DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

CAPACITY IMPROVEMENT IN A REAL MANUFACTURING SYSTEM USING A HYBRID SIMULATION / GENETIC ALGORITHM APPROACH

by
Simge YELKENCİ KÖSE

July, 2010
İZMİR

CAPACITY IMPROVEMENT IN A REAL MANUFACTURING SYSTEM USING A HYBRID SIMULATION / GENETIC ALGORITHM APPROACH

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 Master of Science in Industrial Engineering, Industrial Engineering Program

by
Simge YELKENCİ KÖSE

July, 2010
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"CAPACITY IMPROVEMENT IN A REAL MANUFACTURING SYSTEM USING A HYBRID SIMULATION / GENETIC ALGORITHM APPROACH"** completed by **SİMGE YELKENCİ KÖSE** under supervision of **PROF. DR. SEMRA TUNALI** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Semra TUNALI

Supervisor

(Jury Member)

(Jury Member)

Prof.Dr. Mustafa SABUNCU

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

First and foremost, I would like to point out my gratitude to my advisor Prof. Dr. Semra TUNALI for her guidance, continuing support, encouragement and invaluable advice throughout the progress of this master thesis. Her advices always gave me the direction during the research.

I would like to thank the members of my dissertation committee, Assoc. Prof. Dr. Emine MISIRLI and Assist. Prof. Dr. Gonca TUNÇEL for their precious time to serve on my committee. I am profoundly thankful to all the professors and colleagues in the Department of Industrial Engineering, especially to Research Assist. Leyla DEMİR, Research Assist. Hacer Güner GÖREN, for giving valuable suggestions.

I would like to express my thanks to TÜBİTAK for their financial support through my M.Sc. study.

Last, but the most, I would like to emphasize my thankfullnes with ultimate respect and gratitude to my parents, Kamuran and Bülent Yelkenci, my sweetest sister, Fatoş Yelkenci and my great love, my dear husband, Yasin KÖSE, because of their love, confidence, encouragement and endless support in my whole life.

Simge YELKENCİ KÖSE

CAPACITY IMPROVEMENT IN A REAL MANUFACTURING SYSTEM USING A HYBRID SIMULATION / GENETIC ALGORITHM APPROACH

ABSTRACT

The primary aim of this M.Sc study is to suggest simulation and genetic algorithm based hybrid approach for capacity improvement in a real manufacturing system. In the first phase of the study, a detailed simulation model of the manufacturing system studied is developed using simulation language, Arena 10.0. Following, the verification and the validation of the model developed, potential bottleneck machines in this system are identified using this simulation model. In the following phase, the suggested hybrid method which combines genetic algorithm and simulation is employed to allocate buffers so that the capacity of the production line can be improved.

Keywords: Simulation, genetic algorithm, hybridization, capacity improvement

GERÇEK BİR ÜRETİM SİSTEMİNDE HİBRİD SİMÜLASYON VE GENETİK ALGORİTMA YAKLAŞIMI KULLANILARAK KAPASİTE İYİLEŞTİRME

ÖZ

Bu yüksek lisans çalışmasının esas amacı, gerçek bir üretim sisteminde kapasite iyileştirmek için simülasyon ve genetik algoritma tabanlı hibrid yaklaşımı ortaya koymaktır.

Çalışmanın ilk aşamasında, mevcut üretim sistemi Arena 10.0 ortamında modellenmiştir. İzleyen aşamada geliştirilen model değerlendirilip doğrulandıktan sonra mevcut sistemin simülasyon modeli ile sistemde darboğaz teşkil eden istasyonlar belirlenmiştir. Daha sonra, üretim hattında kapasite iyileştirmek için en uygun ara stok miktarlarını belirlemek amacıyla genetik algoritma ve simülasyon tabanlı hibrid yaklaşım geliştirilmiştir.

Anahtar sözcükler: Simülasyon, genetik algoritma, melezleme, kapasite iyileştirme

CONTENTS

Page

M.Sc THESIS EXAMINATION RESULT FORM ii
ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZ vi

CHAPTER ONE - INTRODUCTION 1

CHAPTER TWO – BACKGROUND INFORMATION 4

2.1 Bottleneck Identification Problem 4
2.2 Buffer Allocation for Capacity Improvement at Bottleneck Stations 7
2.3 Simulation Methodology 12
2.4 Genetic Algorithms 19
2.4.1 Terminology of Genetic Algotihms 21
2.4.2 Identifying Efficient GA Control Parameters 29
2.5 GA - Based Simulation Optimization 29

CHAPTER THREE – REVIEW OF CURRENT LITERATURE 31

3.1 Discussion of Current Relevant Literature 32
3.2 Motivation for This Study 43

CHAPTER FOUR - PROPOSED HYBRID APPROACH 45

4.1 Hybrid Simulation-GA Approach 45
4.2 Specifications of the GA-Based Approach 48
4.2.1 Chromosome Representation 49
4.2.2 Initial Population 50
4.2.3 Fitness Evaluation 51
4.2.4 Selection Scheme 51

4.2.5 Crossover (Recombination) 53
4.2.6 Mutation 54
4.2.7 Replacement Scheme 55
4.2.8 GA Search Termination 56

CHAPTER FIVE – AN INDUSTRIAL CASE STUDY 57

5.1 Application Environment and Problem Definition 57
5.2 Phase I: Bottleneck Identification Using Simulation Modelling 62
5.2.1 Model Conceptualization 63
5.2.2 Simulation Model Development 67
5.2.3 Verification and Validation.. 69
5.2.4 Bottleneck Identification.. 71
5.3 Phase II: Capacity Improvement by Employing the Proposed Hybrid Approach 72
5.3.1 Identifying Efficient GA Control Parameters 74
5.3.2 Structure of the Proposed Hybrid Approach.. 79
5.3.3 Implementation of the Hybrid Approach 82
5.3.3.1 Hybrid GA-Based Simulation Approach Using Random Initialization Scheme 82
5.3.3.2 Hybrid GA-Based Simulation Approach Using Heuristic Initialization Scheme 87
5.3.3.3 Hybrid GA-Based Simulation Approach Using Random Initialization Scheme for Bottleneck Machines 92
5.3.4 Comparison of Alternative Results 97

CHAPTER SIX - CONCLUSION 99

REFERENCES... 101

CHAPTER ONE
INTRODUCTION

In a world of increasing competition, it is of a great importance to improve the capacity of limited production resources. There are various factors affecting the capacity of production systems such as policies for demand management, layout, process technology, bottleneck machines and etc. Within the scope of capacity improvement studies bottleneck identification and optimal buffer allocation have been the subject of many studies and these subjects are still very active research areas.

As illustrated in Figure 1.1, bottleneck machines limit the output of a production system. In other words, the capacity of a production system is determined by bottleneck machines. Any time lost on bottleneck machines affects the capacity of the whole system.

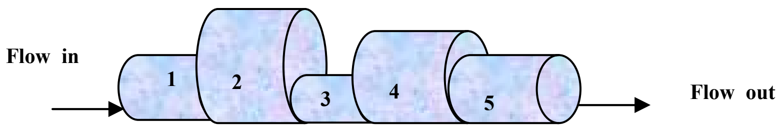


Figure 1.1 A representation of bottlenecks in the product flow

As seen in Figure 1.2, placing buffers in front of bottleneck machines will help to reduce the starving time and blocking time, and increase the utilization of bottleneck machines and in turn, the capacity of the production line will improve. Solving the buffer allocation problem efficiently has a great effect on the performance of a system. However, it should be noted that the improvement in system performance through buffer allocation is achieved at the expense of increasing in-process inventory levels. If the buffers are too large then the capital cost incurred may outweigh the benefit of the increased productivity. If the buffers are too small, the machines will be underutilized or demand will not be met.

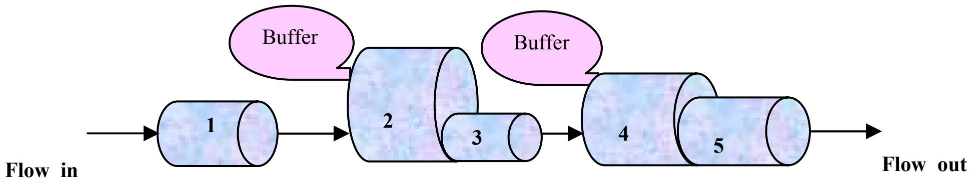


Figure 1.2 A representation of buffers in front of bottlenecks in the product flow

Due to these reasons, bottleneck detection and buffer allocation problems are still attracting many researchers. The most commonly used approaches to solve these problems are analytical methods, simulation modeling, and metaheuristics. Buffer allocation problem can be formulated analytically, but analytical results can be found for very simple cases under very restrictive assumptions. Hence, it becomes necessary to develop alternate techniques which are computationally tractable and able to develop near optimal solutions.

Simulation modeling approach provides many advantages in modeling dynamic and stochastic systems in detail. As computer technology and simulation software have advanced in recent years, the cost of computer time has become much cheaper, and simulation software has become more widely available and in turn the use of simulation in modeling complex systems has become quite widespread. However, it should be noted that simulation modeling is not an optimization method and finding optimal system configuration with simulation is a time consuming process. Therefore, it is essential to employ an optimization method in conjunction with simulation model and this method is called as simulation optimization.

Due to increasing success of metaheuristics such as tabu search, genetic algorithms, simulated annealing and etc. in solving many manufacturing optimization problems, the trend in recent years is to solve buffer allocation problem using these metaheuristics. Among these metaheuristic methods, genetic algorithms (GA) have many advantages such as performing multiple directional searches by using a set of candidate solutions, requiring no domain knowledge and using stochastic transition rules to guide the search.

Considering the advantages of both simulation modeling and genetic algorithms, this M.Sc. study focuses on production lines and proposes a hybrid GA-based simulation approach to allocate limited buffer capacities to the stations so that some capacity improvements can be achieved in the line. As a result, the approach taken in this study combines the key advantages of both simulation modeling and genetic algorithms. Specifically, the proposed approach employs a two-phase simulation-genetic algoritm procedure. In the first phase, a detailed bottleneck analysis has been carried out to identify what limits the capacity of the system by developing a discrete-event simulation model of the system. Following, the proposed hybrid approach is employed to allocate buffers to the machines so as to improve the performance of the system. In this hybrid approach, the simulation model of the production line is used to evaluate the fitness function of the genetic algorithm.

The rest of the study is organized as follows. In Chapter 2, detailed background information about bottleneck identification problem, buffer allocation for capacity improvement at bottleneck stations, simulation methodology, genetic algorithms and GA–based simulation optimization are given. In order to highlight the place of this study in the current literature, the current relevant studies are extensively discussed in Chapter 3. The proposed hybrid GA-based simulation approach is presented in Chapter 4. Finally, concluding remarks and the future research directions are given in the last section.

CHAPTER TWO
BACKGROUND INFORMATION

This section presents detailed background information about bottleneck identification problem, buffer allocation for capacity improvement of bottleneck stations, simulation methodology, genetic algorithms and GA-based simulation optimization.

2.1 Bottleneck Identification Problem

Bottleneck identification problem has gained attention after the book "The Goal" was come onto the market. The author presented in his book a new vision on how to obtain a better process improvement by identifying bottlenecks to improve productivity. It is quite well known that the root cause of many performance problems is linked to the bottlenecks in the system (Lima et al., 2008).

According to Goldratt (1992), the flow of goods of an entire system is limited by the capacity of different machines. Depending on the nature of the system, some machines affect the overall system performance more than other machines. These machines are commonly called as bottlenecks.

In general, the bottleneck types can be classified into three categories:

• Simple Bottleneck (Grosfeld-Nir, 1995),
• Multiple Bottlenecks (Aneja and Punnen, 1999),
• Shifting Bottlenecks (Roser et al., 2002).

In the case of simple bottleneck, there is only one bottleneck machine during the entire period considered as seen in Figure 2.1. For the multiple bottlenecks situation, the system consists of multiple stable bottlenecks through the entire period (see Figure 2.1). In shifting bottlenecks, as seen in Figure 2.1, the location of bottleneck machines in the system may change at any time.

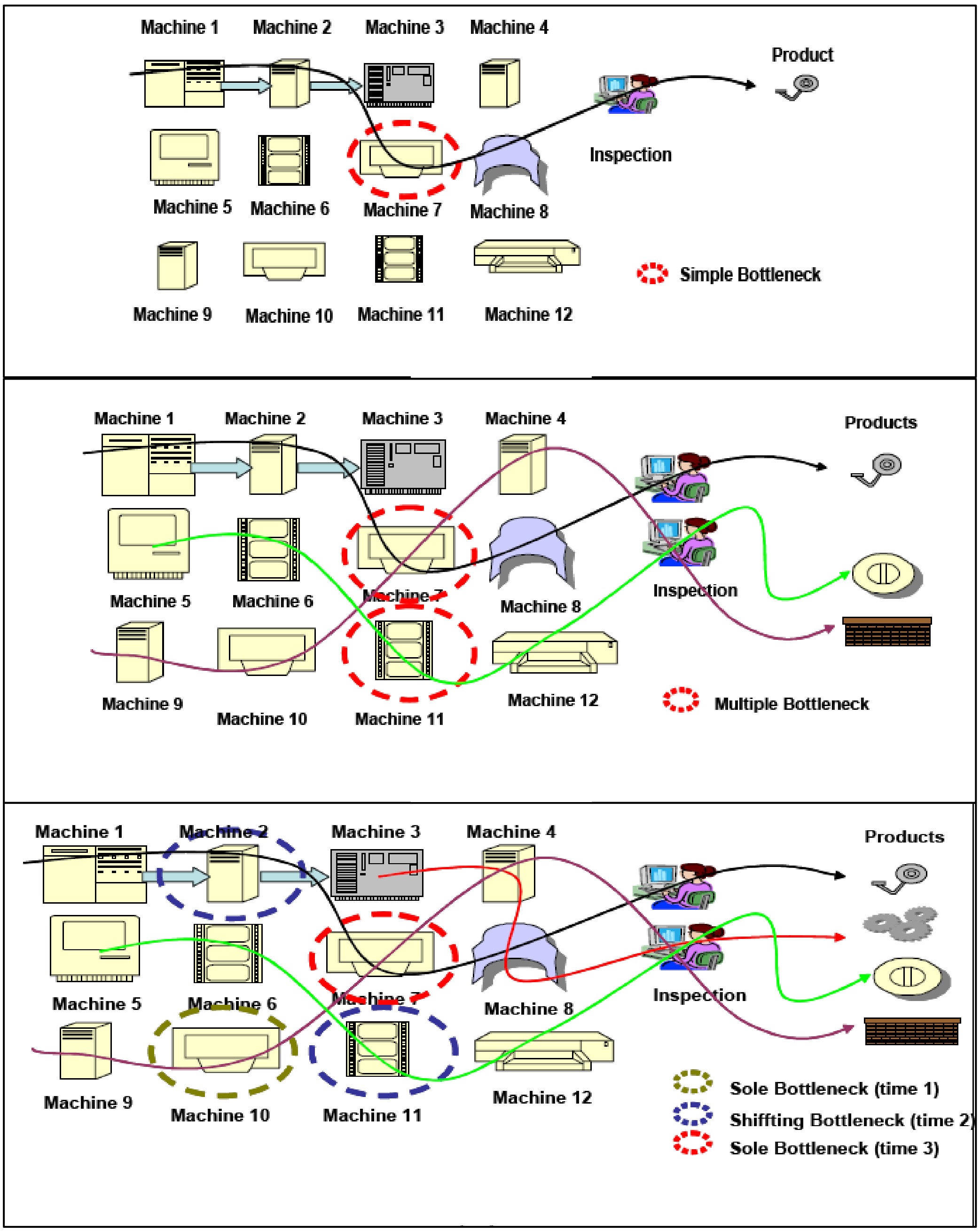


Figure 2.1 The basic configurations of bottleneck types (Lima et al., 2008)

In order to improve the performance of the system, the throughput of the bottlenecks has to be improved. In this case, it is necessary to first detect the bottlenecks in the system. During the survey of relevant literature, a number of methods have been noted to detect the bottlenecks. Some of these methods are based on utilization, using for example a matrix based approach to determine the overall constraint (Luthi, 1998; Luthi and Haring, 1997) or the ratio of the cycle time divided by the processing time (Delp et al. 2003). Other methods use a system theoretic approach to determine the sensitivity of the machine throughput to the system throughput (Chiang, Kuo, and Meerkov, 1998; Chiang, Kuo, and Meerkov, 2000; Chiang, Kuo, and Meerkov, 2002; Kuo, Lim, and Meerkov, 1996; Li and Meerkov, 2000). Bukchin (1998) compared a number of theoretical estimations of the system performance, and found that an estimator based on the machine bottlenecks works best (Roser et al., 2003). Moreover, Kasemset and Kachitvichyanukul presented a simulation-based procedure considering the machine/process utilization, the process utilization and the product bottleneck rate to identify potential bottleneck candidates (Kasemset and Kachitvichyanukul, 2007).

The main characteristics of the bottleneck detection methods are summarized in Table 2.1.

Table 2.1 Bottleneck detection methods (Kasemset and Kachitvichyanukul, (2007); Lima et al., (2008))

Method	**Characteristic**	**Measurement**
1. Utilization Rate (%)	The percentage of time that the production station is in use. The machine with higher utilization would be the bottleneck.	Percentage (%)
2. Bottleneck Rate (Rb)	The bottleneck rate is the rate of parts/jobs per unit time. The machine having low value of output rate would be the bottleneck.	-
3. Queue size in front of machine	The number of products waiting the machine to be available. The machine that has the longest queue would be the bottleneck.	Quantity of products

Continuation of Table 2.1

4. Waiting time in front of the machine	It measures how long a product waits in front of a machine to be processed.	Time
5. Active period time	Two states (i.e., active and non active) are considered. It measures the total time that a machine is in active state. The machine with the highest active period time would be the bottleneck.	Time unit or percentage of time
6. Shifting bottleneck method	Total time (or percentage) that a machine is in the active state without any interruption.	Time unit or percentage of time

In manufacturing systems, there often exists a bottleneck machine whose capacity is equal to or less than the market demand. Any idle or waste time at the bottleneck machine directly impacts the output of the entire plant because it results in a loss of throughput. After finding the bottleneck machine of a system, it is then possible to improve the performance of the bottleneck in order to improve the overall performance of the system. As a result, detecting and managing bottlenecks can have a major impact on performance of a manufacturing system.

2.2 Buffer Allocation for Capacity Improvement at Bottleneck Stations

The buffer allocation problem (BAP) is an NP-hard combinatorial optimization problem. Many manufacturing systems such as transfer lines, flexible manufacturing systems or robotic assembly lines are vulnerable to bottleneck problems. BAP is mainly concerned with how to distribute a certain amount of buffers among the intermediate storage spaces so that the production capacity of the system can be improved.

By providing additional parts, buffers reduce the starving time and blocking time; hence, the utilization of bottleneck machines is increased. Figure 2.2 illustrates a configuration of a manufacturing system with input buffer & output buffer.

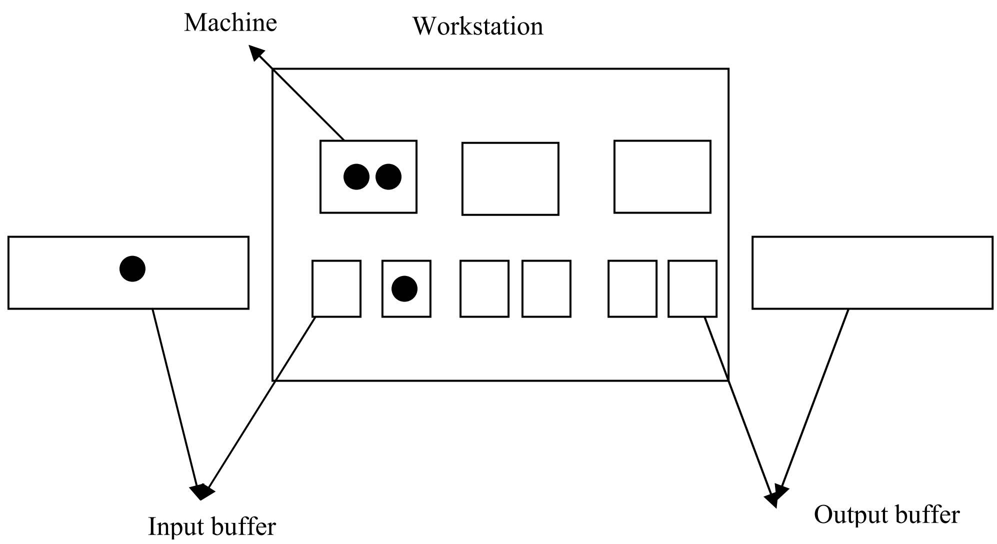


Figure 2.2 Manufacturing system with input buffer & output buffer

However, the improvement in system performance is achieved at the expense of increasing in-process inventory levels. If the buffers are too large then the capital cost incurred may outweigh the benefit of the increased productivity. If the buffers are too small, the machines will be underutilized or demand will not be met. Since buffers have a great effect on the performance of the system, the buffer allocation problem is still a major optimization problem faced by manufacturing designers.

The buffer allocation problem consists of distributing a certain amount of buffer space among the intermediate buffers of a production line. Figure 2.3 shows a serial production line consisting of M machines and $M-1$ buffers.

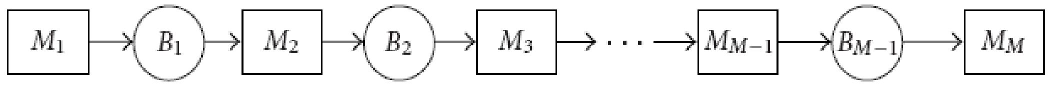


Figure 2.3 A serial production line with buffers.

The buffer allocation problem was considered in the literature with respect to different optimality criteria. The commonly used among them are summarized as follows (Dolgui et al., 2002):

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

Considering these optimality criteria, the buffer allocation problem can be represented mathematically in three forms. As seen below, while one employs the maximization of the throughput rate of the line as an objective function, the second one focuses on the minimization of the total buffer space. In the case of third optimality criterion, the minimization of the inventory cost is achieved.

<u>*Formulation 1*</u>: This formulation expresses the maximization of the throughput rate, given a certain fixed amount of buffers, as follows:

Find $B = (B_1, B_2, B_3, \ldots, B_{M-1})$ so as to

$$\max P(B) \tag{1}$$

subject to

$$\sum_{i=1}^{M-1} Bi = \mathrm{K} \tag{2}$$

Bi nonnegative integers (i = 1,2,…, *M*-1) (3)

where *K* is a fixed nonnegative integer denoting the total buffer space available in the system which has to be allocated among the *M*-1 buffer locations so as to maximize throughput of the production line. In this formulation *B* represents a buffer vector, and *P(B)* represents the throughput rate of the production line.

<u>*Formulation 2*</u>: This formulation expresses achieving the desired throughput rate with the minimum total buffer space, as follows:

Find B = (B_1, B_2, B_3,…, B_{M-1}) so as to

$$\min \sum_{i=1}^{M-1} Bi \tag{4}$$

subject to

$$P(B) \geq P* \tag{5}$$

Bi nonnegative integers (i = 1,2,…, *M*-1) (6)

where *M* is the number of machines in the line, *B* is a buffer vector, *P(B)* is the throughput rate of the production line and P^* is the desired throughput rate.

<u>*Formulation 3*</u>: This formulation expresses the minimization of the average steady-state inventory cost subject to the total buffer constraint.

Find B = (B_1, B_2, B_3,…, B_{M-1}) so as to

$$\min \ Q(B) = \sum_{i=1}^{M-1} c_i B_i \tag{7}$$

subject to

$$\sum_{i=1}^{M-1} Bi \leq \mathrm{K} \tag{8}$$

$$Bi \text{ nonnegative integers } (\mathrm{i} = 1,2,\ldots, M\text{-}1) \tag{9}$$

In this formulation B represents a buffer vector, c_i represents the cost for each buffer location and K is a fixed nonnegative integer denoting the total buffer space available in the system which has to be allocated among the M-1 buffer locations so as to minimize the averae steady-state inventory cost of the production system.

Solution approaches to solve buffer allocation problem involve applying a generative method and an evaluative method in an iterative manner. In other words generative methods and evaluative methods are combined in a closed loop configuration. In this configuration an evaluative method is used to obtain the value of the objective function for a set of inputs. To search for an optimal solution, the value of the objective function is then communicated to the generative model.

Simulation and analytical methods such as traditional Markov state models, decomposition methods, aggregation methods are examples of evaluative approaches. In comparison to simulation, analytical methods are faster but they are usually constructed under some restrictive assumptions which may not be computationally effective in dealing with real world buffer allocation problems. If the objective is to realistically model a large and complex system, as in the case of our study, simulation provides many advantages in comparison to analytical methods. However, simulation is generally an expensive tool in terms of time and monetary resources.

In buffer allocation problem, the simplest method to obtain the optimal buffer sizes is complete enumeration. But the total number of feasible solutions increases exponentially when the total buffer size, K, and the number of machines in the system, M, are large. The number of possible buffer configurations can be calculated as follows:

$$C_{K+M-2}^{M-2} = \frac{(K+1)(K+2)...(K+M-2)}{(M-2)!} \tag{10}$$

For instance, if the production system involves only ten machines and the number of total buffers to allocate is 50, then the total number of feasible buffer allocations becomes 1.916.797.311 indicating the computational difficulty to search through the whole solution space by complete enumeration even for small sized problems. Thus various search methods and metaheuristics have been tried to effectively deal with the combinatorial nature of the buffer allocation problems. The Hooke-Jeeves method, knowledge based methods, dynamic programming based methods and various heuristic procedures are as examples of the search methods category. In recent years, metaheuristic approaches are also widely used to solve buffer allocation problem such as Genetic Algorithms (GAs), Tabu Search, Simulated Annealing, and Ant Colony Optimization.

In particular it is proven that Genetic algorithms is an effective tool for various combinatorial optimization problems. The power and simplicity of GA make it popular for even large scale optimization problems (Boyabatlı&Sabuncuoğlu, 2004). However, as problems get larger and more complex as in real life, pure GAs may lack the capability of exploring the solution space effectively. Hence, over the last years, a number of studies have been reported combining the various methods and metaheuristics named as hybridization. In this M.Sc. study, a hybrid approach combining simulation and GAs is proposed to solve the buffer allocation problem in a real-world production system. The further details about these two methods are explained in the following sections.

2.3 Simulation Methodology

Simulation is one of the most commonly used tool for the design and operation of complex processes or systems (Kozan, 2003). Banks et al. (2001) define simulation 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, rather simulation is a descriptive tool and it only provides estimates of some performance measures. Simulation technique simply provides us with a mechanism to understand and predict the behaviour of a system. Once developed and validated, a model can be used to investigate a wide variety of “what-if” questions about the real world system.

As it is stated above, simulation can be used to investigate systems in the design stage, before such systems are built. Thus, simulation modelling can be used both as an analysis tool to predict the effects of changes to existing systems, and also as a design tool to predict the performance of new systems under varying sets of circumstances (Banks et al., 2001). 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.

It should be noted that besides developing simulation models with animation features using special purpose simulation languages such as ARENA and PROMODEL in a microcomputer environment, some rudimentary simulations can be performed in hand-held pocket computers using spreadsheet software. So all these developments summarize the widespread use of simulation in recent years.

As given in Figure 2.4., a simulation study involves many steps (Banks et al.,2001).

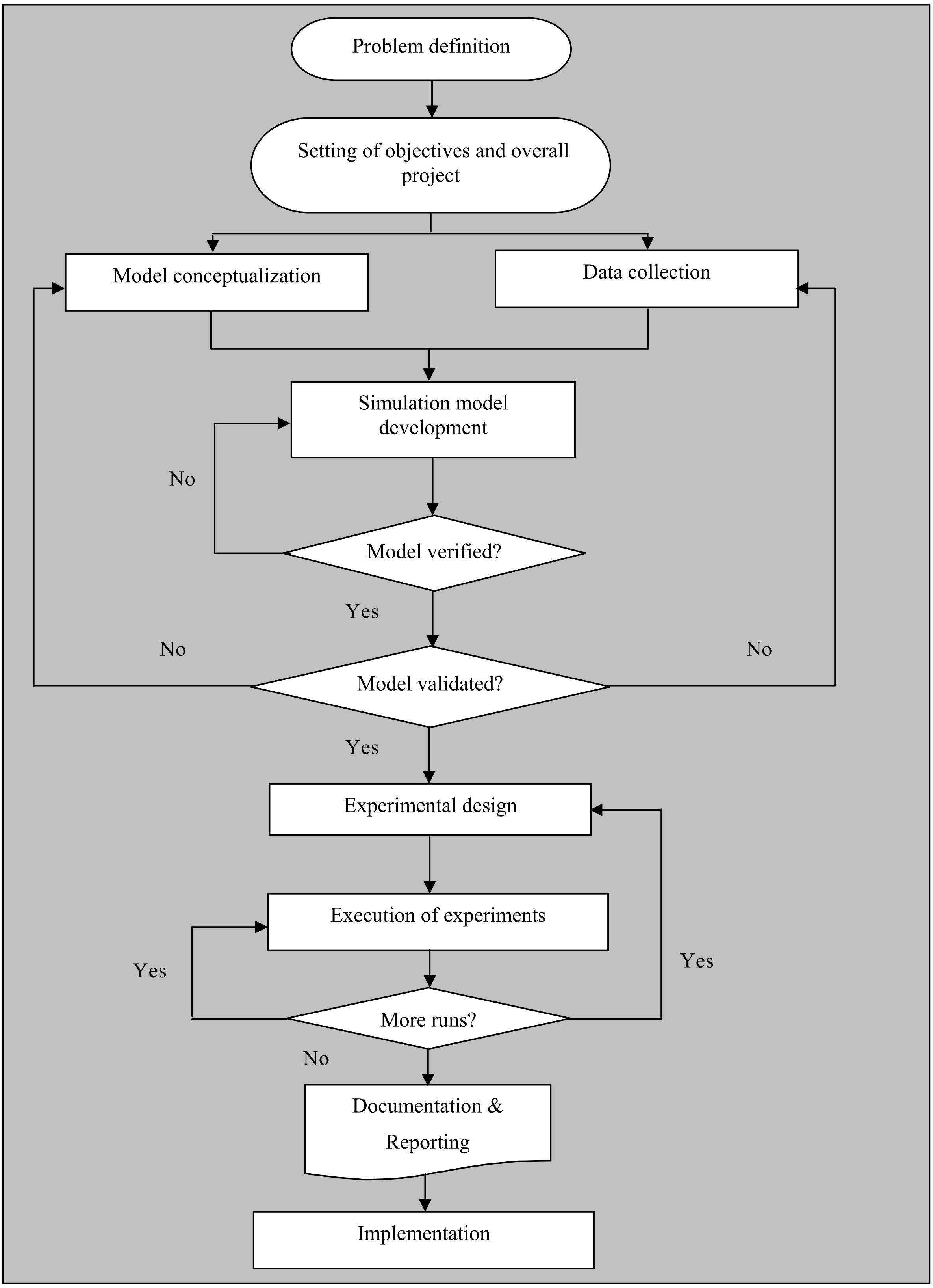


Figure 2.4 Steps of a simulation study

The short explanations for these steps are given as follows:

- *Problem Definition:* Simulation studies are initiated because a problem is faced by a decision maker or group of decision makers and a solution is needed (Ay, 2009). Once the problem at system is defined and decision makers all agree that is a problem, model builder must ensure that the problem being described is clearly understood. During the development process by the model builder, the problem can be reformulated as the study progresses in accordance with desicion makers' demand.

- *Setting of Objectives and Overall Project Plan:* The objectives indicate the questions to be answered by simulation (Banks et al., 2001). 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 by hybridizing simulation with other methods,
  - ✓ Sensitivity analysis, bottleneck analysis.

Following the formulation of the problem and stating the objectives explicitly as given above, it is made sure that simulation is the appropriate methodology and the overall project is planned in terms of cost, the number of people to be involved in this project and time required to accomplish each phase of the work.

- *Model Conceptualization:* The construction of a model of a system is as much art as science. The art of modelling is enhanced by an ability to abstract the essential features of a problem, to select and modify basic assumptions that characterize the system, and then to enrich and elaborate the model until a useful approximation results. Thus it is best to start with a simple model and build toward greater complexity (Banks et al., 2001). Graphical representations (block diagrams, flow charts, etc.), and pseudo-codes are used to conceptualize the model. Figure 2.5 depicts the model conceptualization scheme as follows.

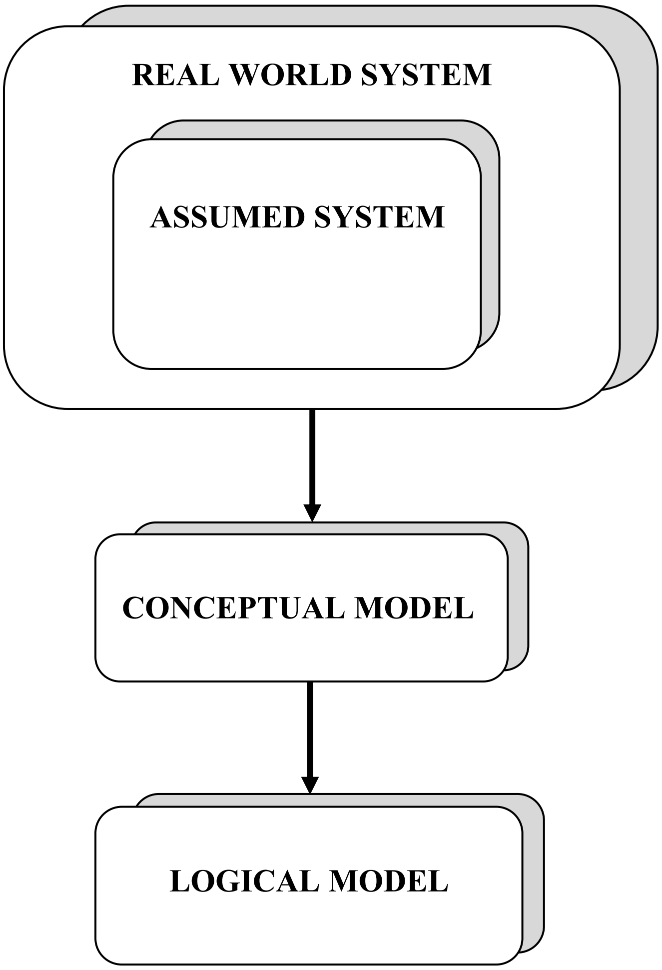


Figure 2.5 Model conceptualization

- *Data Collection:* There is a constant interplay between the construction of the model and the collection of the needed data (Shannon, 1975). The necessary input data can be collected through different information sources. In general, historical data is used for simulation analyzes. Since data collection for simulation study takes such a large portion of the total time reqired to perform a simulation, it is essential to begin for data collection from the early stages of model building.

- *Simulation Model Development:* Since most real world systems result in models that require a great deal of information storage and computation, the model is translated into a computer-recognizable format (Banks et al., 2001). The model builder can achieve the model translation in two ways: programming the model in a general purpose language or in a special purpose simulation software as seen Figure 2.6.

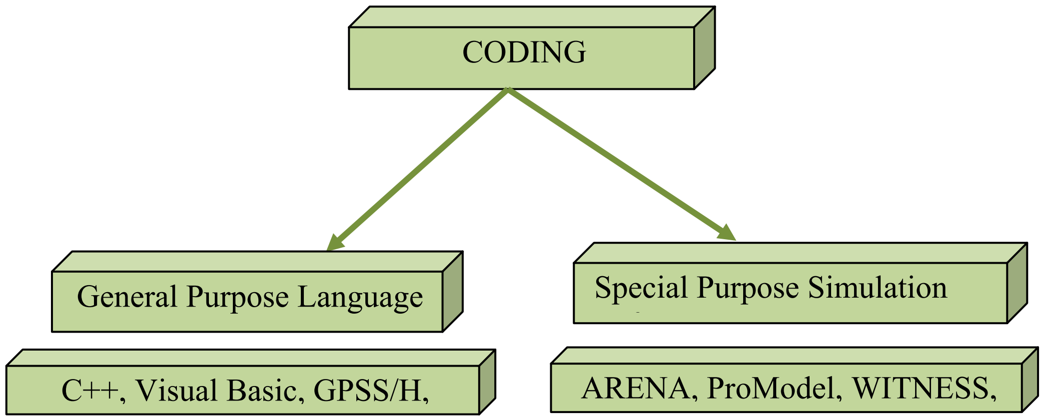


Figure 2.6 Coding schemes

- *Model Verification:* Model verification is the process of determining if the operational logic is correct. In other words, to assure that the conceptual model is reflected accurately in the computerized representation is the main purpose of model verfication. The model builder must observe if the simulation model translated performs accurately during this phase. In complex models, it is difficult and sometimes impossible to debug the simulation software successfully. Many common-sense suggestions which are explained in the book of Banks et al. (2001) are used for the verification process as follows:

    - ✓ asking someone else to check the model,
    - ✓ making a flow diagram that includes each logically possible action a system can take when an event occurs,

- ✓ examining the model output for reasonableness under a variety of input parameter settings,
- ✓ printing the input parameters at the end of the simulation to check that they have not been changed inadvertently,
- ✓ Using graphical representations to simplify the task of model understanding.

Also, it should be noted that use of an interactive run controller, or debugger, is highly encouraged as an aid to the verification process. Once logical structure of the model is correctly represented in the computer, verification has been completed.

- *Model Validation:* Model validation is the determination that the conceptual model is an accurate representation of the real system (Bank 2000). There are many methods to perform validation process. Most common way to validate the model is to compare its output to that of the real system using a wide variety of techniques. Some of them can be summarized as follows:

  - ✓ High face validity: Insuring by consulting knowledgeable people and sensitivity analysis,
  - ✓ Statistical tests: Conducting these tests on assumed distrtibutional forms (i.e. hypothesis tests, confidence interval tests, etc),
  - ✓ Turing test: Utilizing persons' knowledge about the system.

- *Experimental Design:* In this step, decisions need to be made for simulation model in terms of the length of simulation process, the number of replications to be made each run and the length of the initialization period.

- *Execution of Experiments:* Once experimental designs are carried out, the simulation runs and the subsequent analysis are done to estimate performance measures for the model that is being simulated.

- *Documentation&Reporting:* Documentation is an important step for a simulation study. This process eases the modification of the simulation in the future and also allows for others to understand how the program operates. Musselman (1998), discusses progress reports that provide the important, written history of a simulation project. These reports give a chronology of work done and decisions made. This can prove to be of great value in keeping the project on course (Banks et al., 2001).

- *Implementation:* As a last step, all the decisions made as a result of simulation study are implemented in the real system and the performance is observed for follow-up studies.

2.4 Genetic Algorithms

In recent years, metaheuristic approaches have been widely adopted by a number of researchers to solve buffer allocation problems. One of the most popular metaheuristic approaches dealing with this problem is Genetic Algorithms.

Genetic Algorithms (GAs) are adaptive heuristic search algorithm 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 (Calvino et al., 2007).

First pioneered by John Holland in 1975, Genetic Algorithms have been widely studied, experimented and applied in many fields. Many of the real world problems involve finding optimal parameters, which might prove difficult for traditional methods but ideal for GAs. (De Jong, 1993). GAs have been succesfully adapted to solve several combinatorial optimization problems in the literature and have become increasingly popular among metaheuristic approaches for finding optimal or near optimal solutions in a reasonable time. As explained in the book of Haupt&Haupt

(2004), the popularity of GAs among other metahuristic approaches can be attributed to the following features of GAs. GAs

- ✓ Optimize continuous or discrete variables,
- ✓ Work with numerically generated data, experimental data, or analytical functions.
- ✓ Do not require derivative information,
- ✓ Simultaneously search from a wide sampling of the cost surface,
- ✓ Deal with a large number of variables,
- ✓ Are well suited for parallel computers,
- ✓ Optimize variables with extremely complex cost surfaces,
- ✓ Provide a list of optimum variables, not just a single solution, and
- ✓ May encode the variables so that the optimization is done with the encoded variables

Genetic algorithms simulate natural evolution on a computer in which a population of abstract representations (called chromosomes) of candidate solutions (called individuals) to an optimization problem evolves toward better solutions (Akgündüz, 2008). Each solution is represented through a chromosome, which is just an abstract representation (Sivanandam and Deepa, 2008). In GAs, chromosome representation, which considers the structure of the search space and reproduction operators is one of the most difficult task. The chromosomes can be represented with various encoding schemes such as using bits, numbers, arrays, etc. and the encoding scheme depends on the structure of the problem. In other words, the way the encoding process performs differs from problem to problem.

The genetic search initializes with an initial population of individuals and proceeds throughout the generations. In each generation, individuals are stochastically selected from the current population depending on the relative fitness values and following, these individuals are modified to form a new population using crossover and mutation operators. 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 Genetic Algorithm process is illustrated in Figure 2.7.

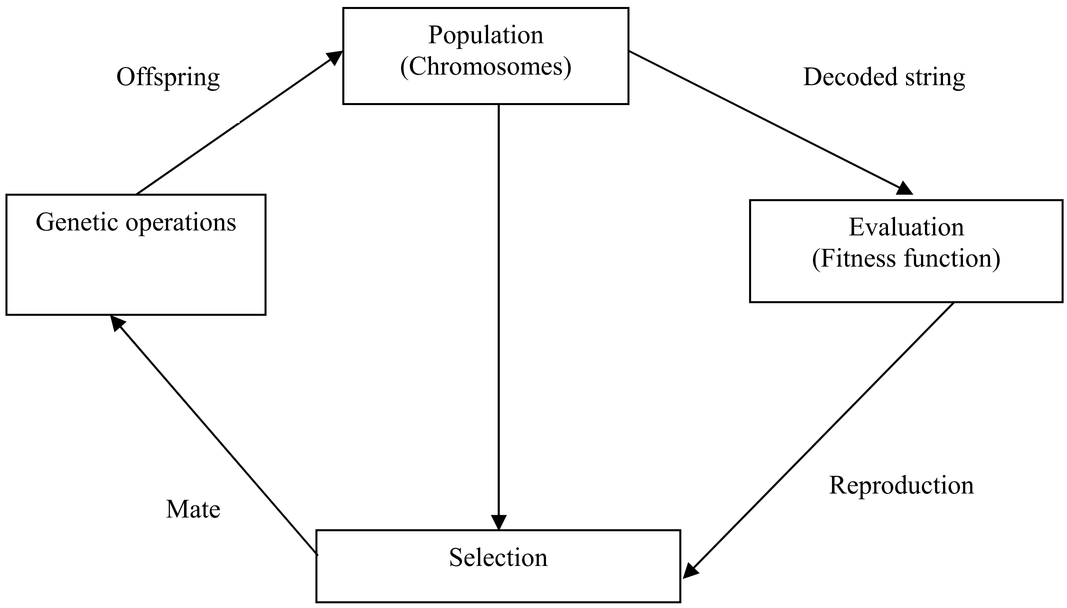


Figure 2.7 Genetic algorithm cycle

2.4.1 Terminology of Genetic Algorithms

In order to understand the philosophy of genetic algorithms, the basic terms relating to GAs must be defined. These basic components include encoding scheme, initial population, fitness function, selection scheme, genetic operators (mutation and crossover), replacement scheme and termination criteria.

In GA terminology, chromosomes are made of discrete units called *genes,* each of them controls one or more features of the chromosome. Genes are assumed to be binary digits in the original implementation of GA by Holland (see Figure 2.8). However various chromosome types have been introduced in later implementations such as given in Figure 2.9.

1	1	0	1	0	0	1	0	1	1

Figure 2.8 Binary chromosome representation

A	C	A	B	C	D	E	D	E	E

Figure 2.9 Value encoding scheme

Normally, a chromosome corresponds to 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. The use of an inappropriate coding scheme has been the cause of many GA failures (Taşan, 2007). The encoding process can be performed using bits, numbers, trees, arrays, lists or any other objects.

GAs operate with a group of chromosomes, called a *population*. The two important aspects of population used in Genetic Algorithms are the initial population generation scheme and the population size.

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.

As for population size, it is generally known that the population size depends on the complexity of the problem. Goldberg has shown that GA efficiency to reach global optimum instead of local ones is largely determined by the size of the population. To sum up, a large population is quite useful. But it requires much more computational cost, memory and time.

The *fitness* of an individual in a genetic algorithm is the 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 is the process that randomly 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 2.10 illustrates the basic selection process.

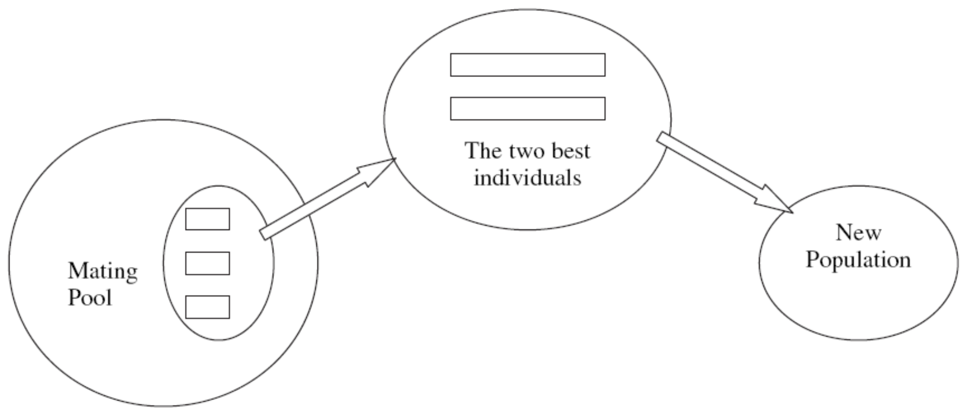


Figure 2.10 Basic selection process (Sivanandam &Deepa, 2008)

Two popular selection schemes are roulette wheel selection and tournament selection. Roulette wheel selection, proposed by Holland (1975), is the best known selection type (Gen and Cheng, 2000). Coley (2003) summarizes the roulette wheel selection as follows:

1. Sum the fitness of all the population members. Call this f_{sum}.

2. Choose a random number, R_s, between 0 and f_{sum}.

3. Add together the fitness of the population members (one at a time) stopping immediately when the sum is greater R_s. The last individual added is the selected individual and copy is passed to the next generation.

Other popular selection scheme, tournament selection is proposed by Goldberg and Deb in 1991. In this scheme, two individuals are randomly choosen from the population, and then a random value (r) is generated for the fittest individual selection. If the random value *(r)* is smaller than a probability value of the individual, that individual is selected. Otherwise, the other one is chosen. Selected individuals are returned to the population and can be chosen again as a parent (Park et al., 2003).

Crossover and *mutation* are the 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. This operation proceeds in three steps:

1. The reproduction operator selects at random a pair of two individual strings for the recombination,
2. A cross site is selected randomly along the string length,
3. Finally, the position values are swapped between the two strings selected following the cross site.

The Figure 2.11 explains one-point crossover operation.

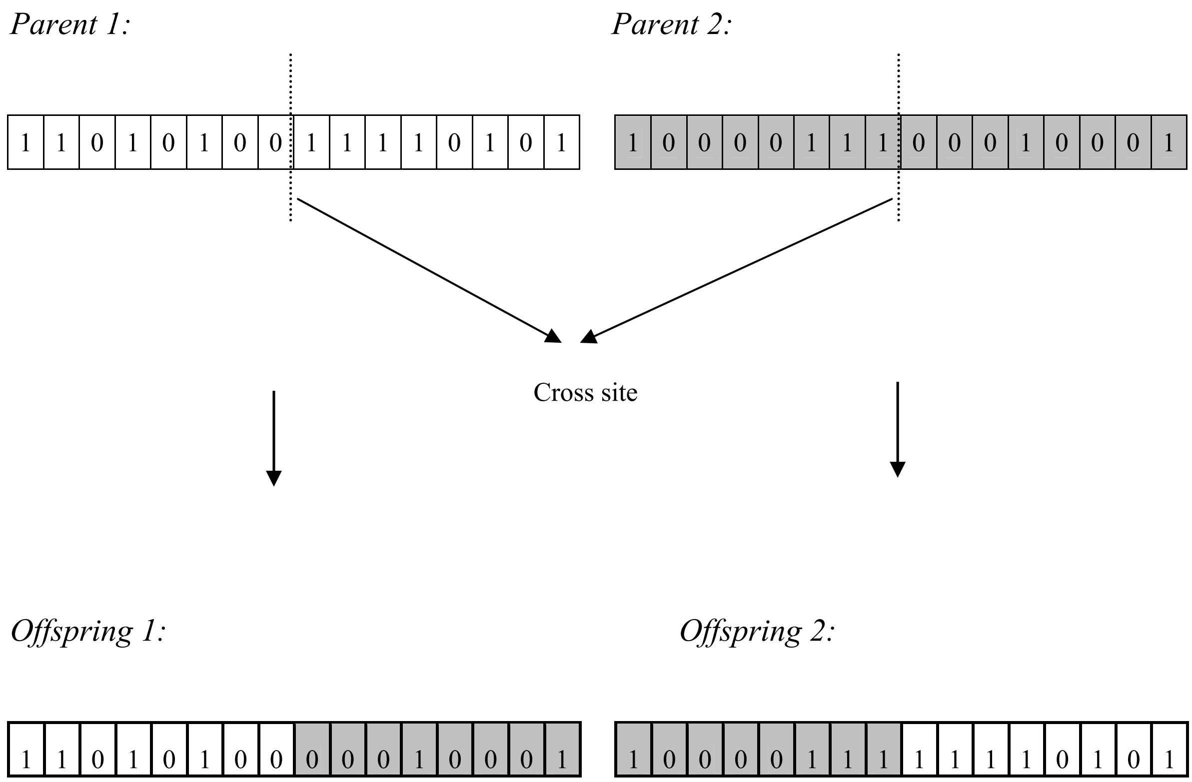


Figure 2.11 One-point crossover

Crossover rate, R_c, is defined as the ratio of the number of offspring produced in each generation to the population size. This ratio controls the expected number of chromosomes to undergo the crossover operation. 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 and Cheng, 1997).

Mutation has traditionally been considered as a simple search operator. Crossover and mutation operators are used in a genetic search in such a way that crossover exploits the current solution to find better ones, and mutation helps for the exploration of the whole search space. Mutation is viewed as a background operator to maintain genetic diversity in the population. It introduces new genetic structures in the population by randomly modifying some of its building blocks. Mutation helps to escape from local minima's trap and maintains diversity in the population. It also keeps the gene pool well stocked, and thus ensuring ergodicity. A search space is said to be ergodic if there is a non-zero probability of generating any solution from any population state (Sivanandam et al., 2008). There are many different forms of mutation for the different kinds of representation. Figure 2.12 shows the simplest mutation, which is performed by changing the value of a randomly selected gene from 0 to 1 (or from 1 to 0) in a binary string.

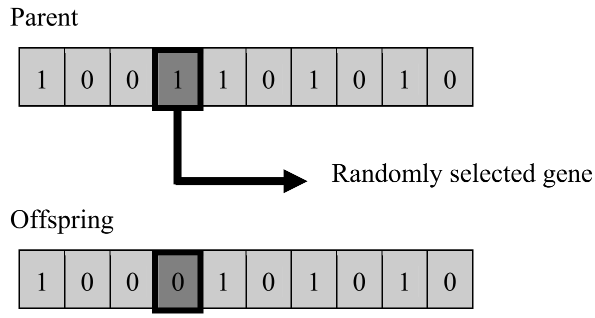


Figure 2.12 The simplest mutation

Mutation rate, denoted by R_m, controls the rate at which new genes are introduced into the population for trial. 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 and Cheng, 1997).

A *replacement scheme* is used to decide which individual stay in a population and which are replaced by offsprings, generated by crossover or mutation. The individuals of the new generation can be generated in three different ways. These ways are summarized as follows:

(i) individuals from the current generation,
(ii) offspring product of crossover,
(iii) individuals who underwent mutation.

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.

Through a *termination criterion* embedded into the genetic algorithm it is decided whether to continue the genetic search or stop the search. The various stopping conditions which are explained in the book of Sivanandam&Deepa (2008) are summarized as follows:

- *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.

As seen in Figure 2.13, the genetic search is carried out following all the steps explained above.

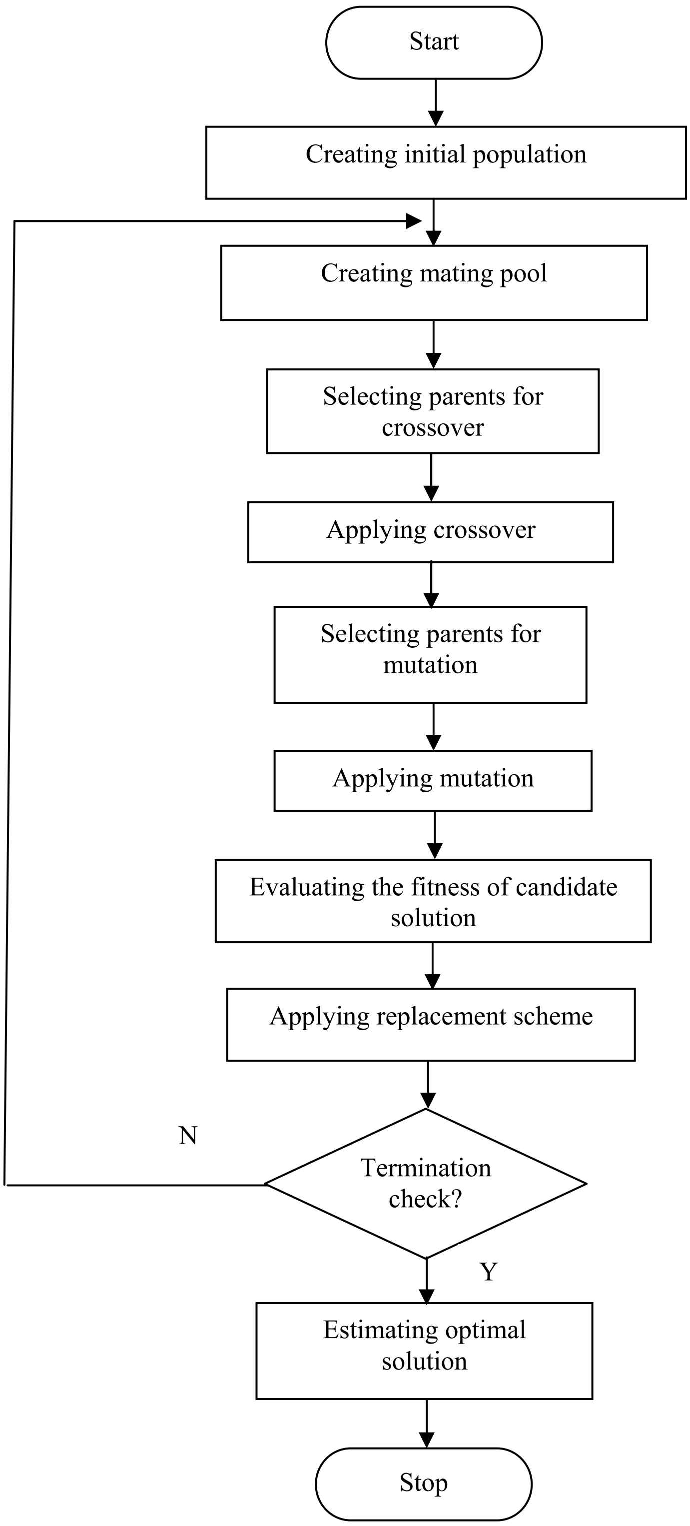


Figure 2.13 The flow chart of genetic algorithm

2.4.2 Identifying Efficient GA Control Parameters

Another important decision faced in many GA applications is the identification of efficient GA control parameters to ensure high performance. Although GA seems to be a robust algorithm which contains same operators and has the same algorithmic logic for different applications, in fact the algorithm itself is significantly different for distinct problems. The main reason is that GA has several parameters and any combination of these parameters has different impacts on the performance of GA (Boyabatlı&Sabuncuoğlu, 2004). As a result, identification of the efficient GA parameters plays a crucial role on the quality of solution and convergence speed of GA application. The control parameters such as the population size, the generation number, the crossover rate, the mutation rate, the selection type and the termination criteria must be chosen with care before the evolution starts. Since, the efficient control parameter values are problem specific, it is necessary to carry out extensive experimental studies to identify the values of these control parameters. But it should be noted that this is often a time consuming task.

The two important aspects of parameter setting efforts which are explained in the study of Eiben et al. (1999) are used to identify the efficient GA control parameters: *parameter tuning* and *parameter control.* Parameter tuning is commonly used in evolutionary computation. Before the algorithm performs, the values of each parameter are selected and then the genetic search starts with these parameter values and these values remain fixed during the iterative search. In contrast to parameter tuning, in the case of parameter control the algorithm starts with some initial parameter values and these values are allowed to be changed during the run.

2.5 GA-based Simulation Optimization

The optimization of manufacturing system simulations is one of the most important and most researched subjects in discrete event simulation (Boesel et al. 2001; Fu et al. 2000). In simulation optimization, simulation is used as a tool to optimize certain parameters of a simulated system in order to improve the system performance. Simulation-based optimization has been a fruitful domain considering the approximate optimization techniques, such as stochastic approximation, random search, metaheuristics (Can et al., 2008). Simulation-based optimization techniques have been widely applied to various combinatorial optimization problems. Among these optimization techniques, use of metaheuristics, in particular genetic algorithms has led to an increased interest in simulation optimization.

In GA-based simulation optimization, GA is integrated with simulation modeling during the calculation of the fitness value of the selected individuals. The fitness value as a performance function is estimated by means of simulation. For every individual of a particular generation simulation results are used to assess the fitness of the corresponding individual. During the survey of current relative literature, we have noted quite number of studies (Bulgak et al., 1995; Wellman and Gemmill, 1995; Ding et al., 2003; Boyabatlı and Sabuncuoğlu, 2004; Gholami and Zandieh, 2008) successfully integrating discrete-event system simulation with GAs for the optimization of manufacturing systems.

CHAPTER THREE
REVIEW OF CURRENT LITERATURE

This chapter presents a review of current relevant literature. Particularly, we focused on studies dealing with the problem of buffer allocation for capacity improvement at bottleneck stations. The surveyed literature has been discussed with respect to methodology (i.e. exact methods or hybrid approaches), application environment (i.e. real or hypothetical), objective function (i.e. single or multiple objective function) and the type of the problem (deterministic or stochastic) as seen in Figure 3.1.

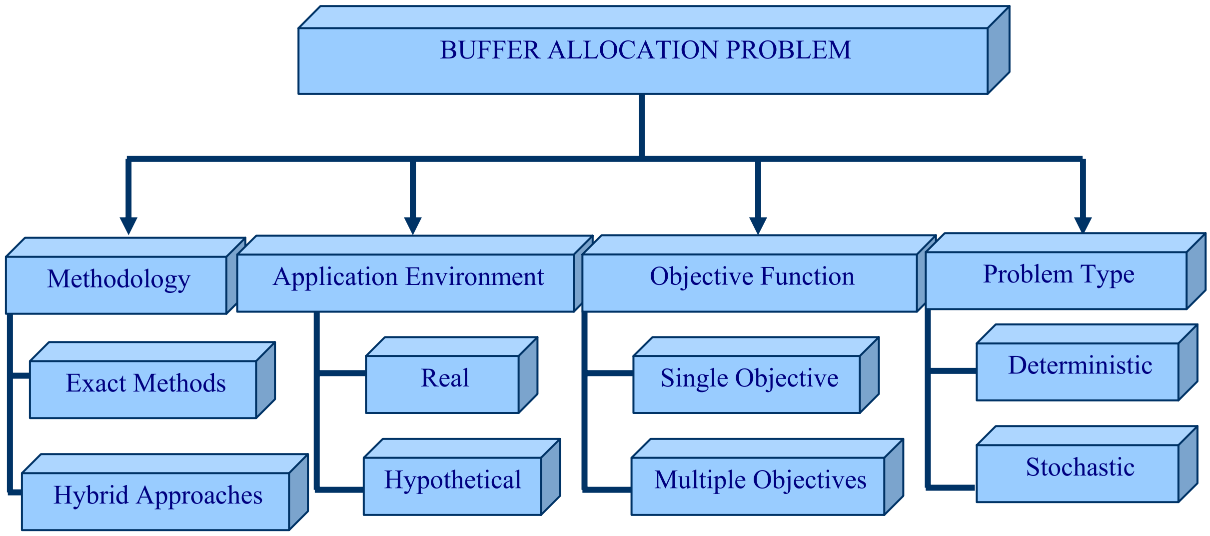


Figure 3.1 Structural framework for reviewing

Following the review of current relevant literature, motivation for this M.Sc. study is explained in detail.

3.1 Discussion of Current Relevant Literature

Buffer allocation problems have been studied by many researchers since 1950s (Vladzievskii, 1950, 1951; Sevastyanov, 1962; Buzacott 1967). During the last twenty years, there has been even a growing interest in this problem. A chronological list of the work published since 1998 is given in Table 3.1 As seen in the table, the features of these studies are summarized with respect to the methodology, the application environment, the objective functions and type of the problems studied.

Table 3.1 Overview on buffer allocation in the literature since 1998

Paper	Methodology		Application Environment		Objective Function		Problem Type	
	Exact Method	**Hybrid**	**Real**	**Hypothetic**	**Single**	**Multiobjective**	**Deterministic**	**Stochastic**
Lutz et al. (1998)		Simulation-Tabu search		x		Max througput Min. İnventory		x
Yamashita and Altiok (1998)		Simulation-Dynamic programming algorithm		x	Min total buffer space			x
Vouros and Papadopoulos (1998)		Simulation-Knowledge-based system (ASBA2)		x	Max throughput			x
Harris and Powell (1999)		Simulation-Simplex search algorithm		x	Max throughput			x
Spinellis and Papadopoulos (2000)		Genetic Algorithm-Simulated annealing algorithm-Decomposition method		x	Max throughput			x
Gershwin and Schor (2000)	Gradient method-dual solution			x		Min total buffer space Max throughput		x
Papadopoulos and Vidalis (2001)	Segmentation approach			x	Max throughput			x

Continuation of Table 3.1

Dolgui et al. (2002)		Genetic Algorithm-Markov-model aggregation approach		x		Min cost Max throughput		x
Altiparmak et al. (2002)		Arttificial neural network, Simulated Annealing-Approximation method		x	Max throughput			x
Patchong et al. (2003)		Simulation-Markov chain models	x			Min capital investment Max profit		x
Shi and Men (2003)		Hybrid nested partitions-Tabu search		x	Max throughput			x
Nourelfath et al. (2005)	Ant system algorithm			x	Max system efficiency			x
Hillier and Hillier (2006)	Cost-based modeling			x	Max revenue			x
Nahas (2006)		Decomposition-type approximation-Degraded ceiling approach		x	Max average throughput			x

Continuation of Table 3.1

Bulgak (2006)		Artificial neural network-Genetic algorithm		x	Max throughput			x
Altiparmak et al. (2007)	Artificial neural network metamodeling			x	Comparison of metamodels			x
Dolgui et al. (2007)		Genetic algorithm-Branch and bound approach		x	Max given function			x
Nourelfath et al. (2008)		Ant colony optimization-Simulated annealing		x	Max throughput			x
Shi and Gershwin (2009)	Nonlinear programming approach			x	Max profit		x	
Massim et al. (2010)		Artificial immune system optimization algorithm-decomposition method		x		Max throughput Max profit		x

In the literature, one broad categorization of methodologies for buffer allocation problems can be made by dividing into those that employ analytical methods and those that employ simulation. A number of studies also use some heuristics in combination with these approaches. Some of the analytical methods use Markov process to analyze the short production lines (Hunt 1956, Buzacott, 1967, Hillier and So, 1991, Hendricks 1992, Hillier et al., 1993, Hillier 2000, Hillier and Hillier 2006) while the others employ approximation methods such as decomposition method (Gershwin and Schor 2000, Helber 2001, Tempelmeir 2003, Shi and Men 2003, Nahas et al., 2006) and aggregation approach (Dolguie et al. 2002, 2007) in conjunction with an optimization method to determine the optimal buffer sizes for long production lines. During the survey of current literature, it has been noted that majority of studies listed in Table 3.1 including Powell and Pyke 1998, Yamashita and Altiok 1998, Harris and Powell 1999, Allon et al. 2005, Bulgak 2006, Sabuncuoglu et al. 2006, and Altiparmak 2007 employed an optimization method (i.e., exact methods or metaheuristics) in conjunction with simulation to solve the buffer allocation problem in production lines. In these studies, simulation is used to obtain the value of the objective function for a set of inputs. To search for an optimal solution, the value of the objective function is then communicated to the optimization method. Considering the capability of modeling large and complex systems by simulation, simulation optimization method is widely used for solving buffer allocation problem as well as other manufacturing design problems.

The optimization methods used for solving buffer allocation problem can be classified as complete enumeration, dynamic programming, various search methods and metaheuristics. Among these methods, metaheuristic methods such as Genetic Algorithms (Dolguie et al, 2002, 2007), Tabu Search (Lutz et al, 1998, Shi and Men, 2003), Simulated Annealing (Spinellis et al. 1999) and Ant Colony Optimization (Nourelfath, 2008) have been successfully used to solve buffer allocation problems.

Lutz et al. (1998) developed a simulation-search heuristic procedure based on tabu search, combined with simulation for the buffer location and storage size problem in a manufacturing line. Simulation is used to model the manufacturing process and tabu search is used to guide the search to overcome the problem of being trapped at local optimal solutions. The procedure employs a Swap Search routine to identify good performing buffer profiles and determine the maximum output level for any given storage level and also Global Search routine to locate promising neighborhood of buffer profiles quickly. The objective of the specified problem is to maximize the output level of the line for the given buffer profile and minimize the total number of storage spaces in the production line given the buffer profile.

Yamashita and Altiok (1998) implemented a dynamic programming algorithm that uses a decomposition method to approximate the system throughput at every stage to find the minimum total buffer allocation for a desired throughput in production lines with phase-type processing times. Powell and Pyke (1998) studied simple asynchronous assembly systems with random processing times and developed simple heuristic rules that can be used to improve existing operations and to support line designers who are faced with increasingly rapid cycles of new product introduction. They also applied these heuristics in their study to several larger systems and discovered that perform quite well. Moreover in 1998, Vouros and Papadopoulos presented a knowledge based system, ASBA2, in close cooperation with a simulation method to maximize throughput of production line. In this study, ASBA2 determines near optimal buffer allocation plans and simulation provides ASBA2 with performance measures concerning production line behaviour.

Harris and Powell (1999) developed a simple search algorithm to determine optimal allocation of a fixed amount of buffer capacity in an *n*-station serial line. The algorithm, which is an adaptation of the Spendley-Hext and Nelder-Mead simplex search algorithms, uses simulation to estimate throughput for every allocation considered.

Spinellis and Papadopoulos (2000) presented two stochastic approaches -genetic algorithms and simulated annealing- and compared them for solving the buffer allocation problem in reliable production lines. The problem entails the determination of near optimal buffer allocation plans in large production lines with the objective of maximizing their throughput. The allocation plan is calculated subject to a given amount of total buffer slots using simulated annealing and genetic algorithms and decomposition method is used to calculate the throughput of the production system. Gershwin and Schor (2000) were concerned with two problems: *primal* problem and *dual* problem. The primal problem, which minimizes total buffer space under a production rate constraint, is solved using the dual solution. The dual problem, which maximizes production rate subject to a total buffer space constraint, is solved by means of a gradient method in their study. Hiller (2000) considered optimal allocation of buffer storage spaces in unpaced production lines with variable processing times.

In 2001, Papadopoulos and Vidalis presented a heuristic approach based on segmentation for buffer allocation problem in short unbalanced production lines consisting of up to six machines that are subject to breakdowns. Sörensen and Janssens (2001) studied on *n*-machines production system which machines are separated by a finite buffer and subject to breakdowns. They investigated how the allocation of buffers can be expressed as a non-linear optimization problem in which the total cost of installing and using the buffers are minimized.

Dolgui et al. (2002) focused on a flow line manufacturing system organized as a series of workstations separated by finite buffers. The authors proposed a genetic algorithm where the tentative solutions are evaluated with an approximate method based on the Markov-model aggregation approach. The performance of the flow-line is measured in terms of average production rate (i.e. the steady-state average number of parts produced per unit of time). In another study, the same authors (Dolgui et al., 2007) focused on the buffer space allocation problem for a tandem production line where the parts are moved from one machine to the next by some kind of transfer mechanism with unreliable machines is considered. They measured the performance of the proposed genetic algorithm with respect to the average steady-state production rate of the line and the buffer equipment acquisition cost. The fitness function is formulated as a maximization function considering amortization time of the line, revenue for the sold production per time unit, buffers acquisition cost.

Grant et al. (2002) employed a simulation-based approach to determine delivery dates of orders based on dynamic buffer adjustment coupled with various methods to forecast the amount of buffer required by. The basic concept of their study is that, if a good job on buffer adjustment can be done, than the current behavior of the system can be exploited more effectively to actually establish promise dates (Grant et al., 2002).

Altiparmak et al. (2002) integrated artificial neural networks metamodel approach with simulated annealing method for buffer size optimization in an asynchronous assembly system. An approximation method using Taylor series expansion probability generating function technique is suggested for the analysis of the average steady state throughput of serial production lines with unreliable machines.

Shi and Men (2003) introduced a hybrid algorithm based on nested partitions and a Tabu search method for production line optimization and they focused on maximizing the production rate of the line under a total buffer space constraint, rather than the profit of the line.

Roser et al. (2003) focused on the area of buffer allocation by creating a prediction model to estimate the effect of additional buffer capacity onto the system performance using only a single simulation. Their proposed method works for large systems, balanced and unbalanced systems, and serial and parallel manufacturing systems and the authors stated that their approach can be adapted to non-manufacturing discrete event systems.

An excellent illustration of the value to industry in solving problems of buffer allocation was given by Patchong et al. in 2003. The authors demonstrated how methods for buffer allocation in designing PSA Peugeot Citroën car-body shop yielded substantial profits.

Chararsoghi and Nahavandi (2003) proposed a heuristic approach to find the optimal allocation of buffers that maximizes throughput in the system. In this study, since the algorithm finds allocation without predetermined total buffer capacity, the proposed algorithm finds the optimal, or near optimal, allocation with less WIP.

Diamantidis and Papadopoulos (2004) also presented a dynamic programming algorithm for optimizing buffer allocation based on the aggregation method given by Lim, Meerkov, and Top (1990). The main focus of the authors was to suggest new dynamic programming based approaches to the production line design, rather than focusing on profit maximization (Shi and Gershwin, 2009).

Nourelfath et al. (2005) developed an efficient heuristic approach to solve optimal design problem. The aim of this study was to maximize the efficiency of system subject to a total cost constraint. The optimal design problem is solved by developing and demonstrating a problem-specific ant system algorithm inspired by the work of real ant colonies that exhibit highly structured. It has been noted that this algorithm can always find near-optimal or optimal solutions quickly. In the next publication of these authors in 2008, to estimate series-parallel production line performance, an analytical decomposition type approximation was proposed. The optimal design problem in this paper was formulated as a combinatorial optimization one where the decision variables are buffers and types of machines. The objective was to maximize production rate subject to a total cost constraint. To solve this design problem, ant colony optimization and simulated annealing methods were used and their performance were compared empirically through several test problems.

Smith and Cruz (2005) solved the buffer allocation problem for general finite buffer queueing networks in which they minimized buffer space cost under the production rate constraint, but they did not consider the average inventory cost. Alon et all. (2005) presented a stochastic algorithm for solving the buffer allocation problem, based on the cross-entropy method. Colledani et al. (2005) presented an approximate analytical method for the performance evaluation of a production line with finite buffer capacity, multiple failure modes and multiple part types.

Bulgak (2006) presented a new approach in optimal buffer allocation problem of split-and-merge unpaced open assembly systems. In this approach, GA based artificial neural networks metamodeling procedure was developed and buffer allocations to accommodate the work-in-process inventories were optimized in an attempt to maximize the overall system production rate using.

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

Nahas et al. (2006) described a new local search approach for solving the buffer allocation problem to maximize the average throughput in unreliable production lines. An analytical decomposition-type approximation was used to estimate the production line throughput. It has been noted that the proposed approach allows the allocation plan to be calculated subject to a given amount of total buffers slots in a computationally efficient way.

Sabuncuoglu et al. (2006) characterized the optimal buffer allocation problem and specifically studied on the cases with single and multiple bottleneck stations under various experimental conditions. Moreover, an efficient heuristic procedure to allocate buffers in serial production lines was developed to maximize throughput. From the results of the computational experiments in this study, it can be stated that the proposed algorithm was very efficient in terms of both solution quality and CPU time requirements.

Altioklar et al. (2007) reviewed various applications of artificial intelligence techniques on manufacturing systems problems, in particular related to artificial neural networks. Due to this context, a metamodeling approach in terms of artificial neural network metamodel has been proposed for asynchronous assembly systems buffer design problems.

Um et al. (2007) presented the simulation methodology for the buffer size determination in flexible manufacturing system cell line which was categorized into cell buffer and machine buffer. The simulation model was designed for developing flexible manufacturing system design in an Aspect-oriented environment. Aspect-oriented approach provides a new way of thinking about flexible manufacturing system simulation design. They used the evolution strategy in order to find the optimal buffer sizes in the flexible manufacturing system cell line. Another simulation based study which discusses an optimal buffer allocation for short unpaced reliable production line was developed by Othman et al. in 2007. Simulation method was used to estimate throughput rate of the production line. As a result of this study showed that the allocation of buffers affects the throughput as an increased rate.

Qudeiri et al. (2008) presented a new GA-simulation-based method to find the nearest optimal design for serial-parallel production lines. 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 have been considered for optimization. As a result, they attempt to find the nearest optimal design of a serial–parallel production line that will maximize production efficiency. According to the authors, one of the important tasks in using a GA is how to express a chromosome. For the efficient use of a GA, their GA methodology is based on a technique that is called the gene family arrangement method (GFAM), which arranges the genes inside individuals.

In a recent study, Shi and Gershwin (2009) introduced an unconstrained problem and they adopted a nonlinear programming approach for maximizing profits through buffer size optimization for production lines. In this study, both buffer space cost and average inventory cost with distinct cost coefficients for different buffers and a nonlinear production rate constraint have been considered. Battini et al. (2009) also focused on allocation of storage capacity in serial production lines. They employed a new experimental cross matrix as a tool to determine the optimal buffer sizes. Using a simulation approach, this study evaluates the effects of workstation reliability parameters on buffer capacity level.

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 is used to determine optimal buffer allocation for maximum line throughput and maximum line economic profit.

3.2 Motivation for This Study

As a result of surveying current relevant research, we might state that various approaches have been applied for the optimization of buffer allocation problem. Some of the most popular approaches include simulation, exact methods, metaheuristics or hybrid methods integrating different approaches. Some researchers employed exact methods to find optimal buffer allocation. However, the applicability of these methods is restricted to very specific systems.

Simulation method provides many advantages in comparison to exact methods to realistically model the buffer allocation problem in a large and complex system. However, the major drawback of simulation methods for practical applications is that it is generally an expensive tool in terms of time and monetary resources. Another drawback of simulation method is that it is a descriptive tool rather than a prescriptive tool. In other words, to use simulation method for optimization of decision variables is computationally very expensive. Due to the success of metaheuristics such as Ant Colony Optimization, Evolutionary Computation, Simulated Annealing, Tabu Search in solving combinatorial optimization problems, the trend in recent years is to integrate metaheuristics with simulation. The advantage of the metaheuristic approaches is that they always reach feasible solutions, but they do not guarantee optimality. We believe that when metaheuristic approaches are integrated with simulation and the simulation model of the production line which models the stochastic behaviour of production line such as random machine breakdowns and stochastic processing times is used to evaluate the fitness function of the metaheuristic approach, the behaviour of real-world systems will be captured more realistically.

Therefore, the main goal of this study is to develop a hybrid method which combines a very widely used metaheuristic method, genetic algorithms and discrete-event simulation model for optimization of buffer allocation problem. It is hoped that, the suggested hybrid approach for buffer allocation problem will be effective in improving the capacity of the production line studied and so that the company's competitiveness in the long run will be enhanced.

CHAPTER FOUR
PROPOSED HYBRID APPROACH

The main objective of this M.Sc. study is to develop a a hybrid approach combining genetic algorithm and simulation to improve the capacity of a production line in a real manufacturing environment. The proposed approach consists of genetic algorithms (GAs) and simulation modeling. GAs search the solution space by building and than evolving a population of solutions. The objective in this hybrid approach is to bring together the advantages of GAs and simulation modeling in solving buffer allocation problem. The main advantage of GAs over those based in sampling the neighbourhood of a single solution is that they are capable of exploring a larger area of the solution space with a a smaller number of objective function evaluations. In this study, during the iterative search the simulation model of the manufacturing system studied is used to evaluate the objective function of GAs. Reflecting dynamic and stochastic features of the manufacturing system, this simulation model helps to solve the buffer allocation problem in a more efficient and also more realistic way.

The following sections present how the GA is adapted to solve the buffer allocation problem and also the specifications of proposed hybrid simulation-GA approach.

4.1 Hybrid Simulation-GA Approach

As given in Figure 4.1, the proposed hybrid simulation-GA approach is implemented in the second phase of this study. First, a simulation-based procedure is implemented to identify bottleneck stations in the system. The steps followed during these two phases are given in Figure 4.1.

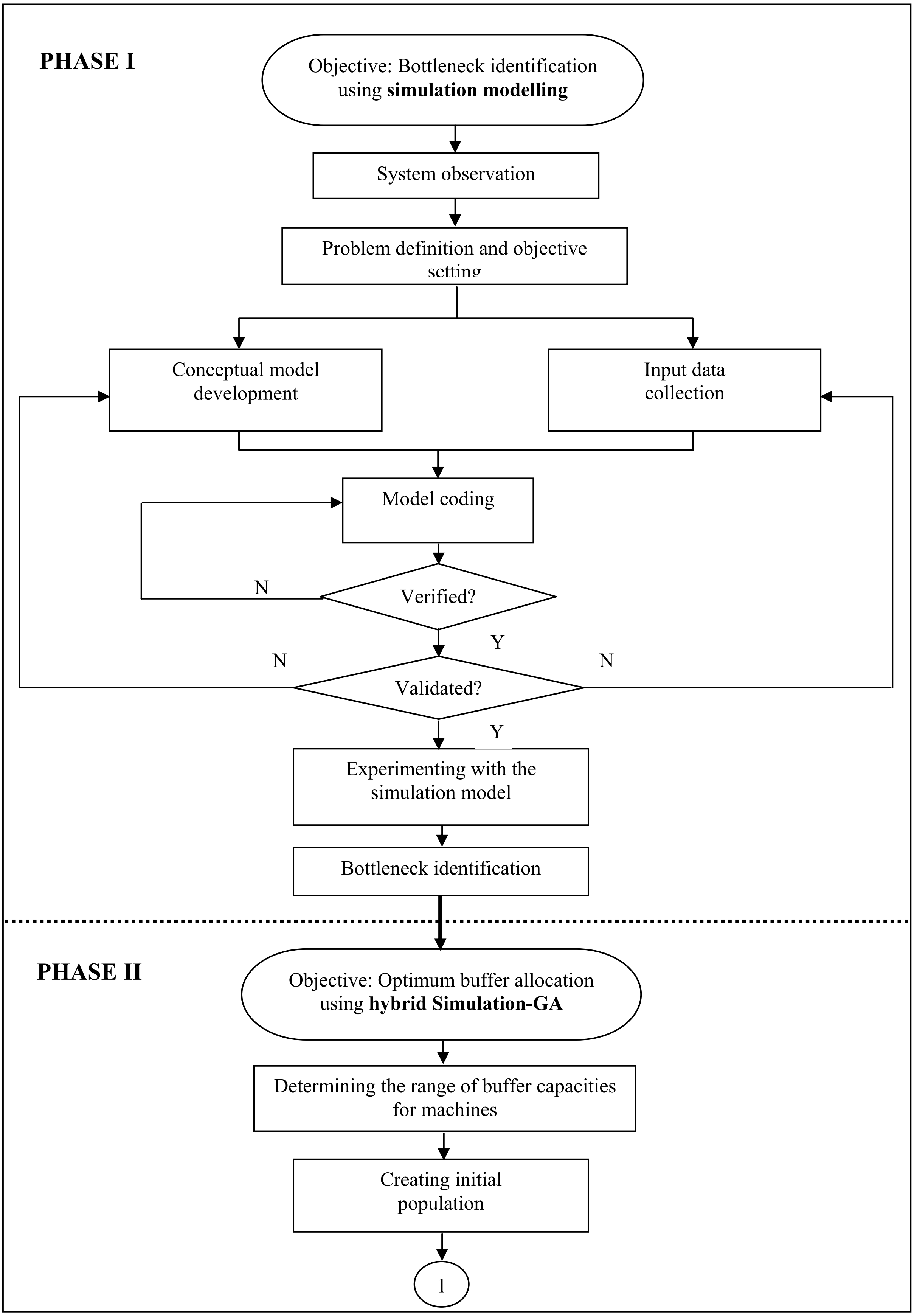
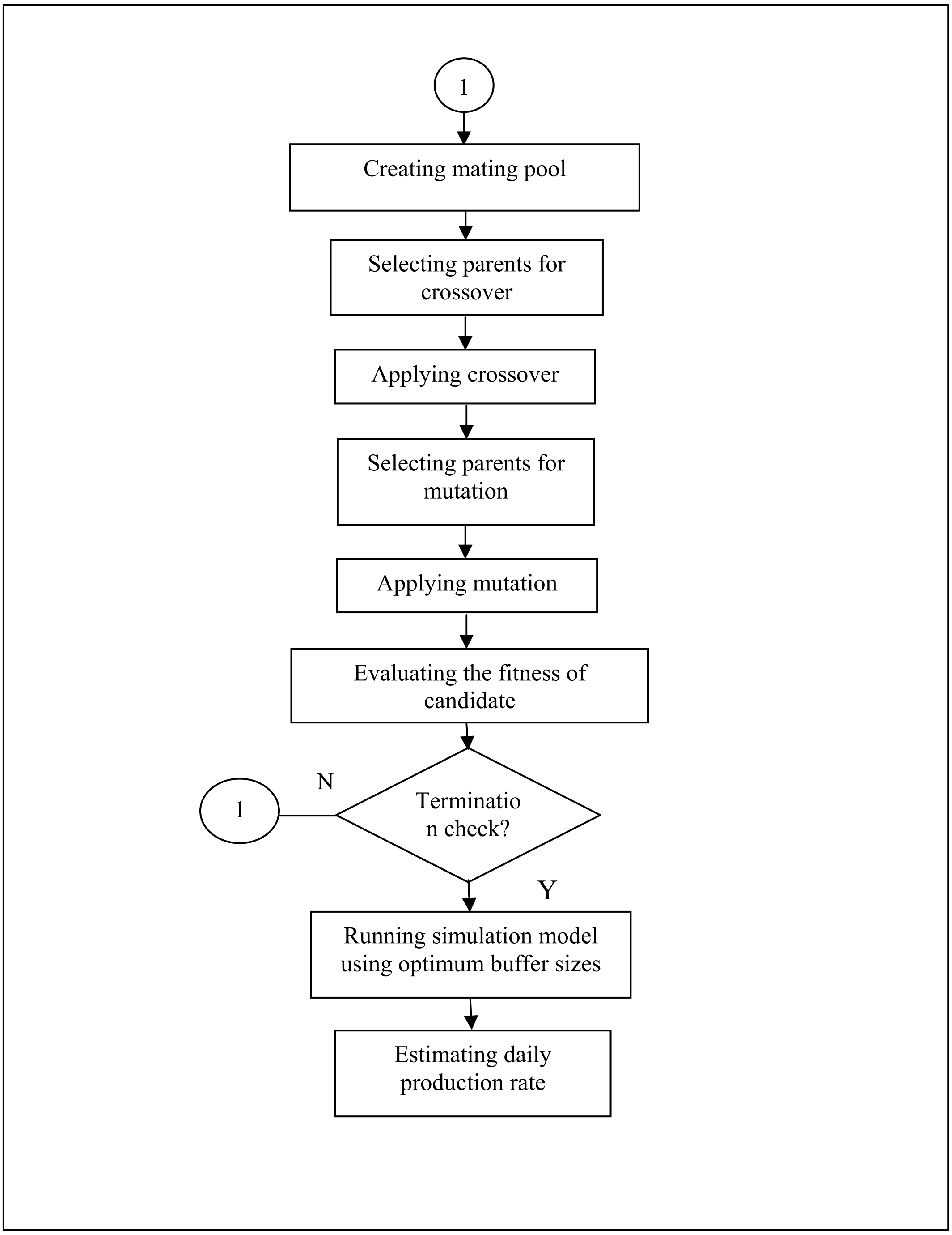


Figure 4.1 The flow diagram of the proposed hybrid approach



Continuation of Figure 4.1

4.2 Specifications of the GA -Based Approach

In this section, the general specifications of the proposed GA-based approach are introduced. As it can be clearly seen from the Figure 4.1, the second phase of this proposed approach is based on the genetic search process for optimum buffer allocation. First, this process begins with the creation of the initial population. Once the initial population is generated, the fitness value of each individual is evaluated by a detailed simulation model of the production system. It should be noted that this simulation model was constructed for bottleneck identification in the first phase of the study. Next, the survival probabilities which are used to select individuals for possible crossover and mutation are calculated.

The selection of the individuals from the current population for reproduction has been done using roulette wheel selection scheme and the selected individual pairs produce the offsprings under genetic operations named as crossover and mutation. In the following phase, the quality of these newly created offsprings is evaluated using simulation model of the production system and the new population is generated by taking into account the replacement strategy (i.e elitism and random replacement). Lastly, the satisfaction of termination criterion is checked. If it is satisfied, this new generation becomes the final generation of the genetic search process and the best individual of this generation is accepted as optimum buffer sizes. If the termination criterion is not met, the generated population is transferred to the next generation for crossover and mutation operations.

In the following subsections, the specifications of the proposed GA based approach such as chromosome representation, initialization, fitness evaluation, selection, crossover, mutation, replacement and search termination schemes are explained in detail.

4.2.1 Chromosome Representation

Chromosome representation is one of the most important task for a successful application of GA. The way bit strings can code differs from problem to problem. In general, a binary string representation is the most common way of encoding genes because of its compatibility with genetic operations. The binary alphabet {0,1} is used to represent these genes which are encoded as a finite length string.

It should be noted that the binary string representation of coding schemes is used for buffer allocation problems. The whole string can represent an integer value, so this has been mapped each buffer size in this problem. Each buffer size represented increases with string length. Having identified bottleneck machines in the production system, next we employ this GA-based hybrid approach to decide how to allocate buffers to the machines in the system so as to improve the performance of the system. In this study, a chromosome represents a possible configuration of the buffer sizes as decision variables. Hence, each chromosome is composed of some unique parts to be allocated. For instance, the buffer size configuration of 13, 4, 15, 7 units means that 13, 4, 15 and 7 units of buffer stock will be allocated to the four machines. The integer value of each buffer size as a decision variable is represented as a binary string and the length of the string depends on the upper bound of each variable. When upper bound of a buffer size equals 15 as a constraint, four binary bits will be used to represent the variable. As a result, a chromosome of the buffer size configuration, *Buffer_sizes = [13, 4, 15, 7],* given in Figure 4.2 will be represented as a string of c*hrom*.

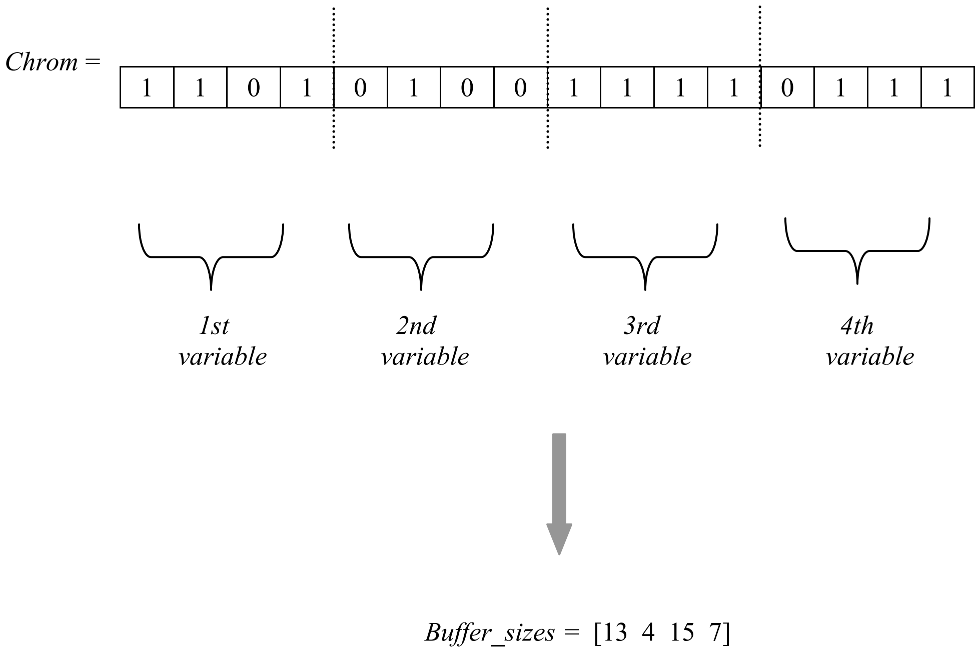


Figure 4.2 Binary coding representation of chromosome

4.2.2 Initial Population

Initial population of chromosomes has been generated randomly and heuristically in this study. Such an initialization approach may yield illegal chromosomes in two aspects: violating the system constraints (i.e. facility capacity constraints, material handling constraints) and / or violating the upper bound (i.e. total number of buffers to allocate) (Gen&Cheng, 1997). It should be noted that a feasible checking step is constructed through the initialization procedure and as a result, infeasible solutions, which violate the system constraints, are not allowed in the population.

The initialization procedure can be summarized as follows:

- ***Initilization Procedure:***

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

generate a chromosome, $chrom_x$;

if *$chrom_x$* is not feasible
i = i-1;
end
end

4.2.3 Fitness Evaluation

The aim of the GA-based simulation optimization approach proposed in this study is to determine optimum buffer capacities. Genetic algorithm module is combined with the simulation model in a closed loop configuration. In this configuration, the simulation model of the production line is used to estimate the fitness value i.e. average daily production rate as a function of buffer sizes. To search the solution space to determine optimal buffer sizes, the estimated fitness values are then communicated to the genetic algorithm module.

The fitness evaluation procedure in our proposed approach is given below:

- ***Fitness Evaluation Procedure:***

for any individual to obtain the fitness value
write *Buffer_sizes* to an external file
execute the system file to run the simulation model
read average fitness value and maximum fitness value from an external file
end

4.2.4 Selection Scheme

The most commonly used traditional GA selection schemes are roulette wheel and tournament selection. In roulette wheel selection, the principle is a linear search through a roulette wheel with the slots in the wheel weighted in proportion to the fitness values of individuals. Unlike the roulette wheel selection, tournament selection provides selective pressure by holding a tournament competition among the individuals. To decide which selection scheme to use in this study, a comparative experimental study has ben carried out considering the existing production system and in this experimental study the total buffer capacity, population size, generation number, crossover rate, mutation rate, and elitism percent are set to 60, 30, 40, 0.6, 0.042, and 25%, respectively. The results of this experimental study comparing the two selection schemes based on average production rate, maximum production rate and total buffer capacity are given in Table 4.1. As it is seen in the table, the performance of the roulette wheel selection scheme dominates the performance of tournament selection scheme in all criteria.

Table 4.1 Results of experiment for selection scheme

	Selection Scheme	
	Tournament selection	**Roulette wheel selection**
Average	0.1345	0.1374
Best solution	0.1351	0.1381
Buffer sizes	B={ 21 13 13 11}	B={ 5 15 12 14}
Total buffer capacity	58	46

Hence, as a result of this pilot experimental study, it has been decided to use the roulette wheel sampling which employs the fitness proportionate selection method. In roulette wheel selection, first, the fitness values of the individuals within the population are scaled and then cumulative survival probabilities are calculated. Below the algorithm of this selection sheme is given.

- ***Fitness Proportionate Procedure:***

for i = 1 to *pop_size*

calculate *Total_Fitness* *i.e.,* the sum of all fitness values in the population.

end

for i = 1 to *pop_size*

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

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

end

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.

The selection procedure is given as follows:

- ***Selection Procedure:***

loop until *offspring_candidate* = 2 i.e., two individuals to select a parent;
generate a random number, *rand_num,* between 0 and 1
for i = 1 to *pop_size*
if *rand_num* less than cumulative survival probability
offspring_candidate has been selected
end
end
end

4.2.5 Crossover (Recombination)

Crossover is the main genetic operator that combines two chromosomes from the current population to produce two new offsprings for the next population. The idea behind crossover operation is that the offspring may be better than both of the parents if it takes the best characteristics from each of the parents. Crossover operation occurs during evolution according to a specified crossover probability rate. The crossover probability rate, R_c, is used to determine if the offspring will represent a blend of the parents. If no crossover takes place according to the crossover probability, the two offsprings are clones of their parents. But, if crossover occurs, the two offsprings are produced for the next population.

Once the fitness proportionate calculations and the selection process have been carried out, crossover operation can begin. The proposed GA performs two-point crossover operation (Sivanandam&Deepa., 2008). In two-point crossover, first, two crossover points are chosen randomly and then the contents between these points are exchanged between two parents to produce two new offspring. The process of crossover operation is illustrated in Figure 4.3.

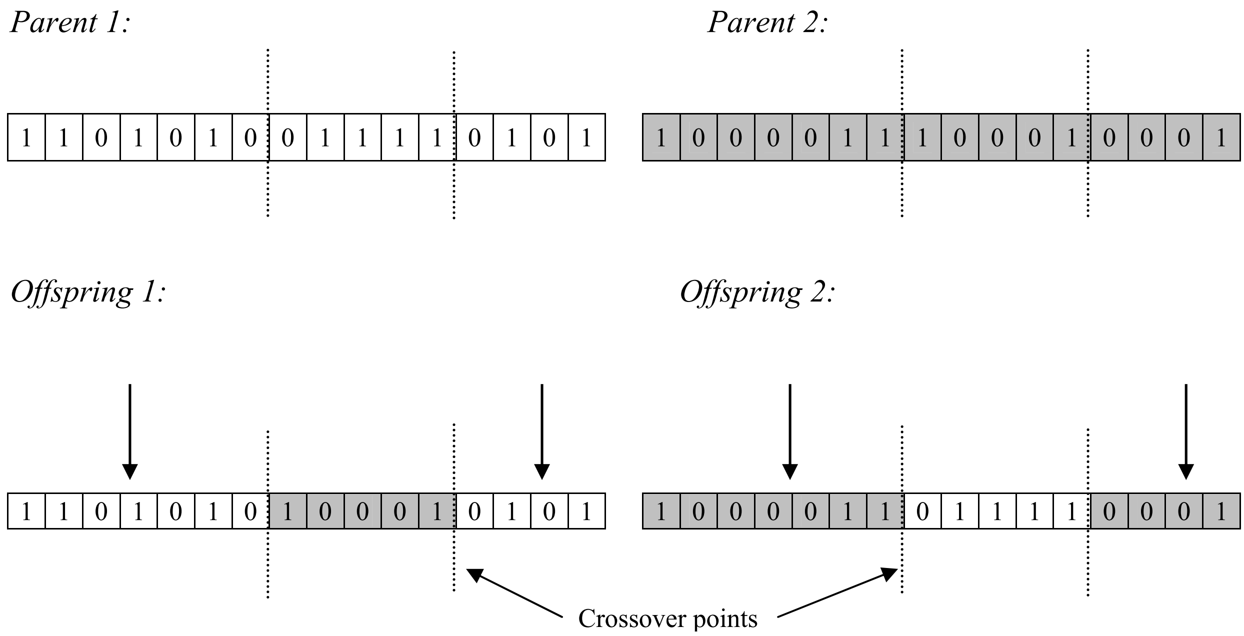


Figure 4.3 Two-point crossover

4.2.6 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. This can result in entirely new chromosomes being added to the next population. Similar to the crossover operation, mutation operation occurs during evolution according to a specified mutation probability rate defined as R_m. This rate is used to decide how often gene values of chromosomes will be mutated.

In our proposed GA, the mutation operator, which is applied to each gene, is implemented by interchanging mutation (Sivanandam&Deepa., 2008). In the interchanging mutation, two random positions on the chromosome are chosen and the bits corresponding those pozitions are interchanged 0 to 1 or 1 to 0. Figure 4.4 illustrates how the interchanging mutation operates.

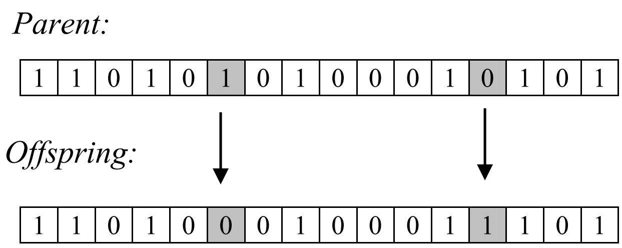


Figure 4.4 Interchanging mutation

4.2.7 Replacement Scheme

Replacement is a vital stage in GAs to form a new population for the next generation. This scheme is used to decide which individual to stay in a population and which are transferred to the next generation. The most commonly used replacement scheme is elitism, which enables the survival of some number of the best individuals at each generation. Hence, this scheme guarantees that these best individuals from previous generation are always present in a population.

In this study, elitism replacement scheme is employed to ensure preservation of the best chromosomes at each generation. The rest of individuals in each generation are selected randomly from the current population using crossover and mutation operators.

The replacement procedure can be summarized as follows:

- ***Replacement Procedure:***

calculate *elitist_value* based on pre-defined percent rate

loop until *elitist_value* is satisfied

select the best individuals to new population for the next generation

end

loop until the value of (*pop_size* - *elitist_value)* is met

select the individuals randomly among the current population and individuals who

underwent crossover and mutation

end

new_population for the next generation has been created

4.2.8 GA Search Termination

Termination criterion is the last decision point by which the genetic algorithm decides whether to continue searching the solution space or stop evolution. Hence, after each generation, it has been checked to see if it is time to terminate the algorithm.

As explained in earlier chapter, the search of solution space is terminated using various stopping conditions. In this study, the performance of genetic process is evaluated by calculating the maximum fitness and average fitness for each generation and this procedure continues for prespecified number of generations. When the number of generations created is equal to this prespecified number, the relative difference between maximum fitness and average fitness is checked. If the difference is less than 0.5 (i.e, commonly used threshold value in published literature), the genetic process ends. Otherwise, the algorithm proceeds with further generations.

CHAPTER FIVE
AN INDUSTRIAL CASE STUDY

This chapter presents an industrial case study carried out a local company operating in Izmir, Turkey. The objective is to improve the capacity of heat exchanger production line by implementing the proposed hybrid approach in two phases. In the first phase of the study, a detailed stochastic and dynamic simulation model of the line is developed to identify bottleneck machines. The objective is to use this information for initial population generation in the second phase of hybrid simulation-GA approach. In the second phase, hybrid simulation-GA approach is employed to decide how to allocate buffers to the machines so that throughput of the line can be maximized. The following section explains the application environment and the problem studied.

5.1 Application Environment and Problem Definition

The manufacturing company considered in this study was founded in Izmir. It is one of the largest European manufacturers of thermotechnology (heating units, etc.) sector. Moreover, this company participates in the areas of automotive and industrial technology, building technology and consumer goods in Turkey. The company adopts customer oriented manufacturing and commits themselves to producing high quality industrial products for domestic and foreign markets.

Since 1991, this manufacturing company has been conducting its operations at its facilities over 25.500 sqm. total production capacity. It produces different types of heat exchangers, copper pipe coils, atmospheric brulors and various copper pipes at this manufacturing area. Among these products, the production of heat exchangers is considerably high in comparison to others. Since thousands of component parts and various models of heat exchangers are produced in the heat exchanger production line, a great portion of the setup times and production times are spent in this line. That is why, capacity improvements in this line is expected to have a great effect on the performance of manufacturing system. Because of these reasons, this industrial case study has been carried out in the heat exchanger production line which is composed of pressing, forming, welding and test stations. As it is seen in Figure 5.1, at some stations parallel machines take place.

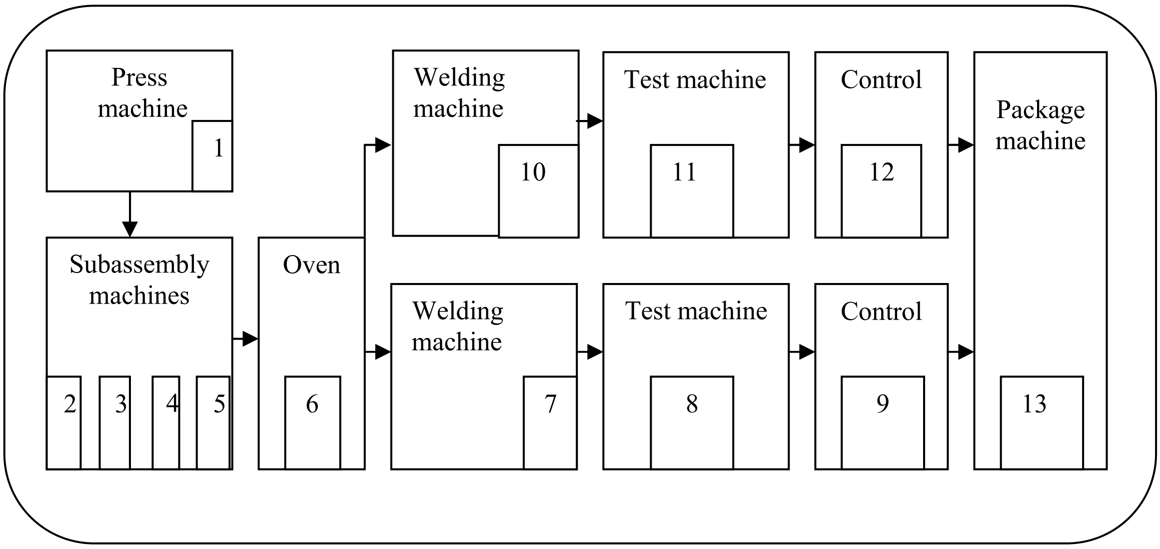


Figure 5.1 Production flow in the heat exchanger production line

In heat exchanger production line, seven types of heat exchangers are produced. In this study, these products are named as HE1, HE2, HE3, HE4, HE5, HE6 and HE7. As it is seen in Figure 5.2, these products are subject to seven processing operations which are pressing, subassembly, oven process, welding, testing-controlling and finally packaging.

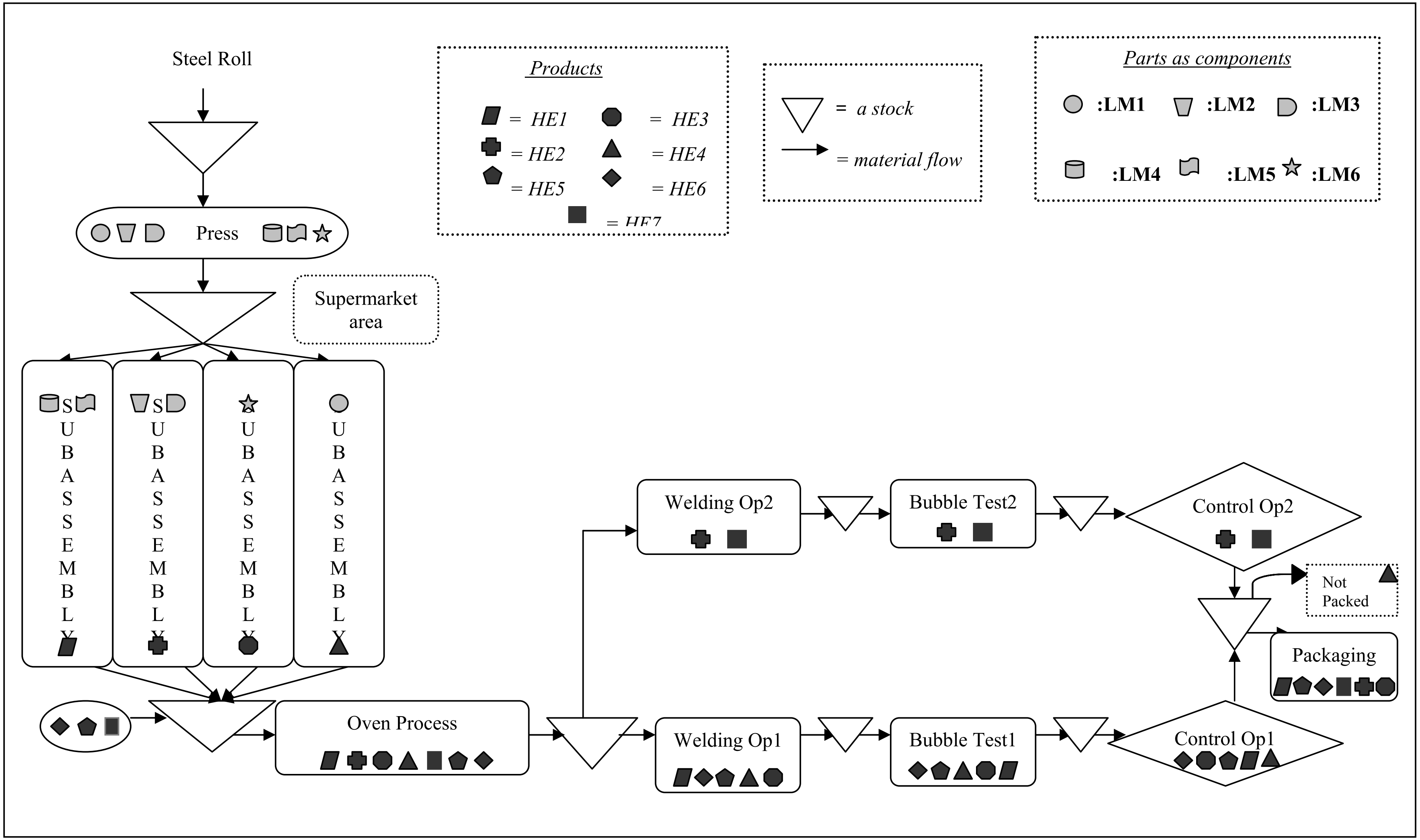


Figure 5.2 Description of heat exchanger process

It should be noted that this company is produce-to-order type. First, based on customer specifications, the steel rolls are smoothed and cut. Next, using different types of moulds for each type, the steel rolls are processed at the press station in batches. As a result of press operation, six types of parts which are named as LM1, LM2, LM3, LM4, LM5 and LM6 are produced (see Table 5.1 for process times) and as seen in Figure 5.1 these parts are temporarily stocked at the supermarket area before they are sent to the subassembly station.

Table 5.1 Process times at the press station

Parts	Process Times (in minutes)
LM1	0.0103
LM2	0.0138
LM3	0.0097
LM4	0.013
LM5	0.0217
LM6	0.013

Changing from one type of mould to another requires a set-up operation. To fit an appropriate distribution to the setup time records kept in the company, Input Analyzer of ARENA 10.0 has been employed, and input distributions given in Table 5.2 are estimated.

Table 5.2 Modeling set-up times at the pres station

	Setup Times (in minutes)					
Parts	LM1	LM2	LM3	LM4	LM5	LM6
LM1	----------	unif(41,47)	unif(27,33)	unif(32,38)	unif(42,48)	unif(37,43)
LM2	unif(27,33)	----------	unif(28,34)	unif(27,33)	unif(27,33)	unif(42,48)
LM3	unif(27,33)	unif(39,45)	----------	unif(32,38)	unif(42,48)	unif(27,33)
LM4	unif(32,38)	unif(35,41)	unif(32,38)	-----------	unif(35,41)	unif(35,41)
LM5	unif(27,33)	unif(32,38)	unif(34,40)	unif(27,33)	----------	unif(35,41)
LM6	unif(35,41)	unif(27,33)	unif(29,35)	unif(27,33)	unif(27,33)	-----------

Following the machine visitation sequences given in Table 5.3, these parts are eventually converted into four types of heat exchangers which are named as HE1, HE2, HE3 and HE4. Unlike these four products, the subassemblies for other three products (i.e., HE5, HE6 and HE7) are outsourced, and they are further processed in heat exchanger production line following the machine visitation sequences in Table 5.3.

It should be noted that like in the case of set-up times, input analysis of ARENA V10.0 has been employed to fit a distribution to process times kept in the company records, and input distributions given in the last column of Table 5.3 are estimated.

Table 5.3 Product routes and process times (in minutes) in each machine

	Product routing - Process times (in minutes)		
Product Type	**Total Number of operations**	**Machine in visitation sequence**	**Process time (in minutes)**
HE1	6	2	unif(15.285, 17.1683)
		6	norm(7.89,1.167)
		7	unif(8.167,8.252)
		8	norm(1.467,0.0333)
		9	unif(0.666,0.917)
		13	unif(1.25,1.417)
HE2	6	3	unif(15.017, 15.55)
		6	norm(7.373, 0.08)
		10	unif(2.75, 3.11)
		11	unif(2.25, 2.42)
		12	unif(1.67, 2.13)
		13	unif(1.25, 1.42)
HE3	6	4	unif(18.167, 16.583)
		6	norm(3.89, 1.08)
		7	unif(18.17, 18.33)
		8	unif(1.19, 1.31)
		9	unif(0.67, 1.25)
		13	unif(1.25, 1.42)

Continuation of Table 5.3

Product Type	Product routing - Process times (in minutes) Total Number of operations	Machine in visitation sequence	Process time (in minutes)
HE4	5	5	unif(22.067, 22.4)
		6	norm(6.89, 1.08)
		7	unif(9.33, 9.53)
		8	unif(1.03, 1.11)
		9	unif(0.73, 0.87)
HE5	5	6	norm(5.89, 1.17)
		7	unif(22.17, 22.25)
		8	norm(2.47, 1.73)
		9	unif(3.67, 4.32)
		13	unif(2.25, 2.42)
HE6	5	6	norm(1.37, 0.08)
		7	unif(14.43, 14.63)
		8	unif(3.19, 4.91)
		9	unif(3.67, 4.25)
		13	unif(2.25, 2.42)
HE7	5	6	norm(5.89, 1.08)
		10	unif(4.75, 6.35)
		11	unif(4.25, 5.42)
		12	unif(5.67, 6.33)
		13	unif(2.25, 2.42)

5.2 Phase I: Bottleneck Identification Using Simulation Modelling

This section presents a simulation-based procedure to identify bottleneck machines in heat exchanger production line. To develop a simulation model of the line, first a conceptual model is developed, next it is coded in ARENA V10.0, and before the model is run to identify the bottleneck machines, it is verified and validated. These steps are given in detail in the following sections.

5.2.1 Model Conceptualization

Model conceptualization is an important part in a simulation study. A conceptual model reflects the control structure of a simulation model. The logic followed to develop the simulation model of the heat exchanger production line is given in figures 5.3 and 5.4. Figure 5.3 represents the logic followed in simulating the production of parts LM1, LM2, LM3, LM4, LM5, LM6. Likewise, Figure 5.4 exemplifies the logic followed in simulating the processing of both in-house produced and also outsourced parts.

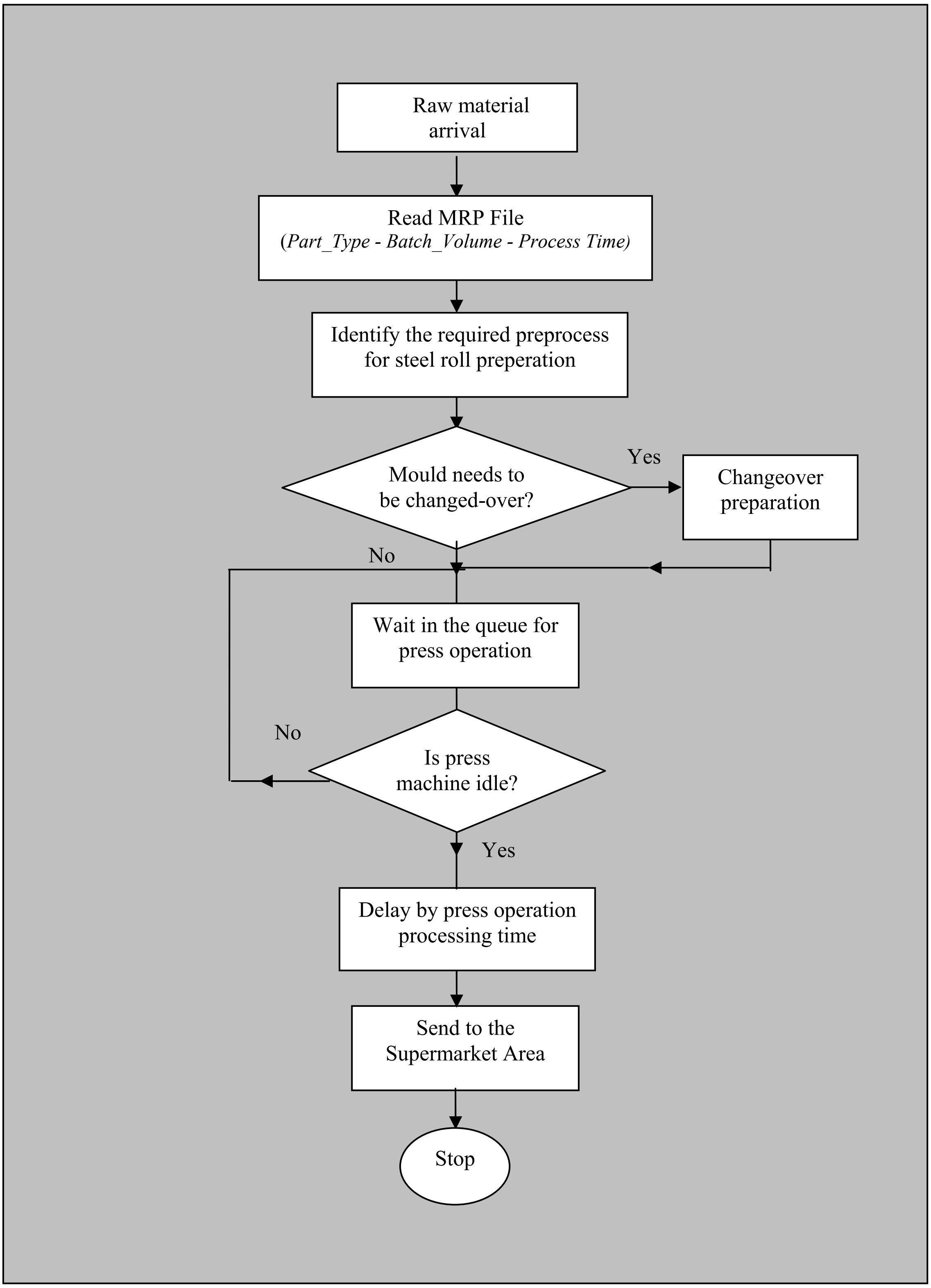


Figure 5.3 The control logic of simulation model for part production

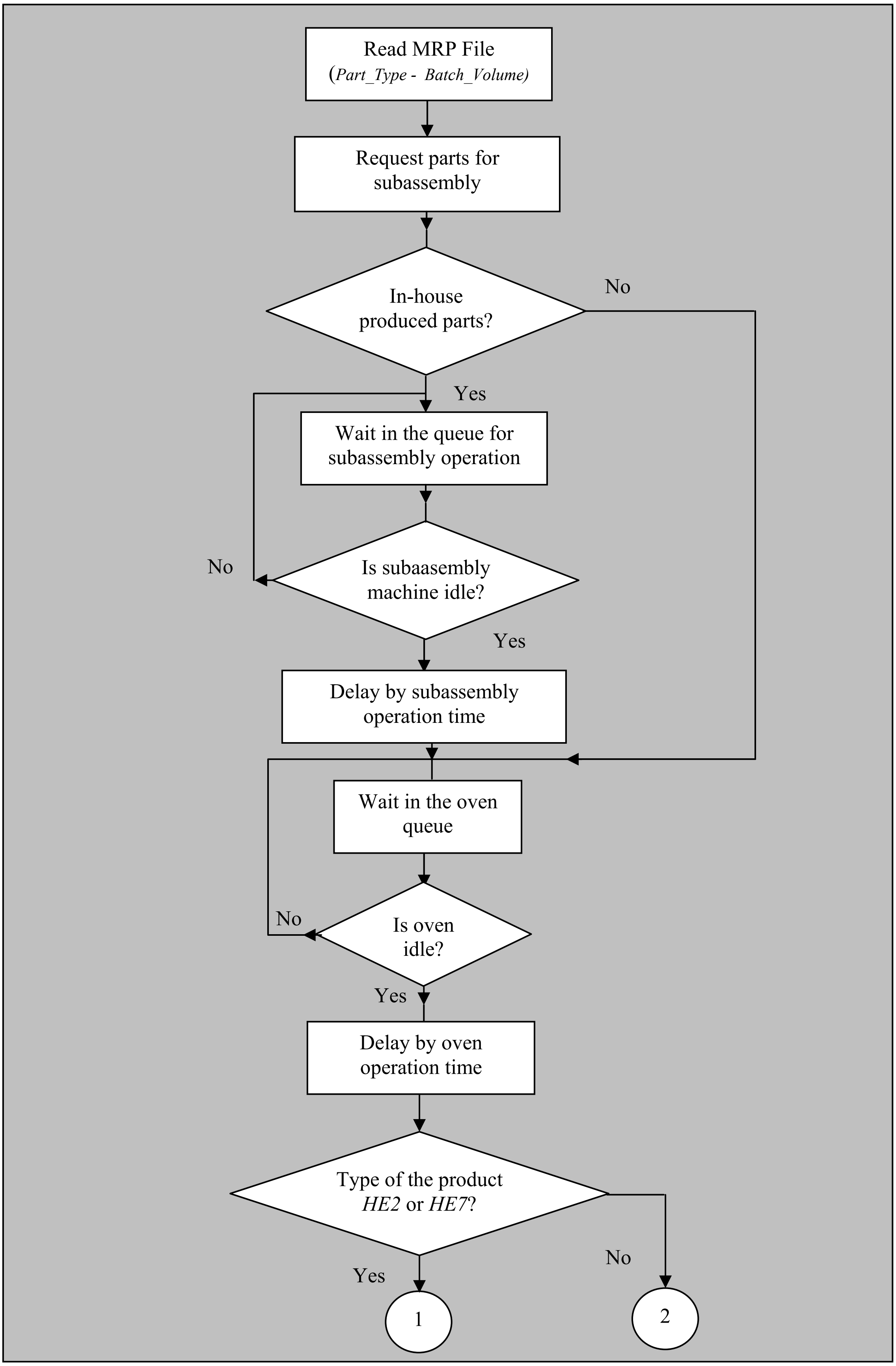
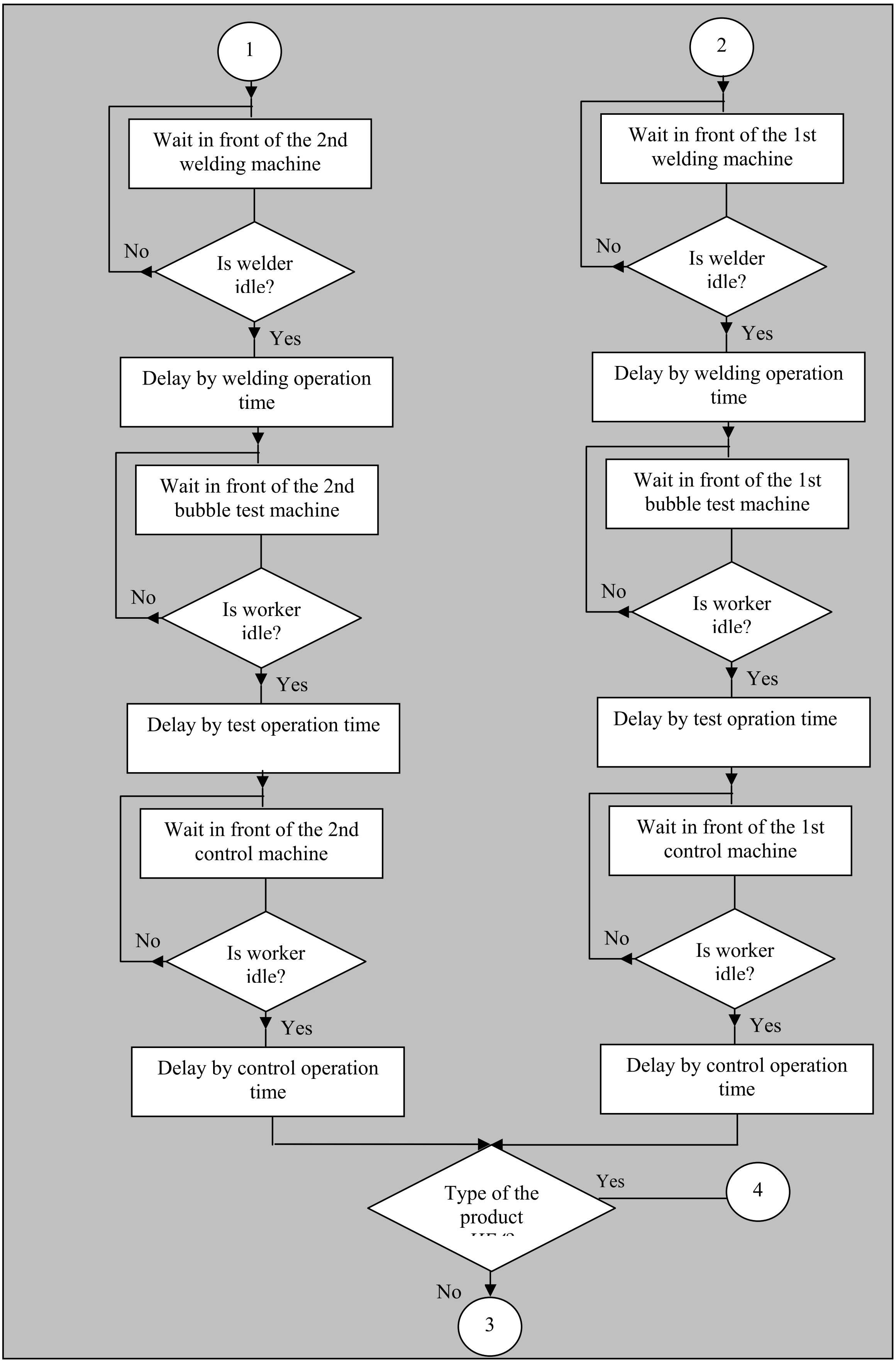
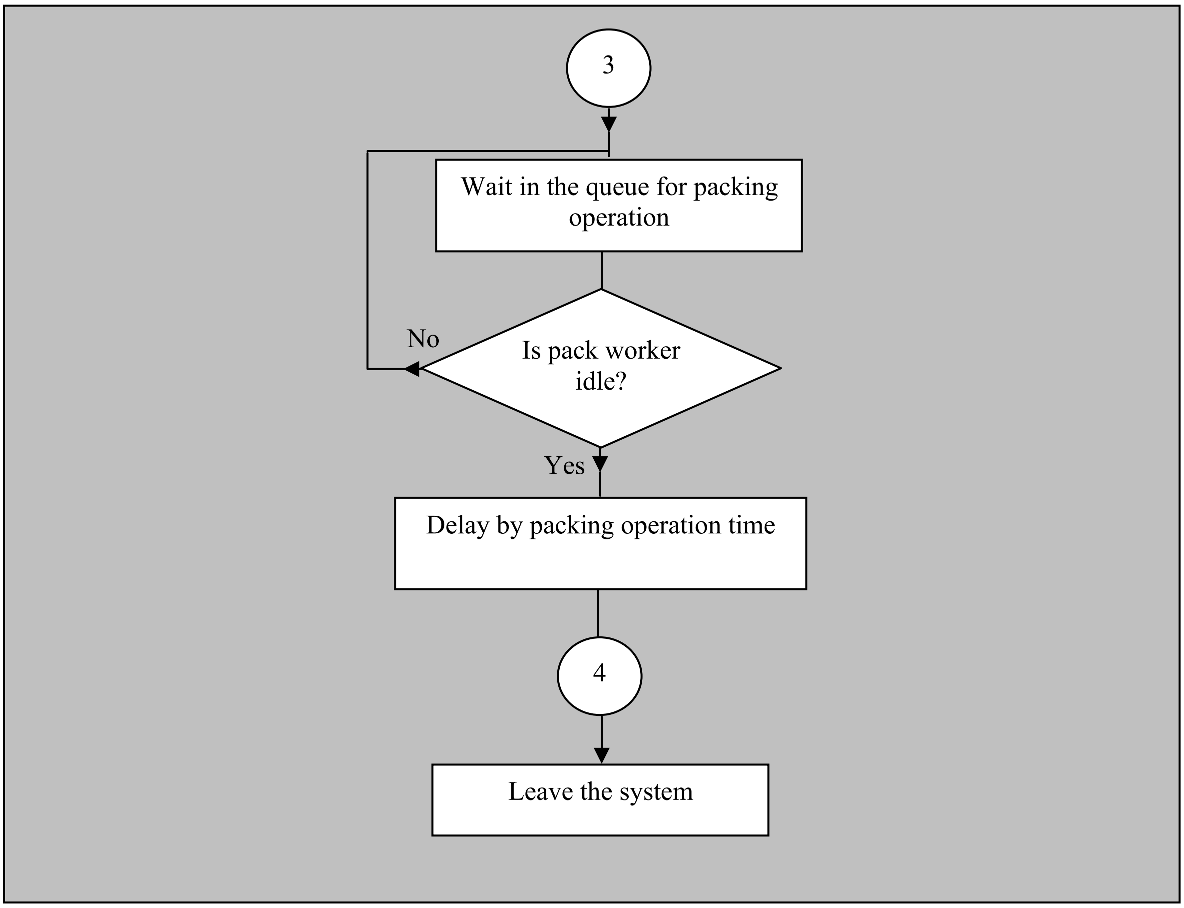


Figure 5.4 The control logic of simulation model for main production system



Continuation of Figure 5.4



Continuation of Figure 5.4

5.2.2 Simulation Model Development

Discrete-event simulation models are generally used to imitate the operations of systems over time. Due to their ability to evalute the performance of systems realistically they are used for decision support in many areas including facility layout design, supply chain management, production planning, etc. In this M.Sc. study, discrete-event simulation modeling has been employed for bottleneck identification in heat exchanger production line. The detailed simulation model of this line has been developed under the following assumptions using simulation language Arena V10.0:

- The plant operates eight hours per day and five days in a week including 50 and 15 minutes breaks for meal and cleaning, respectively.
- Raw materials are always available and there is always space for the finished products,
- The capacity of storage for work-in-process is not limited.
- Each workstation is operated by one operator, except for Welding_Op1 station,
- The processing times at all stations, except for press station which is automated are stochastic,
- Setup times are stochastic,
- Travel times between the stations are negligible,
- Machines are subject to breakdown. Input Analyzer of ARENA V10.0 has been employed to fit appropriate distributions to the records of failure and repair rates kept in the company. (see Table 5.4)

Table 5.4 Modeling machine failures and repair times (in minutes)

	Time between failures (min)	Repair time (min)
Press_machine	expo(140)	expo(20)
Subassembly_machine1	expo(70)	expo(25)
Subassembly_machine2	expo(100)	expo(30)
Subassembly_machine3	expo(55)	expo(36)
Subassembly_machine4	expo(145)	expo(48)
Oven	expo(240)	expo(60)
Welding_machine1	expo(40)	expo(33)
Welding_machine2	expo(150)	expo(20)
Test_machine1	expo(90)	expo(28)
Test_machine2	expo(190)	expo(10)
Control_machine1	expo(200)	expo(115)
Control_machine2	expo(150)	expo(120)
Package_machine	expo(90)	expo(28)

- Since the company operates on the basis of make to order and starts production with no work-in-process, the warm-up period has not been considered.

In simulation modeling, to determine the correctness of the model, two essential functions should be carried out. These two functions are verification and validation. Next section explains verification and validation of the simulation model developed.

5.2.3 Verification and Validation

Model verification makes sure that the conceptual model is correctly converted into a computer simulation model. Unlike model verification, validation of model is usually defined to measure the accuracy of model within its applicability of system. Computerized model should be in a satisfactory range of accuracy consistent with the intended application of the model. Hence, verification and validation are of crucial importance in the development of computer simulations.

In this study, the verification has been done by developing the model in a modular manner, using interactive debuggers, and manually checking the results. Using Arena's build-in Trace element, it is possible to observe whether a product moves on its route harmonized to the system logic or not. As a result, all the movements of products through the model have been observed step by step using the trace element.

Moreover, the operational model has been animated to verify the actual system until no logical errors related to the flow of products are observed. A screenshot of animation environment is represented in Figure 5.5.

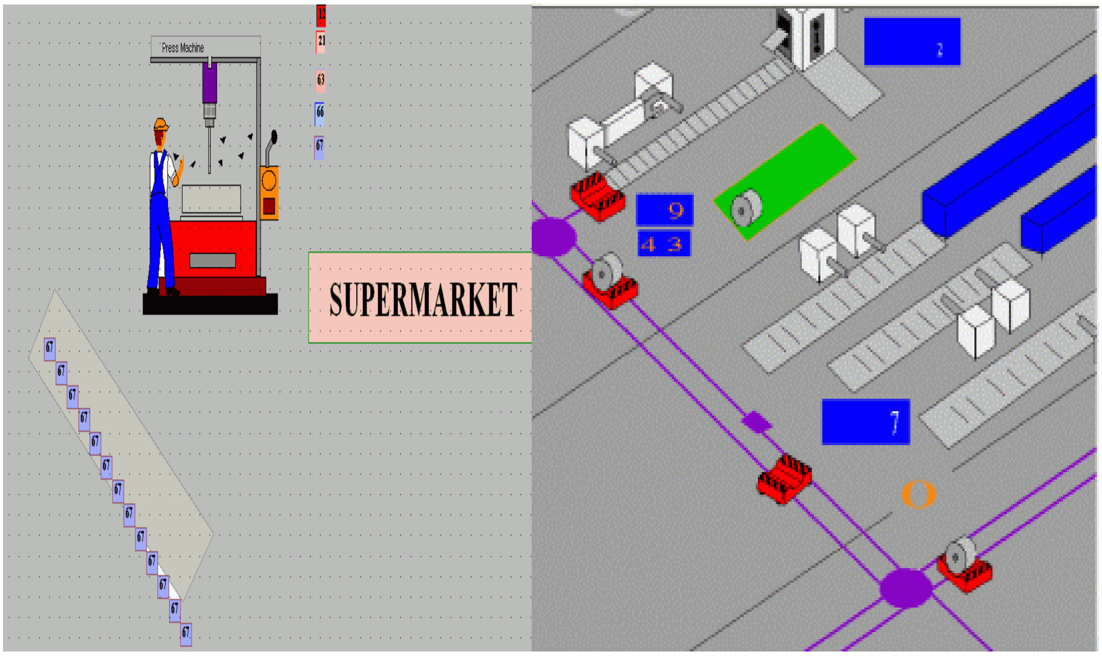
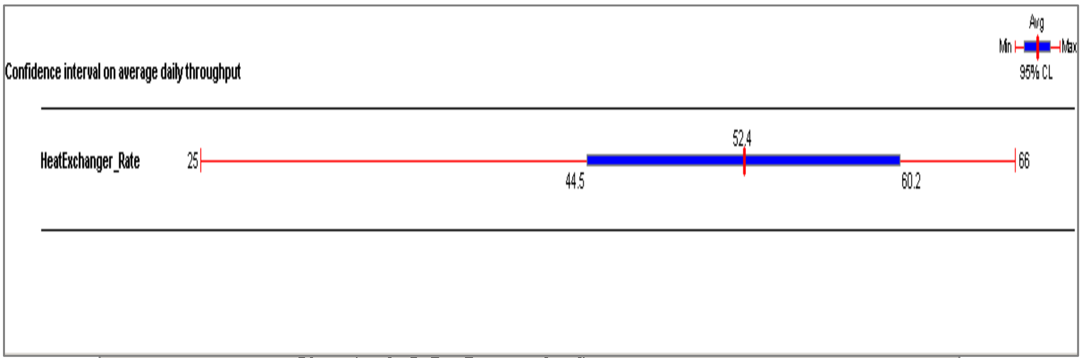


Figure 5.5 Animation view of the whole system

Furthermore, the accuracy of the simulation model developed in this study has been validated by using hypothesis testing and confidence interval testing at a 95% level of confidence. Hypothesis tests are conducted to compare the model output with the observed output of the actual system. Similarly, confidence interval testing is used to evaluate whether the simulation and the real system performance measures are close enough. As a result of a field study at company site, it has been noted that at average, 50 heat exchangers are produced per day. To compare this observed value, with the simulation output, the simulation model of the heat exchanger production line has been replicated 10 times, and the confidence interval at 95% level of confidence has been estimated. As shown in Figure 5.6, since this confidence interval covers the observed daily production, we might state that the simulation model truly represents the real system at 95% level of confidence.



Classical C.I. Intervals Summary
Confidence interval on average daily throughput

IDENTIFIER	AVERAGE	STANDARD DEVIATION	0.950 C.I. HALF-WIDTH	MINIMUM VALUE	MAXIMUM VALUE
HeatExchanger_Rate	52.4	11.7	7.88	25	66

Figure 5.6 Average daily throughput at 95% confidence interval

5.2.4 Bottleneck Identification

Having verified and validated the simulation model, next we carried out various experimental studies to identify the bottleneck machines. For this purpose, we run the simulation model 10 times and recorded average machine utilizations and average number of parts in each machine queue. As it is seen in Figures 5.7 and 5.8, the machines 4, 5, 6 and 7 (i.e. subassemly machine_3, subassemly machine_4, oven, and welding_op1) having highest utilizations and highest number of parts waiting to be processed are identified as bottleneck machines.

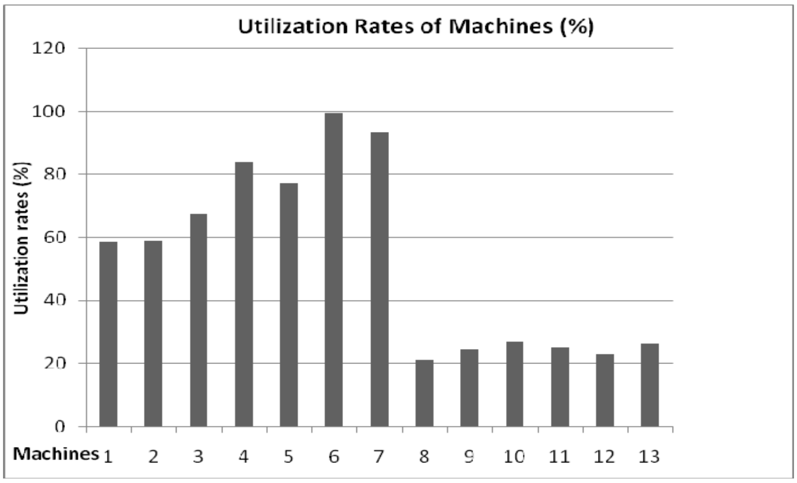


Figure 5.7 Utilization rates of machines

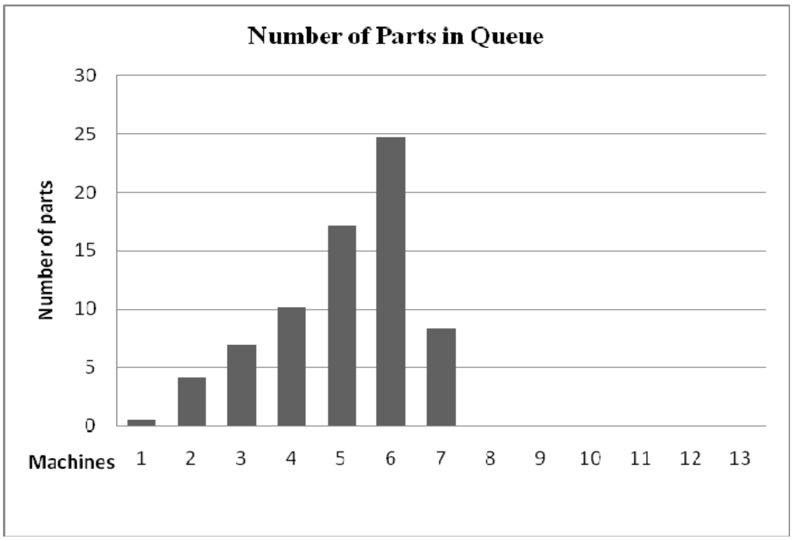


Figure 5.8 Number of parts in queue

The next section explains how the capacity of this line can be improved by using the proposed hybrid approach. It should be noted that in the following phase the information about potential bottleneck stations is taken into consideration in generating the initial population of GA. The objective is to allocate buffers as effectively as possible both with respect to solution time and also solution quality.

5.3 Phase II: Capacity Improvement by Employing the Proposed Hybrid Approach

In this phase, we employed our hybrid simulation-GA approach to make a decision on how to allocate buffers to the machines in the line so that the throughput of the line can be improved.

As it is seen in Figure 5.9, the proposed hybrid approach is based on integration of a genetic algorithm module with a simulation module in a closed loop configuration. It should be noted that as explained in section four the proposed genetic algorithm is adapted to the intrinsic features of the production line studied. Through this integration the genetic algorithm module suggests a buffer configuration at each iteration. Next, this buffer configuration is used as an input in the simulation model of heat exchanger production line. The other set of data provided as input data to the simulation model include daily production plans for each part and product type, product routes, machine failure and repair rates, and machine processing and setup times. Using all these input data, the simulation model evaluates the fitness of a given buffer configuration.

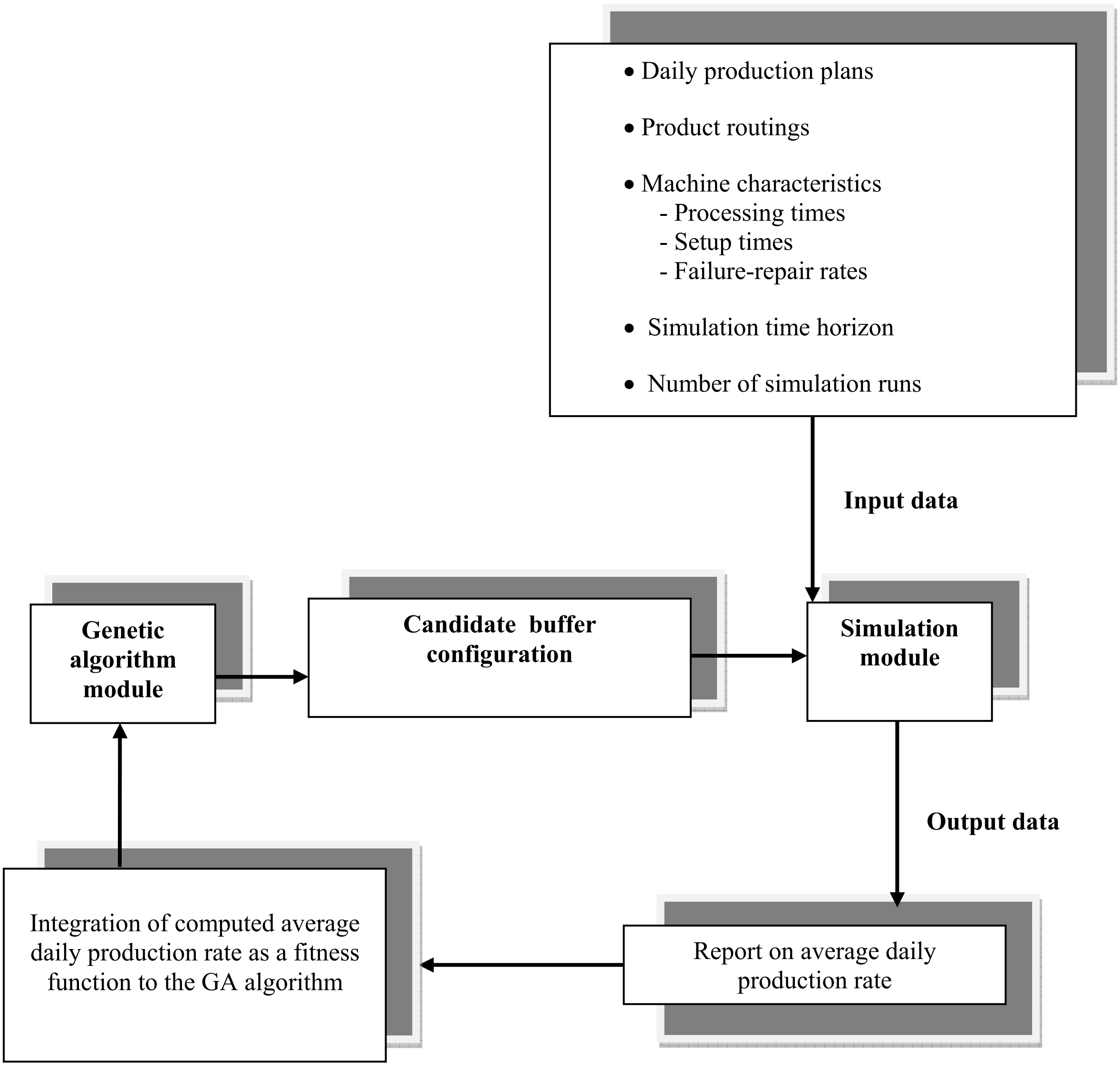


Figure 5.9 Structure of the proposed hybrid approach

Next section presents the computational experiments done for the efficient GA parameter setting.

5.3.1 Identifying Efficient GA Control Parameters

Genetic algorithms have several parameters and any combination of these parameters has different impacts on the performance on GA. For this reason, identification of the efficient GA parameters is very important to the accuracy of solution and convergence speed of GA application.

In this study, by fixing the total number of chromosomes at 1200, the experimental design given in Table 5.5 was employed to identify the effects of different control parameters (i.e. the combination of the population size and number of generations (P/G), the crossover rate (R_c), the mutation rate (R_m) and selection percent for elitism (S_e).

Table 5.5 Experimental factors

Factors	**Levels**	
Population size / Generation number *(P/G)*	20/60	30/40
Crossover rate (R_c)	0.60	0.80
Mutation rate (R_m)	0.028	0.033
Selection percent for elitism (S_e)	25%	50%

As it is seen in Table 5.6, there are four distinct control parameters and hence, 2^4 factorial design is employed to search for efficient levels of these GA control parameters. Five independent runs were carried out at each design point leading to 80 runs and ANOVA was used to determine the statistical significance of each effect on fitness. The fitness function involves estimation of average daily throughput per minute using the simulation model of the heat exchanger production line.

Table 5.6 2^4 full factorial experimental layout

Experiment no	Population size / Generation number P_S/G	Crossover rate, R_c	Mutation rate, R_m	Selection percent for elitism S_e
1	20/60	0.60	0.033	25%
2	20/60	0.60	0.033	50%
3	20/60	0.60	0.028	25%
4	20/60	0.60	0.028	50%
5	20/60	0.80	0.033	25%
6	20/60	0.80	0.033	50%
7	20/60	0.80	0.028	25%
8	20/60	0.80	0.028	50%
9	30/40	0.60	0.033	25%
10	30/40	0.60	0.033	50%
11	30/40	0.60	0.028	25%
12	30/40	0.60	0.028	50%
13	30/40	0.80	0.033	25%
14	30/40	0.80	0.033	50%
15	30/40	0.80	0.028	25%
16	30/40	0.80	0.028	50%

As given in earlier sections, the heat exchanger production line consists of thirteen workstations. In front of twelve stations, buffers are allocated using proposed hybrid approach. Hence, this buffer allocation problem involves twelve decision variables denoting how to allocate the existing buffer capacity to these twelve stations. The genetic algorithm has been coded in Matlab V7.6 language and the results of ANOVA are summarized in Table 5.7. As seen in Table 5.7, only population size/generation number, *P/G* is found to be statistically significant factor as a result of this factorial experiment.

Table 5.7 ANOVA results

Source of variation	df	F_{calc}	Prob(F>F_{calc})
Within + residual	15	4.614	0.000
P/G	1	62.136	0.000 *
R_c	1	0.505	0.205
R_m	1	0.174	0.822
S_e	1	0.194	0.539
$Ps/G * R_c$	1	1.187	0.768
$Ps/G * R_m$	1	0.004	0.717
$Ps/G * S_e$	1	0.000	0.924
$R_c * R_m$	1	0.078	0.662
$R_c * S_e$	1	1.009	0.263
$R_m * S_e$	1	0.151	0.768
$Ps/G * R_c * R_m$	1	3.218	0.075
$Ps/G * R_c * S_e$	1	0.392	0.522
$Ps/G * R_m * S_e$	1	0.020	0.779
$R_c * R_m * S_e$	1	0.019	0.556
$Ps/G * R_c * R_m * S_e$	1	0.119	0.495

A scatter plot of responses for each run (i.e., average daily throughput) with respect to all factors studied is given in Figures 5.10 through 5.13. The scatter plot in Figure 5.10 suggests that the runs involving a population of 30 with 40 generations achieve the higher throughput rate with the minimum spread. Increasing the population size enlarges the search space and apparently more individuals are created. Hence, probability of reaching better solutions increases.

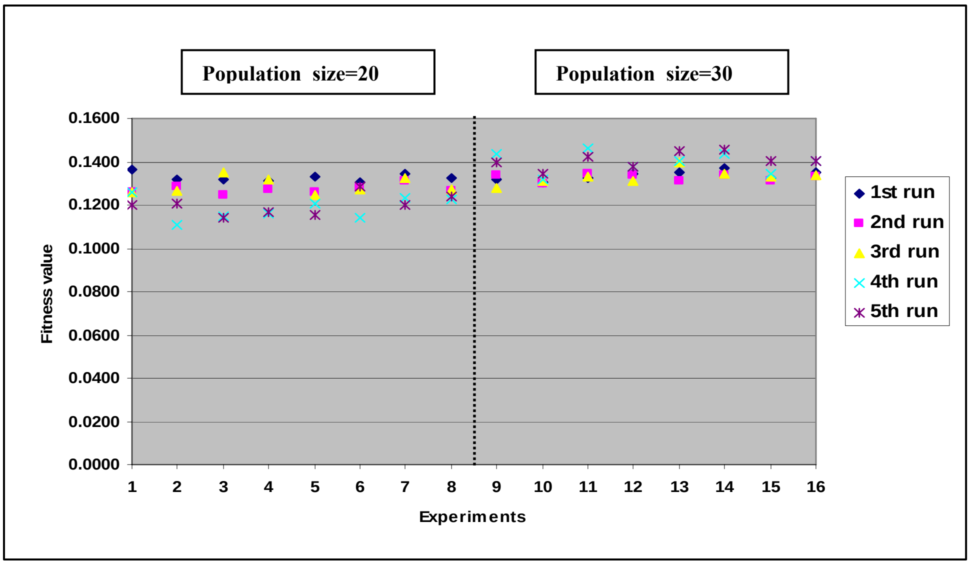


Figure 5.10 Analysis of different population sizes

As it is seen in Figure 5.11 and also supported by the findings of ANOVA analysis, it appears that different levels of crossover rate has no discernible effect on fitness value. It should be noted that a similar behaviour is observed with mutation probability rates and also with selection rates for elitism replacement scheme (see Figures 5.12 and 5.13).

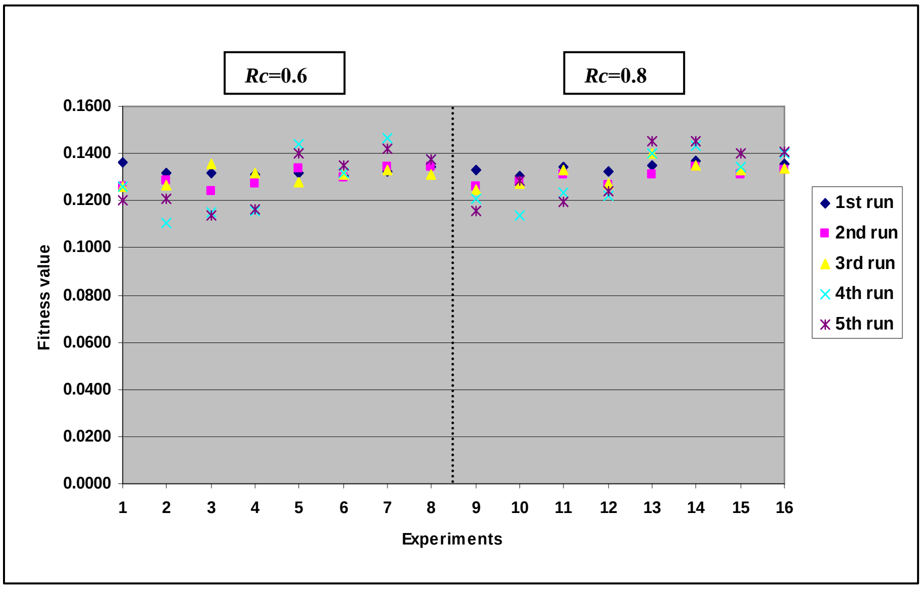


Figure 5.11 Analysis of different crossover probability rates

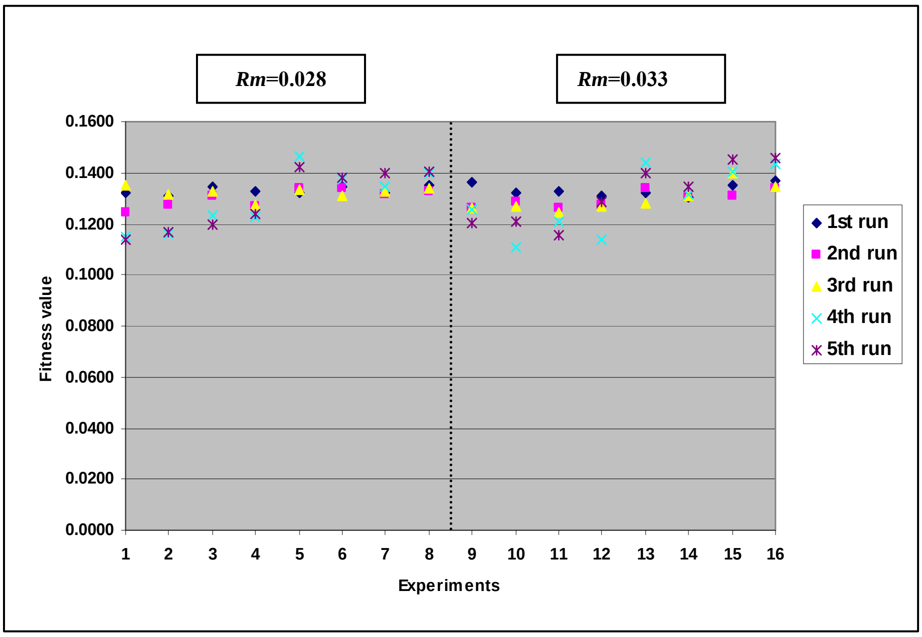


Figure 5.12 Analysis of different mutation probability rates

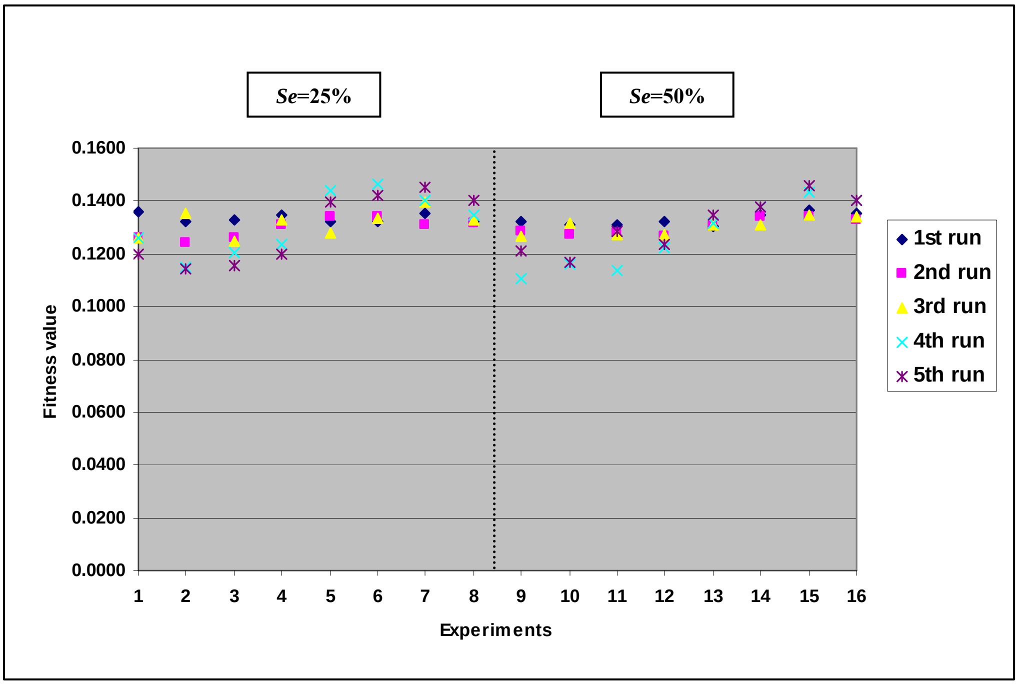


Figure 5.13 Analysis of different selection percent rates for elitism replacement scheme

In summary, except for indicating population size/generation number as a statistically significant factor the results of ANOVA did not help much to give a direction for significant control parameters. Likewise, the scatter plot of responses for each run with respect to each GA parameter led to a consistent pattern only for population size/generation number. Hence, to determine efficient GA parameter settings, we closely examined each scatter plot given above and decided that the GA control parameters, P/G, R_c, R_m, and percent of elitism for replacement strategy with values 30/40, 0.80, 0.033 and 50%, respectively are giving higher fitness values in majority of runs. So the values of these four GA parameters are set accordingly. Next section explains the steps involved in implementation of the proposed hybrid GA-based simulation approach.

5.3.2 Structure of the Proposed Hybrid Approach

In general, the problem of optimal buffer allocation has been considered with respect to different optimality criteria. The most commonly used criterion is the average steady-state production rate, i.e. the average number of parts produced in time unit as in the case of this study. Considering the available floor space in the heat exchanger production line to accommodate buffers, the upper limit of the total number of buffers to allocate is set to 70. It should be noted that the total buffer capacity, K, is considered as system constraint do determine the feasibility of the solutions obtained. Based on this limitation, for total number of buffers, K= 70 and the number of machines, n=13, the whole search space has a volume of C_{81}^{11} = 2.371.707.585 over 2 billion solutions. In this case, the search for a globally optimum solution in such a large search space is very difficult. Hence, using metaheuristic methods to solve such problems becomes inevitable.

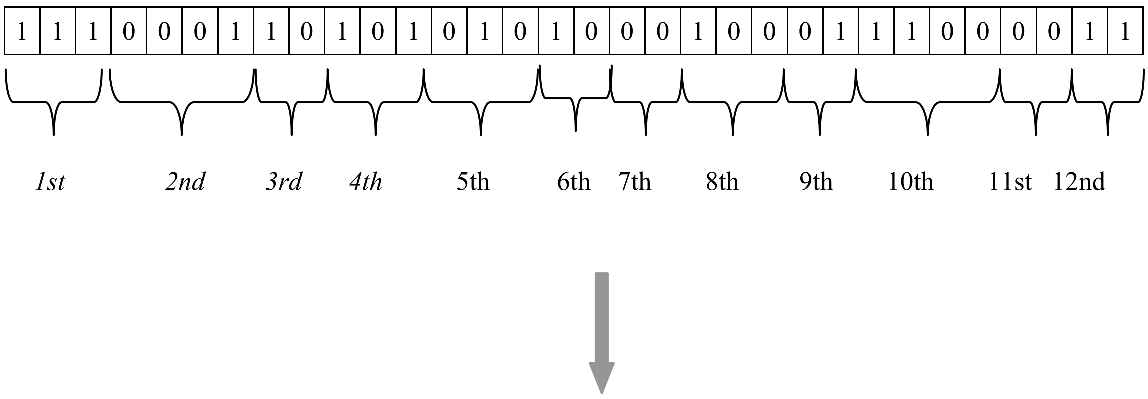
In this study, the genetic algorithm has been implemented in Matlab V7.6 language. The simulation model of the heat exchanger production line which has been developed using Arena 10.0 simulation language. The simulation model has been run for 10 times to obtain the fitness value of the candidate solution, i.e. production rate, and this value is communicated to the genetic search module in an iterative manner.

The details of this hybridized GA-based simulation approach is summarized step by step as follows:

Step 0: Initialization

As a result of experimental studies for efficient GA parameter setting, the population size is set to 30 and the genetic search is terminated after 40 generations. Initially, population of buffer configurations is empty and generation number, k, is set to 0. Since buffers are allocated to twelve machines in the production line, a chromosome is composed of twelve unique parts. So, each chromosome represents a possible configuration of the buffer sizes as a decision variable. The integer value of each decision variable is represented as a binary string. The length of the string depends on the upper bound of total number of buffers to allocate. A chromosome alternative which is named as *chrom1* in Figure 5.14 represents that 7, 1, 2, 5, 2, 2, 0, 4, 1, 12, 0 and 3 units of buffer will be allocated to these twelve machines.

chrom1:



Buffer_sizes = [7 1 2 5 2 2 0 4 1 12 0 3]

Figure 5.14 Binary coding representation of alternative chromosome

Step 1: Initial population generation

At this step, an initial population is generated in two ways: randomly and heuristically. Meanwhile, the feasibility of each individual is checked by considering system constraints such as the maximum capacity of each buffer location and also the total buffer capacity.

Step 2: Evaluation of initial population

Once all chromosomes of the current solution are created, the fitness of each chromosome is evaluated using the simulation module. Then, the generation iteration number, k, is set to 1.

Step 3: Selection of parents

Based on the results of comparative experimental study explained in previous chapter, the roulette wheel selection scheme, which scales the fitness values of the members within the population so that the sum of the rescaled fitness values equals to 1, has been used for the reproduction process in the algorithm.

Step 4: Crossover and mutation

Using 0.80 and 0.033 as values of the crossover and mutation parameters, respectively, new populations are created by crossover and mutation, and individuals are checked for feasibiliy. Infeasible individuals are excluded from the population and the new parents are selected for crossover. After selection, the crossover operation has been repeated and recombination has been carried out. Unlike the crossover operation, individuals which violate the system constraints have been directly eliminated after mutation operation without any repetition.

Step 5: Evaluation of offsprings

The fitness of the newly formed offsprings is evaluated based on daily production rate using the simulation module.

Step 6: Replacement of the individuals to next generation

To survive the individuals to the next generation, elitism replacement scheme, which provides 50% of the best individuals in the new population is used and the rest of other individuals to survive to the next generation is selected randomly among the current population and the new individuals are formed by crossover and mutation. After all, the generation iteration number, k, is set to k+1 and the genetic search continues by Step 3 until the termination criterion is satisfied.

Step 7: Search Termination

As mentioned before, the algorithm is terminated after a specified number of generations by taking into consideration the maximum fitness and average fitness. If there is a significant difference between the maximum fitness and the average fitness, the algorithm proceeds with further generations.

Next section presents the implementation of the proposed hybrid GA-based simulation approach to a real-world industrial problem. Namely, this industrial case study illustrates the benefits (i.e., capacity improvement by buffer allocation to the machines in the line) that can be obtained by implementation of this hybrid approach.

5.3.3 Implementation of the Hybrid Approach

In this section, three alternative genetic searches have been carried out to make a decision on optimum buffer allocation for capacity improvement in the heat exchanger production line. These alternative genetic searches are summarized as follows:

- *Search 1:* Using random initial population generation,
- *Search 2:* Using heuristically initial population generation,
- *Search 3:* Using random initial population generation considering bottleneck machines.

The implementations of these alternative genetic searches are explained in detail in the following subsections.

5.3.3.1 Hybrid GA-Based Simulation Approach Using Random Initialization Scheme

This section presents the implementation of the hybrid approach using random initialization scheme for buffer allocation problem. This scheme involves allocating the total buffer capacity through all the machines in the production line randomly. A representation of buffer areas in the heat exchanger production line is given in Figure 5.15.

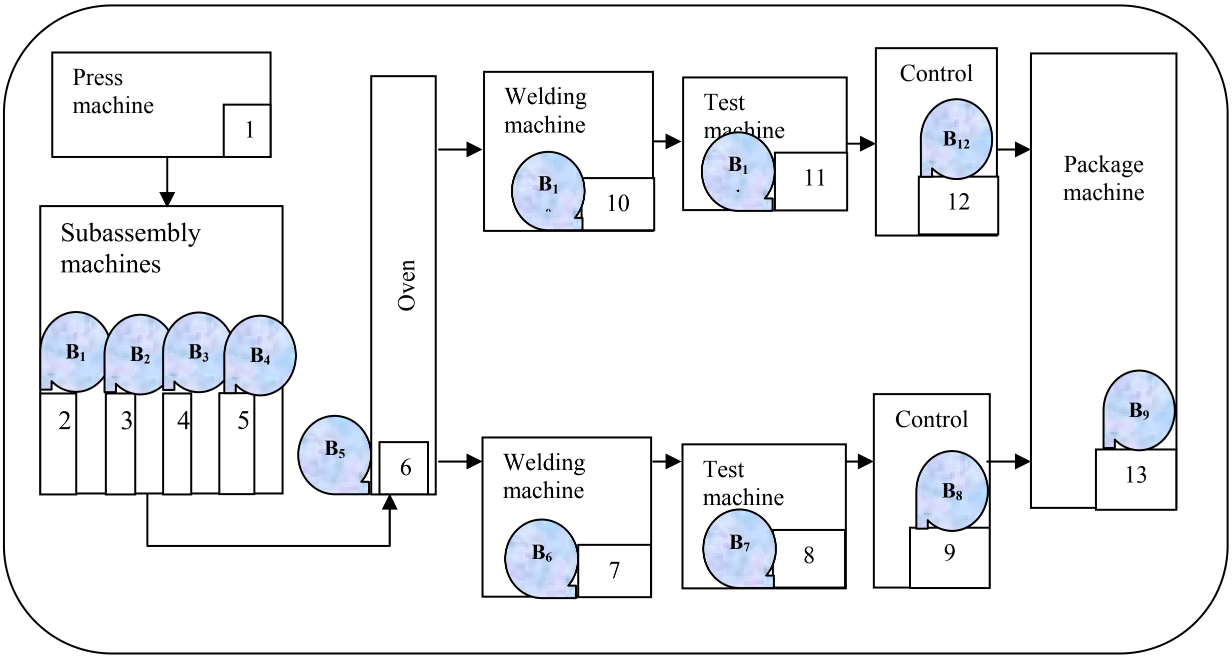


Figure 5.15 Buffer areas in front of all machines in the production line

The mathematical formulation of this buffer allocation problem can be depicted as follows:

Find $B = (B_1, B_2, B_3, \ldots, B_{12})$ so as to

max $P(B)$

subject to $\sum_{i=1}^{12} Bi \leq 70$

$B_1 \leq 5$, $B_2 \leq 7$, $B_3 \leq 15$, $B_4 \leq 20$, $B_5 \leq 25$, $B_6 \leq 15$,

$B_7 \leq 3$, $B_8 \leq 3$, $B_9 \leq 3$, $B_{10} \leq 3$, $B_{11} \leq 3$, $B_{12} \leq 3$

Bi nonnegative integers (i = 0,1,2, …, 12)

This formulation expresses the maximization of the throughput rate $P(B)$, under a given amount of buffers. The total buffer space available in the system which has to be allocated among twelve buffer locations is considered to be at most 70. Moreover, it should be noted that an upper bound is considered for each buffer location.

The results of implementing the proposed alternative approach using random initial population generation to solve the buffer allocation problem in heat exchanger production line are given in Table 5.8. The columns in this table show the genaration number, the buffer size configurations, maximum production rate, average production rate and the relative difference which is found as by dividing (maximum fitness – average fitness) to maximum fitness.

Table 5.8 Results of the genetic search

Gen. No	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	Max Fitness / Production Rate	Average Fitness / Production Rate	Relative Difference	(*)
0	0	0	0	0	15	5	3	3	2	0	1	0	0.0940	0.0496	0.472	
1	3	4	0	0	15	5	3	3	2	0	1	2	0.1325	0.0860	0.351	
2	4	0	0	0	15	8	2	2	2	0	1	2	0.1639	0.1145	0.301	
3	3	4	1	16	23	9	3	3	2	2	1	2	0.1494	0.1145	0.234	
4	1	4	4	0	23	9	3	3	3	0	1	2	0.1470	0.1157	0.213	
5	1	4	0	1	3	9	3	3	3	0	1	2	0.1590	0.1169	0.265	
6	3	5	8	16	15	10	2	2	3	0	3	2	0.1639	0.1248	0.238	
7	1	6	2	1	3	9	3	3	3	0	1	2	0.1687	0.1325	0.214	
8	1	4	8	9	3	9	3	3	3	1	1	2	0.1687	0.1325	0.214	
9	1	6	2	1	3	9	3	3	3	0	1	2	0.1687	0.1325	0.214	
10	3	4	0	9	10	9	3	3	2	3	3	2	0.1687	0.1342	0.204	
11	3	4	0	9	10	9	3	3	2	3	3	2	0.1687	0.1342	0.204	
12	1	4	0	9	10	9	3	3	3	1	1	2	0.1711	0.1349	0.211	
13	1	4	0	9	10	9	3	3	3	1	1	2	0.1711	0.1349	0.211	
14	1	5	0	9	10	9	2	3	3	1	1	2	0.1711	0.1349	0.211	
15	1	6	0	9	10	9	3	3	3	0	1	2	0.1711	0.1349	0.211	
16	1	5	0	9	10	9	2	3	3	1	1	2	0.1711	0.1349	0.211	
17	3	6	0	9	10	9	3	3	3	1	1	2	0.1711	0.1349	0.211	
18	1	6	0	9	10	9	3	3	3	1	1	2	0.1711	0.1349	0.211	
19	1	6	0	9	10	9	3	3	3	1	1	2	0.1711	0.1349	0.211	
20	1	5	0	9	10	9	2	3	3	1	1	2	0.1711	0.1349	0.211	
21	1	4	0	9	10	9	2	3	3	1	1	2	0.1711	0.1349	0.211	
22	1	4	0	9	10	9	3	3	3	1	1	2	0.1711	0.1349	0.211	
23	1	6	0	9	10	9	2	3	3	0	1	2	0.1711	0.1349	0.211	
24	1	5	0	9	10	9	2	3	3	1	1	2	0.1711	0.1349	0.211	
25	5	5	0	13	10	9	2	3	3	1	1	2	0.1711	0.1349	0.211	
26	1	6	0	9	10	9	2	3	3	1	1	3	0.1711	0.1349	0.211	
27	4	5	0	11	10	9	2	3	3	1	1	2	0.1711	0.1349	0.211	
28	1	4	0	9	10	9	2	3	3	1	1	2	0.1711	0.1349	0.211	
29	1	6	0	9	10	9	3	3	3	1	1	2	0.1711	0.1349	0.211	
30	3	6	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	*
31	3	6	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
32	1	4	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
33	1	4	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
34	1	6	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
35	1	4	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
36	1	4	0	8	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
37	1	6	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
38	1	4	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
39	1	4	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	
40	1	4	0	9	18	10	3	3	3	1	1	2	0.1687	0.1357	0.196	

According to these results, it has been noted that the best average throughput rate is found after thirty iterations as a rate of 0.1357 parts per unit time. During the genetic search, the result implies approximately 63% increase in average daily production rate (i.e., from the initial value of approximately 21 to 57). Between the 30th and 40th generations, almost all buffer configurations result in the same average daily production rate with different buffer sizes in configurations. Among these configurations, the buffer configuration of 1, 4, 0, 8, 18, 10, 3, 3, 3, 1, 1, 2 is identified as optimum buffer allocation configuration because of the minimum total buffer capacity usage, 54 (see Table 5.9).

Table 5.9 Buffer configurations that give the best average production rate

Buffer Configuration												Total Buffer Size	Average Production Rate
3	6	0	9	18	10	3	3	3	1	1	2	59	0.1357
1	4	0	9	18	10	3	3	3	1	1	2	55	0.1357
1	6	0	9	18	10	3	3	3	1	1	2	57	0.1357
1	**4**	**0**	**8**	**18**	**10**	**3**	**3**	**3**	**1**	**1**	**2**	**54**	**0.1357**

Furthermore, as seen in Figure 5.16, the relative difference decreases during the iterative search and it converges at a value, 0.196 and after thirty iterations.

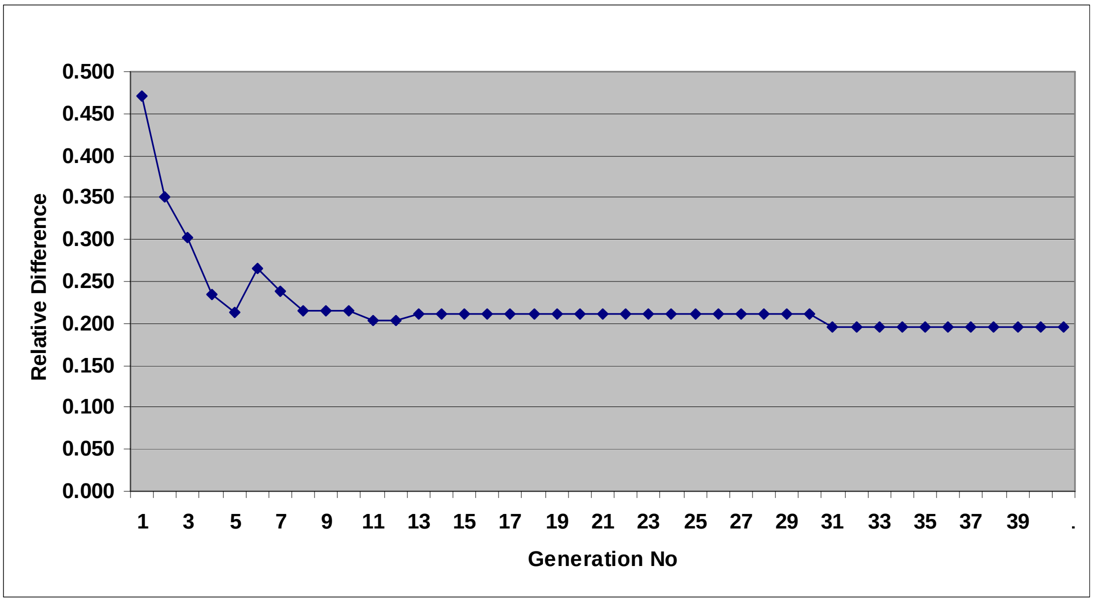


Figure 5.16 Convergence rate of the proposed approach

Having allocated 1, 4, 0, 8, 18, 10, 3, 3, 3, 1, 1, 2 number of buffers to the machines in the heat exchanger production line, the simulation model of the system is run again with specified buffer sizes The proposed approach results in 0.1357/minute which implies 56.3 exchangers per shift, so approximately 12.6% (i.e., from the average daily throughput rate of 50 to 56.3) increase on average daily production rate with a buffer configuration of *B= {1, 4, 0, 8, 18, 10, 3, 3, 3, 1, 1, 2}* is achieved. As seen in Figure 5.17, simulating the production line with these buffer allocation decisions lead to a decrease in number of parts waiting for potential bottleneck machines to be available.

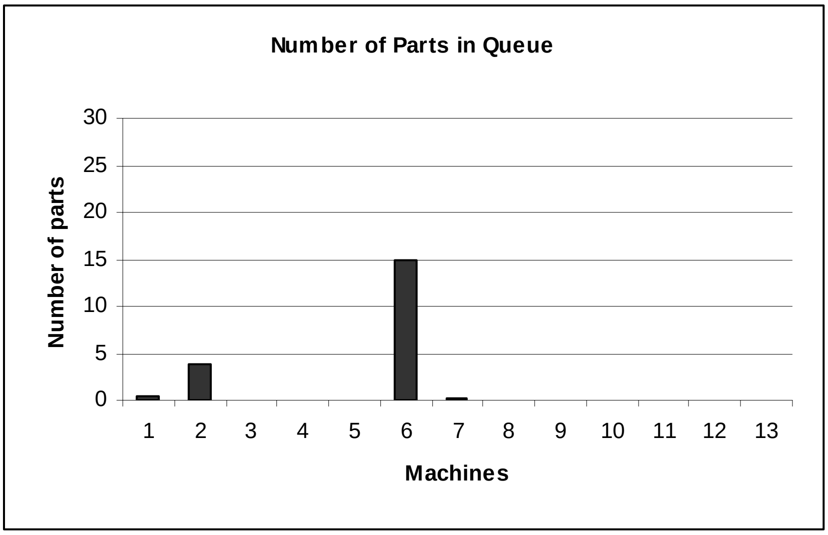


Figure 5.17 Number of parts in queues after buffer allocation

5.3.3.2 Hybrid GA-Based Simulation Approach Using Heuristic Initialization Scheme

In this part of the study, the suggested hybrid approach is used with heuristically generated initial population to solve the same buffer allocation problem given earlier in section 5.3.3.1. As given earlier, in the first phase of the study, a detailed stochastic and dynamic simulation model of the line is developed to identify bottleneck machines. The objective is to use this information for initial population generation in the second phase of hybrid simulation-GA approach. Hence, using this information, initial population is generated in such a way that more buffers allocated to these potential bottleneck machines. Table 5.10 shows the results of computational experiments including the genaration number, the buffer size configurations, maximum production rate, average production rate and the relative difference, respectively.

Table 5.10 Results of the genetic search

Gen. No	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	Max Fitness / Production Rate	Average Fitness / Production Rate	Relative Difference	(*)
0	0	0	14	17	16	8	2	1	2	0	0	0	0.0892	0.0588	0.341	
1	4	1	8	14	20	5	2	2	3	0	1	1	0.1494	0.0935	0.374	
2	4	1	8	17	16	8	2	2	3	1	3	1	0.1470	0.1007	0.315	
3	0	4	14	17	16	9	2	2	2	0	1	3	0.1518	0.1198	0.211	
4	0	4	14	17	16	9	2	2	2	0	1	3	0.1518	0.1198	0.211	
5	0	4	14	17	16	9	2	2	2	0	1	3	0.1518	0.1198	0.211	
6	0	4	9	17	16	9	2	2	2	0	1	3	0.1518	0.1198	0.211	
7	0	4	9	17	16	9	3	2	2	0	1	3	0.1518	0.1198	0.211	
8	0	4	9	17	16	9	3	2	2	0	1	3	0.1518	0.1198	0.211	
9	5	0	8	17	17	8	2	2	3	0	2	3	0.1639	0.1205	0.265	
10	5	0	12	17	17	8	2	2	2	0	2	3	0.1639	0.1205	0.265	
11	5	0	12	17	17	8	2	2	2	0	2	3	0.1639	0.1205	0.265	
12	0	4	9	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
13	0	4	9	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
14	0	4	8	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
15	0	4	12	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
16	0	4	8	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
17	0	4	9	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
18	0	4	9	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
19	0	4	9	17	16	9	1	2	3	2	3	3	0.1735	0.1212	0.301	
20	0	4	9	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
21	0	4	9	17	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
22	0	4	12	16	16	9	1	2	2	0	3	3	0.1735	0.1212	0.301	
23	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	*
24	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
25	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
26	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
27	0	4	5	18	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
28	0	4	9	16	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
29	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
30	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
31	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
32	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
33	0	4	9	19	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
34	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
35	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
36	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
37	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
38	0	4	9	17	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
39	0	4	9	18	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	
40	0	4	9	16	16	9	1	2	2	0	2	2	0.1759	0.1248	0.290	

According to Table 5.10, it has been noted that the proposed approach converges the best average throughput rate after twenty three iterations and the algorithm arrives at an average production rate of 0.1248. The suggested solution remains stagnant during the next 17 generations. In other words, almost all buffer configurations result in the same average daily production rate. Hence, to suggest a buffer configuration, the other criterion which is total buffer size is used and the solution which uses minimum buffer capacity is identified as optimum buffer allocation. As seen in Table 5.11, the buffer configuration of 0, 4, 5, 18, 16, 9, 1, 2, 2, 0, 2 and 2 results in minimum buffer size, 59.

Table 5.11 Buffer configurations that give the best average production rate

Buffer Configuration												Total Buffer Size	Average Production Rate
0	4	9	17	16	9	1	2	2	0	2	2	62	0.1248
0	**4**	**5**	**18**	**16**	**9**	**1**	**2**	**2**	**0**	**2**	**2**	**59**	**0.1248**
0	4	9	16	16	9	1	2	2	0	2	2	61	0.1248
0	4	9	19	16	9	1	2	2	0	2	2	64	0.1248
0	4	9	18	16	9	1	2	2	0	2	2	63	0.1248

Moreover, the convergence rate with respect to the relative difference is summarized in Figure 5.16. Similar to the trend observed in average daily production rate it is obvious that the relative difference converges after twenty three generations.

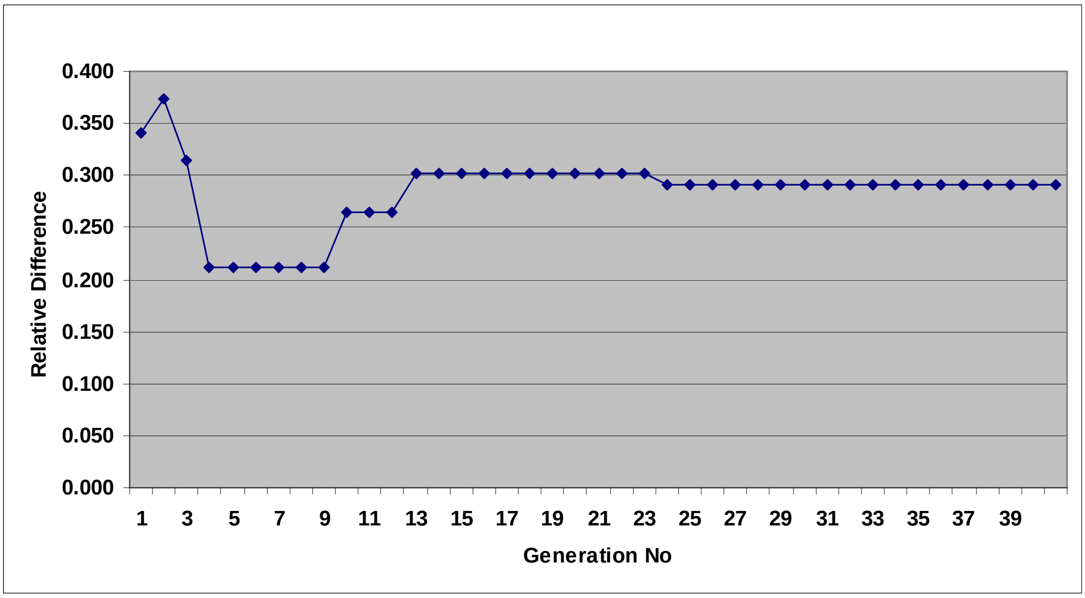


Figure 5.18 Convergence rate of the proposed approach

As a result, 0, 4, 5, 18, 16, 9, 1, 2, 2, 0, 2 and 2 number of buffers have been allocated to the machines in the line and the simulation model of the system is run again with specified buffer sizes, resulting in a production rate of 0.1248/minute, 51.8 exchangers per shift. As it is seen from Figure 5.19, the number of parts waiting in queue is significantly decreased by placing buffer storages in front of bottleneck machines in comparison with Figure 5.8.

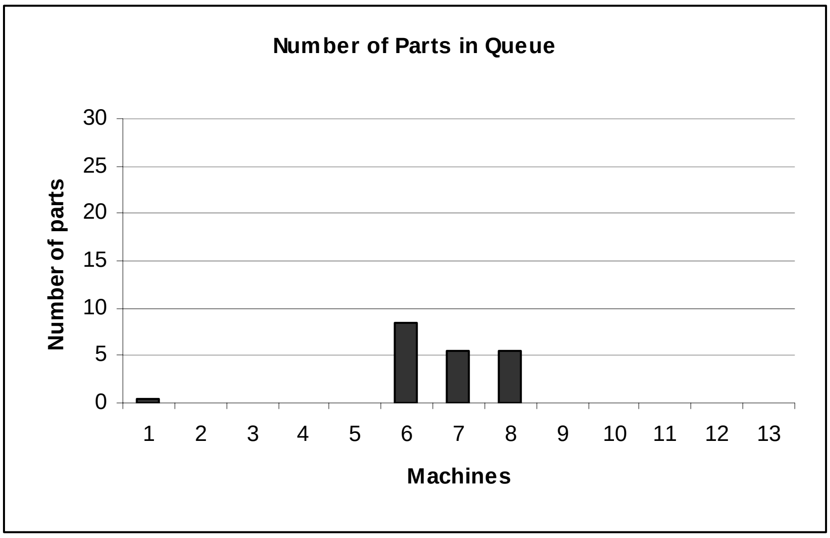


Figure 5.19 Number of parts in queues after buffer allocation

Based on the company records, the average daily throughput is 50 heat exchangers. The optimum solution suggested by the proposed hybrid approach results in average daily production of approximately 52 heat exchangers with a buffer configuration of B= *{0, 4, 5, 18, 16, 9, 1, 2, 2, 0, 2, 2}*. This result implies that just 4% improvement occurs in average daily production.

5.3.3.3 Hybrid GA-Based Simulation Approach Using Random Initialization Scheme for Bottleneck Machines

Lastly, in generating initial population to implement the proposed hybrid approach, as candidate buffer locations, we just considered bottleneck machines identified during the first phase. In another words, we did not take into consideration all of the twelwe machines as candidate buffer locations. Figure 5.20 depicts the representation of buffer areas in front of bottleneck machines in the production line.

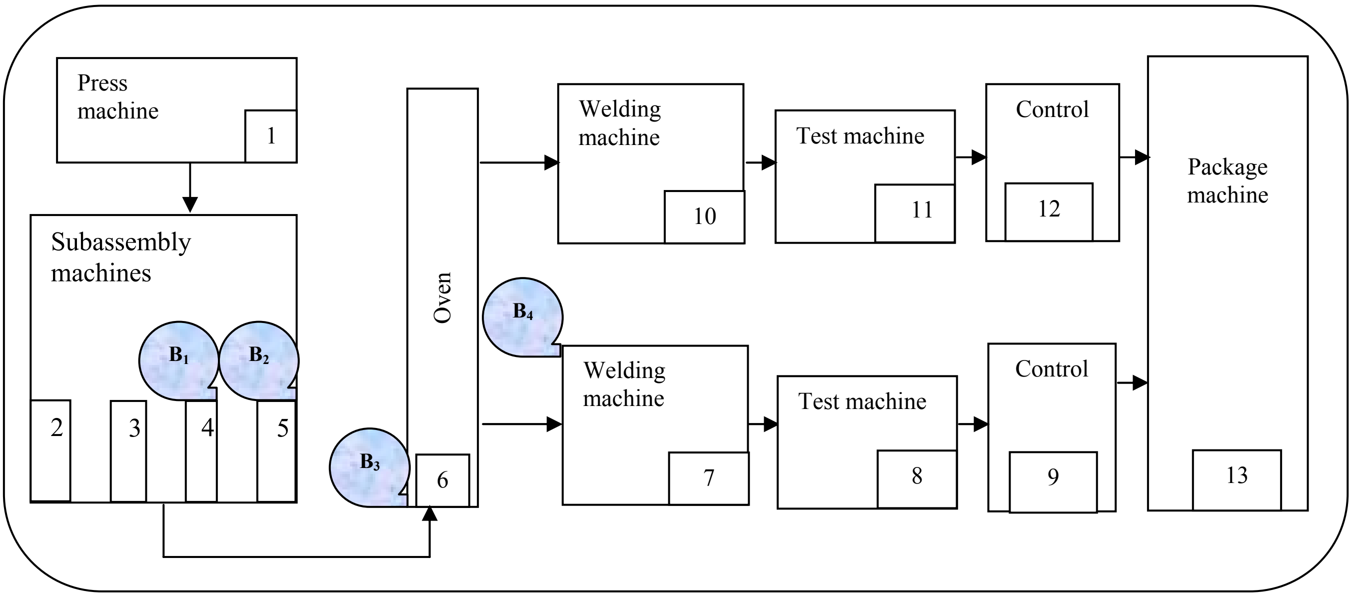


Figure 5.20 Buffer areas in front of identified bottleneck machines in the production line

It should be noted that the upper limit of the total number of buffers to allocate in front of machines is set to 70 considering the available floor space in the heat exchanger production line to accommodate buffers. In similar to the implementations of the alternative searches in previous sections, all GA control parameters such as *P/G*, R_c, R_m, and percent of elitism for replacement strategy are as with the levels of 30/40, 0.80, 0.033 and 50%, respectively. Moreover, the general structure of algorithm in terms of initial population scheme, selection scheme, crossover and mutation techniques, replacement scheme and termination crieria is same as the previous implementation except the chromosome representation scheme. Since four of the machines are identified as bottleneck machines, a chromosome is composed of four unique parts and each chromosome represents a possible buffer configuration. As seen in Figure 5.21, a chromosome alternative represents that 21, 17, 11 and 3 units of buffer will be allocated to these bottleneck machines.

0	1	0	1	0	1	0	1	0	0	0	1	0	0	0	1	1	1	0	0	0	0	1	1

1st variable *2nd variable* *3rd variable* *4th variable*

$$Buffer\_sizes = [21 \;\; 17 \;\; 11 \;\; 3]$$

Figure 5.21 Binary coding representation of alternative chromosome

As a result, the proposed hybrid approch with random initilization for bottleneck machines in the line is employed to solve the buffer allocation problem and the results are given in Table 5.12. According to these results, it has been noted that the proposed approach converges the best average throughput rate after two iterations as a rate of 0.1357 and that occurs 3.68% increase in average daily production rate (i.e., from the initial value of 54.3 to 56.3).

Table 5.12 Results of the genetic search

Gen. No	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Max Fitness / Production Rate	Average Fitness / Production Rate	Relative Difference	(*)
0	1	13	14	18	0.1639	0.1308	0.201	
1	1	13	14	18	0.1639	0.1308	0.201	
2	10	16	14	15	0.1687	0.1357	0.196	*
3	10	16	14	15	0.1687	0.1357	0.196	
4	10	16	14	15	0.1687	0.1357	0.196	
5	10	16	14	15	0.1687	0.1357	0.196	
6	10	17	14	15	0.1687	0.1357	0.196	
7	10	16	14	15	0.1687	0.1357	0.196	
8	10	16	14	15	0.1687	0.1357	0.196	
9	10	16	14	15	0.1687	0.1357	0.196	
10	10	16	14	15	0.1687	0.1357	0.196	
11	12	18	14	11	0.1831	0.1357	0.259	
12	10	16	14	15	0.1687	0.1357	0.196	
13	12	18	14	11	0.1831	0.1357	0.259	
14	12	16	14	15	0.1831	0.1357	0.259	
15	12	16	14	11	0.1831	0.1357	0.259	

Continuation of Table 5.12

Gen. No	Buffer 1	Buffer 2	Buffer 3	Buffer 4	Max Fitness / Production Rate	Average Fitness / Production Rate	Relative Difference	(*)
16	12	18	14	15	0.1831	0.1357	0.259	
17	12	18	14	11	0.1831	0.1357	0.259	
18	12	16	14	11	0.1831	0.1357	0.259	
19	12	18	14	11	0.1831	0.1357	0.259	
20	12	18	14	11	0.1831	0.1357	0.259	
21	12	16	14	11	0.1831	0.1357	0.259	
22	12	18	14	15	0.1831	0.1357	0.259	
23	12	18	14	15	0.1831	0.1357	0.259	
24	12	18	14	11	0.1831	0.1357	0.259	
25	12	18	14	15	0.1831	0.1357	0.259	
26	12	18	14	13	0.1831	0.1357	0.259	
27	12	18	14	12	0.1831	0.1357	0.259	
28	12	16	14	11	0.1831	0.1357	0.259	
29	12	18	14	13	0.1831	0.1357	0.259	
30	12	18	14	12	0.1831	0.1357	0.259	
31	12	18	14	15	0.1831	0.1357	0.259	
32	12	18	14	15	0.1831	0.1357	0.259	
33	12	18	14	11	0.1831	0.1357	0.259	
34	12	18	14	15	0.1831	0.1357	0.259	
35	12	18	14	11	0.1831	0.1357	0.259	
36	12	18	14	15	0.1831	0.1357	0.259	
37	12	18	14	15	0.1831	0.1357	0.259	
38	12	18	14	11	0.1831	0.1357	0.259	
39	12	18	14	15	0.1831	0.1357	0.259	
40	12	18	14	15	0.1831	0.1357	0.259	

The suggested solution remains stagnant during the next 38 generations and for each generation buffer configurations result in the same average daily production rate. In this case, to suggest optimal buffer configuration, minimum buffer capacity used should be considered as an other criterion. Hence, as seen in Table 5.13, the buffer configuration of 12, 16, 14 and 11 which results in minimum buffer size, 53 is identified as optimum buffer allocation.

Table 5.13 Buffer configurations that give the best average production rate

Buffer Configuration				Total Buffer Size	Average Production Rate
10	16	14	15	55	0.1357
10	17	14	15	56	0.1357
12	18	14	11	55	0.1357
12	16	14	15	57	0.1357
12	**16**	**14**	**11**	**53**	**0.1357**
12	18	14	15	59	0.1357
12	18	14	13	57	0.1357
12	18	14	12	56	0.1357

Moreover, the convergence rate with respect to the relative difference is summarized in Figure 5.22. As it is seen clearly in this figure, the relative difference converges quickly within relatively small number of generations. This can be attributed to the fact that the solution space in the case of just considering identified bottleneck machines as candidate buffer storage places is relatively small.

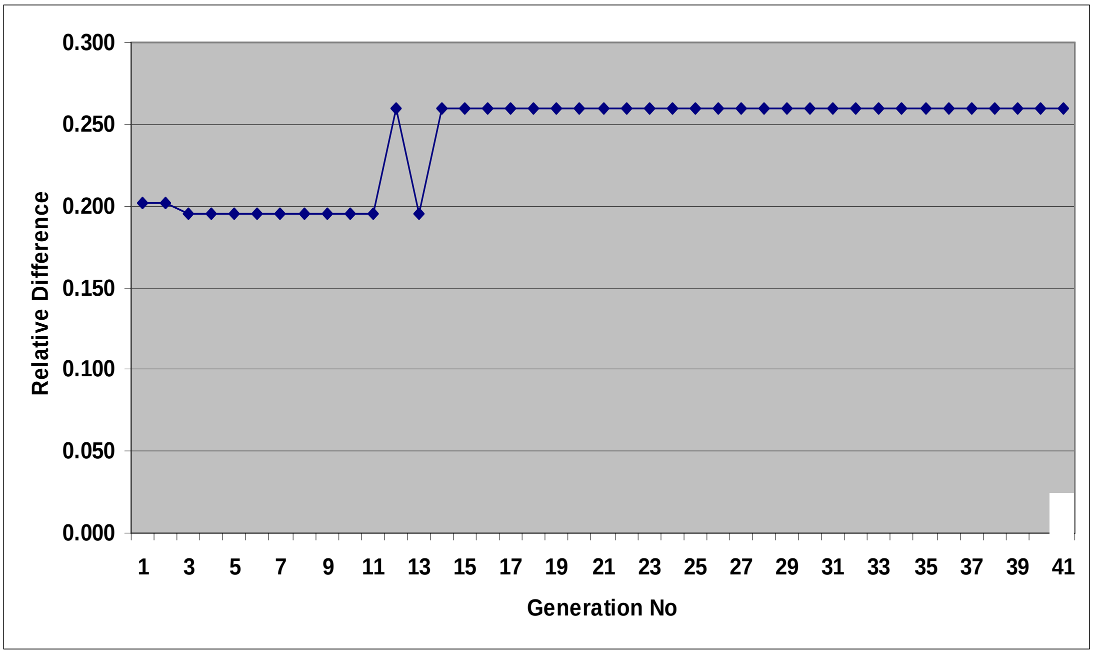


Figure 5.22 Convergence rate of the proposed approach

In conclusion, 12, 16, 14, 11 number of buffers have been allocated to the bottleneck machines and the simulation model of the system is run again with specified buffer sizes, resulting in a production rate of 0.1357/minute, 56.3 exchangers per shift. As it is seen from Figure 5.23, the number of parts waiting in queue is significantly decreased by placing buffer storages in front of bottleneck machines in comparison with Figure 5.8.

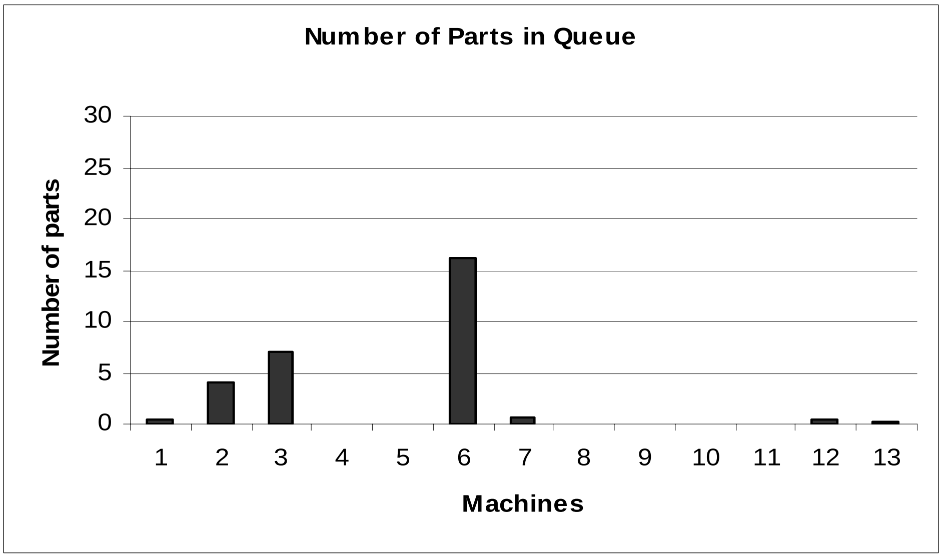


Figure 5.17 Number of parts in queues after buffer allocation

Based on these results, with a buffer configuration of *B= {12, 16, 14, 11},* company records, approximately 13% improvement occurs in average daily production using random initialization scheme of GA considering bottleneck machines in the system.

5.3.4 Comparison of Alternative Results

As mentioned earlier, based on a field study at the company site, it is observed that at average, 50 heat exchangers are produced per day in the pruduction line studied. In order to improve the capacity of this production line, a hybrid simulation-GA approach is proposed for optimum buffer allocation. As a result of experimental studies comparing the performance of the proposed algorithm under three initial population generation schemes, three sets of solutions are found. The solution quality of three approaches are compared in Figure 5.18.

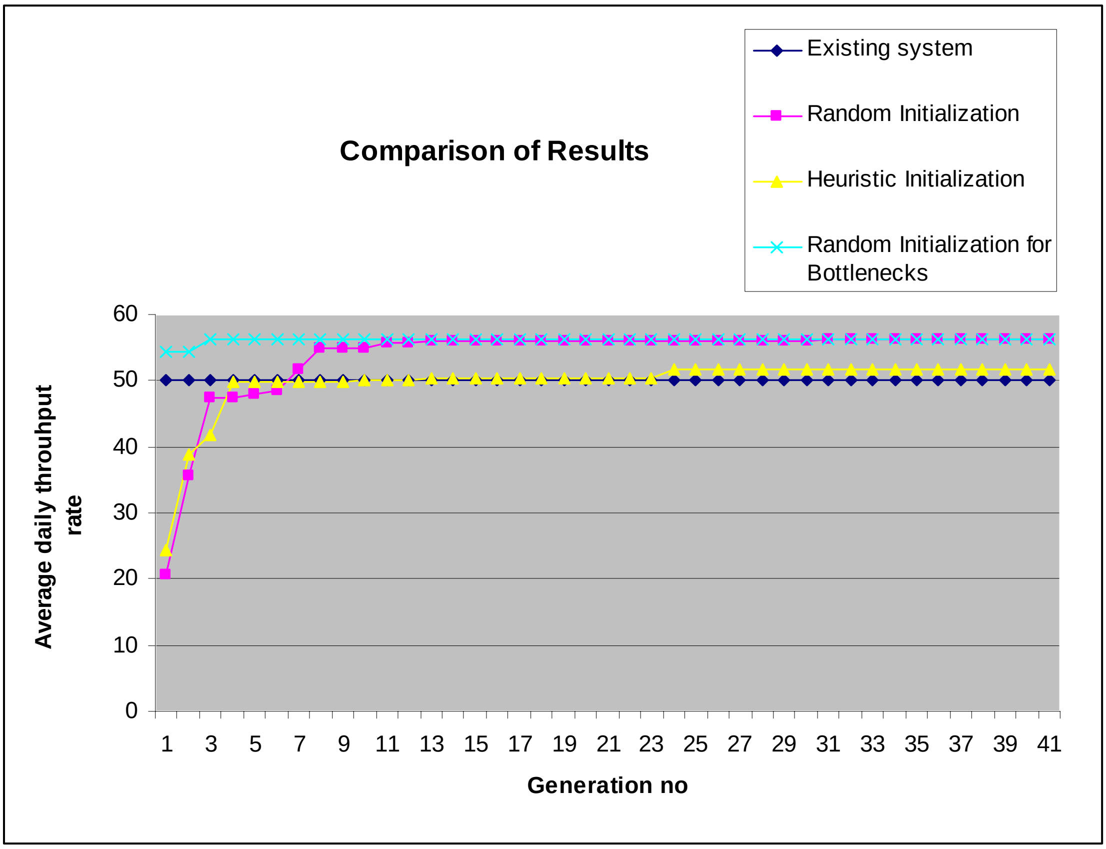


Figure 5.18 Comparison of experimental results

As seen in the figure, the best performance is observed when the proposed hybrid approach is implemented with random initial population and considering bottleneck machines as candidate buffer locations.

CHAPTER SIX
CONCLUSION

The buffer allocation problem is a combinatorial optimization problem which can be observed in many manufacturing systems such as assembly lines, transfer lines or flexible manufacturing systems. A significant number of methods (i.e. analytical methods, approximation methods, aggregation method, simulation, optimization methods and hybrid methods) have been proposed to solve this problem. In this study, a hybrid approach combining the key advantages of both simulation and metaheuristics is presented to find optimal buffer sizes in a real-life production line so as to maximize the production rate, i.e. throughput of the plant.

In the first phase of the M.Sc. study, a detailed bottleneck analysis has been carried out to identify what limits the capacity of the system by developing a stochastic and dynamic simulation model of the system. Having verified and validated the simulation model, various experimental studies have been carried out to identify the bottleneck machines. Considering average machine utilizations and average number of parts in each machine queue for the production line, the bottleneck machines are identified in the heat exchanger production line. In the following phase, GA-based simulation optimization approach is employed to allocate buffers to these bottleneck machines so as to improve the performance of the system. The GA module which includes some problem specific features is combined with a simulation module in a closed loop configuration to solve the buffer allocation problem. Through this integration the genetic algorithm module suggests a buffer configuration at each iteration. Given the candidate buffer configuration, the model generator generates corresponding stochastic and dynamic simulation model. Subsequently, the simulation model of heat exchanger production line has been run to obtain the fitness value of the candidate buffer configuration, i.e. average daily production rate, and this value is communicated to the genetic search module in an iterative manner.

Experimental results show that allocating buffers to the bottleneck machines improves the production rate of the system. It was found that average daily production rate could be increased by about 13% in the real-life production line. Based on these results, it can be stated that the proposed method has great potential to improve the capacity of real manufacturing systems.

One extension of this M.Sc. study could be to integrate this GA-based simulation optimization procedure into a Decision Support System framework. So that both the input data entrance step and also the link among two phases such as bottleneck identification and buffer allocation for capacity improvement can be automated. In doing so, the use of this hybrid approach in an industrial environment will be eased and also the capacity improvement will be achieved with up-to-date data.

Finally, it must be highlighted that, there is usually more than one objective (low costs, low WIP, high revenue) when attempting to optimize the performance of a production system. This necessitates a multi-objective procedure and hence, a future research opportunity for this study would be to employ a multi-objective GA-based simulation optimization procedure for the buffer allocation problem.

REFERENCES

Akgündüz, O. S. (2008). *Solving Mixed-model Assembly Line Sequencing Problem Using Adaptive Genetic Algorithms,*. M.Sc. Thesis, Dokuz Eylül University Graduate School of Natural and Applied Science, Department of Industrial Engineering, Izmir.

Alden, M. J., Burns, D. L., Costy, T., Hutton, R. D., Jackson, C. A., Kim, D. S., Kohls, A. K., Owen, H. J., Turnquist, A. M., Vander Veen, D. J. (2006). General Motors Increases Its Production Throughput, *INFORMS,* Vol. 36, No. 1, pp. 6–25.

AlDurgham, M.M. & Barghash, M.A. (2008). A Generalised Framework for Simulation-Based Decision Support for Manufacturing, *Production Planning & Control* Vol. 19, No. 5, July 2008, pp. 518–534.

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, pp. 137–151.

Altıparmak, F., Dengiz, B., Bulgak, A. (2002). Optimization of Buffer Sizes in Assemly Systems Using Intelligent Techniques, *Proceedings of the 2002 Winter Simulation Conference,* pp. 1157-1162.

Altıparmak, F., Dengiz, B., Bulgak, A. (2007). Buffer Allocation and Performance Modeling in Asynchronous Assembly System Operations: An Artificial Neural Network Metamodeling Approach, *Applied Soft Computing,* pp. 946-956.

Aneja, Y.P. & Punnen, A. P. (1999). Multiple Bottleneck Assignment Problems. *European Journal of Operational Research,* 112, pp.167-173.

Ay, G. (2009). *A Hybrid Approach To Setting Order Promising Times In A Supply Chain Network***,** M.Sc. Thesis, Dokuz Eylül University Graduate School of Natural and Applied Science, Department of Industrial Engineering, Izmir.

Aydın G., Yılmaz D. & Kayman, A. Y. (2004). Bir Aydınlatma Firmasında Darboğaz Yaratan Makine Problemlerinin Alternatif Çözümlerinin Simülasyonu, *Yöneylem Araştırması ve Endüstri Mühendisliği XXIV. Ulusal Kongresi (*YA/EM 2004), Adana.

Balas, E. & Vazacopoulos, A. (1998). Guided Local Search with Shifting Bottleneck for Job Shop Scheduling, *Management Science,* Vol. 44, No. 2, pp. 262 - 275.

Ban, X., Chu, L. & Benouar, H. (2006). Bottleneck Identification and Calibration for Corridor Management Planning, *86th Transportation Research Board Annual Meeting.*

Battini, D., Persona, A. & Regattieri, A. (2009). Buffer Size Design Linked to Reliability Performance: A Simulative Study, *Computers & Industrial Engineering*, 56, pp. 1633–1641.

Boyabatlı, O. & Sabuncuoglu, I. (2004). Parameter Selection in Genetic Algorithms, Systemics, *Cybernetics and Informatics,* Vol. 2, No. 4, pp. 78-83.

Bulgak, A., Diwan, P. D. & Inozu, B. (1995). Buffer Size Optimization in Asynchronous Assembly Systems Using Genetic Algorithms, *Computers & Industrial Engineering*, Vol. 28, pp. 309–322.

Bulgak, A., (2006).Analysis and Design of Split and Merge Unpaced Assembly Systems by Metamodeling and Stochastic Search, *International Journal of Production Research,* Vol. 44, pp. 4067–4080.

Butler, T. (2008). *Design in a Simulation Environment,* A Thesis Presented to The Academic Faculty*,* Georgia Institute of Technology.

Byrne, M.D. & Bakir, M.A. (1999). Production Planning Using A Hybrid Simulation -Analytical Approach, *Int. J. Production Economics*, 59, pp. 305-311.

Ceryan, E. (2008). *Capacity planning in a textile company*. M.Sc. Thesis, Dokuz Eylül University Graduate School of Natural and Applied Science, Department of Industrial Engineering, Izmir.

Chaharsooghi, S. K. & Nahavandi, N. (2003). Buffer Allocation Problem, A Heuristic Approach, *Scientica Ironica,* Vol. 10, No.4, pp. 401-409.

Chiang, S. Y., Kuo, C. T. & Meerkov S. M. (1998). Bottlenecks in Markovian Production Lines: A Systems Approach, *IEEE Tarnsactions on Robotics and Automation,* Vol 14, No.2, pp. 352-359.

Colledani, M., Matta, A. & Tolio, T. (2005). Performance Evaluation of Production Lines with Finite Buffer Capacity Producing Two Different Products, *OR Spectrum*, 27, pp. 243-263.

Can, B., Beham, A. & Heaey, C. (2008). A Comparative Study of Genetic Algorithm Components in Simulation-Based Optimization, *Proceedings of the 2008 Winter Simulation Conference,* pp. 1829-1837.

Dengiz, B. & Alabas, C. (2000). Simulation Optimization Using Tabu Search, *Proceedings of the 2000 Winter Simulation Conference,* pp. 805-810.

De Vin, L. J. Amos H.C. Ng & Oscarsson, Jan. (2004). Simulation Based Decision Support for Manufacturing System Life Cycle Management, *Journal of Advanced Manufacturing Systems,* Vol. 3, No. 2, pp. 115-128.

Ding, H., Benyoucef, L. & Xie, X. (2003). A Simulation-Optimization Approach Using Genetic Search For Supplier Selection, *Proceedings of the 2003 Winter Simulation Conference,* pp. 1260-1267.

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 Modelling and Algorithms,* Vol. 1, pp. 89-104.

Dolgui, A., Eremeev, A. & Sigaev, V. (2007). HBBA: Hybrid Algorithm for Buffer Allocation in Tandem Production Lines, *J Intel Manufs,*18, pp. 411-420.

Duvivier, D., Roux, O., Dhaevers, V., Meskens, N. & Artiba, A. (2007). *Multicriteria Optimisation and Simulation: An İndustrial Application,* Springer Science+Business Media.

Eskandari H., Rabelo, L. & Mollaghasemi, M. (2005). Multiobjective Simulation Optimization Using an Enhanced Genetic Algorithm, *Proceedings of the 2005 Winter Simulation Conference,* pp. 833-841.

Faget P., Eriksson, U. & Herrmann, F. (2005). Applying Discrete Event Simulation And An Automated Bottleneck Analysis As An Aid to Detect Running Production Constraints, *Proceedings of the 2005 Winter Simulation Conference,* pp. 1401-1407.

Gen, M. & Cheng, R. (1997). *Genetic ALgorithms&Engineering Design*, JohnWiley&Sons, Canada.

Gershwin, B. S. & Schor, J. E. (2000). Efficient Algorithms for Buffer Space Allocation*, Annals of Operations Research,* Vol. 93, pp. 117-144.

Gilland, W. G. (2002). A Simulation Study Comparing Performance of CONWIP and Bottleneck-Based Release Rules, *Production Plannıng & Control,* Vol. 13, No. 2, pp. 211-219.

Goldratt, E. (1992). *The Goal: A Process of Ongoing Improvement*, North River Press.

Grant, H., Moses, S. & Goldsman, D. (2002). Using Simulation to Evaluate Buffer Adjustment Methods in Order Promising, *Proceedings of the 2002 Winter Simulation Conference,* pp. 1838-1845.

Grosfeld-Nir, A. (1995). Single Bottleneck Systems With Proportional Expected Yields And Rigid Demand, *European Journal of Operational Research,* 80, pp.297-307.

Harris, J. H. & Powell, S. G. (1999). An Algorithm for Optimal Buffer Placement in Reliable Serial Lines*, IIE Transactions,* Vol. 31, pp. 287-302.

Hasgül, S. & Büyüksünetçi, A.S. (2005). Simulation Modeling And Analysis of a New Mixed Model Production Lines, *Proceedings of the 2005 Winter Simulation Conference,* pp. 1408-1412.

Hsieh, S. (2002). Hybrid Analytic and Simulation Models for Assembly Line Design and Production Planning, *Simulation Modelling Practice and Theory*, 10, pp. 87–108.

Ilgın, M. A. (2006). *Joint Optimization of Spare Parts Inventory and Maintenance Policies Using Hybrid Genetic Algorithms*, M.Sc. Thesis, Dokuz Eylül University Graduate School of Natural and Applied Science, Department of Industrial Engineering, Izmir.

Jahangirian, M., Tillal, E., Nasaer, A., Stergioulas, L. K. & Young, T. (2010). Simulation in Manufacturing and Business: A Review, *European Journal of Operational Research,* 203, pp. 1-13.

Kasemset, C. & Kachitvichyanukul, V. (2007). Simulation-based Procedure for Bottleneck Identification, *AsiaSim 2007*, CCIS 5, pp. 46-55.

Kim, B. & Kim S. (2001). Extended Model for a Hybrid Production Planning Approach, *Int. J. Production Economics,* 73, pp. 165-173.

Konak, A. & Kulturel-Konak S. (2005). Simulation Optimization Using Tabu Search: An Empirical Study, *Proceedings of the 2005 Winter Simulation Conference*, pp. 2686-2692.

Kozan, E. (2003). Container transfer logistics at multimodal container terminals. *School of Mathematical Sciences, Queensland University of Technology.*

Law, A. M. & McComas, M. G. (1999). Simulation of Manufacturing Systems *Proceedings of the Winter Simulation Conference.*

Lawrence, S. & Buss, A. H. (1994). Shifting Production Bottlenecks: Causes, Cures, and Conundrums, *Production and Operations Management* Vol. 3, No. 1, U.S.A.

Lima, E., Chwif, L. & Barreto, M. (2008). Methodology for Selecting the Best Suitable Bottleneck Detection Method, *Proceedings of the 2008 Winter Simulation Conference,* pp.1746-1751.

Liu, C. & Feng-sheng, T. (1994). Buffer Allocation via the Genetic Algorihm, *Proceedings of the 33rd Conference on Decision and Control,* pp. 609-610.

Lutz, M. C., Davis, R. K. & Sun, M. (1998). Determining Buffer Allocation and Size in Production Lines Using Tabu Search, *European Journal of Operational Research,* 106, pp. 301–316.

Massim, Y., Yalaoui, F., Amodeo, L., Chatelet, E. & Zeblah, A. (2009). Efficient Combined Immune-Decomposition Algorithm for Optimal Buffer Allocation in Production Lines for Throughput and Profit Maximization, *Computers & Operations Research,* 00162-2.

Matta, A., Runchina, M., & Tolio, T. (2005). Automated Flow Lines with Shared Buffer, *OR Spectrum*, 27, pp. 265-*286*.

Nahas, N., Ait-Kadi, D. & Nourelfath, M. (2006). A New Approach for Buffer Allocation in Unreliable Production Lines, *Int. J. Production Economics,* 103, pp. 873–881.

Othman, Z., Kamaruddin, S. & Ismail , M. S. (2007). Optimal Buffer Allocation for Unpaced Balanced and Unbaanced Mean Processing Time, *Jurnal Teknologi,* 46 (A), pp. 31–42.

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*, pp. 25-27.

Rekiek, B. & Delchambre, A. (2006). *Assembly Line Design: The Balancing of Mixed-Model Hybrid Assembly Lines with Genetic Algorithms*, Springer-Verlag London Limited.

Roser, C., Nakano, M. & Tanaka, M. (2001). A Practical Bottleneck Detection Method, *Proceedings of the Winter Simulation Conference,* pp. 949-953.

Roser, C., Nakano, M. & Tanaka, M. (2002). Shifting Bottleneck Detection. In Proceedings of the 2002 Winter Simulation Conference, ed. E. Yücesan, C.-H. Chen, J.L. Snowdon, and J. M. Charnes, 1079-1086. San Diego, CA, USA.

Roser, C., Nakano, M. & Tanaka, M. (2003). Comparison of Bottleneck Detection Methods for AGV Systems, *Proceedings of the Winter Simulation Conference,* pp. 1192-1198.

Roser, C., Nakano, M. & Tanaka, M. (2003). Buffer Allocation Model Based on a Single Simulation, *Proceedings of the Winter Simulation Conference,* pp. 1238-1246.

Sengupta, S., Das, K. & Vantil, R. P. (2008). A New Method for Bottleneck Detection, *Proceedings of the Winter Simulation Conference,* pp. 1741-1745.

Shanthikumar, J.G. & Sargent R.G. (1983). A Unifying View of Hybrid Simulation/Analytic Models and Modelling, *Operations Research 31.*

Shi, C. & Gershwin, S. B. (2008). *An Efficient Buffer Design Algorithm for Production Line Profit Maximization*, Department of Mechanical Engineering Massachusetts Institute of Technology Cambridge, Massachusetts, 02139-4307, USA.

Sivanandam, S. N. & Deepa S. N. (2008). *Introduction to Genetic Algorithms*, Springer Berlin Heidelberg, New York.

Smith, J.S. (2003). Survey on the Use of Simulation for Manufacturing System Design and Operation, Journal of Manufacturing Systems, Vol. 22, No. 2, pp. 157-171.

Sower, V. E., Savoie, M. J. & Motwani, J. G. (1993). Use of Simulation for Determining Appropriate Levels of Buffer Inventories in a JIT Environment: A

New Perspective on Implementation of JIT, *International Journal of Management and Systems,* Vol. 9, No.3, pp. 286-299.

Spinellis, D. D. & Papadopoulos, C. T. (1999). Production Line BufferAllocation: Genetic Algrithms Versus Simulated Annealing, *Second International Aegean Conference on the Analysis and Modelling of Maufacturing Systems*, pp. 89-101.

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

Swain, J. J. & Farrington, P.A. (1994). Designing Simulation Experiments for Evaluating Manufacturing Systems, *Proceedings of the Winter Simulation Conference,* pp. 69-76.

Tasan, S. Ö. (2007). Solving Simple and Mixed-Model Assembly Line Balancing Problems Using Hybrid Metaheuristic Approaches, Phd. Thesis, Dokuz Eylül University Graduate School of Natural and Applied Science, Department of Industrial Engineering, Izmir.

Um, I., Lee, H. & Cheon, H. (2007). Determination of Buffer Sizes in Flexible Manufacturing System by Using the Aspect-oriented Simulation, *International Conference on Control, Automation and Systems,* pp. 1729-1733.

Vouros, G. A. & PApadopoulos, H. T. (1998). Buffer Allocation in Unreliable Production Lines Using a Knowledge Based System, *Computers Operations Research,* Vol. 25, pp. 1055-1067.

Wellman, M. A. & Gemmill D. D. (1995). A Genetic Algorithm Approach to Optimization of Asynchronous Automatic, *The International Journal of Flexible Manufacturing Systems,* 7, pp. 27-46.

Zhang, R. & Wu, C. (2009). Bottleneck Identification Procedures for the Job Shop Scheduling Problem with Applications to Genetic Algorithms, *Advanced Manufacturing Technology*, 42, pp. 1153-1164.