ten test cases, the blocking mechanism is the BAS (Blocking After Service) type. Moreover, simulation models are developed and run for 100,000 jobs with 10 replications to estimate the average production rate as an evaluative tool.

Each test case is explained in detail as follows. To compare the performance of the proposed hybrid GAA to the traditional SA and GA, the computational results are summarized in the tables. For all tables, the first column (K) in these tables depicts the total buffer size available in the serial line. The average production rates are presented in the second, third and fourth columns for SA, GA and GAA, respectively. The generated buffer allocation for SA is shown in the fifth column of the tables. For GA, the allocation is presented in the sixth column, and for GAA, it is shown in the seventh column of the tables.

Test Case 1: We first consider the case where the number of buffers and the line length are varied (5, 10, 15, 20, 30, and 50 buffers allocated in 5, 10 and 15-

workstation lines) in balanced reliable production lines. Processing times follow the exponential distribution with a mean of 1.0 time unit (minutes) for all workstations. The experimental results are given in Tables 4.12 - 4.14.

Table 4.12 Test case 1 - Results for a five-workstation balanced reliable production line

K	Average Production Rate			Buffer Size Configuration		
	SA	GA	GAA	SA	GA	GAA
10	0.7054	0.7090	0.7090	{2,3,2,3}	{2,3,3,2}	{2,3,3,2}
15	0.7252	0.7251	0.7252	{1,6,6,2}	{2,6,6,1}	{1,6,6,2}
20	0.7943	0.7943	0.7943	{5,5,5,5}	{5,5,5,5}	{5,5,5,5}
30	0.8334	0.8403	0.8403	{6,8,9,7}	{7,8,8,7}	{7,8,8,7}

Table 4.13 Test case 1 - Results for a 10-workstation balanced reliable production line

K	Average Production Rate			Buffer Size Configuration		
	SA	GA	GAA	SA	GA	GAA
5	0.4926	0.4965	0.4965	{0,1,0,1,0,1,0,1,1}	{1,0,1,0,1,0,1,0,1}	{1,0,1,0,1,0,1,0,1}
10	0.5604	0.5604	0.5604	{1,1,1,1,2,1,1,1,1}	{1,1,1,1,2,1,1,1,1}	{1,1,1,1,2,1,1,1,1}
15	0.6119	0.6158	0.6153	{1,1,2,2,3,2,2,1,1}	{1,2,2,2,2,2,2,1,1}	{1,1,2,2,2,2,2,2,1}
20	0.6512	0.6344	**0.6538**	{2,2,2,2,4,2,2,2,2}	{1,1,2,3,5,3,2,2,1}	**{2,2,2,3,2,3,2,2,2}**
30	0.6442	0.6774	**0.7071**	{1,4,3,5,5,2,3,4,3}	{2,1,5,4,6,4,1,5,2}	**{3,3,4,5,4,5,3,3,0}**
50	0.7756	0.7729	**0.7785**	{6,5,6,5,6,5,6,5,6}	{6,6,6,5,5,5,5,6,6}	**{5,5,6,6,6,6,6,5,5}**

Table 4.14 Test case 1 - Results for a 15-workstation balanced reliable production line

K	Average Production Rate			Buffer Size Configuration		
	SA	GA	GAA	SA	GA	GAA
5	0.4411	0.4594	**0.4601**	{0,1,0,1,0,1,0,1,1,0,0,0,0,0}	{0,1,0,0,0,1,2,1,0,0,0,0,0,0}	**{1,0,0,1,0,0,1,0,0,1,0,0,1,0}**
10	0.4806	0.4947	0.4947	{1,0,0,1,1,1,1,1,1,1,1,1,0,0}	{1,0,1,1,1,1,1,1,1,1,1,0,0,0}	{1,0,1,1,1,1,1,1,1,1,1,0,0,0}
15	0.5489	0.5563	0.5563	{2,1,1,1,1,1,1,1,1,1,1,1,1,1}	{1,1,1,1,1,1,2,1,1,1,1,1,1,1}	{1,1,1,1,1,1,2,1,1,1,1,1,1,1}
20	0.5632	0.5721	**0.5802**	{2,1,2,0,0,3,2,3,1,2,1,0,1,2}	{2,2,2,1,2,3,3,2,1,0,0,1,0,1}	**{2,1,1,1,1,2,3,3,1,1,1,1,1,1}**
30	0.6061	0.6134	**0.6245**	{1,0,1,3,2,4,3,3,4,2,2,3,0,2}	{2,2,2,2,2,2,3,3,2,2,2,2,2,2}	**{1,2,2,2,2,3,3,3,2,2,2,2,2,2}**
50	0.6953	0.7059	**0.7066**	{4,4,4,4,3,3,3,3,3,3,4,4,4,4}	{3,4,3,4,4,3,4,4,3,4,4,3,4,3}	**{3,3,3,4,4,4,4,4,4,4,4,4,3,3}**

Test Case 2: This case examines buffer allocation in reliable production lines. Unlike the test case 1 generated, this test case involves an additional constraint for BAP. For environmental reasons such as layout limitations, each buffer size has an upper value ($0 \leq B_i \leq U_i$, i = 1, 2, …, W-1). Hence, U_i denotes the upper limit of buffer sizes. The number of buffers and line length are varied (20 and 30 buffers allocated in 10 and 15-workstation lines). Workstations have exponentially distributed processing times with a mean of 1.0 minute. For 10-workstation production line, upper bounds of each buffer are 1, 5, 10, 5, 5, 5, 5, 1 and 5, respectively. It is assumed for 15-workstation production line that upper bounds of each buffer are 1, 1, 5, 10, 10, 10, 5, 5, 5, 5, 1, 1, 5 and 5, respectively. The results for the test case 2 are summarized in Tables 4.15 and 4.16.

In test cases 1 and 2, the different total numbers of buffers have been allocated in various types of balanced production lines of different lengths. In 9 of 20 instances, GAA and either the SA or the GA or both find the same buffer allocation maximizing production rate. The proposed hybrid GAA produces a higher production rate in 10 instances, whereas the GA is better in only one instance. The results show that the performance of the GAA is good for long production lines and large buffers. When analyzing buffer allocation for balanced production lines, for test case 1, in most cases in which even buffer allocations cannot be possible, buffers are allocated evenly to all sites, and any remaining capacity is allocated symmetrically from the center out. This phenomenon depicts the bowl phenomenon proposed by Hillier & Boling (1966, 1977, 1979). Additionally, the bowl phenomenon has been observed in test case 2 under the limited buffer capacity.

Table 4.15 Test case 2 - Results for a 10-workstation balanced reliable production line with boundary constraint

K	**Average Production Rate**			**Buffer Size Configuration**		
	SA	**GA**	**GAA**	**SA**	**GA**	**GAA**
20	0.6445	0.6448	**0.6451**	{1,2,3,3,2,3,2,1,3}	{1,3,3,3,2,3,2,1,2}	**{1,3,2,3,2,3,2,1,3}**
30	0.6857	0.6865	0.6865	{1,4,4,4,4,4,4,1,4}	{1,3,4,4,5,4,4,1,4}	{1,3,4,5,4,4,4,1,4}

Table 4.16 Test case 2 - Results for a 15-workstation balanced reliable production line with boundary constraint

K	Average Production Rate			Buffer Size Configuration		
	SA	**GA**	**GAA**	**SA**	**GA**	**GAA**
20	0.5632	0.5778	**0.5785**	{1,1,1,1,2,2,2,2,2,2,1,1,1,1}	{1,1,2,2,1,1,2,1,2,1,1,1,2,2}	**{1,1,2,1,2,1,2,2,2,1,1,1,2,1}**
30	0.6177	0.6167	**0.6201**	{1,1,2,2,3,3,3,3,3,3,1,1,2,2}	{1,1,2,2,2,2,3,3,3,3,1,1,3,3}	{1,1,3,2,3,2,3,3,3,2,1,1,3,2}

Test Case 3: In this case, buffers are allocated in reliable production lines with one bottleneck. The allocation of two and eight buffers in an eight-workstation unbalanced reliable line is considered. The position of the bottleneck is shifted successively from W_2 to W_3 and then to W_4. Moreover, it is assumed that all processing times are exponentially distributed. Non-bottleneck workstations have a mean processing time of 1.0 minute, while bottleneck mean processing times of 1.1, 1.5 and 2.0 minutes are examined. The results for this test case are given in Tables 4.17-4.19.

Test Case 4: An unbalanced reliable production line with symmetrically positioned two bottlenecks is examined in this case. A total of eight buffers are allocated in an eight-workstation production line that has a bottleneck at workstation W_2 and another bottleneck at workstation W_6. In a test case similar to test case 3, workstations have exponentially distributed processing times with a mean of 1.0 minute for non-bottleneck workstations and mean of 1.1, 1.5 and 2.0 minutes for the bottlenecks. The comparative results are shown in Table 4.20.

Test Case 5: Buffer allocation in unbalanced reliable production lines with two bottlenecks that are asymmetrical in position is considered in this case. Eight buffers are allocated in an 8-workstation production system that has a bottleneck at workstation W_2 and another bottleneck at workstation W_4. Processing times are exponentially distributed with a mean processing time of 1.0 minute for all non-bottleneck workstations and with mean processing times of 1.1, 1.5 and 2.0 minutes for bottlenecks. For this case, all experimental results are given in Table 4.21.

In test cases 3 to 5, the buffers are allocated in reliable production lines with single and multiple bottlenecks. In 28 of 36 instances, GAA generates a similar buffer allocation with SA and/or GA. Although SA is better in one instance, GAA generates the best results in six instances. Considering the bottlenecks, an optimal buffer pattern shifts the buffer gradually toward the bottleneck through intermediate locations as the bottleneck mean increases (Powell & Pyke, 1996).

In these test cases, the buffers are allocated evenly to all sites because of similar process times for non-bottleneck workstations. The remaining capacity is also allocated to the bottleneck workstations and near bottleneck workstations to improve the system yields.

Table 4.17 Test case 3 - Results for an eight-workstation reliable production line with a single bottleneck of T=1.1

Bottleneck	*K*	Average Production Rate			Buffer Size Configuration		
		SA	GA	GAA	SA	GA	GAA
W_2	2	0.4573	0.4600	**0.4602**	{2,0,0,0,0,0,0}	{1,0,0,1,0,0,0}	**{1,0,1,0,0,0,0}**
	8	0.5251	0.5386	0.5386	{2,2,0,1,1,1,1}	{2,1,1,1,1,1,1}	{2,1,1,1,1,1,1}
W_3	2	0.4431	0.4437	0.4437	{1,1,0,0,0,0,0}	{0,2,0,0,0,0,0}	{0,2,0,0,0,0,0}
	8	0.5530	0.5461	0.5487	{1,2,1,1,1,1,1}	{0,3,1,1,1,1,1}	{1,2,2,0,1,1,1}
W_4	2	0.4368	0.4374	0.4374	{0,0,2,0,0,0,0}	{0,0,1,1,0,0,0}	{0,0,1,1,0,0,0}
	8	0.5512	0.5512	0.5512	{1,1,2,1,1,1,1}	{1,1,2,1,1,1,1}	{1,1,2,1,1,1,1}

Table 4.18 Test case 3 - Results for an eight-workstation reliable production line with a single bottleneck of T=1.5

Bottleneck	*K*	Average Production Rate			Buffer Size Configuration		
		SA	GA	GAA	SA	GA	GAA
W_2	2	0.4454	0.4512	0.4512	{1,0,0,0,0,0,1}	{1,0,0,0,0,1,0}	{1,0,0,0,0,1,0}
	8	0.5107	0.5107	0.5107	{2,1,1,1,1,1,1}	{2,1,1,1,1,1,1}	{2,1,1,1,1,1,1}
W_3	2	0.4072	0.4072	**0.4103**	{1,1,0,0,0,0,0}	{1,1,0,0,0,0,0}	**{0,1,1,0,0,0,0}**
	8	0.5145	0.5233	0.5197	{1,2,0,2,0,1,2}	{1,3,2,1,0,0,1}	{2,3,1,1,1,1,1}
W_4	2	0.4273	0.4296	0.4296	{0,1,0,1,0,0,0}	{0,0,1,1,0,0,0}	{0,0,1,1,0,0,0}
	8	0.4952	0.5071	**0.5123**	{1,2,2,1,1,1,0}	{1,0,2,1,2,1,1}	**{1,1,2,1,1,1,1}**

Table 4.19 Test case 3 - Results for an eight-workstation reliable production line with a single bottleneck of T=2

Bottleneck	*K*	Average Production Rate			Buffer Size Configuration		
		SA	GA	GAA	SA	GA	GAA
W_2	2	0.3821	0.3821	0.3821	{1,1,0,0,0,0,0}	{1,1,0,0,0,0,0}	{1,1,0,0,0,0,0}
	8	0.4455	0.4514	**0.4580**	{2,0,1,2,1,1,1}	{2,0,1,1,1,1,2}	**{2,1,1,1,1,1,1}**
W_3	2	0.3942	0.4030	0.4030	{1,1,0,0,0,0,0}	{0,2,0,0,0,0,0}	{0,2,0,0,0,0,0}
	8	0.4561	0.4605	0.4605	{1,3,0,1,1,1,1}	{0,2,2,1,1,1,1}	{0,2,2,1,1,1,1}
W_4	2	0.3804	0.3813	0.3813	{0,1,1,0,0,0,0}	{0,0,1,0,1,0,0}	{0,0,1,0,1,0,0}
	8	0.4114	0.4234	0.4234	{1,2,3,1,0,0,1}	{1,2,2,1,1,0,1}	{1,2,2,1,1,0,1}

Table 4.20 Test case 4 - Results for an eight-workstation reliable production line with two symmetrically located bottlenecks (K=8)

$T(W_6)$=1.1	Average Production Rate			Buffer Size Configuration		
$T(W_2)$	SA	GA	GAA	SA	GA	GAA
1.1	0.5541	0.5467	0.5541	{2,1,1,1,2,1,0}	{2,0,0,1,3,1,1}	{2,1,1,1,3,1,1}
1.5	0.5093	0.5125	0.5125	{2,1,1,1,2,1,0}	{3,1,1,1,2,0,0}	{3,1,1,1,2,0,0}
2.0	0.4570	0.4691	**0.4712**	{2,2,0,1,2,1,0}	{2,2,0,0,2,1,1}	**{2,1,1,1,2,1,0}**
$T(W_6)$=1.5 $T(W_2)$						
1.1	0.5240	0.5240	0.5240	{1,1,1,1,2,1,1}	{1,1,1,1,2,1,1}	{1,1,1,1,2,1,1}
1.5	0.4894	0.4954	0.4954	{2,1,1,1,1,1,1}	{1,1,1,1,2,1,1}	{1,1,1,1,2,1,1}
2.0	0.4382	0.4330	0.4382	{2,1,1,1,2,1,0}	{2,1,1,1,2,0,1}	{2,1,1,1,2,1,0}
$T(W_6)$=2.0 $T(W_2)$						
1.1	0.4497	0.4526	0.4526	{1,1,1,0,2,2,1}	{1,1,0,1,2,2,1}	{1,1,0,1,2,2,1}
1.5	0.4331	0.4389	**0.4403**	{1,1,0,1,2,1,2}	{1,1,0,1,2,2,1}	**{2,1,1,1,2,1,1}**
2.0	0.3971	0.4058	0.4058	{2,1,1,1,2,0,1}	{2,1,0,2,2,1,0}	{2,1,0,2,2,1,0}

Table 4.21 Test case 5 - Results for an eight-workstation reliable production line with two asymmetric bottlenecks (K=8)

$T(W_4)$=1.1	Average Production Rate			Buffer Size Configuration		
$T(W_2)$	SA	GA	GAA	SA	GA	GAA
1.1	0.5471	0.5684	0.5684	{3,0,2,1,1,1,0}	{1,1,2,1,1,1,1}	{1,1,2,1,1,1,1}
1.5	0.5176	0.5176	0.5176	{2,1,1,1,1,1,1}	{2,1,1,1,1,1,1}	{2,1,1,1,1,1,1}
2.0	0.4441	0.4437	0.4441	{2,1,2,1,2,1,0}	{2,1,1,1,1,1,1}	{2,1,2,0,1,1,1}
$T(W_4)$=1.5 $T(W_2)$						
1.1	0.5198	0.5198	0.5198	{1,1,2,1,1,1,1}	{1,1,2,1,1,1,1}	{1,1,2,1,1,1,1}
1.5	0.4905	0.4886	0.4905	{2,1,2,1,1,1,0}	{2,1,2,1,1,0,2}	{2,1,2,1,1,1,0}
2.0	0.4381	0.4386	0.4386	{2,2,1,1,1,0,1}	{2,1,1,1,0,1,1}	{2,1,1,1,0,1,1}
$T(W_4)$=2.0 $T(W_2)$						
1.1	0.5581	0.5447	0.5581	{1,1,2,1,1,1,1}	{2,1,2,0,1,1,1}	{1,1,2,1,1,1,1}
1.5	0.4982	0.5203	0.5203	{2,1,1,0,1,1,2}	{1,1,2,1,1,1,1}	{1,1,2,1,1,1,1}
2.0	0.4429	0.4495	0.4495	{1,1,2,2,1,1,0}	{2,1,2,1,1,1,0}	{2,1,2,1,1,1,0}

Test Case 6: This case focuses on buffer allocation in balanced unreliable production lines. The number of buffers, line length, and workstation variability are varied (5, 10, 15, 20, 30 and 50 buffers allocated in 5, 10 and 15-workstation lines). Workstations are subject to failures, and mean time between failures (*MTBF*) and mean time to repair (*MTTR*) are exponentially distributed with 20 and 2 minutes, respectively. All workstations have processing rate of 1.00 job/minute and coefficient of variations for the total time a job spends in a workstation with 0.575. The results for the test case 6 are summarized in Tables 4.22 and 4.24.

Test Case 7: This case examines buffer allocation in balanced unreliable production lines. Unlike the test case 6 generated, this test case involves an additional constraint for BAP as in the test case 2. The number of buffers and the line length are varied (20 and 30 buffers allocated in 10 and 15-workstation lines). Workstations have processing rate of 1.00 job/minute and are subject to failures. Mean time between failures (*MTBF*) and mean time to repair (*MTTR*) are set to 100 and 25 minutes, respectively and exponentially distributed. Workstations have a coefficient of variation for the total time a job spends in a workstation with 0.575. For 10-workstation production line, upper bounds of each buffer are 1, 5, 10, 5, 5, 5, 5, 1 and 5, respectively. For 15-workstation production line, upper bounds of 1, 1, 5, 10, 10, 10, 5, 5, 5, 5, 1, 1, 5 and 5, respectively for each buffer are considered as a constraint in this case. For both production lines, the experimental results are given in Tables 4.25 and 4.26.

In test case 6 and 7, different total numbers of buffers have been allocated in various types of balanced unreliable production lines in different lengths. Test case 7 also focuses on layout limitations like as upper limits of each buffer. GAA and SA and/or GA give the same buffer allocation in 11 instances of 20 instances. Our hybrid GAA approach is the best in eight instances while SA produces the best buffer allocation in only one instance. Most of the best results by GAA are observed in longer line length and larger buffer size. In view of buffer allocation, the bowl phenomenon design rule in which buffers are allocated evenly to all sites and any remaining capacity allocated symmetrically from outer center is observed and gives better results in these cases.

Table 4.22 Test case 6 - Results for five-workstation balanced unreliable production line with failures

K	Average Production Rate			Buffer Size Configuration		
	SA	**GA**	**GAA**	**SA**	**GA**	**GAA**
10	0.8069	0.8085	0.8085	{2,3,2,3}	{2,3,3,2}	{2,3,3,2}
15	0.8370	0.8363	**0.8372**	{3,4,4,4}	{4,4,4,3}	**{3,4,5,3}**
20	0.8547	0.8553	0.8553	{4,6,6,4}	{5,5,5,5}	{5,5,5,5}
30	0.8764	0.8757	0.8764	{7,8,8,7}	{7,8,7,8}	{7,8,8,7}

Table 4.23 Test case 6 - Results for 10-workstation balanced unreliable production line with failures

K	Average Production Rate			Buffer Size Configuration		
	SA	**GA**	**GAA**	**SA**	**GA**	**GAA**
5	0.6394	0.6361	0.6394	{1,0,1,0,1,0,1,0,1}	{0,0,1,1,1,1,1,0,0}	{1,0,1,0,1,0,1,0,1}
10	0.7175	0.7223	0.7223	{2,1,1,1,1,1,1,1,1}	{1,1,1,1,2,1,1,1,1}	{1,1,1,1,2,1,1,1,1}
15	0.7439	0.7491	0.7491	{2,1,2,1,2,1,2,1,2}	{1,1,2,2,3,2,2,1,1}	{1,1,2,2,3,2,2,1,1}
20	0.7814	0.7740	**0.7825**	{2,2,2,2,4,2,2,2,2}	{2,2,1,3,3,3,2,2,2}	**{2,2,2,3,3,2,2,2,2}**
30	0.8157	0.8159	0.8159	{3,3,3,4,4,4,3,3,3}	{3,4,3,4,3,4,3,3,3}	{3,4,3,4,3,4,3,3,3}
50	0.8455	0.8482	**0.8506**	{7,5,5,4,5,6,5,7,6}	{6,6,6,5,5,5,5,6,6}	**{5,5,6,6,6,6,6,5,5}**

Table 4.24 Test case 6 - Results for 15-workstation balanced unreliable production line with failures

K	Average Production Rate			Buffer Size Configuration		
	SA	GA	GAA	SA	GA	GAA
5	0.5897	0.6036	0.6036	{0,0,0,0,1,1,1,1,1,0,0,0,0,0}	{1,0,0,1,0,0,1,0,0,1,0,0,1,0}	{1,0,0,1,0,0,1,0,0,1,0,0,1,0}
10	0.6572	0.6152	0.6572	{0,1,1,0,1,1,1,1,1,0,1,1,0,1}	{0,0,0,1,1,1,2,2,1,1,1,0,0,0}	{0,1,1,0,1,1,1,1,1,0,1,1,0,1}
15	0.7126	0.7082	0.7123	{1,1,1,1,1,1,2,1,1,1,1,1,1,1}	{2,1,1,1,1,1,1,1,1,1,1,1,1,1}	{1,1,1,1,1,1,1,2,1,1,1,1,1,1}
20	0.7319	0.7221	0.7319	{1,1,2,1,1,2,2,1,2,1,1,2,1,2}	{2,1,1,1,1,1,3,3,1,1,1,1,1,2}	{1,1,2,1,1,2,2,1,2,1,1,2,1,2}
30	0.7733	0.7736	**0.7740**	{2,2,2,2,2,2,4,2,2,2,2,2,2,2}	{2,2,2,2,2,2,3,3,2,2,2,2,2,2}	**{2,2,2,2,2,3,3,2,2,2,2,2,2,2}**
50	0.7848	0.8157	**0.8161**	{4,2,2,2,5,6,5,4,6,4,4,3,2,1}	{3,3,3,4,4,4,4,4,4,4,4,3,3,3}	**{34,3,3,4,4,4,4,3,4,5,3,3,3}**

Table 4.25 Test case 7 - Results for 10-workstation balanced unreliable production line with boundary constraint

K	Average Production Rate			Buffer Size Configuration		
	SA	GA	GAA	SA	GA	GAA
20	0.7712	0.7689	**0.7843**	{1,2,3,3,3,3,2,1,2}	{1,3,3,3,3,3,1,1,2}	**{1,3,3,3,3,2,2,1,2}**
30	0.7892	0.7899	**0.7918**	{1,3,4,4,5,5,4,1,3}	{1,3,4,4,4,5,4,1,4}	**{1,4,4,4,4,4,4,1,4}**

Table 4.26 Test case 7 - Results for 15-workstation balanced unreliable production line with boundary constraint

K	Average Production Rate			Buffer Size Configuration		
	SA	GA	GAA	SA	GA	GAA
20	0.6916	0.7291	0.7291	{1,1,0,2,3,1,2,3,1,1,1,1,1,2}	{1,1,1,1,2,2,2,2,2,2,1,1,1,1}	{1,1,1,1,2,2,2,2,2,2,1,1,1,1}
30	0.7475	0.7427	**0.7542**	{1,1,2,1,2,3,4,5,2,2,1,1,3,2}	{1,1,3,2,2,2,4,5,3,2,1,1,1,2}	**{1,2,2,2,2,3,3,4,2,2,1,1,3,2}**

Test Case 8: In this case, buffer allocation in unreliable production lines with one bottleneck is examined. The allocation of two, five and ten buffers in a five-workstation unbalanced unreliable line is considered. The bottleneck is positioned at W_4. It is assumed that the workstations are subject to breakdowns and the processing rate for each workstation is one time unit. Moreover, it is assumed that the failure and repair rates of the workstations are exponentially distributed with 300 and 30 minutes, respectively. In this case, *MTTR* values of 40, 50 and 80 minutes are tested through the simulation runs. The results are given in Table 4.27.

Test Case 9: We address the buffer allocation in an unbalanced unreliable production line with two bottlenecks positioned symmetrically in this case and six buffers are allocated in a six-workstation production system that has a bottleneck at workstation W_2 and another bottleneck at workstation W_5. Processing rates for all workstations are fixed with a rate of 1.0 job/minute. In common with test cases 8, mean time between failures (*MTBF*) and mean time to repair (*MTTR*) are exponentially distributed with 300 and 30 minutes, respectively for non-bottleneck workstations whereas 40, 50, and 80 minutes are used for the bottlenecks. For this case, the comparative results are shown in Table 4.28.

Test Case 10: This case examines a buffer allocation in unbalanced unreliable production lines with two bottlenecks positioned asymmetrically. Six buffers are allocated in a six-workstation line which has a bottleneck workstation, W_2 and another bottleneck, W_4. Workstations have a fixed processing rate of 1.0 job/minute. For non-bottleneck workstations in this production line, mean time between failures (*MTBF*) and mean time to repair (*MTTR*) are exponentially distributed with 300 and 30 minutes, respectively. *MTTR* values of 40, 50, and 80 minutes are used for the bottlenecks. The experimental results are presented in Table 4.29.

In test cases 8-10, the buffers are allocated in unreliable production lines with single and multiple bottlenecks. Since buffers to the bottleneck workstations improve the production rate of the system, an even allocation of buffers to all buffer stages and remaining capacity allocated additionally to the bottleneck workstations

generates the highest production rate. The GAA finds the same production rate with the SA and/or the GA in 20 instances of 27 instances. In other seven instances, the proposed hybrid GAA generates the best buffer allocation for maximum production rate.

Table 4.27 Test case 8 - Results for five-workstation unreliable production line with a single bottleneck

Bottleneck W_2 *(MTTR)*	***K***	**Average Production Rate**			**Buffer Size Configuration**		
		SA	**GA**	**GAA**	**SA**	**GA**	**GAA**
40	**2**	0.6228	0.6229	0.6229	{0,0,2,0}	{0,1,1,0}	{0,1,1,0}
	5	0.6290	0.6290	0.6290	{1,1,2,1}	{1,1,2,1}	{1,1,2,1}
	10	0.6384	0.6397	**0.6399**	{3,2,3,2}	{2,3,3,2}	**{2,2,4,2}**
50	**2**	0.6230	0.6222	0.6230	{0,0,2,0}	{1,0,1,0}	{0,0,2,0}
	5	0.6291	0.6294	**0.6295**	{1,1,2,1}	{0,1,3,1}	**{1,2,2,0}**
	10	0.6398	0.6403	0.6403	{2,3,3,2}	{2,3,4,1}	{2,3,4,1}
80	**2**	0.5518	0.5517	0.5518	{0,0,2,0}	{0,1,1,0}	{0,0,2,0}
	5	0.5590	0.5598	**0.5599**	{1,1,3,0}	{0,2,3,0}	**{0,1,4,0}**
	10	0.5703	0.5701	**0.5817**	{1,4,4,1}	{2,4,4,0}	**{2,3,3,2}**

Table 4.28 Test case 9 - Results for six-workstation unreliable production line with two symmetric bottlenecks (K=6)

***MTTR* (W_5)=40**	**Average Production Rate**			**Buffer Size Configuration**		
***MTTR* (W_2)**	**SA**	**GA**	**GAA**	**SA**	**GA**	**GAA**
40	0.5572	0.5584	0.5584	{2,1,1,1,1}	{1,1,1,2,1}	{1,1,1,2,1}
50	0.5571	0.5583	0.5583	{2,1,1,1,1}	{2,1,1,2,0}	{2,1,1,2,0}
80	0.5559	0.5571	**0.5580**	{4,1,0,1,0}	{3,1,1,1,0}	**{2,1,1,2,0}**
***MTTR* (W_5)=50**						
***MTTR* (W_2)**						
40	0.5442	0.5453	0.5453	{1,1,1,2,1}	{1,1,1,3,0}	{1,1,1,3,0}
50	0.5262	0.5280	0.5280	{2,1,1,1,1}	{1,1,1,2,1}	{1,1,1,2,1}
80	0.4883	0.4892	0.4892	{3,1,0,2,0}	{2,1,1,2,0}	{2,1,1,2,0}
***MTTR* (W_5)=80**						
***MTTR* (W_2)**						
40	0.5046	0.5046	0.5046	{1,1,1,3,0}	{1,1,1,3,0}	{1,1,1,3,0}
50	0.4879	0.4890	0.4890	{2,1,1,2,0}	{1,1,1,3,0}	{1,1,1,3,0}
80	0.4578	0.4553	0.4578	{2,1,1,2,0}	{3,0,0,2,1}	{2,1,1,2,0}

Table 4.29 Test case 10 - Results for six-workstation unreliable production line with two asymmetric bottlenecks (K=6)

***MTTR* (W_4)=40**	**Average Production Rate**			**Buffer Size Configuration**		
***MTTR* (W_2)**	**SA**	**GA**	**GAA**	**SA**	**GA**	**GAA**
40	0.5544	0.5554	0.5554	{2,1,2,1,0}	{1,2,2,1,0}	{1,2,2,1,0}
50	0.5390	0.5387	0.5390	{1,1,2,1,1}	{2,1,2,1,0}	{1,1,2,1,1}
80	0.5064	0.5077	0.5077	{3,1,1,1,0}	{2,1,2,1,0}	{2,1,2,1,0}
***MTTR* (W_4)=50**						
***MTTR* (W_2)**						
40	0.5398	0.5398	0.5398	{1,1,2,1,1}	{1,1,2,1,1}	{1,1,2,1,1}
50	0.5227	0.5245	0.5245	{2,1,1,1,1}	{1,1,2,1,1}	{1,1,2,1,1}
80	0.4886	0.4890	**0.4892**	{3,2,1,0,0}	{2,1,2,0,1}	**{2,1,1,1,1}**
***MTTR* (W_4)=80**						
***MTTR* (W_2)**						
40	0.4900	0.4904	**0.4912**	{1,1,2,1,1}	{1,0,3,1,1}	**{1,1,3,1,0}**
50	0.4754	0.4773	0.4773	{2,1,1,2,0}	{1,2,2,1,0}	{1,2,2,1,0}
80	0.4457	0.4467	0.4467	{2,1,1,1,1}	{1,1,2,1,1}	{1,1,2,1,1}

An analysis of all of the test cases reveals that the hybrid GAA produces the optimal or near-optimal buffer allocation in 100 of 103 instances in total. In 31

instances, only the GAA finds the best buffer allocation. To test the statistical significance of the differences among the proposed hybrid approach, the GAA, traditional SA, and traditional GA used in this study, the Wilcoxon signed-rank test is employed by using MINITAB 14. Non-parametric Wilcoxon signed-rank test for median difference is used when comparing two related populations, situations involving either matched items or repeated measurements on a single sample to assess whether their population averages differ. According to this statistical test, the *p*-values obtained from the comparison of SA with GAA and GA with GAA are less than 0.001 as seen in Table 4.30. Based on the comparison of all reported *p*-values to a significance level of 0.05, these results show that the outputs obtained from SA with GAA and GA with GAA cannot be proved to be the same in either case. Furthermore, it can be noted that our proposed GAA is a good hybrid approach for maximizing the production rate in various configurations of open serial lines.

Table 4.30 Wilcoxon signed-rank test results (α=0.05)

Algorithms	***p*-Value**
SA with GAA	<0.001
GA with GAA	<0.001

*Note: *p*-values of the one-tail test

Another key feature of the GAA is that the proposed hybrid approach yields good solutions in shorter computation times. Table 4.31 clearly shows that the computational times for SA are longer than those for GA and GAA. The proposed hybrid procedure shows an average improvement of 21.2% relative to the traditional SA algorithm and 8.8% relative to the traditional GA.

Table 4.31 Average CPU times for each test case

Methods	Average CPU Time (sec)										
	Test Case 1	Test Case 2	Test Case 3	Test Case 4	Test Case 5	Test Case 6	Test Case 7	Test Case 8	Test Case 9	Test Case 10	Average value (sec)
SA	36,806	27,787	21,601	21,806	23,545	30,350	33,191	25,531	30,658	33,322	28,460
GA	21,157	25,891	18,557	17,431	20,211	27,800	30,764	23,706	29,193	31,084	24,579
GAA	19,723	23,316	16,069	16,057	17,023	26,630	27,065	21,409	27,729	29,256	22,428

As a result, the power of the hybridization is observed in various configurations of serial lines for BAPs by considering the performance of our hybrid approach and its computation time.

4.4.2.5 Comparative Experiments on Very Large Production Lines

In general, real manufacturing systems involve much larger number of workstation, *i.e.*, at least 50 workstations. To help practitiones for designing production lines, long production lines should be considered; thereby more realistic BAPs can be formulated and be solved. In BAP literature, one can also feel the lack in the works dealing with the BAP on very large production lines. For these reasons, we investigate the performance of the proposed hybrid approach and its computation time in very large production lines by comparing the GAA to the traditional GA and traditional SA in this section. We consider eight different cases in which four of them are related to the reliable lines and the rest of them are of unreliable lines. Featuring very large lines, these lines consist of 50, 60, 80 and 100 workstations, respectively. The buffer sizes in the lines are assumed to be three times the number of workstations, as in the study by Spinellis and Papadopoulos (2000a). In the reliable serial lines, processing times are exponentially distributed with a mean processing time of 1.0 minute for all workstations. In the unreliable lines, the mean time between failures (*MTBF*) and mean time to repair (*MTTR*) are exponentially distributed with 20 and 2 minutes, respectively. Additionally, it is assumed that the processing rate for each workstation is one time unit. In all cases, simulation models

are developed and run for 100,000 jobs with 10 replications to estimate the average production rate.

Table 4.32 summarizes the overall results of the experimental study by comparing the proposed hybrid approach to GA and SA with respect to the average production rate, number of iterations for convergence and CPU time for very large lines. According to this computational study, the hybrid GAA and the traditional GA are substantially more efficient than the traditional SA. However, in six instances, the GAA finds better solutions for 50-, 60-, and 100-workstation reliable lines and 50-, 80-, and 100-workstation unreliable lines. As shown in Table 4.32, GAA is faster than both GA and SA and converges in four out of eight cases for very large problems. Additionally, our proposed hybrid procedure finds the average production rates in shorter computational times in all cases, as shown in Table 4.32.

Table 4.32 Computational results for very large lines

Case	Line	*W*	*K*	Search Method	Avg. Production Rate	Iteration	Average CPU Time (sec.)
1	Reliable	50	150	GAA	**0.649998**	**9**	**69,237.7**
				GA	0.647329	12	69,881.3
				SA	0.455492	16	73,462.6
2	Unreliable	50	150	GAA	**0.552007**	**15**	**71,715.1**
				GA	0.533198	14	73,324.7
				SA	0.532471	22	77,028.2
3	Reliable	60	180	GAA	**0.632446**	**5**	**112,860.6**
				GA	0.626573	17	113,290.5
				SA	0.572908	21	115,572.6
4	Unreliable	60	180	GAA	0.752590	18	135,122.8
				GA	0.752590	9	136,616.3
				SA	0.716897	13	141,293.1
5	Reliable	80	240	GAA	0.641707	15	220,040.0
				GA	0.641707	14	220,639.5
				SA	0.570198	18	241,841.0
6	Unreliable	80	240	GAA	**0.739887**	**5**	**101,523.6**
				GA	0.722418	8	124,573.5
				SA	0.715118	20	138,974.2
7	Reliable	100	300	GAA	**0.713545**	**12**	**141,671.3**
				GA	0.699960	16	142,207.7
				SA	0.679989	5	157,948.4
8	Unreliable	100	300	GAA	**0.742554**	**5**	**125,588.6**
				GA	0.740025	9	128,761.0
				SA	0.660553	11	162,223.9

Moreover, the convergence rates with respect to the average production rate in all cases are depicted in Appendix A3. In general, the average production rate converges quickly within a relatively small number of generations. This feature can be attributed to the fact that the proposed approach has great potential for solving BAPs.

4.5 Chapter Summary

In this chapter, a hybrid approach is proposed to obtain optimal or near-optimal buffer configurations in various serial production lines. The hybrid algorithm combining GA and SA is employed using simulations in an iterative manner to solve BAPs. Through this integration, the genetic annealing module suggests a buffer configuration at each

iteration, and the simulation model of the production line is run to obtain the fitness value of the candidate buffer configuration, *i.e.*, average production rate.

To test the performance of the proposed hybrid approach, an extensive experimental study is performed. First, the performance of the proposed approach is compared to other algorithms using benchmark problems previously reported in the literature for small, medium and large production lines. The experimental results demonstrate the efficacy of the GAA for a wide range of problem instances. Next, the proposed hybrid approach is performed under different open serial line conditions, *i.e.*, balanced, unbalanced, reliable, unreliable, upper limits for buffers and some bottleneck position conditions. The results show that the GAA yields the best buffer allocations in 100 of 103 instances. Moreover, only the proposed hybrid approach finds the best results in 31 of 103 instances. Finally, to demonstrate the performance of the proposed hybrid approach in solving the BAP in very large lines, GAA is implemented in eight problem cases consisting of 50-, 60-, 80- and 100-workstation reliable and unreliable production lines, and we compare the results to those obtained from the traditional GA and SA. Based on these results, the solution quality of our proposed hybrid approach in solving BAPs in very large production lines is quite good in terms of the average production rate and CPU time. In summary, we can conclude that the proposed hybrid GAA has great potential for improving the capacity of production systems and solving BAPs.

We consider the problem from a single objective (*i.e.,* average production rate maximization) perspective throughout this chapter. Since we aim at proposing hybrid approaches to solve the BAP in both single and multi-objective manner, in the next chapter, the problem is attempted to solve in a multi-objective manner by using proposed hybrid approach.

CHAPTER FIVE
A HYBRID EVOLUTIONARY APPROACH FOR MULTI-OBJECTIVE BAP IN SERIAL PRODUCTION LINES

5.1 Introduction

In this chapter, we solve the BAP in a multi-objective manner and propose a multi-objective hybrid evolutionary algorithm-based simulation optimization approach (MOHEA) to optimize two conflicting objectives, namely maximizing average production rate and minimizing total buffer size in serial production lines. In the context of hybrid evolutionary algorithm, two meta-heuristics -the elitist non-dominated sorting genetic algorithm (NSGA-II) and a special version of a multi-objective simulated annealing (MOSA) are combined to derive the Pareto optimal set. It should be noted there has been no study attempted with the hybridization of meta-heuristics in the multi-objective BAP literature. An experimental study is carried out to identify the best parameters of the hybrid evolutionary algorithm to improve the search mechanism of the proposed approach. Moreover, in order to demonstrate the efficacy of the proposed hybrid approach, a comparative study is provided for the multi-objective BAP in various serial line configurations.

The remainder of this chapter is organized as follows. Section 5.2 presents the multi-objective BAP along with the specific features for this study. The proposed hybrid solution approach (MOHEA) and its implementation are described in detail in Section 5.3. The results of computational experiments to test the performance of the proposed multi-objective hybrid evolutionary approach are given in Section 5.4. Finally, Section 5.5 summarizes the context of this chapter.

5.2 Problem Statement

This chapter adresses the multi-objective BAP which tries to minimize the total buffer space by appropriately allocating space to each buffer while maximizing average production rate in serial production lines. As given earlier in Chapter 2, for

the multi-objective BAP, the mathematical formulation used in this study can be expressed as follows:

$$\text{Find Pareto optimal set of } \quad B = (B_1, B_2, B_3, \ldots, B_{W\text{-}1}) \text{ to} \tag{5.1}$$

$$\text{Extremize } \quad Z = \{f_1(\bar{x}), f_2(\bar{x})\} \tag{5.2}$$

$$\text{subject to}$$

$$B_i \geq 0 \ (i = 1, 2, \ldots, W\text{-}1) \tag{5.3}$$

where W is the number of workstations in the line, B represents the buffer vector, and f_1 represents the average production rate of the system under a given buffer configuration and f_2 depicts the total buffer size.

Similar to the production lines under consideration in the previous chapter, open production lines in which the first workstation on the line is never starved and the last workstation is never blocked are examined. Moreover, each part goes through all the workstations in exactly the same order. In the following sections, we will explain our proposed solution approach to multi-objective BAP and apply this approach to various cases of open serial production lines to investigate optimal/near optimal buffer configurations that provides maximum average production rates with minimum total buffer sizes, simultaneously.

5.3 Proposed Multi-objective Hybrid Approach

The proposed MOHEA to solving the multi-objective BAP involves applying a search method and an evaluative method in an iterative manner. As a search method, the multi-objective hybrid evolutionary algorithm combining meta-heuristics of NSGA-II and MOSA, yields a set of candidate buffer configurations. As an evaluative tool, discrete event simulation modeling is used to estimate the performance measures for the production systems.

In the special case of multi-objective optimization problems, the selection and elitism operators must be structured to correctly identify the best individuals (Cruz et al., 2010). Additionally, the replacement scheme plays an important role for the next

generation through the algorithm. Hence, in this study, considering these schemes, the hybrid approach blends both NSGA-II and MOSA into a single approach to retain the strengths of these approaches. The proposed approach involves crowding distance estimation to select offspring for crossover and elitism scheme based on the concept of dominance. For the selection scheme, Nam & Parks' modified acceptance rule (2000) is adopted to the hybrid approach used to generate new population. Although the flow of hybrid algorithm and its general structure are similar in the traditional NSGA-II, its performance has been enhanced with the adaptation of an acceptance test in MOSA as the replacement scheme in NSGA-II. The control logic of the proposed multi-objective hybrid algorithm-based simulation optimization approach is depicted in Fig. 5.1. The further details about the features of the proposed MOHEA are explained in depth in the following subsections.

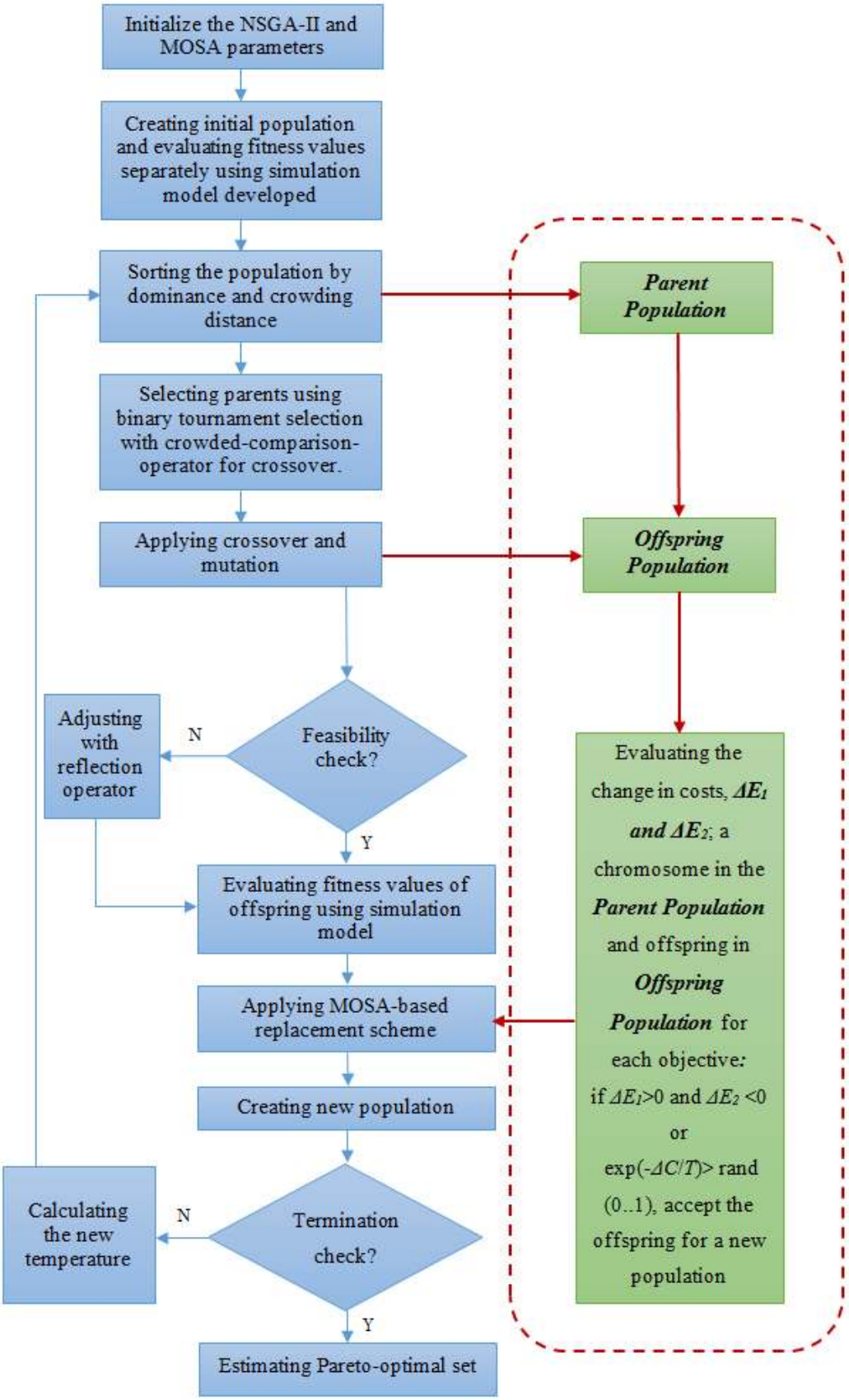


Figure 5.1 The control logic of the proposed MOHEA.

5.3.1 The General Specifications of Proposed Multi-objective Hybrid Approach

The following sections present the prominent features adopted to the proposed hybrid approach, namely, the encoding and initialization scheme, fitness evaluation, selection, crossover, mutation, MOSA-based replacement and search termination schemes.

5.3.1.1 Encoding and Initialization Scheme

Each buffer configuration is composed of an integer-valued coded string that represents the decision variables of the BAP ($B = \{B_1, B_2, \ldots, B_{W-1}\}$) as an initial solution. Moreover, each configuration is created randomly, while considering physical limitations of a facility. These limitations impose a lower (L_i) and an upper limit (U_i) on the number of buffers that can be accommodated in the system ($L_i \leq B_i \leq U_i$, i = 1, 2, …, W-1). The initialization procedure can be summarized with respect to limitations as follows:

Initilization Procedure:

for each candidate buffer configuration to generate initial population **do** (i=1 to population size, *pop_size*)

Initial_Population(i,:)= randint(1,W-1,[L U])

end

5.3.1.2 Evaluation and Selection Scheme

Having generated the initial population, the fitness values of each buffer size configuration in the population *i.e.,* the average production rate and total buffer size, are estimated separately by using the simulation model of the system. Then, these values are communicated to the MOHEA in an iterative manner. The fitness evaluation procedure in our proposed approach is given below:

Fitness Evaluation Procedure:

for each individual to obtain the fitness values **do**

write *Buffer_sizes* to an external file

execute the system file to run the simulation model

read average production rate from an external file

sum total buffer size used as *Buffer_sizes*

end

Next, the initialized population is sorted based on the non-domination sort criteria that is related to the elitism concept. Elitism is based on the concept of dominance in which a buffer size configuration in the search space, $B_i = (B_{i1}, B_{i2}, \ldots, B_{in})$, dominates another configuration, $B_j = (B_{j1}, B_{j2}, \ldots, B_{jn})$, if it is better in one objective without being worse in any other objective. Hence, each non-dominating individual in the population is ranked for each front. For instance, the first non-dominating front is generally assigned a rank of one. Similarly the second non-dominating front has a rank of two and so on. The individuals having lesser rank are the better candidates to be selected in this algorithm. This sorting procedure is described as below:

Non-dominated Sorting Procedure:

for each individual, *p*, in the population *P* **do**

initialize $Sp = Ø$. This set would contain all the individuals that is being dominated by *p*.

initialize $np = 0$. This would be the number of individuals that dominate *p*.

for each individual *q* in *P* **do**

if *p* dominates *q* then

add *q* to the set *Sp, i.e.*, $Sp = Sp \cup \{q\}$

else *q* dominates *p* then

increment the domination counter for *p, i.e.*, $np = np + 1$

end

end

if $np = 0$, *i.e.*, no individuals dominate *p* then *p* belongs to the first front;

set rank of individual *p* to one i.e $p_{rank} = 1$.

update the first front set by adding p to front one, *i.e.,* $F_1 = F_1 \, U \, \{p\}$

end

end

This is repeated for all other fronts, F_i, i=2,…, max.

For the same non-dominating front in the objective space, the crowding distance defined by Deb et al. (2002) is used to measure the distance of the biggest cuboid containing the two neighboring solutions. As seen in Figure 5.2, the average side length of the cuboid depicts the crowding distance of the i^{th} solution.

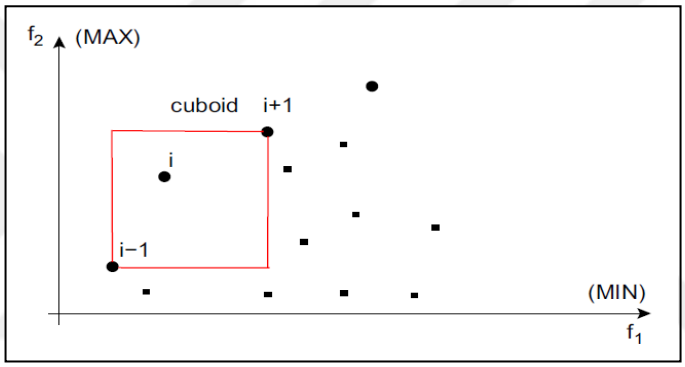


Figure 5.2 Crowding distance (Deb et al., 2002).

The estimation scheme of crowing distance is shown as below:

Crowding Distance Estimation Procedure:

for each front F_i, N is the number of individuals **do**

initialize the crowding distance to be zero for all the individuals, *i.e.,* $F_i(CD_j) = 0$, where j corresponds to the j^{th} individual in front F_i.

end

for each objective function m **do**

sort the individuals in front F_i based on objective m, *i.e.,* I = sort(F_i;m).

assign infinite crowding distance to boundary values for each individual in F_i, *i.e.,* $I(CD_1)=\infty$ and $I(CD_n)=\infty$

for k=2 to (n-1) **do**

$I(CD_k) = I(CD_k) + \frac{I_{k+1}^m - I_{k-1}^m}{f_m^{max} - f_m^{min}}$

end

end

$\boldsymbol{I_k^m}$ is the value of the m^{th} objective function of the k^{th} individual in *I.* The basic idea behind the crowding distance is finding the euclidian distance between each individual in a front based on their *m* objectives in the *m* dimensional hyper space. The individuals in the boundary are always selected since they have infinite distance assignment.

Besides non-domination sorting and calculating crowding distances for the same fronts, the selection scheme is employed using binary tournament selection with the crowding-comparison-distance operator. In this selection scheme, to select each candidate solution for crossover, a random subset of *k* solutions from the original population is selected and then the best solution out of this subset is chosen. If *k* is set to 2, this selection scheme is called binary tournament selection (See Figure 5.3). The comparison between candidate buffer size configurations (B_i and B_j) for crossover is carried out as below:

1. if B_i has a better rank, that is $r_i < r_j$,
2. if they have the same rank but B_i has a better crowding distance than B_j, that is, $r_i = r_j$ and $d_i > d_j$.

Hence, B_i wins a tournament with B_j.

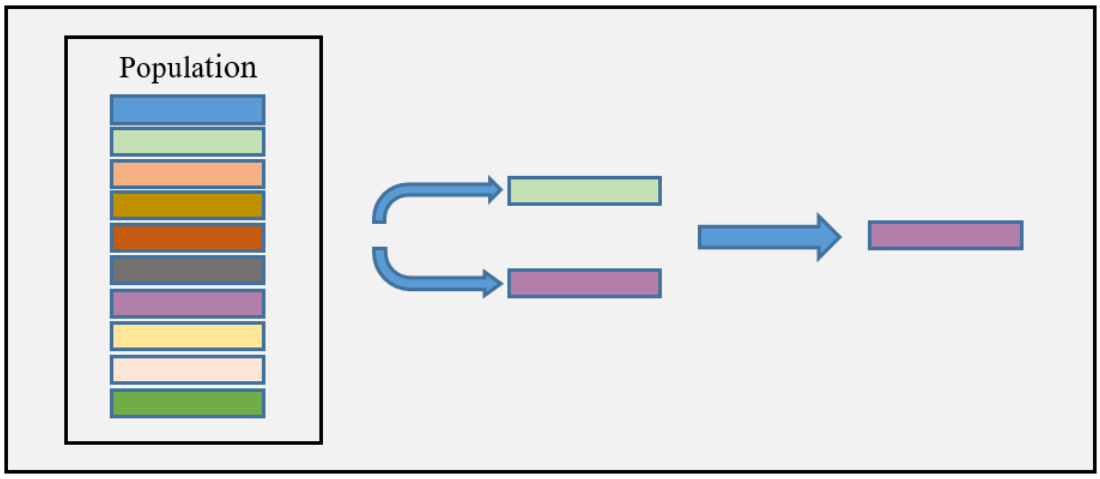


Figure 5.3 Binary tournament selection scheme

5.3.1.3 Crossover and Mutation

To obtain higher quality buffer size configurations, we use crossover and mutation genetic operators in the proposed multi-objective hybrid algorithm. Crossover is the main genetic operator that combines two individuals, namely, parents from the current population to produce two new offspring sharing many of the characteristics of their parents for the next population. Once the selection process is performed, the new population in which all of the individuals are subject to crossover with a probability of crossover (P_c) is selected. We referred to the uniform crossover procedure given in (Cruz et al., 2010) with simulated binary crossover (SBX) operator because of its efficiency in identifying, inheriting, and protecting common genes, as well as re-combining non-common genes (Cruz et al., 2010; Hu & Di Paolo, 2007; Sywerda, 1989). Moreover, SBX operator is a well-suited for real-coded GAs (Deb & Agrawal, 1995).

Two new offspring, $x_{i,(.t+1)}$, are calculated from the parents, $x_{i,(.t)}$, as follows (Cruz et al. 2010):

$$x_{i,(1,t+1)} = 0.5[(1-\beta_q)x_{i,(1,t)} + (1+\beta_q)x_{i,(2,t)}] \quad (5.4)$$

$$x_{i,(2,t+1)} = 0.5[(1+\beta_q)x_{i,(1,t)} + (1-\beta_q)x_{i,(2,t)}] \quad (5.5)$$

where $x_{i,(1,t+1)}$ is the $(t+1)^{th}$ offspring with i^{th} individual, $x_{i,(1,t)}$ and $x_{i,(2,t)}$ are selected parents and β_q is a random variable obtained from the following probability distribution:

$$p(\beta) = 0.5(\eta + 1)\beta^{\eta}, \ \ if\ 0 \leq \beta \leq 1 \tag{5.6}$$

$$p(\beta) = 0.5(\eta + 1)\frac{1}{\beta^{\eta+2}}, \ \ if\ \beta > 1 \tag{5.7}$$

that is specifically designed to create a child solution that has a similar search power as that in a single-point crossover in binary-coded GAs (Deb & Agrawal, 1995). The distribution index η is a user-defined parameter which determines the peakedness of the distributions. This distribution can be obtained from a uniformly sampled random number u between (0,1).

$$\beta(u) = (2u)^{\frac{1}{\eta+1}}, \ \ u \leq 0.5 \tag{5.8}$$

$$\beta(u) = \frac{1}{[2(1-u)]^{\frac{1}{\eta+1}}}, \ \ u > 0.5 \tag{5.9}$$

It should be noted for SBX operator that the average of the real coded values is the same before and after the crossover. Figure 5.4 illustrates the crossover operation for two buffer size configurations selected as parents.

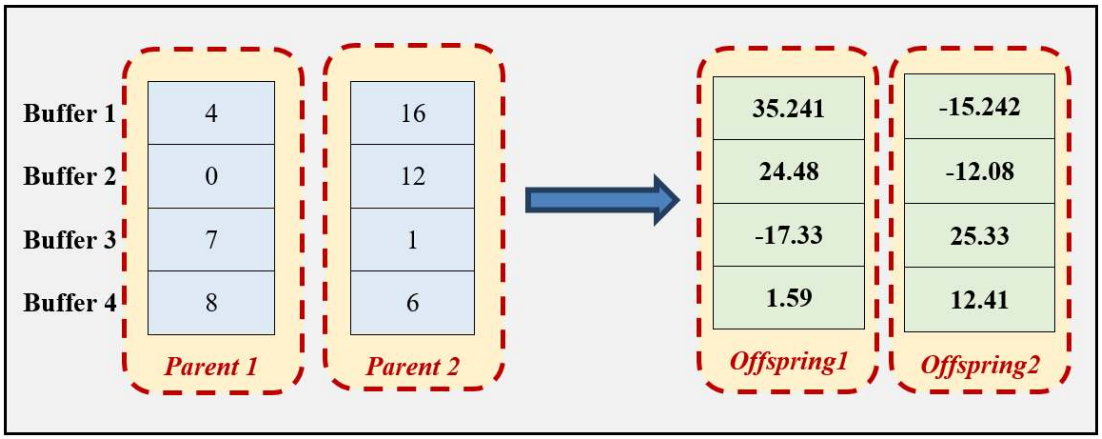


Figure 5.4 Crossover procedure for a four-buffer configuration.

In order to maintain the genetic diversity from one generation of a population to the next, all of the individuals in the population generated after the crossover are subject to mutation with a probability of mutation (P_m) in the proposed MOHEA. We use the same mutation procedure suggested by Deb & Agrawal (1995) in which a Gaussian perturbation, $\varepsilon_i \sim N(0,1)$, is introduced for each individual. The mutation scheme is presented in Figure 5.5. It has been observed that the constraint of integer-valued coding for each buffer sizes may not hold any longer after crossover and mutation.

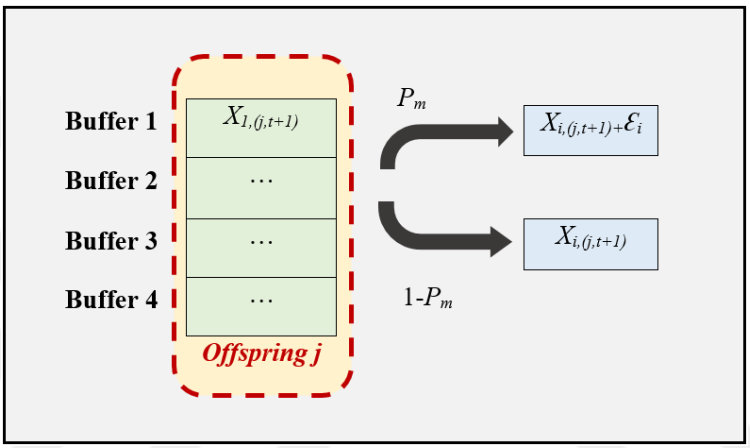


Figure 5.5 Mutation scheme (Cruz et al., 2010).

Therefore, infeasible solutions, which violate the lower-upper limits and integer-valued coding for each buffer sizes, are not allowed in the population. To deal with this situation and guarantee feasibility, the values should be rounded at first and adjusted taking into account of upper and lower limits. If the value is lower than lower limit of buffer size, it is adjusted accordingly by means of the reflection operator as below:

$$x_{i,r} = x_l + |x_i - x_l|, \tag{5.10}$$

where x_l is the lower limit for buffer allocation, x_i is the resulting buffer allocation after crossover and mutation and $x_{i,r}$ is the result after reflection,with |x| being the absolute value of x. If the value is greater than the upper limit of buffer size, the related buffer size is set to the upper limit. As an example, for the buffer size

configuration of B = {-17, 5, 3, 14}, the first buffer size is lower than 0 which is the lower limit for each buffer size, this value is adjusted as follows:

$$x_{i,r} = 0 + |-17 - 0|, = 17 \tag{5.11}$$

5.3.1.4 MOSA-based Replacement Scheme

As the replacement scheme, the acceptance test, similar to that in the MOSA procedure, is adopted to determine which buffer configuration should stay in a population and which buffer configuration should transfer to the next generation. Evaluating individuals from the initial population and the new individuals formed after crossover and mutation, the non-dominated buffer size configurations and the other configurations, which have acceptably average production rates and total buffer sizes at the same time, can be selected for the next generation. The replacement test is designed with the changes in cost (*i.e.*, average production rate and total buffer size) (ΔE_1 *and* ΔE_2) and the acceptance probability, which is determined by a temperature control parameter (*initial temperature*, T_0) that decreases during the MOSA procedure. In the replacement step, the comparison is carried out between the current buffer size configuration (B_{cur}) in the parent population and the new configuration in the offspring population. In this respect, B_{new} can become the new buffer size configuration for the next generation if B_{new} is not dominated by B_{cur}. If B_{cur} is better in one objective at least, B_{new} may still be accepted with a certain probability, p = exp ($-\Delta C/T$). In this acceptance rule, the cost function (ΔC) is based on the random cost criterion exposed by Nam and Park (2000) and calculated as below:

$$\Delta C = \sum_i^m \beta_i \left(f_i(B_{new}) - f_i(B_{cur})\right) \tag{5.12}$$

where β_i is a random variable with uniform probability distribution, and f_i represents the fitness values for m objectives. For the acceptance rule, a random number between zero and one is generated. If the random number is smaller than exp ($-\Delta C/T$), B_{new} is accepted, otherwise, B_{cur} remains in the population for the next generation. Once the new population is generated for the next generation, the

temperature is updated according to the geometric cooling schedule ($T_{k+1}=c*T_k$, where T_k is the temperature at iteration k, c is the cooling rate ($0<c<1$)). The replacement procedure can be summarized as follows:

Replacement Procedure:

calculate the changes in cost, ΔE_1 *and* ΔE_2

normalize the fitness values of *Parent Population* and *Offspring Population*

calculate the Boltzmann's equation with random cost criterion

% for each individual, *x*, in the *Parent Population, x',* in the *Offspring Population*

loop *pop_size* is satisfied

if x does not dominate x' ($\Delta E_1>0$ && $\Delta E_2<0$) then

accept x' to the new population

elseif exp(-$\Delta C/T$)> rand (0..1)

accept x' to the new population

else

accept x to the new population

end

end

5.3.1.5 Termination Criterion

For evolutionary algorithms, setting a fixed number of generations is the most commonly used termination criterion but it may not be the best. Rudenko & Schoenauer (2004) remarked that a much better stopping criterion is when a fixed number of iterations are performed without improvement to avoid wasting precious computation time. Another termination criterion used by Rudenko & Schoenauer (2004) is based on stabilizing the maximal crowding distance, d_l, measured over L generations by calculating the following standard deviation:

$$\sigma_L = \sqrt{\frac{1}{L}\sum_{l=1}^{L}\left(d_l - \overline{d_L}\right)^2} \tag{5.13}$$

where $\overline{d_L}$ is the average of d_l over L generations and the algorithm stops when $\sigma_L<\delta_{lim}$. It has been noted that σ_L should not depend on the actual values of the objective functions because all crowding distances are normalized (Cruz et al., 2010). Regarding all termination criteria described above, in our proposed MOHEA, the criterion will be decided employing a factorial experimental design to examine the effects of two stopping criteria on algorithm search performance in the next section.

5.4 Computational Experiments

In this section, a computational experimentation is carried out in two phases. In the first phase, the efficient control parameters for the proposed hybrid algorithm are identified to ensure high performance of the hybrid algorithm. Next, the performance of the proposed approach and the power of the hybridization are investigated for various serial line configurations. In all experiments, discrete event simulation models are developed by using the simulation language Arena V10.0 to estimate the average production rate and total buffer size allocated as an evaluative tool. Search algorithms in all test cases are implemented in Matlab V7.6. The experiments are performed on a PC with a 2-GHz Pentium (R) 4 CPU processor with 2 GB of RAM.

5.4.1 MOHEA Parameter Setup

Identifying the best parameters in order to improve the performance of the algorithm in solving a problem is definitely necessary task throughout the search process. Since MOHEA combines the specific features of NSGA-II and MOSA, the parameters used in both algorithms should be taken into consideration to discover their best values. In fact, these algorithms have several parameters and any combination of them has different impacts on the performance of the algorithm. Hence, this section is devoted to the identification of efficient MOHEA parameters for solving multi-objective BAP.

The control parameters investigated are:

- population size (*pop_size*),

- crossover probability rate (P_c),
- mutation probability rate (P_m),
- parameter η which controls the dispersion of random variable β_q used in the SBX operator for crossover,
- initial temperature (T_0),
- cooling rate (c).

In this study, since our multi-objective hybrid approach is based on NSGA-II, we have summarized the previous studies on BAP solved with GA and NSGA-II to identify the common used levels of control parameters. For instance, 30 and 50 individuals are the commonly used population sizes in the BAP literature. Moreover, Goldberg (1989) suggested for GA that the choice of high-crossover (0.6-0.9) and low-mutation probabilities (0.01-0.1) and a moderate population size (50-100) result a good algorithm performance. Considering both these explanations about each level of control parameters, the levels of population size are set to 50 and 80. Additionally, other levels of control parameters for NSGA-II (P_c, P_m *and* parameter η) are determined in a similar way for the experimental study. For the MOSA parameters used in the literature, it has been observed that there is no significant difference between the levels, hence, the commonly used parameter values for T_0 and c are set to 0.5 and 0.90 in all experiments.

As mentioned in the previous section, the termination scheme is also experimented to discover the best scheme for the proposed hybrid algorithm. We study on a stopping condition demonstrating any improvement on fitness values through consecutive generations over a fixed maximum number of generations. Moreover, another condition considered in the experimental study is based on stabilizing the maximal crowding distance (d_l). This distance proposed by Rudenko and Schoenauer (2004) measures the standard deviation (σ_L) over L generations and the algorithm stops when the deviation is less than the specified limit by decision maker ($\sigma_L<\delta_{lim}$):

$$\sigma_L = \sqrt{\frac{1}{L}\sum_{l=1}^{L}\left(d_l - \overline{d_L}\right)^2} \tag{5.14}$$

In summary, we use a full factorial design configured with two fixed levels for each control parameter as seen in Table 5.1.

Table 5.1 Experimental factors and levels for MOHEA

Factors	**Levels**	
Population size	50	80
Crossover probability rate	0.60	0.80
Mutation probability rate	0.033	0.1
η	8	16
Stopping condition	Generation number and maximum number of consecutive generations without improvement	Stabilizing maximal crowding distance (MCD)

Further, in order to permit the detection of a possible interaction of factor effects, a 2^5 full factorial experimental layout was used to carry out the experiments. With two levels for each variable, a full factorial design will translate into $2^5 = 32$ experiments as seen in Table 5.2. In all experiments, the maximum number of generations, maximum number of consecutive generations without improvement and the specified limit for stabilizing maximal crowding distance (δ_{lim}) are set to 100, 5 and 0.02, respectively, for the termination schemes.

Table 5.2 2^5 full factorial experimental layout

Exp. No	*Pop_size*	P_c	P_m	η	Stopping condition
1	50	0.60	0.010	8	Gen num (100)
2	50	0.60	0.010	8	*MCD*
3	50	0.60	0.010	16	Gen num (100)
4	50	0.60	0.010	16	*MCD*
5	50	0.60	0.033	8	Gen num (100)
6	50	0.60	0.033	8	*MCD*
7	50	0.60	0.033	16	Gen num (100)
8	50	0.60	0.033	16	*MCD*
9	50	0.80	0.010	8	Gen num (100)
10	50	0.80	0.010	8	*MCD*
11	50	0.80	0.010	16	Gen num (100)
12	50	0.80	0.010	16	*MCD*
13	50	0.80	0.033	8	Gen num (100)
14	50	0.80	0.033	8	*MCD*
15	50	0.80	0.033	16	Gen num (100)
16	50	0.80	0.033	16	*MCD*
17	80	0.60	0.010	8	Gen num (100)
18	80	0.60	0.010	8	*MCD*
19	80	0.60	0.010	16	Gen num (100)
20	80	0.60	0.010	16	*MCD*
21	80	0.60	0.033	8	Gen num (100)
22	80	0.60	0.033	8	*MCD*
23	80	0.60	0.033	16	Gen num (100)
24	80	0.60	0.033	16	*MCD*
25	80	0.80	0.010	8	Gen num (100)
26	80	0.80	0.010	8	*MCD*
27	80	0.80	0.010	16	Gen num (100)
28	80	0.80	0.010	16	*MCD*
29	80	0.80	0.033	8	Gen num (100)
30	80	0.80	0.033	8	*MCD*
31	80	0.80	0.033	16	Gen num (100)
32	80	0.80	0.033	16	*MCD*

For the experimental design, a BAP initially proposed by Ho et al. (1979) with medium complexity is experimented to observe the effects of different control parameters on the proposed hybrid algorithm. The system under consideration is a

production system consisting of seven workstations with exponential processing times and six buffers, which are located between the stations. Moreover, the workstations are subject to breakdown, and the failure (p_i) and repair (r_i) rates of the workstations are exponentially distributed. Each buffer may have a size that is bounded between the lower value (l) which is equal to 1 and the upper value (u) which is equal to 20. The dataset is given in Table 5.3.

Table 5.3 Dataset for seven-workstation production line (Ho et al. 1979)

Workstation	$MTTR=1/r_i$	$MTBF=1/p_i$	T_i
1	450	820	40
2	760	5700	34
3	460	870	39
4	270	830	38
5	270	970	37
6	650	1900	40
7	320	1100	43

The simulation model is developed as an evaluative tool and run for 5,000 pieces each with 50 replications. The performance measure considered throughout the analysis is the mean of diversity metric to measure the extent of spread achieved among the obtained solutions (Deb et al., 2002). Getting a set of solutions that spans the entire Pareto-optimal region, the following metric is used to estimate the non-uniformity in the distribution:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \bar{d}|}{d_f + d_l + (N-1)\bar{d}} \tag{5.15}$$

where d_i is the Euclidean distance between N-1 consecutive solutions in the obtained non-dominated set of solutions, $\bar{d}$ represents the average of all distances, d_f and d_l show the the Euclidean distances between the extreme solutions in the objective space and the boundary solutions of the obtained non-dominated set as depicted in Figure 5.6. The denominator is the value of the numerator for the case when all N solutions lie on one solution (Deb et al., 2002). The smaller the value of this metric, the better the divergence toward the Pareto-optimal front.

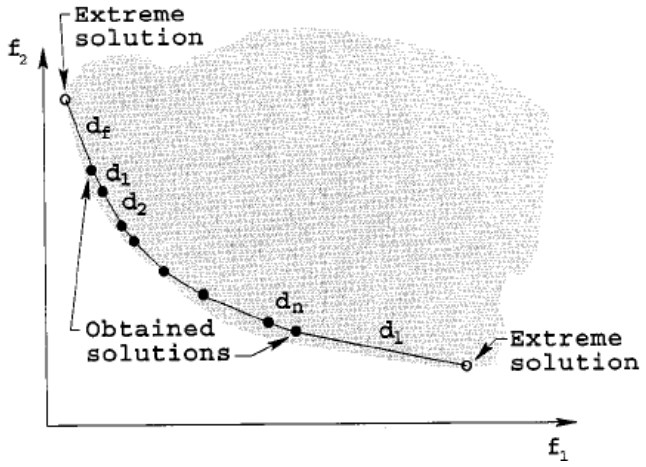


Figure 5.6 Diversity metric (Deb et al., 2002).

In the design of experiments, at each parameter setting, we performed multiple runs, *i.e.,* three runs to determine the variation in the results. In order to determine which control parameter effects are significant, a statistical analysis of variance (ANOVA) is conducted. The estimated effects and coefficients performed in MINITAB is shown in Table 5.4. Using $\alpha = 0.05$, the main effects for *pop_size*, P_c, η, *the stopping condition* and some *interactions* between control parameters are statistically significant; that is, their *p*-values are less than 0.05. This suggests that in order to increase the performance of the hybrid algorithm in seeking the minimum fitness function, we may adjust the control parameters considering the interactions between the parameters.

Table 5.4 Factorial Fit: Mean versus *Pop_size*, P_c, P_m, η and termination scheme

Term	Effect	Coef	SE Coef	*T*	*P*
Constant		0.75094	0.002485	302.15	0.000
Pop_size	**-0.04795**	**-0.02397**	**0.002485**	**-9.65**	**0.000**
P_c	**-0.03978**	**-0.01989**	**0.002485**	**-8.00**	**0.000**
P_m	-0.00420	-0.00210	0.002485	-0.85	0.401
η	**0.02247**	**0.01123**	**0.002485**	**4.52**	**0.000**
Stopping condition	**0.10141**	**0.05071**	**0.002485**	**20.40**	**0.000**
*Pop_size**P_c	-0.00543	-0.00271	0.002485	-1.09	0.279
***Pop_size**P_m**	**0.02769**	**0.01384**	**0.002485**	**5.57**	**0.000**
*Pop_size**η	0.00243	0.00122	0.002485	0.49	0.626
***Pop_size** Stopping condition**	**0.03805**	**0.01903**	**0.002485**	**7.66**	**0.000**
P_c*P_m	-0.00502	-0.00251	0.002485	-1.01	0.317
P_c*η	**0.02995**	**0.01497**	**0.002485**	**6.02**	**0.000**
P_c*Stopping condition	**0.01519**	**0.00760**	**0.002485**	**3.06**	**0.003**
P_m*η	0.00369	0.00185	0.002485	0.74	0.460
P_m* Stopping condition	**-0.01886**	**-0.00943**	**0.002485**	**-3.79**	**0.000**
η*Stopping condition	**0.01492**	**0.00746**	**0.002485**	**3.00**	**0.004**
***Pop_size**P_c*P_m**	**-0.05472**	**-0.02736**	**0.002485**	**-11.01**	**0.000**
***Pop_size**P_c*η**	**-0.01133**	**-0.00566**	**0.002485**	**-2.28**	**0.026**
*Pop_size**P_m*η	0.00723	0.00362	0.002485	1.46	0.150
***Pop_size**P_c*Stopping condition**	**-0.01829**	**-0.00915**	**0.002485**	**-3.68**	**0.000**
***Pop_size**P_m*Stopping condition**	**0.01169**	**0.00584**	**0.002485**	**2.35**	**0.022**
***Pop_size**η*Stopping condition**	**-0.03671**	**-0.01835**	**0.002485**	**-7.39**	**0.000**
P_c*P_m*η	**-0.02429**	**-0.01214**	**0.002485**	**-4.89**	**0.000**
P_c*P_m*Stopping condition	**0.01932**	**0.00966**	**0.002485**	**3.89**	**0.000**
P_c*η* Stopping condition	**0.03253**	**0.01626**	**0.002485**	**6.54**	**0.000**
P_m*η*Stopping condition	0.00596	0.00298	0.002485	1.20	0.235
*Pop_size**P_c*P_m*η	-0.00630	-0.00315	0.002485	-1.27	0.209
***Pop_size**P_c*P_m*Stopping condition**	**-0.01680**	**-0.00840**	**0.002485**	**-3.38**	**0.001**
*Pop_size**P_c*η*Stopping condition	-0.00569	-0.00284	0.002485	-1.14	0.257
*Pop_size**P_m*η*Stopping condition	-0.00674	-0.00337	0.002485	-1.36	0.180
P_c*P_m*η*Stopping condition	**-0.02074**	**-0.01037**	**0.002485**	**-4.17**	**0.000**
*Pop_size**P_c*P_m*η*Stopping condition	-0.00445	-0.00222	0.002485	-0.89	0.374

In order to see which effects influence the mean of diversity, the normal probability plot is evaluated as seen in Figure 5.7.

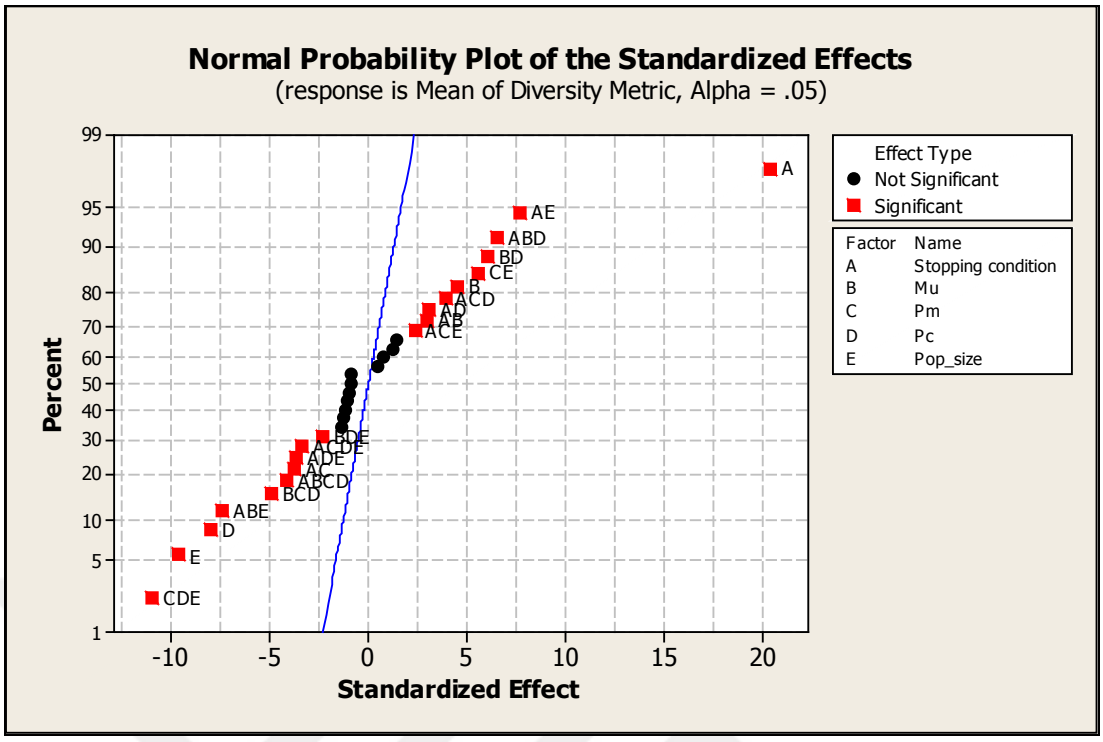


Figure 5.7 Normal probability plot.

Further, to assist in the practical interpretation of this experiment, Figure 5.8 presents the plots of the five main effects for each control parameter. The stopping condition, *pop_size*, P_c and η have a similar effect on the mean of diversity. The plot also indicates that using:

- the generation number that equals to 100 as a stopping criterion yields less diversity mean value than the stopping criterion on stabilizing the maximal crowding distance.
- η =8, gives less diversity mean value than the value of 16.
- the crossover probability rate as 0.8, gives less diversity mean value than the value of 0.6.
- the *pop_size* as 80 yields less diversity mean value than using the value of 50.

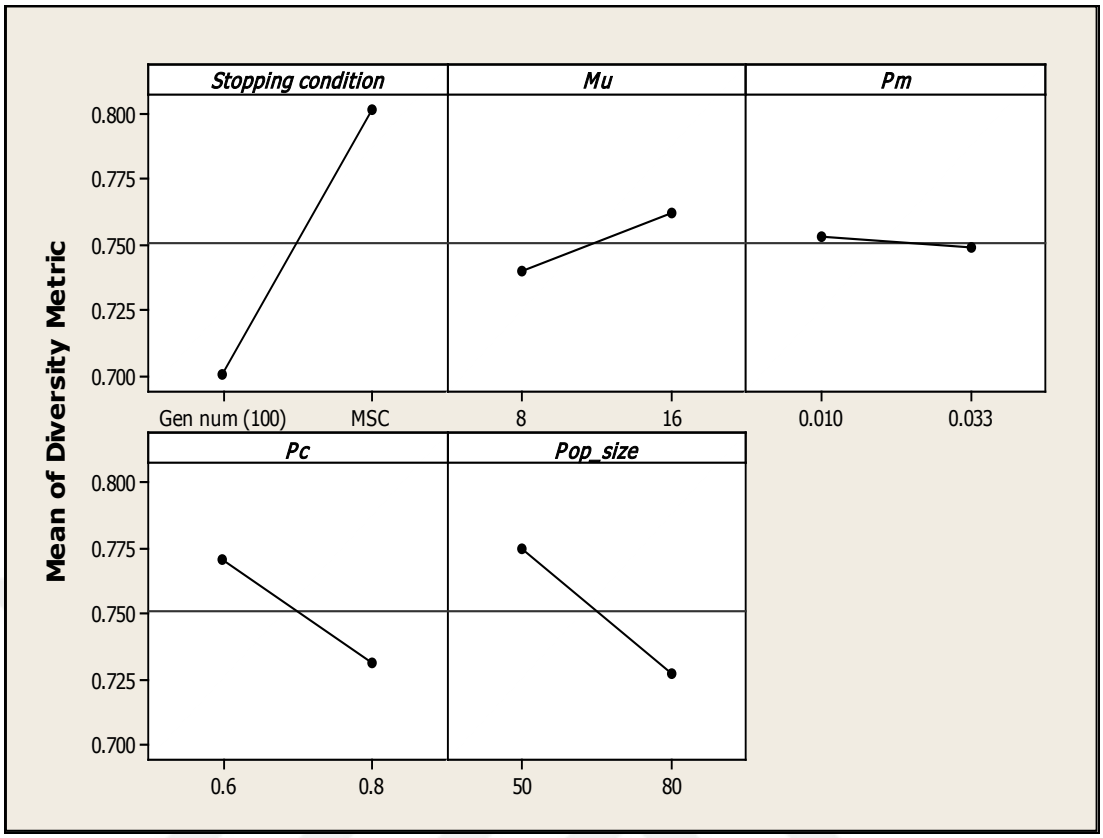


Figure 5.8 Main effects plot.

In particular, since the interactions in this analysis are significant, Figure 5.9 depicts plots of means for each level of a factor with the level of a second factor held constant for judging the presence of interaction. For instance, the first plot shows that using generation number setting to 100 with η=8 gives less mean diversity value than stabilizing the maximal crowding distance as a stopping condition with with η=8 (about 0.65). Using stabilizing the maximal crowding distance criterion as a termination scheme with η=16 gives much more mean diversity value than the generation number setting to 100 as a stopping criterion with η=10 (about 0.85). Because the slope of the line for the generation number setting to 100 is steeper, it is concluded that the parameter η has a greater effect when the generation number is used versus the stabilizing the maximal crowding distance. Based on these experimental results, it is possible to recommend for the algorihm that the values of *pop_size*=80, P_c=0.8, P_m=0.033, η=*8* and using generation number (100 generations) as a termination criterion.

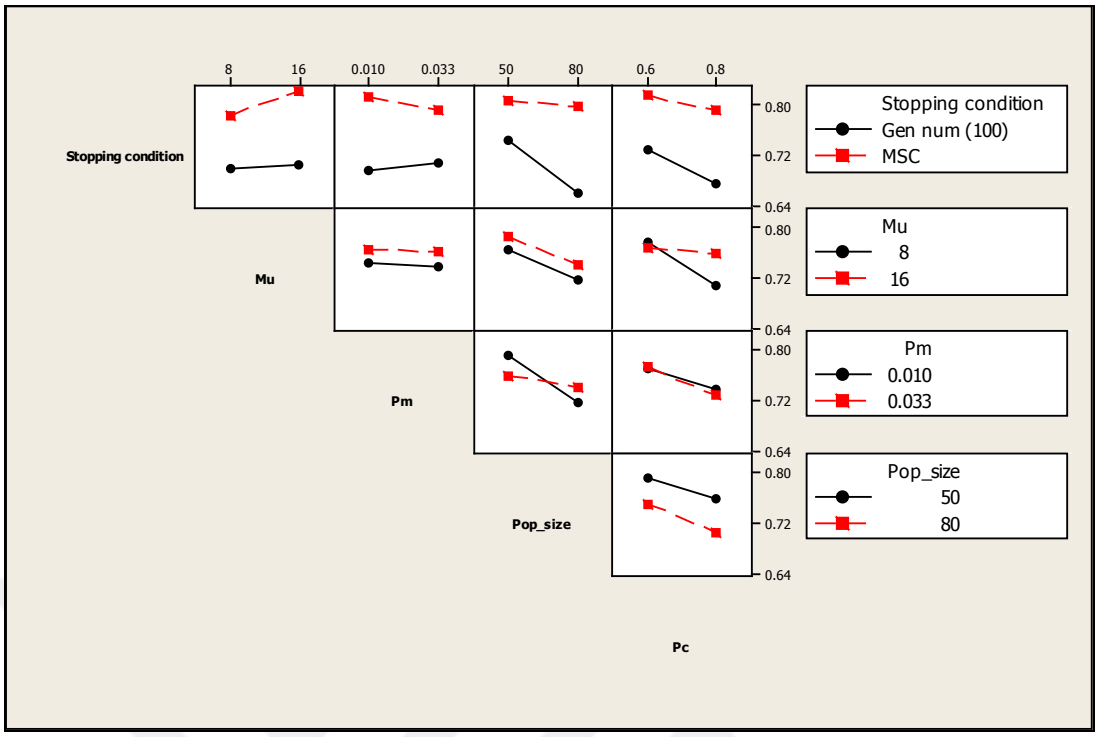


Figure 5.9 Interaction plots.

5.4.2 Comparative Experiments under Different Serial Line Conditions

In this section, the proposed MOHEA is tested in various open serial line designs, *i.e.*, balanced, unbalanced, reliable, unreliable and upper limits for buffers, to demonstrate the power of the hybridization. Although the BAP in serial production lines has been studied to a large extent in the literature, available input data for multi-objective BAP (Ho et al., 1979) and numerical results pertinent to the test cases have rarely been reported (Chehade et al., 2010).

Throughout the experiments discussed in this section, the proposed MOHEA is compared to the traditional NSGA-II based on solution quality. The flow of the traditional NSGA-II and its parameters are similar to those of the proposed hybrid approach. The only difference in the NSGA-II is the replacement test, in which an elitism based scheme is used. To make a meaningful comparison, we first explain comparison criteria that are commonly presented in the literature.

5.4.2.1 Comparison Criteria

In multi-objective optimization, it is necessary to get the following performances: the convergency, the diversity and the spread results to reach the optimization of the non-dominated set (Zitzler & Thiele, 1999). In this study, four measuring criteria are applied to compare two non-dominated fronts obtained by the proposed MOHEA and the traditional NSGA-II. These measures are described as below:

- Number of solutions in an optimal front $n(f_i)$: Let f_1 and f_2 are the fronts of non-dominated solutions, i.e. the first Pareto fronts obtained with corresponding algorithms and $n(f_i)$ be the number of solutions in the Pareto front of non-dominated solutions obtained with the corresponding method.
- Distance proposed by Riise (μ) (Riise, 2002): The distance of Riise, μ, is computed as the sum of the distances d_x between a solution x belonging to front F_1 and its orthogonal projection on front F_2.

$$\mu = \sum_{x=1}^{N} d_x \tag{5.16}$$

Hence, for one given instance, if $\mu(F_1,R)<0$ which means that F_1 is under F_2 for the considered objectives, and positive, otherwise. In other words, an algorithm with a final front F_1 outperforms another with its final front R. Since μ depends on the number of solutions in F_1, a normalized measure, $\overline{\mu}$ is generally taken into consideration to avoid sensibility on the number of points:

$$\bar{\mu} = \frac{1}{n}\sum_{x=1}^{n} d_x \tag{5.17}$$

For this distance, when the fronts are closed to each other, it is not significant (Dugardin, Yalaoui, & Amodeo, 2010). For example, when one front overlaps another front, the d_x is alternatively positive and negative because sometimes a solution of front F_1 is better than a F_2 one and so on. In this case the $\overline{\mu}$ does not depict the relative position of the two fronts.

- Zitlzler measure (C_i) (Zitlzler & Thiele, 1999): The Zitzler measure C_1 represents the percentage of solutions in F_1 dominated by or equal at least one solution in F_2. Taking in consideration that the measure is not symmetric, it is advised to compute C_2 as well. In conclusion, a front F_1 is better than another front F_2 if C_1 is smaller than C_2. As it is stated by Zitzler & Thiele (1999), the measures C_1/C_2 can be used to show that the outcomes of one algorithm dominate the outcomes of another algorithm, although it does not tell how much better it is.
- Diversity metric (Δ) (Deb et al., 2002): This metric based on Euclidean distances between consecutive solutions in the obtained non-dominated set of solutions measures the extent of spread achieved among the obtained solutions. The smaller the value of this metric, the better the divergence toward the Pareto-optimal front.

5.4.2.2 Test Case 1: Unbalanced Unreliable Serial Line

The original information regarding the characteristics of workstations for this case is initially proposed by Ho et al. *(*1979) and used in the study by Chehade et al. (2010). As similar in these studies, this case is realized on small lines with W stations (W=3, 7 and 9) and the workstations are subject to breakdown, and the failure (p_i) and repair (r_i) rates of the workstations are exponentially distributed. The dataset is given in Table 5.5. Moreover, each buffer size is bounded with a lower and upper limit, *i.e.,* 1 and 20, respectively. As indicated in the study by Ho et al. (1979), the simulation models are developed as an evaluative tool and run for 5,000 pieces each with 50 replications to estimate the average production rate and the total buffer size. Figure 5.10 shows the extrapolated fronts for each case in small production lines. Moreover, the Pareto optimal solutions obtained by our approach and NSGA-II are given with detailed results in the tables in Appendix B1.

Table 5.5 Dataset for Test case-1 (Ho et al., 1979)

Workstation	$MTTR=1/r_i$	$MTBF=1/p_i$	T_i
1	450	820	40
2	760	5700	34
3	460	870	39
4	270	830	38
5	270	970	37
6	650	1900	40
7	320	1100	43
8	480	910	39
9	340	1050	41

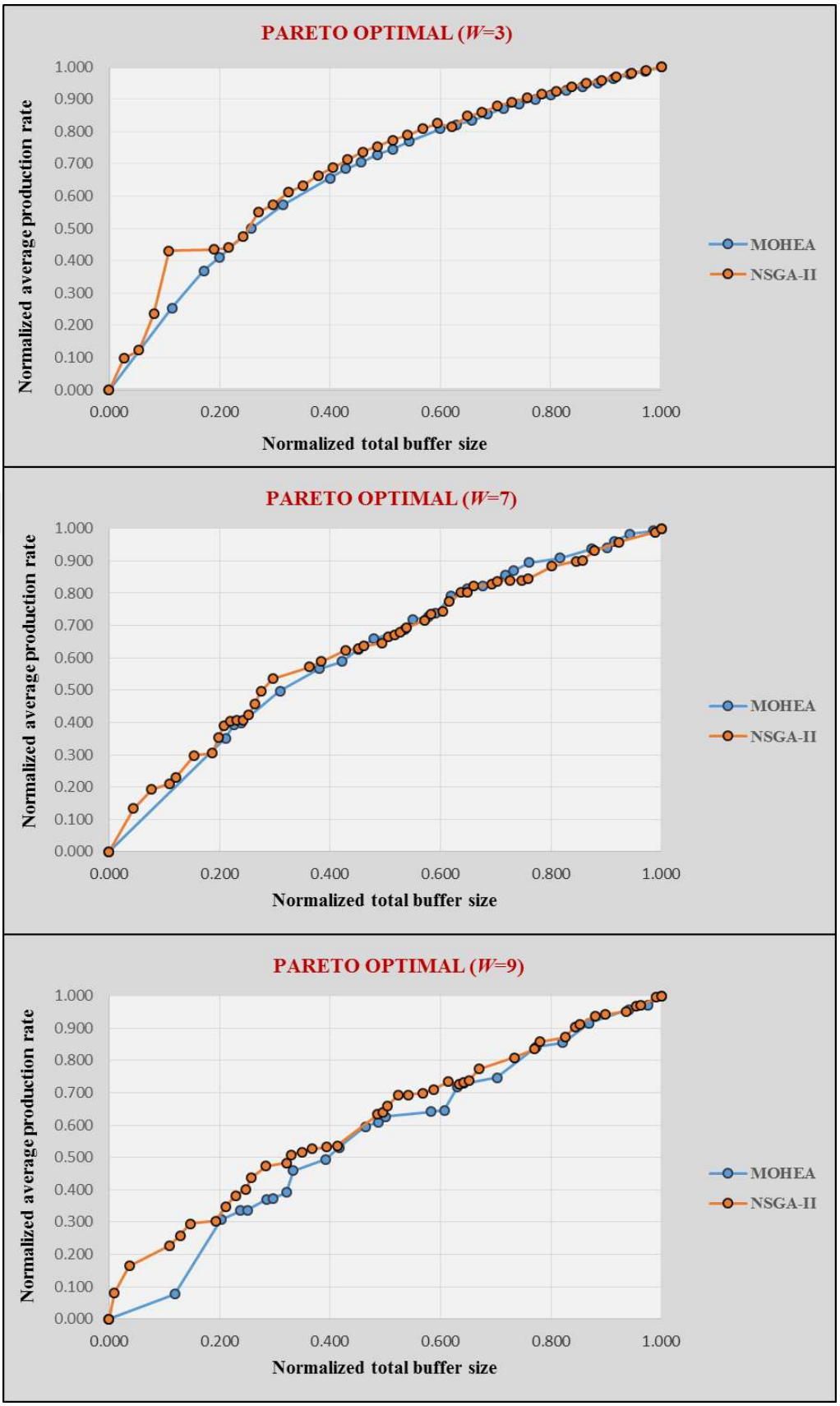


Figure 5.10 Test case 1- Extrapolated fronts for each case.

To compare the performance of the proposed hybrid approach to the traditional NSGA-II, the computational results for this case are summarized in Table 5.6. The first column (W) in Table 5.6 depicts the number of workstations in the serial line. In the second and third columns, nf_1 stands for the number of solutions in the optimal front obtained by the hybrid approach and nf_2 for the NSGA-II. For the fourth column, the distance of Riise is shown under the $\overline{\mu}$ column. In Table 5.6, C_1 and C_2 stand for the Zitzler measure and these measurements are given in the fifth and sixth columns for the front F_1 of hybrid approach and that of front F_2 of the NSGA-II. For the hybrid approach (Δ_1), the mean diversity metric is presented in the seventh column, and for NSGA-II (Δ_2), they are presented in the last column of the table.

Table 5.6 Comparative results for Test case-1

	W	nf_1	nf_2	$\overline{\mu}$	C_1	C_2	Δ_1	Δ_2
	3	27	36	-0.00850	40.74	69.44	0.9909	1.0298
Case 1	**7**	26	45	-0.00067	11.54	28.89	0.7622	0.8749
	9	25	43	-0.02285	0.0004	41.86	0.7487	0.8384
Mean		**26.00**	**41.33**	**-0.0107**	**17.43**	**46.73**	**0.8339**	**0.9144**

It has been observed from the results in Table 5.6 that the number of non-dominated solutions (nf_1) in the best front of the MOHEA is 26.00 instead of 41.33 for the NSGA-II. In fact, this is an advantage for the decision-maker to get fewer solutions. The negative value of the $\overline{\mu}$ column (-0.0107) shows that F_1 is under F_2 (see Figure 5.10) and the proposed MOHEA with a final front F_1 outperforms NSGA-II with its final front R. Columns 5 and 6 show that 17.43% of solutions in the best front of the hybrid approach are dominated by at least one solution in the NSGA-II front while 46.73% of solutions in the NSGA-II front are dominated by at least one solution in the front of proposed MOHEA. As it is seen from the last columns, the mean value of diversity metric of the MOHEA is smaller than the value of NSGA-II; hence, the divergence of the results by MOHEA toward the Pareto-optimal front is better than the results by NSGA-II (0.8339<0.9144).

5.4.2.3 Test Case 2: Balanced Unreliable Serial Line

This case involves balanced unreliable serial lines with W stations (W=3, 7 and 9) which have the same processing time (T_i) in the line. Similar to the previous case, we use the same dataset proposed by Ho et al. *(*1979) and implemented by Chehade et al. (2010) to test the efficiency of the proposed MOHEA. For the input data of the machine characteristics, the values of these parameters are presented in Table 5.7. The Pareto optimal solutions obtained by our approach and NSGA-II for each W-lined production system are given in the tables in Appendix B2. Moreover, the extrapolated fronts for each case are shown in Figure 5.11.

Table 5.7 Dataset for Test case-2 (Ho et al. 1979)

Workstation	***MTTR*=1/ r_i**	***MTBF*=1/ p_i**	T_i
1	7	20	1
2	7	30	1
3	5	22	1
4	10	22	1
5	9	25	1
6	14	40	1
7	5	23	1
8	8	30	1
9	10	45	1

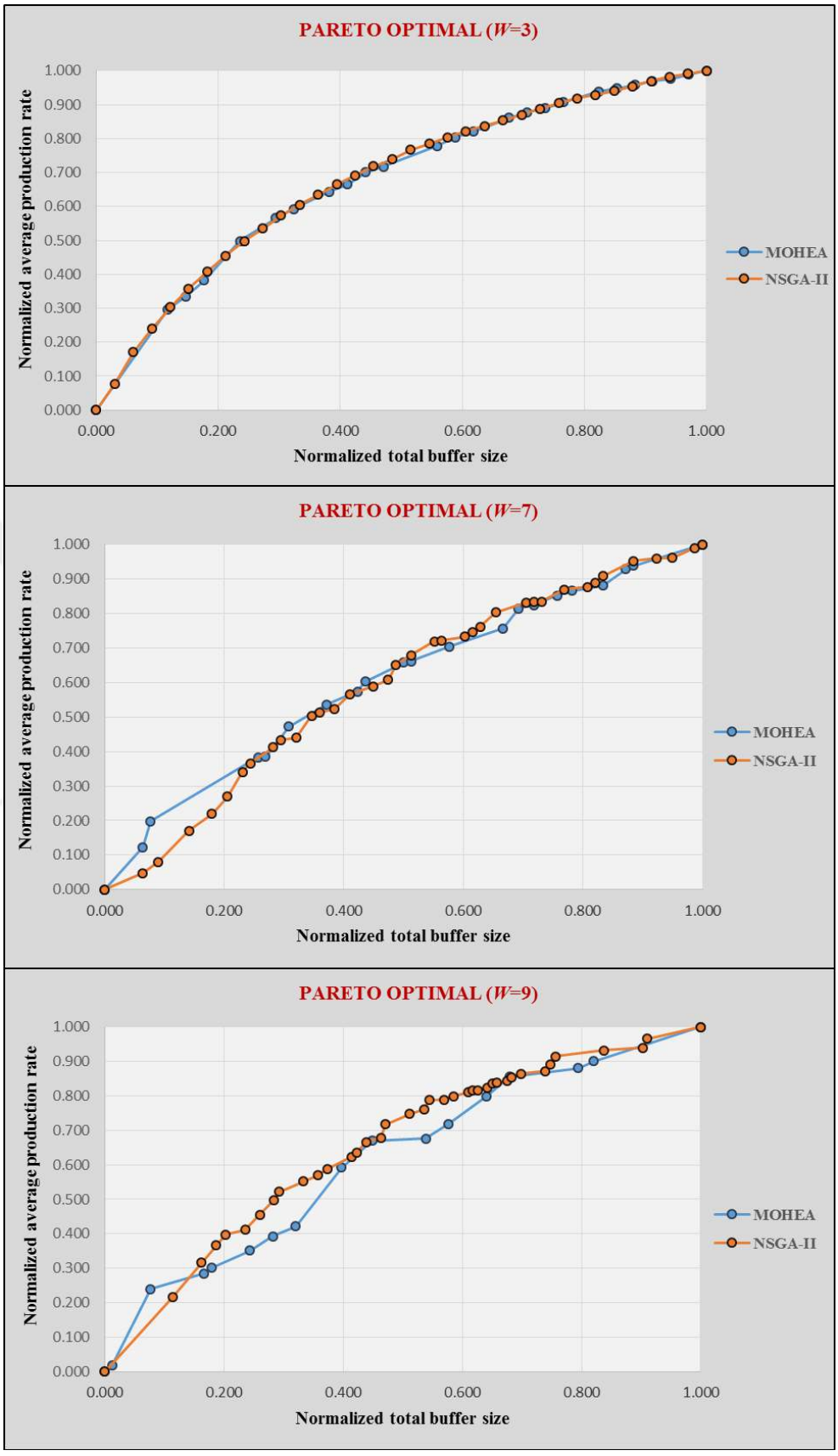


Figure 5.11 Test case 2 - Extrapolated fronts for each case.

Table 5.8 presents the numerical results for Test case 2. Likewise in Test case 1, we can get fewer solutions with the proposed MOHEA compared to the NSGA-II. The number of non-dominated solutions (nf_1) in the best front of the hybrid algorithm is 21.00 instead of 36.33 for the pareto front of NSGA-II. For the Riise distance, the negative value of the $\overline{\mu}$ column (-0.00686) shows that the MOHEA with a final front, F_1, outperforms NSGA-II with its final front R (see Figure 5.11). For the case of W=7, it should be noted that the Riise distance is not negative ($\overline{\mu} = 0.00156$). Since this value is so close 0, it can be concluded that the performances of our proposed approach and NSGA-II are either closer.

Table 5.8 Comparative results for Test case-2

	W	nf_1	nf_2	$\overline{\mu}$	C_1	C_2	Δ_1	Δ_2
Case 2	**3**	25	34	-0.00311	28.00	64.71	0.9376	1.0273
	7	21	37	0.00156	4.76	56.76	0.7155	0.8268
	9	17	38	-0.01903	23.53	50.00	0.7261	0.8953
Mean		**21**	**36.33**	**-0.00686**	**18.76**	**57.16**	**0.7931**	**0.9165**

Moreover, the numerical results show that 18.76% of solutions in the best front of the proposed MOHEA are dominated by at least one solution in the NSGA-II front while 57.16% of solutions in the NSGA-II front are dominated by at least one solution in the front of the proposed MOHEA. As similar in Test case 1, the divergence of the results by the hybrid algorithm toward the Pareto-optimal front is better than the results obtained by NSGA-II (0.7931<0.9165).

5.4.2.4 Test Case 3: Balanced Reliable Serial Line with Upper Bound for Each Buffer

In this test case, an additional constraint of upper bound for each buffer capacity is considered to allow more realistic environment in 10- and 15-workstation production systems. It should be noted that we use the same dataset implemented in the previous chapter as a Test case 2 in solving single objective BAP to compare the performance of the proposed hybrid approach to the traditional NSGA-II. As similar in that case, workstations have exponentially distributed processing times with a mean of 1.0 minute and each buffer size in front of these workstations has an upper

limit denoted by U_i ($0 \leq B_i \leq U_i$, i=1, 2, …, W-1) for environmental limitations. For 10-workstation production line, upper bounds of each buffer are 1, 5, 10, 5, 5, 5, 5, 1 and 5, respectively. It is assumed for 15-workstation production line that upper bounds of each buffer are 1, 1, 5, 10, 10, 10, 5, 5, 5, 5, 1, 1, 5 and 5, respectively. Moreover, the simulation models are run for 100,000 jobs with 10 replications to evaluate the specified performance measures. The extrapolated fronts for 10- and 15-station production lines can be seen in Figure 5.12. Additionally, the Pareto optimal solutions obtained by proposed MOHEA and NSGA-II for each production line are given in the tables in Appendix B3.

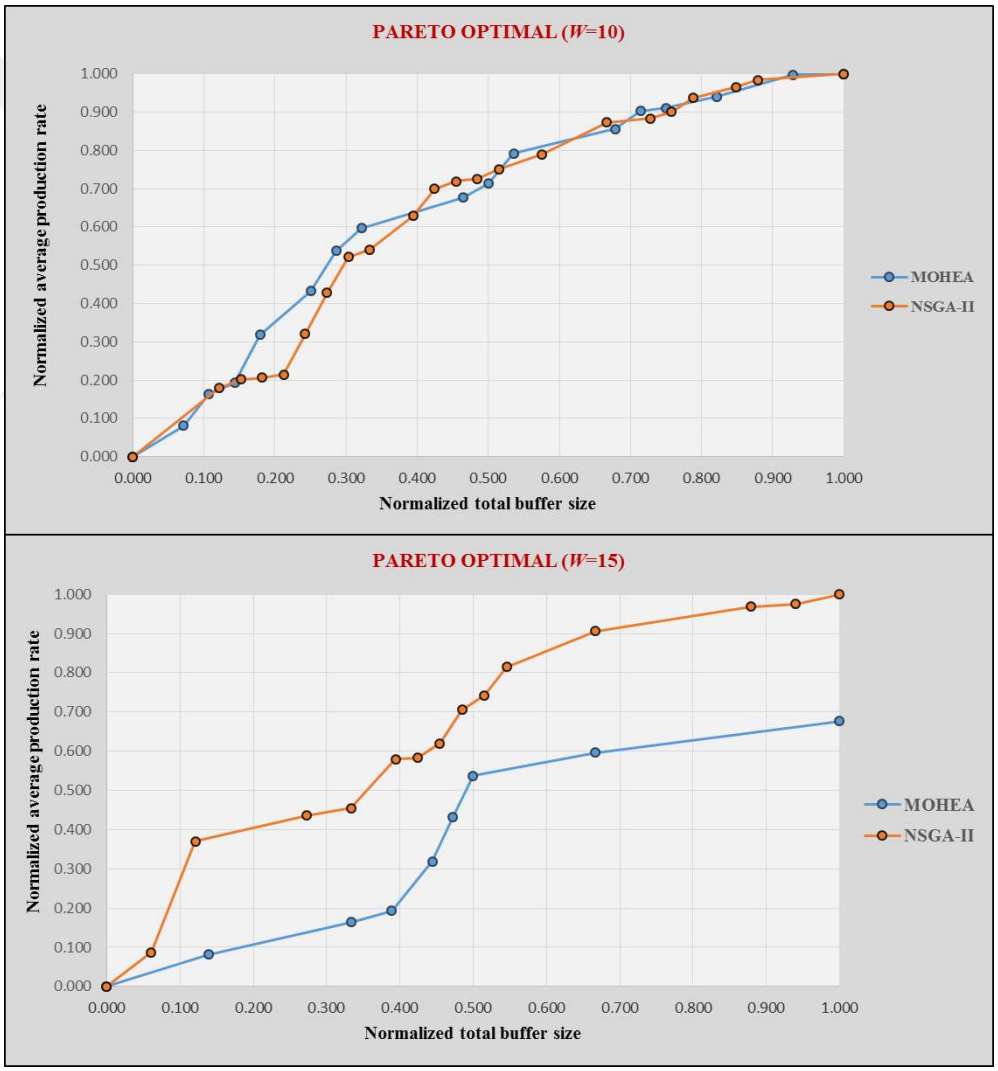


Figure 5.12 Test case 3- Extrapolated fronts for each case.

As can be seen from Table 5.9, the comparative results for Test case 3 show that the number of non-dominated solutions (nf_1) in the best front of the proposed MOHEA is 13.00 instead of 18.50 for the NSGA-II. Hence, we can get fewer solutions with our proposed hybrid approach compared to the NSGA-II.

Furthermore, the Riise distance is negative at average and the distance value shows that F_1 is under F_2. In Figure 5.12, the distances are clear for both production lines. Moreover, 5.89% of solutions in the best front of MOHEA are dominated by at least one solution in the NSGA-II while 37.12% of solutions in the front of NSGA-II are dominated by at least one solution in the proposed MOHEA. As a result of the mean value of diversity metric, the divergence of the results by MOHEA toward the Pareto-optimal front is better than the results by NSGA-II (0.6829<0.8609).

Table 5.9 Comparative results for Test case-3

	W	nf_1	nf_2	$\overline{\mu}$	C_1	C_2	Δ_1	Δ_2
Case 3	**10**	17	22	-0.00219	11.77	40.91	0.6927	0.8525
	15	9	15	-0.04248	0.00	33.33	0.6730	0.8692
Mean		**13**	**18.50**	**-0.02234**	**5.89**	**37.12**	**0.6829**	**0.8609**

5.4.2.5 Test Case 4: Balanced Unreliable Serial Line with Upper Bound for Each Buffer

This case focuses on buffer allocation in balanced unreliable production lines. The line length is varied with 10 and 15-workstation lines for this test case. The upper bounds in this test case are 1, 5, 10, 5, 5, 5, 5, 1, and 5 for a 10-workstation line and 1, 1, 5, 10, 10, 10, 5, 5, 5, 5, 1, 1, 5, and 5 for a 15-workstation line. Furthermore, the processing times of the workstations are assumed to be same and constitute one time unit. Additionally, the workstations have a coefficient of variation for the total time a job spends in a workstation with 0.575. The workstations are subject to breakdown, and the failure and repair times of the workstations are exponentially distributed with a mean of 100 and 25 minutes, respectively. The simulation models are run for 10 runs of 100,000 jobs each. It should be noted that all simulation runs are conducted under the same experimental conditions. Figure 5.13 depicts the extrapolated fronts

for both 10- and 15-station production lines and the Pareto optimal solutions obtained by proposed MOHEA and NSGA-II are given in the tables in Appendix B4.

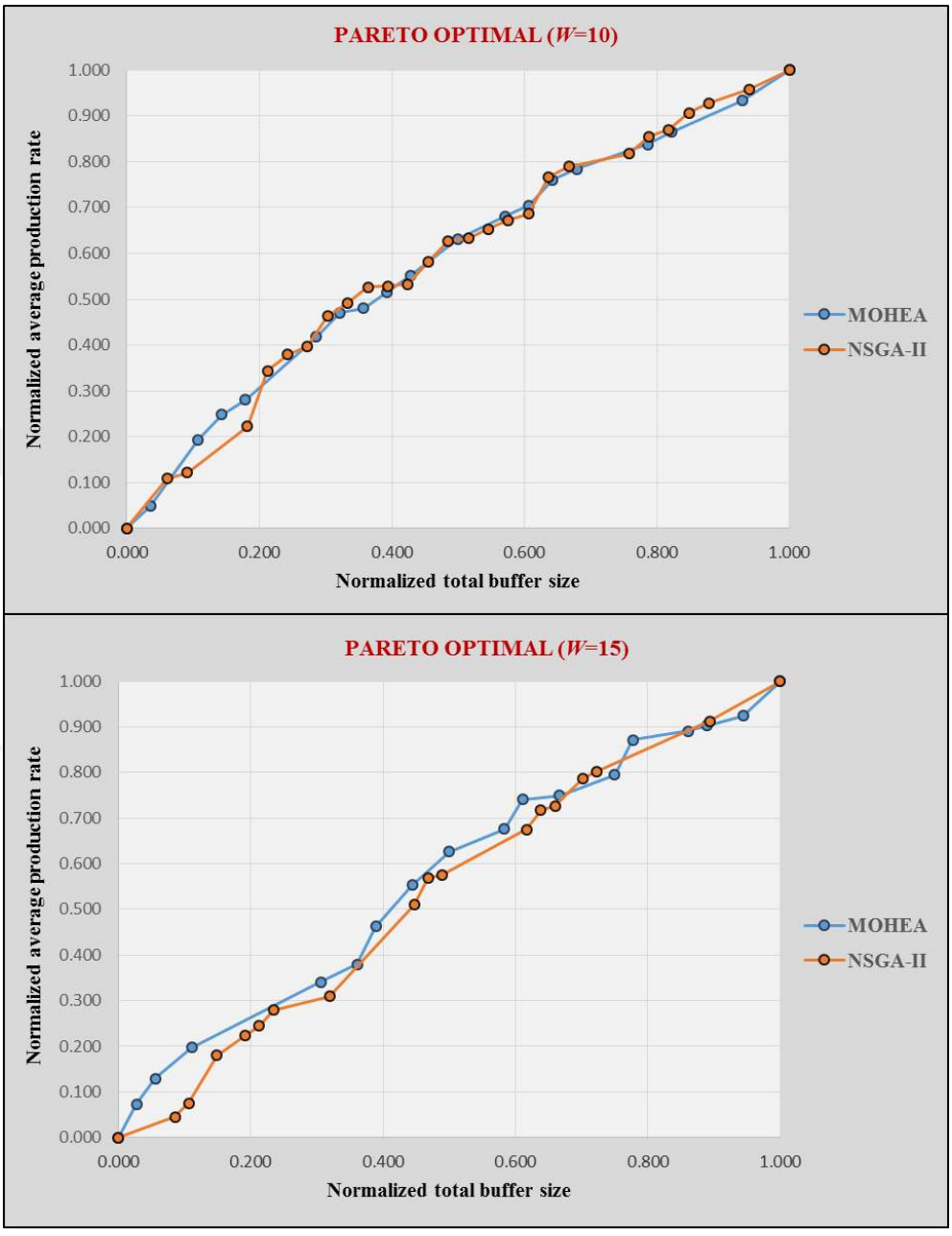


Figure 5.13 Test case 4 - Extrapolated fronts for each case.

Table 5.10 shows the comparative results obtained by the proposed MOHEA and the NSGA-II. The number of non-dominated solutions (nf_1) in the best front of MOHEA is 19.00 instead of 23.00 for the NSGA-II.

Table 5.10 Comparative results for Test case-4

	W	nf_1	nf_2	$\overline{\mu}$	C_1	C_2	Δ_1	Δ_2
Case 4	**10**	20	27	-0.00196	15.00	66.67	0.7260	0.9033
	15	18	19	0.01498	5.56	36.84	0.7495	0.8661
Mean		**19**	**23**	**0.00651**	**10.28**	**51.76**	**0.7378**	**0.8847**

For this test case, the Riise distance value is not negative at average. Since in the case of W=15, there is a slight difference (0.01498) on the distance, it means that NSGA-II shows a bit more performance compared to the proposed hybrid approach. The extrapolated fronts in Figure 5.13 present the positive distances that F_1 is not under F_2 at almost. Moreover, 10.28% of solutions in the best front of hybrid algorithm are dominated by at least one solution in the NSGA-II while 51.76% of solutions in the front of NSGA-II are dominated by at least one solution in the hybrid algorithm. As a result of the mean value of diversity metric, the divergence of the results by the hybrid algorithm toward the Pareto-optimal front is better than the results by NSGA-II (0.7378<0.8847).

5.5 Chapter Summary

In this chapter, we proposed a hybrid evolutionary algorithm-based simulation optimization approach by combining the key advantages of simulation, meta-heuristics, and hybridization to solve the BAP in a multi-objective manner. The proposed MOHEA combines two multi-objective optimization algorithms, namely NSGA-II and MOSA. Based on Pareto dominance relationship, the proposed hybrid approach is employed using simulations in an iterative manner to find to optimal/near optimal buffer configurations that provides maximum average production rates with minimum total buffer sizes simultaneously in production lines. The main aim of hybridization in this study is to increase the performance of the algorithm in terms of solution quality and computing efficiency. Hence, the hybrid approach blends both NSGA-II and MOSA into a single approach to retain the strengths of these approaches. Moreover, NSGA-II is a well-known evolutionary multi-objective algorithm working with population-based solutions, using crossover, mutation operators. In the MOHEA, the performance of NSGA-II has been enhanced

with the adaptation of an acceptance test in MOSA using Nam and Parks' modified acceptance rule (2000) as the replacement scheme in NSGA-II. As a result, the hybridization leads to better solutions relative to those achieved by the traditional NSGA-II examined in this study.

In order to test the performance of the proposed MOHEA, a comprehensive comparative study is performed under different serial line conditions *i.e.,* balanced, unbalanced, reliable, unreliable and upper limits for buffers. Considering four different comparison criteria used in the literature, which are the number of solutions in the optimal front $n(f_i)$, the distance of Riise, the Zitlzler measure (C_i) and the diversity metric (Δ), the results obtained by the MOHEA are compared to the results of the traditional NSGA-II. The experimental results show that the performance of our proposed approach dominates NSGA-II. Non-dominated solutions obtained by the MOHEA in all cases give a valuable idea to the decision-maker. Moreover, the divergence of the results by the proposed MOHEA toward the Pareto-optimal front is better than the results by NSGA-II in all cases. As a result, we can conclude that our proposed multi-objective hybrid approach has a considerable potential to minimize the total buffer space by appropriately allocating space to each buffer while maximizing average production rate in serial production lines.

CHAPTER SIX
CONCLUSION

6.1 Summary of the Thesis

The optimization of production systems is one of the most challenging tasks. The buffer allocation problem is an important optimization problem faced by production system designers. Generally, the problem deals with finding optimal buffer sizes to be allocated into buffer areas in a production system.

Buffers have a significant impact on capacity improvement through the production line by eliminating the disruptive effects in the system. Owing to its importance and complexity, the problem has received considerable attention from many researchers. The most prominent studies on the BAP are based on design rules (or heuristics) and numerical algorithms to solve the problem as stated in Chapter 2. Design rules can assist the decision maker in observing the effects of proposed buffer distributions on the production line without undertaking complex analysis. On the other hand, the numerical algorithms are developed to optimize buffer sizes subject to production constraints such as floor space, budget, scaled production rate in the system. Among the numerical algorithms, meta-heuristics have proven to be highly effective in solving combinatorial problems in recent years. Such algorithms explore the search space efficiently in order to find optimal/near-optimal solutions. Moreover, to increase the performance of the meta-heuristics, a recent trend observed in the literature is to hybrize meta-heuristics to solve any combinatorial optimization problem. Based on the extensive literature survey of approaches developed for BAPs given in Chapter 2, it can be stated that there is a growing need for effective hybrid meta-heuristics.

As far as the performance estimation for a production system through the optimization process is concerned, simulation plays an important role that allows the production systems designers to understand system performance and assist in behavior prediction. As production lines get larger and more complex as in real production systems, the complexities necessitate simulation in order to estimate

performance measures. Hence, using simulation in conjunction with an optimization method is a useful tool for solving problems related to the design of a production system.

Regarding to these research directions, this Ph.D study primarily aimed at solving the BAPs using meta-heuristics-based simulation optimization approaches in serial production lines.

Generally, the BAP can be formulated with different optimality criteria. *In this Ph.D study, we dealed with both single objective BAP and multi-objective BAP.* In this respect, the problem to be considered was twofold. First, a hybrid meta-heuristics-based simulation optimization approach was proposed to solve single objective BAP in which attempted to maximize average production rate with given a certain fixed amount of buffers. The proposed hybrid approach combined GA with SA algorithm and its performance was investigated with an experimental study. The production systems through the experimental study were realized on small, medium, large and very large production lines considering different serial line conditions, *i.e.,* balanced, unbalanced, reliable, unreliable, lower and upper limits for buffers and some bottleneck position conditions. As a result, it could be clearly seen after extensive computational experimentation that this hybridization was capable of improving the capacity of production lines.

Second, we handled the problem to solve from a multi-objective perspective and optimize two conflicting objectives - average production rate maximization and total buffer size minimization in serial production lines, simultaneously. A hybrid evolutionary algorithm-based simulation optimization approach by combining the key advantages of simulation, NSGA-II and special version of SA algorithm (MOSA). In the hybrid approach, the aim was to enhance the performance of NSGA-II with the adaptation of the acceptance test in MOSA. Following a design of experiments to identify the best values/scheme for the MOHEA control parameters, four comparison criteria used in the multi-objective optimization literature were introduced. Using these criteria, comparative experiments based on Pareto

dominance relationship were carried out under different serial line conditions, *i.e.*, balanced, unbalanced, reliable, unreliable to evaluate the performance of the hybrid approach. As a result of this comparative study, it could be stated that the hybridization had a noticeable effect on the performance of the solution approach.

6.2 Contributions

All in all, the original contributions achieved with this thesis can be summarized as follows:

- This study contributes to the single objective BAP literature by hybridizing two meta-heuristics – GA and SA algorithm- to solve the problem. It has been noted that there are only two studies emloying hybrid approach for buffer size optimization in the literature. While Shi & Men (2003) combines a meta-heuristic (namely, tabu search) with the nested partitions method to solve the BAP, Dolgui et al. (2007) hybridizes a meta-heuristic (namely, GA) and branch-and-bound approaches employing as a search method for BAP. Unlike these studies, we introduced a hybridization of two meta-heuristics and this is the first extensive study hybridizing two metaheuristics for BAP.

- To test the performance of proposed hybrid approach, the test cases were realized on different production line configurations. It has been noted that the previous studies on hybrid approaches (Dolgui et al., 2007; Shi & Men, 2003) were attempted to solve BAP in only unreliable production lines. More spesifically, we focus on the BAP in both reliable and unreliable production lines, not only in small-sized and medium-sized, but also large-sized and very large-sized production lines.

- The BAP was considered with buffer space constraints for each buffer location and bottleneck(s) positioned in some test cases. Hence, these parameters made the problem more interesting to the production system designers. Due to the restrictive assumptions, the BAP under different system characteristics realistically represents most of the real-world production lines.

- Considering the capability of modeling large and complex systems by simulation, we modeled the production lines using discrete-event simulation modeling. In the context of simulation optimization, the models were integrated with the codes of generative methods (MATLAB codes). We proceeded to compile both of methods (evaluative-generative methods) and run the code as a standalone application.

- Dealing with the BAP under the objective of average production rate maximization, an efficient initialization procedure based on a pre-specified total buffer size (K) that must be allocated among the W-1 buffer locations in the system was presented to generate the initial population for the proposed hybrid approach.

- Old buffer designs constructed in the BAP literature such as bowl phenomenon, storage bowl phenomenon, Freeman's downtime rule, etc. were also proven to be effective for solving single objective BAP through the computational experiments.

- The problem was also extended to be solved in a multi-objective manner while most of the current relevant studies consider only one objective as a maximization or minimization. From the multi-objective perspective, we handled the problem with two conflicting objectives that are average production rate maximization and total buffer size minimization simultaneously for serial production lines.

- In the literature of multi-objective BAP, this is the first extensive study that attempts to hybridize two meta-heuristics, namely NSGA-II and MOSA. It has been noted that there is only one study in which a hybrid approach combining a meta-heuristic (namely, GA) with line search proposed for unreliable production lines as a generative method by Amiri & Mohtashami (2012) for multi-objective buffer size optimization in the literature.

- Unlike the test cases in the study by Amiri & Mohtashami (2012), the production system under consideration in this Ph.D thesis comprised of both reliable and unreliable production lines and the performance of proposed multi-objective hybrid approach was tested under different production system characteristics.

6.3 Future Research Directions

Several extensions to this research are possible; hence, some of the future research directions can be stated as follows:

- This study will be extended to determine what new features can be added to the proposed hybrid GAA and MOHEA so that the potential of the algorithms can be improved. For instance, new genetic operators, i.e., crossover and mutation, new selection or termination schemes, different neighborhood strategies may be introduced to specialize the GAA while solving single objective BAP and MOHEA while solving multi-objective BAP.

- Researches may evolve towards formulating and solving generalized BAPs with different additional characteristics, such as cost/profit functions, different optimality criteria and others.

- Moreover, the problem considered herein can be tackled with other production design problems simultaneously such as line balancing problem, server allocation problem, workload allocation problem, etc.

- To demonstrate the effectiveness of proposed hybrid approaches for other models of production lines, it would be valuable to implement the algorithms to production systems with complex structures, such as assembly systems and split-merge systems.
- Another further research might address an extensive statistical analysis on limitations for the formulation of the problem. More specifically, it would be interesting to production systems designers to carry out a detailed analysis on efficient upper bounds for each buffer areas and bottleneck analysis while solving the BAPs.

- From the perspective of multi-objective optimization, the results of Pareto optimization methods amount to a Pareto set of non-dominated solutions which the decision maker can select each of them on the basis of his need. Since the set

of non-dominated solutions is usually large, and the corresponding representative Pareto frontier in the objective function space is very crowded, to make selection of the best design from a family of Pareto solutions is so difficult in this context. In dealing with this problem, we can utilize data-mining and visualization techniques for supporting the process of decision making as a future research work.

- Lastly, among the decision making strategies, decision making during the search can also be investigated to observe the effect of buffer allocation on the production line while solving multi-objective BAP. Thus, the decision maker can articulate preferences during the interactive optimization process.

As a corollary, the main objective of this research is to propose robust approaches for buffer allocation in serial production lines. Employing these approaches in a real-world environment, the proposed approaches can provide the production systems designers with valuable information about efficient buffer size management. Concerning the future research work, we intend to integrate these metaheuristics-based simulation optimization procedures into a decision support system framework with a user-friendly interface. In doing so, the use of the proposed hybrid approaches in an industrial environment will be eased and also the capacity improvement will be achieved with up-to-date data.

REFERENCES

Aarts, E. H. L., & Korst, J. H. M. (1989). *Simulated annealing and boltzmann machines*. Chichester: Wiley.

Abdul-Kader, W., Ganjavi, O., & Baki, F. (2011). A nonlinear model for optimizing the performance of a multi-product production line. *International Transactions in Operational Research, 18* (5), 561-577.

Aksoy, K. H., & Gupta, S. M. (2005). Buffer allocation plan for a remanufacturing cell. *Computers&Industrial Engineering, 48,* 657-677.

Aksoy, K. H., & Gupta, S. M. (2010). Near optimal buffer allocation in remanufacturing systems with *N*-policy. *Computers&Industrial Engineering*, *59*, 496-508.

Alon, G., Kroese, D. P., Raviv, T., & Rubinstein. R. Y. (2005). Application of the cross-entropy method to the buffer allocation problem in a simulation-based environment. *Annals of Operations Research*, *134* (1)*,* 137-151.

Almomani, M. H, Abdul Rahman, R., Baharum, A., & Alrefaei, M. H. (2012). A selection approach for solving buffer allocation problem. *International Journal of the Physical Sciences*, *7* (3)*,* 413-422.

Anwar sadath, M. A., & Manoj, P. J. (2015). Optimization of a bi-functional APP problem by using multi-objective genetic algorithm (NSGA-II). *International Research Journal of Engineering and Technology (IRJET)*, *2* (3)*,* 2113-2117.

Altiok, T., & Stidham, S. (1983). The allocation of interstage buffer capacities in production lines. *IIE Transactions*, *18,* 251-261.

Altiparmak, F., Dengiz, B., & Bulgak, A. A. (2007). Buffer allocation and performance modeling in asynchronous assembly system operations: An artificial neural network metamodeling approach. *Applied Soft Computing*, *7,* 946-956.

Amiri, M., & Mohtashami, A. (2012). Buffer allocation in unreliable production lines based on design of experiments, simulation, and genetic algorithm. *International Journal of Advanced Manufacturing*, *62* (1)*,* 371-383.

Back, T., Fogel, D. B., & Michalewicz, Z. (1997). *Handbook of Evolutionary Computation*. New York: Institute of Physics Publishing, Bristol and Oxford University Press.

Battini, D., Persona, A., & Regattieri**,** A. (2009). Buffer size design linked to reliability performance: A simulative study. *Computers&Industrial Engineering*, *56* (4)*,* 1633-1641.

Bekker, J. (2013). Multi-objective buffer space allocation with the cross-entropy method. *International Journal of Simulation Modelling*, *12* (1)*,* 50-61.

Bergey, P. K., Ragsdale, C. T., & Hoskote, M. (2003). A simulated annealing genetic algorithm for the electrical power districting problem. *Annals of Operations Research*, *121*(1)*,* 33-55.

Blum, C., Roli, A., & Alba, E. (2005). *An introduction to metaheuristic techniques*. In: Parallel Metaheuristics, A New Class of Algorithms. John Wiley.

Bulgak, A. A., Diwan, P. D., & Inozu, B. (1995). Buffer size optimization in asynchronous assembly systems using genetic algorithms. *Computers&Industrial Engineering*, *28* (2)*,* 309–322.

Bulgak, A. A. (2006). Analysis and design of split and merge unpaced assembly systems by metamodelling and stochastic search. *International Journal of Production Research*, *44* (18-19)*,* 4067-4080.

Buzacott, J. A. (1967). Automatic transfer lines with buffer stocks. *International Journal of Production Research, 5*, 183-200.

Buzacott, J. A., & Hanifin, L. E. (1978). Models of automatic transfer lines with inventory banks: A review and comparison. *AIIE Transactions*, *10,* 197-207.

Buzacott, J.A., & Shanthikumar, J. G. (1993). *Stochastic models of manufacturing systems*. Michigan: Prentice Hall.

Can, B., & Heavey, C. (2011). Comparison of experimental designs for simulation-based symbolic regression of manufacturing systems. *Computers&Industrial Engineering*, *61* (3)*,* 447–462.

Can, B., & Heavey, C. (2012). A comparison of genetic programming and artificial neural networks in metamodeling of discrete-event simulation models. *Computers&Operations Research, 39,* 424-436.

Carson, Y., & Maria, A. (1997). Simulation optimization: Methods and applications. In Andraróttir S., Healy K.J., Withers D.H., Nelson B.L.: *Proceedings of the 1997 Winter Simulation Conference, USA,* 118-126.

Chaharsooghi, S. K., & Nahavandi, N. (2003). Buffer allocation problem, a heuristic approach. *Scientia Iranica, 10* (4)*,* 401-409.

Chehade, H., Yalaoui, F., Amodeo, L., & Dugardin, F. (2010). Buffers sizing in assembly lines using a Lorenz multiobjective ant colony optimization algorithm. *IEEE International Conference on Machine and Web Intelligence*, 283-287.

Colledani, M., Ekvall, M., Lundholm, T, Moriggi, P., Polato A., & Tolio, T. (2010). Analytical methods to support continuous improvements at Scania. *International Journal of Production Research*, *48* (7)*,* 1913-1945.

Conway, R. W., Maxwell, W. L., McClain, J. O., & Thomas, L. J. (1988). The role of work-in-process inventories in serial production lines. *Operations Research*, *8,* 1183-1196.

Costa, A. Alfieri, A. Matta, A., & Fichera, S. (2015). A parallel tabu search for solving the primal buffer allocation problem in serial production systems. *Computers&Operations Research*, *64*, 97-112.

Cruz, F. R. B., Duarte, A. R., & Van Woensel, T. (2008). Buffer allocation in general single-server queueing networks. *Computers&Operations Research*, *35* (11), 3581-3598.

Cruz, F. R. B., Van Woensel, T., & MacGregor Smith, J. (2010). Buffer and throughput trade-offs in M/G/1/K queueing networks: A bi-criteia approach. *International Journal of Production Economics*, *125,* 224-234.

Cruz, F. R. B., Kendall, G., While, L., Duarte, A. R., & Brito, N. L. C. (2012). Throughput maximization of queueing networks with simultaneous minimization of service rates and buffers. *Mathematical Problems in Engineering*, DOI: 10.1155/2012/692593.

Czyz˙ak, P., & Jaszkiewicz, A. (1998). Pareto simulated annealing – a metaheuristic technique for multiple- objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis, 7*, 34–47.

Dallery, Y., & Gershwin, S. B. (1992). Manufacturing flow line systems: a review of models and analytical results. *Queueing Systems*, *12* (1-2), 3-94.

Deb, K., & Agrawal, R. B. (1995). Simulated binary crossover for continuous search space. *Complex Systems*, *9* (1), 115–148.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective optimization genetic algorithm: NSGA II. *IEEE Transactions on Evolutionary Computation*, *6* (2), 182-197.

Demir, L., Tunali, S., & Løkketangen, A. (2011). A tabu search approach for buffer allocation in production lines with unreliable machines. *Engineering Optimization*, *43* (2), 213-231.

Demir, L., Tunali, S., & Eliiyi, D. T. (2012). An adaptive tabu search approach for buffer allocation problem in unreliable non-homogenous production lines. *Computers&Operations Research*, *39* (7), 1477–1486.

Demir, L., Tunali, S., Eliiyi, D. T., & Løkketangen, A. (2013). Two approaches for solving the buffer allocation problem in unreliable production lines. *Computers&Operations Research*, *40* (10), 2556-2563.

Demir, L., Tunali, S., & Eliiyi, D. T. (2014). The state of the art on buffer allocation problem: A comprehensive survey. *Journal of Intelligent Manufacturing*, *25* (3), 371-392.

Diamantidis, A. C., & Papadopoulos, C. T. (2004). A dynamic programming algorithm for the buffer allocation problem in homogeneous asymptotically reliable serial production lines. *Mathematical Problems in Engineering*, *3,* 209-223.

Dolgui, A., Eremeev, A., Kolokolov, A., & Sigaev, V. (2002). A genetic algorithm for the allocation of buffer storage capacities in a production line with unreliable machines. *Journal of Mathematical Modeling and Algorithms*, *1*, 89-104.

Dolgui, A., Eremeev, A., & Sigaev, V. (2007). HBBA: hybrid algorithm for buffer allocation in tandem production lines. *Journal of Intelligent Manufacturing*, *18*, 411-420.

Dolgui, A., Eremeev, A., Kovalyov, M.Y., & Sigaev, V. (2013). Complexity of buffer capacity allocation problems for production lines with unreliable machines. *Journal of Mathematical Modelling and Algorithms in Operations Research, 12* (2), 155-165.

Dowsland, K. A., & Thompson, J. M. (2012). *Simulated annealing*. In Rozenberg, G., Back, T., & Kok, J., eds., Handbook of Natural Computing. Springer, 1623–1655.

Dugardin, F., Yalaoui, F., & Amodeo, L. (2010). New multi-objective method to solve re-entrant hybrid flow shop scheduling problem. *European Journal of Operation Research*, *203*, 22–31.

Enginarlar, E., Li, J., Meerkov, S. M., & Zhang, R. Q. (2002). Buffer capacity for accommodating machine downtime in serial production lines. *International Journal of Production Research*, *40*, 601–624.

Enginarlar, E., Li, J., & Meerkov, S. M. (2005). How lean can lean buffers be? *IIE Transactions*, *37*, 333-342.

Feoktistov, V. (2006). *Differential evolution: In search of solutions.* Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Fogel, D. B. (1992). *Evolutionary programming society.* In D. B. Fogel and W. Atmar (Eds.), Proceedings of the First Annual Conference on Evolutionary Programming, La Jolla, CA.

Freeman, M. C. (1964). The effects of breakdowns and interstage storage on production line capacity. *Journal of Industrial Engineering*, *15*, 194-200.

Fu, M. C. (2015). *Handbook of Simulation Optimization.* International Series in Operations Research & Management Science, 216, USA: Springer.

Ganesh, K., & Punniyamoorthy, M. (2005). Optimization of continuous-time production planning using hybrid genetic algorithms-simulated annealing. *International Journal of Advanced Manufacturing Technology*, *26* (1)*,* 148-154.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6,* 721–741.

Gen, M., & Cheng, R. (1997). *Genetic algorithms and engineering design.* New York: John Wiley and Sons.

Gershwin, S. B. (1987). An efficient decomposition method for the approximate evaluation of tandem queues with finite storage space and blocking. *Operations Research*, *35* (2)*,* 291-305.

Gershwin, S. B., & Schor. J.E. (2000). Efficient algorithms for buffer space allocation. *Annals of Operations Research*, *93,* 117-144.

Gershwin, S. B., & Werner, M. L. (2007). An approximate analytical method for evaluating the performance of closed-loop flow systems with unreliable machines and finite buffers. *International Journal of Production Research*, *45* (14)*,* 3085-3111.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* MA: Addison-Wesley: Reading.

Goldberg, D. E., & Deb, K. (1991). *A comparative analysis of selection schemes used in genetic algorithms.* In G. Rawlins, editor, Foundations of Genetic Algorithms, 69-93, San Mateo, Morgan Kaufmann.

Gross, D., & Harris, C. (1985). *Fundamentals of queueing theory.* New York: Wiley.

Hajek, B. (1988). Cooling schedules for optimal annealing. *Mathematics of Operations Research*, *13,* 311–329.

Hapke, M., Jaszkiewicz, A., & Slowinski, R. (2000). Pareto simulated annealing for fuzzy multi-objective combinatorial optimization. *Journal of Heuristics, 6* (3)*,* 329-345.

Harris, J. H., & Powell, S. G. (1999). An algorithm for aptimal buffer placement in reliable serial lines. *IIE Transactions, 31*, 287-302.

Helber, S. (1999). *Performance analysis of flow lines with non-linear flow of material.* Lecture Notes in Economics and Mathematical Systems. Berlin: Springer-Verlag.

Helber, S. (2001). Cash-flow-oriented buffer allocation in stochastic flow lines. *International Journal of Production Research*, *39* (14)*,* 3061-3083.

Hillier, F. S., & Boling, R. W. (1966). The effect of some design factors on the efficiency of production lines with variable operation times. *Journal of Industrial Engineering*, *17*, 651-658.

Hillier, F. S., & Boling, R. W. (1967). Finite queues in series with exponential or Erlang service times – A numerical approach. *Operations Research*, *15*, 286-303.

Hillier, F. S., & Boling, R. W. (1977). *Toward characterizing the optimal allocation of work in production line systems with variable operation times.* In: Roubens, M. (ed), Advances in Operations Research, proc Euro, II. North Holland: Amsterdam.

Hillier, F. S., & Boling, R. W. (1979). On the optimal allocation of work in symmetrically unbalanced production line systems with variable operation times. *Management Science*, *25*, 721-728.

Hillier, F., & So, K. C. (1991a). The effect of the coeffcient of variation of operation times on the allocation of storage space in production line systems. *IIE Transactions*, *23*, 198-206.

Hillier, F., & So, K. C. (1991b). The effect of machine breakdowns and interstage storage on production line systems. *Internal Journal of Production Research*, *29*, 2043-2055.

Hillier, F., So, K. C. & Boling, R. W. (1993). Notes: Towards characterizing the optimal allocation of storage space in production line systems with variable processing times. *Management Science, 39* (1)*,* 126-133.

Hillier, S. M. (2000). Characterizing the optimal allocation of storage space in production line systems with variable processing times. *IIE Transactions*, *32*, 1-8.

Hillier, S. M., & Hillier F. S. (2006). Simultaneous optimization of work and buffer space in unpaced production lines with random processing times. *IIE Transactions*, *38*, 39-51.

Himmelblau, D. (1987). *Applied nonlinear programming*. New York: McGraw-Hill.

Ho, Y.C., Eyler, M.A., & Chien, T.T. (1979). A gradient technique for general buffer storage design in a production line. *International Journal of Production Research*, *17* (2), 557-580.

Holland, J. H. (1975/1992). *Adaptation in natural and artificial systems*. Cambridge, MA: MIT Press.

Horn, J. (1997). *F1.9 multicriteria decision making*. In T. Back, D. B. Fogel, and Z. Michalewicz (Eds.), Handbook of Evolutionary Computation. Bristol (UK): Institute of Physics Publishing.

Hu, X.-B., & Di Paolo, E. (2007). An efficient genetic algorithm with uniform crossover for the multi-objective airport gate assignment problem. *In: IEEE Congress on Evolutionary Computation*, 55–62, Singapore.

Huang, M-G., Chang, P-L., & Chou Y-C. (2002). Buffer allocation in flow-shop-type production systems with general arrival and service patterns. *Computers&Operations Research*, *29*, 103-121.

Hwang, C.-L., & Masud, A. S. M. (1979). *Multiple objective decision making - methods and applications a state-of-the-art survey.* Lecture Notes in Economics and Mathematical Systems. Berlin: Springer-Verlag.

Jeong, K.-C., & Kim, Y.-D. (2000). Heuristics for selecting machines and determining buffer capacities in assembly systems. *Computers&Industrial Engineering*, *38*, 341-360.

Kirkpatrick, S., Gelatt, C. D. Jr., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220* (4598), 671-680.

Kose, S. Y. (2010). *Capacity improvement in a real manufacturing system using a hybrid simulation/genetic algorithm approach.* M.Sc. Thesis, Dokuz Eylul University, Izmir.

Kose, Y. S., Demir, L., Tunali, S., & Eliiyi, D. T. (2015). Capacity improvement using simulation optimization approaches: A case study in the thermo-technology industry. *Engineering Optimization*, *47* (2)*,* 149-164.

Kwon, S-T. (2006). On the optimal buffer allocation of an FMS with finite in-process buffers. *Lecture Notes in Computer Science, 3982,* 767-776.

Kumar, A., Saxena, R., & Kumar, A. (2014). A comparative study of the various genetic approaches to solve multi-objective optimization problems. *IEEE, Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014 International Conference*, 109-112.

Lee, H-T., Chen, S-K., & Shunder Chang S. (2009). A meta-heuristic approach to buffer allocation in production line. *Journal of Chung Cheng Institute of Technology*, *38* (1)*,* 167-178.

Liu, C., & Tu., F. S. (1994). *Buffer allocation via the genetic algorithm. In: Proceedings of the 33rd Conference on Decision and Control.* USA: Florida, 609–610.

Li, L., Qian, Y., Yang, Y. M., & Du, K. (2015). A fast algorithm for buffer allocation problem, *International Journal of Production Research*. DOI: 10.1080/00207543.2015.1092612.

Lutz, C. M., Davis, K. R., & Sun, M. (1998). Determining buffer location and size in production lines using tabu search. *European Journal of Operational Research*, *106,* 301-316.

MacGregor Smith, J., & Daskalaki S. (1988). Buffer space allocation in automated assembly lines. *Operations Research*, *36* (2)*,* 343–358.

MacGregor Smith, J., & Cruz, F. R. B. (2005). The buffer allocation problem for general finite buffer queueing networks. *IIE Transactions*, *37* (4)*,* 343-365.

Massim, Y., Yalaoui, F., Amodeo, L., Chatelet, E., & Zeblah, A. (2010). Efficient combined immune-decomposition algorithm for optimal buffer allocation in production lines for throughput and profit maximization. *Computers&Operations Research*, *37* (4), 611-620.

Matta, A., Runchina, M., & Tolio, T. (2005). Automated flow lines with shared buffer. *OR Spectrum*, *27,* 265-286.

McNamara, T., Shaaban, S., & Hudson, S. (2013) Simulation of unbalanced buffer allocation in unreliable unpaced production lines, *International Journal of Production Research*, *5* (6), 1922-1936.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, *21* (6), 1087-1092.

Mitchell, M. (1996). *An introduction to genetic algorithms*. USA: MIT Press.

Nahas, N., Ait-Kadi, D., & Nourelfath, M. (2006). A new approach for buffer allocation in unreliable production lines. *International Journal of Production Economics*, *103*, 873-881.

Nahas, N., Ait-Kadi, D., & Nourelfath, M. (2009). Selecting machines and buffers in unreliable series-parallel production lines. *International Journal of Production Research*, *47* (14), 3741-3774.

Nahas, N. (2014). Buffer allocation and preventive maintenance optimization in unreliable production lines. *Journal of Intelligent Manufacturing*, 1-9.

Nam, D. K., & Park, C. H. (2000). Multiobjective simulated annealing: a comparative study to evolutionary algorithms. *International Journal of Fuzzy Systems*, *2* (2), 87–97.

Nearchou, A. C. (2004). A novel metaheuristic approach for the flow shop scheduling problem. *Engineering Applications of Artificial Intelligence*, *17*, 289–300.

Nourelfath, M., Nahas, N., & Ait-Kadi, D. (2005). Optimal design of series production lines with unreliable machines and finite buffers. *Journal of Quality in Maintenance Engineering*, *11* (2)*,* 121-138.

Oysu, C., & Bingul, Z. (2009). Application of heuristic and hybrid-GASA algorithms to tool-path optimization problem for minimizing air time during machining. *Engineering Applications of Artificial Intelligence*, *22,* 389–396.

Papadopoulos, H. T., Heavey. C., & Browne, J. (1993). *Queueing theory in manufacturing systems analysis and design*. London: Chapman and Hall.

Papadopoulos, H. T., & Heavey, C. (1996). Queueing theory in manufacturing systems analysis and design: A classification of models for production and transfer lines, *European Journal of Operational Research*, *92*, 1-27.

Papadopoulos, H. T., & Vidalis, M. I. (2001). A heuristic algorithm for the buffer allocation in unreliable unbalanced production lines. *Computers&Industrial Engineering*, *41,* 261-277.

Papadopoulos, C. T., O'Kelly, M. E. J., Vidalis, M. J., & Spinellis, D. (2009). A*nalysis and design of discrete part production lines*. New York: Springer Science+Business Media.

Papadopoulos C.T., O'Kelly, M. E. J., & Tsadiras, A.K. (2013). A DSS for the buffer allocation of production lines based on a comparative evaluation of a set of search algorithms, *International Journal of Production Research*, *51,* 4175-4199.

Powell, M.J.D. (1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, *7*, 155–162.

Powell, S. G. (1994). Buffer allocation in unbalanced three-station serial lines. *International Journal of Production Research*, *32*, 2201-2217.

Powell, S. G., & Pyke. D. F. (1996). Allocation of buffers to serial production lines with bottlenecks. *IIE Transactions*, *2,* 18–29.

Price, K. (1994). Genetic annealing. *Dr. Dobb's Journal*, *October*, 127-132.

Qudeiri, J. A., Yamamoto, H., Ramli, R., & Al-Momani, K.R. (2007). Development of production simulator for buffer size decisions in complex production systems using genetic algorithms. *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, *1* (3)*,* 418-429.

Qudeiri, J. A., Yamamoto, H., Ramli, R., & Jamali, A. (2008). Genetic algorithm for buffer size and work station capacity in serial–parallel production lines. *Artificial Life and Robotics*, *12*, 102-106.

Raidl, G. (2006). Hybrid metaheuristics. *Lecture Notes in Computer Science*, *4030*, 1-12.

Razali, N. M., & Geraghty, J. (2011). Genetic algorithm performance with different selection strategies in solving TSP. *In Proceedings of the World Congress on Engineering, 2,* London.

Reeves, C.R. (1996). *Modern heuristic techniques*. In: Rayward-Smith, V.J., Osman, I.H., Reeves, C.R, and Smith, G.D., eds. *Modern heuristic search methods* (1st Ed.) (1-25). England: John Wiley&Sons.

Ribeiro, M. A., Silveira, J. L., & Qassim, R. Y. (2007). Joint optimization of maintenance and buffer size in a manufacturing system. *European Journal of Operational Research*, *176*, 405-413.

Riise. A. (2002). *Comparing genetic algorithms and tabu search for multiobjective optimization. In the Proceedings of the IFORS conference*, July 8-12, Edinburgh.

Rudenko, O., & Schoenauer, M. (2004). *A steady performance stopping criterion for Pareto- based evolutionary algorithms. In: Proceedings of the 6th International Multi- Objective Programming and Goal Programming Conference*, Hammamet, Tunisia.

Sabuncuoglu, I., Erel E., & Kok, A. G. (2002). Analysis of assembly systems for interdeparture time variability and throughput. *IIE Transactions, 34,* 23-40.

Sabuncuoglu, I., Erel, E., & Gocgun, Y. (2006). Analysis of serial production lines: characterization study and a new heuristic procedure for optimal buffer allocation. *International Journal of Production Research*, *44* (13)*,* 2499-2523.

Sánchez, L., & Villar, J. R. (2008). Multiobjective evolutionary search of difference equations-based models for understanding chaotic systems. *Mathematical Modelling: Theory and Applications*, *24*, 181-201.

Schwefel, H. –P. (1981). *Numerical optimization of computer models*. Chichester: John&Wiley Sons.

Seong, D., Chang, Y.S., & Hong, Y. (1995). Heuristic algorithms for buffer allocation in a production line with unreliable machines. *International Journal of Production Research*, *33* (7)*,* 1989-2005.

Seong, D., Chang, Y.S., & Hong, Y. (2000). An algorithm for buffer allocation with linear resource constraints in a continuous-flow unreliable production line. *Asia-Pacific Journal of Operational Research*, *17*, 169-180.

Serafini, P. (1985). *Mathematics of multiobjective optimization*. 289, CISM Courses and Lectures. Berlin: Springer-Verlag.

Serafini, P. (1992). Simulated annealing for multiple objective optimization problems. In: *Proceedings of the Tenth International Conference on Multiple Criteria Decision Making.* Taipei, 19–24 July 1, 87–96.

Serafini, P. (1994). Simulated annealing for multiple objective optimization problems. In: Tzeng GH, Wang HF, Wen VP and Yu PL (eds). *Multiple Criteria Decision Making. Expand and Enrich the Domains of Thinking and Application*. Berlin: SpringerVerlag.

Sevastyanov, B. A. (1962). Influence of storage bin capacity on the average standstill time of a production line. *Theory of Probability and Its Applications*, *7*, 429-438.

Shi, L., & Men, S. (2003). Optimal buffer allocation in production lines. *IIE Transactions*, *35*, 1-10.

Shi, C., & Gershwin, S. B. (2009). An efficient buffer design algorithm for production line profit maximization. *International Journal of Production Economics*, *122*, 725-740.

Shi, C., & Gershwin, S. B. (2014). A segmentation approach for solving buffer allocation problems in large production systems, *International Journal of Production Research,* DOI: 10.1080/00207543.2014.991842.

Shieh, H. J., & Peralta, R. C., (2005). Optimal in situ bioremediation design by hybrid genetic algorithm simulated annealing. *Journal of Water Resources Planning and Management*, *131* (1)*,* 67-78.

Singh, A., & MacGregor Smith, J. (1997). Buffer allocation for an integer nonlinear network design problem. *Computers&Operations Research*, *24* (5)*,* 453–472.

Sivanandam, S. N. & Deepa, S. N. (2008). *Introduction to genetic algorithms*. New York: Springer Berlin Heidelberg.

Spinellis, D., Papadopoulos, C., & MacGregor Smith, J. (2000). Large production line optimization using simulated annealing. *International Journal of Production Research*, *38* (3)*,* 509-541.

Spinellis, D. D., & Papadopoulos, C. T. (2000a). A simulated annealing approach for buffer allocation in reliable production lines. *Annals of Operations Research*, *93*, 373-384.

Spinellis, D. D., & Papadopoulos, C. T. (2000b). Stochastic Algorithms for Buffer Allocation in Reliable Production Lines. *Mathematical Problems in Engineering*, *5*, 441-458.

Srinivas, C. Satyanarayana, B. Ramji, K., & Ravela, N. (2011). Determination of buffer size in single and multi row flexible manufacturing systems through simulation. *International Journal of Engineering Science & Technology*, *3* (5), 3889-3899.

Staley, D.R. (2007). *General design rules for the allocation of buffers in closed serial production lines*. M.Sc. Thesis, Oregon State University, USA.

Staley, D.R., & Kim, D.S. (2012). Experimental results for the allocation of buffers in closed serial production lines. *International Journal of Production Economics*, *137*, 284-291.

Su, C., & Xu, A. (2014). Buffer allocation for hybrid manufacturing/remanufacturing system considering quality grading. *International Journal of Production Research*, *52* (5), 1269-1284.

Suman, B., & Kumar, P. (2006). A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the Operational Research Society, 57*, 1143–1160.

Suppapitnarm, A., Seffen, K. A., Parks, G. T., & Clarkson, P. J. (2000). A simulated annealing algorithm for multiobjective optimization. *Engineering Optimization*, *33,* 59–85.

Swisher, J.R., Hyden, P.D., Jacobson, S.H., & Schruben, L.W. (2000). A survey of simulation optimization techniques and procedures, In Joines J.A., Barton R.R.,

Kang K., Fishwick, P.A., *Proceedings of the 2000 Winter Simulation Conference*. Orlando, USA, 119-128.

Sywerda, G. (1989). *Uniform crossover in genetic algorithms*. In: Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2–9.

Tasan, S. O. (2007). *Solving simple and mixed-model assembly line balancing problems using hybrid metaheuristic approaches.* Ph.D. Thesis, Dokuz Eylul University, Izmir.

Tekin, E., & Sabuncuoglu, I. (2004). Simulation optimization: A comprehensive review on theory and applications. *IIE Transactions*, *36* (11)*,* 1067-1081.

Tempelmeier, H. (2003). Practical considerations in the optimization of flow production systems. *International Journal of Production Research*, *41* (1)*,* 149-170.

Tiacci, L., Saetta, S., & Martini, A. (2006). Balancing mixed-model assembly lines with parallel workstations through a genetic algorithm approach. *International Journal of Industrial Engineering, 13*, 402–411.

Tiacci, L. (2015). Simultaneous balancing and buffer allocation decisions for the design of mixed-model assembly lines with parallel workstations and stochastic *task times. International Journal of Production Economics, 162,* 201-215.

Tuyttens, D., Teghem, J., & El-Sherbeny, N. (2003). *A particular multiobjective vehicle routing problem solved by simulated annealing.* In X. Gandibleux, M. Sevaux, K. Sörensen, and V. T'kindt, editors, Metaheuristics for multiobjective optimisation, volume 535 of Lecture Notes in Economics and Mathematical Systems, 133–152, Springer.

Ulungu, E. L., Teghaem, J., Fortemps, Ph., & Tuyttens, D. (1999). MOSA method: a tool for solving multiobjective combinatorial decision problems. *Journal of Multi-criteria Decision Analysis, 8,* 221–236.

Van Nieuwenhuyse, I., Vandaele, N., Rajaram, K., & Karmarkar, U. S. (2007). Buffer sizing in multi-product multi-reactor batch processes: Impact of allocation and campaign sizing policies. *European Journal of Operational Research, 179*, 424–443.

Van Woensel, T., Andriansyah R., Cruz, F. R. B., MacGregor Smith, J., & Kerbache, L. (2010). Buffer and server allocation in general multi-server queueing Networks. *International Transactions in Operational Research*, *17*, 257-286.

Vergara, H. A., & Kim, D. S. (2009). A new method for the placement of buffers in serial production lines. *International Journal of Production Research*, *47* (16), 4437-4456.

Vigeh, A. (2011). *Investigation of a simulated annealing cooling schedule used to optimize the estimation of the fiber diameter distribution in a peripheral nerve trunk*. M.Sc. Thesis, Faculty of California Polytechnic State University, San Luis Obispo.

Vladzievskii, A. P. (1950). The theory of internal stocks and their influence on the output of automatic lines. *Stanki i Instrumenty*, *21*, 4-7.

Vladzievskii, A. P. (1951). The theory of internal stocks and their influence on the output of automatic lines. *Stanki i Instrumenty*, *22,* 16-17.

Vouros, G.A., & Papadopoulos, H.T. (1998). Buffer allocation in unreliable production lines using a knowledge-based system, *Computers&Operations Research*, *25* (12), 1005-1067.

Wang, L., & Zheng, D. Z. (2001). An effective hybrid optimisation strategy for job-shop scheduling problems. *Computers&Operations Research*, *28,* 585–596.

Yamamoto, H., Qudeiri, J. A., & Marui, E. (2008). Definition of FTL with bypass lines and its simulator for buffer size decision. *International Journal of Production Economics*, *112,* 18-25.

Yamashita, H., & Altiok, T. (1998). Buffer capacity allocation for a desired throughput in production lines. *IIE Transactions*, *30,* 883-891.

Yildiz, G. (2003). *Simulation optimization of dual resource constrained CONWIP controlled lines using response surface methodology.* Ph.D. Thesis, Dokuz Eylul University, Izmir.

Yogeswaran, M., Ponnambalam, S. G., & Tiwari, M. K. (2009). An efficient hybrid evolutionary heuristic using genetic algorithm and simulated annealing algorithm to solve machine loading problem in FMS. *International Journal of Production Research*, *47* (19)*,* 5421-5448.

Yoo, M., & Gen, M. (2007). Scheduling algorithm for real-time tasks using multi objective hybrid genetic algorithm in heterogeneous multiprocessors system. *Computers&Operations Research*, *34,* 3084-3098.

Yuzukirmizi, M., & MacGregor Smith, J. (2008). Optimal buffer allocation in finite closed networks with multiple servers. *Computers&Operations Research*, *35* (8)*,* 2579-2598.

Zhang, C., Li, P., Rao, Y., & Li, S. (2005). A new hybrid GA/SA algorithm for the job shop scheduling problem. *Evolutionary Computation in Combinatorial Optimization Lecture Notes in Computer Science*, 246-259.

Zhao, F., & Zeng, X. (2006). Simulated annealing-genetic algorithm for transit network optimization. *Journal of Computing in Civil Engineering*, *20* (1)*,* 57-68.

Zitzler, E. (1999). E*volutionary algorithms for multi-objective optimization: Methods and applications*. Ph.D. Thesis, Swiss Federal Institute of Technology, Zurich.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation, 3* (4)*,* 257–271.

APPENDICES

APPENDIX A - DETAILED EXPERIMENTAL RESULTS FOR CHAPTER 4

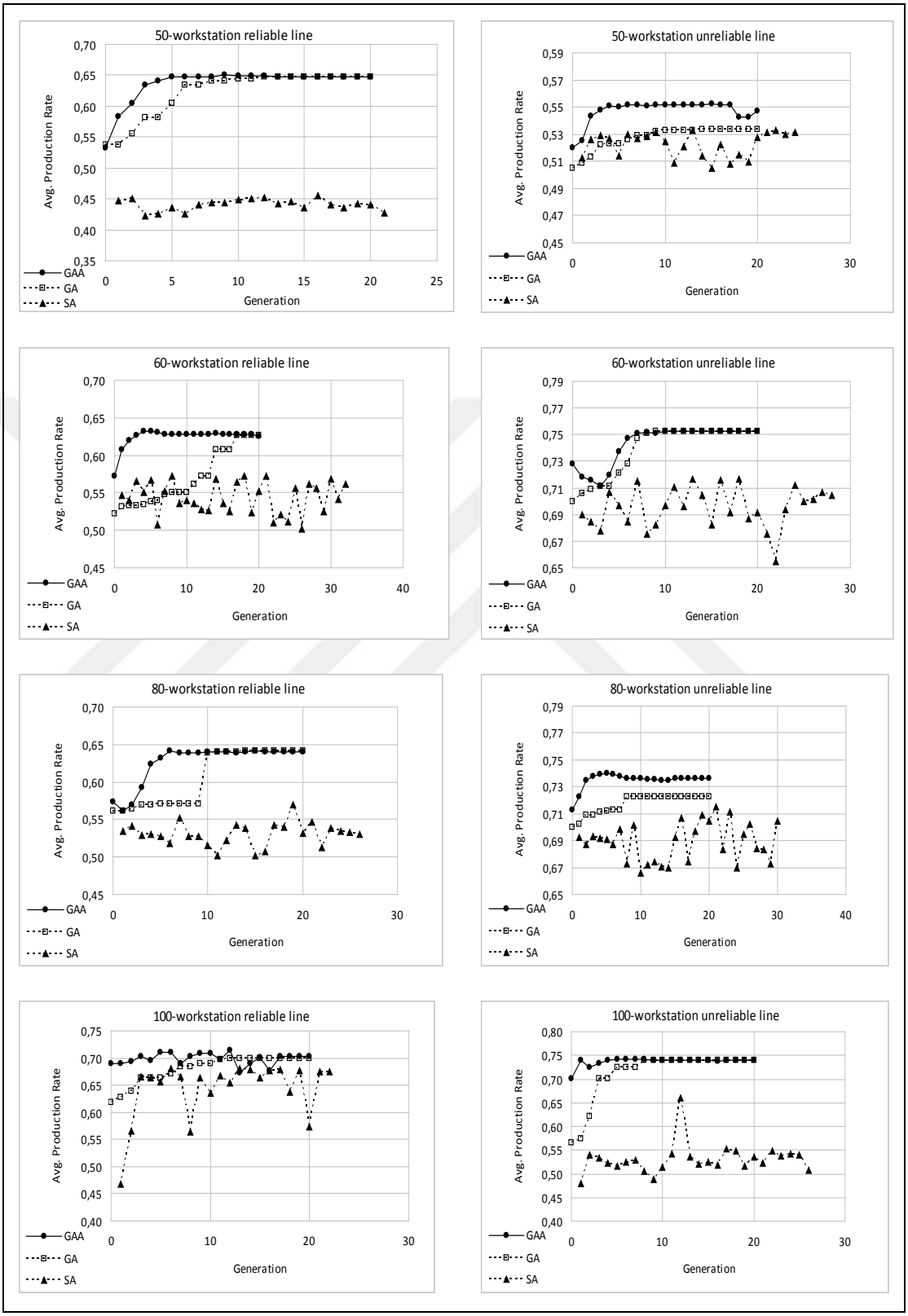
Appendix A1: Evaluating the performance of proposed GAA (*W*=5)

Generation Number	Buffer Size Configuration				Avg. Production Rate	Max. Production Rate
0	4	14	9	4	0.491106	0.495848
1	6	12	6	7	0.487452	0.492516
2	8	11	7	5	0.489809	0.494950
3	6	10	9	6	0.491002	0.496687
…	…	…	…	…	..…	…..
5	5	12	9	5	0.491576	0.496950
6	5	11	9	6	0.490808	0.496410
…	…	…	…	…	..…	…..
9	7	10	7	7	0.488892	0.494501
10	7	11	6	7	0.487499	0.492523
11	8	11	5	7	0.485181	0.490840
…	…	…	…	…	..…	…..
14	8	9	6	8	0.486194	0.491396
15	8	10	6	7	0.487355	0.492453
16	7	12	5	7	0.485284	0.490940
17	**7**	**10**	**10**	**4**	**0.494251**	**0.498779**
20	…	…	…	…	..…	…..

Appendix A2: Evaluating the performance of proposed GAA (*W*=10)

Generation Number	Buffer Size Configuration									Avg. Production Rate	Max. Production Rate
0	26	8	41	73	23	49	25	2	23	0.606720	0.612271
1	32	12	30	61	42	7	8	12	66	0.603109	0.607712
2	9	105	63	37	16	18	5	7	10	0.545766	0.551370
3	8	38	60	71	6	8	10	4	65	0.549762	0.554158
4	9	3	14	49	10	103	35	34	13	0.582124	0.585125
5	50	53	15	2	12	43	32	8	55	0.510954	0.515716
6	60	9	23	45	21	31	14	57	10	0.613764	0.617698
7	…	…	…	…	…	…	…	…	…	…..	…..
8	37	3	28	1	79	79	14	18	11	0.545149	0.550012
9	53	12	3	26	34	38	54	31	19	0.589733	0.593168
10	77	18	53	10	46	1	20	23	22	0.571894	0.575129
11	14	6	15	68	40	10	58	42	17	0.619975	0.623400
12	3	40	41	7	36	16	11	61	55	0.580928	0.585071
13	22	22	27	51	41	14	9	16	68	0.624623	0.627866
14	21	20	21	57	26	78	24	22	1	0.574111	0.581461
15	1	21	6	80	102	11	14	25	10	0.586663	0.590067
16	21	28	50	70	17	24	1	40	19	0.600114	0.603366
17	…	…	…	…	…	…	…	…	…	…..	…..
19	**7**	**16**	**48**	**61**	**24**	**41**	**20**	**34**	**19**	**0.630162**	**0.634182**
20	26	9	30	40	4	102	44	4	11	0.574080	0.578032

Appendix A3: Convergence rates of GAA, GA and SA for very large production lines



APPENDIX B - DETAILED EXPERIMENTAL RESULTS FOR CHAPTER 5

Appendix B1: Test Case 1 - Pareto Optimal solutions obtained by MOHEA (W=3)

Buffer Size Configuration	Total Buffer Size	Avg. Production Rate	Buffer Size Configuration
1	4	5	0.00987
6	3	9	0.01104
3	8	11	0.01157
9	3	12	0.01176
6	8	14	0.01217
10	6	16	0.01250
11	8	19	0.01288
10	10	20	0.01302
11	10	21	0.01311
12	10	22	0.01322
12	11	23	0.01330
12	12	24	0.01341
8	18	26	0.01359
11	16	27	0.01365
12	16	28	0.01371
14	15	29	0.01380
15	15	30	0.01388
15	16	31	0.01394
16	16	32	0.01400
20	13	33	0.01407
18	16	34	0.01413
20	15	35	0.01419
16	20	36	0.01424
20	17	37	0.01430
19	19	38	0.01437
19	20	39	0.01441
20	20	40	0.01447

Test Case 1 - Pareto Optimal solutions obtained by NSGA-II (W=3)

Buffer Size Configuration	Total Buffer Size	Avg. Production Rate	Buffer Size Configuration
2	1	3	0.008943
2	2	4	0.009483
3	2	5	0.009625
3	3	6	0.010245
6	1	7	0.011325
5	5	10	0.011347
6	5	11	0.011376
5	7	12	0.011562
7	6	13	0.011985
7	7	14	0.012118
8	7	15	0.012323
8	8	16	0.012445
8	9	17	0.012610
9	9	18	0.012747
10	9	19	0.012884
11	9	20	0.013018
10	11	21	0.013106
12	10	22	0.013220
12	11	23	0.013301
12	12	24	0.013410
13	12	25	0.013506
12	14	26	0.013443
13	14	27	0.013632
16	12	28	0.013694
16	13	29	0.013797
14	16	30	0.013867
15	16	31	0.013939
16	16	32	0.014002
17	16	33	0.014057
17	17	34	0.014131
18	17	35	0.014186
16	20	36	0.014243
20	17	37	0.014303
19	19	38	0.014365
20	19	39	0.014411
20	20	40	0.014471

Test Case 1 - Pareto Optimal solutions obtained by MOHEA (W=7)

Buffer Size Configuration						Total Buffer Size	Avg. Production Rate
10	3	7	2	6	2	30	0.00839
12	6	8	8	5	6	45	0.00992
5	9	12	4	12	4	46	0.01010
5	7	14	4	12	5	47	0.01013
11	6	13	6	9	7	52	0.01055
11	9	11	10	12	4	57	0.01085
9	13	15	5	14	4	60	0.01096
12	10	9	9	15	7	62	0.01112
11	3	18	13	8	11	64	0.01126
11	10	19	10	14	4	68	0.01139
17	9	15	9	11	8	69	0.01152
7	17	14	15	13	5	71	0.01155
9	8	20	12	11	12	72	0.01160
12	15	16	8	14	9	74	0.01184
10	13	18	15	10	10	76	0.01194
16	12	13	11	15	11	78	0.01197
10	15	20	14	9	13	81	0.01212
8	20	17	11	15	11	82	0.01218
9	20	20	11	14	10	84	0.01229
13	20	20	14	13	8	88	0.01235
8	18	18	20	13	15	92	0.01247
7	20	20	11	20	16	94	0.01249
19	13	20	20	8	15	95	0.01258
13	17	15	19	18	15	97	0.01267
20	20	14	13	20	13	100	0.01272
17	20	20	12	14	18	101	0.01275

Test Case 1 - Pareto Optimal solutions obtained by NSGA-II (W=7)

Buffer Size Configuration						Total Buffer Size	Avg. Production Rate
7	2	2	3	5	5	24	0.00755
8	3	7	3	3	4	28	0.00832
3	9	6	3	4	6	31	0.00866
2	4	11	2	11	4	34	0.00877
9	7	4	4	9	2	35	0.00887
3	6	13	3	11	2	38	0.00927
6	10	14	6	3	2	41	0.00931
4	6	9	4	16	3	42	0.00958
4	10	11	3	9	6	43	0.00980
10	7	9	4	9	5	44	0.00988
3	5	4	20	1	12	45	0.00989
7	4	6	6	14	9	46	0.00989
4	4	13	8	12	6	47	0.01000
5	7	12	10	11	3	48	0.01019
8	8	11	8	7	7	49	0.01041
8	12	6	6	13	6	51	0.01063
3	10	20	7	12	5	57	0.01085
14	5	13	8	11	8	59	0.01094
5	11	20	7	12	8	63	0.01115
11	5	11	14	14	10	65	0.01117
13	5	20	14	6	8	66	0.01122
12	11	9	6	20	11	69	0.01128
15	8	8	17	11	11	70	0.01138
14	16	15	8	14	4	71	0.01141
15	17	9	14	7	10	72	0.01148
19	5	20	12	7	10	73	0.01154
9	20	7	15	14	11	76	0.01168
19	6	16	16	8	12	77	0.01180
20	6	15	15	9	14	79	0.01185
15	11	16	20	9	9	80	0.01202
8	19	18	12	13	12	82	0.01218
14	13	20	8	20	8	83	0.01219
11	19	20	10	13	11	84	0.01229
9	18	20	20	10	10	87	0.01232
18	16	13	13	17	11	88	0.01238
12	15	20	11	12	20	90	0.01239
14	13	14	19	12	20	92	0.01239
20	13	20	11	9	20	93	0.01242
13	9	20	20	15	20	97	0.01264
20	20	16	9	20	16	101	0.01274
12	14	20	20	20	16	102	0.01276
20	13	20	15	16	20	104	0.01292
18	17	20	18	15	20	108	0.01308
16	20	20	20	18	20	114	0.01326
20	20	20	18	20	17	115	0.01332

Test Case 1 - Pareto Optimal solutions obtained by MOHEA (W=9)

Buffer Size Configuration								Total Buffer Size	Avg. Production Rate
2	7	8	9	4	20	4	3	57	0.00870
4	20	12	6	1	10	10	4	67	0.00899
5	20	12	2	4	14	11	6	74	0.00982
9	6	9	12	10	18	12	1	77	0.00993
4	15	6	2	11	14	20	6	78	0.00993
3	20	19	6	4	14	12	3	81	0.01006
6	8	6	15	14	6	10	17	82	0.01007
3	15	13	5	5	19	18	6	84	0.01013
10	7	8	8	20	11	15	6	85	0.01038
10	9	9	7	11	11	17	16	90	0.01051
12	11	10	15	14	6	18	6	92	0.01064
6	9	11	19	9	20	7	15	96	0.01088
7	20	7	16	13	11	20	4	98	0.01093
10	12	10	7	11	11	20	18	99	0.01099
16	18	8	12	20	12	8	12	106	0.01104
7	19	6	12	20	8	20	16	108	0.01106
10	13	11	20	20	8	13	15	110	0.01133
18	10	12	16	20	11	18	6	111	0.01136
13	6	17	13	9	19	20	19	116	0.01143
13	15	16	17	17	20	16	8	122	0.01178
7	20	10	20	20	18	15	16	126	0.01183
18	12	19	14	12	17	18	20	130	0.01205
20	12	20	13	14	17	20	20	136	0.01220
19	12	19	17	18	16	18	20	139	0.01225
20	14	20	15	17	17	18	20	141	0.01235

Test Case 1 - Pareto Optimal solutions obtained by NSGA-II (W=9)

Buffer Size Configuration								Total Buffer Size	Avg. Production Rate
5	1	1	5	5	6	10	4	37	0.00696
9	2	5	3	4	5	9	1	38	0.00741
6	3	6	2	8	4	8	4	41	0.00787
7	3	4	12	2	5	8	8	49	0.00822
8	2	8	7	4	4	16	2	51	0.00839
3	15	5	5	5	5	10	5	53	0.00859
5	13	7	4	10	10	2	7	58	0.00865
8	4	5	9	5	20	6	3	60	0.00889
3	7	16	5	4	14	4	9	62	0.00908
3	11	9	8	4	4	20	5	64	0.00919
12	6	7	12	5	6	11	6	65	0.00939
12	5	8	12	6	6	13	6	68	0.00959
2	12	13	3	8	16	9	9	72	0.00964
3	15	9	8	7	9	11	11	73	0.00978
12	5	10	13	6	6	14	9	75	0.00983
9	12	8	8	9	5	20	6	77	0.00988
13	5	16	7	5	9	15	10	80	0.00991
12	5	13	13	7	11	14	7	82	0.00994
8	13	11	5	14	10	13	16	90	0.01047
9	11	7	16	12	14	15	7	91	0.01051
14	10	10	14	14	10	13	7	92	0.01062
13	10	10	14	16	8	14	9	94	0.01080
9	15	15	11	6	20	13	7	96	0.01081
9	20	8	17	6	15	15	9	99	0.01085
7	12	10	17	7	19	13	16	101	0.01091
12	7	17	11	9	20	17	11	104	0.01104
13	8	17	11	8	19	18	12	106	0.01100
7	20	16	16	11	9	17	11	107	0.01103
10	9	18	12	9	19	16	15	108	0.01106
12	11	15	9	10	20	20	13	110	0.01126
9	13	20	18	7	20	18	12	117	0.01145
17	8	14	20	8	20	18	16	121	0.01160
10	18	15	14	20	20	15	10	122	0.01172
20	14	14	20	9	20	20	10	127	0.01181
17	13	20	14	11	14	20	20	129	0.01198
12	18	20	15	20	16	15	14	130	0.01203
15	13	20	15	20	15	20	15	133	0.01217
20	14	20	13	13	18	18	19	135	0.01220
17	18	17	14	13	20	20	20	139	0.01225
19	14	20	18	12	20	18	20	141	0.01234
16	12	19	20	15	20	20	20	142	0.01235
20	17	18	14	20	20	18	18	145	0.01249
14	19	20	14	19	20	20	20	146	0.01252

Appendix B2: Test Case 2 - Pareto Optimal solutions obtained by MOHEA (W=3)

Buffer Size Configuration		Total Buffer Size	Avg. Production Rate
3	3	6	0.50541
7	3	10	0.56138
4	7	11	0.56849
7	5	12	0.57757
6	8	14	0.59943
11	5	16	0.61213
12	5	17	0.61713
13	6	19	0.62675
11	9	20	0.63075
14	7	21	0.63748
13	9	22	0.64071
13	12	25	0.65225
16	10	26	0.65698
17	10	27	0.66016
18	11	29	0.66782
20	10	30	0.67091
14	17	31	0.67318
16	16	32	0.67643
20	14	34	0.68262
20	15	35	0.68428
20	16	36	0.68618
20	17	37	0.68824
19	19	38	0.68943
20	19	39	0.69189
20	20	40	0.69401

Test Case 2 - Pareto Optimal solutions obtained by NSGA-II (W=3)

Buffer Size Configuration		Total Buffer Size	Avg. Production Rate
2	2	4	0.467800
2	3	5	0.484497
3	3	6	0.505407
4	3	7	0.520715
4	4	8	0.534703
5	4	9	0.546834
5	5	10	0.557694
6	5	11	0.568146
7	5	12	0.577573
7	6	13	0.58568
8	6	14	0.59405
9	6	15	0.60114
9	7	16	0.607792
10	7	17	0.614321
9	9	18	0.620187
11	8	19	0.626308
11	9	20	0.630746
12	9	21	0.636802
13	9	22	0.640707
12	11	23	0.64501
14	10	24	0.648891
14	11	25	0.652191
15	11	26	0.656066
16	11	27	0.659688
16	12	28	0.663677
16	13	29	0.667295
18	12	30	0.670324
19	12	31	0.672415
17	15	32	0.675144
19	14	33	0.678231
19	15	34	0.681606
20	15	35	0.684281
20	16	36	0.686183
20	17	37	0.688236

Test Case 2 - Pareto Optimal solutions obtained by MOHEA (*W*=7)

Buffer Size Configuration						Total Buffer Size	Avg. Production Rate
9	2	5	3	4	5	28	0.36963
7	8	9	4	3	2	33	0.39333
4	2	7	11	6	4	34	0.40777
2	12	20	5	4	5	48	0.44356
3	3	9	15	3	16	49	0.44374
7	9	13	3	17	3	52	0.46065
6	4	19	8	16	4	57	0.47300
9	2	13	20	13	4	61	0.48034
7	6	11	15	16	7	62	0.48597
13	9	15	14	12	4	67	0.49652
17	5	13	13	12	8	68	0.49714
13	8	11	17	13	11	73	0.50545
7	14	20	16	12	11	80	0.51570
10	8	20	17	11	16	82	0.52664
15	12	8	17	20	12	84	0.52870
18	10	20	16	14	9	87	0.53388
6	20	19	15	16	13	89	0.53650
13	15	17	13	16	19	93	0.53959
20	3	20	20	19	14	96	0.54869
8	20	20	15	14	20	97	0.55055
15	19	20	20	12	20	106	0.56244

Test Case 2 - Pareto Optimal solutions obtained by NSGA-II (W=7)

Buffer Size Configuration						Total Buffer Size	Avg. Production Rate
2	4	4	6	4	3	23	0.35257
5	8	3	4	5	3	28	0.36197
9	3	4	4	7	3	30	0.36815
9	3	4	8	4	6	34	0.38580
1	11	5	5	13	2	37	0.39541
2	4	7	9	8	9	39	0.40505
3	7	9	5	14	3	41	0.41902
3	11	6	7	12	3	42	0.42389
4	11	6	9	13	2	45	0.43308
6	9	10	6	11	4	46	0.43730
5	7	10	5	17	4	48	0.43834
4	8	7	10	16	5	50	0.45098
5	4	16	11	12	3	51	0.45256
7	9	7	20	6	4	53	0.45470
3	11	9	11	14	7	55	0.46284
3	7	10	19	13	6	58	0.46763
4	9	9	20	7	11	60	0.47109
11	5	10	13	14	8	61	0.47979
12	6	11	13	16	5	63	0.48505
7	12	11	20	10	6	66	0.49289
13	7	12	13	16	6	67	0.49358
10	4	20	15	11	10	70	0.49559
13	10	8	17	12	11	71	0.49816
8	11	7	18	20	8	72	0.50130
13	6	12	20	16	7	74	0.50926
10	8	20	14	15	11	78	0.51494
5	12	19	16	16	11	79	0.51548
15	8	13	20	14	10	80	0.51550
7	19	9	19	20	9	83	0.52228
10	20	14	17	13	12	86	0.52373
17	7	20	20	13	10	87	0.52619
5	16	20	20	20	7	88	0.52981
11	20	11	20	20	10	92	0.53853
15	18	16	16	20	10	95	0.54004
15	20	14	16	20	12	97	0.54029
8	12	20	20	20	20	100	0.54584
9	16	16	20	20	20	101	0.54775

Test Case 2 - Pareto Optimal solutions obtained by MOHEA (W=9)

Buffer Size Configuration								Total Buffer Size	Avg. Production Rate
18	1	10	10	4	2	10	1	56	0.40662
7	7	3	8	5	17	3	7	57	0.40899
5	4	5	9	7	19	9	4	62	0.43789
13	5	5	5	20	2	13	6	69	0.44387
4	3	16	7	7	19	10	4	70	0.44596
14	8	13	11	6	6	15	2	75	0.45245
4	13	6	14	14	15	10	2	78	0.45791
4	9	15	17	16	7	11	2	81	0.46155
3	6	14	20	16	8	11	9	87	0.48386
20	7	13	18	11	9	7	6	91	0.49393
5	8	20	13	11	11	10	20	98	0.49486
11	5	20	20	16	7	14	8	101	0.50029
10	13	14	20	20	8	14	7	106	0.51073
8	9	17	20	20	13	11	11	109	0.51821
12	16	16	20	12	20	15	7	118	0.52137
7	19	13	16	18	20	18	9	120	0.52399
7	20	15	17	20	20	20	15	134	0.53704

Test Case 2 - Pareto Optimal solutions obtained by NSGA-II (*W*=9)

Buffer Size Configuration								Total Buffer Size	Avg. Production Rate
5	2	1	4	1	4	2	5	24	0.27736
9	2	5	3	4	5	9	1	38	0.33597
9	2	9	5	4	8	6	1	44	0.36326
2	5	13	3	13	5	4	2	47	0.37684
3	3	14	10	4	10	3	2	49	0.38542
11	7	5	9	6	3	10	2	53	0.38886
6	3	12	14	6	5	7	3	56	0.40078
5	5	6	7	10	15	9	2	59	0.41257
11	6	5	10	10	8	8	2	60	0.41898
5	10	4	20	7	4	9	6	65	0.42695
12	5	8	12	6	6	13	6	68	0.43187
2	13	15	11	8	6	11	4	70	0.43680
10	11	11	8	8	10	9	8	75	0.44658
9	8	5	16	20	10	3	5	76	0.44968
4	13	6	14	14	15	10	2	78	0.45791
4	9	15	17	16	7	11	2	81	0.46155
12	8	10	16	14	9	7	6	82	0.47205
7	9	11	14	10	15	13	8	87	0.48004
7	9	12	16	12	20	4	10	90	0.48394
7	8	16	17	16	9	13	5	91	0.49086
13	10	10	14	16	8	14	9	94	0.49121
6	6	20	13	17	14	7	13	96	0.49392
13	6	11	15	19	9	14	12	99	0.49717
11	14	11	14	18	7	14	11	100	0.49861
11	15	10	20	19	6	13	7	101	0.49861
3	20	12	19	20	12	12	5	103	0.50075
4	16	12	20	14	18	15	5	104	0.50378
10	16	9	17	20	12	6	15	105	0.50465
4	20	10	15	20	20	10	8	107	0.50612
13	11	16	13	20	8	20	7	108	0.50858
6	12	12	20	20	8	15	17	110	0.51173
19	11	10	14	20	15	15	11	115	0.51384
14	17	9	20	20	9	19	8	116	0.51874
10	9	20	18	16	17	14	13	117	0.52539
6	20	11	20	20	19	14	17	127	0.53005
6	9	20	20	20	20	20	20	135	0.53221
19	11	20	20	15	17	20	14	136	0.53929
20	12	20	20	18	17	20	20	147	0.54862

Appendix B3: Test Case 3 - Pareto Optimal solutions obtained by MOHEA (W=10)

Buffer Size Configuration									Total Buffer Size	Avg. Production Rate
0	0	2	0	1	1	2	0	5	11	0.51891
0	0	3	1	2	3	0	0	4	13	0.53408
1	2	5	1	3	0	1	1	0	14	0.54947
0	3	1	2	0	2	2	1	4	15	0.55481
1	1	2	2	1	2	1	1	5	16	0.57826
1	0	5	2	2	4	1	1	2	18	0.59949
1	1	1	5	4	2	3	1	1	19	0.61891
1	2	4	1	3	3	2	1	3	20	0.63004
1	1	4	4	4	2	2	1	5	24	0.64484
1	5	3	5	1	2	3	1	4	25	0.65172
1	2	6	2	2	5	4	1	3	26	0.66638
1	5	7	2	2	3	5	1	4	30	0.67823
1	3	6	5	2	4	5	1	4	31	0.68704
1	5	7	3	2	5	4	1	4	32	0.68839
1	3	8	4	4	3	5	1	5	34	0.69410
1	4	6	5	5	5	5	1	5	37	0.70453
1	4	10	5	4	4	5	1	5	39	0.70509

Test Case 3 - Pareto Optimal solutions obtained by NSGA-II (W=10)

Buffer Size Configuration									Total Buffer Size	Avg. Production Rate
0	0	0	1	0	1	1	0	4	7	0.48183
0	3	1	0	2	3	0	1	1	11	0.52195
1	3	2	0	2	2	0	1	1	12	0.52715
1	1	1	1	0	3	1	1	4	13	0.52814
1	2	3	1	2	2	1	1	1	14	0.53008
1	3	1	1	3	3	1	1	1	15	0.55377
1	1	3	1	2	5	1	1	1	16	0.57787
1	3	2	1	2	1	4	1	2	17	0.59893
1	1	2	2	2	5	2	1	2	18	0.60344
1	4	3	1	3	4	2	1	1	20	0.62300
1	3	2	1	2	5	3	1	3	21	0.63885
1	5	2	1	2	3	4	1	3	22	0.64332
1	2	4	2	2	5	5	1	1	23	0.64485
1	2	5	2	3	5	3	1	2	24	0.65044
1	2	3	5	3	3	5	1	3	26	0.65917
1	3	4	5	3	2	5	1	5	29	0.67799
1	5	4	5	3	5	2	1	5	31	0.68018
1	2	7	3	4	4	5	1	5	32	0.68396
1	5	5	3	5	5	5	1	3	33	0.69219
1	5	7	2	5	5	5	1	4	35	0.69874
1	5	7	3	5	4	5	1	5	36	0.70286
1	5	9	5	5	5	4	1	5	40	0.70643

Test Case 3 - Pareto Optimal solutions obtained by MOHEA (W=15)

Buffer Size Configuration														Total Buffer Size	Avg. Production Rate
0	0	2	2	1	2	0	1	0	3	0	1	4	2	18	0.52018
1	1	3	1	3	1	0	5	3	1	1	0	2	1	23	0.56020
1	0	2	1	8	1	1	1	2	4	1	1	5	2	30	0.59373
0	1	3	4	3	1	1	2	4	5	0	1	2	5	32	0.60444
1	1	5	3	0	10	1	0	5	3	1	1	2	1	34	0.60888
1	1	1	1	5	8	5	4	1	4	1	1	1	1	35	0.61462
1	1	2	2	9	3	3	1	5	1	1	1	3	3	36	0.62281
1	1	2	6	6	4	1	4	3	5	1	1	2	5	42	0.63264
1	1	4	10	7	6	2	3	5	3	1	1	5	5	54	0.64743

Test Case 3 - Pareto Optimal solutions obtained by NSGA-II (W=15)

Buffer Size Configuration														Total Buffer Size	Avg. Production Rate
1	1	3	1	0	0	0	1	2	4	0	1	1	1	16	0.49508
0	0	2	0	9	2	0	1	2	0	1	0	1	0	18	0.50820
0	1	2	1	2	2	0	1	2	4	0	1	4	0	20	0.55109
0	1	2	1	10	1	1	2	1	2	0	1	2	1	25	0.56096
1	0	3	6	2	2	0	4	4	1	1	0	2	1	27	0.56391
1	1	4	1	7	1	0	3	1	3	1	1	4	1	29	0.58273
1	1	3	2	0	2	1	5	1	5	1	1	2	5	30	0.58311
1	1	2	2	5	4	3	2	3	3	0	1	2	2	31	0.58878
1	1	2	5	1	1	3	1	5	1	1	1	4	5	32	0.60167
1	1	4	2	1	2	5	1	4	1	1	1	4	5	33	0.60733
1	1	5	1	3	5	4	1	4	2	1	1	2	3	34	0.61821
1	1	3	4	3	2	1	5	5	3	1	1	5	3	38	0.63210
1	1	3	5	4	6	5	1	3	5	1	1	5	4	45	0.64139
1	1	4	8	7	4	4	3	2	5	1	1	3	3	47	0.64247
1	1	5	10	6	1	4	4	2	5	1	1	5	3	49	0.64622

Appendix B4: Test Case 4 - Pareto Optimal solutions obtained by MOHEA (W=10)

Buffer Size Configuration									Total Buffer Size	Avg. Production Rate
1	1	2	1	2	2	0	1	1	11	0.15045
0	1	1	1	0	3	1	1	4	12	0.15539
0	1	1	1	3	3	1	1	3	14	0.16996
0	2	2	1	1	5	1	1	2	15	0.17556
0	1	1	5	2	5	2	0	0	16	0.17885
1	1	5	3	4	2	2	1	0	19	0.19264
1	2	3	5	3	1	3	1	1	20	0.19797
1	1	6	3	4	2	1	1	2	21	0.19895
1	1	6	3	5	2	1	1	2	22	0.20250
0	2	4	1	5	5	4	1	1	23	0.20626
1	1	5	3	5	2	5	1	2	25	0.21430
1	3	2	4	3	5	5	1	3	27	0.21929
1	5	4	5	5	5	1	1	1	28	0.22168
1	2	6	2	5	5	5	1	2	29	0.22730
1	3	7	4	4	3	5	1	2	30	0.22969
0	5	5	5	4	5	5	1	3	33	0.23509
1	5	7	2	5	4	5	1	4	34	0.23784
1	5	8	2	5	5	5	1	5	37	0.24471
1	4	8	5	5	5	5	1	5	39	0.25148

Test Case 4 - Pareto Optimal solutions obtained by NSGA-II (W=10)

Buffer Size Configuration									Total Buffer Size	Avg. Production Rate
0	0	0	1	1	1	0	0	5	8	0.13129
0	0	0	1	3	1	1	1	3	10	0.14482
0	0	2	0	1	1	2	0	5	11	0.14640
1	4	0	0	2	1	3	0	3	14	0.15902
1	1	2	1	1	4	1	1	3	15	0.17420
0	2	2	4	2	5	1	0	0	16	0.17861
1	3	6	1	1	2	1	1	1	17	0.18081
1	2	5	1	4	2	1	1	1	18	0.18901
1	1	5	3	4	2	2	1	0	19	0.19264
0	4	2	2	3	5	2	1	1	20	0.19681
1	4	0	3	4	5	2	1	1	21	0.19711
0	2	10	2	2	3	1	1	1	22	0.19765
1	3	6	2	2	2	3	1	3	23	0.20369
0	3	3	5	5	5	1	1	1	24	0.20939
0	3	4	5	2	2	5	1	3	25	0.21014
1	5	3	5	4	3	4	0	1	26	0.21265
1	2	3	4	5	1	5	1	5	27	0.21500
1	1	9	5	3	5	3	0	1	28	0.21682
1	3	6	5	2	5	4	1	2	29	0.22680
1	3	7	4	4	3	5	1	2	30	0.22969
1	0	9	5	5	5	5	1	2	33	0.23318
1	3	10	5	3	5	3	1	3	34	0.23780
1	5	6	2	5	5	5	1	5	35	0.23971
1	5	7	4	5	5	4	1	4	36	0.24419
1	5	7	5	5	5	4	1	4	37	0.24689
1	3	10	5	5	4	5	1	5	39	0.25058
1	5	10	5	5	5	5	1	4	41	0.25587

Test Case 4 - Pareto Optimal solutions obtained by MOHEA (*W*=15)

Buffer Size Configuration														Total Buffer Size	Avg. Production Rate
1	0	4	0	3	0	0	4	4	1	1	0	2	2	22	0.12531
1	0	4	2	3	1	1	3	4	1	1	0	1	1	23	0.13064
0	1	2	4	2	1	5	1	2	1	1	0	3	1	24	0.13469
1	0	4	1	3	0	0	5	5	2	1	0	2	2	26	0.13966
1	0	5	8	1	1	4	2	5	5	1	0	0	0	33	0.15009
0	1	5	8	6	1	1	4	5	2	1	1	0	0	35	0.15293
1	0	0	7	6	5	4	4	4	1	1	1	2	0	36	0.15899
1	1	1	1	7	9	4	1	2	5	1	1	3	1	38	0.16553
0	1	4	3	7	4	5	4	1	3	1	1	5	1	40	0.17089
0	1	3	5	1	6	4	4	4	4	0	1	5	5	43	0.17452
0	1	5	2	9	5	2	5	4	3	1	0	4	3	44	0.17919
0	1	3	3	10	6	5	5	2	3	1	1	5	1	46	0.17981
1	1	5	10	6	2	5	3	2	5	1	1	3	4	49	0.18311
1	1	5	10	5	4	4	5	3	5	1	1	2	3	50	0.18869
0	1	3	5	9	8	2	5	5	5	1	0	5	4	53	0.19012
1	1	2	10	8	10	3	4	5	3	1	1	5	0	54	0.19098
1	1	5	8	10	4	5	3	2	5	1	1	5	5	56	0.19258
1	0	3	10	7	10	3	5	4	5	1	1	5	3	58	0.19807

Test Case 4 - Pareto Optimal solutions obtained by NSGA-II (W=15)

Buffer Size Configuration														Total Buffer Size	Avg. Production Rate	Buffer Size Configuration
1	1	1	1	1	4	1	3	1	1	0	1	1	0	17	0.11870	0.00000
0	0	0	0	4	7	4	3	0	3	0	0	0	0	21	0.12272	0.08511
1	0	4	0	3	0	0	4	4	1	1	0	2	2	22	0.12531	0.10638
1	1	2	1	2	4	3	1	3	4	0	0	1	1	24	0.13459	0.14894
1	1	0	4	5	4	3	3	1	2	0	1	1	0	26	0.13838	0.19149
0	0	0	8	3	5	0	3	1	3	1	1	2	0	27	0.14034	0.21277
0	1	0	10	4	0	5	2	1	1	1	0	3	0	28	0.14341	0.23404
0	1	1	7	10	0	1	1	3	4	0	1	2	1	32	0.14610	0.31915
0	1	1	8	4	7	1	2	5	4	0	1	4	0	38	0.16380	0.44681
1	0	4	1	4	5	5	5	5	4	1	1	2	1	39	0.16891	0.46809
0	1	4	3	7	4	5	4	1	3	1	1	5	1	40	0.16957	0.48936
1	0	1	2	10	10	3	5	3	5	1	1	2	2	46	0.17837	0.61702
0	1	5	4	5	10	5	4	5	4	1	1	1	1	47	0.18213	0.63830
1	1	0	10	7	1	4	5	5	5	1	1	5	2	48	0.18294	0.65957
0	1	5	4	10	6	3	5	4	5	1	1	5	0	50	0.18828	0.70213
1	1	5	4	6	10	5	5	5	5	1	1	1	1	51	0.18967	0.72340
1	1	2	10	10	10	5	5	4	5	1	1	3	1	59	0.19942	0.89362
1	1	4	10	10	10	2	5	5	5	1	1	5	4	64	0.20713	1.00000