An Integrated Circuit Design for Implementation of a Chemical Sensor Array

By

Yiping Li

A Thesis submitted to the Faculty of Graduate Studies of

The University of Manitoba

in partial fulfilment of the requirements of the degree of

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

University of Manitoba

Winnipeg, Manitoba, Canada

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION

An Integrated Circuit Design for Implementation of a Chemical Sensor Array

BY

Yiping Li

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirement of the degree

Of

MASTER OF SCIENCE

ACKNOWLEDGMENTS

I would like to express my gratitude to all those who gave me the possibility to complete this thesis. I am deeply indebted to my supervisor, Dr. Tarek ElMekkawy, for his advices, encouragement and support throughout my research and writing of this thesis.

I also want to thank my colleagues for their help and suggestions. I am especially grateful to Mr. S. Fahmy who was particularly helpful in my research.

I would like to give my special thanks to my husband for his patient love and support which encouraged me to complete this work.

Lastly, and most importantly, I wish to thank my parents, who bore me, raised me, taught me, and loved me. This thesis is dedicated to my parents.

ABSTRACT

In tandem Automated Guided Vehicle (AGV) systems, the shop floor is partitioned into a group of non-overlapping zones, each served by a single dedicated AGV. Pickup/drop-off (P/D) points are installed to link these zones as transfer points. In this thesis, a genetic algorithm (GA) is proposed for partitioning the tandem AGV systems. The objective is to minimize the maximum AGV workload in order to balance the workload among all the zones to avoid the occurrence of bottlenecks. The performance of the proposed algorithm is evaluated through comparison with the reported results in the literature. The results show that the performance of the proposed algorithm is superior compared to the algorithms reported in previous studies.

The difficulty of applying the GAs to a practical problem is tuning up their parameters such as population size, crossover rate and mutation rate. The performance of a GA is strongly affected by the chosen values of these parameters. In this thesis, Design of Experiments (DOE) is used to define the best combination of the developed GA parameters' values by analyzing the main effect of each parameter and interaction effects between these parameters and some system characteristics on the obtained solutions' quality and computational time. The considered system characteristics are system size, expected zone loading, and the designated number of zones. The obtained results demonstrated the efficiency of the

presented systematic method of tuning the GA's parameters to solve the partitioning problem of tandem AGV systems.

A local search algorithm is then proposed and combined with the developed GA to improve its performance. Hence, a new memetic algorithm (MA) is proposed and applied to optimize the partitioning problem of the tandem AGV systems. Then a performance comparison between the developed GA and MA is then carried out on a group of benchmarking problems. The obtained results demonstrated the efficiency of the developed MA in solving the partitioning problem of tandem AGV systems. In terms of solutions quality, the proposed MA outperforms all the previous approaches as well as pure GA. On the other hand, MA seems worse than pure GA in terms of computational time, especially for large size problems. However, the computational time of MA is still within the accepted range.

As for the vehicle dispatching problem in AGV systems, a simulation study combined with experimental design is conducted to analyze the effects of a number of empty vehicle dispatching rules which are Shortest Time to Travel First (STTF), First-Encountered-First-Served (FEFS), Largest Queue Size (LQS), and First-Come-First-Served (FCFS). Two configurations of a benchmark problem are simulated with the mentioned dispatching rules on three system performance criteria: average vehicle workload, throughput rate, and average queue length.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background

Automated guided vehicles (AGVs) are battery-powered and driverless vehicles. The guide paths of AGVs are pre-designed and the path selection of AGVs can be programmed. AGVs automatically follow their routes and transfer materials in manufacturing systems. They replace the traditional transporters such as forklifts, tractors and roller conveyors.

The AGV first came to the market in the 1950s. Nowadays, AGV systems have become part of many manufacturing operations, including flexible manufacturing systems, warehousing, and service industries. The primary advantages of AGV systems are flexibility, low space utilization, improved safety and low operational cost. AGV systems are extremely flexible, since the vehicles can be easily rerouted to respond to the changes in an existing system. AGVs only occupy workspace when working temporarily in a given area. The space occupied by AGVs can be shared with other vehicles, such as forklifts. Therefore, the overall space utilization in the factory is improved. AGVs are reliable in hazardous and special environments. They also reduce the operational cost since they take up less floor space and direct labor, and they finish the tasks with high efficiency. In addition, AGV systems offer increased control over material flow movement, ability to interface with various peripheral systems, and increased throughput due to the dependable on-time delivery.

An AGV system consists of the vehicles, guide path, guidance system, control system, information transfer system, material pallets, and pickup/delivery (P/D) points (Ross et al., 1996). The P/D points are usually located beside the processor stations and they are where the AGVs pick up or drop off the loads. They are also located at the input and output stations that bring the jobs into and out of the system.

1.2 Configurations of AGV Systems

The conventional configuration is the first applied layout for an AGV system. In this configuration, the AGV guide path passes through all the present stations and every vehicle is allowed to visit any P/D point. The typical application of this configuration is shown in figure1-1. In this figure, the solid lines represent the guide path of the AGVs and the arrows show the permitted direction of each segment. Stations represent processor stations or input and output (I/O) stations. The jobs are brought into and sent out of the system through the I/O stations. Each job in the system requires a sequence of operations to be processed at the processor stations. The P/D points represent the input and output buffers of each station.

In conventional AGV systems, since every vehicle is allowed to visit any P/D point, the possibility of collisions between vehicles is quite high and traffic control becomes difficult. To avoid this problem, Bozer and Srinivasan (1989) introduced the tandem configuration (Figure 1-2).

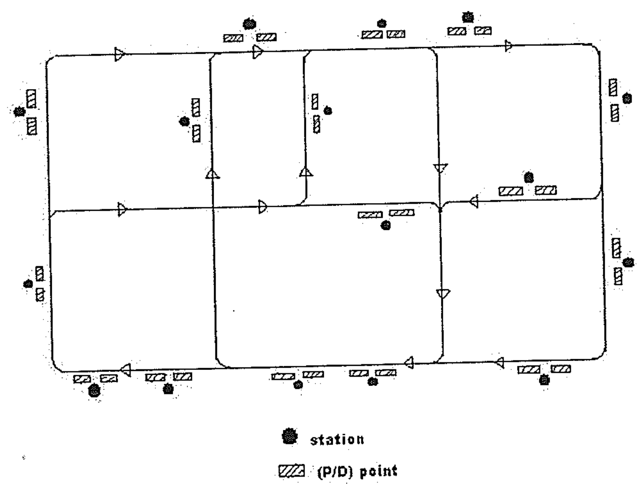


Figure 1-1: Conventional AGV system (Bozer and Srinivasan, 1991)

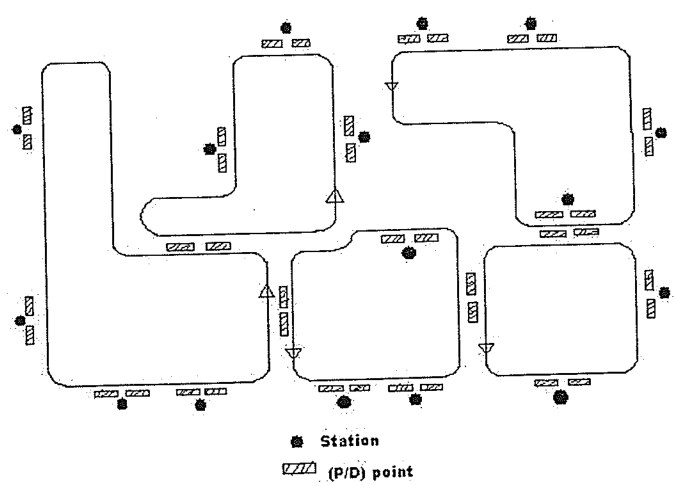


Figure 1-2: Tandem AGV system (Bozer and Srinivasan, 1991)

In the tandem configuration, all the stations are partitioned into a number of non-overlapping zones, where each zone is served by a single dedicated vehicle.

Each zone may have one or more transfer points that link it with other zones. Travel between zones may be achieved using conveyors or any other automated handling equipment. The congestion and conflicts mentioned in the conventional systems can be greatly reduced in the tandem systems since each zone is served by only one AGV. The tandem configuration also features a less complicated control system and offers more flexibility since a zone can be added or removed without affecting other zones.

On the other hand, the tandem configuration suffers from a number of limitations. The jobs may have to be handled by more than one vehicle within and between zones before they reach their destinations. Therefore, delay and other routing problems may be caused. In addition, more floor space and more equipment may be required for the transportation of jobs from one zone to another. Moreover, bottleneck zones may exist without efficient operation and good balancing of the workload among different zones.

1.3 Control and Dispatching Issues

Routing and dispatching rules in AGV systems are usually referred to as the operational issues. Studies concerning these issues usually assume that the guide path, location of P/D stations, and fleet size are already known. Dealing with these problems depends mainly on the type of layout used in the system. In conventional layouts, some traffic problems may occur: more than one vehicle may be dispatched to a waiting job; an AGV must select from variety of routes to reach its destination; a considerable number of paths intersect, resulting in the possibility of conflicts. Moreover, the AGV dispatching problem is either vehicle-initiated or work-centre (station)-initiated. In past literature, many rules were adopted for the solution of this problem in conventional systems, and most of these rules were then evaluated using different techniques like Simulation and Petri-nets. A summary of these rules is shown in Table 1-1.

Table 1-1: AGV dispatching rules (Ganesharajah et al., 1998)

Vehicle initiated rules	Acronym	Work center initiated rules	Acronym
First-come-first-served	FCFS	Farthest vehicle	FV
First-encountered-first-served	FEFS	First available vehicle	FAFS
Largest queue size	LQS	Least cumulative idle time	LIT
Longest inter-arrival time	LIT	Least utilized vehicle	LUV
Longest travel time	LTT	Longest idle vehicle	LIV
Longest waiting time	LWT	Most cumulative idle time	MIT
Maximum demand	MD	Nearest (idle) vehicle	NV or NIV
Maximum outgoing queue size	MOQS	Random vehicle	RV
Minimum remaining outgoing queue size	MROQS		
Minimum work-in-queue	MWQ		
Modified first-come-first-served	MFCFS		
Random work center	RW		
Shortest time to travel first	STTF		
Unit load shop arrival time	ULSAT		
Vehicle looks for work	VLFW		

As mentioned earlier, the tandem configuration greatly reduces the operational problems usually encountered in conventional systems and the complexity of the required control system for the following reasons:

- When a station requires an AGV, only one AGV can be sent to fulfill the required move request, thereby eliminating many of the dispatching problems.
- An AGV can always reach its destination through the shortest route.
- Traffic management is no longer needed since there will never be two or more AGVs that may be occupying the same point in the path.

For the above reasons, the operational issues in tandem systems can be reduced significantly. Since only one AGV can be sent to respond to the request, the work center initiated dispatching problems are eliminated. The only considered dispatching issue is the choice of the vehicle-initiated empty vehicle dispatching rules when an AGV receives simultaneous requests by workstations. In past literature, only the study conducted by Bozer and Srinivasan (1992) considered applying two different dispatching rules in tandem systems. These were the FEFS and the STTF rules. They concluded that the latter resulted in better system performance in terms of average vehicle workload and system throughput.

1.4 Problem Definition

The objective of the current research is to apply a meta-heuristic approach to the design of an efficient zone-partitioning algorithm in the tandem AGV systems so that an optimal balance of the workload among zones can be achieved. In other words, all the stations in the system are grouped into a number of non-overlapping zones, where they are only served by one dedicated vehicle. The algorithm determines the grouping of stations to be included in each zone in order to reach the objective function, which is minimizing the maximum workload in the system. The main idea behind this objective function is to avoid bottleneck zones.

When solving the problem, some information is provided beforehand:

1. The number and coordinates of all the stations in the system (system layout);
2. The number of job types and their processing routes among the stations;
3. The pickup and drop-off time at each P/D point;
4. The traveling speed of each AGV;
5. The number of tandem zones required (the number of AGVs).

A few assumptions are made as follows:

1. Bi-directional movement of all the AGVs is allowed in the system.
2. P/D points are co-located with their stations.
3. The pickup and drop-off times for the AGVs at all the P/D points are equal and constant.
4. AGVs always select the shortest rectilinear path to reach their destinations in a zone when it is loaded.
5. Both loaded and empty trips are considered when the AGV workload is estimated.
6. The Shortest-Time-To-Travel-First (STTF) empty vehicle dispatching policy is adopted in all zones. In other words, when an empty vehicle has requests from a

few stations at the same time, it serves the one to which travel time is the shortest.

7. Each station will be assigned to only one zone.
8. Each zone should have at least two stations.
9. Intersections and overlaps are forbidden among zones.

In addition to the developed meta-heuristic approach in this thesis, a simulation study combined with design of experiments is conducted to analyze the effects of a number of empty vehicle dispatching rules on the performance of the tandem AGV systems. The selected vehicle dispatching rules are Shortest Time to Travel First (STTF), First-Encountered-First-Served (FEFS), Largest Queue Size (LQS), and First-Come-First-Served (FCFS). Two configurations of a benchmark problem are simulated with the above-mentioned dispatching rules on three system performance criteria: average vehicle workload, throughput rate, and average queue length.

1.5 Thesis Outline

This thesis is divided into seven chapters. Chapter 1 presents the background of AGV systems, the introduction of different AGV configurations, and a brief definition of the considered problem in the thesis.

In Chapter 2, literatures about designing the guide-path of AGV systems are reviewed. A detailed discussion of key literature that gives attention to the design of tandem AGV systems is given too.

In Chapter 3, a genetic algorithm (GA) is introduced and applied to solve the partitioning problem in tandem AGV systems. A repair procedure that deals with generated infeasible solutions is proposed. The results are compared with a few benchmark problems.

In Chapter 4, design of experiments (DOE) is devised to further test the performance of the genetic algorithm (GA) and to analyze the relationship between the GA's parameters and the system design characteristics.

In Chapter 5, a memetic algorithm (MA) has been developed by combining the developed GA with a new local search method to solve the partitioning problem of the tandem AGV systems. The results from MA are compared with both pure GA and reported methods in the literature.

A simulation study is carried out in Chapter 6 to analyze the effects of a few selected different AGV dispatching rules on the system performance.

Finally, in Chapter 7, the conclusions of the current research are presented and future work is discussed.

CHAPTER 2

LITERATURE REVIEW

The design of AGV systems involves a physical aspect and a control system development aspect. The physical aspect includes the guide-path design, determination of the locations of P/D points, and the calculation of the number of required vehicles. The design of control systems requires that the material handling tasks assigned to all vehicles follow the most efficient paths to reach their destinations.

The problems of designing the guide-path configuration and the determination of the locations of P/D points have served as the basis for large amounts of research since the advent of AGV systems in industry. The guide-path of the AGV is the track that it follows to reach its destination. The AGV can never go off track and move by itself. Consequently, a few problems have to be solved in designing the guide-path of AGV systems. First, the configuration of this path should be designed in a manner that permits the AGV to reach all its destinations. Second, the amount of floor space and the length of the guide-path have to be decided. Third, the possibility that an AGV may collide with another one has to be eliminated.

In this chapter, the literature available on AGV systems guide-path design is reviewed based on the selected configurations. Moreover, a few key approaches to solving the partitioning problems in tandem AGV systems in the literature are discussed. Finally, the literature review is analyzed and summarized.

2.1 Conventional Configuration

In conventional configuration, the design problem is to find the direction of traffic along different segments of the guide-path and locate the P/D points of each station. Gaskins and Tanchoco (1987) were the first to discuss this problem by proposing a 0-1 integer programming model to find the unidirectional flow path in conventional configuration. They designed a node-arc network in which nodes represent intersections and P/D points, and arcs represent possible directions of travel along the aisles. Each arc was assigned a 0 or 1 integer value. If the assigned arc was chosen in the final solution, its variable will be given a value of 1. The authors suggested the use of the model to determine an initial design and then evaluate it using simulation. However, they did not consider the issues of vehicles traveling empty, blocking, and congestion.

Another integer programming model was designed by Goetz and Egbelu (1990). Their model determined the direction of traffic flow in a unidirectional network and the locations of P/D points simultaneously. They suggested considering only large flows between stations to reduce the problem size.

Sinriech and Tanchoco (1991) proposed a branch and bound method dealing with small set of nodes. Their research was based on an earlier model built by Kaspi and Tanchoco (1990). Only loaded vehicle trips were considered in this study.

Kouvelis et al. (1992) used the node-arc network and considered AGV-empty travel as well as loaded travel. They proposed five heuristic procedures for solving this problem along with different simulated annealing (SA) models. The results indicated a composite heuristic of the proposed procedures would produce solutions of comparable quality in much less time.

In order to find the traffic direction along different segments of the guide path and locate the P/D points of each station, Seo and Egbelu (1995) proposed a flexible design methodology to deal with product-mix changes of arriving jobs.

The above mentioned studies are all based on unidirectional movement systems. Egbelu and Tanchoco (1986) proposed a model to describe the flow and control of AGVs in a bidirectional network. They concluded that the use of bidirectional guide-paths in networks with few AGVs can lead to an increase in productivity.

Kim and Tanchoco (1993) proved that the bidirectional layouts outperform the unidirectional layouts in terms of the number of jobs completed per unit time. In other words, if both systems have the same number of vehicles, a bi-directional system achieves a higher throughput rate than a unidirectional system. Moreover, bi-directional systems require fewer vehicles than unidirectional systems for the same production target. Bi-directional systems increase the flow path reliability due to the greater degree of freedom in selecting travel paths within the network. In addition, bi-directional systems occupy significantly less space than multi-lane unidirectional systems.

2.2 Tandem Configuration

Bozer and Srinivasan (1991) proposed an analytical model to evaluate the performance of an AGV working in a loop that would later form a zone in a tandem configuration. In 1992, they introduced the first partitioning algorithm to divide the existing stations into a group of zones. An analytical model was proposed to examine the performance of an AGV working in a loop that would later form a zone in a tandem configuration. Furthermore, they used a simulation model to compare the Tandem and Conventional Systems. They considered AGV utilization, average output queue of stations, and the average time spent in system by the jobs. Two empty vehicle dispatching rules for the tandem AGV system were implemented. These are first-encountered-first-served (FEFS) rule and the shortest-time-to-travel-first (STTF) rule. It was proved that the STTF gives better results. They concluded that for small systems with three or four vehicles, the tandem configuration is emerging as a strong competitor for conventional layouts. As for larger systems, a tandem AGV system with six vehicles outperformed a conventional system with eight vehicles.

Lin et al. (1994) presented the load routing problem (LRP) and proposed a two-phase solution to deal with the limitations of Tandem AGV systems. Phase one consisted of a procedure to obtain an initial load routine decision. The results estimated by phase one were used in the second simulation phase. Vehicle utilizations, queue lengths of stations, and total loaded-vehicle travel time were checked in the second phase. The purpose of phase two was to verify that the estimated routine satisfies system conditions.

Researchers were interested in comparing the performance of tandem systems and conventional systems under different working conditions and using different techniques. Choi et al. (1994) tested the two systems using the following variables: number of vehicles needed, AGV speed, job-assignment rule and job-arrival distribution. It was concluded that the vehicle speed is the most important and effective factor.

Wang and Hafeez (1994) used generalized stochastic Petri nets to compare the performance of tandem and conventional systems. The tandem configuration always showed better results in terms of throughput, while the AGV utilization was nearly the same in both systems. Kim and Klein (1996) considered the problem of locating P/D points for a given flow path. They formulated a quadratic assignment problem to address this issue, and solved it with the use of heuristics.

Although tandem AGV systems offer flexibility and simplicity, the additional pickup or delivery points and the conveyors connecting the P/D points generally increase the cost and floor space requirements of the system. To solve this problem, Ross et al. (1996) proposed the tandem/loop configuration. An inner loop, in which an extra AGV is responsible for all handling processes between zones, is used to deal with the limitation of AGV systems. The routing congestion between zones is reduced by virtue of the inner loop.

Huang (1997) proposed the idea of the transportation center to solve the LRP as well as the problems of additional transfer points and increased floor space requirements. First, the optimal transfer point has to be found for each zone with the assumption of unidirectional guide paths. Then, all transfer points are connected by several bi-directional tracks to act as a transportation centre. The locations of the transfer points are obtained through the use of an analytical model.

Bozer and Lee (2004) introduced an idea of using existing workstations as transfer points between zones in order to reduce the cost and floor space caused by additional P/D points and the conveyors connecting the P/D points. The transfer stations must be accessible by both vehicles working in the adjacent zones.

Since the tandem AGV systems are vulnerable to vehicle breakdowns, Chuang and Heim (1996) introduced a concept of real-time loop reconfiguration (RTLR) to respond to single-vehicle failures. Alternative reconfiguration guide paths that connect two adjacent zones are predefined, so that an AGV can access its nearest adjacent zone if the AGV in that zone breaks down. Ventura and Lee (2001) proposed a tandem configuration with more than one AGV per zone to solve the problem of zone inaccessibility in case of vehicle failure.

Hsieh and Sha (1996) proposed the idea of designing machine layout and AGV guide path configurations concurrently in tandem systems. Their proposal addressed the problem of partitioning the stations into a set of zones. Based on the concept of variable path routing within a zone, Yu and Egbelu (2001) developed a heuristic partitioning algorithm for tandem systems.

Kim et al. (2003) introduced some new ideas on the matter of partitioning the stations into a set of zones. They proposed a model for designing tandem systems based on the idea of multi-loaded vehicles. They carried out their research based on the first-encountered-first-served (FEFS) empty-vehicle dispatching policy in a unidirectional guide-path system. Ho and Hsien (2004) proposed an algorithm for designing unidirectional tandem AGV systems. Their model was based on multi-loaded vehicles and took different load-carrying capacities into consideration.

Shalaby et al. (2006) proposed a 0-1 integer mathematical model for designing tandem AGV systems, which served a number of objectives. Laporte et al. (2006)

proposed a tabu search (TS) algorithm to partition the stations into zones, thereby minimizing the maximum workload in the system.

2.3 Key Approaches in Tandem Systems

Three approaches in the literature are considered very important to the current research. They are the idea of zone-generation introduced by Bozer and Srinivasan (1992), the heuristic approach proposed by Yu and Egbelu (2001), and the model generated by Shalaby et al. in 2006. These approaches will be discussed below in more detail.

Three partitioning stages were included in the algorithm proposed by Bozer and Srinivasan (1992) to minimize the maximum workload in the tandem AGV systems. First, subsets of the existing workstations are generated to act as a solution space. The Euclidean traveling salesman problems are solved for all stations in the system. Subsequently, the band technique was employed to select a group of candidate zones for use in the final solution. The following steps are required to apply the band technique: First, all the stations are sorted in an ascending order according to the value of their x-coordinate. Second, the same procedure is repeated in the y-direction. Third, more sequences are obtained by dividing the stations horizontally between an upper band and a lower band with equal widths. After that, the first step is used again to obtain two sequences, one for the upper band and another for the lower. The above steps are repeated with dividing vertically to obtain other sequences. The obtained sequences using this generation algorithm are then used to generate the candidate zones.

Second, the candidate zones are checked for feasibility. Vehicles are assumed to be bi-directional and follow the first-encountered-first-served (FEFS) empty vehicle dispatching policy. Under this policy, a vehicle that has just delivered a job to the input queue of a station will continue traveling empty until the first station with request is encountered. The vehicle follows a pre-defined route and inspects the output queue of each station and transfer point in the zone. The vehicle is either traveling loaded or traveling empty until it encounters a waiting job. Last, the partitioning problem is solved using a 0-1 integer programming model with an objective of minimizing the maximum workload.

Yu and Egbelu (2001) developed a heuristic partitioning algorithm for tandem systems based on the concept of variable path routing within a zone. A unidirectional conventional layout was divided into sub-networks to form the tandem system. A vehicle moves along the shortest path within a zone among stations. The objective is to minimize the number of zones for the given existing conventional network. The locations of transfer points are decided on during the partitioning process.

At the beginning of the algorithm, the station that shares borders with the minimum number of stations is selected as the seed station. If ties exist, the authors select the station with maximum flow with other stations in the system. After the first zone is formed, the seed station can be selected from those stations whose borders include the existing transfer points.

The transfer points are selected from the intersections in the conventional network that have the largest number of unassigned stations. The transfer points are assumed to handle flow between zones directly in their work. Each time when the zone is expanded, the location of the transfer point is updated. All unassigned stations that share the same borders with stations already included in the zone are considered to expand the zone. The authors select a station which has the maximum flow with other stations in the zone to expand the zone. They also try to avoid violating the throughput capacity of the vehicle by adding the station into the zone.

Shalaby et al. (2006) proposed a partitioning algorithm for designing tandem systems which serves a number of different objectives: minimizing the total handling cost, minimizing the maximum workload in the system and minimizing the number of trips between zones. The shortest-travel-time-to-serve-first (STTF) empty vehicle dispatching rule is applied in this model.

The algorithm includes two phases. The first phase involves the selection of a pair of seed stations out of all the stations in the system. These stations are then used to form a candidate zone. This generated zone is subsequently checked for integrity and feasibility to avoid the overlapping zones and to ensure that a single vehicle can bear all the workload imposed on that zone. The utilization of the vehicle is estimated to determine its ability to bear the workload. Then, all stations not in present in the current zone are evaluated based on their relationship with the formed zone. The relationship is a weighted combination of distance quantity and flow quantity. A station with a strongest relationship factor is selected for addition to the existing zone. The workload is expressed by two portions of time needed to perform all the tasks required in a zone. The first portion includes the time spent on loaded trips between stations and transfer points within a zone. The time spent on empty trips made in response to waiting jobs in output queues within that zone forms the second portion. The second phase of the algorithm consists of calculating the number of trips between zones and estimating the material handling cost for each zone. Using a selection mathematical model, these two components and the previously estimated workload are considered as objective function coefficients. Thus, the best combination of zones is selected.

2.4 Dispatching Rules of AGVs

Egbelu and Tanchoco (1984) tested the performance of some of the vehicle initiated dispatching rules and they concluded that the rules based on distance measures have some drawbacks if layout conditions are not met.

Bozer and Srinivasan (1992) implemented two dispatching policies for the tandem

AGV system, FEFS and STTF. It was proved that the STTF gave better results. They concluded that for small systems having three or four vehicles the tandem system was emerging as strong competitive for conventional systems. Two years later, they developed the modified first-come-first-served (MFCFS) rule. Under their assumptions, an empty vehicle first inspects if loads are available for transport at the station the AGV just delivered its previous job. If the loads are available, the AGV starts transporting the first in line at this station. Otherwise, it is dispatched to the oldest unassigned request. They found out that the performance of this rule is nearly as good as the performance of the shortest-travel-time-first (STTF) rule.

Kodali (1997) presented a knowledge-based system for selecting an AGV and selecting a work center requesting transport simultaneously. Jeong and Randhawa (2001) developed a multi-attribute dispatching rule which simultaneously considers three criteria. These are the unloaded travel distance of an AGV to the pickup point, the remaining space in the input buffer of a delivery point, and the remaining space in the outgoing buffer of a pickup point. Neural network approach was applied in their study and the weights of the criteria are continuously changing. It was proved that this rule performs better than single attribute rules or a multi-attribute rule with fixed weights.

Naso and Turchiano (2005) described an approach for AGV dispatching in flexible production environments combining different Computational Intelligence (CI) tools with the objectives of maximizing the throughput, minimizing the utilization of AGVs, and minimizing the occurrence of blocking. Bilge et al. (2006) proposed a dispatching strategy with two criteria, which are travel time and output buffer length. Simulation experiments under various scenarios were conducted to compare the performances of the proposed dispatching rules with STTF and MROQS.

2.5 Summary

Many researchers in the past have proposed their approaches in designing either the conventional or the tandem configuration of AGV material handling systems. In general, the conventional systems offer more flexibility but require more complicated control systems to reduce the conflicts that occur due to the vehicles interactions. The studies that compare the performance of both systems have shown that the tandem systems outperform the conventional systems in many different ways. The attention towards the design of tandem systems has been attracted due to the above reasons.

As for the partitioning of tandem systems problem, some points are summarized as follows:

1. The STTF empty vehicle dispatching policy was only applied once in the partitioning process in Shalaby et al. (2006)'s work. Although Bozer and Srinivasan showed by simulation results that applying the STTF rule gave a better system performance than the FEFS rule, not many researchers have

considered this dispatching polity in the partitioning process.

2. It can be noted from the literature that most studies have developed heuristic algorithms to design the tandem configuration. One application of meta-heuristic algorithms in this area is the study of Aarab et al. (1999) who used tabu search (TS) as a subroutine in designing the tandem system. They designed unidirectional single loop flow paths in a given facility layout which is organized into departments based on their geometric shapes and defined boundaries. These departments are later partitioned into different zones. Although TS was applied in this study, it was not considered as the main algorithm of designing the tandem systems. Instead, it was used to improve the initial solutions of generating the single loop. Laporte et al. (2006) applied TS in the tandem system partitioning problem. The mathematical model generated by Bozer and Srinivasan (1992) was adopted in their work. However, no improvement can be found in their results compared with Bozer and Srinivasan's.

3. The presence of overlapping zones is considered as an infeasible condition in all the studies. However, a specific mechanism has not been developed to check this condition.

In this thesis, new algorithms of partitioning the tandem AGV systems are proposed based on meta-heuristic approaches. In addition, simulation is applied to analyze the performance of several different AGV dispatching rules.

CHAPER 3

OPTIMIZING THE PARTITIONING OF TANDEM AGV SYSTEMS USING GENETIC ALGORITHM

The genetic algorithm (GA) was invented by Holland in the 1960s. A GA is a directed random search technique, which can find the global optimal solution in complex multi-dimensional search spaces. A GA is modeled on natural evolution in that the operators it employs are inspired by the natural evolution process (Pham et al. 2000). It works based on the ideas of natural selection and genetic evolution. Instead of searching from solution to solution, genetic algorithms provide a search from population to population. By carrying out the genetic operations including selection, crossover and mutation on the current population, a new population is generated and eventually a population containing the optima or close to optima will be evolved. In other words, the fittest individuals are ensured to survive in the evolution process.

In recent decades, the GA has become one of the most widely used approaches for combinatorial optimization problems. In this chapter, a genetic algorithm (GA) has been developed to solve the partitioning problem of tandem AGV systems. The objective is to minimize the maximum workload to balance the system. Three different types of infeasible conditions are discussed and a repair procedure is developed to deal with them. The performance of the proposed algorithm is compared with the reported results in literature using a group of benchmark problems.

3.1 The Proposed GA

A flowchart of the proposed genetic algorithm is shown in Figure 3-1. These steps are explained in the following sections. The k-means clustering method (MacQueen, 1967) is used to generate the initial population (solutions). Selection will choose the fittest individuals from among the current population to survive. GA operators are employed to propagate the populations from one generation to another to find the optimum solution. The first operator is crossover, which mimics the mating in the biological populations. The second operator is mutation, which maintains the diversity of populations. These operators allow a global search over the entire design space to avoid being trapped in local optima. Since constraints need to be considered in this problem, a repair procedure is developed to restore the infeasible solutions resulting from crossover and mutation operations.

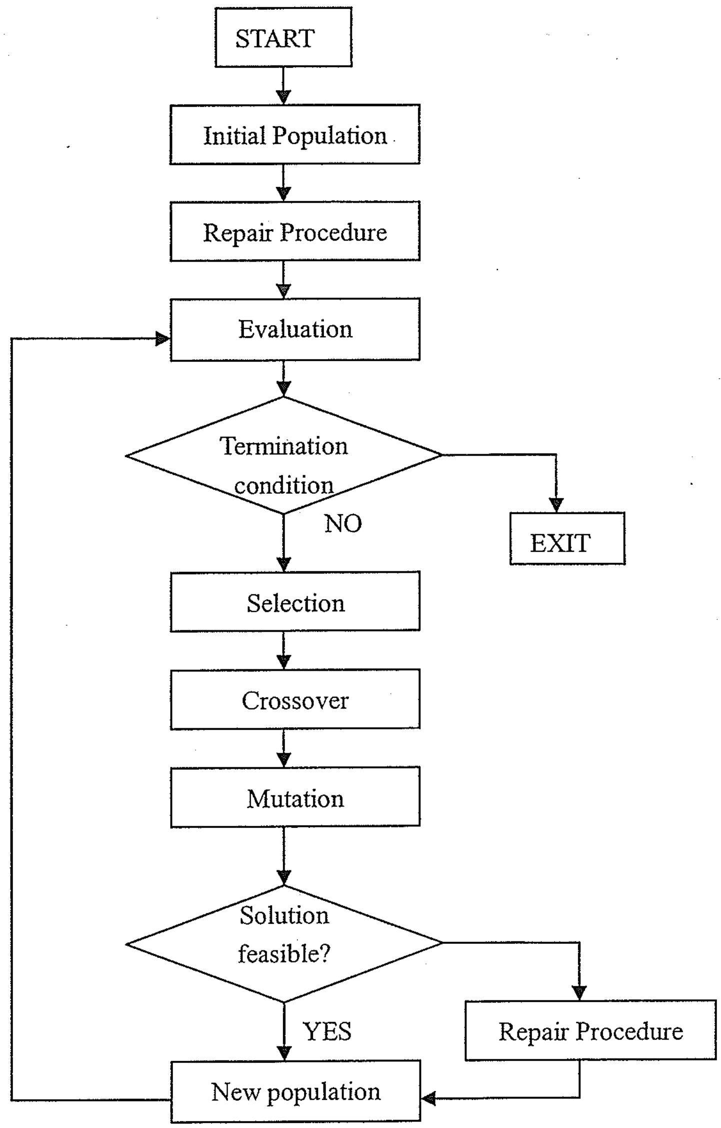


Figure 3-1: The flowchart of the proposed genetic algorithm

3.1.1 Chromosome Representation

Genetic algorithms works on two types of spaces: the coding space and the solution space. The genetic operators work on the coding space. The evaluation and selection work on the solution space. The link between the chromosomes and the performance of decoded solutions is the natural selection. Encoding a solution of a problem into a chromosome is an important issue when applying a GA. In the proposed algorithm, the length of the chromosome indicates the number of stations to be considered in the design problem. A station i can be represented by the position of the gene (x_i) in the chromosome. Also, the value of each gene (x_i) indicates the zone number to which station i is assigned. It is assumed that the number of zones is decided beforehand. An example of the chromosome which represents a solution is shown as follows:

1	2	3	4	5	6	7	8
0	3	0	2	2	1	3	1

Figure 3-2: An example of a chromosome representation

This example shows that there are 8 stations to be partitioned into 4 zones in the system. Stations 1 and 3 are assigned to zone 0, stations 6 and 8 to zone 1, stations 4 and 5 to zone 2, and stations 2 and 7 to zone 3. This chromosome representation ensures that each station is assigned to a zone. There is an equal probability to assign any given station to any zone. However, there is a possibility that some zones may not have any stations, which leads to fewer zones than planned. Therefore, some problems which are caused by infeasible conditions need to be checked and resolved.

3.1.2 Infeasibility Conditions

There are some situations that might occur during the process of generating solutions that lead to infeasibility conditions. In this chapter, three infeasibility conditions are considered. These are overlapping zones, empty zones, and singleton zones. In the proposed algorithm, when these conditions are encountered, the solutions with overlapping zones or singleton zones are repaired by a generated repair procedure, and the solutions with empty zones are rejected.

In terms of dealing with the infeasible conditions, Bozer and Srinivasan (1992) introduced the idea of generating a large amount of possible zones from which the best zones can be selected via a mathematical model. Yu and Egbelu (2001) introduced the idea of selecting seed zones and expanding them. It is evident from the literature that most authors adopted these ideas. In these heuristic approaches, the situation of overlapping zones would be rejected, the empty zones would not exist and the singleton zones were not considered. Laporte et al. (2006) mentioned checking the overlaps and reducing the number of singleton zones. However, to the best of our knowledge, no specific mechanism for resolving the overlaps among generated zones has been developed in the literature.

The difficulty of identifying and resolving the infeasibility conditions, especially the overlapping zones, is attributed to identifying the AGV routing in each zone. Although rectilinear distances are applied for AGV's moves in a real system, it is quite complex to decide the exact rectilinear routes between P/D points. For the sake of simplicity, Laporte et al. (2006) used the Euclidean traveling salesman problem (TSP) model to solve the routing problem in each zone. Since TSP is itself an NP-hard problem, a Nearest Neighbor method is used to reduce the calculation time in our proposed algorithm. When the routes of the AGV in each zone are decided, the overlaps can be checked by searching intersections between every two lines which are formed by four stations from two different zones. In this case, each line is formed by two stations from the same zone. An overlap is certain to occur if one of the intersections caused by these two lines is on these two segments between the four stations. The difference between overlapping and non-overlapping solutions is shown in Figure 3-3. In the proposed algorithm, this kind of infeasibility can be repaired using the developed repair procedure.

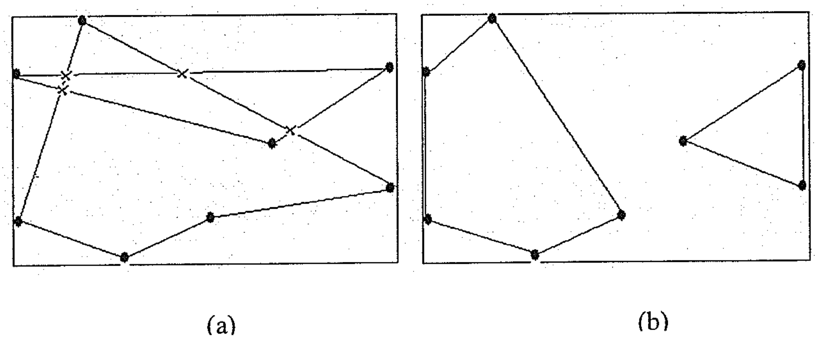


Figure 3-3: An example of overlapping (a) and non-overlapping (b) solutions

The second kind of infeasibility is the empty zones. Empty zones occur when the number of generated zones is less than the designated number, which means that there are zones that contain no stations. Such solutions are rejected. Finally, a zone that contains only one station is called a singleton zone. Solutions with singleton zones are repaired. Although Laporte et al. (2006) mentioned reducing the number of singleton zones, but they did not consider this situation infeasible. They considered the solution which has fewer singleton zones as the best solution even if it has a worse objective function than other solutions. However, in the proposed algorithm, when a singleton zone is found, the nearest station is added to this zone automatically. If another singleton zone is created due to this move, the current move will be cancelled and the next nearest station will be checked. If no stations can be added to the singleton zone or overlaps are caused, the solution is rejected, and hence singleton zones will never exist in the generated solutions. The steps of the repair procedure will be explained in section 3.1.4.

In Laporte et al. (2006), if a solution is generated with an AGV workload greater than 1, it has been considered as an infeasible solution. Since the workload is the proportion of time a vehicle is busy, either loaded or empty, a reasonable value of the workload is always less than 1. However, in the proposed algorithm, this condition is not considered infeasible so that GA is allowed to search a wider area. Gen and Cheng (2000) mentioned that the optimal solutions usually occur at the boundary between the feasible and infeasible areas. In addition, the developed algorithm will always move in a better direction by reducing the AGV workload since the objective is to minimize the maximum workload.

3.1.3 Initial Population

Overlaps between generated zones occur quite often when the initial population is generated randomly. Although they can be repaired using a repair procedure, this is considered to be time-consuming. To avoid this problem, the initial population has to be generated using a more intelligent method than random generation.

A k-means clustering method is adopted in this study in order to avoid overlaps among the generated zones. A group of stations are selected arbitrarily as the centers of the zones. This ensures that at least one station is assigned to each zone. In other words, empty zones are not generated in the initial population. Then, the distances to the centers of all stations not present in these zones are calculated. Each station is then added to the zone with the nearest center to it. For more information about this k-means clustering method, the reader is referred to MacQueen (1967). It has to be emphasized that the locations of the zones' centers are updated every time a station is added. Although overlapping and empty-zones problems do not need to be considered in the initial population, it is possible to generate singleton zones. A solution which has singleton zones is considered an infeasible solution. Therefore, the repair procedure still has to be applied to the initial population to deal with this problem.

3.1.4 Repair Procedure

As mentioned earlier in section 2.2, three conditions are considered infeasible, namely, the empty zones, singleton zones, and overlapping zones. A solution will be rejected if it has empty zones. However, to expand the search space, a repair procedure is developed to repair the other two kinds of infeasible solutions. The repair procedure first recognizes the kind of infeasibility condition. If the infeasibility condition is the singleton zones, the nearest station is added to the zone, provided that no more singleton zones are generated by this adjustment. On the other hand, if the infeasibility is caused by overlaps among zones, the k-means clustering method is applied to repair it. The centers of the zones can be determined based on the original infeasible solution at the beginning, then updated after every move. This procedure continues clustering the stations until the positions of the centers remain the same after updating. The flowchart of the repair procedure is shown in Figure 3-4.

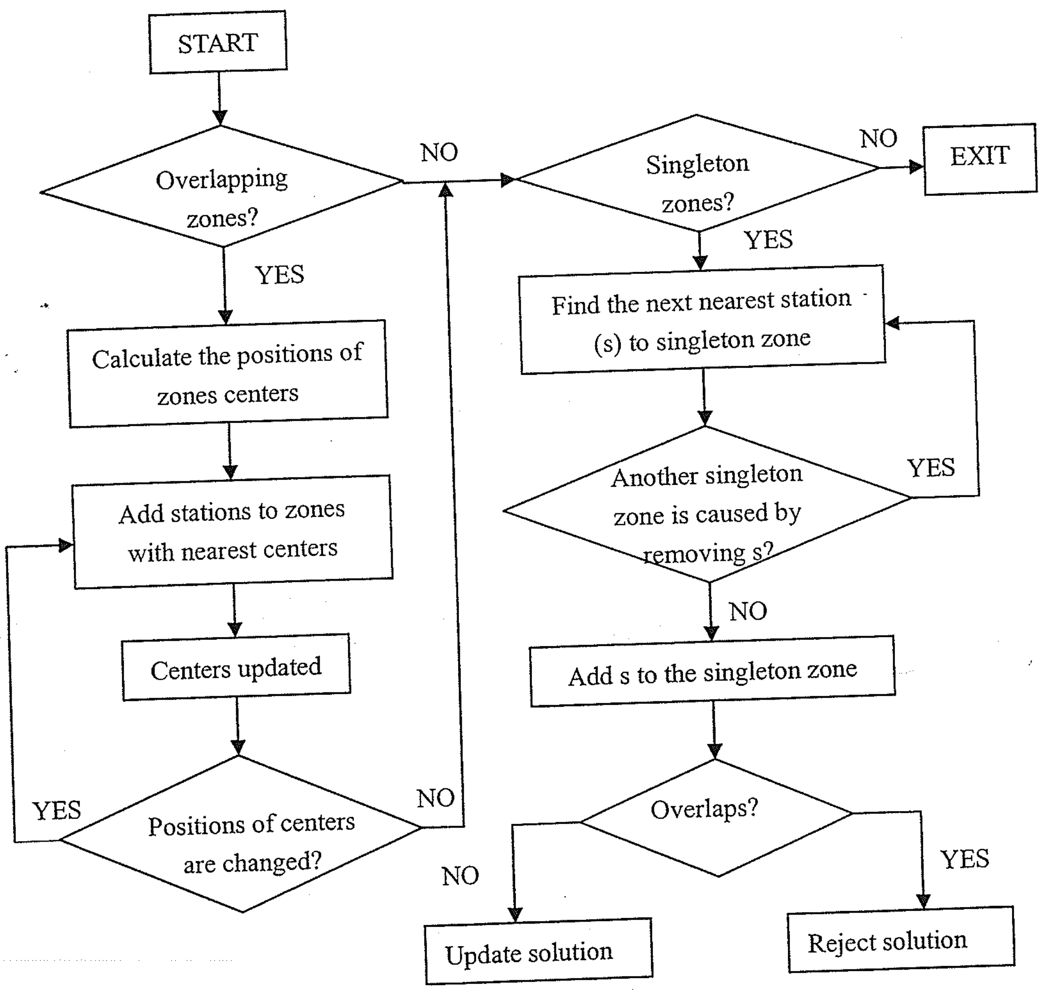


Figure 3-4: The flowchart of the repair procedure

Steps of the repair procedure:

STEP 1: If there is infeasibility caused by overlapping zones, go to STEP 2, else go to STEP 6.

STEP 2: Calculate the positions of the zones' centers.

STEP 3: Add stations to the zones with the nearest centers.

STEP 4: Update centers of zones.

STEP 5: If the locations of the zones' centers have changed, go to STEP 3, else go to STEP 6.

STEP 6: If there is infeasibility caused by singleton zones, go to STEP 7, else EXIT.

STEP 7: Find the nearest unchecked station to the station in the singleton zone.

STEP 8: If adding this station to the singleton zone will result in a new singleton zone, go to STEP 7, else go to STEP 9.

STEP 9: Add the station to the singleton zone.

STEP 10: If there are any overlapping zones caused by the previous move, reject the solution, otherwise accept the move and update the solution.

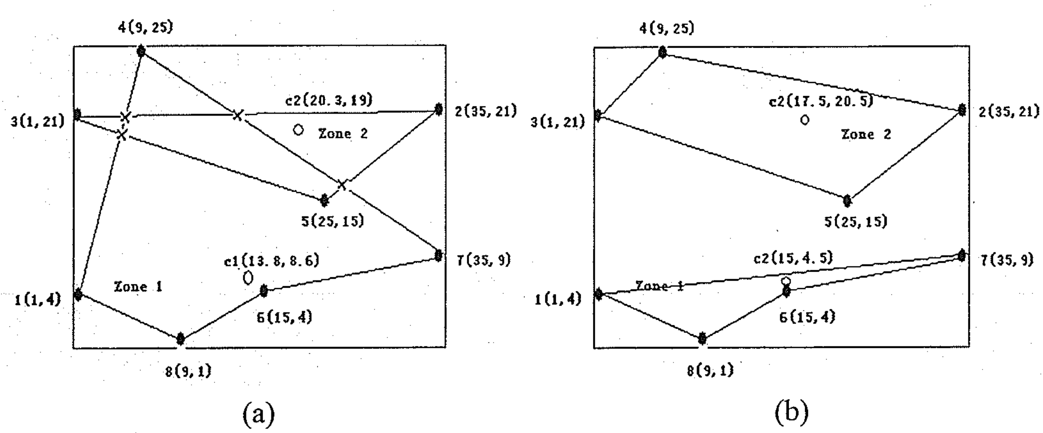


Figure 3-5: A numeric example of the repair procedure

A numeric example is given to illustrate the repair procedure. In the example shown in Figure 3-5 (a), there is an overlap between zones 1 and 2. The center positions of these two zones are calculated as c1 and c2. The rectilinear distances between each station to the centers are listed in Table 3-1.

Table 3-1: Distance between stations and zones' centers in the numeric example

	Station 1	Station 2	Station 3	Station 4	Station 5	Station 6	Station 7	Station 8
C1	17.4	33.6	25.2	21.2	17.6	5.8	21.6	12.4
C2	34.3	16.7	21.3	17.3	8.7	20.3	24.7	29.3

From Table 3-1, it is clear that all the stations are closer to their own center than the other center except station 4. Therefore, station 4 has to be removed from zone 1 and added to zone 2. The new zones are formed and the centers of the zones are updated as shown in Figure 3-5(b). Currently, the nearest center to all the stations is in their own zones, so that no more moves are required and the zones' centers remain at the same positions. Until this step, the overlapping problem is solved.

3.1.5 Fitness Function

The model introduced by Shalaby et al. (2006) is used in the proposed algorithm to estimate the fitness function of a generated solution. The workload (η_p) of zone p can be divided into two proportions. The first one is the time proportion that the AGV spends in the loaded trips (α_p). The second one is the time proportion that the AGV spends in the empty trips which are made to respond to waiting jobs in the output queues of the stations or transfer points (φ_p). The value of α_p can be calculated as follows:

$$\alpha_p = \frac{\sum_{i\in p}\sum_{j\in p} f_{ij} \times (2T + \frac{d_{i,j}}{S}) + \sum_{i\in p}\sum_{j} f_{ij} \times (2T + \frac{d_{i,tp}}{S}) + \sum_{j}\sum_{i\in p} f_{ji} \times (2T + \frac{d_{i,tp}}{S})}{60} \tag{1}$$

Where:

- f_{ij} = flow between station i and station j per unit time (hour)
- $d_{i,j}$ = rectilinear distance between station i and station j
- $d_{i,tp}$ = rectilinear distance between station i and transfer point t of zone p
- S is the AGV speed (unit distance/ min).
- T is the pickup/drop-off time (min).

The value of φ_p can be estimated by a probabilistic approach:

$$P_{ij} = \alpha_p \times (1 - \sum_{k\in K} P(W)_k) \times P(W)_j + (1 - \frac{3}{2}\alpha_p) \times P(W)_j \tag{2}$$

$$E_{ij} = E_i. \times P_{ij} \tag{3}$$

$$\phi_p = \frac{\sum_i \sum_j E_{ij} \times \frac{d_{i,j}}{S}}{60} ....\forall i, j \in Z \tag{4}$$

Where:

- P_{ij} = the probability of assigning an empty trip from station i to station j.
- $P(W)_i$ = the probability that station i has a waiting job in its output queue.
- K is the set of stations or transfer points in zone p closer to station i than station j.

- E_i is the number of empty trips emerging from a station i.

- E_{ij} is the number of empty trips assigned from station i to station j.

It is assumed that the probability of a certain station i having a waiting job in its output queue $P(W)_i$ is equal to the ratio of the loaded trips sent from station i to the total number of loaded trips sent from all the stations in the zone.

The workload (η_p) of zone p can be obtained:

$$\eta_p = \alpha_p + \varphi_p \tag{5}$$

Using this formulation, the workload of each zone can be estimated, and the maximum workload is selected from each individual (solution) as the fitness value. The objective is thus to find the minimum fitness to balance the system.

3.1.6 Selection

A mixed selection approach based on the roulette wheel and elitist selection is adopted in the algorithm. The roulette wheel selection is a traditional selection with the probability of survival equal to the fitness of an individual over the sum of the fitness of all individuals of the population. The probability of selecting a chromosome i is calculated as follows:

$$P_s = f_i \,/ f_{total} \tag{6}$$

Where f_i is the fitness value of chromosome i and f_{total} is the total fitness value of all the chromosomes in the current population.

A random number between 0 and 1 is generated each time to select a chromosome. The elitist selection is employed so that the best individual from the preceding generation is always included in the current generation.

3.1.7 Genetic Operators

A one-cut-point crossover method is used and the crossing point is selected randomly to exchange the right parts of two parents to generate offspring. The crossover rate is the probability of crossover for a chromosome. By multiplying the crossover rate by the population size, the number of crossovers in one generation can be estimated. A random number between 0 and 1 is assigned to each chromosome. A chromosome is selected for crossover if its assigned number is smaller than the designated crossover rate. In this way, each chromosome has equal probability of being selected for crossover. After selecting the parents, a crossing point is chosen randomly in the chromosomes. Then, the right parts of the two parents are exchanged and new children are generated. An example of one-cut-point crossover is shown in Figure 3-6. Crossover is performed on two parents at a randomly picked point, and two children (offspring) are generated.

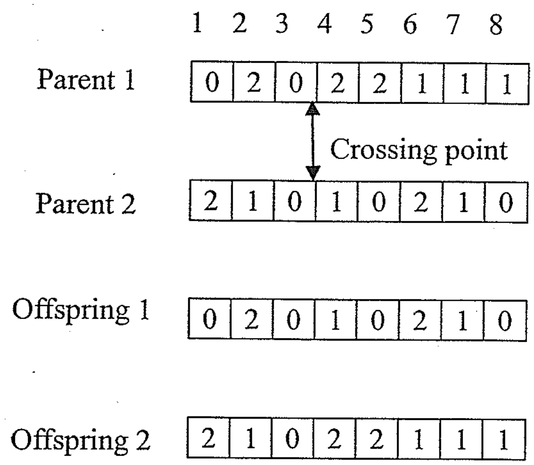


Figure 3-6: An example of one-cut-point crossover

Mutation is another genetic operator. The probability of mutation is defined as the mutation rate. The numbers of mutations in each generation is calculated by multiplying the mutation rate by the total number of genes in the whole population. A sequence of random numbers between 0 and 1 is generated and assigned to each gene. Mutation is carried out on a gene when this random number is smaller than the mutation rate. In the proposed algorithm, to conduct mutation, an integer number is generated randomly within the range [0, total number of zones -1]. The gene chosen for mutation is then replaced by this integer number. For instance, in a 3-zones system, one of the chromosomes is 02022111 and the second gene is selected for mutation. A random integer number 1 is generated between 0 and 2. The second gene, 2, is replaced by 1 in the procedure of mutation. The chromosome becomes: 01022111 after mutation.

3.1.8 Termination Criterion

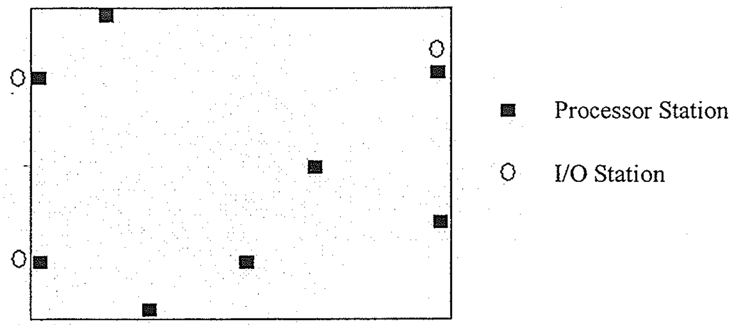
The termination criterion of the proposed algorithm is reached when the number of iterations equals 50 since the best solution has changed. In other words, when 50 same best solutions are found, they are considered to be the optimal solutions.

3.2 Analysis and Comparison of the Proposed Algorithm

The proposed algorithm was coded in Microsoft Visual Studio using C language on a 2.80 GHz Intel Pentium 4 with 512 MB RAM system. The performance of the proposed algorithm is tested and compared with some benchmark problems with the objective of minimizing the maximum workload in the system. The chosen benchmark problems are 8-stations problem, 10-stations problem, and 20-stations problem. The description of these benchmark problems and the results comparison are given in the following sections.

3.2.1 8-stations Problem

The 8-stations problem was solved by Bozer and Srinivasan (1992), and Laporte et al. (2006). 4 zones are designated in this problem. The empty or loaded vehicle speed is 15 units / min, the pickup or drop-off time for each load is 0.2 min and the STTP vehicle dispatching rule is used. The coordinates of the stations, the job sequences and the production rate are shown in Figure 3-7. The comparison of the results is given in Table 3-2. It is clear that the proposed algorithm reaches the same results as Bozer and Srinivasan's and Laporte et al.'s algorithms.



Stations	1	2	3	4	5	6	7	8
Coordinates	(1,4)	(35,21)	(1,21)	(9,25)	(25,15)	(15,4)	(35,9)	(9,1)

Job Type	Process Route	Production Rate/Hr.
A	1-4-5-7-1	1.5
B	3-4-6-1	1.5
C	1-7-5-4-2	3.0
D	3-4-5-6-8-1	3.0

Figure 3-7: Input data of the 8-stations problem (Bozer and Srinivasan, 1992)

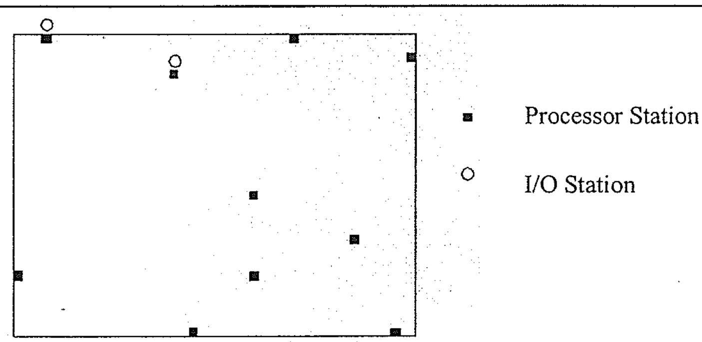
Table 3-2: 8-stations problem (proposed algorithm vs. Bozer and Srinivasan)

Proposed Algorithm		Bozer and Srinivasan	
Resulting Zones	Estimated Workload	Resulting Zones	Estimated Workload
1,8	0.2983	1,8	0.2983
2,5	0.3900	2,5	0.3900
3,4	0.3750	3,4	0.3750
6,7	**0.4417***	6,7	**0.4417***

* Maximum workload

3.2.2 10-stations Problem

A 10-stations tandem AGV system designing problem solved by both by Yu and Egbelu (2001) and Shalaby et al. (2006) is solved by the proposed algorithm. The 10 existing stations are partitioned into 4 non-overlapping zones; the loaded or empty vehicle speed is 60m/min; the pickup or drop-off time for each load is 0.25 min; the maximum capacity per vehicle in the system is 60min/hr, which implies that 100% utilization per vehicle is permitted. The STTP vehicle dispatching rule is used. The coordinates of the stations, the job sequences and the production rate are shown in Figure 3-8. The results, as shown in Tables 3-3, are compared with Yu and Egbelu (2001) and Shalaby et al. (2006).



Stations	1	2	3	4	5
Coordinates	(8,75)	(40,65)	(70,75)	(100,70)	(60,35)
Stations	6	7	8	9	10
Coordinates	(85,25)	(1,15)	(45,1)	(60,15)	(95,1)

Job Type	Process Route	Production Rate/Hr.
A	1-3-6-10-9-2	3.0
B	1-7-10-8-1	3.0
C	1-5-6-7-3-2	3.0
D	1-9-4-7-1	3.0

Figure 3-8: Input data of the 10-stations problem (Yu and Egbelu, 2001)

Table 3-3: 10-stations problem (proposed algorithm vs. Yu and Egbelu and Shalaby et al.)

Proposed Algorithm		Yu and Egbelu		Shalaby et al.	
Resulting Zones	Estimated Workload	Resulting Zones	Workload	Resulting Zones	Estimated Workload
1,3	0.5550	1,7	**0.8083***	1,2	0.4942
2,4,5,6	**0.6058***	2,5	0.2683	3,4,5	0.4079
7,8	0.6008	3,4	0.2642	7,8	0.6008
9,10	0.5433	6,8,9,10	0.7886	6,9,10	**0.7085***
Max Difference	0.0625		0.5441		0.3006

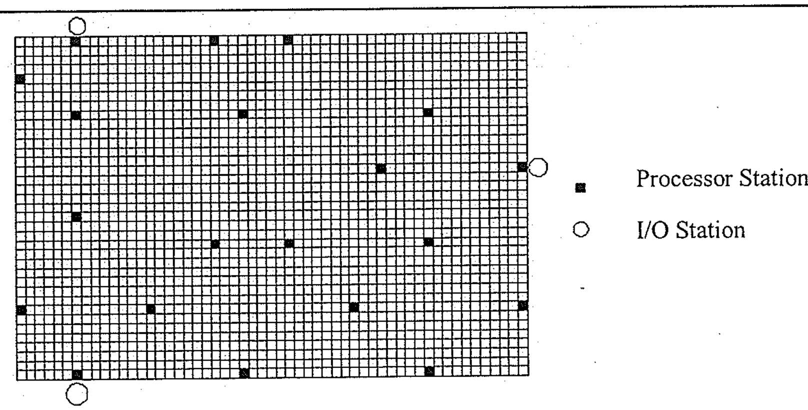
* Maximum workload

According to Table 3-3, the maximum workload is reduced to 0.6058 using the proposed algorithm from Yu and Egbelu's and Shalaby et al.'s algorithms. The balance of workload among different zones is also better using the proposed algorithm.

3.2.3 20-stations Problem

This problem can be divided into two problems considering two different numbers of zones into which the stations can be partitioned. The first one is a 20-stations (6-zones) problem (a), which was presented and solved by Bozer and Srinivasan (1992) and then solved by Shalaby et al. (2006), and Laporte et al. (2006). The second one can be seen as a 20-stations (4-zones) problem (b). It was solved by Yu and Egbelu (2001) to minimize the number of zones. Their algorithm was applied to the same problem data, but 4-zones were considered instead of 6-zones. Shalaby et al. (2006) have also solved the second problem. The assumptions are made as follows:

the empty or loaded vehicle speed is 15units/min; the pickup or drop-off time for each load is 0.2 min; and the STTP vehicle dispatching rule is used. The coordinates of the stations, the job sequences and the production rate are shown in Figure 3-9. The comparison of results for problem (a) and problem (b) is shown in Tables 3-4 and 3-5, respectively. It can be seen that the proposed algorithm performs better in both of the 6-zones and 4-zones problems in the 20-stations system.



Stations	1	2	3	4	5	6	7
Coordinates	(7,37)	(22,37)	(30,37)	(1,33)	(7,29)	(25,29)	(45,29)
Stations	8	9	10	11	12	13	14
Coordinates	(37,23)	(55,23)	(7,16)	(22,16)	(30,16)	(45,16)	(1,7)
Stations	15	16	17	18	19	20	
Coordinates	(16,7)	(37,7)	(55,7)	(7,1)	(25,1)	(45,1)	

Job Type	Process Route	Production Rate/Hr.
A	1-3-6-2-5-4-1	1.5
B	1-6-8-7-9	1.5
C	9-7-8-16-20-17-13-9	1.5
D	18-15-11-12-16-13-17-9	1.5
E	18-15-19-12-11-10-14-18	1.5
F	18-14-10-4-5-1	1.5

Figure 3-9: Input data of the 20-stations problem (Bozer and Srinivasan, 1992)

Table 3-4: 20-stations (6-zones) problem (proposed algorithm vs. Bozer and Srinivasan and Shalaby et al.)

Proposed Algorithm		Bozer and Srinivasan / Laporte et al.		Shalaby et al.	
Resulting Zones	Estimated Workload	Resulting Zones	Estimated Workload	Resulting Zones	Estimated Workload
1,2,4,5	0.3135	1,2,3,6	0.3583	1,2,4,5	0.3135
3,6,10	0.3567	4,5,10,14	**0.4971***	3,6,12	0.3867
7,8,9	0.3477	7,8,9	0.3460	7,8,9	0.3460
13,17,20	0.2938	11,12,13	0.3800	10,14,18	0.2893
11,12,16,19	**0.3746***	15,18,19	0.2913	11,15,19	0.3233
14,15,18	0.3205	16,17,20	0.2967	13,16,17,20	**0.4088***
Max Difference	0.0812		0.2058		0.1195

* Maximum workload

Table 3-5: 20-stations (4-zones) problem (Proposed algorithm vs. Yu and Egbelu and Shalaby et al.)

Proposed Algorithm		Yu and Egbelu		Shalaby et al.	
Resulting Zones	Estimated Workload	Resulting Zones	Estimated Workload	Resulting Zones	Estimated Workload
1,2,3,4,5,6	**0.5349***	1,2,3,4,5,6	0.5349	1,2,3,4,5,6	0.5349
7,9,13,17	0.4824	7,8,9,13,17	**0.5530***	7,8,11,12	**0.5985***
8,12,16,19,20	0.5325	10,11,14,15,18	0.4795	9,13,16,17,20	0.5234
10,11,14,15,18	0.4850	12,16,19,20	0.3676	10,14,15,18,19	0.4772
Max Difference	0.0525		0.1854		0.1213

* Maximum workload

3.2.4 Analysis of the results

The comparisons of results are given in Tables 3-2, 3-3, 3-4 and 3-5. It can be noticed that the proposed algorithm obtained the same results as Bozer and Srinivasan (1992) when the 8-stations problem was solved. As for the 10-stations problem, the maximum workload obtained by the proposed algorithm is 0.6058 which is lower than the ones obtained by Yu and Egbelu (2001) and Shalaby et al (2006) algorithms (0.8083, 0.7085). For the 20-stations problem with 6-zones, the maximum workload obtained is 0.3746 which is again better than the ones obtained by Bozer and Srinivasan (1992) and Shalaby et al. (2006) algorithms (0.4971, 0.4088). Again with 4-zones, the proposed algorithm outperformed the results of Yu and Egbelu (2001), and Shalaby et al. (2006).

It can be noticed from tables 3-3, 3-4, and 3-5 that the proposed algorithm not only outperforms the previous reported results based on the obtained mini-max workload, but also it generates better balance of the workload among all zones. In other words, the differences of workloads among the different zones are smaller. This is quite important because systems with worse balance will have a higher chance of causing bottlenecks.

Laporte et al. (2006) solved the 8-stations problem and the 20-stations (6-zones) problem. Although the details of the results were not given in their paper, they reported the CPU times to be 12.2 seconds and 129.5 seconds respectively using a 2.00 GHz Intel Pentium 4 with 256 MB RAM system. In this thesis, the average CPU times of the proposed algorithm for solving these two problems are 0.49 seconds and 24.6 seconds respectively. It is worth mentioning that the CPU time has not been reported in the other papers [Yu and Egbelu (2001), and Shalaby et al. (2006)].

It has to be mentioned that for the comparison, the workloads of other authors' results shown in Table 3-2, 3-3, 3-4 and 3-5 are obtained from the output of our proposed algorithm instead of the ones from their original papers since minor difference in assumptions might cause different solutions.

CHAPTER 4

DESIGN AND ANALYSIS OF EXPERIMENTS

In the previous chapter, the proposed algorithm was compared with several other algorithms based on their published results. However, only a few benchmark problems in this particular area are available. The literature of tandem AGV systems design is in need of well structured benchmark problems. Although Laporte et al. (2006) generated some random problems for the tandem configuration systems. However, due to the lack of providing clear data in their paper, it is not possible to use these problems as benchmark problems for future comparison. Therefore, a set of tandem configuration experiments are designed in this chapter.

Moreover, the difficulty of applying the GA to a practical problem is tuning up the parameters such as population size, crossover rate and mutation rate. The performance of GA is strongly affected by these strategy parameters. Therefore, developing a method of choosing better values for these parameters is very important so that the GA will perform more efficiently.

There are trials in literature to find optimal parameters of evolutionary algorithms (EA) since the performance of these algorithms is significantly affected by setting their parameters in terms of convergence towards optimum solution and search effort. However, there has not been enough work to establish a systematic method to specify the values of EA's parameters that result in the best performance. Grefenstette (1986) searched for the optimal parameters of GAs for a set of numerical optimization problems using experiments. Davis (1993) demonstrated the delay of convergence caused by the difference between using randomly picked parameter values rather than optimum values. He also introduced an adaptive operator fitness to adjust the parameters based on the feedback from the statistics of the performance of each operator. Srinivas and Patnaik (1994) proposed another adaptive GA. They reported that by assigning low values of crossover rate and mutation rate to high fitness solutions and high values of crossover rate and mutation rate to low fitness solutions, the GA works better than the one with constant crossover and mutation rate for multi-modal problems. Bagchi and Deb (1996) used design of experiments (DOE) to tune up the parameters of GA and they showed the effectiveness of DOE approach in selecting GA's parameters. Rojas et al. (2002) investigated the relevancy and relative importance of parameters involved in GA design by using analysis of variance (ANOVA). Saremi et al. (2007) proposed a memetic algorithm (MA) which is a GA combined with local search methods for solving the vehicle routing problems. DOE is used in their work for tuning the parameters of MA.

Since the parameter values may be problem-specific, instead of only considering the effects of the parameters of GA, the effects of the system characteristics factors of the problem are also considered in this work. In this chapter, design of experiments (DOE) is used to analyze the effects of the input factors including the GA's parameters and the system factors on the solution quality and the computational time. The experimental factors are explained in section 4.1 and the procedure of generating the experiments are discussed in section 4.2. The factorial design is conduced in section 4.3. The computational results are analyzed using ANOVA in section 4.4. The results of the designed experiments using the proposed algorithm are also reported in this section.

4.1 Experimental Factors

The performance of a GA is sensitive to the choice of its parameters, namely, population size, crossover rate and mutation rate. To find the best combination of these factors in different system's characteristics, design of experiments (DOE) has been carried out in this study.

An experiment can be defined as a test or series of tests in which purposeful changes are made to the input variables of a process or system so that we may observe and identify the reasons for changes that may be observed in the output response (Montgomery, 1997). An important feature of design of experiment is the consideration of interaction effect between studied factors rather than considering the effect of individual factors only. This has crucial importance when developing an evolutionary algorithm because the efficacy of an algorithm is rarely dominated by a single factor.

The factors in an experiment include the input variables and the output response. To design experiments in tandem AGV systems, the objective function in this research, minimizing the maximum workload, is obvious one of the most important output responses. Computation time is considered as another output response in order to observe the efficiency of the proposed GA.

The input variables in this problem are divided by two categories. The first category is related to the system characteristics which include the system size, the number of zones, and the zone loading factor. The second kind of input variables considered in this study is the GA's parameters, which are the population size, the crossover rate and the mutation rate.

4.1.1 The system size factor

The system size is represented by the number of stations in the system. It is believed in general that the search space of a problem always has a direct effect on the performance of a search technique. The benchmark problems generated by Bozer and Srinivasan (1992) include an 8-stations problem and a 20-stations problem. The latter was adopted by Yu and Egbelu (2001) and a 10-stations problem was generated later by them. In Laporte et al. (2006) have considered five different sizes including 10, 20, 30, 40 and 50 stations. However, due to the lack of providing clear data in their paper, these problems can not be used and solved as benchmark problems for future comparison.

In the current research, a set of problems are generated based on five different system sizes, which are 10, 20, 30, 40 and 50 stations. Although it might not practical to have 40 or 50 work stations in a real factory, it is important to study the performance of the proposed GA under such large system sizes.

4.1.2 The number of zones factor

The number of zones factor is believed to have significant effect on the performance of the algorithm. For the same number of stations, if the required number of zones is small, a large number of stations will be partitioned into one zone, so that the trips made between different zones can be reduced. However, the AGV workload in each zone would be increased. On the other hand, a large number of zones require a small number of stations in each zone, which may cause a lot of between-zones trips while decreasing the AGV workload in each zone. To test the performance of the proposed algorithm under different conditions, a set of reasonable numbers of zones has to be determined based on the system size factor.

Due to the assumptions made earlier, one zone contains two stations at least. On the other hand, an extremely large number of stations in one zone will cause routing problems inside the zone. Based on the idea of achieving a balance of all the system sizes, 2 to 4 stations will be assigned into a zone on average for the large level of number of zones factor, 5 to 7 stations will be assigned into a zone for the small level of zone numbers factor. The designated numbers of zones in different system sizes are shown in Table 4-1.

Table 4-1: The zone numbers designated for the problems

Zone numbers	10 stations	20 stations	30 stations	40 stations	50 stations
LARGE	4	6	9	12	15
SMALL	2	4	5	7	8

4.1.3 The Zone loading factor

The number of jobs, the required number of operations of each job, their sequences, and the inter-arrival rate of the jobs will define the zone loading. In general, larger number of jobs and more operations of each job introduce more flexibility into the manufacturing environment and more complexity into the relationships between stations and between zones.

Since all these effects can be represented by a from-to flow chart, in order to simplify the design of generated benchmark problems, the zone loading is attributed to the jobs inter-arrival rate only. In another word, the information of the number of jobs, job sequences and the operations among machines will be fixed for a given system size. The inter-arrival rate of a job refers to the number of jobs released to the system in a certain period of time. In this case, the higher the inter-arrival rate the higher the zone loading will be. The number of jobs and the required number of operations of each job and their sequences have been selected in order to balance the loading of each station. In other words, each station, except the I/O stations, will be visited only once in generated sequences of all jobs.

The jobs inter-arrival times of each selected problem size have been selected to give an average upper bound zone utilization of 0.9, 0.7 and 0.5 for SMALL number of zones to keep the utilization within a reasonable range (less than 1). Hence, the three levels of zone loading have been denoted as HIGH, MEDIUM, and LOW respectively.

4.1.4 Population size factor

Instead of point-to-point search approach, the biggest advantage of GA is the population-to-population search approach which attempts to make the search escape from local optima. The population size is the number of individuals in a population, which directly affect the search space of a GA. Small population size results in limited search space. On the other hand, overpopulation size may lead to large searching effort. In the current study, three different population sizes, 50, 100 and 150 are tested.

4.1.5 Crossover factor

Crossover is one of the most important operators of a GA. The offspring inherits characteristics from its parents through the crossover operation. It ensures the commonness of individuals in generations. Crossover rate controls the probability of crossover in a population. Greater crossover rate may lead to faster convergence to local optima. Contrarily, it may also cause pre-maturity. Three values of crossover rate are tested in this work: 0.3, 0.5 and 0.7.

4.1.6 Mutation factor

Mutations create variations in the gene pool. The less favorable individuals are reduced in frequency in the gene pool by natural selection, and the more advantageous individuals tend to accumulate, resulting in evolutionary change. Mutations lead to new varieties of individuals and their future generations may be better adapted to the changes in their environment. However, if the mutation rate is too large, the variation may be unpredictable and uncontrollable, which will break the commonness in the genetic process. On the other hand, it decreases the possibility of finding new possible solutions if the mutation rate is too small. In this work, mutation rate of 0.03, 0.05 and 0.07 will be tested and analyzed.

4.2 Generation of the experiments

As mentioned earlier, the experiments are generated in five system sizes including 10, 20, 30, 40 and 50 stations. The layout of the 10-stations problem is adopted and modified from Bozer and Srinivasan's 8-stations problem. The layout of the 20 stations problem is adopted from Bozer and Srinivasan's. Other layouts are designed based on the desired system factors following the procedure described below.

Some system parameters are fixed and considered constant for all the problems in the generation procedure: The empty or loaded AGV speed is assumed to be 15 grids / minute; the time for an AGV to pickup or delivery a job is 0.2 minute.

The size of the system is enlarged in both the length and the width when the number of stations in the system is increased. The procedure starts by first generating the locations of the stations. After that, the number of job types and the operation sequences of the jobs are determined. Last, the values of the inter-arrival rate which result in the final flow matrix are decided based on the designated utilization threshold for different problems.

STEP 1: Generating the coordinates of the stations

- The whole area is divided into four squares.
- One fourth of the stations are placed in each square randomly. The positions of the stations are adjusted manually to insure a balanced distribution. In other words, it has to be adjusted if some stations are placed at the same position or too close to each other. Also, some stations must be put on the side borders of the shop floor. The coordinates of the generated problems are shown in Table 4-2.

STEP 2: Generating the number of job types and their sequence of operation

- As shown in Table 4-1, each layout of the problem requires two different numbers of zones. The procedure of generating the number of job types tries to generate different number of job types from the designated number of zones in each layout. The reason for that is to avoid the situation that the stations on the operation sequence of the same job will be partitioned into the same zone automatically by the proposed algorithm. The generated numbers of job types are 3, 5, 6, 8 and 10 respectively for each layout.

- A number of input/output (I/O) stations are chosen randomly from the stations placed on the borders of the floor shop in each layout. The chosen numbers of I/O stations are 3, 6 for the 10-stations, and 20-stations layouts respectively. While 9 I/O stations are selected for the 30, 40 and 50 stations layouts. The jobs are released to or sent out of the systems through these I/O stations.
- The jobs' operation sequences are decided based on the rule of each station being visited once and only once except for the I/O stations in order to have balanced utilizations of the processor stations to avoid bottlenecks. A routing of a job starts from a randomly specified I/O station as the input station and ends at another randomly specified I/O station as the output station. A number of near processor stations which are between these two I/O stations are visited. In this way, the routing is designed for each job (refer to Table 4-3).

STEP 3: Generating the jobs' inter-arrival rates

- The jobs' inter-arrival rates are designed to test the performance of the proposed algorithm at different zone loading levels. The value of the inter-arrival rate represents the numbers of jobs released into the system in a certain period of time. These values are tested and adjusted to satisfy the average upper bound of the designated utilizations, 0.9, 0.7 and 0.5, respectively for different zone loading levels. The reason that the same zone loading level has different inter-arrival rates in different layouts is because the distances between stations which affect the utilization are different. The jobs' inter-arrival rates are given in Table 4-3.

Table 4-2: The coordinates of the generated problems

10 stations									
(1,4)	(35,21)	(1,21)	(9,25)	(25,15)	(15,4)	(35,9)	(9,1)	(15,21)	(9,9)
20stations									
(7,37)	(22,37)	(30,37)	(1,33)	(7,29)	(25,29)	(45,29)	(37,23)	(55,23)	(7,16)
(22,16)	(30,16)	(45,16)	(1,7)	(16,7)	(37,7)	(55,7)	(7,1)	(25,1)	(45,1)
30 stations									
(11,39)	(26,39)	(56,39)	(1,34)	(21,34)	(43,34)	(61,34)	(11,29)	(26,29)	(39,29)
(50,29)	(56,29)	(6,23)	(21,23)	(43,23)	(61,23)	(11,14)	(21,14)	(26,14)	(39,14)
(50,14)	(56,14)	(1,11)	(61,11)	(6,6)	(21,6)	(43,6)	(11,1)	(39,1)	(56,1)
40 stations									
(5,47)	(15,47)	(51,47)	(26,43)	(58,43)	(1,39)	(9,39)	(43,39)	(65,39)	(23,35)
(33,35)	(51,35)	(58,35)	(5,30)	(26,30)	(43,30)	(9,21)	(15,25)	(15,35)	(33,25)
(51,25)	(58,25)	(23,18)	(43,18)	(58,21)	(65,18)	(5,13)	(15,13)	(33,16)	(51,13)
(1,9)	(9,9)	(23,9)	(33,9)	(43,9)	(58,9)	(5,4)	(26,4)	(15,1)	(51,1)
50 stations									
(12,54)	(40,54)	(54,54)	(19,50)	(34,50)	(50,50)	(64,50)	(50,40)	(1,46)	(29,46)
(46,46)	(54,46)	(8,40)	(68,40)	(5,35)	(12,35)	(19,46)	(40,35)	(54,35)	(25,29)
(34,29)	(46,29)	(64,29)	(5,24)	(19,24)	(40,24)	(54,24)	(68,24)	(12,20)	(34,20)
(25,35)	(1,17)	(25,17)	(50,17)	(64,17)	(8,12)	(19,12)	(46,12)	(64,5)	(12,8)
(40,8)	(54,8)	(68,8)	(5,5)	(25,5)	(34,8)	(46,5)	(5,50)	(19,1)	(40,1)

Table 4-3: The information of zone loading for the designed problems

Jobs	Processing Sequences	System loading levels		
		HIGH	MID	LOW
10 stations				
A	3*-4-9-2*	5.0	4.0	3.0
B	1*-8-6-10-3	5.0	4.0	3.0
C	2-5-7-1	5.0	4.0	3.0
20 stations				
A	1*-4-5-10-2*	5.5	4.0	3.0
B	2-3-6-8-7-9*	5.5	4.0	3.0
C	14*-18-15-19*	5.5	4.0	3.0
D	19-11-12-16-20*	5.5	4.0	3.0
E	20-17-13-9	5.5	4.0	3.0
30 stations				
A	2*-9-5-1-4*	5.0	4.0	2.5
B	4-8-13-14-17-23*	5.0	4.0	2.5
C	23-25-18-19-26-28*	5.0	4.0	2.5
D	7*-3-6-10-11-12-16*	5.0	4.0	2.5
E	16-15-22-24*	5.0	4.0	2.5
F	29*-27-20-21-30*	5.0	4.0	2.5
40 stations				
A	1*-7-19-18-17-14-6*	4.5	3.5	2.5
B	2*-4-10-15-11-3*	4.5	3.5	2.5
C	3-12-13-5-9*	4.5	3.5	2.5
D	9-21-22-25-26*	4.5	3.5	2.5
E	3-8-16-20-24-26	4.5	3.5	2.5
F	26-30-35-36-40*	4.5	3.5	2.5
G	39*-38-33-23-29-34-40	4.5	3.5	2.5
H	31*-27-28-32-37-39	4.5	3.5	2.5
50 stations				
A	1*-48-13-16-15-9*	4.0	3.0	2.0
B	1-4-17-31-10-5-2*	4.0	3.0	2.0
C	2-11-8-12-6-3*	4.0	3.0	2.0
D	3-7-14-19-23-28*	4.0	3.0	2.0
E	2-22-26-27-28	4.0	3.0	2.0
F	50*-41-38-34-35-28	4.0	3.0	2.0
G	50-47-42-39-43*	4.0	3.0	2.0
H	32*-36-44-40-37-45-49*	4.0	3.0	2.0
I	2-18-21-20-25-24-32	4.0	3.0	2.0
J	32-29-33-30-46-50	4.0	3.0	2.0

* I/O stations

4.3 Factorial Design

Design of experience is an effective approach for evaluating the effect of multiple factors on a process performance. Its efficiency and effectiveness in the analysis of multi-factor cause-response relationship has been studied rigorously (Montgomery, 1997) and its associated data analysis approach (ANOVA) is widely used. By a factorial design, all possible combinations of the factors' levels are investigated in each replication of the experiment. It is most efficient for studying the effects of two or more factors. There are two types of effects in a factorial design, the main effect and the interaction effect. The main effect of a factor is defined to be the change in response produced by a change in the level of the factor. An interaction effect shows the impact of changing the levels of one factor on the main effect of another factor. Therefore, evaluating interaction effects is extremely important. In this section, a full level design is developed and the relationship between the GA's factors and the system factors is analyzed.

In the current work, two output factors, which are the AGV workload and the computational time, are tested as the output responses. In addition, six input factors are considered as mentioned earlier, namely, the system size factor (5 levels), number of zones factor (2 levels), zone loading factor (3 levels), population size factor (3 levels), crossover rate factor (3 levels), and mutation rate factor (3 levels). Hence, $5\times2\times3\times3\times3\times3 = 810$ experiments will be carried out in each replication. Since 5 replications are applied for each experiment, the total number of conducted combinations is $810 \times 5 = 4050$. The experiments are performed on an Intel Pentium 4, 2.8 GHz PC with 512 MB of RAM. The Minitab 14 software is used for performing analysis of variance.

4.4 Analysis of Variance (ANOVA) Results

This section presents the results of experiments performed on the proposed GA using different combinations of the factors' levels. Two sets of experiments are carried out to examine the effect of different factors on the performance of the algorithm for each test case. The first set of experiments measures the performance of the algorithm in terms of solution quality, while the second investigates which factors play the most significant role on the computational time of the algorithm.

The following two sections present the p-value results of the two sets of experiments. P-value is defined as the probability value for a hypothesis test. If the p-value is less than or equal to a predetermined level of significance, the null hypothesis is rejected and the alternative hypothesis is validated as a result. On the other hand, if the p-value is greater than the predetermined level, the null hypothesis is proven true. In this problem, the significance level is determined to be 0.05. The effect of a factor or the interaction of several factors is at 5% significance if the p-value is less than 0.05.

4.4.1 Analysis of Variance: Solution Quality

The main effects of system size, numbers of zones, zone loading, population size, crossover rate, mutation rate, and their interactions are analyzed based on the quality of the solutions obtained. Table 4 shows the ANOVA results from the experiments.

As shown in Table 4-4, with respect to the response of the AGV workload, the main effect of all factors is significant at 5%. The interaction effects of AB, AC, AD, AE, AF, BC, BF, DE, DF and EF are also significant at 5%. Figure 4-1 shows the main effect of the GA parameters plots and Figure 4-2 shows the significant interaction effect plots for the AGV workload.

Table 4-4: The ANOVA results of solution quality

Source	**p-value**	**Source**	**p-value**
System size (A)	**0.000**	BC	**0.000**
Number of zones (B)	**0.000**	BD	0.211
Zone loading (C)	**0.000**	BE	0.290
Population size (D)	**0.000**	BF	**0.000**
Crossover rate (E)	**0.009**	CD	0.330
Mutation rate (F)	**0.000**	CE	0.233
AB	**0.000**	CF	0.172
AC	**0.000**	DE	**0.000**
AD	**0.000**	DF	**0.000**
AE	**0.000**	EF	**0.002**
AF	**0.000**		

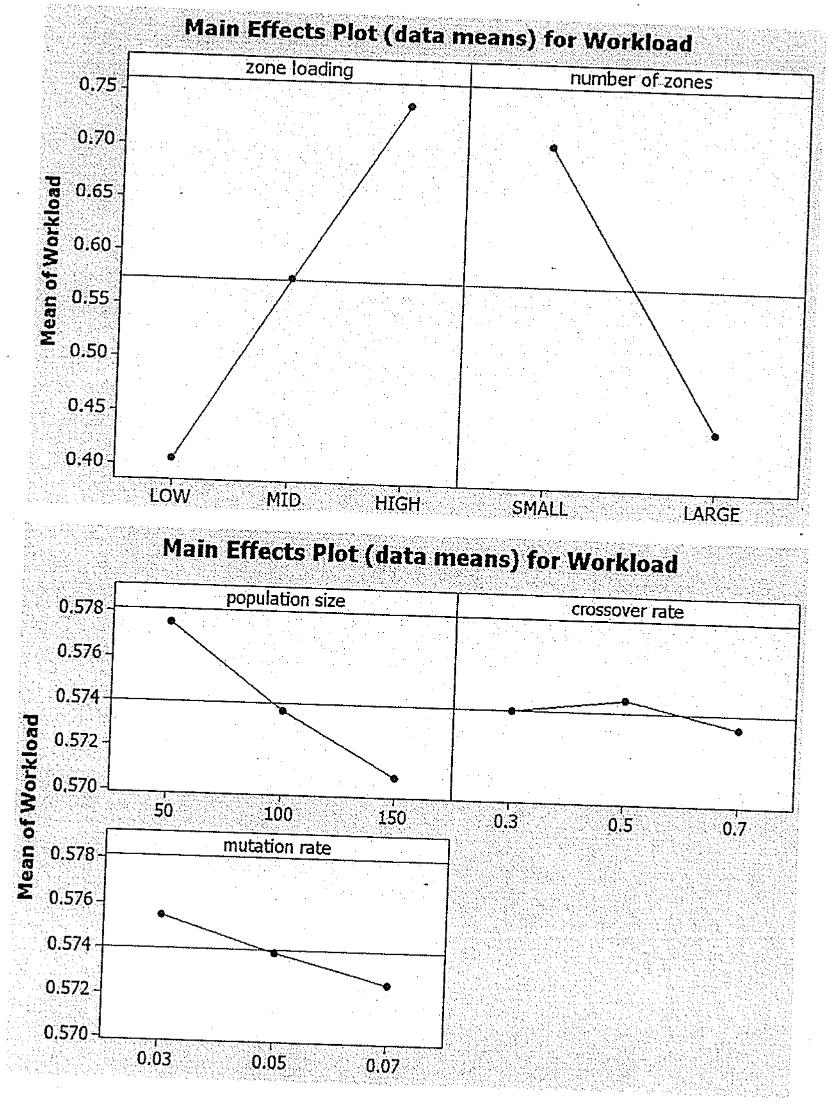
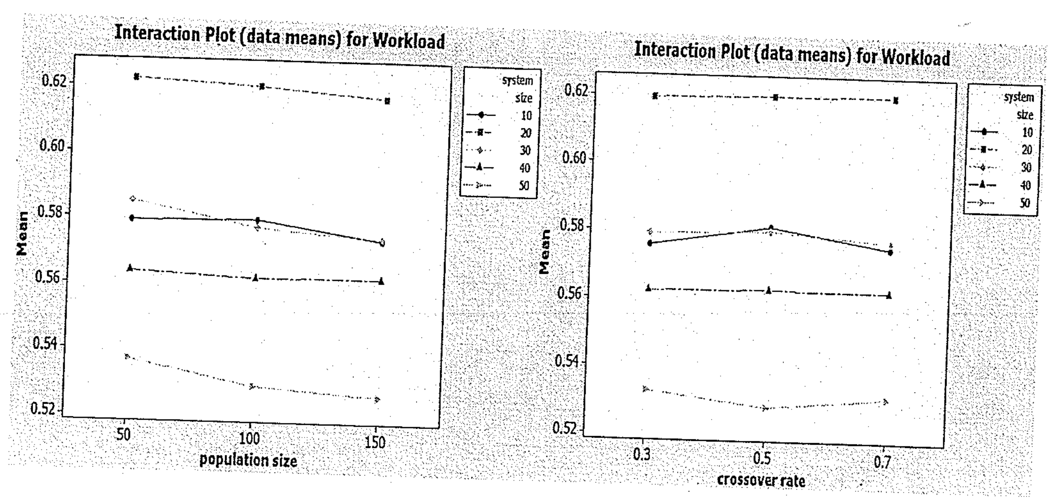


Figure 4-1: Main effect plots for the AGV workload



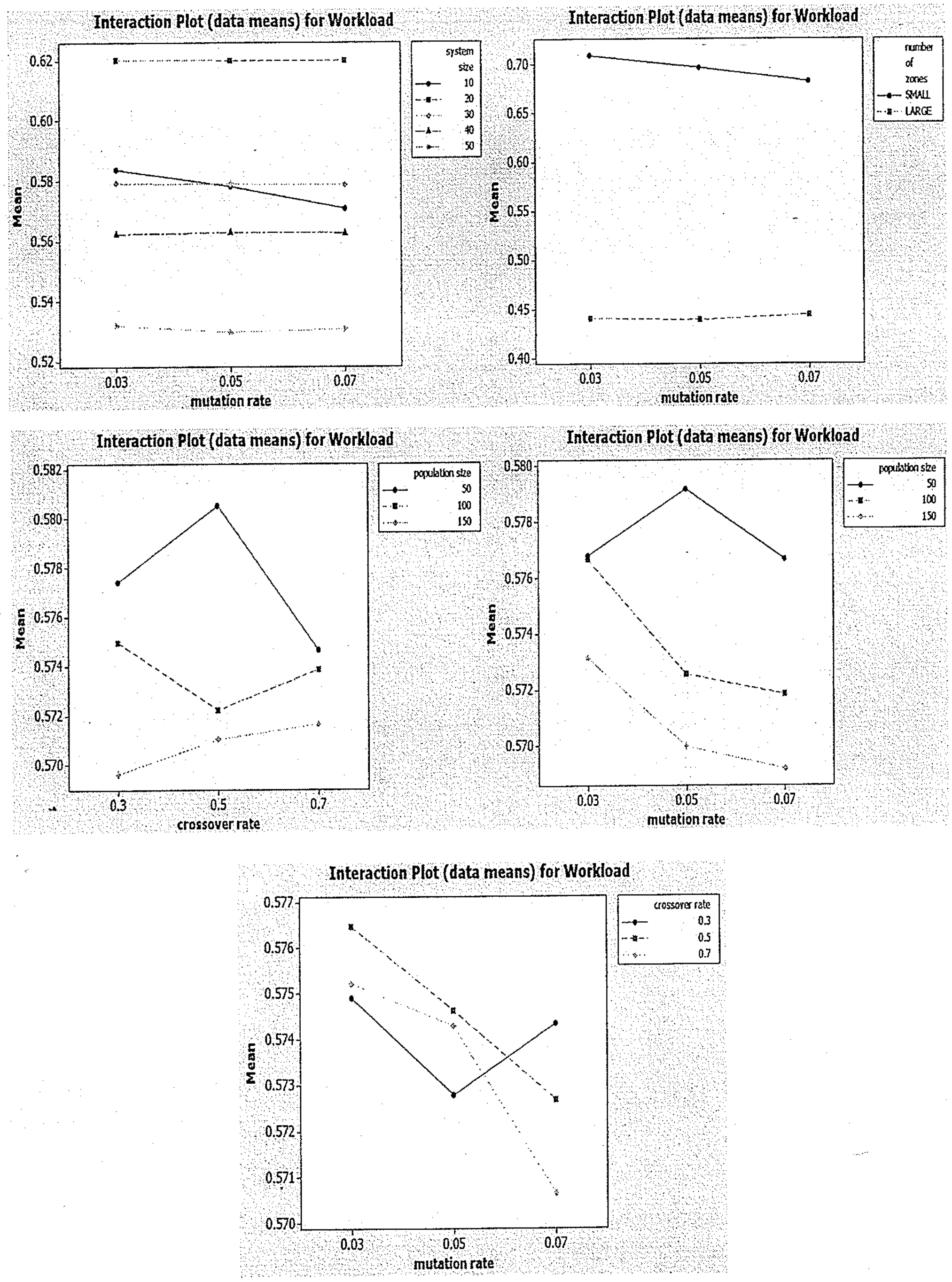


Figure 4-2: Interaction effect plots for AGV workload

Since the different sized problems are distinct, it is not necessary to observe the effect of the system's characteristics factors on the AGV workload. Generally, it is expected that the workload would be reduced when a large number of zones are designated for the same system size and work load. On the other hand, the workload increases as the zone loading factor becomes greater. The focus is on to the main effects resulting from the factors related to the GA parameters and their interaction effect with the factors related to the system's characteristics. For the GA parameters, as the objective is minimizing the workload, it seems that the best values of these parameters are 150 for the population size, 0.7 for crossover rate and 0.07 for the mutation rate (see Figure 4-1). However, the actual situation is more complex when the interaction effects are considered. The interaction effect plots are shown in Figure 4-2.

It is concluded from the interaction effect plots that in any system size, the quality of solutions improves as the population size increases. In this problem, 150 is the best choice for the population size in terms of less AGV workload. It is also concluded that the proposed algorithm performs better when the crossover rate is 0.3 or 0.7 than when it is 0.5 in the 10-stations system. However, in the 20 and 50-stations problems, a crossover rate of 0.5 obtains better solutions. In the 30-stations system, the AGV workload is lower when 0.7 is set as the crossover rate. In the 40-stations system, the crossover rate does not affect the solutions' quality significantly. In the 10-stations system, 0.07 outperforms other values in terms of the mutation rate. In the 20-stations system, better solutions are reached when 0.05 is taken as the mutation rate. All the three values 0.03, 0.05 and 0.07 seem to perform equally in terms of the solutions' quality in the 30-stations and the 40-stations systems. In the 50-stations system, mutation rates of either 0.05 or 0.07 are preferable to 0.03.

When a small number of zones are designated, the mutation rate 0.07 results in better solutions than 0.05 or 0.03. However, it does not affect the solutions significantly when a large number of zones are designated. Larger population size leads to less AGV workload at any zone loading level. For population sizes of 50, 100 and 150, the best crossover rates are 0.7, 0.3, and 0.5, respectively. The ideal mutation rate is 0.07 for all these values of population size.

As for the interaction of crossover rate and mutation rate, when the crossover rate values read 0.3, 0.5 and 0.7, the corresponding values of the mutation rate should read 0.05, 0.07 and 0.07. Table 4-6 shows the best combination of the GA parameters' values that can be selected for the different problems levels of system characteristic factors based on this analysis.

4.4.2 Analysis of Variance: Computational Time

It is important to observe the effects of the factors and their interactions on the computational time as well as on the AGV workload. Factors involved in this experiment are as the same as those considered in the previous study, namely the system size (A), the number of zones (B), the zone loading level (C), the population size (D), the crossover rate (E), and the mutation rate (F). The ANOVA results are shown in Table 4-5.

Table 4-5: The ANOVA results of computational time

Source	p-value	Source	p-value
System size (A)	**0.000**	BC	0.233
Number of zones (B)	**0.000**	BD	**0.000**
Zone loading (C)	**0.016**	BE	0.471
Population size (D)	**0.000**	BF	0.491
Crossover rate (E)	**0.047**	CD	0.090
Mutation rate (F)	**0.000**	CE	0.085
AB	**0.000**	CF	**0.043**
AC	0.138	DE	0.058
AD	**0.000**	DF	**0.007**
AE	0.471	EF	0.570
AF	**0.000**	BC	0.233

The results show that there are significance main effects of factors A, B, C, D, E, F and the interaction effects of AB, AD, AF, BD, CF and DF when considering the computational time. The main effect plots for computational time are shown in Figure 4-3. It is clear that the computational time increases when any of the factors A, B, C or D is increased. When considering crossover rates, 0.5 leads to the longest computational time while 0.7 lead to the shortest. For mutation rates, 0.03 seems to be the most efficient choice while 0.07 is the least efficient in terms of computational time.

Figure 4-4 displays the factors' significant interaction effect. It is critical to note that the required computational time is much greater when large numbers rather than small numbers of zones are designated in big (40-station and 50-station) systems. However, the difference in computational time is not as obvious in small sized systems.

Additionally, the computational time increases as the population size and mutation rate are raised in any system size. Another conclusion drawn from the interaction effect is that the computational time is greater when a larger number of zones rather than a small number are designated. Larger population size results in longer computational time in systems having either a small or large number of zones. However, the growth of computational time along with the increase of the population size is slower when the number of zones is at the small level.

Finally, as far as mutation rates are concerned, 0.03 leads to the shortest computational time while 0.07 leads to the longest at any level of zone loading or population size. When mutation rate is 0.05 or 0.07, the computational time of high zone loading and mid zone loading are quiet close, but the computational time of low zone loading is much less than the other two. When population size is 50 or 150, larger mutation rate results in longer computational time. However, when population size is 100, the computational times obtained when mutation rate is 0.05 and 0.07 are similar.

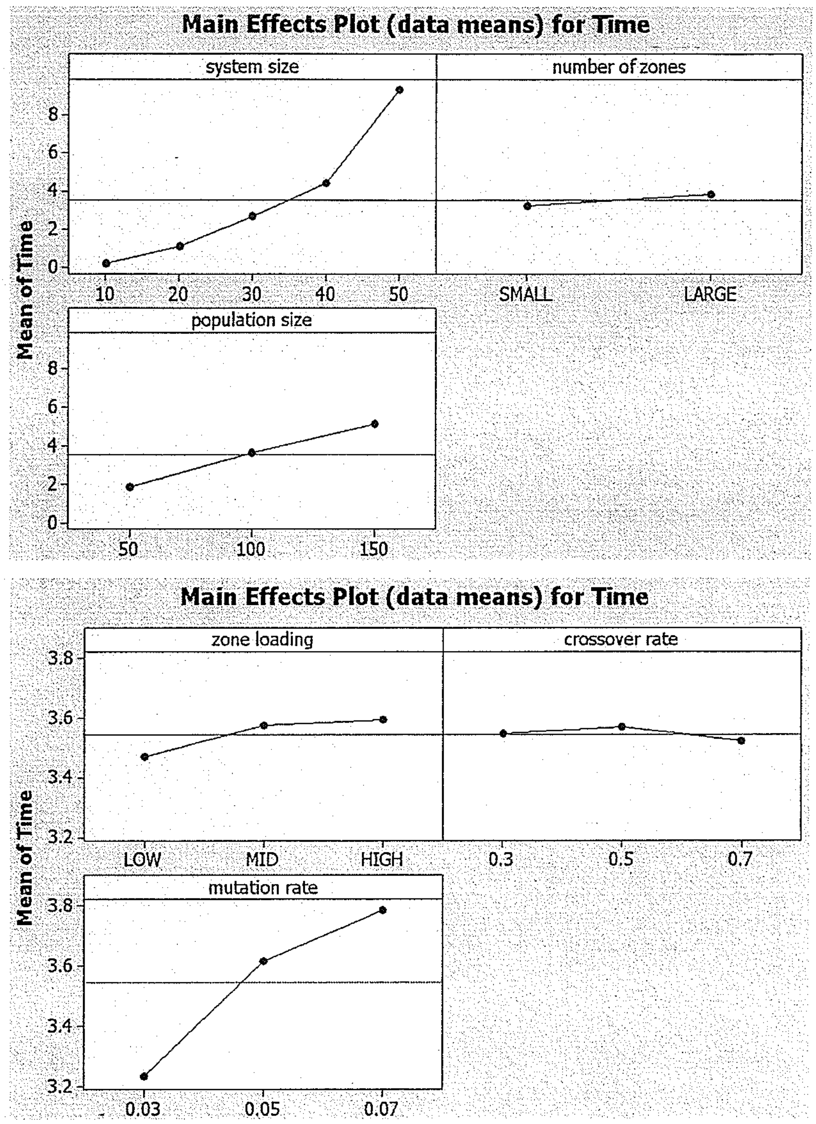
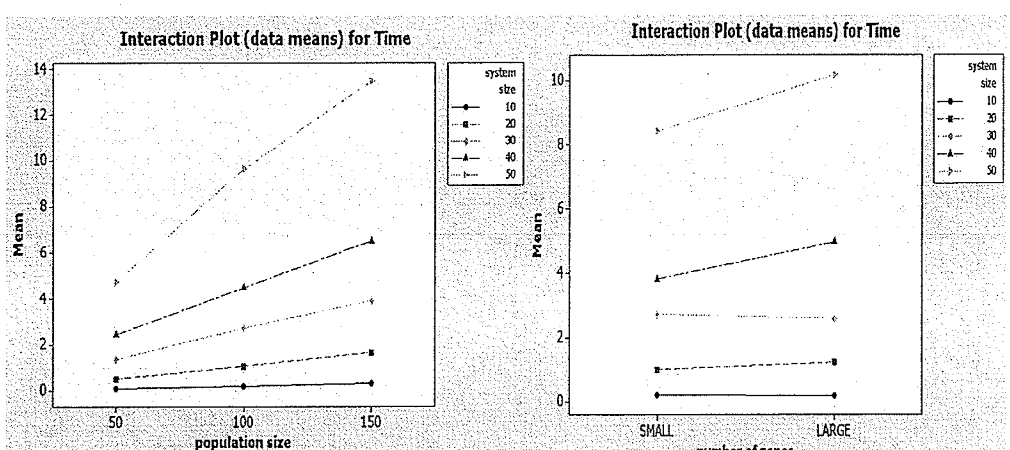


Figure 4-3: Main effect plots for the computational time



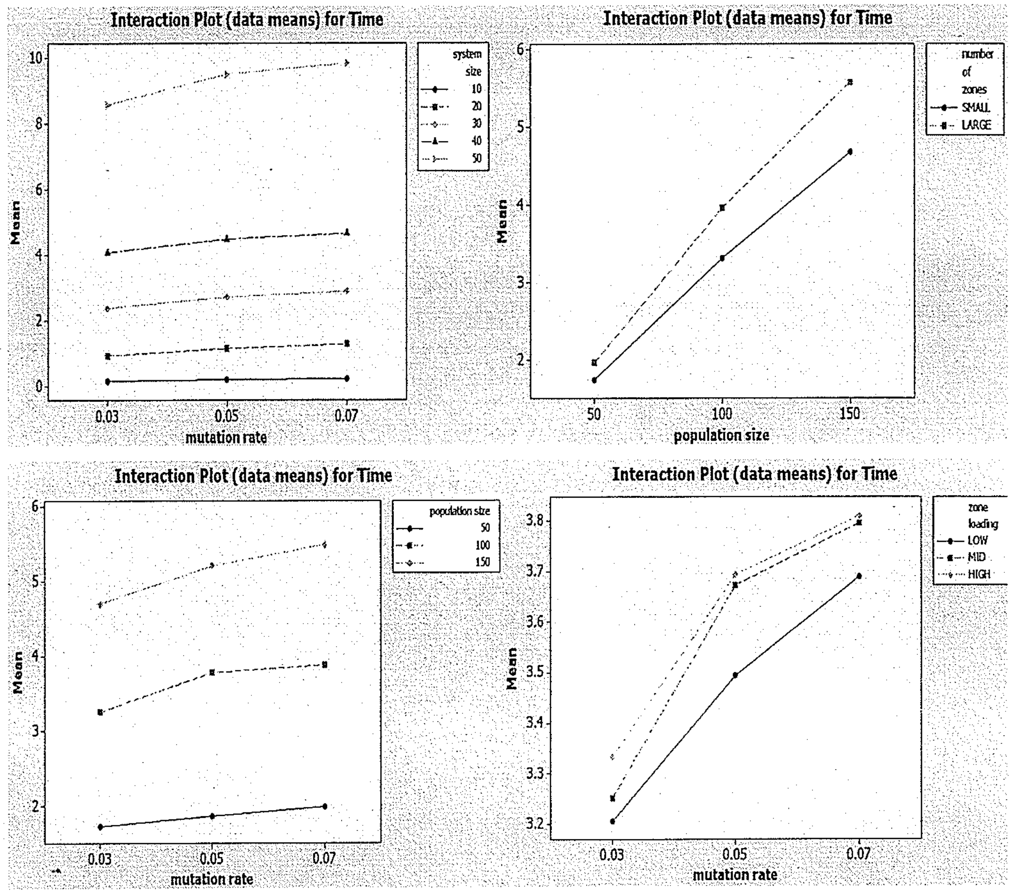


Figure 4-4: Interaction effect plots for computational time

Table 4-6: Summary of the best combination of the factors' values

System size	Zone loading	Number of zones	Solution quality			Computational time		
			Crossover Rate (CR)	Mutation Rate (MR)	Population Size (PS)	CR	MR	PS
10	Any		0.7	0.07	150	0.7	0.03	50
20	Any	LARGE	0.5	0.05				
		SMALL	0.7	0.07				
30	Any		0.7	0.07				
40	Any	LARGE	0.5	0.07				
		SMALL	0.7	0.07				
50	Any		0.5	0.07				

The proposed GA is used to solve the designed experiments discussed in this chapter.

For each combination, 5 replications are solved by the proposed GA. Thus, 150 cases have been solved in total. The GA parameters are set based on the results shown in Table 4-6. The obtained results of these problems are listed in Table 4-7.

Table 4-7: The results of the designed experiments

Problem Size	**Zone Loading**	**Number of Zones**	**Best Solution**	**Worst Solution**	**Average Solution**	**Average CPU Time (s)**
10	HIGH	LARGE	0.539(100%)	-	0.539	0.56
		SMALL	0.857(80%)	0.950(20%)	0.876	0.53
	MID	LARGE	0.431 (100%)	-	0.431	0.52
		SMALL	0.687(20%)	0.770(80%)	0.753	0.49
	LOW	LARGE	0.323(100%)	-	0.323	0.50
		SMALL	0.516(40%)	0.581(60%)	0.556	0.47
20	HIGH	LARGE	0.664(40%)	0.699(20%)	0.681	22.4
		SMALL	0.941(40%)	0.945(60%)	0.943	20.0
	MID	LARGE	0.486(80%)	0.511(20%)	0.491	21.1
		SMALL	0.696(40%)	0.699(60%)	0.698	19.8
	LOW	LARGE	0.366(60%)	0.383(40%)	0.373	21.2
		SMALL	0.517(20%)	0.530(20%)	0.523	19.3
30	HIGH	LARGE	0.583(20%)	0.631(40%)	0.607	43.9
		SMALL	0.897(80%)	0.928(20%)	0.912	32.7
	MID	LARGE	0.467(40%)	0.496(20%)	0.481	37.5
		SMALL	0.726(80%)	0.753(20%)	0.731	29.6
	LOW	LARGE	0.275(20%)	0.315(20%)	0.295	36.0
		SMALL	0.461(80%)	0.464(20%)	0.462	29.3
40	HIGH	LARGE	0.514(40%)	0.565(20%)	0.540	87.2
		SMALL	0.889(20%)	0.891(80%)	0.890	76.4
	MID	LARGE	0.420(40%)	0.427(60%)	0.424	85.1
		SMALL	0.703(100%)	-	0.703	74.3
	LOW	LARGE	0.300(20%)	0.307(80%)	0.306	85.0
		SMALL	0.477(20%)	0.508(60%)	0.493	69.8
50	HIGH	LARGE	0.487(60%)	0.500(20%)	0.493	164.8
		SMALL	0.916(20%)	0.943(20%)	0.930	156.9
	MID	LARGE	0.366(80%)	0.375(20%)	0.368	151.5
		SMALL	0.693(20%)	0.719(40%)	0.706	115.7
	LOW	LARGE	0.245(100%)	-	0.245	142.8
		SMALL	0.462(40%)	0.468(20%)	0.465	112.3

In Table 4-7, the fourth and the fifth column refer to the best and the worst solutions and their percentage in all the five replications. It is noticed that the AGV workload is much higher when small number of zones is required at the same zone loading level. On the other hand, when the number of zones is fixed, the AGV workload goes up along with the increasing the zone loading. The computational time is directly effected by the system size. It takes longer for computing in bigger sized systems. However, the effects of the number of zones or the zone loading on the computational time are not obvious.

CHAPTER 5

SOLVING THE PARTITIONING PROBLEM OF TANDEM AGV SYSTEMS WITH MEMETIC ALGORITHM

5.1 Background

The term of memetic algorithm (MA) was introduced by Moscato and Norman (1992) to describe evolutionary algorithms in which local search plays a significant part. This term is motivated by Richard Dawkins's notion of a meme as a unit of information that reproduces itself as people exchange ideas (Dawkins, 1976). A key difference between genes and memes are that before a meme is passed on, it is typically adapted by the individual who transmits it as that individual thinks, understands and processes the meme, where as genes get passed on whole. Moscato and Norman liken this thinking to local refinement, and therefore promote the term "memetic algorithm" to describe genetic algorithms that use local search heavily.

Radcliffe and Surry (1994) gave a formal description of MA. They mentioned that if a local optimizer is added to a genetic algorithm, and applied to every child before it is inserted into the population then a memetic algorithm can be thought of simply as a special kind of "genetic" search over the subspace of local optima. Recombination and mutation will usually produce solutions that are outside this space of local optima but a local optimizer can then "repair" such solutions to produce final children that lie within this subspace, yielding a memetic algorithm.

Memetic algorithms have been applied on many combinatorial optimization problems successfully. Merz and Freisleben (2000) applied MA on the quadratic assignment problem (QAP). They presented a few experiments comparing three different evolutionary operators for MA. France et al. (2001) proposed a MA for the total tardiness single machine scheduling (SMS) problem with due dates and sequence-dependent setup times. The MA and a pure GA were compared in this work. Merz and Freisleben (2001) illustrated that MA is well-suited for finding near-optimum tours for the traveling salesman problem (TSP) by investigating several instances of traveling salesman problems. It was shown that the MA with genetic recombination was among the best evolutionary algorithms for the TSP. Buriol and Franca (2004) proposed a MA for the asymmetric traveling salesman problem (ATSP). A new local search called Recursive Arc Insertion (RAI) was introduced in their paper. The MA was compared with six other meta-heuristics and it was demonstrated that the MA outperformed all previous approaches.

In this chapter, a new local search is proposed in order to suit the particular problem and it is combined with a genetic algorithm (GA) to be used on partitioning the tandem AGV systems. Although genetic algorithms are very powerful and can be applied widely, they are not well suited for local optimization. With the combination of genetic search and local search, genetic search is used to perform global exploration among the population, and local search is used to perform local exploitation around chromosomes. Some benchmark problems will be solved using

MA as well as a group of designed experiments. The performance of MA will be compared with GA and other approaches obtained from the literature.

5.2 The Proposed Memetic Algorithm

The same assumptions made in chapter 3 are applied on the proposed MA in this chapter: Bi-directional AGVs are used in the system; Each station will be assigned to only one zone; Each zone should have at least two stations; Both loaded and empty trips are considered while the AGV workload is estimated; The AGV always chooses the shortest rectilinear path to its destination when it is loaded; The vehicle follows shortest time to travel first (STTF) dispatching policy when it is empty; The number of zones is given as an input; Intersections and overlaps are forbidden among zones; One transfer point is assigned to each zone and it is co-located with the station which has the maximum flow with other zones.

The flow chart of the proposed MA is shown in Figure 5-1. As described earlier in chapter 3, the k-means clustering method is used to generate the initial population. Selection will choose the fittest individuals from among the current population to survive. Genetic operators, including the crossover and mutation, are employed to propagate the populations from one generation to another to find the optimum solution. These operators allow a global search over the entire design space to avoid being trapped in local optima. The repair procedure developed in chapter 3 is also applied in this algorithm to restore the infeasible solutions resulting from crossover and mutation operations. A new local search is generated and applied to each newly generated individual to move it to a local optimum before injecting it into the population. This procedure is carried out on every population including the initial population.

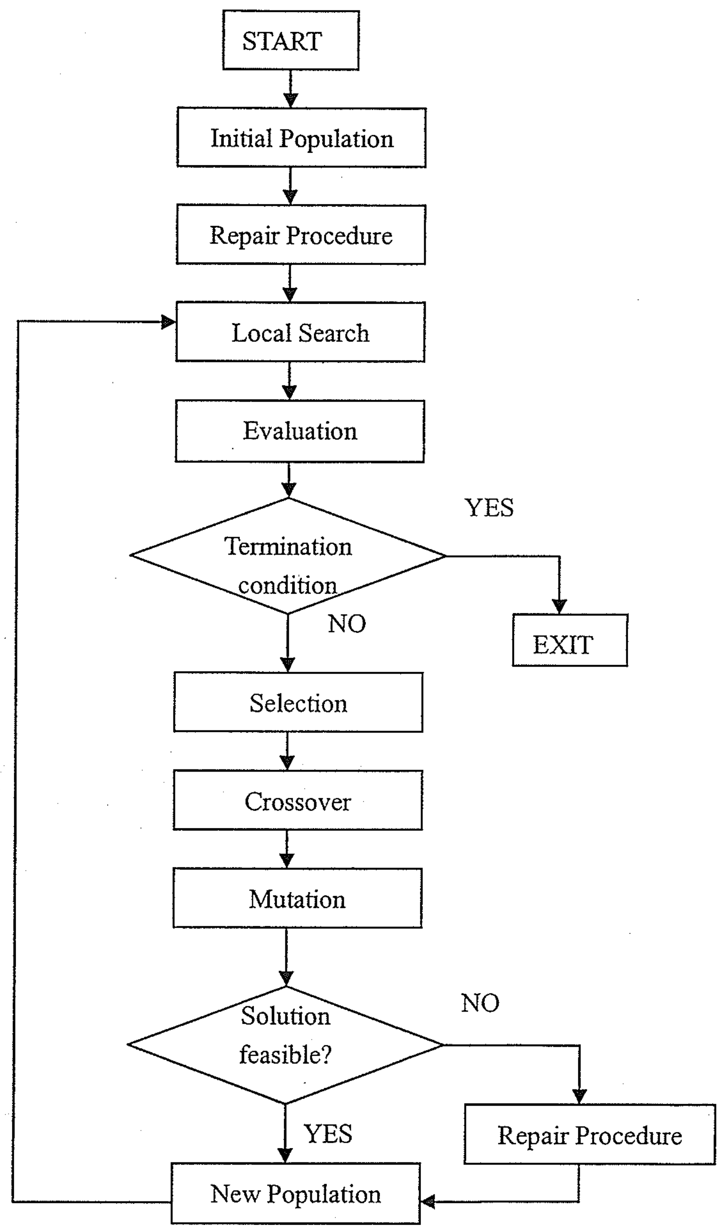
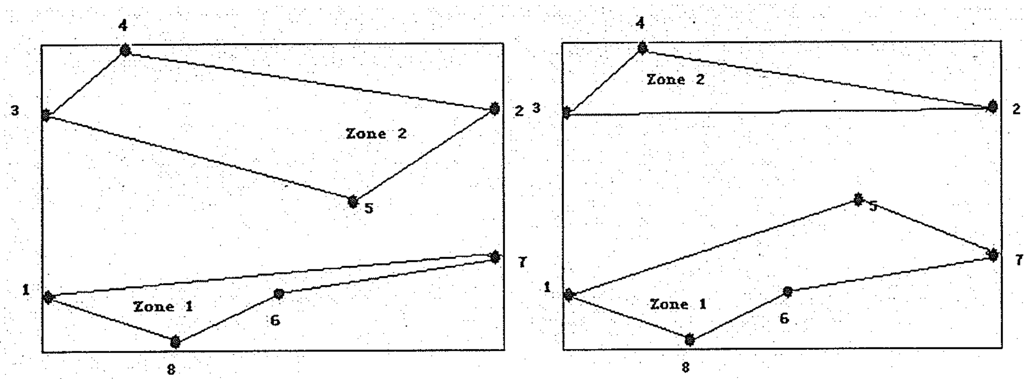


Figure 5-1: Flowchart of the proposed MA

The difference between GA and MA is the existence of a local search heuristic that is combined with the GA to find local optimal solutions around the neighborhood of individuals of a population. Local search heuristics move from solution to solution within the search space (neighborhood) until a solution deemed optimal is found or the time limit has elapsed. The neighborhood of any given solution is created through the movement or swapping of stations between formed zones. In other words, the neighborhood contains all the feasible solutions of the current combinatorial optimization problem which can be reached from the current solution by the move. Therefore, the principal goal in creating a local search heuristic is defining the neighborhoods using these moves.

In the tandem AGV system, the neighborhood of a solution can be established by removing a station from one zone and adding it to another zone. For instance, a solution [12222111], as shown in Figure 5-2 (a), contains 8 stations partitioned into 2 zones. In the decoding space, this solution can be represented as: zone 1 [s1, s6, s7, s8], zone 2 [s2, s3, s4, s5]. Figure 5-2 (b) shows a possible neighborhood solution [12221111] obtained by removing station 5 from zone 2 and adding it into zone 1. This solution represents zone 1 [s1, s5, s6, s7, s8] and zone 2 [s2, s3, s4] in the decoding space.



(a) (b)

Figure 5-2: An example of a neighborhood solution

This thesis proposes a specific local search method using this definition of neighborhood. Some important points in this proposed local search are discussed in the following sections.

5.2.1 The move

As discussed earlier, the main idea behind the move is to remove a station from one zone and add it to another. Moves may sometimes generate infeasible solutions. Therefore, the infeasibility resulting from such moves has to be resolved. Overlaps between zones may occur while moving a station from one zone to another. However, repairing the overlapping zones will change the solution and the fitness value entirely, thus rendering the local search meaningless and consuming a lot of computational time. Hence, in the proposed method, the neighborhood solutions will be checked for feasibility before being compared with the original solution. Should a neighborhood solution contain overlapping zones, it will be rejected directly.

Additionally, singleton zones may occur when a station is removed from a zone in which only two stations were present before the move. An intelligent local search method is required to avoid this problem. In the proposed method, the number of stations in the target zone is calculated first. If the number is greater than 2, a station will be removed from this zone and added to another. Alternatively, to avoid the creation of a singleton zone, stations swapping will be used. In other words, the proposed local search method combines the principles of moving and swapping.

5.2.2 Selection of the target zone and candidate stations

Since the objective of the proposed algorithm is to minimize the maximum workload, the zone that has the maximum workload will be selected as a target zone in order to reduce its workload. At the beginning of the local search, the workload of each zone is calculated to identify the target zone.

When it comes to selecting which stations to move/swap, we have to define a good criterion for this selection since it will be time consuming to consider all stations within the target zone. It is worth mentioning that the zone workload is directly affected by the flow among its stations (in and out). Therefore, the total flow (weight) of stations can be used as a selection criterion. The weight of a station is defined as the sum of the flow entering and exiting the station. Another parameter that has to be defined at this stage is the number of stations that has to be considered to generate the neighborhood solutions. In this study, four stations with the largest weights will be considered. When there are fewer than four stations in the target zone, all the stations will be considered.

5.2.3 Evaluation of the move

The flowchart of the proposed local search algorithm is shown in Figure 5-3. Every trial of the local search algorithm begins with selecting the station in the target zone that has the highest weight and has not been selected before. When a possible move results in a feasible solution, this solution has to be evaluated by estimating its maximum workload. It must be clarified that the objective of the move is not just solely to reduce the workload of the target zone. Instead, all the workloads in the generated solution should be considered and the maximum workload should be minimized or at least reduced to a local optimal value. Therefore, the workload of each zone is calculated after a feasible move and the maximum workload is saved for the generated solution.

At the end of a trial, all the feasible solutions generated by moving or swapping this station and their corresponding maximum workloads are saved. Once the possible moves of all candidate stations to every other zone are considered, the minimum workload obtained from the candidate workloads is compared with the original workload. If a better solution is generated, it replaces the original one. Otherwise, the original solution will be retained. These steps are applied to the next solution in the current population until all solutions are explored.

The steps of the local search algorithm:

STEP 1: Set the solution number (S) = 0.

STEP 2: Select a solution which has not been selected before in the current population.

STEP 3: Find the target zone (z).

STEP 4: Calculate the number of stations (N) in this target zone.

STEP 5: Set the trials number (T) = 0.

STEP 6: Select a station (s, $s \in z$) which has not been selected before with the highest weight.

STEP 7: Chose another zone (z') as a candidate zone.

STEP 8: If N > 2, go to STEP 9. Else, go to STEP 10.

STEP 9: Add station s into zone z', go to STEP 12.

STEP 10: Select the nearest station (s', $s' \in z'$) to s.

STEP 11: Swap s' with s.

STEP 12: Check the feasibility of the generated solution. If the solution is feasible, go to STEP 13. Otherwise, go to STEP 14.

STEP 13: Calculated the maximum workload (W') in the new generated solution and save the corresponding move.

STEP 14: If there are other zones that have not been chosen before as candidate zone, go to STEP 7. Otherwise, go to STEP 15.

STEP 15: Update the trials number T=T+1.

STEP 16: If T < min (N,4), go to STEP 6, else go to STEP 17.

STEP 17: Select the solution with the minimum-maximum workload (minW') and compare it with the original solution (W). If the solution is improved (minW'<W), go to STEP 18, else go to STEP 19.

STEP 18: Update the solution and workload (W=minW').

STEP 19: Update the solution number S=S+1.

STEP 20: If S is less than the population size (total number of solutions in the current population), go to STEP 2. Otherwise, EXIT.

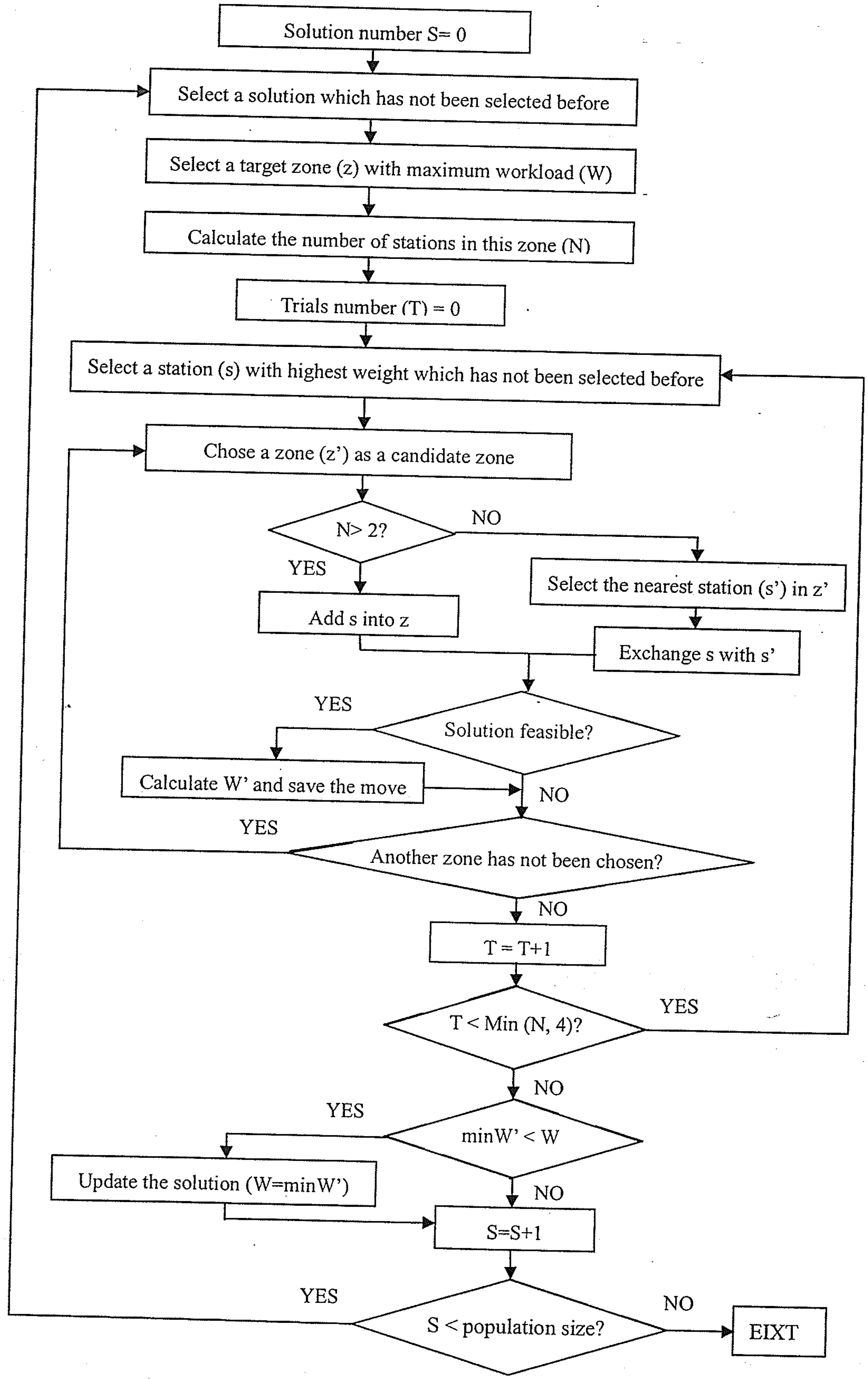


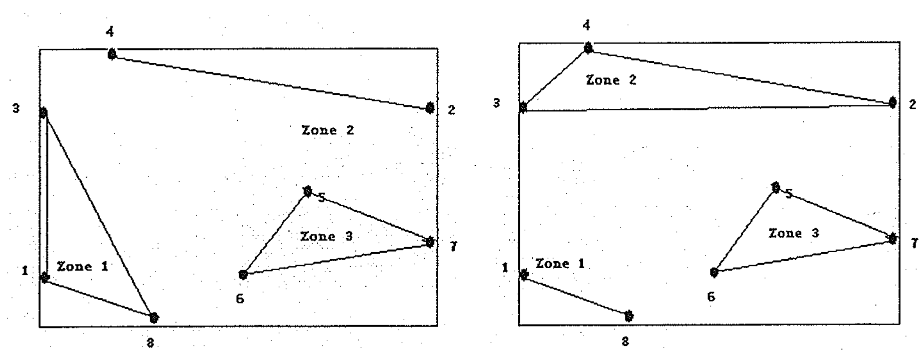
Figure 5-3: Flowchart of the local search

The following example illustrates the basic idea of the local search procedures. Figure 5-4 (a) shows a solution of an 8-stations system to be partitioned into 3 zones: zone 1 [s1, s3, s8], zone 2 [s2, s4], zone 3 [s5, s6, s7]. It is assumed that the maximum workload exists in zone 1 and it is represented by W. According to the local search algorithm, zone 1 is the target zone and the stations present in zone 1 are considered for moves in order to search for neighborhood solutions. Since the total number of stations in zone 1 is 3 (less than 4), it is not necessary to calculate the weights of these stations. Instead, all of them are considered to be candidate stations.

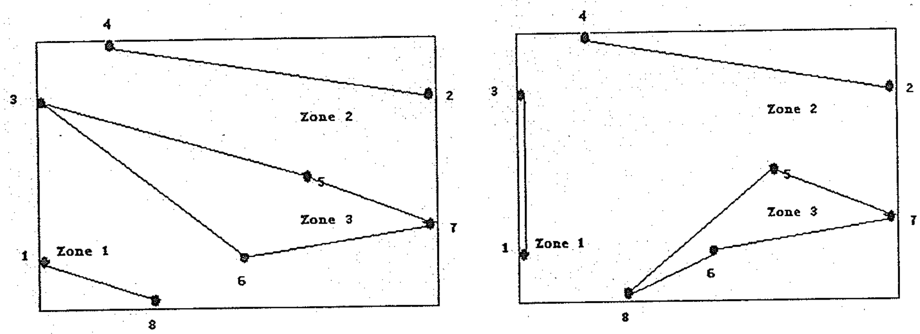
First, station 1 is selected. However, no feasible solutions result from moving station 1 to any other zone. Overlaps will occur in the system when station 1 is added to zones 2 or 3. Under these circumstances, moving station 1 will not generate any feasible neighborhood solutions.

The second trial is to consider station 3. By removing station 3 from zone 1 and adding it to zone 2, a neighborhood solution is generated as shown is Figure 5-4 (b): zone 1 [s1, s8], zone 2 [s2, s3, s4], zone 3 [s5, s6, s7]. The workloads among all the zones are calculated; the maximum workload is displayed as W_1'. There is another possible move for station 3, which involves removing station 3 from zone 1 and adding it to zone 3 as shown in Figure 5-4 (c). The corresponding neighborhood solution is: zone 1 [s1, s8], zone 2 [s2, s4], zone 3 [s3, s5, s6, s7]. The new maximum workload is W_2'.

The last station in the target zone that can be moved is station 8. It is determined that the only feasible move for station 8 is to remove it from zone 1 and add it to zone 3 (Figure 5-4 (d)). This move results in the following new solution: zone 1 [s1, s3], zone 2 [s2, s4], zone 3 [s5, s6, s7, s8]. The workloads in the whole system are calculated and the maximum one is chosen (W_3'). No more stations in the target zone can be considered for moves. Since the objective is minimizing the maximum workload, the smallest value among W_1', W_2' and W_3' is selected as the candidate workload minW' and compared with the original workload W. If minW' is smaller than W, the maximum workload is updated. In other words, the value of minW' is assigned to W. The original solution will be replaced by the new one in the population as well. On the contrary, if minW' is greater than W, the original solution and its workload will be maintained in the population.



(a) (b)



(c) (d)

Figure 5-4: An example of local search algorithm

5.3 Results and Analysis

The proposed MA is applied on the benchmark problems obtained from the literature as described in Chapter 3 and the results are compared with the pure GA proposed in Chapter 3 as well as the reported results from literature. The problems are solved by the MA using C language on a 2.80 GHz Intel Pentium 4 with 512 MB RAM system. The parameters of MA and GA are chosen based on Table 4-6.

It has to be mentioned that the used assumptions and the method of estimating the work loads have direct impact on the reported results. Under our assumptions, one transfer point is assigned to each zone and it is co-located with the station which has the maximum flow with other zones. However, it was not clear that where the transfer points were located and how many transfer points were assigned to each zone in the previous papers. Therefore, in order to perform a fair comparison, we used our model to estimate zones' work load of the reported solutions (zones formation) in the literature.

The results are given in Tables 5-1, 5-2, 5-3 and 5-4. It can be noticed that both the MA and GA obtained the same results as Bozer and Srinivasan (1992) when the 8-stations problem was solved. As for the 10-stations problem, the maximum workload obtained by the MA or GA is the same, 0.6058, which is lower than the ones obtained by Yu and Egbelu (2001) and Shalaby et al (2006) algorithms (0.8083, 0.7085). For the 20-stations problem with 6-zones, the maximum workload obtained by GA is 0.3746 which is again better than the ones obtained by Bozer and Srinivasan (1992) and Shalaby et al. (2006) algorithms (0.4971, 0.4088). By applying MA instead of GA to the 20-stations (6-zones) problem, the maximum workload is reduced from 0.3746 to 0.3705 and the max difference between workloads is reduced from 0.0812 to 0.0738. With 4-zones, the MA and GA generated the same results, which outperformed the results of Yu and Egbelu (2001), and Shalaby et al. (2006).

As for the computational time, Laporte et al. (2006) reported the CPU times for the 8-stations and 20-stations (6-zones) problems, which are 12.2 seconds and 129.5 seconds respectively using a 2.00 GHz Intel Pentium 4 with 256 MB RAM system. In the current work, the average CPU times of solving these two problems are 0.49 seconds and 24.6 seconds respectively when GA is applied, 2.6 seconds and 40.1 seconds respectively when MA is applied.

Table 5-1: Results of the 8-stations problem

MA/GA		Bozer and Srinivasan/ Laporte et al.	
Zones	Estimated workload	Zones	Estimated workload
1,8	0.2983	1,8	0.2983
2,5	0.3900	2,5	0.3900
3,4	0.3750	3,4	0.3750
6,7	**0.4417***	6,7	**0.4417***

* Maximum workload

Table 5-2: Results of the 10-stations problem

MA/GA		Yu and Egbelu		Shalaby et al.	
Zones	Estimated Workload	Zones	Estimated Workload	Zones	Estimated Workload
1,3	0.5550	1,7	**0.8083***	1,2	0.4942
2,4,5,6	**0.6058***	2,5	0.2683	3,4,5	0.4079
7,8	0.6008	3,4	0.2642	7,8	0.6008
9,10	0.5433	6,8,9,10	0.7886	6,9,10	**0.7085***
Max Difference	0.0625		0.5441		0.3006

* Maximum workload

Table 5-3: Results of the 20-stations problem (6-zones)

MA		GA		Bozer and Srinivasan / Laporte et al.		Shalaby et al.	
Zones	Estimated Workload	Zones	Estimated Workload	Zones	Estimated Workload	Zones	Estimated Workload
1,2,4,5	0.3135	1,2,4,5	0.3135	1,2,3,6	0.3583	1,2,4,5	0.3135
3,6,8	0.3575	3,6,10	0.3567	4,5,10, 14	**0.4971***	3,6,12	0.3867
7,9,13	0.3510	7,8,9	0.3460	7,8,9	0.3460	7,8,9	0.3460
10,11, 12,19	**0.3705***	13,17, 20	0.2938	11,12, 13	0.3800	10,14, 18	0.2893
14,15,18	0.3025	11,12, 16,19	**0.3746***	15,18, 19	0.2913	11,15, 19	0.3233
16,17, 20	0.2967	14,15, 18	0.3205	16,17, 20	0.2967	13,16, 17,20	**0.4088***
Max Difference	0.0738		0.0812		0.2058		0.1195

* Maximum workload

Table 5-4: Results of the 20-stations problem (4-zones)

MA/GA		Yu and Egbelu		Shalaby et al.	
Zones	Estimated Workload	Zones	Estimated Workload	Zones	Estimated Workload
1,2,3,4,5,6	**0.5349***	1,2,3,4,5,6	0.5349	1,2,3,4,5,6	0.5349
7,9,13,17	0.4824	7,8,9,13,17	**0.5530***	7,8,11,12	**0.5985***
8,12,16,19,20	0.5325	10,11,14,15,18	0.4795	9,13,16,17,20	0.5234
10,11,14,15,18	0.4850	12,16,19,20	0.3676	10,14,15,18,19	0.4772
Max Difference	0.0525		0.1854		0.1213

* Maximum workload

For further comparison, the designed experiments are solved using GA and MA. The computational results are given in Table 5-5. In Table 5-5, the average improvement is defined as follows:

$$(A_{GA}-A_{MA})/A_{GA}$$

Where:

A_{GA} is the average value of the results generated using GA.

A_{MA} is the average value of the results generated using MA.

Since the objective is to minimize the workload, positive values represent positive improvement of the algorithm. It can be concluded from Table 5-5 that MA makes a lot of improvements in terms of the solution quality. However, MA requires more computational time than GA especially for large size problems. As shown in Figure 5-5, the horizontal axis represents different system sizes and the vertical axis represents the computational time. According to this chart, for both algorithms, the computational time increases with system sizes. However, it rises more rapidly and steeply when MA is applied than GA.

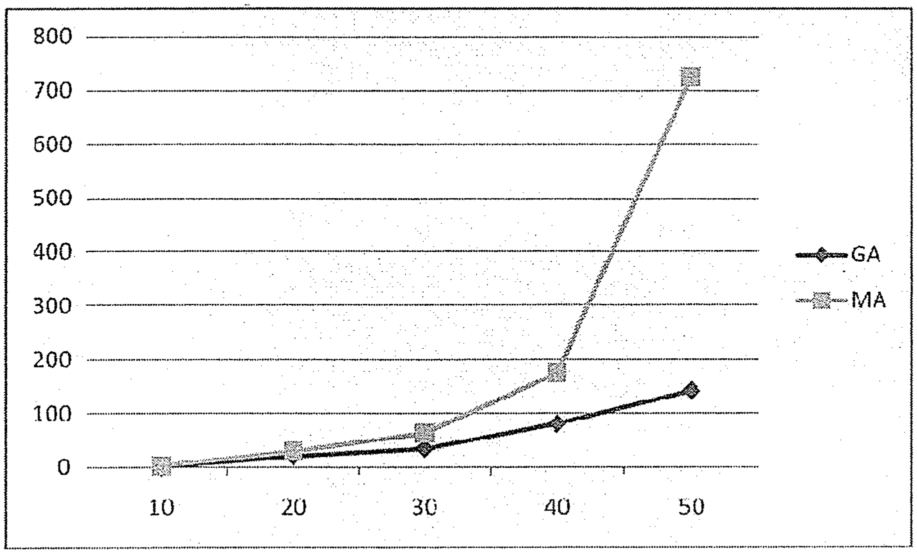


Figure 5-5: Chart of computational time (MA vs. GA)

The obtained results demonstrated the efficiency of the developed MA in solving the partitioning problem of tandem AGV systems. In terms of solutions quality, the proposed MA outperforms all the previous approaches as well as pure GA. On the other hand, MA seems worse than pure GA in terms of computational time, especially for large size problems. However, the computational time of MA is still within the accepted range.

Table 5-5: Results of the designed experiments using GA and MA

Problem Size	Zone Loading	Number of Zones	GA Best solution	GA Worst solution	GA Average	GA CPU time	MA Best solution	MA Worst solution	MA Average	MA CPU time	Average Improvement
10	HIGH	LARGE	0.539(100%)	-	0.539	0.56	0.539(100%)	-	0.539	3.2	0%
		SMALL	0.857(80%)	0.950(20%)	0.876	0.53	0.857(100%)	-	0.857	3.0	2.17%
	MID	LARGE	0.431 (100%)	-	0.431	0.52	0.431(100%)	-	0.431	3.0	0%
		SMALL	0.687(20%)	0.770(80%)	0.753	0.49	0.687(100%)	-	0.687	2.8	8.76%
	LOW	LARGE	0.323(100%)	-	0.323	0.50	0.323(100%)	-	0.323	2.9	0%
		SMALL	0.516(40%)	0.581(60%)	0.556	0.47	0.516(100%)	-	0.516	2.7	7.19%
20	HIGH	LARGE	0.664(40%)	0.699(20%)	0.681	22.4	0.664(40%)	0.691(60%)	0.680	32.6	0.15%
		SMALL	0.941(40%)	0.945(60%)	0.943	20.0	0.902(40%)	0.917(40%)	0.910	29.8	3.50%
	MID	LARGE	0.486(80%)	0.511(20%)	0.491	21.1	0.486(80%)	0.502(20%)	0.489	31.0	0.41%
		SMALL	0.696(40%)	0.699(60%)	0.698	19.8	0.676(60%)	0.699(20%)	0.682	28.2	2.29%
	LOW	LARGE	0.366(60%)	0.383(40%)	0.373	21.2	0.366(40%)	0.377(60%)	0.372	29.9	0.27%
		SMALL	0.517(20%)	0.530(20%)	0.523	19.3	0.517(60%)	0.530(20%)	0.520	26.7	0.57%
30	HIGH	LARGE	0.583(20%)	0.631(40%)	0.607	43.9	0.575(80%)	0.631(20%)	0.586	72.6	3.46%
		SMALL	0.897(80%)	0.928(20%)	0.912	32.7	0.888(40%)	0.928(40%)	0.908	63.1	0.44%
	MID	LARGE	0.467(40%)	0.496(20%)	0.481	37.5	0.435(20%)	0.467(60%)	0.459	69.8	4.57%
		SMALL	0.726(80%)	0.753(20%)	0.731	29.6	0.742(100%)	-	0.742	58.3	-1.50%
	LOW	LARGE	0.275(20%)	0.315(20%)	0.295	36.0	0.275(20%)	0.292(80%)	0.289	64.9	2.03%
		SMALL	0.461(80%)	0.464(20%)	0.462	29.3	0.464(100%)	-	0.464	52.7	0.43%

40	HIGH	LARGE	0.514(40%)	0.565(20%)	0.540	87.2	0.530(40%)	0.546(40%)	0.538	192.5	0.37%
		SMALL	0.889(20%)	0.891(80%)	0.890	76.4	0.889(60%)	0.891(40%)	0.890	169.3	0%
	MID	LARGE	0.420(40%)	0.427(60%)	0.424	85.1	0.420(60%)	0.427(20%)	0.422	188.5	0.47%
		SMALL	0.703(100%)	-	0.703	74.3	0.700(20%)	0.703(80%)	0.702	161.2	0.14%
	LOW	LARGE	0.300(20%)	0.307(80%)	0.306	85.0	0.300(40%)	0.307(40%)	0.304	183.4	0.65%
		SMALL	0.477(20%)	0.508(60%)	0.493	69.8	0.477(20%)	0.508(40%)	0.498	155.1	-1.01%
50	HIGH	LARGE	0.487(60%)	0.500(20%)	0.493	164.8	0.487(60%)	0.500(20%)	0.493	833.4	0%
		SMALL	0.916(20%)	0.943(20%)	0.930	156.9	0.887(80%)	0.892(20%)	0.888	667.3	4.52%
	MID	LARGE	0.366(80%)	0.375(20%)	0.368	151.5	0.366(100%)	-	0.366	829.5	0.54%
		SMALL	0.693(20%)	0.719(40%)	0.706	115.7	0.665(60%)	0.693(20%)	0.672	628.6	4.82%
	LOW	LARGE	0.245(100%)	-	0.245	142.8	0.245(100%)	-	0.245	784.2	0%
		SMALL	0.462(40%)	0.468(20%)	0.465	112.3	0.431(20%)	0.468(40%)	0.456	602.4	1.94%

-: Not applicable (i.e. If all generated solutions have the same value, they are all defined as the best solution so that the corresponding worst solution is not applicable.)

CHAPTER 6

A SIMULATION STUDY OF AGV DISPATCHING RULES IN TANDEM AGV SYSTEMS

Simulation is an important tool for analyzing and studying the complex systems, which can be applied broadly in either engineering or other areas. Simulation can be defined as: the process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behavior of the system and/or evaluating various strategies for the operation of the system (Pegden et al., 1995).

In previous studies, simulation modeling has been used as a significant technique in AGV system design. Lee et al. (1991) developed a simulation model for evaluating the conventional AGV systems in SIMAN. The AGV network was defined first including the number of AGVs, the number of stations and the speed of AGVs. Then they designed the control system, such as the empty vehicle dispatching rules and routing criteria. The performance measures were obtained finally based on the model including the AGV utilization, average stations output queues, average waiting times in intersections, average throughput of the system, and average waiting time in queues.

Bozer and Srinivasan (1992) used simulation modeling in WITNESS to compare the performance of tandem and conventional AGV systems. They considered AGV utilization, average output queue of stations, and the average time spent in systems by the jobs as the performance measures. Two dispatching rules for the tandem system were tested and it was proved that STTF gave better results than FEFS. In addition, they concluded that the tandem layout outperformed the conventional layout for small systems.

Choi et al. (1994) also used simulation in SIMAN to compare the performance of the tandem and conventional systems. The performance measures were the total number of jobs completed, average job flow time, average AGV utilization, and the total blocking time. It was found that the tandem configuration could produce more job completions and reduce the blocking time. On the other hand, the conventional system performed better in the average job flow time and AGV utilization.

One of the most important results obtained in previous evaluation literature is that in conventional systems, the choice of dispatching rule does not influence mean flow time, machine utilization, AGV utilization, or maximum input queue length; it only affects the length of output queues (Russel & Tanchoco, 1984). It was also concluded that the LQS rule performs better than the STTF rule, which in turn has a better performance than the FCFS rule (Shang, 1995).

The tandem configuration greatly reduces the operational problems usually encountered in conventional systems and the complexity of the required control system due to the following reasons:

- When a station requires an AGV, only one AGV can be sent to fulfill the required move request eliminating many of the dispatching problems.
- An AGV can always reach its destination through the shortest route.
- Traffic management is no longer needed since there will never be two or more AGVs that may be occupying the same point in the path.

Due to the above reasons, the operational issues in tandem systems can be reduced to the choice of the vehicle initiated empty vehicle dispatching rule used to respond to simultaneous requests of workstations for an AGV in a zone.

In this chapter, the performance of four different empty vehicle dispatching rules towards three different system performance criteria will be evaluated. The dispatching rules are the STTF, FEFS, LQS, and the FCFS, and the performance criteria are the average AGV workload in the system, system throughput, and the average queue length (*QL)*. The STTF rule is the one adopted in the utilized partitioning algorithm, and was further proved to outperform the FEFS rule in a previous study (Bozer and Srinivasan, 1992). The FEFS rule was the one adopted in the partitioning algorithms proposed by Bozer and Srinivasan (1992) and by Kim and Jae (2003). The LQS and the FCFS rulers were the ones most studied for conventional systems. It is the objective of this study to acquire an idea of the performance of these four rules when applied in tandem systems.

6.1 Simulation experiments

To evaluate the performance of the four different empty-vehicle dispatching rules in tandem systems as mentioned earlier, simulation is conducted on the 20-stations problem (Bozer and Srinivasan, 1992) presented in the Chapter 3. Simulation modeling has been the dominant evaluation technique in most previous studies concerned with AGV systems design because of the randomness and the dynamic behavior of these systems. The two configurations (4 zones and 6 zones) obtained for this problem are both simulated four times each using the four rules. The commercial software WITNESS is used for the simulation purposes, and the values of the three performance criteria are collected. The input data of the simulation model is collected from the literature (Bozer and Srinivasan, 1992; Yu and Egbelu, 2001). The following assumptions and inputs governed the simulation runs:

- Input and output buffers of stations have infinite capacities.
- The time taken by an AGV to travel between the input and the output buffers of a station is negligible.
- AGV tracks are bi-directional and the AGVs always select the shortest route to reach their destinations.
- AGVs always move in a rectilinear manner.

- The speed of all the AGVs and conveyors (for handling between transfer points) are all 15 minutes per unit distance.
- Each simulation run is first warmed up for 10000 minutes followed by 5 replications 6000 minutes long each.
- The average processing time for each workstation is set equal to the value that yields an average workstation utilization of 75%.

Table 6-1: Summarized simulation results

Zones	**Rule**	**Average Workload**	**Average Throughput Rate (unit/min)**	**Average *QL***
4	STTF	0.548	0.148	1.470
4	LQS	0.548	0.149	1.562
4	FCFS	0.558	0.149	1.542
4	FEFS	0.270	0.148	1.680
6	STTF	0.348	0.148	1.449
6	LQS	0.348	0.149	**1.426**
6	FCFS	0.348	0.149	1.446
6	FEFS	**0.186**	0.149	1.558

Table 6-1 displays the average values of the three performance criteria (average AGV workload (*WL*), average system throughput rate (*TP*), and average *QL*) obtained for the four rules using the two configurations from the five replications. It can be easily noticed that the FEFS rule obtained a better average AGV workload for the two configurations than the other three dispatching rules. This is because, for the FEFS rule, the AGV workload was estimated in the simulation based only on the time when the AGVs were traveling loaded, while it was estimated based on the sum of the loaded and the empty traveling times for the other rules. The reason is that, for the other three rules, it was assumed that the AGVs will go to the home position after finishing a loaded trip, and this assumption was not valid for the FEFS rule, since under this rule, the AGV keeps traveling empty along the guide-path until it encounters a waiting load.

6.2 ANOVA analysis

Again, it can be noticed from Table 6-1 that the FEFS rules performed poorly regarding the average *QL* criterion. Further observations or conclusions can not be easily conducted from the shown results. Consequently, a more detailed analysis tool is required to attain rigorous conclusions. Design of experiments (DOE) and analysis of variance (ANOVA) are thus conducted. Two factors are considered in the analysis, the number of zones and the dispatching rule. The former will be only considered to study the interaction effect it has with the dispatching rules on the different performance criteria. This is because, in tandem systems, more zones means more AGVs, and thus less average AGV workload and better system performance in general. The zones factor has two levels, four zones and six zones, and the dispatching rules factor has four levels, each representing a different dispatching rule.

The ANOVA analysis is conducted using the commercial software MINITAB R14.2 with a confidence level of 95%. The *p*-values of the main and the interaction effects of the two factors on the average workload, system throughput, and average *QL* are shown in Table 6-2.

Table 6-2: ANOVA results

Source	Average Workload	System Throughput Rate (unit/min)	Average *QL*
Zones	**0.000**	0.978	**0.002**
Rule	**0.000**	0.999	**0.000**
Zones * Rule	0.462	1.000	0.353

It can be noticed from table 6-2 that changing the number of zones and the applied dispatching rule had significant effects on the resulting average AGV workload and average *QL*. Also table 6-3 shows that the system throughput was not affected by either the number of zones (AGVs) or the dispatching rule. As mentioned earlier, increasing the number of zones means increasing the number of AGVs in the system, and thus distributing the system workload on more AGVs. Consequently, increasing the number of zones has a positive effect on the average AGV workload, and the average *QL*. However, this increase had no effect on the system throughput which may be due to the fact that the current system workload was not high enough in order for the throughput to increase by increasing the number of AGVs. As for the main effects of the dispatching rules on the average AGV workload and the average *QL*, these could be better understood by analyzing the following figures. Figure 6-1 and Figure 6-2 show the main effects of changing the dispatching rule on the average AGV workload and on the average *QL* respectively.

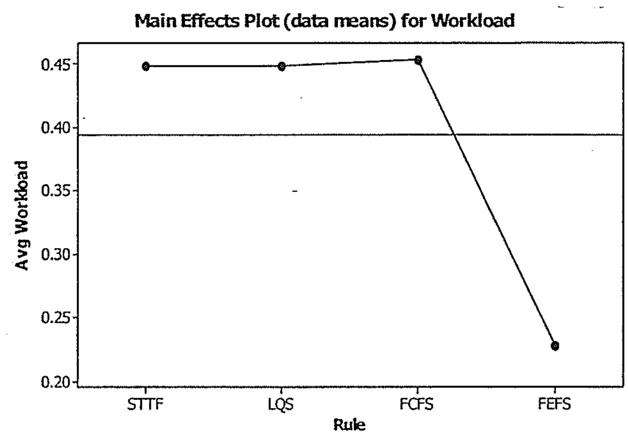


Figure 6-1: Effect of dispatching rules on average AGV workload

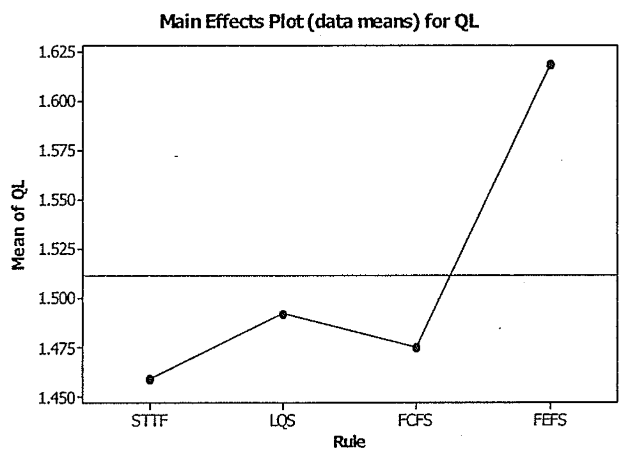


Figure 6-2: Effect of dispatching rules on average *QL*

The above figures show that the significant effects of changing the dispatching rule on the workload and the *QL* are mainly due to the FEFS rule. As mentioned earlier, the effect of the FEFS rule on the average AGV workload can be neglected due to the AGV empty travel time consideration in the other three rules. As for the average *QL*, figure 6-2 shows that the FEFS rule leads to longer queues before and after workstations. As for the other rules, it is clear that they all have the same performance regarding the average AGV workload, however, the STTF and the FCFS rules perform slightly better than the LQS rule when considering the average *QL*.

6.3 Summary

In this chapter, an analysis of the performance of different empty vehicle dispatching rules towards a number of system performance criteria was conducted. It was found that the selection of such rules has no effect on the system throughput. It was also found that the STTF, the FCFS, and the LQS perform equally with regard to the average AGV workload in the system and with a slight difference with regard to the average *QL* in favor of the first two rules. The FEFS rule performed differently than the other three rules as it has a negative effect on the average *QL*.

CHAPTER 7

CONCLUSIONS AND FUTURE RESEARCH

7.1 Conclusions

The design of tandem AGV systems discussed in this study includes the partitioning and dispatching issues. It was summarized from the literature review that some points regarding the partitioning issue have not been considered. Instead of meta-heuristics, most researchers have developed heuristic algorithms to design the tandem configuration. The STTF empty vehicle dispatching policy was only applied once in the partitioning process in Shalaby et al. (2006)'s work. A specific mechanism has not been developed to check the presence of overlapping zones.

In this study, a genetic algorithm is proposed to solve the partitioning problem of the tandem AGV systems. The objective of minimizing the maximum workload is considered. In the proposed algorithm, first, the initial population is generated by a k-means clustering method to avoid the presence of overlaps among zones. Three types of infeasibility conditions are discussed in this study. Solutions with empty zones are rejected directly. The other two kinds, solutions with singleton zones and solutions with overlapping zones, are repaired using a repair procedure. Singleton zones, where only one station exists in a zone, are expanded by including the nearest station(s), provided that no more singleton zones are created by this adjustment. The overlaps between zones are repaired by adding the stations into the zone which has the nearest center to them. This strategy ensures that all the generated solutions are feasible. Design of experiments has been used to tune the parameters of GA (the population size, mutation rate, and crossover rate). The proposed algorithm is evaluated by comparison with the reported results in the literature. The results showed that the performance of the proposed algorithm is superior compared to the algorithms reported in previous studies. Moreover, the CPU time has been tested and shows that the proposed algorithm is efficient.

In addition, a group of experiments have been designed to assist the GA in improving its performance under different conditions. The considered factors in these experiments include system size, number of zones, and zone loading. The parameters of GA that are tuned include population size, crossover rate, and mutation rate. The effects of these factors on the solution quality and the computational time are observed though analysis of variance (ANOVA). The best combinations of the parameters' values for different system characteristics are found through this design of experiments.

To improve the performance of GA, a local search is developed and combined with the developed GA to solve the partition of the tandem AGV systems. The combination of the local search with GA is referred to as memetic algorithm (MA). The local search is applied to each newly generated individual to move it to a local optimum before injecting it back into the population. This procedure is carried out on every population including the initial population. The proposed local search method involves moving and swapping of stations depending on the number of stations in the target zone. The comparison of results showed that MA outperformed GA in terms of the solution quality. However, for large size problems, the computational time of MA seemed to be higher. However, the computational time of MA is still within the accepted range for practical problem sizes.

Finally, a simulation study of the performance of different empty vehicle dispatching rules in tandem AGV systems was performed. It was found that the selection of such rules has no effect on the system throughput. It was also found that the STTF, the FCFS, and the LQS perform equally with regard to the average AGV workload in the system. The STTF and the FCFS rules perform slightly better than the LQS rule when considering the average *QL*. The FEFS rule performed differently than the other three rules as it has a negative effect on the average *QL*.

7.1 Future research

The points that can be considered in the future are summarized as follows:

- The objective of partitioning stations into different zones in the current study is to minimize the maximum workload. However, more objectives may be served by the same problem, such as minimizing the material handling cost, or minimizing the number of zones.
- Other vehicle dispatching rules may be considered instead of STTF in the

partitioning process. Also, other dispatching rules can be used to analyze the performance of the system. The effects of other system factors on the performance of the dispatching rules can be studied as well. These may include the number of workstations, system workload, and the speed of the vehicles.

- One of the factors that contribute to consuming time in the proposed algorithms is using the traveling sales man (TSP) model in the vehicle routing procedure. The efficiency of the algorithm will be improved significantly if a better way of deciding the vehicle routes can be found.

REFERENCES

Aarab, A., Chetto, H. and Radouance, L., 1999, "Flow Path Design for AGV Systems", Studies in Informatics and Control, 8, 2, 97-106.

Bagchi, T.P., and Deb, K., 1996, "Calibration of GA Parameters: The Design of Experiments Approach", Computer Science and Informatic, 26, 4, 46-56.

Bilge, U., Esenduran, G., Varol, N., Ozturk, Z., Aydin, B., and Alp, A., 2006, "Multi-attribute Responsive Dispatching Strategies for Automated Guided Vehicles", International Journal of Production Economics, 100, 65-75.

Buriol, L., Franca, P. M. and Moscato, P., 2004, "A New Memetic Algorithm for the Asymmetric Traveling Salesman Problem", Journal of Heuristics, 10, 483-506.

Bozer, Y. A. and Lee, C-G., 2004, "Using Existing Workstations as Transfer Stations in Tandem AGV Systems", Journal of Manufacturing Systems, 23, 3, 229-241.

Bozer, Y.A., and Srinivasan, M.M., 1989, "Tandem Configurations for Automated Guided Vehicle Systems Offer Simplicity and Flexibility", Industrial Engineering, 21, 2, 23-27.

Bozer, Y.A., and Srinivasan, M.M., 1991, "Tandem Configurations for Automated

Guided Vehicle Systems and the Analysis of Single vehicle loops", IIE Transactions, 23, 1, 72-82.

Bozer, Y.A., and Srinivasan, M.M., 1992, "Tandem AGV systems: A Partitioning Algorithm and Performance Comparison with Conventional AGV Systems", European Journal of Operational Research, 63, 2, 173-192.

Choi, H-G., Kwon, H-J., and Lim, J., 1994, "Traditional and Tandem AGV System Layouts: A Simulation Study", Simulation, 63, 85-93.

Chu, P.C., and Beasley, J.E., 1998, "Constraint Handling In Genetic Algorithms: The Set Partitioning Problem", Journal of Heuristics, 11, 323-357.

Chuang, L.B-S. and Heim, J.A., 1996, "A Simulation Test Bed for Investigation of Tandem AGV Systems", Proceedings of the 1996 Winter simulation Conference, 1115-1122.

Davis, L., 1993, "Handbook of Genetic Algorithms", Van Nostrand Reinhold.

Egbelu, P.J., and Tanchoco, J.M.A., 1986, "Potentials for Bi-directional Guide-path for Automated Guided Vehicle Based Systems", International Journal of Production Research, 24, 5, 1075-1097.

Franca, P. M., Mendes, A. and Moscato, P., 2001, "A Memetic Algorithm for the Total Tardiness Single Machine Scheduling Problem", European Journal of Operational Research, 132, 224-242.

Ganesharajah, T., Hall, N.G., and Sriskandarajah, C., 1998, "Design and Operational Issues in AGV Served Manufacturing Systems", Annuals of Operations Research, 76, 109-154.

Gaskins, R.J., and Tanchoco, J.M.A., 1987, "Flow Path Design for Automated Guided Vehicles", International Journal of Production Research, 25, 667-676.

Gen, M., and Cheng, R., 2000, "Genetic Algorithms and Engineering Optimization: Engineering Design and Automation", New York: John Wiley & Sons.

Goetz, Jr., W.G., and Egbelu, P.J., 1990, "Guide Path Design and Location of Load Pick-up/drop-off Points for an Automated Guided Vehicle System", International Journal of Production Research, 28, 927-941.

Grefenstette, J., 1986, "Optimization of Control Parameters for Genetic Algorithms", IEEE Transactions on Systems, Man, and Cybernetics, 16, 1, 122-128.

Ho, Y-C., and Hsien, P-F, 2004, "A Machine-to-loop Assignment and Layout Design

Methodology for Tandem AGV Systems with Multiple-load Vehicles", International Journal of Production Research, 42, 4, 801-832.

Hsieh, L-F., and Sha, D.Y., 1996, "A Design Process for Tandem Automated Guided Vehicle Systems: the Concurrent Design of Machine Layout and Guided Vehicle Routes in Tandem Automated Guided Vehicle Systems", Integrated Manufacturing Systems, 7, 30-38.

Huang, C., 1997, "Design of Material Transportation System for Tandem Automated Guided Vehicle Systems", International Journal of Production Research, 35, 943-953.

Jeong, B.H., and Randhawa, S.U., 2001, "A Multi-attribute Dispatching Rule for Automated Guided Vehicle Ssystems", International Journal of Production Research, 39 13, 2817-2832.

Kaspi, M., and Tanchoco, J.M.A., 1990, "Optimal Flow Path Design of Unidirectional AGV Systems", International Journal of Production Research, 28, 6, 1023-1030.

Khan, S.S., and Ahmad, A., 2004, "Cluster Center Initialization Algorithm for K-means Clustering", Pattern Recognition Letters, 25, 1293-1302.

Kim, K.S., and Jae, M., 2003, "A Design for a Tandem AGVS with Multi-load AGVs", International Journal of Advanced Manufacturing Technology, 22, 9, 744-752.

Kim, J., and Klein, C.M., 1996, "Location of Departmental Pickup and Delivery Points for an AGV System", International Journal of Production Research, 34, 407-420.

Kim, C.W., and Tanchoco, J.M.A., 1993, "Operational Control of a Bidirectional Automated Guided Vehicle System", International Journal of Production Research, 31, 2123-2138.

Kodali, R., 1997, "Knowledge-based System for Selection of an AGV and a Work Centre for Transport of a Part in Online Scheduling of FMS", Production Planning and Control, 8, 2, 114-122.

Kouvelis, P., Gutierrez, G.J., and Chiang, W-C., 1992, "Heuristic Unidirectional Flow Path Design Approaches for Automated Guided Vehicle Systems", International Journal of Production Research, 30, 1327-1351.

Laporte, G., Farahani, R.Z., and Miandoabchi, E., 2006, "Designing an Efficient Method for Tandem AGV Network Design Problem Using Tabu Search", Applied Mathematics and Computation, 183, 1410-1421.

Lin, J.T., Chang, C.C.K., and Liu, W-C., 1994, "A Load-routing Problem in a Tandem-configuration Automated Guided-Vehicle System", International Journal of Production Research, 32, 411-427.

MacQueen, J. B., 1967, "Some Methods for Classification and Aanalysis of Multivariate Observations", Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1, 281-297

Merz, P. and Freisleben, B., 2000, "Fitness Landscape Analysis and Memetic Algorithms for the Quadratic Assignment Problem", IEEE Transaction on Evolutionary Computation, 4, 4, 337-352.

Merz, P. and Freisleben, B., 2001, "Memetic Algorithm for the Traveling Salesman Problem", Complex Systems, 13, 4, 297-345.

Montgomery, D.C., 2005, "Design and Analysis of Experiments (Sixth Edition)", New York: John Wiley & Sons.

Moscato, P., and Norman, M.G., 1992, "A 'Memetic' Approach for the Traveling Salesman Problem Implementation of a Computational Ecology for Combinatorial Optimization on Message-passing Systems", Parallel Computing and Transputer

Applications, 1, 177-186.

Naso, D. and Turchiano, B. 2005, "Multicriteria Meta-Heuristics for AGV Dispatching Control Based on Computational Intelligence", IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, 35, 2, 208-226.

Radcliffe, N.J., and Surry, P.D., 1994, "Formal Memetic Algorithms", Evolutionary Computing, AISB workshop, selected papers, 1-16.

Rojas, I., Gonzalez, J., Pomares, H., Merelo, J.J., Castillo, P.A., and Romero, G., 2002, "Statistical Analysis of the Main Parameters Involved in the Design of a Genetic Algorithm", IEEE Transactions on Systems, Man, and Cybernetics, 32, 1, 31-37.

Ross, E.A., Mahmoodi, F. and Mosier, C.T., 1996, "Tandem Configuration Automated Guided Vehicle Systems: A Comparative Study", Decision Sciences, 27, 81-102.

Saremi, A., ElMekkawy, T.Y., and Wang, G.G., 2007, "Tuning the Parameters of a Memetic Aalgorithm to Solve Vehicle Routing Problem with Backhauls Using Design of Experiments", International Journal of Operations Research, 4, 4, 1-14.

Seo, Y., and Egbelu, P.J., 1995, "Flexible Guide Path Design for Automated Guided Vehicle Systems", International Journal of Production Research, 33, 1135-1156.

Shalaby, M.A., ElMekkawy, T.Y., and Fahmy, S.A., 2006, "Zones Formation Algorithm in Tandem AGV Systems: A Comparative Study", International Journal of Production Research, 44, 505 – 521.

Sinriech, D., and Tanchoco, J.M.A., 1991, "Intersection Graph Method for AGV Flow Path Design", International Journal of Production Research, 29, 1725-1732.

Smith, T., and Sarin, S., 1992, "A Program for Dispatching and Routing AGVs", Computers and Industrial Engineering, 23, 187-190.

Srinivas, M., and Patnaik, L., 1994, "Adaptive Probabilities of Crossovers and Mutations in Genetic Algorithms", IEEE Transactions on Systems, Man, and Cybernetics, 24, 4, 656-667.

Wang, H-P.B., and Hafeez, S.A., 1994, "Performance Evaluation of Tandem and Conventional AGV Systems Using Generalized Stochastic Petri Nets", International Journal of Production Research, 32, 917-932.

Yu, W., and Egbelu, P.J., 2001, "Design of a Variable Path Tandem Layout for

Automated Guided Vehicle Systems", Journal of Manufacturing Systems, 20, 305-319.