1-1-2002

Performance measurement and operations control in advanced public transit systems

Nader Azizi
Ryerson University

PERFORMANCE MEASUREMENT AND OPERATIONS CONTROL IN ADVANCED PUBLIC TRANSIT SYSTEMS

by

Nader Azizi

A thesis

Presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Applied Science

in

Mechanical Engineering

Toronto, Ontario, Canada

I hereby declare that I am the sole author of this thesis.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Nader Azizi

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Nader Azizi

Ryerson University requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

PERFORMANCE MEASUREMENT AND OPERATIONS CONTROL IN ADVANCED PUBLIC TRANSIT SYSTEMS

A thesis presented to Ryerson University in partial fulfillment of the requirements for the degree of Master of Applied Science in Mechanical Engineering, *Nader Azizi*, 2003

ABSTRACT

In most major cities, levels of traffic congestion are rising along with their associated problems such as travel delays and pollution. While any increase in public transit rider-ship could reduce the level of traffic congestion and related costs, most transit agencies are not able to expand their existing services because of fiscal and physical constraints. As a result, a growing interest has been developing recently to maximize the transit system efficiency and productivity using new emerging technologies.

Recently, the emergence of new technologies such as automatic vehicle location (AVL) and global positioning systems (GPS) has facilitated the design of computer-based real-time decision support systems for public transits. These technologies could significantly help transit agencies improve their operations monitoring and control. In the context of public transit systems, operations monitoring refers to real-time service performance measure and problems detection, and control refers to implementing real time control actions to remedy those problems.

This thesis presents a new approach for operations monitoring and control in public transit systems with real-time information. First, an integrated model that combines both headway-based and schedule-based services is presented. To measure the headway or schedule adherence, the model uses predicted arrival times of vehicles at downstream stops. This feature allows the operational managers to avoid major service interruptions by proactively taking necessary corrective actions. Transit agencies have used and continue to use real-time control strategies to improve quality of their services. These strategies are employed by

inspectors at various points along a route to remedy the problems as they occur. Practice shows that it is difficult to apply such strategies effectively without real-time information. In the second part of this thesis, a mathematical model for holding control strategy with real-time information is described. The proposed model aims at minimization of the total passengers waiting time and considers both cases of overcrowded and underutilized services. Due to complexity of the holding problem, several metaheuristics are proposed and tested. Among all intelligent search algorithms, a new version of simulated annealing algorithm is proposed to solve the real-time holding control model.

ACKNOWLEDGEMENTS

I am lucky to have supportive and encouraging thesis supervisors. My supervisor Professor Saeed Zolfaghari is a dedicated and an honourable educator. His thoughtful comments and meticulous reading have greatly enhanced the quality of this work. Professor Mohamad Y. Jaber, my co-supervisor, asked many insightful questions and provided great assistance in developing and refining this work.

Special thanks to Sindy Wu, Erika Lopez, and Nelson da Silva who provided friendship and encouragement as the work progressed.

Finally, I would like to thank my lovely friends Shina and Siavash whose support and sympathy sustain me through this undertaking.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

Parameters and constants

H	planned headway
f_s	weight value of schedule base performance
f_h	weight value of headway base performance
T^h	prediction horizon
t^{now}	present time
r_k	passenger arrival rate at stop k
C_j	dwell time function parameter
Q_k	passenger alighting fraction at stop k
A_k^f	passenger alighting fraction at stop k that use front door for alighting
A_k^r	passenger alighting fraction at stop k that use rear door for alighting
R_k	running time from stop k-1 to stop k, including acceleration and deceleration
L^{max}	passenger capacity of bus
M_j	sufficiently large numbers
η	expected value of running times on a route
φ	expected value of bus dwell time at stops

Sets

K	set of stations on the route **K**={1,2,3,…,N/2,…,N}
I	set of buses operating on the route **I**={1,2,3,…,i,…,I}

Indices

i	index of buses
k	index of stops
l^i	index of the last stop passed by bus i
l^k	index of the last departed bus from stop k
k_{fst}	index of the first stop in the horizon of bus i

Variables

S_{ik}	scheduled arrival time of bus i at stop k
ΔS_{ik}	schedule adherence of bus i at stop k
F_{ik}	predicted arrival time of bus i at stop k
D_{ik}	running time delay of bus i over the route segment k
h_{ik}	expected headway of bus i at stop k
$\tilde{N}_i$	total number of the downstream stops for which the prediction arrival times of bus i is available
$\tilde{I}_k$	total number of buses for which prediction arrival times at stop k are available
d_{ik}	departure time of bus i from stop k
l_{ik}	load in bus i departing stop k
p_{ik}	passengers left behind by bus i at stop k
D_{ik}	demand for bus i at stop k
v_{ik}	1, if bus i is loaded to capacity when it depart stop k; 0, otherwise
S^u_{ik}	dwell time of bus i at stop k when the passenger load is below the bus capacity
S^c_{ik}	dwell time of bus i at stop k when bus i is loaded to capacity
$\tilde{N}_i$	total number of stops in the horizon of bus i

CHAPTER 1: INTRODUCTION & LITERATURE REVIEW

1.1 Research Background

In transit systems, public or private, providing a reliable service is the main objective of operation managers. Dependable schedules, minimum passengers waiting time, and short travel times are some instances of reliable services. Service reliability can be measured by a number of indicators such as arrival delay, headway delay, and passengers' waiting time. The use of such indicators mostly depends on the frequency of the service.

Most of prior studies focused either on headway- or schedule-based approach to measure the performance of a transit system. Schedule-based approach is aimed at maintaining the punctuality of the service, while headway-based approach is aimed at maintaining the elapsed time between successive services. The choice of headway or schedule-based strategies is clear when the frequency of the service is either too high or too low. However, this choice may not be very obvious when the service frequency is of medium range. An integrated model that could address both headway- and schedule-based services jointly may alleviate this problem.

Public transit systems are subject to irregularities due to randomness in passenger demand, traffic congestion, weather conditions and incidents along the route. Transit agencies usually employ real-time control strategies such as holding, deadheading, and short turning to remedy the specific problems as they occur. Particularly, the aim of real-time operations

control is to minimize passenger cost in a route with given schedule and fleet size. Traditionally, in transit systems control decisions are made by field supervisors at various points along the route with the only information of vehicles that have already passed at the point. In such decision making process, since inspector can only observe deviations at local points, taking any action based on those observations may have negative impact in future and elsewhere in the route. Where the effectiveness of an inspectors' action is highly depended on his/her personal judgment and experience.

In the last decade, many public transportation agencies have begun to use advanced technologies, such as AVL systems, to improve the quality of their services. These systems have the potential to provide accurate real-time information that supports operations' control. While the availability of real-time information has increased, few studies have addressed theory and implementation of real-time control strategies. Thus, there are two objectives to be attained in this thesis. First, to develop an integrated model that addresses simultaneously headway and schedule based approaches to measure the performance of a transit system. Second, objective is to develop a mathematical model for real-time holding strategy applicable in urban bus transit systems.

1.2 Overview of Service Performance Measurement

Reliability problems are of concern to users and operators of transit systems. Service that is not on time affects passengers in terms of increased wait time, travel time uncertainty and a general dissatisfaction with the system. An unreliable service ultimately leads to lost public support when passengers leave transit for alternative resources (Abkowitz and Tozzi 1987, Clotfelter 1993).

In transit systems, unreliable service caused by a number of factors that could be classified as either Endogenous, or Exogenous factors. Endogenous or internal factors include passenger demand variation, route configuration, stop spacing, schedule accuracy, and driver's behaviour. Exogenous factors include traffic congestion and accidents, traffic signalization, on-street parking, and weather conditions (Woodhull 1987).

There are a number of alternative indicators by which service reliability could be measured. Employing of these indicators depends on the frequency of the service. Service frequency is usually defined in terms of *headway*. Headway could be defined as the time interval between successive vehicle on a route and in the same direction as they pass a particular point on that route.

The first indicator of service reliability is arrival delay. This indicator measures the schedule adherence for a given vehicle at a particular stop. Traditionally, transit agencies have used on-time performance (OTP) as a measure of schedule adherence. OTP is a discrete measure that is particularly useful for evaluating the reliability of a transit system from the perspective of the service provider. OTP is generally expressed as the percentage of vehicles that depart on-time from a given location within a predetermined time segment. The second indicator of service reliability is headway delay. Headway delay could be defined as the actual headway less planed headway. This indicator measures deviations in time spacing between vehicles. A negative value of headway delay indicates that the vehicle is early and close to the preceding vehicle. Likewise, a positive value of headway delay implies that the vehicle is late and resulted in an extended headway from the previous vehicle. The third indicator, running time delay, viewed as an important indicator of service performance. Running time could be defined as the elapsed time it takes a vehicle to move from one location to another. Whereas running time delay indicates how well a vehicle is moving along a route. A positive value of running time delay specifies that the vehicle is moving along the route with difficulty, and a negative value indicates a faster than expected traveling between two locations. The running time delay is an appropriate indicator of service reliability for long routes with numerous signalized intersections and irregular ridings (Sterman and Schofer 1976). The fourth, and last, indicator is the estimated excess waiting time experienced by passengers as a result of service irregularity. In addition to the extended waiting time imposed on passengers, a high value of this indicator implies a greater likelihood that passengers would ignore the pre-announced schedule and, most probably, not attempt to coordinate their arrival time with the schedules (Turnquist 1978).

Several studies have addressed the application of headway- and schedule-based approaches in bus transit services. Turnquist (1978) suggested that a schedule-based control approach could be particularly useful on suburban routes, or in other instances where headways are quite large. He further added that the effectiveness of headway-based control approach depends on the identification of an appropriate control point along the route. Turnquist (1978) also suggested that a control point should be located as early along the bus route as possible.

Abkowitz et al. (1986) found that headway-based control approach would be suitable for routes that operate with short and uniform headways. When headways are short and uniform, it is assumed that passengers arrive more randomly at stops and that they are mainly concerned with the headway rather than the schedule. Abkowitz et al. (1986) concurred with Turnquist (1978) that schedule-based control would be suitable for routes characterized by long and/or uneven headways.

Abkowitz and Lepofsky (1990) presented the results of implementing real-time headway-based reliability control along several bus routes. Their procedure involves holding buses at a control point on the route until a prescribed minimum headway is achieved. They concluded that headway-based control approaches were applicable to high-frequency transit routes with sufficiently short headways, where passengers arrive randomly at bus stops without consulting a schedule.

Seneviratne (1989,1990) developed a simulation model to examine the service performance over time. In this model, the control points on the route are optimized according to a specified criterion: the maximum permitted 60 seconds of headway standard deviation. At the end of each simulation set, the point on the route where headway standard deviation exceeds 60 seconds is identified, and a control point is placed at the preceding stop.

Chien and Ding (1999) presented a dynamic headway control model to maintain desired headway between any pair of consecutive vehicles for high frequency light rail transit systems. Their model focuses on adjusting vehicle speed in real-time based on its optimal

arrival time at the next station. Dessouky et al. (1999) developed a performance measurement model based on total passenger delay and number of passengers missing their connections for bus lines with large headways.

The above surveyed works either focused on headway-based or schedule-based approaches to measure performance of a transit system.

When the headway of a transit service is either too low or too high, selecting a strategy is clear. However, it may not be clear which strategy to implement when the headway is neither; i.e., of medium range. There is a general concession among researchers and practitioners that headway of less than 10 minutes could be denoted as high-frequency services (Abkowitz and Engelstein 1984, Abkowitz et al. 1986). Contrary, a threshold for headway in low frequency has not been clearly specified. According to a report published by the Transit Cooperative Research Program (TCRP), a headway of 10 minutes would be considered a high frequency service, while a service with an interarrival time of 60 minutes would be considered as a low frequency service (Pinejames et al. 1998). This may be why practitioners in transit agencies have hard time deciding which of the two available control strategies; headway-based or schedule-based, to implement.

1.3 Overview of Control Strategies

In public transit, reliability is an important measure of service quality that needs continuous attention and improvement. This brings upfront the issue of control strategies. These strategies are aimed at enhancing the reliability of transit services. Abkowitz (1978) suggests that there are three basic methods to improve transit service reliability, categorized as priority, control, and operational. Priority methods involve the special treatment of transit vehicles apart from general vehicular traffic. Examples of this type of strategy are exclusive bus lanes and traffic signal prioritization. Operational methods take place over a longer period of time and include schedule modification, route restructuring, and driver training. Control methods take place in real-time and include vehicle holding, short-turning, stop skipping, and speed modification. Turnquist and Blume (1980) make a distinction between planning control and real-time control strategies. Planning control strategies are long-term

and involve strategies such as re-structuring the routes and schedules. Real-time control strategies are short-term and involve strategies such as adding additional buses and short turning. Real-time control strategies are designed to remedy specific problems as they occur.

Eberlein et al. (1999) divided real-time control strategies into three categories: station controls; inter-station controls, and others. The first category, station controls, includes holding strategy, stop-skipping strategy, and short-turn strategy. The second category, inter-station controls, includes traffic signal pre-emption, and etc. the third category includes such strategies as deadheading, expressing, and adding reserve vehicles.

Among these strategies, holding control has been used frequently by transit agenises to improve the quality of the system. In this thesis, we first briefly review types of control strategies that have been addressed in literature then we focus on the holding strategy.

1.3.1 Stop skipping

Stop skipping is a strategy that involves skipping one or more stops as a vehicle moves along a route. Stop skipping reduces travel time of the vehicle of interest. This would reduce waiting times of passengers on board of a vehicle, and those in downstream stops. However, this might increase the waiting time of passengers at skipped stops, and those who are requested by the driver to alight at a given stop and wait for the next vehicle in service (Wilson et al.1992). For example, consider a route in which three buses, A, B, and C are operating, such that bus B is located on the same route between buses A and C. Then, an ideal scenario for stop skipping is to have a long preceding headway (between buses B and C), a short following headway (between buses A and B), and high passenger demand beyond the segment where skipping is implemented (at stops between buses B and C).

1.3.2 Short turning

Short-turning strategy involves turning a vehicle around before it reaches the route terminus. This strategy is usually adopted when the headway variance or passenger waiting time in the

opposite direction are to be reduced. Referring to the earlier example, an ideal scenario for short turning is to select a bus with a light passenger load, a low preceding headway (between buses B and C), and a low following headway (between buses A and B). Short-turning strategy reduces passengers waiting time in the opposite direction. This benefit is at the expense of increasing waiting time for passengers on board the vehicle who are requested by the driver to alight and transfer to the subsequent vehicle.

1.3.3 Signal priority

Signal priority mechanism reduces vehicle delays at signalled intersections. This mechanism either changes the phase of a signal to green, or it extends the duration of the green phase when a vehicle approaches an intersection. In contrast to holding, which usually delays passengers and increases running time; signal prioritization reduces running times and decreases delay for passengers on-board of buses. Although signal control could reduce waiting time of passengers along the route, it might negatively affect traffic flow at signalled intersections.

1.3.4 Deadheading and expressing

A deadheaded vehicle is a vehicle that usually departs empty from a dispatching terminal point, A, to a designated stop, E, skipping stops in located between A and E. This strategy could reduce both the waiting time of passengers at the stops beyond the skipped ones, and the headway irregularity in the system. However, deadheading comes at a cost (additional waiting time) for those passengers who were not served (skipped) by the deadheaded vehicle (Eberlein et al. 1998). Like deadheading vehicles, express vehicles might skip stops on a route. However, an express vehicle could be dispatched from any stop, and might not necessarily run empty.

1.3.5 Reserve vehicle

Reserve vehicles are useful when there are unexpected interruptions in the system (e.g., vehicle breakdown). In such circumstances, it may be desirable to add a reserve bus to fill the existing gap in demand (i.e., passengers waiting at stops). However, adding a reserve vehicle to a service could reduce passenger-waiting time and prevent headway irregularity in the system, it might inflict additional costs on transit agencies.

1.3.6 Holding control strategies

Holding control strategies delay vehicles deliberately when they are ahead of their schedules. If adopted, holding control strategies could significantly reduce headway variances and the average waiting time of passengers. Despite of these advantages, holding control strategies could increase in-vehicle time of passengers and vehicle travel time. Holding has attracted the most attention in the bus transportation literature. This maybe attributed to its ease to apply. Since early 1970's, many researchers have attempted to develop mathematical or simulation models for finding optimal holding policies. Holding control strategies could be classified into two types. The first type use threshold-base control models to hold buses at control stops to correct the headway between consecutive buses. The second type use mathematical programming models with holding times as decision variables and passenger-waiting time as the cost function to be minimized.

1.3.6.1 Threshold based control models

In literature, there are several studies that tackled models for threshold-base control. Among the earliest studies was that of Osuna and Newell (1972) who presented an analytic method that determines the optimal strategy to hold buses at a terminal point. In their analysis, the arrival rate of passengers was assumed to be uniformly distributed, and with no bus bunching, for a system of two buses. However, Osuna and Newell (1972) considered the case of a single bus operating along the route. Barnett (1974) consider a route with two terminals and one control stop in- between. Unlike, the work of Osuna and Newell (1972), Barnett

(1974) developed a two-point, discrete, approximate distribution of vehicle delay, with the intention of reducing the complexity of the problem discussed. In his model, Barnett (1974) structured the cost function as the sum of the weighted passengers' waiting time and on-board passengers delay time at the control stop with the threshold-holding time, x, as a decision variable. Koffman et al. (1978) developed a simulation model to analyze a one-way bus route. They tested several control strategies for buses in real-time. For holding strategy, two threshold values, 75% and 65% of desirable headway, were analyzed. Their results indicated that higher threshold values correspond to lower waiting times and higher bus travel time. Abkowitz and Engelstein (1984) found that the optimal control point, defined as a stop where a headway deviation is measured, is sensitive to the ratio of passengers on board to those waiting at downstream stops. Abkowitz et al. (1986) developed an empirical headway deviation function using Monte Carlo simulation to estimate the waiting time of passengers. In their model, the cost function measures the waiting time of passengers along a route with threshold value of holding and location of control point as decision variable. The result of Abkowitz et al. (1986) indicate that using headway-based holding control could reduce the wait times of passengers by 5 to 15 percent. They recommended that a control point be placed prior to high demand stops.

In summary, the above survey works has the following three features in common:

a) These works presented models based on threshold-based holding control strategies. The purpose of these models is to determine the optimal fixed threshold headway, and the optimal position of a control point along a route.

b) Perhaps the lack of real-time information what led these researchers to assume forms of probability density functions when investigating headways and/or running times in their models.

c) Most of the above developed models are simulation based. This might be attributed to the complexity of the problems addressed.

1.3.6.2 Mathematical programming models

The emergence of technologies such as *automatic vehicle location* (AVL) and *global positioning systems* (GPS) facilitated the design of computer-based real-time decision support systems for public transit.

Eberlein et al. (1999) presented the first research on real-time routine control problems. Three types of control strategies were studied. These strategies were holding, deadheading, and expressing. They considered a one-way loop transit network of two terminals and number of intermediate stations. On this route, vehicles are assumed to operate with evenly scheduled headway. Eberlein et al. (1999) adopted a rolling horizon approach to formulate control problems addressed. Each time the optimization problem is solved considering only a limited number of vehicles in the route, known as impact set, where an impact set was defined as a set of vehicles operating on a route. Then, The result of optimal control policy is applied to the first vehicle on the set. To illustrate their solution procedure For example, Consider a case where there are ten vehicles operating on a route. First, the optimization problem is solved considering only vehicles 1, 2, and 3 and the result of optimal control policy is only applied to vehicle 1. In the second step, vehicles 10, 1, and 2 are considered and the optimization problem is solved for these vehicle set and the optimization control result is applied to vehicle 10, the first vehicle in the impact set. This process repeated for all vehicles on the route. Most probably, Eberlein et al. (1999) used the concept of impact set to reduce the complexity of the problem. However, this comes at the price of eliminating the effect of vehicles interaction on the one-way loop route

Eberlein et al. (1999) attempted at minimizing the passengers waiting time at stations beyond the impact set, the effectiveness of these control models was tested using empirical data. They concluded that the holding control strategy was found to be more effective than the other two strategies.

O'Dell and Wilson (1999) presented formulations for disruption control problems in rail transit systems with more than one rail branch. They investigated several holding and short

turning strategies. O'Dell and Wilson (1999) objective was to minimize waiting time of passengers within and beyond an impact set, with an impact set being defined as a set of trains and stations preceding and succeeding a disruption point. Using empirical data, they concluded applying the holding control strategies, passengers waiting time could be reduced by 15 to 40 percent.

Several common aspects of the above two studies (Eberlein et al. 1999) and (O'Dell and Wilson 1999) on modeling holding control by are summarized below.
a) The models are of mathematical formulation type with passengers waiting time as objective and departure time of vehicles from downstream stations as decision variables.

b) The holding models constructed based on availability of real-time information. Such information includes information the current position of vehicles, the number of passengers on board, and departure (arrival) time of trains from (at) visited stations.

c) The dwell time of trains at stations is approximated using a linear function. It is calculated based on the number of passengers boarding and alighting a train at a given station. Generally, in rail transit systems, alighting and boarding of passengers is sequential. Thus, the dwell time could be represented as the sum of alighting and boarding times.

1.4 Research Scope and Objectives

This thesis is divided into two major parts. In the first part, a multi-attribute performance measurement model is presented for advanced public transit systems. Depending on the frequency of the transit service, either headway-based or schedule-based approaches are used to evaluate the reliability of a transit system. The focus of earlier studies was either on headway-based or scheduled-based approach, but not both. Thus, The objective herein is to develop an integrated model to measure the performance of transit systems with varying service frequencies. It is believed that this thesis is the first attempt at developing such a performance measurement model that could be used in advanced public transit systems.

In the second part of this thesis, a real-time holding control problem is formulated with real-time information for a high frequency bus transit system. The cost function adopted in this thesis is more of a total measure than those adopted in earlier studies. It is the sum of the waiting times of passengers at stops, and the additional time that some passengers may encounter when left not served because of an overloaded bus.

While, the availability of real-time information has increased by emerging new technologies such AVL and GPS systems, few are those researchers who investigated control strategies for transit systems in real-time environment. This thesis takes a logical step in pursuing this direction of research.

1.5 Thesis Organization

This thesis is organized as follows. In chapter 2 we present a performance measurement model. Two numerical examples are provided to illustrate the application of the proposed model. In chapter 3, a new approach to model holding control problem is presented. Assumptions and limitations along with the control process will be discussed. Two adaptations of simulated annealing, adaptive-SA and adaptive tabu -SA, are proposed in chapter 4. Performance of the proposed methods is compared with conventional simulated annealing using several job shop scheduling problem. Computational results are compared using statistical analysis to evaluate the performance of the methods. In chapter 5 we apply adaptive simulated annealing to holding control problem described in chapter 3. The effectiveness of the proposed holding control model is illustrated using numerical examples. Finally, conclusions and future research are presented in chapter 6.

CHAPTER 2: A MULTI-ATTRIBUTE PERFORMANCE MEASUREMENT MODEL FOR ADVANCED PUBLIC TRANSIT SYSTEMS

This chapter presents a new approach to measure the performance of a service in advanced public transit systems. The novelty of the work presented herein lies in integrating two operation control tools, which are schedule and headway adherences applicable respectively to high and low frequency services. These control tools aid managers in depicting deviations in schedules, and take proactive corrective actions to effectively prevent service interruptions.

2.1 Problem Description

The problem considered here consists of I buses and N stops on a route. A *route* is defined as a one-way loop that connects a sequence of geographical points labelled *stops* and *terminals* where passengers can board and/or alight. A *trip* could be defined as the run of a bus in two directions, from the *dispatching* terminal to *intermediate* terminal and from intermediate terminal to the dispatching terminal following a schedule that gives the arrival (departure) times of buses at (from) each stop along the route (Figure 2.1). Each route has two directions. The first direction has a set of stops numbered 1 through $N/2$, and the second direction has a set of stops numbered $N/2+1$ through N. where N represents the dispatching terminal. Predicted arrival times of buses at downstream stops are provided to controllers in real-time. These values of expected arrival times could be used to calculate the performance of service provided by each bus with respect to the schedule or planned headways. The performance

measure calculated reflects deviation from schedule that could be used to devise appropriate control strategies. Thus, upon adequate action by controllers, service interruptions or lengthy delays could be avoided.

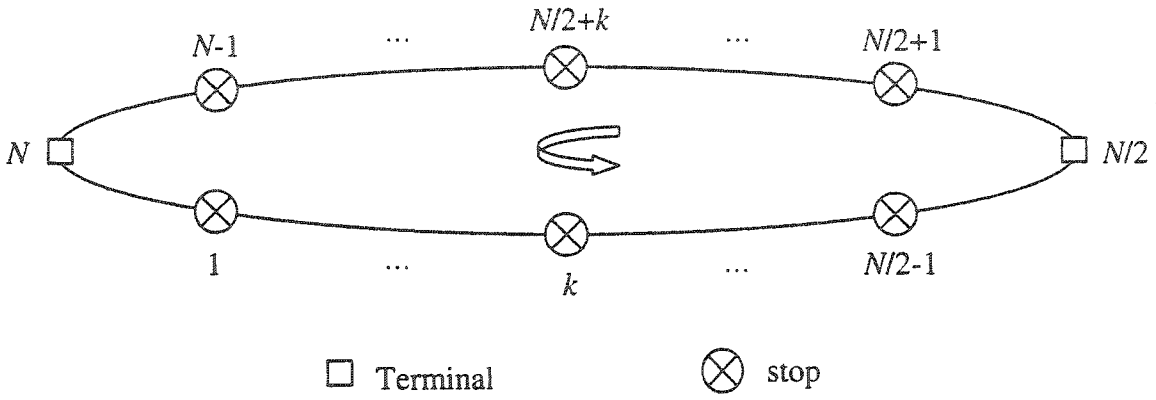


Figure 2.1 The schematic presentation of the one way loop route/multi-vehicle public transit problem

2.2 Notation List

The following variables and parameters are used in the formulations:

i	index of buses, $i = 1,\ldots,I$
k	index of stops, $k = 1,\ldots,N$ (starting terminal: $k = N$, intermediate terminal: $k = N/2$)
S_{ik}	scheduled arrival time of bus i at stop k
ΔS_{ik}	schedule adherence of bus i at stop k
F_{ik}	predicted arrival time of bus i at stop k
D_{ik}	running time delay of bus i over the route segment k
h_{ik}	expected headway of bus i at stop k
H	planned headway
ΔH_{ik}	headway adherence of bus i at stop k
P_s	schedule base performance index

P_h headway base performance index

W_{ik} weight of stop k for bus i

w_{ik} normalized weight of stop k for bus i

f_s weight value of schedule base performance

f_h weight value of headway base performance

T^h prediction horizon

$\tilde{N}_i$ total number of the downstream stops for which the prediction arrival times of bus i is available (stops in the prediction horizon)

l^i index of the last stop passed by bus i

l^k index of the last departed bus from stop k

$\tilde{I}_k$ total number of buses for which prediction arrival times at stop k are available

t^{now} present time

As buses progress along the route, the passenger information system predicts the arrival time of each bus at downstream stops. Using this predicted values, the deviation from the scheduled arrival time could be calculated as follows:

$$\Delta S_{ik} = F_{ik} - S_{ik} \tag{2.1}$$

Note that $\Delta S_{ik} < 0$, $\Delta S_{ik} > 0$ and $\Delta S_{ik} = 0$ indicate respectively that bus is either early, late or on time. The difference in schedule adherence of two consecutive stops is the running time delay associated with the route segment between these stops that is given as:

$$D_{ik} = \Delta S_{ik} - \Delta S_{i,k-1} \tag{2.2}$$

The delay D_{ik} can be zero, negative or positive. A zero delay indicates that bus i has traveled through the segment between stops (k-1) and (k) at expected speed. This does not mean that the possible lateness of the bus at stop (k-1) has been recovered. For example, if the bus was behind schedule at stop (k-1) by 5 minutes and $D_{ik} = 0$, then it can be concluded that the bus is still behind the schedule by the same 5 minutes at stop k. However, for the same bus, if $D_{ik} = +2$, it means the bus has traveled the route segment at a slower speed than expected and is

behind schedule at stop k by 7 minutes. By the same token, a negative running time delay indicates that the bus has traveled through the segment faster than expected.

Examining the relationship between D_{ik} and ΔS_{ik}, the following equation can be derived:

$$\Delta S_{ik} = \sum_{j=1}^{k} D_{ij} \qquad \begin{array}{l} i = 1,\ldots,I \\ k = l^i+1,\ldots, l^i+\tilde{N}_i \end{array} \tag{2.3}$$

Equation (2.3) indicates that the schedule adherence of bus i at any downstream stop k equals its cumulative delay of all segments preceding stop k. Since the predicted arrival times of bus i are available only for the next $\tilde{N}_i$ stops starting from stop l^i+1, the schedule adherence ΔS_{ik} is calculated for those stops.

Another important element of the public transit operations is the headway. As defined earlier, headway is the interarrival time between successive buses of the same route and in the same direction as they pass a particular point on that route. In planning operations of public transit systems, it is usually desirable to have a constant headway along a route. Therefore, the relationship between the planned headway, H, and the scheduled arrival time at any stop is:

$$H = S_{ik} - S_{i-1,k} \qquad \begin{array}{l} i = 1,\ldots,I \\ k = 1,\ldots,N \end{array} \tag{2.4}$$

However, due to deviations in arrival times along a route, the actual headway may not remain constant and may fluctuate from one stop to another stop and from one bus to another bus. The expected headway of bus i at stop k could be computed as follows:

$$h_{ik} = F_{ik} - F_{i-1,k} \qquad \begin{array}{l} i = 1,\ldots,I \\ k = l^i+1,\ldots, l^i+\tilde{N}_i \end{array} \tag{2.5}$$

Then, the headway adherence, i.e. the deviation of headway from the planned headway, could be computed as:

$$\Delta H_{ik} = h_{ik} - H \qquad \begin{array}{l} i = 1,\ldots,I \\ k = l^i+1,\ldots, l^i+\tilde{N}_i \end{array} \tag{2.6}$$

The headway adherence (2.6) could be either of a negative, or a positive value. A negative ΔH_{ik} indicates a shorter than planned headway, whereas a positive ΔH_{ik} shows a longer than expected headway. This could be used for headway control on high frequency routes, where regularity is of concern. The above-developed equations are important elements of the performance measurement model, which is discussed in the following section.

2.3 Performance Measurement Model

The purpose of developing a performance measurement model is to determine how well buses in operation adheres with desired headway and schedule. As aforementioned, the choice between a schedule-based and a headway-based model is highly dependent on the frequency of service. A high frequency service demands attention to maintaining a constant headway, whereas a low frequency service requires on time arrivals and departures based on a pre-announced schedule. The dividing line between the high- and the low-frequency services is vague. There are many instances where a service might be interpreted as a medium frequency. Then, a performance measurement model that incorporates both headway and schedule adherences might be desirable. Figure 2.2 illustrates the tabular structure of the proposed integrated performance model. Columns represent stops and rows represent buses running on the same route. Each cell associated with a pair of a bus and a stop contains three values. These values are: a) the schedule adherence ΔS_{ik}, b) the headway adherence ΔH_{ik}, and c) the weight value of stop k for bus i, w_{ik}.

In an ideal case, where buses are running on time with appropriate interarrival times as planned, the adherence values are zero, $\Delta S_{ik} = \Delta H_{ik} = 0$. In such a case, the performance measure is 100 percent. However, if some buses fall behind or ahead of their schedules, the performance model should be able to capture them and show a lower performance. The last column shows the total un-weighted schedule adherence of a bus for all stops in the prediction horizon that is calculated using equation (2.7). Since the schedule adherence at any stop can take both positive and negative values, the absolute values are used to eliminate the effect of cancellation.

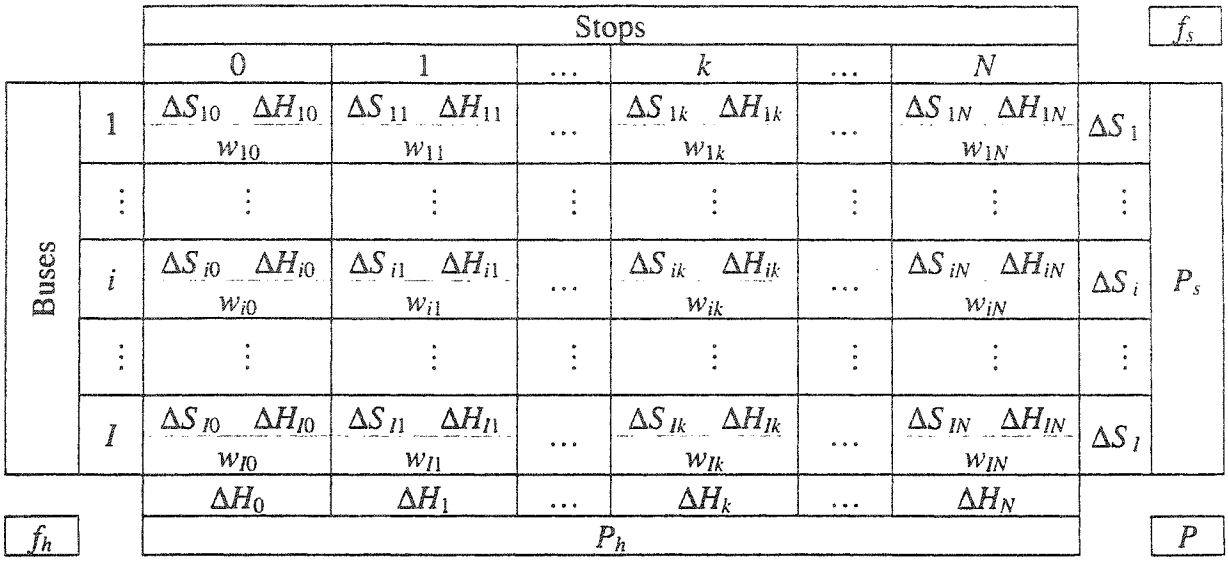


Figure 2.2 The tabular structure of the proposed performance measurement model

$$\Delta S_i = \sum_{k=l^i+1}^{l^i+\tilde{N}_i} |\Delta S_{ik}| \qquad i = 1,\ldots, I \tag{2.7}$$

Similarly, the last row indicates the total headway adherence of all buses at a particular stop in the prediction horizon. It is calculated according to the following formula (2.8).

$$\Delta H_k = \sum_{i=l^k+1}^{l^k+\tilde{I}_k} |\Delta H_{ik}| \qquad k = 1,\ldots, N \tag{2.8}$$

Since it is unrealistic to predict the bus movements far into the future, we assume that the prediction takes place for a finite horizon T^h. Therefore, depending on the distance between stops, the predicted arrival time of bus i only at a few downstream stops will be available. In above formulas, $\tilde{N}_i$ represents the number of downstream stops of bus i for which predicted arrival times are available. Also, we assume when prediction takes place, if bus i is at stop k, then l^i, the index of the last stop visited by bus i, is k-1.

The assignment of weights presented in Figure 2.2, are discussed in the subsequent section.

2.4 Bus Stop Weighting System

It is worth noting that when calculating performance measures, each stop may have an associated weight value for a bus that dynamically changes as the bus progresses along the route. The rational for this dynamic weight assignment is the increasing chance of prediction error for farther stops. In previous section, we introduced an integrated performance measurement model for public transit system. This model measures the performance of the system based on two known operation control tools, which are headway and schedule adherences. Calculating the performance of the system based on the predicted arrival times of buses at stops is one of the distinctive features of the proposed model. Although this approach helps the controller to act before facing possible interruptions, it could adversely affect the performance of the system as some unnecessary actions may be taken as a result of the prediction errors. In other word, the prediction arrival time of bus i at stop k is less accurate when the bus is too far from stop k. To alleviate this problem, we propose the following descending weight equations. As illustrated in Figure 2.3, the weight factor of the stop k for bus i is dependent on how distant bus i is from stop k.

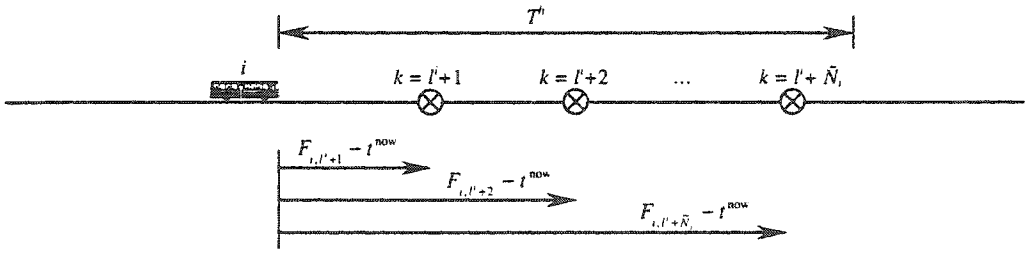


Figure 2.3 Bus i and its downstream stops in the planning horizon T^h

$$W_{ik} = \frac{T^h - \left(F_{ik} - t^{now}\right)}{T^h} \qquad \begin{matrix} i = 1, \ldots, I \\ k = l^i+1, \ldots, l^i+\tilde{N}_i \end{matrix} \tag{2.9}$$

In order to normalize the weight values for each bus, first we calculate the sum of the weights as follows:

$$\sum_{j=l^i+1}^{l^i+\tilde{N}_i} W_{ij} = \frac{\tilde{N}_i\left(T^h + t^{\text{now}}\right) - \sum_{j=l^i+1}^{l^i+\tilde{N}_i} F_{ij}}{T^h} \qquad i = 1, \ldots, I \tag{2.10}$$

Thus, the normalized weights are:

$$w_{ik} = \frac{T^h - \left(F_{ik} - t^{\text{now}}\right)}{\tilde{N}_i\left(T^h + t^{\text{now}}\right) - \sum_{j=l^i+1}^{l^i+\tilde{N}_i} F_{ij}} \qquad \begin{array}{l} i=1, \ldots, I \\ k = l^i+1, \ldots, l^i+\tilde{N}_i \end{array} \tag{2.11}$$

The magnitude of the weight value for any bus is mostly dependent on the time distance of the bus from the stop. At any given instant, the farther downstream stops have less weight values than the immediate stops. The number of stops, $\tilde{N}_i$, for which the predicted arrival times are available, depends on the prediction horizon T^h.

2.5 Formulations

As described in section 2.3, the proposed performance measurement model, consists of two components, a schedule-based and a headway-based components. Depending on the frequency of the service, each of these two components could either be used separately as a partial performance measure or jointly as total performance measure. The schedule, P_s, and headway, P_h, performance indices are respectively computed as:

$$P_s = 1 - \frac{\sum_{i=1}^{I}\sum_{k=l^i+1}^{l^i+\tilde{N}_i} w_{ik}\left|\Delta S_{i,k}\right|}{I\left(\alpha . T^h\right)} \tag{2.12}$$

$$P_h = 1 - \frac{\sum_{i=1}^{I} \sum_{k=l'+1}^{l'+\tilde{N}_i} w_{ik} \left| \Delta H_{ik} \right|}{I.H} \tag{2.13}$$

where P_s and P_h are normalized over the range of [0,1] with 1 being the best performance. In (2.12), it is assumed that the schedule adherence of a bus at any stop has an upper bound of αT^h (i.e., $\Delta S_{ik} \leq \alpha T^h$), where T^h is the prediction horizon and α is a user-selected coefficient that controls the magnitude of the upper bound of schedule adherences. Similarly, in (2.13),H is the upper limit for headway adherence at any stop.

Each of the above partial performance indices provides a valuable means for service control. In low frequency transit services, where service punctuality is critical, it is appropriate to use the schedule adherence performance index, P_s. whereas in high frequency transit services, where the interarrival times of buses are relatively small, maintaining a constant headway between consecutive buses is important. In this case, it is appropriate to use the headway performance index, P_h. However, a combined performance measure could be used and it is presented as:

$$P = f_s P_s + f_h P_h \tag{2.14}$$

Where f_s and f_h are weight values between 0 and 1 for schedule and headway indices, respectively.

2.6 Numerical Examples

To illustrate the application of the proposed performance measurement model, the following two examples are provided.

Example 2.1:

Suppose there are five buses running on a route consisting of eight intermediate stops and two terminals as illustrated in Figure 2.4. The planned headway is H = 5 minutes, the prediction horizon is T^h = 8 minutes, and the number of downstream stops in the prediction

horizon of bus i is assumed to be $\tilde{N}_i$ =3 for all buses. The scheduled arrival time, S_{ik}, and predicted arrival time, F_{ik} ,for all buses at their next three stops are given in Table 2.1.

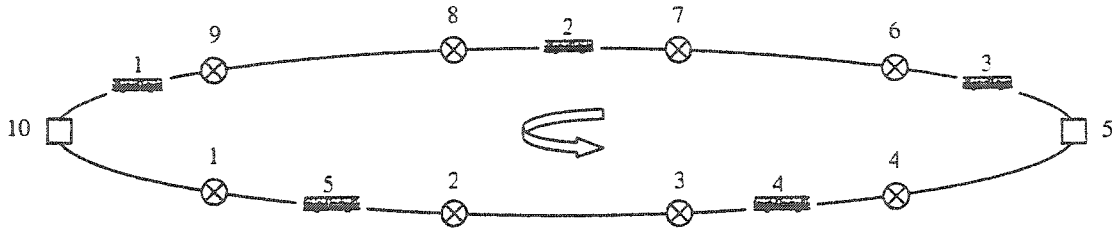


Figure 2.4 Stops and position of buses for Example 2.1

		Scheduled arrival times									
		Stop 1	Stop 2	Stop 3	Stop 4	Stop 5	Stop 6	Stop 7	Stop 8	Stop 9	Stop 10
	Bus 1	4	6							✓	1
	Bus 2							✓	1	4	6
(a)	Bus 3					✓	1	4	6		
	Bus 4			✓	1	4	6				
	Bus 5	✓	1	4	6						
		Predicted arrival times									
		Stop 1	Stop 2	Stop 3	Stop 4	Stop 5	Stop 6	Stop 7	Stop 8	Stop 9	Stop 10
	Bus 1	4	6								1
	Bus 2								1	4	6
(b)	Bus 3						3	6	8		
	Bus 4				1	4	8				
	Bus 5		1	4	6						

Table 2.1 The scheduled and predicted arrival times for Example 2.1
Check mark (✓) indicates the last visited stop of bus i, l^i.

Based on the information given in this table, the performance matrix can be constructed using equations 2.5 to 2.14, with results summarized in Figure 2.5. For example, the three values associated with Bus 1 and Stop 2 are computed as follows.

First, according to equation 2.1, the schedule adherence of Bus 1 at Stop 2 is calculated, ΔS_{12}= 4 – 4 = 0. In order to compute the headway adherence, first the predicted headway of Bus 1 at Stop 2 is calculated using equation 2.5, $h_{12} = F_{12} - F_{52} = 6 - 1 = 5$. Then, the headway adherence is computed according to equation 2.6, $\Delta H_{12} = 5 - 5 = 0$. Finally, the

weight of Stop 2 for Bus 1 is calculated using equation 2.11, (8 – (6–0))/(3×(8+0) –11) = 0.16. In calculation of the overall performance P, it is assumed that both headway and schedule adherences are of equal importance, that is $f_s = f_h = 0.50$. In this example, the schedule and the headway performance indices, P_s and P_h, are respectively 95 and 92 percent where the ideal performance of the system is 100%.

		Stops											0.50
		1	2	3	4	5	6	7	8	9	10		
Buses	1	0 0	0 0								0 0	0	95%
		0.30	0.16								0.54		
	2								0 0	0 0	0 0	0	
									0.54	0.3	0.16		
	3						2 2	2 2	2 2			6	
							0.71	0.29	0				
	4				0 0	0 0	2 0					2	
					0.64	0.36	0						
	5		0 0	0 0	0 0							0	
			0.54	0.3	0.16								
		0	0	0	0	0	2	2	2	0	0		
0.50		92%											93.5%

Figure 2.5 The performance measurement matrix for Example 2.1

Example 2.2:

Now, assume for the same route discussed in the above example, $\tilde{N}_i$ is equal to 1 and all five buses are dispatched as scheduled, but due to a traffic congestion, it is expected that based on the current position of buses and prediction horizon, all buses encounter 5 minutes late arrival to their downstream stops. Therefore, the predicted arrival times change as shown in Table 2.2.

The performance matrix for this problem is shown in Figure 2.6. Note that for any bus at the next stop, ΔS_{ik} is 5 as there will be a 5-minute late arrival. Similarly, ΔH_{ik} is 5 for any bus i at the next stop k as there will be a 5-minute deviation from planned headway, H. This results in the worst performance at any stop with respect to headway and it is reflected by $P_h = 0$. The schedule based performance P_s is not zero because the schedule deviations at stops are not at their maximum level of $T^h = 8$.

	Scheduled arrival times									
	Stop 1	Stop 2	Stop 3	Stop 4	Stop 5	Stop 6	Stop 7	Stop 8	Stop 9	Stop 10
Bus 1	✓	1	4	6	9					
Bus 2			✓	1	4	6	9			
Bus 3					✓	1	4	6	9	
Bus 4	9						✓	1	4	6
Bus 5	4	6	9						✓	1
	Predicted arrival times									
	Stop 1	Stop 2	Stop 3	Stop 4	Stop 5	Stop 6	Stop 7	Stop 8	Stop 9	Stop 10
Bus 1	-	6								
Bus 2				6						
Bus 3						6				
Bus 4								6		
Bus 5										6

Table 2.2 The scheduled and predicted arrival times for Example 2.2
Check mark (✓) indicates the last visited stop of bus i, l^i

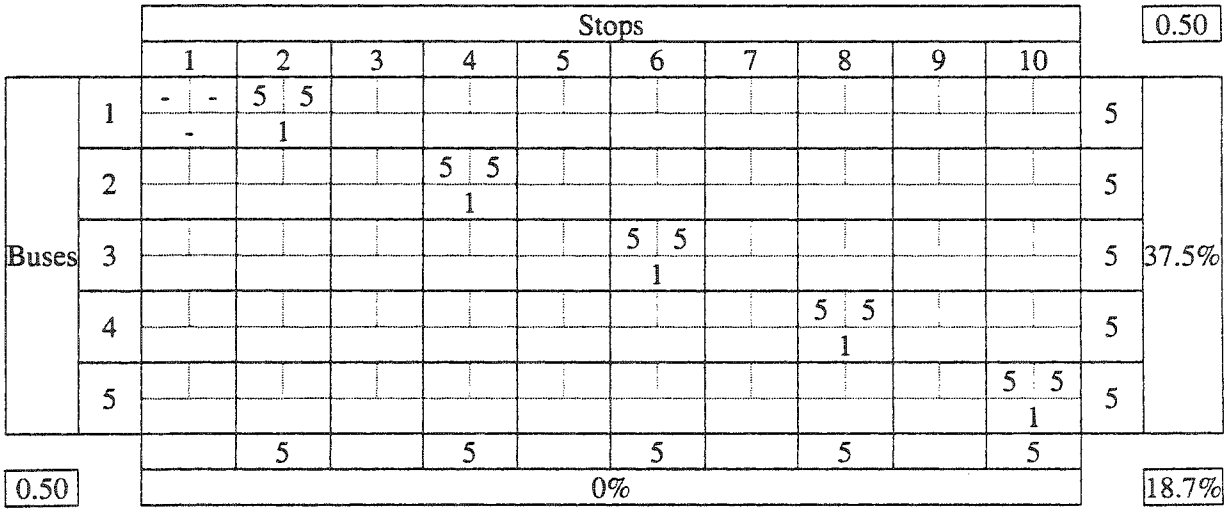


Figure 2.6 The performance measurement matrix for Example 2.2

2.7 Chapter Summary

Unlike earlier studies that focused on either schedule or headway adherences as control approach, we presented a new model to integrate both schedule and headway adherences. This feature of the model make it possible to be applied to any bus transit service regardless of its frequency; e.g., high, medium, or low. Furthermore, to measure schedule or headway adherences, the proposed model uses predicted rather than actual arrival times. The advantage of doing so is that it allows the operation mangers to avoid major service interruptions by proactively taking necessary corrective actions in advance.

CHAPTER 3: A MODEL FOR HOLDING STRATEGIES IN PUBLIC TRANSIT SYSTEMS WITH REAL-TIME INFORMATION

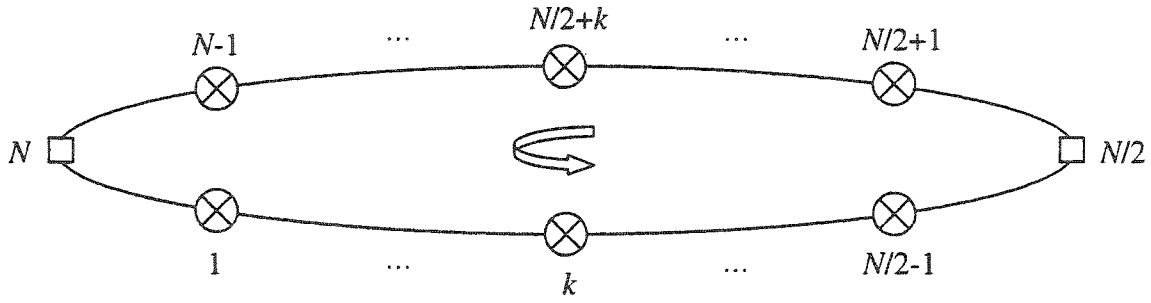
In this chapter, a mathematical model for a holding control strategy is developed. Particularly, this model uses real-time information of locations of buses along a specified route. The objective of the developed model is to minimize the waiting time of passengers at all stops on that route. Furthermore, the model is characterized by the flexibility of adopting situations where bus occupancy could be either high, or low.

3.1 Problem Description

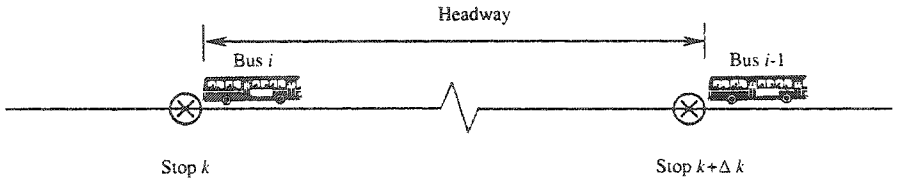
The problem considered herein investigates holding strategy that could be applied to control an operation of i buses on a specified route with N stops. A *route* is defined as a one-way loop that connects a sequence of geographical points labelled as *stops* and *terminals*. Where passengers could board and alight at the stops, but either board or alight at the terminals. We divide the route into two directions. In the first direction (includes stops 1, 2, ...,$N/2$-1), buses are running from the dispatching terminal (N) to intermediate terminal ($N/2$), whereas in the second direction (includes stops $N/2$+1, $N/2$+2,..., N-1), they are running from intermediate terminal ($N/2$) to the dispatching terminal (N) following a given headway (see Figure 3.1 for illustration). It is assumed that at any given instant, and using a real-time passenger information system and automatic vehicle location system (AVL system), the necessary information is transmitted to controllers. This information includes: (1) the departure times of buses from the last visited stops, (2) indices of last visited stops, and (3)

loads on buses departing their last visited stops. However, there is information that will not be transmitted, such as: (1) passenger arrival rate, (2) fraction of passengers on board of buses alighting at stops, (2) dwell time parameters, and (4) running time between stops include acceleration and deceleration times. We assume that these parameters could be determined from available historical data.

At any given instant, departure time of buses from the last visited stop, indices of last visited stops, and loads on buses departing last visited stops are provided by real-time passenger information and automatic vehicle location system (AVL) systems. Other required data including passenger arrival rate and alighting fraction at each stop, dwell time parameters, and running time between stops include acceleration and deceleration time are assumed to be estimated from available historical data.



(a) One-way loop



(b) Headway

Figure 3.1 The schematic presentation of the one-way loop and headway

3.1.1 Assumptions and limitations

The model is subject to the following assumptions and limitations:

Assumptions:

- When the bus is not loaded to capacity, it is assumed that all boarding take place at the front door and alighting take place at rear door.
- When bus is loaded to capacity, the front door can be used for both boarding and alighting.

Limitations:

- Running times are approximated by their expected values.
- Dwell time function is approximated by a linear function.
- During controlling the vehicles we consider a limited number of downstream stops for each bus, which is called "rolling horizon".

3.1.2 Data requirements

In this thesis, it is assumed that having an AVL system and historical data available, the following data could be estimated.

Provided through AVL system:

- Actual departure times of buses from the most recently visited stops
- Predicted departure times of buses from downstream stops (maximum $\tilde{N}_i$ stops for each bus i)

Estimated from historical data:

- Expected value of bus running times between stops
- Expected value of bus dwell times at stops
- Passenger arrival rates and alighting fractions at each stop
- Dwell time function parameters

3.1.3 Notation list

The following variables and parameters are used in the proposed formulations:

i	index of vehicles, $i = 1,\ldots,I$
k	index of stops, $k = 1,\ldots,N$
d_{ik}	departure time of bus i from stop k
l_{ik}	load in bus i departing stop k
p_{ik}	passengers left behind by bus i at stop k
D_{ik}	demand for bus i at stop k
r_k	passenger arrival rate at stop k
v_{ik}	1, if bus i is loaded to capacity when it depart stop k; 0, otherwise
S^u_{ik}	dwell time of bus i at stop k when the passenger load is below the bus capacity
S^c_{ik}	dwell time of bus i at stop k when bus i is loaded to capacity
C_j	dwell time function parameter
Q_k	passenger alighting fraction at stop k
A_k^f	passenger alighting fraction at stop k that use front door for alighting
A_k^r	passenger alighting fraction at stop k that use rear door for alighting
R_k	running time from stop k-1 to stop k, including acceleration and deceleration
L^{max}	passenger capacity of bus
M_j	sufficiently large numbers
$\mathbf{K}$	set of stations on the route $\mathbf{K}=\{1,2,3,\ldots,N/2,\ldots,N\}$
$\mathbf{I}$	set of buses operating on the route $\mathbf{I}=\{1,2,3,\ldots,i,\ldots,I\}$
$\tilde{N}_i$	total number of stops in the horizon of bus i
T^h	rolling horizon (unit of time, i.e., min.)
t^{now}	present time (time now)
η	expected value of running times on a route
φ	expected value of bus dwell time at stops
l^i	index of the last visited stop by bus i
k_{fst}	the first stop in the horizon of bus i

3.1.4 Rolling horizon

As discussed above, the transit system considered in this study is a one-way loop in which buses are operating on the route following a planned headway. When controlling the route, and for each bus, only a limited number of immediate downstream stops are considered. This way, the problem could be reduced to a reasonable size if there are a large number of stops and buses on the route. The number of downstream stops, $\tilde{N}_i$, depends on the length of the rolling horizon T^h as described in Figure 3.2. The rolling horizon is a common term in many fields. Even in the transportation literature, despite similarities it has been defined differently (e.g., Newell 1998, Choong et al. 2001). In this thesis, the rolling horizon is defined as a reasonable length of time in which bus movement on the route could be project fairly. It is a "horizon" as it is a finite future time period, and it is "rolling" because it moves forward as buses progress on their routes. Since stops are not equidistant, then the total number of stops in a given rolling horizon could be different for each bus operating on the same route. In order to ensure that the waiting time of passengers at all stops on the route are considered, the length of the rolling horizon, T^h, must be selected such that:

$$T^h > \max\left\{d_{i,l^i-1} - t^{now}\right\} \qquad \forall i \in \mathrm{I} \tag{3.1}$$

Furthermore, the relationship between the rolling horizon, T^h, and the number of downstream stops in the horizon of bus i, $\tilde{N}_i$, could be defined as follows:

$$\tilde{N}^i = \frac{T^h}{\eta + \varphi} \tag{3.2}$$

Where η is the expected value of the route running times, and φ is the expected value of the bus dwell time at stops.

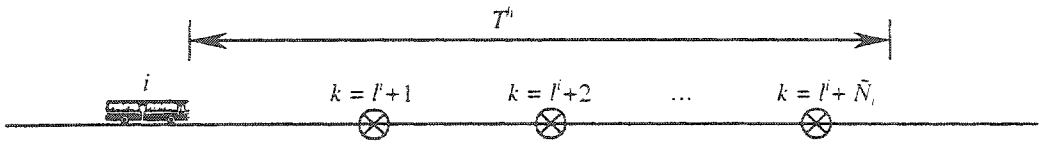


Figure 3.2 Rolling horizon for bus i

3.1.5 Cost function

In this thesis, the cost (objective) function sums the passengers waiting times at stops along a route. If bus i is crowded when departing from stop k, then there is an additional waiting time for passengers who could not board that bus. These passengers are assumed to wait for the next bus; i.e., bus i+1. The total waiting time of passengers is then calculated as

$$\sum_{i=1}^{I} \sum_{k=l^i+1}^{l^i+\tilde{N}_i} \left[\frac{r_k \left(d_{i,k} - d_{i-1,k}\right)^2}{2} + p_{i,k} \left(d_{i+1,k} - d_{i,k}\right) \right] \tag{3.3}$$

The first term in the cost function computes the waiting time for passengers who arrive randomly at stop k since the last bus departed that stop. The second term computes the additional waiting time for passengers who are left behind at stop k because of an overloaded bus.

3.1.6 Dwell time functions

Guenthner and Sinha (1983) presented a mathematical model for estimating bus dwelling time at bus stop using data from *Milwaukee and Lafa*yette. They assumed that the number of passenger boarding and alighting at each bus stop follows a Poisson distribution, and that the passenger demand is uniformly distributed along a bus route.

Dwell time is an important element in determining the departure time of buses from stops. It is directly related to the number of passengers boarding and alighting, and whether a bus has reached capacity or not. Although the true dwell time may be non-linear, several studies indicate that for short headway services, dwell time could be approximated by a linear function (Lin and Wilson 1992), (Eberlein et al. 1998). Following are two linear functions used to calculate bus dwell time at stop in this thesis. When the busload is below capacity, the dwell time is computed as:

$$S^u{}_{i,k} = C_0 + \max\left\{ C_1 A_k^r \; l_{i,k-1}, C_2 \left[\; r_k \left(d_{i,k} - d_{i-1,k}\right) + p_{i-1,k} \; \right] \right\} \tag{3.3}$$

where the first term C_0 is a constant, C_1 and C_2 are, respectively, the alighting and boarding times per passenger when load in a bus is less than the bus capacity. The total number of passengers boarding bus i at stop k is the summation of those passengers who arrive at stop since last bus $(i-1)$ departed and the number of people left behind by bus $i-1$; i.e., $[r_k(d_{i,k} - d_{i-1,k}) + p_{i-1,k}]$. The number of passengers alighting at stop k is calculated using the load on bus when i departing previous stop $k-1$ multiplied by alighting fraction at stop k, $A_k^r l_{i,k-1}$.

If bus i is loaded to capacity when departing stop k, then the dwell time is computed as:

$$S^f{}_{i,k} = C_3 + \max\{C_4 A_k^r l_{i,k-1}, C_4 A_k^f l_{i,k-1} + C_5 \min\{\lfloor L^{\max} - (1-Q_k) l_{i,k-1} \rfloor, \lfloor r_k(d_{i,k} - d_{i-1,k}) + p_{i-1,k} \rfloor\}\} \quad (3.4)$$

where C_3 is a constant, C_4 and C_5 are, respectively, the alighting and boarding times per passenger in crowded condition. In this case, the number of passengers boarding is either the number of passengers at stop k who desire to board bus i, or the number of available seats on bus i, which ever is minimum.

3.2 Mathematical Model

Three types of holding strategy have been addressed for rail transit systems in literature (O'Dell and Wilson 1999). The first strategy, known as holding-all, permits holding each train at any of the stations in the impact set. An impact set represents the number of trains before and after a disruption point. The second strategy is to hold each train at the first station it reaches. The third strategy is to hold each train at only one station in the impact set that reduces the waiting time of passengers. The holding station for this strategy could be any station in the impact set.

In this section, and using the second strategies of O'Dell and Wilson (1999), a mathematical model is developed for holding strategy. The objective of this model is to alleviate the effect

of service irregularities on passengers' waiting times in bus transit systems. The question this paper attempts to answer is which bus (buses) has (have) to be held at the first downstream stop in order to minimize total passengers' waiting time. Then the mathematical programming problem could be presented as:

$$\text{minimize} \quad \sum_{i=1}^{I} \sum_{k=l^i+1}^{l^i+\tilde{N}_i} \left[\frac{r_k \left(d_{i,k} - d_{i-1,k}\right)^2}{2} + p_{i,k}\left(d_{i+1,k} - d_{i,k}\right) \right]$$

Subject to:

$$l_{i,k} \le D_{i,k} \qquad \forall\, i,k \in (\mathrm{I},\mathrm{K}) \qquad (3.5)$$

$$l_{i,k} \le L^{\max} \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \qquad (3.6)$$

$$l_{i,k} \ge L^{\max} v_{i,k} \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \qquad (3.7)$$

$$l_{i,k} \ge D_{i,k} - M_1 v_{i,k} \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \qquad (3.8)$$

$$d_{i,k} \ge d_{i,k-1} + R_k + S^u{}_{i,k} - M_2 v_{i,k} \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \qquad (3.9)$$

$$d_{i,k} \ge d_{i,k-1} + R_k + S^c{}_{i,k} - M_3(1 - v_{i,k}) \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \qquad (3.10)$$

$$d_{i,k} \le d_{i,k-1} + R_k + S^u{}_{i,k} + M_4 v_{i,k} \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \mid k \ne k_{frst} \qquad (3.11)$$

$$d_{i,k} \le d_{i,k-1} + R_k + S^c{}_{i,k} + M_5(1 - v_{i,k}) \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \mid k \ne k_{frst} \qquad (3.12)$$

$$D_{i,k} = (1 - Q_k) l_{i,k-1} + r_k \left(d_{i,k} - d_{i-1,k}\right) + p_{i-1,k} \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \qquad (3.13)$$

$$p_{i,k} = \max \left\{ 0, D_{i,k} - L^{\max} \right\} \qquad \forall i,k \in (\mathrm{I},\mathrm{K}) \qquad (3.14)$$

$$Q_k = A_k^f + A_k^r \qquad \forall k \in \mathrm{K} \qquad (3.15)$$

$$0 \le Q_k \le 1 \qquad \forall k \in \mathrm{K} \qquad (3.16)$$

$$v_{i,k} \in \{0,1\} \qquad \forall i,k \qquad (3.17)$$

$$d_{i,k}, l_{i,k}, p_{i,k}, \ge 0 \quad \text{integer} \qquad \forall i,k \qquad (3.18)$$

Constraints (3.5) to (3.8) restrict the number of passengers on board when bus departing stop bus-stop. The number of passengers on board is calculated as a function of the load on the bus when arrives at station k and the number of boarding and alighting. When bus i at bus-stop k is at full capacity, $v_{ik}=1$, the constraints (3.5) and (3.6) are binding ($l_{i,k} = l^{\max}$). When bus i at bus-stop k is not at full capacity, $v_{ik} = 0$, then constraints (3.5) and (3.8) are binding. The constraints (3.9) to (3.12) restrict the earliest departure time of bus i at station k. To reduce headway irregularities in the service, the holding strategy assumed herein, allows for each bus to be held at the first stop it for a finite period (e.g., 5 minutes). When bus i is not loaded to capacity as it departs stop k, $v_{ik} = 0$, and depending on whether or not stop k is the first stop reached by bus i, constraint (3.9) or constraints (3.9) and (3.11) will apply. If stop k is the first stop reached by bus i, constraint (3.9) is binding. Therefore, The earliest departure time of bus i could be equal to, or greater than, the sum of the departure time from stop k–1, the running time from stop k–1 to stop k, and the dwell time at stop k. Otherwise, constraints (3.9) and (3.11) are binding. When bus i is at full capacity, $v_{ik} = 1$, and depending on whether or not stop k is the first stop reached by bus i, constraint (3.10) or constraints (3.10) and (3.12) will apply. If stop k is the first stop reached by bus i, constraint (3.10) is binding. Otherwise, both constraints (3.10) and (3.12) are binding.

The demand for bus i at stop k is calculated by constraint (3.13). Constraint (3.13) represents the load on the bus i departing stop k–1 less the number of passengers alighting at stop k, added to the number of passengers arrived at stop k since bus i–1 departed, and to the number of passengers who were left behind at stop k by bus i-1, $p_{i,k}$, with the later given by constraint (3.14). Note that the value $p_{i,k}$ depends on the demand for bus i at the stop k, and on the number of available spots on the bus after alighting. When the number of available spots on bus i is more than, or equal to, the demand for bus i at stop k, then the number of passengers left behind at stop k is equal to zero. Otherwise, the number of passengers left behind at stop k is the demand at that stop less the capacity of bus i.

3.3 Chapter Summary

In this chapter, we developed a mathematical model for holding control problem applicable in urban bus transit operations. The model has the flexibility of adopting both conditions where the passenger load on the bus is below the bus capacity and where a bus is overloaded departing any stop along the route. The objective is to minimize the total passengers' waiting time including new arrivals at stops and those who are left behind by an overloaded bus with the departure time of downstream stops as decision variables. The proposed dwell time functions calculate the delay times at stops based on the number of passengers boarding and alighting from the front and the rear door of the bus.

CHAPTER 4: ADAPTIVE TEMPERATURE CONTROL FOR SIMULATED ANNEALING

Simulated annealing is a stochastic search method capable of moving out of local optima by means of a transition probability. The transition probability is controlled by a temperature function that changes during the search. In this chapter, we focuses on testing and development of two cooling mechanisms for simulated annealing method. The performance of the methods is evaluated using a set of job shop scheduling problems selected from the literature.

4.1 Problem Description

Metaheuristics have gained considerable attentions and have experienced remarkable growth over the past decade. Among their benefits is their flexibility in applications to complex optimization problems. Simulated annealing, tabu search and genetic algorithms are among the most popular metaheuristics. In their standard forms, each of these heuristics has a unique search mechanism that allows them to escape local optima. The performance of these heuristics can be further improved if used in conjunction with an application-specific procedure. For instance, in tabu search, tabu list is a standard feature that forbids the move to the most recently visited solutions, and in simulated annealing, transition probability is a standard feature that allows a non-improving move to be made. Any of these methods may be used for a local search step in an application-specific algorithm. For example in application to job shop scheduling problems, Taillard (1994) proposed a tabu search to find a good

schedule for any sub-problem generated by a divide-and-conquer algorithm. Similarly, Yamada and Nakano (1995) used *shifting bottleneck* method to enhance the performance of their simulated annealing algorithm. In this study, the focus is on standard features of simulated annealing regardless of the area of application. However, to evaluate the performance of the proposed method, the job shop scheduling problem has been selected. The job shop scheduling problem studied in this research is to schedule a set of jobs on a set of machines such that each job has a specific sequence of operations without preemption and each machine can handle at most one job at a time. The objective is to minimize the *makespan*, the duration in which all jobs are completed. Job shop scheduling problem is among the hardest combinatorial optimization problems and proved to be NP-hard (Lenstra and Rinnooy Kan 1979). The problem can be formulated as follows.

Let

$J = \{1,\ldots,n\}$ denote the set of jobs;

$M = \{1,\ldots,m\}$ denote the set of machines;

O_j be the set of pair of operations (k, r) of job j constrained by the precedence relations;

p_{jk} denote the processing time of job j on machine k; and

x_{jk} be the start time of job j on machine k.

Given the above definitions, the problem can be stated as:

$$\text{Minimize} \quad C_{\max} = \max_{j,k}\{x_{jk} + p_{jk}\} \tag{4.1}$$

subject to :

$$x_{jr} \geq x_{jk} + p_{jk} \qquad \forall j \in J, (k,r) \in O_j$$

$$x_{ik} \geq x_{jk} + p_{jk} \quad \vee \quad x_{jk} \geq x_{ik} + p_{ik} \qquad \forall i, j \in J, k \in M$$

$$x_{jk} \geq 0 \qquad \forall j \in J, k \in M$$

In the above model, the first set of constraints describes the precedence relations among operations of each job, and the second set is to avoid having two jobs scheduled simultaneously on the same machine. This formulation is a disjunctive programming problem

due to the either-or constraint. Applegate and Cook (1991) discussed a mixed-integer programming transformation of this problem.

One of the optimization algorithms for job shop scheduling is branch and bound where a dynamically constructed tree representing all feasible schedules is searched. Through rules and procedures of this technique, large portions of the tree can be iteratively eliminated until an optimal solution is found. For many years, branch and bound was the most popular technique. However, due to limitations of enumeration techniques, approximation methods became another alternative (Jain and Meeran 1999). The shifting bottleneck procedure (Adams et al. 1988) is among the well-known approximation methods for job shop scheduling. This procedure employs a sequence of decomposition, bottleneck identification, single-machine scheduling and re-optimization. Nonetheless, other researches showed that this method suffers from difficulty in re-optimization phase and infeasible solutions (Balas et al. 1995). Iterative algorithms with local search are another popular category of job shop scheduling methods, and simulated annealing is one of the most popular techniques in this category (Kirkpatrick et al. 1983). The driving mechanism of simulated annealing is based on a neighbourhood search in which a probability function determines the transition from one solution to another. The magnitude of this probability depends in part on a temperature parameter that declines according to a cooling schedule. In this study, we propose two adaptations of simulated annealing. The first method uses an adaptive temperature control instead of a constantly descending function, and the second one has the same temperature control mechanism but it is accompanied by a tabu list that keeps track of recently visited solutions.

4.1.1 Conventional simulated annealing

Simulated annealing (SA) is among the most popular iterative methods that have been applied widely to solve many combinatorial optimization problems. This method is a random local search technique based on principles of physics. The search starts from an initial feasible solution. Each solution has a specific cost value. A small change in one or a combination of some variables can generate a neighbouring solution with a different cost value. In simulated annealing, the neighbouring solution is generated randomly. If the cost

value of the candidate solution is lower than that of the current solution, a move to the candidate solution is made. However, if the candidate does not improve the current solution, there is still a chance of transition according to the following probability function.

$$\mathrm{P(transition)} = \min\left\{1, \exp\left(-\frac{\Delta C_i}{\theta_i}\right)\right\} \tag{4.2}$$

where ΔC_i is the cost difference between the candidate solution and the current solution in iteration i, and θ_i is the control parameter known as temperature. In each iteration, the above transition probability is compared with a uniform random number. If the probability value is greater than or equal to the random number, then the transition to the worse solution is accepted. If the transition from the current solution to the candidate solution is rejected, another solution in the neighbourhood will be generated and evaluated. The temperature decreases during the search according to a function known as *cooling schedule* or *annealing schedule*. As a lower temperature contributes to a lower transition probability, the end of the search has less chance of escaping local minima while in the beginning, the search is more likely to move out of local minimum solutions.

Simulated annealing algorithm

Step 1:

Set $i = 1$ and select $\theta_{\min}$.

Select an initial solution and set the current solution B_i and the best solution O_i equal to the initial solution.

Step 2:

Select a candidate solution B^c from the neighbourhood of B_i

IF $Z(B^c) < Z(B_i)$ THEN

Set $B_{i+1} = B^c$

IF $Z(B^c) < Z(O_i)$ THEN set $O_i = B^c$.

Go to Step 3

ENDIF

Generate a random number u from a U(0,1) distribution;

Compute the transition probability $P(B_i, B^c) = \exp[(Z(B_i) - Z(B^c)) / \theta_i]$

IF $u \leq P(B_i, B^c)$, THEN set $B_{i+1} = B^c$.

Step 3:

Select θ_{i+1}

Set $i = i + 1$.

IF termination condition is satisfied, THEN stop; ELSE, go to Step 2.

END

Van Laarhoven et al. (1992) applied an approximation algorithm based on simulated annealing to minimize the makespan in job-shop scheduling problems. Their computational experiments showed that the algorithm could find shorter makespan than two other approximation approaches proposed by Adams et al.(1988) and Matsuo et al.(1988). The most important elements of Van Laarhoven et al. (1992) algorithm are as follows.

Cost function: The cost function is the makespan of the job shop scheduling problem.

Neighbouring solution: A neighbour is generated by interchanging the position of two jobs on a randomly selected machine.

Cooling schedule: The temperature declines according to the following function.

$$\theta_{i+1} = \frac{\theta_i}{1 + \frac{\theta_i \ln(1+\delta)}{3\sigma_i}} \tag{4.3}$$

where θ_i is the temperature in iteration i, σ_i is the standard deviation of the previously visited solutions and δ is an empirical distance parameter.

In this study, we have developed an algorithm based on the above cooling schedule. The initial value of the temperature is 0.95 and the distance parameter is set to 0.01. From now on, we refer to this algorithm as *conventional simulated annealing* or simply SA algorithm and we use it as a benchmark for the performance evaluation of the proposed methods.

4.1.2 Adaptive simulated annealing

In this section, we propose a modified version of the simulated annealing algorithm for job shop scheduling problems and we call it *adaptive SA*. The distinct feature of this method is the temperature change mechanism, which is an important part of the transition probability equation. In the conventional simulated annealing, the search begins with a high temperature

allowing a higher chance of transition to a worse solution. By doing so, the search is able to move out of local minima. However, as the search continues, the temperature continuously declines resulting in reduced chance of uphill transition. Such an approach could be useful if the local minima are near the start point, but may not lead to a near optimal solution if some local minima are encountered at a relatively low temperature toward the end of the search. To alleviate this difficulty, we propose an *adaptive simulated annealing* method that takes into consideration the characteristics of the search trajectory. In this method, instead of the previous cooling schedule (equation 4.3) the following *temperature control function* is used.

$$\theta_i = \theta_{\min} + \lambda \ln(1 + r_i) \tag{4.4}$$

where $\theta_{\min}$ is the minimum value that the temperature can take, λ is a coefficient that controls the rate of temperature rise, and r_i is the number of consecutive upward moves at iteration i. The initial value of r_i is zero, thus the initial temperature $\theta_0 = \theta_{\min}$. At any point during the search, if a move is made to a solution with a cost higher than that of the previous solution (i.e., upward move), then the counter r_i increases by 1. If the new solution has an equal cost, r_i remains unchanged; and if the cost is improved, then r_i is equal to zero. That is:

$$r_i = \begin{cases} r_{i-1} + 1 & \text{if } \Delta C_i > 0 \\ r_{i-1} & \text{if } \Delta C_i = 0 \\ 0 & \text{if } \Delta C_i < 0 \end{cases} \tag{4.5}$$

The rationale behind this approach is that there is a good chance of downhill moves in the beginning of the search, thus, there is less need to a high temperature to push the search out of local minima. However, as the search continues, the cost value of the solutions reduces, and the chance of getting trapped in a local minimum increases. Therefore, the need to a high temperature that could move the search out of local minima becomes more evident. With this new temperature control scheme, the search is given a higher chance of uphill move once it started climbing up regardless of the iteration number k. This provides a chance to explore low cost solutions surrounded by high cost neighbours; and since the temperature is not linked with the iteration number k, the neighbouring solutions in the beginning of the search

are treated the same way as the neighbouring solutions near the end of the search. Another benefit of this method is that the search would not freeze if the computational time were extended. This will be further discussed in section 4.2.

4.1.3 Adaptive tabu-simulated annealing

The third method we developed in this study is called *adaptive tabu-SA*. The only difference between this method and the previous one is the addition of a tabu list that keeps track of recently visited solutions. The idea of tabu search was first introduced by Glover (1977). Tabu search is a deterministic search approach that is designed to deal with the local minima problem. The search selects the best solution in the neighbourhood if it is not listed as tabu. Every time a solution is selected, it is added to the tabu list. This way, the cycling and revisiting is reduced. Compared to simulated annealing, tabu search has some drawbacks mostly due to its deterministic nature. Interested readers could refer to Zolfaghari and Liang (1999) for further discussions on the performance of tabu search and simulated annealing.
In this study, we propose a hybrid method that appends a tabu list to the adaptive SA and we call it the *adaptive tabu-simulated annealing* method. The proposed adaptive tabu-SA method has the benefits of both tabu search and the adaptive SA methods. This hybrid technique takes the advantages of the stochastic simulated annealing to escape local minima and at the same time it improves the performance of the search by having a tabu list on the side. It is worthwhile to mention that the length of the tabu list in the following test problems is 20. This value for the tabu length is decided after comparing the performance of different tabu list sizes. The performance of these methods is discussed in the following section.

4.2 Implementation and Comparison

4.2.1 Statistical analysis

To compare the performance of the above methods, all three methods have been coded in Visual Basic and tested using several well-known problems selected from the literature. The selected problems are known as LA7, LA15, LA16, LA23 (Lawrence 1984) ORB4

(Applegate and Cook 1991) and MT10 (Muth and Thompson 1963). Since we are interested in comparison of the three methods, we eliminated the effects of other factors by choosing the same neighbourhood generation scheme and computational time for all methods. In addition to methods, initial solution (seeds) could also play an important role in the performance of a search algorithm. Some start points could lead to a near optimal solution relatively faster than some other points. Several studies have shown that in simulated annealing algorithms, if the initial value of the temperature is sufficiently high, the start point does not have any effect on the final solution. In our study, however, this may not be the case as we have a distinct temperature control mechanism that does not require an initial temperature. Therefore, it is worthwhile to include the start point as a factor in our analysis. Furthermore, as all three methods are based on simulated annealing, which has a stochastic nature, the final solution may vary from one run to another. This is mostly due to the sequence of random numbers used in the search. For these reasons, we have designed a two-factor ANOVA with 6 replications. The first factor is "Method" with three levels, and the second factor is "seed" or initial solution with four levels. The computational results and statistical analyses are summarized in tables 4.1 to 4.6. In these tables, the overall best solution of the total 72 replications is shown in bold. It is worthwhile to mention that such solutions are the best solutions of a fixed computational time for comparison purpose only, therefore, they may not necessarily represent the optimal solutions. The optimal makespan for all six problems can be obtained through an extended computational time.

The purpose of the above ANOVA tables is to evaluate the effects of two factors (search methods and start points) as well as the existence of interactions between these factors. Three hypotheses of interest here are:

$H_0^{(1)}$: All three methods have equal mean values

$H_0^{(2)}$: All four start points (seeds) have equal mean values

$H_0^{(3)}$: No interactions exist between search methods and start points

Method	Seed 1		Seed 2		Seed 3		Seed 4		Mean
	Obs.	Mean	Obs.	Mean	Obs.	Mean	Obs.	Mean	
Conventional SA	902	919	909	923	897	927	944	923	923.04
	894		907		897		917		
	967		938		915		900		
	908		938		925		962		
	939		942		957		898		
	904		905		969		919		
Adaptive SA	909	901	947	928	904	919	939	920	916.96
	892		909		896		914		
	906		931		915		927		
	913		927		923		932		
	894		914		925		915		
	894		939		952		**890**		
Adaptive Tabu-SA	**890**	904	914	900	899	909	907	912	906.17
	904		893		**890**		932		
	949		**890**		897		910		
	890		**890**		916		898		
	900		899		938		902		
	890		914		914		922		
Mean	908.06		917.00		918.28		918.22		915.39

Optimal makespan = 890

Table 4.1(a) Computational results for the job shop scheduling test problem LA7 (15×5)

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F_0	p-Value	$F_{0.05}$
Methods	3505.86	2	1752.93	4.41	0.0163	3.15
Seeds	1309.44	3	436.48	1.10	0.3567	2.76
Interaction	1612.14	6	268.69	0.68	0.6690	2.25
Error	23829.67	60	397.16			
Total	30257.11	71				

Table 4.1(b) Analysis of variance table for problem LA7 (15×5)

	$\overline{C}_{i\cdot} - \overline{C}_{j\cdot}$	Significant?
Conventional SA vs. Adaptive SA	6.08	No
Adaptive SA vs. Adaptive Tabu-SA	10.79	No
Conventional SA vs. Adaptive Tabu-SA	16.87	Yes, $\overline{C}_{1\cdot} > \overline{C}_{3\cdot}$

$T_{0.05} = 13.83$

Table 4.1(c) Multiple-comparison Tukey's test for problem LA7 (15×5)

Method	Seed 1		Seed 2		Seed 3		Seed 4		Mean
	Obs.	Mean	Obs.	Mean	Obs.	Mean	Obs.	Mean	
Conventional SA	1384	1298	1255	1246	1287	1310	1226	1248	1275.17
	1247		1267		1325		1220		
	1314		1246		1340		1232		
	1299		1239		1307		1264		
	1305		1237		1331		1288		
	1238		1230		1267		1256		
Adaptive SA	1308	1275	1220	1235	1249	1270	1223	1247	1256.63
	1246		1249		1274		1220		
	1251		1224		1272		1250		
	1290		1227		1253		1265		
	1283		1240		1277		1270		
	1269		1251		1293		1255		
Adaptive Tabu-SA	1280	1272	1215	1238	1279	1277	1214	1231	1254.25
	1286		1251		1271		1251		
	1315		1264		1326		1246		
	1254		1249		1289		1220		
	1234		1234		1226		1232		
	1261		**1213**		1272		1220		
Mean	1281.33		1239.50		1285.44		1241.78		1262.01

Optimal makespan = 1207

Table 4.2(a) Computational results for the job shop scheduling test problem LA15 (20×5)

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F_0	p-Value	$F_{0.05}$
Methods	6295.53	2	3147.76	4.50	0.0151	3.15
Seeds	33094.93	3	11031.64	15.77	< 0.0001	2.76
Interaction	3062.03	6	510.34	0.73	0.6277	2.25
Error	41978.50	60	699.64			
Total	84430.99	71				

Table 4.2(b) Analysis of variance table for problem LA15 (20×5)

	$\overline{C}_{i\cdot} - \overline{C}_{j\cdot}$	Significant?
Conventional SA vs. Adaptive SA	18.54	Yes, $\overline{C}_{1\cdot} > \overline{C}_{2\cdot}$
Adaptive SA vs. Adaptive Tabu-SA	2.38	No
Conventional SA vs. Adaptive Tabu-SA	20.92	Yes, $\overline{C}_{1\cdot} > \overline{C}_{3\cdot}$

$T_{0.05} = 18.36$

Table 4.2(c) Multiple-comparison Tukey's test for problem LA15 (20×5)

Method	Seed 1		Seed 2		Seed 3		Seed 4		Mean
	Obs.	Mean	Obs.	Mean	Obs.	Mean	Obs.	Mean	
Conventional SA	1086	1120	1095	1136	1110	1140	1147	1125	1129.96
	1089		1095		1213		1164		
	1191		1149		1139		1133		
	1137		1174		1107		1064		
	1101		1125		1172		1100		
	1115		1177		1097		1139		
Adaptive SA	1101	1097	1092	1118	1070	1091	1080	1110	1103.96
	1105		1115		1063		1074		
	1100		1111		1129		1128		
	1058		1121		1063		1144		
	1134		1127		1149		1110		
	1083		1144		1069		1125		
Adaptive Tabu-SA	1088	1093	1137	1114	1079	1097	1098	1111	1103.63
	1128		1082		1108		1120		
	1084		1122		1106		1147		
	1084		1108		1084		1115		
	1093		1090		1106		**1058**		
	1080		1144		1098		1128		
Mean	1103.17		1122.67		1109		1115.22		1112.51

Optimal makespan = 1005

Table 4.3(a) Computational results for the job shop scheduling test problem ORB4 (10×10)

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F_0	p-Value	$F_{0.05}$
Methods	10956.44	2	5478.22	5.72	0.0053	3.15
Seeds	3782.38	3	1260.79	1.32	0.2772	2.76
Interaction	2573.67	6	428.95	0.45	0.8435	2.25
Error	57434.50	60	957.19			
Total	74743.99	71				

Table 4.3(b) Analysis of variance table for problem ORB4 (10×10)

	$\overline{C}_{i\cdot} - \overline{C}_{j\cdot}$	Significant?
Conventional SA vs. Adaptive SA	26.00	Yes, $\overline{C}_{1\cdot} > \overline{C}_{2\cdot}$
Adaptive SA vs. Adaptive Tabu-SA	0.33	No
Conventional SA vs. Adaptive Tabu-SA	26.33	Yes, $\overline{C}_{1\cdot} > \overline{C}_{3\cdot}$

$T_{0.05} = 21.47$

Table 4.3(c) Multiple-comparison Tukey's test for problem ORB4 (10×10)

Method	Seed 1		Seed 2		Seed 3		Seed 4		Mean
	Obs.	Mean	Obs.	Mean	Obs.	Mean	Obs.	Mean	
Conventional SA	1063 1002 1027 1039 1038 1045	1036	1032 1069 1040 1041 1013 1027	1037	1011 1083 1007 994 1027 1038	1027	1001 1028 1012 1001 1026 1078	1024	1030.92
Adaptive SA	1019 982 1044 1046 1013 1019	1021	986 1015 1019 1021 1007 1000	1008	993 999 1028 1026 1038 994	1013	1002 1012 986 1007 1001 1008	1003	1011.04
Adaptive Tabu-SA	979 998 1035 990 1046 1012	1010	1019 1023 999 1025 1021 1039	1021	1031 994 993 1046 994 972	1005	1018 1008 985 994 994 1020	1003	1009.79
Mean	1022.10		1022.00		1014.89		1010.10		1017.25

Optimal makespan = 945

Table 4.4(a) Computational results for the job shop scheduling test problem LA16 (10×10)

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F_0	p-Value	$F_{0.05}$
Methods	6742.75	2	3371.38	7.20	0.0016	3.15
Seeds	1853.83	3	617.94	1.32	0.2763	2.76
Interaction	1063.25	6	177.21	0.38	0.8899	2.25
Error	28091.67	60	468.19			
Total	37751.50	71				

Table 4.4(b) Analysis of variance table for problem LA16 (10×10)

	$\overline{C}_{i\cdot} - \overline{C}_{j\cdot}$	Significant?
Conventional SA vs. Adaptive SA	19.88	Yes, $\overline{C}_{1\cdot} > \overline{C}_{2\cdot}$
Adaptive SA vs. Adaptive Tabu-SA	1.25	No
Conventional SA vs. Adaptive Tabu-SA	21.13	Yes, $\overline{C}_{1\cdot} > \overline{C}_{3\cdot}$

$T_{0.05} = 15.02$

Table 4.4(c) Multiple-comparison Tukey's test for problem LA16 (10×10)

Method	Seed 1		Seed 2		Seed 3		Seed 4		Mean
	Obs.	Mean	Obs.	Mean	Obs.	Mean	Obs.	Mean	
Conventional SA	981	980	967	1091	1046	1074	1132	1104	1061.96
	976		1128		1090		1084		
	986		1132		1052		1069		
	951		1062		1121		1107		
	965		1135		1086		1118		
	1019		1122		1047		1111		
Adaptive SA	958	983	1094	1031	1018	1027	1077	1081	1030.33
	988		1078		1048		1085		
	973		1012		973		1057		
	1008		1005		1057		1100		
	972		995		992		1064		
	997		1000		1072		1105		
Adaptive Tabu-SA	955	970	1049	1022	1011	1028	1054	1041	1015.00
	992		1085		1042		1069		
	982		1030		1043		1028		
	949		1032		961		1031		
	971		948		1093		1033		
	968		990		1016		1028		
Mean	977.28		1048		1042.67		1075.11		1035.77

Optimal makespan = 930

Table 4.5(a) Computational results for the job shop scheduling test problem MT10 (10×10)

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F_0	p-Value	$F_{0.05}$
Methods	27522.00	2	13761.35	10.82	< 0.0001	3.15
Seeds	92991.60	3	30997.20	24.36	< 0.0001	2.76
Interaction	10805.53	6	1800.92	1.42	0.2236	2.25
Error	76333.17	60	1272.22			
Total	207652.99	71				

Table 4.5(b) Analysis of variance table for problem MT10 (10×10)

	$\overline{C}_{i\cdot} - \overline{C}_{j\cdot}$	Significant?
Conventional SA vs. Adaptive SA	31.63	Yes, $\overline{C}_{1\cdot} > \overline{C}_{2\cdot}$
Adaptive SA vs. Adaptive Tabu-SA	15.33	No
Conventional SA vs. Adaptive Tabu-SA	46.96	Yes, $\overline{C}_{1\cdot} > \overline{C}_{3\cdot}$

$T_{0.05} = 24.75$

Table 4.5(c) Multiple-comparison Tukey's test for problem MT10 (10×10)

Method	Seed 1		Seed 2		Seed 3		Seed 4		Mean
	Obs.	Mean	Obs.	Mean	Obs.	Mean	Obs.	Mean	
Conventional SA	1168	1169	1119	1097	1177	1131	1076	1118	1128.88
	1295		1079		1096		1160		
	1161		1061		1135		1103		
	1184		1116		1133		1081		
	1148		1133		1124		1118		
	1058		1075		1121		1172		
Adaptive SA	1171	1134	1166	1116	1168	1137	1158	1139	1131.63
	1122		1107		1115		1165		
	1109		1144		1174		1115		
	1150		1089		1137		1127		
	1129		1087		1089		1149		
	1125		1104		1141		1118		
Adaptive Tabu-SA	1137	1125	1061	1077	1079	1108	1152	1103	1103.17
	1146		1080		1130		1114		
	1105		1090		1113		1093		
	1154		1079		1098		1076		
	1116		1086		1127		**1057**		
	1091		1066		1100		1126		
Mean	1142.72		1096.78		1125.39		1120.00		1121.22

Optimal makespan = 1032

Table 4.6(a) Computational results for the job shop scheduling test problem LA23 (15×10)

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F_0	p-Value	$F_{0.05}$
Methods	11826.86	2	5913.43	4.98	0.0100	3.15
Seeds	19415.44	3	6471.81	5.44	0.0022	2.76
Interaction	5997.47	6	999.58	0.84	0.5432	2.25
Error	71302.67	60	1188.38			
Total	108542.44	71				

Table 4.6(b) Analysis of variance table for problem LA23 (15×10)

	$\overline{C}_{i\cdot} - \overline{C}_{j\cdot}$	Significant?
Conventional SA vs. Adaptive SA	-2.75	No
Adaptive SA vs. Adaptive Tabu-SA	28.46	Yes, $\overline{C}_{2\cdot} > \overline{C}_{3\cdot}$
Conventional SA vs. Adaptive Tabu-SA	25.7	Yes, $\overline{C}_{1\cdot} > \overline{C}_{3\cdot}$

$T_{0.05} = 23.92$

Table 4.6(c) Multiple-comparison Tukey's test for problem LA23 (15×10)

In order to have a sound analysis, we begin with the interaction hypothesis $H_0^{(3)}$. The third row in the above ANOVA tables show that $F_0 < F_{0.05}$ indicating there is no significant interaction between two factors: methods and start points in the set of tested problems. This will allow us to proceed to the next two tests that evaluate the effects of two factors.

The first two rows of the ANOVA tables are related to $H_0^{(1)}$ and $H_0^{(2)}$, respectively. Based on these analyses, for all test problems, the hypothesis $H_0^{(1)}$ is rejected indicating that there are at least two methods having a significant difference between their mean values. Such a conclusion is supported by p-values, as they are significantly less than $\alpha = 0.05$. For the effect of start points, however, such a solid conclusion cannot be drawn. For three test problems LA15, MT10 and LA23, $H_0^{(2)}$ is rejected suggesting that there is a significant difference between pairs of seeds while for the other three problems the effects of seed levels are not significant.

Having significant difference between methods alone does not provide enough information as to which method is superior; therefore, a paired comparison is required. A multiple-comparison Tukey's test is conducted for all test problems. The paired comparison of the two proposed methods with the conventional SA method shows that the adaptive SA produces significantly better solutions than the conventional SA in 4 out of 6 test problems, and the adaptive tabu-SA method outperforms the conventional SA in all six instances. Although the mean values of the computational results indicate a better performance by the adaptive tabu-SA as compared to the adaptive SA, the multiple-comparison Tukey's test does not show a significant difference except in the last problem LA23.

Based on these observations, one may conclude that the improvement caused by adding the tabu list to SA method is somewhat lighter than that of the proposed temperature control scheme. Nonetheless, the statistical analysis clearly shows that the combination of tabu list and the proposed temperature control function has significant effect on the performance of the search as reflected in the results obtained by the adaptive tabu-SA method.

4.2 Convergence

Computational results indicate that the adaptive tabu-SA is significantly superior to the conventional simulated annealing. To obtain further insights into the convergence behaviour of this method, from this point on, we focus on the adaptive tabu-SA method as compared to the benchmark SA method. Throughout extensive computations, it was noticed that the adaptive tabu-SA continuously improves the solution regardless of the search maturity. This behaviour is depicted in Figure 4.1 that compares the convergence of the conventional SA and the adaptive tabu-SA methods for 10 minutes search on the test problem MT10.

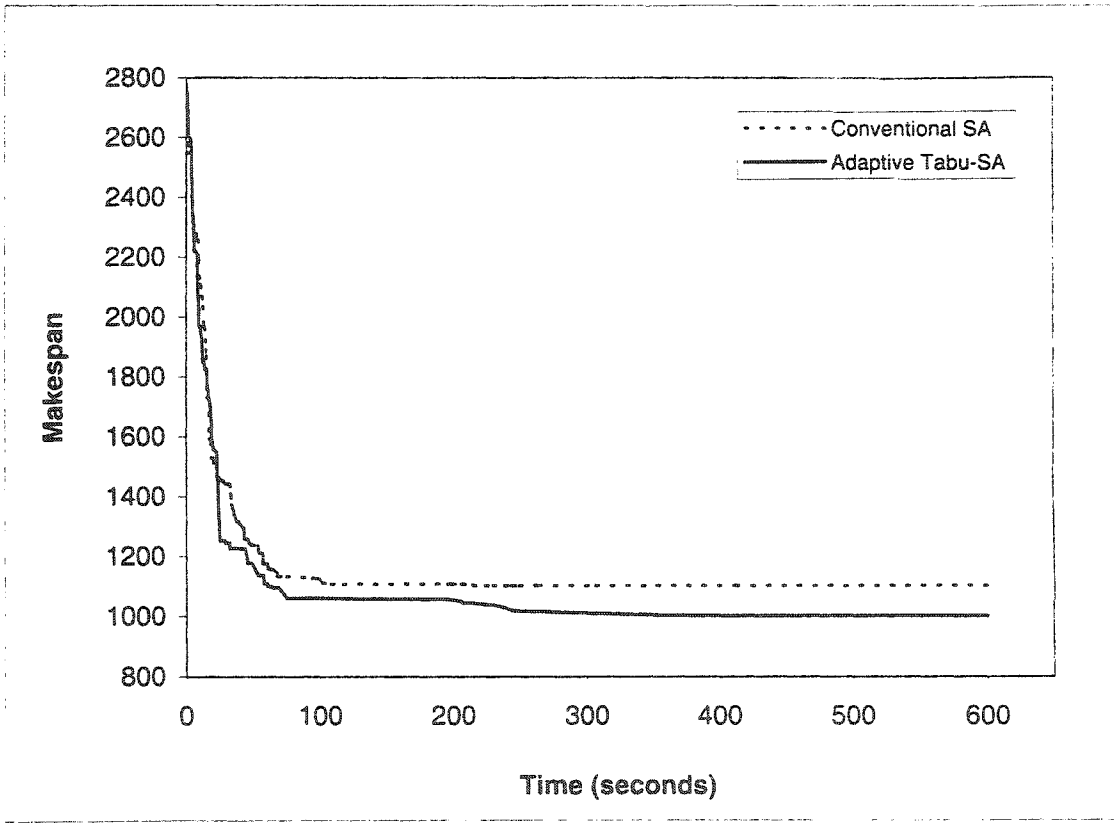
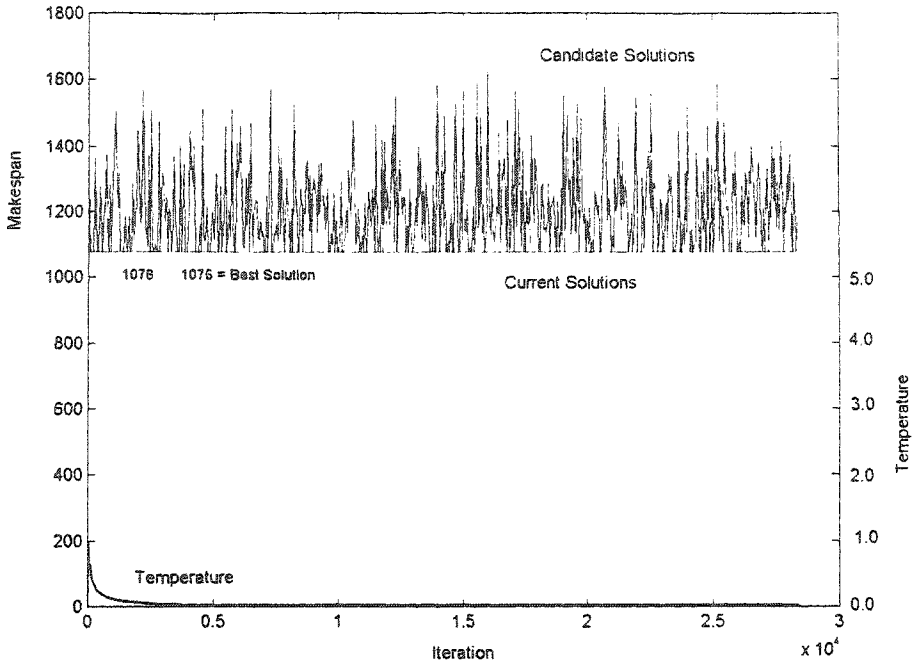


Figure 4.1 Comparison of the convergence of SA and the adaptive tabu-SA methods

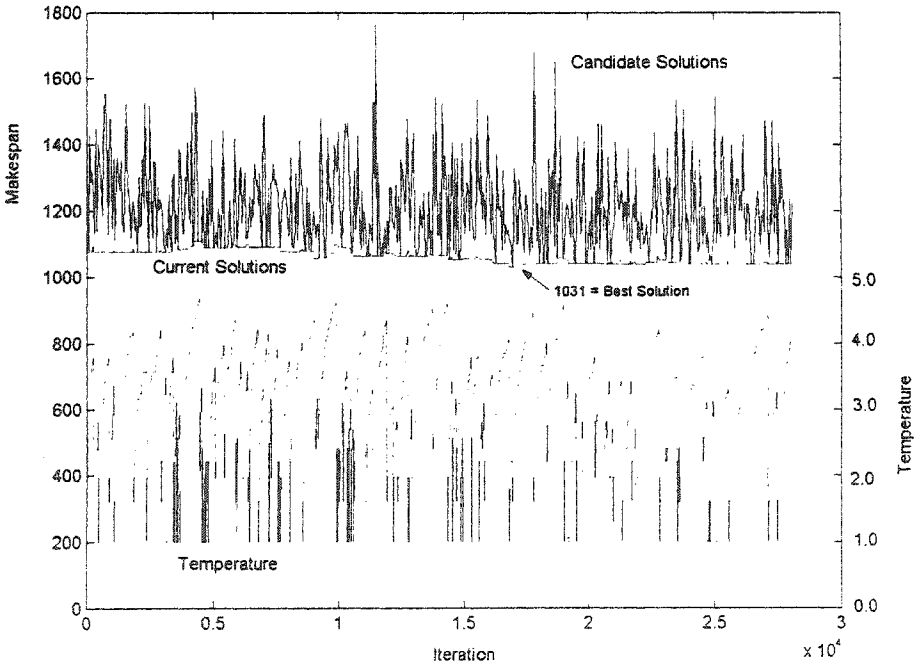
In this example, the last improvement of SA method was seen at 215 seconds while the adaptive tabu-SA continued the improvement until 412 seconds.

Further examination of the convergence curves reveals that the conventional SA method could freeze sooner if the search starts from a near optimal solution. To investigate this, both methods were tested on a near optimal solution of the test problem ORB4. An improved solution with makespan 1078 was generated and used as the seed solution for both methods. Figure 4.2 illustrates a 10-minute search of the two methods starting from this point. This seed is assumed to be relatively near optimal solution because its makespan is much smaller than that of many randomly generated seeds. For instance, the makespan of the seed solutions used in Table 4.3(a) were all in the range of 3604 and 3879 whereas the optimal makespan for ORB4 is reported to be 1005.

As is evident from the figure in the conventional SA method, the current solution stays at 1078 level until it finds the better solution of 1076 (as marked by an asterisk) and stays at this level for the rest of the search. In the adaptive tabu-SA method, however, the current solution moves continuously up and down and therefore searches a variety of solutions. This is mainly due to the difference in the temperature control mechanism of the two methods. In the SA method, the temperature drops constantly from the beginning of the search. Therefore, by the time the search hits a new solution, it has lost most of its energy and may hardly move upward. In other words, after losing the temperature, the search becomes near-deterministic. The adaptive tabu-SA method, however, fluctuates the temperature frequently regardless of the length of the search. Therefore, there is always a chance for uphill moves.



(a) Conventional SA



(b) Adaptive tabu-SA

Figure 4.2 Comparison of the conventional SA and the adaptive tabu-SA methods

4.3 Chapter Summary

In this study, we proposed two adaptations to the standard simulated annealing based on a new temperature control scheme in conjunction with a tabu list. To test the performance of these proposed methods, they were applied to a number of well-known job shop scheduling problems selected from the literature. The distinctive feature of these methods is a new temperature-control scheme that changes based on the characteristics of the search trajectory. In addition to the new temperature control mechanism, in one of the proposed methods we used a hybrid tabu-simulated annealing approach to reduce the chance of cycling and revisits. The comparisons of the proposed methods and the conventional simulated annealing clearly demonstrate that both adaptive-SA and adaptive tabu-SA methods are able to find better solutions than conventional simulated annealing. This conclusion is supported by our statistical analysis.

CHAPTER 5: APPLICATION OF ADAPTIVE-SIMULATED ANNEALING TO HOLDING CONTROL PROBLEM

In this chapter, we use the adaptive-SA method to solve the holding control problem described in chapter 3. The model is a non-linear mixed integer program intended for real-time use. To apply the model, we need an efficient algorithm that can solve the model optimally or near optimally in a short period of time. In previous chapter, the performance of the adaptive-SA method was tested by using several classical job shop problems. The computational result indicate that the algorithm outperform conventional simulated annealing.

5.1 Holding Control Problem by Adaptive-SA

The holding control problem studied in chapter 3 is to decide which bus (buses) has (have) to be held at any stop along the route to minimize the total passengers' waiting time. The decision variable is departure time of buses in a number of downstream stops. The problem formulation is given in equations (3.5)-(3.18) the first four constraints calculate the number of passengers on the bus using departing any stop. The next four constraints restrict the earliest departure time of each bus from the downstream stops. Constraint (3-13) calculated the demand for bus i at stop k as function of load on the bus departing from previous stop, the number boarding and alighting at stop k, and the number of passengers left behind by previous bus. The number of passengers left behind at stop k is calculated by constraint (3-

14) as a function of the demand for bus i at stop k and bus capacity. The formulation is a non-linear mixed integer program that can't be solved quickly by commercial software packages.

Simulated annealing is a local search technique that has been widely employed to solve many combinatorial optimization problems. Similar to other search methods, this technique has a unique search mechanism that allows it to escape local minima. In conventional simulated annealing, the temperature declines constantly, providing the search with a higher transition probability in the beginning of the search and lower probability toward the end of the search. As a lower temperature contributes to a lower transition probability, the end of the search has less chance of escaping local minima while in the beginning, the search is more likely to move out of local minimum solutions. Such an approach could be useful if the local minima are near the start point, but may not lead to a near optimal solution if some local minima are encountered at a relatively low temperature toward the end of the search. To alleviate this difficulty, we proposed an adaptive simulated annealing method that takes into consideration the characteristics of the search trajectory (Chapter 4).

5.1.1 Elements of adaptive-SA algorithm

The most important elements of the adaptive-SA algorithm are defined as follows:
Cost function: The cost function is the total passengers' waiting time.
Neighbouring solution: A neighbour is generated by adding (deducting) a prescribed holding time value to (from) the departure time of a randomly selected bus.
Temperature control function: We used the following temperature control function:

$$\theta_i = 1 + \ln(1 + r_i) \tag{5.1}$$

The initial value of r_i is zero, thus the initial temperature $\theta_0 = 1$. The predicted departure time of the buses from downstream stops based on current location of each bus is considered to be an initial feasible solution. However, the initial solution could also be generated randomly. The search begins with an initial feasible solution. The cost function for the initial solution is calculated and considered as the best cost. To generate a neighbouring solution, first one bus is selected randomly, then a holding value is generated randomly from a

reasonable range of integer numbers (i.e., 1,2,...,15). The holding value is either added to or deducted from the departure time of the bus. If the generated neighbour is feasible, the value of the cost function for the candidate solution is calculated and will be compared with the cost of the initial (current) solution. In case of improvement, the candidate solution will be replaced with the initial solution. If the cost of candidate solution is greater than that of the initial solution, the value of the transition probability will be compared with a uniform random number. If the probability value is greater than or equal to the random number, then the transition to the worse solution is accepted. Otherwise, another solution in the neighbourhood will be generated and evaluated. Each time a solution is accepted, the value of the temperature function must be decided. This process will continue until the stop criterion is satisfied.

The algorithm was coded in visual basic and implemented on a Pentium III with 450 MHZ CPU. The average computational time of the procedure is about 5 seconds for a problem with 26 stops and 18 buses described in the next section.

5.2 Numerical Examples

To illustrate the application of the adaptive-SA holding control model, the following two examples are provided.

Example 5.1

Consider a case where there are seven buses operating on a route consisting of four intermediate stops and two terminals as illustrated in Figure 5.1(a). The planned headway is 4 units of time, $H = 4$, and a rolling horizon of 13 units of time, $T^h = 13$. Then, the length of rolling horizon has resulted in $\tilde{N}_i = 3$ for all buses. Three cases might arise. These cases are discussed below.

Case I: Regular Service

Assume that the positions of buses at clock-time 5 are as described in Figure 5.1(a). Also assume that the information transmitted at that time includes: (1) the index of the last visited

stop, k-1, with respect to bus i (2) bus i actual departure time from the last visited stop, (3) bus i predicted departure time from downstream stops, and (4) load on bus when departing the last visited stop, with values summarized in Table 5.1(a).

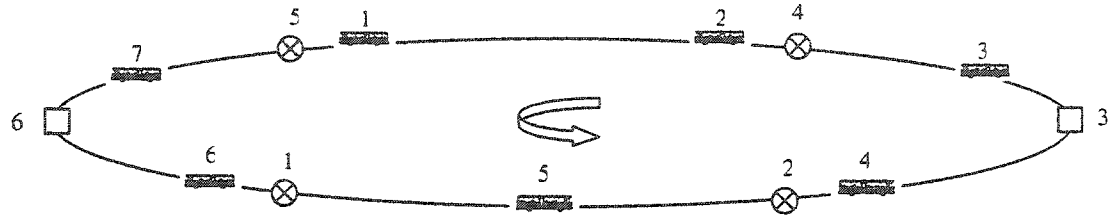


Figure 5.1(a) Stops and bus positions for Example 5.1(*case* I)

It is worth to note that, in order to generate those values for actual and predicted departure times of buses, we assumed bus 1 has departed from stop 4 at time 0. Since the given expected running time between stop 4 and stop 5 is 6 units of time (Table 5.1(b)), it is expected to reach the stop 5 at time 6. Adding 1 unit of time to the arrival time of the bus at stop 5 as the expected dwell time, the predicted departure time of bus 1 from stop 5 is at time 7. In the same way, the predicted departure time of bus 1 from the rest of the downstream stops, 6, and 1 is respectively at time 11 and 15. Using the value of planned headway, H, and the expected running times between stops, those information for the reminder buses are calculated. Table 5.1(b), summarizes the further information including: (1) passenger arrival rate, (2) passenger alighting fraction, and (3) the running times of buses between two consecutive stops, where this information could be determined from available historical data.

Then, substituting these values into the mathematical model described by equations (3.5)-(3.18), the cost function, which computes the total passengers waiting time, yields a value of 760 units of time. This value represent the minimum value of passenger's waiting time in the next 13 units of time (T^{h}) for the route described above.

	Stops:	*1*	*2*	*3*	*4*	*5*	*6*
Bus 1	The last visited stop				*		
	Actual departure time from the last visited stop				0		
	Predicted departure time from down stream stops	15				7	11
	Load on bus when departing the last visited stop				12		
Bus 2	The last visited stop				*		
	Actual departure time from last visited stop				4		
	Predicted departure time from down stream stops	19				11	15
	Load on bus when departing last visited stop				17		
Bus 3	The last visited stop			*			
	Actual departure time from last visited stop			4			
	Predicted departure time from down stream stops				8	15	19
	Load on bus when departing last visited stop			15			
Bus 4	The last visited stop		*				
	Actual departure time from last visited stop		4				
	Predicted departure time from down stream stops			8	12	19	
	Load on bus when departing last visited stop		18				
Bus 5	The last visited stop	*					
	Actual departure time from last visited stop	3					
	Predicted departure time from down stream stops		8	12	16		
	Load on bus when departing last visited stop	22					
Bus 6	The last visited stop						*
	Actual departure time from last visited stop						3
	Predicted departure time from down stream stops	7	12	16			
	Load on bus when departing last visited stop						14
Bus 7	The last visited stop					*	
	Actual departure time from last visited stop					3	
	Predicted departure time from down stream stops	11	16				7
	Load on bus when departing last visited stop					14	

Table 5.1(a) Buses data for Example 5.1

Stop:	*1*	*2*	*3*	*4*	*5*	*6*
Passenger arrival rate	4	4	5	4	4	6
Passenger alighting fraction	0.2	0.3	1	0.2	0.3	1
Running time	3	4	3	3	6	3

Table 5.1(b) Stops data for Example 5.1

Case II: One bus ahead of schedule

Now, assume for the same route discussed in *case* I, Bus 2 has been running ahead of schedule. The position of the buses on the route is as depicted in Figure 5.1(b). Under such condition, the new transmitted information related to bus 2 is summarized in Table 5.2. Substituting these values into the mathematical model, the total waiting time for passengers, yields a value of 928 units of time. Indicating that if the controller does nothing, the expected total passengers waiting time will increase by 168 units of time (928-760=168) in compare with the situation in which bus headways are even (*case* I).

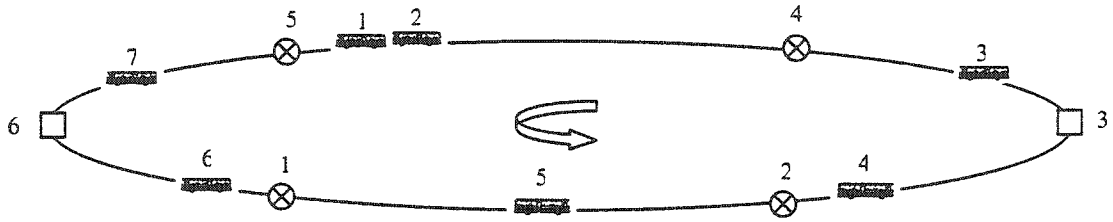


Figure 5.1(b) Stops and bus positions for Example 5.1(*case* II)

	Stops:	*1*	*2*	*3*	*4*	*5*	*6*
Bus 2	The last visited stop				*		
	Actual departure time from the last visited stop				2		
	Predicted departure time from down stream stops	15				7	11
	Load on bus when departing the last visited stop				17		

Table 5.2 New transmitted information for bus 2 in Example 5.1 (*case* II)

Applying the proposed holding model to the problem. The best solution found after average 3 seconds computational time is to hold bus 2 at stop 5 for 3 units of time. By doing so, the total expected passenger' waiting time yields the value 796 units of time. Indicating that 79% (vis., 1-(796-760)/(928-760)) of the additional passengers' waiting time caused by service irregularity has been reduced. Computational results have been summarized in Table 5.3.

	Regular service		Irregular service	
	Do nothing	Holding	Do nothing	Holding
Total waiting time	760	-	928	796
Additional waiting time	-	-	168	36
Percentage saving	-	-	0	79%

Table 5.3 Computational results for Example 5.1(*case* II)

Case III: Two buses ahead of schedule

Consider a scenario for the same route described in *case* I where bus 2 and bus 5 are running ahead of schedule. Figure 5.1(c) illustrates the position of the buses on the route at clock-time 5.

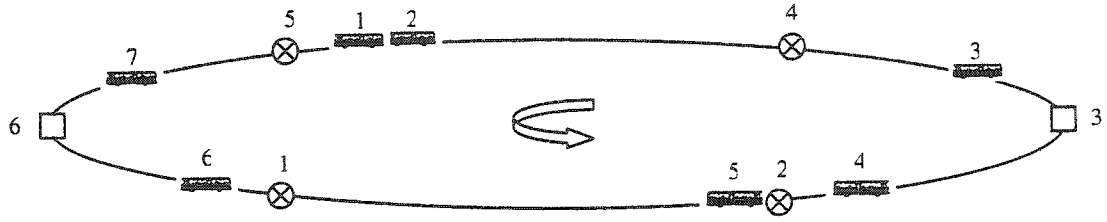


Figure 5.1(c) Stops and bus positions for Example 5.1(*case* III)

	Stops:	*1*	*2*	*3*	*4*	*5*	*6*
Bus 2	The last visited stop				*		
	Actual departure time from the last visited stop				2		
	Predicted departure time from down stream stops	15				7	11
	Load on bus when departing the last visited stop				17		
Bus 5	The last visited stop	*					
	Actual departure time from last visited stop	3					
	Predicted departure time from down stream stops		6	10	14		
	Load on bus when departing last visited stop	22					

Table 5.4 New transmitted information for bus 2 and bus 5 in Example 5.1 (*case* III)

Table 5.4 summarized the new transmitted information for bus 2 and bus 5. Substituting these values into the mathematical model, the total waiting time for passengers, yields a value of 940 units of time. Indicating that if the controller does nothing, the expected total

passengers waiting time will increase by 168 units of time (940-760=180). The best solution found after 3 seconds computational time is as follows: bus 2 has to be held for 3 units of time at stop 5. Bus 5 also has to be for 2 unit of time at stop 2. As a result, the total passenger waiting time could reduce to 796 times unit. Indicating that 80%(vis., 1-(796-760)/(940-760)) of the additional passengers' waiting time caused by service irregularity has been reduced (see Table 5.5).

	Regular service		Irregular service	
	Do nothing	Holding	Do nothing	Holding
Total waiting time	760	-	940	796
Additional waiting time	-	-	180	36
Percentage saving	-	-	0	80%

Table 5.5 Computational results for Example 5.1(*case* III)

Example 5.2

Consider a case where there are eighteen buses operating on a route consisting of twenty four intermediate stops and two terminals as illustrated in Figure 5.2(a). The planned headway is 6 units of time, $H = 6$, and a rolling horizon of 20 units of time, $T^h = 20$. Then, the length of rolling horizon has resulted in $\tilde{N}_i = 5$ for all buses. Two cases might arise. Either all buses are running on schedule or there is one or more headway irregularity in the system. These cases are discussed below.

Case I: Regular Service

Assume that the positions of buses at clock-time 5 are as described in Figure 5.2(a). Also assume that the information transmitted at that time includes: (1) the index of the last visited stop, k-1, with respect to bus i (2) bus i actual departure time from the last visited stop, (3) bus i predicted departure time from downstream stops, and (4) load on bus when departing the last visited stop, with values summarized in Table 5.6(a). Table 5.6(b), summarizes the further information including: (1) passenger arrival rate, (2) passenger alighting fraction, and (3) the running times of buses between two consecutive stops, where this information could

be determined from available historical data. Then, substituting these values into the mathematical model described by equations (3.5)-(3.18), the cost function, which computes the total waiting time for passengers, yields a value of 3289 units of time. This value represent the minimum value of passenger's waiting time in the next 20 unit of time (T^h) for the route described above with planned headway, $H = 6$, and eighteen buses on the service.

Case II: Irregular Service

Now, assume Bus 16 has been running head of schedule. The position of the buses on the route is as depicted in Figure 5.2(b). Under such condition, the new transmitted information related to bus 16 is summarized in Table 5.7. Substituting these values into the mathematical model described by equations (3.5)-(3.18), the cost function, which computes the total waiting time for passengers, yields a value of 3390 units of time. Indicating that if the controller does nothing, the expected total passengers waiting time will increase by 101units of time (3390-3289=101).

Applying the proposed holding model to the problem. The best solution found after average 3 seconds computational time is to hold bus 16 at stop 10 for 2 units of time. By doing so, the total expected passenger' waiting time yields the value 3342 units of time. Indicating that 47% (vis., 1-(3342-3289)/(3390-3289)) of the additional passengers' waiting time caused by service irregularity has been reduced (see table 5.8).

	Regular service		Irregular service	
	Do nothing	Holding	Do nothing	Holding
Total waiting time	3289	-	3390	3342
Additional waiting time	-	-	101	53
Percentage saving	-	-	0	47.5%

Table 5.8 Computational results for Example 5.2(*case* II)

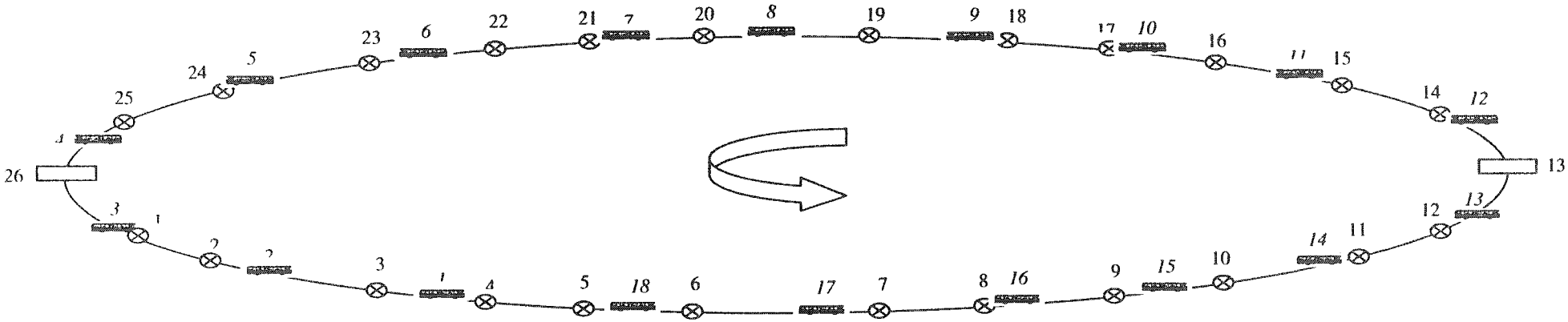


Figure 5.2(a) Stops and bus positions for Example 5.2(*case* I)

	Stops:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Bus 1	The last visited stop			*																							
	Actual departure time from the last visited stop			3																							
	Predicted departure time from down stream stops				7	10	14	19	23																		
	Load on bus when departing the last visited stop			21																							
Bus 2	The last visited stop		*																								
	Actual departure time from the last visited stop		4																								
	Predicted departure time from down stream stops			9	13	16	20	25																			
	Load on bus when departing the last visited stop		20																								
Bus 3	The last visited stop																										*
	Actual departure time from the last visited stop																										2
	Predicted departure time from down stream stops	6	10	15	19	22																					
	Load on bus when departing the last visited stop																										24
Bus 4	The last visited stop																									*	
	Actual departure time from the last visited stop																									4	
	Predicted departure time from down stream stops	12	16	21	25																						8
	Load on bus when departing the last visited stop																									20	
Bus 5	The last visited stop																							*			
	Actual departure time from the last visited stop																							1			
	Predicted departure time from down stream stops	18	22																						6	10	14
	Load on bus when departing the last visited stop																							19			
Bus 6	The last visited stop																						*				
	Actual departure time from the last visited stop																						3				
	Predicted departure time from down stream stops	24																						7	12	16	20
	Load on bus when departing the last visited stop																						22				
Bus 7	The last visited stop																				*						
	Actual departure time from the last visited stop																				2						
	Predicted departure time from down stream stops																					6	9	13	18	22	
	Load on bus when departing the last visited stop																				20						
Bus 8	The last visited stop																			*							
	Actual departure time from the last visited stop																			3							
	Predicted departure time from down stream stops																				8	12	15	19	24		
	Load on bus when departing the last visited stop																			21							

Table 5.6(a) Buses data for Example 5.2

	Stops:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Bus 9	The last visited stop																		*								
	Actual departure time from the last visited stop																		5								
	Predicted departure time from down stream stops																			9	14	18	21	25			
	Load on bus when departing the last visited stop																		18								
Bus 10	The last visited stop																*										
	Actual departure time from the last visited stop																2										
	Predicted departure time from down stream stops																	6	11	15	20	24					
	Load on bus when departing the last visited stop																23										
Bus 11	The last visited stop															*											
	Actual departure time from the last visited stop															4											
	Predicted departure time from down stream stops																8	12	17	21	26						
	Load on bus when departing the last visited stop															14											
Bus 12	The last visited stop													*													
	Actual departure time from the last visited stop													2													
	Predicted departure time from down stream stops														6	10	14	18	23								
	Load on bus when departing the last visited stop													24													
Bus 13	The last visited stop												*														
	Actual departure time from the last visited stop												4														
	Predicted departure time from down stream stops													8	12	16	20	24									
	Load on bus when departing the last visited stop												21														
Bus 14	The last visited stop										*																
	Actual departure time from the last visited stop										2																
	Predicted departure time from down stream stops											6	10	14	18	22											
	Load on bus when departing the last visited stop										23																
Bus 15	The last visited stop									*																	
	Actual departure time from the last visited stop									4																	
	Predicted departure time from down stream stops										8	12	16	20	24												
	Load on bus when departing the last visited stop									22																	
Bus 16	The last visited stop								*																		
	Actual departure time from the last visited stop							1	5																		
	Predicted departure time from down stream stops									10	14	18	22	26													
	Load on bus when departing the last visited stop								18																		

Table 5.6(a) Buses data for Example 5.2

	Stops:	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*
Bus 17	The last visited stop						*																				
	Actual departure time from the last visited stop						2																				
	Predicted departure time from down stream stops							7	11	16	20	24															
	Load on bus when departing the last visited stop						20																				
Bus 18	The last visited stop					*																					
	Actual departure time from the last visited stop					4																					
	Predicted departure time from down stream stops						8	13	17	22	26																
	Load on bus when departing the last visited stop					22																					

Table 5.6(a) Buses data for Example 5.2

Stops:	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*
Passenger arrival rate	1.5	1.5	1.5	2	1.5	2	1.5	1.5	2	2.5	2	2	4	2	2	2	2.5	2.5	2	2	1.5	2	1.5	1.5	1.5	4
Passenger alighting fraction	0.1	0.1	0.2	0.2	0.3	0.2	0.3	0.2	0.2	0.2	0.3	0.2	1	0.1	0.1	0.1	0.2	0.2	0.2	0.3	0.2	0.3	0.2	0.1	0.1	1
Running time	3	3	4	3	2	3	4	3	4	3	3	3	3	3	3	3	3	4	3	4	3	2	3	4	3	3

Table 5.6(b) Stops data for numerical Example 5.2

	Stops:	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*
Bus 16	The last visited stop									*																	
	Actual departure time from the last visited stop									5																	
	Expected departure time from down stream stops										9	13	17	21	23												
	Load on bus when departing the last visited stop									10																	

Table 5.7 New transmitted information for bus 16 in Example 5.2 (*case* II)

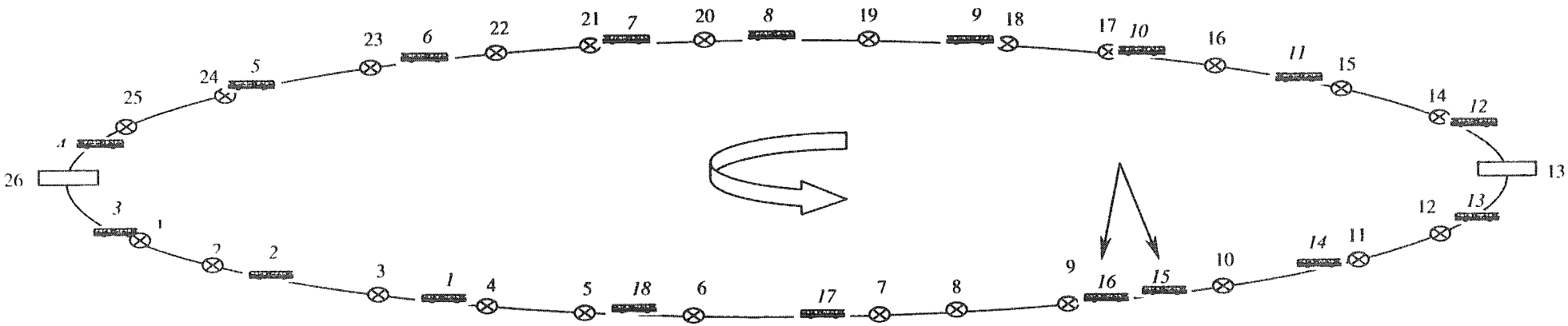


Figure 5.2(b) Stops and bus positions for Example 5.2 (*case* II)

5.3 Chapter Summary

In this chapter, we applied the adaptive-SA algorithm to holding control problem described in chapter 3. The model and solution algorithm were coded in Visual Basic and implemented on a Pentium III with 450 MHZ CPU. Two numerical examples were generated. The first example describes a route with 6 stops in which 7 buses are operating. In the second example, a route was depicted with 26 stops and 18 buses on service. To investigate the effect of holding policy on passenger waiting time, in both examples several possible cases for which holding might be considered as a control policy were investigated. Computational results indicate that by applying holding control, the additional waiting time caused by headway irregularity in the service could reduce significantly between 47% and 80%. Furthermore, The average computational time of the solution procedure was about 5 seconds for the problem with 26 stops and 18 buses. Such small computational time clearly meet the needs of a real-time control.

CHAPTER 6: CONCLUSIONS AND FUTURE RESEARCH

6.1 Concluding Remarks

In this thesis, we have investigated service monitoring and operations control in public transit systems. We first studied several methods that have been frequently employed by transit agencies to measure the performance of a transit system. These methods could be categorized as either headway-based or schedule-based measures. In general, depending on the frequency of the service, an appropriate policy is chosen to measure the service reliability. Unfortunately, there is no concession between researchers about the definition of low and high frequency. Furthermore, when the headway is in medium range, it is difficult to select between headway and schedule based approaches. We were able to present an integrated performance measure model that could simultaneously address both headway and schedule-based services. This facilitates applying the model in a responsive manner to any bus transit service regardless of its frequency; e.g., high, medium, or low. To measure schedule or headway adherences, this study adopted predicted rather than actual arrival times. The advantage of doing so is that it allows the operation mangers to avoid major service interruptions by proactively taking necessary corrective actions in due course.

In the second part of this thesis, we studied several types of real-time control strategies. These strategies are applied to remedy the service disruptions as they occur. Among these strategies, holding control has attracted more attention in the literature. This strategy also has been used frequently by transit agencies to reduce the headway variations and passengers'

waiting time. We presented deterministic formulations to model holding control problem assuming real-time information about the location of vehicles on the route is available. The model has the flexibility of adapting both high-occupancy and low-occupancy of vehicles. The objective is to minimize the total passenger's waiting time with the departure time of vehicles as decision variable. Unlike a prior study on holding strategy that employed a sequential control process (i.e., considering a set of vehicles at a time) that might eliminate interaction between vehicles, the proposed model considers all vehicles in the system simultaneously.

Finally, due to complexity of the problem, a heuristic approach based on simulated annealing was developed. The proposed heuristics uses an adaptive temperature control that tunes the temperature according to the search path. The efficiency of the solution algorithm was compared with conventional simulated annealing by applying both methods to several classical job shop scheduling problems.

6.2 Suggestions for Future Research

Real-time monitoring and operations control in public transit system is an extensive area; and this thesis is in no way a complete coverage of this topic. Possible future researches based on the work presented here are as follows:

6.2.1 Extension of the performance measurement model

The output of the proposed model reflects the efficiency of the system from the operator's point of view. It would be better to extend the model such that passenger waiting time also be considered as a factor in measuring the performance of the system.

6.2.2 Extension of the holding control model

The following suggestions are given as the possible future research on holding control problem:

a) Although in this thesis we used several examples with different sizes to test the effectiveness of the holding model, the performance of the model has to be verified using collected data from bus routes.

b) In the model the objective function minimize the total passenger waiting time at stops and does not include the on-board waiting time. It would be useful to compare the effectiveness of the holding control using other criteria.

c) We developed a deterministic model to study the behaviour of a transit system. To do so, the stochastic elements of the model such as bus running times between stops have been approximated by their expected values. It would be more realistic to extend this deterministic model to a stochastic model that includes random inputs such as demand, arrivals, speed, etc.

d) In the model presented here, routes are assumed to be independent. However, in reality, routes are connected at some transfer points to allow passengers to continue their trips toward their destinations. Holding control could be helpful to facilitate such transfers. An extended model that incorporates the waiting time of passengers at transfer points could be very useful.

REFERENCES

Abkowitz, M. D. (1978) Transit service reliability, Cambridge, MA: USDOT transportation systems centre and multi-systems Inc. (NTIS No. UMTA/MA-06-0049-78-1).

Abkowitz, M. D., Eiger, A., Engelstein, I. (1986) Optimal control of headway variation on transit routes, *Journal of Advanced Transportation*, (20), pp. 73-88.

Abkowitz, M. D., Engelstein, I. (1984) Methods for maintaining transit service regularity, *Transportation Research Record*, (961), pp. 1-8.

Abkowitz, M., Tozzi, J. (1987) Research contributions to managing transit service reliability, *Journal of Advanced Transportation*, (21), pp. 47-65.

Abkowitz, M. D., Lepofsky, M. (1990) Implementing headway-based reliability control on transit routes, *Journal of Transportation Engineering*, 116, (1), pp. 49-63.

Adams, J., Balas, E., Zawack, D. (1988) The shifting bottleneck procedure for job shop scheduling, *Management Science*, 34, pp. 391-401.

Applegate, D., Cook, W. (1991) A computational study of job shop scheduling problem, *ORSA Journal on Computing*, 3, (2), pp. 149-156.

Balas, E., Lenstra, J. K., Vazacopoulos, A. (1995) The One-machine problem with delayed precedence constraints and its use in job shop scheduling, *Management Science*, 41, pp. 94-109.

Barnett, A. I. (1974) On controlling randomness in transit operations. *Transportation Science*, 8 (2), pp. 101-116.

Chien, S., Ding, Y. (1999) A dynamic headway control strategy for transit operations, *Proceedings of the 6th World Congress on Intelligent Transportation Systems*, Toronto, Canada.

Choong, S., Cole, M. H., Kutanoglu, E. (2001) Empty Container Management for Container-on-Barge (COB) Transportation: Planning Horizon Effects on Empty Container Management in a Multi-Modal Transportation Network, MBTC-2003, Mack-Blackwell Transportation Center, University of Arkansas, Fayetteville, Arkansas.

Clotfelter, C. (1993) The private life of public economics, *Southern Economic Journal*, 59, (4), pp. 579-596.

Dessouky, M. M., Hall, R., Nowroozi, A., Mourikas, K. (1999) Bus dispatching at timed transfer transit stations using bus tracking technology, *Transportation Research, Part C: Emerging Technologies*, (7), pp. 187-208.

Eberlein, X. J., Wilson, M. H. M., Barnhart, C., Bernstein, D. (1998) The real-time deadheading problem in transit operations control, *Transportation Research (Part B)*, 32, (2), pp. 77-100.

Eberlein, X., Wilson, N. H. M., Bernstein, D. (1999) Modeling real-time control strategies in public transit operations, *Lecture Note in Economics and Mathematical Systems* #471: Computer Aided Transit Scheduling, Wilson, N.H.M. (Ed.), Springer-Verlag Berlin Heidelberg, pp. 325-346.

Gidas, B. (1985) Non-stationary Markov chains and convergence of the annealing algorithm, J. Statis. Phys., 39, pp. 73-131.

Glover, F. (1977) Heuristics for integer programming using surrogate constraints, *Decision Sciences*, 8, pp. 156-166.

Hajek, B. (1986) Cooling schedules for optimal annealing, *Mathematics of Operations Research.*

Henderson, G., Kwong, P., Adkins, H. (1991) Regularity indices for evaluating transit performance, *Transportation Research Record*, (1297), pp. 3-9.

Hounsell, N., McLeod, F. (1998) AVL implementation application and benefits in the U.K., *Proceedings of the 77th annual meeting of the Transportation Research Board*, Washington, D.C.

Jain, A. S., Meeran, S. (1999) Deterministic job-shop scheduling: Past, present and future, *European Journal of Operational Research*, 113, (2), pp. 390-434.

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

Koffman, D. (1978) A simulation study of alternative real-time bus headway control strategies, *Transportation Research Record*, 663, pp. 41-46.

Lawrence, S. (1984) Supplement to resource constrained project scheduling, an experimental investigation of heuristic techniques. *Graduate School of Industrial Administration, Carnegie-Mellon University*, Pittsburgh, PA.

Lenstra, J. K., Rinnooy, K. A. H. G. (1979) Computational complexity of discrete optimization problems, *Annals of Discrete Mathematics*, 4, pp. 121-140.

Lin, T., Wilson, N. H. M. (1993) Dwell time relationships for light rail systems, *Transportation Research Record*, 1361, pp. 296-304.

Matsuo, H., Suh C. J., Sullivan, R. S. (1988) A controlled search simulated annealing method for the general job-shop scheduling problem. *Working Paper #03-04-88*, Graduate School of Business, The University of Texas at Austin, Austin.

Muth, J. F., Thompson, G. L. (1963) *Industrial Scheduling*, Prentice Hall: New Jersey.

Newell, G. F. (1998) The rolling horizon scheme of traffic signal control, *Transportation Research (Part A)*, 32, pp. 39-44.

O'Dell, S. W., Wilson, N. H. M. (1999) Optimal real-time control strategies for rail transit operations during disruptions, *Lecture Notes in Economics and Mathematical Systems* #471: Computer aided scheduling, Wilson N. H. M. (ED.), Springer-Verlag, pp. 299-323.

Osuna, E. E., Newell, G. F. (1972) Control strategies for an idealized public transportation system, *Transportation Science*, 16, (1), pp. 52-72.

Pinejames, R., Niemeyer, J., Chisholm, R. (1998) Transit scheduling: basic and advanced manuals, report 30, transit cooperative research program, *Transportation Research Board*, National Research Council, National Academy Press, Washington D.C.

Seneviratne, P. N., Tam, L., Javid, M. (1989) Scheduling fixed route bus services using simulation, *Proceedings of the International Conference on Microcomputers in Transportation*, pp. 1042-1 053.

Seneviratne, P. N. (1990) Analysis of on-time performance of bus services using simulation, *Journal of Transportation Engineering*, 166, (4), pp. 5 17-531.

Sterman, B., Schofer, J. (1976) Factors affecting reliability of urban bus services, *Transportation Engineering Journal*, pp. 147-159.

Taillard, E. D. (1994) Parallel taboo search techniques for the job shop scheduling problem. *ORSA Journal on Computing*, 6, (2), pp. 108-117.

Turnquist, M. A. (1978) Strategies for improving reliability of bus transit service, *Transportation Research Board, Transportation Research Record*, (818), pp. 7-13.

Turnquist, M. A., Blume, S. W. (1980) Evaluating potential effectiveness of headway control strategies for transit system, *Transportation Research Board, Transportation Research Record*, (746), pp. 25-29.

Vaessens, R. J. M., Aarts, E. H. L., Lenstra, J. K. (1996) Job shop scheduling by local search. *ORSA Journal on Computing*, 8, (3), pp. 302-317.

Van Laarhoven, P. J. M., Aarts, E. H. L., Lenstra, J. K. (1992) Job shop scheduling by simulated annealing, *Operation Research*, 40, pp. 113-126.

Van Laarhoven, P. J. M., Aarts, E. H. L. (1987) *Simulated annealing: Theory and applications*, Kluwer Ac. Publ., Dordrecht.

Wilson, N., Macchi, R. A., Fellows, R. E., Deckoff, A. A. (1992) Improving service on the MBTA Green Line through better operations control, *Transportation Research Record*, 1361, pp. 10-15.

Woodhull, J. (1987) Issues in on-time performance of bus systems, *unpublished manuscript.* Los Angeles, CA: Southern California Rapid Transit District.

Yamada, T., Nakano, R. (1995) Job shop scheduling by simulated annealing combined with deterministic local search, *Proceedings of the Metaheuristics International Conference*, Colorado, USA, pp. 344-349.

Zolfaghari, S., Liang, M. (1999) Jointly solving the group scheduling and machining speed selection problems: a hybrid tabu search and simulated annealing approach, *International Journal of Production Research*, 37, (10), pp. 2377-2397.

APENDIX I

Job shop scheduling test problems data

Processing times and operations sequence for the test problem LA7 (15×5)

	Operations Sequence				
Jobs	O1	O2	O3	O4	O5
J 1	1	5	2	4	3
J 2	1	2	5	4	3
J 3	4	1	3	2	5
J 4	1	2	5	4	3
J 5	4	2	1	3	5
J 6	2	3	1	5	4
J 7	3	2	1	5	4
J 8	3	4	5	1	2
J 9	5	1	3	2	4
J 10	5	1	2	4	3
J 11	5	1	2	3	4
J 12	4	3	1	5	2
J 13	5	2	1	3	4
J 14	5	1	4	3	2
J 15	5	2	1	3	4
	Processing Times				
J 1	47	57	71	96	14
J 2	75	60	22	79	65
J 3	32	33	69	31	58
J 4	44	34	51	58	47
J 5	29	44	62	17	8
J 6	15	40	97	38	66
J 7	58	39	57	20	50
J 8	57	32	87	63	21
J 9	56	84	90	85	61
J 10	15	20	67	30	70
J 11	84	82	23	45	38
J 12	50	21	18	41	29
J 13	16	52	52	38	54
J 14	37	54	57	74	62
J 15	57	61	81	30	68

Processing times and operations sequence for the test problem LA7 (20×5)

	Operations Sequence				
Jobs	O1	O2	O3	O4	O5
J 1	1	3	2	4	5
J 2	3	4	1	5	2
J 3	2	5	3	4	1
J 4	3	5	1	4	2
J 5	3	1	2	4	5
J 6	1	5	2	4	3
J 7	5	4	2	3	1
J 8	1	3	2	5	4
J 9	5	1	4	3	2
J 10	2	1	5	3	4
J 11	1	2	3	5	4
J 12	3	1	4	2	5
J 13	1	3	2	4	5
J 14	1	4	3	2	5
J 15	2	1	5	4	3
J 16	2	3	5	1	4
J 17	2	5	3	1	4
J 18	4	1	3	5	2
J 19	1	2	3	4	5
J 20	2	3	5	1	4
	Processing Times				
J 1	6	40	81	37	19
J 2	40	32	55	81	9
J 3	46	65	70	55	77
J 4	21	65	64	25	15
J 5	85	40	44	24	37
J 6	89	29	83	31	84
J 7	59	38	80	30	8
J 8	80	56	77	41	97
J 9	56	91	50	71	17
J 10	40	88	59	7	80
J 11	45	29	8	77	58
J 12	36	54	96	9	10
J 13	28	73	98	92	87
J 14	70	86	27	99	96
J 15	95	59	56	85	41
J 16	81	92	32	52	39
J 17	7	22	12	88	60
J 18	45	93	69	49	27
J 19	21	84	61	68	26
J 20	82	33	71	99	44

Processing times and operations sequence for the test problem ORB4 (10×10)

	Operations Sequence									
Jobs	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10
J 1	1	2	3	4	5	6	7	8	9	10
J 2	1	9	4	6	3	2	10	7	8	5
J 3	1	7	2	4	9	3	6	5	8	10
J 4	1	2	5	9	6	8	7	10	4	3
J 5	1	6	10	9	8	5	4	3	2	7
J 6	2	6	1	10	3	9	4	7	8	5
J 7	1	4	9	8	10	3	5	7	2	6
J 8	1	10	5	2	6	8	3	7	4	9
J 9	1	2	10	8	9	7	5	6	3	4
J 10	1	3	5	10	2	6	9	4	7	8
	Processing Times									
J 1	9	81	55	40	32	37	6	19	81	40
J 2	21	70	65	64	46	65	25	77	55	15
J 3	85	37	40	24	44	83	89	31	84	29
J 4	80	77	56	8	30	59	38	80	41	97
J 5	91	40	88	17	71	50	59	80	56	7
J 6	8	9	58	77	29	96	45	10	64	36
J 7	70	92	98	87	99	27	86	96	28	73
J 8	95	92	85	52	81	32	39	59	41	56
J 9	60	45	88	12	7	22	93	49	69	27
J 10	21	61	68	26	82	71	44	99	33	84

Processing times and operations sequence for the test problem LA16 (10×10)

	Operations Sequence									
Jobs	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10
J 1	2	7	10	9	8	3	1	5	4	6
J 2	5	3	6	10	1	8	2	9	7	4
J 3	4	3	9	2	5	10	8	7	1	6
J 4	2	4	3	8	9	10	7	1	6	5
J 5	3	1	6	7	8	2	5	10	4	9
J 6	3	4	6	10	5	7	1	9	2	8
J 7	4	3	1	2	10	9	7	6	5	8
J 8	2	1	4	5	7	10	9	6	3	8
J 9	5	3	9	6	4	8	2	7	10	1
J 10	9	10	3	5	4	1	8	7	2	6
	Processing Times									
J 1	21	71	16	52	26	34	53	21	55	95
J 2	55	31	98	79	12	66	42	77	77	39
J 3	34	64	62	19	92	79	43	54	83	37
J 4	87	69	87	38	24	83	41	93	77	60
J 5	98	44	25	75	43	49	96	77	17	79
J 6	35	76	28	10	61	9	95	35	7	95
J 7	16	59	46	91	43	50	52	59	28	27
J 8	45	87	41	20	54	43	14	9	39	71
J 9	33	37	66	33	26	8	28	89	42	78
J 10	69	81	94	96	27	69	45	78	74	84

Processing times and operations sequence for the test problem MT10 (10×10)

	Operations Sequence									
Jobs	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10
J 1	1	2	3	4	5	6	7	8	9	10
J 2	1	3	5	10	4	2	7	6	8	9
J 3	2	1	4	3	9	6	8	7	10	5
J 4	2	3	1	5	7	9	8	4	10	6
J 5	3	1	2	6	4	5	9	8	10	7
J 6	3	2	6	4	9	10	1	7	5	8
J 7	2	1	4	3	7	6	10	9	8	5
J 8	3	1	2	6	5	7	9	10	8	4
J 9	1	2	4	6	3	10	7	8	5	9
J 10	2	1	3	7	9	10	6	4	5	8
	Processing Times									
J 1	29	78	9	36	49	11	62	56	44	21
J 2	43	90	75	11	69	28	46	46	72	30
J 3	91	85	39	74	90	10	12	89	45	33
J 4	81	95	71	99	9	52	85	98	22	43
J 5	14	6	22	61	26	69	21	49	72	53
J 6	84	2	52	95	48	72	47	65	6	25
J 7	46	37	61	13	32	21	32	89	30	55
J 8	31	86	46	74	32	88	19	48	36	79
J 9	76	69	76	51	85	11	40	89	26	74
J 10	85	13	61	7	64	76	47	52	90	45

Processing times and operations sequence for the test problem LA23 (15×10)

	Operations Sequence									
Jobs	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10
J 1	8	6	9	3	5	7	4	2	10	1
J 2	7	2	5	6	3	4	8	9	10	1
J 3	1	6	9	10	4	7	5	8	3	2
J 4	6	3	4	7	5	8	9	10	2	1
J 5	10	5	7	8	1	3	9	6	4	2
J 6	7	6	2	8	9	5	1	3	10	4
J 7	8	1	9	5	3	2	10	4	7	6
J 8	4	6	10	2	9	3	5	7	1	8
J 9	1	8	3	9	5	7	6	2	10	4
J 10	3	2	8	7	5	1	6	4	10	9
J 11	9	3	6	1	8	4	2	5	10	7
J 12	3	10	1	2	9	7	4	8	6	5
J 13	5	10	8	7	1	6	3	2	9	4
J 14	3	6	7	5	4	10	8	2	9	1
J 15	2	9	8	7	4	10	3	6	5	1
	Processing Times									
J 1	84	58	77	44	97	89	5	58	96	9
J 2	21	87	15	39	81	85	31	57	73	77
J 3	40	71	34	82	70	22	10	80	48	49
J 4	75	17	7	72	11	62	47	35	91	55
J 5	20	12	71	67	64	94	15	50	75	90
J 6	93	93	57	70	77	58	52	29	7	68
J 7	56	95	48	26	82	63	36	27	87	6
J 8	76	15	78	8	41	36	30	84	36	76
J 9	75	13	81	29	54	82	88	78	40	13
J 10	6	26	32	64	54	52	82	6	88	54
J 11	62	67	32	62	69	61	35	72	5	93
J 12	78	90	85	72	64	63	11	82	88	7
J 13	28	11	50	88	44	31	27	66	49	35
J 14	14	39	56	62	97	66	69	7	47	76
J 15	18	93	58	47	69	57	41	53	79	64

APENDIX II

Computer programs

Main module

Global variables:

```
Public Type Idx
  Row As Integer
  Col As Integer
End Type
Public TempO As Idx
Public r As Integer
Public Temtur As Integer
Public nJob As Integer
Public nMachine As Integer
Public JD ( ) As Integer
Public JJ ( ) As Integer
Public DT ( ) As Integer
Public MD ( ) As Integer
Public CT ( ) As Integer
Public BCT( ) As Integer
Public bMinTime As Integer
Public CoTime As Integer
Public CoTmin As Integer
Public m As Integer
Public nebvalid As Boolean
Public Ra1 As Integer
Public Ra2 As Integer
Public Ra3 As Integer
Public Cnbhd As Integer
Public Moveno1 As Integer
```

```
Sub read_data1 (DFname1 As String)
        On Error GoTo ShErr:
        Open DFname1 For Input As #1
        Dim i As Integer, j As Integer
                Input #1, nJob
        nJob = nJob - 1
                Input #1, nMachine
'Define dynamic array boundaries
        ReDim JJ (0 To nJob, 0 To nMachine) As Integer
        ReDim JD (0 To nJob, 1 To nMachine) As Integer
For i = 0 To nJob
  For j = 1 To nMachine
                Input #1, JJ (i, j) 'Read data to array
                JD (i, j) = JJ (i, j)
  Next j
```

```
Next i
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub

_____________________________
Sub read_data2 (DFname2 As String) ' Reading sequence of the jobs
        On Error GoTo ShErr:
        Open DFname2 For Input As #1
        Dim j As Integer
                Input #1, nJob
                Input #1, nMachine
                nMachine = nMachine - 1
' Define dynamic array boundaries
        ReDim MD (1 To nJob, 0 To nMachine) As Integer
For j = 1 To nJob
  For k = 0 To nMachine
                Input #1, MD (j, k) ' Read data to array
  Next k
Next j
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub

_____________________________
Sub read_data3 (DFname3 As String)
        On Error GoTo ShErr:
        Open DFname3 For Input As #1
        Dim i As Integer
                Input #1, nJob
                Input #1, nMachine
        ReDim DT (1 To nJob, 1 To nMachine) As Integer
For i = 1 To nJob
          For k = 1 To nMachine
                Input #1, DT (i, MD (i, k)) ' Read data to array
     Next k
 Next i
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub

_____________________________
Sub Writedata (DFname4 As String)
        On Error GoTo ShErr:
        Open DFname4 For Output As #1
        Dim i As Integer
        Dim j As Integer
For i = 1 To nJob
```

```
    For j = 1 To nMachine
                        Print #1, Tab (j + 4); BCT(i, j) ' Read data to array
    Next j
Next i
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub
```

```
Public Sub StartPoint( )
        ReDim JD (0 To nJob, 0 To nMachine) As Integer
        Dim valid As Boolean
        Dim Rdm As Integer
        Randomize
For L = 1 To nMachine
                JD (0, L) = 0
Next L
  For i = 1 To  nJob
    Do
                Rdm = Int ((nJob) * Rnd + 1)
                 JD (i, 1) = Rdm
                k = 1
                Valid = True
      Do
                        If JD (i, 1) = JD (i - k, 1) Then
                        Valid = False
                         Else
                         k = k + 1
                         End If
     Loop Until k > i or valid = False
    Loop Until k > i
  Next i
  For i = 1 To  nJob
         For L = 2 To  nMachine
                        JD (i, L) = JD(i, 1)
        Next L
 Next i
End Sub
```

```
Public Sub Schedul ( )
        Dim tempComTime As Integer, EndTime As Integer, L As Integer
        Dim succss As Boolean
        Dim schedl As Boolean
        Dim a As Integer
        Dim b As Integer
        Dim e As Integer
        ReDim machinTime (1 To nMachine)
        ReDim jobTime (1 To nJob)
'schduling the jobs on machines and computing the complition time
        ReDim CT (0 To nJob, 0 To nMachine)
```

```
For i = 1 To  nJob
                MD (i, 0) = 0
Next i
For i = 1 To  nJob
        For L = 1 To  nMachine
                        CT (JD (i, L), 0) = 1
        Next L
Next i
For L = 1 To  nMachine
                        JD (0, L) = 0
Next L
For L = 1 To  nMachine
                        CT (0, L) = 1
Next L
nebvalid = True
e = 0
t = 0
tempComTime = 0
EndTime = 0
Do
  t = t + 1
  L = 0
  i = 0
  k = 0
  Do
    L = L + 1
    Do
      i = i + 1
      a = JD(i, L)
      Do
        k = k + 1
        b = MD(a, k)
        If L = b Then
          If CT (JD(i, L), MD(a, k)) = 0 Then
            If CT (JD(i, L), MD(a, k - 1)) > 0 Then
                If jobTime (JD(i, L)) > machinTime (MD(a, k)) Then
                    CT (JD(i, L), MD(a, k)) = jobTime (JD(i, L)) + DT(JD(i, L), MD(a, k))
                    succss = True
                    e = e + 1
                    tempComTime = CT(JD(i, L), MD(a, k))
                    jobTime (JD(i, L)) = CT(JD(i, L), MD(a, k))
                    machinTime (MD(a, k)) = CT(JD(i, L), MD(a, k))
                    If EndTime < tempComTime Then EndTime = tempComTime
                Else
                   CT (JD(i, L), MD(a, k)) = machinTime (MD(a, k)) + DT(JD(i, L), MD(a, k))
                    succss = True
                    e = e + 1
                    tempComTime = CT(JD(i, L), MD(a, k))
                    jobTime(JD(i, L)) = CT(JD(i, L), MD(a, k))
                    machinTime(MD(a, k)) = CT(JD(i, L), MD(a, k))
                    If EndTime < tempComTime Then EndTime = tempComTime
```

```
                End If
              Else
            succss = True
            End If
          Else
          schedl = True
          End If
        End If
      Loop Until k = nMachine or succss = True or schedl = True
      schedl = False
      k = 0
    Loop Until i = nJob or succss
    succss = False
    i = 0
  Loop Until L = nMachine
              If t > e Then
                    nebvalid = False
              End If
Loop Until e = nJob * nMachine or nebvalid = False
      If nebvalid Then CoTime = EndTime
End Sub
```

```
'Generating a random number to select a neighbour (by interchanging the position of two jobs)
Public Sub selctneibour( )
      Dim Temp As Integer
      Randomize Time
      Ra1 = Int (((nJob) * Rnd + 1))
      i = Ra1
      Ra2 = Int (((nMachine) * Rnd + 1))
      L = Ra2
      Temp = JD (i, L)
      Ra3 = Int (((nJob) * Rnd + 1))
      JD (i, L) = JD (Ra3, L)
      JD (Ra3, L) = Temp
'Generating a random number to select a neighbour (by interchanging the position of two adjacent jobs)
      Ra1 = Int (((nJob - 1) * Rnd + 1))
      i = Ra1
      Ra2 = Int (((nMachine - 1) * Rnd + 1))
      L = Ra2
      Temp = JD (i, L)
      JD (i, L) = JD (i + 1, L)
      JD (i + 1, L) = Temp
 End Sub
```

```
Public Sub Resetneibour( )      'Return the jobs position after selecting neighbour
      Dim Temp As Integer
      Temp = JD (Ra1, Ra2)
      JD (Ra1, Ra2) = JD (Ra3, Ra2)
      JD (Ra3, Ra2) = Temp
 'Return the jobs position after selecting neighbour
      Temp = JD (Ra1, Ra2)
```

```
        JD (Ra1, Ra2) = JD (Ra1 + 1, Ra2)
        JD (Ra1 + 1, Ra2) = Temp
End Sub
```

```
Public Sub BestSchdul ( )
        ReDim BCT (1 To nJob, 1 To nMachine)
' Results :(completion time of jobs on machines)
For i = 1 To nJob
         For j = 1 To nMachine
                        BCT (i, j) = CT(i, j)
         Next j
Next i
End Sub
```

```
Sub nextMove( )
        Dim e As Currency
        Dim CoTdf As Integer
        Dim EXx As Double
        Dim F As Double
        Dim MRdn As Currency
        Dim Success As Boolean
        Cnbhd = CoTime
        neibourNO = 0
Do
  Do
                 selctneibour
                Schedul
        If nebvalid = False Then
                 Resetneibour
                CoTime = Cnbhd
        End If
  Loop While nebvalid = False
                                Temtur = 1 + Log (1 + r)
                                CoTdf = CoTime - Cnbhd
                                 F = CoTdf / Temtur
                                 EXx = Exp (-F)
                If EXx >= 1 Then
                                Success = True
                Else
                                MRdn = Rnd
                                        If MRdn < EXx Then
                                                Success = True
                                        Else
                                                Resetneibour
                                                CoTime = Cnbhd
                                        End If
                 End If
Loop Until Success
End Sub
```

```
Sub SimulationSearch (maxMove As Integer)
        Dim SearchTime As Double, Start As Double, Finish As Double, MinTime As Double
         Start = Timer
        SearchTime = ViewDFrm.MaxMov.Text
        MinTime = Start
        CoTmin = CoTime
        S = 1
        r = 0
        m = 1
        Randomize Time
Do While Timer < Start + SearchTime
                        nextMove
                 If CoTime < Cnbhd Then
                        r = 0
                Else
                        r = r + 1
                End If
                        If CoTime < CoTmin Then
                                CoTmin = CoTime
                                BestSchdul
                                 bMinTime = Timer - Start
                                 Moveno1 = bMinTime
                                 Moveno1 = m
                        End If
                          If (m Mod 5) = 0 Then
                                        ViewDFrm.caLabel.Caption = "Calculating...." & Temtur
                                         ViewDFrm.caLabel.Refresh
                         End If
                         m = m + 1
Loop
End Sub
```

Form 1 (*Aform*) *codes:*

```
Private Sub ExitCmd_Click( )
End
End Sub

Private Sub RandStart_Click ( )
        strtpntcmd.Enabled = True
        ViewDFrm.Startpnt
End Sub

Private Sub Readdata1Cmd_Click ( )
         read_data1 DFname1.Text
         strtpntcmd.Enabled = True
 End Sub

Private Sub Readdata2Cmd_Click ( )
        read_data2 DFname2.Text
```

```
        readdata3cmd.Enabled = True
End Sub

Private Sub Readdata3Cmd_Click ( )
        read_data3 DFname3.Text
        RandStart.Enabled = True
        readdata1cmd.Enabled = True
End Sub

Private Sub strtpntcmd_Click ( )
        ViewDFrm.Show
        ViewDFrm.drawTable
End Sub
```

Form 2 (*ViewDFrm) codes:*

```
        Dim Tlns( ) As Line
        Public Sub drawTable( )
        On Error GoTo ShErr:
        Dim i As Integer, j As Integer
        Tabpic.Scale (-1, (nJob + 1))-((nMachine + 1), -1)
        Tabpic.Cls
        Tabpic.Line (0, 0)-(0, nJob)
        Tabpic.Line (0, 0)-(nMachine, 0)
        FillColor = 2
For i = 1 To nJob
        Tabpic.Line (0, i)-(nMachine, i)
Next i
        For i = 1 To nJob
                Tabpic.Line (i, 0)-(i, nJob)
        Next i
                For i = 1 To nJob
                        For j = 1 To nMachine
                                Tabpic.CurrentY = (nJob - i + 0.75) - 0.1
                                Tabpic.CurrentX = j - 0.8
                                Tabpic.Print JD(i, j)
                        Next j
                Next i
                        Tabpic.FontBold = True
For i = 1 To nJob
         Tabpic.CurrentX = -0.3
         Tabpic.CurrentY = (nJob - i + 0.75) - 0.1
         Tabpic.Print " " & i
Next i
        Tabpic.CurrentY = nJob + 0.9
        Tabpic.CurrentX = (nMachine / 2) - 3
        Tabpic.Print "Starting Sequence of Jobs on Machines:"
        Tabpic.CurrentY = nJob
For i = 1 To nMachine
        Tabpic.CurrentY = nJob + 0.4
        Tabpic.CurrentX = i - 0.8
```

```
            Tabpic.Print "M" & i
Next i
        Tabpic.FontBold = False
Exit Sub
ShErr:
         MsgBox "Error!" & Err.Description
End Sub
_____________________________
Public Sub Startpnt ( )
Do
        StartPoint
        Schedul
Loop Until nebvalid = True
        ReDim JJ(0 To nJob, 0 To nMachine) As Integer
For i = 1 To nJob
  For j = 1 To nMachine
        JJ(i, j) = JD(i, j)
  Next j
Next i
End Sub
_____________________________
Public Sub drwTblSchdl( )
        On Error GoTo ShErr:
        Dim i As Integer, j As Integer
        Tabpic.Scale (-1, (nJob + 1))-((nMachine + 1), -1)
        Tabpic.Cls
        Tabpic.Line (0, 0)-(0, nJob)
        Tabpic.Line (0, 0)-(nMachine, 0)
        FillColor = 2
For i = 1 To nJob
        Tabpic.Line (0, i)-(nMachine, i)
Next i
For i = 1 To nJob
        Tabpic.Line (i, 0)-(i, nJob)
Next i
For i = 1 To nJob
         For j = 1 To nMachine
                  Tabpic.CurrentY = (nJob - i + 0.75) - 0.1
                 Tabpic.CurrentX = j - 0.8
                  Tabpic.Print BCT(i, j)
         Next j
Next i
        Tabpic.FontBold = True
For i = 1 To nJob
        Tabpic.CurrentX = -0.3
        Tabpic.CurrentY = (nJob - i + 0.75) - 0.1
        Tabpic.Print "J" & i
Next i
        Tabpic.CurrentY = nJob + 0.8
        Tabpic.CurrentX = (nMachine / 2) - 3
        Tabpic.Print "Completion Times of Jobs on Machines:"
```

```
        Tabpic.CurrentY = nJob
For i = 1 To nMachine
        Tabpic.CurrentY = nJob + 0.4
        Tabpic.CurrentX = i - 0.8
        Tabpic.Print "M" & i
Next i
        Tabpic.FontBold = False
Exit Sub
ShErr:
        MsgBox "Error!" & Err.Description
End Sub
```

```
Private Sub Write_Click( )
        Writedata DFname4.Text
End Sub
```

```
Private Sub rescdulcmd_Click( )
For i = 1 To nJob
        For j = 1 To nMachine
                JD(i, j) = JJ(i, j)
        Next j
Next i
        drawTable
End Sub
```

```
Private Sub reDrawCmd_Click( )
End Sub

Private Sub Schdcmd_Click( )
        Schdcmd.Enabled = False
        caLabel.Caption = "Calculating.."
        caLabel.Refresh
        Schedul
        BestSchdul
        Dim mxMov As Integer
        mxMov = MaxMov.Text
        SimulationSearch mxMov
        drwTblSchdl
        ComTicmd.Caption = CoTmin
        Moveno.Caption = Moveno1
        caLabel.Caption = ""
        caLabel.Refresh
        Schdcmd.Enabled = True
End Sub
```

Job Shop scheduling by adaptive tabu-SA

Main module

Global variables:

```
Public Type Idx
  Row As Integer
  Col As Integer
End Type
Public TempO As Idx
Public taboList( ) As Integer, tabuHits As Long
Public NofTabu As Integer, tabuIdx As Integer, tabuFilled As Boolean
Public s As Integer
Public r As Integer
Public nJob As Integer
Public nMachine As Integer
Public JD( ) As Integer
Public JJ( ) As Integer
Public DT( ) As Integer
Public MD( ) As Integer
Public CT( ) As Integer
Public BCT( ) As Integer
Public CoTime As Integer
Public CoTmin As Integer
Public bMinTime As Long
Public Cursoltn( ) As Integer
Public bMinTime( ) As Currency
Public Temtur As Currency
Public m As Integer
Public nebvalid As Boolean
Public Ra1 As Integer
Public Ra2 As Integer
Public Ra3 As Integer
Public Cnbhd As Integer
 Public Moveno1 As Integer
```

```
Sub read_data1(DFname1 As String)
      On Error GoTo ShErr:
      Open DFname1 For Input As #1
      Dim i As Integer, j As Integer
         Input #1, nJob
      nJob = nJob - 1
         Input #1, nMachine
'Define dynamic array boundaries
      ReDim JJ(0 To nJob, 0 To nMachine) As Integer
      ReDim JD(0 To nJob, 1 To nMachine) As Integer
For i = 0 To nJob
   For j = 1 To nMachine
```

```
        Input #1, JJ(i, j) ' Read data to array
        JD(i, j) = JJ(i, j)
    Next j
Next i
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub
```

```
Sub addJDtoTabu( )
        Dim i As Integer, j As Integer
For i = 0 To nJob
    For j = 1 To nMachine
        taboList(tabuIdx, i, j) = JD(i, j)
    Next j
  Next i
If tabuIdx = NofTabu Then
        tabuIdx = 1
        tabuFilled = True ' changes first time that tabu has been filled
Else
        tabuIdx = tabuIdx + 1
End If
End Sub
```

```
Function isJDinTabu( ) As Boolean
        Dim i As Integer, j As Integer, k As Integer
        Dim nTab As Integer
        Dim FoundInTabu As Boolean, Equal As Boolean
        FoundInTabu = False

If tabuFilled Then nTab = NofTabu Else nTab = tabuIdx
        k = 1
Do While k < nTab + 1 And Not FoundInTabu
    Equal = True
    i = 1
        Do While i < nJob + 1 And Equal
                j = 1
                        Do While j < nMachine + 1 And Equal
                                If taboList(k, i, j) <> JD(i, j) Then
                                Equal = False
                                End If
                                j = j + 1
                        Loop
                i = i + 1
      Loop
   If Equal Then
                FoundInTabu = True
   End If
                k = k + 1
Loop
```

```
        isJDinTabu = FoundInTabu
End Function
_____________________________
Sub read_data2(DFname2 As String)       'Reading the sequence of the jobs
        On Error GoTo ShErr:
        Open DFname2 For Input As #1
        Dim j As Integer
                Input #1, nJob
                Input #1, nMachine
        nMachine = nMachine - 1
'Define dynamic array boundaries
        ReDim MD(1 To nJob, 0 To nMachine) As Integer
For j = 1 To nJob
  For k = 0 To nMachine
                Input #1, MD(j, k) 'Read data to array
  Next k
Next j
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub
_____________________________
Sub read_data3(DFname3 As String)
      On Error GoTo ShErr:
      Open DFname3 For Input As #1
      Dim i As Integer
         Input #1, nJob
         Input #1, nMachine
      ReDim DT(1 To nJob, 1 To nMachine) As Integer
For i = 1 To nJob
        For k = 1 To nMachine
          Input #1, DT(i, MD(i, k)) 'Read data to array
      Next k
Next i
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub
_____________________________
Sub appendData(ns As Integer)
      On Error GoTo ShErr:
      Dim DFname4 As String
      DFname4 = ViewDFrm.DFname4.Text
      Open DFname4 For Append As #1
      Dim i As Integer
         Input #1, nJob
         Input #1, nMachine
      ReDim DT(1 To nJob, 1 To nMachine) As Integer
      L = 1
```

```
For i = 1 To ns
        Write #1, Cursoltn(i), bMinTime(i) ' write data to file
Next i
Close #1
Exit Sub
ShErr:
         MsgBox "Error in appending to the file!"
End Sub
```

```
Public Sub StartPoint( )
      ReDim JD(0 To nJob, 0 To nMachine) As Integer
      Dim valid As Boolean
      Dim Rdm As Integer
      Randomize
For L = 1 To nMachine
         JD(0, L) = 0
Next L

For i = 1 To nJob
       Do
          Rdm = Int((nJob) * Rnd + 1)
          JD(i, 1) = Rdm
          k = 1
          valid = True
         Do
                  If JD(i, 1) = JD(i - k, 1) Then
                  valid = False
                  Else
                  k = k + 1
                  End If
      Loop Until k > i Or valid = False
    Loop Until k > i
Next i
 For i = 1 To nJob
   For L = 2 To nMachine
                JD(i, L) = JD(i, 1)
   Next L
 Next i
End Sub
```

```
Public Sub Schedul( )
      Dim tempComTime As Integer, EndTime As Integer, L As Integer
      Dim succss As Boolean
      Dim schedl As Boolean
      Dim a As Integer
      Dim b As Integer
      Dim e As Integer
      ReDim machinTime(1 To nMachine)
      ReDim jobTime(1 To nJob)
'schduling the jobs on machines and computing the complition time
      ReDim CT(0 To nJob, 0 To nMachine)
```

```
For i = 1 To nJob
        MD(i, 0) = 0
Next i
For i = 1 To nJob
   For L = 1 To nMachine
                CT(JD(i, L), 0) = 1
   Next L
Next i
For L = 1 To nMachine
                JD(0, L) = 0
Next L
For L = 1 To nMachine
                CT(0, L) = 1
Next L
     nebvalid = True
     e = 0
     t = 0
     tempComTime = 0
     EndTime = 0
Do
      t = t + 1
     L = 0
     i = 0
     k = 0
  Do
      L = L + 1
    Do
        i = i + 1
       a = JD(i, L)
      Do
        k = k + 1
        b = MD(a, k)
        If L = b Then
          If CT(JD(i, L), MD(a, k)) = 0 Then
            If CT(JD(i, L), MD(a, k - 1)) > 0 Then
               If jobTime(JD(i, L)) > machinTime(MD(a, k)) Then
                  CT(JD(i, L), MD(a, k)) = jobTime(JD(i, L)) + DT(JD(i, L), MD(a, k))
                  succss = True
                  e = e + 1
                  tempComTime = CT(JD(i, L), MD(a, k))
                  jobTime(JD(i, L)) = CT(JD(i, L), MD(a, k))
                  machinTime(MD(a, k)) = CT(JD(i, L), MD(a, k))
                  If EndTime < tempComTime Then EndTime = tempComTime
               Else
                 CT(JD(i, L), MD(a, k)) = machinTime(MD(a, k)) + DT(JD(i, L), MD(a, k))
                  succss = True
                  e = e + 1
                  tempComTime = CT(JD(i, L), MD(a, k))
                  jobTime(JD(i, L)) = CT(JD(i, L), MD(a, k))
                  machinTime(MD(a, k)) = CT(JD(i, L), MD(a, k))
                  If EndTime < tempComTime Then EndTime = tempComTime
```

```
                End If
              Else
              succss = True
              End If
            Else
            schedl = True
            End If
          End If
        Loop Until k = nMachine Or succss = True Or schedl = True
        schedl = False
        k = 0
      Loop Until i = nJob or succss
      succss = False
      i = 0
    Loop Until L = nMachine
      If t > e Then
     nebvalid = False
    End If
Loop Until e = nJob * nMachine Or nebvalid = False
If nebvalid Then CoTime = EndTime
End Sub
```

```
Public Sub selctneibour( )
  Dim Temp As Integer
  Dim inTabu As Boolean
  Dim nofTry As Integer, maxTry As Integer
  Randomize Time
  inTabu = True
  nofTry = 0
  maxTry = 100
Do While inTabu   'look until find  not in tabu
        Randomize Time
         Ra1 = Int(((nJob) * Rnd + 1))
         i = Ra1
         Ra2 = Int(((nMachine) * Rnd + 1))
         L = Ra2
         Temp = JD(i, L)
         Ra3 = Int(((nJob) * Rnd + 1))
         JD(i, L) = JD(Ra3, L)
         JD(Ra3, L) = Temp
         inTabu = isJDinTabu
    If inTabu Then 'If it is in tabu list then return to previous state
                nofTry = nofTry + 1
      If nofTry > maxTry Then
                inTabu = False
      Else
                Temp = JD(i, L)
                JD(i, L) = JD(Ra3, L)
                JD(Ra3, L) = Temp
                tabuHits = tabuHits + 1
```

```
                ViewDFrm.NTabuHits.Caption = tabuHits
      End If
    End If
  Loop
  addJDtoTabu  'add this neighbor to the tabu list
 End Sub
_____
'Return the jobs position after selecting neighbour
Public Sub Resetneibour( )
      Dim Temp As Integer
      Temp = JD(Ra1, Ra2)
      JD(Ra1, Ra2) = JD(Ra3, Ra2)
      JD(Ra3, Ra2) = Temp
End Sub
_____
Public Sub BestSchdul( )
      ReDim BCT(1 To nJob, 1 To nMachine)
      'reslts :(completion time of jobs on machines)
For i = 1 To nJob
   For j = 1 To nMachine
        BCT(i, j) = CT(i, j)
   Next j
Next i
End Sub
_____
Sub nextMove( )
      Dim CoTdf As Integer
      Dim EXx As Double
      Dim F As Currency
      Dim MRdn As Currency
      Dim Success As Boolean
      Cnbhd = CoTime
      neibourNO = 0


Do
  Do
        selctneibour
       Schedul
             If nebvalid = False Then
              Resetneibour
              CoTime = Cnbhd
             End If
  Loop While nebvalid = False
       addJDtoTabu    'add this neighbour to the tabu list
       Temtur = 1 + Log(1 + r)
       CoTdf = CoTime - Cnbhd
       F = CoTdf / Temtur
       EXx = Exp(-F)
             If EXx >= 1 Then
                   Success = True
```

```
                Else

                        MRdn = Rnd
                        If MRdn < EXx Then
                                Success = True
                        Else
                                Resetneibour
                                CoTime = Cnbhd
                        End If
                End If
Loop Until Success
End Sub
```

```
Sub SimulationSearch(maxMove As Integer)
    r = 0
    Dim SearchTime As Double, Start As Double, Finish As Double, MinTime As Double
    Start = Timer
    SearchTime = ViewDFrm.MaxMov.Text
     MinTime = Start
    Dim i As Integer
    Const nofItems As Integer = 1              'Change this number for different sizes
    ReDim Cursoltn(1 To nofItems)
    ReDim bMinTime(1 To nofItems)
  For i = 1 To nofItems
                Cursoltn(i) = 0
                bMinTime(i) = 0
  Next i
        s = 1
    CoTmin = CoTime
    r = 1
    m = 1
Randomize Time
Do While Timer < Start + SearchTime
     nextMove
  If CoTime < Cnbhd Then
       r = 0
  Else
     r = r + 1
   End If
  If CoTime < CoTmin Then
    CoTmin = CoTime
    Cursoltn(s) = CoTime
    bMinTime = Timer - Start
    Moveno1 = bMinTime
     s = s + 1
                If s > nofItems Then
                        appendData nofItems
                        s = 1

                End If
    BestSchdul
```

```
    Moveno1 = m
  End If
        If (m Mod 5) = 0 Then
                ViewDFrm.caLabel.Caption = "Calculating...." & r
                ViewDFrm.caLabel.Refresh
        End If
                m = m + 1
Loop
appendData s - 1 ' write the final part of data
End Sub
```

Form 1 codes: Use the Form 1(*Aform*) *codes* in the Job shop scheduling by adaptive-SA.

Form 2 codes: Substitute the following codes in the Form 2(*ViewDFrm)* at " Job shop scheduling by adaptive-SA".

```
Private Sub Schdcmd_Click( )
        Schdcmd.Enabled = False
        caLabel.Caption = "Calculating.."
        caLabel.Refresh
        selctneibour
        NofTabu = ViewDFrm.tabuLength.Text
        ReDim taboList(1 To NofTabu, 0 To nJob, 1 To nMachine) As Integer
        tabuIdx = 1
        tabuHits = 0
        ViewDFrm.NTabuHits.Caption = tabuHits
        tabuFilled = False
        addJDtoTabu  ' add the start point to the tabu list
        Schedul
        BestSchdul
        Dim mxMov As Integer
        mxMov = MaxMov.Text
        SimulationSearch mxMov
        drwTblSchdl
        ComTicmd.Caption = CoTmin
        Moveno.Caption = Moveno1
        caLabel.Caption = ""
        caLabel.Refresh
        Schdcmd.Enabled = True
End Sub
```

Job Shop scheduling by conventional simulated annealing

Main module: Substitute the following codes in the main module of the Job shop scheduling by adaptive-SA program codes.

```
Sub SimulationSearch(maxMove As Integer)

        Dim TotlCoTime As Long
        Dim MinCoTime As Currency
        Dim DevCoTime As Currency
        Dim TotDevCoTime As Currency
        Dim SDCoTime As Currency
        Dim Delta As Currency
        Dim SearchTime As Double, Start As Double, Finish As Double, MinTime As Double
        Start = Timer
        SearchTime = ViewDFrm.MaxMov.Text
        MinTime = Start
        r = 0

        CoTmin = CoTime
        TotlCoTime = CoTime
        ReDim nComTime(0 To 50000) As Integer

         nComTime(0) = CoTime
         CTemptur = 0.95
        Delta = 0.1
         i = 0
        m = 1
        Randomize Time
Do While Timer < Start + SearchTime
        nextMove
                If m = 1 Then
                        CTemptur = 0.95
                Else
                          i = i + 1
                        nComTime(i) = CoTime
                        DevCoTime = 0
                        TotDevCoTime = 0
                        SDCoTime = 0
                        TotlCoTime = TotlCoTime + CoTime
                        MinCoTime = TotlCoTime / (m + 1)

                For i = 0 To m
                         DevCoTime = ((nComTime(i) - MinCoTime) ^ 2) / (m + 1)
                        TotDevCoTime = TotDevCoTime + DevCoTime
                 Next i
                         SDCoTime = Sqr(TotDevCoTime)
```

```
                CTemptur = CTemptur / (1 + (CTemptur * Log(1 + Delta) / (3 * SDCoTime)))
                 If CTemptur < 0.000001 Then CTemptur = 0.000001

                End If

                        If CoTime < CoTmin Then
                                CoTmin = CoTime
                                BestSchdul

                                Moveno1 = m
                        End If
  If (m Mod 5) = 0 Then
                 ViewDFrm.caLabel.Caption = "Calculating...." & CTemptur
                 ViewDFrm.caLabel.Refresh
  End If
                 m = m + 1
Loop

End Sub
```

Form 1 codes: Use the Form 1 (*Aform*) *codes* in Job shop scheduling by adaptive-SA.

Form 2 codes: Use the Form 2 (*ViewDFrm) codes* in Job shop scheduling by adaptive-SA.

Holding Control Problem by adaptive-SA

Main module

Global variables:

```
Public Bs_num As Integer
Public Sp_num As Integer
Public LoadMax As Integer
Public RndnBus As Integer
Public RndnHoldT As Integer
Public RndnAdoRdu As Currency
Public neibrvalid As Boolean
Public BDprtT( ) As Integer
Public DprtT( ) As Integer
Public IndxlstS( ) As Integer
Public HoriznSize As Integer
Public Ariv_rat( ) As Currency
Public ALit_frcn( ) As Currency
Public Lod_onbus( ) As Integer
Public Lod_onbusls( ) As Integer
Public RunnigT( ) As Integer
Public Pep_lftbehnd( ) As Integer
Public Cnbhd As Integer
Public TotwitT As Integer
Public TotwitTmin As Integer
```

```
Sub read_data2 (DFname2 As String)  'Index of last visited stop
        On Error GoTo ShErr:
        Open DFname2 For Input As #1
        Dim k As Integer
                Input #1, Bs_num
        ReDim IndxlstS(1 To Bs_num) As Integer
For i = 1 To Bs_num
                Input #1, IndxlstS(i) ' Read data to array
  Next i
Close #1
Exit Sub
ShErr:
          MsgBox "Error in reading from file \n Check file name and path."
End Sub
```

```
Sub read_data1 (DFname1 As String)  'departure time of each bus from last visited stop(s)
        On Error GoTo ShErr:
        Open DFname1 For Input As #1
        Dim i As Integer, k As Integer
                Input #1, Bs_num
                Input #1, Sp_num
        ' Define dynamic array boundaries
```

```
        ReDim DprtT(1 To Bs_num, 0 To Sp_num) As Integer
For i = 1 To Bs_num
  For k = 1 To Sp_num
            Input #1, DprtT(i, k) ' Read data to array
  Next k
Next i
Close #1
Exit Sub
ShErr:
         MsgBox "Error in reading from file \n Check file name and path."
End Sub
```

```
Sub read_data3 (DFname3 As String) ' Passenger arrival rate at stops
        On Error GoTo ShErr:
        Open DFname3 For Input As #1
        Dim k As Integer
                Input #1, Sp_num
                ReDim Ariv_rat (1 To Sp_num) As Currency ' Define dynamic array boundaries
For k = 1 To Sp_num
                Input #1, Ariv_rat (k) ' Read data to array
Next k
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub
```

```
Sub read_data4 (DFname4 As String) 'passenger alighting fraction at stops
        On Error GoTo ShErr:
        Open DFname4 For Input As #1
        Dim k As Integer
                Input #1, Sp_num
        ReDim ALit_frcn(1 To Sp_num) As Currency
For k = 1 To Sp_num
                Input #1, ALit_frcn(k) ' Read data to array
Next k
Close #1
Exit Sub
ShErr:
         MsgBox "Error in reading from file \n Check file name and path."
End Sub
```

```
Sub read_data5 (DFname5 As String)  'Load on bus departing last visited stop
        On Error GoTo ShErr:
        Open DFname5 For Input As #1
        Dim i As Integer, k As Integer
                Input #1, Bs_num
                Input #1, Sp_num
        ReDim Lod_onbusls (1 To Bs_num, 1 To Sp_num) As Integer
For i = 1 To Bs_num
  For k = 1 To Sp_num
```

```
                Input #1, Lod_onbusls(i, k) ' Read data to array
  Next k
Next i
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub
_____________________________
Sub read_data6 (DFname6 As String) 'Running time of buses between two stops
        On Error GoTo ShErr:
        Open DFname6 For Input As #1
        Dim k As Integer
                Input #1, Sp_num
        ' Define dynamic array boundaries
        ReDim RunnigT(1 To Sp_num) As Integer
For k = 1 To Sp_num
                Input #1, RunnigT(k) ' Read data to array
Next k
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub
_____________________________
Sub Writedata(DFname4 As String)
        On Error GoTo ShErr:
        Open DFname4 For Output As #1
        Dim i As Integer
        Dim k As Integer
For i = 1 To Bs_num
   For k = 1 To Sp_num
                Write #1, Tab (k + 4); BDprtT(i, k) ' Read data to array
   Next k
Next i
Close #1
Exit Sub
ShErr:
        MsgBox "Error in reading from file \n Check file name and path."
End Sub
_____________________________
Sub Randarive( )                'generate passenger arrival rate at stops
        ReDim Ariv_rat(0 To Sp_num) As Currency
For k = 0 To Sp_num
        Ariv_rat(k) = (2 * Rnd + 0.2)
Next k
End Sub
_____________________________
Sub Aligtfrctn( )               'Generate passenger alighting factions at stops
        ReDim ALit_frcn(1 To Sp_num) As Currency
For k = 1 To Sp_num
```

```
        ALit_frcn(k) = Rnd
Next k
End Sub

---------------------------------
Sub Calcuload( )        'Calculate load on bus i departing stop k
        ReDim Lod_onbus (1 To Bs_num, 1 To Sp_num) As Integer
        ReDim Pep_lftbehnd (1 To Bs_num, 1 To Sp_num) As Integer
        Dim Load As Integer
        Dim s As Integer
        HoriznSize = 5
        Dim i As Integer, k As Integer
        LoadMax = 40
    For i = 1 To Bs_num
      For k = 1 To Sp_num
                Lod_onbus(i, k) = Lod_onbusls(i, k)
      Next k
    Next i
For i = 1 To Bs_num
        s = 0
For k = IndxlstS (i) + 1 To IndxlstS(i) + HoriznSize
   If k <= Sp_num And i > 1 Then
              Load = Ariv_rat(k) * (DprtT(i, k) - DprtT(i - 1, k)) + Pep_lftbehnd (i - 1, k)
           If Load < LoadMax Then
             Lod_onbus(i, k) = Load
         Else
            Lod_onbus(i, k) = LoadMax
         End If
         If (Lod_onbus(i, k) - LoadMax) < 0 Then
           Pep_lftbehnd(i, k) = 0
         Else
           Pep_lftbehnd(i, k) = Lod_onbus(i, k) - LoadMax
         End If
   Else
    If k > Sp_num + s And i = 1 Then
         s = s + 1
         Load = Ariv_rat(s) * (DprtT(i, s) – DprtT (Bs_num, s)) + Pep_lftbehnd (Bs_num, s)
         If Load < LoadMax Then
         Lod_onbus(i, s) = Load
         Else
         Lod_onbus(i, s) = LoadMax
         End If
         If (Lod_onbus(i, s) - LoadMax) < 0 Then
              Pep_lftbehnd(i, s) = 0
         Else
              Pep_lftbehnd(i, s) = Lod_onbus(i, s) - LoadMax
         End If
     Else
          If i = 1 Then
          Load = Ariv_rat(k) * (DprtT(i, k) - DprtT(Bs_num, k)) + Pep_lftbehnd(Bs_num, k)
          If Load < LoadMax Then
          Lod_onbus(i, k) = Load
```

```
            Else
            Lod_onbus(i, k) = LoadMax
            End If
            If (Lod_onbus(i, k) - LoadMax) <= 0 Then
              Pep_lftbehnd(i, k) = 0
            Else
              Pep_lftbehnd(i, k) = Lod_onbus(i, k) - LoadMax
            End If
          Else
              If k > Sp_num + s Then
              s = s + 1
              Load = Ariv_rat(s) * (DprtT(i, s) - DprtT(i - 1, s)) + Pep_lftbehnd(i - 1, s)

              If Load < LoadMax Then
                Lod_onbus(i, s) = Load
              Else
                Lod_onbus(i, s) = LoadMax
              End If
              If (Lod_onbus(i, s) - LoadMax) <= 0 Then
                Pep_lftbehnd(i, s) = 0
              Else
                Pep_lftbehnd(i, s) = Lod_onbus(i, s) - LoadMax
              End If

            End If
          End If

      End If
    End If
  Next k
Next i

End Sub
```

```
Sub CalcuTotlwaiT( ) 'Calculate the total waiting time of passengers
        Dim s As Integer
        Dim i As Integer, k As Integer
        Dim Pep_lftbehndwt As Integer
        Dim Pep_atstpwt As Integer
        Dim SbTotwitT As Integer
For i = 1 To Bs_num
         s = 0
For k = IndxlstS(i) + 1 To IndxlstS(i) + HoriznSize
 If k <= Sp_num And 1 < i And i < Bs_num Then
      Pep_lftbehndwt = Pep_lftbehnd(i, k) * (DprtT(i + 1, k) - DprtT(i, k))
      Pep_atstpwt = (Ariv_rat(k) * (DprtT(i, k) - DprtT(i - 1, k)) ^ 2) / 2
      SbTotwitT = Pep_atstpwt + Pep_lftbehndwt

 Else
      If k > Sp_num Then
          s = s + 1
```

```
            If i = 1 Then
              Pep_lftbehndwt = Pep_lftbehnd(i, s) * (DprtT(i + 1, s) - DprtT(i, s))
              Pep_atstpwt = (Ariv_rat(s) * (DprtT(i, s) - DprtT(Bs_num, s)) ^ 2) / 2
              SbTotwitT = Pep_atstpwt + Pep_lftbehndwt
            End If
            If i = Bs_num Then
              Pep_lftbehndwt = Pep_lftbehnd(i, s) * (DprtT(1, s) - DprtT(i, s))
              Pep_atstpwt = (Ariv_rat(s) * (DprtT(i, s) - DprtT(i - 1, s)) ^ 2) / 2
              SbTotwitT = Pep_atstpwt + Pep_lftbehndwt
            End If
            If 1 < i And i < Bs_num Then
             Pep_lftbehndwt = Pep_lftbehnd(i, s) * (DprtT(i + 1, s) - DprtT(i, s))
             Pep_atstpwt = (Ariv_rat(s) * (DprtT(i, s) - DprtT(i - 1, s)) ^ 2) / 2
             SbTotwitT = Pep_atstpwt + Pep_lftbehndwt
            End If
        Else
          If i = 1 Then
            Pep_lftbehndwt = Pep_lftbehnd(i, k) * (DprtT(i + 1, k) - DprtT(i, k))
            Pep_atstpwt = (Ariv_rat(k) * (DprtT(i, k) - DprtT(Bs_num, k)) ^ 2) / 2
            SbTotwitT = Pep_atstpwt + Pep_lftbehndwt
          Else
            Pep_lftbehndwt = Pep_lftbehnd(i, k) * (DprtT(1, k) - DprtT(i, k))
            Pep_atstpwt = (Ariv_rat(k) * (DprtT(i, k) - DprtT(i - 1, k)) ^ 2) / 2
            SbTotwitT = Pep_atstpwt + Pep_lftbehndwt
          End If
       End If
    End If
            ContrTotwitT = ContrTotwitT + SbTotwitT
  Next k
Next i
          TotwitT = ContrTotwitT
End Sub
```

```
Sub Solutnvalid( )
          ReDim LwerBDprtT(1 To Bs_num, 1 To Sp_num) As Double
          Dim PasgeraligtT As Double
          Dim FPasgeraligtT As Double
          Dim PasgerbordT As Double
          Dim Bsunusdsits As Integer
          Dim Pasgerbord As Double
          Dim Mxalgtbord As Double
          Dim Mxbording As Integer
          neibrvalid = True
          Dim s As Integer
          Dim C0 As Currency
          Dim C1 As Currency
          Dim C2 As Currency
          Dim C3 As Currency
          Dim C4 As Currency
          Dim C5 As Currency
          C0 = 0.16
```

```
        C1 = 0.067
        C2 = 0.05
        C3 = 0.16
        C4 = 0.067
        C5 = 0.067
For i = 1 To Bs_num
        s = 0
  For k = IndxlstS(i) + 1 To IndxlstS(i) + HoriznSize
     If k > Sp_num Then
        s = s + 1
       If Lod_onbus(i, s) < LoadMax Then
         If s = 1 Then
           PasgeraligtT = C1 * ALit_frcn(s) * Lod_onbus(i, Sp_num)
           DprtT(i, 0) = DprtT(i, Sp_num)
         Else
           PasgeraligtT = C1 * ALit_frcn(s) * Lod_onbus(i, s - 1)
         End If
         If i = 1 Then
          PasgerbordT = C2 * (Ariv_rat(s) * (DprtT(i, s) - DprtT(Bs_num, s)) +
            Pep_lftbehnd(Bs_num, s))
         Else
           PasgerbordT = C2 * (Ariv_rat(s) * (DprtT(i, s) - DprtT(i - 1, s)) + Pep_lftbehnd(i - 1, s))
         End If
           If (PasgeraligtT - PasgerbordT) <= 0 Then
              Mxalgtbord = PasgerbordT
           Else
              Mxalgtbord = PasgeraligtT
           End If
           LwerBDprtT(i, s) = DprtT(i, s - 1) + RunnigT(s) + C0 + Mxalgtbord
       Else
           If s = 1 Then
              FPasgeraligtT = C4 * ALit_frcn(s) * 0.3 * Lod_onbus(i, Sp_num)
              DprtT(i, 0) = DprtT(i, Sp_num)
              Bsunusdsits = LoadMax - (1 - ALit_frcn(s)) * Lod_onbus(i, Sp_num)
           Else
              PasgeraligtT = C4 * ALit_frcn(s) * 0.3 * Lod_onbus(i, s - 1)
              Bsunusdsits = LoadMax - (1 - ALit_frcn(s)) * Lod_onbus(i, s - 1)
           End If
           If i = 1 Then
                Pasgerbord = (Ariv_rat(s) * (DprtT(i, s) - DprtT(Bs_num, s)) +
                Pep_lftbehnd(Bs_num, s))
           Else
            Pasgerbord = (Ariv_rat(s) * (DprtT(i, s) - DprtT(i - 1, s)) + Pep_lftbehnd(i - 1, s))
           End If
           If (Bsunusdsits - Pasgerbord) <= 0 Then
              Mxbording = Bsunusdsits
           Else
             Mxbording = Pasgerbord
           End If
             LwerBDprtT(i, s) = DprtT(i, s - 1) + RunnigT(s) + C3 + PasgeraligtT + C5 *
              Mxbording
```

```
        End If
      Else
        If Lod_onbus(i, k) < LoadMax Then
          If k = 1 Then
            PasgeraligtT = C1 * ALit_frcn(k) * Lod_onbus(i, Sp_num)
            DprtT(i, 0) = DprtT(i, Sp_num)
          Else
            PasgeraligtT = C1 * ALit_frcn(k) * Lod_onbus(i, k - 1)
          End If
          If i = 1 Then
            PasgerbordT = C2 * (Ariv_rat(k) * (DprtT(i, k) - DprtT(Bs_num, k)) +
             Pep_lftbehnd(Bs_num, k))
          Else
            PasgerbordT = C2 * (Ariv_rat(k) * (DprtT(i, k) - DprtT(i - 1, k)) + Pep_lftbehnd(i -
            1, k))
          End If

          If (PasgeraligtT - PasgerbordT) <= 0 Then
            Mxalgtbord = PasgerbordT
          Else
            Mxalgtbord = PasgeraligtT
          End If
            LwerBDprtT(i, k) = DprtT(i, k - 1) + RunnigT(k) + C0 + Mxalgtbord
        Else
          If k = 1 Then
            FPasgeraligtT = C4 * ALit_frcn(k) * 0.3 * Lod_onbus(i, Sp_num)
            DprtT(i, 0) = DprtT(i, Sp_num)
            Bsunusdsits = LoadMax - (1 - ALit_frcn(k)) * Lod_onbus(i, Sp_num)
          Else
            PasgeraligtT = C4 * ALit_frcn(k) * 0.3 * Lod_onbus(i, k - 1)
            Bsunusdsits = LoadMax - (1 - ALit_frcn(k)) * Lod_onbus(i, k - 1)
          End If
          If i = 1 Then
            Pasgerbord = (Ariv_rat(k) * (DprtT(i, k) - DprtT(Bs_num, k)) +
            Pep_lftbehnd(Bs_num, k))
          Else
            Pasgerbord = (Ariv_rat(k) * (DprtT(i, k) - DprtT(i - 1, k)) + Pep_lftbehnd(i - 1, k))
          End If
          If (Bsunusdsits - Pasgerbord) <= 0 Then
            Mxbording = Bsunusdsits
          Else
            Mxbording = Pasgerbord
          End If
            LwerBDprtT(i, k) = DprtT(i, k - 1) + RunnigT(k) + C3 + PasgeraligtT + C5 *
            Mxbording
        End If
      End If
    Next k
  Next i
    If IndxlstS(RndnBus) = Sp_num Then IndxlstS(RndnBus) = 0
     For k = IndxlstS(RndnBus) + 1 To IndxlstS(RndnBus) + HoriznSize
```

```
      If DprtT(RndnBus, IndxlstS(RndnBus) + 1) < LwerBDprtT(RndnBus,
         IndxlstS(RndnBus) + 1) Then neibrvalid = False
        If neibrvalid = False Then Exit For
   Next k
End Sub

---------------------------------
Public Sub selctneibor( )
        Dim m As Integer
        Dim i As Integer
        Dim k As Integer
        Randomize Time
        m = 0
        RndnBus = Int(Bs_num * Rnd + 1)
        i = RndnBus
        RndnHoldT = Int(5 * Rnd)
        RndnAdoRdu = Rnd
If RndnHoldT > 0 Then
 If RndnAdoRdu > 0.5 Then
   For k = IndxlstS(i) + 1 To IndxlstS(i) + HoriznSize
     If k <= Sp_num Then
       DprtT(i, k) = DprtT(i, k) + RndnHoldT
     Else
       m = m + 1
       DprtT(i, m) = DprtT(i, m) + RndnHoldT
     End If
   Next k
 Else
   For k = IndxlstS(i) + 1 To IndxlstS(i) + HoriznSize
     If k <= Sp_num Then
       DprtT(i, k) = DprtT(i, k) - RndnHoldT
     Else
       m = m + 1
       DprtT(i, m) = DprtT(i, m) - RndnHoldT
     End If
   Next k
 End If
End If
End Sub

---------------------------------
Public Sub Resetneibor( )
        i = RndnBus
        m = 0
If RndnAdoRdu > 0.5 Then
   For k = IndxlstS(i) + 1 To IndxlstS(i) + HoriznSize
    If k <= Sp_num Then
      DprtT(i, k) = DprtT(i, k) - RndnHoldT
    Else
      m = m + 1
      DprtT(i, m) = DprtT(i, m) - RndnHoldT
    End If
 Next k
```

```
Else
  For k = IndxlstS(i) + 1 To IndxlstS(i) + HoriznSize
    If k <= Sp_num Then
      DprtT(i, k) = DprtT(i, k) + RndnHoldT
    Else
      m = m + 1
      DprtT(i, m) = DprtT(i, m) + RndnHoldT
    End If
  Next k
End If
End Sub
```

```
Sub nextMove ( )
        Dim CoTdf As Integer
        Dim EXx As Double
        Dim F As Currency
        Dim MRdn As Currency
        Dim Success As Boolean
        Cnbhd = TotwitT
        neibourNO = 0
Do
   Do
            selctneibor
           Calcuload
           Solutnvalid
        If neibrvalid = False Then
           Resetneibor
           TotwitT = Cnbhd
        Else
          CalcuTotlwaiT
        End If
    Loop While neibrvalid = False
         Temtur = 1 + Log(1 + r)
        CoTdf = TotwitT - Cnbhd
         F = CoTdf / Temtur
        EXx = Exp(-F)
         If EXx >= 1 Then
                  Success = True
         Else
                 MRdn = Rnd
         If MRdn < EXx Then
                 Success = True
         Else
                  Resetneibor
                 TotwitT = Cnbhd
          End If
  End If
Loop Until Success
End Sub
```

```
Public Sub BestSchdul( )
        ReDim BDprtT(1 To Bs_num, 1 To Sp_num)
        ' reslts :(departure time of buses from stops)
For i = 1 To Bs_num
   For k = 1 To Sp_num
        BDprtT(i, k) = DprtT(i, k)
   Next k
Next i
End Sub
```

```
Sub SimulationSearch (maxMove As Integer)

        Dim SearchTime As Double, Start As Double, Finish As Double, MinTime As Double
        Start = Timer
        SearchTime = Form2.MaxMov.Text
        MinTime = Start
        TotwitTmin = TotwitT
        r = 1
        m = 1
        Randomize Time

Do While Timer < Start + SearchTime
                nextMove
                        If TotwitT < Cnbhd Then
                                r = 0
                        Else
                                r = r + 1
                        End If

                 If  TotwitT < TotwitTmin Then
                          TotwitTmin = TotwitT
                          Moveno1 = Timer - Start
                         BestSchdul
                End If
  If (m Mod 5) = 0 Then
          Form2.caLabel.Caption = "Searching...." & Temtur
         Form2.caLabel.Refresh
  End If
  m = m + 1
Loop
End Sub
```

Form 1 codes:

```
Private Sub ExitCmd_Click ( )
End
End Sub

Private Sub simlte_Click ( )
        Form2.Show
        Form2.drawTable
```

```
End Sub

Private Sub Readdata1Cmd_Click ( )
        read_data1 DFname1.Text
        readdata1cmd.Enabled = True
 End Sub

Private Sub Readdata2Cmd_Click ( )
        read_data2 DFname2.Text
        readdata2cmd.Enabled = True
End Sub

Private Sub Readdata3Cmd_Click ( )
        read_data3 DFname3.Text
        readdata3cmd.Enabled = True
End Sub
Private Sub Readdata4Cmd_Click ( )
        read_data4 DFname4.Text
        readdata4cmd.Enabled = True
End Sub


Private Sub Readdata5Cmd_Click ( )
        read_data5 DFname5.Text
        readdata5cmd.Enabled = True
End Sub

Private Sub Readdata6Cmd_Click( )
        read_data6 DFname6.Text
        readdata6cmd.Enabled = True
End Sub
```

Form 2(viewdatafrm) codes:

```
Public Sub drawTable( )
        On Error GoTo ShErr:
        Dim i As Integer, j As Integer
        Tabpic.Scale (-1, (Bs_num + 1))-((Sp_num + 1), -1)
        Tabpic.Cls
        Tabpic.Line (0, 0)-(0, Bs_num)
        Tabpic.Line (0, 0)-(Sp_num, 0)
        FillColor = 2
For i = 1 To Bs_num
        Tabpic.Line (0, i)-(Sp_num, i)
Next i
For i = 1 To Bs_num + 3
        Tabpic.Line (i, 0)-(i, Bs_num)
Next i
For i = 1 To Bs_num
   For j = 1 To Sp_num
```

```
            Tabpic.CurrentY = (Bs_num - i + 0.75) - 0.1
            Tabpic.CurrentX = j - 0.8
            Tabpic.Print DprtT(i, j)
    Next j
Next i
Tabpic.FontBold = True
For i = 1 To Bs_num
            Tabpic.CurrentX = -0.8
             Tabpic.CurrentY = (Bs_num - i + 0.75) - 0.1
            Tabpic.Print "BS " & i
Next i
            Tabpic.CurrentY = Bs_num + 0.9
            Tabpic.CurrentX = (Sp_num / 2) - 3
            Tabpic.Print "Departure time of buses from last visited stop:"
            Tabpic.CurrentY = Bs_num
For i = 1 To Sp_num
            Tabpic.CurrentY = Bs_num + 0.4
            Tabpic.CurrentX = i - 0.8
            Tabpic.Print "STP" & i
Next i
            Tabpic.FontBold = False
Exit Sub
ShErr:
             MsgBox "Error!" & Err.Description
End Sub
```

```
Public Sub drawTableReslt( )
            On Error GoTo ShErr:
            Dim i As Integer, j As Integer
            Tabpic.Scale (-1, (Bs_num + 1))-((Sp_num + 1), -1)
            Tabpic.Cls
            Tabpic.Line (0, 0)-(0, Bs_num)
            Tabpic.Line (0, 0)-(Sp_num, 0)
            FillColor = 2
For i = 1 To Bs_num
            Tabpic.Line (0, i)-(Sp_num, i)
Next i
For i = 1 To Bs_num + 3
            Tabpic.Line (i, 0)-(i, Bs_num)
Next i
For i = 1 To Bs_num
    For j = 1 To Sp_num
            Tabpic.CurrentY = (Bs_num - i + 0.75) - 0.1
            Tabpic.CurrentX = j - 0.8
            Tabpic.Print BDprtT(i, j)
    Next j
Next i
            Tabpic.FontBold = True
For i = 1 To Bs_num
            Tabpic.CurrentX = -0.8
            Tabpic.CurrentY = (Bs_num - i + 0.75) - 0.1
```

```
        Tabpic.Print "BS " & i
Next i
        Tabpic.CurrentY = Bs_num + 0.9
        Tabpic.CurrentX = (Sp_num / 2) - 3
        Tabpic.Print "Departure time of buses from last visited stop:"
        Tabpic.CurrentY = Bs_num
For i = 1 To Sp_num
        Tabpic.CurrentY = Bs_num + 0.4
        Tabpic.CurrentX = i - 0.8
        Tabpic.Print "STP" & i
Next i
        Tabpic.FontBold = False
Exit Sub
ShErr:
         MsgBox "Error!" & Err.Description
End Sub
_____________________________
Private Sub Write_Click( )
        Writedata DFname4.Text
End Sub
_____________________________
Private Sub ExitCmd_Click( )
End
End Sub
_____________________________
Private Sub Schdcmd_Click( )
        Schdcmd.Enabled = False
        caLabel.Caption = "Searching.."
        caLabel.Refresh
        Calcuload
        CalcuTotlwaiT
        BestSchdul
        Dim mxMov As Integer
        mxMov = MaxMov.Text
        SimulationSearch mxMov
        drawTableReslt
        ComTicmd.Caption = TotwitTmin
        Moveno.Caption = Moveno1
        caLabel.Caption = ""
        caLabel.Refresh
        Schdcmd.Enabled = True

End Sub
_____________________________________
```