THE AGV SCHEDULING PROBLEM WITH ALTERNATIVE PICK UP AND DELIVERY NODES

INTRODUCTION

Productivity and flexibility, which are the primary goals of today's automation technology, can only be achieved in fully integrated manufacturing environments. A carefully designed and efficiently managed material handling system is an important part. The study of automated guided vehicle (AGV) system was initiated in 1987 by Tanchoco and Moodie. AGV systems are among the fastest growing classes of equipment in the material handling system in industry. AGVs are battery-powered, unmanned vehicles with programming capabilities for selecting the traveling path, positioning the pick up and delivery points, responding to frequently changing transport patterns, and integrating into fully automated intelligent control systems. These features make AGVs to be a viable alternative to other material handling methods, especially in job shop environments where the variety of products are proceeded in fluctuating transport requirements. In such a dynamic and sophisticated environment, the job scheduling is one of the key factors in a successful implementation of an AGV system.

Both job sequencing and scheduling are important parts of any kind of vehicle routing design problem, including an AGV system design. Designing an AGV system is a complex task because of factory layout, the number of nodes and the AGV's traffic system. One of the main purposes of a single/multi AGV scheduling problem is concerning about how the scheduling can provide the minimum total traveling distance of AGVs. Normally, the scheduling problem is considered or designed with the routing problem concomitantly, for any kind of vehicle system management. The ordinary vehicle scheduling and routing problem, as the single/multi AGV scheduling problem, is a problem with a set of specific pick up and delivery nodes that can be modeled by the existing network problem approach, which is the such as TSP/MTSP.

The TSP/MTSP is one of the most interested approaches because there is a network structure that can be modified and applied to the AGV scheduling problem. Because Dantzig, Fulkerson and Johnson (1954) proposed that determining the optimal solution of the TSP for large numbers of nodes requires much time, heuristic methods are considered when the TSP is applied to any kind of problems. Many papers proposed heuristic algorithms for finding the AGV scheduling and traveling path such as Maxwell and Muckstadt (1982), Gaskins and Tanchoco (1987). As NP-hard nature of the original TSP, the vehicle scheduling problem with alternative pick up and delivery (P/D) nodes may be considered as a class of NP-hard problem also when the problem structure falls into the TSP category. According to this point, the potential problem for studying the single/multi AGV scheduling problem is extended to be more realistic when the original TSP problem is modified by adding the structure of alternative P/D nodes. The main purpose is to find the solution of AGV scheduling problem with alternative P/D nodes (AGVsp-P/D) that can provide the minimum total traveling distance of AGVs.

The original TSP/MTSP is one of the applications of network problems; it is necessary to choose a sequence of nodes to be visited so as to accomplish a specified objective. The TSP/MTSP is a network problem that given a network and a cost (or distance) associated with each arc, it is necessary to start from a specified originating or depot node, visit each and every other nodes exactly one, and return to the starting node with the lowest cost. For example, a bus that leaves the school yard must stop at various locations to pick up students and ultimately return to the school yard in the shortest possible distance. As another example, research considers the AGV system that can start from a specified originating or depot node, visit each and every other nodes, which have some alternatives for selection to visit exactly once, and return to the starting node in the shortest distance. The TSP/MTSP can be solved to determine the scheduling of normal uncapacitated vehicle routing problems but in this case, the original TSP/MTSP has to be modified to support the AGVsp-P/D.

The concept of TSP/MTSP will be applied with some generated techniques of assignment problem to solve the AGVsp-P/D to determine the minimum traveling

distance of each AGV from the starting node or depot to some appropriate selected nodes, and then come back to the starting node. This procedure is based on the branch and bound process with solving assignment subproblems to search for the optimal tour. The formulated mathematical model will be presented in this research. The assignment subproblems with alternative P/D nodes, and the branch and bound algorithm for TPS/MTSP with alternative P/D nodes are considered as an important part of this research. The assignment subproblem model and the solving approach for finding the lower bound of the AGVsp-P/D will be proposed. The ordinary assignment problem is an integer programming (IP) problem, but it has special structures that make it can be solved by linear programming (LP) approach, not considering IP constraints. When the constraint of alternative P/D nodes is added to the system, the problem loses the property of regular assignment problem, which causes it becomes the 0-1 IP problem.

Thus, a new 0-1 IP model of assignment problem with alternative P/D nodes is created. The implementation of the generated model is tested using the Excel Solver and MATLAB 7.0. After the lower bound of the AGVsp-P/D is found by solving the assignment problem with alternative P/D nodes, the branch and bound algorithm for finding the TSP/MTSP with alternative P/D nodes will be studied. Because the branch and bound approach takes much time for the large problem, the heuristics for solving the lower bound of the AGVsp-P/D are proposed. Benders'decomposition approach is applied to create the heuristic for solving the lower bound of the AGVsp-P/D. However, this Benders'decomposition algorithm still uses the 0-1 IP problem, but with a smaller problem size than the direct method. To avoid solving the 0-1 IP problem, three heuristics for selecting the alternative nodes and an alternative selection improvement heuristic are proposed.

The lower bound solutions from solving both the 0-1 IP problem and heuristic approaches may provide the single TSP tour or subtours. For the single TSP tour solution, the subtour elimination approach is applied to the lower bound solution to create the single TSP tour from subtours. The modified Eastman's algorithm for TSP with the lower bound model of the AGVsp-P/D is proposed for solving the single TSP

tour. When multi AGVs are considered, heuristics for solving the multi AGVsp-P/D are presented. There are two approaches, which are the heuristic of splitting the single TSP tour to multi tours for the lower bound of the multi AGVsp-P/D and the approach of solving the MTSP as the standard TSP for the solution of the multi AGVsp-P/D.

Finally, the computer program for solving single/multi AGVsp-P/D using the Excel Solver and MATLAB 7.0 are developed for testing the model of AGVsp-P/D. The program of the 0-1 IP problem of AGVsp-P/D and heuristics are applied to some size levels of tested problems. The tested results are analyzed by statistical methods to verify the performance and quality of the AGVsp-P/D model.

An introduction to the research on the AGVsp-P/D including research questions, problem statement, research objectives, research significance and research assumption will be presented as follows.

Research Questions

Consider the modern manufacturing system, the AGV system is used to transport items among departments in the factory. Let's assume the problem that the factory has a particular layout of departments for an AGV system, as in figure 1. The AGV layout can be drawn as a network. The AGVs move through the network between nodes (labeled A, B, C, D, E, F, G, H, and I). Bidirectional flow of the AGVs along the aisles is assumed.

In general, each job of AGVs consists of picking up a load at one node and delivering it to a fixed destination node. For this research, the special characteristic of alternatives P/D nodes is represented by some jobs that can have alternative pick up and delivery nodes at more than one fixed point. For example, let job No. 1 of the AGV is to pick up an item from a turning process at department (node) B and deliver to a drilling process which can be performed at departments E or G or I. The AGV has to travel from pick up the node B and can choose to deliver the item to only one node at departments E or G or I, which is described as alternatives P/D nodes. If an

AGV travels from node B and selects to deliver to node E, the total traveling distance of AGVs may different from selects to deliver to node G or node I. An example of a job list for AGVsp-P/D is shown as table 1. The job scheduling (for example, starting with job No. 1, follows by job No. 6, No. 5, and ending the schedule when all jobs are completed) and selecting the appropriate alternative node effects on the traveling distance of the AGV. The objective is the selecting of alternatives and scheduling for all jobs such that the total AGV traveling distance is minimized. Therefore, the research questions can be described as follows:

1. Given the information of some specific daily tasks of a specific factory and a specific route path with the distances among departments, what is the scheduling of the single AGV with appropriate selected alternative P/D nodes that can provides the minimum total traveling distance? An example of specific daily tasks is shown on table 1.

2. Given the information of some specific daily tasks of a specific factory, a specific route path with the distances among departments and a specific number of AGVs, what is the schedul of each AGV with appropriate selected alternative P/D nodes that can provide the minimum total traveling distance?

Table 1 The example of AGVsp-P/D jobs

Job No.	Pick up Department	Delivery Department
1	B	C
2	A	I
3	B	E or G or I
4	G	C
5	D	E
6	B or D or H	F
7	I	C or D or E
8	C	H
9	F	E
10	H	I or G

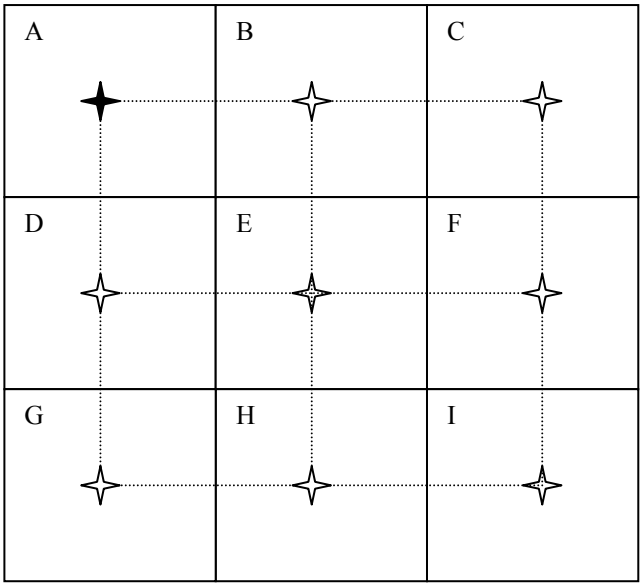


Figure 1 The example of the layout of AGV systems

Problem Statement

Let the problem has a set of n jobs J such that job $J_i = \{P_{i\,a}, D_{i\,b}\}$, $i = 1, 2, \ldots, n$ where $P_{i\,a}$ is a set of alternative pick up departments a of job J_i, $a = \{1, 2, \ldots, k(i)\}$ and $D_{i\,b}$ is a set of alternative delivery departments b of job J_i , $b = \{1, 2, \ldots, l(i)\}$. $k(i)$ is the number of alternative departments a for job J_i. $l(i)$ is the number of alternative departments b for job J_i. When job $J_j = \{P_{j\,a}, D_{j\,b}\}$, $j = 1, 2, \ldots, n$ is scheduled after job J_i, $c_{ia\,jb}$ is the traveling distance of an AGV that starts from a selected pick up department a of job J_i , goes to a selected delivery department b of job J_i , goes to a selected pick up department a of job J_j , then goes to a selected delivery department b of job J_j , which is a non-negative number and $c_{ia\,ia} = \infty$.

The AGVsp-P/D is the problem that selects one alternative department from set a and one alternative department from set b of all jobs J, called $x_{ia\,jb}$ such that $x_{ia\,jb} = 1$ if an AGV travels from a selected pick up department a of job J_i to a selected delivery department b of job J_j or $x_{ia\,jb} = 0$ otherwise and sequences all those jobs J

with their selected alternatives to form single/multi tours (TSP/MTSP tour) that can provide the minimized total traveling distance.

Research Objectives

1. To study and develop the mathematical model of the single/multi AGVsp-P/D that can describe the characteristics and structures of this problem.

2. To develop heuristic algorithms for solving the problem.

3. To create the code or program of developed algorithms by using a builder software for solving the problem with some specific sizes and structures of problem as a tool for verifying and validating efficiency and quality of proposed algorithms.

4. To analyze statistically the result of solving tested problems by using proposed algorithms.

Research Significance

The trend of the modern manufacturing industry is to become more computerized and automated systems. Improvements in production planning with respect to the scheduling process of traveling vehicles (AGVs) will provide more effective production planning that helps to improve the productivity. This research establishes some kind of algorithms or production planning tools in the form of static models that provide near-optimal solutions for the AGVsp-P/D which never been studied and modeled before. The established algorithms are the extension and modification from the existing AGV scheduling problem to capture the special structure of the AGVsp-P/D.

Research Scope and Assumptions

1. This research considers only the constant speed AGV with undirected paths in manufacturing factories.

2. The daily task of the AGV system is considered to be a static condition during the shift period.

3. The task of pick up and delivery activities is considered as a unit load that can be assigned to only one AGV or can not be spitted.

4. This research results provide the mathematical model of the single/multi AGVsp-P/D that can describes the structures and characteristics of this problem for analysis and developments of all solving procedures.

LITERATURE REVIEW

This chapter provides the background information on the key subjects for this research. The first part of this chapter presents the AGV systems which explain about the vehicle and the driving system. The description of all types of AGVs, function criteria, and the dispatching systems are explained. The nature of dispatching systems can be related to the scheduling approach, which is the main proposal of the research.

The next part explains about the AGV problems which all factories, using AGVs, have to face with. Many cases of AGV problems are explained and analyzed in order to diagnose and solve the problem. Then single/multi traveling salesman problem (TSP/MTSP) is explained next with its applications and transformations. This part presents the mathematical model of TSP/MTSP that can be applied to solve many real world problems of vehicle routing applications. The transformation approach for solving the MTSP as a standard TSP is reviewed for generating the concept of solving the multi AGVsp-P/D.

The last part explains about the relevant statistical methods for analyzing the data from the research results. The probability distribution of data sets is the first issue that should be considered because most of statistic methods assume the normal probability distribution of the data set. The normality test is explained in this part. Then, the statistical hypothesis test and the analysis of variance, which are applied to analyze some parts of research results, are reviewed.

Automated Guided Vehicle Systems

The AGV system is one of the most exciting and dynamic areas in material handling systems. AGV systems were invented around 1950's, they were called the driverless systems. AGV systems combine electromagnetic technology with existing industrial truck equipment to create more flexible and self-steering vehicles. Technological developments may have given AGVs more flexibility and capability for operating in computer integrated manufacturing systems. In the future manufacturing, AGV systems are expected to be widely used as material handling equipments. These vehicles transport tools and materials among different work cells in flexible manufacturing systems. AGVs are programmed independently but all of them are correlated with the scheduling and the traffic control system. These characteristics confer the flexibility and the adaptability to the material handling system. AGVs circulate on a network of guide paths connecting the various work cells at load transfer points, also called P/D nodes, which are located on paths of the network. In the design of AGV systems, many types of design problems can be identified such as a design of the network layout, a design of load pick up and delivery point locations, a design of fleet size, and a design of traffic management systems.

Egbelu (1986) proposed the paper of AGV dispatching heuristics that are related to the pull versus push strategy for AGV load movement in a batch manufacturing system. The purpose of this paper is to justify the use of demand based dispatching rules for AGVs in the manufacturing system. The algorithm of the pull strategy (demand base) algorithm is presented and compared to several push strategy (source base) algorithms to demonstrate its effectiveness. The traditional source based dispatching rules do not have the flexibility, required by just-in-time (JIT) manufacturing systems, so there is a need to develop some useful dispatching rules for such applications. When the pull strategy is used, direct access load retrieval systems must be used in order to pick up parts from any position of the queue, not only the first part. In developing the dispatching algorithm, some assumptions are made such as 1. A vehicle can transport only one unit load at a time, 2. No look-ahead capacity for

future events is considered, and 3. No job is assigned a global priority over all others at the time of entry into the production system.

A hierarchical demand driven dispatching rule is developed and tested against some commonly used source driven dispatching rules. There are mainly two steps for this algorithm. First is to identify workstations that have the greatest demand for all parts, then the sources of parts can be selected according to some preset rules. If no workstations meet the minimum requirement in the first step, the rule automatically reverts to a source driven rule. A FORTRAN based discrete event simulation language, AGVSim, is used to investigate the effectiveness of the proposed method. The author compares the pull system to the widely used push systems in three separate cases. From the simulation results of all cases, the demand driven dispatching rule proves itself to be the competitiveness of the push system. In all cases, the pull system shows that it is superior to the push system.

Tanchoco and Moodie (1987) proposed the special issue of the study of AGV systems in the Material Flow journal that consists of many points of view on the study of AGV systems. This special issue brings together, under one cover, a collection of papers dealing with new concepts in designing, planning, and analyzing**.** The paper by G.A. Koff (1987) provided an introduction to the AGV system, its major functions, and how these functions are executed. Koff illustrated that there are several types of AGV that they are:

1. AGV towing vehicles; were the first type introduced and are still a very popular type until now. It can pull a multitude of trailer. AGV towing applications involve the bulk movement of product into and out of warehouse areas. Often side-path spurs are place in receiving or shipping areas so that trains can be loaded or unloaded off the main line and thereby not hinder the movement of other trains on the main path.

2. AGV unit load vehicles; are equipped with decks which permit unit load transportation and often automated load transfer. AGV unit load applications usually

involve specific mission assignment for individual pallet movement. Unit load carries are quite popular in applications of integrating conveyors with storage-retrieval systems.

3. AGV pallet trucks; are designed to transport palletized loads to and form floor level and eliminate the need for fixed load stands. AGV pallet trucks are generally used in distribution functions. Vehicles can be loaded in two ways, either they are capable of automatically reversing into pallets on the floor or operators will manually board the vehicle and back them into pallets.

4. AGV fork trucks; are a relative new guided vehicle which has the ability to service palletized loads both at floor level and on stands. AGV fork trucks are used when the system requires the automatic pick up and delivery of loads from floor or stand level, and where the heights of load transfer vary at stop locations. The guided fork truck has the ability to pick up or deliver a load automatically without any human interface.

5. Light load AGVs; are vehicles which have capacities to transport small parts. They are design to operate in areas with limited space. Light load AGVs applications are used in light manufacturing processes. The product can be distributed from a small parts storage area to individual work stations where operators do light assembly.

6. AGV assembly line vehicles; are an adaptation of the light load AGVs for application involving serial-assembly processes. Assembly line AGV is adaptations of the small, light-load AGVs for an assembly line process. The guided vehicles carry major subassemblies such as motors or transmission to which parts are added in serial assembly process.

The basic functions of AGVs consist of five functions as follows:

1. Guidance: this function allows the vehicle to follow a predetermined route, which is optimized for the material flow pattern of a given application.

2. Routing: this function is the vehicle's ability to make decision along the guidance path in order to select optimal route to specific destination.

3. Traffic management: this function is a system or vehicle's ability to avoid collisions with other vehicles, while at the same time maximizing vehicle flows and therefore load movements throughout the system.

4. Load transfer: this function is the pick up and delivery method for AGVs, which may be simple or integrated with other subsystems.

5. System Management: this function is the method of system control used to dictate system operations. The proper method of selection for each function and its ability to work with the other functions is determines in by measuring the degree of successfulness of a given system.

The manufacturing industry consists of several machine centers performing different machining functions. A part or unit load visits several centers before its machining requirements are satisfied. A unit load continues to circulate in the shop among work centers until receiving the last service. It is the transition of unit loads or parts that generate the vehicle dispatching or task assignment problem in an AGV system.

Egbelu and Tanchoco (1984) presented some heuristic rules for dispatching AGV in a job shop environment. The vehicle dispatching decisions fall into two categories that are the work center initiated task assignment problem and the vehicle initiated task assignment problem. The first category is a decision involving the selection of a vehicle from a set of idle vehicles to assign to a unit load pick up task

generated at some parts of the factory. This class of decisions involves a single work center and one or more vehicles. The decision is generally the result of a request from a work center for vehicle service. The secondary category of decisions involves the selection of a work center from a set of work centers simultaneously requesting the services of any vehicle, a decision, which usually involves a single vehicle and multi work centers. The decision is to prioritize the departments and to dispatch vehicles to the departments with the highest priority. Two vehicle dispatching decisions are explained as follows:

1. The work center initiated task assignment problems: a typical machining center in an AGV system consists of one or more machines, an incoming unit load queue, and an outgoing unit load queue. The unit loads are drawn from the incoming queue, processed, and released into the output queue at the same rate. The deposition of a unit load into the output queue also initiates a request by the department for an unassigned vehicle for the immediate removal of the deposited load. Several heuristic rules can be employed to assign the priority of vehicles for dispatching such as Random Vehicle rule, Nearest Vehicle rule, Farthest Vehicle rule, Longest Idle Vehicle rule, and Least Utilized Vehicle rule.

2. Vehicle initiated task assignment problems: from an operational point of view, the most desirable level of handling effectiveness is to ensure that unit loads completed at a work center are removed promptly and transported to their subsequent destinations with a minimum of delays. Like the work center initiated task assignment problem, several heuristic rules are available for ranking work centers requesting unassigned vehicles. Possible assignment rules are:

1. Random Work Center rule
2. Shortest Travel Time/Distance rule
3. Longest Travel Time/Distance rule
4. Maximum Outgoing Queue size rule
5. Minimum Remaining Outgoing Queue Space rule
6. Modified First Come First Serve rule

Several combinations of all above rules are tested on a factory using a simulation technique. A simulation program, AGVSim, was developed specifically to simulate an AGV system. Using unit load throughput as a measure of rule performance with 2 trials per rule combination, all experiments are conducted under similar conditions. The demonstrations indicated that rules, which are derivatives of distance measures, have several drawbacks if the appropriate layout conditions of factory and equipment locations are not met.

In the design of an AGV system, one of the fundamental problems is the determination of the number of vehicles that are required to provide a given level of transportation service. There are so many methods for the fleet-size determination process that use mathematical or simulation based techniques. Tanchoco, Egbelu and Taghaboni (1987) proposed the effectiveness of CAN-Q software in determining the number of AGVs and compared to a simulation based method (AGVsim). The analysis indicates that the results obtained from the software provide a good starting search point for a simulation technique. When two approaches are used jointly, the number of simulation runs which is required to generate a solution is potentially reduced. Simulation is the most reliable method to data estimating vehicle requirements for complex system. However, since simulation is expensive in the cost of data correction and time consuming, several non-simulations based calculation approaches vehicle estimation are generated.

Egbelu (1987) proposed four cases of the method for estimating the number of vehicles through hand calculation. They are

Case 1: it is assumed that the distance covered by vehicles making empty run is equal to the distance traveled by loaded vehicles.

Case 2: it requires the estimation of blocking time factors and idle time factors. This estimation is used to refine the estimate on the vehicle requirement.

Case 3: this method requires the computation of the net traffic flow into the work center. For the work center i, the net in-flow is F_i.

- If $F_i > 0$, it implies that there are more number of vehicles coming to deliver a load into the work center *i* then coming to pick up a load. It is a net exporter of empty vehicle.
- If $F_i = 0$, the method assumes that no empty runs will be made from the work center *i*. Every vehicle that delivers a load from the center *i* will leave with a load to another center.
- If $F_i < 0$, it implies that the center *i* will be a net importer of empty vehicles.

Case 4; this method assumes a job shop environment for a work center *i*, the sequence of jobs, which request to pick up load are generated from the work centers that is varied randomly.

A modern manufacturing factory is usually managed by computer control systems that obtain the production plans and monitor the current state of each job. In such a dynamic and sophisticated environment, the job scheduling is one of key factors in a successful implementation of the AGV system.

Automated Guided Vehicle Problems

When the vehicle management problem is considered, the vehicle routing and scheduling problems are the most interested problem. There are many interesting papers about AGV problems, which relate to the vehicle routing and scheduling problem that are reviewed as follows.

Maxwell (1981) presented about solving material handling problems using Operations Research (OR). The objective of this paper is to provide a broad overview of OR techniques that can be used to solve interplant material handling problems. The author first identifies the primary variables generally associated with material handling problems. These variables such as flow rates, weights, sizes, distances, and velocities can usually be handled in OR with matrices and joint probability distributions. OR techniques are used to solve a vehicle requirement problem in the simple AGV. It is assumed that the factory layout and the AGV truck layout are already designed. To simplify the analysis, the AGV are considered unidirectional and only one way flow is

allowed in each plant aisle. The minimum number of vehicles is determined heuristically by first summing the total travel time required, total pick up and delivery time required, and total blocking time encountered by vehicles, and thendividing this value by the total operating time per a unit of vehicle. Blocking time is the time that two or more AGVs are in conflict for a route, causing one or more of them to be blocked. In order to determine the minimum vehicles requirement, blocking time is assumed to be zero. The total pick up and delivery time is determined by time estimation techniques, such as analyzing past productions data, or performing time studies on similar operations. The total operation time per a unit of vehicle is simply a constant base on an estimation of the available operating time of each vehicle over a typical shift. The total traveling time is estimated by using the shortest path to determine optimal route for all possible pairs of nodes, and the problem can finally be treated as a transshipment problem. The author admits that many OR techniques are in their infancy, and these techniques are underutilized in material handling problems. The paper does show the applicability of the shortest route algorithm and transshipment problem to solving an AGV design problem.

Maxwell and Muckstudt (1982) presented the problem of determining an optimum schedule for dispatching AGV that the results a minimum number of AGVs by focusing on the empty traveling distance. The system is designed to ensure proper vehicle's utilization. The items such as raw materials or work pieces are moved from receiving stations to storage facilities and the production lines according to needs as they arise. These functions cannot be carried out effectively unless considerable thought and design effort has gone into planning of the vehicle dispatching and control system. One purpose of their article is to show how the design of an AGV can determine the minimum number of required vehicles. Determination of the optimal number of vehicles is exactly difficult when considering detailed time-phased pick up and drop off requirements, pick up and drop off areas, floor space capacities, and track congestions. A large scale IP can be formulated including all these factors. The second goal is to present other analysis tools that can be used to evaluate the time-dependent behavior of an AGV. The procedure for dispatching vehicles is developed and shown how to measure the blocking time caused by congestions and the size of

storage areas. The problem is designed by assuming the particular layout for a system already exists. The track layout can be defined as a network. The vehicles move through the network among nodes on directed segments which correspond to a guide path connecting one node with the others. The nodes correspond to the intersection points of the various segments of guide path. Each segment has some number of pick up or drop off stations. The problem is analyzed by assuming the system characteristics as follows.

1. vehicles move in only one direction on any segment.
2. a zone control system is used to prevent collisions particularly at intersections.
3. load and unload times are known for each location.
4. traveling speeds among stations for loaded and unloaded vehicles are known. The requirements, which are moved from one station to the other are known. The data are given in integer vehicle loads and that vehicles are always dispatched to pick up and drop off completed loads.

If a unit load must be moved from station i to station j, then one AGV is used to accomplish this task. Splitting of loads is not allowed. Let v_{ij} be the number of vehicle loads that must be moved from station i to station j during a shift and the layout consists of n stations. The value of $\sum_{j=1}^{n} v_{ij}$ is the number of AGVs that are needed at station i to move materials, and $\sum_{i=1}^{n} v_{ij}$ is the number of AGVs that arrives at station j during the shift. For stations, which are not the storage points for AGVs, they must have the total vehicle flow into the station within the shift equal to the total flow out. The model's objective function is to measure the total travel time for empty vehicles moving among stations.

To formulate the problem, the net AGV flows into each station are determined. The net flows for station i is $\sum_{j=1}^{n} v_{ji} - \sum_{j=1}^{n} v_{ij}$. If f_i is the number of AGVs that are

available at the beginning of the shift and g_i is the number of AGVs that are required at the end of the shift, the net flow for station i is

$$NF(i) = \sum_{j=1}^{n} v_{ji} - \sum_{j=1}^{n} v_{ij} + f_i - g_i$$

For the problem to be well defined, $\sum_{i=1}^{n} NF(i) = 0$. Thus, the problem is to determine how to allocate the vehicles that are available [$NF(i) > 0$] at station i to satisfy the deficits at other stations j [$NF(j) < 0$] so that the totaling traveling time for moving empty AGVs are as small as possible.

Let $a_i = NF(i)$, if $NF(i) \geq 0$
= 0, otherwise.
$b_i = NF(i)$, if $NF(i) < 0$
= 0, otherwise.
t_{ij} = the shortest travel time from station i to station j when a vehicle is unloaded.
x_{ij} = the number of empty vehicle trips that should be made from station i to station j during the shift.

The problem is to find the values for the variables x_{ij} for all i,j that

Minimize $\sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij}x_{ij}$

subject to

$\sum_{j=1}^{n} x_{ij} = a_i$, for all station i,

$-\sum_{k=1}^{n} x_{ki} = b_i$, for all station i,

$x_{ij} \geq 0$ for all i, j.

It is easy to see that the above problem is a simple transportation problem. The solution indicates how many vehicle trips should be made with empty vehicles between station i and j. Because this problem is a transportation problem, all variables

will have integer values. The total traveling time for empty AGVs moving between stations is measured. Thus, if this total is H hours and h hours are available on each AGV shift per, and then H/h vehicles are required for material movement plan.

An optimal flow path design is one of the interesting topic for an AGV system planning. The AGV technologies are constantly growing due to better sensors, improved robotics, low-cost high-performance computers, and sophisticated control methods and software. In such a modern manufacturing environment the path routing is one of key factors in a successful implementation of the AGV system.

Blair, Charnsethikul and Vasques (1987) presented the optimum routing problem of AGVs among the workstations as the TSP. An algorithm for the near optimal routing of AGVs in such a system is presented which seeks to organize material moves into tours with the objective of minimizing the maximum tour length. In their paper, they assume that the sequence of move transactions, which are assigned to each AGV, is a tour. The tour distance is the total distance to be traversed in order for the assigned AGVs to go from its initial location to the location of the first move and then to pick up and deliver each move in the prescribed sequence. The tour may requires the AGV to travel empty from the destination of one move to the origin of the next. Each move consists of a unit load, which will consume the capacity of AGVs. The objective function is to minimize the maximum tour distance of all tours. The AGV activity scheduling task can be easily formulated as either two well known network optimization problems. In the first formulation, work centers are represented as nodes. Each move transaction is represented as a directed arc from the origin of the move to its destination. The other formulation represents the move transactions as nodes. Arcs are used to represent the sequence of performance. Each node is connected to every other node by a set of corresponding arcs. This is a modification of the classic TSP appropriately called the multiple traveling salesman problem or MTSP. The heuristic method, which they presented in this work, is composed of two phases. In the first phase, the AGV routing problem is formulated as a standard MTSP. The MTSP is solved by using a modification of the branch and bound technique, first proposed by Eastman (1958). The resulting solution is a minimum total distance over

all tours. The second phase is a tour improvement process, which starts with the feasible set of tours prescribed by phase one. At each iteration of phase two, the longest tour is reduced by removing a node from it. Two new subproblems are defined. One subproblem is the restructuring of the largest tour by treating it as a single TSP. The second subproblem is a reduced MTSP, which includes all the nodes in the other tours plus the node recently removed from the largest tour. Solving the TSP for the largest tour, minus the removed node, provides an optimal patching of the remaining set of nodes in the tour. Solving the remaining tours using an MTSP provides an optimal allocation of the removed nodes to one of the remaining tours. This algorithm has been coded into a BASIC program. The program was tested at three levels:

1. Level I: small-sized problems, 15 moves and 3 tours;
2. Level II: medium-sized problems, 35 moves and 4 tours;
3. Level III: large-sized problems, 50 moves and 5 tours.

For each level, 100 randomly generated move transaction lists of the appropriate size were generated and solved by the program. Each replication is evaluated with respect to two performance measures: the optimization performance ratio (OPR) and the corresponding computation (CPU) time.

Gaskins and Tanchoco (1987) first formulated the flow path design for AGV systems by using IP approach. The objective of this study is to find the optimal flow path for an AGV so that the total traveling distance of the loaded vehicles will be minimized. The 0-1 IP model with considerations of the given facility layout and P/D stations is used to determine the optimal flow path in this paper. However, the paper only considers the unidirectional path network, which has lower utilization than the bidirectional network. The traveling distance by the unloaded vehicles is not taken into consideration. The main limitation of this study is that it only considers a fleet of AGVs with the same origin and destination every time. These AGVs run along the same route. Therefore, routing control is trivialized because issue such as congestion, deadlocks, and conflicts will never occur.

The first procedure of their work is to formulate the objective function. It consists of the distance between pairs of nodes when a particular path is taken, the flow intensity between pairs of nodes, and the decision variables for determining which pairs of nodes are selected. The objective function represents the total distance traveled by loaded vehicle. Besides the objective function, a set of constraint equations are also required to ensure that the shortest route among all pairs of shortest path are taken and all other limitations are satisfied. These constraints include unidirectional flow, at least one input and one output arc that are selected for each nodes, and finally the constraint equations use to ensure the shortest path is taken. The problem is solved by first determining the shortest route between pairs of nodes, and then putting them into the objective function. From this procedure, the minimum total traveling distance is obtained. Unused arcs in the layout can be either removed from the layout or included as alternative routs when blocking occurs.

Kaspi, Kesselman, and Tanchoco (2002) presented the optimal flow path layout design method. The problem is analyzed and formulated by a mixed integer programming (MIP) problem. A searching procedure, based on the branch and bound technique, is proposed to solve the problem. The procedure is implemented as a computer program and yields an optimal solution in a small number of iterations. Using the transportation model for calculating the required and optimal flow of empty vehicles, system balance is achieved. The problem is formulated as a node-arc network where the nodes represent pick up and delivery stations and arcs are guide paths connecting the nodes. Empty vehicle flows are also taken into account when the checking the feasibility of a partially or fully directed guide path is done. The objective of the flow path layout problem is to set directions for each arc in an undirected flow path network such that the total traveling distance of both loaded and unloaded vehicles is minimized. The assumption that the network is fully unidirectional and the reach ability constraints eliminate the issue of blocking.

The authors stated that the formulated linear mixed variables (0-1 and continuous) model is quite difficult to solve. The main difficulty in finding the optimal solution to this problem is the large number of binary variables required for a

realistic size problem. A general approach, which is used to solve this problem, is the branch and bound procedure. When using the branch and bound method, the search function deals with subproblems of the main problem at each step and ignore the global aspect. It is possible that great computational effort can be directed to a branch in which the optimal, or even a feasible, solution cannot be found. The specific used technique is the branch and bound method with depth-first search and backtracking, rather than the jump tracking approach (known also as best-first search). Using the backtracking method, a feasible solution is obtained quite quickly and the required memory is much less than for the jump tracking method. The backtracking procedure is invoked any time when a feasible complete solution is obtained when a branch is bounded or branching is impossible. The backtracking procedure returns to the source branch. The procedure determines the optimal flow of the unloaded vehicles by solving the transportation problem for each step in branching process. The direction of each arc in the system is determined and optimal objective function is obtained.

Traveling Salesman Problem

The real world task of a salesman is trying to sales the products that a salesman has to travel to possible customers at any cities. If a salesman, starts from the depot or head office city, visits each city exactly once on a given list of possible customers and return to the starting point, it is plausible for him to select the order in which he visits all cities so that the distances traveled in his tour is as small as possible. Assume a salesman knows, for every pairs of cities, the distance from one city to the others. Then he has all the data necessary to find the minimum tour distance, but it is by means obvious how to use these data in order to get the answer. This kind of problem is called “Traveling Salesman Problem” or “TSP”.

Lawler *et al.* (1985) presented the survey of knowledge on the TSP. The TSP is one of combinatorial optimization problems that attempt to minimize the total distance of the tour. The problem is one of optimization problems, but cannot immediately employ the methods of differential calculus by setting derivative to zero, because it is in a combinatorial situation that its choice of solution is not over a

continuum but over the set of a tour. A different optimization method comes from linear programming (LP). The continuous history began in the late 1940s with George Dantzig, treats the problem of finding the minimum of linear function on a polyhedron by a system of linear equations. The LP can be used as a tool of combinatorial optimizations by its principle. There are three aspects of the history of any mathematical problem, which are:

1. how to arose
2. how research on it influences other developments of mathematics
3. how the problem is finally solved.

If the TSP is one of the mathematical problems which developed algorithms which satisfy formal or informal standard of efficiency, this problem can be considered that it has not yet been solved. So the TSP is the most prominent of the unsolved combinatorial optimization problem. And that is why it continues to influence the development of optimization concepts and algorithms.

One of the earlier problems of the combinatorial mathematics arises in the theory of graphs. A graph is a finite set of vertices and some pairs of which are joined by edges. A cycle in the graph is a set of vertices of the graph which such that it is possible to move from one vertex to another vertex, along edges of the graph, so that all vertices are encountered exactly once, and it must finish where it started. If a cycle contains all the vertices of the graphs, it is called “Hamiltonian cycle”. The TSP for a graph with specified edge lengths is the problem of finding a Hamiltonian cycle with the shortest length. Lawler *et al.* (1985) presented the survey that many papers relate to the Hamiltonian cycle and the TSP as following examples.

Kirkman (1856) considered Hamiltonian cycles in a general context. He asserted a sufficient condition for a polyhedral graph to admit such a cycle, and also showed that a polyhedron on an odd number of vertices, in which each face has an even number of edges, cannot have such a cycle.

Hamilton (1856) invented a system of noncommutative algebra, for which the actions of the basis elements could be interpreted in terms of paths on the regular dodecahedron. Hamilton named this algebra as “The Icosian Calculus”, and used the graphical interpretation as the basis for a puzzle, marketed the game in name “The Icosian Game”. The game consisted of various problems, such as finishing a cycle when the first five positions are given.

Lin and Kernighan (1973) proposed an effective heuristic algorithm for solving the TSP. The general concept is to transfer arcs which are not included in the previous tour into a new tour by exchanging nodes. They presented several algorithms to show methods which can be used to generate a set of tours from an available tour. A method, which is widely used, is called the 3-OPT procedure. The process of 3-OPT is to choose three arcs out of the old tour and find three new arcs to replace them. Several new tours are generated. An objective function, which is minimizing tour length, must be evaluated and the process stops when all new tours show no improvement in the objective value. Otherwise, a tour with improvement is chosen to start the process again.

The TSP/MTSP can be formulated as IP. Orman and Williams (2004) presented a survey of different IP formulations of the TSP such as the conventional formulation that is presented by Dantzig, Fulkerson and Johnson (1954) and Miller, Tucker and Zemilin (1960). The 0-1 IP model of TSP is defined on a complete directed graph $G = (V, A)$, on n vertices, with vertex a set $V = \{1, 2, \ldots, n\}$, arc a set $A = \{(i, j) | \, i, j = 1, 2, \ldots, n \}$, nonnegative cost or distance c_{ij} associated with arcs (i, j) and $c_{ii} = \infty$ for all $i, j \in V$.

Dantzig, Fulkerson and Johnson (1954) formulated the standard problem of TSP as a 0-1 IP model as follows

$$Min \;\; Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \qquad (1)$$

Subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j = 1,\ 2, ..., n \qquad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1,\ 2, ..., n \qquad (3)$$

$$\sum_{i,j \in S} x_{ij} \le |S| - 1 \qquad \forall S \subseteq V \ and \ 2 \le |S| \le n-1, \qquad (4)$$

$$x_{ij} = 0 \, or \, 1 \qquad \forall i, j \in V$$

where $V = \{1, 2, \ldots, n\}$, $x_{ij} = 1$ if arc (i, j) is in the solution and $x_{ij} = 0$, otherwise. The constraints (2) and (3) are the assignment constraints. The constraints (4) represent the subtour elimination constraints. This formulation has $2^{n-1} + n$-1 constraints and n (n-1) of 0-1 variables x_{ij}. The exponential number of constraints makes it impractical to solve directly. The branch and bound approach can be applied and solved this model iteratively.

The sequential formulation is the Miller, Tucker and Zemlin (1960) formulation of the classical TSP that is given as follows

$$Min \;\; Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \qquad (6)$$

Subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j = 1,\ 2, ..., n \qquad (7)$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1,\ 2, ..., n \qquad (8)$$

$$y_i - y_j + nx_{ij} \le n - 1 \qquad \forall i \ne j \qquad (9)$$

$$x_{ij} = 0 \, or \, 1 \qquad \forall i, j.$$

The number of cities is n, the distances are c_{ij} and the arcs in the tour are represented by the variable x_{ij} for all i, j . The c_{ij} is the distance from city i to j ($c_{ij} = \alpha$ for $i = j$). The variable x_{ij} is 1 if the salesman travels from city i to j and 0 otherwise. The

variables y_i are arbitrary real numbers which satisfy the subtour elimination constrain (9). The constraints (7) and (8) are the assignment constraints. This formulation has $n^2 - n + 2$ constraints and n (n-1) of 0-1 variables x_{ij}. The mathematical formulation of the MTSP can be formed by applying the transformation idea to the Miller Tucker and Zemlin(1960) formulation. Svestka and Huckfeldt (1973) gave the MTSP formulation for m salesmen as following.

$$Min \;\; Z = \sum_{i=1}^{r}\sum_{j=1}^{r} d_{ij}x_{ij} \qquad (10)$$

Subject to

$$\sum_{i=1}^{r} x_{ij} = 1, \qquad j = 1, 2, ..., r \qquad (12)$$

$$\sum_{j=1}^{r} x_{ij} = 1, \qquad i = 1, 2, ..., r \qquad (13)$$

$$y_i - y_j + (n+m-1)x_{ij} \le n+m-2 \qquad \forall i \ne j \qquad (14)$$

$$x_{ij} = 0 \, or \, 1 \qquad \forall i, j.$$

where $r = n+m-1$

d_{ij} denotes the new distances for MTSP and all other terms have the same definitions as the Miller Tucker and Zemlin(1960) formulation. The new distance matrix [d_{ij}] are defined from the original distance c_{ij}, which augment the original distance matrix [c_{ij}] with m-1 new rows and columns, where each new row and column is a duplicate of the first row and column of the matrix [c_{ij}]. It is assumed that the first row and column correspond to the home city. Set all other new elements on new rows and columns of the augment matrix to infinity. All other terms have the same value as the original matrix [c_{ij}].

Bellmore and Hong (1974) proposed another method to solve MTSP. Suppose all n cities must be visited by one of m salesmen. They presented the transformation of the MTSP for m salesmen to the classical TSP by adding m-1 dummy nodes to the original network as the artificial starting node and solved the MTSP from solving the TSP of the modified network.

According to this point, the TSP is seductively easy to state. It takes no mathematical background to understand the problem and no great talent to find good solutions to large problems. Thus, it is exciting to work on the way to solve the problem on any sizes. The TSP has resisted all efforts to find a good optimization algorithm or even an approximation algorithm that is guaranteed to be effective. There are also practical reasons for the importance of the TSP. Many significant real world problems can be formulated as instances of the TSP. The application of TSP can describe various problem transformations, related combinatorial problems, and generalizations of the basic TSP.

Generalizations of the TSP and related problems

There are many problems related or have some relationships with the TSP. Lawler *et al.* (1985) illustrated the relationships of the TSP to several other optimization problems, which are shown as follows.

1. The assignment problem: the problem of n cities is considered. Let x_{ij} be a 0-1 variable indicating whether or not the salesman goes directly from city i to city j for all i, j and c_{ij} be the corresponding distance. The length of salesman tour is then

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \qquad (15)$$

which is to be minimized. Clearly,

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1,...,n, \qquad (16)$$

since a unique city is visited directly after each city, and similarly,

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j = 1,...,n, \qquad (17)$$

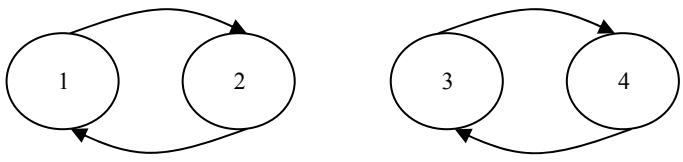


Figure 2 Subtours from the assignment solution

Now, (15), (16) and (17) describe the well-solved assignment problem. It follows that the TSP must involve some additional complications. In particular, the missing constraints in the above formulation involve subtours. For example, if $n = 4$ then $x_{12} = x_{21} = x_{34} = x_{43} = 1$ and $x_{ij} = 0$ otherwise satisfies (16), (17) but represents two subtours (1, 2), (3, 4) of figure 2 rather than a single tour. Thus, the assignment problem is a relaxation of the TSP or, equivalently, the TSP is the restriction of the assignment problem obtained by adding the constraint of a single tour, which is:

'no subtours allowed'. (18)

2. Integer linear programming: There are a number of ways to enforce (18) mathematically. For instance, (18) can be replace with

$$\sum_{i \in s} \sum_{j \in s} x_{ij} \leq |S| - 1 \qquad (18a)$$

or with

$$\sum_{i \in s} \sum_{j \in s} x_{ij} \geq 1 \qquad (18a)$$

for every proper, nonempty subtour S of $N = \{1, \ldots, n\}$ where |.| denotes cardinality. Clearly, any subtour violates (18a) and (18b) for some S (In figure 2, $S = \{1, 2\}$ and $\{3, 4\}$). Of course (18a) and (18b) represent a large number of constraints: $2^n - 2$ to be exact. However, these formulations, due to Dantzig, Fulkerson and Johnson (1954), do have at least one characteristic of good formulations, namely a well-solved relaxation.

A more compact variation of (18a) and (18b) is proposed by Miller, Tucker and Zemline (1960). Arbitrarily designate vertex 1 to be the home base. Then the constraints

$$y_i - y_j + n x_{ij} \leq n-1, \qquad i, j = 2, \ldots, n, \qquad (18c)$$

where y_i and y_j are arbitrary real numbers, block all tours not containing vertex 1. To see that (18c) in conjunction with (16), (17) blocks subtours, consider an arbitrary

subtour (i_1 , …., i_k) where 1 is not in $\{i_1$, …., $i_k\}$. If a set of x_{ij} satisfying (16), (17) represents more than one subtour, then it also represents at least one subtour not containing vertex 1. But addition of the constraints (18c), represented by this subtour yields $nk \leq (n\text{-}1)k$ which is clearly false since $n, k \geq 2$. Furthermore, every TSP tour remains feasible with these additional constraints. Every tour can be assumed to start at city 1. If city i is visited j^{th} after city 1, let $y_i = j$. As example, consider the tour (1, 4, 3, 2). For this, set $y_1 = 0$, $y_4 = 1$, $y_3 = 2$, and $y_2 = 3$. It is straightforward to verify that this procedure works in general.

Note that the model (15), (16), (17), (18c) with binary variables x_{ij} is a mixed integer program since it has n-1 continuous variables, and that (18c) represents only $(n\text{-}1)^2$ constraints. It is also shown by Miller, Tucker and Zemline (1960) that an extension of the TSP can be modeled in the same way. Suppose the salesman visits the cities in a number of subtours, each beginning and ending at city 1, and no subtour can contain more than r cities, which $r < n$. Then (18c) can be replace with

$$y_i - y_j + rx_{ij} \leq r\text{-}1, \qquad i, j = 2, \ldots, n, \qquad (18d)$$

Of course, since city 1 can be visited more than once, the constraints

$\sum_{j=1}^{n} x_{1j} = 1$ and $\sum_{i=1}^{n} x_{i1} = 1$ should be replaced by $\sum_{j=1}^{n} x_{1j} \geq 1$ and $\sum_{i=1}^{n} x_{i1} \geq 1$.

The model obtains the solution, which is a set of subtours of r cities.

Branch and bound methods for TSP

Lawler *et al.* (1985) stated that the origins of the branch and bound idea go back to the work of Dantzig, Fulkerson & Johnson (1954 and 1959) on the TSP. The first attempt to solve TSP by enumerative approach is apparently due to the work of Eastman (1958). In a sense the TSP has served as a testing ground for the development of solution methods for discrete optimization, in that many procedures and devices were first developed for the TSP and then, after successful testing, extended to more general integer programming.

Gillett (1976) stated that the enumerative (branch and bound, implicit enumeration) methods solve a discrete optimization problem by breaking up its feasible set into successively smaller subset, calculating bounds on the objective function value over each subset, and using them to discard certain subsets from further consideration. The bounds are obtained by replacing the problem over a given subset with an easier (relaxed) problem, such that the solution value of the latter bounds that of the former. The procedure ends when each subset has either produced a feasible solution, or has been shown to contain no better solution than the one already in hand. The best solution found during the procedure is a global optimum. A number of branch and bound algorithms that find the exact solution for a small to moderate size of TSP (fewer than 50 cities) have appeared in many literatures, but most of them are base on the algorithm by Eastman (1958).

Little *et al.* (1963) presented an algorithm that is a branch and bound method for solving TSP. The set of all tours (feasible solutions) is broken up into smaller subsets by a procedure called branching. For each subset, a lower bound on the length of tours is calculated. Eventually, a subset is found that it contains a single tour whose length is less than or equal to some lower bound for every tour. This algorithm modifies both the branching and bounding procedures by modifying the Eastman's algorithm to eliminate two cities subtours. Since the Eastman and Little's algorithm form the basis for all TSP branch and bound algorithms, one of them, namely, Eastman's algorithm is presented as follows.

Eastman's algorithm for TSP

Gillett (1976) presented an Eastman's algorithm for TSP which is the branch and bound algorithm for solving the single TSP tour. Let $c(i,j)$ be the distance from city i to city i for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$. Where n is the number of cities and $c(i,i) = \infty$ for $i = 1, 2, \ldots, n$. A tour is a complete route or cycle through n cities where no city is visited more than once. If the salesman visits a certain city and returns to that city later, the cities involved form a subtour. Of course, this cannot occur if a route is feasible (each city is visited once and only once). The Eastman's algorithm

solves the easier assignment problem that allows subtours and then systematically forbids subtours until finally the single tour is obtained that are the optimal. An illustrative procedure follows the algorithm, which is:

Step 1:
Let CLUB represents the current least upper bound on the optimal solution of the TSP. Set CLUB = 10^{10} (CLUB equal to a large positive number)

Step 2:
Solve the associated assignment problem, where the distances $c(i,j)$ are the elements of the distance matrix. The solution provides a lower bound on the optimal solution of the TSP. If at least one subtour exists in the solution, go to step 3, otherwise the optimal solution of the assignment problem is also an optimal solution of the TSP, so stop.

Step 3:
Select a subtour and let k be the number of arcs in the selected subtour. Eastman selects the subtour with the smallest number of arcs. All other subtours at this node can be ignored.

Step 4:
Branch into k subproblems. If the subtour is:
$i_1 - i_2 - \ldots i_k - i_l$
Then for subproblem 1 let $c(i_1, i_2) = \infty$, for subproblem 2 let $c(i_2, i_3) = \infty$, etc., and for subproblem k let $c(i_k, i_l) = \infty$.

Step 5:
Solve the k new assignment problems. Each solution distance is a lower bound for the corresponding subproblem.

Step 6:
If there are one or more feasible solutions from step 5 and if the smallest total distance for these feasible solutions, say STD, is smaller than CLUB, set CLUB = STD and save the corresponding feasible solution. Otherwise CLUB remains unchanged.

Step 7:
If CLUB is less than the lower bounds on all other unexplored subproblems, then the solution corresponding to CLUB is an optimal solution of the TSP, so stop; otherwise, goes to step 8. By unexplored subproblems, it means subproblems that have not been divided into further subproblems.

Step 8:
From the set of all unexplored nonfeasible (subtours present) subproblems with a bound less than CLUB, select the subproblem with smallest lower bound for further branching. Go back to step 3

Applications of the TSP

Despite the fact that the TSP can be applied to many useful situations directly, most of reported applications are quite different. Seemingly there are many unrelated problems that can be solved by formulating them as instances of the TSP. Lawler *et al.* (1985) illustrated some examples of applications of the TSP. Many applications descried below are the versatility of the TSP model.

1. Vehicle routing: by vehicle routing it means the problem of determining for a fleet of vehicle which customers should be served by which vehicles, and in what order each vehicle should visit its customers. Constraints generally include capacities of the vehicles as well as time windows for each the customers. Some algorithms for this problem use the TSP model for the subproblem of ordering each vehicle's customers.

2. Computer wiring: this problem occurs repeatedly in the design of the computer's component and other digital systems. A system consists of a number of modules and several pins are located on each module. A given set of pins has to be interconnected by wires. In order to avoid signal crossing and to improve ease and neatness of wiring, the total wire length should be minimized. A minimum length Hamiltonian path can be solved by using an (n+1)-city symmetric TSP.

3. Cutting wallpaper: this situation needs to cut n sheets of wallpaper from a single long roll of paper by minimizing waste. Sheet i starts at position S_i and finishes at position F_i, with respect to a pattern that repeats at one unit intervals. Thus, $F_i = S_i + L_i$ (mod 1) where the length of sheet i is L_i pattern units. The amount of wallpaper that is wasted if sheet j is cut from the roll immediately after sheet i is then;

$C_{ij} = S_i - F_i$ if $F_i \leq S_i$, otherwise $C_{ij} = 1 + S_i - F_i$

or equivalently,

$C_{ij} = S_i - F_i$ (mod 1).

Now suppose that when begin cutting, the end of the roll is at position F_o and that after cutting the last sheet worker must makes one final cut to restore the roll to the same starting points $S_0 = F_0$. If create a dummy sheet *0* is created, the starting and finishing point, the problem of cutting the n sheets from the roll become an (n+1)-city TSP with distance matrix defined by C_{ij}.

4. Job sequencing: Consider the problem of sequencing n jobs on a single machine. The jobs can be done in any order and the objective is to complete all of them in the shortest possible time. Assume that the machine must be in a certain state S_i in order to do job j and that the beginning and ending state for the machine is S_o. Let the time required to complete job j directly after job i be

$$T_{ij} = C_{ij} + P_i$$

where C_{ij} is the time required to transform the machine from S_i to S_j and P_i is the actual time to perform job j (with $P_0 = 0$). The TSP can be used to solve this kind of problem by using the distance matrix defined by T_{ij}.

5. The stacker crane problem: The motivation for this problem is expressed by ignoring the stacker cranes and considering the delivery trucks. Suppose a truck must perform a collection of pick up and delivery, subject to the constraint that each loads, which is picked up completely, fills the truck and goes to a single destination, Hence, no picks up or deliveries can be combined. The stacker crane problem is a generalization of the TSP in which the desired tour must contain certain edges, and must traverse them in specified directions. An instance is a set of cities (and corresponding distance matrix C is defined by $[C_{ij}]$) together with a set A of arcs, where each arc is an ordered pairs of cities and every city occurs in exactly one arc. If $(i, j) \in A$, this means that a load must be picked up at city i and delivered to city j. The goal is to save fuel, by minimizing the total length of the route that is used to make all movements. The TSP with distance matrix $[C_{ij}]$ can be applied to this problem.

6. Problem of postal service: The stacker crane problem is related to a number of other problems which are concerned more with traversing arcs (or edges) then with visiting vertices. The undirected analogue of the stacker crane problem is called the rural postman problem, Orloff (1974). The information is given a set of required edges (rather than arcs) and asks for a route of the minimum length, which will traverse each edge at least once (the direction of traversal dose not matter). This model is the problem that a mail carrier designs an optimum route, with each edge corresponding to a street along which the mail must be delivered. The TSP can be applied to help a mail carrier to solve an optimum route.

Benders'Decomposition Algorithm

J. F. Benders (1962) proposed a technique in which the mixed integer programming (MIP) problem can be written as an IP problem. Using the linear programming duality theory, it is possible to show that any the MIP problem can be written as an integer program. The equivalent IP problem is solved after generating only a subset of its constraints. The remaining "implicitly enumerated" constraints are relaxed from the IP problem. The Benders decomposition procedure partitions the

MIP problem into an integer and a continuous part, consisting respectively of the integer and the continuous variables of the original problem.

The decomposition algorithm works by successive solving a continuous programming problem and an integer programming problem, considering the linear case. A LP produces an extreme point and a single constraint for the IP problem. Also, the value of the LP optimal solution gives an upper bound for the optimal solution to the MIP problem. After the IP problem, which is the MIP problem's equivalent when it has all cut constraints, is solved, it yields a nondecreasing lower bound. When the two bounds coincide, the optimal MIP solution has been found and the process terminates.

Consider a class of linear MIP problem, which is the Benders'master problem, as follows.

$$\begin{aligned} \text{(MIP)} \qquad & \text{Minimize} \quad c^T x + d^T y \\ & \text{Subject to} \quad \boldsymbol{A}x + \boldsymbol{B}y \geq b, \\ & \qquad\qquad\quad x \geq 0,\ y \in \boldsymbol{Y} \end{aligned}$$

where $\boldsymbol{A}$ is a m by n coefficient matrix of vector x,

$\boldsymbol{B}$ is a m by n' coefficient matrix of vector y,

c is a n by 1 cost vector of vector x,

d is a n' by 1 cost vector of vector y

x is a n by 1 vector of continuous variable x,

y is a n' by 1 vector of variable y with $\boldsymbol{Y} = \{y \mid y_i \in \{0, 1\};\ i = 1, 2, \ldots, n'\}$

A concept of the Benders'algorithm is that the partitioning of the variables into two sets (x and y) and projecting the problem onto the y variables. If let $\boldsymbol{Y}$ denote the set of binary or all possible nonnegative integer vectors y, then MIP may be written as follows.

Let $\quad v(y) = d^T y + min\ \{\ c^T x \mid \boldsymbol{A}x \geq b - \boldsymbol{B}y,\ x \geq 0\}$

the Benders'master problem is clearly seen to be equivalent to:

$$\begin{aligned} &\text{Minimize} && v(y) \\ &\text{Subject to} && y \in \boldsymbol{Y} \end{aligned}$$

for a fixed y, the minimization problem is the LP problem

$$\begin{aligned} \text{(LP)} \qquad &\text{Minimize} && c^T x \\ &\text{Subject to} && \boldsymbol{A}x \geq b - \boldsymbol{B}y, \\ & && x \geq 0 \end{aligned}$$

its dual programming (DL) problem is

$$\begin{aligned} \text{(DL)} \qquad &\text{Maximize} && (b - \boldsymbol{B}y)^T u \\ &\text{Subject to} && \boldsymbol{A}^T u \leq c, \\ & && u \geq 0 \end{aligned}$$

where u is a m by 1 vector of dual variable u,

In principle, it is possible to identify and enumerate all of extreme points of the dual feasible region and choose the best. That is, the function $v\,(y)$ can be evaluated by:

$$v\,(y) = d^T y + \text{Maximize } \{(b - \boldsymbol{B}y)^T u \mid \boldsymbol{A}^T u \leq c,\ u \geq 0\}$$

Suppose $\boldsymbol{Y}$ consists of p sets of vector y, a fixed vector y is defined by y^j for all $j = 1, 2, \ldots, p$ and $v_k\,(y^j)$ is a function that a vector y^j is supplied to the function $v\,(y)$ at iteration k for finding the k^{th} solution for all $k = 1, 2, \ldots, p$. However, the $v\,(y)$ is to be evaluated by solving the LP problem, not by identifying all sets of dual extreme points (p sets) and computing the corresponding linear objective function of y. If k^{th} iterations are used with the sets of y^j (where $1 \leq k \leq p$), it can provided an approximation function, which is an underestimate of $v\,(y)$, defined by $\mathrm{v}_k\,(y^j) =$ Maximize $\{(b - \boldsymbol{B}\,y^j)^T u^j \mid \boldsymbol{A}^T u^j \leq c,\ u^j \geq 0\}$, that is the Benders'subproblem.

The initial ($k = 1$) value of variables $y \in \boldsymbol{Y}$ can be generated by selecting any arbitrary value of vector y (it is the first y^j, which is y^1) that provides the feasible solution to the Benders'subproblem, Maximum $v_k\,(y^j) =$ Maximize $\{(b - \boldsymbol{B}\,y^j)^T u^j \mid \boldsymbol{A}^T u^j \leq c,\ u^j \geq 0\}$. The solution of the Benders'subproblem is evaluated by solve LP

with fixed y^j. The Benders'subproblem solution is a generated vector u^j of dual variable u and its objective function value of $v_k(y^j)$ that corresponding with a selected y^j. When a generated vector u^j from solving the Benders'subproblem is put into the Benders'master problem, which is the Benders'partial master problem of iteration k^{th} that is:

$$\text{Minimize} \quad v(y) = d^T y + \text{Maximize} \{ [(b - \boldsymbol{B}y)^T u^j]_k \}$$
$$\text{Subject to} \quad y \in \boldsymbol{Y}$$

This step provides the Benders'partial master problem with one underestimate function of $v(y)$, called the Benders'cut of the iteration k^{th} ($[(b - \boldsymbol{B}y)^T u^j]_k$). For each iteration, the algorithm must solve this Benders'partial master problem to generate the new vector y, for replacing the previous selected vector y, and the new solution value of master problem, $v(y)$**.**

If $v_k(y^j) = v(y)$, the solution can be accepted and terminate the algorithm otherwise the algorithm has to improve by adding the new approximation function, Benders'cut by using new set of dual extreme points to generate the new arbitrary y^j of iteration k+1for solving the new partial master problem. Therefore, the Benders'decomposition algorithm for solving the lower bound of AGVsp-P/D can be summarized step by step as follows.

Step 1: Initialization: set $v(y) = 0$, select a fixed vector $y^j \in \boldsymbol{Y}$, set $j = 1$ and set $k = 1$

Step 2: Solve the Benders'subproblem: evaluate the value of $v_k(y^j)$ with its corresponding set of the dual extreme point (vector u^j) by solving LP with a fixed vector y^j

Step 3: Stopping criterion: if $v_k(y^j) = v(y)$ then stop, otherwise go to step 4

Step 4: Improve the approximation function: by using a set of dual extreme point (vector u^j) to generate the Benders'cut for forming iteration k^{th} Benders'partial master problem

Step 5: Solve the Benders'partial master problem: that is the minimizing of $v(y)$ with Benders'cut from step 4 for updating the value of $v(y)$ and then set $j=j+1$ and updating the new vector y^j, set $k = k+1$ and go to step 1.

Statistical methods for data analysis

Montgomery and Runger (2002) illustrated and applied statistics for using in the engineering research. The data probability distribution is the first issue that should be considered because most of statistic assumptions are assuming the normal probability distribution of the data set. The normality test is explained in this part. Then the analysis of variance that is the important method to conclude the solving result is reviewed.

1. Probability plots

Montgomery and Runger (2002) explained that the probability plot is a graphical method for determining whether sample data conform to a hypothesized distribution based on a subjective visual examination of the data set. The general procedure is very simple and can be performed quickly. Probability plotting typically uses special graph paper, known as the probability paper that has been designed for the hypothesized distribution. Probability plotting is wildly used for the normal distribution because most of statistical methods are using normal probability distribution data.

A normal probability plot can also be constructed on an ordinary graph paper by plotting the standardized normal scores z_j against $x_{(j)}$, where the standardized normal scores satisfy $\frac{j-0.5}{n} = P(Z \le z_j) = \phi(z_j)$. Almost of statistical software can perform the normality test by doing the normal probability and showing the result in same form as plotting on an ordinary graph paper. If the specific type I error is α, the probability distribution of the data set is the normal probability distribution when the P-value of the normal probability plot of the data set is grater than α, otherwise the data set is not the normal probability distribution. The example of the normal probability plot that is obtained from normal distribution data by using MATLAB is shown as follows.

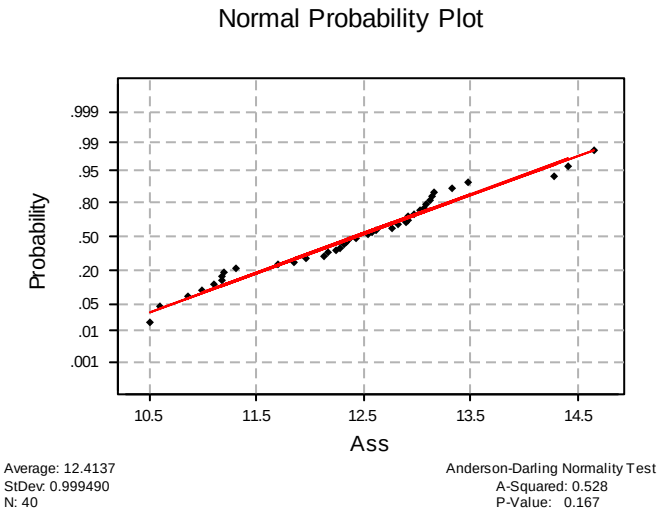


Figure 3 The example of the normal probability plot of the assignment data with 50 nodes from table 31 by using MINITAB

When the obtained data is non-normality, MINITAB has Two Box-Cox transformation procedures, which are a stand-alone command and a transformation option that can be useful for correcting both non-normality and highly skewed. First, use the stand-alone command as an exploratory tool to determine the best lambda value for the transformation. Then, use the transformation option to transform the data. The Box-Cox transformation is used to make the data "more normal." The transformation takes the original data to the power l, unless l = 0, in which case the natural log is taken. (l is pronounced "lambda.") To use this option, the data must be positive. The options subdialog box lists the common transformations natural log (l=0) and square root (l= 0.5). User can also choose any value between -5 and 5 for l. In most cases, user should not choose an l outside the range of -2 and 2.

2. Hypothesis testing

Montgomery and Runger (2002) illustrated that many research problems require the conclusion that the results will be accepted or rejected, based on some parameters. Normally, the researchers decide whether accept or reject a statement about the research results of some parameters. The statement is called a hypothesis,

and the decision making procedure about the hypothesis is called hypothesis testing. This is one of the most useful aspects of statistical inferences, since many types of decision making problems, tests, or experiments in the research can be formulated as hypothesis testing problems. Normally, research considers the hypothesis test about the mean μ of a single normal population distribution where the variance of the population σ^2 is known. The hypothesis can be formally stated as

$$H_0: \mu = \mu_0$$

$$H_1: \mu \neq \mu_0$$

It is usually more convenient to standardize the sample mean and use a test statistical based on the standard normal distribution. That is, the test procedure for H_0 uses the test statistic

$$Z_0 = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}}$$

If the specific type I error is α, the hypothesis H_0: $\mu = \mu_0$ cannot be rejected when the observed value of the test statistic Z_0 is $-Z_{\alpha/2} \leq Z_0 \leq Z_{\alpha/2}$. When a research considers hypothesis testing about the mean μ of population with unknown variance of the population σ^2, the test procedure for H_0 uses the test statistic

$$T_0 = \frac{\bar{X} - \mu_0}{S/\sqrt{n}}$$

T_0 has a *t* distribution with n-1 degrees of freedom. If the specific type I error is α, the H_0: $\mu = \mu_0$ cannot be rejected when the observed value of the test statistic is $-t_{\alpha/2, n-1} \leq t_0 \leq t_{\alpha/2, n-1}$.

3. The analysis of variance (ANOVA)

Montgomery and Runger (2002) presented that many single-factor experiments require that more than two levels of the factor be considered. For example, an industrial engineer may want to investigate three different methods. The ANOVA, can be used for comparing means when there are more than two levels of a single factor. Suppose all experiments have different levels of a single factor that the

researchers wish to compare. Each factor level is called a treatment, a very general term that can be traced to the early applications of the experimental design methodology. The response from the experiment for each of the k treatments is a random variable. The researchers are interested in testing the equality of the k treatment means $\mu_1, \mu_2, \ldots, \mu_k$. The hypothesis can be formally stated as:

H_0: $\mu_1 = \mu_2 = \ldots = \mu_k$

H_1: $\mu_i \neq \mu_j$ for at least one pair of all i, j

Thus, if each observation consists of the overall mean μ plus a realization of the random error component, this is equivalent to saying that all N observations are taken from the normal probability distribution with mean μ and variance σ^2. Therefore, if the hypothesis H_0 is not rejected, the changing of the level of the factor has not affect on the mean μ of response. Table 2 is called the ANOVA table of k treatments, n observations and $N = kn$ total number of observations.

Table 2 The ANOVA table for a single-factor experiment, fixed effects model

Source of Variation	Sum of Squares	Degree of Freedom	Mean Squares	f_0
Treatment	$SS_{Treatment}$	$k-1$	$MS_{Treatment}$	$f_0 = MS_{Treatment} / MS_E$
Error	SS_E	$N-k$	MS_E	
Total	SS_T	$N-1$		

The value of test statistic is

$$SS_T = \sum_{i=1}^{k}\sum_{j=1}^{ni} y_{ij}^2 - \frac{y_{..}^2}{N}$$

$$SS_{Treatment} = \sum_{i=1}^{k} \frac{y_{i.}^2}{n_i} - \frac{y_{..}^2}{N}$$

$$SS_E = SS_T - SS_{Treatment}$$

$$MS_{Treatment} = \frac{SS_{Treatment}}{k-1}$$

$$MS_E = \frac{SS_E}{N-k}$$

$$f_0 = \frac{MS_{Treatment}}{MS_E}$$

where y_{ij} is a data point from the experiment by using treatment i and replication j^{th}, $i = 1, 2, \ldots, k$ and $j = 1, 2, \ldots, n$

$y_{i.}$ is the summation value of all observations from the experiment by using treatment $i, (y_i = \sum_{j=1}^{n} y_{ij})$

$y_{..}$ is the grand total, $(y_{..} = \sum_{i=1}^{k}\sum_{j=1}^{n} y_{ij})$

If the specific type I error is α, the hypothesis H_0 cannot be rejected when $f_0 < f_{\alpha, v1, v2} = f_{\alpha, k-1, N-k}$ from the table of F- probability distribution.

4. Nonparametric statistics

Montgomery and Runger (2002) explained that most of the hypothesis testing and confidence interval procedures, which is discussed in the previous part, are based on the assumption that it works with random samples from normal populations. Traditionally, these procedures are called parametric methods because they are based on a particular parametric family of distributions, the normal in this case. Alternatively, sometimes it can be said that these procedures are not distribution-free because they depend on the assumption of normality. Nonparametric or distribution-free procedures do not utilize all the information, which provides by the sample. As a result, a nonparametric procedure will be less efficient than the corresponding parametric procedure when the underlying population is normal. There are many nonparametric methods involve the analysis of data.

The Kruskal-Wallis test is a nonparametric method in the analysis of variance for a single-factor experiment. The Kruskal-Wallis can perform a test of the equality of medians for two or more populations. This test offers a nonparametric alternative

to the single-factor (one-way) analysis of variance. Suppose that $N = \sum_{i=1}^{a} n_i$ is the total number of observations n_i for all factor levels $i = 1, 2,\ldots, a$, $R_{i.}$ demote the n_i ranks in the i^{th} treatment and $\overline{R_{i.}}$ denote the average value of $R_{i.}$. The Kruskal-Wallis hypotheses are:

H_0: the population medians are all equal versus H_1: the medians are not all equal

The Kruskal-Wallis test statistic measures the degree, which the actual observed average rank $\overline{R_{i.}}$, to the different from their expected value $(N + 1)/2$. If this difference is sufficiently large, the hypothesis H_0 is rejected. The test statistic is

$$H = \frac{12}{N(N+1)} \sum_{i=1}^{a} n_i \left(\overline{R}_{i.} - \frac{N+1}{2} \right)^2$$

H has approximately a chi-square distribution with a-1 degrees of freedom. Since large values of H imply that H_0 is false, H_0 will be rejected if the observed value $H \geq x^2_{\alpha,a-1}$. When observations are tied, assign an average rank to each of the tied observation. The test statistic is

$$H = \frac{1}{S^2} \left[\sum_{i=1}^{a} \frac{R_{i.}^2}{n_i} - \frac{N(N+1)^2}{4} \right]$$

where n_i is the number of observations in the i^{th} treatment, N is the total number of observations, and

$$S^2 = \frac{1}{N-1} \left[\sum_{i=1}^{a} R_i^2 - \frac{N(N-1)^2}{4} \right]$$

An assumption for this test is that samples from the different populations are independent random samples from continuous distributions, with the distributions having the same shape.

MATERIALS AND METHODS

This chapter presents research methods, which include materials for researching the problem formulation, the mathematical model of AGVsp-P/D and solving algorithms to find the solution and verify the model quality.

Materials

The materials for this research could be categorized into three groups as follows:

1. Computer

A personal computer, CPU Pentium IV 2.0GHz with 2 GB RAM, was used to generate the data of simulated problems, process data sets, formulate the mathematical model, program the algorithms, and run programs to solve the tested problems.

2. Software

2.1 Microsoft Excel program was used to solve the formulated integer programming by using Solver, form the information sheets, and create the tables and graphs for this research document.

2.2 MATLAB 7.0 was used to generate the tested problems, program the algorithm and run the program to solve the generated problems.

2.3 Microsoft Word was used to create this research document.

2.4 Minitab was used to perform all statistical analysis.

3. Literatures and related papers

Most of literatures and related papers have been received from many professors, which are Dr. Peerayuth Charnsethikul (the advisor of this thesis), Dr. Kamlesh Mathur, Dr. Danil Solow, and Dr. George Viraktarakis. A lot of books and papers have been collected from the Kelvin Smith library of Case Western Reserve University (CWRU), USA, the main library of Asian Institution of Technology (AIT), Thailand, the library of the faculty of Engineering and the main library of Kasetsart University, Thailand and download from electronic online journals, which available at OHIOLINK on the Internet.

Methods

The motivation of the mathematical model and solving approaches are due to the fact that the routing problems normally are difficult to solve and can not satisfy some real situations, because it relates to some problems in NP-hard class such as TSP/MTSP. If some real world constraints are added to a kind of routing problem, it becomes a much more difficult problem to be modeled and solved.

The original single/multi AGV scheduling problem with specific P/D nodes can be formulated and solved as TSP/MTSP (Blair, Charnsethikul and Vasques, 1987). When the original AGV problem is modified to capture the special network structure that is the network, which has alternatives for some nodes, the problem becomes the AGVsp-P/D. TSP/MTSP with alternative P/D nodes will be considered for finding the solution of this special AGV scheduling problem.

The key successfulness of this thesis will be creating the appropriate mathematical model and heuristic approaches for solving the AGVsp-P/D. The research sequence will be conducted following the steps that consist of the study and analysis of the AGVsp-P/D, compare the AGVsp-P/D with the existing related problems from literatures, formulate the mathematical model of this problem, create the heuristic algorithms for solving these created mathematical model which is the

modified TSP approach, and perform the test and evaluation of the created model by programming the model on MATLAB 7.0 and solving some generated examples of this problem. The proposed research will be explained step by step as follows.

1. The problem of AGV with alternative pick up and delivery nodes (AGVsp-P/D)

Job sequencing and scheduling is the important part of AGV system design. The main goal of this step is to define the problem of AGVsp-P/D clearly in detail and structure for doing analysis and studies in the next step. Designing AGV systems are complex tasks. One of the main purposes of the scheduling problem for single/multi AGV is how the scheduling can provide the minimum total traveling distance of AGV. Normally, the scheduling problems have been considered or designed with the routing problem concomitantly. The ordinary vehicle scheduling and routing problem as single/multi AGV scheduling problem is the problem with a single specific P/D node that can be simulated by a network problem approach such as TSP/MTSP.

According to this point, the potential problem for studying the single/multi AGV scheduling problem is extended to be more realistic that the original TSP/MTSP problem is modified by adding the structure of alternative P/D nodes. The main purpose is to find the scheduling of AGV problems with alternative P/D nodes. This kind of problems is presented in section 2.1.

The original TSP/MTSP is one of the applications of network problems, it is necessary to choose a sequence of nodes to visit so as to accomplish a specified objective. When the AGVsp-P/D is considered, the TSP/MTSP approach can be applied for solving the schedule of problem like normal vehicle routing problems, but the approach has to be modified to support the special structures of AGVsp-P/D. The concept of TSP/MTSP will be applied by using the generated technique of assignment problem with alternative P/D nodes to solve the AGVsp-P/D for determining the minimum traveling distance of each AGV from the starting depot to some appropriate selected nodes and then come back to the starting depot. This procedure based on the

branch and bound with solving assignment subproblems for determining the optimal schedule. The formulated mathematical model is presented in section 2.2.

The assignment problem with alternative P/D nodes which is the lower bound of the AGVsp-P/D is considered as one of important parts of this research. The mathematical models of TSP/MTSP are formulated in form of the 0-1 IP problem. For large 0-1 IP problems, it takes much time to solve the problem. The Benders decomposition approach is considered for lower bound of the AGVsp-P/D. The generated Benders'decomposition algorithm for solving the lower bound of the AGVsp-P/D is described in section 2.3.

The ordinary assignment problem is the 0-1 IP problem. An assignment problem can be solved as a regular LP without concerning of 0-1 integer constraints because of the unimodularlity of the network structure. The result is an integer solution automaticly (Mathur and Solow, 1994). When the alternative nodes constraints are added to the system, the properties of the problem will be changed. The heuristics for the alternative selection and the improvement of selection for solving the lower bound of AGVsp-P/D as solving the regular assignment problem are presented in section 2.4. The lower bound model of AGVsp-P/D and its solving approaches are programmed and tested on the computer by using MATLAB 7.0 and Excel Solver.

The solutions of many tested problems, which are presented in the next chapter, will sometimes form the single TSP tour but sometimes will not. After the lower bound model is completed, the modified branch and bound and heuristic approaches are applied to generate the TSP/MTSP tour of the schedule for multi/single AGVsp-P/D. The modified Eastman's algorithm for TSP of the AGVsp-P/D is presented in section 2.5. The last section presents the heuristic for solving multi AGVsp-P/D by using the methods of solving MTSP as standard TSP and using the heuristic of splitting TSP tour.

2. The Problem Analysis and Solution Technique

2.1 AGV Scheduling Problems Analysis

Let consider the problem that the factory has a particular layout of departments for the AGV system as figure 1. From the example layout, let assume that the distance between each department (node) is shown in table 3.

Table 3 The distance table of the example layout from figure 1

From \ To	A	B	C	D	E	F	G	H	I
A	∞	1	2	1	2	3	2	3	4
B		∞	1	2	1	2	3	2	3
C			∞	3	2	1	4	3	2
D				∞	1	2	1	2	3
E					∞	1	2	1	2
F						∞	3	2	1
G							∞	1	2
H								∞	1
I									∞

Normally, the list of jobs for AGV problems can be defined as the example on table 4. In general, each job of AGVs composes of pick up the items at one node and delivers them at one fixed destination node. For example, let consider a job No. 1 on table 4, the AGV travels from a starting department (node A) to the pick up node B for getting the items and then travels to the delivery node C for finally sending items.

Table 4 The example of a part of job list for a regular AGV problem

Job No.	Pick up Department	Delivery Department
1	B	C
2	A	I
3	B	H
4	G	C
5	D	E
6	H	F

Blair, Charnsethikul and Vasques (1987) modeled the optimum routing problem of AGVs among the workstations as TSP/MTSP, mentioned previously. An algorithm for the near optimal routing of AGVs in such a system is presented which seeks to organize materials move into tours with the objective of minimizing the maximum tour length.

For this research, the specific characteristic of alternative P/D nodes here is considered the jobs that can have the alternative pick up and delivery nodes to select more than one fixed point. Suppose in some parts of the example, the list of jobs that one AGV is used to complete all jobs is shown as follows.

Table 5 The example of a part of job list for the single AGVsp-P/D

Job No.	Pick up Department	Delivery Department
1	B	C
2	A	I
3	B	H or G or I
4	G	C
5	D	E
6	D or H	F

The meaning of each job of AGVsp-P/D can be explained as the following example. Let consider the job No. 3 on table 5, the AGV job is the item movements that pick up the items from the turning process at department (node) B and deliver at the drilling process, which can be performed at departments H or G or I. The AGV

has to travel from pick up node B and can select to deliver the items at nodes H or G or I that is called “alternative pick up and delivery nodes” (alternative P/D nodes). If the AGV travels from node B and select to deliver at node H, the total AGV traveling distance may different from selecting to deliver at node G or node I. The job scheduling of jobs (for example started with job No. 1 and followed by job No. 6, No. 5, and ended the schedule when all jobs done) and selecting of alternative P/D nodes appropriately may provides the minimized total traveling distance of the AGV, which is the objective of this research.

The AGVsp-P/D can be transformed to TSP for solving the special situation that some jobs of AGVs have the P/D alternatives. The distance matrix $[c_{ij}]$ of AGVsp-P/D in a form of TSP is defined, which the table is consisted of the distance of AGVs that move from the starting point of the current job to the starting point of the next job. So the TSP distance table for the AGV problem is an asymmetric distance table. Suppose the distance table of the previous AGVs job list on table 5, which is the distance from the considered job to the others in a form of TSP distance table, is shown as follows.

Table 6 The example of distance matrix $[c_{ij}]$ of the AGVsp-P/D in a form of TSP

To / From	Job *j* No. (*n*)		1	2		3		4	5	6	
Job *j* No.	Alternative (job *i*, Alt. *a*)		1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
		n	1	2	3	4	5	6	7	8	9
1	1.1	1	∞	3	2	2	2	5	4	2	4
2	2.1	2	7	∞	7	7	7	6	7	7	7
	3.1	3	2	3	∞	∞	∞	3	2	2	2
3	3.2	4	6	5	∞	∞	∞	3	4	6	4
	3.3	5	6	7	∞	∞	∞	5	6	6	6
4	4.1	6	5	6	5	5	5	∞	7	5	7
5	5.1	7	2	3	2	2	2	3	∞	2	2
	6.1	8	4	7	4	4	4	5	4	∞	∞
6	6.2	9	3	5	4	4	4	6	7	∞	∞

From table 6, assume that job No. 1 is a starting job with a starting node B (depot) for the AGV. Job No. 1 of the movement from node B to node C is a regular job (not has any alternative P/D nodes) that has label 1.1, but job No. 3 of the movement from node B to the selected alternative delivery nodes H or G or I is an alternative job that are labeled as 3.1, 3.2, and 3.3. Job No. 3 can be separated to 3.1 (B to H), 3.2 (B to G), and 3.3 (B to I). The distance on table 6 is the distance of the AGV that move from the pick up nodes of the current job to the pick up nodes of the next job. For example, the distance of the AGV that move from job 1.1 to job 2.1 is 3 units, which is the summation distance from the pick up node of job 1.1 (node B) to the delivery node of job 1.1 (node C), 1 unit, and the distance from node C to the pick up node of job 2.1 (node A), 2 units, that equal to 1+2 = 3 units. When the TSP approach is applied to this table with alternative P/D nodes constraints, the solution of AGVsp-P/D can be generated.

2.2 Problem Formulation of the AGVsp-P/D

When mathematical formulations of routing problems are studied, there are so many kinds of mathematical models as mentioned previously in the literature review part. When the presented problem statement of AGVsp-P/D is compared to the TSP, the detail can be analyzed as follows.

Refer to the stated problem statement, given a set of n jobs J such that job $J_i = \{P_{i\,a}, D_{i\,b}\}$, $i = 1, 2, \ldots, n$ where $P_{i\,a}$ is a set of alternative pick up departments a of job J_i, $a = \{1, 2, \ldots, k(i)\}$ and $D_{i\,b}$ is a set of alternative delivery departments b of job J_i, $b = \{1, 2, \ldots, l(i)\}$. $k(i)$ is the number of alternative departments a for job J_i. $l(i)$ is the number of alternative departments b for job J_i. When job $J_j = \{P_{j\,a}, D_{j\,b}\}$, $j = 1, 2, \ldots, n$ is scheduled after job J_i, $c_{ia\,jb}$ is the traveling distance of an AGV that starts from a selected pick up department a of job J_i, goes to a selected delivery department b of job J_i, goes to a selected pick up department a of job J_j, then goes to a selected delivery department b of job J_j, which is a non-negative number and $c_{ia\,ia} = \infty$. The AGVsp-P/D is the problem that selects one alternative department from set a and one

alternative department from set b of all jobs J, called $x_{ia\,jb}$ such that $x_{ia\,jb} = 1$ if an AGV travels from a selected pick up department a of job J_i to a selected delivery department b of job J_j or $x_{ia\,jb} = 0$ otherwise and sequences all those jobs J with their selected alternatives to form single/multi tours (TSP/MTSP tour) that provide minimized the total traveling distance. The single/multi AGVsp-P/D relates directly to TSP/MTSP.

Refer to Miller-Tucker-Zemlin (1960)'s formulation of classical TSP and Svestka and Huckfeldt (1973)'s formulation of MTSP. TSP/MTSP variable x_{ij} is equal 1 if the salesman travels from node i to node j. When the alternative P/D nodes structure is analyzed on the model's variables, if the salesman travels from node i with alternative a to node j with alternative b the variable should be $x_{ia,\,jb} = 1$. By similar idea, the TSP/MTSP with alternative P/D can be formulated base on the original model but change x_{ij} to $x_{ia,\,jb}$. This research does the analysis and creates the model of TSP/MTSP with alternative P/D nodes, n nodes and m AGVs, which similar idea with the original TSP/MTSP model as follows.

$$MinZ = \sum_{i=1}^{r}\sum_{a=1}^{k(i)}\sum_{j=1}^{r}\sum_{b=1}^{l(j)} x_{i\,a\,jb}\, c_{i\,a\,jb} \qquad (19)$$

Subject to

$$\sum_{i=1}^{r}\sum_{a=1}^{k(i)} x_{i\,a\,jb} = 1, \qquad \forall j = 1,\, 2,\ldots, r \;\; and \;\; \forall b = 1,\, 2,\ldots, l(j) \qquad (20)$$

$$\sum_{j=1}^{r}\sum_{b=1}^{l(j)} x_{i\,a\,jb} = 1, \qquad \forall i = 1,\, 2,\ldots, r \;\; and \;\; \forall a = 1,\, 2,\ldots, k(i) \qquad (21)$$

$$y_i - y_j + r x_{i\,a\,jb} \le r - 1 \qquad \forall\, i \ne j, a, b \qquad (22)$$

$$x_{i\,a\,jb} = 0\, or\, 1 \qquad \forall\, i, a, j, b$$

$$a, b > 0\, and\ Integer$$

where

r $= n + m - 1$

$x_{ia\,jb}$ = 1; If one AGV travel from node i with alternative a to node j with alternative b (for example, $x_{11\,31}$ is that the AGV travel from node No. 1 with alternative No. 1 to node No. 3 with alternative No. 1)

= 0; otherwise

$c_{ia\,jb}$ represents the distance from node i with alternative a to node j with alternative b

$k(i)$ represents the number of alternative departments a for node J_i.

$l(i)$ represents the number of alternative departments b for node J_i.

m represents the number of AGVs (m = 1 when considers the single AGV case)

This mathematical model is a TSP/MTSP with alternative P/D nodes that can simulate the model of single/multi AGVsp-P/D by identifying nodes of the TSP/MTSP as jobs of AGVs. This model can not be solved regularly same as the original TSP/MTSP. Therefore, the model has to be modified as follows.

When relax the subtour elimination constraints (22) and consider the single AGV case, this problem looks like the assignment problem, but there are the alternative P/D nodes for each job. This assignment problem with alternative P/D is the 0-1IP model that is the relaxation of TSP/MTSP with alternative nodes. For solving TSP/MTSP with alternative P/D nodes, the solving algorithm has to apply the branch and bound approach with solving the assignment problem with alternative P/D nodes as a subproblem of each branching. The solution from solving this assignment problem provides the lower bound of AGVsp-P/D and can be modified to be the simpler mathematical model. The variable $x_{ia\,jb}$ is considered to be eliminated the subscribes a and b. The main propose of the modification is to create the model that can be solved by similar approaches of solving the regular assignment problem. For clearly explaining, the table 6 (the example of cost matrix $[c_{ij}]$ of the AGVsp-P/D in a form of TSP) and table 7 (The assignment solution of variable x_{ij} of AGVsp-P/D from table 6) is considered concomitantly. The modified mathematical model is explained as follows, where h = number of all nodes (all rows or columns) that consist of all alternatives and n = number of jobs.

Let the new variable x_{ij} is a 0-1 integer variable, which indicating whether the schedule of the AGV is accomplished from node (row) i to node (column) j or not and can be solved as an original assignment problem. The dummy variable Z is introduced to the model to capture the alternative structures and eliminate the subscribes a and b of the variables $x_{ia\,jb}$. It represents to the summation of variables x_{ij} for each row and column. Consider table 7 for clearly explaining, job No.1 is a regular job, which not has P/D alternatives. It consists of one row (row No.1) of job 1.1 and one column on table 7. The dummy variable $Z_{(i)}$ of row No.1 ($Z_{(1)}$) is the summation of the solution of variables x_{ij} of row No.1, which equal to 1. For job No. 3, it has 3 alternative P/D nodes, which it consists of row No.3, row No.4 and row No.5 of job 3.1, job 3.2 and job 3.3 sequentially. Table 7 shows that the dummy variables $Z_{(i)}$ of row No.3 ($Z_{(3)}$) equals to 1, row No.4 ($Z_{(4)}$) equals to 0 and row No.5 ($Z_{(5)}$) equals to 0. Because only one alternative of job No. 3 (job 3.1or 3.2 or 3.3) will be selected, only one row of job No.3 (row No.3 or row No.4 or row No.5) will has the summation of the solution of variable x_{ij} equal to 1, which is row No.3 of job 3.1 in this case. According to this point, all $Z_{(i)}$ variables of job No.3 ($Z_{(3)}$, $Z_{(4)}$, and $Z_{(5)}$) will have the summation equal to 1 ($Z_{(3)} + Z_{(4)} + Z_{(5)} = 1$). Table 7 is the example of AGVsp-P/D, which consists of 9 nodes (9 rows and columns) with 6 jobs ($h = 9$ and $n = 6$). Set $S(k)$ is defined to represent the set of all alternatives of any job k ($k = 1, 2, \ldots, n$). From table 7, there are six sets $S_{(k)}$ of 6 jobs that are set $S_{(1)}$ of job No.1, which consists of row No.1 of job 1.1, set $S_{(2)}$of job No.2, which consists of row No.2 of job 2.1, set $S_{(3)}$ of job No.3, which consists of rows No.3, 4 and 5 of jobs 3.1, 3.2 and 3.3, set $S_{(4)}$ of job No.4, which consists of row No.6 of jobs 4.1, set $S_{(5)}$ of job No.5, which consists of row No.7 of job 5.1 and set $S_{(6)}$ of job No.6, which consists of rows No.8 and 9 of jobs 6.1 and 6.2. Therefore, the constraints of alternative P/D nodes of the modified model is that the summation of all dummy variables $Z_{(i)}$ of all rows No. i in each set $S_{(k)}$ of any jobs k will equal to 1.

According to this point, the created lower bound model of AGVsp-P/D is shown as follows.

$$MinZ = \sum_{i=1}^{h}\sum_{j=1}^{h} x_{ij}\, c_{ij}$$

Subject to

$$\sum_{i=1}^{h} x_{ij} = Z_{(j)} \qquad \forall j = 1,\ 2,\ldots, h$$

$$\sum_{j=1}^{h} x_{ij} = Z_{(i)} \qquad \forall i = 1,\ 2,\ldots, h$$

$$\sum_{i \in S_{(k)}} Z_{(i)} = 1 \qquad \forall k = 1,\ 2,\ldots, n$$

$$x_{ij} = 0 \text{ or } 1 \qquad \forall i, j$$

where

x_{ij} = 1; If one AGV travel from node i to node j

= 0; otherwise

c_{ij} represents the distance from pick up node i through the path to delivery node j

$S_{(k)}$ represents the set of all P/D alternatives of job k

n represents the number of jobs

h represents the number of nodes

Now the lower bound model of AGVsp-P/D can be programmed in Excel Solver and MATLAB 7.0. After the lower bound model is implemented with the example of distance matrix [c_{ij}] of the AGVsp-P/D on table 6 by using Excel Solver, the result is shown on table 7.

Table 7 The assignment solution of variable x_{ij} of AGVsp-P/D from table 6

Job No.			1	2		3		4	5	6		
(n)	Job i, Alt a		1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2	
		h	1	2	3	4	5	6	7	8	9	Z_i
1	1.1	1	∞	0	1	0	0	0	0	0	0	1
2	2.1	2	0	∞	0	0	0	1	0	0	0	1
	3.1	3	0	0	∞	∞	∞	0	1	0	0	1
3	3.2	4	0	0	∞	∞	∞	0	0	0	0	0
	3.3	5	0	0	∞	∞	∞	0	0	0	0	0
4	4.1	6	0	1	0	0	0	∞	0	0	0	1
5	5.1	7	0	0	0	0	0	0	∞	0	1	1
	6.1	8	0	0	0	0	0	0	0	∞	∞	0
6	6.2	9	1	0	0	0	0	0	0	∞	∞	1
		Z_j	1	1	1	0	0	1	1	0	1	

From the Excel Solver solution, the assignment solution is 1.1 - 3.1, 2.1 - 4.1, 3.1 - 5.1, 4.1 - 2.1, 5.1 - 6.2, 6.2 - 1.1 and the solution value of minimum total distance that is 21 units. The alternative job No. 3.1 (B to H) and job No. 6.2 (H to F) are selected. The solution forms 2 subtours that are 1.1-3.1-5.1-6.2-1.1 and 2.1-4.1-2.1. The meaning of this solution is that one AGV starts at node B of job No.1 and accomplishes job No.1 at node C, travels to node B, which is the starting job of job No. 3.1and accomplishes job No. 3.1 at node H, travels to node D, which is the starting job of job No.5 and accomplishes job No.5 at node E, travels to node H, which is the starting job of job No. 6.2 and accomplishes job No. 6.2 at node F, then travels back to node B which is the starting node of this AGV. Another AGV starts at node A of job No.2.1 and accomplishes job No.2.1 at node I, travels to node G which is the starting job of job No.4 and accomplishes job No.4.1 at node C, then travels back to node A, which is the starting node of this AGV.

According to this point, the problem size is increased and the research found that the ordinary version of Microsoft Excel Solver can run only 13 nodes (h = 13). Solver shows that "Too many adjustable cells" and terminate running. Then MATLAB 7.0 is applied. The experiment is performed by randomly generating simulated problem of 10, 20, 30, 40, and 50 nodes with some numbers of 2 alternative jobs and some numbers of regular jobs. The running time of this lower bound model is

examined and compared to the original assignment problem of same problem size. The result is shown in the next chapter.

After using MATLAB 7.0 on a 2 GB RAM computer to perform the experiment, the research found that MATLAB 7.0 can run steadily up to 50 nodes (h = 50) with an average running time about 50 seconds but MATLAB 7.0 shows "Out of memory" of running the *bintprog* function, which is the function of solving 0-1 IP, beyond 50 nodes. Because this mathematical model of the lower bound of AGVsp-P/D is 0-1 IP, the operation find the optimal solution by using branch and bound approach for solving 0-1 IP but if a considered situation is a larger scale problem, IP may take much more memory to run. From the experimental results, the resource of required memory is the problem, not the running time, so that the researcher attempts to solve the lower bound model of AGVsp-P/D in LP rather than 0-1 IP. The decomposition techniques and other heuristic methods are considered for solving the larger scale problem to avoid running out of memory. Next, the research tries to apply Benders'decomposition to solve the lower bound model of AGVsp-P/D.

2.3 The lower bound of AGVsp-P/D by Benders'decomposition approach

Let consider the Benders'decomposition approach for MIP. The lower bound model of AGVsp-P/D in a similar form of MIP of the Benders'decomposition will be considered. Refer to Benders'decomposition algorithm, it can be applied to the lower bound model of AGVsp-P/D by partitioning the variables of the into two sets which are x_{ij} and $Z_{(i)}$ and projecting the problem onto $Z_{(i)}$ variables. Consider the example on table 7 for clearly explaining, there are nine $Z_{(i)}$ variables of nine nodes (rows), which can be separated into six sets of $S_{(k)}$. They are $S_{(1)} = \{ Z_{(1)}\}$, $S_{(2)} = \{ Z_{(2)}\}$, $S_{(3)} = \{ Z_{(3)}, Z_{(4)}, Z_{(5)}\}$, $S_{(4)} = \{ Z_{(6)}\}$, $S_{(5)} = \{ Z_{(7)}\}$ and $S_{(6)} = \{ Z_{(8)}, Z_{(9)}\}$. Because only one alternative node of each job will be selected in each iteration of solving the model by using Benders'decomposition, only one variable $Z_{(i)}$ of each set $S_{(k)}$ will be fixed to be 1 and the others will be 0. The alternative nodes, which have the value of variable $Z_{(i)}$ = 0, can be ignored from the model so that the model becomes the regular assignment problem. All variables x_{ij} , which are 0-1 integer, can be ignore to become $x_{ij} \geq 0$,

because of the property of assignment problem (Mathur and Solow, 1994). Let $\boldsymbol{Z}$ denotes the sets of all feasible 0-1 integer vectors Z, then the lower bound model of AGVsp-P/D in a similar form of MIP by the Benders'decomposition approach can be written as following.

$$\begin{aligned} \text{Minimize} \quad & c^T x + d^T Z \\ \text{Subject to} \quad & \boldsymbol{A}x + \boldsymbol{B}Z \geq b, \\ & x \geq 0,\ Z \in \boldsymbol{Z} \end{aligned}$$

where $\boldsymbol{A}$ is a m by n coefficient matrix of vector x,

$\boldsymbol{B}$ is a m by n' coefficient matrix of vector Z,

c is a n by 1 cost vector of vector x,

d is a n' by 1 dummy cost vector of vector Z, which is a zero vector

x is a n by 1 vector of variable x_{ij},

Z is a n' by 1 vector of variable $Z_{(i)}$ with $\boldsymbol{Z} = \{Z \mid Z_{(i)} \in \{0, 1\};\ i = 1, 2, \ldots, n'\}$

Let $v(Z) = d^T Z + \text{maximize } \{(b - \boldsymbol{B}Z)^T u \mid \boldsymbol{A}^T u \leq c,\ u \geq 0\}$

where u is a m by 1 vector of dual variable u.

When the Benders'algorithm is applied for solving the example of AGVsp-P/D on table 6, the function $v(Z)$ is:

$$\begin{aligned} v(Z = [Z_{(1)}, Z_{(2)}, Z_{(3)}, Z_{(4)}, Z_{(5)}, Z_{(6)}, Z_{(7)}, Z_{(8)}, Z_{(9)}]^{\mathrm{T}}) \\ = \text{maximize } \{(b - \boldsymbol{B}Z)^T u \mid \boldsymbol{A}^T u \leq c,\ u \geq 0\}. \end{aligned}$$

Because this example is the 9 nodes problem, the dual problem of this example consists of 18 dual variables u and 81 constraints. The example of applying Benders'decomposition approach to solve the lower bound model of AGVsp-P/D of the example on table 6 is explained as follows.

The Benders'decomposition algorithm for the variables x_{ij} and $Z_{(i)}$ for this example is:

Step 1: Initialization: set $v(Z) = 0$, select a fixed vector $Z^j \in \boldsymbol{Z}$, set $j = 1$ and set $k = 1$

Step 2: Solve the Benders'subproblem: evaluate the value of $v_k(Z^j)$ with its corresponding set of dual extreme point (vector u^j) by solving LP with a fixed vector Z^j

Step 3: Stopping criterion: if $v_k(Z^j) = v(Z)$ then stop, otherwise go to step 4

Step 4: Improve an approximation function: by using a set of dual extreme point (vector u^j) to generate the Benders'cut for forming iteration k^{th} of Benders'partial master problem

Step 5: Solve the Benders'partial master problem: that is minimizing $v(Z)$ with Benders'cut from step 4 for updating the value of $v(Z)$ and then set $j=j+1$ and updating the new vector Z^j, set $k = k+1$ and go to step 2.

When the algorithm is implemented, the result is illustrated as follows.

Iteration 1:

Step 1: Initialization:

Let set $v(Z) = 0$, select $v(Z = [Z_{(1)}, Z_{(2)}, Z_{(3)}, Z_{(4)}, Z_{(5)}, Z_{(6)}, Z_{(7)}, Z_{(8)}, Z_{(9)}]^T) = v(Z^1 = [1, 1, 0, 0, 1, 1, 1, 0, 1]^T)$, set $j = 1$ and set $k = 1$.

Step 2: Solve the Benders'subproblem:

The first Benders'subproblem of the example on table 6, which is:

Maximize $v_1(Z^1)$ = Maximize $\{(b - \boldsymbol{B} Z^1)^T u^1 \mid \boldsymbol{A}^T u^1 \leq c, u^1 \geq 0\}$,

is solved. The maximum occurs at the extreme point $u^1 = [u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, u_9, u_{10}, u_{11}, u_{12}, u_{13}, u_{14}, u_{15}, u_{16}, u_{17}, u_{18}]^T = [0, 5, 0, 2, 4, 3, 0, 1, 0, 2, 3, 2, 2, 2, 1, 2, 2, 2]$ and maximum value of $v_1(Z^1) = 24$.

Step 3: Stopping Criterion:

Now the value of $v(Z) = 0$, $v_1(Z^1) = 24 \neq v(Z)$ then go to step 4

Step 4: Improve the approximations function:

Using the dual extreme point u^1 generates the approximations function ($v(Z)$), with a Benders'cut, for the Benders'partial master problem of the iteration 1. The Benders'partial master problem is:

Minimize $v(Z) = d^T Z +$ maximize $\{ [(b - \boldsymbol{B}Z)^T u^1]_1 \}$

Subject to $Z \in \boldsymbol{Z}$

A Benders'cut of the iteration 1 is $[(b - \boldsymbol{B}Z)^T u^1]_1$ that is:

$[0Z_{(1)}+5Z_{(2)}+0Z_{(3)}+2Z_{(4)}+4Z_{(5)}+3Z_{(6)}+0Z_{(7)}+1Z_{(8)}+0Z_{(9)}+2Z_{(1)}+3Z_{(2)}+2Z_{(3)}+2Z_{(4)}+2Z_{(5)}+1Z_{(6)}+2Z_{(7)}+2Z_{(8)}+2Z_{(9)}]_1$

$= 2Z_{(1)}+8Z_{(2)}+2Z_{(3)}+4Z_{(4)}+6Z_{(5)}+7Z_{(6)}+2Z_{(7)}+3Z_{(8)}+2Z_{(9)}$

Because a vector d is a zero vector, the Benders'partial master problem for iteration 1is:

Minimize $v(Z) =$ maximize $\{ [2Z_{(1)}+8Z_{(2)}+2Z_{(3)}+4Z_{(4)}+6Z_{(5)}+7Z_{(6)}+2Z_{(7)}+3Z_{(8)}+2Z_{(9)}]_1 \}$

Subject to $Z \in \boldsymbol{Z}$

Step 5: Solve the Benders'partial master problem:

Update $j = 2$, $k = 2$ and the value of $v(Z)$ from solving the Benders'partial master problem = 20 with is new vector $Z = Z^2 = [1, 1, 1, 0, 0, 1, 1, 0, 1]^T$

Iteration 2:

Step 2: Solve the Benders'subproblem:

The Benders'subproblem of iteration 2 is:

Maximize $v_2(Z^2) =$ Maximize $\{(b - \boldsymbol{B}Z^2)^T u^2 \mid \boldsymbol{A}^T u^2 \leq c,\ u^2 \geq 0\}$,

is solved. The maximum occurs at the extreme point $u^2 = [u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, u_9, u_{10}, u_{11}, u_{12}, u_{13}, u_{14}, u_{15}, u_{16}, u_{17}, u_{18}]^T = [0, 5, 0, 1, 1, 3, 0, 1, 1, 2, 3, 2, 2, 1, 1, 2, 2, 2]$ and maximum value of $v_1(Z^1) = 21$.

Step 3: Stopping Criterion:

Now the current value of $v(Z) = 20$. Because $v_1(Z^1) = 21 \neq v(Z)$, not terminate, then go to step 4

Step 4: Improve the approximations function:

Using the dual extreme point u^2 generates an approximations function (v (Z)), with a Benders'cut. A Benders'cut of the iteration 2 is $[(b - \boldsymbol{B}Z)^T u^2]_2$ that is:

$[0Z_{(1)}+5Z_{(2)}+0Z_{(3)}+1Z_{(4)}+1Z_{(5)}+3Z_{(6)}+0Z_{(7)}+1Z_{(8)}+1Z_{(9)}+2Z_{(1)}+3Z_{(2)}+2Z_{(3)}+2Z_{(4)}+1Z_{(5)}+1Z_{(6)}+2Z_{(7)}+2Z_{(8)}+2Z_{(9)}]_1$

$= 2Z_{(1)}+8Z_{(2)}+2Z_{(3)}+3Z_{(4)}+2Z_{(5)}+4Z_{(6)}+2Z_{(7)}+3Z_{(8)}+3Z_{(9)}$

The Benders'partial master problem for iteration 2 is:

Minimize v (Z) = maximize { $[2Z_{(1)}+8Z_{(2)}+2Z_{(3)}+4Z_{(4)}+6Z_{(5)}+7Z_{(6)}+2Z_{(7)}+3Z_{(8)}+2Z_{(9)}]_1$,

$[2Z_{(1)}+8Z_{(2)}+2Z_{(3)}+3Z_{(4)}+2Z_{(5)}+4Z_{(6)}+2Z_{(7)}+3Z_{(8)}+3Z_{(9)}]_2$}

Subject to $Z \in \boldsymbol{Z}$

Step 5: Solve the Benders'partial master problem:

Update j = 3, k = 3 and the value of v (Z) from solving the Benders'partial master problem = 21 with is new vector $Z = Z^3 = [1, 1, 1, 0, 0, 1, 1, 0, 1]^T$

Iteration 3:

Step 2: Solve the Benders'subproblem:

The Benders'subproblem of iteration 3 is:

Maximize v_3 (Z^3) = Maximize $\{(b - \boldsymbol{B}\, Z^3)^T u^3 \mid \boldsymbol{A}^T u^3 \le c,\ u^3 \ge 0\}$,

is solved. The maximum occurs at the extreme point $u^2 = [u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, u_9, u_{10}, u_{11}, u_{12}, u_{13}, u_{14}, u_{15}, u_{16}, u_{17}, u_{18}]^T = [0, 5, 0, 1, 1, 3, 0, 1, 1, 2, 3, 2, 2, 1, 1, 2, 2, 2]$ and maximum value of v_3 (Z^3) =21.

Step 3: Stopping Criterion:

Now the current value of v (Z) = 21. Because v_3 (Z^3) =21 = v (Z), terminate the algorithm and stop.

The solution from the Benders' algorithm now is same as the solution from IP which the assignment is 1.1 - 3.1, 2.1 - 4.1, 3.1 - 5.1, 4.1 – 2.1, 5.1 - 6.2 and 6.2 – 1.1 with the solution value of minimized total distance of 21 units, that same as the solution from solving 0-1 linear programming. Alternative 3.1 and 6.2 are selected.

The example result shows that the solution provide two subtours that can assigns to 2 AGVs which starts at node 1 and node 2. This solution can be the lower bound of the AGVsp-P/D. This lower bound can be used in branch and bound approach to find a single TSP tour that is the optimal schedule of the single AGVsp-P/D.

There are some solutions provide the assignment solution as a single tour that is the TSP solution, but most of them provide subtours. According to this point, the model can be applied to any size of problem but the Benders'decomposition algorithm may consists of many iterations for generating one lower bound solution. The algorithm is so complicate and still has to solve 0-1 IP in step 5 for finding vectors *Z*. However, refer to the result section, the 0-1 IP lower bound problem with 50 nodes can be solved by MATLAB 7.0 but most of problems which have more than 50 nodes (h=50) cause MATLAB 7.0 shows "Out of memory" in calculation of binary problem. Benders'decomposition algorithm can be applied for the larger scale problem, which MATLAB 7.0 can not generate solution, because the problem size of 0-1 IP of the Benders'partial master problem in each iterations is smaller than the problem size of 0-1 IP of the original lower bound model for the same tested problem.

For example, let consider a problem of 60 nodes, the 0-1 IP model has the matrix [$\boldsymbol{A}$] with the size of 120 x 3600, 3600 x_{ij} variables, that cause MATLAB 7.0 can not calculate binary problems and shows "out of memory". The example of applying the Benders'decomposition algorithm to the 60 nodes problem is shown in an appendix. However, the research attempts to examine the methods to solve the lower bound of AGVsp-P/D without solving 0-1 IP and easier to process than Benders' decomposition algorithm. The research forms some heuristic approaches. The following section presents heuristics for selecting alternative nodes that can provide that assignment solutions, which close to solutions from the 0-1 IP model.

2.4 Heuristic approaches for solving the lower bound of AGVsp-P/D

The lower bound of AGVsp-P/D, which is the assignment problem with alternative P/D nodes, can be solved by selecting the appropriate alternative nodes first and then solving the regular assignment problem. The considered problem is "How to select the best alternative that can provide the best assignment solution?" This section presents heuristics for selecting the appropriate alternatives of each job and the heuristic for improving the selected alternatives that can provide the assignment solution, which close to the solution from the 0-1 IP model. Suppose the example of distance matrix, called the master matrix, of 6 jobs that job 3 has 3 alternatives and job 6 has 2 alternatives, is shown on table 8 as follows.

Table 8 The example of master matrix

Job No.		1	2		3		4	5	6	
	Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1	1.1	∞	2	11	3	35	94	30	13	97
2	2.1	3	∞	57	73	86	23	21	61	83
	3.1	85	27	∞	∞	∞	41	11	66	27
3	3.2	48	57	∞	∞	∞	52	46	73	52
	3.3	80	66	∞	∞	∞	58	79	63	28
4	4.1	61	37	33	0	56	∞	88	87	9
5	5.1	72	16	68	14	20	485	∞	4	70
	6.1	22	43	62	17	88	21	44	∞	∞
6	6.2	96	18	86	60	34	42	15	∞	∞

The solution of this master matrix from solving the 0-1 IP model is 74 units with the assignment solution 1.1-2.1, 2.1-1.1, 3.1-5.1, 4.1-3.1, 5.1-6.1, and 6.1-4.1. Manually, the assignment problem with alternative P/D nodes can be solved by selecting the appropriate alternative for job No. 3 and job No. 6 first and then solve the regular assignment problem. This research tries to create 3 heuristics for alternative selection that can provide the initial solution and the heuristic for improving the alternative selection. All for alternative selection heuristics are:

Heuristic-1 for selecting the alternative nodes

Step 1**:** For all rows of job *j* that have alternative nodes, compute the average of all cost elements in each row

Step 2: For all columns of job *j* that have alternative nodes, compute the average of all cost elements in each column

Step 3: Compute the average value of cost elements from all rows, from step 1, and the correlated columns, from step 2, of each alternative job**s** *j*

Step 4: Select the alternatives for each job *j* that have the minimum value from step 3 and solve the assignment problem with this selected alternatives

Heuristic-2 for selecting the alternative nodes

Step 1: Solve the assignment problem of the master matrix

Step 2: Calculate the average of the assignment solution of all rows and columns of each alternative

Step 3**:** Select the alternatives that have the minimum value of the average of assignment solution and solve the assignment problem with this selected alternatives

Heuristic-3 for selecting the alternative nodes

Step 1: Create the distance matrix**,** $[d_{ij}]$, that consists of minimum distance of all jobs

Step 2: Solve the assignment problem of the distance matrix**,** $[d_{ij}]$

Step 3: Select feasible alternatives of master matrix, which provide the minimum increasing (penalty) of cost elements from the assignment solutions of step 2

Step 4: Solve the feasible assignment solution.

For explaining, the master matrix on table 8 is used to show the implementation of all heuristics. The details are shown step by step for all heuristics as follows.

The example of implementing the Heuristic-1 of the alternative nodes selection:

Step 1 and step 2 are explained as follows.

Step 1**:** For all rows of job *j* that have alternatives, compute the average of the cost in each row

Step 2: For all columns of job *j* that have alternatives, compute the average of the cost in each column

Table 9 The example of step 1 and step 2 of Heuristic-1

Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2	
1.1	-	2	11	3	35	94	30	13	97	
2.1	3	-	57	73	86	23	21	61	83	Avg.
3.1	85	27	-	-	-	41	11	66	27	***42.833***
3.2	48	57	-	-	-	52	46	73	52	***54.667***
3.3	80	66	-	-	-	58	79	63	28	***62.333***
4.1	61	37	33	0	56	-	88	87	9	
5.1	72	16	68	14	20	485	-	4	70	
6.1	22	43	62	17	88	21	44	-	-	***42.429***
6.2	96	18	86	60	34	42	15	-	-	***50.143***
		Avg.	***52.833***	***27.833***	***53.167***			***52.429***	***52.286***	

Step 3: Compute the average value of each alternative job**s**

3.1: (42.83+52.83)/2 = 47.83

3.2: (54.66+27.83)/2 = ***41.25***

3.3: (62.33+53.16)/2 = 57.75

6.1: (42.43+52.43)/2 = ***47.43***

6.2: (50.14+52.28)/2 = 51.21

Step 4: Select the alternatives that have the minimum value of average cost and solve the assignment problem with this selection

From step 3, job 3.2 and 6.1 are selected. The assignment solutions with the selected jobs are shown on table 10 as follows. The solution is 76 units.

Table 10 The assignment solution of the example of Heuristic-1

Job *i*, Alt *a*	1.1	2.1	3.2	4.1	5.1	6.1
1.1	∞	***2***	3	94	30	13
2.1	***3***	∞	73	23	21	61
3.2	48	57	∞	52	***46***	73
4.1	61	37	***0***	∞	88	87
5.1	72	16	14	485	∞	***4***
6.1	22	43	17	***21***	44	∞

The example of implementing the Heuristic-2 for the alternative nodes selection:

Step 1: Solve the assignment problem of the master matrix

The assignment solution with the selected jobs is shown as follow.

Table 11 The assignment solution of step 1of Heuristic-2

Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	2	***11***	3	35	94	30	13	97
2.1	***3***	∞	57	73	86	23	21	61	83
3.1	85	27	∞	∞	∞	41	***11***	66	27
3.2	48	***57***	∞	∞	∞	52	46	73	52
3.3	80	66	∞	∞	∞	58	79	63	***28***
4.1	61	37	33	***0***	56	∞	88	87	9
5.1	72	16	68	14	20	485	∞	***4***	70
6.1	22	43	62	17	88	***21***	44	∞	∞
6.2	96	18	86	60	***34***	42	15	∞	∞

Step 2: Calculate the average of assignment solution of all rows and columns of each alternative

From table 11, the column of job 3.1 has the assignment solution =11 and the rows of job 3.1 has the assignment solution =11. The average assignment solution of job 3.1 is (11+11)/2 = 11. The average assignment solutions of all alternative jobs are:

3.1: (11+11)/2 = 11

3.2: (0+57)/2 = 28.5

3.3: (34+28)/2 = 31

and

6.1: (4+21)/2 = 12.5

6.2: (28+34)/2 = 31

Step 3**:** Select the alternatives that have the minimum value of the average of assignment solution and solve the assignment problem with this selected alternatives From step 2, job 3.1 and 6.1 are selected. The assignment solution with selected jobs is shown on the table 12 as follows. The solution is 74 units

Table 12 The assignment solution of step 2 of Heuristic-2

Job *i*, Alt *a*	1.1	2.1	3.1	4.1	5.1	6.1
1.1	∞	***2***	11	94	30	13
2.1	***3***	∞	57	23	21	61
3.1	85	27	∞	41	***11***	66
4.1	61	37	***33***	∞	88	87
5.1	72	16	68	485	∞	***4***
6.1	22	43	62	***21***	44	∞

The example of implementing the Heuristic-3 for the alternative nodes selection:

Step 1: Create the distance matrix**,** $[d_{ij}]$ that consists of minimum distance of all jobs

The distance matrix, $[d_{ij}]$ is shown on table 13 as follows. For all alternative jobs *j*, the notation of *j*.X represents the job with the minimum distance element. For example, job 3.X represents the job that all distance elements are the minimum value from jobs 3.1, 3.2 and 3.3.

Table 13 The distance matrix**,** $[d_{ij}]$ of step 1 of Heuristic-3

Job *i*, Alt *a*	1.1	2.1	3.X	4.1	5.1	6.X
1.1	∞	2	3	94	30	13
2.1	3	∞	57	23	21	61
3.X	48	27	∞	41	11	27
4.1	61	37	0	∞	88	9
5.1	72	16	14	485	∞	4
6.X	22	18	17	21	15	∞

Step 2: Solve the assignment problem of the distance matrix**,** [d_{ij}]

The assignment solution of table 13 is shown on table 14 as follows.

Table 14 The assignment of minimum distance matrix of Heuristic-3

Job *i*, Alt *a*	1.1	2.1	3.X	4.1	5.1	6.X
1.1	∞	***2***	3	94	30	13
2.1	***3***	∞	57	23	21	61
3.X	48	27	∞	41	***11***	27
4.1	61	37	***0***	∞	88	9
5.1	72	16	14	485	∞	***4***
6.X	22	18	17	***21***	15	∞

The assignment solution is 1.1-2.1, 2.1-1.1, 3.X-5.1, 4.1-3.X, 5.1-6.X, and 6.X-4.1 with the total cost = 41.

Step 3: Select the appropriate alternatives from the assignment solutions of step 2

Let consider job 3.X, the assignment solution shows that the assignment is 3.X-5.1 with cost =11. The alternative 3.1 must be selected for making the solution to be feasible, but when consider assignment of 4.1-3.X with cost = 0, the alternative 3.2 must be selected. The problem is which alternative 3.1 or 3.2, should be selected. For the master matrix, the algorithm selects the alternative that provides the minimum increasing (penalty) cost. For example,

1. If select 3.1 to replace 3.X, the assignment of 4.1-3.X becomes 4.1-3.1 with the cost increasing from 0 to 33 and the assignment of 3.X-5.1 becomes 3.1-5.1 with the same cost = 11. The total increasing cost is the penalty cost = (33-0) + (0) = 33.

2. If select 3.2 to replace 3.X, the assignment of 3.X-5.1 becomes 3.2-5.1 with the cost increasing from 11 to 46 and the assignment of 4.1-3.X becomes 4.1-3.2 with the same cost = 0. The total increasing cost is the penalty cost = (46-11) + (0) = 35.

The alternative 3.1 is selected for this problem and used the same procedure for considering job 6.X.

Let consider job 6.X, the assignment solution shows that the assignment 6.X to 4 with cost =21. The alternative 6.1 must be selected for making the solution to be feasible, but when consider assignment 5.1-6.X with cost = 4, the alternative 6.1 must be selected also. Therefore, the alternative 6.1 is selected without considering the penalty cost.

Step 4: Solve the feasible assignment solution

The distance matrix is updated by using the feasible alternative in the matrix, $[d_{ij}]$. The feasible distance matrix and assignment solution is shown on table 15 as follows.

Table 15 The assignment solution of step 4 of Heuristic-3

Job *i*, Alt *a*	1.1	2.1	3.1	4.1	5.1	6.1
1.1	∞	***2***	11	94	30	13
2.1	***3***	∞	57	23	21	61
3.1	85	27	∞	41	***11***	66
4.1	61	37	***33***	∞	88	87
5.1	72	16	68	485	∞	***4***
6.1	22	43	62	***21***	44	∞

The assignment solution is 1.1-2.1, 2.1-1.1, 3.1-5.1, 4.1-3.1, 5.1-6.1, and 6.1-4.1 with cost 74 units that is the same as the 0-1 IP solution.

30 tested problems are generated to verify the quality of solutions for all heuristics. All heuristics are applied to select the alternative of all tested problems and then the assignment solution of the selected alternative is solved and compared to the IP solution from the master problem. The results are showed in the next chapter.

All heuristics for selecting alternatives discussed can provide the initial solution of the lower bound of the AGVsp-P/D. The previously procedures, can usually be classified as methods of constructive heuristics. The solutions can be improved by applying the procedure that is the improvement heuristic base on a given initial solution. The next part presents a heuristic for improving selected alternatives

from previous alternative selection heuristics. The purpose is to improve the selected alternatives that can provide that the assignment solution close to the solution from the 0-1 IP model. The alternative selection improvement heuristic is:

Step 1**:** For all rows of job *j* that have alternatives, compute the summation of the cost in each row

Step 2**:** For all columns of job *j* that have alternative, compute the summation of the cost in each column

Step 3**:** Compute the summation value of each alternative job and label the selected solution from any alternative selected heuristics **(**Heuristic-1 or 2 or 3 can be used)

Step 4: Select one of the un-label alternatives that has the minimum value of summation label the new selection, change the selected alternative to the new selection and solve the assignment problem of the distance matrix with new selected alternatives

Step 5: If the assignment solution is not improved, go back to step 4, otherwise keep the improved solution, label all alternatives of this considered job and then go to back step 4. Continue until all jobs (in step 3) are labeled

Suppose the example of the master matrix for explaining the implementation of this heuristic is shown on table 16 as follows.

Table 16 The example of the alternative selection improvement heuristic

Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	71	16	56	90	47	75	97	99
2.1	6	∞	10	33	25	43	94	50	2
3.1	73	69	∞	∞	∞	67	38	1	87
3.2	66	2	∞	∞	∞	83	42	63	81
3.3	11	32	∞	∞	∞	5	92	76	37
4.1	79	54	22	94	11	∞	16	7	23
5.1	67	6	24	70	70	45	∞	0	20
6.1	97	45	56	62	60	9	86	∞	∞
6.2	89	39	95	0	54	27	67	∞	∞

The solution of this distance matrix from solving the 0-1 IP model is 54 units with selected alternative 3.1 and 6.1 and the assignment of 1.1-3.1, 2.1-1.1, 3.1-6.1, 4.1-5.1, 5.1-2.1, and 6.1-4.1. When the alternative selection Heuristic-3 is applied to this example, the alternatives 3.2 and 6.1 are selected with cost 89 units. The solution is shown as follow.

Table 17 The assignment solution of table 16 by using Heuristic-3

Job *i*, Alt *a*	1.1	2.1	3.2	4.1	5.1	6.1
1.1	∞	71	***56***	47	75	97
2.1	***6***	∞	33	43	94	50
3.2	66	***2***	∞	83	42	63
4.1	79	54	94	∞	***16***	7
5.1	67	6	70	45	∞	***0***
6.1	97	45	62	***9***	86	∞

The Heuristic-3 solution deviates from the IP solution about 64.81% that is too much. When the alternative selection improvement heuristic is applied to the generated solution from Heuristic-3, the example of implementation of the alternative selection improvement heuristic algorithm is shown step by step as follows.

Step 1**:** For all rows of job *j* that have alternatives, compute the summation of the cost in each row

Step 2**:** For all columns of job *j* that have alternative, compute the summation of the cost in each column

The detail of step 1 and step 2 are shown on table 18 as follows

Table 18 The result of implementation of step 1 and 2 of the improvement heuristic

Job *i*, Alt *a*	1.1	2.1	3.1	***3.2***	3.3	4.1	5.1	***6.1***	6.2	
1.1	-	71	16	56	90	47	75	97	99	
2.1	6	-	10	33	25	43	94	50	2	Sum.
3.1	73	69	-	-	-	67	38	1	87	335
3.2	66	2	-	-	-	83	42	63	81	337
3.3	11	32	-	-	-	5	92	76	37	253
4.1	79	54	22	94	11	-	16	7	23	
5.1	67	6	24	70	70	45	-	0	20	
6.1	97	45	56	62	60	9	86	-	-	415
6.2	89	39	95	0	54	27	67	-	-	371
		Sum.	223	315	310			294	349	

Step 3**:** Compute the summation value of each alternative job and label the selected solution from the alternative selection heuristic

3.1: (335+223) = 558

3.2: (337+315) = 652

3.3: (253+310) = 563

6.1: (415+294) = 709

6.2: (371+349) = 702

From the alternative selection Heuristic-3, job 3.2 and 6.1 are the selected from Heuristic-3, which are labeled at this step.

Step 4: Select one of the unlabel alternatives that has the minimum value of summation label the new selection, change the selected alternative to the new selection and solve the assignment problem of the distance matrix with new selected alternatives

From step 3, the alternative 3.1 has the minimum value of summation. The existing selected alternative is changed from the alternative 3.2 and 6.1 to the alternative 3.1 and 6.1 and labels the alternative 3.1. The new distance matrix and the assignment solution are shown on table 19 as follows.

Table 19 The result of step 4 of the alternative improvement heuristic

Job *i*, Alt *a*	1.1	2.1	3.1	4.1	5.1	6.1
1.1	∞	71	***16***	47	75	97
2.1	***6***	∞	10	43	94	50
3.1	73	69	∞	67	38	***1***
4.1	79	54	22	∞	***16***	7
5.1	67	***6***	24	45	∞	0
6.1	97	45	56	***9***	86	∞

The assignment solution value is 54 units

Step 5: If the assignment solution is not improved, go to step 4, otherwise keep the improved solution, label all alternatives of this considered job and then go to step 4. Continue until all jobs (in step 3) are labeled

From step 4, the solution is improved from 89 units to 54 units. This solution with alternative selection of 3.1 and 6.1 is kept. Then label all alternatives of job No.3 and continue to step 3 by considering the rest of unlabeled jobs. This example is ended at the iteration 2 with the solution of alternatives 3.1 and 6.1 that is same as the IP solution.

The same set of tested problems of alternative selection heuristics are used to verify the quality of solution for this alternative improvement heuristic. This improvement heuristic is applied to improve the initial solution from the alternative selection heuristic of tested problems. All results are shown in the next chapter.

2.5 The modified Eastman's algorithm for TSP of the AGVsp-P/D

Now the solution of the assignment problem with alternative P/D nodes by solving the 0-1 IP model, which is the lower bound of the AGVsp-P/D, provides a single tour or subtours. The goal of this part is to propose the heuristics approach to create TSP tours from the lower bound solutions.

Eastman's algorithm for solving the TSP is considered for solving a single TSP tour from the lower bound of AGVsp-P/D. Charnsethikul (1993) presented that Eastman's algorithm has some advantages over Little's algorithm. For instance, there is no difference in the level of branching or fathoming between solving the MTSP and the TSP using Little's algorithm, because the algorithm treats the MTSP same as the TSP. Eastman's algorithm has difference rules to fathom an active node, it considers whether the tour is feasible or not for the MTSP. For example, consider a problem with five nodes and two vehicles. Suppose that the solution from the assignment problem is 6-1-2-6, and 7-3-5-4-7, where 6 and 7 represent the starting point of each vehicle. This tour is feasible for the MTSP, but it is not feasible for the TSP. If the MTSP is solved by Eastman's algorithm and use its rule for solving the TSP, it has to continue branching and searching for the solution of the TSP. In fact, there is already has a feassible tour in the first step. This was illustrated by Svestka and Huckfeldt (1973) when they modified this rule to Eastman's algorithm. The results showed that solving the MTSP usually required fewer steps than solving the TSP.

To satisfy the subtour elimination constrain of the TSP with alternative P/D nodes, the heuristic techniques are applied to the problem of minimum total distance solved by the method as similar idea in the previous paragraph. The goal of the heuristic is to create the TSP tour from the solution of solving the assignment problem with alternative P/D nodes, which is the lower bound of the TSP. The procedure deletes each link (i, j) of the first found subtour from the lower bound solution, where (i, j) is a sequence of node in the first found subtour by assigning c_{ij} (cost of traveling from node i to node j) equal to infinity, the assignment problem with alternative P/D nodes corresponding to each deletion of links (i, j) is solved. Suppose there are k links (i, j) in the first subtour, thus the heuristic produces the new k solutions, then selects the best solution, which provides a single TSP tour. If it can not found the single TSP tour in this set of k solutions, continue searching in the next found subtours of the lower bound solution until the single TSP tour is found. When all subtours are searched and still exist no single TSP tour, the best improvement solution is selected to be the new lower bound solution and continued searching in the same process until the single TSP tour is found.

From the previous procedure, the algorithm can be described as follows.

Step 1: Solves the assignment problem with alternative P/D nodes (which is a TSP relaxation).

Step 2: If the solution from step 1 is a single TSP tour, stops and keeps the solution, otherwise, from the solution in step 1 or step 3, selects the first produced subtour.

Step 3: Starts to deletes each link (*i, j*), which is called de-link, in that selected subtour from step 2 and resolves the assignment problem with alternative P/D nodes corresponding to each deletion of links (*i, j*).

Step 4: If a single TSP tour is found in step 3, stops and keeps the best solution, otherwise, selects the next produced subtours from the solution in step 1 and goes back to step 3 until all produced subtours are investigated. If a TSP tour can not be found, goes to step 5.

Step 5: Select the best solution in step 3 as the current lower bound solution and goes to back step 2.

Consider the example of the distance matrix in table 8, the solution of this distance matrix from solving the 0-1 IP model is 74 units with the assignment of 1.1-2.1, 2.1-1.1, 3.1-5.1, 4.1-3.1, 5.1-6.1, and 6.1-4.1. The solution is the optimum for 2 AGVs which stat at node 1 and node 3.1, because there are 2 subtours which are 1.1-2.1-1.1 and 3.1-5.1-6.1-4.1-3.1.

For a single AGV, subtour elimination constraints will be added for creating the single TSP tour. By applying the modified Eastman's branch and bound algorithm for the TSP, the procedure is to select the first subtour, eliminates each arc in the tour, called de-link, and solves the corresponding assignment problem with alternative P/D nodes. Base on the generated lower bound solution, a first found subtour, which is

subtour 1-2-1, is selected and link 1-2 and 2-1 are de-linked, which are shown on table 20 and 22, and solved the assignment problem with alternative P/D nodes. The results are shown on table 21and 23 as follows.

Table 20 The cost table of de-link 1-2 , by assigning the cost of the link 1-2 to ∞

Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	∞	11	3	35	94	30	13	97
2.1	3	∞	57	73	86	23	21	61	83
3.1	85	27	∞	∞	∞	41	11	66	27
3.2	48	57	∞	∞	∞	52	46	73	52
3.3	80	66	∞	∞	∞	58	79	63	28
4.1	61	37	33	0	56	∞	88	87	9
5.1	72	16	68	14	20	485	∞	4	70
6.1	22	43	62	17	88	21	44	∞	∞
6.2	96	18	86	60	34	42	15	∞	∞

Table 21 The assignment solution of table 20

Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2	$z_{(i)}$
1.1	∞	0	1	0	0	0	0	0	0	1
2.1	1	∞	0	0	0	0	0	0	0	1
3.1	0	0	∞	∞	∞	0	1	0	0	1
3.2	0	0	∞	∞	∞	0	0	0	0	0
3.3	0	0	∞	∞	∞	0	0	0	0	0
4.1	0	1	0	0	0	∞	0	0	0	1
5.1	0	0	0	0	0	0	∞	1	0	1
6.1	0	0	0	0	0	1	0	∞	∞	1
6.2	0	0	0	0	0	0	0	∞	∞	0
$z_{(j)}$	1	1	1	0	0	1	1	1	0	

The assignment solution of table 20 is 1.1- 3.1- 5.1 - 6.1 - 4.1 - 2.1 - 1.1 with total distance of 87 units. It implies the TSP tour which is stated and ended at node 1.

Table 22 The cost table of de-link 2-1, by assigning the cost of the link 2-1 to ∞

Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	2	11	3	35	94	30	13	97
2.1	∞	∞	57	73	86	23	21	61	83
3.1	85	27	∞	∞	∞	41	11	66	27
3.2	48	57	∞	∞	∞	52	46	73	52
3.3	80	66	∞	∞	∞	58	79	63	28
4.1	61	37	33	0	56	∞	88	87	9
5.1	72	16	68	14	20	485	∞	4	70
6.1	22	43	62	17	88	21	44	∞	∞
6.2	96	18	86	60	34	42	15	∞	∞

Table 23 The The assignment solution of table 22

Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2	$z_{(i)}$
1.1	∞	1	0	0	0	0	0	0	0	1
2.1	0	∞	0	0	0	1	0	0	0	1
3.1	0	0	∞	∞	∞	0	1	0	0	1
3.2	0	0	∞	∞	∞	0	0	0	0	0
3.3	0	0	∞	∞	∞	0	0	0	0	0
4.1	0	0	1	0	0	∞	0	0	0	1
5.1	0	0	0	0	0	0	∞	1	0	1
6.1	1	0	0	0	0	0	0	∞	∞	1
6.2	0	0	0	0	0	0	0	∞	∞	0
$z_{(j)}$	1	1	1	0	0	1	1	1	0	

The assignment solution of table 22 is 1.1- 2.1 - 4.1 - 3.1 - 5.1- 6.1 - 1.1 with the total distance of 95 units. This solution implies the single TSP tour which stat at node 1, but this solution is not better than the previous one. This heuristic would appear to require a lot of memory and time of computation for solving the sequence of assignment subproblems with alternative P/D nodes which is 0-1 IP. In fact, the algorithm starts searching the first produced subtour and then goes to the next subtours for saving running time, instead of searching from all found links (*i, j*) from the solution in step 1, which may provides the better solution but takes much more time. The results of some simulated problems are presented in the next chapter.

3. The algorithms for solving the multi AGVsp-P/D

The previous section provides the procedure for solving the case of single AGVsp-P/D. The results from applying the procedure to solve the distance matrix are sets of the single TSP tour. When the case of multi AGVsp-P/D is considered, a specific number of AGVs is given to the problem. How can all vehicles will be utilized is considered. The solutions of multi AGVsp-P/D are the sets of multi TSP tours. The procedure of solving the MTSP from the existing TSP solutions will be applied to form the sets of TSP tours from the single TSP tour. The research presents two heuristic procedures as follows.

3.1 The heuristic of splitting a TSP tour for solving the lower bound of multi AGVsp-P/D

Let assume that a regular AGV has speed equal to 1 meter/minute. If the problem defines a special AGV that has speed equal to *M* meters /minute, it can accomplish the same job (same total distance) faster than a regular AGV by the normal mission time divided by *M* or can travels more distance by using the same amount of time. Refer to table 8, let this table is the distance matrix of the regular AGV, which is used for explaining the example of multi AGVsp-P/D as follows.

Table 24 The example of the regular AGV distance matrix for multi AGVsp-P/D

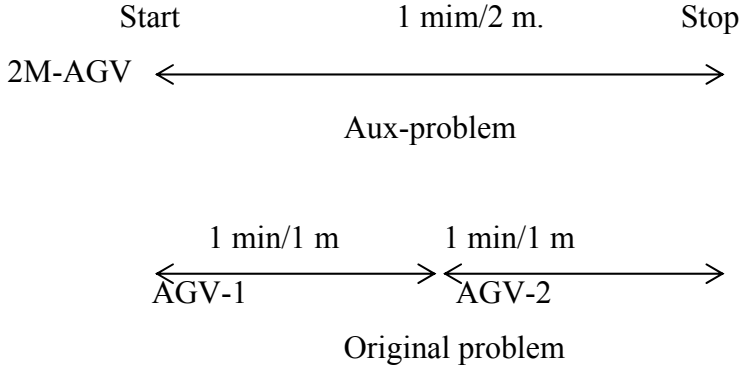
Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	2	11	3	35	94	30	13	97
2.1	3	∞	57	73	86	23	21	61	83
3.1	85	27	∞	∞	∞	41	11	66	27
3.2	48	57	∞	∞	∞	52	46	73	52
3.3	80	66	∞	∞	∞	58	79	63	28
4.1	61	37	33	0	56	∞	88	87	9
5.1	72	16	68	14	20	485	∞	4	70
6.1	22	43	62	17	88	21	44	∞	∞
6.2	96	18	86	60	34	42	15	∞	∞

The table shows that the regular AGV can travel from the starting point of job 1.1 to the starting point of job 2.1 by the total distance of 2 meters. From the assumption of the regular AGV speed, that it takes 2 minutes. Let define that the auxiliary problem, called Aux-problem, is the problem of the special AGV that the traveling time from node *i* to node *j* equal to the original distance matrix divided by *M*. The auxiliary cost matrix is used with the special AGV of *M* times faster. The auxiliary cost matrix of 2 times faster AGVs (*M* = 2), called 2*M*-AGV, is shown on table 25 as follows.

Table 25 The cost matrix of Aux-problem of 2 times faster AGVs

Job *i*, Alt *a*	1.1	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	1	5.5	1.5	17.5	47	15	6.5	48.5
2.1	1.5	∞	28.5	36.5	43	11.5	10.5	30.5	41.5
3.1	42.5	13.5	∞	∞	∞	20.5	5.5	33	13.5
3.2	24	28.5	∞	∞	∞	26	23	36.5	26
3.3	40	33	∞	∞	∞	29	39.5	31.5	14
4.1	30.5	18.5	16.5	0	28	∞	44	43.5	4.5
5.1	36	8	34	7	10	242.5	∞	2	35
6.1	11	21.5	31	8.5	44	10.5	22	∞	∞
6.2	48	9	43	30	17	21	7.5	∞	∞

The Aux-problem considers the traveling time (because the distance of each pair of jobs is same as the regular AGV), not the distance. It can see that from the starting point of job1.1 to the starting point of job2.1 takes 1 minute by 2*M*-AGV. It means that this is the lower bound of the mission time for 2 regular AGVs, which are AGV-1 and AGV-2. If the mission time of the Aux- problem is splitted into two parts, the explanation is shown as follows.



From the above chart, let assume that AGV-1 and AGV-2 can be started everywhere and every time. Therefore, the relationship between the original problem and the Aux-problem is:

2M-AGV time ≤ AGV-1 time + AGV-2 time

For a given, the optimal TSP tour from the starting depot to all jobs and back to depots of AGV-1 and AGV-2, both of tour distances can be added together and get the optimal total distance. For explaining, the relationship of AGV-1, AGV-2 and 2*M*-AGV is examined as follows.

AGV-1 X m. tour 1-2-3-4-5-1= X meters / X min

X min.

AGV-2 Y m. tour 1-6-7-8-9-1=Y meters / Y min

Y min

2M-AGV X+Y m.

(X+Y)/2 min

Tour 1-2-3-4-5-1-6-7-8-9-1 = X+Y meters / (X+Y)/2 min

Let a tour 1-2-3-4-5-6-7-8-9-1 of the Aux-problem is the optimal TSP tour. The distance of X+Y (1-2-3-4-5-1-6-7-8-9-1) is ≥ the Aux-problem TSP tour distance of 1-2-3-4-5-6-7-8-9-1. Assume that distance of X+Y is the optimal solution of the original problem so that it can claim that the optimal tour of Aux-problem is the lower bound of the original problem with 2 AGVs. If the TSP tour of the Aux-problem can be solved, it can get the lower bound of problems with *M* number of AGVs for each AGV by splitting the Aux-problem tour into *M* parts. Therefore, the original distance value of *M* multiplied by Aux-tour distance is the lower bound distance of each regular AGV.

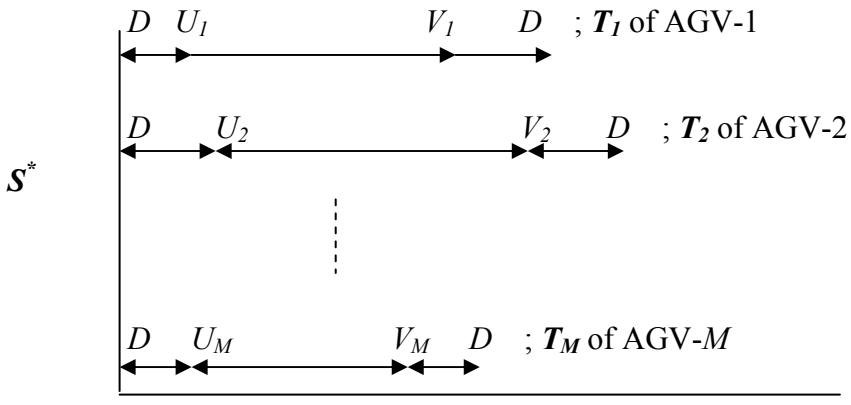
According to this point, solutions of a single TSP tour from the original problem man provide *M* balance subtours, which is the lower bound of multi AGVsp-P/D by splitting the optimal TSP tour distance of the Aux-problem into *M* parts, (*M* subtours) and converting to the original distance of all subtours. The proof is shown as follows.

Lemma: Let construct the Aux-problem of a single AGV where the Aux-problem distance matrix $[C'_{ij}] = (1/M) \times C_{ij}$; C_{ij} is the original distance matrix. If $\boldsymbol{T}^*_{AUX}$ is the optimal TSP tour for the single AGV of the Aux-problem, the length of the optimal tour $\boldsymbol{T}^*_{AUX}$, $l(\boldsymbol{T}^*_{AUX})$, is a lower bound of the length of the optimal tour $\boldsymbol{T}^*$, $l(\boldsymbol{T}^*)$, of the original problem with *M* number of AGVs

That it is; $$M \times l(\boldsymbol{T}^*_{AUX}) \leq l(\boldsymbol{T}^*)$$

Proof: Given an optimal tour, $\boldsymbol{T}^*$, of *M* AGVs with the original distance matrix by assuming that all AGVs are started at the starting depot and can start at the same time.

Let set $\boldsymbol{S}^*$ is a set of subtour of the optimal tour $\boldsymbol{T}^*$ which consists of subtours $\boldsymbol{T_i} = \{\boldsymbol{T_1}, \boldsymbol{T_2}, \boldsymbol{T_3}, \ldots \boldsymbol{T_M}\}$ for each AGV. Set $\boldsymbol{S}^*$ can form the tours as the following diagram.



Set $\boldsymbol{S}^*$ of subtours of the optimal tour $\boldsymbol{T}^*$

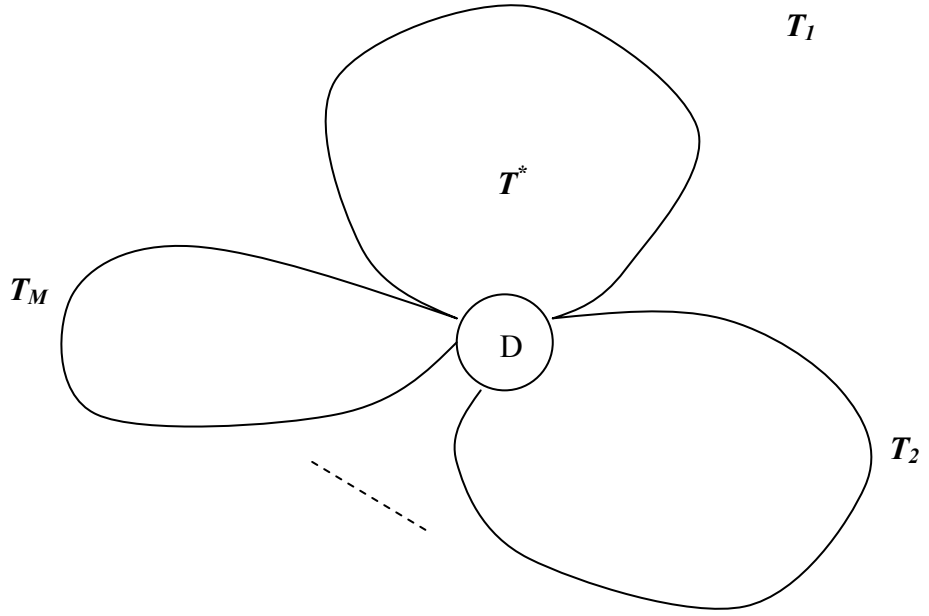


Figure 4 Subtours T_i and tour T^*

$$T^* = T_1 + T_2 + T_3 + \ldots + T_M$$

$$l(T^*) = \sum_{i=1}^{M} l(T_i)$$

Consider any subtours T_i for all $i = 1, 2, \ldots, M$, all of them have some common jobs which are travel from last node (job) of subtours T_i to the starting depot, D, and travel from the starting depot to the first job of subtours T_{i+1}. For example, tour T_1 consists of the travel distance from the starting depot D to job U_1 and go to the next job, follows the optimal sequence, until finishing the last job V_1 of this subtour and then travel from V_1 back to the starting depot D.

Let $U_1, U_2, \ldots, U_M$ be the first nodes after the starting node D and $V_1, V_2, \ldots, V_M$ be the last nodes before the starting node D. From Set S^* of subtours of the optimal tour T^*, let construct a single tour T'' by:

1. Disconnect the arcs from the last node, V_i to the starting node D of all subtours T_i for all i,
2. Disconnect the arcs from the starting node D to the first node, U_i of all

subtours T_i for all i, except for the arc from the starting node D to node U_1 and from V_M to the starting node depot D , and then

3. Connect the node V_i to node U_{i+1} for all subtours T_i for for all i.

A single tout T'' is constructed which is a single AGV starts at depot, travels only one time to the first node U_1, continuous travels along the optimal sequence to the last node V_M , and travels back to the starting node D only one time.

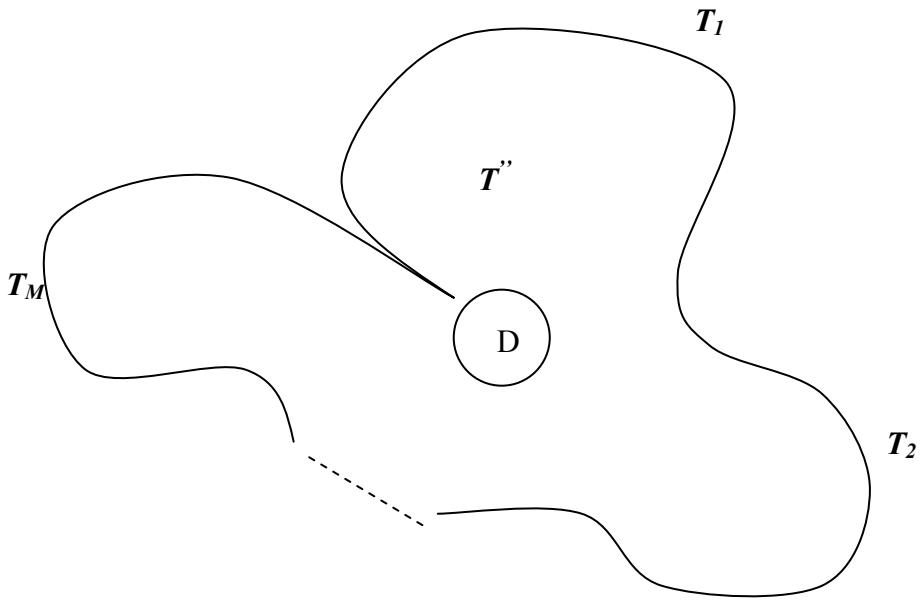


Figure 5 The TSP tour ***T"***

Let the length between node i and node j is $l(i, j)$. From the property of triangle inequality,

$$l(V_1, D) + l(D, U_2) \geq l(V_1, U_2)$$

It can be derived that:

$$l(\boldsymbol{T}'') = \sum_{i=1}^{M} l(T_i) - [l(\boldsymbol{V_1}, \boldsymbol{D}) + l(\boldsymbol{D}, \boldsymbol{U_2})] + l(\boldsymbol{V_1}, \boldsymbol{U_2})$$

$$- [l(\boldsymbol{V_2}, \boldsymbol{D}) + l(\boldsymbol{D}, \boldsymbol{U_3})] + l(\boldsymbol{V_2}, \boldsymbol{U_3})$$

$$\vdots$$

$$- [l(\boldsymbol{V_{M-1}}, \boldsymbol{D}) + l(\boldsymbol{D}, \boldsymbol{U_M})] + l(\boldsymbol{V_{M-1}}, \boldsymbol{U_M})$$

Because of $-l(\boldsymbol{V_1}, \boldsymbol{D}) - l(\boldsymbol{D}, \boldsymbol{U_2}) + l(\boldsymbol{V_1}, \boldsymbol{U_2}) \leq 0$, so that

$$l(\boldsymbol{T}'') \leq l(\boldsymbol{T}^*)$$

Now procedure claim that $M \times l(\boldsymbol{T}^*_{\boldsymbol{AUX}}) \leq l(\boldsymbol{T}'')$ include every tours $\boldsymbol{T_i}$ of the original problem that has length $l(\boldsymbol{T}^*)/M$ in the Aux-problem. Since $l(\boldsymbol{T}^*_{\boldsymbol{AUX}})$ is the optimal length of tour of the Aux-problem, it can conclude that:

$$l(\boldsymbol{T}^*_{\boldsymbol{AUX}}) \leq l(\boldsymbol{T}'')/M$$

Therefore,

$$l(\boldsymbol{T}^*_{\boldsymbol{AUX}}) \leq l(\boldsymbol{T}^*)/M$$

$$M \times l(\boldsymbol{T}^*_{\boldsymbol{AUX}}) \leq l(\boldsymbol{T}^*)$$

This heuristic can be used to split a single TSP tour to multi tours. Solutions of the lower bound of multi AGVsp-P/D are the sets of multi tours, not the sets of multi TSP tours or MTSP solution. The assumption of this algorithm is that vehicles can be started and ended everywhere. Therefore, this heuristic can not satisfy the objective of the multi AGVsp-P/D, which same as the objective of the MTSP, but can be used for solving the lower bound of the multi AGVsp-P/D by splitting a single TSP tour to a set of *M* routes for *M* AGVs. A set of *M* routes from splitting a single TSP tour to *M* parts can be form a set of *M* tours, if the distance from the starting depot to the starting job and the ending job to the starting depot of each route is added. Actually, the goal is attempting to form the algorithm which can support the assumption of the MTSP, which is shown on the next part. The experiment on the next chapter is performed to

compare the solution of the algorithm of splitting a single TSP and the algorithm of solving the MTSP as the standard TSP

3.2 The algorithm of solving the MTSP as the standard TSP

Refer to Svestka and Huckfeldt (1973), if 2 AGVs are given for the multi AGVsp-P/D, the problem on table 8 can be modified for using the algorithm of solving the MTSP as the standard TSP. The new distances matrix $[d_{ij}]$ are created from the original distances matrix $[c_{ij}]$, which is shown on table 26 as follows

Table 26 The MTSP distances matrix $[d_{ij}]$

	1.1	1.2	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	∞	2	11	3	35	94	30	13	97
1.2	∞	∞	2	11	3	35	94	30	13	97
2.1	3	3	∞	57	73	86	23	21	61	83
3.1	85	85	27	∞	∞	∞	41	11	66	27
3.2	48	48	57	∞	∞	∞	52	46	73	52
3.3	80	80	66	∞	∞	∞	58	79	63	28
4.1	61	61	37	33	0	56	∞	88	87	9
5.1	72	72	16	68	14	20	485	∞	4	70
6.1	22	22	43	62	17	88	21	44	∞	∞
6.2	96	96	18	86	60	34	42	15	∞	∞

Table 27 The AGVsp-P/D solution of table 26

	1.1	1.2	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	∞	0	0	0	0	0	1	0	0
1.2	∞	∞	1	0	0	0	0	0	0	0
2.1	1	0	∞	0	0	0	0	0	0	0
3.1	0	0	0	∞	∞	∞	0	0	0	0
3.2	0	1	0	∞	∞	∞	0	0	0	0
3.3	0	0	0	∞	∞	∞	0	0	0	0
4.1	0	0	0	0	1	0	∞	0	0	0
5.1	0	0	0	0	0	0	0	∞	1	0
6.1	0	0	0	0	0	0	1	0	∞	∞
6.2	0	0	0	0	0	0	0	0	∞	∞

Table 28 The corresponding cost from the solution of table 27

	1.1	1.2	2.1	3.1	3.2	3.3	4.1	5.1	6.1	6.2
1.1	∞	∞	0	0	0	0	0	30	0	0
1.2	∞	∞	2	0	0	0	0	0	0	0
2.1	3	0	∞	0	0	0	0	0	0	0
3.1	0	0	0	∞	∞	∞	0	0	0	0
3.2	0	48	0	∞	∞	∞	0	0	0	0
3.3	0	0	0	∞	∞	∞	0	0	0	0
4.1	0	0	0	0	0	0	∞	0	0	0
5.1	0	0	0	0	0	0	0	∞	4	0
6.1	0	0	0	0	0	0	21	0	∞	∞
6.2	0	0	0	0	0	0	0	0	∞	∞

Table 26 shows that there is one additional first row and column, which is row and column No. 1.2. They represent the dummy starting node for the problem of 2 AGVs. The solution of AGVsp-P/D of table 26 is shown on table 27 with its corresponding cost, which is shown on table 28.

From table 27, the assignment solution is 1.1 - 5 - 6.1 - 4 - 3.2 - 1.2 - 2 - 1.1 with the distance of 108 units. If the assignment solution is not a single TSP tour, the modified Eastman's algorithm for the TSP of the AGVsp-P/D is applied. By the

solution from table 27, it shows that the assignment solution consists of 2 TSP tours for 2 AGVs which are:

the TSP tour 1 for AGV-1 that is: 1.1 (depot) - 5.1 - 6.1 - 4.1 - 3.2 - 1.2 (depot) and the TSP tour 2 for AGV-2that is: 1.2 (depot) - 2.1 - 1.1 (depot).

This algorithm provides the solution of the multi AGVsp-P/D by solving the single TSP tour of the modified distance matrix [d_{ij}]. The running time of solving the TSP and the MTSP may not different significantly when the number of AGVs (M) increasing, because in the real world problems of n nodes, it may have the constraints of the cost of increasing the number of salesman or vehicle. This algorithm is tested for the implementation with many levels of problem sizes that are shown in the next chapter.

All presented methods of this research can provide the results that satisfy the research objectives. The single/multi AGVsp-P/D can be solved with some levels of problem sizes. The results of all experiments are explained in the next chapter.

RESULTS AND DISCUSSIONS

This chapter presents the results of all experiments of this research, which includes the result analysis, conclusions and discussions.

Results

The detailed results of all tested problems are displayed in the form of tables and graphs. Tested problems of the AGVsp-P/D are generated randomly as similar as the example on table 6. The formulated mathematical model of AGVsp-P/D is programed using MATLAB 7.0 for solving lower bound solutions, single TSP tours, and multi TSP tours of the single/multi AGVsp-P/D.

1. The results of solving the lower bound of the AGVsp-P/D by integer linear programming

The simulated problems of the AGVsp-P/D with 10, 20, 30, 40, and 50 nodes are generated randomly with some numbers of 2 alternative jobs and some numbers of regular jobs. The running times of solving the lower bound solution is examined and compared to the regular assignment problem of the same problem size. The simulated problems are generated, which are:

1. 10 nodes with 5 jobs of 2 alternatives
2. 20 nodes with 5 jobs of 2 alternatives and 10 regular jobs
3. 30 nodes with 5 jobs of 2 alternatives and 20 regular jobs
4. 40 nodes with 5 jobs of 2 alternatives and 30 regular jobs
5. 50 nodes with 5 jobs of 2 alternatives and 40 regular jobs

that all for them are in a set of problems called 2Al-5 of n nodes (n = 10, 20, 30, 40 and 50). The other sets of simulated problems are:

1. 10 nodes with 5 jobs of 2 alternatives, called 2Al-5
2. 20 nodes with 10 jobs of 2 alternatives, called 2Al-10
3. 30 nodes with 15 jobs of 2 alternatives, called 2Al-15
4. 40 nodes with 20 jobs of 2 alternatives, called 2Al-20
5. 50 nodes with 25 jobs of 2 alternatives, called 2Al-25

that all of them are in a set of problem, called 2Al-Max. The running time of 40 replications for each of the level of problem size are compared with the regular assignment problems with 10, 20, 30, 40, and 50 jobs, which are solved using the lower bound model with alternative P/D nodes.

According to this experiment, the main purpose is to examine that whether the increasing of number of alternative jobs, from 2Al-5 to 2Al-Max, affects on the average running time for solving the problem or not. The research assumes that the type I error, α is 0.05. The hypothesis test will be examined after the experiments done. The results of all experiments of this section using MATLAB 7.0 are shown as follows.

Table 29 The running time in seconds of simulated problems of 2Al-5

Problem No.	10node	20node	30node	40node	50node
1	0.237	0.920	2.188	5.375	12.437
2	0.253	0.527	2.887	6.286	15.814
3	1.073	0.576	5.770	6.535	14.133
4	0.383	0.605	1.900	5.567	11.435
5	0.420	0.618	1.930	4.850	12.047
6	0.391	0.599	1.711	5.333	16.538
7	0.436	0.551	3.743	5.025	16.167
8	0.396	0.531	2.101	5.243	12.177
9	0.737	1.292	1.752	8.186	16.165
10	0.282	0.910	3.618	4.745	11.653
11	0.224	0.615	2.014	4.731	11.587
12	0.511	0.598	1.919	4.845	11.054
13	0.546	0.997	1.798	5.535	12.677
14	0.701	0.584	3.584	5.097	25.688
15	0.231	0.574	2.930	4.584	10.257
16	0.340	0.635	1.890	5.754	22.572
17	0.217	0.560	3.757	8.197	12.422
18	0.952	0.543	1.810	5.010	10.695
19	0.558	1.693	1.522	4.68	13.052
20	0.615	0.994	1.712	4.397	10.861
21	0.286	0.546	2.014	4.961	12.478
22	0.219	0.691	1.921	5.064	37.135
23	0.175	0.976	1.960	4.912	11.232
24	0.797	0.622	1.982	4.980	11.485
25	0.257	0.513	1.860	4.760	11.323
26	0.662	0.633	1.804	5.06	12.594
27	0.456	0.648	1.701	5.659	11.331
28	0.230	0.575	1.709	5.266	11.463
29	0.401	0.548	1.723	4.806	11.872
30	1.009	0.938	3.833	9.179	10.668
31	0.269	0.617	1.924	5.584	14.824
32	1.000	0.619	2.126	4.993	13.29
33	0.204	0.565	2.017	5.161	10.866
34	0.256	0.941	2.695	5.546	11.880
35	0.293	0.662	2.114	4.855	10.570
36	0.767	0.582	3.601	4.328	11.755
37	1.320	0.575	3.692	6.780	11.946
38	0.284	0.638	1.883	5.005	17.767
39	0.209	0.619	2.031	4.477	15.268
40	0.377	0.555	1.644	4.475	11.330

Table 30 The running time in seconds of simulated problems of 2Al-Max

Problem No.	10node (2Al-5)	20node (2Al-10)	30node (2Al-15)	40node (2Al-20)	50node (2Al-25)
1	0.237	1.500	4.293	29.315	59.497
2	0.253	1.065	6.390	19.752	13.864
3	1.073	2.014	9.446	36.411	31.589
4	0.383	1.944	4.504	36.797	91.136
5	0.420	2.376	9.662	28.656	22.187
6	0.391	2.968	11.68	37.923	49.681
7	0.436	0.756	13.357	43.830	115.214
8	0.396	3.570	13.681	43.389	12.944
9	0.737	3.641	3.714	7.674	27.961
10	0.282	1.590	3.274	12.666	59.073
11	0.224	0.988	2.236	14.049	13.920
12	0.511	0.662	6.230	6.591	31.656
13	0.546	0.637	5.866	7.332	91.336
14	0.701	2.794	7.680	21.991	22.085
15	0.231	1.904	6.577	22.744	49.614
16	0.340	2.329	3.328	12.535	115.064
17	0.217	2.423	3.827	11.762	12.787
18	0.952	1.152	5.813	12.317	70.256
19	0.558	1.015	3.427	8.901	21.932
20	0.615	3.729	7.756	4.899	37.877
21	0.286	2.248	2.926	26.677	16.208
22	0.219	0.743	8.94	89.939	74.394
23	0.175	0.584	2.934	9.427	26.248
24	0.797	3.570	9.691	8.390	32.616
25	0.257	1.246	2.077	37.673	39.466
26	0.662	2.793	3.898	8.304	29.349
27	0.456	2.446	6.359	57.296	78.340
28	0.230	1.424	6.765	16.613	67.905
29	0.401	3.164	7.865	84.195	11.305
30	1.009	2.868	14.005	41.498	71.857
31	0.269	1.430	6.913	64.475	37.215
32	1.000	1.614	2.805	32.793	80.843
33	0.204	3.288	3.748	16.388	66.105
34	0.256	1.410	6.405	25.121	46.362
35	0.293	1.168	5.972	25.765	64.666
36	0.767	1.473	9.716	10.831	23.741
37	1.320	3.505	9.225	32.688	77.782
38	0.284	2.327	4.920	5.297	63.787
39	0.209	3.745	9.378	26.874	89.043
40	0.377	0.741	7.505	7.965	57.916

Table 31 The running time in second of the regular assignment problems

Problem No.	10node	20node	30node	40node	50node
1	0.1859	0.672	2.297	5.976	10.600
2	0.174	0.558	2.373	5.183	12.343
3	0.244	0.738	2.203	5.437	10.852
4	0.203	0.627	2.384	5.555	14.656
5	0.168	0.587	2.07	5.863	12.167
6	0.242	0.726	2.076	4.932	12.587
7	0.224	0.691	2.038	5.863	11.106
8	0.230	0.718	2.318	5.660	14.293
9	0.197	0.690	2.016	5.212	12.923
10	0.199	0.674	2.028	4.678	14.413
11	0.195	0.714	2.031	5.782	13.157
12	0.256	0.772	2.331	5.250	11.171
13	0.165	0.640	1.972	5.795	12.326
14	0.174	0.721	2.024	5.833	13.488
15	0.235	0.642	2.104	5.289	12.356
16	0.161	0.640	2.138	5.362	13.078
17	0.191	0.714	1.963	5.309	12.426
18	0.274	0.658	2.371	5.972	11.694
19	0.282	0.756	2.068	5.749	12.767
20	0.254	0.716	2.284	5.607	11.204
21	0.224	0.669	2.074	5.572	12.924
22	0.203	0.663	2.309	5.446	12.273
23	0.236	0.628	2.007	5.914	11.959
24	0.161	0.599	2.107	5.001	12.547
25	0.249	0.778	2.134	5.438	12.241
26	0.205	0.666	2.244	5.755	12.624
27	0.216	0.695	2.253	5.109	11.317
28	0.258	0.673	2.260	5.643	12.981
29	0.237	0.724	2.059	4.989	13.321
30	0.194	0.702	2.302	5.146	10.499
31	0.245	0.710	1.991	5.466	13.068
32	0.218	0.678	2.054	5.249	12.833
33	0.215	0.688	2.655	5.355	12.136
34	0.217	0.690	2.052	5.440	11.000
35	0.250	0.633	2.008	5.284	13.150
36	0.254	0.68	2.049	5.347	13.127
37	0.247	0.677	2.315	5.865	11.848
38	0.210	0.660	2.205	5.531	12.891
39	0.231	0.669	2.181	5.902	13.032
40	0.224	0.705	2.268	5.396	11.170

Table 32 The summarized running time in seconds of solving the lower bound of AGVsp-P/D

Size Levels		Assignment Problems	2Al-5 Problems	2Al-Max Problems
10 nodes	Mean	0.2186975	0.47435	0.47435
	S.D.	0.031660438	0.287858633	0.287858633
	Min	0.161	0.175	0.175
	Max	0.282	1.320	1.320
20 nodes	Mean	0.681025	0.699625	2.0211
	S.D.	0.04663084	0.238417613	1.009444549
	Min	0.558	0.513	0.584
	Max	0.778	1.693	3.745
30 nodes	Mean	2.1654	2.36925	6.6197
	S.D.	0.152020545	0.906358924	3.189694184
	Min	1.963	1.522	2.077
	Max	2.655	5.770	14.005
40 nodes	Mean	5.478875	5.39565	26.193575
	S.D.	0.318063144	1.05462081	20.3977164
	Min	4.678	4.328	4.899
	Max	5.976	9.179	89.939
50 nodes	Mean	12.4137	13.7627	50.120525
	S.D.	0.999490388	4.941659133	28.89604306
	Min	10.499	10.257	11.305
	Max	14.656	37.135	115.214

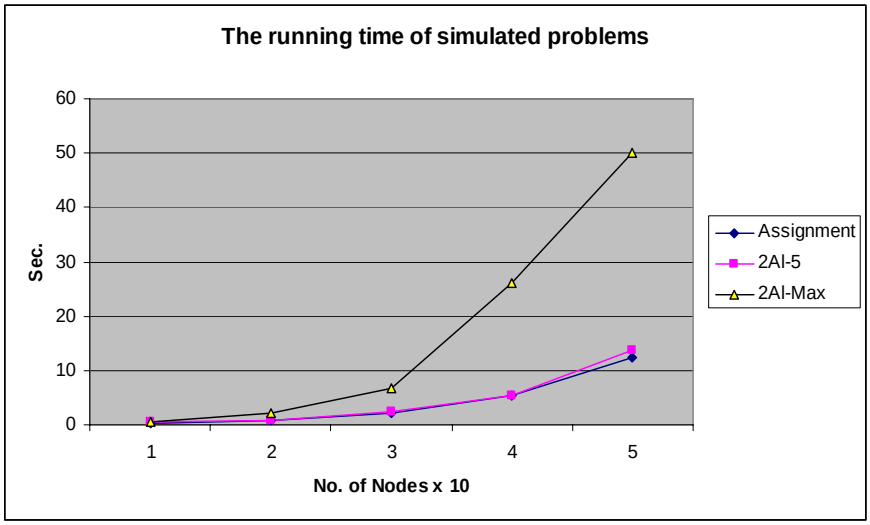


Figure 6 The graph of running time in second for obtaining the lower bound of AGVsp-P/D

Let consider the data set of tables 29, 30 and 31 of 50 nodes, the normal probability plot results using Minitab are performed sequentially as follows.

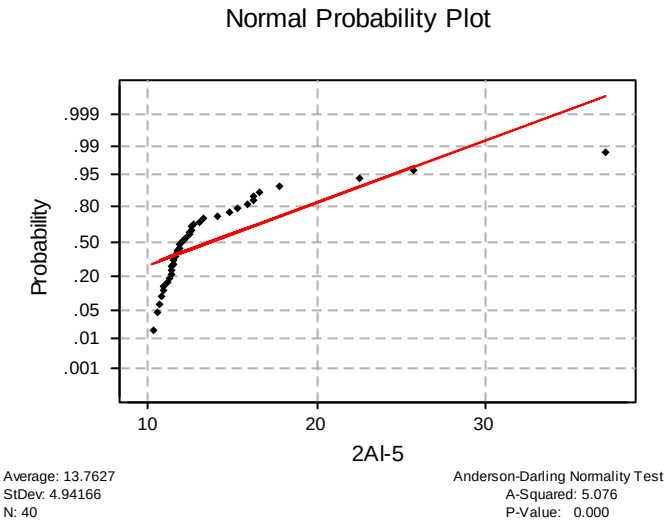


Figure 7 The normal probability plot of 2Al-5 data with 50 nodes from table 29

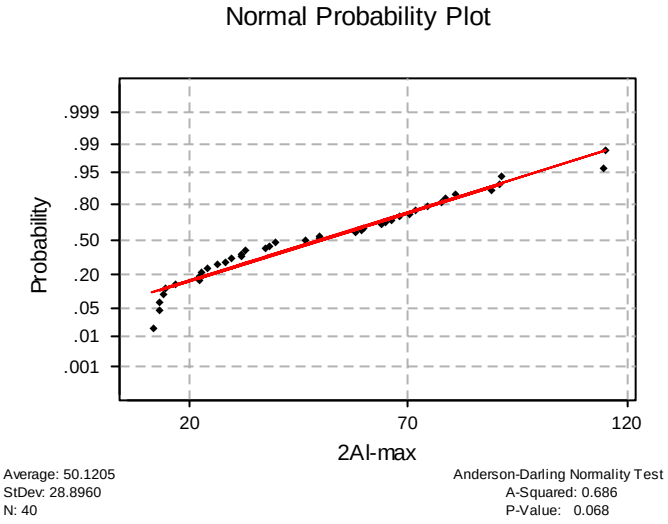


Figure 8 The normal probability plot of the 2Al-Max data with 50 nodes from table 30

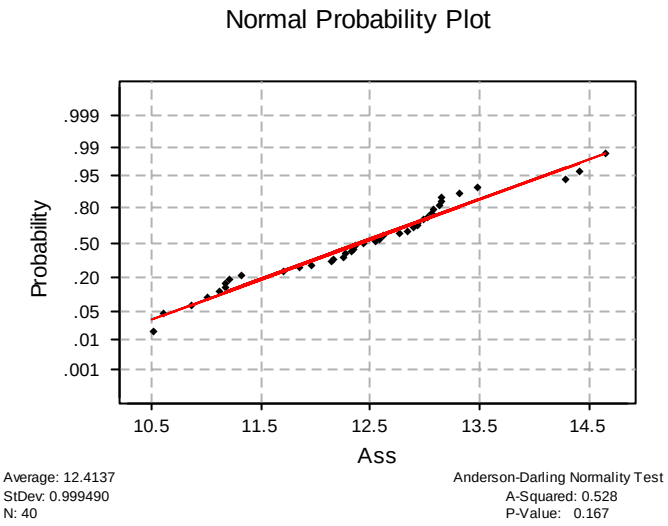


Figure 9 The normal probability plot of the regular assignment data with 50 nodes from table 31

The results of the normal probability plot show that the data set of 2Al-5 of 50 nodes, figure 7, is not statistically normal distribution, because the P-value less than 0.05. Before performing any statistical analysis, the data should be transformed or adjusted to be the normal distributed data set. The Box-Cox transformation function in Minitab is used to adjust and transform the data set from the experimental results for forming the normal probability data set. The data set from the figure 7 is transformed using the Box-Cox transformation function and tested the normality by performing the normal probability plot of the transformed data set. The result of the normality test of the transformed data set of the figure 7 is shown on figure 10 as follows.

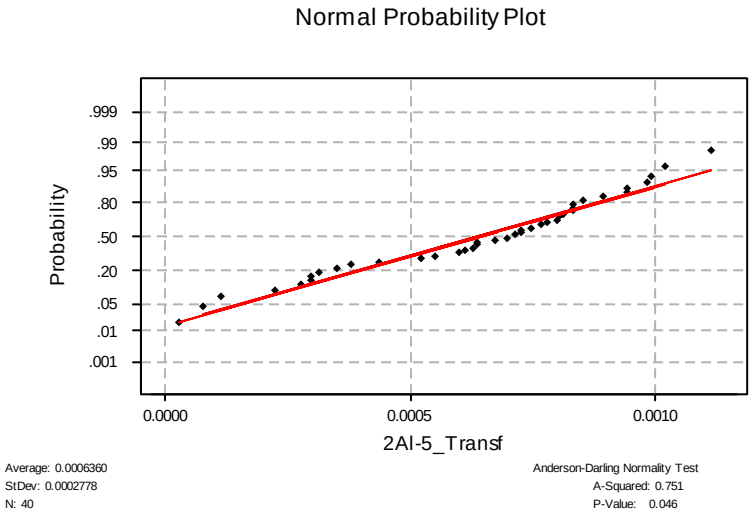


Figure 10 The normal probability plot of the Box-Cox transformation data of 2Al-5 with 50 nodes of table 29

The normal probability plot on figure 10 shows that the transformed data set of 2Al-5 with 50 nodes is still not the normal probability distribution, because the P-value is still less than 0.05. Therefore, the data of table 29, 30 and 31 will be analyzed by using nonparametric statistics. The analysis can be performed using the Kruskal-Wallis test. This test offers a nonparametric statistic of the one-way analysis of variance.

According to the experimental results, because the effect of increasing on the number of alternative jobs with a specific size level of problem is considered, research wish to test the hypothesis that the mean values of running times of assignment ($\mu_{\text{Assignment}}$), 2Al-5 ($\mu_{\text{2AL-5}}$), and 2Al-Max ($\mu_{\text{2AL-Max}}$) experiments are equal or not. Let consider the size level of 50 nodes experiments with 40 replications and type I error of $\alpha = 0.05$, the Kruskal-Wallis hypotheses can be formally stated as:

H_0: $\mu_{\text{Assignment}} = \mu_{\text{2AL-5}} = \mu_{\text{2AL-Max}}$ $(\mu_1 = \mu_2 = \mu_3)$

H_1: $\mu_i \neq \mu_j$ for at least one pair of all i, j, where $i, j = 1, 2$ and 3

The Kruskal-Wallis test is used for analyzing data sets on tables 28, 29 and 30 for 3 treatments of assignment (Ass), 2Al-5, and 2Al-Max of 50 nodes ($k = 3$) and 40 replications ($n_i = 40$ and $N = 120$). The result from Minitab is shown as follows.

Kruskal-Wallis Test: C6 versus C7

```
Kruskal-Wallis Test on C6

C7          N    Median    Ave Rank         Z
2Al-five   40     12.00        42.3     -4.05
2Al-max    40     47.99        96.3      7.97
Ass        40     12.49        42.9     -3.92
Overall   120                  60.5

H = 63.56  DF = 2  P = 0.000
```

Figure 11 The Kruskal-Wallis test of the data set on Ass, 2Al-5, and 2Al-Max of 50 nodes

If the type I error of $\alpha = 0.05$, the result from figure 11 obtains that the P-value of the Kruskal-Wallis test of data sets on Ass, 2Al-5, and 2Al-Max of 50 nodes equal to 0.000, which less than 0.05. Therefore, the hypothesis H_0: $\mu_{\text{Assignment}} = \mu_{\text{2AL-5}} = \mu_{\text{2AL-Max}}$ can be rejected and can conclude base on the inference statistics that the increasing on the number of alternative jobs with a specific size level of problem affects on the average running time.

2. The results of solving the lower bound by the alternative selection heuristics

This part attempts to test all heuristics for selecting the alternative nodes. The 30 tested problems are generated randomly to verify the quality of solution for all 3 heuristics (Heu-*i* Sol. for all $i = 1, 2$ and 3) by consider the deviation (Dev) of the heuristic solutions from the IP solutions in a form of the percent deviations (% Dev). The tested problems are the distance matrix, which has the same format as the example on table 6. The assignment problem with alternative P/D nodes can by solved by selecting the appropriate alternative jobs (Alt. Sel.) for job No. 3, and job No. 6 first and then solve the regular assignment problem. The research tries to evaluate which heuristic can perform the best solution. The appropriate heuristic should provide the minimum average of % Dev.

Refer to the section 2.4 for the materials and methods chapter, all 3 heuristics are applied to select the alternatives of 30 tested problems and then the assignment solution of the selected alternative problem is solved and compared to the IP solution (IP Sol.) of the master problem. The result of solutions and the %Dev between the heuristics and the IP solutions are shown as follows.

Table 33 The %Dev of alternative selection Heuristic-1 solutions from the IP solutions

Problem No.	IP Sol.	Alt. Sel.	Heu-1 Sol.	Dev	Alt. Sel.	%Dev
1	74	3.1,6.1	76	2	3.2,6.1	2.7027
2	99	3.3,6.2	110	11	3.2,6.2	11.1111
3	138	3.2,6.2	144	6	3.2,6.1	4.34783
4	138	3.1,6.2	218	80	3.2,6.1	57.971
5	109	3.1,6.2	121	12	3.3,6.2	11.0092
6	98	3.2,6.1	98	0	3.2,6.1	0
7	148	3.2,6.1	172	24	3.3,6.1	16.2162
8	71	3.3,6.1	76	5	3.2,6.1	7.04225
9	72	3.2,6.1	100	28	3.3,6.1	38.8889
10	83	3.3,6.1	105	22	3.3,6.2	26.506
11	129	3.1,6.2	129	0	3.1,6.2	0
12	81	3.3,6.2	97	16	3.3,6.1	19.7531
13	60	3.2,6.1	60	0	3.2,6.1	0
14	77	3.2,6.1	88	11	3.3,6.1	14.2857
15	95	3.1,6.2	95	0	3.1,6.2	0
16	127	3.2,6.1	133	6	3.1,6.1	4.72441
17	68	3.1,6.1	79	11	3.3,6.1	16.1765
18	146	3.1,6.1	154	8	3.2,6.2	5.47945
19	82	3.3,6.1	85	3	3.2,6.1	3.65854
20	91	3.3,6.1	94	3	3.2,6.1	3.2967
21	104	3.1,6.1	104	0	3.1,6.1	0
22	54	3.1,6.1	54	0	3.1,6.1	0
23	143	3.1,6.2	143	0	3.1,6.2	0
24	67	3.1,6.2	111	44	3.2,6.2	65.6716
25	72	3.1,6.1	72	0	3.1,6.1	0
26	122	3.2,6.1	139	17	3.3,6.2	13.9344
27	111	3.1,6.1	111	0	3.1,6.1	0
28	65	3.2,6.1	65	0	3.2,6.1	0
29	98	3.1,6.2	98	0	3.1,6.2	0
30	217	3.1,6.2	217	0	3.1,6.2	0

Table 34 The %Dev of alternative selection Heuristic-2 solutions from the IP solutions

Problem No.	IP Sol.	Alt. Sel.	Heu-2 Sol.	Dev	Alt. Sel.	%Dev
1	74	3.1,6.1	74	0	3.1,6.1	0
2	99	3.3,6.2	110	11	3.2,6.2	11.1111
3	138	3.2,6.2	144	6	3.2,6.1	4.34783
4	138	3.1,6.2	152	14	3.1,6.1	10.1449
5	109	3.1,6.2	143	34	3.2,6.2	31.1927
6	98	3.2,6.1	98	0	3.2,6.1	0
7	148	3.2,6.1	148	0	3.2,6.1	0
8	71	3.3,6.1	76	5	3.2,6.1	7.04225
9	72	3.2,6.1	97	25	3.1,6.1	34.7222
10	83	3.3,6.1	126	43	3.2,6.1	51.8072
11	129	3.1,6.2	129	0	3.1,6.2	0
12	81	3.3,6.2	97	16	3.3,6.1	19.7531
13	60	3.2,6.1	63	3	3.1,6.1	5
14	77	3.2,6.1	122	45	3.2,6.2	58.4416
15	95	3.1,6.2	95	0	3.1,6.2	0
16	127	3.2,6.1	133	6	3.1,6.1	4.72441
17	68	3.1,6.1	89	21	3.1,6.2	30.8824
18	146	3.1,6.1	154	8	3.2,6.2	5.47945
19	82	3.3,6.1	106	24	3.1,6.1	29.2683
20	91	3.3,6.1	94	3	3.2,6.1	3.2967
21	104	3.1,6.1	120	16	3.1,6.2	15.3846
22	54	3.1,6.1	91	37	3.2,6.2	68.5185
23	143	3.1,6.2	149	6	3.3,6.2	4.1958
24	67	3.1,6.2	67	0	3.1,6.2	0
25	72	3.1,6.1	72	0	3.1,6.1	0
26	122	3.2,6.1	160	38	3.2,6.2	31.1475
27	111	3.1,6.1	111	0	3.1,6.1	0
28	65	3.2,6.1	65	0	3.2,6.1	0
29	98	3.1,6.2	98	0	3.1,6.2	0
30	217	3.1,6.2	249	32	3.2,6.1	14.7465

Table 35 The %Dev of alternative selection Heuristic-3 solutions from the IP solutions

Problem No.	IP Sol.	Alt. Sel.	Heu-3 Sol.	Dev	Alt. Sel.	%Dev
1	74	3.1,6.1	74	0	3.1,6.1	0
2	99	3.3,6.2	110	11	3.2,6.2	11.1111
3	138	3.2,6.2	138	0	3.2,6.2	0
4	138	3.1,6.2	152	14	3.1,6.1	10.1449
5	109	3.1,6.2	121	12	3.3,6.2	11.0092
6	98	3.2,6.1	98	0	3.2,6.1	0
7	148	3.2,6.1	164	16	3.1,6.1	10.8108
8	71	3.3,6.1	76	5	3.2,6.1	7.04225
9	72	3.2,6.1	72	0	3.2,6.1	0
10	83	3.3,6.1	105	22	3.3,6.2	26.506
11	129	3.1,6.2	129	0	3.1,6.2	0
12	81	3.3,6.2	81	0	3.3,6.2	0
13	60	3.2,6.1	63	3	3.1,6.1	5
14	77	3.2,6.1	88	11	3.3,6.1	14.2857
15	95	3.1,6.2	95	0	3.1,6.2	0
16	127	3.2,6.1	133	6	3.1,6.1	4.72441
17	68	3.1,6.1	68	0	3.1,6.1	0
18	146	3.1,6.1	152	6	3.1,6.2	4.10959
19	82	3.3,6.1	85	3	3.2,6.1	3.65854
20	91	3.3,6.1	91	0	3.3,6.1	0
21	104	3.1,6.1	104	0	3.1,6.1	0
22	54	3.1,6.1	89	35	3.2,6.1	64.8148
23	143	3.1,6.2	143	0	3.1,6.2	0
24	67	3.1,6.2	67	0	3.1,6.2	0
25	72	3.1,6.1	72	0	3.1,6.1	0
26	122	3.2,6.1	139	17	3.3,6.2	13.9344
27	111	3.1,6.1	111	0	3.1,6.1	0
28	65	3.2,6.1	77	12	3.3,6.1	18.4615
29	98	3.1,6.2	98	0	3.1,6.2	0
30	217	3.1,6.2	217	0	3.1,6.2	0

The interested result is considered that the %Dev of all 3 alternatives selection heuristics, which are shown on table 36. A summary of the %Dev is shown on table 37 as follows.

Table 36 The comparison of the %Dev for all 3 heuristics

Problem No.	IP Sol.	%Dev of Heuristic-1	%Dev Heuristic-2	%Dev Heuristic-3
1	74	2.7027	0	0
2	99	11.1111	11.1111	11.1111
3	138	4.34783	4.34783	0
4	138	57.971	10.1449	10.1449
5	109	11.0092	31.1927	11.0092
6	98	0	0	0
7	148	16.2162	0	10.8108
8	71	7.04225	7.04225	7.04225
9	72	38.8889	34.7222	0
10	83	26.506	51.8072	26.506
11	129	0	0	0
12	81	19.7531	19.7531	0
13	60	0	5	5
14	77	14.2857	58.4416	14.2857
15	95	0	0	0
16	127	4.72441	4.72441	4.72441
17	68	16.1765	30.8824	0
18	146	5.47945	5.47945	4.10959
19	82	3.65854	29.2683	3.65854
20	91	3.2967	3.2967	0
21	104	0	15.3846	0
22	54	0	68.5185	64.8148
23	143	0	4.1958	0
24	67	65.6716	0	0
25	72	0	0	0
26	122	13.9344	31.1475	13.9344
27	111	0	0	0
28	65	0	0	18.4615
29	98	0	0	0
30	217	0	14.7465	0

Table 37 The summary of the %Dev for all 3 heuristics

	% Dev of Heuristic-1	% Dev of Heuristic-2	% Dev of Heuristic-3
Mean	10.7592	15.2925	6.85378
S.D.	16.739	18.8379	12.8988
Min	0	0	0
Max	65.67164	68.51852	64.814815

According to the experimental results, the hypothesis is to test whether the mean values of the %Dev of the solution from the different alternative selection heuristics are equal or not. The hypothesis can be formally stated as:

H_0: $\mu_{\text{Heuristic-1}} = \mu_{\text{Heuristic-2}} = \mu_{\text{Heuristic-3}}$ $(\mu_1 = \mu_2 = \mu_3)$

H_1: $\mu_i \neq \mu_j$ for at least one pair of all i, j, where $i, j = 1, 2$ and 3

Let consider the results on table 36 with 30 replications and the error of $\alpha = 0.05$. The normal probability plots of the data set on table 36 are performed as follow.

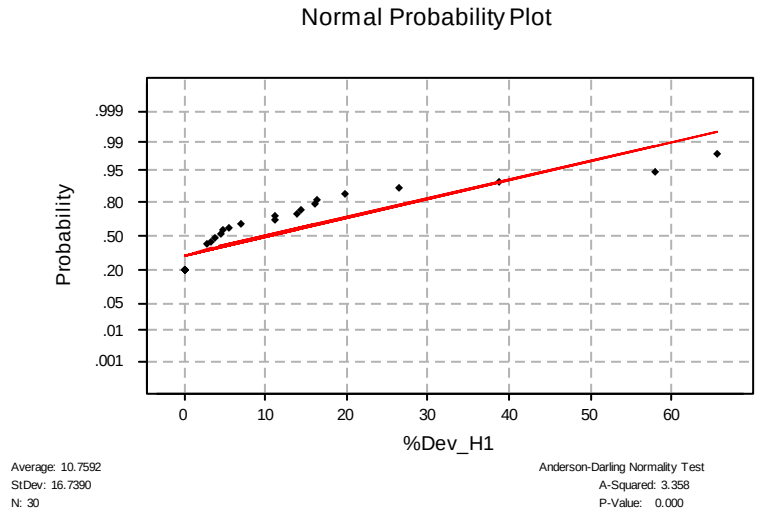


Figure 12 The normal probability plot of the %Dev of Heuristic-1 from table 36

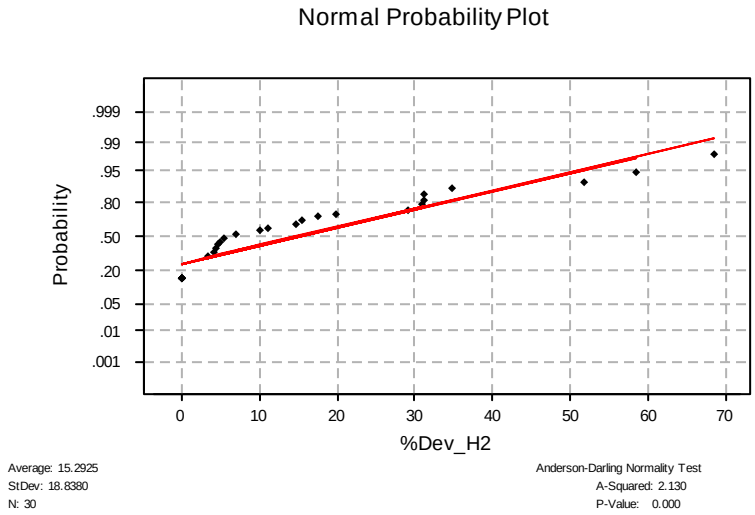


Figure 13 The normal probability plot of the %Dev of Heuristic-2 from table 36

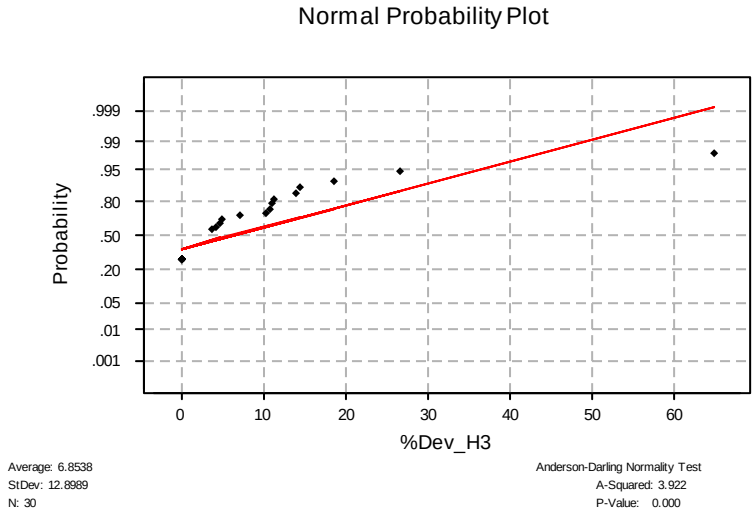


Figure 14 The normal probability plot of the %Dev of Heuristic-3 from table 36

The normal probability plots show that all 3 sets of data from table 36 are not the normally distributed. The Box-Cox transformation function in Minitab is used to transform the data set from the experimental results. After all 3 sets of data from table 36 are transformed, and then normality tests by the normal probability plot are performed, which are shown as follows.

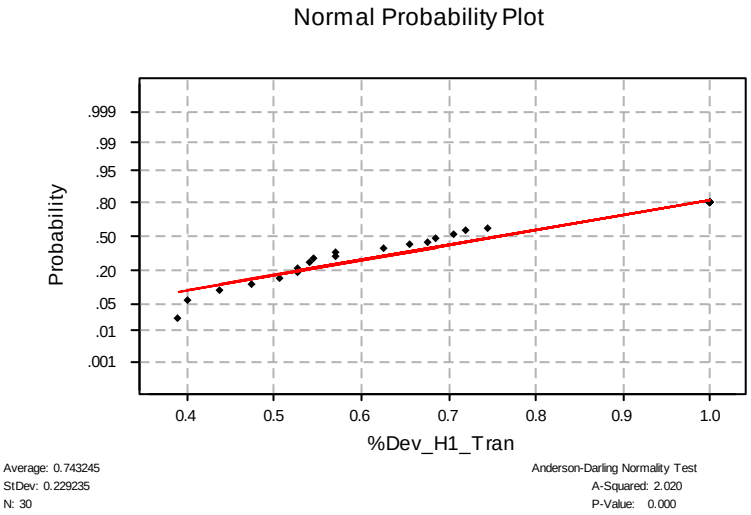


Figure 15 The normal probability plot of the Box-Cox transformation data of the %Dev of Heuristic-1 from table 36

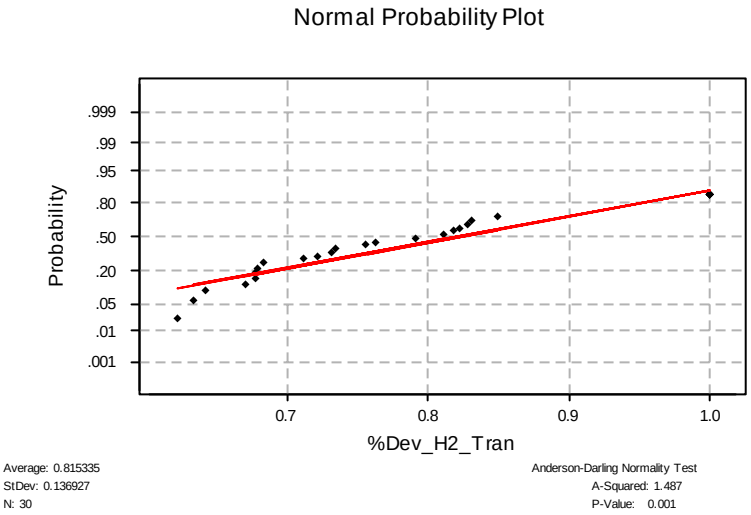


Figure 16 The normal probability plot of the Box-Cox transformation data of the %Dev of Heuristic-2 from table 36

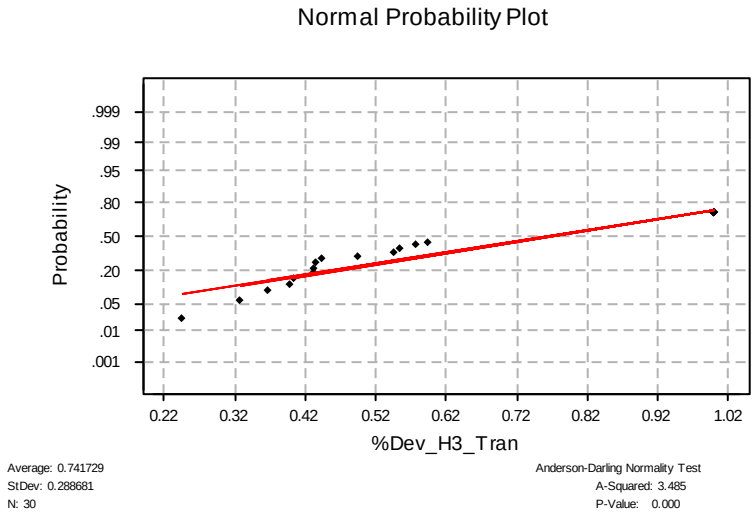


Figure 17 The normal probability plot of the Box-Cox transformation data of the %Dev of Heuristic-3 from table 36

Normal probability plots on figures 14, 15 and 16 show that all sets of transformed data are not normal distributed so that all data sets will be analyzed by using nonparametric statistics. The Kruskal-Wallis test is used for analyzing the data sets on figures 11, 12 and 13 for 3 treatments of Heuristic-1, Heuristic-2 and Heuristic-3 ($k = 3$) and 30 replications ($n_i = 30$ and $N = 90$). The result from Minitab is shown as follows.

Kruskal-Wallis Test: %Dev versus Heu-

Kruskal-Wallis Test on %Dev

Heu-	N	Median	Ave Rank	Z
1	30	4.00E+00	45.3	-0.05
2	30	6.26E+00	52.7	1.85
3	30	0.00E+00	38.5	-1.80
Overall	90		45.5	

H = 4.44 DF = 2 P = 0.108
H = 4.78 DF = 2 P = 0.092 (adjusted for ties)

Figure 18 The Kruskal-Wallis test of the %Dev for all 3 heuristics

If the type I error of $\alpha = 0.05$ is specified, then the result of from the Kruskal-Wallis test provides P-value = 0.092 > 0.05. The H_0: $\mu_{\text{Heuristic-1}} = \mu_{\text{Heuristic-2}} = \mu_{\text{Heuristic-3}}$ cannot be rejected and can be concluded base on the inference statistics that the average value of the %Dev of each alternative selection heuristic is not significantly different.

According to the results, all heuristics will be used appropriately for solving the large scale problem because the IP problem may requires too much memory and take too much time than solving the regular assignment problem with some heuristics of alternative selection. From the result on table 36, there are some cases that having much %Dev such as the tested problem No.22 of Heuristic-3, which has the %Dev of 64.815. The heuristic for improving the alternative selection heuristic is applied. The same 30 tested problems on table 33, 34 and 35 are used to verify the quality of solutions for the heuristic of alternative selection improvement heuristics by considering the deviation of solutions.

Let consider the tested problems on tables 33, 34 and 35 that have the deviation value grater than zero such as case numbers 2, 4, 5, and so on. The alternative selection improvement heuristic is applied and the results of solutions of the deviation of alternative selection heuristics, the deviation of alternative selection improvement heuristic solutions (Imp. Heu) and the IP solutions (IP Sol.) are shown as follows.

Table 38 The result of alternative selection improvement for Heuristic-1

No.	IP Sol.	Alt. Sel.	Heu-1 Sol.	Dev	Alt. Sel.	%Dev	Imp. Heu	Dev	Alt. Sel.	Iterations
1	74	3.1,6.1	76	2	3.2,6.1	2.7027	74	0	3.1,6.1	2
2	99	3.3,6.2	110	11	3.2,6.2	11.1111	99	0	3.3,6.2	2
3	138	3.2,6.2	144	6	3.2,6.1	4.34783	138	0	3.2,6.2	3
4	138	3.1,6.2	218	80	3.2,6.1	57.971	138	0	3.1,6.2	2
5	109	3.1,6.2	121	12	3.3,6.2	11.0092	109	0	3.1,6.2	3
6	98	3.2,6.1	98	0	3.2,6.1	0				
7	148	3.2,6.1	172	24	3.3,6.1	16.2162	148	0	3.2,6.1	2
8	71	3.3,6.1	76	5	3.2,6.1	7.04225	71	0	3.3,6.1	2
9	72	3.2,6.1	100	28	3.3,6.1	38.8889	72	0	3.2,6.1	3
10	83	3.3,6.1	105	22	3.3,6.2	26.506	83	0	3.3,6.1	3
11	129	3.1,6.2	129	0	3.1,6.2	0				
12	81	3.3,6.2	97	16	3.3,6.1	19.7531	81	0	3.3,6.2	3
13	60	3.2,6.1	60	0	3.2,6.1	0				
14	77	3.2,6.1	88	11	3.3,6.1	14.2857	77	0	3.2,6.1	3
15	95	3.1,6.2	95	0	3.1,6.2	0				
16	127	3.2,6.1	133	6	3.1,6.1	4.72441	127	0	3.2,6.1	2
17	68	3.1,6.1	79	11	3.3,6.1	16.1765	68	0	3.1,6.1	2
18	146	3.1,6.1	154	8	3.2,6.2	5.47945	146	0	3.1,6.1	2
19	82	3.3,6.1	85	3	3.2,6.1	3.65854	82	0	3.3,6.1	2
20	91	3.3,6.1	94	3	3.2,6.1	3.2967	91	0	3.3,6.1	2
21	104	3.1,6.1	104	0	3.1,6.1	0				
22	54	3.1,6.1	54	0	3.1,6.1	0				
23	143	3.1,6.2	143	0	3.1,6.2	0				
24	67	3.1,6.2	111	44	3.2,6.2	65.6716	67	0	3.1,6.2	2
25	72	3.1,6.1	72	0	3.1,6.1	0				
26	122	3.2,6.1	139	17	3.3,6.2	13.9344	122	0	3.2,6.1	3
27	111	3.1,6.1	111	0	3.1,6.1	0				
28	65	3.2,6.1	65	0	3.2,6.1	0				
29	98	3.1,6.2	98	0	3.1,6.2	0				
30	217	3.1,6.2	217	0	3.1,6.2	0				

Table 39 The result of alternative selection improvement for Heuristic-2

No.	IP Sol.	Alt. Sel.	Heu-2	Dev	Alt. Sel.	%Dev	Imp. Heu	Dev	Alt. Sel.	Iteration
1	74	3.1,6.1	74	0	3.1,6.1	0				
2	99	3.3,6.2	110	11	3.2,6.2	11.1111	99	0	3.3,6.2	2
3	138	3.2,6.2	144	6	3.2,6.1	4.34783	138	0	3.2,6.2	3
4	138	3.1,6.2	152	14	3.1,6.1	10.1449	138	0	3.1,6.2	2
5	109	3.1,6.2	143	34	3.3,6.2	31.1927	109	0	3.1,6.2	3
6	98	3.2,6.1	98	0	3.2,6.1	0				
7	148	3.2,6.1	148	0	3.2,6.1	0				
8	71	3.3,6.1	76	5	3.2,6.1	7.04225	71	0	3.3,6.1	2
9	72	3.2,6.1	97	25	3.1,6.1	34.7222	72	0	3.2,6.1	2
10	83	3.3,6.1	126	43	3.2,6.1	51.8072	83	0	3.3,6.1	2
11	129	3.1,6.2	129	0	3.1,6.2	0				
12	81	3.3,6.2	97	16	3.3,6.1	19.7531	81	0	3.3,6.2	3
13	60	3.2,6.1	63	3	3.1,6.1	5	60	0	3.2,6.1	2
14	77	3.2,6.1	122	45	3.2,6.2	58.4416	77	0	3.2,6.1	3
15	95	3.1,6.2	95	0	3.1,6.2	0				
16	127	3.2,6.1	133	6	3.1,6.1	4.72441	127	0	3.2,6.1	2
17	68	3.1,6.1	89	21	3.1,6.2	30.8824	68	0	3.1,6.1	3
18	146	3.1,6.1	154	8	3.2,6.2	5.47945	146	0	3.1,6.1	3
19	82	3.3,6.1	106	24	3.1,6.1	29.2683	82	0	3.3,6.1	3
20	91	3.3,6.1	94	3	3.2,6.1	3.2967	91	0	3.3,6.1	2
21	104	3.1,6.1	120	16	3.1,6.2	15.3846	104	0	3.1,6.1	2
22	54	3.1,6.1	91	37	3.2,6.2	68.5185	54	0	3.1,6.1	2
23	143	3.1,6.2	149	6	3.3,6.2	4.1958	143	0	3.1,6.2	2
24	67	3.1,6.2	67	0	3.1,6.2	0				
25	72	3.1,6.1	72	0	3.1,6.1	0				
26	122	3.2,6.1	160	38	3.2,6.2	31.1475	122	0	3.2,6.1	3
27	111	3.1,6.1	111	0	3.1,6.1	0				
28	65	3.2,6.1	65	0	3.2,6.1	0				
29	98	3.1,6.2	98	0	3.1,6.2	0				
30	217	3.1,6.2	249	32	3.2,6.1	14.7465	217	0	3.1,6.2	2

Table 40 The result of alternative selection improvement for Heuristic-3

No.	IP Sol.	Alt. Sel.	Heu-3	Dev	Alt. Sel.	%Dev	Imp. Heu	Dev	Alt. Sel.	Iteration
1	74	3.1,6.1	74	0	3.1,6.1	0				
2	99	3.3,6.2	110	11	3.2,6.2	11.1111	99	0	3.3,6.2	2
3	138	3.2,6.2	138	0	3.2,6.2	0				
4	138	3.1,6.2	152	14	3.1,6.1	10.1449	138	0	3.1,6.2	3
5	109	3.1,6.2	121	12	3.3,6.2	11.0091	109	0	3.1,6.2	3
6	98	3.2,6.1	98	0	3.2,6.1	0				
7	148	3.2,6.1	164	16	3.1,6.1	10.8108	148	0	3.2,6.1	3
8	71	3.3,6.1	104	33	3.2,6.1	46.4788	71	0	3.3,6.1	2
9	72	3.2,6.1	72	0	3.2,6.1	0				
10	83	3.3,6.1	105	22	3.3,6.2	26.5060	83	0	3.3,6.1	3
11	129	3.1,6.2	129	0	3.1,6.2	0				
12	81	3.3,6.2	81	0	3.3,6.2	0				
13	60	3.2,6.1	63	3	3.1,6.1	5	60	0	3.2,6.1	2
14	77	3.2,6.1	88	11	3.3,6.1	14.2857	77	0	3.2,6.1	3
15	95	3.1,6.2	95	0	3.1,6.2	0				
16	127	3.2,6.1	133	6	3.1,6.1	4.72440	127	0	3.2,6.1	2
17	68	3.1,6.1	68	0	3.1,6.1	0				
18	146	3.1,6.1	152	6	3.1,6.2	4.1095	146	0	3.1,6.1	3
19	82	3.3,6.1	85	3	3.2,6.1	3.6585	82	0	3.3,6.1	2
20	91	3.3,6.1	91	0	3.3,6.1	0				
21	104	3.1,6.1	104	0	3.1,6.1	0				
22	54	3.1,6.1	89	35	3.2,6.1	64.8148	54	0	3.1,6.1	2
23	143	3.1,6.2	143	0	3.1,6.2	0				
24	67	3.1,6.2	67	0	3.1,6.2	0				
25	72	3.1,6.1	72	0	3.1,6.1	0				
26	122	3.2,6.1	139	17	3.3,6.2	13.9344	122	0	3.2,6.1	2
27	111	3.1,6.1	111	0	3.1,6.1	0				
28	65	3.2,6.1	77	12	3.3,6.1	18.4615	65	0	3.2,6.1	2
29	98	3.1,6.2	98	0	3.1,6.2	0				
30	217	3.1,6.2	217	0	3.1,6.2	0				

The results from table 38, 39 and 40 show all heuristics of improving alternative selection can be performed well for all tested problems. All heuristics can provide the same solution as the IP model. For example, when consider tested problem No.2, the alternative selection Heuristic-3 provide the solution with the deviation of 11 units from the IP solution and then the result of alternative selection improvement heuristic shows that the solution is 99 units, which equal to the IP solution. The deviation becomes zero on the iteration 2 of running. The heuristic of alternative selection improvement appropriates for solving the lower bound of AGVsp-P/D with all alternative selection heuristics.

According to the experimental results, the hypothesis is to test whether the mean values of the %Dev of the solutions from the different alternative selection heuristics with the alternative selection improvement heuristic are equal or not. The hypothesis can be formally stated as:

H_0: $\mu_{\text{Heuristic-1+Imp}} = \mu_{\text{Heuristic-2+Imp}} = \mu_{\text{Heuristic-3+Imp}}$ ($\mu_1 = \mu_2 = \mu_3$)

H_1: $\mu_i \neq \mu_j$ for at least one pair of all i, j, where $i, j = 1, 2$ and 3

The data sets of the number of iterations from tables 38, 39 and 40 are not the normally distributed obviously, because the data are discrete numbers. The data set will be analyzed by using the nonparametric statistics. The Kruskal-Wallis test is used for analyzing data sets of the numbers of iterations from tables 38, 39 and 40 for 3 treatments of all heuristics ($k = 3$) and 30 replications ($n_i = 30$ and $N = 90$). The result from Minitab is showed as follows.

Kruskal-Wallis Test: No. of Iteration versus Heu-

Kruskal-Wallis Test on No. of I

Heu-	N	Median	Ave Rank	Z
1	30	2.00E+00	46.3	0.20
2	30	2.00E+00	49.4	1.01
3	30	0.00E+00	40.8	-1.21
Overall	90		45.5	

H = 1.68 DF = 2 P = 0.432
H = 1.92 DF = 2 P = 0.382 (adjusted for ties)

Figure 19 The Kruskal-Wallis test of the %Dev for all 3 heuristics with the alternatives selection improvement heuristic

If a type I error of $\alpha = 0.05$, the results provide P-value = 0.382 which is > 0.05. The H_0: $\mu_{\text{Heuristic-1+Imp}} = \mu_{\text{Heuristic-2+Imp}} = \mu_{\text{Heuristic-3+Imp}}$ cannot be rejected and can be concluded base on the inference statistics that all 3 alternative selection heuristics with the alternative selection improvement heuristic are not different significantly.

3. The results of solving the single TSP tour of the AGVsp-P/D using the modified Eastman's algorithm

The research attempts to implement the modified Eastman's algorithm for the TSP with lower the bound model of assignment problems with alternative P/D nodes on MATLAB 7. The 40 simulated problems with 10, 20, 30, 40, and 50 nodes, which consist of one job of 2 alternatives, one job of 3 alternatives and some regular jobs, are generated randomly. The running time of solving the single TSP tour of AGVsp-P/D, using the modified Eastman's algorithm, of the simulated problems are shown on table 41 and compared with the running time of solving the regular assignment problem with 10, 20, 30, 40, and 50 jobs, shown on table 42.

Table 41 The running time in second of solving the TSP tour of AGVsp-P/D

Problem No.	10 nodes	20 nodes	30 nodes	40 nodes	50 nodes
1	1.693	14.436	66.797	231.027	766.072
2	3.335	14.641	137.112	247.091	639.657
3	1.808	14.026	67.063	5.240	636.986
4	1.856	13.612	207.756	235.259	712.353
5	0.258	57.574	352.885	241.898	704.137
6	1.877	27.765	62.591	214.908	701.258
7	1.828	0.739	328.193	225.427	647.43
8	1.651	0.713	69.698	224.664	656.990
9	3.392	0.7111	79.219	219.988	668.780
10	1.682	14.103	69.182	6.183	734.886
11	0.226	55.339	134.911	241.618	734.411
12	0.262	16.011	67.598	719.097	744.121
13	0.283	27.951	304.586	204.507	679.370
14	1.825	14.284	73.322	236.092	642.580
15	1.859	13.929	80.206	425.945	637.226
16	0.298	14.475	73.39	474.657	639.772
17	1.832	13.888	76.527	231.260	708.768
18	1.698	15.442	65.905	237.361	711.008
19	0.307	15.025	64.765	224.364	712.425
20	1.692	0.769	64.591	252.56	657.413
21	1.747	30.800	70.174	226.488	636.859
22	1.674	28.348	71.670	236.647	732.089
23	1.740	0.792	114.147	231.182	647.651
24	0.284	13.745	64.398	246.789	676.477
25	0.285	32.151	69.559	5.246	732.258
26	0.263	13.072	70.199	235.618	734.277
27	1.757	65.791	2.161	261.750	732.742
28	0.237	13.280	78.426	237.382	673.567
29	1.652	58.381	72.749	255.169	640.159
30	0.211	13.693	75.813	205.515	625.663
31	0.290	41.085	65.514	266.024	629.462
32	1.719	14.342	64.394	6.832	625.209
33	1.731	14.264	64.568	266.383	702.128
34	1.717	56.760	69.746	787.577	681.357
35	0.229	0.721	71.241	232.691	682.662
36	1.713	0.672	113.415	246.183	771.671
37	0.314	13.932	64.098	426.096	764.973
38	1.796	14.332	69.131	475.67	770.507
39	1.841	14.932	70.087	230.799	708.948
40	0.198	53.106	2.160	235.654	647.383

Table 42 The comparison of the running time in second of solving the single TSP tour and the regular assignment problem

Size Levels	Statistics	Assignment Problem	TSP Tour Problem
10 nodes	Mean	0.2186975	1.2765
	S.D.	0.031660438	0.870312
	Min	0.161	0.198
	Max	0.282	3.392
20 nodes	Mean	0.681025	20.9908
	S.D.	0.04663084	18.14552
	Min	0.558	0.672
	Max	0.778	65.791
30 nodes	Mean	2.1654	94.74868
	S.D.	0.152020545	74.87147
	Min	1.963	2.16
	Max	2.655	352.885
40 nodes	Mean	5.478875	260.371
	S.D.	0.318063144	153.5101
	Min	4.678	5.24
	Max	5.976	787.577
50 nodes	Mean	12.4137	688.7921
	S.D.	0.999490388	45.38168
	Min	10.499	625.209
	Max	14.656	771.671

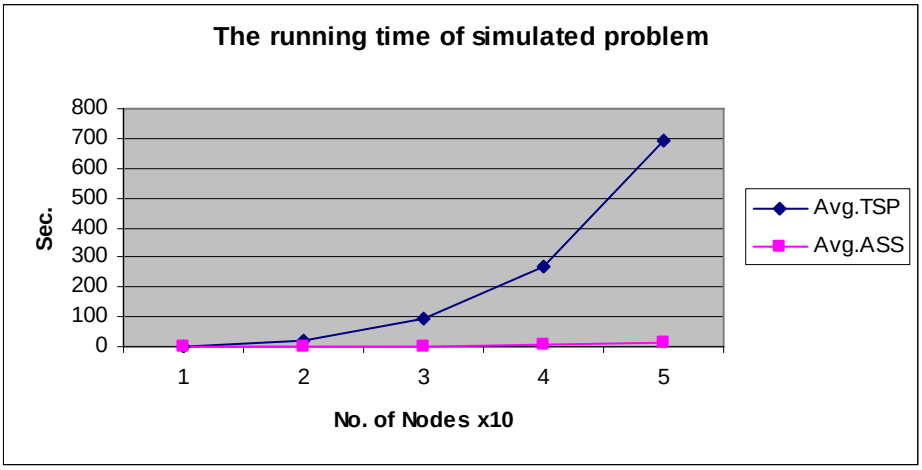


Figure 20 The graph of running time in second of solving the single TSP tour of the AGVsp- P/D

Consider the graph on figure 20, the graph shows that the running time of solving the single TSP tour of the AGVsp-P/D increases dramatically when the number of AGVs is increased. This result shows that the heuristic can be used to solve the case of single AGVsp-P/D.

4. Results of solving the multi AGVsp-P/D

This part of the experiment of solving multi AGVsp-P/D using the heuristic of solving the MTSP as the standard TSP is performed. The solution of multi AGVsp-P/D is the sets of multi TSP tours. The heuristic is programmed on MATLAB 7.0. The 50 simulated problems with 10, 20, 30, 40, and 50 nodes, which consist of one job of 2 alternatives, one job of 3 alternatives and some regular jobs for all cases of single AGV ($M = 1$), 2 AGVs ($M = 2$) and 3 AGV ($M = 3$) are generated randomly. The data sets of running time of solving the multi AGVsp-P/D by considering only the calculation time, not include the problem set up time, are shown on table 43, 45 and 47.

According to this experiment, the main purpose is to examine that whether the increasing of the number of AGVs affects on the average running time of solving the multi AGVsp-P/D or not. The hypothesis test will be examined after all experiments done

Table 43 The running time in second of 10 nodes MTSP for the multi AGVsp-P/D

Problem No.	10 nodes		
	$M = 1$	$M = 2$	$M = 3$
1	0.5781	0.4963	0.3187
2	1.3044	0.4534	1.3768
3	0.245	0.5547	0.2213
4	0.2081	0.2613	0.3131
5	0.8105	0.3016	0.6008
6	1.2781	0.844	2.0474
7	0.223	0.4001	1.026
8	0.4473	0.6251	0.9685
9	0.2722	3.6764	0.6313
10	0.8397	0.4283	0.4077
11	1.1375	0.4061	0.7901
12	0.2273	1.7481	0.2893
13	0.1793	0.3521	0.7662
14	0.470	0.4544	0.5212
15	0.9578	0.364	0.9125
16	0.25105	0.238	0.9261
17	0.6137	0.3895	0.4226
18	0.7162	0.6538	0.7331
19	1.122	1.3593	0.5414
20	0.266	0.2857	2.3444
21	0.3687	0.4212	1.0174
22	0.473	0.272	1.0084
23	0.2933	0.4277	2.5028
24	1.0118	0.3127	0.2272
25	0.4057	0.4355	0.8508
26	0.1926	1.598	0.7059
27	1.3251	0.2247	0.7774
28	0.6122	0.2395	2.2004
29	0.2553	0.375	1.1827
30	0.2766	3.846	0.5311
31	0.516	0.2879	0.403
32	0.7142	1.756	0.2493
33	0.2956	0.8841	0.8959
34	0.208	0.6551	0.8793
35	0.6997	0.3407	0.4914
36	0.3625	0.4416	4.081
37	0.2044	0.7505	0.7278
38	0.3185	0.8979	0.302
39	0.5303	0.5229	1.9412
40	0.5628	0.773	0.4439

Table 43 (Continued)

Problem No.	10 nodes		
	$M = 1$	$M = 2$	$M = 3$
41	0.3808	0.698	0.5543
42	0.258	0.6145	0.3318
43	0.3888	0.889	3.0799
44	0.259	0.6725	0.7411
45	0.506	1.2406	0.8989
46	0.2321	0.7288	1.071
47	0.2642	0.823	1.9991
48	0.4068	0.3143	0.498
49	0.3919	0.8437	2.5709
50	0.536	0.1283	1.134

Table 44 The summary of 10 nodes MTSP for the multi AGVsp-P/D

	10 nodes		
	$M = 0$	$M = 1$	$M = 2$
Mean	0.507943	0.734138	1.009128
S.D.	0.31895549	0.72995078	0.816217924
Min	0.1793	0.1283	0.2213
Max	1.3251	3.846	4.081

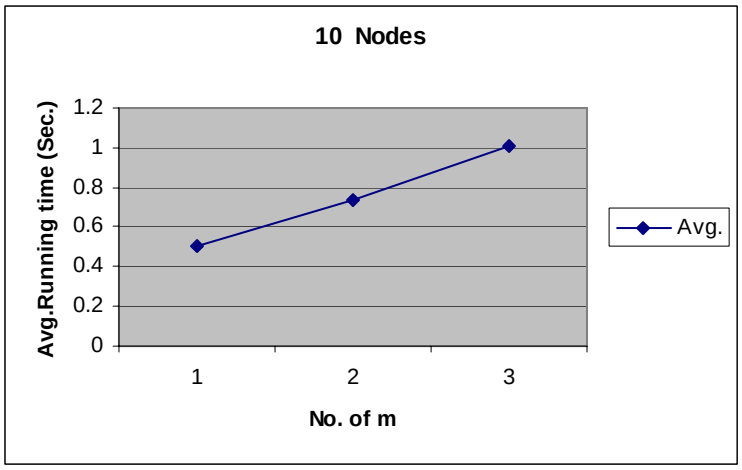


Figure 21 The graph of the average running time in second of 10 nodes MTSP

Table 45 The running time in second of 20 nodes MTSP for the multi AGVsp-P/D

Problem No.	20 nodes		
	$M = 1$	$M = 2$	$M = 3$
1	7.2711	1.4687	10.4991
2	5.3215	4.666	1.4812
3	0.6562	9.2695	4.9078
4	2.4304	7.9018	9.1667
5	1.6493	10.6857	4.6736
6	1.2746	1.3451	1.4609
7	2.2308	8.7386	9.7711
8	9.3824	5.4041	1.4887
9	3.9393	2.3899	9.8529
10	2.9384	31.7611	5.4141
11	1.9332	3.9518	0.8323
12	8.9992	15.4405	6.7089
13	6.9871	1.296	27.2017
14	20.3591	2.1797	19.8646
15	1.1316	4.7616	6.8771
16	21.499	2.6524	57.4077
17	7.4028	7.9937	7.7732
18	1.2898	4.4645	25.7845
19	6.4044	4.7286	6.7818
20	5.7406	25.9307	20.7878
21	3.406	4.2891	0.8185
22	3.5275	13.9361	11.7984
23	2.284	2.0237	13.7122
24	0.653	13.2995	12.4737
25	1.3032	11.3131	5.241
26	1.1372	17.0743	5.1957
27	24.5203	12.4049	4.4442
28	2.6884	7.6847	6.9935
29	2.662	10.4358	7.4455
30	1.7965	8.6316	17.0687
31	1.2876	45.5607	8.242
32	7.5661	13.0314	4.9419
33	3.0469	5.3899	0.7621
34	1.7372	14.5202	14.315
35	13.2586	12.085	11.0408
36	7.1424	4.5083	3.7207
37	8.8102	6.4901	12.2831
38	27.3407	7.4976	19.7948
39	3.8289	41.4066	0.8236
40	1.363	4.8039	8.2879

Table 45 (Continued)

Problem No.	20 nodes		
	$M = 1$	$M = 2$	$M = 3$
41	23.9681	1.3514	23.1394
42	1.7786	11.9583	18.5901
43	2.2326	2.8668	4.8258
44	7.5768	9.2478	14.582
45	2.5632	21.5848	28.713
46	1.3223	2.5375	39.9687
47	15.2526	4.4379	26.2798
48	4.7328	7.7225	8.5443
49	12.6934	2.7995	6.5797
50	5.1685	10.4343	8.324

Table 46 The summary of 20 nodes MTSP for the multi AGVsp-P/D

	20 nodes		
	$M = 1$	$M = 2$	$M = 3$
Mean	6.309788	9.767146	11.753716
S.D.	6.76525129	9.38018444	10.7872062
Min	0.653	1.296	0.7621
Max	27.3407	45.5607	57.4077

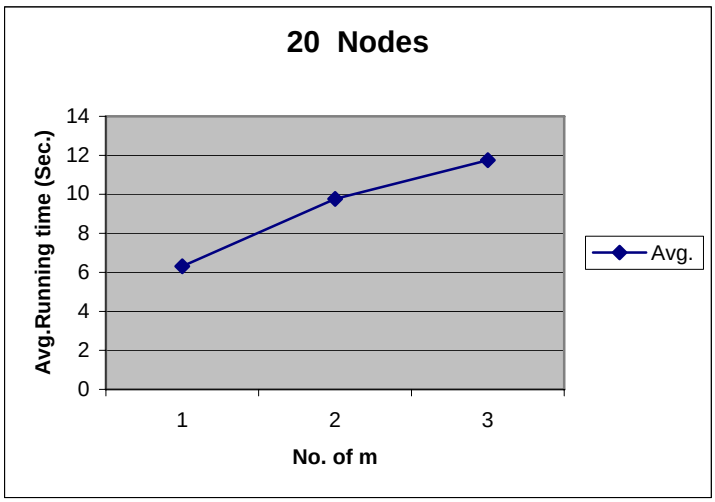


Figure 22 The graph of the average running time in second of 20 nodes MTSP

Table 47 The running time in second of 30 nodes MTSP for the multi AGVsp-P/D

Problem No.	30 nodes		
	$M = 1$	$M = 2$	$M = 3$
1	14.468	25.7828	33.311
2	96.8074	31.2256	11.646
3	17.1893	133.2994	81.8107
4	136.9473	10.0478	28.5088
5	102.1074	33.8301	5.4162
6	22.7224	16.1166	178.181
7	42.1609	43.8518	18.4791
8	54.9581	40.2743	31.0544
9	23.0809	16.9561	19.5557
10	61.138	20.6118	12.8187
11	24.7432	78.939	71.4472
12	32.4986	14.7052	89.8392
13	75.8807	34.8351	84.0734
14	23.8733	46.0669	138.6934
15	14.4361	174.4891	30.6783
16	10.9639	46.729	82.247
17	129.4007	21.0921	14.3649
18	4.0006	39.324	42.9922
19	62.6249	96.6452	26.1129
20	27.143	17.8762	47.5797
21	65.4614	33.1484	6.8357
22	29.9751	24.1449	23.0791
23	28.8981	28.4337	8.9306
24	11.624	22.0013	51.8817
25	13.7934	43.0344	65.8268
26	25.5864	50.598	25.8491
27	14.9759	68.6874	163.0203
28	7.876	24.099	24.2744
29	74.7315	27.4172	27.8802
30	6.5012	26.2887	25.8394
31	43.0726	90.2681	94.7307
32	12.4687	14.7459	59.423
33	155.261	52.2774	66.3459
34	10.0452	42.5896	30.3738
35	6.1793	102.03667	10.0852
36	23.533	19.4913	26.7812
37	6.7356	2.2572	94.643
38	26.025	14.2288	117.7158
39	28.208	58.6066	20.4345
40	35.1858	32.0658	27.4943

Table 47 (Continued)

Problem No.	30 nodes		
	$M = 1$	$M = 2$	$M = 3$
41	37.3528	19.2568	31.293
42	19.916	2.1943	41.5333
43	10.3684	36.4643	62.6321
44	6.0149	4.5291	35.7456
45	10.421	63.5197	42.5076
46	32.1945	141.0189	15.1661
47	16.7367	29.7821	81.5883
48	127.6941	93.0225	28.6096
49	37.7534	69.2408	96.3396
50	5.4234	38.6061	76.6012

Table 48 The summary of 30 nodes MTSP for the multi AGVsp-P/D

	30 nodes		
	$M = 1$	$M = 2$	$M = 3$
Mean	38.143142	44.3350614	50.645418
S.D.	37.5907785	36.4018846	40.0213276
Min	4.0006	2.1943	5.4162
Max	155.261	174.4891	178.181

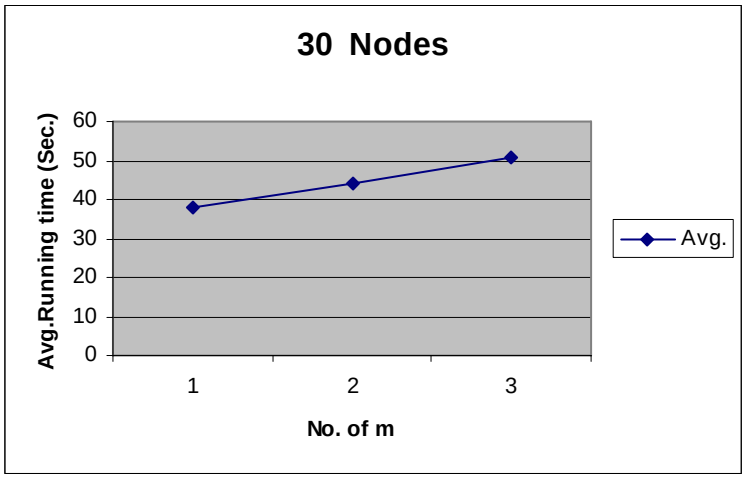


Figure 23 The graph of the average running time in second of 30 nodes MTSP

According to the experimental results, the hypothesis is to test whether the mean values of the running time from solving the multi AGVsp-P/D of the different number of AGVs are equal or not. The hypothesis can be formally stated as:

H_0: $\mu_{M=1} = \mu_{M=2} = \mu_{M=3}$ ($\mu_1 = \mu_2 = \mu_3$)

H_1: $\mu_i \neq \mu_j$ for at least one pair of all i, j, where $i, j = 1, 2$ and 3

Let consider experiments of $M = 1$, $M = 2$ and $M = 3$ of 30 nodes simulated problems with 50 replications and the type I error of $\alpha = 0.05$. The normal probability plots of data sets on table 47 are performed as follow.

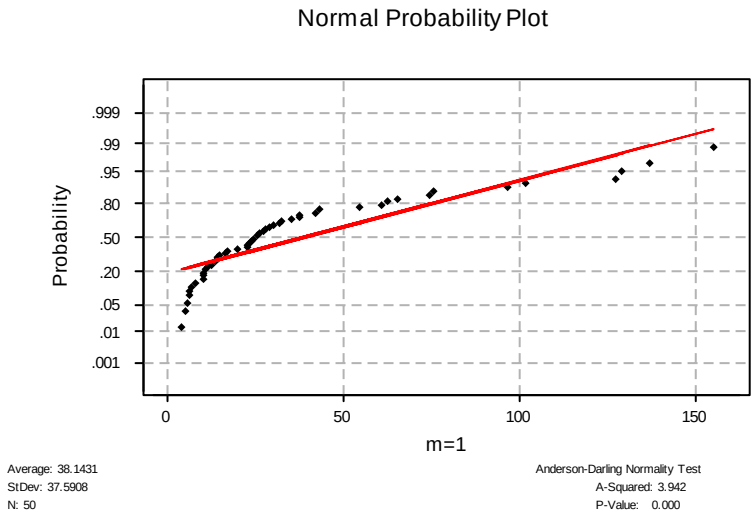


Figure 24 The normal probability plot of the average running time of $M = 1$ from table 47

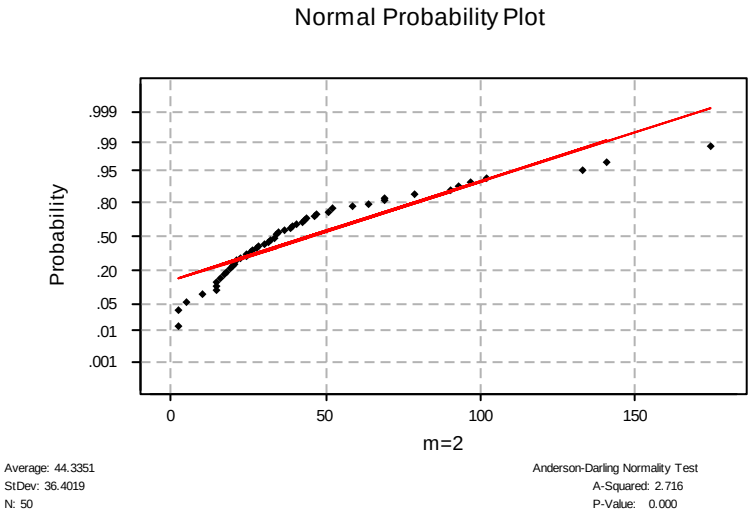


Figure 25 The normal probability plot of the average running time of M =2 from table 47

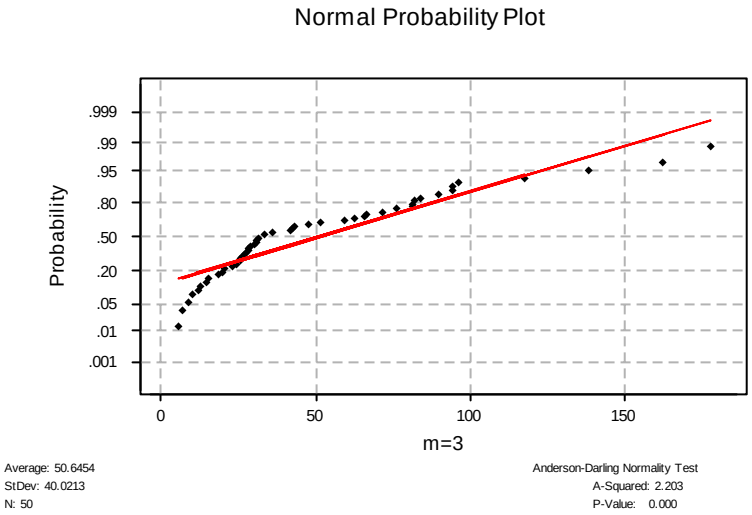


Figure 26 The normal probability plot of the average running time of M =3 from table 47

The normal probability plots show that all data sets of table 47 are not normally distributed. The Box-Cox transformation function in Minitab is used to transform all data sets from experimental results. All data sets of table 47 are transformed, which are M = 1-Trans, M = 2-Trans and M = 3-Trans, shown on table 49. The normality tests by normal probability plots are performed, which are shown on figures 27, 28 and 29 as follows.

Table 49 The data set from Box-Cox transformation of table 47

Problem No.	30 nodes		
	M = 1-Trans	M = 2-Trans	M = 3-Trans
1	0.739668	2.07227	1.48538
2	0.596858	2.16321	1.31925
3	0.725420	2.99521	1.64391
4	0.573944	1.67758	1.45951
5	0.593279	2.20241	1.21005
6	0.702929	1.86506	1.79486
7	0.655561	2.33434	1.38982
8	0.636240	2.29022	1.47367
9	0.701688	1.88642	1.39873
10	0.628634	1.97083	1.33362
11	0.696202	2.66323	1.61897
12	0.675105	1.82713	1.66137
13	0.613492	2.21692	1.64898
14	0.699020	2.36028	1.74482
15	0.739852	3.18161	1.47165
16	0.763185	2.36784	1.64490
17	0.577627	1.98104	1.35087
18	0.855154	2.27799	1.52878
19	0.626931	2.78686	1.44513
20	0.688966	1.90890	1.54637
21	0.623805	2.19238	1.24226
22	0.681292	2.04200	1.42512
23	0.684112	2.11825	1.28031
24	0.758166	1.99987	1.56155
25	0.743665	2.32452	1.60407
26	0.693574	2.41046	1.44347
27	0.736793	2.58145	1.77694
28	0.792216	2.04113	1.43327
29	0.614550	2.10103	1.45585
30	0.809554	2.08132	1.44341
31	0.653981	2.74453	1.67134
32	0.752188	1.82826	1.58565
33	0.565871	2.42817	1.60550
34	0.770760	2.31911	1.46999
35	0.814207	2.82099	1.29800
36	0.700153	1.94629	1.44925
37	0.806324	1.20027	1.67117
38	0.692245	1.81369	1.71283
39	0.685980	2.49119	1.40568
40	0.669079	2.17612	1.45356

Table 49 (Continued)

Problem No.	30 nodes		
	M = 1-Trans	M = 2-Trans	M = 3-Trans
41	0.664582	1.94101	1.47494
42	0.713465	1.19268	1.52283
43	0.768011	2.23976	1.59509
44	0.816689	1.40311	1.49726
45	0.767572	2.53657	1.52682
46	0.675822	3.03326	1.35917
47	0.727608	2.14037	1.64341
48	0.578494	2.76309	1.46010
49	0.663782	2.58609	1.67452
50	0.826286	2.26860	1.63175

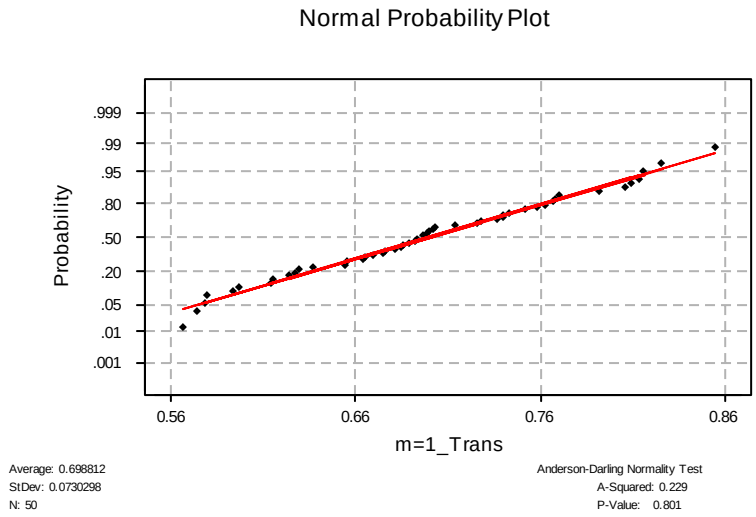


Figure 27 The normal probability plot of Box-Cox transformation data of M = 1 from table 49

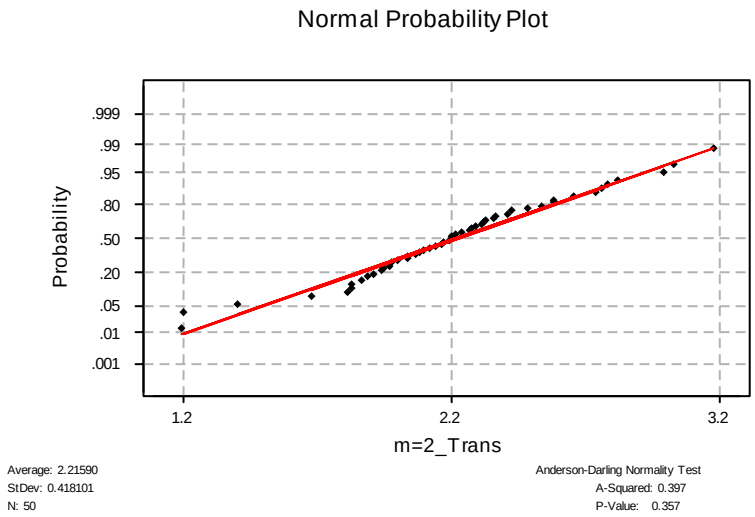


Figure 28 The normal probability plot of Box-Cox transformation data of $M = 2$ from table 49

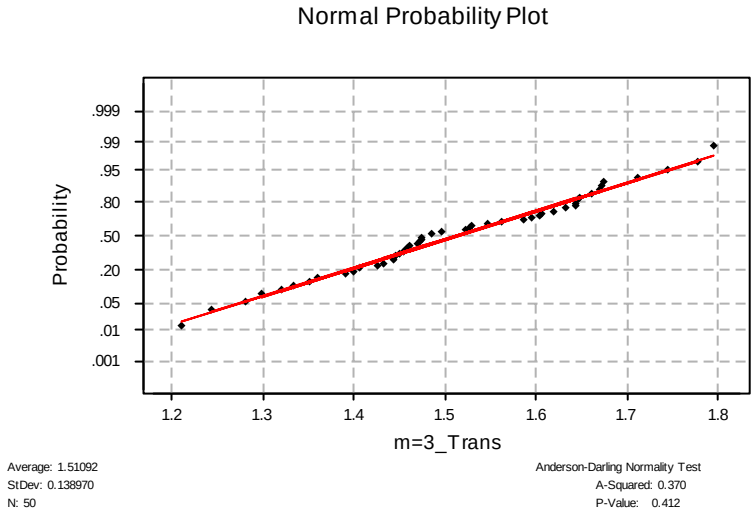


Figure 29 The normal probability plot of Box-Cox transformation data of $M = 3$ from table 49

The normal probability plots on figures 27, 28 and 29 show all transformed data sets are normally distributed. According to this point, all data sets of table 49 on figures 27, 28 and 29 can be used to perform the hypothesis test by using regular statistic methods. The ANOVA is used for analyzing the transformed data set of table 49 for 3 treatments, which are M = 1-Trans, M = 2-Trans and M = 3-Trans, (k = 3) and 50

experiments (n_i = 50 and N = 150), that implies the analysis of original data set. The ANOVA table is shown as follows.

Table 50 The ANOVA table of the data set on table 49

Source of Variation	Sum of Squares	Degree of Freedom	Mean Squares	F statistic (f_0)
Treatment	57.63443	2	28.8172149	433.441717
Error	9.773242	147	0.06648464	
Total	67.40767	149		

If the type I error of $\alpha = 0.05$ is specified, the result from the ANOVA table provides the value of $f_0 = 433.441$. From the statistical table of F distribution, the value of $f_{0.05, v1, v2}$ of this ANOVA is $f_{0.05, 2, 147} = 3.057$. The result of the ANOVA is $f_0 > f_{0.05, 2, 147}$. Therefore, the H_0: $\mu_{M=1} = \mu_{M=2} = \mu_{M=3}$ ($\mu_1 = \mu_2 = \mu_3$)can be rejected ($f_0 > f_{0.05, 2, 147}$) and can conclude base on the inference statistics that the different number of AGVs affects on the mean value of the running time of solving multi AGVsp-P/D.

The results show that the mean values of the running time of solving the multi AGVsp-P/D with the different numbers of AGVs are not equal, at the type I error α = 0.05, because this heuristic solves the MTSP as solving the standard TSP with the additional dummy rows and columns. According to this point, this heuristic can be used to solve the multi AGVsp-P/D with the specific number of AGVs.

5. The result of the heuristic of splitting the single TSP tour for solving the multi AGVsp-P/D

From the previous section, when the number of AGVs is increased, the average running time of solving multi AGVsp-P/D increases. If the larger problem of AGVsp-P/D with a lot of AGVs (vehicles) is considered, it may take a lot of computational time and memory to solve the problem. The large numbers of AGVs (M) may cause the running out of memory of MATLAB 7.0. The heuristic of splitting the single TSP tour is used for solving the lower bound solution of multi AGVsp-P/D. The lower bound of multi AGVsp-P/D solutions is the sets of multi tours, not multi TSP tours. The 30 simulated problems with 10 nodes, which consist of one job of 2 alternatives, one job of 3 alternatives and 5 regular jobs are generated randomly for the cases of 2 AGVs (M =2). The solution of M TSP tours from solving the MTSP as the standard TSP and the solutions of M tours from the heuristic of splitting a single TSP tour are compare. The %Dev of the solutions from both methods is examined as follows.

Table 51 The solutions of multi tours from the heuristic of splitting a TSP tour (Spliting TSP Heu) and the solutions of multi TSP tours from the heuristic of solving MTSP as a standard TSP(MTSP Heu)

No.	MTSP Heu	Spliting TSP Heu	Dev	%Dev
1	152	236	84	55.263
2	96	169	73	76.042
3	170	223	53	31.176
4	165	221	56	33.939
5	96	157	61	63.542
6	102	102	0	0.000
7	177	225	48	27.119
8	168	226	58	34.524
9	155	198	43	27.742
10	170	239	69	40.588
11	119	177	58	48.739
12	178	190	12	6.742
13	135	168	33	24.444
14	142	185	43	30.282
15	219	234	15	6.849
16	134	156	22	16.418
17	195	288	93	47.692
18	138	179	41	29.710
19	164	198	34	20.732
20	171	213	42	24.561
21	119	170	51	42.857
22	178	210	32	17.978
23	135	214	79	58.519
24	142	185	43	30.282
25	219	236	17	7.763
26	134	156	22	16.418
27	155	224	69	44.516
28	74	87	13	17.568
29	171	213	42	24.561
30	132	203	71	53.788

Table 52 The statistical summary of table 51

	MTSP Heu	Spliting TSP Heu	Dev	%Dev
Mean	150.167	196.067	45.900	32.012
S.D.	34.097	40.914	23.307	18.251
Min	74.000	87.000	0.000	0.000
Max	219.000	288.000	93.000	76.042

From the result, the splitting heuristic can be used to form the multi tours for *M* AGVs from the single TSP tour of the single AGV. The % deviation of the splitting TSP Heu and MTSP Heu solutions show that the solutions of Spliting TSP Heu deviate so much, about 32% on the average, from the MTSP Heu, but the Spliting TSP Heu can generate the feasible solution easily and quickly for the large AGVsp-P/D systems with the large number of AGVs.

Results Analysis Based on the Tested Problem

This section presents the analysis of experimental results based on all results of all tested problems. The analysis will be conducted following the sequences, which are 1. analysis of the lower bound solution of AGVsp-P/D by integer programming, 2. analysis of the lower bound solution of AGVsp-P/D by alternative selection heuristics, 3. analysis of results of solving the single TSP tour of the AGVsp-P/D by the modified Eastman's algorithm, and 4. analysis of results of solving the multi AGVsp-P/D.

1. Analysis of the lower bound solution of AGVsp-P/D by integer programming

Recall table 32, it is the statistical summary of the running time in seconds of the lower bound solution of the AGVsp-P/D and figure 6, which is the graph of table 32**.** The analysis of this section is based on the 40 generated problems. The result of the average running time of all cases, which are assignment, 2Al-5 and 2Al-Max problems, increases when the numbers of nodes are increased. The increasing in each case is not linearly proportional to the number of nodes, which the trend is much more rapidly increasing. The formulated 0-1 IP of the lower bound model can provide the solution well but may required a lot of memory take too much time for the large problem. Based on the experiment of 2Al-Max, the problem size of 50 nodes takes 50.12 seconds to run by MATLAB 7.0 on average, but for the 30 nodes problems takes only 6.61 seconds. The increasing of the average running time grow rapidly, because problems are 0-1 IP model that are solved by branch and bound approach. More number of nodes means more variable to solve, which form more branching.

When the problem is larger than 50 nodes, the average running time may increase dramatically as same as the required memory, which is used to solve the problem. The experiment did not go further more 50 nodes because the MATLAB 7.0 will give the warning of "out of MEMORY" on the "*bintprog*" function.

Another point of consideration is the number of alternative jobs. The research does the testing on no-alternative jobs (the regular assignment problems), 2Al-5, and 2Al-Max. The graph in figure 6 shows that the number of alternative jobs affects so much on the increasing of the average running time. Consider the problems of 50 nodes of the regular assignment problem, 2Al-5 (five pairs of alternative jobs and 40 normal jobs) and 2Al-Max (twenty five pairs of alternative jobs and no normal jobs) the number of 2 alternative jobs is increased from 0 to 5 jobs and from 5 to 25 jobs. The average running time is increased from 12.41 to 13.76 seconds and from 13.76 to 50.12 seconds. From the inference statistic, the result expresses the same conclusion as the descriptive statistic, which is mentioned previously that the same size of problem but with a different number of alternative jobs may provide the different average running time. Because the lower bound model is solved by branch and bound approach, more alternative numbers form more branching of alternative selections for the same size of problems.

2. Analysis of the lower bound solution of AGVsp-P/D by alternative selection heuristics

From table 37, the comparison of results of the % deviation (%Dev) of all alternatives selection heuristics is analyzed. From the descriptive statistics show that Heuristic-3 provides the minimum value of the %Dev on the average, standard deviation of the %Dev and the maximum number of problems, which obtain the heuristic solution the same as the IP solution. Considering the inference statistic, the Kruskal-Wallis test shows that all heuristics do not perform differently in term of the mean value of the % Dev. The results of testing shows that all heuristics can be used equivalently, but the descriptive statistics shows that the Heuristic-3 is the most efficient (based on table 37). Heuristic-3 provides better solutions than the others on

the average but it is also the most complex approach. The %Dev is the main consideration because the research wants to find the solutions from the alternative selection heuristics as close as the IP solution of the same problem. The experiment does not mention about the running time of all heuristics because all heuristics are not the complicated algorithm and do not take many steps. When all heuristics are applied, the main part of running time will be taken by running the assignment problem, not from the methods of alternative selection from heuristics. It can conclude that all heuristics can provide the solutions close to IP solutions by solving the regular assignment problems that it is efficiently for the large problem.

There are some cases having too much %Dev and the research attempts to test the heuristic for improving the alternative selection heuristics. The research applies the alternative selection improvement heuristic to all alternative selection heuristics and considers the improvement of the reduction of the %Dev. The results show that all cases can provide the heuristic solution same as the IP solution after applying the alternative selection improvement heuristic to the initial solution from alternative selection heuristics, but different number of iterations. The improvement heuristic is the searching heuristic so that some cases may search all possible alternatives. If the initial solution is not the optimum, this heuristic can provides the better solution exactly.

3. Analysis of results of solving the single TSP tour of the AGVsp-P/D by the modified Eastman's algorithm

When the modified Eastman's algorithm is applied to the lower bound solutions of the AGVsp-P/D, the assignment solutions become the TSP tours. This searching procedure is the branch and bound approach so that the running time increases dramatically when the problem size is increased because the 0-1 IP subproblems are solved for all branches. The objective of this experiment is the modified Eastman's algorithm. The research shows that this algorithm performs well but takes quite a lot of computation time and required memory for solving the AGVsp-P/D using MATLAB 7.0.

Now the research found the TSP tour solutions for the AGVsp-P/D by solving the 0-1 IP subproblem with the modified Eastman's algorithm for the TSP. Table 42 and figure 20 show that the average running time of the 50 nodes problem is 688.79 seconds. This result leads to the conclusion that the average running time will grow dramatically and not linear proportion when the number of nodes is inceased.

4. Analysis of the results from solving the multi AGVsp-P/D

This analysis focuses the effect of the running time, when the additional AGVs are added to the system of the AGVsp-PD. Let's consider the results on table 51 and 52, the ANOVA of the experiment and the descriptive statistic show that the increasing of the number of AGVs affects on the average running time that consider only the calculation time. The average running time of 10 nodes AGVsp-P/D increases about 0.3 seconds when the number of AGVs is increased from 2 AGVs to 3 AGVs, but when the 30 node problems are considered, the running time will be increased about 5 seconds. This algorithm performs adding one node to the problem when one additional AGV is added. The MTSP can be solved as the standard TSP of the problem with some additional nodes. The running time of solving the MTSP does not increase much compared to solving the TSP of same problem size. It can say that this algorithm performs well on solving multi AGVsp-P/D and provides the solution, which is the set of TSP tours. If the problem is very large and uses a lot of AGVs, the heuristic of splitting the single TSP tour may be used appropriately. The solution of the multi AGVsp-P/D in the form of multi tours, not multi TSP tours, may suitable for the large manufacturing system.

Discussions

Based on research results, this research accomplishes all research objectives, which are studying the problem of single/multi AGVsp-P/D, developing the algorithms to solve the problem and creating some computer programs for solving single/multi AGVsp-P/D for testing the quality of the model. This section discusses many issues such as the weaknesses of all algorithms of this research on the objective s perspective, real world applications of AGVsp-P/D and difficulties on the implementation of AGVsp-P/D model.

1. Problem of Single/Multi AGVsp-P/D

This AGVsp-P/D is a special case of TSP in both cases of single and multisalesman. Bases on literature search results, there are not shown any literature that explains about the TSP with the special structure of alternative P/D nodes. The research generated the mathematical model for the AGVsp-P/D in the form of modified TSP/MTSP. The generated model can provide a solution in the form of the schedule of jobs for the AGV with alternative selection nodes but this model does not consider many constraints in the real world. The assumptions of the static job list and fixed plaint layout make the model inflexible for some kinds of product layout manufacturing. Many manufacturing process concepts such as “just in time” or lean system may produce response to the changing of demand by minimizing the stock. This AGV system that cannot support the dynamic demand provides a poor solution in a real production system. Therefore, the research model appropriates for applying to the manufacturing layout in which the product items do different steps in different departments with static environments. Another issue is about the problem size. The generated AGV system in the form of the 0-1 IP lower bound model with modified Eastman’s algorithm may take too much time for implementing in the real world situations because of the size of problem. Although the model can provide the solutions well, on average, they are suitable for problems of static layout that are not larger than 50 nodes.

2. The weakness of the model of AGVsp-P/D

There are two weaknesses of the formulated model of AGVsp-P/D, which are 1. problem size and 2. computation time.

The first weakness is the problem size. The current approach uses the 0-1 IP approach for forming the mathematical model of AGVsp-P/D. The research creates the heuristics to support the larger problem size. However, the result still illustrates preferences of the IP model because of the obtained solution quality. The branch and bound is used to solve the TSP tour of the AGVsp-P/D by using modified Eastman's algorithm. Generally, branch and bound approach is an exhaustive search that is used to solve the 0-1 IP. The research use MATLAB 7.0 that has the function to solve the 0-1 IP. This software can provide a stable running condition up to about 50 nodes for this research model. Because of the nature of 0-1 IP, the problem takes a lot of memory for computing on MATLAB 7.0, which is limited on the regular personal computer with 2 GB RAM. This research uses MATLAB 7.0 because it provides the flexibility to program and can run automatically through the algorithms. For future improvement, because the *bintprog* function of solving 0-1 IP is the main consuming of required memory, the way to program the model of AGVsp-P/D should be changed by improving the program structure of solving the 0-1 IP without using the *binprog* function to avoid the "out of MEMORY" for extending to support the larger problems.

The second weakness is about the computation time. The combinatorial nature of the TSP/MTSP affects on the computation time of the problem obviously. Because the generated model of the AGVsp-P/D is a special case of the TSP, the increasing of running time is not the polynomial function, exactly. From the results, the model takes about 10 minutes, on the average, to solve 50 nodes for the TSP solution of the AGVsp-P/D. For the real implementation, the software for programming the AGVsp-P/D may be much more powerful than MATLAB 7.0. The average running time may be improved by reprogramming the model on the other software.

3. The application of AGVsp-P/D

In some applications of the regular AGV systems, only few numbers of vehicles and jobs are involved. The aim of the regular AGV scheduling is to dispatch a set of AGVs to achieve the goals for a batch of pick up and delivery jobs under the certain constraints such as batch size, deadlines, priority and etc. The goals are normally related to the processing time or utilization of resources, such as minimizing the total traveling time or distance of all vehicles. Qiu and Hsu (2002) proposed the survey paper of the AGV scheduling. The paper showed that most of current AGV system uses simple scheduling algorithms. Jobs are usually handled in a first-come-first-serve (FCFS) fashion, and the nearest idle vehicle is usually chosen to serve a next job. When the problem of scheduling of AGVs in the real manufacturing situation is different from the conventional path problem such as this AGVsp-P/D, the AGV systems still operate under the human monitoring and decision making, not exactly automatic.

The AGVsp-P/D model is developed directly under the real manufacturing application that is automated materials handling systems. For example, the AGV starts at the recharging depot then goes to pick up its first job at some process station, such as pick up the items from a drilling station, and goes to deliver the items at some process station, such as a milling station. Because the factory may have many milling stations, the AGV has to select the appropriate milling station for minimizing the total traveling distance. Askin and Standridge (1993) illustrated some examples of applications of the AGV system. Many real world applications descried below are the versatilities, which the AGVsp-P/D system can be applied to solve the problem.

3.1. Satellite signal transferring

Consider the signal transferring systems, the signal is sent from the ground station to the satellite and comes back to the station. The signal is generated from the main controller device and transferred to the ground transition posts having multiple routers. Next, the signal will being into one router that transfers the signal to the next

nearest ground transition post until be transferred reaching the satellite transition station of the specific satellite. Then, the signal is sent to the satellite to operate the equipments and backs to the main controller device. On the path of signal traveling, there are so many posts and routers to choose in the transferring system. The first-come-first-serve (FCFS) fashion, and the nearest idle router may be usually chosen to serve a transferring, but it may not provides good efficiency. According to this problem structure, the AGVsp-P/D can be applied to the signal transferring problems.

3.2. Circuit board wiring

This kind of problem appears normally in the design of any wiring such as the car's computers and the other digital systems. A system consists of the numbers of modules and several pins that are located on each module. A given set of pins has to be connected by the wire. Some modules, which have sever common pins such as a ground pin, have alternative pins to be selected for connecting to other modules in the circuit. In order to avoid the signal crossing and to minimize the length of wire, the AGVsp-P/D model can be applied to this problem also.

3.3. Messenger scheduling

The following type of problem occurs repeatedly in the decision making process of the messengers, for example in a financial agent. The messenger job is similar to the AGV job that starts at the office and goes to pick up the financial documents, such as the checks or cash, from one customer and deliver them to the specific bank (or post office) and goes to the next customer until all customers are served then goes back to the office. There are so many banks for serving the customer needs, for example the messenger has to go to deliver the checks of one customer to KBank. There are many KBanks that messenger can select to deliver the documents and then goes to the next job. It is similar to the AGVsp-P/D model. A simulated example of this application is showed in an appendix.

4. The difficulty on the implementation of the AGVsp-P/D model

The first difficulty is concerned with how the user can apply the model of AGVsp-P/D to solve the real world situation. The research is created based on many assumptions that make it possible for solving by some mathematical approaches. The research does the AGVsp-P/D model with specific kinds of variables and many constraints, because the research want to capture the real world situation as much as possible, but there are many real world situations that hard to be formed into the mathematical model. For example, some real conditions that are the AGVs speed, the pick up and delivery period, the traffic jam conditions or the maintenance activities are not considered in this research. When the model is implemented, the obtained solutions should be adjusted to avoid the infeasible implementation. The suitable set up of the material handling system should be considered concomitantly with implementing the AGVsp-P/D system.

Secondly, based on the result of the tested problems of the multi AGVsp-P/D, when the additional AGVs are added to the problem, the running time is increased but not much. However, the research does not consider the cost of additional AGVs. The model considers solving the problem by minimizing the total traveling distance with the specific numbers of AGVs, but without considers the operating cost, which is an important factor in the real world situation. Users should conduct the trade off analysis between the optimum solutions and the appropriate implementation conditions.

CONCLUSION AND RECOMMENDATION

Conclusion

In this research, the mathematical model of the AGVsp-P/D and the relaxation model which is the assignment problem with alternative P/D nodes are proposed with the solving approaches by integer linear programming, Benders'decomposition and both constructive and improvement heuristics. The assignment problem with alternative P/D nodes is solved for finding the lower bound of the AGVsp-P/D. This study is conducted because the assignment problem is the subproblem that has to be solved in all iterations of solving the TSP/MTSP by a branch and bound approach. This research creates a knowledge base for studying the TSP/MTSP with alternative nodes that cab be applied for solving the AGVsp-P/D.

The assignment problem with alternative P/D nodes has a special structure that is different from the original assignment problem and cannot be solved by the traditional solving approach of the original assignment problem. This special structure creates an effect on the unimodular property of the assignment problem, which the 0-1 IP model of original assignment problem can be solved as a regular linear programming without concerning the 0-1 integer constraints. When the alternative P/D nodes constraints are added, the model will lose the unimodular property. The research creates a new mathematical model for formulating the assignment problem with alternative P/D nodes, which is the lower bound of AGVsp-P/D by modifying the original assignment problem structure. The created model is still the 0-1 IP, look like the assignment problem, and can be solved by using branch and bound approach that can be programmed on MATLAB 7.0 for solving the problem. The procedure of solving 0-1 IP requires a lot of memory to run on MATLAB 7.0 and makes the program cannot solve the lower bound of AGVsp-P/D beyond 50 nodes, because MATLAB 7.0 shows "out of MEMORY" for solving the binary problem using the "*bintprog*" function.

For solving larger problem size, the research creates the heuristics for solving the lower bound of the AGVsp-P/D without solving the created 0-1 IP model, but solving the linear programming or the regular assignment problem with some heuristic methods. Benders'decomposition approach is applied to solve the lower bound model of AGVsp-P/D. The created algorithm of solving the lower bound of AGVsp-P/D by Benders'decomposition is a complicate procedure and still solving the 0-1 IP in the Benders partial mater problem, but the problem sizes of the 0-1 IP is smaller than the direct method. This method can be use to solve the lower bound of the AGVsp-P/D, which larger than 50 nodes. Then, the research attempts to create other heuristics without solving the 0-1 IP model. The created heuristics consist of three alternatives selection heuristics and one alternative selection improvement heuristic. All alternatives selection heuristics are the constructive heuristic methods that can provides the initial solution of assignment problem with alternative P/D nodes, which is the initial lower bound then, the initial solution is improve by the alternative selection improvement heuristic that can provide the improved solutions, which is the better lower bound solution of the AGVsp-P/D.

After that the modified Eastman's algorithm for TSP is applied to the lower bound solution of the AGVsp-P/D, which can provide the single TSP tour for the cases of single AGVsp-P/D. Finally, the heuristics for solving the multi AGVsp-P/D are created by using the TSP tours solution. Two heuristics that are the heuristic of solving the MTSP as the standard TSP and the heuristic of solving multi tours from splitting a single TSP tour are compared to show that the heuristic of solving the MTSP as the standard TSP can provide the solution of multi TSP tours with less total tour distance than the solution of multi tours from the heuristic of splitting a single TSP. The advantage of the heuristic of splitting a single TSP is that the heuristic can provide the feasible solution quickly for a larger problem size that the heuristic of solving the MTSP as the standard TSP cannot run on MATLAB 7.0.

The simulated problem is generated to verify and validate the quality of the AGVsp-P/D model by using MATLAB 7.0. All results from experiments are analyzed by statistical methods with type I error $\alpha = 0.05$. The results of average running time

of solving the lower bound model of AGVsp-P/D with different size levels of problem using the 0-1 IP model are analyzed using statistical methods. The conclusion is that the averaging running time increases when the number of node is increased. The experiments of solving the lower bound model of AGVsp-P/D using three alternatives selection heuristics and alternative selection improvement heuristic focuses on the %Dev of heuristic solutions from IP solutions. The statistical analysis results show that all three created heuristics provide not different on the mean value of the %Dev of heuristic solutions from IP solutions, which imply all heuristics can by used to solve the lower bound solution of AGVsp-P/D equivalently. When the modified Eastman's algorithm for TSP is applied to the lower bound solution of the AGVsp-P/D, the result is the TSP tour solution of AGVsp-P/D. The average running time of 50 nodes problem is about 688 seconds. For multi AGVsp-P/D cases, the average running time of solving the problem using the heuristic of solving MTSP as the standard TSP of 30 nodes problem with 3 AGVs is about 50 seconds. The statistical analysis shows that the average running time is increased not much when one AGV is added to the system, but it is significant at the type I error $\alpha = 0.05$. When the heuristic of splitting a single TSP tour for multi tours of the multi AGVsp-P/D is applied, the average %Dev of this heuristic solutions and the heuristic of solving MTSP as the standard TSP solutions is about 32%. The heuristic of splitting a single TSP tour provided much %Dev of the solution but it can provide the solution quickly for the larger size of AGVsp-P/D. According to this point, all solutions and analysis results from all simulated problems satisfy all model constraints, can provide the solution of AGVsp-P/D and can be applied to use in the real situations.

Recommendation

The AGVsp-P/D model is formulated by applying the TSP approach that the created model is the 0-1 IP model. This research conducts the study and analysis to create the knowledge bases of AGV problem with some special structure. The created model attempts to capture the structure of alternative P/D nodes, but ignore many real world constraints so that the model may feasible for some real applications by relaxing unconsidered constraints, but may not feasible for many cases. This model is suitable for the fixed layout of traveling path, fixed job list and constant AGV speed that are not compatible with many real flexible manufacturing systems. The implementations of this created AGVsp-P/D model will success when the obtained solutions should be adjusted to handle the realistic situations.

The objective of the single/multi AGVsp-P/D is to minimize the total traveling distance, which is total tour/tours length. When the multi AGVsp-P/D cases are considered, the obtained solutions are the route of multi TSP tours, which is the minimum total traveling distance, but not balance the length of each TSP tour. For the real situation, if the additional AGVs are supplied to the system, all AGVs should be utilized equivalently because the additional AGV make the increasing of operating cost. For example, the solution may provide two TSP tour for 10 nodes problem with 2 AGVs that consists of one TSP tour of seven nodes and another TSP tour of three nodes. One AGV may still running, but another AGV is already finished and becomes the unutilized at the same period time. For the suitable implementation, all AGVs should be scheduled and utilized equivalently. The solution of multi AGVsp-P/D provides the minimum total traveling distance, but not provides the maximum AGV utilization and the minimum operating cost that is the most important issue in any real world situations. When the multi AGVsp-P/D model is implementing to the real manufacturing or applications, the obtained solutions should be analyzed and concerned about the operating cost. All considered AGVs should be utilized as equivalent as possible. The obtained solutions from solving the AGVsp-P/D should conduct the trade-off analysis between the minimum total traveling distance and the minimum operating cost.

The problem of the single/multi AGVsp-P/D with 50 nodes can be solved by MATLAB 7.0 on a personal computer with 2 GB RAM, but the research found that most of problems having more than 50 nodes cause MATLAB 7.0 to be "out of MEMORY" in solving the 0-1 IP. The model works well for solving single/multi AGVsp-P/D with fewer nodes. If the larger problem sizes are considered, the heuristics of alternative selection and improvement or Benders' decomposition approach should be applied.

From the limitation previously, the future research should extend to cover more realistic situation for more accuracy and reality of the obtained solution. The other solving approaches should be considered instead of branch and bound approach. The cost of operation and the dynamic job list constraints should be studied. The researcher believes that this research can be modified to cover the more realistic events and can still be solved under some mathematical approaches.

LITERATURE CITED

Ahuja, R.K., T.L. Magnanti and J.B. Orlin. 1993. **Network Flows: Theory, Algorithms, and Applications.** Prentice-Hall International, Inc., New Jersey.

Askin, R. G. and C. R. Standridge. 1993. **Modeling and Analysis of Manufacturing Systems.** John Wiley and Sons, Inc., New York.

Bellmore, M. and S. Hong. 1974. Transformation of the multisalesmen problem to the standard traveling salesman problem. **J. Assoc. Comput. Mach.** 21: 500-504.

Bellmore, M. and G.L. Nemhauser. 1968. The traveling salesman problem: A survey. **Oper. Res.** 16: 538-558.

Benders, J.F. 1962. Partitioning procedure for solving mixed-variable programming problems. **Numer. Math.** 4: 238-252.

Blair, E.L., P. Charnsethikul and A. Vasques. 1987. Optimal routing of driverless vehicles to support flexible manufacturing. **Material Flow**. 4: 73-83.

Charnsethikul, P. 1993. The multi-traveling salesman problem with balancing criteria. **Thai Journal of Development Administration** 33: 217-229.

Chartrand, G. and O.R. Oellerman. 1993. **Applied and Algorithmic Graph Theory.** McGraw-Hill, Inc., New York.

Clarke, G. and J.W. Wright. 1964. Scheduling of vehicles from a central depot to a number of delivery points. **Oper. Res.** 12: 568-581.

Claus, A. 1984. A new formulation for the traveling salesman problem. **SIAM J. Alg. Disc. Math.** 5: 21-25.

Dantzig, G.B., D.R. Fulkerson and S.M. Johnson. 1954. Solution of a large-scale traveling salesman problem. **Oper. Res.** 2: 393-410.

Dantzig, G.B., D.R. Fulkerson and S.M. Johnson. 1959. On a linear-programming, combinatorial approach to the traveling salesman problem. **Oper. Res.** 7: 58-66.

Dantzig, G.B. and J.H. Ramser. 1959. The truck dispatching problem. **Management Sci.** 6: 80-91.

Eastman, W.L. 1958. **Linear Programming with Pattern Constraints.** Ph.D. Thesis, Harvard University.

Egbelu, P.J. 1986. Pull versus push strategy for automated guided vehicle load movement in a batch manufacturing system. **J. Manu. Sys.** 6: 209-220.

Egbelu, P.J. 1987. The use of non-simulation approach in estimating vehicle requirements in an automated guided vehicle based transport system. **Material Flow** 4: 4-16.

Egbelu, P.J. and J.M.A. Tanchoco. 1984. Characterizations of automated guided vehicle dispatching rules. **Int. J. Prod. Res.** 22: 359-374.

Gaskins, R.J. and J.M.A. Tanchoco. 1987. Flow path design for automated guided vehicle systems. **Int. J. Prod. Res.** 25: 667-676.

Gavish, B. and S.C. Graves. 1987. **The Traveling Salesman Problem and Related Problems.** Working Paper OR-078-78. Operations Research Center, MIT.

Gillete, B.E. 1976. **Introduction to Operations Research.** McGraw-Hill, Inc., New York.

Hamilton, W.R. 1856. Memorandum respecting a new system of roots of unity (The Icosian Calculus). **Philos. Mag.** 12: 446.

Kaspi, M., U. Kesselman and J.M.A. Tanchoco 2002. Optimal solution for the flow path design problem of a balance unidirectional AGV system. **Int. J. Prod. Res.** 40: 389-401.

Kaspi, M. and J.M.A. Tanchoco 1990. Optimal flow path design of unidirectional AGV systems. **Int. J. Prod. Res.** 28: 1023-1030.

Kirkman, T.P. 1856. On the representation of polyhedral. **Philos. Trans. Roy. Soc. London Ser. A**. 146: 413-418.

Koff, G.A. 1987. Automated guided vehicle systems: Applications, controls and planning. **Material Flow** 4: 3-16.

Laporte, G. 1997. Modeling and solving several classes of arc routing problems as traveling salesman problems. **Comput. Oper. Res.** 24: 1057-1061.

Lawler, E.L., J.K. Lenstra, A.H.G. R. Kan and D.B. Shmoys. 1995. **The Traveling Salesman Problem.** John Wiley and Sons, Inc., Chichester.

Lin, S. and B.W. Kernighan 1973. An effective heuristic algorithm for the traveling salesman problem. **Oper. Res.** 21: 498-516.

Little, J.D.C., K.G. Murty, D.W. Sweeney and C. Karel. 1963. An algorithm for the traveling salesman problem. **Oper. Res.** 11: 972-989.

Maxwell, W.L. 1981. Solving material handling design problems with OR. **Industrial Engineering** 4: 58-69.

Maxwell, W.L. and J. A. Muckstadt. 1982. Design of automated guided vehicle systems. **IIE Transactions.** 14: 114-124.

Miller, C.E., A.W. Tucker and R.A. Zemlin. 1960. Integer programming formulation of traveling salesman problems. **J. Assoc. Comput. Mach.** 7: 326-329.

Montgomery, D. C. and G. C. Runger. 2002. **Applied Statistics and Probability for Engineers.** 4th ed. John Wiley and Sons, Inc., New York.

Orloff, C.S. 1974. A fundamental problem in vehicle routing. **Networks** 4: 35-64.

Orman, A.J. and H.P. Williams. 2004. **A Survey Report of Different Integer Programming Formulations of the Traveling Salesman Problem.** Working Paper No. LSEOR 04.67. London School of Economics and Political Science, London.

Padberg, M. and T.Y. Sung. 1991. An analysis comparison of different formulations of the traveling salesman problems. **Math. Programming** 52: 315-352.

Shapiro, D.M. 1966. **Algorithm for the Solution of the Optimal Cost and Bottleneck Traveling Salesman Problems**, Sc.D. Thesis, Washington University.

Svestka, J. A. and V. E. Huckfeldt 1973. Computational experience with an m-salesman traveling salesman algorithm. **Management Sci.** 19: 790-799.

Tanchoco, J.M.A. and C.L. Moodie. 1987. Automated guided vehicle systems: Special issue. **Material Flow** 4: 1-126.

APPENDIX

This part shows some simulated examples that can help to understand some content of this research. There are two examples, the example of lower bound of AGVsp-P/D by Benders' decomposition approach, and the example of application of AGVsp-P/D that are shown as follows.

1. The example of solving the lower bound of the AGVsp-P/D by the Benders'decomposition approach

Refer to the result section of solving the lower bound of the AGVsp-P/D by integer programming, most of problems which have more than 50 nodes causes MATLAB 7.0 out of memory in calculation of binary problems but the Benders'decomposition approach can be applied for a larger problem that the direct solving method using branch and bound on MATLAB 7.0 can not generate solutions.

Let consider the generated problem of 60 nodes with 60 of variables *Z* that are 55 regular jobs, one of the 2 alternatives job and one of the 3 alternatives job. The distance matrix of 60 nodes problem can be shown as follows.

Columns 1 through 10

∞	54	85	16	71	74	19	61	2	88
26	∞	66	46	29	34	91	64	12	52
52	91	∞	29	47	13	82	80	43	41
29	47	61	∞	43	87	33	94	57	39
33	61	56	88	∞	58	87	56	29	1
23	18	86	97	93	∞	24	96	51	15
39	86	75	99	28	68	∞	13	57	41
3	64	27	71	97	65	81	∞	92	47
30	79	85	17	76	91	49	80	∞	10
78	3	40	90	33	91	68	94	85	∞
11	60	43	43	63	59	33	21	44	27
96	64	40	49	93	52	45	80	53	68
30	34	79	13	85	57	14	8	51	36

9	15	45	75	61	13	30	41	64	83
52	39	46	25	98	42	32	75	64	29
51	23	73	6	46	13	92	62	49	57
71	60	69	91	31	70	7	86	71	87
1	76	27	23	55	82	4	53	95	71
69	12	74	29	5	9	69	26	71	93
98	98	7	4	31	31	58	60	46	26
73	71	90	10	1	71	76	94	7	88
17	18	32	19	68	43	29	7	95	67
48	90	11	68	83	23	24	82	27	69
4	14	85	28	83	29	91	83	52	86
31	40	29	91	67	35	75	64	3	76
57	40	89	77	43	34	91	89	82	15
18	33	23	30	28	37	45	38	25	20
86	60	66	41	46	93	49	48	75	65
0	45	65	65	99	94	18	79	1	10
67	30	10	66	34	91	22	52	98	84
75	52	7	79	45	14	26	6	34	48
74	93	21	4	28	40	45	84	14	96
45	75	84	56	37	4	2	16	24	76
95	45	69	56	96	69	83	65	48	37
41	87	63	36	53	2	95	74	16	86
5	63	70	38	77	2	66	36	13	63
46	24	57	8	22	78	93	92	35	88
67	19	31	64	64	47	20	79	81	16
49	91	94	26	62	48	10	12	57	53
39	73	60	4	13	47	34	33	15	95
33	91	2	2	90	16	82	79	14	55
57	37	29	83	12	32	61	76	42	60
93	58	66	91	94	71	42	73	49	40
73	22	60	48	44	47	25	89	85	45
6	8	96	52	0	87	71	17	97	13

67	38	82	81	74	58	58	36	1	4
27	23	4	76	4	45	32	98	97	4
85	75	26	66	85	83	2	80	53	31
63	48	7	97	65	82	88	81	32	59
93	51	22	68	94	77	66	70	3	83
5	42	82	22	87	58	78	58	3	85
97	43	23	75	38	49	64	9	93	17
41	41	60	11	74	68	48	3	21	15
71	52	69	52	22	76	24	20	12	77
44	97	44	80	29	52	61	54	25	59
90	91	26	57	87	29	35	24	99	18
86	98	41	94	32	77	38	20	95	73
50	27	54	45	27	34	12	78	52	73
50	36	41	55	20	32	24	26	86	96
46	65	37	18	45	5	44	30	78	50

Columns 11 through 20

13	65	27	17	35	59	14	32	35	91
91	83	18	8	50	71	58	31	76	51
13	46	59	77	26	16	71	56	57	33
15	28	75	50	67	19	10	24	81	39
79	38	28	98	52	35	45	93	19	36
51	79	15	50	68	91	58	98	50	79
6	78	13	7	46	44	91	69	78	26
10	18	45	14	81	74	33	97	67	77
3	58	81	74	12	63	11	97	57	14
47	38	24	59	83	65	65	74	9	9
∞	12	73	84	97	61	55	59	88	93
98	∞	48	32	83	8	84	75	30	76
44	33	∞	28	9	93	58	66	42	8
36	13	89	∞	54	97	46	57	43	27
40	89	30	66	∞	68	73	19	68	12

98	5	49	10	31	∞	71	34	84	33
50	69	74	30	51	45	∞	52	15	94
4	2	21	9	32	3	97	∞	74	56
5	33	34	62	55	25	48	37	∞	27
4	53	24	54	52	85	7	12	98	∞
89	73	48	11	89	92	80	64	67	19
47	70	95	18	44	53	33	5	40	34
9	32	67	36	49	60	60	98	37	90
96	76	85	69	69	52	0	72	73	88
26	96	71	33	76	49	45	62	2	64
57	35	57	92	88	75	44	34	7	84
97	89	43	65	12	20	70	75	97	11
94	5	72	59	33	1	29	98	45	31
40	97	17	24	84	83	17	18	98	95
87	32	17	61	77	18	48	58	56	59
53	28	86	56	88	64	67	26	94	62
84	59	92	7	50	47	2	53	24	82
25	88	21	93	69	1	51	74	37	50
32	55	78	84	51	50	13	14	26	31
22	17	12	56	88	40	8	87	12	93
58	55	53	15	2	92	74	13	49	81
80	65	12	70	73	3	59	50	39	98
64	59	10	46	8	58	24	19	99	28
84	54	70	58	52	95	1	61	24	92
72	42	63	24	82	0	43	10	26	11
74	34	72	33	40	39	93	40	28	86
69	8	98	91	79	44	49	13	30	59
59	77	92	32	83	84	68	84	53	17
28	64	19	78	48	71	75	97	12	83
22	71	10	59	77	48	82	98	41	51
16	43	35	98	14	34	2	76	9	7
71	6	76	95	18	65	76	68	38	40

12	8	39	98	15	50	32	70	85	11
67	70	88	74	13	36	75	31	35	24
41	11	70	88	36	37	92	82	51	13
33	59	10	3	55	79	64	13	83	88
69	82	33	79	3	9	65	43	50	54
8	32	2	78	20	54	81	49	7	62
89	66	66	44	79	56	90	28	35	65
22	10	26	62	51	56	73	79	97	98
1	1	63	47	59	55	83	25	12	47
48	64	67	74	75	95	82	27	97	24
43	93	78	36	7	87	97	86	13	21
67	32	46	45	85	2	56	16	0	71
16	61	27	75	38	7	81	45	69	68

Columns 21 through 30

67	56	76	19	0	4	62	9	95	38
72	49	41	29	23	83	22	46	91	65
76	73	83	3	41	7	5	56	14	60
21	8	16	97	48	94	12	66	81	41
8	4	48	84	26	22	25	16	33	80
63	14	98	77	93	15	70	32	60	23
31	85	77	13	62	43	91	97	29	75
43	76	25	50	70	1	19	58	82	25
39	79	38	39	58	20	95	23	22	83
90	77	15	11	22	71	34	23	65	26
81	14	72	96	77	22	69	71	15	33
65	31	65	38	33	1	47	79	46	52
17	15	23	73	84	77	50	95	40	55
42	3	9	89	39	44	88	72	45	81
95	10	45	94	4	47	88	22	65	52
22	26	76	89	57	28	23	80	42	60
80	18	97	93	0	4	30	40	43	92

87	11	29	95	80	79	45	1	7	12
84	13	95	96	28	17	83	81	46	55
97	56	86	84	38	81	34	72	24	3
∞	43	30	22	57	62	27	74	13	84
4	∞	12	88	92	31	98	48	28	56
60	80	∞	37	65	48	83	66	91	28
32	50	35	∞	91	24	77	62	96	97
51	97	22	6	∞	31	8	4	6	28
23	79	25	3	96	∞	67	38	3	41
39	11	87	86	96	10	∞	3	94	81
92	48	22	53	11	12	15	∞	53	27
40	25	33	79	42	90	88	77	∞	50
33	27	31	68	80	67	83	79	95	∞
70	42	34	1	45	65	33	16	58	37
75	53	29	69	93	84	73	41	16	30
9	32	22	14	40	58	59	74	93	26
73	24	70	88	54	79	82	71	59	73
2	26	37	11	96	94	8	71	67	3
18	36	70	34	40	73	20	77	15	45
28	5	15	46	73	64	72	51	76	95
76	88	82	33	65	15	47	95	0	92
95	97	43	18	89	5	16	11	3	51
17	77	55	1	59	79	21	55	96	24
14	18	2	90	64	56	27	98	76	32
9	2	43	51	95	48	61	45	53	88
0	34	13	1	15	67	39	21	74	67
88	40	14	62	80	79	24	44	26	65
42	47	55	92	83	86	47	39	86	81
57	14	90	68	7	97	27	66	98	8
45	82	45	33	51	52	4	71	42	65
17	88	51	64	22	89	65	56	46	98
60	61	21	62	67	20	89	57	30	32

92	29	15	57	61	86	67	74	83	32
96	64	54	68	5	50	48	58	11	16
61	73	93	41	47	31	93	47	18	26
66	65	33	90	16	67	78	23	13	48
97	96	51	74	39	3	25	73	8	82
79	67	75	81	55	7	10	15	86	36
29	63	59	56	2	92	20	77	82	90
50	26	63	54	15	98	15	91	93	64
95	43	8	89	53	50	48	65	9	75
54	9	32	8	77	20	29	43	49	73
71	9	50	61	90	90	74	82	89	98

Columns 31 through 40

8	82	63	35	36	11	40	90	10	60
53	19	45	62	64	37	58	50	86	18
20	12	40	7	97	57	1	68	81	86
12	16	34	73	35	11	12	71	74	11
56	68	74	21	81	32	46	34	69	97
58	85	51	3	51	48	74	48	6	49
98	9	78	35	70	13	23	39	76	97
66	62	63	11	78	9	33	84	94	20
72	88	50	45	86	27	89	14	38	42
45	64	18	46	53	61	41	41	67	92
20	3	37	72	36	98	49	45	57	93
0	24	8	99	93	98	63	9	97	13
40	19	24	92	11	1	14	46	9	31
63	83	20	80	91	74	36	2	91	11
48	76	48	78	36	22	80	92	77	62
50	9	57	61	70	33	60	63	17	37
35	39	74	9	49	69	44	3	71	26
31	17	10	3	36	1	57	91	20	92
70	45	32	72	25	8	94	23	23	15

53	86	75	41	80	9	57	74	39	67
51	19	25	34	33	3	87	83	33	89
7	57	62	67	4	91	2	7	29	70
80	42	67	66	93	96	63	18	18	65
17	77	13	1	79	13	10	86	91	55
18	9	33	5	32	4	57	5	10	12
41	36	72	50	29	27	78	84	50	79
27	89	1	53	21	77	52	85	78	54
88	18	1	11	91	71	57	90	8	26
8	15	61	17	61	91	18	47	79	48
0	53	62	11	69	51	1	24	21	88
∞	42	72	83	35	52	10	93	72	51
21	∞	73	48	97	71	23	2	75	57
92	34	∞	48	10	5	28	73	60	60
65	96	92	∞	15	48	30	49	9	17
72	51	1	98	∞	84	2	94	63	72
11	81	18	58	24	∞	26	68	42	82
28	41	71	18	46	69	∞	90	50	68
69	97	26	35	52	79	43	∞	82	85
75	51	23	58	24	48	62	24	∞	75
46	69	93	9	38	88	18	97	65	∞
27	12	75	76	43	75	74	56	11	21
34	58	95	19	30	20	87	72	77	89
59	43	29	2	40	56	71	59	93	32
97	26	91	8	13	53	74	96	68	12
77	78	68	76	42	7	96	46	57	37
30	18	89	92	49	94	12	44	15	10
95	47	67	12	69	32	82	78	79	98
65	22	91	72	50	4	75	57	95	92
54	28	17	81	27	63	10	68	67	82
96	27	3	68	78	64	72	38	66	79
45	92	70	86	36	86	50	95	95	48

61	96	30	67	85	51	27	53	52	67
2	26	79	47	43	90	6	85	43	8
48	48	92	44	96	67	40	95	93	41
33	58	16	63	14	53	64	61	3	87
62	45	11	24	52	2	51	98	13	66
69	57	48	4	30	78	41	34	89	10
72	97	50	48	16	63	42	72	89	19
10	40	21	93	97	32	19	30	72	13
93	11	50	51	20	71	90	29	4	45

Columns 41 through 50

37	23	35	83	72	11	35	43	76	74
72	1	61	99	92	88	79	81	42	30
33	59	87	20	35	46	31	8	32	18
94	43	62	11	10	80	79	85	6	42
32	17	18	81	80	1	32	17	37	1
52	85	17	50	57	43	41	31	37	50
44	33	30	16	5	86	41	82	35	44
65	72	88	33	73	78	38	70	84	58
54	76	5	35	58	94	35	67	2	81
4	91	41	0	33	22	57	77	29	22
25	68	44	36	50	19	41	92	49	71
2	97	67	26	15	47	54	86	36	41
51	50	25	36	98	43	97	62	60	16
11	0	2	25	38	17	43	45	34	74
80	61	41	77	8	24	14	49	63	84
35	38	45	30	70	62	68	15	91	63
57	1	68	12	12	54	76	1	7	3
61	60	41	37	79	40	75	9	88	91
93	4	48	95	12	52	13	97	81	35
82	39	19	58	65	63	28	22	20	21
21	20	75	33	94	30	1	86	98	39

7	25	37	2	95	27	74	81	91	70
43	65	84	81	30	44	78	55	89	88
38	70	45	85	57	45	69	62	36	10
54	74	75	86	57	6	63	14	52	82
36	63	39	54	45	55	84	10	10	61
0	76	24	66	11	6	17	22	25	51
12	73	95	51	18	89	43	90	9	27
81	6	39	95	48	37	57	93	98	59
47	5	58	35	4	12	64	22	52	52
63	74	43	55	43	7	82	73	65	13
12	97	97	45	7	3	24	68	64	83
8	38	72	80	23	57	35	87	4	25
3	85	5	70	59	28	59	60	4	67
87	5	74	77	23	43	44	54	52	25
80	15	42	77	61	49	53	14	7	33
8	44	36	66	8	72	54	35	1	97
91	62	46	80	11	82	28	93	99	60
28	15	16	56	29	51	43	62	5	51
31	19	38	65	79	59	27	66	52	10
∞	99	67	95	77	24	81	96	84	62
2	∞	24	53	67	67	38	19	69	46
15	21	∞	60	79	35	49	71	84	77
46	32	2	∞	94	89	28	8	23	36
52	96	33	59	∞	20	60	59	32	89
69	67	10	22	94	∞	28	63	56	64
86	81	94	56	11	64	∞	9	98	74
75	67	12	56	55	82	88	∞	19	23
84	12	22	64	88	77	18	70	∞	39
62	1	91	62	25	73	46	20	72	∞
4	92	65	17	64	47	50	25	97	58
51	38	50	82	67	37	98	82	29	2
30	9	31	51	73	47	75	8	24	20

54	66	62	42	70	20	11	67	74	73
85	39	54	88	40	24	39	84	55	9
65	7	93	55	58	14	36	53	24	69
42	18	1	5	67	47	23	69	86	94
69	99	7	41	66	24	3	61	67	92
2	39	9	54	1	97	48	63	8	60
21	31	92	56	66	67	85	57	78	95

Columns 51 through 60

94	92	46	31	39	13	15	93	63	47
13	95	45	57	83	71	35	46	90	35
74	74	20	76	37	81	37	45	37	55
74	97	10	35	73	35	56	67	81	70
7	27	41	71	75	16	64	41	71	72
6	93	52	88	73	9	70	51	72	88
56	17	57	71	39	82	33	64	19	35
5	84	58	51	47	74	16	7	29	16
69	98	48	58	92	14	14	71	77	18
54	86	45	71	33	41	19	82	86	61
98	63	2	51	40	25	35	1	22	39
40	53	1	85	37	19	63	29	93	56
46	83	6	28	62	20	87	15	85	95
13	59	60	13	40	3	72	47	90	24
93	78	33	20	19	16	13	63	91	18
28	66	74	11	34	87	6	73	25	48
25	31	90	76	95	77	54	60	24	14
62	47	89	38	27	48	33	2	47	78
15	99	56	48	55	87	34	62	56	8
79	8	80	6	37	63	75	56	52	25
31	2	96	83	86	15	6	75	62	43
56	54	60	56	62	63	65	42	11	98
75	8	5	27	38	6	78	79	27	18

78	98	7	15	75	4	64	41	12	50
1	35	29	27	95	93	40	93	77	85
29	62	45	66	10	57	0	73	49	49
12	5	62	60	8	45	49	62	41	24
26	28	10	92	82	1	77	97	16	85
94	18	52	44	76	17	93	31	76	77
79	66	34	31	1	38	48	6	28	46
74	21	54	43	5	20	87	98	40	71
20	85	46	5	55	21	80	51	75	9
56	18	20	75	7	78	70	19	4	92
13	85	15	17	9	70	10	53	68	75
99	30	52	56	96	22	13	49	27	81
77	67	16	9	85	61	79	21	53	58
33	52	43	72	79	76	24	25	43	67
51	60	39	21	81	80	38	68	93	70
32	1	25	2	4	87	26	54	59	58
68	17	77	0	54	90	36	22	75	92
20	71	54	39	59	69	41	85	44	61
5	94	59	26	81	74	82	40	64	28
53	16	13	72	75	9	75	31	5	16
82	15	89	22	52	88	66	13	2	74
51	40	28	48	78	27	87	35	15	88
55	88	31	60	70	83	78	17	9	10
54	42	78	80	62	32	30	5	60	22
14	31	93	57	22	45	55	52	28	30
78	61	35	26	61	19	44	5	86	51
73	14	2	47	74	31	4	5	10	23
∞	29	9	4	34	8	28	59	5	48
75	∞	23	78	45	71	68	72	9	40
48	76	∞	14	48	57	21	41	33	46
1	99	97	∞	5	42	12	92	60	90
10	46	87	38	∞	11	20	71	63	47

45	85	89	83	14	∞	∞	96	78	91
63	66	72	75	20	∞	∞	99	28	56
22	17	80	8	35	25	23	∞	∞	∞
77	40	62	82	77	32	18	∞	∞	∞
81	73	97	82	66	98	54	∞	∞	∞

When considering the Benders'algorithm for solving this example, the process is explained as follows.

Iteration 1:

Step 1: Initialization:

Let set $v\,(Z) = 0$, select $v\,(Z = [Z_{(1)}, Z_{(2)},\ldots, Z_{(60)}]^{T})$, set $j = 1$ and set $k = 1$

For this example, from node No.1 to node No. 55 are the normal jobs. Node No. 56 and No. 57 are the component of a 2 alternatives job. Node No. 58, No. 59 and No. 60 are the component of a 3 alternatives job. Therefore, all variables of $Z_{(1)}$ to $Z_{(55)}$ equal to 1 and the rest of them are $Z_{(i)} \in \{0, 1\}$, i = 56, 57, …, 60. The first $v\,(Z = [Z_{(1)}, Z_{(2)},\ldots, Z_{(60)}]^{T})$ is:

$v\,(Z^{1} = [1_{(1)}, 1_{(2)},\ldots, 1_{(56)}, 1_{(56)}, 0_{(57)}, 1_{(58)}, 0_{(59)}, 0_{(60)}]^{T})$

Step 2: Solve the Benders'subproblem:

The first the Benders'subproblem of this example is:

Maximize $v_1\,(Z^{1}) = \text{Maximize}\ \{(b - \boldsymbol{B}\,Z^{1})^{T} u^{1} \mid \boldsymbol{A}^{T} u^{1} \le c,\ u^{1} \ge 0\}$,

is solved. The maximum occurs at the vector of extreme point u^{1} and the maximum value of $v_1\,(Z^{1}) = 115$.

Step 3: Stopping Criterion:

Now the value of $v\,(Z) = 0$, $v_1\,(Z^{1}) = 115 \neq v\,(Z)$ then go to step 4

Step 4: Improve the approximations function:

Using the dual extreme point u^1 generates the approximations function ($v(Z)$), with the Benders'cut, for the Benders'partial master problem of the iteration 1. The Benders'partial master problem is:

Minimize $v(Z) = d^T Z +$ maximize $\{ [(b - \boldsymbol{B}Z)^T u^1]_1 \}$

Subject to $Z \in \boldsymbol{Z}$

The Benders'cut of the iteration 1 is $[(b - \boldsymbol{B}Z)^T u^1]_1$ that is:

$[u^1_{(1)} Z_{(1)} + u^1_{(2)} Z_{(2)} + \ldots + u^1_{(60)} Z_{(60)}]_1$

Because a vector d is a zero vector, the Benders'partial master problem for iteration 1 is:

Minimize $v(Z) =$ maximize $\{ [u^1_{(1)} Z_{(1)} + u^1_{(2)} Z_{(2)} + \ldots + u^1_{(60)} Z_{(60)}]_1 \}$

Subject to $Z \in \boldsymbol{Z}$

Step 5: Solve the Benders'partial master problem:

Update $j = 2$, $k = 2$ and the value of $v(Z)$ from solving the Benders'partial master problem = 143 with is new vector $Z =$

$Z^2 = [1_{(1)}, 1_{(2)}, \ldots, 1_{(56)}, 1_{(56)}, 0_{(57)}, 0_{(58)}, 1_{(59)}, 0_{(60)}]^T$

Iteration 2:

Step 2: Solve the Benders'subproblem:

The Benders'subproblem of iteration 2 is:

Maximize $v_2(Z^2) =$ Maximize $\{(b - \boldsymbol{B}Z^2)^T u^2 \mid \boldsymbol{A}^T u^2 \le c,\ u^2 \ge 0\}$,

is solved. The maximum occurs at the vector of extreme point u^2 and the maximum value of $v_2(Z^2) = 153$.

Step 3: Stopping Criterion:

Now the current value of $v(Z) = 143$. Because $v_2(Z^2) = 153 \neq v(Z)$, not terminate, then go to step 4

Step 4: Improve the approximations function:

Using the dual extreme point u^2 generates the approximations function ($v(Z)$), with the Benders'cut, for the Benders'partial master problem of the iteration 2. The Benders'partial master problem is:

Minimize $v(Z) = d^T Z +$ maximize $\{[(b - \boldsymbol{B}Z)^T u^1]_1, [(b - \boldsymbol{B}Z)^T u^2]_2\}$

Subject to $Z \in \boldsymbol{Z}$

The Benders'cut of the iteration 2 is $[(b - \boldsymbol{B}Z)^T u^2]_2$ that is:

$[u^2_{(1)}Z_{(1)} + u^2_{(2)}Z_{(2)} + \ldots + u^2_{(60)}Z_{(60)}]_2$

Because a vector d is a zero vector, the Benders'partial master problem for iteration 2 is:

Minimize $v(Z) =$ maximize $\{ [u^1_{(1)}Z_{(1)} + u^1_{(2)}Z_{(2)} + \ldots + u^1_{(60)}Z_{(60)}]_1,$
$[u^2_{(1)}Z_{(1)} + u^2_{(2)}Z_{(2)} + \ldots + u^2_{(60)}Z_{(60)}]_2\}$

Subject to $Z \in \boldsymbol{Z}$

Step 5: Solve the Benders'partial master problem:

Update $j = 3$, $k = 3$ and the value of $v(Z)$ from solving the Benders'partial master problem = 149 with is new vector $Z =$

$Z^3 = [1_{(1)}, 1_{(2)}, \ldots, 1_{(56)}, 1_{(56)}, 0_{(57)}, 0_{(58)}, 0_{(59)}, 1_{(60)}]^T$

Iteration 3:

Step 2: Solve the Benders'subproblem:

The Benders'subproblem of iteration 3 is:

Maximize $v_3(Z^3) =$ Maximize $\{(b - \boldsymbol{B}Z^3)^T u^3 \mid \boldsymbol{A}^T u^2 \le c, u^3 \ge 0\}$,

is solved. The maximum occurs at the vector of extreme point u^3 and the maximum value of $v_3(Z^3) = 159$.

Step 3: Stopping Criterion:

Now the current value of $v(Z) = 149$. Because $v_3(Z^3) = 159 \neq v(Z)$, not terminate, then go to step 4

Step 4: Improve the approximations function:

Using the dual extreme point u^3 generates the approximations function ($v(Z)$), with the Benders'cut, for the Benders'partial master problem of the iteration 3. The Benders'partial master problem is:

Minimize $v(Z) = d^T Z +$ maximize $\{[(b - \boldsymbol{B}Z)^T u^1]_1, [(b - \boldsymbol{B}Z)^T u^2]_2,$
$[(b - \boldsymbol{B}Z)^T u^3]_3\}$

Subject to $Z \in \boldsymbol{Z}$

The Benders'cut of the iteration 3 is $[(b - \boldsymbol{B}Z)^T u^3]_3$ that is:

$[u^3_{(1)}Z_{(1)} + u^3_{(2)}Z_{(2)} + \ldots + u^3_{(60)}Z_{(60)}]_3$

Because a vector d is a zero vector, the Benders'partial master problem for iteration 3 is:

Minimize $v(Z) =$ maximize $\{ [u^1_{(1)}Z_{(1)} + u^1_{(2)}Z_{(2)} + \ldots + u^1_{(60)}Z_{(60)}]_1,$
$[u^2_{(1)}Z_{(1)} + u^2_{(2)}Z_{(2)} + \ldots + u^2_{(60)}Z_{(60)}]_2$
$[u^3_{(1)}Z_{(1)} + u^3_{(2)}Z_{(2)} + \ldots + u^3_{(60)}Z_{(60)}]_3\}$

Subject to $Z \in \boldsymbol{Z}$

Step 5: Solve the Benders'partial master problem:

Update $j = 4$, $k = 4$ and the value of $v(Z)$ from solving the Benders'partial master problem = 150 with is new vector $Z =$

$Z^4 = [1_{(1)}, 1_{(2)}, \ldots, 1_{(56)}, 1_{(56)}, 0_{(57)}, 0_{(58)}, 1_{(59)}, 0_{(60)}]^T$

Iteration 4:

Step 2: Solve the Benders'subproblem:

The Benders'subproblem of iteration 4 is:

Maximize $v_4(Z^4) =$ Maximize $\{(b - \boldsymbol{B}Z^4)^T u^4 \mid \boldsymbol{A}^T u^4 \leq c, u^4 \geq 0\}$,

is solved. The maximum occurs at the vector of extreme point u^3 and the maximum value of $v_4(Z^4) = 152$.

Step 3: Stopping Criterion:

Now the current value of $v(Z) = 150$. Because $v_4(Z^4) = 152 \neq v(Z)$, not terminate, then go to step 4

Step 4: Improve the approximations function:

Using the dual extreme point u^4 generates an approximations function ($v(Z)$), with the Benders'cut, for the Benders'partial master problem of the iteration 4. The Benders'partial master problem is:

Minimize $v(Z) = d^T Z +$ maximize $\{[(b - \boldsymbol{B}Z)^T u^1]_1, [(b - \boldsymbol{B}Z)^T u^2]_2,$
$[(b - \boldsymbol{B}Z)^T u^3]_3, [(b - \boldsymbol{B}Z)^T u^4]_4\}$

Subject to $Z \in \boldsymbol{Z}$

The Benders'cut of the iteration 4 is $[(b - \boldsymbol{B}Z)^T u^4]_4$ that is:

$[u^4_{(1)}Z_{(1)} + u^4_{(2)}Z_{(2)} + \ldots + u^4_{(60)}Z_{(60)}]_4$

Because a vector d is a zero vector, the Benders'partial master problem for iteration 4 is:

Minimize $v(Z) =$ maximize $\{ [u^1_{(1)}Z_{(1)} + u^1_{(2)}Z_{(2)} + \ldots + u^1_{(60)}Z_{(60)}]_1,$
$[u^2_{(1)}Z_{(1)} + u^2_{(2)}Z_{(2)} + \ldots + u^2_{(60)}Z_{(60)}]_2$
$[u^3_{(1)}Z_{(1)} + u^3_{(2)}Z_{(2)} + \ldots + u^3_{(60)}Z_{(60)}]_3$
$[u^4_{(1)}Z_{(1)} + u^4_{(2)}Z_{(2)} + \ldots + u^4_{(60)}Z_{(60)}]_4\}$

Subject to $Z \in \boldsymbol{Z}$

Step 5: Solve the Benders'partial master problem:

Update $j = 5$, $k = 5$ and the value of $v(Z)$ from solving the Benders'partial master problem = 152 with is new vector $Z =$

$Z^5 = [1_{(1)}, 1_{(2)}, \ldots, 1_{(56)}, 1_{(56)}, 0_{(57)}, 0_{(58)}, 1_{(59)}, 0_{(60)}]^T$

Iteration 5:

Step 2: Solve the Benders'subproblem:

The Benders'subproblem of iteration 4 is:

Maximize $v_5(Z^5) =$ Maximize $\{(b - \boldsymbol{B}Z^5)^T u^5 \mid \boldsymbol{A}^T u^5 \leq c, u^5 \geq 0\}$,

is solved. The maximum occurs at the vector of extreme point u^5 and the maximum value of $v_5(Z^5) = 152$.

Step 3: Stopping Criterion:

Now the current value of v (Z) = 152. Because v_5 (Z^5) =152, stop

This example can be solved by using the Benders'decomposition as above.

2. The example of the application of the AGVsp-P/D

Refer to the problem of messenger scheduling, one of the proposed application of the AGVsp-P/D. This problem occurs repeatedly in the decision making process of the messenger of any agent such as the government agent, the financial business agent, the private postal service agent and etc.

For example, the messenger of the engineering faculty has a job list for one round as follows.

1. get the document at the Faculty office(ENG) and deliver at ME department
2. pick up the document at Financial office(FIN) and deliver to EE department
3. pick up the printed sheets at Copy shop(CPY) and deliver at Library(LBY)
4. pick up the document at IE department and deliver to graduated school(GRD)
5. buy some cashier checks at either TBank or Abank and go to pay at Computer training center (COM)
6. buy some stamps at the post office or shop1 or shop2 and deliver at the faculty

The messenger job list for one round can be shown as follows.

Job No.	Pick up Department	Delivery Department
1.	ENG	ME
2.	FIN	EE
3.	CPY	LBY
4.	IE	GRD
5.	TBank or ABank	COM
6.	Post or Shop1 or Shop2	ENG

Suppose the distances among all of locations are known and transformed into a form of the TSP distance table, by same method of table 5 that is explained previously. The distance table of this problem is shown as follows.

To / From Job No.	Job No. (*h*) / Alternative (job *i*, alt. *a*)		***1***	***2***	***3***	***4***	***5***		***6***		
			1.1	***2.1***	***3.1***	***4.1***	***5.1***	***5.2***	***6.1***	***6.2***	***6.3***
		n	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
1	***1.1***	***1***	∞	31	25	43	83	18	50	45	32
2	***2.1***	***2***	38	∞	75	87	45	79	13	6	37
3	***3.1***	***3***	37	48	∞	34	25	58	52	16	48
4	***4.1***	***4***	49	83	80	∞	60	56	61	10	16
	5.1	***5***	41	55	27	78	∞	∞	58	55	20
5	***5.2***	***6***	9	92	26	20	∞	∞	54	9	63
	6.1	***7***	44	7	37	25	92	62	∞	∞	∞
6	***6.2***	***8***	43	97	9	52	54	28	∞	∞	∞
	6.3	***9***	83	14	17	7	82	13	∞	∞	∞

This simulated problem is programmed in MATLAB 7.0 and solved by AGVsp-P/D model. The result is the schedule of the messenger that is the sequence of 1 - 6 - 4 - 8 - 3 - 2 - 1 according to number of nodes *n* with the total distance of 143 units. The result of solving this problem using the AGVsp-P/D model on MATLAB 7.0 is shown as follows.

cnew =

∞	31	25	43	83	18	50	45	32
38	∞	75	87	45	79	13	6	37
37	48	∞	34	25	58	52	16	48
49	83	80	∞	60	56	61	10	16
41	55	27	78	∞	∞	58	55	20
9	92	26	20	∞	∞	54	9	63
44	7	37	25	92	62	∞	∞	∞
43	97	9	52	54	28	∞	∞	∞
83	14	17	7	82	13	∞	∞	∞

Optimization terminated.

Xnod =

(2,1)	1
(3,2)	1
(8,3)	1
(6,4)	1
(1,6)	1
(4,8)	1

fval =

143

The Optimal TSP Tour is
tour =

1
6
4
8
3
2
1

subtour =

0

Elapsed time is 0.384500 seconds.

CURRICULUM VITAE

NAME : Mr. Chatpun Khamyat

BIRTH DATE : August 19, 1978

BIRTH PLACE : Bangkok, Thailand

EDUCATION :

YEAR	**INSTITUTE**	**DEGREE/DEPLOMA**
2000	Kasetsart Univ.	B.Eng (Industrial)
2002	Kasetsart Univ.	M.Eng (Industrial)

POSITION/TITLE : Lecturer

WORK PLACE : Faculty of Engineering, Kasetsart University

SCHOLARSHIP/AWARDS : Thai Government Scholarship 2000-2005