DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

APPLICATION OF *Q*-LEARNING ALGORITHM TO BICRITERIA DYNAMIC SCHEDULING PROBLEM

by

Eren YEŞİLYAPRAK

September, 2007

İZMİR

APPLICATION OF *Q*-LEARNING ALGORITHM TO BICRITERIA DYNAMIC SCHEDULING PROBLEM

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

by

Eren YEŞİLYAPRAK

September, 2007
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"APPLICATION OF *Q*-LEARNING ALGORITHM TO BICRITERIA SCHEDULING PROBLEM"** completed by **EREN YEŞİLYAPRAK** under supervision of **ASST. PROF. DR. GÖKALP YILDIZ** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

.....

Supervisor

.....

(Jury Member)

.....

(Jury Member)

Prof.Dr. Cahit HELVACI
Director
Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

I would like to express my thanks to my supervisor Asst. Prof. Dr. Gökalp YILDIZ for his guidance and patience to complete my thesis.

I would also thank to Özlem UZUN ARAZ for her helpful suggestions, encouragement and friendship.

I wish to give special thanks to Leyla DEMİR for her endless support, friendship and special help.

I would like to acknowledge Banu YETKİN EKREN, Emrah Bünyamin EDİS and Rahime SANCAR EDİS for their encouragement and suggestions.

Finally, I would to give the deepest thanks to my family for their love and endless supports. This thesis is dedicated to my little sister, Gözde.

Eren YEŞİLYAPRAK

APPLICATION OF *Q*-LEARNING ALGORITHM TO BICRITERIA DYNAMIC SCHEDULING PROBLEM

ABSTRACT

Most of research in manufacturing scheduling is concerned with the minimization of a single criterion. However, scheduling problems often involve more than one objective and therefore require multi criteria analysis. This thesis deals with bicriteria dynamic scheduling problems. The main purpose of this study is to find out a compromising solution for the system objectives. The *Q*-learning algorithm, an agent based approach, is proposed to find a good schedule for the systems. The proposed methodology consists of three phases. In the first phase, selected dispatching rules are performed on the system under different conditions for both of the objectives. In the second phase, learning potential of the *Q*-agent is investigated. In the third phase, *Q*-learning agent is trained to minimize the system objectives on dynamic scheduling problem. Dispatching rules which are involved in the learning process are determined according to results in the first phase. In this thesis, bicriteria dynamic scheduling problem is investigated on a single machine and a flow shop separately. Furthermore, the performance of the *Q*-learning agent is evaluated and compared to other dispatching rules by using a new ranking method which is also presented in this thesis.

Keywords: Reinforcement learning, *Q*-learning algorithm, bicriteria dynamic scheduling, simulation

Q-ÖĞRENME ALGORİTMASININ İKİ KRİTERLİ DİNAMİK ÇİZELGELEME PROBLEMİNE UYGULANMASI

ÖZ

Üretim çizelgelemeye ilişkin yapılan çalışmalar genel olarak tek bir kriterin eniyilenmesiyle ilgilidir. Fakat çizelgeleme problemleri birden fazla amaç içerebilir ve bu nedenle çok kriterli analiz gerektirebilirler. Bu tez, iki kriterli dinamik çizelgeleme problemlerini konu almaktadır. Bu çalışmanın ana hedefi her iki sistem amacı için iyi olan bir sonuç bulmaktır. Ajan tabanlı bir yaklaşım olan *Q*-öğrenme algoritması sistem için iyi bir çizelge elde etmek için önerilmiştir. Önerilen çözüm yaklaşımı üç aşamadan oluşmaktadır. İlk aşamada, seçilen dağıtım kuralları farklı sistem koşulları altında her iki amaç için test edilmiştir. İkinci aşamada, *Q*-öğrenme algoritmasının öğrenme potansiyeli araştırılmıştır. Son aşamada ise, *Q*-öğrenme algoritması, dinamik çizelgeleme problemindeki sistem amaçlarının eniyilemesi için eğitilmiştir. Öğrenme sürecine dahil edilen dağıtım kuralları, birinci aşamadaki sonuçlara göre belirlenmiştir. Bu çalışmada, iki kriterli dinamik çizelgeleme problemi, tek makinalı üretim sistemi ve akış tipi üretim sistemi için ayrı ayrı incelenmiştir. Ayrıca, bu tezde yeni bir sıralama metodu geliştirilmiştir. *Q*-öğrenme algoritmasının performansı, önerilen sıralama metodu ile değerlendirilmiş ve diğer dağıtım kurallarının performansları ile karşılaştırılmıştır.

Anahtar Kelimeler: Destekli öğrenme, *Q*-öğrenme algoritması, iki kriterli dinamik çizelgeleme, benzetim

CONTENTS

Page

CHAPTER ONE
INTRODUCTION

1.1 Motivation of the Research

Scheduling is a study that concerns the allocation of limited resources to the tasks, which can be in several forms, from service industry to manufacturing systems, or information processes (Pinedo, 1995). Scheduling plays an important role in shop floor planning. A schedule shows the planned time when processing of a specific job will start on each machine that the job requires. It also indicates when the job will be completed on every machine. Thus, scheduling is a timetable for both jobs and machines (Sule, 1997).

As the industrialized work develops, more and more resources are becoming critical. Machine, manpower and facilities are now commonly thought of as critical resources in production and service activities. Scheduling these resources leads to increased efficiency and utilization (Sule, 1997). Manufacturing scheduling problems have been studied for a long time. Specific scheduling problems on different systems such as single machine, flow shop and job shop have attracted researchers. Solution approaches and algorithms were developed to find solutions for these problems. However, in most of these studies, a static environment where the all jobs are available before processing was considered. Dynamic scheduling problems where the arrivals of new jobs are allowed during the processing have been rarely mentioned in the literature.

Finding a good objective to maximize or minimize can be difficult for a scheduling problem for several reasons. First, such important objectives as costumer satisfaction with quality or promptness are difficult to quantify and do not appear among the accounting numbers. Second, a shop is usually dealing with three types of objectives which are to maximize shop throughput, to satisfy customer desires and to minimize current costs (Baker, 1974). These objectives are measured by system performances. Thus, performance measures of the system should be

evaluated simultaneously to find a good schedule under these objectives. In recent years, scheduling problems with multiple performance measures have received considerable attention from researchers. To remain competitive in the market, a decision maker is often concerned with maximizing costumer satisfaction as well as reducing production costs. Therefore, minimizing the inventory cost (measured by mean flow time) and minimizing the mean waiting time of the costumers (measured by mean tardiness) may be required at the same time. Although the two criteria have been studied individually and optimal schedules are easy to determine, it is clear that the optimal schedule for one criterion might be far from optimal for the other in many cases. This is valid even with measures like number of tardy jobs and maximum tardiness, which are both due date oriented. If the total flow time criterion is considered, the situation may become even more complicated.

In this study, a solution approach for bicriteria dynamic scheduling problem is developed. An agent based methodology, Reinforcement Learning, is proposed to find a good schedule for the systems. The proposed approach is applied to a single machine system and a flow shop separately. There are a number of studies related to multi criteria single machine and flow shop scheduling problems when the static system conditions are considered. However, dynamic multi criteria scheduling problems have been rarely studied in the literature. Besides, multi criteria reinforcement learning applications is a new research area and it has not been investigated fully yet. A new ranking methodology is also proposed in the thesis to guide the decision maker to find out the compromising solution.

1.2 Thesis Outline

The main objective of this thesis is to develop a solution approach for bicritera dynamic single machine and flow shop scheduling problems.

In Chapter Two, a detailed machine scheduling description is presented with objectives, assumptions, classification, solution approaches and literature review.

Chapter Three gives information about Reinforcement Learning (*RL*) which is used for developing a solution approach for bicriteria dynamic scheduling problem. This chapter includes the structure of a *RL* model, solution approaches for *RL* problems and a *RL* literature review related to machine scheduling.

Chapter Four presents a solution approach to a bicriteria dynamic single machine scheduling problem. The problem involves two objectives, namely minimizing mean tardiness (*MT*) and minimizing number of tardy jobs (*NTJ*). An agent based solution approach is proposed to find out the compromising solution under different system conditions.

In Chapter Five, proposed solution approach is applied to a bicriteria dynamic flow shop scheduling problem. The problem consists of two different objectives; minimizing mean tardiness (*MT*) and minimizing mean flow time (*MF*). The proposed solution approach is executed to minimize these performance measures under different system conditions.

Finally, Chapter Six summarizes the research work and outlines the directions for future research.

CHAPTER TWO
REVIEW OF SCHEDULING

2.1 Introduction

Scheduling concerns with the allocation of limited resources to tasks over time. It consists of planning and prioritizing activities that need to be performed in an orderly sequence of operations. Scheduling is a tool that optimizes use of available resources. It leads to increased efficiency and capacity utilization, reduced time required to complete jobs and consequently increased the profitability of an organization. Efficient scheduling of resources such as machines, labor and material is a necessity in today's extremely competitive environment (Sule, 1997).

The ultimate objective of the scheduling task is to select a schedule that optimizes some pre-stated goal. In this context, a schedule is formally defined as a set of start and completion times for a group of jobs on a group of machines that satisfy all problem constraints. According to Baker (1974), the three types of decision-making goals that are prevalent in scheduling are:

- Efficient utilization of resources
- Rapid response to demands
- Close conformance to prescribed deadlines

For any enterprise, a planner develops a scheduling function that covers all of the jobs in the firm or different functions are defined for the group of jobs. Scheduling function should have a relationship with several other important functions in the organization. First of all, it is affected by the production planning process, which handles medium and long-term planning for the entire organization. At this point, scheduling process should consider inventory levels, forecasts, and resource requirements to optimize, the product mix and resource allocation for long term period, then it focuses on unexpected events in the shop-floor such as machine breakdowns, or random processing times, and so on. As known, planning is a whole

system and the scheduling is one of the stages of this system. Figure 2.1 shows where scheduling is placed in the planning system as an information flow.

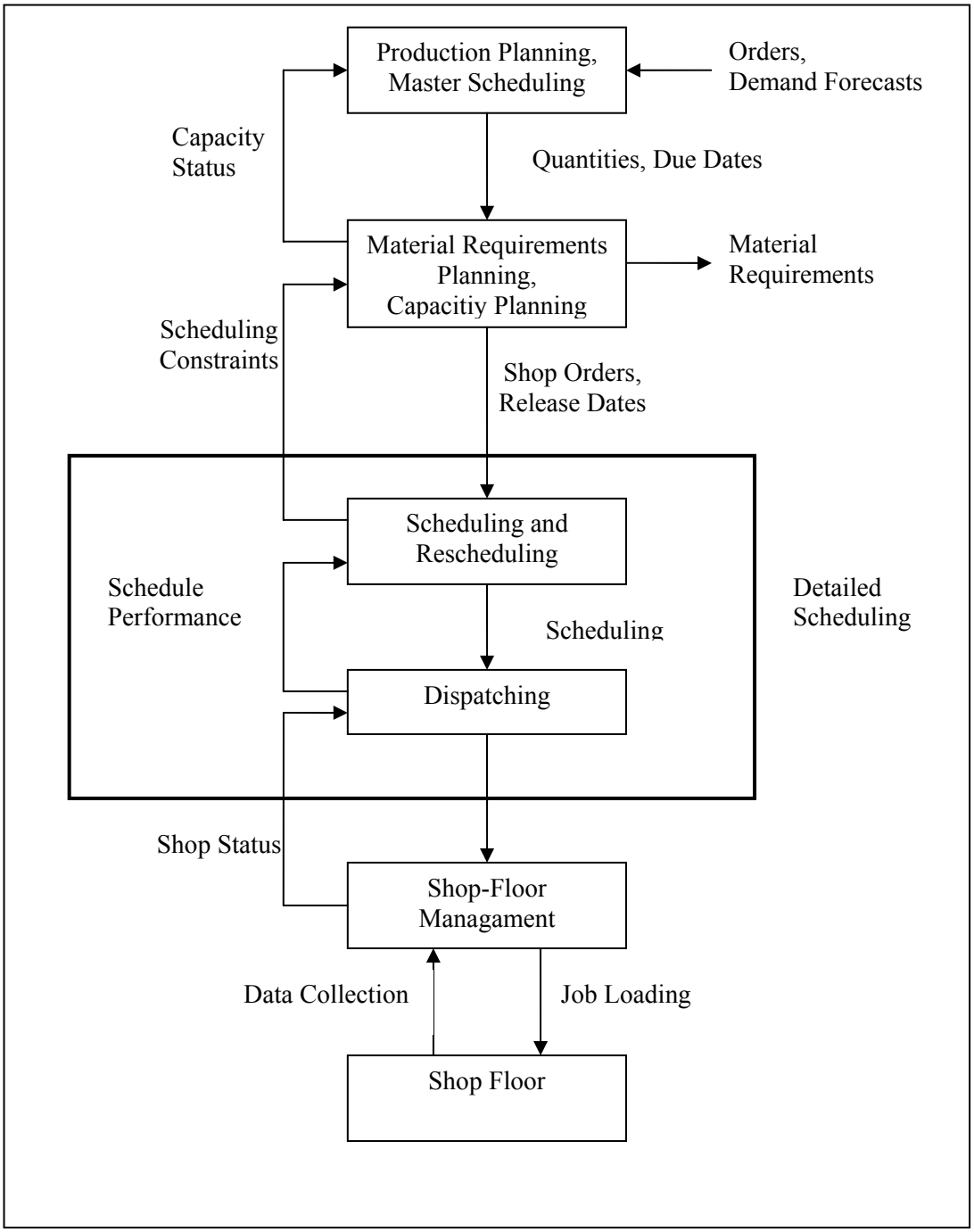

Figure 2.1 Information flow diagram in a manufacturing system (Pinedo and Chao, 1999, p.7)

The scheduling function uses mathematical techniques or heuristic methods to allocate those limited resources to the processing of tasks. A proper allocation of

resources enables the company to optimize its objectives and achieve its goals. Resources may be machines in a job shop, runways at an airport, crews at a construction site, or processing units in a computing environment. Tasks may be operations in a job shop, takeoffs and landings at an airport or stages in a construction project. Each task may have a priority level such as an earliest possible starting time, and a due date. The objectives may also take many forms such as minimizing the time to complete all tasks or minimizing the number of tasks completed after their due dates. (Pinedo& Chao, 1999, p.2)

Although to define a scheduling problem in words is often easy, unfortunately, scheduling can be difficult to perform and implement. Since the time function goes into the scheme, solution branches grow up to a huge amount, at once. Then, implementation difficulties arise related to the modeling of the real world scheduling problems whereas technical difficulties come across the solution methodology and procedures. Resolving these difficulties takes skill and experience but is often financially and operationally well worth the effort. (Pinedo& Chao, 1999, p.5)

2.2 Objectives of Scheduling Problems

There are several objectives investigated in scheduling problems. The classical objective functions are either of the ‘minsum’ or ‘minmax’ type. These functions are built by considering the following elementary functions (Brandimarte and Villa, 1995, p.110).

Flow Time: $F_i = C_i - t_i$

Lateness: $L_j = C_j - d_j$

Tardiness: $T_j = \max \{0, C_j - d_j\}$

Earliness: $E_j = \max \{0, d_j - C_j\}$

Where C is the completion time, d is the due date and t is the arrival time of job j. The most significant minsum objective functions are:

Total completion time: $\sum_{j=1}^{n} C_j$ Total weighted completion time: $\sum_{j=1}^{n} w_j C_j$

Total tardiness: $\sum_{j=1}^{n} T_j$ Total weighted tardiness: $\sum_{j=1}^{n} w_j T_j$

Number of Tardy Jobs: $\sum_{j=1}^{n} U_j$ Weighted number of tardy jobs: $\sum_{j=1}^{n} w_j U_j$

Where n is the number of jobs and U takes the value of 1 if the job is tardy; otherwise it is equal to zero. The weight w_j of job j is a priority factor, denoting the importance of job j relative to other jobs in the system. The weight can be a holding or inventory cost, or it can be the amount of value already added to the job. The most important minmax objective functions are:

Maximum lateness: $L_{max} = max_j\ L_j$

Maximum tardiness: $T_{max} = max_j\ T_j$

Makespan: $C_{max} = max_j\ C_j$

2.3 General Assumptions of Scheduling Problems

To simplify the complexity of manufacturing and service environments, most of the scheduling problems are solved under assumptions associated with job characteristics and objective functions. The following list gives a set of these assumptions and points out when some of them are inadequate to represent a realistic scheduling problem (Brandimarte and Villa, 1995, pg.104-106):

- Single parts and batches of parts are always treated as a single job.

Although this assumption may be appropriate in many situations, in case of large lots, it may produce poor quality schedules. The classical machine scheduling formulation does not allow starting processing on a latter machine until all the parts in the lot are processed on the first one. However, a better schedule may be obtained by transferring a sub-batch of a part to the preceding machine before the completion of the whole batch; such sub-batches are called transfer batches.

- Job cancellation is not allowed.

All the jobs are to be processed eventually. This is not always true; for instance, it may be useful to draw a distinction between high-priority and low-priority jobs; low priority jobs may be scheduled only if the execution of the high-priority ones leaves enough idle time on the machines.

- Preemption is not allowed

Preemption occurs when an operation is stopped and resumed at a later time. This is clearly unacceptable from the technological point of view in most manufacturing environments. However, this is not the case, when scheduling tasks on computer systems.

- Each job visits all machines exactly once.

In practical problems, some machines may not be required for a certain job. In other cases, a machine may be visited more than once by the same job, *e.g.*, reworking of defective parts.

- Machines are always available

In practice, machines may not be available because of controllable events (*e.g.*, preventive maintenance) and uncontrollable events (*e.g.*, failures).

- Jobs are all known in advance

This is the characteristic that distinguishes static and dynamic scheduling problems. In a static scheduling environment, all jobs are ready for processing at time zero.

- The problem is purely deterministic.

In practice, scheduling problems are stochastic in nature. Machine failures, unpredictable processing times, (*e.g.*, in the case of manual operations) causes this uncertainty.

- Processing times are independent of the schedule.

Essentially, this assumption implies that setup times are sequence-independent and can be included in the processing time. This is not the case when sequence dependent setup times are to be dealt with. A classical case occurs in paint production, where sequencing a lot of white paint after the production of black paint is a very bad idea, due to the large setup time needed to clean the machine.

- Machines are the only resources modeled.

In practice, it may be necessary to model additional resources such as transportation devices (*e.g.*, automated guided vehicles), tools, or skilled labor.

- Work in process is allowed.

Jobs may wait in a queue until next machine required for processing is idle.

- Machines are able to process one job at a time.

This is a correct assumption for many mechanical operations. Batch processors such as thermal treatment machine, PCB processors, etc violates the assumption.

2.4 Classification of the Scheduling Problems

Scheduling problems are classified into different categories in the literature. There is a huge variety of scheduling problems related to different manufacturing systems under different system conditions. Scheduling problems may be classified into three fields; these are static vs. dynamic, deterministic vs. stochastic and the type of manufacturing system.

2.4.1 Static Scheduling vs. Dynamic Scheduling

In static scheduling problem, a fixed set of jobs which are ready to be processed at the beginning of the scheduling period are scheduled. The typical assumptions of the static scheduling are the concurrent arrival of the entire set of jobs and availability of the resources. Static scheduling problems are investigated by using either deterministic or stochastic processing times. Methods based on the deterministic processing times can be divided into subtypes of those producing optimum schedules and those utilizing heuristic scheduling procedures. As the computational difficulty increases exponentially with the problem size, optimization methods are used mostly for small problems. For more complex problems, heuristic procedures such as dispatching or sequencing rules are used.

In dynamic scheduling problem, arrivals of the new jobs are allowed during the production. Dynamic scheduling is based on the prioritization between jobs. The

rules that are used in dynamic scheduling may differ from the rules used in static scheduling in terms of the properties which are considered for scheduling process.

2.4.2 Deterministic Problems vs. Stochastic Problems

Production environments in real life are subject to many sources of uncertainty. Among the sources with major impact are machine breakdowns and unexpected releases of high priority jobs. Another source of uncertainty is processing times, which are not known in advance typically (Pinedo, 1995). While the stochastic models are constructed under these conditions, the uncertainties are not considered in deterministic models.

2.4.3 The Type of Manufacturing System

Pinedo (1995) classified the scheduling problems considering machine environment. Basic features of these systems are presented below.

- **Single Machine**

There is only one machine (server) available and arriving jobs require services from this machine. Jobs are processed by the machine one at a time. Each job has a processing time and due date and may have other characteristics such as priority. Single machine models may also be utilized in decomposition approaches, where scheduling problems in more complicated environments are broken down into a number of smaller, single-machine scheduling problems. The case of a single machine is the special and simplest case of all possible machine environments.

- **Parallel Machines**

Parallel machine models can be explained as special case of flexible flow shops that are used in practice so often. In this type of scheduling, the most common objective functions are minimization of total completion time and minimization of

makespan. Parallel machine problems deal with routing and sequencing of the jobs. For most parallel machine environments, preemption is allowed. Preemption means that any product that has not finished yet can be disposed at any time (Biskup, 1999). Pinedo (1995) distinguished the parallel machines into three sub-fields.

In the case of identical parallel machines**,** there are *m* identical parallel machines. Job *i* requires a single operation and may be processed on any one of the *m* machines or on any one belonging to a given subset.

In the case of parallel machines with different speed*s***,** there are *m* parallel machines with different speeds; the speed of machine *j* is denoted by v_j and determined as proportional (relative) to speeds of other machines. The processing time for a job assigned to one machine, is the job processing time divided by machine speed, $p_{ij} = p_i / v_j$. This environment is also referred to as *uniform* machines. If all machines have the same speed, that is, $v_j = 1$ for all *j* and $p_{ij} = p_i$, then the environment is identical to the previous one.

In the case of unrelated parallel machines, there are *m* different machines in parallel, and there is no particular relationship among processing times for each job. Machine *j* can process job *i* at speed v_{ij}. The time p_{ij}, assuming it is processed only on machine *j*, is equal to p_i / v_{ij}. The processing times for the jobs assigned to one machine are not proportional to the processing times that correspond to another.

- **Flow Shop**

There are *m* machines in series. Each job has to be processed on each one of the *m* machines. If the routes of all jobs are identical, that is, all jobs visit the same machines in the same order, the environment is referred to as a flow shop. The machines are set up in series. A job completes its processing on one machine, and then joins the queue at the next. A generalization of the flow shop is the flexible flow shop, which consists of a number of stages in series with a number of machines in

parallel at each stage. Jobs are processed at each stage on any of the parallel machines.

- **Job Shop**

Job shop scheduling is an important and complex activity in manufacturing. It can be described as a set of jobs composed of sequences of operations that are processed on a set of machines. It is a generalization of a flow shop. (A flow shop is a job shop in which each and every job has the same route.) The simplest job shop models assume that a job may be processed on a particular machine at most once on its route through the system. In others, a job may visit a given machine several times on its route through the system.

2.5 Solution Approaches for Scheduling Problems

Solution approaches can be classified as exact solution approaches and heuristic approaches. Exact solution approaches include integer programming and dynamic programming. On the other side, heuristic methods are dispatching rules and search techniques.

2.5.1 Exact Solution Approaches

2.5.1.1 Integer Programming

An integer program (*IP*) is basically a linear program with the additional requirement that the variables have to be integers. If only a subset of the variables are required to be integers and the remaining ones are allowed to be real, the problem is referred to mixed integer program. Many scheduling problems can be formulated as integer programs. There are several ways for dealing with *IP*. The best known approaches are branch and bound techniques and cutting plane techniques (Pinedo, 1995, p.328)

Branch and bound method is basically a sophisticated way of doing complete enumeration that can be applied to many combinatorial problems. The branching refers to a partitioning of the solution space; each part of the solution space is then considered separately. The bounding refers to the development of lower bounds for parts of solution space. If a lower bound on the objective in one part of the solution space is larger than an integer solution already found in a different part of solution space, the corresponding part of the former solution space can be disregarded. (Pinedo, 1995, p.329)

Cutting plane methods focuses on the linear program (*LP*) relaxation of the *IP*. The techniques aim at generating additional linear constraints that have to be satisfied for variables to be integer. These additional inequalities constrain the feasible set more than the original set of linear inequalities without cutting off integer solutions. Solving the *LP* relaxation of the *IP* with the additional inequalities then yields a different solution. If the solution is integer, the procedure stops because the solution obtained is an optimal solution for original *IP*. If the variables are not integer, more inequalities are generated (Pinedo, 1995, p.329).

2.5.1.2 Dynamic Programming

Dynamic programming is basically a complete enumeration scheme that attempts, with a divide-and-conquer approach, to minimize the amount of computation to be done. The approach solves a series of subproblems until it finds the solution of the original problem. It determines the optimal solution for each subproblem, and its contribution to the objective function. At each iteration, it determines the optimal solution for a subproblem, which is larger than all previously solved subproblems. It finds a solution for the current problem by utilizing all the information obtained before in solutions of all the previous subproblems (Pinedo, 1995, p.333).

2.5.2 Heuristic Approaches

Exact solution approaches can solve small problems in a reasonable time. Most of the scheduling problems are *NP-hard* and it is impossible to solve the larger size problems with these exact methods in a reasonable time. Thus, various heuristic approaches are developed for approximate solution of the large problems.

2.5.2.1 Dispatching rules

Dispatching rule is the most simplified and easy-to-implement scheduling tool for production system. In recent years, many dispatching rules have been developed and proven to be effective for manufacturing systems (Horng, 2006). These rules can be classified in various ways. A distinction can be made between *static* and *dynamic* rules. Static rules are not time dependent. They are just a function of the job and/or machine data. On the other hand, dynamic rules are time dependent. They give priority to the jobs depending on the processing times, arrival times, remaining processing times or due dates. A second way of classifying rules is according to the information they are based upon. A *local* rule uses only information pertaining to either the queue where the job is resident or to the machine where the job is queued. A *global* rule may use information regarding other machines or queues (Pinedo, 1995). The dispatching rules used in this thesis are presented in Chapter Four.

2.5.2.2 Search Techniques

- ***Neighborhood Search***

Neighborhood search is a rather general technique used for scheduling. First, a feasible starting solution is selected using any method. Then, all possible ways of changing schedule are tried slightly and each resulting schedule is evaluated. If no improvement is observed, searching is stopped. Otherwise, the largest improvement is taken and small changes are searched, and so on. It generally arrives at any local optimum (Morton and Pentico, 1993).

- ***Tabu Search***

Tabu search is a neighborhood search with a list of recent search positions. A tabu search procedure can be viewed as a modified form of neighborhood search and operates much like a neighborhood search. Instead of stopping when a local optimum is encountered, tabu search strategy accepts a new seed even if its solution value worse than the previous seed. When the new seed worse than the previous seed, the procedure may cycle indefinitely. To avoid this type of cycling, a move back to the previous seed is designated as tabu. In the other words, a list of tabu moves is kept and at each stage the procedure selects the best solution from the neighborhood that are not on the tabu list (Baker, 1974).

- ***Simulated Annealing***

The algorithm searches a neighborhood that is generated by some mechanism which randomly changes the schedule to another in the same locality. The objective function is tested and in the case of an observed improvement, the new schedule is taken as being the best. Otherwise the schedule is tested according to a specific criterion before the decision to disregard the schedule as the source of an overall improvement is made.

- ***Hill Climbing***

Hill climbing is a standard local search procedure. It selects the first or the best improvement in the neighborhood to replace the seed. If the neighborhood of a solution is small, then it is reasonable and feasible to generate them all and select the best. If the neighborhood is very large, then to even generate them all might take a very long time. In this case it may be best to select the first improvement. One possible variation may be to select the best improvement from a fixed number of sampled schedules in the neighborhood. It is worthwhile noticing that there is no way

to escape the first encountered local optimum since any move in the local neighborhood produces a worse schedule.

- ***Genetic Algorithms***

Genetic algorithms constitute a class of techniques that are more general and abstract than both tabu search and simulated annealing. Genetic algorithms, when applied to scheduling, view sequences and schedules as *individuals*, which are members of a *population*. Each individual is characterized by its *fitness*. The fitness of an individual is measured by the associated value of the objective function. The procedure works iteratively and each iteration is referred to as a *generation*. A generation of individuals consists of surviving individuals of previous generation and new solutions or *children* from the previous generation (Pinedo, 1995). In each generation, the best solutions are allowed to produce new solutions (children) by mixing features of the parents or by mutation. The worst children die off to keep the population stable (Morton and Pentico, 1993).

2.5.3 Novel Approaches

One of the most popular novel approaches is neural networks (*NN*). Principles of neural network theory directly come from biology. Neural networks are supposed to reproduce the same mechanisms as in the brain. *NN* able to solve very complicated problems in real time. They are widely used with excellent results for pattern recognition problems. Neural networks are also capable to learn the underlying mechanisms of time series. *NN* often used for market dynamics in trading applications. Precisely, *NN* are not an optimization tool like *GA*, but a kind of interpolation tool. A neural network is made of nodes, called, by analogy with the brain, neurons, connected by weighted edges, called synapses. Typically, the nodes are divided into different layers. There is an input layer and an output layer, and hidden layers in between. This forms the topology of the neural network. During a training phase, weight over the synapses is setup in order to pass knowledge through the network.

Another solution method for scheduling is an agent based approach called reinforcement learning. It has been investigated in several researches related to artificial intelligence and machine learning. However, application of reinforcement learning in industrial systems has not thoroughly explored yet. In this thesis, a solution approach based on reinforcement learning is proposed to solve a dynamic bicriteria scheduling problem. A detailed definition of reinforcement learning is presented in Chapter Three.

2.6 Literature Review

In this section, scheduling studies in dynamic single machine systems and dynamic flow shops are focused on. Multiobjective static studies are also included to the review since this thesis deals with bicriteria scheduling problems.

2.6.1 Single Machine Scheduling Studies

In the past decades, a considerable amount of research has been done in the field of single machine scheduling. Even though a great deal of time and effort has been devoted to a variety of scheduling scenarios, most of these studies have been investigated in static environment. Therefore, this section is classified depending on the environment and objective.

2.6.1.1 Single Machine Scheduling Studies in Static Environment - Multiple Performance Measures

Koksalan et al. (1998) considered the bicriteria scheduling problem of minimizing flow time and maximum earliness on a single machine. The problem is also known to be *NP-hard*. They developed heuristic procedures for generating all efficient sequences for the cases where machine idle time is either allowed or not allowed. For both cases they also discussed an algorithm that finds the best of the approximately efficient sequences for a given objective function by generating only a

small subset of those sequences. They presented computational results which demonstrate that the heuristic procedures and the algorithms perform very well.

Liaw (1999) addressed the problem of scheduling a given set of independent jobs on a single machine to minimize the sum of weighted earliness and weighted tardiness without considering machine idle time. They developed efficient lower and upper bounds. The lower bound was obtained based on a Lagrangian relaxation that decomposes the problem into two sub problems and an efficient lower bounding procedure for each of the sub problems. The upper bound was obtained using a two-phase heuristic procedure that combines a priority dispatching rule with a local improvement procedure. They proposed a branch-and-bound algorithm incorporating these bounds and some dominance rules. Computational experiments on problems with up to 50 jobs show that both the lower and upper bounds are very tight and the branch-and-bound algorithm performs very well.

Köksalan (1999) considered the problem of sequencing jobs on a single machine while minimizing a nondecreasing function of two criteria. They developed a heuristic procedure that quickly finds a good solution for bicriteria scheduling. The procedure is based on using several arcs in the criterion space that are representative of the possible locations of non dominated solutions. By sampling a small number of points on these arcs, they identified a promising point in the criterion space for each arc. Then, they found an efficient sequence in the neighborhood of each of the promising points and selected the best of these efficient sequences as the heuristic solution. They implemented the procedure for two different bicriteria scheduling problems: (i) minimizing total flow time and maximum tardiness and (ii) minimizing total flow time and maximum earliness. Their results show that the heuristic approach is very robust and yields good solutions.

Köksalan and Karasakal (2000) applied a simulated annealing approach to two bicriteria scheduling problems on a single machine. The first problem is the strongly *NP-hard* problem of minimizing total flow time and maximum earliness. The second one is the *NP-hard* problem of minimizing total flow time and number of tardy jobs.

They experimented on different neighborhood structures as well as other parameters of the simulated annealing approach to improve its performance. Their computational experiments show that the developed approach yields solutions that are very close to lower bounds and hence very close to the optimal solutions of their corresponding problems for the minimization of total flow time and maximum earliness. For the minimization of total flow time and number tardy, their experiments show that the simulated annealing approach yields results that are superior to randomly generated schedules. Köksalan and Keha (2003) considered two bicriteria scheduling problems on a single machine: minimizing flow time and number of tardy jobs, and minimizing flow time and maximum earliness. Both problems are known to be *NP-hard*. For the first problem, they developed a heuristic that produces an approximately efficient solution (*AES*) for each possible value the number of tardy jobs can take over the set of efficient solutions. They developed a genetic algorithm (*GA*) that further improves the *AES'*s. They then adapted the *GA* for the second problem by exploiting its special structure. They presented computational experiments that show that the *GA*s performs well.

Azizoglu, Kondakci and Köksalan (2003) studied the bicriteria scheduling problem of minimizing the maximum earliness and the number of tardy jobs on a single machine. They assumed idle time insertion is not allowed. They first examined the problem of minimizing maximum earliness while keeping the number of tardy jobs to its minimum value. They then proposed a general procedure for generating all efficient schedules for bicriteria problems. They also developed a general procedure to find the efficient schedule that minimizes a composite function of the two criteria by evaluating only a small fraction of the efficient solutions. They adapted the general procedures for the bicriteria problem of minimizing maximum earliness and the number of tardy jobs.

Daya, Duffuaa and Raouf (2003) proposed a hybrid branch and bound algorithm for solving the problem of minimizing mean tardiness for a single machine problem subject to minimum number of tardy jobs. The proposed algorithm uses traditional branch and bound scheme where lower bounds on mean tardiness are calculated

coupled with using the information that the number of tardy jobs is known. Their algorithm also uses an insertion algorithm which determines the optimal mean tardiness once the subset of tardy jobs is specified.

Duenas and Petrovic (2004) presented a new approach to single machine production scheduling problems in the presence of uncertainty and multiple scheduling objectives. The uncertain parameter is considered as job's due dates, modeled by a fuzzy set where membership degrees represent a decision maker's satisfaction grades with respect to the job completion time. Their objectives are to minimize the maximum and the average tardiness of the jobs. They used an aggregation operator to consider fuzzy multiobjectives simultaneously. In order to find a job schedule that maximizes the grade of satisfaction achieved, they applied a hybrid algorithm that combines a multiobjective genetic algorithm with local search. They used the fuzzy multiobjective approach to solve a real-life problem defined through collaboration with a manufacturing pottery company. Their results show that combining a genetic algorithm with local search can yield better results than applying the genetic algorithm on its own.

Hino, Ronconi and Mendes (2005) examined the single-machine scheduling problem involving both earliness and tardiness cost. They measured the performance with the minimization of the sum of earliness and tardiness penalties of the jobs. They proposed a tabu search-based heuristic and a genetic algorithm which exploit specific properties of the optimal solution. They also analyzed hybrid strategies to improve the performance of these methods. Their results indicate that hybrid metaheuristic can yield good results with small computational effort. M'Hallal (2006) also investigated same problem in their study. They solved this *NP-hard* scheduling problem using a hybrid heuristic which combines local search heuristics (dispatching rules, hill climbing and simulated annealing) and an evolutionary algorithm based on genetic algorithms. The heuristic involves low and high, relay and teamwork hybridization. Their computational results reflect the sizeable solution quality improvement induced by hybridization, and assess the impact of each type of hybridization on the efficiency of the hybrid heuristic.

Soroush (2006) studied a static stochastic single machine scheduling problem in which jobs have random processing times with arbitrary distributions, due dates are known with certainty, and fixed individual penalties (or weights) are imposed on both early and tardy jobs. The objective is to find an optimal sequence that minimizes the expected total weighted number of early and tardy jobs. In their research, they developed certain conditions under which the problem is solvable exactly. An efficient heuristic is also introduced to find a candidate for the optimal sequence of the general problem. Their illustrative examples and computational results demonstrate that the heuristic performs well in identifying either optimal sequences or good candidates with low errors. Furthermore, they showed that special cases of the problem studied may be reduced to some classical stochastic single machine scheduling.

Eren and Güner (2006) considered a bicriteria scheduling problem with sequence-dependent setup times on a single machine. The objective function of their problem is minimization of the weighted sum of total completion time and total tardiness. They developed an integer programming model for the problem which belongs to *NP-hard* class. Their results show that the proposed model is effective in solving problems with up to 12 jobs. For solving problems containing large number of jobs, they proposed a special heuristic algorithm. Besides the proposed heuristic algorithm, they used tabu search based heuristic for a number of jobs problems. To improve the performance of tabu search method, they considered the result of the proposed heuristic algorithm as an initial solution of tabu search method. According to their computational results, both heuristic algorithms are effective in finding problem solutions with up to 1000 jobs.

2.6.1.2 Single Machine Scheduling Studies in Dynamic Environment - Single Performance measures

Chand, Traub and Uzsoy (1996) considered the single machine dynamic scheduling problem where the jobs have different arrival times and the objective is to

minimize the sum of completion times. They developed decomposition results for this problem such that a large problem can be solved by combining optimal solutions for several smaller problems. They denoted that the decomposition results can be used with any implicit enumeration method to develop an optimal algorithm. Their computational experiment indicates that the computational efficiency of the currently best available branch and bound algorithm can be improved with the use of these decomposition results. Madureira, Ramos and Silva (2001) proposed two interrelated genetic algorithms for the minimization of the weighted tardiness on a static single machine scheduling problem. Their results show that the proposed algorithms perform well for the cases studied and find good solutions in a short time. They also proposed a scheduling system based on genetic algorithms to solve dynamic version of the problem.

Jang (2002) investigated the problem of minimizing the expected number of tardy jobs on a single machine. They considered that jobs having stochastic processing times and deterministic due dates arrive randomly. The objectives of their research are to evaluate the role of the variance of processing time and to develop a dynamic scheduling policy, which is simple and robust, so that it can be used in rapidly changing and uncertain manufacturing environments. They developed a heuristic based on a myopically optimal solution. They also provided the sufficient conditions for the optimality and the worst case analysis. Their computational study validates its effectiveness by comparison with optimal solutions. Lasso et al. (2004) proposed two approaches to face dynamic tardiness problems in single machine environments. The first approach uses job features, processing time, due dates and weights. The second approach uses conventional heuristics and a hybrid evolutionary algorithm to reorder jobs in the waiting queue. They presented computational results which demonstrate that the heuristic procedures perform very well.

2.6.1.3 Single Machine Scheduling Studies in Dynamic Environment – Multiple Performance Measures

Gupta and Sivakumar (2006) presented a preliminary analysis of the typical scheduling environment in semiconductor manufacturing involving multiple job families, and where more than one objective such as cycle time, machine utilization and the due date accuracy needs to be simultaneously considered. In their study, the *NP-hard* problem of scheduling *N* independent jobs on a single testing machine with due dates and sequence-dependent setup times is addressed, where the multiple objectives are to minimize average cycle time, to minimize average tardiness, and to maximize machine utilization. They generated a Pareto optimal solution, which is not inferior to any other feasible solutions in terms of all objectives, combining the analytically optimal and conjunctive simulated scheduling approach. First, they modeled the machine scheduling problem using the discrete event simulation approach and divided the problem into simulation clock based lot selection sub-problems. Then, they selected a Pareto optimal lot using the compromise programming technique for multiobjective optimization at each decision instant in simulated time. With the help of a broad experimental design, they compared this developed solution is with common heuristic-dispatching rules such as SPT and EDD, which show better results for all the objectives over a wide range of problems. The developed scheduling method shows approximately 16.7% reduction in average cycle time, 25.6% reduction in average tardiness, and 21.6% improvement in machine utilization over the common dispatching rules, *SPT* and *EDD*.

Chen and Sheen (2007) considered a single machine scheduling problem with the objective of minimizing the summation of the weighted earliness and tardiness, subject to the number of tardy jobs. There are *n* jobs with a given common due date and each job has different weights for earliness and tardiness. They proposed a pareto-optimal solution algorithm, by which they can efficiently generate pareto-optimal solutions for any possible number of tardy jobs. They also worked with the benchmark problems to show the superior accuracy and run time of their algorithm. Their results show that pareto-optimal solutions can be found for different numbers

of tardy jobs. The computational analysis also shows that approximately 47.15% of the nodes in the branching tree can be eliminated. They indicated that the pareto-optimal solution may provide a manager with a tool to find the schedule with a specific objective value subject to different numbers of tardy jobs.

Valente (2007) considered the single machine earliness/tardiness scheduling problem with job-independent penalties, and no machine idle time. They proposed several dispatching heuristics and analyzed their performance on a wide range of instances in their study. Their heuristics include simple scheduling rules, as well as a procedure that takes advantage of the strengths of each of those rules. They also considered early/tardy dispatching procedures, and a heuristic method based on existing adjacent precedence conditions. They also proposed a procedure which can be used to improve the schedules generated by the heuristics. Their computational tests show that the best results are given by the early/tardy dispatching rules. Their heuristics are also quite fast, and are capable of quickly solving even very large instances. They recommended the use of the improvement procedure, since it improves the solution quality with little additional computational effort.

In a similar study, Valente and Alves (2007) investigated the single machine scheduling problem with quadratic earliness and tardiness costs, and no machine idle time. They proposed several dispatching heuristics, and analyzed their performance on a wide range of instances. The heuristics include simple and widely used scheduling rules, as well as adaptations of those rules to a quadratic objective function. They also proposed heuristic procedures that specifically address the earliness and the tardiness penalties, as well as the quadratic cost function. They analyzed several improvement procedures. These procedures are applied as an improvement step, once the heuristics have generated a schedule. Their computational experiments show that the best results are provided by the heuristics that explicitly consider both early and tardy costs, and the quadratic objective function. They denoted that it is indeed important to specifically address the quadratic feature of the cost function, instead of simply using procedures originally developed for a linear objective function. They found their heuristics are quite fast,

and are capable of quickly solving even very large instances. They recommended the use of an improvement step, since it usually improves the solution quality with little additional computational effort.

Table 2.1 and 2.2 summarize the literature of the single machine scheduling problems.

Table 2.1 Summary of the scheduling studies on single machine – single performance measure

Authors	Objective	Method	Condition
Chand et al. (1996)	Sum of Completion Times	Heuristics	Dynamic
Madureira et al. (2001)	Total Weighted Tardiness	Genetic Algorithms	Dynamic
Jang (2002)	Number of Tardy Jobs	Heuristics	Dynamic
Lasso et al. (2004)	Mean Tardiness	Dispatching Rules	Dynamic

Table 2.2 Summary of the scheduling studies on single machine – multiple performance measures

Authors	Objectives	Method	Condition
Koksalan et al. (1998)	Mean Flow Time and Maximum Earliness	Heuristic	Static
Liaw (1999)	Weighted Earliness and Weighted Tardiness	Branch and Bound	Static
Köksalan (1999)	Minimizing Total Flow Time and Maximum Tardiness	Heuristic	Static
Karasakal and Köksalan (2000)	Total Flow Time and Maximum Earliness	Simulated Annealing	Static
Köksalan and Keha (2003)	Total Flow Time and Number of Tardy Jobs, Total Flow Time and Maximum Earliness	Genetic Algorithm and Heuristics	Static
Azizoglu et. al (2003)	Maximum Earliness and the Number of Tardy Jobs	Heuristic	Static

Table 2.2 Continue

Authors	Objectives	Method	Condition
Daya et. al (2003)	Mean Tardiness and Number of Tardy Jobs	Branch and Bound	Static
Duenas and Petrovic (2004)	Mean Tardiness and Maximum Tardiness	Genetic Algorithm and Heuristics	Static
Hino et. al (2005)	Mean Earliness and Mean Tardiness	Genetic Algorithm and Tabu Search	Static
Gupta and Sivakumar (2006)	Mean Cycle Time and Mean Tardiness	Heuristic	Dynamic
M'Hallal (2006)	Mean Earliness and Mean Tardiness	Hill Climbing	Static
Soroush (2006)	Total Weighted Number of Early and Tardy Jobs	Heuristic	Static
Eren and Güner (2006)	Weighted Sum of Total Completion Time and Total Tardiness	Integer Programming	Static
Valente (2007)	Mean Earliness and Mean Tardiness	Heuristic	Dynamic
Valente and Alves (2007)	Quadratic Earliness and Earliness Costs	Heuristic	Dynamic
Chen and Sheen (2007)	Weighted Earliness and Tardiness	Heuristic	Dynamic

2.6.2 Flow Shop Scheduling Studies

In an *m*-machine flow shop, arriving jobs have to be processed on *m* machines in the same technological order. Each job has *m* operations and operation *i* of each job is to be performed on machine *i*. Operation times for each job on different machines may be different. This section presents a literature review of flow shop scheduling studies and.

2.6.2.1 Flow Shop Scheduling Studies in Static Environment – Multiple Performance Measures

Neppali, Chen and Gupta (1996) considered the two-stage bicriteria flow shop scheduling problem with the objective of minimizing the total flow time subject to obtaining the optimal makespan. They proposed two Genetic Algorithms (*GA*) based approaches to solve the problem. The effectiveness of the proposed *GA* based approaches is demonstrated by comparing their performances with the only known heuristic for the problem. Their computational experiments show that the proposed *GA* based approaches are effective in solving the problem and recommend that the proposed *GA* based approaches are useful for solving the multi-machine, multi-criteria scheduling problems.

Sivrikaya and Ulusoy (1998) considered a scheduling problem of n jobs on two machines which are assumed to be continuously available. All jobs are available at the beginning of the scheduling period. The objective is the minimization of a weighted combination of the average job flow time and the makespan. They compared three branch and bound approaches which differ mainly in their branching strategies. They performed a computational study to evaluate the three different branch and bound approaches on 30 problems each for n=10, n=14, and n=18; and they solved each one of these problems for five different levels of the weighting factor.

Murata and Ishibuchi (1998) proposed a hybrid algorithm for finding a set of non dominated solutions of a multiobjective optimization problem that aims to minimize makespan and maximum tardiness. In the proposed algorithm, they applied a local search procedure to each solution generated by genetic operations. Their algorithm uses a weighted sum of multiple objectives as a fitness function. They denoted that the fitness function is utilized when a pair of parent solutions is selected for generating a new solution by crossover and mutation operations. They applied a local search procedure to the new solution to maximize its fitness value. One characteristic feature of their algorithm is to randomly specify weight values whenever a pair of

parent solutions is selected. That is, each selection is performed by a different weight vector. Another characteristic feature of their algorithm is not to examine all neighborhood solutions of a current solution in the local search procedure. Only a small number of neighborhood solutions are examined to prevent the local search procedure from spending almost all available computation time in their algorithm. High performance of their algorithm is demonstrated by applying it to multiobjective flow shop scheduling problem.

Sayın and Karabatı (1999) addressed the problem of minimizing makespan and sum of completion times simultaneously in a two-machine flow shop environment. They formulate the problem as a bicriteria scheduling problem, and developed a branch-and-bound procedure that iteratively solves restricted single objective scheduling problems until the set of efficient solutions is completely enumerated. They explored certain properties of the set of efficient solutions.

Allahverdi (2002) considered the flow shop scheduling problem with the objective of minimizing the weighted sum of makespan and mean flow time. In this paper, a comparison of these available heuristics in the literature is conducted. Moreover, they proposed three new heuristic algorithms, which can be utilized for both the two-machine and *m*-machine problems. Their computational experiments indicate that the proposed heuristics are superior to all the existing heuristics in the literature including a genetic algorithm. They also developed two dominance relations; one for the two-machine and the other for the three-machine problem. Their experimental results show that both relations are efficient. In a similar study, Allahverdi (2003) also addressed the *m*-machine flow shop problem with the objective of minimizing a weighted sum of makespan and maximum tardiness. Two types of the problem are considered. The first type is to minimize the objective function subject to the constraint that the maximum tardiness should be less than a given value. The second type is to minimize the objective without the constraint. They proposed a new heuristic and compared it with two existing heuristics. Their computational experiments indicate that the proposed heuristic is much better than the existing ones.

Rajendran and Ziegler (2003) presented heuristics for scheduling jobs in a static flow shop with sequence-dependent setup times of jobs. The objective is to minimize the sum of weighted flow time and weighted tardiness of jobs. They used two heuristic preference relations to construct a good heuristic permutation sequence of jobs. Thereafter, they implemented an improvement scheme, once and twice, on the heuristic sequence to enhance the quality of the solution. An existing heuristic, they used a random search procedure and a greedy local search as benchmark methods for relatively evaluating the proposed heuristics. Their analyses show that the proposed heuristics are computationally faster and more effective in yielding solutions of better quality than the benchmark procedures.

Armentano and Arroyo (2004) proposed a new tabu search algorithm for multiobjective combinatorial problems with the goal of obtaining a good approximation of the Pareto-optimal or efficient solutions. Their algorithm works with several paths of solutions in parallel, each with its own tabu list, and the Pareto dominance concept is used to select solutions from the neighborhoods. They applied the algorithm to the permutation flow shop scheduling problem in order to minimize the criteria of makespan and maximum tardiness. They tested performance of the algorithm against a branch and bound algorithm proposed in the literature for two machines, and compared with a tabu search algorithm and a genetic local search algorithm, both from the literature for more than two machines. Their computational results show that the heuristic yields a better approximation than these algorithms.

Eren and Güner (2006) considered a bicriteria two-machine flow shop scheduling problem with setup times. The objective function of the problem is minimization of the weighted sum of total completion time and total tardiness. They developed an integer programming model for the problem which belongs to *NP-hard* class. They demonstrated that only small size problems with up to 20 jobs can be solved by the proposed integer programming model and heuristic methods are also used to solve large size problems. Their results show that their proposed method is effective for the problem. Eren and Güner (2007) also considered a tricriteria two-machine flow shop scheduling problem with a tardiness criterion. The objective is to minimize a

weighted sum of total completion time, total tardiness and makespan. They proposed an integer programming model for the problem which also belongs to *NP-hard* class. Their analysis shows that the heuristics are quite efficient. Table 2.3 summarizes the literature of static multiobjective studies on flow shop scheduling problems

Table 2.3 Summary of the static scheduling studies on flow shops – multiple performance measures

Authors	**Objectives**	**Method**
Neppali et. al (1996)	Total Flow Time and Makespan	Genetic Algorithm
Sivrikaya and Ulusoy (1998)	Mean Flow Time and Makespan	Branch and Bound
Murata and Ishibuchi (1998)	Maximum Tardiness and Makespan	Heuristic
Sayın and Karabatı (1999)	Total Completion Time and Makespan	Branch and Bound
Ali Allahverdi (2002)	Weighted Sum of Makespan and Mean Flow Time	Heuristic
Ali Allahverdi (2003)	Weighted Sum of Makespan and Maximum Tardiness	Heuristic
Rajendran and Ziegler (2003)	Sum of Weighted Flow Time and Weighted Tardiness of Jobs	Heuristic
Armentano and Arroyo (2004)	Makespan and Maximum Tardiness	Tabu Search
Eren and Güner (2006)	Weighted Sum of Total Completion Time and Total Tardiness	Integer Programming
Eren and Güner (2007)	Weighted Sum of Total Completion Time, Total Tardiness and Makespan	Integer Programming

2.6.2.2 Flow Shop Scheduling Studies in Dynamic Environment – Single Performance Measures

Dispatching rules are used in most of dynamic flow shop researches. While considering dynamic job arrival in flow shops, Sarper and Henry (1996) showed that *MDD* (modified due date) and *SPTL* (shortest processing time local) perform better than others in mean flow time, maximum flow time, and mean tardiness criteria for different utilization rates and due date factors. In dynamic flow shops, Barrett and Kadipasaoglu (1990) evaluated the performance of five dynamic and four static dispatching rules. They found that *SPT* performs well in mean flow time and percentage of tardiness, but worse in mean tardiness. They also found that there is no evidence that the performance differences between dynamic dispatching rules and static dispatching rules are significant. Rajendran and Holthaus (1999) generated new rules by compounding processing time with other information such as work in the next queue and job arrival time. They showed that, based on mean flow time criterion, *PT + WINQ* (processing time plus work in next queue) performs as good as *SPT* rule. Table 2.5 summarizes the best dispatching rules from literature for flow shops under different performance criteria.

2.6.2.3 Flow Shop Scheduling Studies in Dynamic Environment – Multiple Performance Measures

There are limited studies on multiobjective flow shop scheduling in the literature. Su and Chou (2000) addressed the two machine bicriteria dynamic flow shop problem where setup time of a job is separated from its processing time and is sequenced independently. They considered the simultaneous minimization of total flow time and makespan, which is more effective in reducing the total scheduling cost compared to the single objective. They first proposed a frozen-event procedure to transform a dynamic scheduling problem into a static one. To solve the transformed static scheduling problem, an integer programming model is formulated. They provided a heuristic algorithm and developed a decision index as the basis for

the heuristic. Their experimental results show that the proposed heuristic algorithm is effective and efficient.

Table 2.4 Summary of best dispatching rules for dynamic flow shops

Criteria	Best Rule(s)	Authors
Mean Flow Time	*SPT*	Hunsucker and Shah (1994)
	MDL, *SPT*	Sarper and Henry(1996)
	SPT	Barrett and Kadipasaoglu(1990)
	SPT, *PT+WINQ*	Rajendran and Holthaus(1999)
Maximum Flow Time	*SPT*, *FIFO*	Hunsucker and Shah (1994)
	MDD, *SPTL*	Sarper and Henry(1996)
	PT/TIS, *AT-RPT*, *AT*	Rajendran and Holthaus(1999)
	PT+WINQ+AT,*FIFO*	
	(*PT+WINQ*)/*TIS*	
Variance of Flow Time	*PT/TIS*,(*PT+WINQ*)/*TIS*	Rajendran and Holthaus(1999)
Number of Tardy Jobs	*SPT*	Barrett and Kadipasaoglu(1990)
	RR, *SPT*	Rajendran and Holthaus(1999)
Mean Tardiness	*MDD*, *SPTL*	Sarper and Henry(1996)
	CR, *OPCR*	Barrett and Kadipasaoglu(1990)
	RR, *COVERT*	Rajendran and Holthaus(1999)
Maximum Tardiness	*RR*, *PT+WINQ+SL*	Rajendran and Holthaus(1999)
Variance of Tardiness	*RR*, *PT+WINQ+SL*	Rajendran and Holthaus(1999)

2.7 Summary

A scheduling function dynamically makes decisions about matching activities and resources in order to finish jobs and project needing these activities in a reasonable time while simultaneously maximizing throughput and minimizing direct operation costs. A detailed description of machine scheduling was presented in this chapter. Furthermore, studies on single machine and flow shop in the literature were mentioned. Multiobjective researches were also presented since this thesis deals with bicriteria scheduling problems. Reinforcement learning, an agent based approach, is

used as an optimization tool in this study. Thus, description and principles of reinforcement learning are presented in the next chapter. Reinforcement learning applications in scheduling is also given and discussed in the same chapter.

CHAPTER THREE
REINFORCEMENT LEARNING

3.1 Introduction

Reinforcement learning (*RL*) has attracted a lot of researchers in the machine learning and artificial intelligence communities in recent years. In order to understand what reinforcement learning is relative to other forms of learning in artificial intelligence, an important distinction should be noted between supervised and unsupervised learning. Learning is the way of acquiring knowledge in its basic concept. Supervised learning is learning from examples provided by an external supervisor. However, it is not adequate for learning from interaction. In interactive problems it is often impractical to obtain examples of desired behavior that are both correct and representative of all the situations in which the agent has to act (Sutton and Barto, 1999). On the other hand, unsupervised learning is analogous to an explorer independently discovering new concepts. Therefore, *RL* is an unsupervised learning method.

RL is an agent based approach and it deals with the problem of how an autonomous agent can learn to select proper actions in specific states for achieving its goals. In the *RL* framework, a learning agent must be able to perceive information from its environment. The perceived information is used to determine the current state of the environment. The agent then chooses an action to perform based on the perceived state. The action taken may result in a change in the state of the environment. Based on the new state, there is an immediate reinforcement that is used to reward or penalize the selected action. These interactions between the agent and its environment continue until the agent learns a decision-making strategy that maximizes the total reward. *RL* is to learn how to map situations to actions so as to maximize a numerical reward signal (Kaelbling et al., 1996). The learner is not told which actions to take, as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them. Besides, actions may affect not only the immediate reward but also the next situation. These two

characteristics, trial and error search and delayed reward, are the most important distinguishing features of reinforcement learning (Sutton and Barto, 1999).

Sutton and Barto (1999) defined four key elements for dealing with the *RL* problems: a policy, a reward function, a value function and a model of the environment. A policy defines the agent's behavior in a given state. It is a mapping from perceived states of the environment to actions to be taken when in those states. A reward function specifies the overall goal of the agent that guides the agent towards learning to achieve the goal. A reinforcement learning agent's sole objective is to maximize the total reward it receives in the long run. The reward function defines what are the good and bad events for the agent. For example, if an action selected by the policy is followed by low reward, then the policy may be changed to select some other action in that situation in the future. A value function specifies the value of a state or a state-action pair indicating how good it (the state or the state-action pair) is in the long run. The *value* of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state. A model of the environment predicts the next state given the current state and a proposed action.

3.2 Reinforcement Learning Model

In the standard reinforcement-learning model, an agent is connected to its environment via perception and action, as depicted in Figure 3.1 (Kaelbling, Litmann and Moore, 1996). On each step of the interaction, the agent receives an input, *i*, and some indication of the current state, *s*, of the environment. The agent then chooses an action, *a*, to generate as output. The action changes the state of the environment, and the value of this state transition, *T*, is communicated to the agent through a scalar reinforcement signal, *r*. The agent's behavior, *B*, should choose actions that tend to increase the long-run sum of values of the reinforcement signal. The agent can learn to do this over time by systematic trial and error, guided by *RL* algorithms.

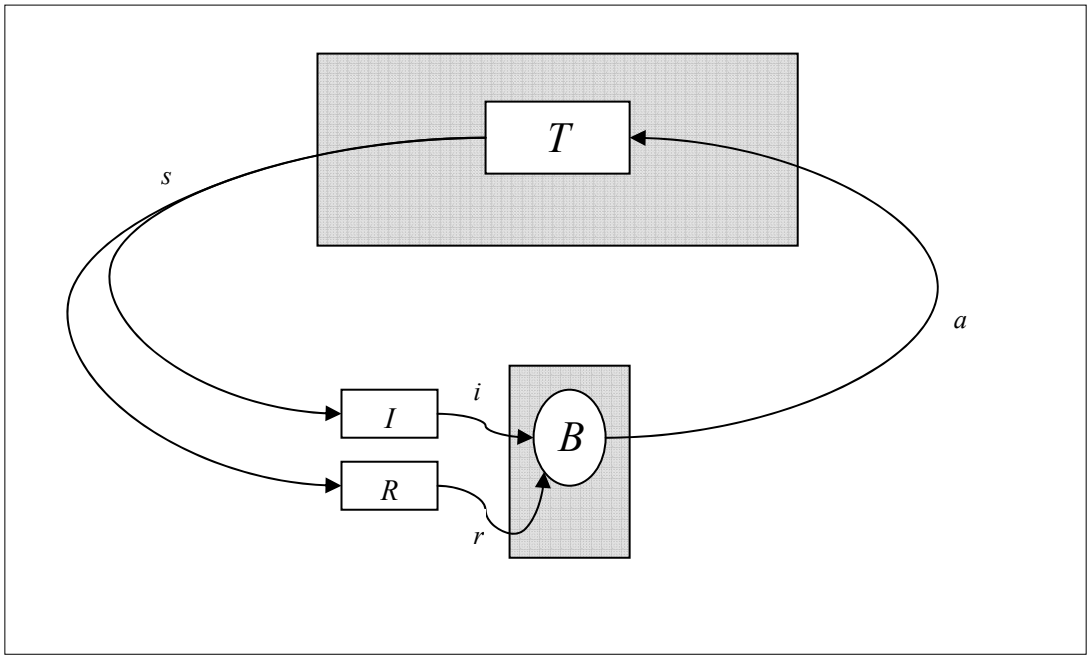

Figure 3.1 Reinforcement agent and its interaction with environment

The model consists of a discrete set of environment states, *S*, a discrete set of agent actions, *A*, and a set of scalar reinforcement signals, *R*. Figure 3.1 also includes an input function *I*. This parameter settles on how the agent views the environment state. Kaelbling et al. (1996) presented following example dialogue to understand the relation between the agent and its environment.

Environment: You are in state 65. You have 4 possible actions.
Agent: I'll take action 2.
Environment: You received a reinforcement of 7 units. You are now in state 15. You have 2 possible actions.
Agent: I'll take action 1.
Environment: You received a reinforcement of -4 units. You are now in state 65. You have 4 possible actions.
Agent: I'll take action 2.
Environment: You received a reinforcement of 5 units. You are now in state 44. You have 5 possible actions.
...
...

3.3 Markov Property

In the reinforcement learning framework, the agent makes its decisions as a function of a signal from the environment called the environment's *state*. If an environment has the Markov property, then its next state and expected next reward can be predicted given the current state and action. Markov Decision Process (*MDP*) consists of:

-a set of states, S
-a set of actions, A
-a state transition function T: $S \times A$

$T(s, a, s')$ represents the probability of making a transition from state s to state s' using action a. The state transition function probabilistically specifies the next state of the environment as a function of its current state and the agent's action. The reward function specifies expected instantaneous reward as a function of the current state and action. The model has Markov property if the state transitions are independent of any previous environment states or agent actions. Thus, interaction between agent and its environment has markov property.

3.4 Delayed Reward

In reinforcement learning, most of the case, the agent's actions determine not only its immediate reward, but also the next state of the environment. The objective is to find an optimal policy. Thus, the agent must take into account the next state as well as the immediate reward when it decides which action to take. In the long run, the agent also determines the future rewards. It may take a long sequence of actions to arrive at a state with high reinforcement after receiving unimportant reinforcement, and then finally arrive at a state with high reinforcement. The agent must be able to learn which of the actions are desirable based on reward that may receive arbitrarily far in the future.

3.5 Solution Methods for Reinforcement Learning Problem

A *RL* problem can be thought as a combination of two subproblems: a problem of estimating unknown values and finding an optimal policy. *RL* methods explore the environment and gathers information about it. In this section, the solution methods for reinforcement learning problem are presented.

3.5.1 Dynamic Programming

Dynamic programming consists of a set of methods for finding optimal policies when the complete information about the environment is available. In a complete model of environment, if a policy is arbitrarily chosen for navigating the environment, the value function of each state can be calculated through an iterative computation. Since this iterative method converges to the true value of the states for given policy, this approach is known as iterative policy evaluation. However the ultimate goal is not to evaluate arbitrary policies, but to obtain an optimal policy. An optimal policy may be found through policy iteration, value iteration or asynchronous dynamic programming algorithms (Sutton and Barto, 1999).

Policy iteration uses two steps. The first step is the evaluation of a policy and second step is the optimization process. First, the policy is fixed and then the value of each state is found. After the values of each state are found for the given policy, the policy is improved by finding a new policy for each state. A sequence of policy evaluation and improvement continues until the policy converges to an optimal policy.

An alternative to policy iteration is value iteration which combines the evaluation and iteration steps into a single iteration that performs both evaluation and improvement simultaneously.

Asynchronous dynamic programming algorithms assign values to states in any order. Ultimately, all states must be backed up for the policy and state values to

converge. However, some states may be backed up several times before others. Asynchronous dynamic programming methods can be faster by avoiding updates of states that are not necessary for optimal behavior. Dynamic programming methods are limited use since they require information about of the system. This situation rarely occurs in real life problems. In order to address real life problems, methods such as Monte Carlo and Temporal Difference were developed which sample from the environment and use those samples to perform the policy evaluation and optimizations.

3.5.2 Monte Carlo Method

Monte Carlo in reinforcement learning refers to an approach to policy value estimation where information about the environment is collected through simulated experience. Monte Carlo methods are defined for systems which have terminal states. In this case, the sampling runs can be divided into episodes such that each episode ends in a terminal state. Value estimates for all states in the system are found by collecting reward information for each state visited following a policy. The equation that updates the value estimates is given as follows.

$$V(s) = V(s) + \alpha(R - V(s)) \tag{3.1}$$

Where $V(s)$ is the value of the state s following a policy, R is the average reward obtained from following policy and α is the learning rate. The advantage of this approach is the $V(s)$ can be estimated without waiting for completion of sampling. However, in large number of runs, some state action pairs may be rarely or never chosen, giving a poor estimate of the value function at those states. One approach to overcome this problem is exploring the new states.

3.5.3 Temporal Difference Learning

Temporal differencing (*TD*), similar to the Monte Carlo approach, uses sample runs to obtain data on the system to form its value function and to perform policy

evaluation and optimization. *TD* differs from Monte Carlo by using bootstrapping to speed up its estimates (Sutton and Barto, 1999). Bootstrapping is an approach which uses old state values to update current state values so that values for the states can be updated immediately after visiting a state instead of at the and of an episode. Essentially, the difference between *TD* and Monte Carlo comes down to how they update their value functions. Equation 3.2 presents the update formula of *TD*.

$$V(s) = V(s) + \alpha\,(R' - \gamma V(s') - V(s)) \tag{3.2}$$

Where $V(s')$ is the value of next state and R' is the reward obtained from new state. The parameter γ represents the discount rate. The advantage of the *TD* approach is that no model of the system is required, but is also uses bootstrapping to perform immediate updates of value function at each visit of state.

An example of *TD* is a method called *Q*-learning algorithm (Watkins, 1989). A detailed description of this algorithm is given in next section.

3.6 *Q*-Learning Algorithm

Reinforcement learning is one of the most important topics in research on intelligent agents. This technique is based on a reinforcement signal which is produced by environment against agent's actions. The *Q* learning algorithm, a popular reinforcement algorithm, was proposed Watkins in 1989. *Q*-learning is an iterative method for action-policy learning in autonomous agents. It is based on the state-action value, *Q(s,a),* which represents the long-term expected reward of the action *a* that is executed in state *s*. These *Q* values converge to the optimal state-action values, *Q** (Watkins, 1989). The steps of the *Q* learning algorithm are given as follows:

Step1: Initialize the *Q(s,a)* value function.

Step 2: Perceive the current state, s_0.

Step 3: Select an action (*a*) for the given state s_0.

Step 4: Apply the action a and receive immediate reward r, then perceive the next state s_1.

Step 5: Update the value function as follows:

$$Q(s_0,a) = Q(s_0,a) + \alpha[r + \gamma \, max_b \, Q(s_1, b) \; - Q(s_0,a)] \tag{3.3}$$

Step 6: Let $s_0 = s_1$.

Step 7: Go to step 3 until state s_0 represents terminal state.

Step 8: Repeat steps 2-7 for a number of episodes.

The parameter, α, is the step size parameter which affects the learning rate. The value of this factor can be constant, $0 \leq \alpha \leq 1$, or varied from step to step. The parameter, γ, is the discount-rate parameter. The value of γ is between one and zero. If γ approaches to zero, the agent does not take long term reward into account, it attaches importance to the immediate reward. If γ approaches to one, the effect of long term reward increases. The $Q(s,a)$ values can be initialized arbitrarily. If no actions for any specific states are preferred, then $Q(s,a)$ values in the policy table are initialized with the same value, otherwise, the agent may prefer taking those actions in the beginning by initializing those $Q(s,a)$ values with larger than the other values. Then these actions are selected initially in specific states.

Several studies in literature (Sutton and Barto, 1999) about Q-learning algorithm use these parameters with settings of $\alpha = 0.1$ and $\gamma = 0.9$. In this thesis, same parameter settings are used in all experimental runs.

3.6.1 Exploration and Exploitation

Exploration and exploitation is very important issue for learning cycle. Exploitation means that the agent prefers the actions that were selected before and rewarded. On the other hand, exploration is that the agent must try something that has not been done before to get more reward. The balance of exploration and exploitation may affect the performance of Q-learning algorithm. In general, the selection of actions (Third step of the Q-learning algorithm) is carried out according

to ε-greedy method (Wang and Usher, 2004) which allows exploration of the actions. If value of ε is set to 0.1, 10% of the time, the strategy will be the randomly selection of the actions independent of their $Q(s,a)$ values. In the other 90% of the time, the agent selects the actions with the best $Q(s,a)$ value. Wang and Usher (2004) and Kong and Wu (2005) set the value of ε to 0.1 in their researches. In this thesis, same parameter setting is used for all simulation models.

3.7 Literature Review

Reinforcement learning has received a lot of attention from researchers in recent years. There is a variety of research areas within *RL* such as robotics, industrial process control and multi-agent problems. In this section, a review of reinforcement learning applications in industrial area is presented. Mutiobjective applications are mentioned besides scheduling of manufacturing systems since this thesis also focuses on bicriteria problems.

3.7.1 Reinforcement Learning Applications in Production and Operation Management

Crites and Barto (1996) applied the *Q*-learning to a four elevator, ten-floor system. Each elevator made its own decision independently of the other elevators. There were some constraints placed on the decisions. The system they dealt with had more than 1022 states. Crites and Barto also employed a neural network to represent the action-value function. Their *RL*-based dispatchers outperformed other existing dispatching heuristics on the customer's average waiting time and average squared waiting time. Singh and Bertsekas (1997) used the *TD* algorithm to find dynamic channel allocation policies in cellular telephone systems. Their study showed that *RL* with a linear function approximation is able to find better dynamic channel allocation policies than two other existing policies.

Laurent and Piat (2002) presented a reinforcement learning algorithm based on *Q*-learning algorithm. They used this algorithm to control a two axis micro

manipulator system. The objective is to learn complex behaviors as reaching target positions and avoiding obstacles at the same time. The simulation results show that their algorithm is able to learn simultaneously opposite behaviors. Raicevic (2006) presented a parallel learning method for agents with an actor–critic architecture based on artificial neural networks. The agents have multiple modules, where the modules can learn in parallel to further increase learning speed. Each module solves a sub-problem and receives its own separate reward signal with all modules trained concurrently. They used the method on a grid world navigation task and their results show that parallel learning can significantly reduce learning time.

Valluri and Croson (2005) developed an agent-based model to study the performance of a supplier selection model which displays a complicated reward and punishment profile under incomplete information. They modeled two methods of determining exploration reference points which are auction-style model focusing on probability of success and newsvendor-style model focusing on profitability. Their simulation shows that the tournament structure suffices to reach convergence at high-quality levels whenever the number of suppliers exceeds three, punishment length and number of suppliers are substitutes and shorter punishments improve learning speed of convergence. They denoted that it is strictly better for the buyer to transact with relatively few suppliers rather than through assumptions that explicitly penalize supplier proliferation.

Whiteson and Stone (2004) proposed a new network simulator designed to study the application of machine learning methods from a system-wide perspective. They also presented learning-based methods for addressing the problems of job routing and *CPU* scheduling in the simulated networks. Their experimental results verify that methods using machine learning outperform reasonable heuristic and hand-coded approaches on example networks designed to capture many of the complexities that exist in real systems.

Scheduling of manufacturing systems has been studied for a long time. Different solution methods and methodologies are applied to job shops, flow shops, flexible

manufacturing systems or single machine systems to find a good schedule. There are several studies related to reinforcement learning applications in scheduling manufacturing systems even though it is a new research area. Rabelol, Jones and Yih (1994) proposed a scheme for the scheduling of flexible manufacturing systems which divides the scheduling function into four different steps: candidate rule selection, transient phenomena analysis, multi criteria compromise analysis, and learning. This scheme is based on a hybrid architecture which utilizes neural networks, simulation, genetic algorithms, and induction mechanism. They investigated the candidate rule selection process, which selects a small list of scheduling rules from a larger list of such rules. They developed this rule selector by using the integration of dynamic programming and neural networks. Their system achieves real-time learning using this approach. In addition, since an expert scheduler is not available, it utilizes reinforcement signals from the environment. They denoted that this integration technique has the potential to solve real world sequencing and scheduling problems in real time.

Zhang and Dietterich (1995) applied *RL* to a job shop scheduling problem involving the scheduling of the various tasks that must be performed to install and test the payloads placed in the cargo bay of the NASA space shuttle for each mission. The objective of this problem was to schedule a set of tasks without violating any resource constraints while minimizing the total duration. The scheduling approach Zhang and Dietterich employed was an iterative repair-based scheduling method that started with generating a critical path schedule by ignoring the resource constraints and incrementally repairing the schedule to find a shortest conflict-free schedule. In their system, each state is a complete schedule and each action is a schedule modification. They applied the temporal difference algorithm *TD* algorithm to this scheduling problem. After taking an action to repair the schedule the scheduler receives a negative reward if the new state still contains constraint violations. This reward function essentially forces the scheduler to not only find a conflict-free schedule but to do it in less iteration. The performance of the iterative repair-based procedure with a simulated annealing method was compared with the one using the

TD method. Their results showed that single iteration of the method with *TD* is equivalent to about 1.8 iterations of the method with simulated annealing.

Kim and Lee (1995) focused on the development of a learning-based heuristic for the machine scheduling problem, which automatically captures the search control knowledge or the common features of good schedules while generating a number of schedules. In their study, a learner or a scheduler learns to select the right action in each state of the machine shop, using the reward from a schedule evaluator for executing the action. They also developed genetic reinforcement learning (*GRL*) approach to apply machine scheduling problem. They applied *GRL* to various classes of machine scheduling problems, such as job shop scheduling, flow shop scheduling and open shop scheduling problems. The performance evaluation of *GRL* with a number of different problem instances has shown that the learning-based heuristic is robust and its performance is comparable with the other problem-specific heuristics or search-oriented heuristics in the quality of solutions.

.

Mahadevan et al. (1997a, 1997b) developed a new model free average reward algorithm called *SMART* for continuous time semi-Markov decision processes. They applied the *SMART* algorithm to the problem of optimal preventative maintenance in a production inventory system. In their system, there was a single machine capable of producing multiple types of products with multiple buffers for storing each of the different products. Whenever a job is finished, the machine needs to decide to either undergo maintenance or start another job. Machine maintenance costs and time are less than repair costs and time. In other words, frequent maintenance may be not economical but machine failures resulting from rare maintenance will require more repair costs and time. In their maintenance problem, the state of the system is a 10-dimensional vector of integers that consists of the numbers of five different products manufactured since the last repair or maintenance and the buffer levels of the five products. They compared the maintenance policy learned from *SMART* to two well-known maintenance heuristics. They found that *SMART* is more flexible than the two heuristics in finding proper maintenance schedules as the costs are varied. Mahadevan and Theocharous (1998) applied *SMART* to the problem of optimizing a

3-machine transfer line producing a single product type. The system goal is to minimize the Work-In-Process (*WIP*) inventory and failures. They compared the policy from *SMART* to the kanban heuristic. Their results showed that the policy learned by *SMART* requires fewer items in inventory and results in fewer failures than with the Kanban heuristic.

Paternina and Das (2001) developed a methodology to find a dynamic control policy via intelligent agents. They applied reinforcement learning on a four station serial line to achieve this goal. They compared the control policy attained by the application of a reinforcement learning algorithm with the other existing policies on the basis of total average work in process and average cost of work in process. Paternina and Das also developed a heuristic control policy. This heuristic policy named Behavior-Based Control (BBC), although placed second to the *RL* policy, proved to be more efficient and leaner control policy than most of the existing policies in the literature.

Aydin and Oztemel (2000) proposed an intelligent agent-based scheduling system in which agents are trained by a new *RL* algorithm. They employed *Q*-III to train the resource agents to dynamically select dispatching rules. Their state determination criteria consist of the buffer size of the machine and the mean slack time of the queue. The rewards were generated based on some selection rules obtained from the literature. The thresholds used in the rules for determining the systems status were obtained through trial-and-error procedures. Three dispatching rules: *SPT*, *COVERT*, and *CR*, are available for each resource agent to select for their use. The authors compared the proposed scheduling system trained by their *RL* mechanism to the above three dispatching rules. Their results showed the *RL*-scheduling system outperformed the use of each of the three rules individually in mean tardiness for most of the testing cases

Wang and Usher (2004) applied *Q*-learning algorithm to a dynamic single machine scheduling problem. They investigated the application potential of *Q*-learning to a dispatching rule selection problem on a single machine to determine if it

can be used to enable a single machine agent to learn commonly accepted dispatching rules for three example cases in which the best dispatching rules have been previously defined. Their results confirm that a machine agent with *Q*-learning algorithm is able to learn the best rules for different objectives. Wang and Usher (2007) also used *Q*-learning agent in a job routing problem in job shop environment. They considered a factorial experiment design for studying the settings used to apply *Q*-learning to the job routing problem. Their study not only investigates the effects of this *Q*-learning application but also provides recommendations for factor settings and useful guidelines for future applications of *Q*-learning to agent-based production scheduling. In a similar study, Kong and Wu (2005) proposed an intelligent agent based single machine scheduling system, where the agent is trained by a new improved *Q*-learning algorithm. They trained the agent by a new simulated annealing based *Q*-learning algorithm. The simulation results show that the simulated annealing based *Q*-learning agent is able to learn to select the best dispatching rule for different system objectives. Their results also indicate that the agent could perform well for all criteria, which is impossible when using only one dispatching rule independently

Paternina and Das (2005) developed a multi-agent reinforcement learning approach to obtaining dynamic control policies for stochastic lot scheduling problem. The dynamic policy optimizes a cost function based on the work in process inventory, the backorder penalty costs and the setup costs, while meeting the productivity constraints for the products. They tested their *RL*-based methodology on a stochastic lot-scheduling problem having a state–action space of size 1.8×10^7. The dynamic policies obtained through the *RL*-based approach outperformed various cyclic policies. They implemented the *RL* approach via a multi-agent control architecture where a decision agent was assigned to each of the products. They used a neural network based approach to approximate the reinforcement value function during the implementation of the *RL*-based methodology. Finally, they extracted the dynamic control policy over the large state space from the reinforcement values using a commercially available tree classifier tool.

Zi and Yang (2005) developed a reinforcement learning approach to dynamic job shop scheduling. They proposed composite rules considering both machine selection and job selection. They trained the dynamic system to enhance its learning and adaptive capability via *Q*-learning algorithm. They defined the conception of pressure to describe the system state. Their results show that their methodology can be used as real time scheduling technology. Yu and Ram (2006) proposed a multiagent scheduling system with a good solution quality and robustness. Their multi-agent approach was designed for dynamic job shops with routing flexibility and sequence-dependent setup. Their strategy was accomplished using a computational model which is composed of response threshold, response intention, and machine-centred reinforcement learning. Yu and Ram compared bio-inspired scheduling with an agent-based approach and a dispatching rule-based approach. Their results show that the proposed bio-inspired scheduling model performs better than the other two methods on all eight common scheduling metrics.

Reinforcement learning applications with multiple objectives are rarely presented in the literature. Myung and Bien (2003) considered a cart-pole system. They proposed a new reinforcement learning method for a multiobjective control problem in consideration of its convergence. Their proposed method, in which objective eligibility is considered for handling multirewards, reformulates a multiobjective control problem in a form of a reinforcement learning problem under non-Markov environment. They found their proposed method limited theoretically.

There are also multiobjective reinforcement learning studies in scheduling of manufacturing systems. Hong and Prabhu (2004) considered a family set up flow shop manufacturing system. They proposed an approach that is suitable for Just-In-Time (*JIT*) production for multi-objective scheduling problem in dynamically changing shop floor environment. They integrated their distributed learning and control (*DLC*) approach with part-driven distributed arrival time control (*DATC*) and machine-driven distributed reinforcement learning based control. The system objectives are minimizing mean squared due date deviation and minimizing waiting time in the buffer. They modeled the machine control problem as Semi Markov

Decision Process (*SMDP*) and solved using *Q*-learning. They evaluated the *DLC* algorithms using simulation for two types of manufacturing systems, family scheduling and dynamic batch sizing. Their results show that *DLC* algorithms achieve significant performance improvement over usual dispatching rules in complex real-time shop floor control problems for *JIT* production.

Mariano, Yamanaka and Morales (2006) described an approach that considers two criteria, namely, the minimization of freshwater consumption and the minimization of the infrastructure cost required to build the network. Their optimization model considers water reuse between operations and wastewater treatment as the main mechanisms to reduce freshwater consumption. They used multi-objective distributed *Q*-learning (*MDQL*) which is a heuristic approach based on the exploitation of knowledge acquired during the search process. In order to compare the quality of the results obtained with *MDQL*, they applied the reduced gradient method to solve a weighted combination of the two objective functions used in the model. The proposed approach was tested on three cases includes a single contaminant four unitary operations problem, a four contaminants real-world case with ten unitary operations and the water distribution network operation of Cuernavaca and Mexico. Their results show that the proposed approach can solve highly constrained real-world multi-objective optimization problems

3.8 Summary

Reinforcement learning is a way of teaching agents, decision-makers, near optimal control policies. This is achieved by assigning punishments and rewards for their actions based on the temporal feedback obtained during active interactions of the learning agents with dynamic systems. Reinforcement learning is the problem faced by an agent that learns a state-action mapping or a policy receiving the scalar evaluation of its action, called reinforcement, from its environment. Agent's experience of the environment can be represented by a value function. The concepts of value and value functions are the key features of the reinforcement learning methods. *RL* requires that the agent learns by directly interacting with the system (its

environment) and responding to the receipt of rewards or penalties based on the effect of each action.

Reinforcement learning has become very active research field in machine learning. However, applications of *RL* to manufacturing systems have not been thoroughly explored yet. In the following chapters, the *Q*-learning algorithm is applied to two manufacturing systems, dynamic single machine and dynamic flow shop, to solve the bicriteria scheduling problem.

CHAPTER FOUR
APPLICATION OF *Q*-LEARNING ALGORITHM TO A BICRITERIA DYNAMIC SINGLE MACHINE SCHEDULING PROBLEM

4.1 Introduction

In recent years, single machine scheduling problems with multiple performance measures have received considerable attention from researchers. There are a number of studies related to multiobjective single machine scheduling problem if the static system condition is considered. This chapter presents a solution approach to a dynamic single machine bicriteria scheduling problem. The problem involves two objectives, namely minimizing mean tardiness (*MT*) and minimizing number of tardy jobs (*NTJ*). An agent based solution approach is proposed to minimize these measures of performance under different system conditions. Next section presents a detailed description of the proposed methodology.

4.2 Proposed Methodology

Studies related to multi criteria single machine scheduling are generally static. In some of these studies, a weighted combination of two or more independent criteria is optimized with mathematical modeling or some heuristic algorithms. Some of these researches give priority to one of the objectives to solve this problem. In a static environment, all jobs are available for processing at time zero. However, arrivals of the jobs are allowed during the processing in dynamic systems. In this study, *Q*-learning algorithm, a Reinforcement learning algorithm, is proposed as a solution approach to dynamic single machine bicriteria scheduling problem.

Reinforcement learning (*RL*) receives attention in recent years from researchers. *RL* is an agent based approach and it deals with the problem of how an autonomous agent can learn to select proper actions in specific states for achieving its goals. In several studies, integrated agents into different manufacturing systems are used to schedule these systems and it is observed that they give significant results. Wang and

Usher (2004) applied the *Q*-learning algorithm to a dynamic single machine dispatching rule selection problem. They investigate the application potential of *Q*-learning to a dispatching rule selection problem on a single machine to determine if it can be used to enable a single machine agent to learn commonly accepted dispatching rules for three measures of performance for which the best dispatching rules have been previously defined. They state that the agent has a potential to learn to execute the best dispatching rule for these three criteria. In this study, *Q*-learning agent is trained to minimize mean tardiness and number of tardy jobs on a dynamic single machine scheduling problem.

This chapter mainly consists of three phases. In the first phase, six dispatching rules that are generally used on dynamic single machine scheduling problem in the literature are included in the study. Performances of these dispatching rules are tested individually for mean tardiness and number of tardy jobs under conditions of different utilization levels and due date assignments. Best dispatching rules for these measures of performance are determined according to individually results.

In the second phase, learning potential of the agent to minimize one of the performance measures, *MT* and *NTJ*, is investigated. Three dispatching rules, including the best dispatching rule, are involved in the test for each measure of performance and system condition. Purpose of these tests is to determine if the single machine agent enables to learn to select the best dispatching rule for *MT* and *NTJ* which are presented in the first phase.

In the third phase, *Q*-learning agent is trained to minimize mean tardiness and number of tardy jobs on a dynamic single machine scheduling problem. The best rules for each measure of performance, according to results of first phase, are included in training process of the agent. State of the system may change, and it depends on the load level of the shopfloor or some other factors. The *Q*-learning agent is trained to learn which rule to use in a perceived system state.

Before these applications, system description is mentioned and performance measures and dispatching rules are detailed in following sections.

4.3 Description of the System

In this study, a dynamic single machine production system is considered. System consists of a machine and a single buffer for storing jobs awaiting processing. Following assumptions are considered for the system:

1. Machine can process only one job at a time.
2. There are no interruptions and cancellations between jobs.
3. Resource is continuously available for the process.
4. The job is processed immediately when it arrives to the machine.
5. If the machine is not idle then job is sent to the buffer.
6. Each job consists of only one operation requiring a variant processing time.
7. Selection of the next job from the buffer for processing is conducted based on a predefined dispatching rule.

4.4 Performance Measures

Different types of objectives may be under consideration in scheduling problems. Mean tardiness and number of tardy jobs are determined as measures of performance for this system.

Lateness is the amount of time (positive or negative) by which the completion time of the job exceeds its due date. Tardiness is the lateness of the job if it is positive; if the lateness of the job is negative, the tardiness is zero (Morton and Pentico, 1993). Equations (4.3) and (4.4) give the formulas of these performance measures.

$$L_j = C_j - d_j \tag{4.1}$$

$$T_j = \max\{0, L_j\} \tag{4.2}$$

$$\text{Mean Tardiness} = \frac{\sum_{j=1}^{n} T_j}{n} \tag{4.3}$$

$$\text{Number of tardy jobs} = \sum_{j=1}^{n} U_j \tag{4.4}$$

Where L is the lateness, C is the completion time, d is the due date, T is the tardiness and n is the number of jobs processed. U takes the value of 1 if the job is tardy; otherwise it is equal to zero and j indicates the job index.

4.5 Dispatching Rules

Dispatching rules have been applied consistently to dynamic scheduling problems because of their ease of implementation and low computational requirement. The performance of these rules varies under different conditions. Some dispatching rules are very powerful for finding a good schedule with regard to a specific system objective. Criteria for the dispatching rule selection are system dependent. The shop status and the overall system objective usually should be taken into account to determine these criteria.

Dispatching rules included in this study are investigated several times in dynamic single machine scheduling problems in the literature. Before the dispatching rules considered in this study are presented, the terminology used in this section is introduced in Table 4.1.

Table 4.1 Terminology used for the definitions of dispatching rules

Parameter	Description
τ	Time at which the dispatching decision is made
p_i	Process time for job *i*
d_i	Due date for job *i*
t_i	Time of arrival of job *i*
s_i	Slack time of job *i*
Z_i	Priority value assigned to the job *i* at the time of decision of dispatching

- **Shortest Processing Time (*SPT*)**

This rule is most commonly used rule for dynamic single machine scheduling and is found to be very effective in minimizing mean flow time and also minimizing mean tardiness under highly loaded shopfloor conditions (Haupt, 1989). The job that has the minimum processing time is chosen for loading.

- **Hodgson Algorithm (*HA*)**

Hodgson Algorithm is very effective in minimizing number of tardy jobs when the static condition is considered (Morton and Pentico, 1993). All jobs are ready for processing at time zero under this condition. Wang and Usher (2004) use this algorithm on a dynamic single machine scheduling problem through executing it on all decision points. They get a new schedule for every time and the job that on the first sequence is loaded to the machine. The priority of job *i* is given as follows:

$$Z_i = (s_i \mid s_i \geq p_i) \tag{4.5}$$

Where the slack time, $s_{i,}$ as follows is given by

$$s_i = d_i - \tau - p_i \tag{4.6}$$

The job with the minimum Z_i is chosen for loading.

- **First In First Out (*FIFO*)**

The job that has entered the queue at the earliest is chosen for loading. The *FIFO* rule is an effective rule minimizing the maximum flow time and variance of flow time (Rajendran and Holthaus, 1997).

- **Last In First Out (*LIFO*)**

The job that has entered the queue at the latest is chosen for loading. *LIFO* is effective in minimizing the number of tardy jobs.

- **Earliest Due Date (*EDD*)**

This rule is often used in industries for its simplicity implementation on the shopfloor. This rule performs well with respect to minimizing maximum tardiness and variance of tardiness in the single machine scheduling problem (Baker, 1974). The job with the smallest d_i is chosen for loading.

- **Rule by Raghu and Rajendran (*RR*)**

This rule seeks to minimize both mean flow time and mean tardiness of jobs. The job with the minimum Z_i is chosen for loading. The priority index, Z_i, of job i is given as follows:

$$Z_i = ((s_i * exp(-\mu) * t_i) / p_i) + exp(-\mu) * t_i \quad (4.7)$$

Where μ refers to utilization level of the machine on which the job is to be loaded.

4.6 First Phase: Performance Analysis of Dispatching Rules Individually

In this section, dispatching rules designated in previous section are tested for mean tardiness and number of tardy jobs under different system conditions in simulation models. The simulation model for the single machine system is constructed in ARENA 10.0 software.

4.6.1 Experimental Design for the Simulation Study

A single machine scheduling simulation is carried out to measure the performances of dispatching rules on *MT* and *NTJ*. Four machine utilization levels (*U*) are tested in the experiments; 80%, 85%, 90% and 95%. The time between job arrivals to the system follows an exponential distribution with a mean of 10. The estimated processing times (*EPT*) of jobs are arranged according to utilization levels. The total work-content (*TWK*) is used for due date settings in all experiments (Blackstone et al., 1982). Equation (4.8) gives the formulation of this method:

$$\text{Due date} = \text{Arrival time} + \text{Allowance Factor} * EPT \tag{4.8}$$

Due date tightness is controlled by adjusting the allowance factor. In this study, the allowance factor (k) is drawn from the uniform distribution between 1.2 and 1.8 for jobs with tight due dates (*TDD*) and between 1.7 and 2.3 for jobs with loose due dates (*LDD*). The real processing time (*RPT*) of each job is generated using a normal distribution with a mean of *EPT* and standard deviation of *EPT* / 10.

In this study, each simulation experiment consists of 10 different replications. In each run, the shop is continuously loaded with job orders that are numbered. In order to determine when the system reaches a steady state, shop parameters such as utilization of the machine, mean flow time of the jobs are observed. It has been found that the shop reaches a steady state after arrival of about 5000 job orders. Simulation run length is fixed as 500000 completed job orders for every simulation experiment. Table 4.2 and Table 4.3 give the all parameter settings.

Table 4.2 Parameter settings for loose due dates

	Arrivals	***EPT***	***RPT***	***k***
80% *U*	Expo(10)	Unif(7,9)	Norm(EPT, EPT /10)	Unif(1.7,2.3)
85% *U*		Unif(8,9)		Unif(1.7,2.3)
90% *U*		Unif(8,10)		Unif(1.7,2.3)
95% *U*		Unif(9,10)		Unif(1.7,2.3)

Table 4.3 Parameter settings for tight due dates

	Arrivals	***EPT***	***RPT***	***k***
80% *U*	Expo(10)	Unif(7,9)	Norm(EPT, EPT /10)	Unif(1.2,1.8)
85% *U*		Unif(8,9)		Unif(1.2,1.8)
90% *U*		Unif(8,10)		Unif(1.2,1.8)
95% *U*		Unif(9,10)		Unif(1.2,1.8)

4.6.2 Results and Discussion

In this study, there are four utilization levels and two due date settings. Thus, 12 simulation experiments are implemented for every dispatching rule. The performances of 6 dispatching rules under study are evaluated with respect to mean tardiness and number of tardy job. The results of the simulation experiments for all system conditions are presented in Table 4.4 and Table 4.5.

Table 4.4 Individual simulation results for loose due dates

Dispatching Rule	**80% *U***		**85% *U***		**90% *U***		**95% *U***	
	MT	***NTJ***	***MT***	***NTJ***	***MT***	***NTJ***	***MT***	***NTJ***
SPT	10.81	178490	18.54	204053	32.58	221469	81.00	243350
HA	11.84	143195	19.43	157816	35.44	173523	84.76	189590
FIFO	10.78	284832	18.63	329859	33.83	381032	82.56	437465
LIFO	12.41	148008	20.06	159154	36.33	175569	85.61	195973
EDD	10.71	285560	18.73	330779	33.75	381679	82.54	437909
RR	10.97	281096	18.98	329622	33.66	337768	82.50	437222

Table 4.5 Individual simulation results for tight due dates

Dispatching Rule	80% *U*		85% *U*		90% *U*		95% *U*	
	MT	***NTJ***	***MT***	***NTJ***	***MT***	***NTJ***	***MT***	***NTJ***
SPT	12.66	283767	20.67	310720	35.08	330296	83.83	352356
HA	13.79	246706	21.63	265185	38.07	283605	87.70	302660
FIFO	13.32	349438	21.06	384040	37.44	420660	86.82	459336
LIFO	13.88	255127	21.76	266943	38.27	289362	87.80	308847
EDD	13.27	350707	21.05	385244	37.39	421530	86.81	459867
RR	13.20	346598	21.03	383994	37.26	418283	86.76	459203

In the case of minimizing mean tardiness, the *SPT* rule gives the best results for all utilization levels when the jobs with the tight due dates are considered. For the loose due date settings, the *SPT* rule continues to be the best for this objective except for one case where the shop is loaded with a low utilization level of 80%. The *EDD* and the *FIFO* perform better than the *SPT* rule but the performances of these rules are almost comparable, since *SPT* is not significantly worse than *EDD* and *FIFO*.

With respect to minimizing number of tardy jobs, Hodgson's Algorithm emerges to be the best for utilization levels and due date settings. It is also observed that the performance of the *LIFO* rule is seems to be fairly well.

4.7 Second Phase: Analysis of Learning Potential of the *Q*-learning Agent

In this phase, the application potential of the *Q*-learning algorithm to a dynamic single machine dispatching rule selection problem is investigated. The purpose of this study is to determine if the single machine agent enables to learn to select the well performed dispatching rule for mean tardiness and number of tardy jobs which are previously stated in the first phase.

Wang and Usher (2004) studied the same problem in their investigation. Their results confirmed that a machine agent with *Q*-learning algorithm is able to learn the best rules for different system objectives. In another similar study, Kong and Wu (2005) trained the agent with a simulated-annealing based *Q*-learning algorithm. They observed that the agent selects the best dispatching rule for three different objectives.

In this thesis, investigation of the learning potential of the agent is included in the study since it forms an important base for the third phase of the study.

4.7.1 Minimizing Mean Tardiness

In this section, learning potential of the agent is investigated when the objective is minimizing mean tardiness. It is obvious that the *SPT* dispatching rule performed better than the other dispatching rules for *MT* according to the results of the first phase. Thus, it is expected that the agent learns to select the *SPT* under different system conditions.

Before describing the learning process, set of actions, system states and reward function which are the main parts of *Q*-learning algorithm are determined in following sections.

4.7.1.1 Set of Actions

In this research, selection of the next job for processing from the buffer is conducted based on three dispatching rules that are well performed for mean tardiness in individual simulation experiments in the first phase. The best three dispatching rules under all system conditions, *EDD*, *SPT* and *FIFO*, are determined as the action set of the *Q*-learning agent. The learning agent applies the *Q*-learning algorithm to select a dispatching rule from the action set that contains *EDD*, *SPT* and *FIFO*. The agent perceives the current situation of the system (the system state) from the environment, evaluates it and selects the one of the proper dispatching rule based on the $Q(s,a)$ values. The set of actions, A, is defined as follows:

$A = \{a_1, a_2, a_3\}$

where,

a_1 = execute the *EDD*

a_2 = execute the *SPT*

a_3 = execute the *FIFO*

4.7.1.2 State Determination Criteria

The agent makes a decision based on the current system state to select one of the actions when the machine becomes idle. The table of system states can be constructed respect to several measures such as the number of the jobs in the buffer, the number of the tardy jobs in the buffer or the lateness or average slack time of those jobs.

Wang and Usher (2004) adopted the number of jobs in the buffer and estimation of the total lateness as state determination criteria. Aydın and Öztemel (1999) determined the systems states respect to sizes of the queue and mean slack time of the queue. Kong and Wu (2005) defined different system state constructs according to different system objectives. They designated the average slack time in the queue as the criteria when the objective is minimizing mean tardiness and maximum slack value of the jobs waiting in the buffer when the objective is minimizing maximum tardiness. Zi and Yang (2005) presented a novel state determination criterion in which the state is defined based on the concept of pressure that donated by the ratio of estimated mean lateness and estimated mean remaining processing time.

In our thesis, estimation of the total lateness (*ETL*) is adopted as system determination criteria. The *EPT* value of the jobs is taken into account to evaluate the *ETL*. There are two special conditions that need to be considered when the agent attempts to select an action. The first one is when the buffer is empty and the second one occurs when there is only one job in the buffer. In these conditions, there is no need to select a dispatching rule since there are less than two jobs in the buffer. The conditions for these two special cases are presented in the state table as dummy state. Therefore, only when there are two or more jobs in the buffer, the agent selects an action for the current state.

In this research, 10 *ETL* intervals are evaluated to define state action table. The ranges of the states are determined through trial and error experiments by balancing the number of visits to each state. When the *Q*-learning algorithm in progress, if the

previous state is a dummy state, there is no need to update the $Q(s,a)$ value because the decision of taking next action is made without considering learning algorithm. However, If the previous state is not a dummy state but the current state is a dummy state, updating of $Q(s,a)$ value is executed using Equation 4.10 since $Q(s,a)$ values of the dummy states are equal to zero (Wang and Usher, 2004).

$$Q(s_0,a) = Q(s_0,a) + \alpha r \tag{4.10}$$

In the case of both of the states are not dummy, updating of the Q value is evaluated according to Equation 3.3 that has been previously defined.

In this study, eight different state action tables are defined since there are eight different system conditions. The state action tables for the conditions of tight due dates and 95% utilization level and loose due dates and 85% utilization level are presented in Table 4.6 and Table 4.7. The rest of the state action tables are illustrated in Appendix A.

Table 4.6 State action table for 95% utilization level and tight due dates

State	**Determination Criteria**	***EDD***	***SPT***	***FIFO***
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ETL < 50$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $50 \leq ETL < 100$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $100 \leq ETL < 250$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $250 \leq ETL < 400$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $400 \leq ETL < 600$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $600 \leq ETL < 750$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $750 \leq ETL < 900$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $900 \leq ETL < 1000$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $1000 \leq ETL < 1100$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $1100 \leq ETL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table 4.7 State action table for 85% utilization level and loose due dates

State	Determination Criteria	*EDD*	*SPT*	*FIFO*
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ETL < 10$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $10 \leq ETL < 25$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $25 \leq ETL < 45$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $45 \leq ETL < 80$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $80 \leq ETL < 120$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $120 \leq ETL < 220$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $220 \leq ETL < 330$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $330 \leq ETL < 450$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $450 \leq ETL < 630$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $630 \leq ETL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

4.7.1.3 Determination of Reward Function

In *Q*-learning algorithm, the purpose of the agent is formalized in terms of a specified reward signal passing from environment through agent. These signals guide the learning agent to achieve its goal. After agent selects an action in a state for a specified objective, *Q*-learning mechanism gives an immediate reward or punishment to agent's action according to reward function and agent learns to maximize its reward during the learning cycle.

Wang and Usher (2004) construct their reward function in their single machine scheduling study in case of mean tardiness based on the principle that the later a job is finished, the larger its negative reward; likewise the earlier a job is completed, the larger its reward. In case of minimizing maximum lateness, if lateness of the job is greater than the maximum lateness, *Q*-learning mechanism gives -1, otherwise it gives +1. Kong and Wu (2005) construct the same reward function in case of minimizing mean tardiness.

Since the learning agent tries to maximize the total reward, reward function is used for guiding the learning agent to achieve its goal. In this experiment, the reward function aims at minimizing mean tardiness. Thus, it is constructed based on the tardiness level of the job. If the job is not tardy, learning agent receives a reward of +1. Otherwise, the learning agent receives a negative reward depending on the tardiness level of the job. Ten different tardiness levels are defined for each system

condition. The ranges of those levels are determined through trial and error experiments as in the other studies (Wang and Usher, 2004). The reward function tables for tight due dates and 95% utilization level and loose due dates and 85% utilization level are shown in Table 4.8. The reward function tables for other system conditions are given in Appendix B.

Table 4.8 Reward function tables

Tight Due Dates and 95% Utilization		**Loose Due Dates and 85% Utilization**	
Condition	**Reward**	**Condition**	**Reward**
$Tardiness \leq 0$	1	$Tardiness \leq 0$	1
$0 < Tardiness \leq 30$	-1	$0 < Tardiness \leq 5$	-1
$30 < Tardiness \leq 50$	-2	$5 < Tardiness \leq 10$	-2
$50 < Tardiness \leq 70$	-3	$10 < Tardiness \leq 15$	-3
$70 < Tardiness \leq 90$	-4	$15 < Tardiness \leq 20$	-4
$90 < Tardiness \leq 110$	-5	$20 < Tardiness \leq 25$	-5
$110 < Tardiness \leq 130$	-6	$25 < Tardiness \leq 35$	-6
$130 < Tardiness \leq 160$	-7	$35 < Tardiness \leq 50$	-7
$160 < Tardiness \leq 200$	-8	$50 < Tardiness \leq 60$	-8
$200 < Tardiness$	-9	$60 < Tardiness$	-9

4.7.1.4 The Structure of the Q-learning Agent

The *Q*-learning algorithm is used during the interaction of the agent with the simulated environment. The aim of the *Q*-learning mechanism is to give rewards for the actions by observing agent's behavior. Typical learning model contains four elements which are the environment, the learning agent, a set of actions, and the environmental response (Sutton and Barto, 1999). In a reinforcement model, an agent is connected to its environment. On each step of interaction the agent perceives the current state, and then selects an action for this state. The execution of the action changes the state of the environment. Based on the new state, an immediate reinforcement signal that is used to reward or penalize the selected action is produced by environment.

The interactions between the agent, *Q*-learning mechanism and simulated environment are presented in Figure 4.1. The agent perceives the estimated total lateness of the jobs in the buffer and then selects the most proper dispatching rule to

be performed. *Q*-learning mechanism produces a reward for the agent's decision. The agent takes the reward into account in updating the *Q* values for future actions. These interactions between the agent and the simulated environment continue until the agent learns a decision making strategy that maximizes the total reward.

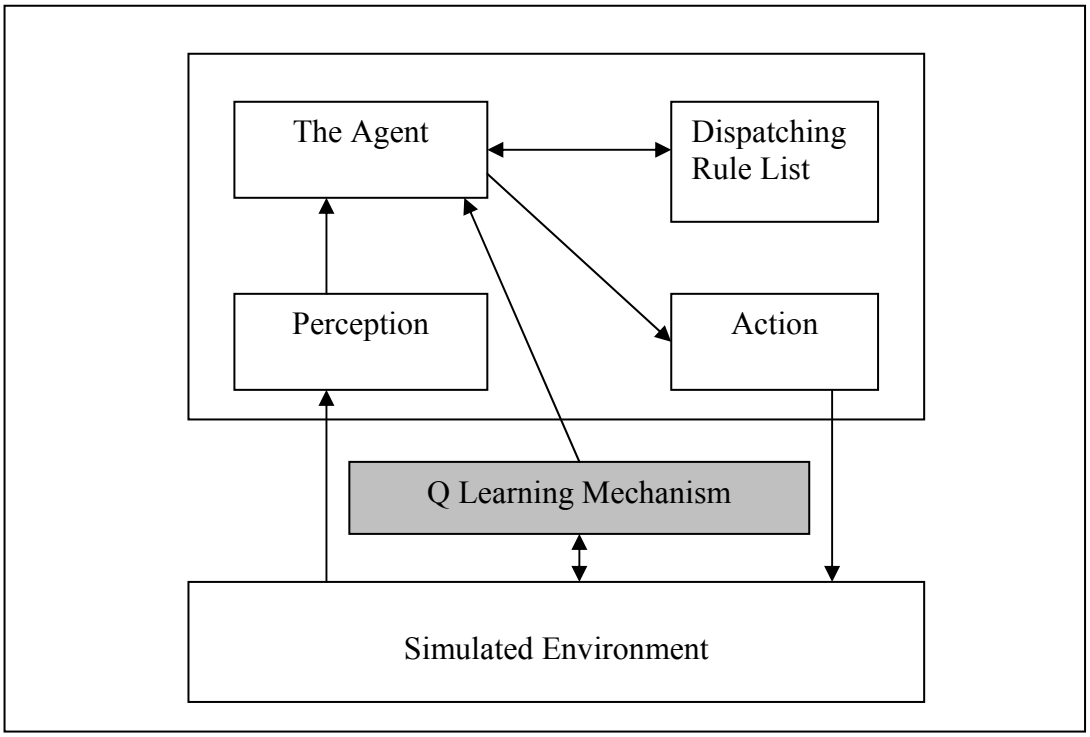

Figure 4.1 The structure of the *Q*-learning agent

4.7.1.5 Simulation and Results

The *Q*-learning agent is located into the single machine simulation model to investigate if the agent can learn to select the best dispatching rule, determined in the first phase, for all system conditions. The parameter settings which are defined in the first phase is used for all simulation experiments and all the values of state-action pairs, $Q(s,a)$, are initialized to zero since all the actions for all states have equal chance to be selected. The *Q*-learning algorithm is applied with the setting of $\varepsilon = 0.1$ (ε-greedy coefficient) for exploration of the actions. The step size parameter, α, is set to 0.1 and the value of the discount rate parameter, γ, is determined as 0.9.

The results are presented in Table 4.9 and Table 4.10. The *SPT* is defined as the best dispatching rule for the case of minimizing mean tardiness except one system condition in the first phase. The number of times that each rule is selected during the simulation run is computed and then selection percentages of these rules are calculated. The agent selects the *SPT* dispatching rule for most of the run time in all system conditions as shown in Table 4.9.

Table 4.9 Selecting percentages of actions for all system conditions

System Condition	**Selecting Percentages of Dispatching Rules (Actions)**		
	EDD	***SPT***	***FIFO***
TDD* and 80% *U	0.16	0.80	0.04
TDD* and 85% *U	0.11	0.84	0.05
TDD* and 90% *U	0.06	0.90	0.04
TDD* and 95% *U	0.04	0.93	0.03
LDD* and 80% *U	0.18	0.76	0.06
LDD* and 85% *U	0.12	0.81	0.07
LDD* and 90% *U	0.11	0.84	0.05
LDD* and 95% *U	0.06	0.89	0.05

Table 4.10 Individual results for all system conditions

System Condition	**Individual Results**			
	EDD	***SPT***	***FIFO***	***Q* Agent**
TDD* and 80% *U	13.27	12.66	13.32	12.72
TDD* and 85% *U	21.05	20.67	21.06	20.85
TDD* and 90% *U	37.39	35.08	37.44	35.28
TDD* and 95% *U	86.81	83.83	86.82	85.12
LDD* and 80% *U	10.71	10.81	10.78	10.75
LDD* and 85% *U	18.73	18.54	18.63	18.59
LDD* and 90% *U	33.75	32.58	33.83	32.64
LDD* and 95% *U	82.54	81.00	82.56	81.42

The agent selects *SPT* instated of *EDD* under the condition of tight due date settings and 80% utilization level although the *EDD* outperforms the *SPT* in individual results. Wang and Usher (2004) expounded this dilemma in their study. They stated that *SPT* may cause some jobs with long processing time to be very tardy causing the overall mean tardiness to be worse even though there are only a few tardy jobs. In our reward function, only non-tardy jobs receive reward, therefore it is not surprising that the selection percentage of *SPT* for this system condition is rather high. Table 4.10 presents the individual results, including the agent's performance

for all system conditions. The selection percentages of *EDD* and *FIFO* are so close for all system conditions since the performance of these dispatching rules are nearly the same. The performance of *Q* agent converges to the performance of *SPT* when the selection percentage of this rule increases. It is also observed that the selection percentage of *SPT* increases with system utilization level. This result means that the agent selects the proper dispatching rule rather frequently when the system is heavy loaded. The number of tardy job and tardiness of these jobs is higher under heavy system condition, therefore the agent receives more reward or punishment in number for its actions and perceives the favor rule earlier. The results demonstrate that the *Q*-learning agent is able to learn to select the proper dispatching rule for each of the system conditions when the system objective is to minimize mean tardiness.

4.7.2 Minimizing Number of Tardy Jobs

In this section, learning potential of the agent is investigated when the objective is to minimize number of tardy jobs. The Hodgson Algorithm (*HA*) outperforms other dispatching rules for minimizing number of tardy jobs (*NTJ*) according to the results of the first phase. Thus, it is expected that the agent learns to select the *HA* under different system conditions.

4.7.2.1 Determination of States and Actions

In this experiment, as it was designated in Section 4.8.1.1, the best three dispatching rules for minimizing number of tardy jobs under all system conditions, *HA*, *SPT* and *LIFO,* are determined as the action set of the *Q*-learning agent. The set of actions, *A*, is defined as follows;

$A = \{a_1, a_2, a_3\}$

Where,

a_1 = execute the *HA*

a_2 = execute the *SPT*

a_3 = execute the *LIFO*

State determination tables presented in Section 4.7.1.2 are also used in this experiment.

4.7.2.2 Determination of Reward Function

In this study, the reward function aims at minimizing the number of tardy jobs. Thus, it is constructed based on being tardy of the job. If the job is not tardy, learning agent receives a reward of +1. Otherwise, the learning agent receives a reward of -1. The reward function for all system conditions given as follows:

Table 4.11 Reward function table

Condition	**Reward**
Tardiness = 0	+1
Tardiness >0	-1

4.7.2.3 Simulation and Results

The *Q*-learning agent is integrated with the dynamic single machine simulation model to determine if the agent can learn to select the best dispatching rule, stated in the first phase, for all system conditions. The parameter settings that defined in Section 4.7.1.5 are used for all simulation experiments.

The *HA* is defined as the best dispatching rule for the case of minimizing number of tardy jobs in the first phase. Selection percentages and individual results are presented in Table 4.12 and Table 4.13. The agent selects the *HA* for most of the run time in all system conditions as shown in Table 4.12. The performance of *Q* agent converges to *HA* when the selection percentage of this rule increases as it was in Section 4.7.1.5, also the selection percentage of *HA* increases with utilization level because of the same reason denoted in Section 4.7.1.5.

Table 4.12 Selecting percentages of actions for all system conditions

System Condition	Selecting Percentages of Dispatching Rules (Actions)		
	HA	***SPT***	***LIFO***
TDD* and 80% *U	0.80	0.07	0.13
TDD* and 85% *U	0.84	0.05	0.11
TDD* and 90% *U	0.86	0.03	0.11
TDD* and 95% *U	0.89	0.03	0.08
LDD* and 80% *U	0.72	0.08	0.20
LDD* and 85% *U	0.76	0.09	0.15
LDD* and 90% *U	0.78	0.07	0.15
LDD* and 95% *U	0.79	0.06	0.15

Table 4.13 Individual results for all system conditions

System Condition	Individual Results			
	HA	***SPT***	***LIFO***	***Q Agent***
TDD* and 80% *U	246706	283767	250127	250176
TDD* and 85% *U	265185	310720	266943	270112
TDD* and 90% *U	283605	330296	289362	288344
TDD* and 95% *U	302660	352356	308847	307982
LDD* and 80% *U	143195	178490	148008	147583
LDD* and 85% *U	157816	204053	159154	159324
LDD* and 90% *U	173523	221469	175569	177225
LDD* and 95% *U	189590	221469	195569	193225

The results indicate that the *Q*-learning agent is able to learn to select the proper dispatching rule for each of the system conditions when the system objective is to minimize number of tardy jobs.

4.8 Third Phase: Application of *Q*-learning Algorithm to Bicriteria Problem

In the first phase of the study, performances of six dispatching rules were tested on a dynamic single machine manufacturing system for two separate cases under different system conditions. It is observed that when the objective is to minimize mean tardiness the *SPT* dispatching rule outperforms the other rules except for one condition where the system has a low utilization level of 80% and a loose due date setting, but *SPT* is not significantly worse than best performing rule. On the other hand, *HA* emerges to the best for all system conditions in the case of minimizing number of tardy jobs.

In the second phase, learning potential of the *Q* agent was investigated in cases of minimizing mean tardiness and minimizing number of tardy jobs individually. It is observed that the agent learns to select the best dispatching rule for each case. The aim of performing the second phase was to test the suitability of the *Q*-learning algorithm application to a dynamic single machine scheduling problem.

In this phase, *Q*-learning agent is trained to minimize mean tardiness and number of tardy jobs on a dynamic single machine bicriteria scheduling problem. The best rules for each measure of performance according to results of first phase, *SPT* and *HA*, are included in training process of the agent. The *Q*-learning agent is trained to learn minimizing which performance measure in which system state.

4.8.1 Determination of States and Actions

The best dispatching rules for each performance measures, *SPT* and *HA*, are determined as the action set of the *Q*-learning agent. Although the *EDD* outperforms the *SPT* in the condition of 80% utilization level and loose due date settings, the *SPT* dispatching rule is preferred as the action for minimizing mean tardiness because of its well performance in minimizing number of tardy jobs besides its close performance to *EDD*. The agent perceives the current system state from the environment and it determines which objective brings in more reward, then selects one of the proper dispatching rule according to the $Q(s,a)$ values. The set of actions, *A*, is defined as follows:

$A = \{a_1, a_2\}$

where,

a_1 = minimize the mean tardiness (execute the *SPT*)

a_2 = minimize the number of tardy jobs (execute the *HA*)

The state determination tables presented in Section 4.7.1.2 are also used in this experiment.

4.8.2 Determination of Reward Function

In Section 4.7.1.3, the reward function was constructed based on the tardiness level of the job, since the objective was to minimize mean tardiness. In the case of minimizing number of tardy jobs, the reward function was constructed based on being tardy of the jobs in Section 4.7.2.2. In this phase, simultaneous minimization of these objectives is considered. Thus, two separate reward functions are developed for each action. The reward function table for 90% utilization level and tight due dates is presented in Table 4.14. The tables for other conditions are shown in Appendix C.

Table 4.14 Reward function table

Minimizing *MT*		**Minimizing *NTJ***	
Condition	**Reward**	**Condition**	**Reward**
$0 \leq s_1 < 5$ and Tardiness< 0	4	$0 \leq s_1 < 5$ and Tardiness < 0	4
$5 \leq s_1 < 8$ and Tardiness< 0	3	$5 \leq s_1 < 8$ and Tardiness < 0	3
$8 \leq s_1 <10$ and Tardiness < 0	2	$8 \leq s_1 < 10$ and Tardiness < 0	2
$s_1 = 10$ and Tardiness < 0	1	$s_1 = 10$ and Tardiness < 0	1
0 ≤ Tardiness <50	-1	$0 \leq s_1 < 3$ and Tardiness > 0	-1
50 ≤ Tardiness <90	-2	$3 \leq s_1 < 6$ and Tardiness > 0	-2
90 ≤ Tardiness <130	-3	$6 \leq s_1 < 8$ and Tardiness > 0	-3
130 ≤ Tardiness <200	-4	$8 \leq s_1 < 10$ and Tardiness > 0	-4
200 ≤ Tardiness	-5	s_1=10 and Tardiness > 0	-5

The s_1 represents the new state. In the case of minimizing mean tardiness, there are ten tardiness levels in the reward function, therefore ten reward values,+1 to -9, are considered for the agent's actions. However, there are only two reward values, +1 and -1, in the reward function when the objective is to minimize number of tardy jobs. In this study, the objective is to minimize these performance measures together. Thus, the reward functions developed in Sections 4.7.1.3 and 4.7.2.2 are common in this experiment. A new classification scheme is considered to balance the number of reward values of these reward functions. The Agent receives more reward if the new state is a good state; on the other hand, if the new state is a bad state, the agent receives more negative reward. The state tables are constructed according to estimated total lateness (*ETL*) of the jobs in the buffer. If the *ETL* of the jobs is low,

then, it refers to a good state. The reward which agent receives decreases as the *ETL* increases.

The *Q*-learning algorithm used a single reward function in the first phase. A dual reward function is used in the third phase. If the *SPT* is selected as an action, the agent receives the reward from the first reward function since the objective is to minimize mean tardiness. Otherwise, the agent receives the reward from the second function when the selected action is *HA*. The simulation model of the system for 90% utilization level and tight due dates is presented in Appendix N.

4.8.3 Simulation and Results

The *Q*-learning agent is inserted to a dynamic single machine simulation model to determine minimizing which objective is more effective in which system state. The parameter settings which are defined in Section 4.7.1.5 are used for all simulation experiments. Table 4.15 presents the selection percentages of the dispatching rules under all system conditions.

Table 4.15 Selection percentages of dispatching rules for all conditions

Dispatching Rule	**80% *U***		**85% *U***		**90% *U***		**95% *U***	
	TDD	***LDD***	***TDD***	***LDD***	***TDD***	***LDD***	***TDD***	***LDD***
HA	0.58	0.57	0.48	0.49	0.51	0.56	0.53	0.51
SPT	0.42	0.43	0.52	0.51	0.49	0.44	0.47	0.49

For instance, selection percentages of *SPT* and *HA* are 58%and 42% under conditions of 80% utilization level and tight due date settings. In addition to Table 4.15, selection percentages of these dispatching rules in each state are shown in Table 4.16 and Table 4.17.

According to Table4.16 and Table 4.17, it can be seen that the agent prefers *HA* on system states of 1 to 6. This means that minimizing number of tardy jobs on these states is more effective and. On the other hand, *SPT* dispatching rule is selected on the states of 7 to 10 generally. The changes in the percentages are clearly seen in Figures 4.2 to 4.9.

Table 4.16 Selection percentages of dispatching rules in each state for tight due dates

State No	80% *U*		85% *U*		90% *U*		95% *U*	
	HA	***SPT***	***HA***	***SPT***	***HA***	***SPT***	***HA***	***SPT***
1	0.86	0.14	0.89	0.11	0.90	0.10	0.87	0.13
2	0.89	0.11	0.85	0.15	0.85	0.15	0.85	0.15
3	0.87	0.13	0.88	0.12	0.85	0.15	0.85	0.15
4	0.81	0.19	0.81	0.19	0.84	0.16	0.83	0.17
5	0.77	0.23	0.80	0.20	0.83	0.17	0.81	0.19
6	0.56	0.44	0.60	0.40	0.75	0.25	0.75	0.25
7	0.19	0.81	0.22	0.78	0.71	0.29	0.71	0.29
8	0.10	0.90	0.12	0.88	0.35	0.65	0.37	0.63
9	0.08	0.92	0.08	0.92	0.29	0.71	0.25	0.75
10	0.09	0.91	0.09	0.91	0.11	0.89	0.12	0.88

Table 4.17 Selection percentages of dispatching rules in each state for loose due dates

State No	80% *U*		85% *U*		90% *U*		95% *U*	
	HA	***SPT***	***HA***	***SPT***	***HA***	***SPT***	***HA***	***SPT***
1	0.61	0.39	0.53	0.47	0.56	0.44	0.88	0.12
2	0.78	0.22	0.70	0.30	0.78	0.22	0.85	0.15
3	0.71	0.29	0.64	0.36	0.77	0.23	0.81	0.19
4	0.65	0.35	0.54	0.56	0.68	0.32	0.78	0.22
5	0.62	0.38	0.52	0.48	0.71	0.29	0.75	0.25
6	0.44	0.56	0.43	0.57	0.59	0.41	0.60	0.40
7	0.45	0.55	0.39	0.61	0.51	0.49	0.35	0.65
8	0.25	0.75	0.28	0.72	0.41	0.59	0.28	0.72
9	0.20	0.80	0.29	0.71	0.40	0.60	0.18	0.82
10	0.27	0.73	0.31	0.67	0.46	0.54	0.15	0.85

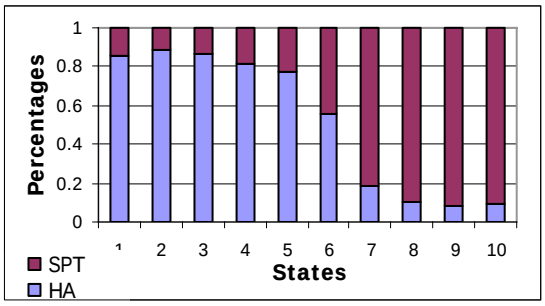

Figure 4.2 Selection percentages for 80% and *TDD*

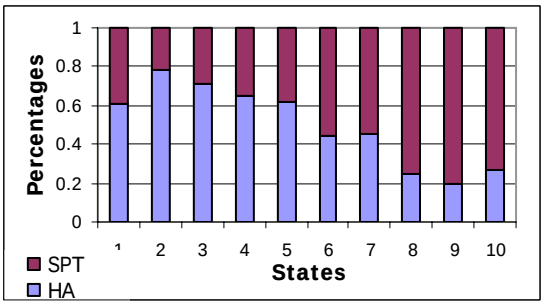

Figure 4.3 Selection percentages for 80% and *LDD*

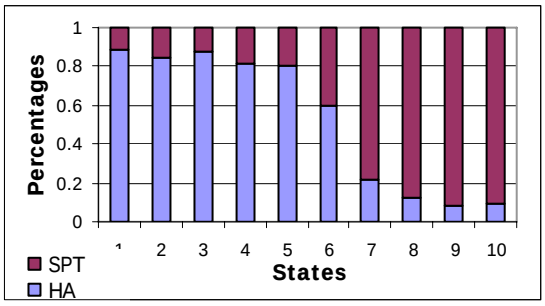

Figure 4.4 Selection percentages for 85% and *TDD*

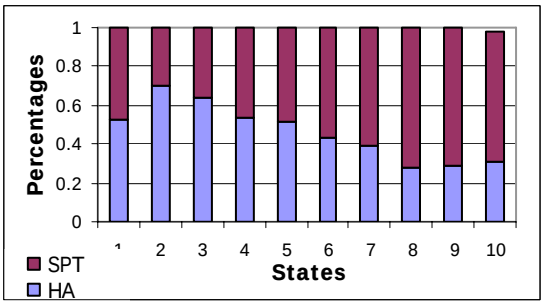

Figure 4.5 Selection percentages for 85% and *LDD*

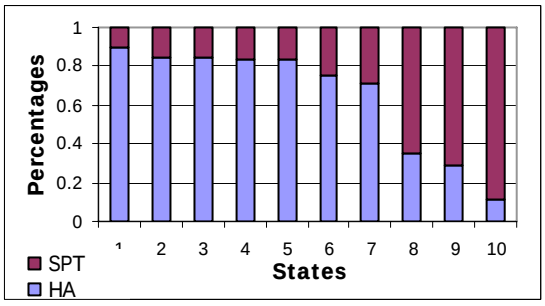

Figure 4.6 Selection percentages for 90% and *TDD*

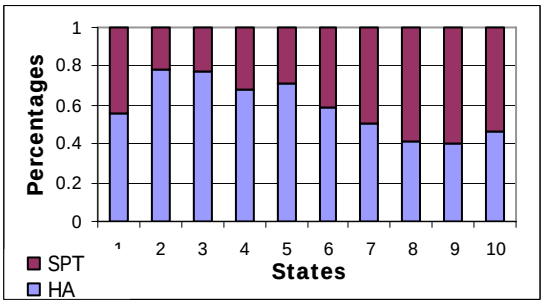

Figure 4.7 Selection percentages for 90% and *LDD*

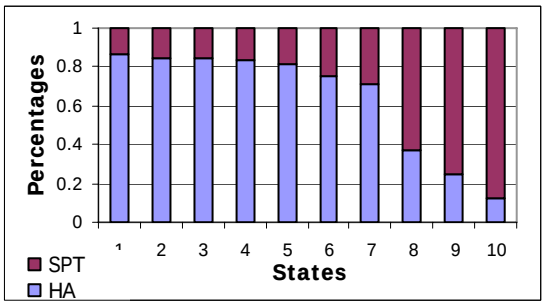

Figure 4.8 Selection percentages for 95% and *TDD*

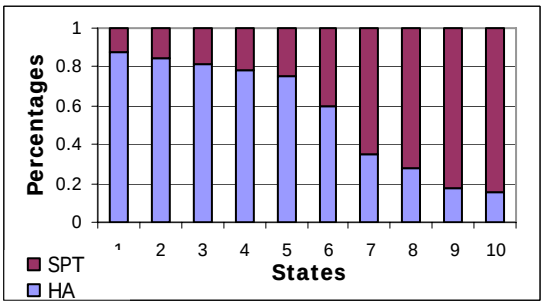

Figure 4.9 Selection percentages for 95% and *LDD*

It is clearly seen that the dominance of the selected dispatching rule is more distinct when the tight due date parameter settings are considered. For instance, selection percentage of *HA* is obviously greater then *SPT*'s on states 1 to 7 under the system condition of 90% utilization level and tight due dates. Likewise, *SPT* outstands *HA* on states 8, 9 and 10 under the same system condition. However, when the jobs have loose due dates, the clarity of the dominant rule reduces. It is also observed that the agent preferred to execute the *HA* on states 1 to 7. That means agent receives more reward from minimizing number of tardy jobs when the *ETL* is low. On the other hand, it is effective to minimize mean tardiness when the *ETL* increases. The performances of *Q*-agent for all system conditions and individual results of other dispatching rules are given in Table 4.18 and Table 4.19.

Table 4.18 Performance of the *Q* agent and individual results for tight due dates

Dispatching Rule	**80% *U***		**85% U**		**90% *U***		**95% *U***	
	MT	***NTJ***	***MT***	***NTJ***	***MT***	***NTJ***	***MT***	***NTJ***
SPT	12.66	283767	20.67	310720	35.08	330296	83.83	352356
HA	13.79	246706	21.63	265185	38.07	283605	87.70	302660
FIFO	13.32	349438	21.06	384040	37.44	420660	86.82	459336
LIFO	13.88	255127	21.76	266943	38.27	289362	87.80	291847
EDD	13.27	350707	21.05	385244	37.39	421530	86.81	459867
RR	13.20	346598	21.03	383994	37.26	418283	86.76	459203
***Q*-Agent**	12.98	263275	20.87	286840	35.19	302844	84.10	321457
Test 1	13.32	268813	21.13	296045	36.61	308352	85.10	329457
Test 2	13.32	272423	21.15	295601	36.63	307738	85.18	329582
Test 3	13.26	268997	21.06	294665	36.40	318897	84.95	331552
Test 4	13.05	266445	20.91	290554	35.45	305827	84.32	325165

Table 4.19 Performance of the *Q* agent and individual results for loose due dates

Dispatching Rule	**80% *U***		**85% U**		**90% *U***		**95% *U***	
	MT	***NTJ***	***MT***	***NTJ***	***MT***	***NTJ***	***MT***	***NTJ***
SPT	10.81	178490	18.54	204053	32.58	221469	81.00	243350
HA	11.84	143195	19.43	157816	35.44	173523	84.76	189590
FIFO	10.78	284832	18.63	329859	33.83	381032	82.56	437465
LIFO	12.41	148008	20.06	159154	36.33	175569	85.61	195973
EDD	10.71	285560	18.73	330779	33.75	381679	82.54	437909
RR	10.97	281096	18.98	329622	33.66	337768	82.50	437222
***Q*-Agent**	11.05	164375	18.69	184986	32.93	193185	81.45	212335
Test 1	11.40	168496	18.95	190587	34.24	197734	82.28	220505
Test 2	11.41	168388	18.95	189665	34.23	197169	82.33	221225
Test 3	11.21	171014	18.93	190579	33.63	211045	82.05	222567
Test 4	11.11	166228	18.75	186556	33.02	196027	81.58	216163

As seen in Table 4.18 and Table 4.19, four test methods are also investigated to evaluate the performance of *Q* agent. These test methods are applied according to the results in Table 4.15, 4.16 and 4.17. In the first test method (Test 1), *HA* is selected continuously as a dispatching rule at the first part of the simulation run. The range of this part is determined with respect to the percentages in Table 4.15. For instance, *HA* is executed during the first 58% part of the simulation run for 80% utilization level and tight due dates. Then *SPT* is selected for the second part of 42%. The second test method (Test 2) is the inverse of the Test 1. *SPT* dispatching rule is preferred in the first part of the simulation run. In the third test method (Test 3), these two rules are implemented randomly respect to the percentages in Table 4.15. It is observed that *Q*

agent outperforms these three test methods although the selection percentages of dispatching rules are equal for the agent and test methods. A forth test method (Test 4) is also investigated to support the performance of the agent. In this test method, the dominant dispatching rule is executed for all states when these states occur. For instance, *HA* is selected for the second state and *SPT* is selected for the eighth state under the condition of 90% utilization level and tight due dates according to Table 4.16. Test 4 performed well and outperformed other test methods.

The performance of *Q*-learning agent is between *HA* and *SPT* as expected. The agent outperforms the *SPT* when the number of tardy jobs is considered but *SPT* is better then the *Q*-learning agent for minimizing mean tardiness. On the other hand, the agent outperforms the *HA* in case of mean tardiness but *HA* emerges to the best for number of tardy jobs. The results are normalized and evaluated to investigate the performance of the *Q*-learning agent under all conditions in following section.

4.8.4 Performance Analysis of Q-learning Agent

Generally, different attributes have different scales of measurement. Thus, attributes often conflict with each other and become incomparable. Normalization aims at obtaining comparable scales. Normalization of the attributes is necessary for multiobjective decision making methods. One of the most popular of these methods is Simple Additive Weighting (*SAW*) method (Hwang and Yoon, 1981). The *SAW* method uses the following equation to evaluate the utility value of the x^{th} alternative;

$$U_x = \sum_{i=1}^{m} w_i Y_{ix} \tag{4.11}$$

Where,

U_x: The utility value of x^{th} alternative, x: 1,…, v

v: The total number of alternatives

m: The number of objectives

w_i: The weight for i^{th} objective

Y_{ix}: The normalized value of i^{th} objective for x^{th} alternative

Simple additive weighting method provides alternatives to satisfy decision maker's requirements depending on the importance (weight) of the objectives. In multiobjective studies using *SAW* method, generally, weights of the criteria are determined by decision maker. Various weights are tested to evaluate the performances of the alternatives. However, expressing the importance of the criteria with weights and determining the values of these weights may be difficult. In this thesis, a new ranking method is developed to analyze the performances of the alternatives. The aim of the proposed method is to obtain a weight-independent-rank for each alternative. The new method is constructed and simulated in *C*. The program gets the values of each alternative for criterion as inputs. Before ranking the alternatives, the program normalizes the values according to following equation;

$$Y_i = \frac{x_i - \frac{\max_x - \min_x}{2}}{\frac{\max_x + \min_x}{2}}, \quad i=1,2,\ldots,n \tag{4.12}$$

where,

n = Total number of methods

Y_i = Normalized value of the i^{th} alternative

x_i = Value of the i^{th} alternative

$\min_x$ = minimum x_i , i=1,2…,n

$\max_x$ = maximum x_i , i=1,2…,n

Steps of the proposed ranking method are given as follows;

Step 1: Normalize the values of each criterion between -1 and 1

Step 2: Multiple each normalized value by -1 (or 1)

Step 3: Generate a number, a, between 0 and 1

Step 4: Let $w_1 = a$ and $w_2 = 1 - a$

Step 5: Evaluate the utility values of each alternative, U_{ij}, according to weights

Step 6: Rank the alternatives according to U_{ij}

Step 7: Let $R_{itk} = R_{itk} + 1, \quad i=1,2\ldots m$

Step 8: Repeat step 3 to 7 for n

Step 9: Let $AR_{ik} = \dfrac{\sum_{t=1}^{m} R_{itk} \times t}{n}, \quad i=1,2\ldots m$

Step 10: Repeat step 3 to 9 for k

where,

w_1 = Weight of the first criteion

w_2 = Weight of the second criteion

U_{ij} = Utility values of the i^{th} alternative for j^{th} generated number

m = Total number of alternatives

n = Number of generated numbers

k = Total number of replication

t = rank indicator

R_{itk} = Total times of taking place at the t^{th} rank of the i^{th} alternative for k^{th} replication

AR_{ik} = Average rank of the i^{th} alternative for k^{th} replication

After the normalization, values are multiplied by -1 if the criterion is cost based since the minimum value is the best in this case. Thereafter, very large amount of weights between 0 and 1 are generated and weights are determined. For each time a number is generated, utility values of the methods are calculated. Utility values vary between -1 and 1. Ranks of the methods are determined depending on the utility values. Then, total times of taking place at each rank of each alternative is traced and evaluated during the simulation. At the end of the replication, these values are multiplied by related rank and summed each other. Finally, total value is divided by total number of generated numbers and average rank is calculated for each alternative. Consequently, a weight independent rank is obtained for each alternative through using this proposed ranking method.

In this study, eleven methods (six dispatching rules, *Q*-learning agent and four test methods) are tested for two performance measures (*MT* and *NTJ*). Thus, values

of eleven alternatives for two criteria are the inputs of the program. All Simulation experiments consist of 30 replications. In each replication, 200000 weights between 0 and 1 were generated. The w_1 represents the weight of the mean tardiness and w_2 is the weight of number of tardy jobs. Simulation results for 90% utilization level and tight due dates are presented in Table 4.20. Importance of the both objectives is equal since weights vary between 0 and 1 ($\overline{w}_1 = \overline{w}_2$). *Q*-learning agent outperforms all the other methods when the weight independent ranking considered. Test 4 is also performs well under this system condition, that is, preferred actions of the agent in each state are effective. Average rank of the *Q*-agent is 1.37326. Therefore, the agent placed at first rank for most of the weights generated.

The half width for each method is calculated to measure the precision of the average ranking estimation of the true but unknown mean. Half width gives a 95% confidence interval for the exact results which are centered at simulation results. Maximum (Upper confidence limits) values, minimum (lower confidence limits) values and the half-widths of the methods are also given in Table.4.20. The results for other conditions are presented in Appendix J.1.

Table 4.20 Ranking results for 90% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	3.61626	3.61847	3.62068	0.00221
HA	6.09310	6.09571	6.09831	0.00260
FIFO	10.33200	10.33261	10.33322	0.00061
LIFO	7.56614	7.56852	7.57090	0.00238
EDD	9.88662	9.88749	9.88836	0.00087
RR	8.53993	8.54051	8.54110	0.00059
***Q*-Agent**	1.37326	1.37388	1.37450	0.00062
Test 1	5.11920	5.11964	5.12009	0.00044
Test 2	5.28911	5.28972	5.29034	0.00061
Test 3	5.49345	5.49474	5.49604	0.00129
Test 4	2.67815	2.67870	2.67926	0.00055

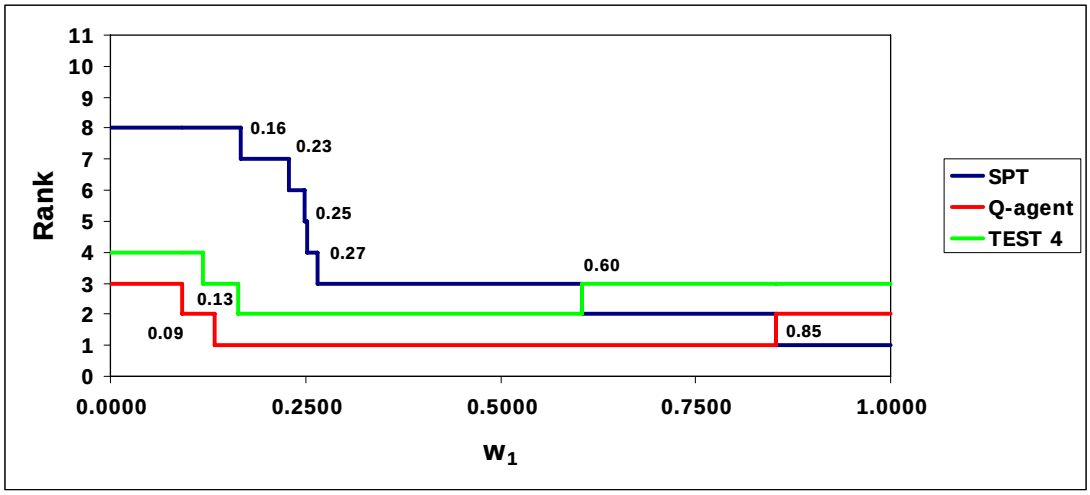

Figure 4.10 Ranks of the first three methods with respect to w_1

The ranks of the first three methods (well performed) for whole all weights generated for the condition of 90% *U* and *k*=3 are presented in Figure 4.10. The ranking figures for the other conditions are illustrated in Appendix M.1. It is shown that the region under the each line gives the average ranking of the methods. For instance, the region under the line which represents the *SPT* is 3.62. Hence, average ranks of the all methods can be calculated by determining edge points. However, it takes too much time to determine all these points. Besides, if number of the objectives is more than two, it is impossible to evaluate the regions. Thus, proposed method is more effective and not bounded by the number of objectives. It is also observed that the half width value of the *Q* agent is lower. In the other words, the performance of the agent is robust for each weight. Even though the importance of the objectives switches, rank of the *Q*-learning agent does not change as much as the other methods. On the other hand, performance of the *SPT* is unstable as seen in Figure 4.10. Ranks of the *SPT* vary between 1 and 8. It is also seen that the half width value of the *FIFO* is as lower as the agent but its average rank is 10.33261. Thus, a lower half width value is effective only if the rank is also lower.

The results show that the *Q* agent emerges to the best when the weight independent ranking is considered. Test 4 also performed well under all conditions.

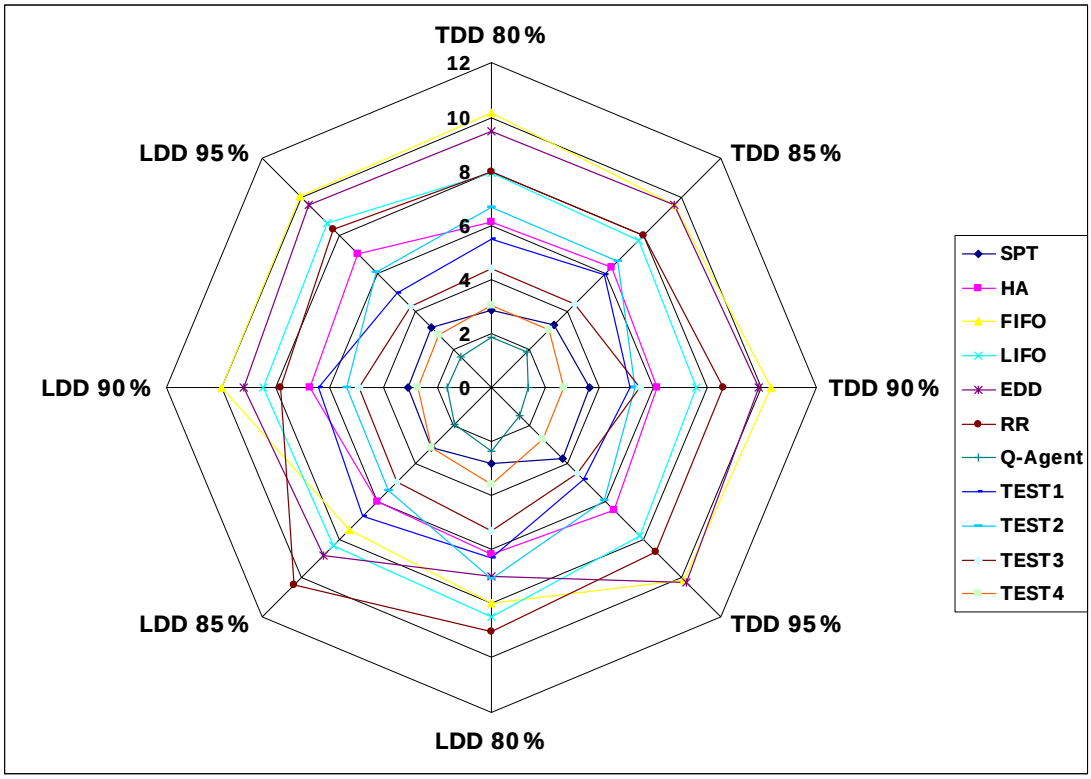

Figure 4.11 Ranks of all methods for all conditions

Figure 4.11 also shows the ranks of all methods for all conditions. It is clearly seen that the agent places at the middle of the radar graphic and outperforms other methods.

Proposed ranking method is also executed for different two cases. Decision maker may attach importance one of two objectives. In this sense, weights are generated between 0 and 1, as it was in the first experiment, but the bigger weight is assigned always to the objective which is important. Parameters used in the first experiment are also used in these two experiments. The results when $\overline{w}_1 > \overline{w}_2$ for 90% utilization level and tight due dates are presented in Table 4.21 and Figure 4.12.

Table 4.21 Ranking results for 90% *U* and *TDD* when $\overline{w}_1 > \overline{w}_2$

Methods	Maximum	Average	Minimum	Half Width
SPT	1.91352	1.91406	1.91460	0.00054
HA	8.09489	8.09588	8.09687	0.00099
FIFO	10.23836	10.23903	10.23971	0.00067
LIFO	9.38136	9.38261	9.38386	0.00125
EDD	9.20068	9.20140	9.20212	0.00072
RR	8.08026	8.08107	8.08188	0.00081
***Q*-Agent**	1.29503	1.29535	1.29567	0.00032
Test 1	4.92513	4.92530	4.92548	0.00018
Test 2	5.94217	5.94234	5.94251	0.00017
Test 3	4.13202	4.13235	4.13269	0.00034
Test 4	2.79027	2.79059	2.79091	0.00032

When $\overline{w}_1 > \overline{w}_2$, *Q* agent has the best performance only for two conditions, 90%, 95% and tight due dates. *SPT* dispatching rule outperforms the agent under the rest of the conditions since the *SPT* performed well for number of tardy jobs as well as for mean tardiness in individual results. It is also seen that the *SPT* placed at middle of the graphic in Figure 4.12.

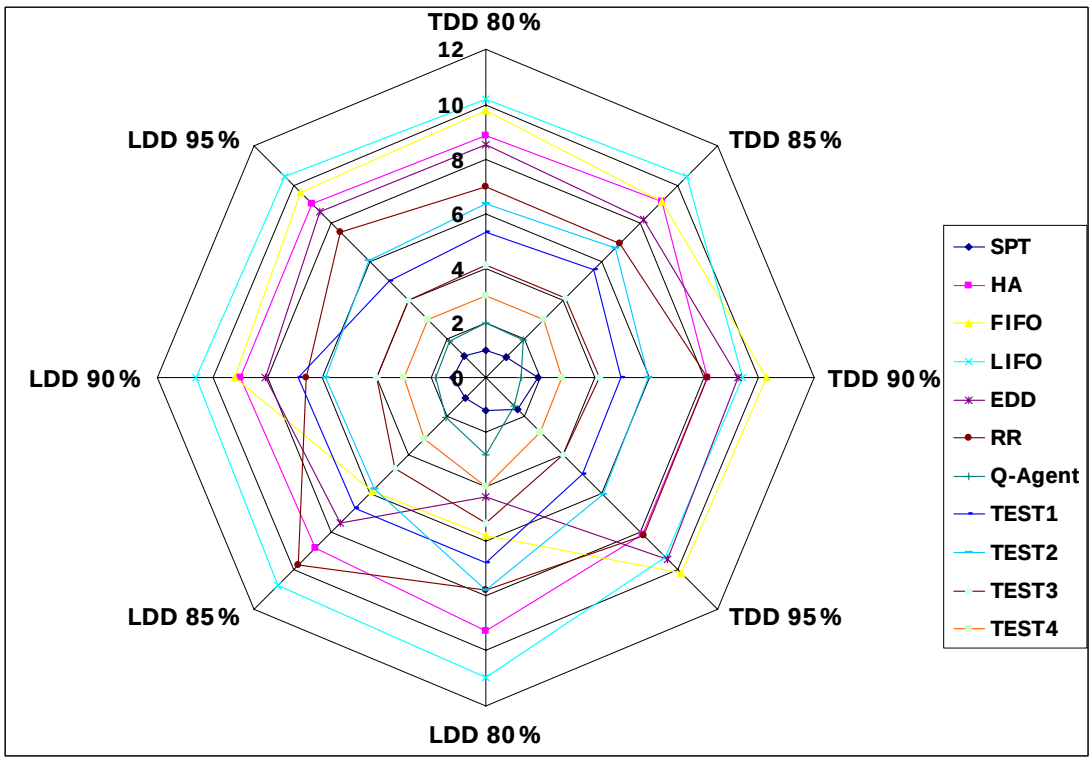

Figure 4.12 Ranks of all methods for all conditions when $\overline{w}_1 > \overline{w}_2$

The results when $\overline{w}_1 < \overline{w}_2$ for 90% utilization level and tight due dates are presented in Table 4.22 and Figure 4.13.

Table 4.22 Ranking results for 90% *U* and *TDD* when $\overline{w}_1 < \overline{w}_2$

Methods	Maximum	Average	Minimum	Half Width
SPT	5.32206	5.32344	5.32482	0.00138
HA	4.09335	4.09495	4.09655	0.00160
FIFO	10.42577	10.42608	10.42640	0.00031
LIFO	5.75353	5.75501	5.75650	0.00149
EDD	10.57360	10.57392	10.57423	0.00031
RR	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.45111	1.45165	1.45219	0.00054
Test 1	5.31338	5.31389	5.31441	0.00052
Test 2	4.63574	4.63618	4.63662	0.00044
Test 3	6.85822	6.85874	6.85926	0.00052
Test 4	2.56549	2.56614	2.56678	0.00065

When $\overline{w}_1 < \overline{w}_2$, *Q* agent outperforms all other methods including *HA* since the *HA* has a poor performance for mean tardiness although it emerges to the best for number of tardy jobs in individual results. It is seen that the *Q* agent placed at middle of the graphic in Figure 4.13. It is also observed that Test 4 placed at the second rank for all conditions.

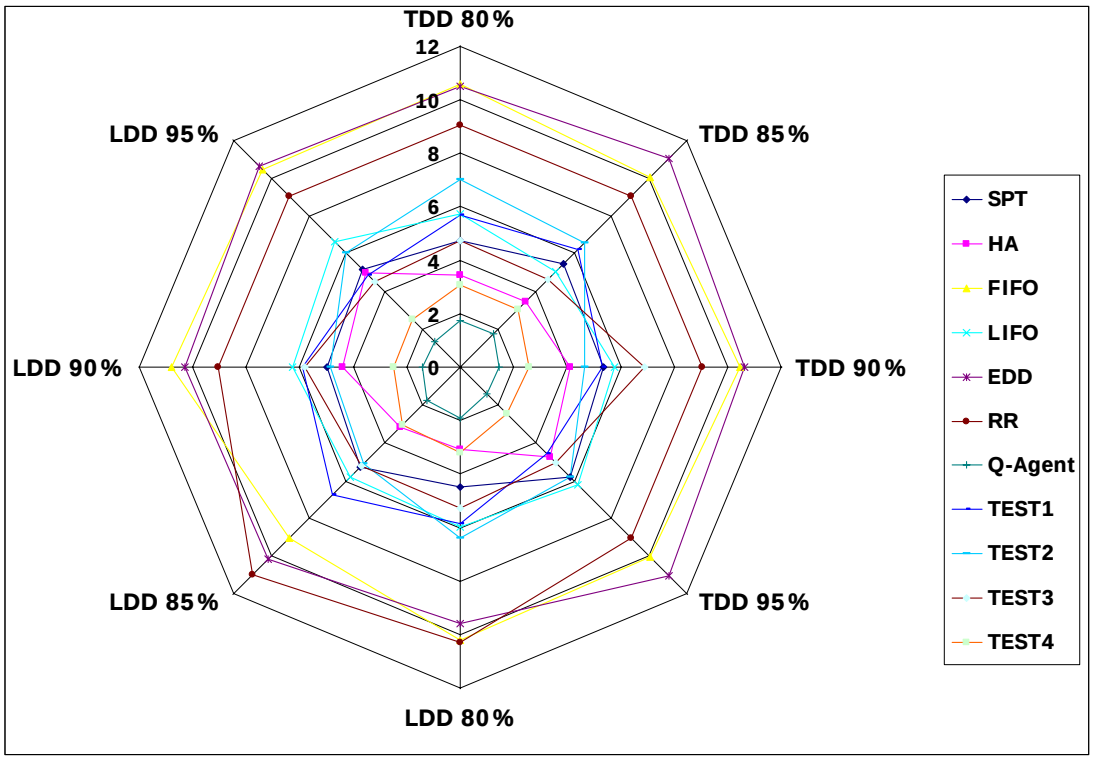

Figure 4.13 Ranks of all methods for all conditions $\overline{w}_1 < \overline{w}_2$

The results for these two cases under other conditions are presented in Appendix J.2 and Appendix J.3.

CHAPTER FIVE
APPLICATION OF *Q*-LEARNING ALGORITHM TO A BICRITERIA DYNAMIC FLOW SHOP SCHEDULING PROBLEM

5.1 Introduction

This chapter presents a solution approach to a dynamic flow shop bicriteria scheduling problem. The problem consists of two different objectives; minimizing mean tardiness (*MT*) and minimizing mean flow time (*MF*). The *Q*-learning algorithm is determined as a solution approach to minimize these performance measures under different system conditions.

The methodology used in the previous chapter is also used for this problem. Thus, this chapter involves three phases. As it was in the first chapter, the performances of dispatching rules are tested on a dynamic flow shop system under different system conditions in the first phase. In the second phase, learning potential of the *Q*-learning agent is investigated for *MT* and *MF* individually. Finally, in the third phase, *Q*-learning agent is trained to minimize mean tardiness and mean flow time on a dynamic flow shop scheduling problem.

The description of the system, performance measures of system and dispatching rules involved in the study are presented in following sections.

5.2 Description of the System

In this study, a dynamic flow shop manufacturing system is considered. System consists of ten machines and ten buffers for storing jobs waiting for processing. Following assumptions are considered for the system:

1. Machines can process only one job at a time.
2. There are no interruptions and cancellations between jobs.
3. Resources are continuously available for the process

4. The job is processed immediately when it arrives to the machines.
5. If the machine is not idle then job is sent to the related buffer.
6. Each job consists of only one operation requiring a variant processing time.
7. All jobs undergo processing on all ten machines sequentially starting from the first machine.
8. Selection of the next job from the buffer for processing is conducted based on a predefined dispatching rule.

5.3 Performance Measures

In this study, mean tardiness and mean flow time are investigated under different system conditions in three phases. The mean tardiness was presented in Section 4.4. Equation (5.2) gives the definition of mean flow time.

$$F_j = C_j - t_j \tag{5.1}$$

$$\text{Mean Flow Time} = \frac{\sum_{j=1}^{n} F_j}{n} \tag{5.2}$$

Where F_j is the flow time, C_j is the completion time, t_j is the arrival time of the job j and n is the number of jobs processed.

5.4 Dispatching rules

The dispatching rules involving this study are observed many times in flow shop and job shop scheduling researches in the literature. The *SPT*, *FIFO* and *EDD* dispatching rules were defined in Section 4.5. The other dispatching rules considered in this study are presented below with the terminology used in Section 4.5.

Slack Per Remaining Operation (*S/OPN*)

The job with the minimum slack per remaining number of operations is chosen for loading. The *S/OPN* rule is effective in minimizing tardiness related measures of performance. The priority of job *i* is given as follows:

$$Z_i = \begin{cases} s_i /(o(i)-j+1) & s_i \geq 0 \\ s_i \times (o(i)-j+1) & s_i < 0 \end{cases} \tag{5.3}$$

where $o(i)$ is total number of operations of job *i* and *j* is the operation index.

- **A Rule by Rajendran and Holthaus (*AT-RPT*)**

Rajendran and Holthaus (1999) presented *AT-RPT* rule in their study. This rule tries to minimize the maximum flow time and variance of flow time of the jobs. The job with the least Z_i is chosen for loading. The priority index is expressed as follows:

$$Z_i = -(\tau - t_i) - \sum_{q=j}^{o(i)} p_{iq} \tag{5.4}$$

- **Cost Over Time (*COVERT*)**

This rule is effective when the tardiness related performance measures are considered in job shop and flow shop scheduling. The *COVERT* rule calculates a penalty function, c_i, depending upon the slack of job *i*, s_i, and sum of the expected waiting times for job's uncompleted operations, WT_i, and determines the priority index, Z_i, of the job. The job with the largest Z_i is chosen for loading. The priority index is given as follows:

$$Z_i = c_i / p_{ij} \tag{5.5}$$

where c_i is;

$$c_i = \begin{cases} (WT_i - s_i)/WT_i & 0 \le s_i < WT_i \\ 0 & s_i \ge WT_i \\ 1 & s_i < 0 \end{cases} \quad (5.6)$$

- **A Rule by Rajendran and Holthaus (*PT/TIS*)**

Rajendran and Holthaus (1999) denoted that this rule is developed based on the process time and the resident time of the job. The job with the minimum Z_i is chosen for loading. The priority index is defined as follows;

$$Z_i = p_{ij} / (\tau - t_i) \quad (5.7)$$

- **Process Time Plus Work-in-Next Queue Plus Arrival Time (*PT+WINQ+AT*)**

This rule is also proposed by Rajendran and Holthaus (1997), and is found to be quite effective in minimizing the maximum flow time and variance of flow time. The job with the minimum Z_i is chosen for loading. The priority index is given as follows;

$$Z_i = p_{ij} + W_i + t_i \quad (5.8)$$

W_i indicates the total work content of jobs in the queue of the next operation of job i.

- **Process Time Plus Work-in-Next Queue Plus Negative Slack (*PT+WINQ+SL*)**

This rule is also developed by Rajendran and Holthaus (1997) and minimizes the maximum tardiness and variance of tardiness of the jobs. The job with the smallest Z_i is chosen for loading. The priority index is expressd as follows:

$$Z_i = p_{ij} + W_i + \min \{s_i; 0\} \quad (5.9)$$

5.5 First Phase: Performance Analysis of Dispatching Rules Individually

In this section, dispatching rules determined in previous section are tested for mean tardiness and mean flow time under different system conditions in simulation models. The model for flow shop simulation is constructed in ARENA 10.0 software.

5.5.1 Experimental Design for the Simulation Study

The experiments have been conducted in a dynamic flow shop simulation model, consisting ten machines. The processing times are drawn from a uniform distribution ranging from 1 to 59. The total work content (*TWK*) method is used for due date assignments of the jobs with the allowance factors of 3, 4, 5 and 6. The job arrivals are generated using an exponential distribution. Three machine utilization levels, 85%, 90% and 95%, are considered in the experiments. Thus, there are four different due date settings and three different utilization levels, giving a total number of 12 simulation experiment for every dispatching rule.

Each simulation experiment consists of 20 replications. In each run, the shop is continuously loaded with job orders. It has been found that the shop reaches a steady state after arrival of about 2500 job orders. The simulation run length is fixed as 50000 completed job orders for every simulation experiment. Table 5.1 gives all the parameter settings for all allowance factors:

Table 5.1 Parameter settings for all experiments

	85% *U*	**90% *U***	**95% *U***
Arrivals	Expo(35.3)	Expo(33.3)	Expo(31.58)
Processing time	Unif(1,59)	Unif(1,59)	Unif(1,59)

5.5.2 Results and Discussion

The performances of nine dispatching rules are observed for the objectives of minimizing mean tardiness and mean flow time. The results of the simulation study are presented in Tables 5.2, 5.3, 5.4 and 5.5. The term *k* refers to allowance factor.

Table 5.2 Individual results for *k*=3

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	148.21	913.43	442.86	1293.30	1473.12	2368.02
AT-RPT	155.42	920.8	449.85	1299.63	1477.99	2372.80
EDD	120.85	887.72	412.33	1266.98	1446.40	2342.34
S/OPN	124.306	894.66	417.10	1273.14	1452.54	2348.61
COVERT	52.37	777.91	185.14	992.55	676.06	1558.27
SPT	92.87	706.16	250.24	898.27	780.81	1416.45
PT/TIS	60.64	744.93	190.11	998.90	741.90	1613.99
PT+WINQ+AT	123.006	869.53	394.24	1234.09	1397.86	2291.11
PT+WINQ+SL	77.37	787.74	297.57	1091.99	1291.02	2176.31

Table 5.3 Individual results for *k*=4

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	58.53	913.43	257.94	1293.30	1192.64	2368.02
AT-RPT	62.51	920.8	263.73	1299.63	1197.44	2372.80
EDD	37.19	881.25	221.03	1259.01	1153.56	2332.29
S/OPN	38.55	887.61	224.53	1265.84	1161.46	2340.72
COVERT	15.56	772.87	79.83	985.86	489.79	1613.01
SPT	34.0021	706.16	120.04	898.90	593.01	1416.45
PT/TIS	25.06	744.93	83.51	998.90	515.71	1613.99
PT+WINQ+AT	46.16	869.53	223.21	1234.09	1121.61	2291.11
PT+WINQ+SL	30.9	782.09	130.77	1008.77	928.51	2053.42

The *COVERT* is the best rule for minimizing mean tardiness except for one case where utilization level is 85% and the allowance factor is 6. The *PT/TIS* rule emerges to the best in this case but the *COVERT* rule is not significantly worse than the *PT/TIS* rule.

The *SPT* rule outperforms the other dispatching rules for minimizing mean tardiness except for two cases. The *PT+WINQ+SL* rule is better than *SPT* under conditions of 85% utilization level and loose due date settings of k=5 and k=6.

Table 5.4 Individual results for k=5

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	21.99	913.43	142.07	1293.3	939.33	2368.02
AT-RPT	23.91	920.8	146.23	1299.63	943.95	2372.8
EDD	9.82	875.99	107.2	1251.68	889.45	2323.67
S/OPN	10.17	881.45	109.03	1257.89	895.36	2331.1
COVERT	3.27	728.25	31.1	960.23	334.01	1632.79
SPT	43.15	706.16	50.7	898.9	490.99	1416.45
PT/TIS	3.99	744.93	37.88	998.9	345.53	1613.98
PT+WINQ+AT	16.71	869.53	119.98	1234.09	875.45	2291.11
PT+WINQ+SL	8.85	703.42	43.02	966.4	633.01	1925.2

Table 5.5 Individual results for k=6

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	8.34	913.43	75.72	1293.3	722.16	2368.02
AT-RPT	9.17	920.8	78.41	1299.63	726.42	2372.80
EDD	2.4	870.22	48.14	1244.41	662.60	2314.37
S/OPN	2.56	875.97	48.92	1250.12	666.02	2320.35
COVERT	1.23	721.56	13.99	956.96	214.27	1621.27
SPT	31.85	706.16	25.4	898.9	365.77	1416.45
PT/TIS	1.09	744.93	20.21	998.9	225.58	1613.98
PT+WINQ+AT	6.2	869.53	60.67	1234.09	667.28	2291.11
PT+WINQ+SL	5.45	696.83	22.31	957.88	330.69	1801.84

5.6 Second Phase: Analysis of Learning Potential of the *Q*-learning Agent

In this phase, the application potential of the *Q*-learning algorithm to a dynamic flow shop dispatching rule selection problem is investigated. Purpose of this study is to determine whether the single machine agent enables to learn to select the well-performed-rules which are designated in the first phase.

5.6.1 Minimizing Mean Tardiness

In this section, learning potential of the agent is investigated when the objective is to minimize mean tardiness. It is obvious that the *COVERT* dispatching rule performed better than the other dispatching rules for *MT* except for one case according to the results of the first phase. Thus, it is expected that the agent learns to

select the *COVERT* rule under different system conditions and learns to select the *PT/TIS* rule when the utilization level is 85% and allowance factor is equal to 6. Before performing the learning process, set of the actions, system states and reward function are determined in the following sections.

5.6.1.1 Set of Actions

In this experiment, selection of the next job from the buffer for processing is conducted according to three dispatching rules which are well-performed for mean tardiness in individual simulation experiments in the first phase. The set of best three dispatching rules changes depending on the system conditions. The *COVERT* and the *PT/TIS* dispatching rules take place in the action set for all system conditions. The action sets for all conditions are presented in Table 5.6.

Table 5.6 Set of actions for all system conditions

System Condition	Action 1	Action 2	Action 3
85% *U* and *k*=3	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
85% *U* and *k*=4	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
85% *U* and *k*=5	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
85% *U* and *k*=6	*COVERT*	*PT/TIS*	*EDD*
90% *U* and *k*=3	*COVERT*	*PT/TIS*	*SPT*
90% *U* and *k*=4	*COVERT*	*PT/TIS*	*SPT*
90% *U* and *k*=5	*COVERT*	*PT/TIS*	*SPT*
90% *U* and *k*=6	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
95% *U* and *k*=3	*COVERT*	*PT/TIS*	*SPT*
95% *U* and *k*=4	*COVERT*	*PT/TIS*	*SPT*
95% *U* and *k*=5	*COVERT*	*PT/TIS*	*SPT*
95% *U* and *k*=6	*COVERT*	*PT/TIS*	*PT+WINQ+SL*

5.6.1.2 State Determination Criteria

In this study, average slack time of the jobs in the buffer (*ASL*) is considered as state determination criteria. Ten *ASL* intervals are evaluated to define the state action table. The ranges of the states are determined through trial and error experiments according to the *ASL* value of the jobs. Dummy states and the applied method in these states described in Section 4.8.1.2 are also taken into account in this experiment. The definition of *ASL* is given as follows:

$$ASL = \frac{\sum_{i=1}^{n} d_i - \tau - RPT_i}{n} \tag{5.10}$$

The *RPT* is the remaining processing time of the job *i*, *n* is the number of jobs in the buffer and τ is the current time. There are twelve different system conditions (three levels of utilization and four levels of due date settings). Thus, twelve different state action tables are defined. The state action table for the condition of 90% utilization level and k=3 is presented in Table 5.7. The state action tables for the other conditions are given in Appendix D.

Table 5.7 State action table for 90% utilization level and k=3

State	Determination Criteria	*COVERT*	*PT/TIS*	*SPT*
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ASL <$-325	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and -325 $\leq ASL <$ -100	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and -100 $\leq ASL < 0$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $0 \leq ASL < 100$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $100 \leq ASL < 200$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $200 \leq ASL < 300$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $300 \leq ASL < 375$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $375 \leq ASL < 450$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $450 \leq ASL < 525$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $525 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

5.6.1.3 Determination of Reward Function

In this experiment, the reward function aims at minimizing mean tardiness. Thus, it is constructed based on the tardiness level of the job. The agent receives a reward depending on the tardiness level of the job after the execution of the action. For a single machine system, the decision on whether the job is tardy or not can be made only when the job is released from the resource. However, for a flow shop, it is not definite that the job is tardy when the job is released from a machine since the route of the job contains more than one operation. Therefore, reward function is constructed based on the estimation of the tardiness level of a job when the job is

released from the last machine. The estimation of the completion time of job i is given as follows:

$$C_{ij} = t + \sum_{k=j+1}^{m} q_{ik} + \sum_{k=j+1}^{m} p_{ik} \tag{5.11}$$

The C is the estimated completion of job i when it released from machine j, m is the total number of machines, q is the estimated waiting time of job i in the buffer of machine j, p is the processing time of job i on machine j and t is the current time. Tardiness of the job is calculated based on Equation 4.2. It must be noted that C is taken into account as completion time. The reward function for 90% utilization level and k=3 constructed respect to these assumptions is presented in Table 5.8. The reward functions for the rest of the conditions are illustrated in Appendix E.

Table 5.8 Reward function for 90% utilization level and k=3

Condition	Reward
$Tardiness \leq 0$	1
$0 < Tardiness \leq 30$	-1
$30 < Tardiness \leq 60$	-2
$60 < Tardiness \leq 100$	-3
$100 < Tardiness \leq 170$	-4
$170 < Tardiness \leq 220$	-5
$220 < Tardiness \leq 260$	-6
$260 < Tardiness \leq 320$	-7
$320 < Tardiness \leq 400$	-8
$400 < Tardiness$	-9

5.6.1.4 Simulation and Results

The Q-learning agent is integrated into the dynamic flow shop simulation model to investigate if the agent can learn to select the best dispatching rule, determined in first phase, for all system conditions. The parameter settings which are defined in the first phase is used for all simulation experiments and all the values of state-action pairs, $Q(s,a)$, are initialized to zero since all the actions for all states have equal chance to be selected. The Q-learning algorithm is applied with the setting of $\varepsilon = 0.1$

for exploration of the actions. The step size parameter, α, is set to 0.1 and the value of the discount rate parameter, γ, is determined as 0.9.

The results are presented in Table 5.9 and Table 5.10. The *COVERT* rule is defined as the best dispatching rule for the case of minimizing mean tardiness except one system condition in the first phase. The number of times that each rule is selected during the simulation run is computed and then selection percentages of these rules are calculated. The agent selects the *COVERT* dispatching rule for most of the run time in all system conditions as shown in Table 5.9.

Table 5.9 Selecting percentages of actions for all system conditions

System Condition	Selecting Percentages of Dispatching Rules (Actions)		
	Action 1	Action 2	Action 3
85% *U* and *k*=3	0.85	0.11	0.04
85% *U* and *k*=4	0.84	0.10	0.06
85% *U* and *k*=5	0.79	0.18	0.03
85% *U* and *k*=6	0.25	0.69	0.06
90% *U* and *k*=3	0.83	0.13	0.04
90% *U* and *k*=4	0.86	0.08	0.06
90% *U* and *k*=5	0.88	0.07	0.05
90% *U* and *k*=6	0.92	0.05	0.03
95% *U* and *k*=3	0.91	0.06	0.03
95% *U* and *k*=4	0.83	0.12	0.05
95% *U* and *k*=5	0.83	0.13	0.04
95% *U* and *k*=6	0.86	0.10	0.04

Table 5.10 Individual results for all system conditions

System Condition	Individual Results			
	Action 1	Action 2	Action 3	*Q*-Agent
85% *U* and *k*=3	52.37	60.64	77.37	56.21
85% *U* and *k*=4	15.56	25.06	30.9	18.64
85% *U* and *k*=5	3.27	3.99	8.85	4.85
85% *U* and *k*=6	1.23	1.09	2.40	1.56
90% *U* and *k*=3	185.14	190.11	250.24	192.86
90% *U* and *k*=4	79.83	83.51	120.04	84.44
90% *U* and *k*=5	31.1	37.88	50.7	34.78
90% *U* and *k*=6	13.99	20.21	22.31	15.23
95% *U* and *k*=3	676.06	741.90	780.81	685.95
95% *U* and *k*=4	489.79	515.71	593.01	499.84
95% *U* and *k*=5	334.01	345.53	490.99	342.71
95% *U* and *k*=6	214.27	225.58	330.69	221.45

The agent selects the *PT/TIS* more than *COVERT* rule only for the case where the system utilization level is 85% and k=6, since *PT/TIS* is better than the *COVERT* for minimizing mean tardiness in this system condition according to the individual results.

The results indicate that the *Q*-learning agent is able to learn to select the proper dispatching rule for each of the system conditions when the system objective is to minimize mean tardiness on dynamic flow shop system.

5.6.2 Minimizing Mean Flow Time

In this section, learning potential of the agent is investigated when the objective is to minimize number of tardy jobs. The *SPT* outperforms the other dispatching rules for minimizing mean flow time (*MF*) except for two cases according to the results of the first phase. The *PT+WINQ+SL* is better than *SPT* under the conditions of 85% utilization level and loose due date settings of k=5 and k=6. Thus, it is expected that the agent learns to select the *SPT* under most of the system conditions and select the *PT+WINQ+SL* rule for the specific conditions.

5.6.2.1 Determination of States and Actions

In this experiment, as it was designated in Section 5.6.1.1, the best three dispatching rules for minimizing mean flow time under all system conditions, *COVERT*, *PT/TIS* and *SPT,* are determined as the action set of the *Q*-learning agent. State determination tables presented in Section 5.6.1.2 are also used in this experiment. The set of actions, A, is defined as follows;

$A = \{a_1, a_2, a_3\}$

where,

a_1 = execute the *COVERT*

a_2 = execute the *PT/TIS* (execute the *PT+WINQ+SL* for %85U and k=5, 6)

a_3 = execute the *SPT*

5.6.2.2 Determination of Reward Function

In this research, the reward function aims minimizing mean flow time. Thus, it is constructed based on the time that jobs spent in the system. The completion times of the jobs are estimated according to Equation 5.11 and flow time of the jobs are calculated respect to Equation 5.1. The reward function for 90% utilization level and k=3 is presented in Table 5.11. The reward functions for the other the conditions are given in Appendix F.

Table 5.11 Reward function for 90% utilization level and k=3

Condition	**Reward**
Flow Time ≤ 400	1
400 < *Flow Time* ≤ 480	-1
480 < *Flow Time* ≤ 530	-2
530 < *Flow Time* ≤ 620	-3
620 < *Flow Time* ≤ 700	-4
700 < *Flow Time* ≤ 800	-5
800 < *Flow Time* ≤ 900	-6
900 < *Flow Time* ≤ 1000	-7
1000 < *Flow Time* ≤1200	-8
1200 < *Flow Time*	-9

5.6.2.3 Simulation and Results

The *Q*-learning agent is inserted into the dynamic flow shop simulation model to determine if the agent can learn to select the best dispatching rule stated in first phase, for all system conditions. The parameter settings that defined in Section 5.6.1.4 are used for all simulation experiments. The results are presented in Table 5.12 and Table 5.13.

Table 5.12 Selecting percentages of actions for all system conditions

System Condition	Selecting Percentages of Dispatching Rules (Actions)		
	COVERT	***PT/TIS* or *PT+WINQ+SL***	***SPT***
85% *U* and *k*=3	0.04	0.10	0.86
85% *U* and *k*=4	0.04	0.10	0.86
85% *U* and *k*=5	0.08	0.68	0.24
85% *U* and *k*=6	0.07	0.72	0.21
90% *U* and *k*=3	0.05	0.03	0.92
90% *U* and *k*=4	0.06	0.03	0.91
90% *U* and *k*=5	0.07	0.04	0.89
90% *U* and *k*=6	0.09	0.03	0.88
95% *U* and *k*=3	0.06	0.03	0.91
95% *U* and *k*=4	0.04	0.05	0.91
95% *U* and *k*=5	0.04	0.06	0.90
95% *U* and *k*=6	0.04	0.06	0.90

The *SPT* is defined as the best dispatching rule for minimizing mean flow time except for two cases in the first phase. The agent selects the *SPT* for most of the run time in all system conditions except these specific ones as shown in Table 5.12 and selects the *PT+WINQ+SL* rule under conditions of 85% utilization level and *k*=5, 6.

The results demonstrate that the *Q*-learning agent is able to learn to select the proper dispatching rule for each of the system conditions when the system objective is minimizing mean flow time.

Table 5.13 Individual results for all system conditions

System Condition	Individual Results			
	COVERT	***PT/TIS* or *PT+WINQ+SL***	***SPT***	***Q*-Agent**
85% *U* and *k*=3	777.91	744.93	706.16	720.66
85% *U* and *k*=4	772.87	744.93	706.16	719.33
85% *U* and *k*=5	724.25	703.42	706.16	712.02
85% *U* and *k*=6	721.56	696.83	706.16	710.23
90% *U* and *k*=3	992.55	998.9	898.27	912.73
90% *U* and *k*=4	985.86	998.9	898.27	910.26
90% *U* and *k*=5	960.23	998.9	898.27	908.26
90% *U* and *k*=6	956.96	998.9	898.27	907.91
95% *U* and *k*=3	1558.27	1613.99	1416.45	1442.89
95% *U* and *k*=4	1613.01	1613.99	1416.45	1445.31
95% *U* and *k*=5	1632.79	1613.99	1416.45	1448.26
95% *U* and *k*=6	1621.27	1613.99	1416.45	1447.04

5.7 Third Phase: Application of *Q-learning* Algorithm to Bicriteria Problem

In this phase, *Q*-learning agent is trained to minimize mean tardiness and mean flow time on a dynamic flow shop bicriteria scheduling problem. The best rules for each measure of performance for each system condition according to results of first phase are included in training process of the agent. The *Q*-learning agent is trained to learn minimizing which performance measure in which system state.

5.7.1 Determination of States and Actions

Although the *PT/TIS* outperforms the *COVERT* under the condition of 85% utilization level and k=6, the *COVERT* dispatching rule is preferred as the action for minimizing mean tardiness because of its well performance in minimizing mean flow time besides its close performance to *PT/TIS*. On the other hand, the *SPT* rule is considered as the action for minimizing mean flow time except for two cases where the utilization level is 85% and k=5, 6. In these conditions, *PT+WINQ+SL* rule is determined as the action for minimizing mean flow time since it is better then *SPT* according to individual results. The set of actions, A, is defined as follows;

$A = \{a_1, a_2\}$

where,

a_1 = minimize the mean tardiness (execute the *COVERT*)

a_2 = minimize the mean flow time (execute the *SPT* or *PT+WINQ+SL*)

The state determination tables presented in Section 5.6.1.2 are also used in this experiment.

5.7.2 Determination of Reward Function

In this phase, concurrent minimization of mean tardiness and mean flow time is considered. Thus, two separate reward functions are developed for each action. The

reward function table for 90% utilization level and k=3 is presented in Table 5.14. The tables for other conditions are shown in Appendix G.

Table 5.14 Reward function table

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	*Flow Time* ≤ 400	1
0 < *Tardiness* ≤ 25	-1	400 < *Flow Time* ≤ 480	-1
25 < *Tardiness* ≤ 50	-2	480 < *Flow Time* ≤ 530	-2
50 < *Tardiness* ≤ 70	-3	530 < *Flow Time* ≤ 620	-3
70 < *Tardiness* ≤ 100	-4	620 < *Flow Time* ≤ 700	-4
100 < *Tardiness* ≤ 140	-5	700 < *Flow Time* ≤ 800	-5
140 < *Tardiness* ≤ 210	-6	800 < *Flow Time* ≤ 900	-6
210 < *Tardiness* ≤ 335	-7	900 < *Flow Time* ≤ 1000	-7
335 < *Tardiness* ≤ 600	-8	1000 < *Flow Time* ≤1200	-8
600 < *Tardiness*	-9	1200 < *Flow Time*	-9

The *Q*-learning algorithm was used with a single reward function in the first phase. A dual reward function is used in the third phase. If the *COVERT* rule is selected as an action, the agent receives the reward from the first reward function since the objective is to minimize mean tardiness. Otherwise, the agent receives the reward from the second function when the selected action is *SPT* (or *PT+WINQ+SL*).

5.7.3 Simulation and Results

The *Q*-learning agent is inserted to a dynamic flow shop simulation model to determine minimizing which objective is more effective in which system state. The parameter settings that defined in Section 5.6.1.4 are used for all simulation experiments. Tables 5.15 to 5.18 present the selection percentages of the dispatching rules under all system conditions.

Table 5.15 Selection percentages of dispatching rules for k=3

Dispatching Rule	**85% *U***	**90% *U***	**95% *U***
COVERT	0.54	0.53	0.52
SPT	0.46	0.47	0.48

Table 5.16 Selection percentages of dispatching rules for k=4

Dispatching Rule	85% *U*	90% *U*	95% *U*
COVERT	0.57	0.56	0.53
SPT	0.43	0.44	0.47

Table 5.17 Selection percentages of dispatching rules for k=5

Dispatching Rule	85% *U*	Dispatching Rule	90% *U*	95% *U*
COVERT	0.60	***COVERT***	0.57	0.55
PT+WINQ+SL	0.40	***SPT***	0.43	0.45

Table 5.18 Selection percentages of dispatching rules for k=6

Dispatching Rule	85% *U*	Dispatching Rule	90% *U*	95% *U*
COVERT	0.63	***COVERT***	0.61	0.60
PT+WINQ+SL	0.37	***SPT***	0.39	0.40

Table 5.19 Selection percentages of dispatching rules in each state for k=3

State	85% *U*		90% *U*		95% *U*	
	COVERT	***SPT***	***COVERT***	***SPT***	***COVERT***	***SPT***
1	0.27	0.73	0.25	0.75	0.20	0.80
2	0.30	0.70	0.27	0.73	0.22	0.78
3	0.28	0.72	0.30	0.70	0.25	0.75
4	0.35	0.65	0.32	0.68	0.30	0.70
5	0.48	0.52	0.40	0.60	0.35	0.65
6	0.63	0.37	0.55	0.45	0.52	0.48
7	0.68	0.32	0.65	0.35	0.67	0.33
8	0.73	0.27	0.74	0.26	0.78	0.22
9	0.77	0.23	0.78	0.22	0.80	0.20
10	0.76	0.24	0.79	0.21	0.83	0.17

Table 5.20 Selection percentages of dispatching rules in each state for k=4

State	85% *U*		90% *U*		95% *U*	
	COVERT	***SPT***	***COVERT***	***SPT***	***COVERT***	***SPT***
1	0.22	0.78	0.20	0.80	0.17	0.83
2	0.25	0.75	0.23	0.77	0.18	0.82
3	0.27	0.73	0.27	0.73	0.25	0.75
4	0.32	0.68	0.30	0.70	0.28	0.72
5	0.45	0.55	0.43	0.57	0.35	0.65
6	0.60	0.40	0.65	0.35	0.60	0.40
7	0.68	0.32	0.72	0.28	0.70	0.30
8	0.75	0.25	0.76	0.24	0.75	0.25
9	0.80	0.20	0.78	0.22	0.78	0.22
10	0.85	0.15	0.82	0.18	0.80	0.20

In addition to these tables, selection percentages of the dispatching rules in each state are shown in Table 5.19 to 5.22

Table 5.21 Selection percentages of dispatching rules in each state for k=5

State	85% *U*		90% *U*		95% *U*	
	COVERT	***PT+WINQ+SL***	***COVERT***	***SPT***	***COVERT***	***SPT***
1	0.20	0.80	0.18	0.82	0.17	0.83
2	0.28	0.72	0.25	0.75	0.20	0.80
3	0.35	0.65	0.30	0.70	0.27	0.73
4	0.39	0.61	0.37	0.63	0.40	0.60
5	0.48	0.52	0.45	0.55	0.43	0.57
6	0.69	0.31	0.65	0.35	0.60	0.40
7	0.78	0.22	0.74	0.26	0.70	0.30
8	0.83	0.17	0.80	0.20	0.78	0.22
9	0.88	0.12	0.86	0.14	0.80	0.20
10	0.90	0.10	0.88	0.12	0.85	0.15

Table 5.22 Selection percentages of dispatching rules in each state for k=6

State	85% *U*		90% *U*		95% *U*	
	COVERT	***PT+WINQ+SL***	***COVERT***	***SPT***	***COVERT***	***SPT***
1	0.25	0.75	0.20	0.80	0.17	0.83
2	0.32	0.68	0.28	0.72	0.25	0.75
3	0.39	0.61	0.34	0.66	0.32	0.68
4	0.45	0.55	0.42	0.58	0.40	0.60
5	0.71	0.29	0.70	0.30	0.63	0.37
6	0.78	0.22	0.75	0.25	0.72	0.28
7	0.83	0.17	0.78	0.22	0.79	0.21
8	0.85	0.15	0.84	0.16	0.80	0.20
9	0.89	0.11	0.88	0.12	0.85	0.15
10	0.91	0.09	0.90	0.10	0.88	0.12

The agent prefers to select *SPT* (or *PT+WINQ+SL*) on system states of 1 to 5 except for one condition where the allowance factor is 6. The agent selects that action on system states of 1 to 4, that is, minimizing mean flow time on these states is more effective and brings in more reward. On the other hand, *COVERT* dispatching rule is selected on the states of 5 to 10 generally. The changes of the percentages are clearly seen in Figures 5.1 to 5.12.

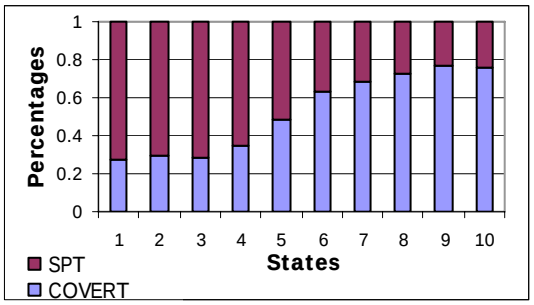

Figure 5.1 Selection percentages for 85% and *k*=3

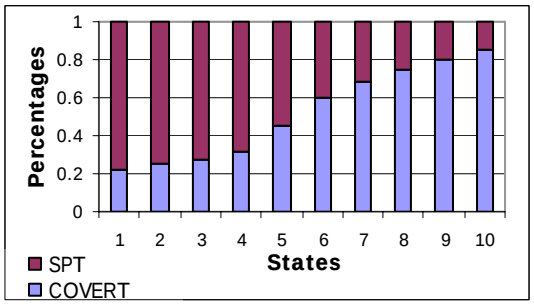

Figure 5.2 Selection percentages for 85% and *k*=4

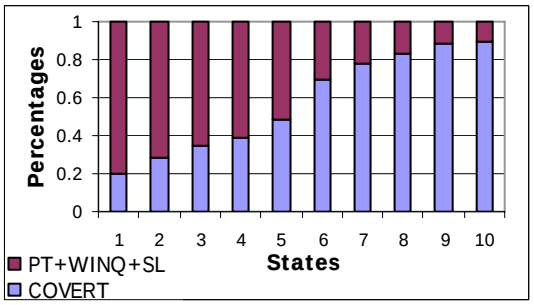

Figure 5.3 Selection percentages for 85% and *k* =5

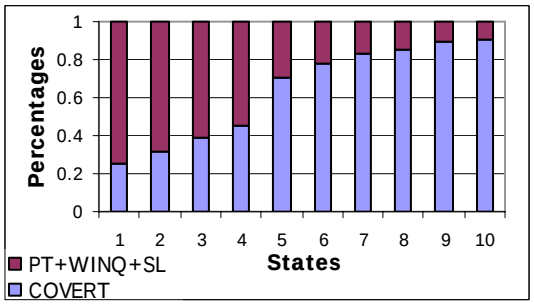

Figure 5.4 Selection percentages for 85% and *k* =6

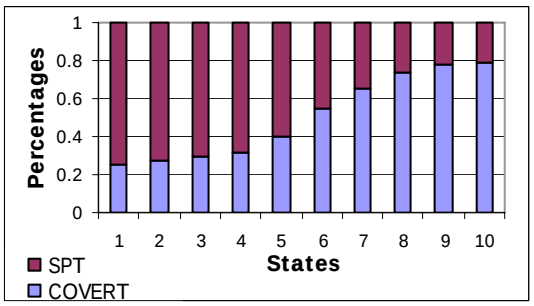

Figure 5.5 Selection percentages for 90% and *k* =3

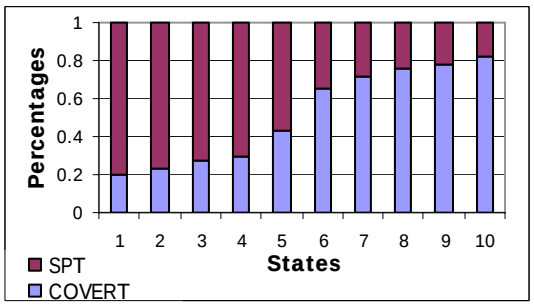

Figure 5.6 Selection percentages for 90% and *k* =4

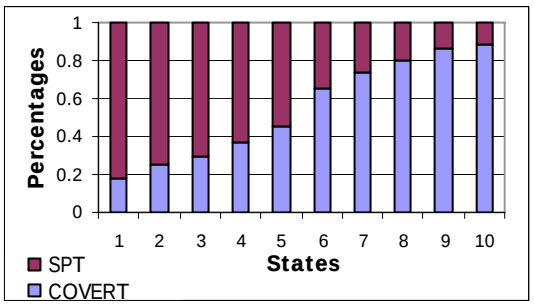

Figure 5.7 Selection percentages for 90% and *k* =5

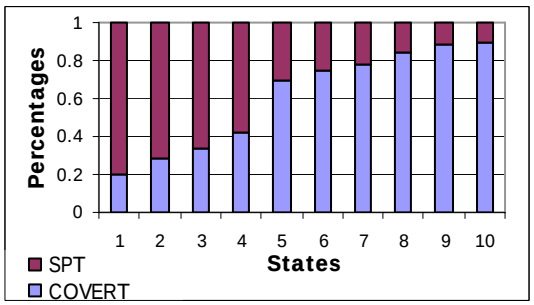

Figure 5.8 Selection percentages for 90% and *k* =6

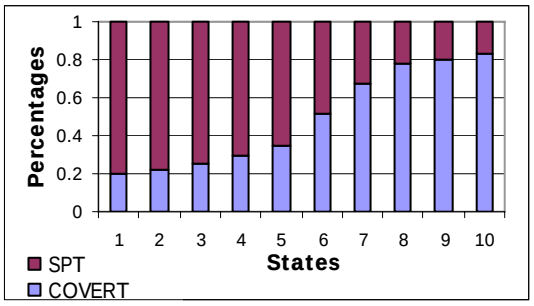

Figure 5.9 Selection percentages for 95% and k =3

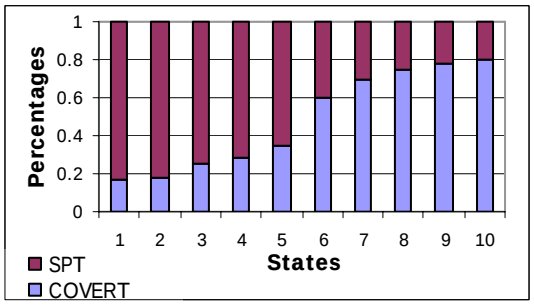

Figure 5.10 Selection percentages for 95% and k =4

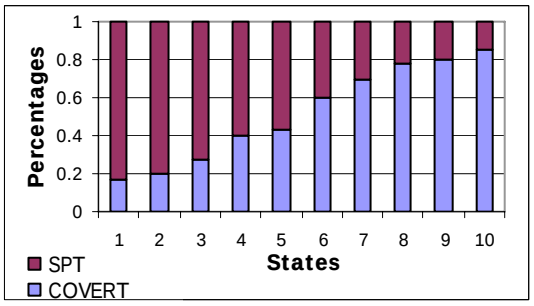

Figure 5.11 Selection percentages for 95% and k =5

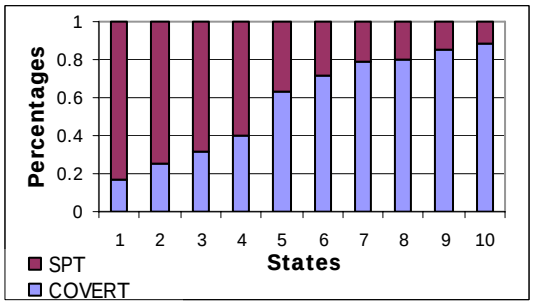

Figure 5.12 Selection percentages for 95% and k =6

It is obviously seen that the dominance of the selected dispatching rule is more distinct on heavy utilization levels. For instance, selection percentage of *SPT* is greater then *COVERT*'s on states 1 to 4 under the system condition of 95% utilization level and k=6. Likewise, *COVERT* is selected more than on states 5 to 10 under the same system condition. It is also observed that the agent minimizes the mean tardiness when the *ASL* is high since probability of being tardy of a job decreases when the job has more slack time for its due date. Therefore, the agent receives more reward if it executes *COVERT* because reward function gives more reward if the job is not tardy or the tardiness level of the job is low. On the other hand, it is effective to minimize mean flow time when the *ASL* is low since the tardiness level of the jobs increases and the agent begins to receive negative reward if it selects the *COVERT* rule. Thus, the agent executes the *SPT* rule (or *PT+WINQ+SL*) on the low level of *ASL*. The performances of *Q* agent for all system conditions and individual results of other dispatching rules are given in Table 5.23 to 5.26.

Table 5.23 Performance of the *Q* agent and individual results for *k* =3

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	148.21	913.43	442.86	1293.30	1473.12	2368.02
AT-RPT	155.42	920.8	449.85	1299.63	1477.99	2372.80
EDD	120.85	887.72	412.33	1266.98	1446.40	2342.34
S/OPN	124.30	894.66	417.10	1273.14	1452.54	2348.61
COVERT	52.37	777.91	185.14	992.55	676.06	1558.27
SPT	92.87	706.16	250.24	898.27	780.81	1416.45
PT/TIS	60.64	744.93	190.11	998.90	741.90	1613.99
PT+WINQ+AT	123.00	869.53	394.24	1234.09	1397.86	2291.11
PT+WINQ+SL	77.37	787.74	297.57	1091.99	1291.02	2176.31
***Q*-Agent**	61.01	738.56	197.23	920.54	702.65	1441.32
Test 1	72.54	746.77	212.43	951.33	727.83	1482.76
Test 2	73.87	748.23	214.49	954.81	729.66	1486.13
Test 3	70.22	744.55	211.26	948.67	724.98	1478.88
Test 4	66.32	743.28	207.87	939.19	716.34	1469.65

Table 5.24 Performance of the *Q* agent and individual results for *k* =4

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	58.53	913.43	257.94	1293.30	1192.64	2368.02
AT-RPT	62.51	920.8	263.73	1299.63	1197.44	2372.80
EDD	37.19	881.25	221.03	1259.01	1153.56	2332.29
S/OPN	38.55	887.61	224.53	1265.84	1161.46	2340.72
COVERT	15.56	772.87	79.83	985.86	489.79	1613.01
SPT	34.01	706.16	120.04	898.90	593.01	1416.45
PT/TIS	25.06	744.93	83.51	998.90	515.71	1613.99
PT+WINQ+AT	46.16	869.53	223.21	1234.09	1121.61	2291.11
PT+WINQ+SL	30.90	782.09	130.77	1008.77	928.51	2053.42
***Q*-Agent**	22.23	736.56	90.54	918.35	524.65	1445.75
Test 1	26.32	747.65	102.64	945.21	545.64	1512.73
Test 2	27.39	749.31	104.39	948.82	549.91	1519.32
Test 3	25.43	745.87	100.11	941.32	539.43	1502.44
Test 4	24.98	742.19	96.24	932.15	531.66	1483.72

Table 5.25 Performance of the *Q* agent and individual results for *k* =5

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	21.99	913.43	142.07	1293.3	939.33	2368.02
AT-RPT	23.91	920.8	146.23	1299.63	943.95	2372.8
EDD	9.82	875.99	107.2	1251.68	889.45	2323.67
S/OPN	10.17	881.45	109.03	1257.89	895.36	2331.10
COVERT	3.27	728.25	31.10	960.23	334.01	1632.79
SPT	43.15	706.16	50.70	898.9	490.99	1416.45
PT/TIS	3.99	744.93	37.88	998.9	345.53	1613.98
PT+WINQ+AT	16.71	869.53	119.98	1234.09	875.45	2291.11
PT+WINQ+SL	8.85	703.42	43.02	966.4	633.01	1925.20
***Q*-Agent**	5.29	715.42	36.75	917.11	370.87	1490.67
Test 1	6.12	717.22	41.23	932.12	412.35	1532.11
Test 2	6.19	718.11	41.99	934.19	416.67	1540.76
Test 3	5.92	716.64	40.12	928.72	402.88	1523.83
Test 4	5.65	716.35	38.24	926.55	389.91	1515.95

Table 5.26 Performance of the *Q* agent and individual results for *k*=6

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	8.34	913.43	75.72	1293.30	722.16	2368.02
AT-RPT	9.17	920.80	78.41	1299.63	726.42	2372.80
EDD	2.40	870.22	48.14	1244.41	662.60	2314.37
S/OPN	2.56	875.97	48.92	1250.12	666.02	2320.35
COVERT	1.23	721.56	13.99	956.96	214.27	1621.27
SPT	31.85	706.16	25.40	898.90	365.77	1416.45
PT/TIS	1.09	744.93	20.21	998.90	225.58	1613.98
PT+WINQ+AT	6.20	869.53	60.67	1234.09	667.28	2291.11
PT+WINQ+SL	5.45	696.83	22.31	957.88	330.69	1801.84
***Q*-Agent**	2.51	712.11	18.65	916.99	241.47	1488.32
Test 1	3.28	714.33	19.58	930.27	285.83	1528.88
Test 2	3.36	714.99	19.92	933.74	289.91	1537.28
Test 3	3.21	713.32	19.37	927.49	278.62	1520.92
Test 4	3.02	712.89	18.83	922.86	268.16	1511.87

The test methods investigated in chapter four are also considered in this experiment. It is observed that *Q*-agent outperforms these three test methods although the selection percentages of dispatching rules are equal for the agent and test methods. Also, Test 4 performed well and outperformed other test methods and most of the dispatching rules. The performance of *Q*-learning agent is between *COVERT* and *SPT* as expected. The agent is better than the *SPT* when the mean tardiness is considered but *SPT* is better than the *Q*-learning agent for minimizing mean flow

time. On the other hand, the agent outperforms the *COVERT* in case of mean flow time but *COVERT* emerges to the best for mean tardiness. The results are normalized and evaluated to investigate the performance of the *Q*-learning agent under all conditions in following section.

5.7.4 Performance Analysis of Q-learning Agent

Proposed ranking method used in Section 4.9.4 is also used in this section. Simulation results for 90% utilization level and k=3 are presented in Table 5.27 and Figure 5.13. The result tables for othr conditions are given in Appendix K.1. Importance of the both objectives is equal since weights vary between 0 and 1 ($\overline{w}_1 = \overline{w}_2$).

Table 5.27 Ranking results for 90% utilization level and k=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.37443	4.37652	4.37860	0.00209
SPT	4.92747	4.92963	4.93179	0.00216
PT/TIS	5.82895	5.83095	5.83294	0.00200
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.54015	1.54075	1.54135	0.00060
Test 1	5.39698	5.39742	5.39786	0.00044
Test 2	6.50734	6.50774	6.50814	0.00040
Test 3	4.32393	4.32444	4.32495	0.00051
Test 4	3.09183	3.09255	3.09327	0.00072

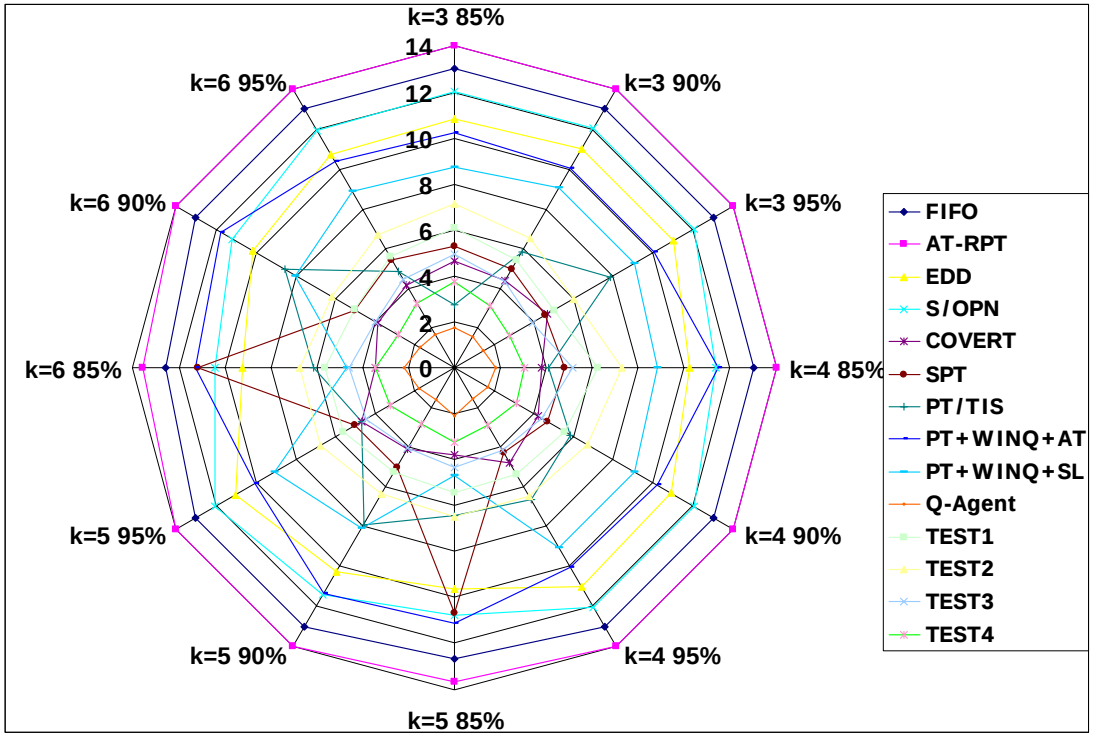

Figure 5.13 Ranks of all methods for all conditions

The results show that the *Q*-agent emerges to the best when the weight independent ranks are considered. Test 4 also performed well under all conditions. The ranks of the first three methods (well performed) for all weights generated for the condition of 90% *U* and *k*=3 are presented in Figure 5.14. The figures representing the ranks of the methods for the other conditions are illustrated in Appendix M.2. The w_1 is the weight of mean tardiness.

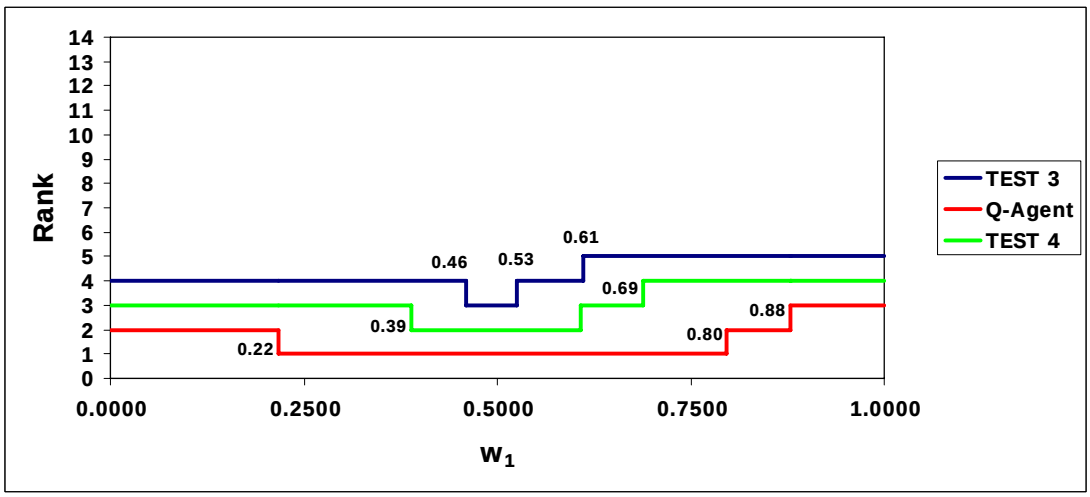

Figure 5.14 Ranks of the first three methods with respect to w_1

The results when $\overline{w}_1 > \overline{w}_2$ (w_1 is the weight of mean tardiness, w_2 is the weight mean flow time) for 90% utilization level and k=3 are presented in Table 5.28 and Figure 5.15.

Table 5.28 Ranking results for 90% U and k=3 when $\overline{w}_1 > \overline{w}_2$

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	1.85916	1.85995	1.86075	0.00080
SPT	7.92939	7.92970	7.93000	0.00030
PT/TIS	3.66017	3.66169	3.66322	0.00152
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.64791	1.64865	1.64940	0.00074
Test 1	5.83112	5.83148	5.83185	0.00036
Test 2	6.93302	6.93321	6.93341	0.00020
Test 3	4.72778	4.72831	4.72884	0.00053
Test 4	3.40625	3.40700	3.40775	0.00075

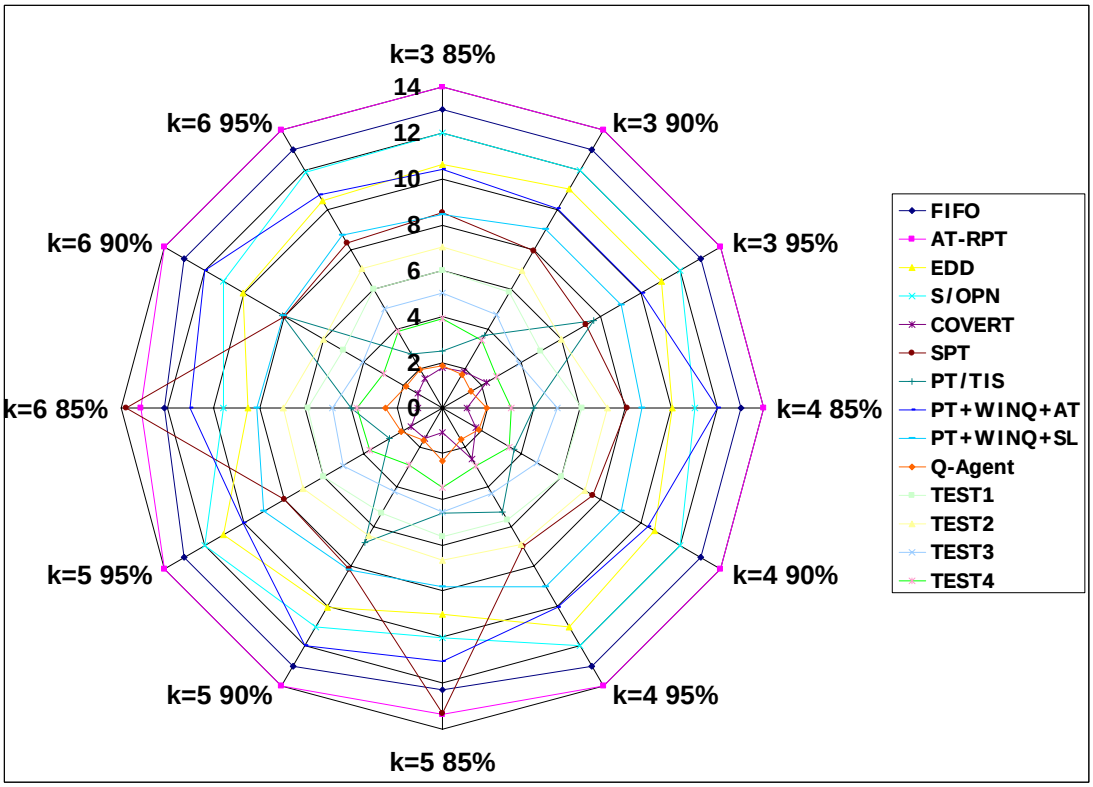

Figure 5.15 Ranks of all methods for all conditions when $\overline{w}_1 > \overline{w}_2$

The results when $\overline{w}_1 < \overline{w}_2$ for 90% utilization level and k=3 are presented in Table 5.29 and Figure 5.16.

Table 5.29 Ranking results for 90% utilization level and k=3 when $\overline{w}_1 < \overline{w}_2$

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	6.89248	6.89284	6.89319	0.00036
SPT	1.92887	1.92965	1.93042	0.00077
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.43326	1.43363	1.43399	0.00037
Test 1	4.96303	4.96320	4.96336	0.00017
Test 2	6.08206	6.08233	6.08261	0.00028
Test 3	3.92013	3.92041	3.92068	0.00027
Test 4	2.77766	2.77795	2.77825	0.00030

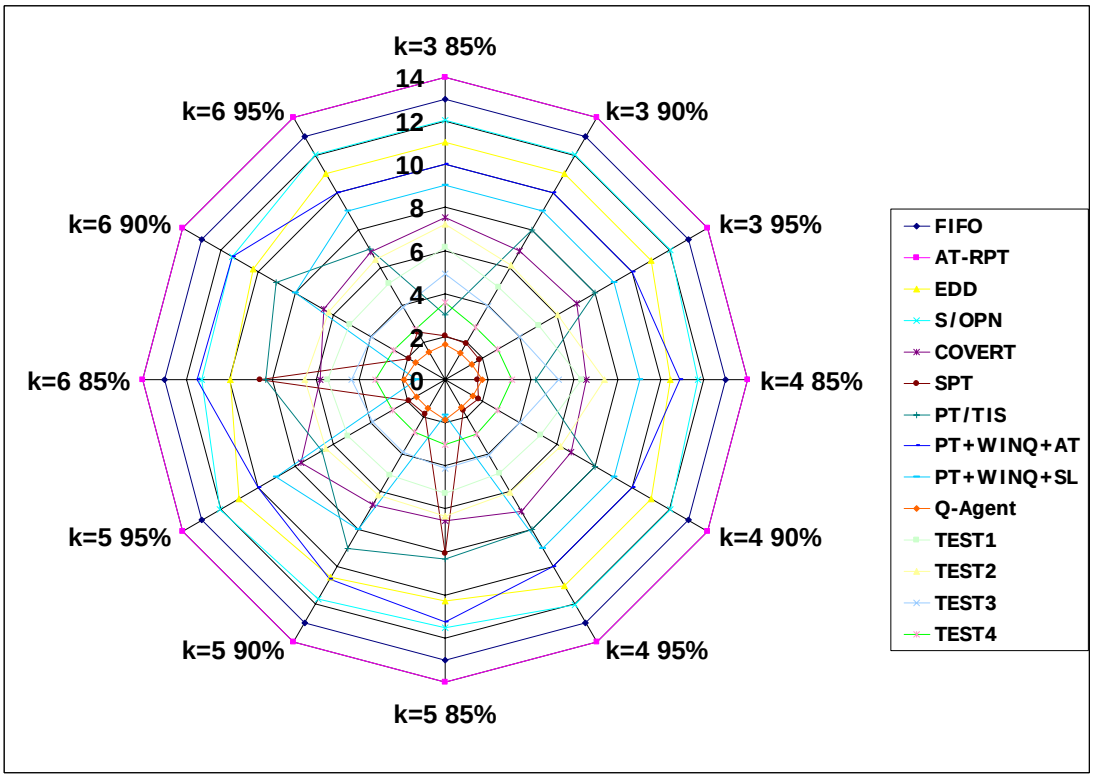

Figure 5.16 Ranks of all methods for all conditions when $\overline{w}_1 < \overline{w}_2$

When the $\overline{w}_1 > \overline{w}_2$ for the decision maker, *Q* agent has the best performance only for three cases where the system conditions are 90% *U* and *k*=3, 95% *U* and *k*=3, 95% *U* and *k*=4. *COVERT* dispatching rule outperforms the agent under the rest of the conditions since the *COVERT* performed well for mean flow time as well as for mean tardiness in individual results.

When $\overline{w}_1 < \overline{w}_2$ for the decision maker, *Q* agent outperforms all other methods including *SPT* since the *SPT* has a poor performance for mean tardiness although it performed well for mean flow time in individual results. *SPT* is better than the agent only for one condition, k=4 and 85 %, but it is not a significant difference. It is seen that the *Q* agent placed at middle of the graphic in Figure 5.16. The results for these two cases under other conditions are presented in Appendix K.2 and Appendix K.3.

5.8 Application of *Q*-learning Algorithm with Multi State Determination Criteria

In this section, the state table is constructed based on a new determination criterion. In the Section 5.7.1, average slack time (*ASL*) of the jobs in the buffer is considered as the determination criteria. However, the number of jobs in the buffer (*N*) may effects the results since tardiness and waiting time increase if the number of jobs is high. On the other hand, two states with the same *ASL* value may be unconcern from each other if the numbers of jobs in the queues are different in these states.

The number of jobs (*N*) in the buffer is considered as determination criteria with the average slack time of the jobs in the buffer. In the Section 5.7.1, there were 10 different states in the system. In this research, 4 intervals are also evaluated to define the levels of number of tardy jobs in the buffer (*N*). The ranges of these levels are determined through trial and error experiments depend on balancing the number of visits to each state. Thus, there are 40 different states in this experiment. The state table for 90% utilization level and *k*=3 is presented in Table 5.30. The state tables for the other conditions are given in Appendix H.

Table 5.30 State table for 90% utilization level and k=3

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and $ASL < 0$	$N = 2$
2	$N > 1$ and $ASL < 0$	$N = 3$
3	$N > 1$ and $ASL < 0$	$N = 4$
4	$N > 1$ and $ASL < 0$	$N > 4$
5	$N > 1$ and $0 \leq ASL < 100$	$N = 2$
6	$N > 1$ and $0 \leq ASL < 100$	$N = 3$
7	$N > 1$ and $0 \leq ASL < 100$	$N = 4$
8	$N > 1$ and $0 \leq ASL < 100$	$N > 4$
9	$N > 1$ and $100 \leq ASL < 200$	$N = 2$
10	$N > 1$ and $100 \leq ASL < 200$	$N = 3$
11	$N > 1$ and $100 \leq ASL < 200$	$N = 4$
12	$N > 1$ and $100 \leq ASL < 200$	$N > 4$
13	$N > 1$ and $200 \leq ASL < 300$	$N = 2$
14	$N > 1$ and $200 \leq ASL < 300$	$N = 3$
15	$N > 1$ and $200 \leq ASL < 300$	$N = 4$
16	$N > 1$ and $200 \leq ASL < 300$	$N > 4$
17	$N > 1$ and $300 \leq ASL < 350$	$N = 2$
18	$N > 1$ and $300 \leq ASL < 350$	$N = 3$
19	$N > 1$ and $300 \leq ASL < 350$	$N = 4$
20	$N > 1$ and $300 \leq ASL < 350$	$N > 4$
21	$N > 1$ and $350 \leq ASL < 400$	$N = 2$
22	$N > 1$ and $350 \leq ASL < 400$	$N = 3$
23	$N > 1$ and $350 \leq ASL < 400$	$N = 4$
24	$N > 1$ and $350 \leq ASL < 400$	$N > 4$
25	$N > 1$ and $400 \leq ASL < 450$	$N = 2$
26	$N > 1$ and $400 \leq ASL < 450$	$N = 3$
27	$N > 1$ and $400 \leq ASL < 450$	$N = 4$
28	$N > 1$ and $400 \leq ASL < 450$	$N > 4$
29	$N > 1$ and $450 \leq ASL < 500$	$N = 2$
30	$N > 1$ and $450 \leq ASL < 500$	$N = 3$
31	$N > 1$ and $450 \leq ASL < 500$	$N = 4$
32	$N > 1$ and $450 \leq ASL < 500$	$N > 4$
33	$N > 1$ and $500 \leq ASL < 560$	$N = 2$
34	$N > 1$ and $500 \leq ASL < 560$	$N = 3$
35	$N > 1$ and $500 \leq ASL < 560$	$N = 4$
36	$N > 1$ and $500 \leq ASL < 560$	$N > 4$
37	$N > 1$ and $560 \leq ASL$	$N = 2$
38	$N > 1$ and $560 \leq ASL$	$N = 3$
39	$N > 1$ and $560 \leq ASL$	$N = 4$
40	$N > 1$ and $560 \leq ASL$	$N > 4$

The reward function, action sets and the parameter settings determined in Section 5.7 are used for this experiment. Tables 5.31 to 5.34 present the selection percentages of the dispatching rules under all system conditions.

Table 5.31 Selection percentages of dispatching rules for k=3

Dispatching Rule	85% *U*	90% *U*	95% *U*
COVERT	0.53	0.53	0.51
SPT	0.47	0.47	0.49

Table 5.32 Selection percentages of dispatching rules for k=4

Dispatching Rule	85% *U*	90% *U*	95% *U*
COVERT	0.56	0.54	0.53
SPT	0.44	0.46	0.47

Table 5.33 Selection percentages of dispatching rules for k=5

Dispatching Rule	85% *U*	Dispatching Rule	90% *U*	95% *U*
COVERT	0.61	***COVERT***	0.57	0.56
PT+WINQ+SL	0.39	***SPT***	0.43	0.44

Table 5.34 Selection percentages of dispatching rules for k=6

Dispatching Rule	85% *U*	Dispatching Rule	90% *U*	95% *U*
COVERT	0.62	***COVERT***	0.61	0.59
PT+WINQ+SL	0.38	***SPT***	0.39	0.41

The selection percentages of dispatching rules are close to results of experiment with single state determination criteria. The percentage of the COVERT rule is increases with the allowance factor and decreases with the utilization level. The selection percentages of these dispatching rules in each state for the allowance factor of 3 are presented in Table 5.35. The tables for other conditions are illustrated in Appendix I. The changes of the percentages are clearly seen in Figure 5.17 and 5.18 for the utilization level of 90% and k=3, 6. The other related figures are also given in Appendix I.

Table 5.35 Selection percentages of dispatching rules in each state for k=3

State	85% *U*		90% *U*		95% *U*	
	COVERT	***SPT***	***COVERT***	***SPT***	***COVERT***	***SPT***
1	0.32	0.68	0.28	0.72	0.25	0.75
2	0.29	0.71	0.26	0.74	0.22	0.78
3	0.25	0.75	0.23	0.77	0.20	0.80
4	0.23	0.77	0.22	0.78	0.18	0.82
5	0.33	0.67	0.30	0.70	0.25	0.75
6	0.30	0.70	0.28	0.72	0.21	0.79
7	0.30	0.70	0.26	0.74	0.17	0.83
8	0.28	0.72	0.25	0.75	0.17	0.83
9	0.32	0.68	0.33	0.67	0.29	0.71
10	0.27	0.73	0.31	0.69	0.26	0.74
11	0.28	0.72	0.28	0.72	0.24	0.76
12	0.28	0.72	0.27	0.73	0.21	0.79
13	0.40	0.60	0.35	0.65	0.35	0.65
14	0.37	0.63	0.33	0.67	0.31	0.69
15	0.33	0.67	0.30	0.70	0.29	0.71
16	0.29	0.71	0.28	0.72	0.26	0.74
17	0.57	0.43	0.42	0.58	0.40	0.60
18	0.55	0.45	0.40	0.60	0.36	0.64
19	0.46	0.54	0.38	0.62	0.36	0.64
20	0.44	0.56	0.37	0.63	0.33	0.67
21	0.68	0.32	0.58	0.42	0.57	0.43
22	0.65	0.35	0.56	0.44	0.55	0.45
23	0.62	0.38	0.45	0.55	0.43	0.57
24	0.60	0.40	0.43	0.57	0.42	0.58
25	0.72	0.28	0.75	0.25	0.75	0.25
26	0.70	0.30	0.70	0.30	0.70	0.30
27	0.68	0.32	0.64	0.36	0.62	0.38
28	0.66	0.34	0.63	0.37	0.60	0.40
29	0.77	0.23	0.80	0.20	0.83	0.17
30	0.74	0.26	0.78	0.22	0.80	0.20
31	0.72	0.28	0.74	0.26	0.76	0.24
32	0.70	0.30	0.72	0.28	0.74	0.26
33	0.84	0.16	0.85	0.15	0.82	0.18
34	0.80	0.20	0.82	0.18	0.80	0.20
35	0.78	0.22	0.76	0.24	0.79	0.21
36	0.75	0.25	0.74	0.26	0.79	0.21
37	0.85	0.15	0.84	0.16	0.85	0.15
38	0.82	0.18	0.82	0.18	0.81	0.19
39	0.78	0.22	0.77	0.23	0.77	0.23
40	0.77	0.23	0.76	0.24	0.75	0.25

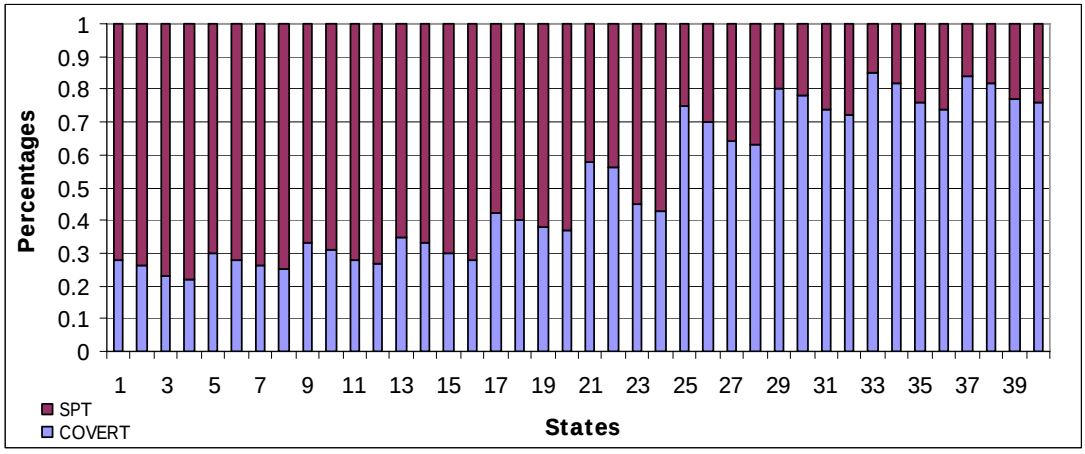

Figure 5.17 Selection percentages for 90% and *k*=3

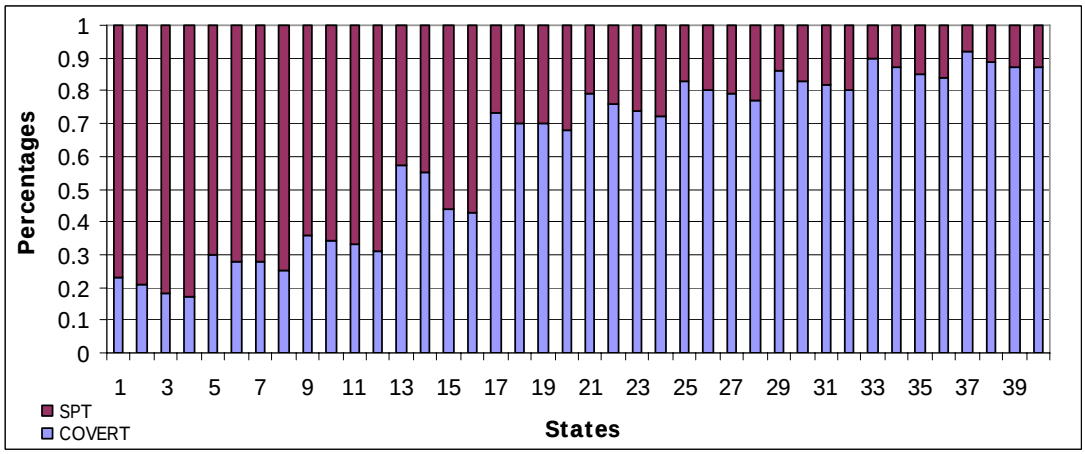

Figure 5.18 Selection percentages for 90% and *k*=6

It is observed that the selection percentage of the *COVERT* rule is high when the number of jobs in the buffer is low. The probability of being tardy of the jobs reduces if the number of jobs in the buffer is low. Therefore, the agent prefers to select the *COVERT* on these states since it receives more reward if the job is not tardy or tardiness level of the job is low.

The results are presented in Table 5.36 to 5.39. The performance of the *Q* agent is better than its performance in single state criterion model. Test4 also increases its performance. However, these performance improving is not significant.

Table 5.36 Performance of the *Q* agent and individual results for *k*=3

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	148.21	913.43	442.86	1293.30	1473.12	2368.02
AT-RPT	155.42	920.8	449.85	1299.63	1477.99	2372.80
EDD	120.85	887.72	412.33	1266.98	1446.40	2342.34
S/OPN	124.31	894.66	417.10	1273.14	1452.54	2348.61
COVERT	52.37	777.91	185.14	992.55	676.06	1558.27
SPT	92.87	706.16	250.24	898.27	780.81	1416.45
PT/TIS	60.64	744.93	190.11	998.90	741.90	1613.99
PT+WINQ+AT	123.00	869.53	394.24	1234.09	1397.86	2291.11
PT+WINQ+SL	77.37	787.74	297.57	1091.99	1291.02	2176.31
***Q*-Agent**	59.87	736.11	196.22	917.21	700.31	1438.43
Test 1	73.93	745.01	212.43	951.33	730.19	1478.29
Test 2	75.22	748.12	214.49	954.81	732.81	1483.91
Test 3	71.91	742.82	211.26	948.67	727.23	1473.46
Test 4	65.35	740.77	204.29	933.82	715.17	1460.83

Table 5.37 Performance of the *Q* agent and individual results for *k*=4

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	58.53	913.43	257.94	1293.30	1192.64	2368.02
AT-RPT	62.51	920.80	263.73	1299.63	1197.44	2372.80
EDD	37.19	881.25	221.03	1259.01	1153.56	2332.29
S/OPN	38.55	887.61	224.53	1265.84	1161.46	2340.72
COVERT	15.56	772.87	79.83	985.86	489.79	1613.01
SPT	34.01	706.16	120.04	898.90	593.01	1416.45
PT/TIS	25.06	744.93	83.51	998.90	515.71	1613.99
PT+WINQ+AT	46.16	869.53	223.21	1234.09	1121.61	2291.11
PT+WINQ+SL	30.90	782.09	130.77	1008.77	928.51	2053.42
***Q*-Agent**	21.76	734.14	88.72	917.12	521.72	1440.27
Test 1	27.58	745.23	104.18	942.89	545.64	1512.73
Test 2	28.99	748.56	107.72	945.23	549.91	1519.32
Test 3	26.32	743.18	101.78	938.55	539.43	1502.44
Test 4	24.11	741.33	95.46	929.42	529.62	1480.16

Table 5.38 Performance of the *Q* agent and individual results for *k*=5

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	21.99	913.43	142.07	1293.3	939.33	2368.02
AT-RPT	23.91	920.80	146.23	1299.63	943.95	2372.8
EDD	9.82	875.99	107.2	1251.68	889.45	2323.67
S/OPN	10.17	881.45	109.03	1257.89	895.36	2331.10
COVERT	3.27	728.25	31.10	960.23	334.01	1632.79
SPT	43.15	706.16	50.7	898.90	490.99	1416.45
PT/TIS	3.99	744.93	37.88	998.90	345.53	1613.98
PT+WINQ+AT	16.71	869.53	119.98	1234.09	875.45	2291.11
PT+WINQ+SL	8.85	703.42	43.02	966.40	633.01	1925.20
***Q*-Agent**	5.18	713.83	35.99	915.78	368.53	1486.88
Test 1	6.03	719.38	41.23	932.12	409.45	1536.37
Test 2	6.11	721.43	41.99	934.19	413.65	1543.74
Test 3	5.85	719.91	40.12	928.72	398.77	1529.29
Test 4	5.53	715.11	37.87	924.92	385.37	1512.48

Table 5.39 Performance of the *Q* agent and individual results for k=6

Dispatching Rule	85% *U*		90% *U*		95% *U*	
	MT	***MFT***	***MT***	***MFT***	***MT***	***MFT***
FIFO	8.34	913.43	75.72	1293.30	722.16	2368.02
AT-RPT	9.17	920.80	78.41	1299.63	726.42	2372.80
EDD	2.40	870.22	48.14	1244.41	662.60	2314.37
S/OPN	2.56	875.97	48.92	1250.12	666.02	2320.35
COVERT	1.23	721.56	13.99	956.96	214.27	1621.27
SPT	31.85	706.16	25.40	898.90	365.77	1416.45
PT/TIS	1.09	744.93	20.21	998.90	225.58	1613.98
PT+WINQ+AT	6.20	869.53	60.67	1234.09	667.28	2291.11
PT+WINQ+SL	5.45	696.83	22.31	957.88	330.69	1801.84
***Q*-Agent**	2.45	709.94	18.13	913.59	238.63	1485.81
Test 1	3.35	712.53	19.58	930.27	290.25	1523.28
Test 2	3.41	713.02	19.92	933.74	293.74	1529.85
Test 3	3.28	711.67	19.37	927.49	282.19	1516.39
Test 4	2.97	710.51	18.69	920.26	267.33	1509.28

5.8.1 Performance Analysis of Q-learning Agent

Proposed ranking method used in Section 4.9.4 is also used in this section for the same three cases. The results are close to ones in Section 5.7.4. When the objectives have the same importance and flow time is more important, same rankings are observed for all methods under all conditions. Simulation results and ranking graphics related to these cases are presented in Appendix L and Appendix M.3. On the other

hand, results differ from Section 5.7.4 for some conditions when the mean tardiness is more important. The ranks of all methods for all conditions when the mean tardiness is important are shown in Figure 5.19.

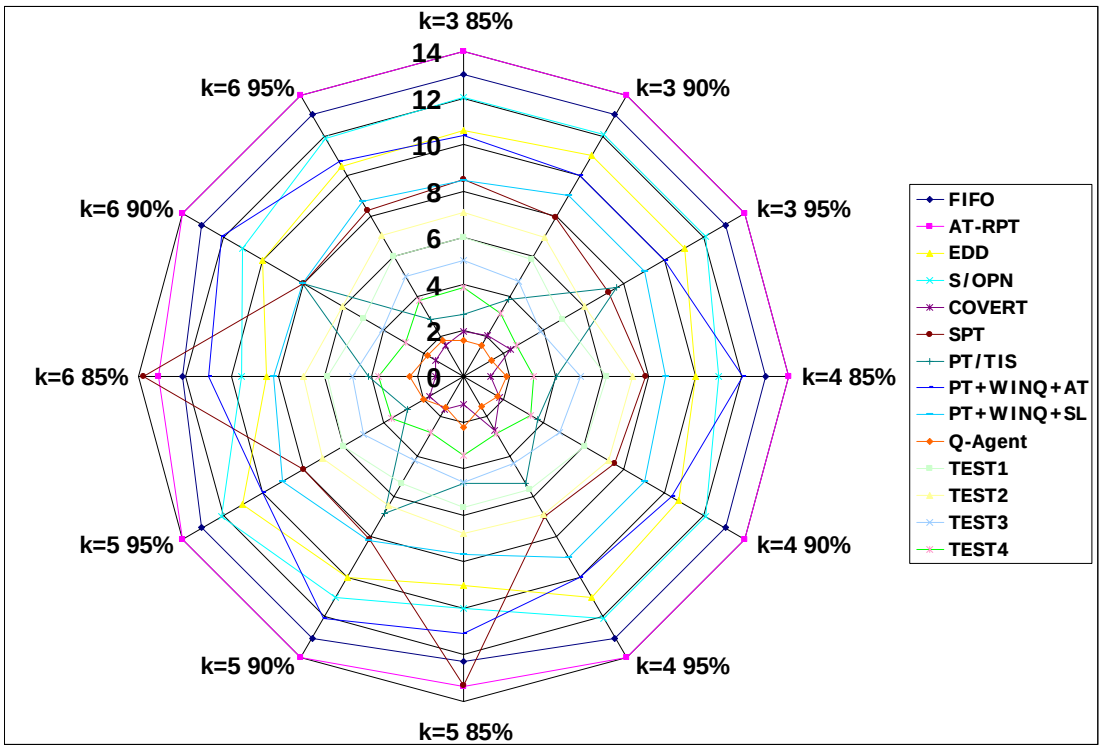

Figure 5.19 Ranks of all methods for all conditions when $\overline{w}_1 > \overline{w}_2$

When the mean tardiness is important for the decision maker, *Q* agent had the best performance only for three conditions, 90% k=3, 95% k=3 and 95% k=4, in Section 5.7.4. *COVERT* dispatching rule outperformed the agent under the rest of the conditions. However, in this experiment, agent also outperforms the *COVERT* rule under conditions of 90% k=5, 90% k=4 and 85% k=3 through the new state construction.

CHAPTER SIX
CONCLUSION

6.1 Conclusion

Studies related to multi criteria machine scheduling have been studied for a long time. However, in most of these studies, a static environment where the all jobs are available before processing was considered. Dynamic scheduling problems have been rarely mentioned in the literature. In this thesis, dynamic bicriteria scheduling problem was studied on two separate systems. *Q*-learning algorithm, a Reinforcement learning algorithm, was proposed as a solution approach to find a good schedule for these systems.

In chapter four, proposed solution approach was applied to a dynamic single machine bicriteria scheduling problem. The problem involves two objectives, namely minimizing mean tardiness (*MT*) and minimizing number of tardy jobs (*NTJ*). Proposed methodology consists of three phases. In the first phase, six dispatching rules that are generally used on dynamic single machine scheduling problem in the literature were included to the study. Performances of these dispatching rules were tested individually for mean tardiness and number of tardy jobs under conditions of different utilization levels and due date assignments. It is observed that *SPT* dispatching rule gives the best results under all system conditions for mean tardiness. On the other hand, *HA* outperforms all dispatching rules when the number of tardy jobs is considered.

In the second phase, learning potential of minimizing one performance measure of the agent was investigated for *MT* and *NTJ*. Three dispatching rules, including the best dispatching rule, were involved to the test for each measure of performance and system condition. Purpose of these tests was to determine if the single machine agent enables to learn to select the best dispatching rule for *MT* and *NTJ* which were presented in the first phase. The results demonstrated that the *Q* learning agent is able

to learn to select the proper dispatching rule for each of the system conditions for each objective in a dynamic single machine system.

In the third phase, *Q*-learning agent was trained to minimize mean tardiness and number of tardy jobs on a dynamic single machine scheduling problem. The best rules for each measure of performance, *SPT* and *HA*, were included to training process of the agent. The *Q*-learning algorithm was used as an optimization tool differently from its usage in second phase The *Q*-learning agent was trained to learn minimizing which performance measure in which system state. The performance of *Q*-learning agent is between *HA* and *SPT* as it was expected. The agent outperforms the *SPT* when the number of tardy jobs is considered but *SPT* is better then the *Q*-learning agent for minimizing mean tardiness. On the other hand, the agent outperforms the *HA* in case of mean tardiness but *HA* emerges to the best for number of tardy jobs.

A new ranking method was developed to investigate the performance of *Q*-learning agent thoroughly. In this method, all the results for two objectives were normalized. Thereafter, very large amount of weights between 0 and 1 are generated. For each weight, utility values of the methods were calculated and ranks of the methods were determined depending on the utility values. The proposed method evaluates the average rank of the each method. Therefore, a weight independent rank was obtained for each method. Three different cases were investigated with this new method.

In the first case, importance of the both objectives is equal since weights vary between 0 and 1. *Q*-learning agent outperforms all the other methods when the weight independent ranking considered. Also, Test 4 performs well under this system condition. In the other two cases, a preference was given to each objective. In this sense, weights were generated between 0 and 1, as it was in the first experiment, but the bigger weight was assigned always to the objective which is more important. When the mean tardiness is important for the decision maker, *Q*-agent has the best performance only for two conditions. *SPT* dispatching rule outperforms the agent

under the rest of the system conditions. When the number of tardy jobs is important for the decision maker, *Q* agent outperforms all other methods including *HA*.

In chapter five, proposed solution approach was applied to a dynamic flow shop bicriteria scheduling problem. The problem consists of two different objectives; minimizing mean tardiness (*MT*) and minimizing mean flow time (*MF*). The *Q* learning algorithm was determined as a solution approach to minimize these performance measures under different system conditions. The methodology used in the chapter four was also used for this problem. In the first phase, the performances of nine dispatching rules were tested on a dynamic flow shop system under different system conditions for each objective individually. The *COVERT* rule is the best for minimizing mean tardiness except for one case. The *SPT* rule outperforms the other dispatching rules for minimizing mean tardiness except for two cases. In the second phase, the application potential of the *Q* learning algorithm to a dynamic flow shop dispatching rule selection problem was investigated. The results indicate that the *Q* learning agent is able to learn to select the proper dispatching rule for each of the system conditions on dynamic flow shop system for each objective.

In the third phase, *Q* learning agent was trained to minimize mean tardiness and mean flow time on a dynamic flow shop bicriteria scheduling problem. The best rules for each measures of performance for each system condition according to results of first phase were included to training process of the agent. The *Q* learning agent was trained to learn minimizing which performance measure in which system state. The new ranking method was also used in this chapter to investigate the performance of *Q* learning agent. The results show that the *Q* agent emerges to the best when the weight independent ranking is considered. Test 4 also performed well under all conditions. When the mean tardiness is important, *Q* agent has the best performance only for three conditions. *COVERT* dispatching rule outperforms the agent under the rest of the conditions. When the mean flow time is important for the decision maker, *Q* agent outperforms all other methods including *SPT* except for one condition.

In chapter five, *Q* learning model was modified through construction of a new state framework. The results indicated that when the objectives have the same importance and flow time is more important, same rankings were observed for all methods under all conditions. However, the performance of the *Q* learning agent was increased when the mean tardiness is important.

Consequently, an agent based approach was proposed and applied to dynamic bicriteria scheduling problems. Results show that application of *Q* learning algorithm to dynamic single machine systems and flow shops is effective when the multiobjective decision making is considered.

6.2 Future Research

This study focused on dynamic bicriteria scheduling problem on a single machine and a flow shop manufacturing systems. For future studies, proposed methodology may be applied to different manufacturing systems like job shops and flexible manufacturing systems.

In this study, two objectives were included to the each research work. More objectives may be investigated in each application. Also, different performance measures may be incorporated to the study.

REFERENCES

Allahverdi A. (2002). The two and m-machine flowshop scheduling problems with bicriteria of makespan and mean flowtime, *European Journal of Operational Research, 147(2), Fuzzy Sets in Scheduling and Planning*, 373-396.

Allahverdi A. (2003). A new heuristic for m-machine flowshop scheduling problem with bicriteria of makespan and maximum tardiness, *Computers and Operations Research*, *31(2)*, 157-180.

Armentano V. A. & Arroyo J.E.C. (2004). An Application of a Multi-Objective Tabu Search Algorithm to a Bicriteria Flowshop Problem, *Journal of Heuristics, 10(5),* 463–481.

Aydın M.E. & Öztemel E. (2000). Dynamic job-shop scheduling using reinforcement learning agents, *Robotics and Autonomous Systems, 33(2)*,169–178.

Azizoğlu M. , Kondakçı S. & Köksalan M. (2003). Single machine scheduling with maximum earliness and number tardy, *Computers and Industrial Engineering, 45(2),* 257-268.

Baker, K.R. (1974). *Introduction to sequencing and scheduling.* New York: Wiley and Sons.

Barrett, R. and Kadipasaoglu, S. N. (1990). Evaluation of dispatching rules in a dynamic flow shop, *Production and Inventory Management Journal*, *31*(1), 54-58.

Biskup, D. & Chang Edwin, T.C, (1999). Multiple Machine Scheduling With Earliness /Tardiness And Completion Time Penalties, *Computers and Operations Research, 26(1),* 45-57.

Brandimarte P.,&Villa A. (1995). *Advanced Models for Manufacturing Systems Management,* Florida: CRC Press, Inc.

Chand S. , Traub R. & Uzsoy R. (1996). Single-Machine Scheduling with Dynamic Arrivals: Decomposition Results and an Improved Algorithm, *Naval Research Logistics, 43(5),* 709-719.

Chen W. & Sheen G. (2007). Single-machine scheduling with multiple performance measures: Minimizing job-dependent earliness and tardiness subject to the number of tardy jobs, Int. *J. Production Economics 109* . 214–229.

Crites, R. H. & Barto, A.G. (1996). Improving Elevator Performance using Reinforcement Learning. *Advances in Neural Information Processing Systems, 8,* 1017-1023.

Daya M., Raouf A.& Duffuaa S.O.(2003). Minimizing mean tardiness subject to unspecified minimum number tardy for a single machine, *European Journal of Operational Research, 89(1)*,100-107.

Das, T. K., Gosavi, A., Mahadevan, S., & Marchalleck, N. (1999). Solving Semi-Markov Decision Problems Using Average Reward Reinforcement Learning. *Management Science, 45(4)*, 560-574.

Duenas A. & Petrovic D.(2004). A New Approach to Multi-objective Single Machine Scheduling Problems under Fuzziness, *Decision Support in an Uncertain and Complex World: The IFIP TC8/WG8.3 International Conference,* 223-232.

Eren T. & Güner E. (2006). A bicriteria flowshop scheduling problem with setup times, *Applied Mathematics and Computation (New York)*,*183(2)*,1292-1300.

Eren T. & Güner E. (2006). A bicriteria scheduling with sequence-dependent setup times, *Applied Mathematics and Computation (New York)*, *179(1),* 378-385.

Eren T. & Güner E. The tricriteria flowshop scheduling problem (2007). *The International Journal of Advanced Manufacturing Technology*.

Gupta A. K. & Sivakumar A. I. (2006). Multi-objective scheduling of two-job families on a single machine, *Winter Simulation Conference, Proceedings of the 38th conference on Winter simulation,* 1749-1756.

Hino C.M., Ronconi D.P. & Mendres A.B. (2005). Minimizing earliness and tardiness penalties in a single-machine problem with a common due date, *European Journal of Operational Research, 160*, 190–201.

Hong J. & Prabhu V. (2004). Distributed Reinforcement Learning Control for Batch Sequencing and Sizing in Just-In-Time Manufacturing Systems, *Applied Intelligence, 20(1),* 71-87.

Horng H. (2006). Comparing Steady-state Performance of Dispatching Rule-pairs in Open Shops*, International Journal of Applied Science and Engineering,4(3),* 259-273.

Hunsucker, J.L. and Shah, J.R. (1994). Comparative performance analysis of priority rules in a constrained flow shop with multiple processors environment, *European Journal of Operational Research, 72(1) ,*102-114.

Hwang, C. L. & Yoon, K. (1981). *Multiple Attribute Decision Making: Methods and Applications*. Berlin: Springer.

Jang W. (2002). Dynamic scheduling of stochastic jobs on a single machine, *European Journal of Operational Research, 138(3),* 518-530.

Kaelbling L. P., Littman M. L.& Moore A. W. (1996). Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research, 23(4),* 237-285.

Kim G.H. &Lee C.S.G.(1995). Genetic reinforcement learning approach to the machine scheduling problem, *Proceedings - IEEE International Conference on Robotics and Automation, 1,* 196-201.

Köksalan M. & Keha A.B. (2003). Using Genetic Algorithms For Single-Machine Bicriteria Scheduling Problems, *European Journal of Operational Research*, 145(3), 543-556.

Köksalan M. & Köktener Karasakal E. (2000). A Simulated Annealing Approach to Bicriteria Scheduling Problems on a Single Machine, *Journal of Heuristics, 6,* 311–327.

Köksalan M. (1999). A Heuristic Approach to Bicriteria Scheduling, *Naval Research Logistics, 46(7),* 777-789.

Köksalan M. , Azizoğlu M. & Köksalan Kondakçı S. (1998). Minimizing Flowtime and maximum earliness on a single machine*, IIE Transactions, 30(2)*, 192-200.

Kong L. & Wu J. (2005). Dynamic Single Machine Scheduling Using Q-Learning Agent*, International Conference on Machine Learning and Cybernetics*, *5*, 3237-3241.

Laurent G.J. & Piat E. (2002). Learning mixed behaviours with parallel *Q* learning, *International Conference on Intelligent Robots and Systems*, 1002-1007

Lasso M. , Pandolfi D. , De San Pedro M. ,Villagra A. & Gallard R. (2004). Solving dynamic tardiness problems in single machine environments, *Proceedings of the 2004 Congress on Evolutionary Computation, 1*,1143-1149.

Liaw C. (1999). A branch-and-bound algorithm for the single machine earliness and tardiness scheduling problem, *Computers & OperationsResearch, 26(7),* 679-693.

M'Hallah R. (2006). Minimizing total earliness and tardiness on a single machine using a hybrid heuristic, *Computers and Operations Research, 34(10),* 3126-3142.

Madureira A., Ramos C. & Carmo Silva S. (2001). A GA Based Scheduling System For Dynamic Single Machine Problem, *Proceedings of the 4th IEEE International Symposium on Assembly and Task Planning Soft Research Park, Fukuoka, Japan,* 262-267.

Mahadevan, S. & Theocharous, G. (1998). Optimizing Production Manufacturing using Reinforcement Learning. *The 11th International FLAIRS Conference, AAAI Press,* 372-377.

Mahadevan, S., Khaleeli, N., & Marchalleck, N. (1997a). Designing Agent Controllers using Discrete-Event Markov Models. *AAAI Fall Symposium on Model-Directed Autonomous Systems, Cambridge, MA.*

Mahadevan, S., Marchalleck, N., Das, T. K., & Gosavi, A. (1997b). Self-Improving Factory Simulation using Continuous-time Average-Reward Reinforcement Learning. *Proceedings of the 4th International Machine Learning Conference,* 202-210.

Mariano-Romero C.E. , V.H.Alcocer-Yamanaka & Morales E. F. (2007). Multi-objective optimization of water-using systems, *European Journal of Operational Research, 181(3),* 1691-1707.

Morton T.E. & Pentico D.W. (1993) *Heuristics Scheduling Systems*. Canada: John Wiley & Sons. Inc..

Muratha T. & Ishibuchi H. (1998). A Multi-Objective Genetic Local Search Algorithm and Its Application to Flowshop Scheduling*, IEEE Transactions On Systems, Man, And Cybernetics Part C: Applications And Reviews, 28(3),* 392-403.

Myung H. & Bien Z. Z. (2003). Design of the Fuzzy Multiobjective Controller Based on the Eligibility Method*, International Journal of Intelligent Systems, 18(5),* 509-528.

Neppalli V. R. ,Chen C. & Gupta J.N.D. (1996). Genetic algorithms for the two-stage bicriteria flowshop problem*, European Journal of Operational Research ,95(2),* 356-373.

Paternina-Arboleda C.D. & Das T.K. (2001). Intelligent dynamic control policies for serial production lines, *IEE Transactions 33 ,* 65-77.

Paternina-Arboleda C.D. & Das T.K. (*2005*). A multi-agent reinforcement learning approach to obtaining dynamic control policies for stochastic lot scheduling problem, *Simulation Modelling Practice and Theory 13,* 389-406.

Pinedo, M., (1995*). Scheduling theory, Algorithms, and Systems.* Prentice-Hall.

Pinedo, M.,& Chao, X. (1999). *Operations Scheduling with Applications in Manufacturing and Services.* Singapore: McGraw-Hill

Rabelol L.C., Jones A. & Yih Y. (1994). Development of a real-time learning scheduler using reinforcementlearning concepts*, Intelligent Control, Proceedings of the 1994 IEEE International Symposium* ,291-296.

Raicevic P. (2006). Parallel reinforcement learning using multiple reward signals, *Neurocomputing*, 69 (16-18).

Rajendran C. & Ziegler H. (2003). Scheduling to minimize the sum of weighted flowtime and weighted tardiness of jobs in a flowshop with sequence-dependent setup times*, European Journal of Operational Research, 149(3),* 513-522.

Rajendran, C. and Holthaus, O. (1999). A comparative study of dispatching rules in dynamic flow shops and job shops, *European Journal of Operational Research, 116*: 156-170.

Sarper, H. and Henry, M. (1996). Combinational evaluation of six dispatching rules in a dynamic two-machine flow shop, *Omega, 24(1),* 73-81.

Sayın S. & Karabatı S. (1999). A bicriteria approach to the two-machine flow shop scheduling problem, *European Journal of Operational Research ,113(2),* 435-449.

Singh, S. P. & Bertsekas, D. (1997). Reinforcement Learning for Dynamic Channel Allocation in Cellular Telephone Systems. *In Advances in Neural Information Processing Systems: Proceedings of the1996 Conferance,* 974-980.

Sivrikaya F. & Ulusoy G. (1998). A bicriteria two-machine permutation flowshop problem*, European Journal of Operational Research, 107(2),* 414-430.

Su L. & Chou F. (2000). Heuristic for scheduling in a two-machine bicriteria dynamic flowshop with setup and processing times separated, *Production Planning and Control*, 11(8), 806-819.

Soroush H. M. (2006). Minimizing the weighted number of early and tardy jobs in stochastic single machine scheduling problem, *Europan Journal of Operation Researc, 181(1)*, 266-287

Sule D. (1997). *Industrial Scheduling*. Boston: PWS Pub. Co.

Sutton, R. S. & Barto, A. G. (1999). *Reinforcement Learning: An Introduction.* The MIT Press.

Tesauro, G. (1995). Temporal Difference Learning and TD-Gammon. *Commun. of the ACM, 38(3)*, 58-67.

Valente J.M.S. & Alves R.A. (2007). Heuristics for the single machine scheduling problem with quadratic earliness and tardiness penalties, Computers and Operations Researc,

Valente J.M.S. (2007). Dispatching heuristics for the single machine early/tardy scheduling problem with job-independent penalties*, Computers & Industrial Engineering 52(4),* 434–447.

Valluri A. & Croson D. (2005). Agent learning in supplier selection models, *Decision Support Systems 39(2),* 219– 240.

Wang Y. & Usher J.M. (2004). Application of reinforcement learning for agent-based production scheduling, *Engineering Applications of Artificial Intelligence 18,* 73–82.

Wang Y. & Usher J. M.(2007) A reinforcement learning approach for developing routing policies in multi-agent production scheduling. *International Journal of Advanced Manufacturing Technology*, *33(3-4)*, 323-333.

Watkins, C. (1989). *Learning from Delayed Rewards.* Doctoral dissertation, Cambridge University.

Whiteson S. & Stone P. (2004). Adaptive job routing and scheduling, *Engineering Applications of Artificial Intelligence 17(7),* 855–869.

Yu X. & Ram B. (2006). Bio-inspired scheduling for dynamic job shops with flexible routing and sequence-dependent setups*, International Journal of Production Research, 44(22),* 4793-4813.

Zhang, W. & Dietterich, T. G. (1995). A Reinforcement Learning Approach to Job-Shop Scheduling. *Proceedings of the 14th International Joint Conference on Artificial Intelligence, 31(5),* 1114-1120.

Zi W. Y. & Yang Z. M. (2005). Reinforcement learning-based approach to dynamic job-shop scheduling*, Zidonghua Xuebao/Acta Automatica Sinica, 31(5),* 765 771.

APPENDIX A: State-Action Tables for Single Machine Applications

Table A.1 State action table for 80% utilization level and tight due dates

State	Determination Criteria	*EDD*	*SPT*	*FIFO*
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ETL < 5$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $5 \leq ETL < 15$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $15 \leq ETL < 40$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $40 \leq ETL < 60$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $60 \leq ETL < 90$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $90 \leq ETL < 150$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $150 \leq ETL < 250$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $250 \leq ETL < 350$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $350 \leq ETL < 550$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $550 \leq ETL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table A.2 State action table for 80% utilization level and loose due dates

State	Determination Criteria	*EDD*	*SPT*	*FIFO*
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ETL < 5$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $5 \leq ETL < 10$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $10 \leq ETL < 35$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $35 \leq ETL < 50$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $50 \leq ETL < 80$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $80 \leq ETL < 130$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $130 \leq ETL < 220$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $220 \leq ETL < 330$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $330 \leq ETL < 520$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $520 \leq ETL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table A.3 State action table for 85% utilization level and tight due dates

State	Determination Criteria	EDD	SPT	FIFO
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ETL < 10$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $10 \leq ETL < 30$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $30 \leq ETL < 50$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $50 \leq ETL < 100$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $100 \leq ETL < 150$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $150 \leq ETL < 250$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $250 \leq ETL < 350$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $350 \leq ETL < 480$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $480 \leq ETL < 660$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $660 \leq ETL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table A.4 State action table for 90% utilization level and tight due dates

State	Determination Criteria	*EDD*	*SPT*	*FIFO*
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ETL < 30$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $30 \leq ETL < 60$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $60 \leq ETL < 120$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $120 \leq ETL < 180$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $180 \leq ETL < 250$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $250 \leq ETL < 350$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $350 \leq ETL < 450$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $450 \leq ETL < 600$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $600 \leq ETL < 750$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $750 \leq ETL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table A.5 State action table for 90% utilization level and loose due dates

State	Determination Criteria	*EDD*	*SPT*	*FIFO*
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ETL < 25$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $25 \leq ETL < 50$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $50 \leq ETL < 100$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $100 \leq ETL < 150$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $150 \leq ETL < 230$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $230 \leq ETL < 330$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $330 \leq ETL < 430$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $430 \leq ETL < 550$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $550 \leq ETL < 700$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $700 \leq ETL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table A.6 State action table for 95% utilization level and loose due dates

State	Determination Criteria	*EDD*	*SPT*	*FIFO*
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ETL < 50$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $50 \leq ETL < 80$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $80 \leq ETL < 200$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $200 \leq ETL < 300$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $300 \leq ETL < 500$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $500 \leq ETL < 650$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $650 \leq ETL < 800$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $800 \leq ETL < 900$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $900 \leq ETL < 1000$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $1000 \leq ETL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

APPENDIX B: Reward Functions for Minimizing Mean Tardiness on Single Machine Problem

Table B.1 Reward function for 80% utilization level and tight due dates

Condition	Reward
Tardiness ≤ 0	1
0 < Tardiness ≤ 2.5	-1
2.5 < Tardiness ≤ 5.5	-2
5.5 < Tardiness ≤ 10.5	-3
10.5 < Tardiness ≤ 15.5	-4
15.5 < Tardiness ≤ 20	-5
20 < Tardiness ≤ 30	-6
30 < Tardiness ≤ 40	-7
40 < Tardiness ≤ 45	-8
45 < Tardiness	-9

Table B.2 Reward function for 80% utilization level and loose due dates

Condition	Reward
Tardiness ≤ 0	1
0 < Tardiness ≤ 2	-1
2 < Tardiness ≤ 4	-2
4 < Tardiness ≤ 8	-3
8< Tardiness ≤ 12	-4
12 < Tardiness ≤ 18	-5
18 < Tardiness ≤ 25	-6
25 < Tardiness ≤ 35	-7
35 < Tardiness ≤ 40	-8
40 < Tardiness	-9

Table B.3 Reward function for 85% utilization level and tight due dates

Condition	Reward
Tardiness ≤ 0	1
0 < Tardiness ≤ 5	-1
5 < Tardiness ≤ 10	-2
10 < Tardiness ≤ 15	-3
15 < Tardiness ≤ 20	-4
20 < Tardiness ≤ 30	-5
30 < Tardiness ≤ 40	-6
40 < Tardiness ≤ 55	-7
55 < Tardiness ≤ 65	-8
65 < Tardiness	-9

Table B.4 Reward function for 90% utilization level and tight due dates

Condition	Reward
Tardiness ≤ 0	1
0 < Tardiness ≤ 10	-1
10 < Tardiness ≤ 25	-2
25 < Tardiness ≤ 35	-3
35 < Tardiness ≤ 45	-4
45 < Tardiness ≤ 55	-5
55 < Tardiness ≤ 65	-6
65 < Tardiness ≤ 75	-7
75 < Tardiness ≤ 90	-8
90 < Tardiness	-9

Table B.5 Reward function for 90% utilization level and loose due dates

Condition	Reward
Tardiness ≤ 0	1
0 < Tardiness ≤ 10	-1
10 < Tardiness ≤ 20	-2
20 < Tardiness ≤ 30	-3
30 < Tardiness ≤ 40	-4
40 < Tardiness ≤ 50	-5
50 < Tardiness ≤ 60	-6
60 < Tardiness ≤ 70	-7
70 < Tardiness ≤ 80	-8
80 < Tardiness	-9

Table B.6 Reward function for 95% utilization level and loose due dates

Condition	Reward
Tardiness ≤ 0	1
0 < Tardiness ≤ 25	-1
25 < Tardiness ≤ 40	-2
40 < Tardiness ≤ 60	-3
60 < Tardiness ≤ 80	-4
80 < Tardiness ≤ 100	-5
100 < Tardiness ≤ 120	-6
120 < Tardiness ≤ 140	-7
140 < Tardiness ≤ 190	-8
190 < Tardiness	-9

APPENDIX C: Reward Functions for Dynamic Single Machine Bicriteria Scheduling Problem

Table C.1 Reward function for 80% utilization level and tight due dates

Minimizing *MT*		**Minimizing *NTJ***	
Condition	**Reward**	**Condition**	**Reward**
$0 \le s_1 < 5$ and Tardiness< 0	4	$0 \le s_1 < 5$ and Tardiness < 0	4
$5 \le s_1 < 8$ and Tardiness< 0	3	$5 \le s_1 < 8$ and Tardiness < 0	3
$8 \le s_1 < 10$ and Tardiness < 0	2	$8 \le s_1 < 10$ and Tardiness < 0	2
$s_1 = 10$ and Tardiness < 0	1	$s_1 = 10$ and Tardiness < 0	1
0 ≤ Tardiness <5.5	-1	$0 \le s_1 < 3$ and Tardiness > 0	-1
5.5 ≤ Tardiness <15.5	-2	$3 \le s_1 < 6$ and Tardiness > 0	-2
15.5 ≤ Tardiness <30	-3	$6 \le s_1 < 8$ and Tardiness > 0	-3
30 ≤ Tardiness <45	-4	$8 \le s_1 < 10$ and Tardiness > 0	-4
45 ≤ Tardiness	-5	$S_1 = 10$ and Tardiness > 0	-5

Table C.2 Reward function for 80% utilization level and loose due dates

Minimizing *MT*		**Minimizing *NTJ***	
Condition	**Reward**	**Condition**	**Reward**
$0 \le s_1 < 5$ and Tardiness< 0	4	$0 \le s_1 < 5$ and Tardiness < 0	4
$5 \le s_1 < 8$ and Tardiness< 0	3	$5 \le s_1 < 8$ and Tardiness < 0	3
$8 \le s_1 < 10$ and Tardiness < 0	2	$8 \le s_1 < 10$ and Tardiness < 0	2
$s_1 = 10$ and Tardiness < 0	1	$S_1 = 10$ and Tardiness < 0	1
0 ≤ Tardiness <4	-1	$0 \le s_1 < 3$ and Tardiness > 0	-1
4 ≤ Tardiness <12	-2	$3 \le s_1 < 6$ and Tardiness > 0	-2
12 ≤ Tardiness <25	-3	$6 \le s_1 < 8$ and Tardiness > 0	-3
25 ≤ Tardiness <40	-4	$8 \le s_1 < 10$ and Tardiness > 0	-4
40 ≤ Tardiness	-5	$s_1 = 10$ and Tardiness > 0	-5

Table C.3 Reward function for 85% utilization level and tight due dates

Minimizing *MT*		**Minimizing *NTJ***	
Condition	**Reward**	**Condition**	**Reward**
$0 \le s_1 < 5$ and Tardiness< 0	4	$0 \le s_1 < 5$ and Tardiness < 0	4
$5 \le s_1 < 8$ and Tardiness< 0	3	$5 \le s_1 < 8$ and Tardiness < 0	3
$8 \le s_1 < 10$ and Tardiness < 0	2	$8 \le s_1 < 10$ and Tardiness < 0	2
$s_1 = 10$ and Tardiness < 0	1	$S_1 = 10$ and Tardiness < 0	1
0 ≤ Tardiness <10	-1	$0 \le s_1 < 3$ and Tardiness > 0	-1
10 ≤ Tardiness <20	-2	$3 \le s_1 < 6$ and Tardiness > 0	-2
20 ≤ Tardiness <30	-3	$6 \le s_1 < 8$ and Tardiness > 0	-3
30 ≤ Tardiness <50	-4	$8 \le s_1 < 10$ and Tardiness > 0	-4
50 ≤ Tardiness	-5	$s_1 = 10$ and Tardiness > 0	-5

Table C.4 Reward function for 85% utilization level and loose due dates

Minimizing *MT*		**Minimizing *NTJ***	
Condition	**Reward**	**Condition**	**Reward**
$0 \leq s_1 < 5$ and Tardiness< 0	4	$0 \leq s_1 < 5$ and Tardiness < 0	4
$5 \leq s_1 < 8$ and Tardiness< 0	3	$5 \leq s_1 < 8$ and Tardiness < 0	3
$8 \leq s_1 <10$ and Tardiness < 0	2	$8 \leq s_1 < 10$ and Tardiness < 0	2
$s_1 = 10$ and Tardiness < 0	1	$S_1 = 10$ and Tardiness < 0	1
0 ≤ Tardiness <10	-1	$0 \leq s_1 < 3$ and Tardiness > 0	-1
10 ≤ Tardiness <15	-2	$3 \leq s_1 < 6$ and Tardiness > 0	-2
15 ≤ Tardiness <35	-3	$6 \leq s_1 < 8$ and Tardiness > 0	-3
35 ≤ Tardiness <45	-4	$8 \leq s_1 < 10$ and Tardiness > 0	-4
45 ≤ Tardiness	-5	$s_1 =10$ and Tardiness > 0	-5

Table C.5 Reward function for 90% utilization level and loose due dates

Minimizing *MT*		**Minimizing *NTJ***	
Condition	**Reward**	**Condition**	**Reward**
$0 \leq s_1 < 5$ and Tardiness< 0	4	$0 \leq s_1 < 5$ and Tardiness < 0	4
$5 \leq s_1 < 8$ and Tardiness< 0	3	$5 \leq s_1 < 8$ and Tardiness < 0	3
$8 \leq s_1 <10$ and Tardiness < 0	2	$8 \leq s_1 < 10$ and Tardiness < 0	2
$s_1 = 10$ and Tardiness < 0	1	$S_1 = 10$ and Tardiness < 0	1
0 ≤ Tardiness <10	-1	$0 \leq s_1 < 3$ and Tardiness > 0	-1
10 ≤ Tardiness <15	-2	$3 \leq s_1 < 6$ and Tardiness > 0	-2
15 ≤ Tardiness <35	-3	$6 \leq s_1 < 8$ and Tardiness > 0	-3
35 ≤ Tardiness <55	-4	$8 \leq s_1 < 10$ and Tardiness > 0	-4
55 ≤ Tardiness	-5	$s_1 =10$ and Tardiness > 0	-5

Table C.6 Reward function for 95% utilization level and tight due dates

Minimizing *MT*		**Minimizing *NTJ***	
Condition	**Reward**	**Condition**	**Reward**
$0 \leq s_1 < 5$ and Tardiness< 0	4	$0 \leq s_1 < 5$ and Tardiness < 0	4
$5 \leq s_1 < 8$ and Tardiness< 0	3	$5 \leq s_1 < 8$ and Tardiness < 0	3
$8 \leq s_1 <10$ and Tardiness < 0	2	$8 \leq s_1 < 10$ and Tardiness < 0	2
$s_1 = 10$ and Tardiness < 0	1	$S_1 = 10$ and Tardiness < 0	1
0 ≤ Tardiness <50	-1	$0 \leq s_1 < 3$ and Tardiness > 0	-1
50 ≤ Tardiness <90	-2	$3 \leq s_1 < 6$ and Tardiness > 0	-2
90 ≤ Tardiness <130	-3	$6 \leq s_1 < 8$ and Tardiness > 0	-3
130 ≤ Tardiness <200	-4	$8 \leq s_1 < 10$ and Tardiness > 0	-4
200 ≤ Tardiness	-5	$s_1 =10$ and Tardiness > 0	-5

Table C.7 Reward function for 95% utilization level and loose due dates

Minimizing *MT*		**Minimizing *NTJ***	
Condition	**Reward**	**Condition**	**Reward**
$0 \leq s_1 < 5$ and Tardiness< 0	4	$0 \leq s_1 < 5$ and Tardiness < 0	4
$5 \leq s_1 < 8$ and Tardiness< 0	3	$5 \leq s_1 < 8$ and Tardiness < 0	3
$8 \leq s_1 <10$ and Tardiness < 0	2	$8 \leq s_1 < 10$ and Tardiness < 0	2
$s_1 = 10$ and Tardiness < 0	1	$S_1 = 10$ and Tardiness < 0	1
0 ≤ Tardiness <40	-1	$0 \leq s_1 < 3$ and Tardiness > 0	-1
40 ≤ Tardiness <80	-2	$3 \leq s_1 < 6$ and Tardiness > 0	-2
80 ≤ Tardiness <120	-3	$6 \leq s_1 < 8$ and Tardiness > 0	-3
120 ≤ Tardiness <190	-4	$8 \leq s_1 < 10$ and Tardiness > 0	-4
190 ≤ Tardiness	-5	$s_1 =10$ and Tardiness > 0	-5

APPENDIX D: State-Action Tables for Flow Shop Applications

Table D.1 State action table for 85% utilization level and k=3

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
Dummy State	$N < 2$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
1	$N > 1$ and $ASL < 0$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
2	$N > 1$ and $0 \leq ASL < 100$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
3	$N > 1$ and $100 \leq ASL < 200$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
4	$N > 1$ and $200 \leq ASL < 300$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
5	$N > 1$ and $300 \leq ASL < 350$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
6	$N > 1$ and $350 \leq ASL < 400$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
7	$N > 1$ and $400 \leq ASL < 450$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
8	$N > 1$ and $450 \leq ASL < 500$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
9	$N > 1$ and $500 \leq ASL < 560$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$
10	$N > 1$ and $560 \leq ASL$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$

Table D.2 State action table for 85% utilization level and k=4

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
Dummy State	$N < 2$	0	0	0
1	$N > 1$ and $ASL < 300$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $300 \leq ASL < 450$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $450 \leq ASL < 530$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $530 \leq ASL < 600$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $600 \leq ASL < 650$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $650 \leq ASL < 700$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $700 \leq ASL < 750$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $750 \leq ASL < 800$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $800 \leq ASL < 900$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $900 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.3 State action table for 85% utilization level and k=5

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL < 55$ 0	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $550 \leq ASL < 700$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $700 \leq ASL < 800$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $800 \leq ASL < 870$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $870 \leq ASL < 950$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $950 \leq ASL < 1020$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $1020 \leq ASL < 1080$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $1080 \leq ASL < 1150$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $1150 \leq ASL < 1250$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $1250 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.4 State action table for 85% utilization level and k=6

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL < 800$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $800 \leq ASL < 950$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $950 \leq ASL < 1080$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $1080 \leq ASL < 1150$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $1150 \leq ASL < 1220$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $1220 \leq ASL < 1300$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $1300 \leq ASL < 1380$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $1380 \leq ASL < 1470$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $1470 \leq ASL < 1580$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $1580 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.5 State action table for 90% utilization level and k=4

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL < 20$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $20 \leq ASL < 190$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $190 \leq ASL < 340$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $340 \leq ASL < 460$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $460 \leq ASL < 540$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $540 \leq ASL < 610$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $610 \leq ASL < 680$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $680 \leq ASL < 750$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $750 \leq ASL < 850$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $850 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.6 State action table for 90% utilization level and k=5

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL < 300$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $300 \leq ASL < 550$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $550 \leq ASL < 700$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $700 \leq ASL < 800$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $800 \leq ASL < 870$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $870 \leq ASL < 950$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $950 \leq ASL < 1020$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $1020 \leq ASL < 1100$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $1100 \leq ASL < 1200$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $1200 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.7 State action table for 90% utilization level and k=6

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL < 650$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $650 \leq ASL < 850$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $850 \leq ASL < 1000$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $1000 \leq ASL < 1100$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $1100 \leq ASL < 1170$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $1170 \leq ASL < 1250$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $1250 \leq ASL < 1320$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $1320 \leq ASL < 1400$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $1400 \leq ASL < 1550$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $1550 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.8 State action table for 95% utilization level and k=3

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL < -2200$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $-2200 \leq ASL < -1050$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $-1050 \leq ASL < -550$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $-550 \leq ASL < -250$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $-250 \leq ASL < -80$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $-80 \leq ASL < 40$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $40 \leq ASL < 200$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $200 \leq ASL < 350$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $350 \leq ASL < 450$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $450 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.9 State action table for 95% utilization level and k=4

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL < -1500$	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and $-1500 \leq ASL < -550$	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and $-550 \leq ASL < -150$	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and $-150 \leq ASL < 0$	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and $0 \leq ASL < 200$	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and $200 \leq ASL < 400$	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and $400 \leq ASL < 525$	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and $525 \leq ASL < 625$	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and $625 \leq ASL < 750$	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and $750 \leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.10 State action table for 95% utilization level and k=5

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL <$-700	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and -700 $\leq ASL <$ -100	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and -100 $\leq ASL <$100	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and 100 $\leq ASL <$ 400	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and 400 $\leq ASL <$ 575	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and 575 $\leq ASL <$ 700	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and 700 $\leq ASL <$ 810	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and 810 $\leq ASL <$ 920	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and 920 $\leq ASL <$ 1050	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and 1050 $\leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

Table D.11 State action table for 95% utilization level and k=6

State	Determination Criteria	*COVERT*	*PT/TIS*	*PT+WINQ+SL*
D.S	$N < 2$	0	0	0
1	$N > 1$ and $ASL <$-100	$Q(1,1)$	$Q(1,2)$	$Q(1,3)$
2	$N > 1$ and -100 $\leq ASL <$ 300	$Q(2,1)$	$Q(2,2)$	$Q(2,3)$
3	$N > 1$ and 300 $\leq ASL <$580	$Q(3,1)$	$Q(3,2)$	$Q(3,3)$
4	$N > 1$ and 580 $\leq ASL <$ 750	$Q(4,1)$	$Q(4,2)$	$Q(4,3)$
5	$N > 1$ and 750 $\leq ASL <$ 875	$Q(5,1)$	$Q(5,2)$	$Q(5,3)$
6	$N > 1$ and 875 $\leq ASL <$ 975	$Q(6,1)$	$Q(6,2)$	$Q(6,3)$
7	$N > 1$ and 975 $\leq ASL <$ 1100	$Q(7,1)$	$Q(7,2)$	$Q(7,3)$
8	$N > 1$ and 1100 $\leq ASL <$ 1250	$Q(8,1)$	$Q(8,2)$	$Q(8,3)$
9	$N > 1$ and 1250 $\leq ASL <$ 1400	$Q(9,1)$	$Q(9,2)$	$Q(9,3)$
10	$N > 1$ and 1400 $\leq ASL$	$Q(10,1)$	$Q(10,2)$	$Q(10,3)$

APPENDIX E: Reward Functions for Minimizing Mean Tardiness on Flow Shop Problem

Table E.1 Reward function for 85% utilization level and *k*=3, 4

Condition	Reward	Condition	Reward
Tardiness ≤ 0	1	Tardiness ≤ 0	1
0 < Tardiness ≤ 20	-1	0 < Tardiness ≤ 10	-1
20 < Tardiness ≤ 40	-2	10 < Tardiness ≤ 20	-2
40 < Tardiness ≤ 70	-3	20 < Tardiness ≤ 30	-3
70 < Tardiness ≤ 90	-4	30 < Tardiness ≤ 40	-4
90 < Tardiness ≤ 110	-5	40 < Tardiness ≤ 50	-5
110 < Tardiness ≤ 150	-6	50 < Tardiness ≤ 70	-6
150 < Tardiness ≤ 190	-7	70 < Tardiness ≤ 80	-7
190 < Tardiness ≤ 260	-8	80 < Tardiness ≤ 90	-8
260 < Tardiness	-9	90 < Tardiness	-9

Table E.2 Reward function for 85% utilization level and *k*=5, 6

Condition	Reward	Condition	Reward
Tardiness ≤ 0	1	Tardiness ≤ 0	1
0 < Tardiness ≤ 0.8	-1	0 < Tardiness ≤ 0.8	-1
0.8 < Tardiness ≤ 1.5	-2	0.8 < Tardiness ≤ 1.5	-2
1.5 < Tardiness ≤ 2.8	-3	1.5 < Tardiness ≤ 2.2	-3
2.8 < Tardiness ≤ 4	-4	2.2 < Tardiness ≤ 2.8	-4
4 < Tardiness ≤ 6	-5	2.8 < Tardiness ≤ 3.5	-5
6 < Tardiness ≤ 10	-6	3.5 < Tardiness ≤ 4	-6
10 < Tardiness ≤ 13	-7	4.4 < Tardiness ≤ 4.8	-7
13 < Tardiness ≤ 18	-8	4.8 < Tardiness ≤ 5.4	-8
18 < Tardiness	-9	5.4 < Tardiness	-9

Table E.3 Reward function for 90% utilization level and *k*=4

Condition	Reward
Tardiness ≤ 0	1
0 < Tardiness ≤ 20	-1
20 < Tardiness ≤ 50	-2
50 < Tardiness ≤ 70	-3
70 < Tardiness ≤ 90	-4
90 < Tardiness ≤ 110	-5
110 < Tardiness ≤ 130	-6
130 < Tardiness ≤ 150	-7
150 < Tardiness ≤ 180	-8
180 < Tardiness	-9

Table E.4 Reward function for 90% utilization level and *k*=5, 6

Condition	Reward	Condition	Reward
Tardiness ≤ 0	1	Tardiness ≤ 0	1
0 < Tardiness ≤ 10	-1	0 < Tardiness ≤ 4	-1
10 < Tardiness ≤ 15	-2	4 < Tardiness ≤ 8	-2
15 < Tardiness ≤ 20	-3	8 < Tardiness ≤ 10	-3
20 < Tardiness ≤ 30	-4	10 < Tardiness ≤ 13	-4
30 < Tardiness ≤ 40	-5	13 < Tardiness ≤ 18	-5
40 < Tardiness ≤ 50	-6	18 < Tardiness ≤ 25	-6
50 < Tardiness ≤ 70	-7	25 < Tardiness ≤ 30	-7
70 < Tardiness ≤ 90	-8	30 < Tardiness ≤ 45	-8
90 < Tardiness	-9	45 < Tardiness	-9

Table E.5 Reward function for 95% utilization level and *k*=3, 4

Condition	Reward	Condition	Reward
Tardiness ≤ 0	1	Tardiness ≤ 0	1
0 < Tardiness ≤ 250	-1	0 < Tardiness ≤ 150	-1
250 < Tardiness ≤ 350	-2	150 < Tardiness ≤ 250	-2
350 < Tardiness ≤ 450	-3	250 < Tardiness ≤ 350	-3
450 < Tardiness ≤ 600	-4	350 < Tardiness ≤ 450	-4
600 < Tardiness ≤ 750	-5	450 < Tardiness ≤ 600	-5
750 < Tardiness ≤ 850	-6	600 < Tardiness ≤ 750	-6
850 < Tardiness ≤ 950	-7	750 < Tardiness ≤ 850	-7
950 < Tardiness ≤ 1100	-8	850 < Tardiness ≤ 1000	-8
1100 < Tardiness	-9	1000 < Tardiness	-9

Table E.6 Reward function for 95% utilization level and *k*=5, 6

Condition	Reward	Condition	Reward
Tardiness ≤ 0	1	Tardiness ≤ 0	1
0 < Tardiness ≤ 100	-1	0 < Tardiness ≤ 50	-1
100 < Tardiness ≤ 150	-2	50 < Tardiness ≤ 100	-2
150 < Tardiness ≤ 200	-3	100 < Tardiness ≤ 150	-3
200 < Tardiness ≤ 300	-4	150 < Tardiness ≤ 250	-4
300 < Tardiness ≤ 350	-5	250 < Tardiness ≤ 300	-5
350 < Tardiness ≤ 450	-6	300 < Tardiness ≤ 350	-6
450 < Tardiness ≤ 550	-7	350 < Tardiness ≤ 450	-7
550 < Tardiness ≤ 700	-8	450 < Tardiness ≤ 550	-8
700 < Tardiness	-9	550 < Tardiness	-9

APPENDIX F: Reward Functions for Minimizing Mean Flow Time on Flow Shop Problem

Table F.1 Reward function for 85% utilization level and *k*=3, 4

Condition	Reward	Condition	Reward
Flow Time ≤ 390	1	Flow Time ≤ 380	1
390 < Flow Time ≤ 440	-1	380 < Flow Time ≤ 430	-1
440 < Flow Time ≤ 500	-2	430 < Flow Time ≤ 460	-2
500 < Flow Time ≤ 550	-3	460 < Flow Time ≤ 500	-3
550 < Flow Time ≤ 650	-4	500 < Flow Time ≤ 550	-4
650 < Flow Time ≤ 750	-5	550 < Flow Time ≤ 620	-5
750 < Flow Time ≤ 850	-6	620 < Flow Time ≤ 730	-6
850 < Flow Time ≤ 950	-7	730 < Flow Time ≤ 900	-7
950 < Flow Time ≤1050	-8	900 < Flow Time ≤1000	-8
1050 < Flow Time	-9	1000 < Flow Time	-9

Table F.2 Reward function for 85% utilization level and *k*=5, 6

Condition	Reward	Condition	Reward
Flow Time ≤ 370	1	Flow Time ≤ 370	1
370 < Flow Time ≤ 410	-1	370 < Flow Time ≤ 400	-1
410 < Flow Time ≤ 440	-2	400 < Flow Time ≤ 430	-2
440 < Flow Time ≤ 480	-3	430 < Flow Time ≤ 470	-3
480 < Flow Time ≤ 530	-4	470 < Flow Time ≤ 520	-4
530 < Flow Time ≤ 600	-5	520 < Flow Time ≤ 590	-5
600 < Flow Time ≤ 710	-6	590 < Flow Time ≤ 700	-6
710 < Flow Time ≤ 800	-7	700 < Flow Time ≤ 920	-7
800 < Flow Time ≤950	-8	920 < Flow Time ≤1000	-8
950 < Flow Time	-9	1000 < Flow Time	-9

Table F.3 Reward function for 90% utilization level and *k*=4

Condition	Reward
Flow Time ≤ 400	1
400 < Flow Time ≤ 480	-1
480 < Flow Time ≤ 530	-2
530 < Flow Time ≤ 620	-3
620 < Flow Time ≤ 700	-4
700 < Flow Time ≤ 800	-5
800 < Flow Time ≤ 900	-6
900 < Flow Time ≤ 1000	-7
1000 < Flow Time ≤1200	-8
1200 < Flow Time	-9

Table F.4 Reward function for 90% utilization level and *k*=5, 6

Condition	Reward	Condition	Reward
Flow Time ≤ 390	1	Flow Time ≤ 390	1
390 < Flow Time ≤ 480	-1	390 < Flow Time ≤ 480	-1
480 < Flow Time ≤ 530	-2	480 < Flow Time ≤ 530	-2
530 < Flow Time ≤ 620	-3	530 < Flow Time ≤ 620	-3
620 < Flow Time ≤ 690	-4	620 < Flow Time ≤ 690	-4
690 < Flow Time ≤ 780	-5	690 < Flow Time ≤ 780	-5
780 < Flow Time ≤ 900	-6	780 < Flow Time ≤ 900	-6
900 < Flow Time ≤ 1000	-7	900 < Flow Time ≤ 1000	-7
1000 < Flow Time ≤1150	-8	1000 < Flow Time ≤1150	-8
1150 < Flow Time	-9	1150 < Flow Time	-9

Table F.5 Reward function for 95% utilization level and *k*=3, 4

Condition	Reward	Condition	Reward
Flow Time ≤ 450	1	Flow Time ≤ 450	1
450 < Flow Time ≤ 550	-1	450 < Flow Time ≤ 550	-1
550 < Flow Time ≤ 650	-2	550 < Flow Time ≤ 650	-2
650 < Flow Time ≤ 750	-3	650 < Flow Time ≤ 750	-3
750 < Flow Time ≤ 850	-4	750 < Flow Time ≤ 850	-4
850 < Flow Time ≤ 950	-5	850 < Flow Time ≤ 950	-5
950 < Flow Time ≤1100	-6	950 < Flow Time ≤1100	-6
1100 < Flow Time ≤ 1250	-7	1100 < Flow Time ≤ 1250	-7
1250 < Flow Time ≤1650	-8	1250 < Flow Time ≤1650	-8
1650 < Flow Time	-9	1650 < Flow Time	-9

Table F.6 Reward function for 95% utilization level and *k*=5, 6

Condition	Reward	Condition	Reward
Flow Time ≤ 450	1	Flow Time ≤ 450	1
450 < Flow Time ≤ 550	-1	450 < Flow Time ≤ 550	-1
550 < Flow Time ≤ 630	-2	550 < Flow Time ≤ 630	-2
630 < Flow Time ≤ 740	-3	630 < Flow Time ≤ 740	-3
740 < Flow Time ≤ 860	-4	740 < Flow Time ≤ 860	-4
860 < Flow Time ≤ 940	-5	860 < Flow Time ≤ 940	-5
940 < Flow Time ≤1100	-6	940 < Flow Time ≤1100	-6
1100 < Flow Time ≤ 1250	-7	1100 < Flow Time ≤ 1250	-7
1250 < Flow Time ≤1650	-8	1250 < Flow Time ≤1650	-8
1650 < Flow Time	-9	1650 < Flow Time	-9

APPENDIX G: Reward Functions for Dynamic Flow Shop Bicriteria Scheduling Problem

Table G.1 Reward function for 85% utilization level and k=3

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 390	1
0 < Tardiness ≤ 20	-1	390 < Flow Time ≤ 440	-1
20 < Tardiness ≤ 40	-2	440 < Flow Time ≤ 500	-2
40 < Tardiness ≤ 70	-3	500 < Flow Time ≤ 550	-3
70 < Tardiness ≤ 90	-4	550 < Flow Time ≤ 650	-4
90 < Tardiness ≤ 110	-5	650 < Flow Time ≤ 750	-5
110 < Tardiness ≤ 150	-6	750 < Flow Time ≤ 850	-6
150 < Tardiness ≤ 190	-7	850 < Flow Time ≤ 950	-7
190 < Tardiness ≤ 260	-8	950 < Flow Time ≤1050	-8
260 < Tardiness	-9	1050 < Flow Time	-9

Table G.2 Reward function for 85% utilization level and k=4

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 390	1
0 < Tardiness ≤ 10	-1	390 < Flow Time ≤ 440	-1
10 < Tardiness ≤ 30	-2	440 < Flow Time ≤ 500	-2
30 < Tardiness ≤ 50	-3	500 < Flow Time ≤ 550	-3
50 < Tardiness ≤ 70	-4	550 < Flow Time ≤ 650	-4
70 < Tardiness ≤ 90	-5	650 < Flow Time ≤ 750	-5
90 < Tardiness ≤ 110	-6	750 < Flow Time ≤ 850	-6
110 < Tardiness ≤ 130	-7	850 < Flow Time ≤ 950	-7
130 < Tardiness ≤ 150	-8	950 < Flow Time ≤1050	-8
150 < Tardiness	-9	1050 < Flow Time	-9

Table G.3 Reward function for 85% utilization level and k=5

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 370	1
0 < Tardiness ≤ 7	-1	370 < Flow Time ≤ 410	-1
7 < Tardiness ≤ 20	-2	410 < Flow Time ≤ 440	-2
20 < Tardiness ≤ 35	-3	440 < Flow Time ≤ 480	-3
35 < Tardiness ≤ 55	-4	480 < Flow Time ≤ 530	-4
55 < Tardiness ≤ 70	-5	530 < Flow Time ≤ 600	-5
70 < Tardiness ≤ 95	-6	600 < Flow Time ≤ 710	-6
95 < Tardiness ≤ 115	-7	710 < Flow Time ≤ 800	-7
115 < Tardiness ≤ 150	-8	800 < Flow Time ≤950	-8
150 < Tardiness	-9	950 < Flow Time	-9

Table G.4 Reward function for 85% utilization level and *k*=6

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 370	1
0 < Tardiness ≤ 5	-1	370 < Flow Time ≤ 400	-1
5 < Tardiness ≤ 15	-2	400 < Flow Time ≤ 430	-2
15 < Tardiness ≤ 28	-3	430 < Flow Time ≤ 470	-3
28 < Tardiness ≤ 45	-4	470 < Flow Time ≤ 520	-4
45 < Tardiness ≤ 63	-5	520 < Flow Time ≤ 590	-5
63 < Tardiness ≤ 87	-6	590 < Flow Time ≤ 700	-6
87 < Tardiness ≤ 110	-7	700 < Flow Time ≤ 920	-7
110 < Tardiness ≤ 150	-8	920 < Flow Time ≤1000	-8
150 < Tardiness	-9	1000 < Flow Time	-9

Table G.5 Reward function for 90% utilization level and *k*=4

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 400	1
0 < Tardiness ≤ 20	-1	400 < Flow Time ≤ 480	-1
20 < Tardiness ≤ 40	-2	480 < Flow Time ≤ 530	-2
40 < Tardiness ≤ 60	-3	530 < Flow Time ≤ 620	-3
60 < Tardiness ≤ 80	-4	620 < Flow Time ≤ 700	-4
80 < Tardiness ≤ 120	-5	700 < Flow Time ≤ 800	-5
120 < Tardiness ≤ 180	-6	800 < Flow Time ≤ 900	-6
180 < Tardiness ≤ 300	-7	900 < Flow Time ≤ 1000	-7
300 < Tardiness ≤ 580	-8	1000 < Flow Time ≤1200	-8
580 < Tardiness	-9	1200 < Flow Time	-9

Table G.6 Reward function for 90% utilization level and *k*=5

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 390	1
0 < Tardiness ≤ 15	-1	390 < Flow Time ≤ 480	-1
15 < Tardiness ≤ 35	-2	480 < Flow Time ≤ 530	-2
35 < Tardiness ≤ 45	-3	530 < Flow Time ≤ 620	-3
45 < Tardiness ≤ 65	-4	620 < Flow Time ≤ 690	-4
65 < Tardiness ≤ 100	-5	690 < Flow Time ≤ 780	-5
100 < Tardiness ≤ 150	-6	780 < Flow Time ≤ 900	-6
150 < Tardiness ≤ 270	-7	900 < Flow Time ≤ 1000	-7
270 < Tardiness ≤ 550	-8	1000 < Flow Time ≤1150	-8
550 < Tardiness	-9	1150 < Flow Time	-9

Table G.7 Reward function for 90% utilization level and *k*=6

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 390	1
0 < Tardiness ≤ 10	-1	390 < Flow Time ≤ 480	-1
10 < Tardiness ≤ 25	-2	480 < Flow Time ≤ 530	-2
25 < Tardiness ≤ 37	-3	530 < Flow Time ≤ 620	-3
37 < Tardiness ≤ 50	-4	620 < Flow Time ≤ 690	-4
50 < Tardiness ≤ 70	-5	690 < Flow Time ≤ 780	-5
70 < Tardiness ≤ 105	-6	780 < Flow Time ≤ 900	-6
105 < Tardiness ≤ 180	-7	900 < Flow Time ≤ 1000	-7
180 < Tardiness ≤ 400	-8	1000 < Flow Time ≤1150	-8
400 < Tardiness	-9	1150 < Flow Time	-9

Table G.8 Reward function for 95% utilization level and *k*=3

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 450	1
0 < Tardiness ≤ 40	-1	450 < Flow Time ≤ 550	-1
40 < Tardiness ≤ 70	-2	550 < Flow Time ≤ 630	-2
70 < Tardiness ≤ 100	-3	630 < Flow Time ≤ 740	-3
100 < Tardiness ≤ 135	-4	740 < Flow Time ≤ 860	-4
135 < Tardiness ≤ 180	-5	860 < Flow Time ≤ 940	-5
180 < Tardiness ≤ 260	-6	940 < Flow Time ≤1100	-6
260 < Tardiness ≤ 450	-7	1100 < Flow Time ≤ 1250	-7
450 < Tardiness ≤ 800	-8	1250 < Flow Time ≤1650	-8
800 < Tardiness	-9	1650 < Flow Time	-9

Table G.9 Reward function for 95% utilization level and *k*=4

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 450	1
0 < Tardiness ≤ 30	-1	450 < Flow Time ≤ 550	-1
30 < Tardiness ≤ 60	-2	550 < Flow Time ≤ 650	-2
60 < Tardiness ≤ 80	-3	650 < Flow Time ≤ 750	-3
80 < Tardiness ≤110	-4	750 < Flow Time ≤ 850	-4
110 < Tardiness ≤ 155	-5	850 < Flow Time ≤ 950	-5
155 < Tardiness ≤ 240	-6	950 < Flow Time ≤1100	-6
240 < Tardiness ≤ 420	-7	1100 < Flow Time ≤ 1250	-7
420 < Tardiness ≤ 760	-8	1250 < Flow Time ≤1650	-8
760 < Tardiness	-9	1650 < Flow Time	-9

Table G.10 Reward function for 95% utilization level and k=5

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 450	1
0 < Tardiness ≤ 20	-1	450 < Flow Time ≤ 550	-1
20 < Tardiness ≤ 40	-2	550 < Flow Time ≤ 650	-2
40 < Tardiness ≤ 60	-3	650 < Flow Time ≤ 750	-3
60 < Tardiness ≤ 90	-4	750 < Flow Time ≤ 850	-4
90 < Tardiness ≤ 130	-5	850 < Flow Time ≤ 950	-5
130 < Tardiness ≤ 210	-6	950 < Flow Time ≤1100	-6
210 < Tardiness ≤ 390	-7	1100 < Flow Time ≤ 1250	-7
390 < Tardiness ≤ 730	-8	1250 < Flow Time ≤1650	-8
730 < Tardiness	-9	1650 < Flow Time	-9

Table G.11 Reward function for 95% utilization level and k=6

Minimizing *MT*		**Minimizing *MF***	
Condition	**Reward**	**Condition**	**Reward**
Tardiness ≤ 0	1	Flow Time ≤ 450	1
0 < Tardiness ≤ 20	-1	450 < Flow Time ≤ 550	-1
20 < Tardiness ≤ 40	-2	550 < Flow Time ≤ 650	-2
40 < Tardiness ≤ 70	-3	650 < Flow Time ≤ 750	-3
70 < Tardiness ≤100	-4	750 < Flow Time ≤ 850	-4
100 < Tardiness ≤ 140	-5	850 < Flow Time ≤ 950	-5
140 < Tardiness ≤ 180	-6	950 < Flow Time ≤1100	-6
180 < Tardiness ≤ 280	-7	1100 < Flow Time ≤ 1250	-7
280 < Tardiness ≤ 650	-8	1250 < Flow Time ≤1650	-8
650 < Tardiness	-9	1650 < Flow Time	-9

APPENDIX H: Multi State Tables for Flow Shop Applications

Table H.1 State table for 85% utilization level and k=3

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and $ASL < 0$	$N = 2$
2	$N > 1$ and $ASL < 0$	$N = 3$
3	$N > 1$ and $ASL < 0$	$N = 4$
4	$N > 1$ and $ASL < 0$	$N > 4$
5	$N > 1$ and $0 \leq ASL < 100$	$N = 2$
6	$N > 1$ and $0 \leq ASL < 100$	$N = 3$
7	$N > 1$ and $0 \leq ASL < 100$	$N = 4$
8	$N > 1$ and $0 \leq ASL < 100$	$N > 4$
9	$N > 1$ and $100 \leq ASL < 200$	$N = 2$
10	$N > 1$ and $100 \leq ASL < 200$	$N = 3$
11	$N > 1$ and $100 \leq ASL < 200$	$N = 4$
12	$N > 1$ and $100 \leq ASL < 200$	$N > 4$
13	$N > 1$ and $200 \leq ASL < 300$	$N = 2$
14	$N > 1$ and $200 \leq ASL < 300$	$N = 3$
15	$N > 1$ and $200 \leq ASL < 300$	$N = 4$
16	$N > 1$ and $200 \leq ASL < 300$	$N > 4$
17	$N > 1$ and $300 \leq ASL < 350$	$N = 2$
18	$N > 1$ and $300 \leq ASL < 350$	$N = 3$
19	$N > 1$ and $300 \leq ASL < 350$	$N = 4$
20	$N > 1$ and $300 \leq ASL < 350$	$N > 4$
21	$N > 1$ and $350 \leq ASL < 400$	$N = 2$
22	$N > 1$ and $350 \leq ASL < 400$	$N = 3$
23	$N > 1$ and $350 \leq ASL < 400$	$N = 4$
24	$N > 1$ and $350 \leq ASL < 400$	$N > 4$
25	$N > 1$ and $400 \leq ASL < 450$	$N = 2$
26	$N > 1$ and $400 \leq ASL < 450$	$N = 3$
27	$N > 1$ and $400 \leq ASL < 450$	$N = 4$
28	$N > 1$ and $400 \leq ASL < 450$	$N > 4$
29	$N > 1$ and $450 \leq ASL < 500$	$N = 2$
30	$N > 1$ and $450 \leq ASL < 500$	$N = 3$
31	$N > 1$ and $450 \leq ASL < 500$	$N = 4$
32	$N > 1$ and $450 \leq ASL < 500$	$N > 4$
33	$N > 1$ and $500 \leq ASL < 560$	$N = 2$
34	$N > 1$ and $500 \leq ASL < 560$	$N = 3$
35	$N > 1$ and $500 \leq ASL < 560$	$N = 4$
36	$N > 1$ and $500 \leq ASL < 560$	$N > 4$
37	$N > 1$ and $560 \leq ASL$	$N = 2$
38	$N > 1$ and $560 \leq ASL$	$N = 3$
39	$N > 1$ and $560 \leq ASL$	$N = 4$
40	$N > 1$ and $560 \leq ASL$	$N > 4$

Table H.2 State table for 85% utilization level and k=4

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and $ASL < 300$	$N = 2$
2	$N > 1$ and $ASL < 300$	$N = 3$
3	$N > 1$ and $ASL < 300$	$N = 4$
4	$N > 1$ and $ASL < 300$	$N > 4$
5	$N > 1$ and $300 \leq ASL < 450$	$N = 2$
6	$N > 1$ and $300 \leq ASL < 450$	$N = 3$
7	$N > 1$ and $300 \leq ASL < 450$	$N = 4$
8	$N > 1$ and $300 \leq ASL < 450$	$N > 4$
9	$N > 1$ and $450 \leq ASL < 530$	$N = 2$
10	$N > 1$ and $450 \leq ASL < 530$	$N = 3$
11	$N > 1$ and $450 \leq ASL < 530$	$N = 4$
12	$N > 1$ and $450 \leq ASL < 530$	$N > 4$
13	$N > 1$ and $530 \leq ASL < 600$	$N = 2$
14	$N > 1$ and $530 \leq ASL < 600$	$N = 3$
15	$N > 1$ and $530 \leq ASL < 600$	$N = 4$
16	$N > 1$ and $530 \leq ASL < 600$	$N > 4$
17	$N > 1$ and $600 \leq ASL < 650$	$N = 2$
18	$N > 1$ and $600 \leq ASL < 650$	$N = 3$
19	$N > 1$ and $600 \leq ASL < 650$	$N = 4$
20	$N > 1$ and $600 \leq ASL < 650$	$N > 4$
21	$N > 1$ and $650 \leq ASL < 700$	$N = 2$
22	$N > 1$ and $650 \leq ASL < 700$	$N = 3$
23	$N > 1$ and $650 \leq ASL < 700$	$N = 4$
24	$N > 1$ and $650 \leq ASL < 700$	$N > 4$
25	$N > 1$ and $700 \leq ASL < 750$	$N = 2$
26	$N > 1$ and $700 \leq ASL < 750$	$N = 3$
27	$N > 1$ and $700 \leq ASL < 750$	$N = 4$
28	$N > 1$ and $700 \leq ASL < 750$	$N > 4$
29	$N > 1$ and $750 \leq ASL < 800$	$N = 2$
30	$N > 1$ and $750 \leq ASL < 800$	$N = 3$
31	$N > 1$ and $750 \leq ASL < 800$	$N = 4$
32	$N > 1$ and $750 \leq ASL < 800$	$N > 4$
33	$N > 1$ and $800 \leq ASL < 900$	$N = 2$
34	$N > 1$ and $800 \leq ASL < 900$	$N = 3$
35	$N > 1$ and $800 \leq ASL < 900$	$N = 4$
36	$N > 1$ and $800 \leq ASL < 900$	$N > 4$
37	$N > 1$ and $900 \leq ASL$	$N = 2$
38	$N > 1$ and $900 \leq ASL$	$N = 3$
39	$N > 1$ and $900 \leq ASL$	$N = 4$
40	$N > 1$ and $900 \leq ASL$	$N > 4$

Table H.3 State table for 85% utilization level and k=5

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and ASL <55 0	$N = 2$
2	$N > 1$ and ASL <55 0	$N = 3$
3	$N > 1$ and ASL <55 0	$N = 4$
4	$N > 1$ and ASL <55 0	$N > 4$
5	$N > 1$ and $550 \leq ASL < 700$	$N = 2$
6	$N > 1$ and $550 \leq ASL < 700$	$N = 3$
7	$N > 1$ and $550 \leq ASL < 700$	$N = 4$
8	$N > 1$ and $550 \leq ASL < 700$	$N > 4$
9	$N > 1$ and $700 \leq ASL < 800$	$N = 2$
10	$N > 1$ and $700 \leq ASL < 800$	$N = 3$
11	$N > 1$ and $700 \leq ASL < 800$	$N = 4$
12	$N > 1$ and $700 \leq ASL < 800$	$N > 4$
13	$N > 1$ and $800 \leq ASL < 870$	$N = 2$
14	$N > 1$ and $800 \leq ASL < 870$	$N = 3$
15	$N > 1$ and $800 \leq ASL < 870$	$N = 4$
16	$N > 1$ and $800 \leq ASL < 870$	$N > 4$
17	$N > 1$ and $870 \leq ASL < 950$	$N = 2$
18	$N > 1$ and $870 \leq ASL < 950$	$N = 3$
19	$N > 1$ and $870 \leq ASL < 950$	$N = 4$
20	$N > 1$ and $870 \leq ASL < 950$	$N > 4$
21	$N >1$ and $950 \leq ASL < 1020$	$N = 2$
22	$N >1$ and $950 \leq ASL < 1020$	$N = 3$
23	$N >1$ and $950 \leq ASL < 1020$	$N = 4$
24	$N >1$ and $950 \leq ASL < 1020$	$N > 4$
25	$N > 1$ and $1020 \leq ASL < 1080$	$N = 2$
26	$N > 1$ and $1020 \leq ASL < 1080$	$N = 3$
27	$N > 1$ and $1020 \leq ASL < 1080$	$N = 4$
28	$N > 1$ and $1020 \leq ASL < 1080$	$N > 4$
29	$N > 1$ and $1080 \leq ASL < 1150$	$N = 2$
30	$N > 1$ and $1080 \leq ASL < 1150$	$N = 3$
31	$N > 1$ and $1080 \leq ASL < 1150$	$N = 4$
32	$N > 1$ and $1080 \leq ASL < 1150$	$N > 4$
33	$N > 1$ and $1150 \leq ASL < 1250$	$N = 2$
34	$N > 1$ and $1150 \leq ASL < 1250$	$N = 3$
35	$N > 1$ and $1150 \leq ASL < 1250$	$N = 4$
36	$N > 1$ and $1150 \leq ASL < 1250$	$N > 4$
37	$N > 1$ and $1250 \leq ASL$	$N = 2$
38	$N > 1$ and $1250 \leq ASL$	$N = 3$
39	$N > 1$ and $1250 \leq ASL$	$N = 4$
40	$N > 1$ and $1250 \leq ASL$	$N > 4$

Table H.4 State table for 85% utilization level and k=6

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and $ASL < 800$	$N = 2$
2	$N > 1$ and $ASL < 800$	$N = 3$
3	$N > 1$ and $ASL < 800$	$N = 4$
4	$N > 1$ and $ASL < 800$	$N > 4$
5	$N > 1$ and $800 \leq ASL < 950$	$N = 2$
6	$N > 1$ and $800 \leq ASL < 950$	$N = 3$
7	$N > 1$ and $800 \leq ASL < 950$	$N = 4$
8	$N > 1$ and $800 \leq ASL < 950$	$N > 4$
9	$N > 1$ and $950 \leq ASL < 1080$	$N = 2$
10	$N > 1$ and $950 \leq ASL < 1080$	$N = 3$
11	$N > 1$ and $950 \leq ASL < 1080$	$N = 4$
12	$N > 1$ and $950 \leq ASL < 1080$	$N > 4$
13	$N > 1$ and $1080 \leq ASL < 1150$	$N = 2$
14	$N > 1$ and $1080 \leq ASL < 1150$	$N = 3$
15	$N > 1$ and $1080 \leq ASL < 1150$	$N = 4$
16	$N > 1$ and $1080 \leq ASL < 1150$	$N > 4$
17	$N > 1$ and $1150 \leq ASL < 1220$	$N = 2$
18	$N > 1$ and $1150 \leq ASL < 1220$	$N = 3$
19	$N > 1$ and $1150 \leq ASL < 1220$	$N = 4$
20	$N > 1$ and $1150 \leq ASL < 1220$	$N > 4$
21	$N > 1$ and $1220 \leq ASL < 1300$	$N = 2$
22	$N > 1$ and $1220 \leq ASL < 1300$	$N = 3$
23	$N > 1$ and $1220 \leq ASL < 1300$	$N = 4$
24	$N > 1$ and $1220 \leq ASL < 1300$	$N > 4$
25	$N > 1$ and $1300 \leq ASL < 1380$	$N = 2$
26	$N > 1$ and $1300 \leq ASL < 1380$	$N = 3$
27	$N > 1$ and $1300 \leq ASL < 1380$	$N = 4$
28	$N > 1$ and $1300 \leq ASL < 1380$	$N > 4$
29	$N > 1$ and $1380 \leq ASL < 1470$	$N = 2$
30	$N > 1$ and $1380 \leq ASL < 1470$	$N = 3$
31	$N > 1$ and $1380 \leq ASL < 1470$	$N = 4$
32	$N > 1$ and $1380 \leq ASL < 1470$	$N > 4$
33	$N > 1$ and $1470 \leq ASL < 1580$	$N = 2$
34	$N > 1$ and $1470 \leq ASL < 1580$	$N = 3$
35	$N > 1$ and $1470 \leq ASL < 1580$	$N = 4$
36	$N > 1$ and $1470 \leq ASL < 1580$	$N > 4$
37	$N > 1$ and $1580 \leq ASL$	$N = 2$
38	$N > 1$ and $1580 \leq ASL$	$N = 3$
39	$N > 1$ and $1580 \leq ASL$	$N = 4$
40	$N > 1$ and $1580 \leq ASL$	$N > 4$

Table H.5 State table for 90% utilization level and k=4

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and $ASL < 20$	$N = 2$
2	$N > 1$ and $ASL < 20$	$3 \leq N < 5$
3	$N > 1$ and $ASL < 20$	$5 \leq N < 7$
4	$N > 1$ and $ASL < 20$	$7 \leq N$
5	$N > 1$ and $20 \leq ASL < 190$	$N = 2$
6	$N > 1$ and $20 \leq ASL < 190$	$3 \leq N < 5$
7	$N > 1$ and $20 \leq ASL < 190$	$5 \leq N < 7$
8	$N > 1$ and $20 \leq ASL < 190$	$7 \leq N$
9	$N > 1$ and $190 \leq ASL < 340$	$N = 2$
10	$N > 1$ and $190 \leq ASL < 340$	$3 \leq N < 5$
11	$N > 1$ and $190 \leq ASL < 340$	$5 \leq N < 7$
12	$N > 1$ and $190 \leq ASL < 340$	$7 \leq N$
13	$N > 1$ and $340 \leq ASL < 460$	$N = 2$
14	$N > 1$ and $340 \leq ASL < 460$	$3 \leq N < 5$
15	$N > 1$ and $340 \leq ASL < 460$	$5 \leq N < 7$
16	$N > 1$ and $340 \leq ASL < 460$	$7 \leq N$
17	$N > 1$ and $460 \leq ASL < 540$	$N = 2$
18	$N > 1$ and $460 \leq ASL < 540$	$3 \leq N < 5$
19	$N > 1$ and $460 \leq ASL < 540$	$5 \leq N < 7$
20	$N > 1$ and $460 \leq ASL < 540$	$7 \leq N$
21	$N > 1$ and $540 \leq ASL < 610$	$N = 2$
22	$N > 1$ and $540 \leq ASL < 610$	$3 \leq N < 5$
23	$N > 1$ and $540 \leq ASL < 610$	$5 \leq N < 7$
24	$N > 1$ and $540 \leq ASL < 610$	$7 \leq N$
25	$N > 1$ and $610 \leq ASL < 680$	$N = 2$
26	$N > 1$ and $610 \leq ASL < 680$	$3 \leq N < 5$
27	$N > 1$ and $610 \leq ASL < 680$	$5 \leq N < 7$
28	$N > 1$ and $610 \leq ASL < 680$	$7 \leq N$
29	$N > 1$ and $680 \leq ASL < 750$	$N = 2$
30	$N > 1$ and $680 \leq ASL < 750$	$3 \leq N < 5$
31	$N > 1$ and $680 \leq ASL < 750$	$5 \leq N < 7$
32	$N > 1$ and $680 \leq ASL < 750$	$7 \leq N$
33	$N > 1$ and $750 \leq ASL < 850$	$N = 2$
34	$N > 1$ and $750 \leq ASL < 850$	$3 \leq N < 5$
35	$N > 1$ and $750 \leq ASL < 850$	$5 \leq N < 7$
36	$N > 1$ and $750 \leq ASL < 850$	$7 \leq N$
37	$N > 1$ and $850 \leq ASL$	$N = 2$
38	$N > 1$ and $850 \leq ASL$	$3 \leq N < 5$
39	$N > 1$ and $850 \leq ASL$	$5 \leq N < 7$
40	$N > 1$ and $850 \leq ASL$	$7 \leq N$

Table H.6 State table for 90% utilization level and k=5

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and $ASL < 300$	$N = 2$
2	$N > 1$ and $ASL < 300$	$3 \leq N < 5$
3	$N > 1$ and $ASL < 300$	$5 \leq N < 7$
4	$N > 1$ and $ASL < 300$	$7 \leq N$
5	$N > 1$ and $300 \leq ASL < 550$	$N = 2$
6	$N > 1$ and $300 \leq ASL < 550$	$3 \leq N < 5$
7	$N > 1$ and $300 \leq ASL < 550$	$5 \leq N < 7$
8	$N > 1$ and $300 \leq ASL < 550$	$7 \leq N$
9	$N > 1$ and $550 \leq ASL < 700$	$N = 2$
10	$N > 1$ and $550 \leq ASL < 700$	$3 \leq N < 5$
11	$N > 1$ and $550 \leq ASL < 700$	$5 \leq N < 7$
12	$N > 1$ and $550 \leq ASL < 700$	$7 \leq N$
13	$N > 1$ and $700 \leq ASL < 800$	$N = 2$
14	$N > 1$ and $700 \leq ASL < 800$	$3 \leq N < 5$
15	$N > 1$ and $700 \leq ASL < 800$	$5 \leq N < 7$
16	$N > 1$ and $700 \leq ASL < 800$	$7 \leq N$
17	$N > 1$ and $800 \leq ASL < 870$	$N = 2$
18	$N > 1$ and $800 \leq ASL < 870$	$3 \leq N < 5$
19	$N > 1$ and $800 \leq ASL < 870$	$5 \leq N < 7$
20	$N > 1$ and $800 \leq ASL < 870$	$7 \leq N$
21	$N > 1$ and $870 \leq ASL < 950$	$N = 2$
22	$N > 1$ and $870 \leq ASL < 950$	$3 \leq N < 5$
23	$N > 1$ and $870 \leq ASL < 950$	$5 \leq N < 7$
24	$N > 1$ and $870 \leq ASL < 950$	$7 \leq N$
25	$N > 1$ and $950 \leq ASL < 1020$	$N = 2$
26	$N > 1$ and $950 \leq ASL < 1020$	$3 \leq N < 5$
27	$N > 1$ and $950 \leq ASL < 1020$	$5 \leq N < 7$
28	$N > 1$ and $950 \leq ASL < 1020$	$7 \leq N$
29	$N > 1$ and $1020 \leq ASL < 1100$	$N = 2$
30	$N > 1$ and $1020 \leq ASL < 1100$	$3 \leq N < 5$
31	$N > 1$ and $1020 \leq ASL < 1100$	$5 \leq N < 7$
32	$N > 1$ and $1020 \leq ASL < 1100$	$7 \leq N$
33	$N > 1$ and $1100 \leq ASL < 1200$	$N = 2$
34	$N > 1$ and $1100 \leq ASL < 1200$	$3 \leq N < 5$
35	$N > 1$ and $1100 \leq ASL < 1200$	$5 \leq N < 7$
36	$N > 1$ and $1100 \leq ASL < 1200$	$7 \leq N$
37	$N > 1$ and $1200 \leq ASL$	$N = 2$
38	$N > 1$ and $1200 \leq ASL$	$3 \leq N < 5$
39	$N > 1$ and $1200 \leq ASL$	$5 \leq N < 7$
40	$N > 1$ and $1200 \leq ASL$	$7 \leq N$

Table H.7 State table for 90% utilization level and k=6

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and $ASL < 650$	$N = 2$
2	$N > 1$ and $ASL < 650$	$3 \leq N < 5$
3	$N > 1$ and $ASL < 650$	$5 \leq N < 7$
4	$N > 1$ and $ASL < 650$	$7 \leq N$
5	$N > 1$ and $650 \leq ASL < 850$	$N = 2$
6	$N > 1$ and $650 \leq ASL < 850$	$3 \leq N < 5$
7	$N > 1$ and $650 \leq ASL < 850$	$5 \leq N < 7$
8	$N > 1$ and $650 \leq ASL < 850$	$7 \leq N$
9	$N > 1$ and $850 \leq ASL < 1000$	$N = 2$
10	$N > 1$ and $850 \leq ASL < 1000$	$3 \leq N < 5$
11	$N > 1$ and $850 \leq ASL < 1000$	$5 \leq N < 7$
12	$N > 1$ and $850 \leq ASL < 1000$	$7 \leq N$
13	$N > 1$ and $1000 \leq ASL < 1100$	$N = 2$
14	$N > 1$ and $1000 \leq ASL < 1100$	$3 \leq N < 5$
15	$N > 1$ and $1000 \leq ASL < 1100$	$5 \leq N < 7$
16	$N > 1$ and $1000 \leq ASL < 1100$	$7 \leq N$
17	$N > 1$ and $1100 \leq ASL < 1170$	$N = 2$
18	$N > 1$ and $1100 \leq ASL < 1170$	$3 \leq N < 5$
19	$N > 1$ and $1100 \leq ASL < 1170$	$5 \leq N < 7$
20	$N > 1$ and $1100 \leq ASL < 1170$	$7 \leq N$
21	$N > 1$ and $1170 \leq ASL < 1250$	$N = 2$
22	$N > 1$ and $1170 \leq ASL < 1250$	$3 \leq N < 5$
23	$N > 1$ and $1170 \leq ASL < 1250$	$5 \leq N < 7$
24	$N > 1$ and $1170 \leq ASL < 1250$	$7 \leq N$
25	$N > 1$ and $1250 \leq ASL < 1320$	$N = 2$
26	$N > 1$ and $1250 \leq ASL < 1320$	$3 \leq N < 5$
27	$N > 1$ and $1250 \leq ASL < 1320$	$5 \leq N < 7$
28	$N > 1$ and $1250 \leq ASL < 1320$	$7 \leq N$
29	$N > 1$ and $1320 \leq ASL < 1400$	$N = 2$
30	$N > 1$ and $1320 \leq ASL < 1400$	$3 \leq N < 5$
31	$N > 1$ and $1320 \leq ASL < 1400$	$5 \leq N < 7$
32	$N > 1$ and $1320 \leq ASL < 1400$	$7 \leq N$
33	$N > 1$ and $1400 \leq ASL < 1550$	$N = 2$
34	$N > 1$ and $1400 \leq ASL < 1550$	$3 \leq N < 5$
35	$N > 1$ and $1400 \leq ASL < 1550$	$5 \leq N < 7$
36	$N > 1$ and $1400 \leq ASL < 1550$	$7 \leq N$
37	$N > 1$ and $1550 \leq ASL$	$N = 2$
38	$N > 1$ and $1550 \leq ASL$	$3 \leq N < 5$
39	$N > 1$ and $1550 \leq ASL$	$5 \leq N < 7$
40	$N > 1$ and $1550 \leq ASL$	$7 \leq N$

Table H.8 State table for 95% utilization level and k=3

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and ASL <-2200	$2 \leq N < 4$
2	$N > 1$ and ASL <-2200	$4 \leq N < 7$
3	$N > 1$ and ASL <-2200	$7 \leq N <9$
4	$N > 1$ and ASL <-2200	$9 \leq N$
5	$N > 1$ and -2200 $\leq ASL <$ -1050	$2 \leq N < 4$
6	$N > 1$ and -2200 $\leq ASL <$ -1050	$4 \leq N < 7$
7	$N > 1$ and -2200 $\leq ASL <$ -1050	$7 \leq N <9$
8	$N > 1$ and -2200 $\leq ASL <$ -1050	$9 \leq N$
9	$N > 1$ and -1050 $\leq ASL <$-550	$2 \leq N < 4$
10	$N > 1$ and -1050 $\leq ASL <$-550	$4 \leq N < 7$
11	$N > 1$ and -1050 $\leq ASL <$-550	$7 \leq N <9$
12	$N > 1$ and -1050 $\leq ASL <$-550	$9 \leq N$
13	$N > 1$ and -550 $\leq ASL <$ -250	$2 \leq N < 4$
14	$N > 1$ and -550 $\leq ASL <$ -250	$4 \leq N < 7$
15	$N > 1$ and -550 $\leq ASL <$ -250	$7 \leq N <9$
16	$N > 1$ and -550 $\leq ASL <$ -250	$9 \leq N$
17	$N > 1$ and -250 $\leq ASL <$ -80	$2 \leq N < 4$
18	$N > 1$ and -250 $\leq ASL <$ -80	$4 \leq N < 7$
19	$N > 1$ and -250 $\leq ASL <$ -80	$7 \leq N <9$
20	$N > 1$ and -250 $\leq ASL <$ -80	$9 \leq N$
21	$N > 1$ and -80 $\leq ASL <$ 40	$2 \leq N < 4$
22	$N > 1$ and -80 $\leq ASL <$ 40	$4 \leq N < 7$
23	$N > 1$ and -80 $\leq ASL <$ 40	$7 \leq N <9$
24	$N > 1$ and -80 $\leq ASL <$ 40	$9 \leq N$
25	$N > 1$ and 40 $\leq ASL <$ 200	$2 \leq N < 4$
26	$N > 1$ and 40 $\leq ASL <$ 200	$4 \leq N < 7$
27	$N > 1$ and 40 $\leq ASL <$ 200	$7 \leq N <9$
28	$N > 1$ and 40 $\leq ASL <$ 200	$9 \leq N$
29	$N > 1$ and 200 $\leq ASL <$ 350	$2 \leq N < 4$
30	$N > 1$ and 200 $\leq ASL <$ 350	$4 \leq N < 7$
31	$N > 1$ and 200 $\leq ASL <$ 350	$7 \leq N <9$
32	$N > 1$ and 200 $\leq ASL <$ 350	$9 \leq N$
33	$N > 1$ and 350 $\leq ASL <$ 450	$2 \leq N < 4$
34	$N > 1$ and 350 $\leq ASL <$ 450	$4 \leq N < 7$
35	$N > 1$ and 350 $\leq ASL <$ 450	$7 \leq N <9$
36	$N > 1$ and 350 $\leq ASL <$ 450	$9 \leq N$
37	$N > 1$ and 450 $\leq ASL$	$2 \leq N < 4$
38	$N > 1$ and 450 $\leq ASL$	$4 \leq N < 7$
39	$N > 1$ and 450 $\leq ASL$	$7 \leq N <9$
40	$N > 1$ and 450 $\leq ASL$	$9 \leq N$

Table H.9 State table for 95% utilization level and k=4

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and ASL <-1500	$2 \leq N < 4$
2	$N > 1$ and ASL <-1500	$4 \leq N < 7$
3	$N > 1$ and ASL <-1500	$7 \leq N <9$
4	$N > 1$ and ASL <-1500	$9 \leq N$
5	$N > 1$ and -1500 $\leq ASL <$ -550	$2 \leq N < 4$
6	$N > 1$ and -1500 $\leq ASL <$ -550	$4 \leq N < 7$
7	$N > 1$ and -1500 $\leq ASL <$ -550	$7 \leq N <9$
8	$N > 1$ and -1500 $\leq ASL <$ -550	$9 \leq N$
9	$N > 1$ and -550 $\leq ASL <$-150	$2 \leq N < 4$
10	$N > 1$ and -550 $\leq ASL <$-150	$4 \leq N < 7$
11	$N > 1$ and -550 $\leq ASL <$-150	$7 \leq N <9$
12	$N > 1$ and -550 $\leq ASL <$-150	$9 \leq N$
13	$N > 1$ and -150 $\leq ASL < 0$	$2 \leq N < 4$
14	$N > 1$ and -150 $\leq ASL < 0$	$4 \leq N < 7$
15	$N > 1$ and -150 $\leq ASL < 0$	$7 \leq N <9$
16	$N > 1$ and -150 $\leq ASL < 0$	$9 \leq N$
17	$N > 1$ and 0 $\leq ASL <$ 200	$2 \leq N < 4$
18	$N > 1$ and 0 $\leq ASL <$ 200	$4 \leq N < 7$
19	$N > 1$ and 0 $\leq ASL <$ 200	$7 \leq N <9$
20	$N > 1$ and 0 $\leq ASL <$ 200	$9 \leq N$
21	$N > 1$ and 200 $\leq ASL <$ 400	$2 \leq N < 4$
22	$N > 1$ and 200 $\leq ASL <$ 400	$4 \leq N < 7$
23	$N > 1$ and 200 $\leq ASL <$ 400	$7 \leq N <9$
24	$N > 1$ and 200 $\leq ASL <$ 400	$9 \leq N$
25	$N > 1$ and 400 $\leq ASL <$ 525	$2 \leq N < 4$
26	$N > 1$ and 400 $\leq ASL <$ 525	$4 \leq N < 7$
27	$N > 1$ and 400 $\leq ASL <$ 525	$7 \leq N <9$
28	$N > 1$ and 400 $\leq ASL <$ 525	$9 \leq N$
29	$N > 1$ and 525 $\leq ASL <$ 625	$2 \leq N < 4$
30	$N > 1$ and 525 $\leq ASL <$ 625	$4 \leq N < 7$
31	$N > 1$ and 525 $\leq ASL <$ 625	$7 \leq N <9$
32	$N > 1$ and 525 $\leq ASL <$ 625	$9 \leq N$
33	$N > 1$ and 625 $\leq ASL <$ 750	$2 \leq N < 4$
34	$N > 1$ and 625 $\leq ASL <$ 750	$4 \leq N < 7$
35	$N > 1$ and 625 $\leq ASL <$ 750	$7 \leq N <9$
36	$N > 1$ and 625 $\leq ASL <$ 750	$9 \leq N$
37	$N > 1$ and 750 $\leq ASL$	$2 \leq N < 4$
38	$N > 1$ and 750 $\leq ASL$	$4 \leq N < 7$
39	$N > 1$ and 750 $\leq ASL$	$7 \leq N <9$
40	$N > 1$ and 750 $\leq ASL$	$9 \leq N$

Table H.10 State table for 95% utilization level and k=5

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and ASL <-700	$2 \leq N < 4$
2	$N > 1$ and ASL <-700	$4 \leq N < 7$
3	$N > 1$ and ASL <-700	$7 \leq N <9$
4	$N > 1$ and ASL <-700	$9 \leq N$
5	$N > 1$ and -700 $\leq ASL <$ -100	$2 \leq N < 4$
6	$N > 1$ and -700 $\leq ASL <$ -100	$4 \leq N < 7$
7	$N > 1$ and -700 $\leq ASL <$ -100	$7 \leq N <9$
8	$N > 1$ and -700 $\leq ASL <$ -100	$9 \leq N$
9	$N > 1$ and -100 $\leq ASL <$100	$2 \leq N < 4$
10	$N > 1$ and -100 $\leq ASL <$100	$4 \leq N < 7$
11	$N > 1$ and -100 $\leq ASL <$100	$7 \leq N <9$
12	$N > 1$ and -100 $\leq ASL <$100	$9 \leq N$
13	$N > 1$ and 100 $\leq ASL <$ 400	$2 \leq N < 4$
14	$N > 1$ and 100 $\leq ASL <$ 400	$4 \leq N < 7$
15	$N > 1$ and 100 $\leq ASL <$ 400	$7 \leq N <9$
16	$N > 1$ and 100 $\leq ASL <$ 400	$9 \leq N$
17	$N > 1$ and 400 $\leq ASL <$ 575	$2 \leq N < 4$
18	$N > 1$ and 400 $\leq ASL <$ 575	$4 \leq N < 7$
19	$N > 1$ and 400 $\leq ASL <$ 575	$7 \leq N <9$
20	$N > 1$ and 400 $\leq ASL <$ 575	$9 \leq N$
21	$N > 1$ and 575 $\leq ASL <$ 700	$2 \leq N < 4$
22	$N > 1$ and 575 $\leq ASL <$ 700	$4 \leq N < 7$
23	$N > 1$ and 575 $\leq ASL <$ 700	$7 \leq N <9$
24	$N > 1$ and 575 $\leq ASL <$ 700	$9 \leq N$
25	$N > 1$ and 700 $\leq ASL <$ 810	$2 \leq N < 4$
26	$N > 1$ and 700 $\leq ASL <$ 810	$4 \leq N < 7$
27	$N > 1$ and 700 $\leq ASL <$ 810	$7 \leq N <9$
28	$N > 1$ and 700 $\leq ASL <$ 810	$9 \leq N$
29	$N > 1$ and 810 $\leq ASL <$ 920	$2 \leq N < 4$
30	$N > 1$ and 810 $\leq ASL <$ 920	$4 \leq N < 7$
31	$N > 1$ and 810 $\leq ASL <$ 920	$7 \leq N <9$
32	$N > 1$ and 810 $\leq ASL <$ 920	$9 \leq N$
33	$N > 1$ and 920 $\leq ASL <$ 1050	$2 \leq N < 4$
34	$N > 1$ and 920 $\leq ASL <$ 1050	$4 \leq N < 7$
35	$N > 1$ and 920 $\leq ASL <$ 1050	$7 \leq N <9$
36	$N > 1$ and 920 $\leq ASL <$ 1050	$9 \leq N$
37	$N > 1$ and 1050 $\leq ASL$	$2 \leq N < 4$
38	$N > 1$ and 1050 $\leq ASL$	$4 \leq N < 7$
39	$N > 1$ and 1050 $\leq ASL$	$7 \leq N <9$
40	$N > 1$ and 1050 $\leq ASL$	$9 \leq N$

Table H.11 State table for 95% utilization level and k=6

State	Determination Criteria 1	Determination Criteria 2
Dummy State	$N < 2$	$N < 2$
1	$N > 1$ and ASL <-100	$2 \leq N < 4$
2	$N > 1$ and ASL <-100	$4 \leq N < 7$
3	$N > 1$ and ASL <-100	$7 \leq N <9$
4	$N > 1$ and ASL <-100	$9 \leq N$
5	$N > 1$ and -100 $\leq ASL <$ 300	$2 \leq N < 4$
6	$N > 1$ and -100 $\leq ASL <$ 300	$4 \leq N < 7$
7	$N > 1$ and -100 $\leq ASL <$ 300	$7 \leq N <9$
8	$N > 1$ and -100 $\leq ASL <$ 300	$9 \leq N$
9	$N > 1$ and 300 $\leq ASL <$580	$2 \leq N < 4$
10	$N > 1$ and 300 $\leq ASL <$580	$4 \leq N < 7$
11	$N > 1$ and 300 $\leq ASL <$580	$7 \leq N <9$
12	$N > 1$ and 300 $\leq ASL <$580	$9 \leq N$
13	$N > 1$ and 580 $\leq ASL <$ 750	$2 \leq N < 4$
14	$N > 1$ and 580 $\leq ASL <$ 750	$4 \leq N < 7$
15	$N > 1$ and 580 $\leq ASL <$ 750	$7 \leq N <9$
16	$N > 1$ and 580 $\leq ASL <$ 750	$9 \leq N$
17	$N > 1$ and 750 $\leq ASL <$ 875	$2 \leq N < 4$
18	$N > 1$ and 750 $\leq ASL <$ 875	$4 \leq N < 7$
19	$N > 1$ and 750 $\leq ASL <$ 875	$7 \leq N <9$
20	$N > 1$ and 750 $\leq ASL <$ 875	$9 \leq N$
21	$N > 1$ and 875 $\leq ASL <$ 975	$2 \leq N < 4$
22	$N > 1$ and 875 $\leq ASL <$ 975	$4 \leq N < 7$
23	$N > 1$ and 875 $\leq ASL <$ 975	$7 \leq N <9$
24	$N > 1$ and 875 $\leq ASL <$ 975	$9 \leq N$
25	$N > 1$ and 975 $\leq ASL <$ 1100	$2 \leq N < 4$
26	$N > 1$ and 975 $\leq ASL <$ 1100	$4 \leq N < 7$
27	$N > 1$ and 975 $\leq ASL <$ 1100	$7 \leq N <9$
28	$N > 1$ and 975 $\leq ASL <$ 1100	$9 \leq N$
29	$N > 1$ and 1100 $\leq ASL <$ 1250	$2 \leq N < 4$
30	$N > 1$ and 1100 $\leq ASL <$ 1250	$4 \leq N < 7$
31	$N > 1$ and 1100 $\leq ASL <$ 1250	$7 \leq N <9$
32	$N > 1$ and 1100 $\leq ASL <$ 1250	$9 \leq N$
33	$N > 1$ and 1250 $\leq ASL <$ 1400	$2 \leq N < 4$
34	$N > 1$ and 1250 $\leq ASL <$ 1400	$4 \leq N < 7$
35	$N > 1$ and 1250 $\leq ASL <$ 1400	$7 \leq N <9$
36	$N > 1$ and 1250 $\leq ASL <$ 1400	$9 \leq N$
37	$N > 1$ and 1400 $\leq ASL$	$2 \leq N < 4$
38	$N > 1$ and 1400 $\leq ASL$	$4 \leq N < 7$
39	$N > 1$ and 1400 $\leq ASL$	$7 \leq N <9$
40	$N > 1$ and 1400 $\leq ASL$	$9 \leq N$

APPENDIX I: Selecting Percentages for Multi State Flow Shop Applications

Table I.1 Selection percentages of dispatching rules in each state for k=4

State	85% U		90% U		95% U	
	COVERT	***SPT***	***COVERT***	***SPT***	***COVERT***	***SPT***
1	0.25	0.75	0.25	0.75	0.20	0.80
2	0.23	0.77	0.23	0.77	0.19	0.81
3	0.20	0.80	0.17	0.83	0.16	0.84
4	0.19	0.81	0.15	0.85	0.16	0.84
5	0.27	0.73	0.26	0.74	0.22	0.78
6	0.25	0.75	0.23	0.77	0.20	0.80
7	0.25	0.75	0.21	0.79	0.18	0.82
8	0.22	0.78	0.21	0.79	0.16	0.84
9	0.32	0.68	0.30	0.70	0.28	0.72
10	0.29	0.71	0.28	0.72	0.25	0.75
11	0.28	0.72	0.26	0.74	0.25	0.75
12	0.25	0.75	0.25	0.75	0.23	0.77
13	0.35	0.65	0.33	0.67	0.33	0.67
14	0.30	0.70	0.31	0.69	0.30	0.70
15	0.28	0.72	0.27	0.73	0.26	0.74
16	0.27	0.73	0.25	0.75	0.26	0.74
17	0.57	0.43	0.56	0.44	0.40	0.60
18	0.56	0.44	0.55	0.45	0.36	0.64
19	0.43	0.57	0.42	0.58	0.34	0.66
20	0.42	0.58	0.40	0.60	0.34	0.66
21	0.68	0.32	0.68	0.32	0.63	0.37
22	0.63	0.37	0.66	0.34	0.60	0.40
23	0.63	0.37	0.63	0.37	0.60	0.40
24	0.58	0.42	0.60	0.40	0.58	0.42
25	0.73	0.27	0.76	0.24	0.74	0.26
26	0.71	0.29	0.73	0.27	0.71	0.29
27	0.69	0.31	0.70	0.30	0.68	0.32
28	0.67	0.33	0.68	0.32	0.65	0.35
29	0.80	0.20	0.80	0.20	0.78	0.22
30	0.78	0.22	0.76	0.24	0.76	0.24
31	0.73	0.27	0.73	0.27	0.75	0.25
32	0.71	0.29	0.70	0.30	0.71	0.29
33	0.86	0.14	0.83	0.17	0.81	0.19
34	0.83	0.17	0.80	0.20	0.79	0.21
35	0.83	0.17	0.78	0.22	0.77	0.23
36	0.77	0.23	0.75	0.25	0.77	0.23
37	0.89	0.11	0.86	0.14	0.85	0.15
38	0.87	0.13	0.83	0.17	0.81	0.19
39	0.84	0.16	0.80	0.20	0.79	0.21
40	0.84	0.16	0.79	0.21	0.77	0.23

Table I.2 Selection percentages of dispatching rules in each state for k=5

State	85% U		90% U		95% U	
	COVERT	***PT+WINQ+SL***	***COVERT***	***SPT***	***COVERT***	***SPT***
1	0.22	0.78	0.20	0.80	0.19	0.81
2	0.20	0.80	0.18	0.82	0.16	0.84
3	0.20	0.80	0.18	0.82	0.15	0.85
4	0.17	0.83	0.16	0.84	0.15	0.85
5	0.30	0.70	0.28	0.72	0.23	0.77
6	0.27	0.73	0.25	0.75	0.20	0.80
7	0.26	0.74	0.24	0.76	0.18	0.82
8	0.24	0.76	0.22	0.78	0.16	0.84
9	0.38	0.62	0.33	0.67	0.30	0.70
10	0.36	0.64	0.31	0.69	0.28	0.72
11	0.33	0.67	0.28	0.72	0.26	0.74
12	0.32	0.68	0.26	0.74	0.26	0.74
13	0.40	0.60	0.39	0.61	0.43	0.57
14	0.38	0.62	0.37	0.63	0.41	0.59
15	0.38	0.62	0.36	0.64	0.38	0.62
16	0.36	0.64	0.36	0.64	0.36	0.64
17	0.60	0.40	0.58	0.42	0.57	0.43
18	0.57	0.43	0.56	0.44	0.55	0.45
19	0.45	0.55	0.44	0.56	0.43	0.57
20	0.44	0.56	0.43	0.57	0.43	0.57
21	0.73	0.27	0.69	0.31	0.65	0.35
22	0.70	0.30	0.67	0.33	0.61	0.39
23	0.67	0.33	0.65	0.35	0.60	0.40
24	0.65	0.35	0.65	0.35	0.58	0.42
25	0.83	0.17	0.80	0.20	0.73	0.27
26	0.80	0.20	0.78	0.22	0.70	0.30
27	0.78	0.22	0.74	0.26	0.68	0.32
28	0.78	0.22	0.72	0.28	0.67	0.33
29	0.87	0.13	0.85	0.15	0.83	0.17
30	0.85	0.15	0.83	0.17	0.80	0.20
31	0.80	0.20	0.78	0.22	0.78	0.22
32	0.77	0.23	0.75	0.25	0.76	0.24
33	0.92	0.08	0.88	0.12	0.85	0.15
34	0.89	0.11	0.86	0.14	0.83	0.17
35	0.87	0.13	0.86	0.14	0.80	0.20
36	0.85	0.15	0.83	0.17	0.76	0.24
37	0.93	0.07	0.90	0.10	0.87	0.13
38	0.91	0.09	0.88	0.12	0.85	0.15
39	0.88	0.12	0.86	0.14	0.82	0.18
40	0.86	0.14	0.83	0.17	0.80	0.20

Table I.3 Selection percentages of dispatching rules in each state for k=6

State	85% U		90% U		95% U	
	COVERT	***PT+WINQ+SL***	***COVERT***	***SPT***	***COVERT***	***SPT***
1	0.28	0.72	0.23	0.77	0.20	0.80
2	0.26	0.74	0.21	0.79	0.18	0.82
3	0.24	0.76	0.18	0.82	0.16	0.84
4	0.23	0.77	0.17	0.83	0.16	0.84
5	0.35	0.65	0.30	0.70	0.28	0.72
6	0.32	0.68	0.28	0.72	0.26	0.74
7	0.30	0.70	0.28	0.72	0.26	0.74
8	0.27	0.73	0.25	0.75	0.23	0.77
9	0.42	0.58	0.36	0.64	0.35	0.65
10	0.40	0.60	0.34	0.66	0.33	0.67
11	0.38	0.62	0.33	0.67	0.31	0.69
12	0.38	0.62	0.31	0.69	0.30	0.70
13	0.59	0.41	0.57	0.43	0.43	0.57
14	0.57	0.43	0.55	0.45	0.41	0.59
15	0.45	0.55	0.44	0.56	0.38	0.62
16	0.44	0.56	0.43	0.57	0.38	0.62
17	0.74	0.26	0.73	0.27	0.67	0.33
18	0.72	0.28	0.70	0.30	0.65	0.35
19	0.68	0.32	0.70	0.30	0.62	0.38
20	0.67	0.33	0.68	0.32	0.60	0.40
21	0.83	0.17	0.79	0.21	0.76	0.24
22	0.80	0.20	0.76	0.24	0.73	0.27
23	0.76	0.24	0.74	0.26	0.71	0.29
24	0.75	0.25	0.72	0.28	0.69	0.31
25	0.86	0.14	0.83	0.17	0.81	0.19
26	0.84	0.16	0.80	0.20	0.80	0.20
27	0.83	0.17	0.79	0.21	0.80	0.20
28	0.80	0.20	0.77	0.23	0.78	0.22
29	0.88	0.12	0.86	0.14	0.84	0.16
30	0.86	0.14	0.83	0.17	0.81	0.19
31	0.84	0.16	0.82	0.18	0.80	0.20
32	0.82	0.18	0.80	0.20	0.77	0.23
33	0.92	0.08	0.90	0.10	0.87	0.13
34	0.90	0.10	0.87	0.13	0.85	0.15
35	0.88	0.12	0.85	0.15	0.82	0.18
36	0.86	0.14	0.84	0.16	0.81	0.19
37	0.94	0.06	0.92	0.08	0.90	0.10
38	0.91	0.09	0.89	0.11	0.88	0.12
39	0.89	0.11	0.87	0.13	0.88	0.12
40	0.86	0.14	0.87	0.13	0.86	0.14

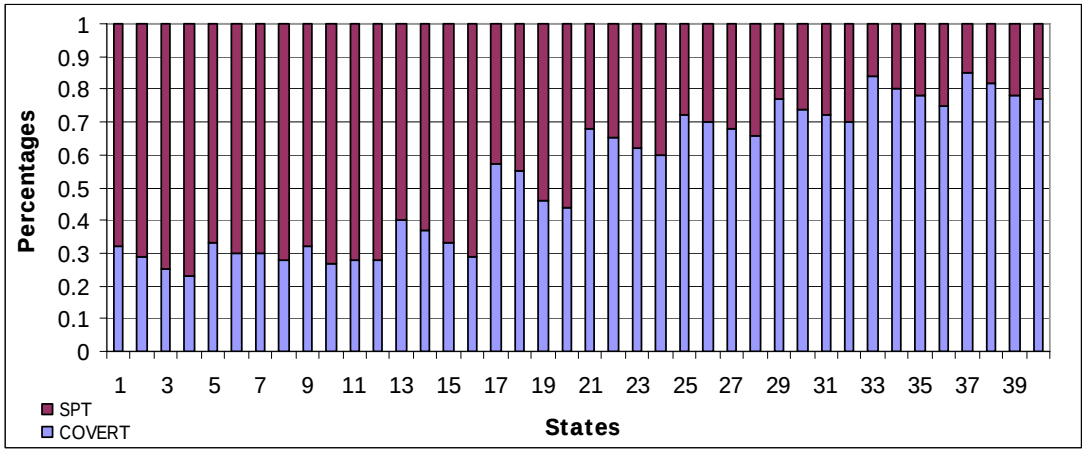

Figure I.1 Selection percentages for 85% and *k*=3

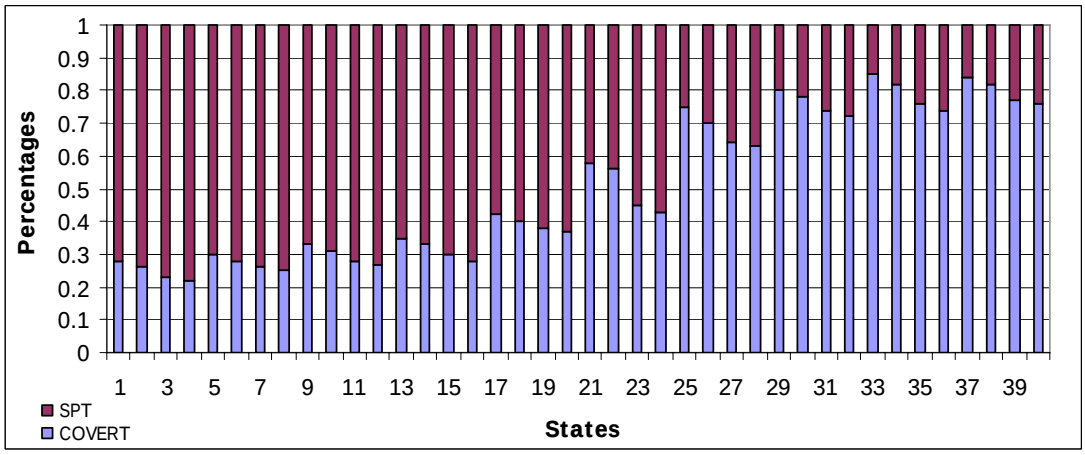

Figure I.2 Selection percentages for 90% and *k*=3

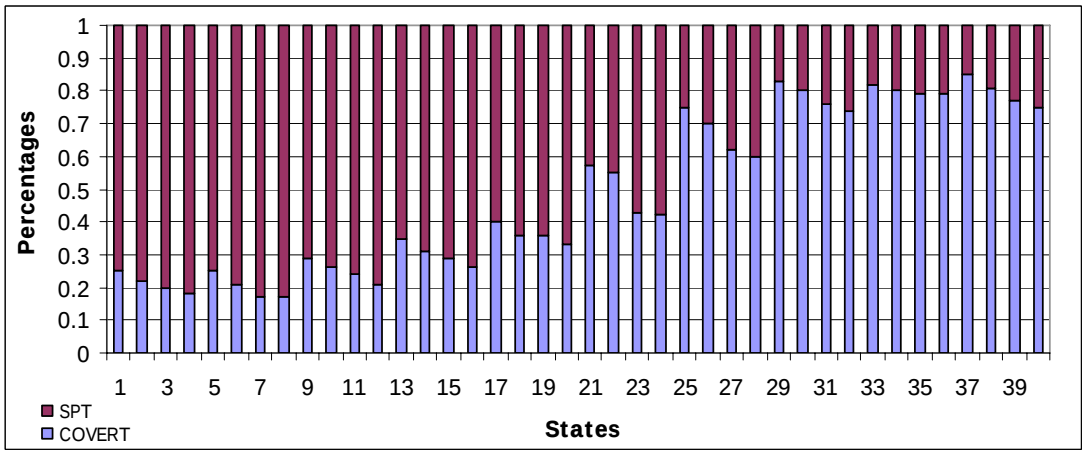

Figure I.3 Selection percentages for 95% and *k*=3

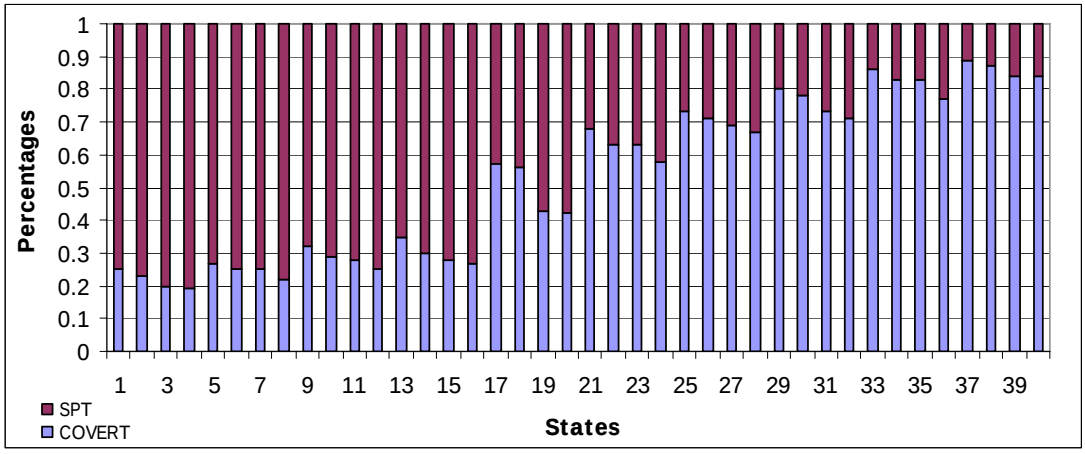

Figure I.4 Selection percentages for 85% and *k*=4

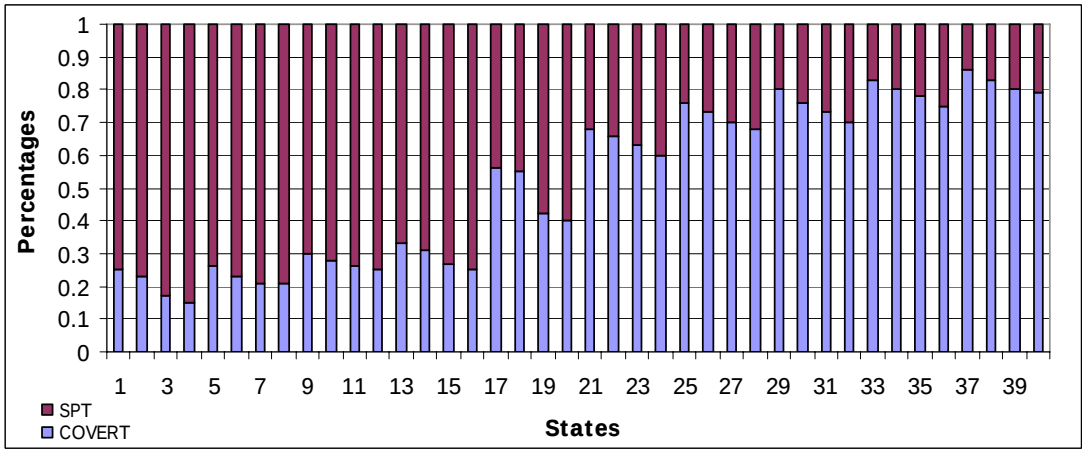

Figure I.5 Selection percentages for 90% and *k*=4

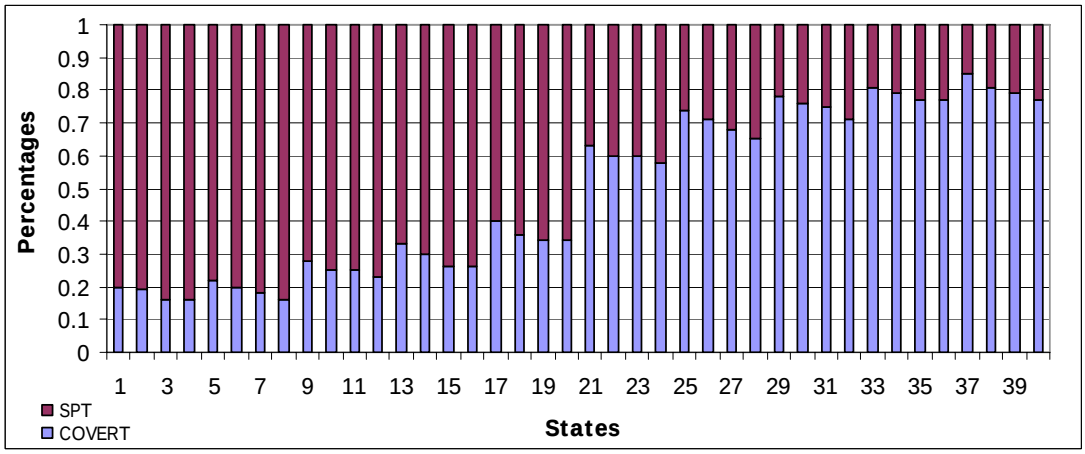

Figure I.6 Selection percentages for 95% and *k*=4

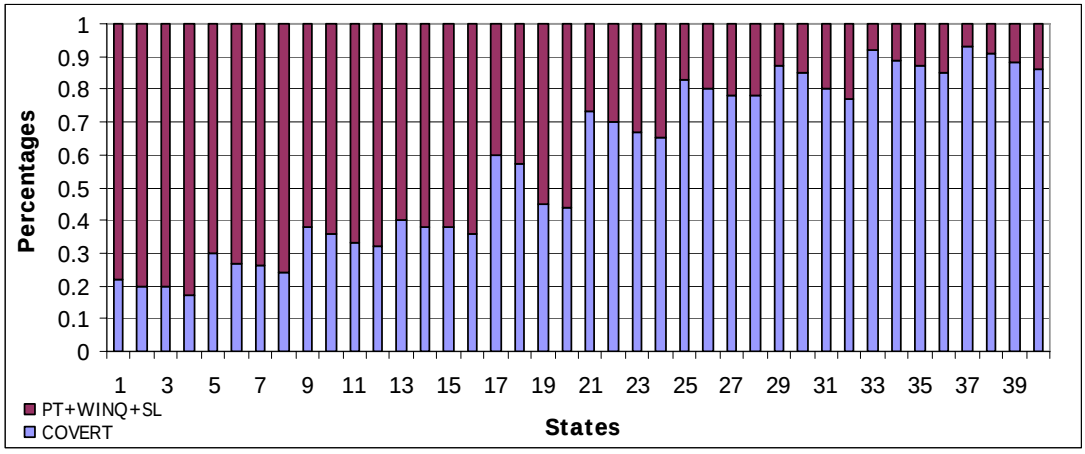

Figure I.7 Selection percentages for 85% and *k*=5

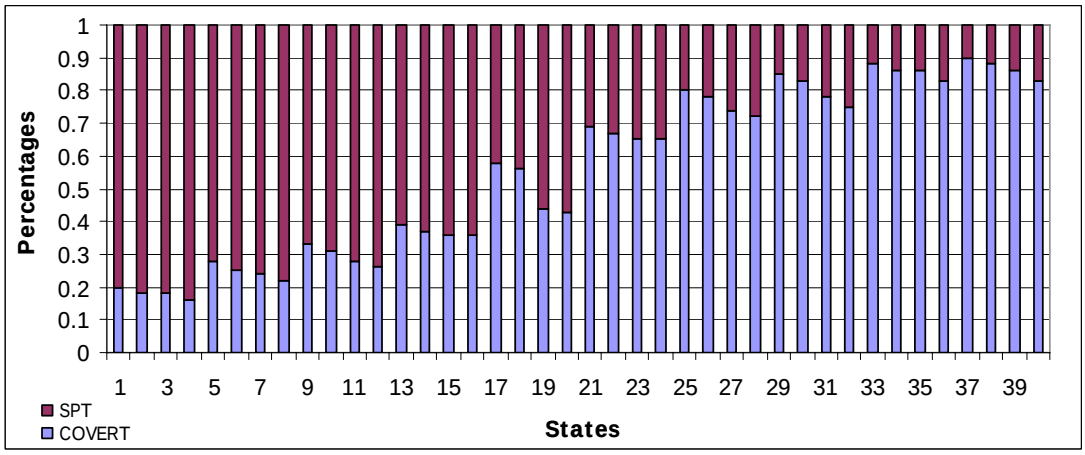

Figure I.8 Selection percentages for 90% and *k*=5

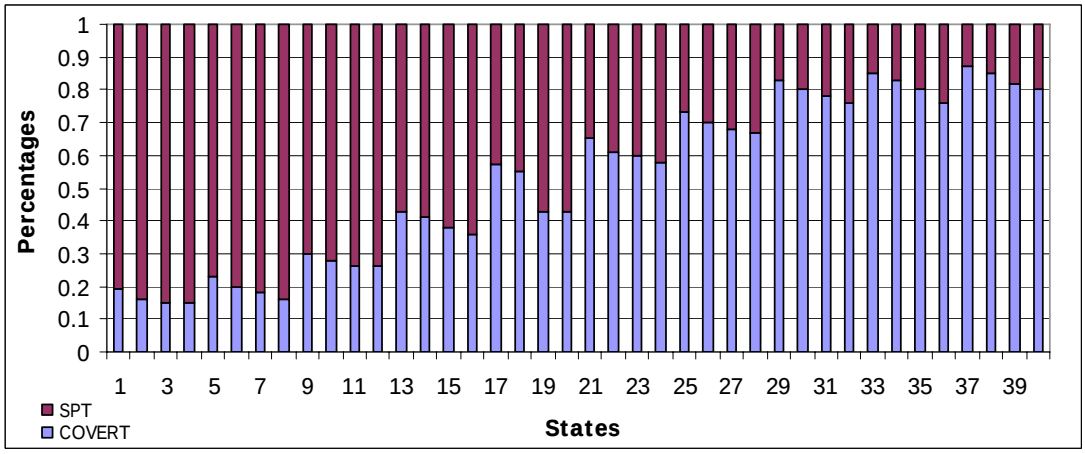

Figure I.9 Selection percentages for 95% and *k*=5

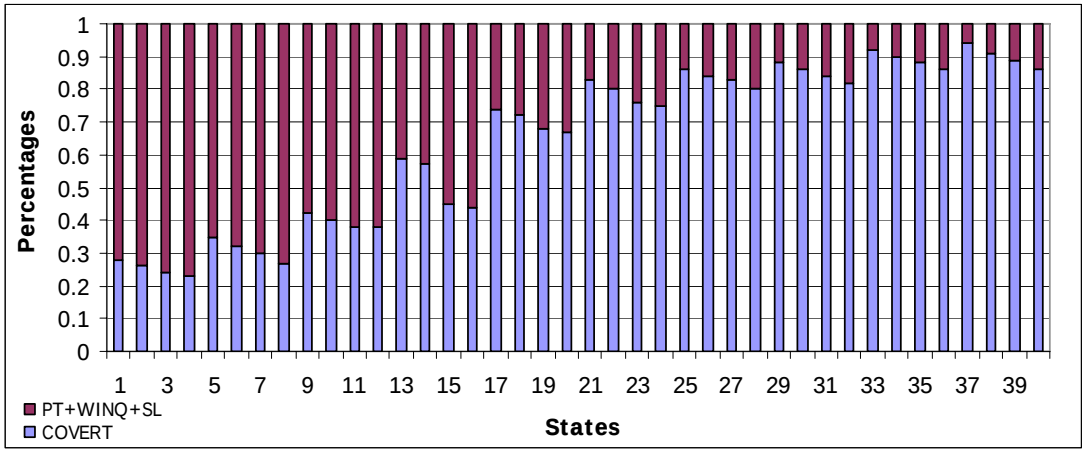

Figure I.10 Selection percentages for 85% and *k*=6

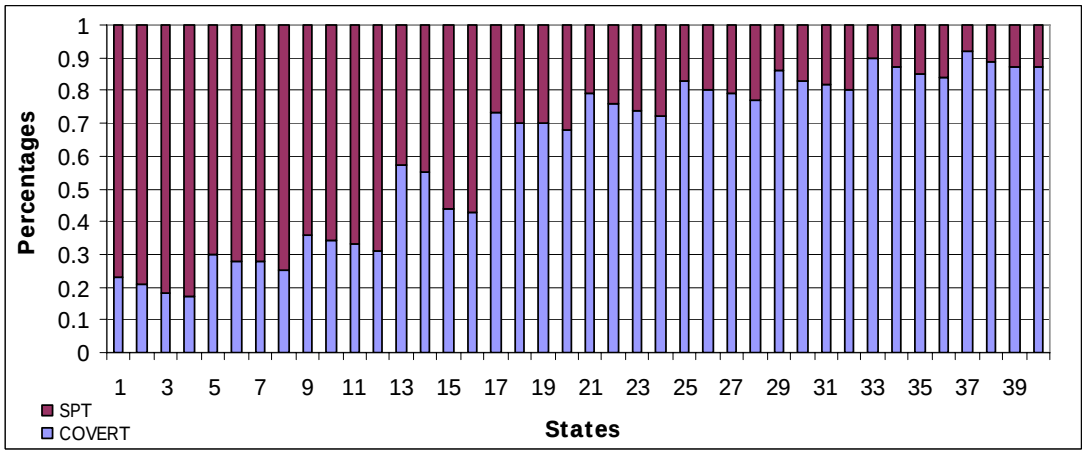

Figure I.11 Selection percentages for 90% and *k*=6

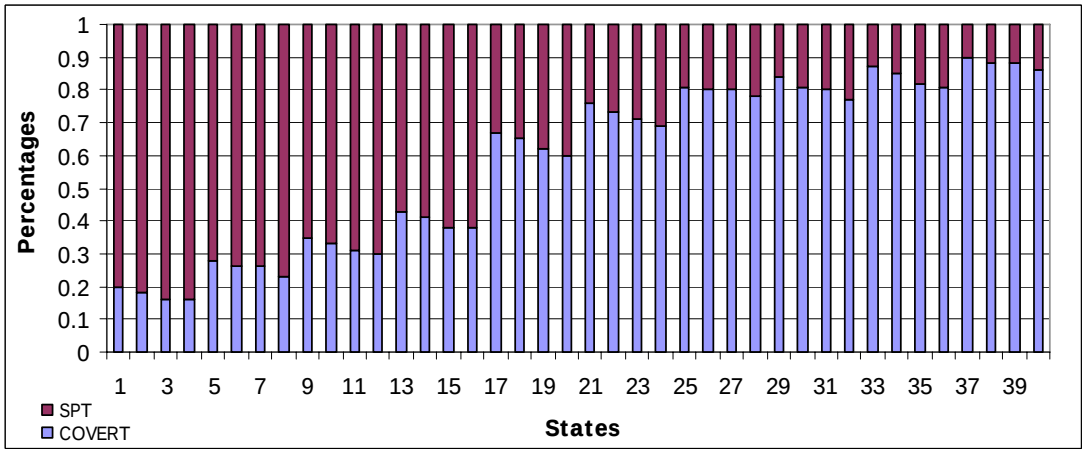

Figure I.12 Selection percentages for 90% and *k*=6

APPENDIX J.1: Ranking Results for Single Machine Applications

Table J.1 Ranking results for 80% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	2.86421	2.86614	2.86808	0.00194
HA	6.12621	6.12839	6.13057	0.00218
FIFO	10.15374	10.15425	10.15476	0.00051
LIFO	7.92885	7.93079	7.93274	0.00195
EDD	9.47597	9.47690	9.47782	0.00092
RR	7.97660	7.97775	7.97890	0.00115
***Q*-Agent**	1.85999	1.86044	1.86088	0.00045
Test 1	5.49898	5.49942	5.49986	0.00044
Test 2	6.66121	6.66182	6.66243	0.00061
Test 3	4.41027	4.41063	4.41099	0.00036
Test 4	3.03310	3.03347	3.03384	0.00037

Table J.2 Ranking results for 85% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	3.23866	3.24098	3.24330	0.00232
HA	6.25227	6.25489	6.25752	0.00262
FIFO	9.55446	9.55527	9.55608	0.00081
LIFO	7.70135	7.70388	7.70641	0.00253
EDD	9.56683	9.56812	9.56942	0.00130
RR	7.95946	7.96055	7.96164	0.00109
***Q*-Agent**	1.85274	1.85348	1.85422	0.00074
Test 1	5.87816	5.87901	5.87987	0.00086
Test 2	6.61314	6.61398	6.61483	0.00084
Test 3	4.35895	4.35940	4.35985	0.00045
Test 4	3.00975	3.01043	3.01110	0.00068

Table J.3 Ranking results for 90% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	3.61626	3.61847	3.62068	0.00221
HA	6.09310	6.09571	6.09831	0.00260
FIFO	10.33200	10.33261	10.33322	0.00061
LIFO	7.56614	7.56852	7.57090	0.00238
EDD	9.88662	9.88749	9.88836	0.00087
RR	8.53993	8.54051	8.54110	0.00059
***Q*-Agent**	1.37326	1.37388	1.37450	0.00062
Test 1	5.11920	5.11964	5.12009	0.00044
Test 2	5.28911	5.28972	5.29034	0.00061
Test 3	5.49345	5.49474	5.49604	0.00129
Test 4	2.67815	2.67870	2.67926	0.00055

Table J.4 Ranking results for 95% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	3.71782	3.72002	3.72223	0.00221
HA	6.41995	6.42197	6.42398	0.00201
FIFO	10.03989	10.04038	10.04086	0.00048
LIFO	7.72542	7.72725	7.72907	0.00182
EDD	10.18353	10.18444	10.18536	0.00091
RR	8.59271	8.59334	8.59396	0.00062
***Q*-Agent**	1.45236	1.45284	1.45332	0.00048
Test 1	4.79419	4.79456	4.79494	0.00037
Test 2	5.90224	5.90245	5.90265	0.00021
Test 3	4.49551	4.49649	4.49747	0.00098
Test 4	2.66582	2.66627	2.66673	0.00046

Table J.5 Ranking results for 80% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	2.82772	2.82942	2.83113	0.00171
HA	6.15736	6.15959	6.16181	0.00223
FIFO	7.96217	7.96424	7.96631	0.00207
LIFO	8.45067	8.45264	8.45462	0.00197
EDD	6.95676	6.95896	6.96116	0.00220
RR	9.02725	9.02901	9.03077	0.00176
***Q*-Agent**	2.35731	2.35820	2.35910	0.00089
Test 1	6.29185	6.29274	6.29363	0.00089
Test 2	7.08098	7.08196	7.08294	0.00098
Test 3	5.28229	5.28345	5.28460	0.00116
Test 4	3.58887	3.58979	3.59071	0.00092

Table J.6 Ranking results for 85% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	3.15948	3.16176	3.16404	0.00228
HA	5.95224	5.95455	5.95685	0.00231
FIFO	7.42851	7.43023	7.43195	0.00172
LIFO	8.27966	8.28221	8.28476	0.00255
EDD	8.79693	8.79835	8.79978	0.00142
RR	10.30582	10.30652	10.30722	0.00070
***Q*-Agent**	1.90270	1.90332	1.90395	0.00062
Test 1	6.70791	6.70862	6.70933	0.00071
Test 2	5.39667	5.39738	5.39808	0.00070
Test 3	4.95061	4.95142	4.95223	0.00081
Test 4	3.10509	3.10564	3.10620	0.00056

Table J.7 Ranking results for 90% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	3.08010	3.08181	3.08352	0.00171
HA	6.68740	6.68963	6.69185	0.00222
FIFO	9.94930	9.95042	9.95153	0.00112
LIFO	8.41472	8.41669	8.41866	0.00197
EDD	9.14991	9.15119	9.15246	0.00127
RR	7.77484	7.77614	7.77744	0.00130
***Q*-Agent**	1.61561	1.61605	1.61650	0.00045
Test 1	6.36536	6.36633	6.36731	0.00097
Test 2	5.32977	5.33075	5.33174	0.00098
Test 3	4.88667	4.88785	4.88903	0.00118
Test 4	2.73276	2.73314	2.73352	0.00038

Table J.8 Ranking results for 95% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	3.14912	3.15148	3.15384	0.00236
HA	6.95042	6.95298	6.95554	0.00256
FIFO	9.97881	9.97967	9.98053	0.00086
LIFO	8.53710	8.53941	8.54171	0.00230
EDD	9.55998	9.56115	9.56231	0.00117
RR	8.25976	8.26070	8.26164	0.00094
***Q*-Agent**	1.61940	1.61996	1.62052	0.00056
Test 1	4.91640	4.91658	4.91677	0.00018
Test 2	6.00173	6.00199	6.00225	0.00026
Test 3	4.26135	4.26211	4.26286	0.00075
Test 4	2.75351	2.75397	2.75443	0.00046

APPENDIX J.2: Ranking Results for Single Machine Applications When $\overline{w}_1 > \overline{w}_2$

Table J.9 Ranking results for 80% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	1.00000	1.00000	1.00000	0.00000
HA	8.82768	8.82864	8.82961	0.00097
FIFO	9.76602	9.76666	9.76731	0.00064
LIFO	10.15194	10.15289	10.15384	0.00095
EDD	8.49460	8.49536	8.49612	0.00076
RR	6.95465	6.95566	6.95666	0.00101
***Q*-Agent**	2.00000	2.00000	2.00000	0.00000
Test 1	5.33116	5.33157	5.33198	0.00041
Test 2	6.34514	6.34556	6.34598	0.00042
Test 3	4.12343	4.12366	4.12389	0.00023
Test 4	3.00000	3.00000	3.00000	0.00000

Table J.10 Ranking results for 85% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	1.04007	1.04025	1.04042	0.00018
HA	9.09973	9.10083	9.10193	0.00110
FIFO	9.10633	9.10723	9.10813	0.00090
LIFO	10.40438	10.40543	10.40648	0.00105
EDD	8.12972	8.13084	8.13195	0.00112
RR	6.91509	6.91630	6.91751	0.00121
***Q*-Agent**	1.95958	1.95975	1.95993	0.00018
Test 1	5.56239	5.56331	5.56423	0.00092
Test 2	6.67961	6.68063	6.68164	0.00102
Test 3	4.09511	4.09544	4.09576	0.00033
Test 4	3.00000	3.00000	3.00000	0.00000

Table J.11 Ranking results for 90% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	1.91352	1.91406	1.91460	0.00054
HA	8.09489	8.09588	8.09687	0.00099
FIFO	10.23836	10.23903	10.23971	0.00067
LIFO	9.38136	9.38261	9.38386	0.00125
EDD	9.20068	9.20140	9.20212	0.00072
RR	8.08026	8.08107	8.08188	0.00081
***Q*-Agent**	1.29503	1.29535	1.29567	0.00032
Test 1	4.92513	4.92530	4.92548	0.00018
Test 2	5.94217	5.94234	5.94251	0.00017
Test 3	4.13202	4.13235	4.13269	0.00034
Test 4	2.79027	2.79059	2.79091	0.00032

Table J.12 Ranking results for 95% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	1.65259	1.65313	1.65367	0.00054
HA	8.11292	8.11411	8.11530	0.00119
FIFO	10.07973	10.08052	10.08131	0.00079
LIFO	9.25069	9.25191	9.25313	0.00122
EDD	9.36667	9.36752	9.36838	0.00085
RR	8.18513	8.18593	8.18673	0.00080
***Q*-Agent**	1.51365	1.51410	1.51455	0.00045
Test 1	5.00000	5.00000	5.00000	0.00000
Test 2	6.00000	6.00000	6.00000	0.00000
Test 3	4.00000	4.00000	4.00000	0.00000
Test 4	2.83252	2.83277	2.83302	0.00025

Table J.13 Ranking results for 80% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	1.19046	1.19085	1.19125	0.00040
HA	9.25943	9.26040	9.26137	0.00097
FIFO	5.76770	5.76963	5.77155	0.00192
LIFO	10.94941	10.94957	10.94973	0.00016
EDD	4.37290	4.37492	4.37694	0.00202
RR	7.77699	7.77862	7.78025	0.00163
***Q*-Agent**	2.80396	2.80476	2.80557	0.00080
Test 1	6.76130	6.76243	6.76356	0.00113
Test 2	7.78204	7.78315	7.78426	0.00111
Test 3	5.32361	5.32460	5.32559	0.00099
Test 4	4.00014	4.00107	4.00200	0.00093

Table J.14 Ranking results for 85% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	1.05517	1.05533	1.05549	0.00016
HA	8.78102	8.78203	8.78305	0.00102
FIFO	5.84513	5.84700	5.84886	0.00187
LIFO	10.75045	10.75095	10.75146	0.00051
EDD	7.51582	7.51720	7.51858	0.00138
RR	9.70229	9.70291	9.70353	0.00062
***Q*-Agent**	2.03440	2.03469	2.03499	0.00030
Test 1	6.71689	6.71767	6.71846	0.00078
Test 2	5.71322	5.71401	5.71479	0.00079
Test 3	4.67246	4.67319	4.67391	0.00072
Test 4	3.20460	3.20501	3.20543	0.00042

Table J.15 Ranking results for 90% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	1.20581	1.20622	1.20664	0.00042
HA	8.97319	8.97424	8.97529	0.00105
FIFO	9.15477	9.15584	9.15692	0.00108
LIFO	10.58000	10.58085	10.58170	0.00085
EDD	8.04795	8.04903	8.05012	0.00108
RR	6.55192	6.55299	6.55407	0.00108
***Q*-Agent**	1.81388	1.81426	1.81464	0.00038
Test 1	6.85115	6.85214	6.85312	0.00098
Test 2	5.83394	5.83490	5.83587	0.00096
Test 3	4.00000	4.00000	4.00000	0.00000
Test 4	2.97941	2.97952	2.97963	0.00011

Table J.16 Ranking results for 95% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	1.12226	1.12257	1.12288	0.00031
HA	8.95551	8.95671	8.95791	0.00120
FIFO	9.54245	9.54316	9.54387	0.00071
LIFO	10.44047	10.44151	10.44255	0.00104
EDD	8.53700	8.53769	8.53838	0.00069
RR	7.52023	7.52094	7.52165	0.00071
***Q*-Agent**	1.87712	1.87743	1.87774	0.00031
Test 1	5.00000	5.00000	5.00000	0.00000
Test 2	6.00000	6.00000	6.00000	0.00000
Test 3	4.00000	4.00000	4.00000	0.00000
Test 4	3.00000	3.00000	3.00000	0.00000

APPENDIX J.3: Ranking Results for Single Machine Applications When $\overline{w}_1 < \overline{w}_2$

Table J.17 Ranking results for 80% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	4.73233	4.73457	4.73681	0.00224
HA	3.42474	3.42659	3.42843	0.00185
FIFO	10.54066	10.54110	10.54154	0.00044
LIFO	5.70477	5.70677	5.70877	0.00200
EDD	10.45846	10.45890	10.45934	0.00044
RR	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.72086	1.72146	1.72206	0.00060
Test 1	5.66695	5.66732	5.66770	0.00038
Test 2	6.97756	6.97798	6.97840	0.00042
Test 3	4.69779	4.69812	4.69845	0.00033
Test 4	3.06661	3.06718	3.06776	0.00057

Table J.18 Ranking results for 85% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	5.43741	5.43976	5.44211	0.00235
HA	3.41233	3.41461	3.41688	0.00228
FIFO	10.00000	10.00000	10.00000	0.00000
LIFO	5.00523	5.00777	5.01031	0.00254
EDD	11.00000	11.00000	11.00000	0.00000
RR	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.74776	1.74853	1.74931	0.00078
Test 1	6.19460	6.19539	6.19619	0.00079
Test 2	6.54722	6.54786	6.54850	0.00064
Test 3	4.62337	4.62380	4.62423	0.00043
Test 4	3.02148	3.02228	3.02308	0.00080

Table J.19 Ranking results for 90% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	5.32206	5.32344	5.32482	0.00138
HA	4.09335	4.09495	4.09655	0.00160
FIFO	10.42577	10.42608	10.42640	0.00031
LIFO	5.75353	5.75501	5.75650	0.00149
EDD	10.57360	10.57392	10.57423	0.00031
RR	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.45111	1.45165	1.45219	0.00054
Test 1	5.31338	5.31389	5.31441	0.00052
Test 2	4.63574	4.63618	4.63662	0.00044
Test 3	6.85822	6.85874	6.85926	0.00052
Test 4	2.56549	2.56614	2.56678	0.00065

Table J.20 Ranking results for 95% utilization level and tight due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	5.78206	5.78437	5.78668	0.00231
HA	4.73011	4.73282	4.73553	0.00271
FIFO	10.00000	10.00000	10.00000	0.00000
LIFO	6.20298	6.20538	6.20779	0.00240
EDD	11.00000	11.00000	11.00000	0.00000
RR	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.39077	1.39138	1.39200	0.00062
Test 1	4.58866	4.58908	4.58950	0.00042
Test 2	5.80444	5.80471	5.80499	0.00027
Test 3	4.99107	4.99267	4.99427	0.00160
Test 4	2.49882	2.49958	2.50034	0.00076

Table J.21 Ranking results for 80% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	4.46663	4.46890	4.47116	0.00227
HA	3.05658	3.05841	3.06024	0.00183
FIFO	10.15921	10.15948	10.15976	0.00028
LIFO	5.95255	5.95484	5.95713	0.00229
EDD	9.54287	9.54366	9.54444	0.00078
RR	10.27872	10.27954	10.28036	0.00082
***Q*-Agent**	1.91032	1.91100	1.91168	0.00068
Test 1	5.82298	5.82343	5.82388	0.00045
Test 2	6.38008	6.38088	6.38167	0.00080
Test 3	5.24122	5.24215	5.24308	0.00093
Test 4	3.17720	3.17772	3.17824	0.00052

Table J.22 Ranking results for 85% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	5.26736	5.26885	5.27033	0.00148
HA	3.12547	3.12702	3.12858	0.00155
FIFO	9.01170	9.01179	9.01188	0.00009
LIFO	5.81048	5.81231	5.81414	0.00183
EDD	10.07793	10.07813	10.07832	0.00020
RR	10.90984	10.91009	10.91034	0.00025
***Q*-Agent**	1.77152	1.77214	1.77276	0.00062
Test 1	6.70027	6.70055	6.70082	0.00028
Test 2	5.08126	5.08154	5.08181	0.00028
Test 3	5.23018	5.23066	5.23114	0.00048
Test 4	3.00642	3.00694	3.00746	0.00052

Table J.23 Ranking results for 90% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	4.95432	4.95637	4.95841	0.00204
HA	4.40284	4.40493	4.40702	0.00209
FIFO	10.74552	10.74588	10.74625	0.00037
LIFO	6.25004	6.25227	6.25450	0.00223
EDD	10.25375	10.25412	10.25448	0.00037
RR	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.41717	1.41781	1.41844	0.00063
Test 1	5.87982	5.88014	5.88045	0.00032
Test 2	4.82554	4.82587	4.82620	0.00033
Test 3	5.77434	5.77550	5.77667	0.00117
Test 4	2.48646	2.48712	2.48779	0.00067

Table J.24 Ranking results for 95% utilization level and loose due dates

Methods	Maximum	Average	Minimum	Half Width
SPT	5.17601	5.17795	5.17989	0.00194
HA	4.94981	4.95171	4.95361	0.00190
FIFO	10.41582	10.41623	10.41664	0.00041
LIFO	6.63738	6.63896	6.64054	0.00158
EDD	10.58336	10.58377	10.58418	0.00041
RR	9.00000	9.00000	9.00000	0.00000
***Q*-Agent**	1.36173	1.36221	1.36269	0.00048
Test 1	4.83308	4.83333	4.83357	0.00025
Test 2	6.00382	6.00409	6.00435	0.00027
Test 3	4.52285	4.52369	4.52453	0.00084
Test 4	2.50752	2.50807	2.50862	0.00055

APPENDIX K.1: Ranking Results for Flow Shop Applications

Table K.1 Ranking results for 85% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.80173	10.80201	10.80230	0.00028
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.64094	4.64306	4.64519	0.00212
SPT	5.27270	5.27532	5.27795	0.00262
PT/TIS	2.74177	2.74234	2.74291	0.00057
PT+WINQ+AT	10.19770	10.19799	10.19827	0.00028
PT+WINQ+SL	8.71601	8.71632	8.71664	0.00031
Q-Agent	1.74933	1.74987	1.75040	0.00054
Test 1	6.06359	6.06384	6.06409	0.00025
Test 2	7.11647	7.11677	7.11707	0.00030
Test 3	4.95688	4.95706	4.95724	0.00018
Test 4	3.73506	3.73541	3.73576	0.00035

Table K.2 Ranking results for 90% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.37443	4.37652	4.37860	0.00209
SPT	4.92747	4.92963	4.93179	0.00216
PT/TIS	5.82895	5.83095	5.83294	0.00200
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.54015	1.54075	1.54135	0.00060
Test 1	5.39698	5.39742	5.39786	0.00044
Test 2	6.50734	6.50774	6.50814	0.00040
Test 3	4.32393	4.32444	4.32495	0.00051
Test 4	3.09183	3.09255	3.09327	0.00072

Table K.3 Ranking results for 95% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.62109	4.62302	4.62495	0.00193
SPT	4.50814	4.51015	4.51215	0.00201
PT/TIS	7.80944	7.80973	7.81001	0.00028
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.42359	1.42392	1.42425	0.00033
Test 1	4.96187	4.96202	4.96217	0.00015
Test 2	6.00297	6.00300	6.00303	0.00003
Test 3	3.90741	3.90764	3.90787	0.00023
Test 4	2.76010	2.76052	2.76095	0.00042

Table K.4 Ranking results for 85% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.22166	10.22197	10.22228	0.00031
S/OPN	11.34126	11.34165	11.34205	0.00040
COVERT	3.80618	3.80852	3.81086	0.00234
SPT	4.76676	4.76899	4.77122	0.00223
PT/TIS	4.09509	4.09530	4.09550	0.00021
PT+WINQ+AT	11.43570	11.43637	11.43705	0.00068
PT+WINQ+SL	8.84235	8.84265	8.84294	0.00029
Q-Agent	1.81683	1.81720	1.81757	0.00037
Test 1	6.20277	6.20311	6.20346	0.00035
Test 2	7.28466	7.28505	7.28545	0.00039
Test 3	5.12870	5.12894	5.12919	0.00025
Test 4	3.05009	3.05023	3.05038	0.00014

Table K.5 Ranking results for 90% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.83968	10.84001	10.84034	0.00033
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.20010	4.20240	4.20470	0.00230
SPT	4.68157	4.68422	4.68687	0.00265
PT/TIS	5.80270	5.80482	5.80695	0.00213
PT+WINQ+AT	10.15966	10.15999	10.16032	0.00033
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.64785	1.64856	1.64927	0.00071
Test 1	5.53671	5.53720	5.53770	0.00049
Test 2	6.66051	6.66112	6.66172	0.00060
Test 3	4.37753	4.37794	4.37836	0.00041
Test 4	3.08320	3.08373	3.08426	0.00053

Table K.6 Ranking results for 95% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.78654	4.78910	4.79166	0.00256
SPT	4.28743	4.29045	4.29347	0.00302
PT/TIS	6.60647	6.60842	6.61036	0.00194
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.52728	1.52769	1.52810	0.00041
Test 1	5.31496	5.31540	5.31584	0.00044
Test 2	6.43041	6.43089	6.43137	0.00048
Test 3	4.14346	4.14388	4.14429	0.00042
Test 4	2.89377	2.89418	2.89459	0.00041

Table K.7 Ranking results for 85% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	12.64153	12.64194	12.64234	0.00040
AT-RPT	13.67085	13.67129	13.67173	0.00044
EDD	9.62871	9.62931	9.62992	0.00060
S/OPN	10.74276	10.74337	10.74398	0.00061
COVERT	3.80310	3.80550	3.80790	0.00240
SPT	10.68371	10.68648	10.68924	0.00277
PT/TIS	6.44929	6.45158	6.45387	0.00229
PT+WINQ+AT	11.13342	11.13402	11.13461	0.00060
PT+WINQ+SL	4.69558	4.69822	4.70086	0.00264
Q-Agent	2.08061	2.08114	2.08167	0.00053
Test 1	5.41333	5.41375	5.41416	0.00042
Test 2	6.47916	6.47961	6.48007	0.00045
Test 3	4.32762	4.32802	4.32842	0.00040
Test 4	3.23531	3.23578	3.23624	0.00047

Table K.8 Ranking results for 90% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.28314	10.28357	10.28399	0.00043
S/OPN	11.38393	11.38447	11.38501	0.00054
COVERT	4.08613	4.08861	4.09109	0.00248
SPT	4.96457	4.96771	4.97086	0.00315
PT/TIS	7.88967	7.89112	7.89256	0.00145
PT+WINQ+AT	11.33102	11.33196	11.33290	0.00094
PT+WINQ+SL	8.07101	8.07122	8.07144	0.00021
Q-Agent	1.58547	1.58597	1.58647	0.00050
Test 1	5.20633	5.20667	5.20702	0.00034
Test 2	6.31121	6.31158	6.31196	0.00038
Test 3	4.04595	4.04638	4.04681	0.00043
Test 4	2.83027	2.83073	2.83119	0.00046

Table K.9 Ranking results for 95% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.62283	4.62553	4.62823	0.00270
SPT	4.96156	4.96426	4.96696	0.00270
PT/TIS	4.62734	4.62925	4.63115	0.00190
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.81557	1.81622	1.81687	0.00065
Test 1	5.59458	5.59510	5.59562	0.00052
Test 2	6.70399	6.70459	6.70519	0.00060
Test 3	4.43471	4.43513	4.43556	0.00043
Test 4	3.22935	3.22992	3.23048	0.00057

Table K.10 Ranking results for 85% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	12.54740	12.54789	12.54838	0.00049
AT-RPT	13.56482	13.56529	13.56576	0.00047
EDD	9.19173	9.19277	9.19381	0.00104
S/OPN	10.41792	10.41903	10.42014	0.00111
COVERT	3.44124	3.44360	3.44595	0.00236
SPT	11.15745	11.16036	11.16328	0.00291
PT/TIS	6.13493	6.13763	6.14034	0.00270
PT+WINQ+AT	11.24670	11.24714	11.24757	0.00043
PT+WINQ+SL	4.70133	4.70446	4.70758	0.00312
Q-Agent	2.18442	2.18495	2.18549	0.00053
Test 1	5.64998	5.65058	5.65118	0.00060
Test 2	6.72076	6.72133	6.72190	0.00057
Test 3	4.56422	4.56480	4.56539	0.00059
Test 4	3.45955	3.46016	3.46077	0.00061

Table K.11 Ranking results for 90% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.11675	10.11703	10.11732	0.00028
S/OPN	11.17960	11.17990	11.18020	0.00030
COVERT	3.84769	3.84950	3.85131	0.00181
SPT	4.96520	4.96730	4.96940	0.00210
PT/TIS	8.51345	8.51405	8.51466	0.00060
PT+WINQ+AT	11.70251	11.70307	11.70362	0.00055
PT+WINQ+SL	7.99574	7.99579	7.99583	0.00004
Q-Agent	1.72119	1.72151	1.72183	0.00032
Test 1	5.03000	5.03014	5.03028	0.00014
Test 2	6.11889	6.11918	6.11947	0.00029
Test 3	3.96395	3.96409	3.96423	0.00014
Test 4	2.83813	2.83844	2.83874	0.00031

Table K.12 Ranking results for 95% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.72654	10.72694	10.72734	0.00040
S/OPN	11.92526	11.92550	11.92574	0.00024
COVERT	4.15649	4.15901	4.16152	0.00252
SPT	5.42964	5.43251	5.43538	0.00287
PT/TIS	4.84869	4.85059	4.85248	0.00189
PT+WINQ+AT	10.34702	10.34756	10.34810	0.00054
PT+WINQ+SL	8.85447	8.85478	8.85508	0.00030
Q-Agent	1.70326	1.70389	1.70453	0.00063
Test 1	5.58951	5.58992	5.59033	0.00041
Test 2	6.69718	6.69769	6.69819	0.00051
Test 3	4.45060	4.45114	4.45169	0.00055
Test 4	3.25985	3.26048	3.26110	0.00062

APPENDIX K.2: Ranking Results for Flow Shop Applications When $\overline{w}_1 > \overline{w}_2$

Table K.13 Ranking results for 85% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.60379	10.60414	10.60450	0.00035
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	1.77236	1.77323	1.77410	0.00087
SPT	8.53602	8.53646	8.53690	0.00044
PT/TIS	2.47037	2.47072	2.47107	0.00035
PT+WINQ+AT	10.39550	10.39586	10.39621	0.00035
PT+WINQ+SL	8.43260	8.43299	8.43337	0.00039
Q-Agent	1.84279	1.84347	1.84414	0.00067
Test 1	6.00000	6.00000	6.00000	0.00000
Test 2	7.03042	7.03056	7.03070	0.00014
Test 3	5.00000	5.00000	5.00000	0.00000
Test 4	3.91233	3.91258	3.91283	0.00025

Table K.14 Ranking results for 90% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	1.85916	1.85995	1.86075	0.00080
SPT	7.92939	7.92970	7.93000	0.00030
PT/TIS	3.66017	3.66169	3.66322	0.00152
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.64791	1.64865	1.64940	0.00074
Test 1	5.83112	5.83148	5.83185	0.00036
Test 2	6.93302	6.93321	6.93341	0.00020
Test 3	4.72778	4.72831	4.72884	0.00053
Test 4	3.40625	3.40700	3.40775	0.00075

Table K.15 Ranking results for 95% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	2.24526	2.24664	2.24801	0.00137
SPT	7.22590	7.22674	7.22758	0.00084
PT/TIS	7.61886	7.61940	7.61994	0.00054
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.42622	1.42676	1.42730	0.00054
Test 1	4.92384	4.92405	4.92426	0.00021
Test 2	6.00593	6.00599	6.00606	0.00006
Test 3	3.84709	3.84737	3.84765	0.00028
Test 4	2.70273	2.70306	2.70339	0.00033

Table K.16 Ranking results for 85% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.00000	10.00000	10.00000	0.00000
S/OPN	11.00000	11.00000	11.00000	0.00000
COVERT	1.08699	1.08721	1.08743	0.00022
SPT	8.04785	8.04851	8.04917	0.00066
PT/TIS	4.00000	4.00000	4.00000	0.00000
PT+WINQ+AT	12.00000	12.00000	12.00000	0.00000
PT+WINQ+SL	8.68509	8.68542	8.68575	0.00033
Q-Agent	1.91257	1.91279	1.91301	0.00022
Test 1	6.08296	6.08320	6.08343	0.00023
Test 2	7.17601	7.17630	7.17659	0.00029
Test 3	5.00652	5.00658	5.00663	0.00006
Test 4	3.00000	3.00000	3.00000	0.00000

Table K.17 Ranking results for 90% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.67925	10.67970	10.68015	0.00045
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	1.69043	1.69111	1.69179	0.00068
SPT	7.59728	7.59806	7.59885	0.00078
PT/TIS	3.60814	3.60964	3.61115	0.00151
PT+WINQ+AT	10.31985	10.32030	10.32075	0.00045
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.83240	1.83322	1.83404	0.00082
Test 1	5.97370	5.97384	5.97398	0.00014
Test 2	7.14020	7.14052	7.14084	0.00032
Test 3	4.76721	4.76760	4.76799	0.00039
Test 4	3.38528	3.38600	3.38672	0.00072

Table K.18 Ranking results for 95% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	2.57513	2.57674	2.57835	0.00161
SPT	6.95751	6.95902	6.96054	0.00152
PT/TIS	5.21262	5.21448	5.21633	0.00185
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.57402	1.57462	1.57523	0.00060
Test 1	5.63086	5.63122	5.63158	0.00036
Test 2	6.86183	6.86217	6.86250	0.00034
Test 3	4.28765	4.28824	4.28883	0.00059
Test 4	2.89288	2.89351	2.89414	0.00063

Table K.19 Ranking results for 85% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	12.28426	12.28468	12.28509	0.00042
AT-RPT	13.34279	13.34330	13.34381	0.00051
EDD	9.00000	9.00000	9.00000	0.00000
S/OPN	10.00000	10.00000	10.00000	0.00000
COVERT	1.07585	1.07606	1.07627	0.00021
SPT	13.30857	13.30959	13.31062	0.00103
PT/TIS	4.59826	4.60073	4.60320	0.00247
PT+WINQ+AT	11.06223	11.06243	11.06263	0.00020
PT+WINQ+SL	7.77243	7.77280	7.77316	0.00036
Q-Agent	2.31066	2.31127	2.31188	0.00061
Test 1	5.59038	5.59085	5.59131	0.00046
Test 2	6.62425	6.62474	6.62522	0.00048
Test 3	4.54188	4.54237	4.54285	0.00049
Test 4	3.48070	3.48120	3.48169	0.00049

Table K.20 Ranking results for 90% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.00000	10.00000	10.00000	0.00000
S/OPN	11.00000	11.00000	11.00000	0.00000
COVERT	1.52529	1.52592	1.52655	0.00063
SPT	8.10393	8.10483	8.10573	0.00090
PT/TIS	6.77740	6.77910	6.78080	0.00170
PT+WINQ+AT	12.00000	12.00000	12.00000	0.00000
PT+WINQ+SL	8.14274	8.14298	8.14322	0.00024
Q-Agent	1.62612	1.62652	1.62691	0.00040
Test 1	5.30092	5.30130	5.30169	0.00039
Test 2	6.43726	6.43767	6.43807	0.00041
Test 3	4.19783	4.19822	4.19861	0.00039
Test 4	2.88313	2.88346	2.88380	0.00033

Table K.21 Ranking results for 95% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	1.58865	1.58920	1.58975	0.00055
SPT	7.96765	7.96776	7.96787	0.00011
PT/TIS	2.66231	2.66279	2.66327	0.00048
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	2.06533	2.06596	2.06659	0.00063
Test 1	6.00000	6.00000	6.00000	0.00000
Test 2	7.03213	7.03224	7.03235	0.00011
Test 3	4.99419	4.99425	4.99431	0.00006
Test 4	3.68728	3.68781	3.68833	0.00053

Table K.22 Ranking results for 85% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	12.09497	12.09523	12.09549	0.00026
AT-RPT	13.13012	13.13038	13.13064	0.00026
EDD	8.46800	8.46904	8.47007	0.00104
S/OPN	9.55890	9.55980	9.56070	0.00090
COVERT	1.09030	1.09053	1.09076	0.00023
SPT	13.77389	13.77439	13.77488	0.00050
PT/TIS	3.97666	3.97900	3.98134	0.00234
PT+WINQ+AT	11.00000	11.00000	11.00000	0.00000
PT+WINQ+SL	8.08731	8.08843	8.08955	0.00112
Q-Agent	2.48190	2.48248	2.48305	0.00057
Test 1	5.87749	5.87808	5.87867	0.00059
Test 2	6.95501	6.95556	6.95611	0.00055
Test 3	4.79046	4.79104	4.79163	0.00059
Test 4	3.70548	3.70604	3.70661	0.00056

Table K.23 Ranking results for 90% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.00000	10.00000	10.00000	0.00000
S/OPN	11.00000	11.00000	11.00000	0.00000
COVERT	1.22084	1.22130	1.22176	0.00046
SPT	7.96771	7.96848	7.96925	0.00077
PT/TIS	8.02832	8.02906	8.02980	0.00074
PT+WINQ+AT	12.00000	12.00000	12.00000	0.00000
PT+WINQ+SL	7.99144	7.99151	7.99159	0.00008
Q-Agent	1.84041	1.84075	1.84109	0.00034
Test 1	5.00000	5.00000	5.00000	0.00000
Test 2	6.01089	6.01095	6.01101	0.00006
Test 3	4.00000	4.00000	4.00000	0.00000
Test 4	2.93779	2.93795	2.93811	0.00016

Table K.24 Ranking results for 95% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.45337	10.45377	10.45417	0.00040
S/OPN	11.85072	11.85099	11.85126	0.00027
COVERT	1.48818	1.48869	1.48921	0.00052
SPT	8.29032	8.29065	8.29098	0.00033
PT/TIS	2.74164	2.74226	2.74289	0.00062
PT+WINQ+AT	10.69466	10.69524	10.69583	0.00058
PT+WINQ+SL	8.70902	8.70935	8.70968	0.00033
Q-Agent	1.93430	1.93501	1.93572	0.00071
Test 1	6.00000	6.00000	6.00000	0.00000
Test 2	7.00000	7.00000	7.00000	0.00000
Test 3	5.00000	5.00000	5.00000	0.00000
Test 4	3.83359	3.83403	3.83447	0.00044

APPENDIX K.3: Ranking Results for Flow Shop Applications When the $\overline{w}_1 < \overline{w}_2$

Table K.25 Ranking results for 85% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	7.51271	7.51378	7.51485	0.00107
SPT	2.01101	2.01259	2.01418	0.00159
PT/TIS	3.01394	3.01459	3.01525	0.00066
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.65574	1.65610	1.65646	0.00036
Test 1	6.12743	6.12769	6.12795	0.00026
Test 2	7.20233	7.20272	7.20311	0.00039
Test 3	4.91376	4.91401	4.91426	0.00025
Test 4	3.55800	3.55852	3.55903	0.00052

Table K.26 Ranking results for 90% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	6.89248	6.89284	6.89319	0.00036
SPT	1.92887	1.92965	1.93042	0.00077
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.43326	1.43363	1.43399	0.00037
Test 1	4.96303	4.96320	4.96336	0.00017
Test 2	6.08206	6.08233	6.08261	0.00028
Test 3	3.92013	3.92041	3.92068	0.00027
Test 4	2.77766	2.77795	2.77825	0.00030

Table K.27 Ranking results for 95% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	7.00000	7.00000	7.00000	0.00000
SPT	1.79206	1.79286	1.79366	0.00080
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.42087	1.42133	1.42180	0.00046
Test 1	5.00000	5.00000	5.00000	0.00000
Test 2	6.00000	6.00000	6.00000	0.00000
Test 3	3.96770	3.96786	3.96802	0.00016
Test 4	2.81760	2.81794	2.81828	0.00034

Table K.28 Ranking results for 85% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.44427	10.44468	10.44510	0.00042
S/OPN	11.68353	11.68391	11.68430	0.00039
COVERT	6.53009	6.53178	6.53347	0.00169
SPT	1.48792	1.48861	1.48931	0.00070
PT/TIS	4.19039	4.19072	4.19104	0.00032
PT+WINQ+AT	10.87067	10.87140	10.87214	0.00073
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.72145	1.72184	1.72223	0.00039
Test 1	6.32177	6.32216	6.32256	0.00039
Test 2	7.39272	7.39308	7.39344	0.00036
Test 3	5.25097	5.25130	5.25163	0.00033
Test 4	3.10024	3.10051	3.10077	0.00026

Table K.29 Ranking results for 90% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	6.71335	6.71384	6.71433	0.00049
SPT	1.76937	1.77016	1.77096	0.00079
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.46382	1.46430	1.46478	0.00048
Test 1	5.10011	5.10034	5.10057	0.00023
Test 2	6.18171	6.18200	6.18228	0.00028
Test 3	3.98807	3.98815	3.98823	0.00008
Test 4	2.78084	2.78121	2.78158	0.00037

Table K.30 Ranking results for 95% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	7.00000	7.00000	7.00000	0.00000
SPT	1.62177	1.62241	1.62305	0.00064
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.48116	1.48164	1.48213	0.00048
Test 1	5.00000	5.00000	5.00000	0.00000
Test 2	6.00000	6.00000	6.00000	0.00000
Test 3	4.00000	4.00000	4.00000	0.00000
Test 4	2.89571	2.89595	2.89619	0.00024

Table K.31 Ranking results for 85% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.25969	10.26008	10.26047	0.00039
S/OPN	11.48796	11.48839	11.48882	0.00043
COVERT	6.53894	6.54043	6.54192	0.00149
SPT	8.05721	8.05879	8.06037	0.00158
PT/TIS	8.30727	8.30754	8.30782	0.00028
PT+WINQ+AT	11.20477	11.20535	11.20593	0.00058
PT+WINQ+SL	1.61582	1.61712	1.61841	0.00129
Q-Agent	1.84956	1.85004	1.85053	0.00048
Test 1	5.23574	5.23612	5.23649	0.00038
Test 2	6.33337	6.33376	6.33415	0.00039
Test 3	4.11264	4.11288	4.11313	0.00025
Test 4	2.98908	2.98950	2.98992	0.00042

Table K.32 Ranking results for 90% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.56652	10.56694	10.56736	0.00042
S/OPN	11.76842	11.76880	11.76918	0.00038
COVERT	6.64787	6.64857	6.64926	0.00069
SPT	1.83283	1.83384	1.83486	0.00101
PT/TIS	9.00000	9.00000	9.00000	0.00000
PT+WINQ+AT	10.66352	10.66426	10.66499	0.00074
PT+WINQ+SL	8.00000	8.00000	8.00000	0.00000
Q-Agent	1.54520	1.54566	1.54612	0.00046
Test 1	5.11253	5.11276	5.11299	0.00023
Test 2	6.18521	6.18558	6.18596	0.00037
Test 3	3.89469	3.89492	3.89514	0.00023
Test 4	2.77831	2.77867	2.77903	0.00036

Table K.33 Ranking results for 95% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	7.66600	7.66663	7.66726	0.00063
SPT	1.95614	1.95719	1.95825	0.00106
PT/TIS	6.59731	6.59796	6.59861	0.00065
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.56488	1.56528	1.56568	0.00040
Test 1	5.18951	5.18993	5.19035	0.00042
Test 2	6.37652	6.37712	6.37772	0.00060
Test 3	3.87439	3.87465	3.87491	0.00026
Test 4	2.77095	2.77123	2.77152	0.00029

Table K.34 Ranking results for 85% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	9.91550	9.91580	9.91611	0.00031
S/OPN	11.27718	11.27770	11.27823	0.00053
COVERT	5.79460	5.79608	5.79757	0.00149
SPT	8.54696	8.54860	8.55025	0.00164
PT/TIS	8.29488	8.29526	8.29565	0.00039
PT+WINQ+AT	11.49290	11.49346	11.49402	0.00056
PT+WINQ+SL	1.32262	1.32327	1.32392	0.00065
Q-Agent	1.88695	1.88734	1.88772	0.00038
Test 1	5.42240	5.42281	5.42322	0.00041
Test 2	6.48586	6.48635	6.48685	0.00050
Test 3	4.33836	4.33878	4.33921	0.00043
Test 4	3.21412	3.21453	3.21494	0.00041

Table K.35 Ranking results for 90% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.23395	10.23424	10.23454	0.00029
S/OPN	11.35957	11.35994	11.36031	0.00037
COVERT	6.47672	6.47776	6.47880	0.00104
SPT	1.96340	1.96475	1.96609	0.00134
PT/TIS	9.00000	9.00000	9.00000	0.00000
PT+WINQ+AT	11.40524	11.40582	11.40640	0.00058
PT+WINQ+SL	8.00000	8.00000	8.00000	0.00000
Q-Agent	1.60202	1.60236	1.60270	0.00034
Test 1	5.05981	5.05999	5.06018	0.00018
Test 2	6.22686	6.22723	6.22760	0.00037
Test 3	3.92839	3.92864	3.92888	0.00024
Test 4	2.73888	2.73927	2.73966	0.00039

Table K.36 Ranking results for 95% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	6.82565	6.82697	6.82829	0.00132
SPT	2.57565	2.57750	2.57935	0.00185
PT/TIS	6.95555	6.95645	6.95735	0.00090
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.47269	1.47312	1.47355	0.00043
Test 1	5.17995	5.18026	5.18058	0.00031
Test 2	6.39505	6.39561	6.39618	0.00057
Test 3	3.90246	3.90283	3.90320	0.00037
Test 4	2.68688	2.68726	2.68764	0.00038

APPENDIX L.1: Ranking Results for Multi-State Flow Shop Applications

Table L.1 Ranking results for 85% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.80167	10.80195	10.80223	0.00028
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.69086	4.69305	4.69524	0.00219
SPT	5.29078	5.29332	5.29587	0.00254
PT/TIS	3.08854	3.08934	3.09014	0.00080
PT+WINQ+AT	10.19777	10.19805	10.19833	0.00028
PT+WINQ+SL	8.71598	8.71628	8.71659	0.00031
Q-Agent	1.57530	1.57566	1.57602	0.00036
Test 1	6.07325	6.07342	6.07358	0.00017
Test 2	7.14785	7.14813	7.14840	0.00027
Test 3	4.91067	4.91090	4.91112	0.00023
Test 4	3.49948	3.49990	3.50032	0.00042

Table L.2 Ranking results for 90% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.45521	4.45728	4.45935	0.00207
SPT	5.00858	5.01105	5.01352	0.00247
PT/TIS	5.90987	5.91174	5.91361	0.00187
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.47186	1.47244	1.47302	0.00058
Test 1	5.39722	5.39763	5.39804	0.00041
Test 2	6.50759	6.50798	6.50837	0.00039
Test 3	4.32417	4.32458	4.32499	0.00041
Test 4	2.91672	2.91730	2.91789	0.00058

Table L.3 Ranking results for 95% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.66967	4.67136	4.67305	0.00169
SPT	4.59068	4.59261	4.59453	0.00193
PT/TIS	7.80979	7.81002	7.81026	0.00024
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.38009	1.38039	1.38070	0.00031
Test 1	4.95239	4.95259	4.95278	0.00019
Test 2	6.01733	6.01745	6.01757	0.00012
Test 3	3.88989	3.89014	3.89039	0.00025
Test 4	2.68510	2.68544	2.68578	0.00034

Table L.4 Ranking results for 85% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.22158	10.22197	10.22236	0.00039
S/OPN	11.34130	11.34170	11.34211	0.00041
COVERT	3.84785	3.85032	3.85278	0.00247
SPT	4.72389	4.72628	4.72867	0.00239
PT/TIS	4.32777	4.32813	4.32848	0.00036
PT+WINQ+AT	11.43560	11.43633	11.43706	0.00073
PT+WINQ+SL	8.84220	8.84251	8.84283	0.00032
Q-Agent	1.75524	1.75561	1.75598	0.00037
Test 1	6.23618	6.23655	6.23693	0.00038
Test 2	7.36532	7.36575	7.36617	0.00043
Test 3	4.90347	4.90393	4.90440	0.00047
Test 4	2.99085	2.99092	2.99099	0.00007

Table L.5 Ranking results for 90% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.83951	10.83976	10.84000	0.00024
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.24288	4.24506	4.24724	0.00218
SPT	4.67868	4.68140	4.68412	0.00272
PT/TIS	5.81917	5.82138	5.82358	0.00220
PT+WINQ+AT	10.16000	10.16024	10.16049	0.00024
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.55247	1.55284	1.55322	0.00037
Test 1	5.55826	5.55872	5.55918	0.00046
Test 2	6.72800	6.72863	6.72926	0.00063
Test 3	4.39931	4.39972	4.40014	0.00042
Test 4	3.01163	3.01225	3.01286	0.00061

Table L.6 Ranking results for 95% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.82304	4.82522	4.82740	0.00218
SPT	4.35157	4.35400	4.35643	0.00243
PT/TIS	6.64752	6.64942	6.65133	0.00190
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.44254	1.44302	1.44349	0.00047
Test 1	5.31508	5.31550	5.31593	0.00043
Test 2	6.43065	6.43108	6.43152	0.00043
Test 3	4.14335	4.14378	4.14421	0.00043
Test 4	2.83748	2.83797	2.83847	0.00049

Table L.7 Ranking results for 85% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	12.64227	12.64264	12.64301	0.00037
AT-RPT	13.67147	13.67186	13.67224	0.00039
EDD	9.62960	9.63013	9.63066	0.00053
S/OPN	10.74376	10.74432	10.74488	0.00056
COVERT	3.68631	3.68829	3.69028	0.00199
SPT	10.65546	10.65783	10.66020	0.00237
PT/TIS	6.44952	6.45163	6.45374	0.00211
PT+WINQ+AT	11.13315	11.13364	11.13412	0.00049
PT+WINQ+SL	4.64440	4.64667	4.64895	0.00227
Q-Agent	1.96943	1.96998	1.97054	0.00055
Test 1	5.15137	5.15207	5.15276	0.00070
Test 2	6.64021	6.64068	6.64114	0.00046
Test 3	4.83181	4.83226	4.83271	0.00045
Test 4	3.13748	3.13801	3.13853	0.00052

Table L.8 Ranking results for 90% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.28314	10.28351	10.28389	0.00038
S/OPN	11.38410	11.38448	11.38486	0.00038
COVERT	4.14732	4.14952	4.15172	0.00220
SPT	5.01081	5.01353	5.01625	0.00272
PT/TIS	7.90719	7.90850	7.90980	0.00130
PT+WINQ+AT	11.33130	11.33201	11.33272	0.00071
PT+WINQ+SL	8.07108	8.07129	8.07151	0.00022
Q-Agent	1.52430	1.52470	1.52510	0.00040
Test 1	5.20666	5.20693	5.20719	0.00026
Test 2	6.31111	6.31143	6.31176	0.00033
Test 3	4.04566	4.04602	4.04637	0.00036
Test 4	2.76775	2.76809	2.76842	0.00033

Table L.9 Ranking results for 95% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	4.67324	4.67529	4.67733	0.00205
SPT	4.99711	4.99922	5.00132	0.00211
PT/TIS	4.68644	4.68791	4.68938	0.00147
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.75972	1.76040	1.76109	0.00068
Test 1	5.59827	5.59872	5.59917	0.00045
Test 2	6.69869	6.69928	6.69988	0.00060
Test 3	4.42959	4.43005	4.43051	0.00046
Test 4	3.14856	3.14913	3.14970	0.00057

Table L.10 Ranking results for 85% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	12.54752	12.54796	12.54840	0.00044
AT-RPT	13.56505	13.56548	13.56592	0.00043
EDD	9.19074	9.19171	9.19268	0.00097
S/OPN	10.41389	10.41491	10.41592	0.00101
COVERT	3.70122	3.70401	3.70681	0.00279
SPT	11.20181	11.20472	11.20763	0.00291
PT/TIS	6.19485	6.19749	6.20014	0.00264
PT+WINQ+AT	11.24666	11.24700	11.24734	0.00034
PT+WINQ+SL	4.84651	4.84963	4.85276	0.00312
Q-Agent	2.04789	2.04837	2.04886	0.00049
Test 1	5.57927	5.57989	5.58051	0.00062
Test 2	6.63033	6.63092	6.63151	0.00059
Test 3	4.50466	4.50530	4.50595	0.00065
Test 4	3.31212	3.31260	3.31308	0.00048

Table L.11 Ranking results for 90% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.11669	10.11694	10.11718	0.00025
S/OPN	11.17962	11.17990	11.18018	0.00028
COVERT	3.91980	3.92239	3.92497	0.00258
SPT	5.05083	5.05399	5.05715	0.00316
PT/TIS	8.51378	8.51458	8.51539	0.00080
PT+WINQ+AT	11.70265	11.70316	11.70367	0.00051
PT+WINQ+SL	7.99566	7.99571	7.99576	0.00005
Q-Agent	1.61745	1.61785	1.61825	0.00040
Test 1	5.02990	5.03005	5.03020	0.00015
Test 2	6.11889	6.11913	6.11938	0.00024
Test 3	3.96405	3.96421	3.96437	0.00016
Test 4	2.78179	2.78209	2.78239	0.00030

Table L.12 Ranking results for 95% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.72637	10.72673	10.72708	0.00035
S/OPN	11.92498	11.92521	11.92544	0.00023
COVERT	4.19472	4.19682	4.19892	0.00210
SPT	5.45400	5.45636	5.45872	0.00236
PT/TIS	4.89064	4.89228	4.89393	0.00164
PT+WINQ+AT	10.34753	10.34806	10.34860	0.00053
PT+WINQ+SL	8.85406	8.85438	8.85470	0.00032
Q-Agent	1.63720	1.63778	1.63836	0.00058
Test 1	5.59345	5.59379	5.59413	0.00034
Test 2	6.67788	6.67833	6.67878	0.00045
Test 3	4.45752	4.45799	4.45847	0.00048
Test 4	3.23164	3.23226	3.23288	0.00062

APPENDIX L.2: Ranking Results for Multi-State Flow Shop Applications When $\overline{w}_1 > \overline{w}_2$

Table L.13 Ranking results for 85% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.60369	10.60399	10.60429	0.00030
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	1.92632	1.92731	1.92830	0.00099
SPT	8.50054	8.50103	8.50151	0.00048
PT/TIS	2.68570	2.68609	2.68648	0.00039
PT+WINQ+AT	10.39571	10.39601	10.39631	0.00030
PT+WINQ+SL	8.43256	8.43291	8.43326	0.00035
Q-Agent	1.54384	1.54419	1.54454	0.00035
Test 1	6.00000	6.00000	6.00000	0.00000
Test 2	7.06583	7.06606	7.06630	0.00023
Test 3	5.00000	5.00000	5.00000	0.00000
Test 4	3.84205	3.84241	3.84276	0.00035

Table L.14 Ranking results for 90% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	2.02468	2.02533	2.02598	0.00065
SPT	7.92925	7.92950	7.92975	0.00025
PT/TIS	3.82546	3.82659	3.82771	0.00112
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.56771	1.56816	1.56861	0.00045
Test 1	5.83106	5.83135	5.83163	0.00028
Test 2	6.93286	6.93304	6.93321	0.00018
Test 3	4.72755	4.72797	4.72840	0.00042
Test 4	3.15743	3.15806	3.15870	0.00063

Table L.15 Ranking results for 95% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	2.34296	2.34442	2.34588	0.00146
SPT	7.22216	7.22290	7.22364	0.00074
PT/TIS	7.61921	7.61955	7.61990	0.00035
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.38853	1.38886	1.38919	0.00033
Test 1	4.90501	4.90525	4.90550	0.00025
Test 2	6.03480	6.03492	6.03504	0.00012
Test 3	3.83652	3.83684	3.83716	0.00032
Test 4	2.64678	2.64726	2.64773	0.00048

Table L.16 Ranking results for 85% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.00000	10.00000	10.00000	0.00000
S/OPN	11.00000	11.00000	11.00000	0.00000
COVERT	1.15451	1.15485	1.15518	0.00034
SPT	7.84771	7.84850	7.84928	0.00079
PT/TIS	4.00000	4.00000	4.00000	0.00000
PT+WINQ+AT	12.00000	12.00000	12.00000	0.00000
PT+WINQ+SL	8.68466	8.68502	8.68537	0.00035
Q-Agent	1.84482	1.84515	1.84549	0.00034
Test 1	6.14159	6.14190	6.14221	0.00031
Test 2	7.29793	7.29830	7.29866	0.00037
Test 3	5.02616	5.02629	5.02642	0.00013
Test 4	3.00000	3.00000	3.00000	0.00000

Table L.17 Ranking results for 90% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.67926	10.67967	10.68007	0.00040
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	1.80654	1.80730	1.80806	0.00076
SPT	7.49957	7.50045	7.50132	0.00088
PT/TIS	3.64533	3.64683	3.64833	0.00150
PT+WINQ+AT	10.31993	10.32033	10.32074	0.00040
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.68295	1.68373	1.68452	0.00078
Test 1	6.01135	6.01146	6.01157	0.00011
Test 2	7.25708	7.25744	7.25780	0.00036
Test 3	4.79464	4.79494	4.79524	0.00030
Test 4	3.29700	3.29785	3.29869	0.00085

Table L.18 Ranking results for 95% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	2.64961	2.65091	2.65222	0.00131
SPT	6.95592	6.95719	6.95846	0.00127
PT/TIS	5.29811	5.29953	5.30096	0.00142
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.48855	1.48904	1.48953	0.00049
Test 1	5.63067	5.63102	5.63137	0.00035
Test 2	6.86183	6.86208	6.86234	0.00025
Test 3	4.28665	4.28729	4.28793	0.00064
Test 4	2.82244	2.82293	2.82342	0.00049

Table L.19 Ranking results for 85% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	12.28433	12.28473	12.28512	0.00039
AT-RPT	13.34303	13.34345	13.34387	0.00042
EDD	9.00000	9.00000	9.00000	0.00000
S/OPN	10.00000	10.00000	10.00000	0.00000
COVERT	1.19315	1.19346	1.19378	0.00031
SPT	13.30850	13.30939	13.31028	0.00089
PT/TIS	4.59118	4.59335	4.59551	0.00217
PT+WINQ+AT	11.06219	11.06243	11.06267	0.00024
PT+WINQ+SL	7.66276	7.66333	7.66391	0.00058
Q-Agent	2.18340	2.18389	2.18438	0.00049
Test 1	5.62480	5.62523	5.62566	0.00043
Test 2	6.75212	6.75254	6.75297	0.00043
Test 3	4.58055	4.58096	4.58137	0.00041
Test 4	3.40674	3.40723	3.40772	0.00049

Table L.20 Ranking results for 90% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.00000	10.00000	10.00000	0.00000
S/OPN	11.00000	11.00000	11.00000	0.00000
COVERT	1.64742	1.64819	1.64896	0.00077
SPT	8.10295	8.10398	8.10500	0.00102
PT/TIS	6.81265	6.81444	6.81623	0.00179
PT+WINQ+AT	12.00000	12.00000	12.00000	0.00000
PT+WINQ+SL	8.14256	8.14280	8.14304	0.00024
Q-Agent	1.55295	1.55339	1.55384	0.00045
Test 1	5.30067	5.30109	5.30152	0.00042
Test 2	6.43760	6.43799	6.43838	0.00039
Test 3	4.19747	4.19787	4.19826	0.00039
Test 4	2.79987	2.80025	2.80062	0.00038

Table L.21 Ranking results for 95% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	1.68575	1.68653	1.68730	0.00077
SPT	7.97556	7.97567	7.97579	0.00011
PT/TIS	2.78185	2.78253	2.78322	0.00068
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.98301	1.98384	1.98467	0.00083
Test 1	6.00000	6.00000	6.00000	0.00000
Test 2	7.02421	7.02433	7.02444	0.00011
Test 3	4.98304	4.98316	4.98328	0.00012
Test 4	3.56320	3.56394	3.56469	0.00074

Table L.22 Ranking results for 85% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	12.09514	12.09539	12.09564	0.00025
AT-RPT	13.13007	13.13035	13.13063	0.00028
EDD	8.46554	8.46630	8.46707	0.00077
S/OPN	9.55021	9.55094	9.55167	0.00073
COVERT	1.21675	1.21707	1.21740	0.00033
SPT	13.77375	13.77425	13.77476	0.00050
PT/TIS	4.09405	4.09582	4.09759	0.00177
PT+WINQ+AT	11.00000	11.00000	11.00000	0.00000
PT+WINQ+SL	8.17617	8.17687	8.17757	0.00070
Q-Agent	2.31193	2.31245	2.31297	0.00052
Test 1	5.83864	5.83912	5.83960	0.00048
Test 2	6.89752	6.89802	6.89852	0.00050
Test 3	4.78757	4.78802	4.78847	0.00045
Test 4	3.65491	3.65538	3.65586	0.00048

Table L.23 Ranking results for 90% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.00000	10.00000	10.00000	0.00000
S/OPN	11.00000	11.00000	11.00000	0.00000
COVERT	1.36835	1.36884	1.36933	0.00049
SPT	7.96712	7.96793	7.96874	0.00081
PT/TIS	8.02869	8.02951	8.03033	0.00082
PT+WINQ+AT	12.00000	12.00000	12.00000	0.00000
PT+WINQ+SL	7.99149	7.99157	7.99165	0.00008
Q-Agent	1.74445	1.74477	1.74509	0.00032
Test 1	5.00000	5.00000	5.00000	0.00000
Test 2	6.01091	6.01099	6.01107	0.00008
Test 3	4.00000	4.00000	4.00000	0.00000
Test 4	2.88615	2.88639	2.88664	0.00024

Table L.24 Ranking results for 95% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.45332	10.45372	10.45412	0.00040
S/OPN	11.85091	11.85116	11.85141	0.00025
COVERT	1.55773	1.55825	1.55877	0.00052
SPT	8.29004	8.29044	8.29084	0.00040
PT/TIS	2.82621	2.82673	2.82725	0.00052
PT+WINQ+AT	10.69453	10.69512	10.69571	0.00059
PT+WINQ+SL	8.70916	8.70956	8.70996	0.00040
Q-Agent	1.82145	1.82217	1.82290	0.00072
Test 1	6.00000	6.00000	6.00000	0.00000
Test 2	7.00000	7.00000	7.00000	0.00000
Test 3	5.00000	5.00000	5.00000	0.00000
Test 4	3.79240	3.79285	3.79329	0.00045

APPENDIX L.3: Ranking Results for Multi-State Flow Shop Applications When $\overline{w}_1 < \overline{w}_2$

Table L.25 Ranking results for 85% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	7.45987	7.46074	7.46161	0.00087
SPT	2.08200	2.08326	2.08452	0.00126
PT/TIS	3.49208	3.49291	3.49375	0.00084
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.60676	1.60712	1.60747	0.00035
Test 1	6.14637	6.14664	6.14692	0.00027
Test 2	7.22971	7.23006	7.23040	0.00034
Test 3	4.82169	4.82192	4.82215	0.00023
Test 4	3.15706	3.15735	3.15764	0.00029

Table L.26 Ranking results for 90% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	6.89267	6.89299	6.89330	0.00032
SPT	2.08918	2.09002	2.09086	0.00084
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.37507	1.37550	1.37593	0.00043
Test 1	4.96293	4.96311	4.96328	0.00018
Test 2	6.08197	6.08218	6.08239	0.00021
Test 3	3.92033	3.92054	3.92075	0.00021
Test 4	2.67534	2.67566	2.67598	0.00032

Table L.27 Ranking results for 95% utilization level and *k*=3

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	7.00000	7.00000	7.00000	0.00000
SPT	1.95960	1.96047	1.96134	0.00087
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.37212	1.37266	1.37321	0.00055
Test 1	5.00000	5.00000	5.00000	0.00000
Test 2	6.00000	6.00000	6.00000	0.00000
Test 3	3.94290	3.94306	3.94322	0.00016
Test 4	2.72341	2.72381	2.72421	0.00040

Table L.28 Ranking results for 85% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.44423	10.44468	10.44512	0.00044
S/OPN	11.68344	11.68374	11.68405	0.00030
COVERT	6.54742	6.54871	6.55001	0.00129
SPT	1.60089	1.60154	1.60218	0.00064
PT/TIS	4.65637	4.65678	4.65719	0.00041
PT+WINQ+AT	10.87093	10.87158	10.87222	0.00065
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.66624	1.66653	1.66683	0.00029
Test 1	6.33052	6.33082	6.33112	0.00030
Test 2	7.43235	7.43270	7.43306	0.00035
Test 3	4.78051	4.78111	4.78170	0.00059
Test 4	2.98168	2.98180	2.98192	0.00012

Table L.29 Ranking results for 90% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	6.68684	6.68728	6.68773	0.00045
SPT	1.85731	1.85793	1.85854	0.00062
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.42112	1.42153	1.42195	0.00042
Test 1	5.10482	5.10500	5.10517	0.00017
Test 2	6.19825	6.19854	6.19883	0.00029
Test 3	4.00362	4.00368	4.00374	0.00006
Test 4	2.72577	2.72604	2.72630	0.00027

Table L.30 Ranking results for 95% utilization level and *k*=4

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	7.00000	7.00000	7.00000	0.00000
SPT	1.74903	1.74984	1.75065	0.00081
PT/TIS	8.00000	8.00000	8.00000	0.00000
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.39638	1.39692	1.39746	0.00054
Test 1	5.00000	5.00000	5.00000	0.00000
Test 2	6.00000	6.00000	6.00000	0.00000
Test 3	4.00000	4.00000	4.00000	0.00000
Test 4	2.85286	2.85324	2.85362	0.00038

Table L.31 Ranking results for 85% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.25938	10.25976	10.26015	0.00039
S/OPN	11.48819	11.48870	11.48920	0.00050
COVERT	6.18050	6.18207	6.18364	0.00157
SPT	8.00888	8.01049	8.01210	0.00161
PT/TIS	8.30687	8.30726	8.30765	0.00039
PT+WINQ+AT	11.20467	11.20543	11.20619	0.00076
PT+WINQ+SL	1.62997	1.63098	1.63198	0.00100
Q-Agent	1.75558	1.75611	1.75663	0.00052
Test 1	4.67783	4.67867	4.67951	0.00084
Test 2	6.52815	6.52860	6.52905	0.00045
Test 3	5.08303	5.08336	5.08368	0.00032
Test 4	2.86811	2.86858	2.86905	0.00047

Table L.32 Ranking results for 90% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.56625	10.56677	10.56729	0.00052
S/OPN	11.76867	11.76906	11.76944	0.00038
COVERT	6.64821	6.64889	6.64958	0.00069
SPT	1.92469	1.92579	1.92689	0.00110
PT/TIS	9.00000	9.00000	9.00000	0.00000
PT+WINQ+AT	10.66333	10.66418	10.66503	0.00085
PT+WINQ+SL	8.00000	8.00000	8.00000	0.00000
Q-Agent	1.49535	1.49583	1.49631	0.00048
Test 1	5.11246	5.11270	5.11294	0.00024
Test 2	6.18494	6.18527	6.18560	0.00033
Test 3	3.89474	3.89498	3.89522	0.00024
Test 4	2.73615	2.73653	2.73692	0.00039

Table L.33 Ranking results for 95% utilization level and *k*=5

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	7.66490	7.66568	7.66647	0.00079
SPT	2.02051	2.02189	2.02327	0.00138
PT/TIS	6.59283	6.59367	6.59451	0.00084
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.53616	1.53657	1.53699	0.00041
Test 1	5.19688	5.19742	5.19796	0.00054
Test 2	6.37396	6.37469	6.37542	0.00073
Test 3	3.87605	3.87637	3.87669	0.00032
Test 4	2.73336	2.73371	2.73405	0.00035

Table L.34 Ranking results for 85% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	9.91549	9.91584	9.91618	0.00034
S/OPN	11.27709	11.27761	11.27813	0.00052
COVERT	6.18688	6.18830	6.18972	0.00142
SPT	8.63616	8.63759	8.63902	0.00143
PT/TIS	8.29462	8.29496	8.29530	0.00034
PT+WINQ+AT	11.49337	11.49377	11.49418	0.00041
PT+WINQ+SL	1.52602	1.52692	1.52781	0.00090
Q-Agent	1.78440	1.78478	1.78516	0.00038
Test 1	5.32117	5.32151	5.32185	0.00034
Test 2	6.36419	6.36455	6.36492	0.00036
Test 3	4.22302	4.22331	4.22360	0.00029
Test 4	2.97062	2.97086	2.97110	0.00024

Table L.35 Ranking results for 90% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	10.23379	10.23411	10.23442	0.00032
S/OPN	11.35968	11.36008	11.36047	0.00039
COVERT	6.47683	6.47780	6.47876	0.00097
SPT	2.13737	2.13871	2.14006	0.00135
PT/TIS	9.00000	9.00000	9.00000	0.00000
PT+WINQ+AT	11.40517	11.40582	11.40647	0.00065
PT+WINQ+SL	8.00000	8.00000	8.00000	0.00000
Q-Agent	1.49013	1.49054	1.49094	0.00041
Test 1	5.05988	5.06009	5.06030	0.00021
Test 2	6.22701	6.22741	6.22781	0.00040
Test 3	3.92815	3.92838	3.92860	0.00022
Test 4	2.67664	2.67707	2.67750	0.00043

Table L.36 Ranking results for 95% utilization level and *k*=6

Methods	Maximum	Average	Minimum	Half Width
FIFO	13.00000	13.00000	13.00000	0.00000
AT-RPT	14.00000	14.00000	14.00000	0.00000
EDD	11.00000	11.00000	11.00000	0.00000
S/OPN	12.00000	12.00000	12.00000	0.00000
COVERT	6.83639	6.83766	6.83893	0.00127
SPT	2.61756	2.61937	2.62119	0.00182
PT/TIS	6.96010	6.96070	6.96129	0.00060
PT+WINQ+AT	10.00000	10.00000	10.00000	0.00000
PT+WINQ+SL	9.00000	9.00000	9.00000	0.00000
Q-Agent	1.45166	1.45213	1.45261	0.00048
Test 1	5.18686	5.18714	5.18741	0.00028
Test 2	6.35591	6.35648	6.35705	0.00057
Test 3	3.91503	3.91543	3.91583	0.00040
Test 4	2.67066	2.67109	2.67152	0.00043

APPENDIX M.1: Ranks of the First Three Methods for Single Machine

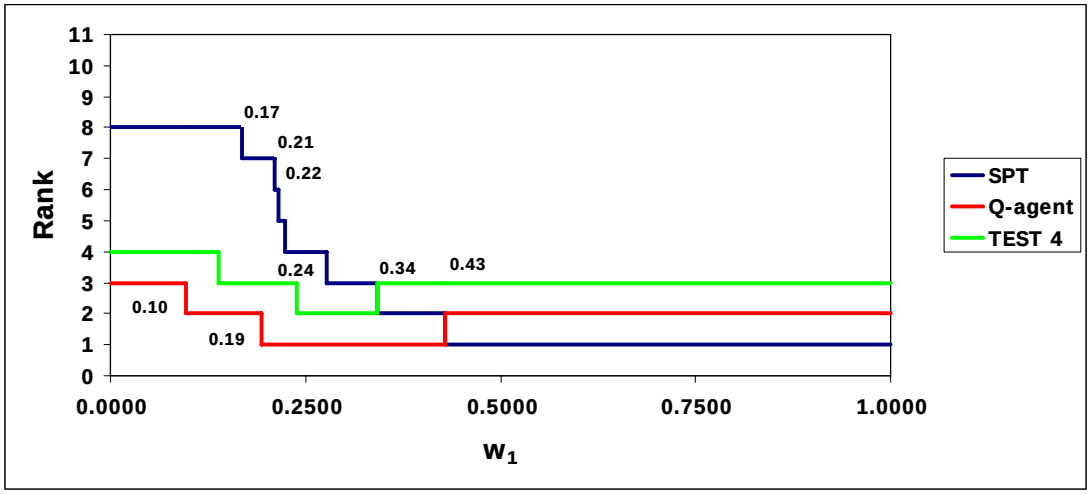

Figure M.1 Ranks of the first three methods for 80% utilization level and tight due dates

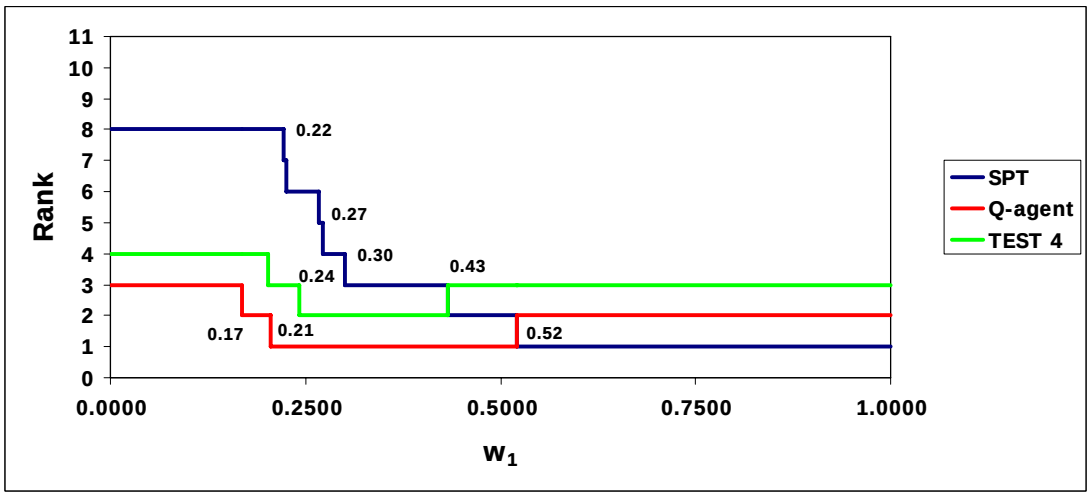

Figure M.2 Ranks of the first three methods for 85% utilization level and tight due dates

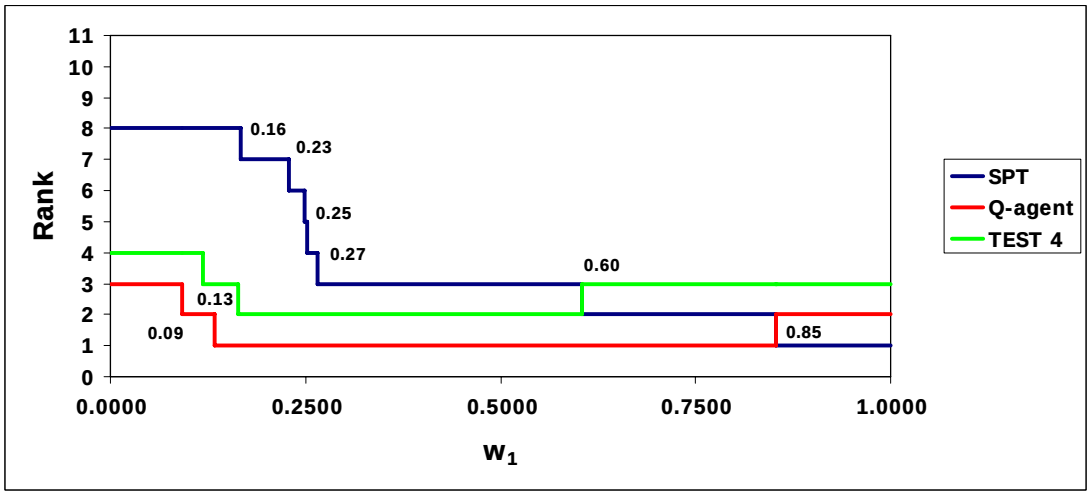

Figure M.3 Ranks of the first three methods for 90% utilization level and tight due dates

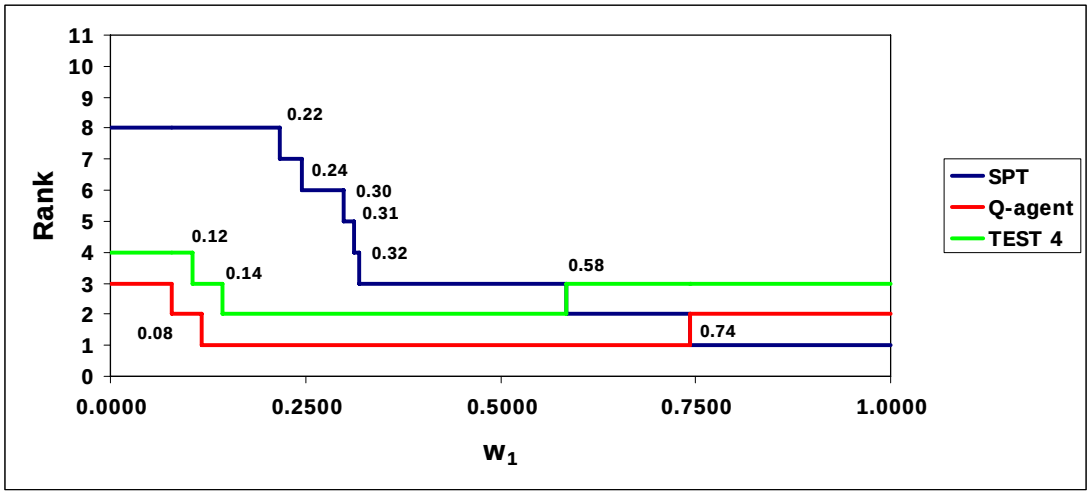

Figure M.4 Ranks of the first three methods for 95% utilization level and tight due dates

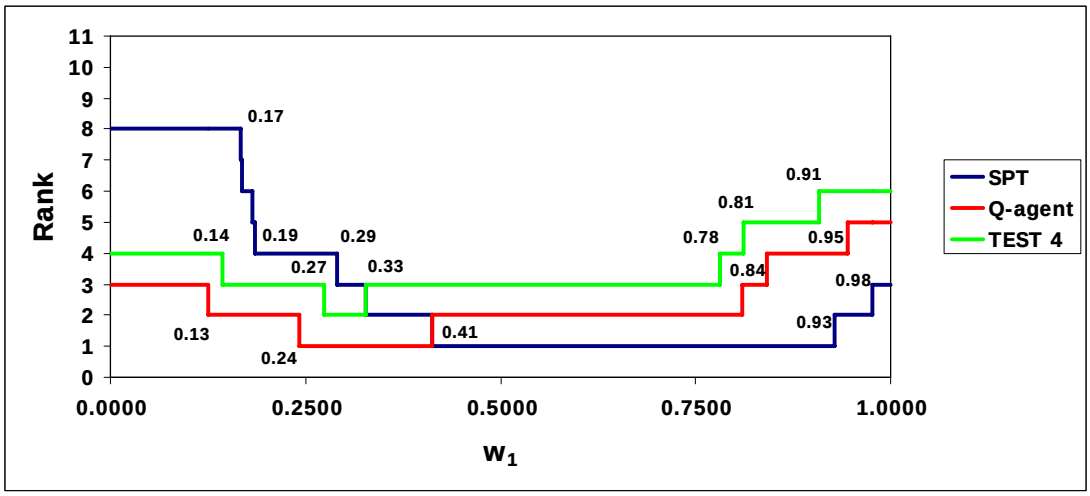

Figure M.5 Ranks of the first three methods for 80% utilization level and loose due dates

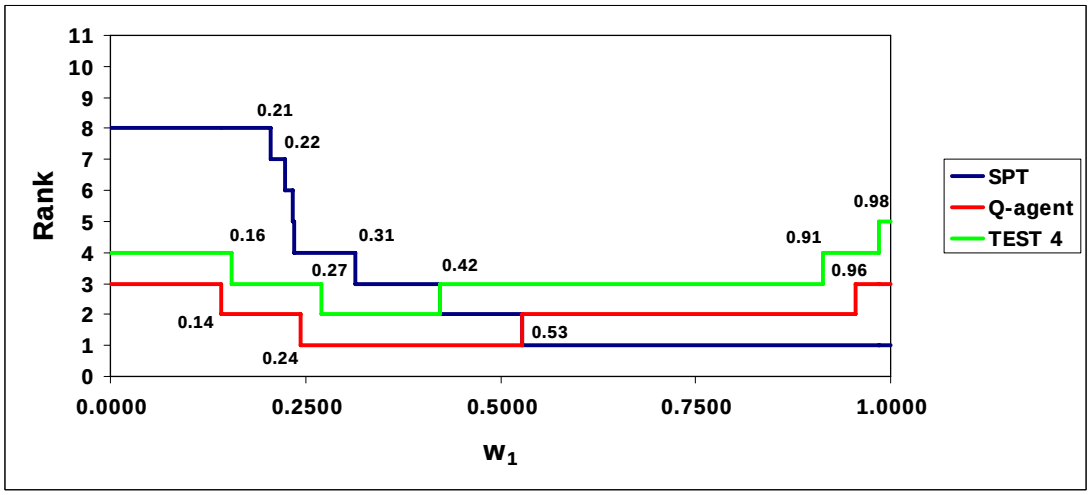

Figure M.6 Ranks of the first three methods for 85% utilization level and loose due dates

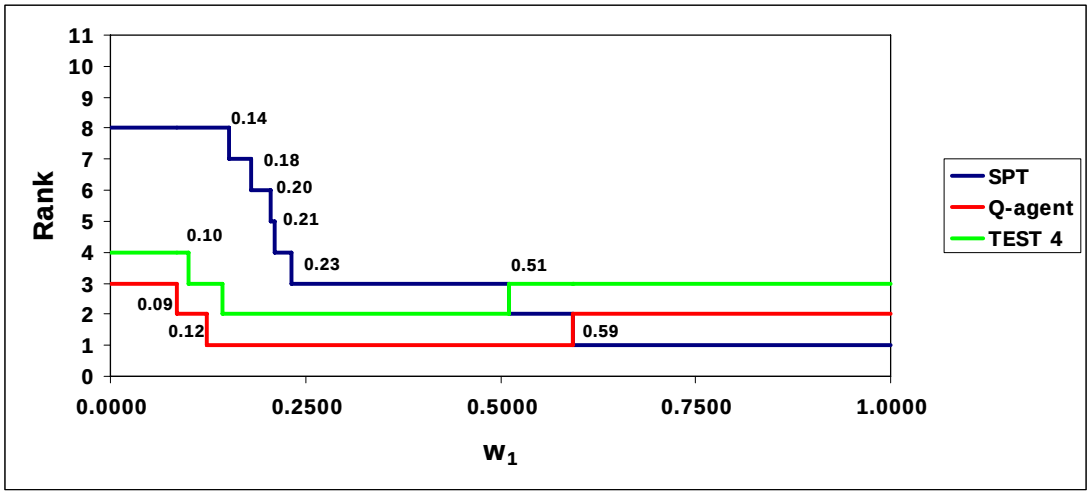

Figure M.7 Ranks of the first three methods for 90% utilization level and loose due dates

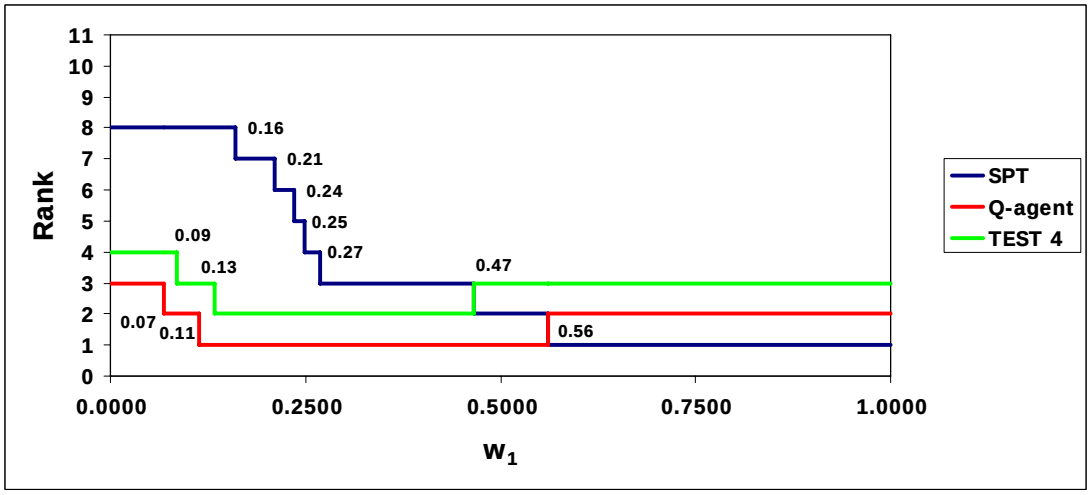

Figure M.8 Ranks of the first three methods for 95% utilization level and loose due dates

APPENDIX M.2: Ranks of the First Three Methods for Flow Shop

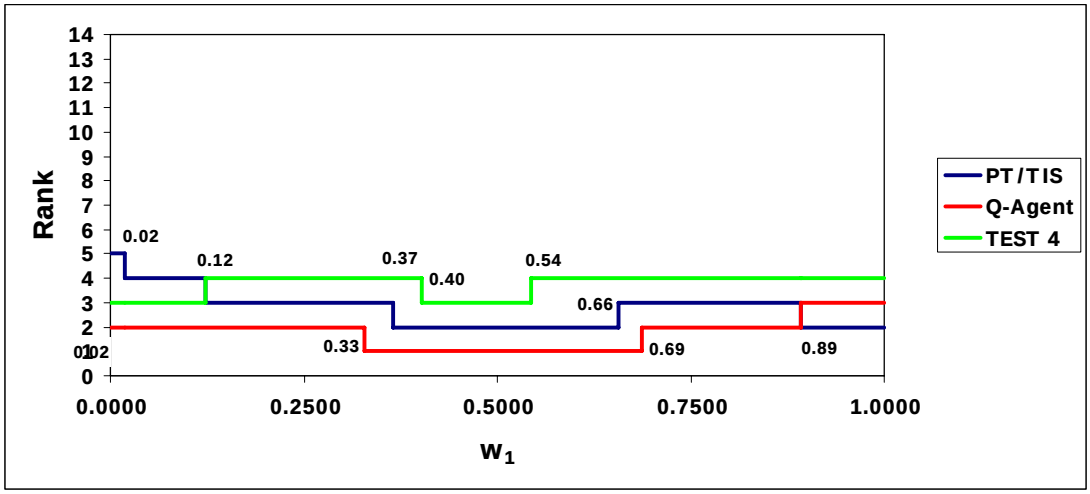

Figure M.9 Ranks of the first three methods for 85% utilization level k=3

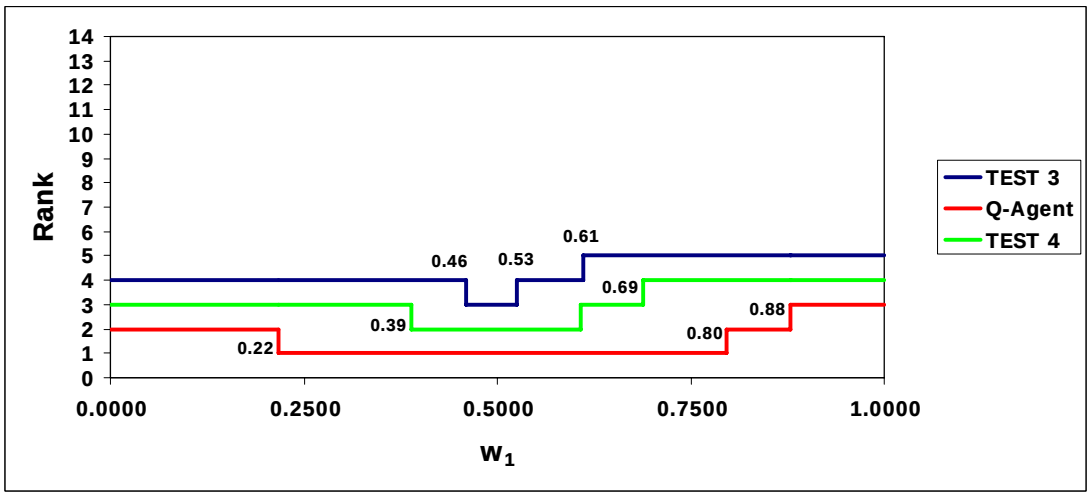

Figure M.10 Ranks of the first three methods for 90% utilization level k=3

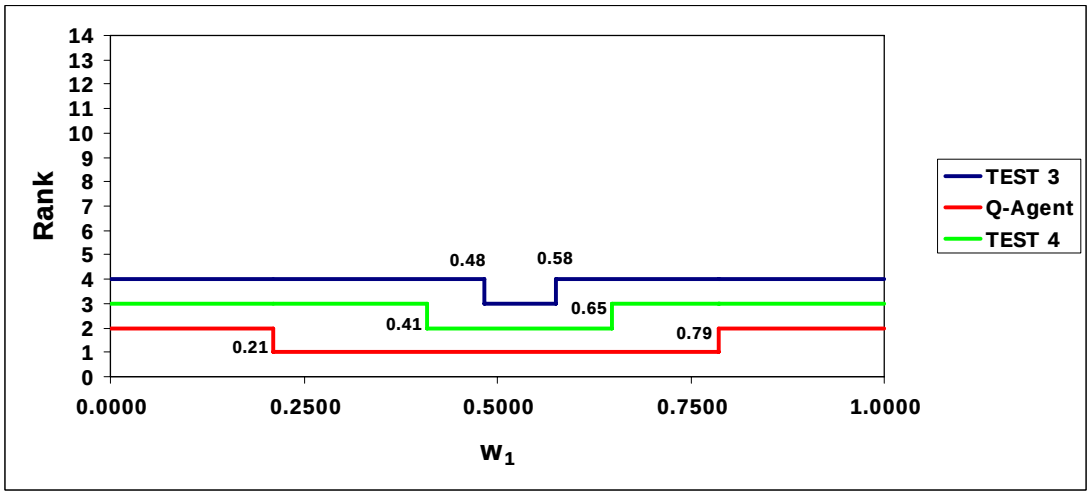

Figure M.11 Ranks of the first three methods for 95% utilization level k=3

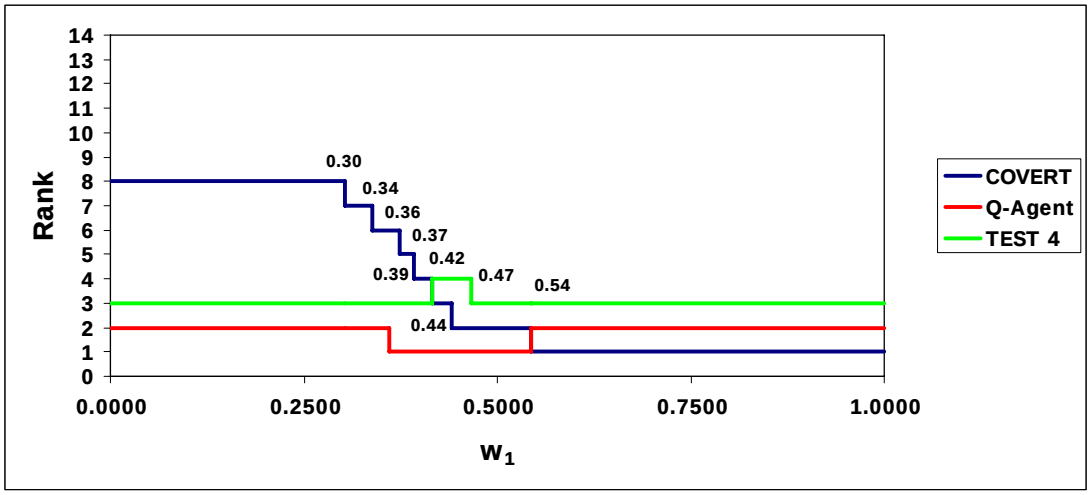

Figure M.12 Ranks of the first three methods for 85% utilization level k=4

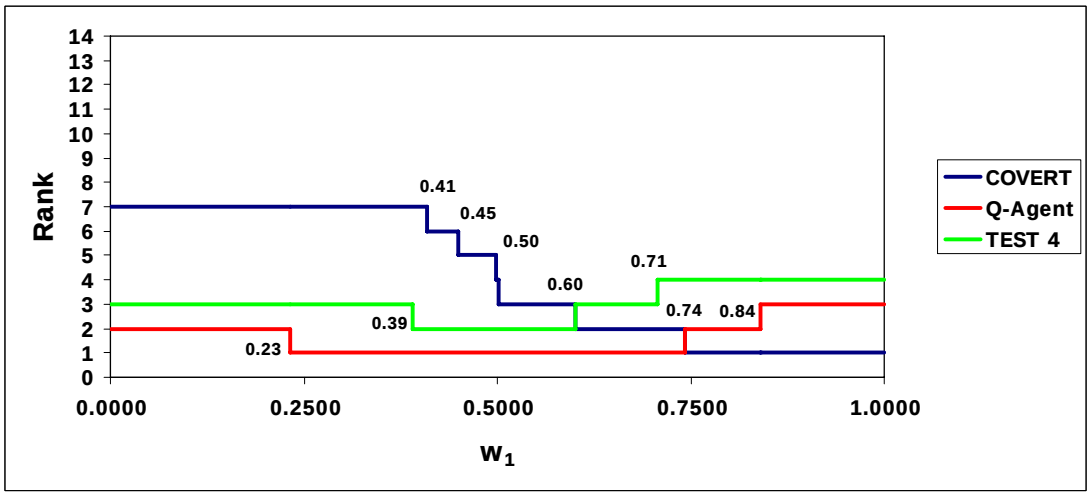

Figure M.13 Ranks of the first three methods for 90% utilization level k=4

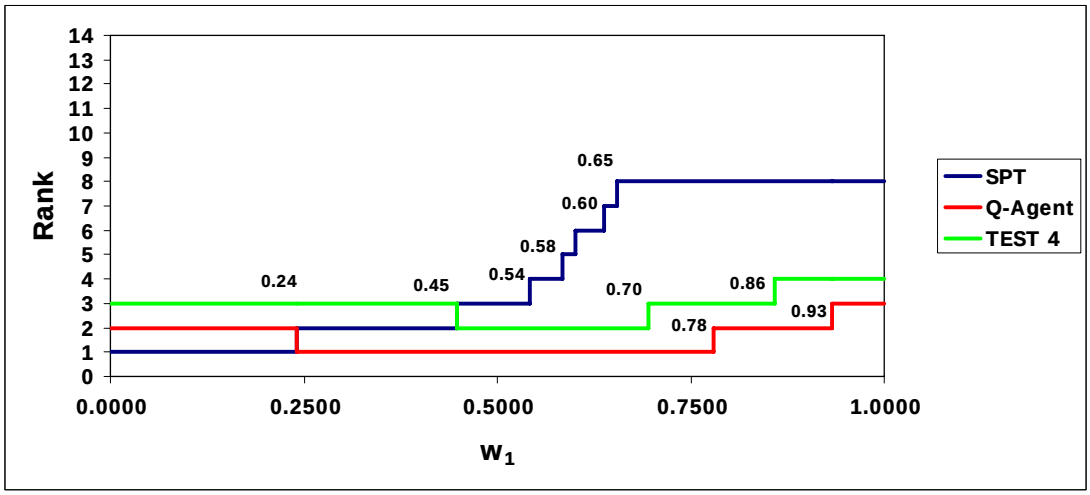

Figure M.14 Ranks of the first three methods for 95% utilization level k=4

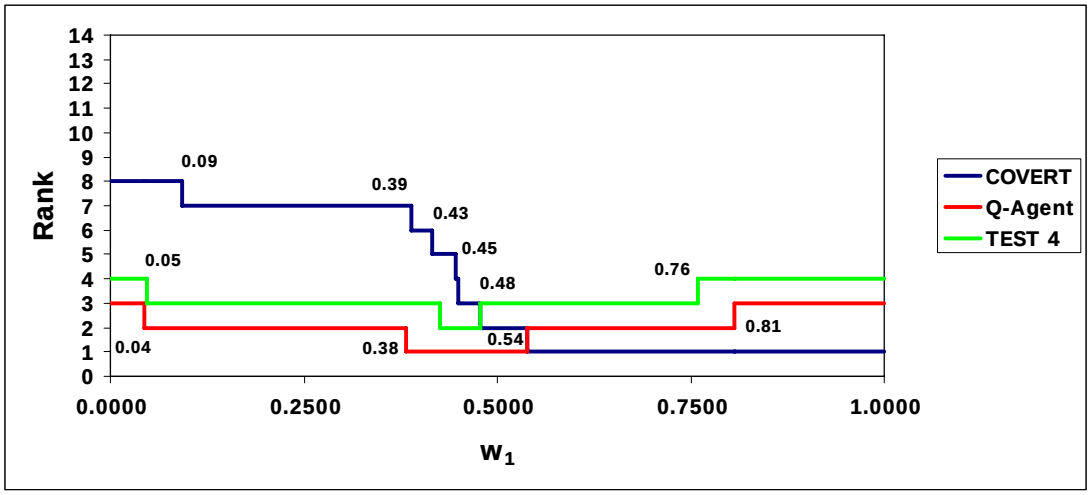

Figure M.15 Ranks of the first three methods for 85% utilization level *k*=5

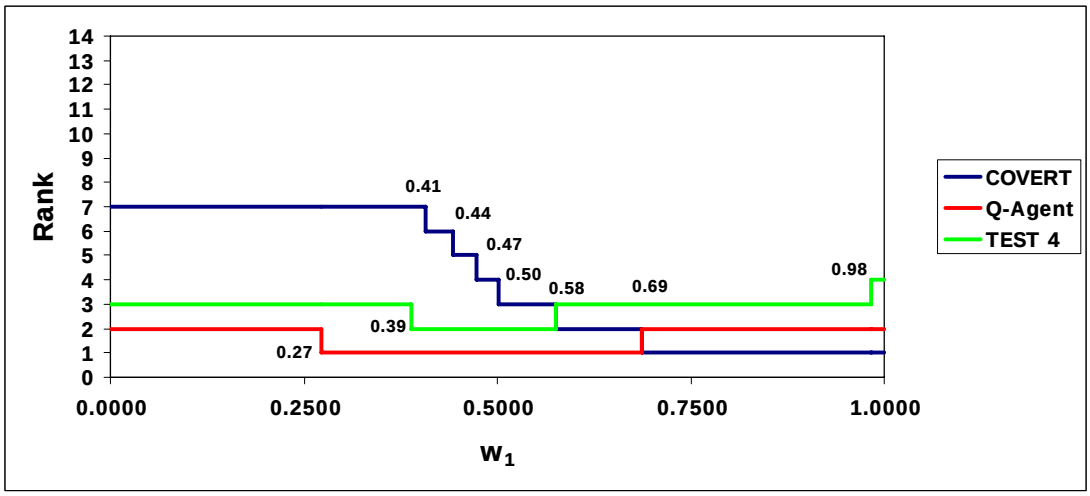

Figure M.16 Ranks of the first three methods for 90% utilization level *k*=5

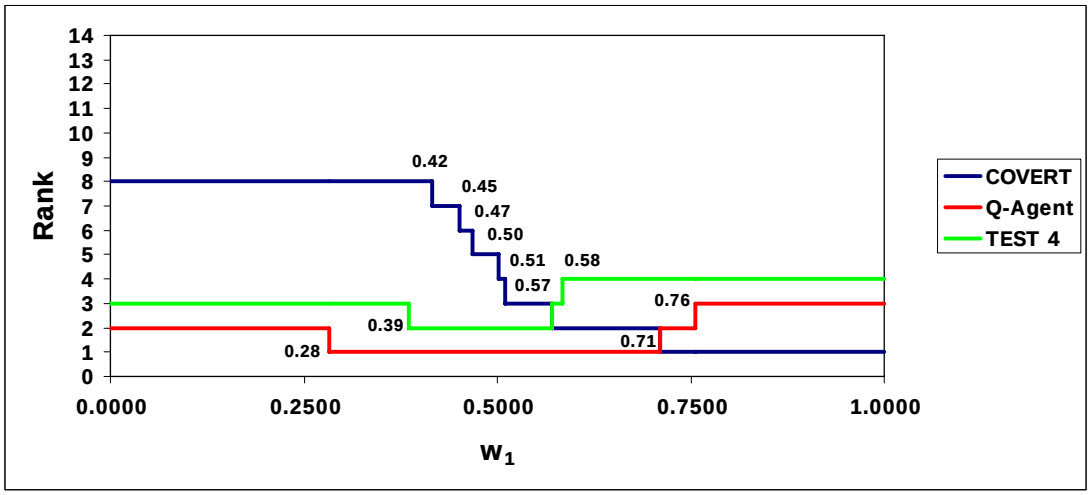

Figure M.17 Ranks of the first three methods for 95% utilization level *k*=5

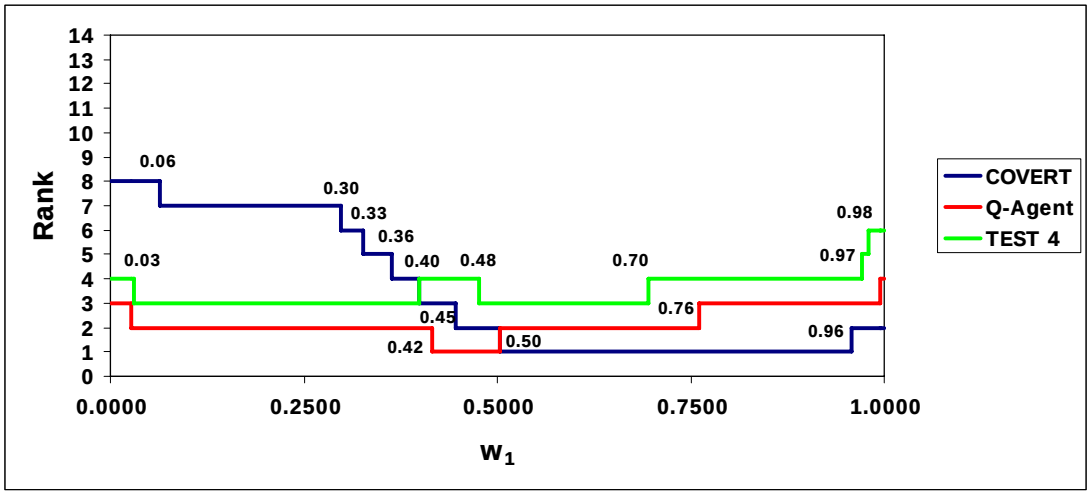

Figure M.18 Ranks of the first three methods for 85% utilization level k=6

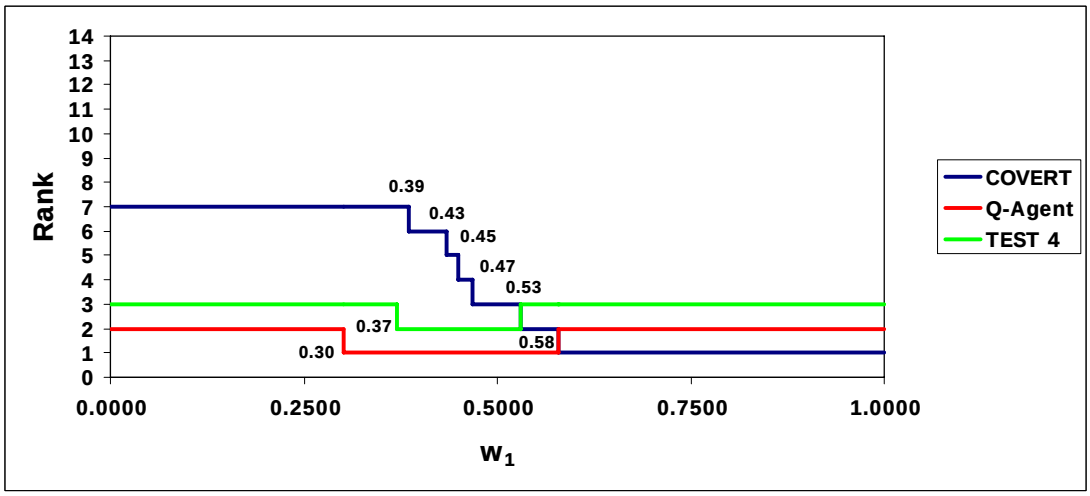

Figure M.19 Ranks of the first three methods for 90% utilization level k=6

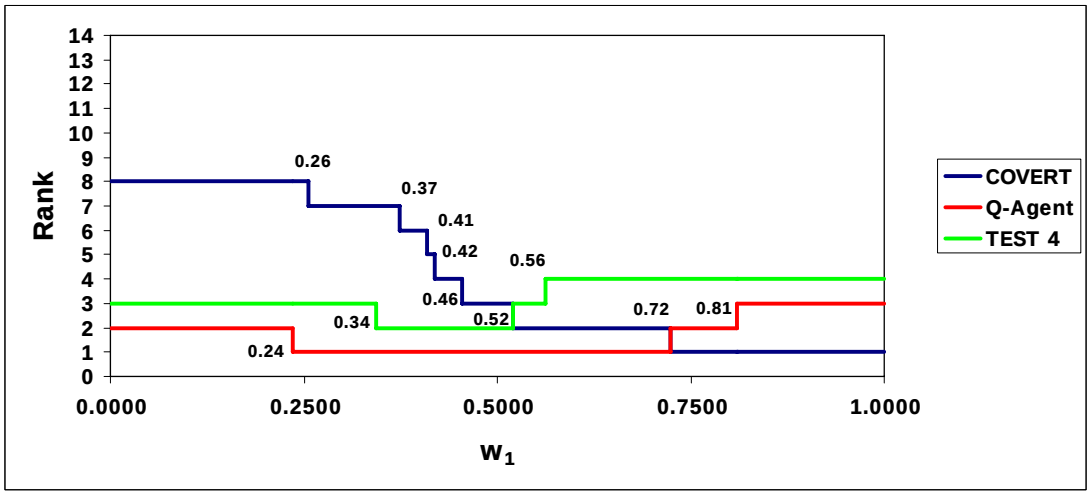

Figure M.20 Ranks of the first three methods for 95% utilization level k=6

APPENDIX M.3: Ranks of the First Three Methods for Multi-State Flow Shop

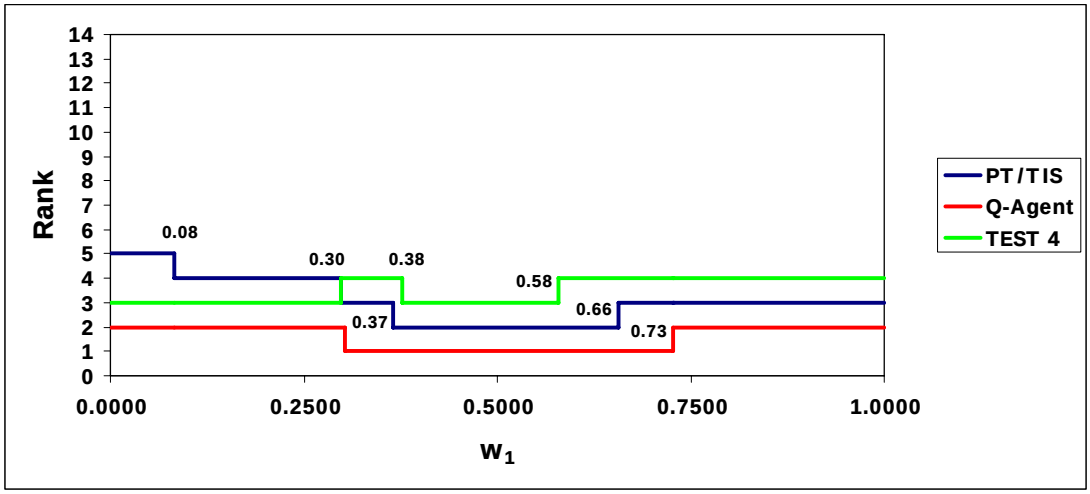

Figure M.21 Ranks of the first three methods for 85% utilization level k=3

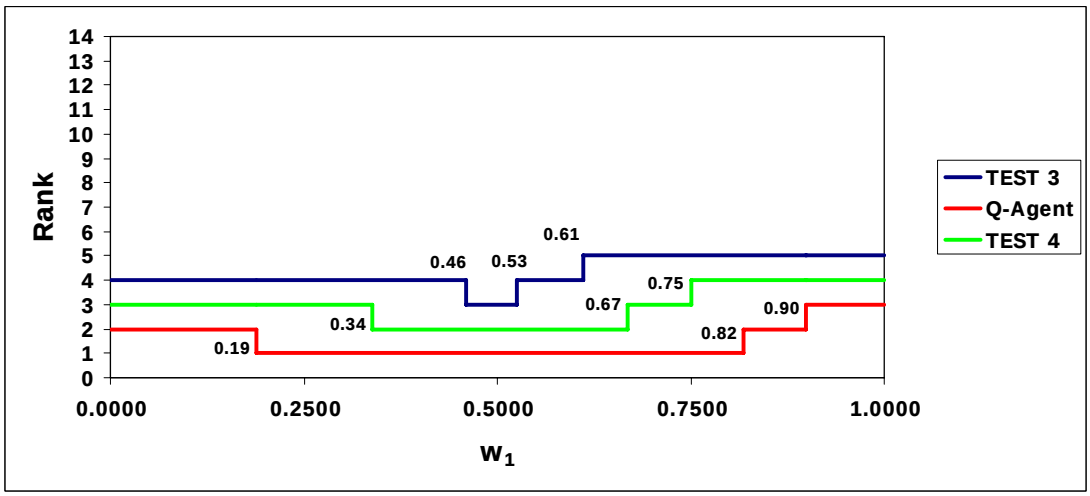

Figure M.22 Ranks of the first three methods for 90% utilization level k=3

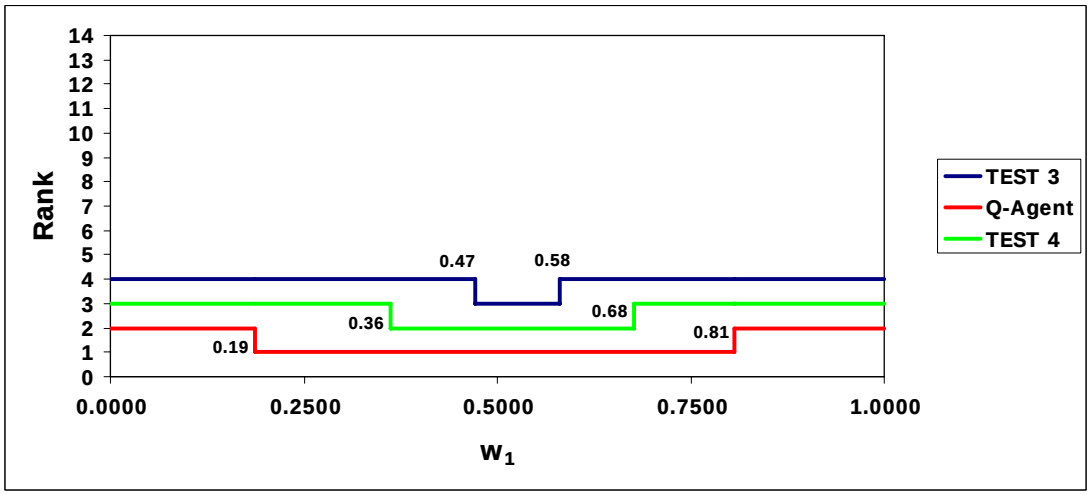

Figure M.23 Ranks of the first three methods for 95% utilization level k=3

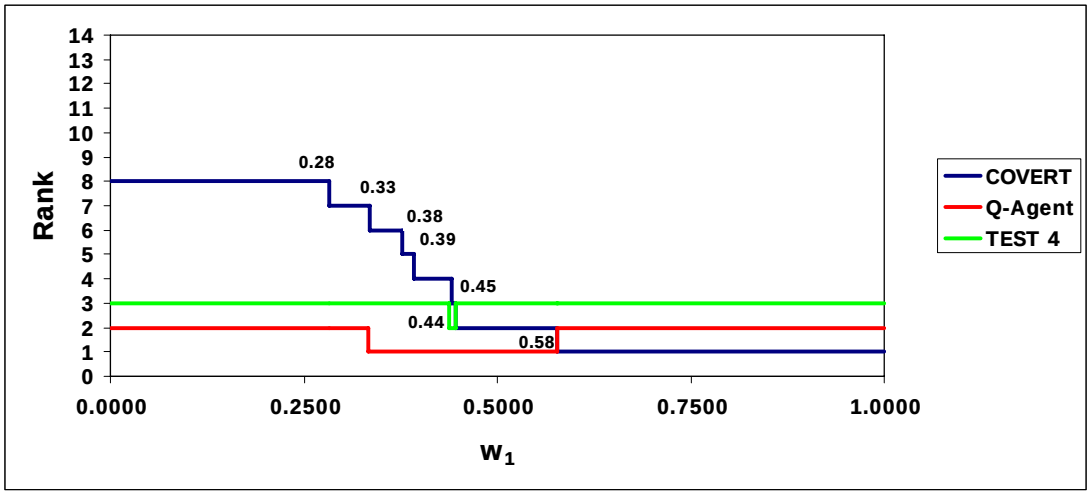

Figure M.24 Ranks of the first three methods for 85% utilization level *k*=4

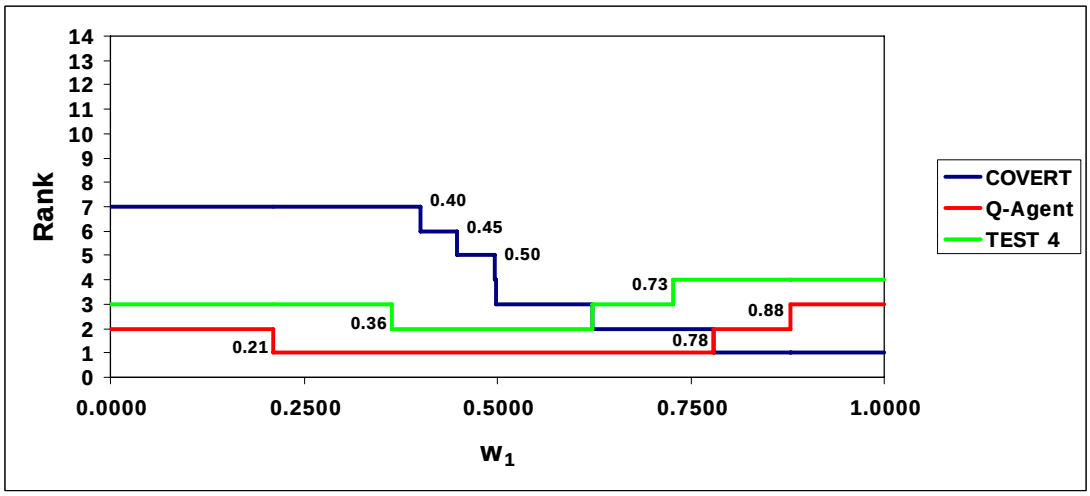

Figure M.25 Ranks of the first three methods for 90% utilization level *k*=4

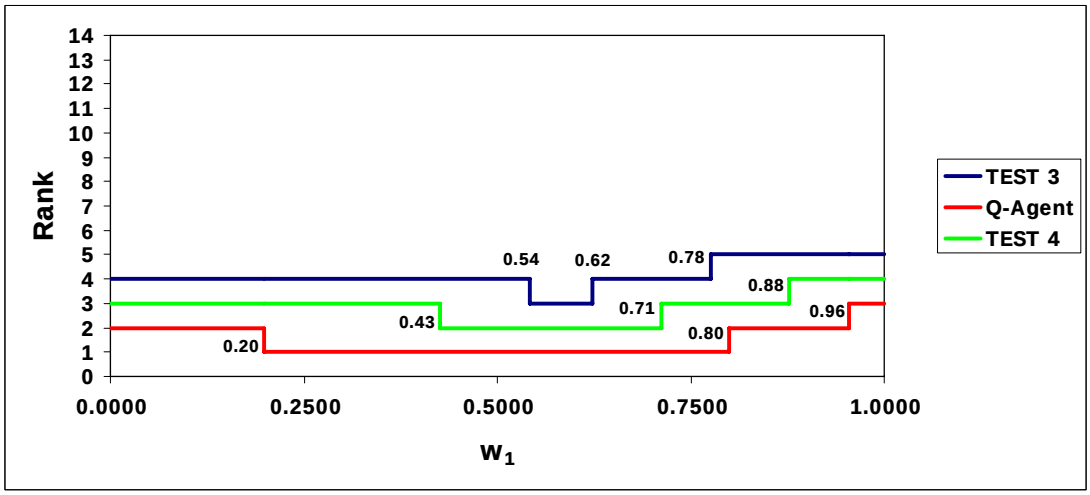

Figure M.26 Ranks of the first three methods for 95% utilization level *k*=4

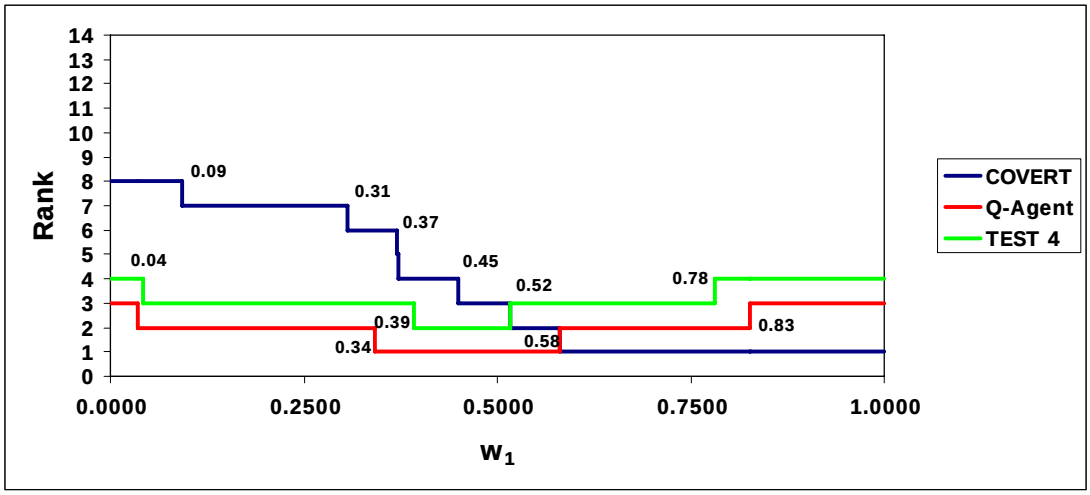

Figure M.27 Ranks of the first three methods for 85% utilization level *k*=5

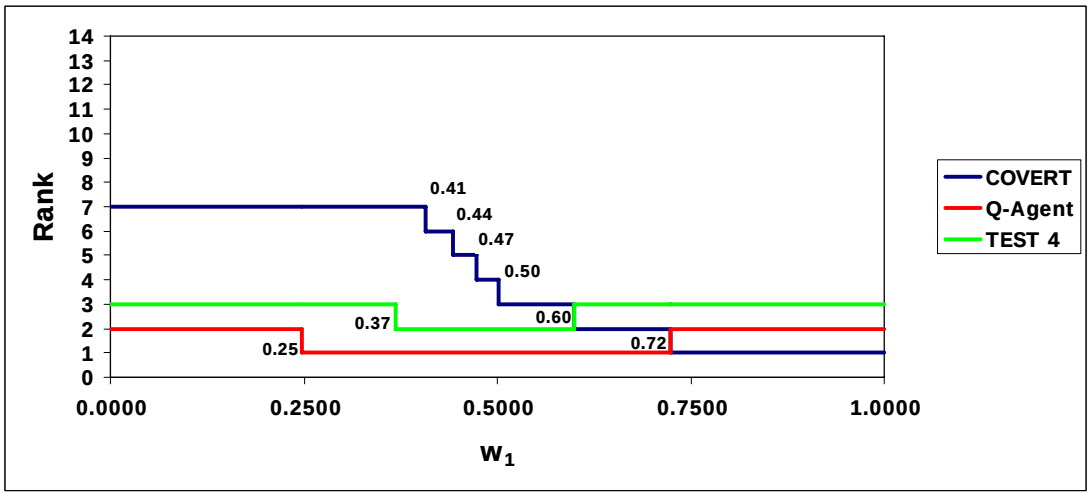

Figure M.28 Ranks of the first three methods for 90% utilization level *k*=5

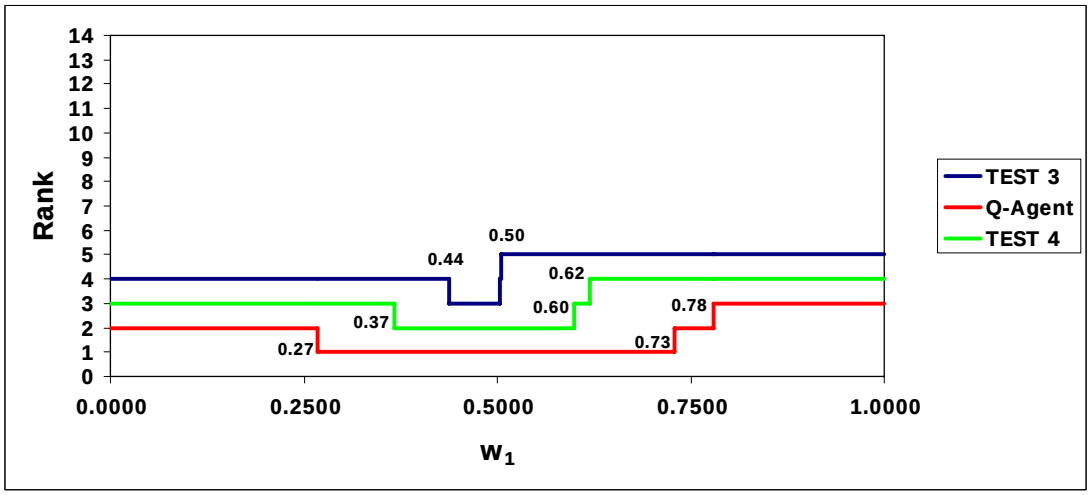

Figure M.29 Ranks of the first three methods for 95% utilization level *k*=5

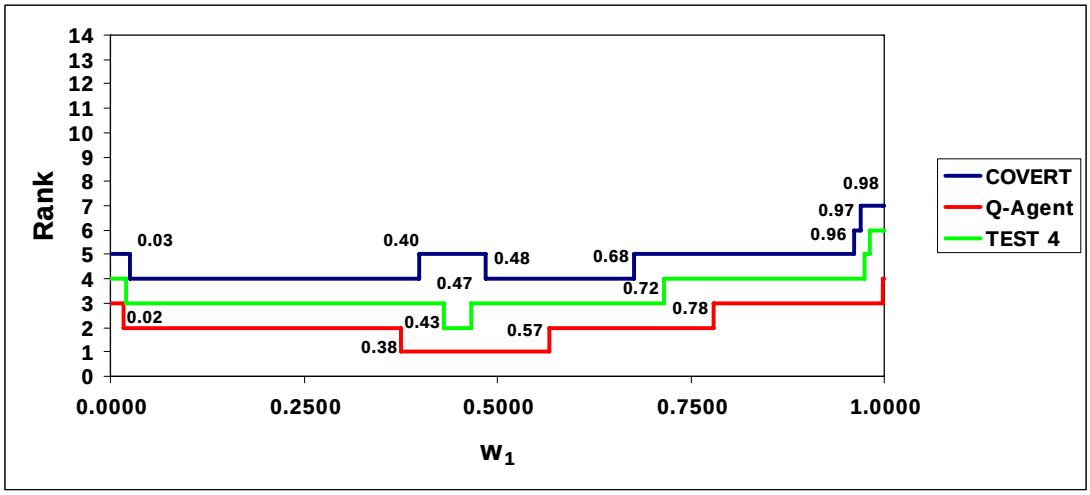

Figure M.30 Ranks of the first three methods for 85% utilization level k=6

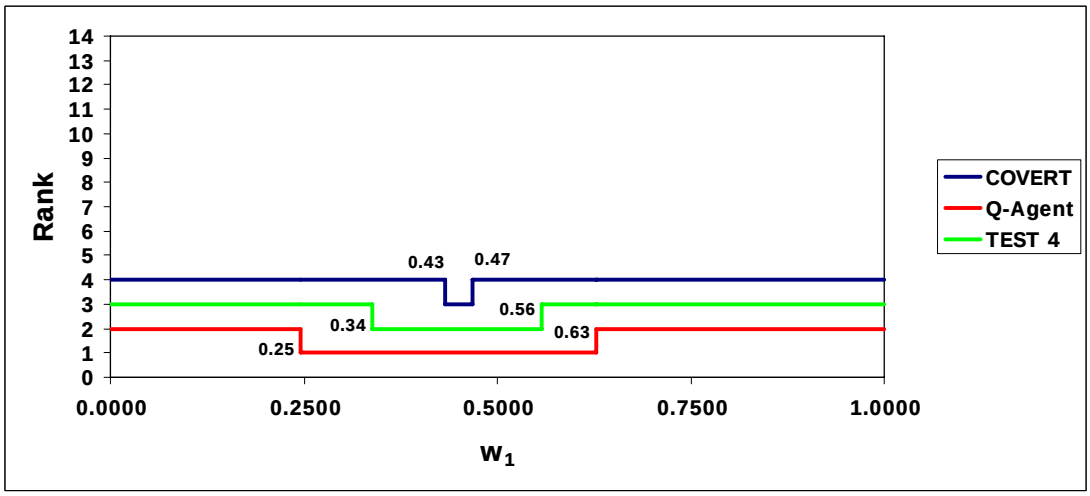

Figure M.31 Ranks of the first three methods for 90% utilization level k=6

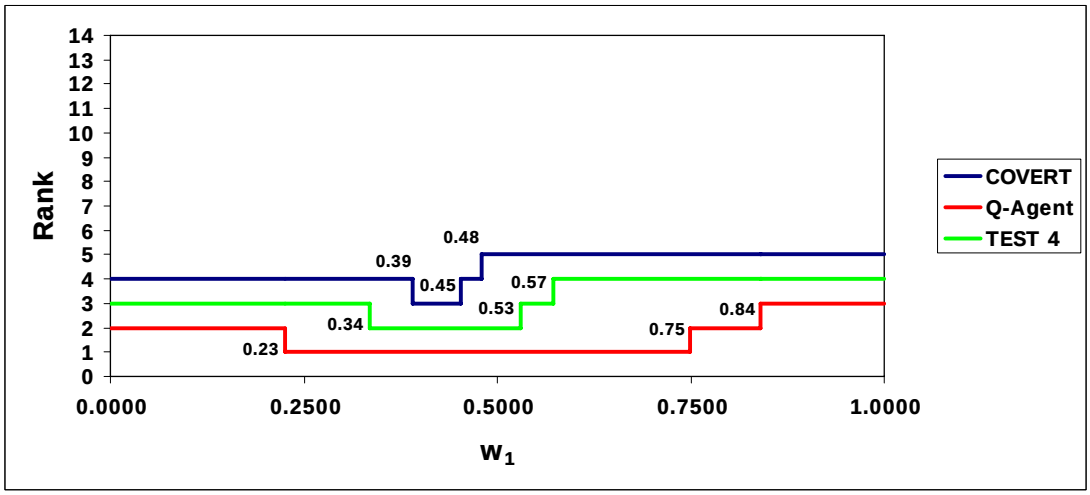

Figure M.32 Ranks of the first three methods for 95% utilization level k=6