YANGON INSTITUTE OF ECONOMICS

Ph.D. PROGRAMME

A Study on Queueing Networking System with Application of Open Queueing Circuit to Accommodating and Using Information Technology in Myanmar

LAY KYI

DECEMBER, 2006

A Study on Queueing Networking System with Application of Open Queueing Circuit to Accommodating and Using Information Technology in Myanmar

Partial fulfillment of requirement for the Degree of

Ph.D.

of the Department of Statistics

Yangon Institute of Economics

Submitted by

Lay Kyi

December 2006

A Study on Queueing Networking System with Application of Open Queueing Circuit to Accommodating and Using Information Technology in Myanmar

The viva voce examination of the Ph.D. candidate Maung Lay Kyi was successfully held on August 31, 2005 from 14:00 hours to 16:30 hours at the room number (15) of Yangon Institute of Economics (kamayut Campus) in front of the Rector of Yangon Institute of Economics Professor Dr. Kan Zaw and examination board members.

CERTIFICATION

I hereby certify that the content of this thesis is wholly my own work unless otherwise referenced or acknowledged. Information from sources is referenced with original comments and ideas from the writer him/herself.

Lay Kyi

December 2006

A Study on Queueing Networking System with Application of Open Queueing Circuit to Accommodating and Using Information Technology in Myanmar

Lay Kyi

December 2006

Board of Examiners

Professor Dr. Kan Zaw
(Chairman)
Rector, Yangon Institute of Economics

Professor Akihiro TAMAKI
(External Examiner)
Tokyo University of Information Sciences
Japan

Professor U Ngwe Soe
(Supervisor)
Professor (Retired)
Department of Statistics

Board of Examiners

External Examiners:

Professor Dr. Khin Maung Nyunt
Rector (Retired)
Yangon Institute of Economics

Professor Dr. Pyke Tin
Rector (Retired)
University of Computer Science, Yangon

Professor Dr. Kyaw Thein
Rector (Retired)
University of Computer Science, Yangon

Professor Dr. Daw Than Toe
Professor (Retired)
Department of Statistics

Internal Examiners:

Professor Dr. Daw Khin May Hla
Pro-Rector
Yangon Institute of Economics

Professor U Kyaw Min Tun
Pro-Rector
Yangon Institute of Economics

Dr. Nu Nu Yin
Professor/ Head
Department of Management Studies

CONTENTS

A Study on Queueing Networking System with Application of Open Queueing Circuit to Accommodating and Using Information Technology in Myanmar

Abstract

This research intends to analyse the computer system performance. Its goal is to apply queueing network modelling in computer network, as tools to assist in answering the questions of cost and performance that arise throughout the life of a computer system. Queueing network modelling in computer network is a methodology for the analysis of computer systems.

Our approach to queueing network modelling arises from our collective experience in contributing to the theory of queueing network modelling, in embodying this theory in performance analysis. Although queueing network models are not a panacea, they are the appropriate tools in many computer system design and analysis applications. The most important characteristic of a computer system analyst is a detailed understanding of computer systems. On the one hand, mathematical sophistication is not required to analyze computer systems intelligently and successfully through queueing network models because the algorithms for evaluating queueing network models are well developed. Queueing network modelling software packages do not guarantee success in computer system analysis. Defining and parameterizing a queueing network model of a particular computer system is a blend of art and science.

The study is divided into five Chapters. In Chapter I we provide five types of background material: a general discussion of basic concept of queueing network modelling, an overview of the way in which a modelling study is conducted, an introduction to the interesting performance quantities in computer systems and relationships that must hold among them, and a discussion of the inputs and outputs of queueing network modeling.

In Chapter II are presented the techniques used to evaluate queueing network models - to obtain outputs such as utilizations, residence times, queue lengths, and throughputs from inputs such as workload intensities and service demands. Chapter III deals with study of the parameterization of queueing network models of basic systems, developing systems, and future systems. In Chapter IV the Ministry of Education (MOE) Intranet system, including the analysis of computer communication networks and database concurrency control mechanisms are surveyed. Additionally the structure and

use of queueing network modelling software packages are examined. In Chapter V, the conclusions are drowned based on different queueing network parameter values computed from system measurement data, and programs for implementing the queueing network evaluation techniques described in Chapter II. They are included to illustrate various aspects of computer system analysis using queueing network models. The results obtained from analysis of MOE intranet system should not be misconstrued as making general statements on the relative performance of various systems; these results have significance only for the specific configurations and workloads under consideration.

The intention is to provide enough information for understanding fully the essential aspects of each technique. Theoretical details of significance to the implementation of a technique, when we feel that these details might obscure the more fundamental concepts, are omitted.

To the best of knowledge and experience, practicing computer system analysts are relatively skilled in techniques such as workload characterization, system measurement, interpretation of performance data, and system tuning, and are at least familiar with basic statistical methods and with simulation. Most interesting and important future research area in queueing network modelling also is briefly put forward; possible approach to each problem felt to be best suited for application is also presented.

Acknowledgements

First and foremost, I would like to thank Professor Dr. Kan Zaw, Rector of the Yangon Institute of Economics, for giving me the opportunity to undertake this doctoral research. Next, I wish to express my deep gratitude to Professor Dr. Daw Than Toe for her constant encouragement and constructive suggestions to complete this research work successfully. I am also grateful to professor Daw Hta Hta , U Nyan Lynn and U Mya Thein for giving valuable suggestions.

I wish to thank the staff of Yangon Institute of Economics Library for providing me access to documents and for their kind cooperation and assistance.

I am highly and tremendously grateful to my supervisor, Professor U Ngwe Soe for reading and supervising the entire work and providing many valuable suggestions for and constant criticism improvement.

The active support from my friends and colleges, too numerous to mention by name, for their advice and encouragement is gratefully acknowledged.

Last but not least, I wish to express my gratitude and appreciation to my wife, Daw Mya Mya Shein, for her full support and encouragement from inception of this research work to its end.

List of Figures

List of Tables

Chapter I

Queueing Network Model

This first chapter of the thesis provides five different sorts of background material as a prelude to our study of quantitative system performance. In section 1.1, we survey queueing network modelling, discussing some applications and comparing it to more traditional approaches to computer system analysis. This section has surveyed the questions of cost and performance that arise throughout the life of a computer system, the nature of queueing network models, and the role that queueing network models can play in answering these questions. Queueing network models have achieved a favorable balance between accuracy and cost, are the appropriate tools in many computer system design and analysis applications.

In section 1.2, we explore various aspects of conducting a modelling study. Our objective is to provide some perspective on the "pieces" of the process that will be studied in the research. The most challenging aspect of computer system analysis using queueing network models is not the technical details of defining, parameterizing, and evaluating the models. It is the process of tailoring the general "methodology" of queueing network modelling to a specific computer system analysis context.

In section 1.3, we provide a technical foundation for our work by defining a number of quantities of interest, introducing the notation that will use in referring to these quantities, and deriving various relationships among these quantities.

In section 1.4, we describe the inputs and the outputs of separable queueing network models. The availability of inputs and outputs is determined by assumptions imposed to ensure the efficient evaluation of the model. The practical impacts of assumptions on the accuracy of the model are also considered. Often, separable models are adequate by themselves, because complex system characteristics are captured implicitly in the measurement data used to parameterize them. In other cases, separable models must be augmented with procedures that calculate revised estimates for those portions of various service demands that are indirect representations of relevant system characteristics.

In section 1.5, we discuss some application on computer communication networks. We have studied models of computer communication networks, local area networks, software resources, and database concurrency control. These models have

employed techniques such as flow equivalent service centers (FESCs) with global balance, FESCs whose service rates are determined through probabilistic analysis, two-level hierarchical iteration, and hybrid modelling.

1.1 An Overview of Queueing Network Modelling

1.1.1. Introduction

This research is the computer system performance analyst. Its goal is to apply queueing network models in the computer system, as tools to help in answering the questions of cost and performance that arise throughout the life of a computer system. Our approach arises from our collective experience in the theory of queueing network modelling (Allen 1978), to manifest this theory in performance analysis tools and in computer system analysis tools.

Although queueing network models are not a panacea, they are the appropriate tools in many computer system design and analysis applications. On the one hand, mathematical sophistication is not required to analyze computer systems intelligently and successfully using queueing network models. The algorithms for evaluating queueing network models are well developed. This is the case because defining and parameterizing a queueing network model of a particular computer system is a blend of art and science, requiring training and experience.

1.1.2 Queueing Network

Today's computer systems are more complex, more rapidly developing (Beizer 1978). The result is an increasing need for tools and techniques that assist in understanding the behavior of these systems. A computer manufacturer is considering various architectures and protocols during design and implementation stage for connecting terminals to mainframes using a packet oriented communications network. An energy utility must assess the longevity of its current configuration, given forecasts of workload growth. Knowing what the system bottleneck will be is desirable during evolution of the configuration and workload, and the relative cost effectiveness of various alternatives for alleviating it. In particular, since this is a virtual memory system, tradeoffs among memory sizes, CPU power, and paging device speed must be evaluated.

In considering questions such as these, one must begin with a thorough grasp of the system, the application, and the objectives of the study. With this as a basis, several approaches are available. One is the use of intuition and trend extrapolation. Few substitutes for experience and insight yield reliable intuition. Unfortunately, those who possess these qualities in sufficient quantity are rare.

Another is the experimental evaluation of alternatives. Experimentation is always valuable, often required, and sometimes the approach of choice. It is also expensive. A further drawback is that an experiment is likely to yield accurate knowledge of system behavior under one set of assumptions that would allow generalization.

These two approaches are in some sense at opposite extremes of a spectrum. Intuition is rapid and flexible, but its accuracy is suspect because it relies on experience and insight that are difficult to find and verify. Experimentation yields excellent accuracy, but is laborious and inflexible. Between these extremes' lies a third approach, the general subject of this research:

1.1.3. Queueing Network Model

A model is an abstraction of a system: an attempt to extract, from the mass of details. A model is the system itself, exactly those aspects that are essential to the system's behavior. Once a model has been defined through this abstraction process, it can be parameterized to reflect any of the alternatives under study, and then evaluated to determine its behavior under this alternative. Using a model to investigate system behavior is less laborious and more flexible than experimentation, because the model is an abstraction that avoids unnecessary detail. It is more reliable than intuition, because it is more methodical: each particular approach to modelling provides a framework for the definition, parameterization, and evaluation of models. Of equal importance, using a model enhances both intuition and experimentation. Intuition is enhanced because a model makes it possible to investigate the behavior of a system under a wide range of alternatives. Experimentation is enhanced because the framework provided by each particular approach to modelling gives guidance about which experiments are necessary to define and parameterize the model. Modelling, then, provides a framework for gathering, organizing, evaluating, and understanding information about a computer system.

Queueing network modelling is a particular approach to computer system modelling in which the computer system is represented as a network of queue which is evaluated analytically. A network of queues is a collection of service centers, which represent system resources, and customers, which represent users or transactions. Analytic evaluation involves using software to solve efficiently a set of equations induced by the network of queues and its parameters.

1.1.3.1. Single Service Centers

Figure 1.1.1 illustrates a single service center. Customers arrive at the service center, wait in the queue if necessary, receive service from the server, and depart. In fact, this service center and its arriving customers form a queueing network model.

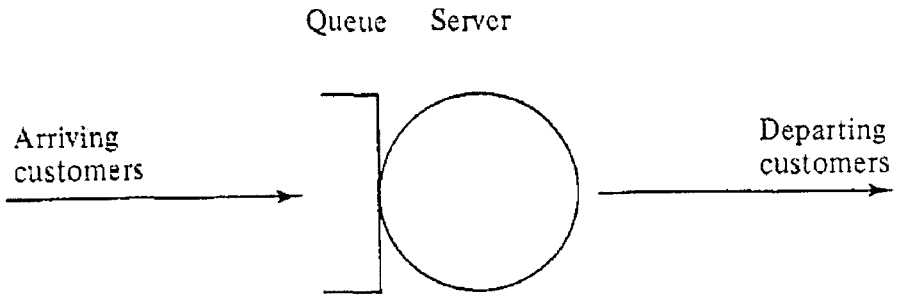


Figure 1.1.1 - A Single Service Center (Ferrari 1978)

This model has two parameters. First is the workload intensity, which is the rate at which customers arrive. Second is the service demand, which is the average service requirement of a customer. For specific parameter values, it is possible to evaluate this model by solving some simple equations, yielding performance measures such as utilization, residence time, queue length, and throughput.

The principal thing to observe is that the evaluation of the model yields performance measures that are qualitatively consistent with intuition and experience. Consider residence time (Ferrari 1978). When the workload intensity is low, an arriving customer will seldom encounter competition for the service center, so will enter service immediately and will have a residence time roughly equal to its service requirement. As the workload intensity rises, congestion increases, and residence time along with it. Initially, this increase is gradual. As the Utilization: loads grow, however, residence time increases at a faster and faster rate, until, as the service center approaches saturation, small increases in arrival rate results in dramatic increases in residence time.

1.1.3.2. Multiple Service Centers

Imagining it characterizing a contemporary computer system by two parameters is difficult, as would be required to use the model of Figure 1.1.1. Figure 1.1.2 shows a more realistic model in which each system resource is represented by a separate service center.

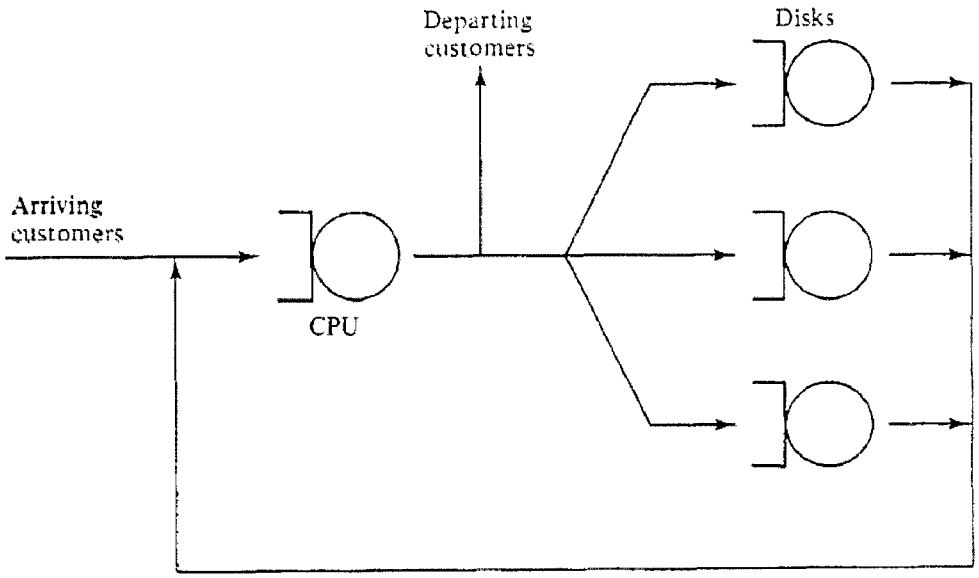


Figure 1.1.2 - A Network of Queues (Ferrari 1978)

The parameters of this model are analogous to those of the previous one. The workload intensity is the rate at which customers arrive. Service demands provide a separate service demand for each service center. If the customers corresponding to transactions in the system, the workload intensity corresponds to the rate at which users submit transactions to the system. Service demand at each service center corresponds to the total service requirement per transaction at the corresponding resource in the system. As indicated by the lines in the figure, it can think of customers as arriving, circulating among the service centers, and then departing. The pattern of circulation among the centers is not important, however; only the total service demand at each center.

1.1.4. Parameterizing of Queueing Network Models

Defining a queueing network model of a particular system is made relatively straight forward by the close correspondence between the attributes of queueing network models and the attributes of computer systems (Kleinrock 1976). For example, service centers in queueing network models naturally correspond to hardware resources and their software queues in computer systems, and customers in queueing network models naturally correspond to users or transactions in computer systems.

Queueing network models have a richer set of attributes. As an example, specifying the rate at which customers arrive is only one of three available means to describe workload intensity. A second approach is to state the number of customers in the model. A third approach is to specify the number of customers and the average time that

each customer "spends thinking" between interactions. In Figure 1.1.3 modified the model of Figure 1.1.2 so that the workload intensity is described using this last approach.

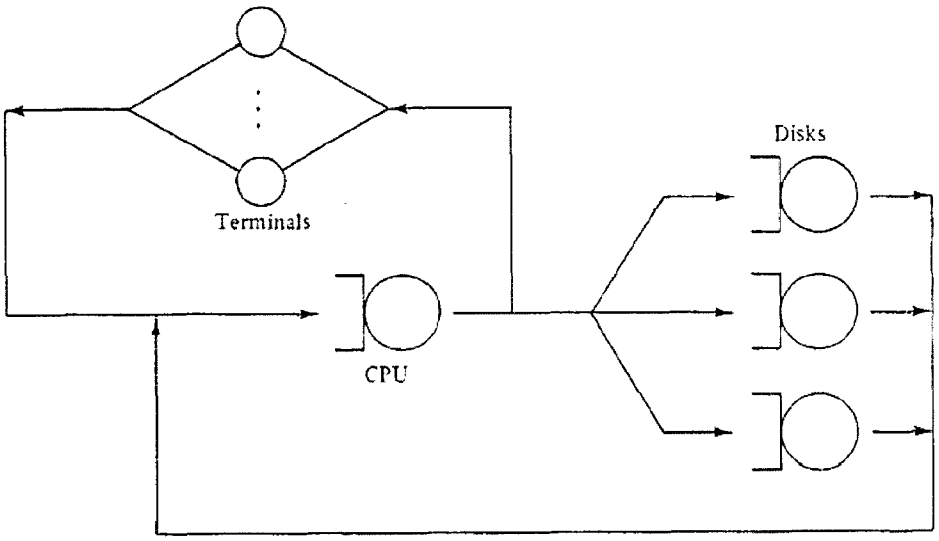


Figure 1.1.3 - A Model with a Terminal-Driven Workload (Ferrari 1978)

Most computer systems have several identifiable workload components. Although the queueing network models that have considered a single customer class, distinguishing it between a system's workload components in a queueing network model by making use of multiple customer classes is possible, and service demands. Modeling directly a computer system in which four workload components are possible: transaction processing, background batch, interactive database inquiry, and interactive program development. Defining the model would specify four customer classes and the relevant service centers. In parameterizing the model, it would provide workload intensities for each class. It would provide service demands for each class at each service center. Evaluating the model would obtain performance measures in the aggregate (e.g., total CPU utilization), and on a per-class basis.

1.1.4.1. Parameterization

The parameterization of queueing network models, like their definition, is straightforward. Imagine calculating the CPU service demand for a customer in a queueing network model of an existing system. It would observe the system in operation and would measure two quantities: the number of seconds that the CPU was busy, and the number of user requests processed. The busy time would be divided by the number of request completions to find out the average number of seconds of CPU service attributable to each request, the required parameter.

A major strength of queueing network models is the relative ease with which parameters can be modified to obtain answers to "what if " questions.

- What if we balance the I/O activity among the disks?
- What if the workload subsequently increases by 20%?
- What if we then upgrade to a CPU 30% faster?

Considerable insight can be required to conduct such a modification analysis. The performance measures obtained from evaluating the model can be only as accurate as the workload intensities and service demands provided as inputs. Anticipating every effect on these parameters of a change to the configuration or workload is not always easy. Consider the first "what if" question listed above. If it assume that the system's disks are physically identical then the primary effect of balancing disk activity can be reflected the parameter values of the model by setting the service demand at each disk to the average value. However, there may be secondary effects of the change. If this secondary effect is anticipated, then reflecting it in the parameter values of the model is easy, and the model, when evaluated, will yield accurate values for performance measures. If not, the model will yield pessimistic results. Fortunately, the primary effects of modifications, which dominate performance, tend to be relatively easy to anticipate.

Models with multiple customer classes are more common than models with single customer classes because they simplify answering many "what if" questions. Single class models, though, have the advantage that they are especially easy to parameterize, requiring few assumptions by the analyst. Using contemporary computer system measurement tools, figuring out correctly resource consumption by workload component is notoriously difficult, especially concerning system overhead and I/O subsystem activity (Ferrari et al. 1983). Since single class models can be parameterized with greater ease and accuracy, they are quicker and more reliable than multiple class models for answering those questions to which they are suited.

1.1.4.2. Evaluation

We distinguish two approaches to evaluating queueing network models. The first involves calculating bounds on performance measures, rather than specific values. For example, we might find out upper and lower bounds on response time for a particular set of parameter values (workload intensity and service demands). The virtue of this approach is that the calculations are simple enough to be carried out by hand, and the resulting bounds can contribute significantly to understanding the system under study.

The second approach involves calculating the values of the performance measures. While the algorithms for doing this are sufficiently complicated that the use of computer programs is necessary, emphasizing that these algorithms are extremely efficient is important. Specifically, the running time of the most efficient general algorithm grows according to the product of the number of service centers with the number of customer classes. The running time is largely independent of the number of customers in each class. A queueing network model with 100 service centers and 10 customer classes can be evaluated in only seconds of CPU time.

The algorithm for evaluating queueing network models is the lowest level of a queueing network modelling software package. Higher levels typically include transformation routines to map the characteristics of specific subsystems onto the general algorithms at the lowest level, and high-level front ends that assist in obtaining model parameter values from system measurement data.

1.1.5. Appropriate Queueing Network Models

Models in general, and queueing network models in particular, have become important tools in the design and analysis of computer systems (Gelenbe & Mitrani 1980). This is because, for many applications, queueing network models achieve a favorable balance between accuracy and efficiency.

In terms of accuracy, a large body of experience shows that queueing network models can be expected to be accurate to within 5 to 10% for utilizations and throughputs and to within 10 to 30% for response times. This level of accuracy is consistent with the requirements of a variety of design and analysis applications. Important, it is consistent with the accuracy achievable in other components of the computer system analysis process, such as workload characterization.

About efficiency, it has been shown in the previous section that queueing network models can be defined, parameterized, and evaluated at low cost. Definition is eased by the close correspondence between the attributes of queueing network models and the attributes of computer systems. Parameterization is eased by the small number of high-level parameters. Evaluation is eased by the recent development of algorithms whose running time grows as the product of the number of service centers with the number of customer classes (Kobayashi 1978).

Queueing network models achieve high accuracy at low cost. The incremental cost of achieving greater accuracy is high - significantly higher than the incremental benefit, for a wide variety of applications.

1.1.6. Related Techniques

Our informal description of queueing network modelling has taken several liberties that should be acknowledged to avoid confusion. These liberties can be summarized as follows:

- We have not described networks of queues in their full generality, but a subset that can be evaluated efficiently.
- We have incorrectly implied that the only analytic technique for evaluating networks of queues is the use of software to solve a set of equations induced by the network of queues and its parameters.
- We have neglected the fact that simulation can be used to evaluate networks of queues.
- We have not explored the relationship of queueing network models to queueing theory.

The following subsections explore these issues.

1.1.6.1. General Networks of Queues

This research is concerned with a subset of general networks of queues. This subset consists of the separable queueing networks (a name used for historical and mathematical reasons), extended where necessary for the accurate representation of particular computer system characteristics.

We restrict our attention to the members of this subset because of the efficiency with which they can be evaluated. This efficiency is mandatory in analyzing contemporary computer systems, which may have hundreds of resources and dozens of workload components, each consisting of many users or jobs.

Restriction to this subset implies certain assumptions about the computer system under study. On the one hand, these assumptions are seldom satisfied strictly. On the other hand, the inaccuracies resulting from violations of these assumptions typically are, at worst, comparable to those arising from other sources (e.g., inadequate measurement data).

General networks of queues, which avoid many of these assumptions, can be evaluated analytically, but the algorithms require time and space that grow prohibitively quickly with the size of the network. They are useful in certain specialized circumstances, but not for the direct analysis of realistic computer systems.

1.1.6.2. Simulation

The principal strength of simulation is its flexibility. There are few restrictions on the behavior that can be simulated, so a computer system can be represented at an arbitrary level of detail. At the abstract end of this spectrum is the use of simulation to evaluate networks of queues. At the concrete extreme, running a benchmark experiment is in some sense using the system as a detailed simulation model of itself.

The principal weakness of simulation modelling is its relative expense. Simulation models generally are expensive to define, because this involves writing and debugging a complex computer program. (In the specific domain of computer system modelling, however, this process has been automated by packages that generate the simulation program from a model description.) They can be expensive to parameterize, because a highly detailed model implies many parameters. Finally, they are expensive to evaluate, because running a simulation requires substantial computational resources, especially if narrow confidence intervals are needed.

A tenet of this research, for which there is much supporting evidence, is that queueing network models provide an appropriate level of accuracy for a wide variety of computer system design and analysis applications. Therefore, our primary interest in simulation is as a means to evaluate certain sub-models in a study that is primarily analytic. This technique, known as hybrid modeling, is motivated by a desire to use analysis where possible, since the cost of evaluating a simple network of queues using simulation exceeds of magnitude the cost of evaluating the same model using analysis.

1.1.6.3. Queueing Network Models and Queueing Theory

Queueing network modelling can be viewed as a small subset of the techniques of queueing theory, selected and specialized for modeling computer systems (Lavenberg 1983). Much of the queueing theory is oriented towards modelling a complex system using a single service center with complex characteristics. Sophisticated mathematical techniques are employed to analyze these models.

Rather than single service centers with complex characteristics, queueing network modelling employs networks of service centers with simple characteristics, Benefits arise from the fact that the application domain is restricted to computer systems. An appropriate subset of networks of queues can be selected, and evaluation algorithms can be designed to obtain meaningful performance measures with an appropriate balance between accuracy and efficiency. These algorithms can be packaged with interfaces based on the terminology of computer systems so that only a minimal understanding of the theory underlying these algorithms is required to apply them successfully.

1.2 Study on Modelling

1.2.1. Introduction

We take a broad look at how, when confronted with a specific computer system analysis problem, to apply the general "methodology" of queueing network modelling. The success of queueing network modelling is because the low-level details of a system are largely irrelevant to its high-level performance characteristics (Sauer & Chandy 1981). Queueing network models appear abstract when compared with other approaches to computer system analysis. Queueing network modelling is inherently a top-down process. The underlying philosophy is to begin by identifying the principal components of the system and the ways in which they interact, then supply any details that are necessary.

There is a strong incentive to identify and eliminate irrelevant details. In fact, it will adopt a liberal definition of "irrelevant" in this context by generally including any system characteristic that will not have a primary effect on the results of the study. - A system may have many identifiable workload components. Then, choose to employ a model with only two classes, one representing the workload component of interest and the other representing the aggregate effect of all other workload components (Kienzle & Sevcik 1979). - The primary effect of a CPU upgrade will be a decrease in CPU service demands. A change in the average paging and swapping activity per job may also result, but if so, this is a secondary effect.

The measurement tools available on contemporary computer systems often fail to provide directly the quantities required to parameterize queueing network models. Queueing network models require a small number of carefully selected inputs. Measurement tools, largely for historical reasons, provide a large volume of data, most of which is of limited use for our purposes. Typically, most CPU activity is not attributed to

specific workload components. Since the CPU tends to be a heavily utilized resource, correct attribution of its usage is important to the accuracy of a multiple class model. Determining the multiprogramming level of a batch workload is sometimes difficult, because some system tasks may be counted by the measurement tool.

Restriction is needed for ease of evaluation of general networks of queues that can be evaluated efficiently. Extremely high variability in the service requirement at a particular resource can cause performance to degrade. Direct representation of this characteristic makes queueing network models costly to evaluate, though, and examples where it is a major determinant of performance are rare. It generally is omitted from models. Memory admission policies typically are complex, and the memory requirements of programs differ. The evaluation of a model is considerably eased, if it is assumed that the memory admission policy is either first-come-first-served or class-based priority, and that programs have similar memory requirements, at least within each class.

Skill in introducing and assessing assumptions is the key to conducting a successful modelling study. Overall, being explicit concerning the assumptions made is important, the motivations for their introduction, and the arguments for their plausibility. This allows the analyst's reasoning to be examined, and simplifies evaluating the sensitivity of the results to the assumptions.

1.2.2. The Modelling Cycle

The most common application of queueing network modeling involves projecting the effect on performance of changes to the configuration or workload of an existing system. There are three phases to such a study. In the validation phase, a baseline model of the existing system is constructed and its sufficiency established. In the projection phase, this model is used to forecast the effect on performance of the anticipated modifications. In the verification phase, the actual performance of the modified system is compared with the model's projections (Lo 1980). Taken together, these three phases are called the modelling cycle, illustrated in Figure 1.2.1.

The validation phase begins with the definition of the model, which includes selection of those system resources and workload components that will be represented. The identification of any system characteristics may require special attention. System characteristics are choice of model structure, and procedures for obtaining the necessary parameters from the available measurement data. Next, the system is measured to obtain workload measures, from which model inputs will be calculated, and performance

measures, which will be compared to model outputs. Sometimes these are the same; for instance, device utilizations are workload measures and performance measures. On the other hand, the multiprogramming level of a batch workload is strictly a workload measure, and system response time is strictly a performance measure. The workload measures then are used to parameterize the model, a step that may require various transformations. The model is evaluated, yielding outputs. These are compared with the system's performance measures. Discrepancies indicate flaws in the process, such as system characteristics ignored or represented inappropriately, or model inputs whose values were established incorrectly. Unfortunately, the absence of such discrepancies does not guarantee that the model will project properly the effect of system or workload modifications. Confidence in a model's predictive abilities may come from two sources. The first is repetitive validation over most measurement intervals, perhaps involving selected modifications. For example, if the objective of a modelling study is to assess the benefits of additional memory, repeating the validation phase may be possible while various amounts of existing memory are disabled. The second is completion of the verification phase, discussed below.

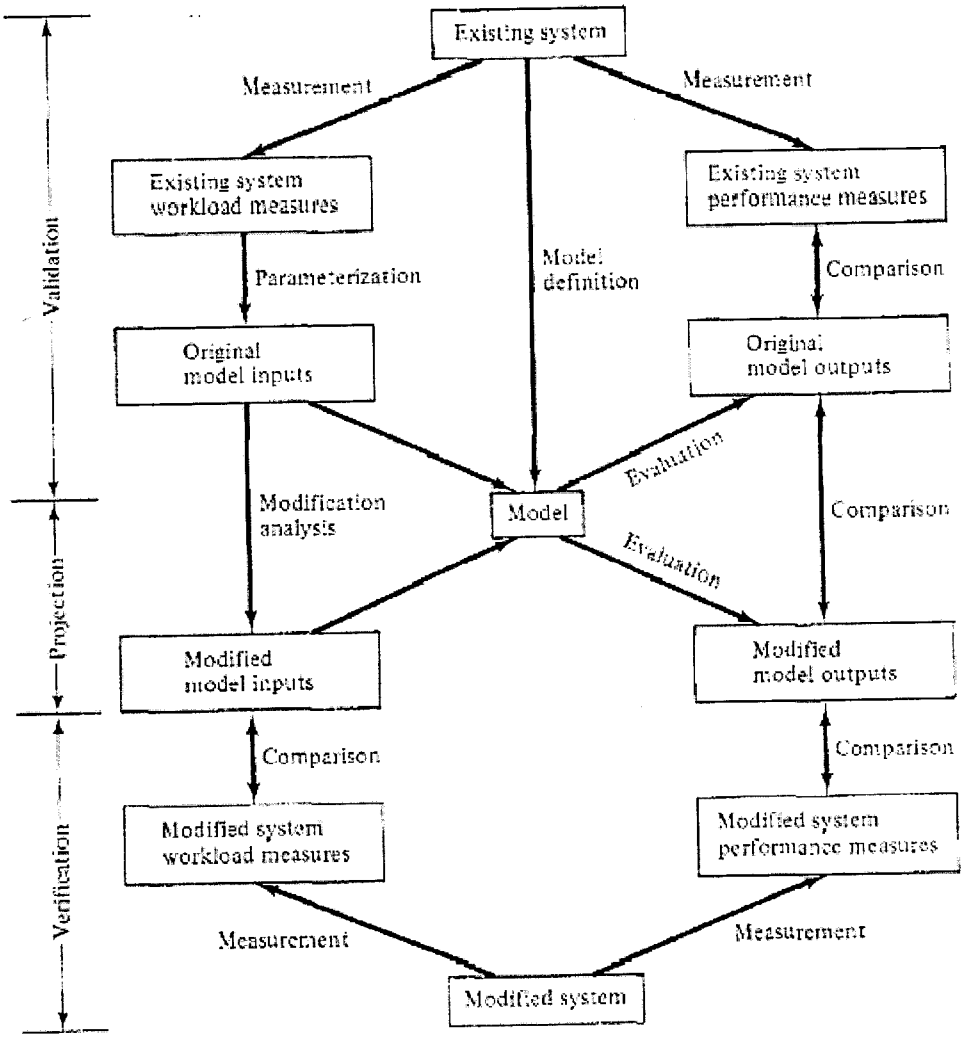


Figure 1.2.1 - The Modelling Cycle (Kienzle & Sevcik 1979)

In the projection phase, model inputs are modified to reflect the anticipated changes to the system or workload. The model then is evaluated. The difference between the modified model outputs and the original model outputs is the projected effect of the modification.

Finally, in the verification phase, the modified system is measured and two comparisons are made. First, its performance measures are compared with the model outputs. Second, its workload measures are compared with the model inputs. Discrepancies between the projections of the model and the performance of the system can arise from two sources: the omission or misrepresentation of (retrospectively) significant system characteristics, and the evolution of the system in a way that differs from that which was anticipated. Understanding and evaluating these sources of discrepancy is crucial to gaining confidence in queueing network modelling as a computer system analysis technique (Trivedi 1982). The accuracy of a model's

performance projections can be no greater than the accuracy of the workload projections furnished as input.

Although we have presented the modelling cycle, conducting a modelling study is hardly a strictly sequential process. There are strong dependencies among the various components of the validation and projection phases. Compatibility must be achieved between the definition of the model, the measurements used to parameterize the model, and the techniques used to evaluate the model. Achieving this compatibility, and reconciling it with the objectives of a particular modelling study, is inherently iterative in nature.

The validation phase of a modelling study obviously requires a thorough understanding of the computer system under consideration. Perhaps a thorough understanding of the objectives of the study is obviously important. In fact, though, this latter understanding is a key component of the top-down philosophy of queueing network modelling. Many system characteristics that would need to be represented in a fully general model may be irrelevant in a particular study. Identifying these characteristics leads to a simpler model and a simpler modelling study.

Devising such a procedure is feasible, but it adds considerably to the complexity of the modelling study, and it provides a level of generality that is not required. The objective of this study was restricted to estimating the relative performance of each of the fifteen benchmarks on the two configurations. We can achieve a significant simplification by assuming that the paging and swapping activity of each user, while sensitive to changes in the mix of workload components, are insensitive to changes in CPU speed. This assumption allows the paging and swapping service demands of each workload component to be measured for each of the benchmarks during the RTE experiments.

The two approaches to this computer system analysis problem are contrasted in Figure 1.2.2. The assumption on which the simplified approach relies is not valid universally, but any inaccuracies that result are strictly secondary.

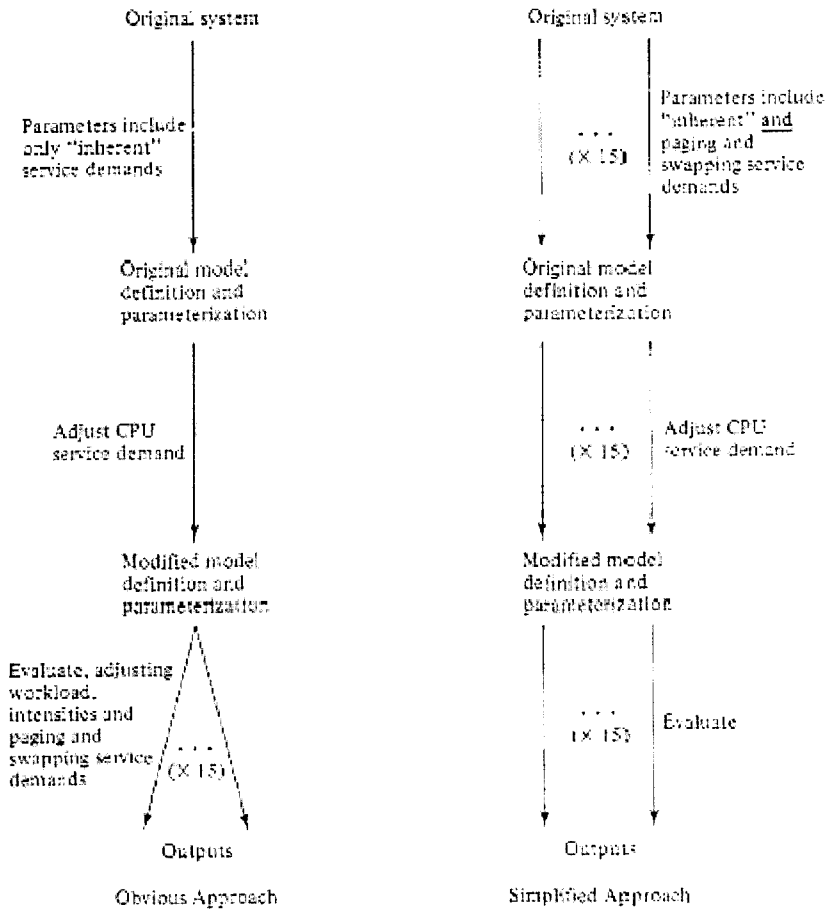


Figure 1.2.2 - Two Approaches to Modelling a CPU Replacement (Kienzle & Sevcik 1979)

1.2.3. Workload Characterization

In discussing the validation phase of the modelling cycle, measurement is identified as the process of obtaining workload measures. These activities, while not necessarily straightforward, are often less difficult than workload characterization: the process of selecting the workload or workloads on which to base the performance study.

Difficult questions for uncertainties arise even in considering an existing computing environment. These uncertainties are compounded in considering an environment that cannot be measured directly. Every approach to computer system analysis - intuition and trend extrapolation, experimental evaluation of alternatives, or modelling - requires workload characterization. Strangely, the imprecision inherent in workload characterization argues for queueing network models. In principle, greater accuracy might be obtained (at significantly greater cost) through experimentation or through simulation modelling. In practice, however, the dominant source of error is apt to

lie with the workload characterization, even when queueing network models are employed.

1.2.4. Sensitivity Analysis

Every computer system analyst encounters situations in which questionable assumptions must be introduced. Sensitivity analysis can be used to determine the extent to which such assumptions cast doubt on the conclusions of the study (Lazowska 1980). The analyst may test the robustness of the results to the assumption in question. This involves evaluating the model a number of times for variations in the assumption, and comparing the results. The analyst may obtain bounds on the expected performance, by evaluating the model for extreme values of the assumption.

Difficulty was encountered during the validation phase because a significant proportion of the system's I/O activity was not attributed to specific workload components by the available measurement tools. For example, determining the total number of swaps during a measurement interval was possible, and the average disk service demand per swap. However, determining which user was not possible or workload component was the "victim" of the swap. Had the study been based on a single class model, this would not have been a problem.

Various methods of allocating this measured I/O activity among the four workload components yielded different values for some input parameters of the model (Myhre [1979). Not surprisingly, different response time projections from the model resulted.

1.3. Fundamental Laws

1.3.1. Introduction

It has three objectives. The first is to define a number of quantities of interest and to introduce the notation that we will use in referring to these quantities. The second is to derive various algebraic relationships among these quantities, some of which, because of their importance, will be identified as fundamental laws (Buzen 1976). The third is to explore thoroughly the most important of these fundamental laws, Little's law (named for J.D.C. Little). Little's law states that the average number of requests in a system must equal the product of the throughput of that system and the average time spent in that system by a request.

1.3.2. Basic Quantities

If we were to observe the abstract system shown in Figure 1.3.1 we might imagine measuring the following quantities:

T, the length of time we observed the system

A, the number of request arrivals we observed

C, the number of request completions we observed

From these measurements we can define the following additional quantities:

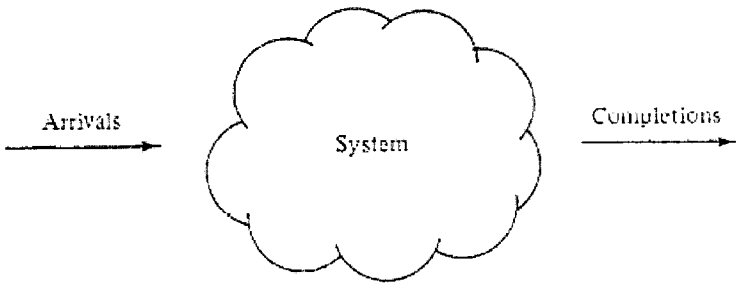


Figure 1.3.1 - An Abstract System

λ, the arrival rate: $\lambda \equiv \frac{A}{T}$

If we observe 8 arrivals during an observation interval of 4 minutes, then the arrival rate is 8/4 = 2 requests/minute.

X, the throughput: $X \equiv \frac{C}{T}$

If we observe 8 completions during an observation interval of 4 minutes, then the throughput is 8/4 = 2 requests/minute.

If the system consists of a single resource, we also can measure:

B, the length of time that the resource was observed to be busy. Two more defined quantities now are meaningful:

U, the utilization: $U \equiv \frac{B}{T}$

If the resource is busy for 2 minutes during a 4 minute observation interval, then the utilization of the resource is 2/4, or 50%.

S, the average service requirement per request: $S \equiv \frac{B}{C}$

If we observe 8 completions during an observation interval and the resource is busy for 2 minutes during that interval, then on the average each request requires 2/8 minutes of service.

We now can derive the first of our fundamental law. Algebraically, $\frac{B}{T} = \frac{C}{T}\frac{B}{C}$, From the three preceding definitions, $\frac{B}{T} \equiv U$, $\frac{C}{T} \equiv X$, and $\frac{B}{C} \equiv S$ Hence.

The Utilization Law: U = XS

That is, the utilization of a resource is equal to the product of the throughput of that resource and the average service requirement at that resource.

1.3.3. Little's Law

The utilization law in fact is a special case of Little's law, which we now will derive in a more general setting (Little 1961). Figure 3.2 is a graph of the total number of arrivals and completions occurring at a system over time. Each step in the higher step function signifies the occurrence of an arrival at that instant; each step in the lower signifies a completion. At any instant, the vertical distance between the arrival and completion functions represents the number of requests present in the system. Over any interval, the area between the arrival and completion functions represents the accumulated time in system during that interval, measured in request-seconds. For example, if there are three requests in the system during a two-second period, then six request-seconds are accumulated. This area is shaded in Figure 3.2 for an observation interval of length T = 4 minutes. We temporarily denote accumulated time in system by W. We define:

N, the average number of requests in the system: $N \equiv \frac{W}{T}$

If a total of 2 request-minutes of residence time are accumulated during a 4 minute observation interval, then the average number of requests in the system is 2/4 = 0.5.

R, the average system residence time per request: $R \equiv \frac{W}{C}$

If a total of 2 request-minutes of residence time are accumulated during an observation interval in which 8 requests complete, then the average contribution of each completing request (informally, the average system residence time per request) is 2/8 = 0.25 minutes.

Algebrically, $\frac{W}{T} = \frac{C}{T}\frac{W}{C}$. But $\frac{W}{T} \equiv N$, $\frac{C}{T} \equiv X$, and $\frac{W}{C} \equiv R$.

Hence:

Little's Law: N = XR

That is, the average number of requests in a system is equal to the product of the throughput of that system and the average time spent in that system by a request.

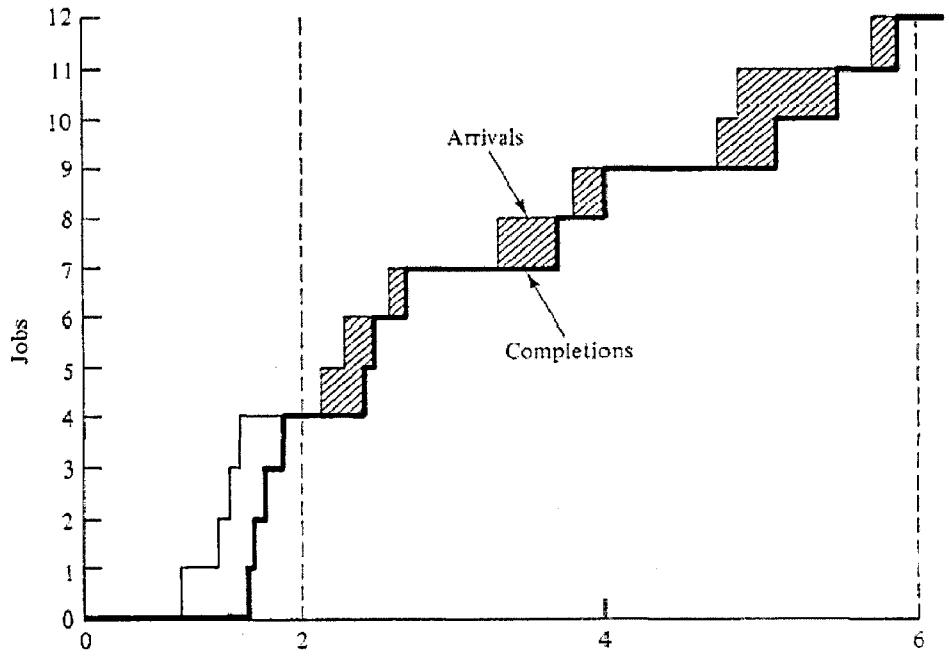


Figure 1.3.2 - System Arrivals and Completions (Little 1961)

A subtle but an important point in our derivation of Little's law is that the quantity R does not necessarily correspond to our intuitive notion of average residence time or response time - the expected time from arrival to departure. This discrepancy is due to end effects: knowing how to account for requests that are present just is hard prior to the start or just after the end of an observation interval. For the time being, suffice it to say that if the number of requests passing through the system during the observation interval is much greater than the number present at the beginning or end, then R corresponds closely to our intuition, and if the observation interval begins and ends at instants when system is empty, then this correspondence is exact.

Little's law is important for three reasons. First, because it is so widely applicable (it requires only very weak assumptions), it will be valuable to us in checking the consistency of measurement data. Second, in studying computer systems we frequently will find that we know two of the quantities related by Little's law and desire to know the third. Third, Little's law is central to the algorithms for evaluating queueing network models. Given a computer system, Little's law can be applied at many different levels: to a single resource, to a subsystem, or to the system as a whole. The key to success is consistency: the definitions of population, throughput, and residence time must be compatible with one another. In Figure 1.3.3 we illustrate this by applying Little's law to

a hypothetical timesharing system at four different levels, as indicated by the four boxes in the figure.

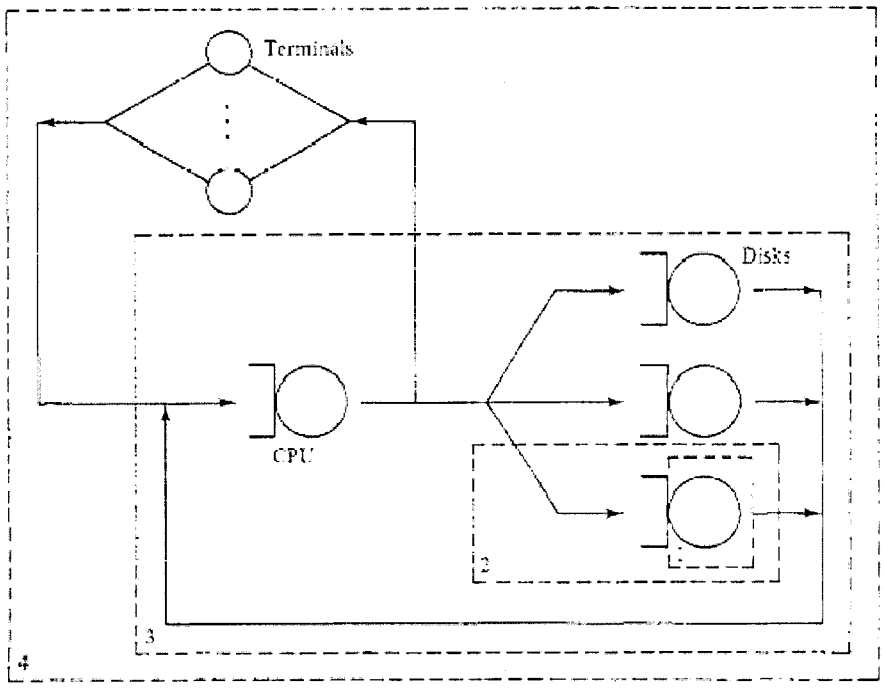


Figure 1.3.3 - Little's Law Applied at Four Levels (Little 1961)

Box 1 is perhaps the most subtle; it illustrates the application of Little's law to a single resource, nor including its queue. In this example, population corresponds to the utilization of the resource, throughput corresponds to the rate at which the resource is satisfying requests, and residence time corresponds to the average service requirement per request at the resource. This application of Little's law constitutes an alternative derivation of the utilization law.

Box 2 illustrates the application of Little's law to the same resource, this time including its queue. Now, population corresponds to the total number of requests either in queue or in service, throughput remains the rate at which the resource is satisfying requests, and residence time corresponds to the average time that a request spends at the resource per visit, both queueing time and service time.

Box 3 illustrates the application of Little's law to the central subsystem - the system without its terminals. The definition of "request" changes at this level: it is no longer interested in visits to a particular resource, but rather in system-level interactions. Population corresponds to the number of customers in the central subsystem, i.e., those

users not thinking. Throughput corresponds to the rate at which interactions flow between the terminals and the central subsystem.

Finally, box 4 illustrates the application of Little's law to the entire system, including its terminals. Here, population corresponds to the total number of interactive users, throughput corresponds to the rate at which interactions flow between the terminals and the system, and residence time corresponds to the sum of system response time and user think time.

If Z denote think time then it can write this incarnation of Little's law as N = X(R + Z). As with the utilization law, this application is so ubiquitous that its own name and notation, expressing R in terms of the other quantities:

The Response Time Law: R = N/X - Z

1.3.4. The Forced Flow Law

In discussing Little's law, it allowed view to range from an individual resource to an entire system. At different levels of detail, different definitions of "request" are appropriate. The relationship between these two views of a system is expressed by the forced flow law, which states that the flows (throughputs) in all parts of a system must be proportional to one another (Denning & Buzen 1978). Suppose that during an observation interval it can be count not only system completions, but also the number of completions at each resource. It define the visit count of a resource to be the ratio of the number of completions at that resource to the number of system completions. If a variable with the subscript k refer to the k-th resource, then we can write this definition as:

V_k, the visit count of resource k: $V_k = C_k/C$

If we rewrite this definition as $C_k = V_k\ C$ and recall that the completion count divided by the length of the observation interval is defined to be the throughput, then the throughput of resource k is given by:

The Forced Flow Law: $X_k = V_kX$

An informal statement of the forced flow law is that the various components of a system must do comparable amounts of work (measured in "transaction's worth") in a given time interval. Little's law becomes especially powerful when combined with the forced flow law. To apply the response time law: $R = (N/X) - Z$. The number of terminals and the average think time need to be known, but are missing the throughput.

Note that it can describe an interaction's disk service requirement in either of two ways: by saying that an interaction makes a certain number of visits to the disk and

:quires a certain amount of service on each visit, or by specifying the total amount of isk service required by an interaction. These two points of view are equivalent, and vhichever is more convenient should be chosen. We define:

D_k, the service demand at resource k: $D_k = V_k S_k$

From now on S_k will be used to refer to the Service requirement per visit at resource k, and D_k to refer to the total service requirement at that resource. D with no subscript is defined to be the sum of the D_k: the total service demanded by a job at all resources.

Again, consistency is crucial to success. Consider using the utilization law to calculate the utilization of a resource. It can be express throughput in terms of visits to that resource (X_k), in which case service requirement must be expressed as service requirement per visit (S_k). Using the forced flow law, it can also express throughput in terms of system-level interactions (X), in which case service requirement must be expressed on a per-interaction basis (D_k). In other words, $U_k = X_k S_k = X D_k$.

Service demands are one of the parameters required by queueing network models. If it follows a system for an interval of length T, it can easily be obtain the utilizations of the various resources, U_k, and the system-level completion count, C. The service demands at the various resources then can be calculated as $D_k = B_k/C = U_k T/C$. It is fortunate that queueing network models can be parameterized in terms of the D_k rather than the corresponding V_k and S_k, since the former typically are much more easily obtained from measurement data than the latter.

1.3.5. The Flow Balance Assumption

Frequently it will be convenient to assume that systems satisfy the flow balance property, namely, that the number of arrivals equals the number of completions, and thus the arrival rate equals the throughput:

The Flow Balance Assumption: $A = C$, therefore $\lambda = X$

The flow balance assumption can be tested over any measurement interval, and it can be strictly satisfied by careful choice of measurement interval. When used in conjunction with the flow balance assumption, Little's law and the forced flow law allow us to calculate device utilizations for systems whose workload intensities are described in terms of an arrival rate.

Assuming that the system is able to handle the offered load, the flow balance assumption is reasonable. Thus, the throughput of the system will be the same as the arrival rate to the system. The forced flow law guarantees that the various devices in the system will do comparable amounts of work (measured Departures in "transaction's worth") in a given time. Interference between transactions does not affect this. Rather, it causes an increase in the average number of transactions resident in the system, which causes a corresponding increase in response time (by Little's law).

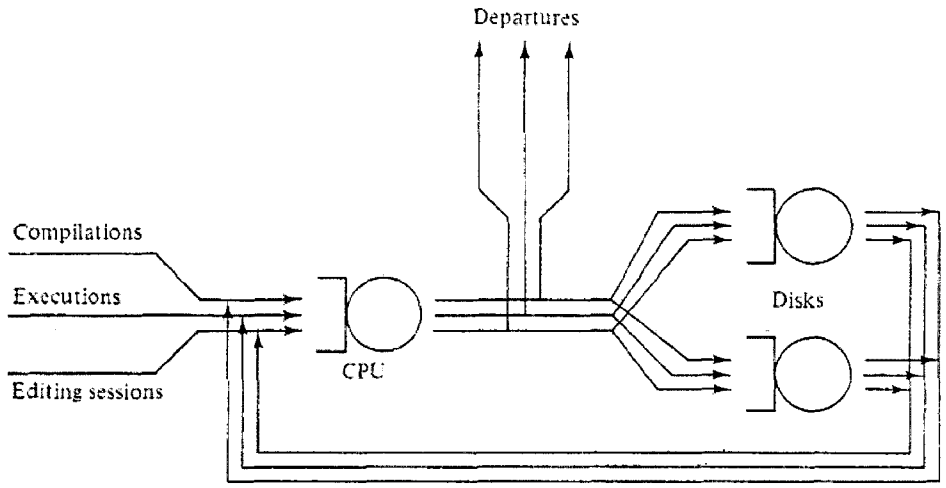


Figure 1.3.5 - Calculating Utilizations Using Flow Balance (Denning & Buzen 1978)

1.4 Queueing Network Model Inputs and Outputs

1.4.1. Introduction

We need to state precisely the inputs and outputs of queueing network models. It extended where necessary for the accurate representation of particular computer system characteristics. We have described the inputs and outputs of separable queueing networks. For notational simplicity we first present this material in models with a single customer class (Kleinrock 1976). We discuss certain computer system characteristics that cannot be represented directly using the inputs available for separable models, and certain performance measures that cannot be obtained directly from the available outputs.

1.4.2. Model Inputs

The basic entities in queueing network models are service centers, which represent system resources, and customers, which represent users or jobs or transactions. Table 1.4.1 lists the inputs of single class queueing network models, which describe the

relationships between customers and service centers. In the subsections that follow, these parameters are discussed in some detail.

1.4.2.1. Customer Description

The workload intensity may be described in any of three ways, named to suggest the computer system workloads they are best suited to representing:

customer decription	*The workload intensity, one of:* *l, the arrival rate (for transaction workloads), or* *N, the population (for batch workloads), or* *N and Z, the think time (for terminal workloads)*
center description	*K, the number of service centers* *For each service center k:* *its type, either queueing or delay*
service demands	*For each service center k:* *$Dk \equiv V_k S_k$, the service demand*

Table 1.4.1 - Single Class Model Inputs

- A transaction workload has its intensity specified by a parameter A, indicating the rate at which requests (customers) arrive. A transaction workload has a population that varies over time. Customers that have completed service leave the model.
- A batch workload has its intensity specified by a parameter N, indicating the average number of active jobs (customers). (N need not be an integer.) A batch workload has a fixed population. Customers that have completed service can be thought of as leaving the model and being replaced instantaneously from a backlog of waiting jobs.
- A terminal workload has its intensity specified by two parameters: N, indicating the number of active terminals (customers), and Z, indicating the average length of time that customers use terminals ("think") between interactions. (Again, N need not be an integer.)

A terminal workload is similar to a batch workload in that its total population is fixed. In fact, a terminal workload with a thinks time of zero is in every way equivalent to a batch workload. On the other hand, a terminal workload is similar to a transaction workload in that the population of the central subsystem (the system excluding the terminals) varies, if the terminal workload has a non-zero think time. Note that N is an upper bound on the central subsystem population of a terminal workload, whereas no upper bound exists for a transaction workload.

We sometimes refer to models with transaction workloads as open models. Open model is an infinite stream of arriving customers. Models with batch or terminal

workloads are called closed models. This distinction is made because the algorithms used to evaluate open models differ from those used for closed models. It highlights the similarity between batch and terminal workloads.

1.4.2.2. Center Description

Service centers may be of two types: queueing and delay. These are represented as shown in Figure 1.4.1.

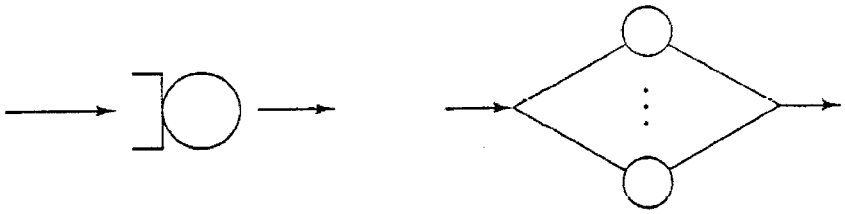


Figure 1.4.1 - Queueing and Delay Service Centers

Customers at a queueing center compete for the use of the server. Thus, the time spent by a customer at a queueing center has two components: time spent waiting, and time spent receiving service. Queueing centers are used to represent any system resource at which users compete for service, e.g., the CPU and I/O devices. As shown in the figure, a queueing center is drawn as a queue plus a server. Because customers in a single class model are indistinguishable, specifying the scheduling discipline at a queueing center is not necessary. The same performance measures will result from any scheduling discipline in which exactly one customer is in service whenever there are customers at the center.

Customers at a delay center each (logically) are allocated their own server, so there is no competition for service. Thus the residence time of a customer at a delay center is exactly that customer's service demand there. The most common use of a delay center is to represent the think time of terminal workloads. However, delay centers are useful in any situation in which imposing some known average delay is necessary. For instance, a delay center could be used to represent the delay incurred by sending large amounts of data over a dedicated low speed transmission line. As shown in the figure, an icon suggesting concurrent activity is used to represent a delay center.

1.4.2.3. Service Demands

The service demand of a customer at center k, D_k, is the total amount of time the customer requires in service at that center (Trivedi 1982). Thus the set of service demands

(one for each center) characterizes the behavior of the customer in terms of processing requirements. In a single class model, customers are indistinguishable with respect to their service demands, which can be thought of as representing the "average customer" in the actual system. D_k can be calculated directly as B_k/C (the measured busy time of device k divided by the measured number of system completions), or may be thought of as the product of V_k the number of visits that a customer makes to center k, and S_k the service requirement per visit. It is possible to parameterize queueing network models at this more detailed level. However, a surprising characteristic of separable queueing networks is that their solutions depend only on the product of V_k and S_k at each center, and not on the individual values. Thus a model in which customers make 100 visits to the CPU, each for 10 milliseconds of service, is equivalent to one in which customers make a single visit for one second of service. For simplicity (to reduce the number of parameters and to facilitate obtaining their values) we generally will choose to parameterize our models in terms of D_k. Note that we defiy D to be the total service demand of a customer at all centers:

$$D = \sum_{k=1}^{k} D_k \text{ D.}$$

1.4.3. Model Outputs

Table 1.4.2 lists the outputs obtained by evaluating a single class queueing network model. Comments appear in the subsections that follow.

system measures	R	*average system response time*
	X	*system throughput*
	Q	*average number in system*
center measures	U_k	*utilization of center k*
	R_k	*average residence time at center k*
	X_r	*throughput of center k*
	Q_k	*average queue length at center k*

Table 1.4.2 - Single Class Model Outputs

The values of these outputs depend upon the values of all of the model inputs. It will be especially useful to be able to specify that an output value corresponds to a particular workload intensity value. To do so, we follow the output with the parenthesized workload intensity: X(N) is the throughput for a batch or terminal class with population N, Qk (λ) is the average queue length at center k for a transaction class with arrival rate λ, etc.

1.4.3.1. Utilization

The utilization of a center may be interpreted as the proportion of time, and the device is busy, or, equivalently, as the average number of customers in service there.

1.4.3.2. Residence Time

Just as D, is the total service demand of a customer at center k (in contrast to S_k, the service requirement per visit), R_k is the total residence time of a customer at center k (as opposed to the time spent there on a single visit). If the model is parameterized in terms of V_k and S_k, then the time spent per visit at center k can be calculated as R_k / V_k.

System response time, R, corresponds to our intuitive notion of response time; for example, the interval between submitting a request and receiving a response on an interactive system. Obviously, system response time is the sum of the residence times at the various centers:

$$R = \sum_{k=1}^{k} R_k$$

1.4.3.3. Throughput

If a model is parameterized in terms of D_k then we can obtain system throughput, X, but do not have sufficient information to calculate device throughputs, X_k.. If a model is parameterized in terms of V_k, and S_k, then device throughputs can be calculated using the forced flow law, as $X_k = V_k X$.

1.4.3.4. Queue Length

The average queue length at center k, Q_k, includes all customers at that center, whether waiting or receiving service. The number of customers waiting can be calculated as Q_k - U_k since U, can be interpreted as the average number of customers receiving service at center k. Q denotes the Qverage number in system. For a batch class, $Q = N$. For a transaction class, $Q = XR$ (by Little's law). For a terminal class, $Q = N - XZ$ ($Q = XR$, and $R = N/X - Z$.). In general, the average population of any subsystem can be obtained either by multiplying the throughput of the subsystem by the residence time there, or by summing the queue lengths at the centers belonging to the subsystem.

1.4.3.5. Other Outputs

Various other outputs can be computed at some additional cost. As one example, we occasionally will wish to know the queue length distribution at a center: the

proportion of time that the queue length has each possible value. We denote the proportion of time that the queue length at center k has the value i by $P[Q_k=i]$.

1.4.4. Multiple Class Models

1.4.4.1. Inputs

Multiple class models consist of C customer classes, each of which has its own workload intensity (λ_c, N_c or N_c and Z_c) and its own service demand at each center ($D_{c.k}$). Within each class, the customers are indistinguishable. Multiple class models consisting entirely of open (transaction) classes are referred to as open models. Models consisting entirely of closed (batch or terminal) classes are referred to as closed. Models consisting of both types of classes are referred to as mixed.

The overall workload intensity of a multiple class model is described by a vector with an entry for each class: $\lambda \equiv (\lambda_1, \lambda_2, \ldots\ldots, \lambda_c)$ if the model is open, N = (N_1 , N_2 , .,. , N_c) if it is closed (in point of fact, 3 also must be included for terminal classes), and I =G (N_1 or λ_c , N_2 or λ_2 , . . . , N_c or λ_c) if it is mixed.

As was the case for single class models, we do not specify the scheduling discipline at a queueing center. Roughly, the assumption made is that the scheduling discipline is class independent, i.e., it does not make use of information about the class to which a customer belongs. The same performance measures will result from any scheduling discipline that satisfies this assumption, along with the earlier assumption that exactly one customer is in service whenever there are customers at the center.

Table 1.4.3 summarizes the inputs of multiple class models. By analogy to the single class case, we define D_c to be the total service demand of a , class c customer at all centers:

$$D_c \cong \sum_{k=1}^{k} D_{c.k}$$

1.4.4.2. Outputs

All performance measures can be obtained on a per-class basis (e.g., U, k and X,) as well as on an aggregate basis (e.g., $U_{c,\,k}$ and X_c. For utilization, queue length, and throughput, the aggregate performance measure

customer description	*C, the number of customer classes* *For each class c:* *its workload intensity, one of* *λ_c, the arrival rate (for transactions workloads), or* *N_c, the population (for batch workloads), or* *N_c and Z_c, the think time (for terminal workloads)*
center description	*K, the number of service centers* *For each service center k;* *its type, either queueing or delay*
service demands	*For each class c and center k;* *$D_{c,k} \equiv V_{c,k}S_{c,k}$, the service demand*

Table 1.4.3 - Multiple Class Model Inputs

equals the sum of the per-class performance measures (e.g., $U_k U_k \equiv \sum_{c=1}^{C} U_{c,k}$.). For residence time and System response time, however, the per-class measures must be weighted by relative throughput, as follows:

$$R = \sum_{c=1}^{C} \frac{R_c X_c}{X} \qquad R_k = \sum_{c=1}^{C} \frac{R_{c,k} X_c}{X}$$

This makes intuitive sense, and can be demonstrated formally using Little's law.

Table 1.4.4 summarizes the outputs of multiple class models. The following reminders, similar to comments made in the context of single class models, should be noted in studying the table:

- The basic outputs are average values (e.g., average response time) rather than distributional information (e.g., the 90th percentile of response time). Thus the word "average" should be understood even if it is omitted.
- X_k and $X_{c,k}$ are meaningful only if the model is parameterized in terms of $V_{c,k}$ and $S_{c,k}$, rather than $D_{c,k}$.
- To specify that an output value corresponds to a particular workload intensity value, we follow the output symbol with the parenthesized workload intensity.

system measures	*aggregate*	R	*average system response time*
		X	*system throughput*
		Q	*average number in system*
	per class	R_i	*average class c system response time*
		X_i	*class c system throughput*
		Q_i	*average class c number in system*
center measures	aggregate	U_k	*utilization of center k*
		R_k	*average residence time at center k*
		X_i	*throughput at center k*
		Q_k	*average queue length at center k*
	per class	$U_{c,k}$	*class c utilization of center k*
		$R_{c,k}$	*average class c residence time at center k*
		$X_{c,k}$	*class c throughput at center k*
		$Q_{c,k}$	*average class c queue length at center k*

Table1.4.4 - Multiple Class Model Outputs

1.5 Application on Computer Communication Networks

1.5.1 Introduction

Computer communication networks use a variety of flow control policies to achieve high throughput, low delay, and stability. Here, we model the flow control policy of IBM's System Network Architecture (SNA). SNA routes messages from sources to destinations by way of intermediate nodes which temporarily buffer the messages. Message buffers are a scarce resource (Schwartz 1977). The flow control policy regulates the flow of messages between source/destination pairs trying to avoid problems such as deadlock and starvation, which could result from poor buffer management. SNA has a window flow control policy. The key control parameter is the window size, W. When a source starts sending messages to a particular destination, a pacing count at the source is initialized to the value of W. This pacing count is decremented every time a message is sent. If the pacing count reaches zero, the transmission of messages is halted.

When the first message of a window reaches the destination, a pacing response is returned to the source. Upon receipt, the source increments the current value of the pacing count by W (Schwartz 1982). Another pacing response is sent to the source by the destination each time an additional W messages have been received. Thus, the maximum number of messages that can be en route from source to destination at any time is 2 W- 1. Our objective is to model the "response time" of messages between a single source/destination pair - the average time required for messages to flow from source to destination. The most convenient model, for simplicity and ease of evaluation, is an open

queueing network. There are M centers, representing the source node, the destination node, and M-2 intermediate nodes. (Obviously, M is determined by adding two to the number of intermediate nodes.) Customers, which represent messages, arrive at the source node at rate X. They flow from node to node, requiring D units of service at each node. This model is shown in Figure 1.5.1.

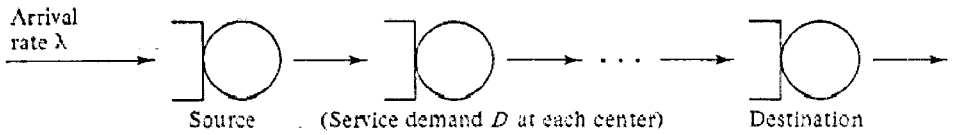


Figure 1.5.1 - Open Model of SNA Flow Control (1982 IEEE)

Response times can be calculated easily for this model. Unfortunately, the model makes a significant simplifying assumption which impacts the applicability of the results: there is no representation of the flow control policy! The source continues to transmit, despite the number of outstanding messages.

A more realistic approach, therefore, is to use a closed model, in which representing the limit on the number of outstanding messages will be possible, Figure 5.2 shows this model. There are 2 W- 1 customers, representing the possible outstanding messages. As in the open model, M centers are corresponding to the source node, the destination node, and M-2 intermediate nodes. Customers have service demand D at each of these centers. In addition, there are a "message generation" center and a "pacing box". Together, the message generation center and the pacing box mimic the flow control policy, in the following way.

The pacing box "stores" up to W-l messages. When the W-th message arrives, it triggers the discharge of all W messages into the queue of the message generation center. The message generation center has service rate X; as long as its queue is non-empty, it will generate message traffic at this rate. A bit of thought will reveal that the arrival of the W-th message to the pacing box corresponds to the source's receipt of a pacing response from the destination; such receipt carries with it the right to initiate W additional messages.

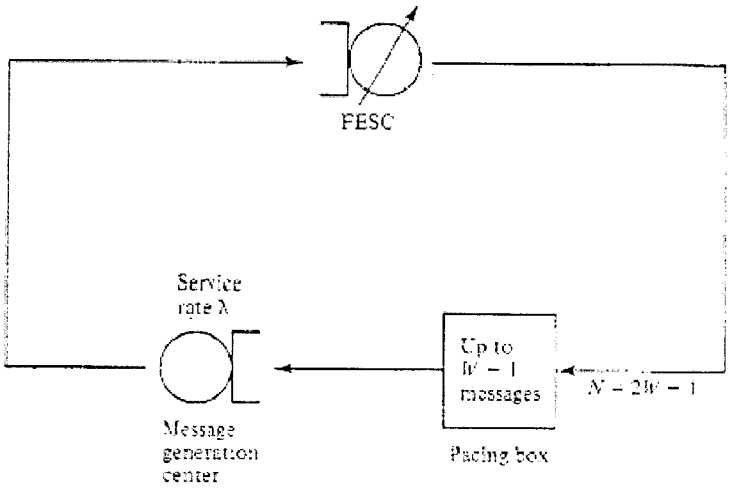


Figure 1.5.2 - Closed Model of SNA Flow Control (1982 IEEE)

The model of Figure 1.5.2, while realistic, is not separable, because of the unusual characteristics of the pacing box. The model could be evaluated directly using the global balance approach. However, the potentially large size of the model makes this approach infeasible in general. A alternative, is to replace the A4 centers representing the source, destination, and intermediate nodes with an FESC. The resulting three node model of Figure 1.5.3 still is not separable, but it is small enough for global balance to be practical.

The load dependent service rates of the FESC are estimated in the usual way. A closed, separable model consisting of the M centers representing the nodes, each with service demand D, is evaluated for each feasible message population, from 1 to 2 W- 1. Throughputs are determined, and used to define the FESC. Once this has been accomplished, writing the global balance equations and numerically evaluating them to obtain the equilibrium state probabilities is tedious but straightforward. These probabilities yield system throughput and average queue length at the FESC. Little's law then can be applied to determine average response time.

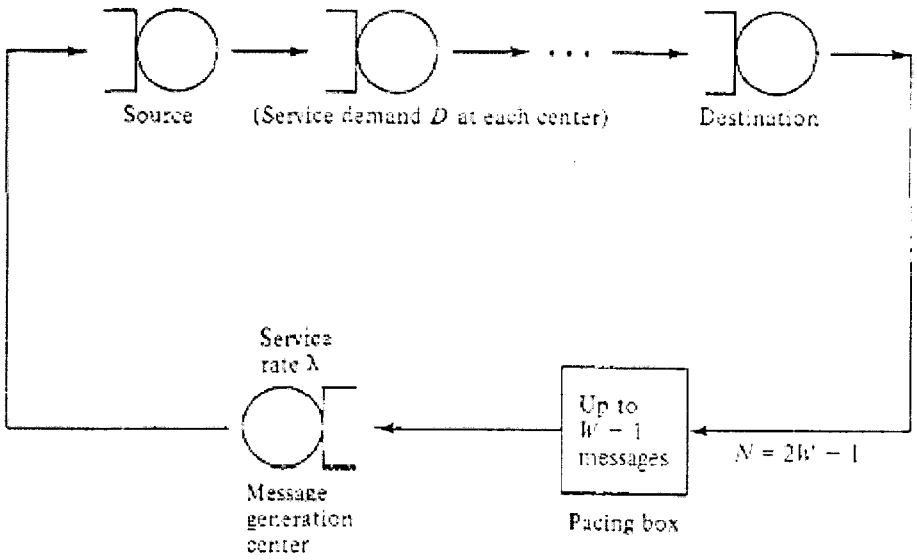


Figure 1.5.3 - FESC Representing the Message Path (1982 IEEE)

One assumption made by this modelling approach is that the only traffic passing through a node is due to the source/destination pair of interest. This unrealistic assumption can be eliminated by modifying the separable model used to estimate the load dependent service rates of the FESC. As one approach, if the traffic at each node due to other source/destination pairs is known, it can be represented as an open class. This type presence will impede the progress of messages associated with the source/destination pair of interest, with a resulting decrease in FESC rates.

Computer communication networks such as SNA are designed to perform well over long distances at moderate bandwidths (Metcalfe & Boggs 1976). Local area networks, on the other hand, are optimized for use over moderate distances (say, 1 km.) at high bandwidths (10 MHz. or greater).

1.5.2. Software Resources

The usual viewpoint in constructing queueing network models is that service centers correspond to hardware resources. It also is the case, though, that queueing delays in computer systems can arise from contention for software resources: operating system critical sections, non-reentrant software modules, etc. In this section we consider the use of queueing network models to evaluate software system structures (Agre & Tripathi 1982).

Our approach will be to define a software-level queueing network model in which customers, as usual, correspond to users, but in which service centers correspond to

software modules. The service demand at each center will be equal to the time the customer spends executing the corresponding software module. The queueing delay at each center, calculated when the model is evaluated, will be an estimate of the time the customer is blocked awaiting access to the corresponding software module. A reentrant software module will be represented as a delay center, since a customer is never blocked awaiting access. A nonreentrant module will be represented as a queueing center, since only one customer can be executing it at a time.

Obviously, the service demand at each center in the software-level model includes various service requirements and queueing delays incurred in executing the corresponding software module on the underlying computer system. This service demand can be thought of as the "response time" of the user once access to the software module has been granted. This service demand will be estimated using a more conventional hardware-level queueing network model, in which customers correspond to users executing software modules, and centers correspond to hardware resources (Smith and Browne 1980). The service demands are easily obtained for this hardware-level model, but the customer population cannot be known. The degree of concurrency at the hardware level depends upon the extent to which users are blocked awaiting accesses to modules at the software level. Thus, an iterative solution is required, in which the hardware-level model provides service demand estimates for the software-level model, which in turn provides customer population estimates for the hardware level model.

A simple example of a software-level model is shown in Figure 1.5.4. There are centers corresponding to various software activities: editing, compilation, linking, loading, and execution. There are various possible "execution sequences": edit and compile; compile, link, and execute; load and execute; etc. Each execution sequence is represented as a separate customer class (Agrawal & Buzen 1983). The number of customers in each class is the number of users performing the corresponding execution sequence.

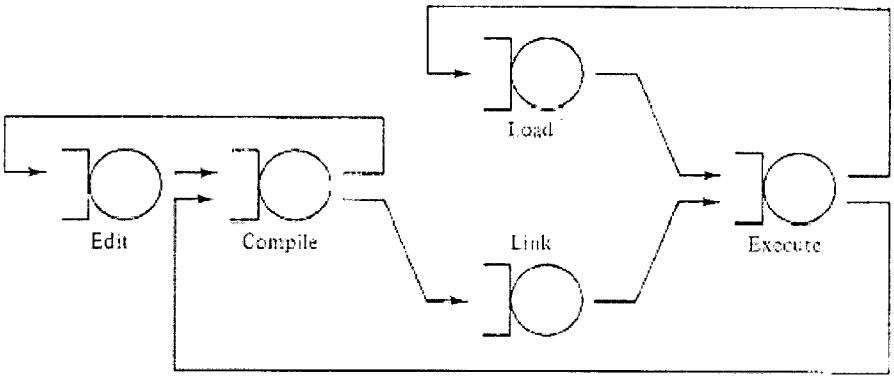


Figure 1.5.4 - A Software-Level Queueing Network Model (1982 IEEE)

Once the service demands for the centers in the software-level model are known, the model can be evaluated. From the results, the average number of users concurrently executing each software module can be estimated. If a module is reentrant it will be represented as a delay center, and the average population at that delay center will be the average number of users concurrently executing the module. If a module is non-reentrant it will be represented as a queueing center, and the utilization of that center will be the proportion of time that a user is executing the module.

To estimate the service demands for the centers in the software-level model, we use the hardware-level model (Jacobson & Lazowska 1983). As noted earlier, customers in this model correspond to users executing software modules. One class represents each module. The service demands of the various classes at the various centers are determined by the resource requirements of the corresponding software modules. The response time of a class in this hardware-level model determines the service demand at the center corresponding to the same software module in the software-level model.

1.5.3. Database Concurrency Control

In any database system, many users will wish to access and update the database concurrently (Sevcik 1983). Problems may arise if this concurrency is undisciplined:

- The database may become inconsistent because of an unfortunate interleaving of reads and writes by various users.
- Even if the database remains consistent, individual users may "see" inconsistent views, again because of an unfortunate interleaving of activity.

To free the user from concern for problems such as these, the concept of a transaction has been devised. The key property of a transaction is atomicity :

- The user executing a transaction is guaranteed a single, consistent view of the database, regardless of the activities of other users.
- Other users perceive a transaction as a single action, rather than as a series of separate reads and writes of data items.

The job of a concurrency control mechanism is to allow transactions to be executed concurrently while guaranteeing that the consistency of the database is preserved (Bernstein & Goodman 1981). A crude concurrency control mechanism would grant exclusive access to the entire database to one transaction for its duration. (Concurrency is restricted unnecessarily by this simple solution: two transactions that reference entirely different sets of data items would be unable to proceed concurrently.) A more reasonable mechanism would grant exclusive access to various data items to one transaction for its duration. Other possibilities exist. Clearly, the presence of a concurrency control mechanism can have a significant effect on system performance - an effect somewhat analogous to that of a memory constraint. Equally clearly, a queueing network model that represents the concurrency control mechanism directly will be non-separable: customers may be blocked when data items they require are held by other customers.

Chapter II

General Analytic Techniques for Queueing Network Model

Chapter II of this research discusses algorithms for evaluating separable queueing network models. To evaluate a queueing network model is to obtain outputs such as utilizations, residence times, queue lengths, and throughputs, from inputs such as workload intensities and service demands.

In section 2.1 we show how to obtain bounds on performance, using extremely straightforward reasoning and simple computations that can be performed by hand. Using fundamental laws, bounds on center utilizations and throughputs can be calculated from the asymptotic and balanced system bounds on system throughput.

In section 2.2 we have examined the construction and evaluation of single class, separable queueing network models. Separable models have the following desirable characteristics: efficiency of evaluation, accuracy of results, direct correspondence with computer systems, generality.

In section 2.3 we have focused on multiple class, separable queueing network models. We are interested in separable networks because they are reasonably accurate models of computer systems and can be solved efficiently; more general models require excessively large amounts of time and space. Exact solutions of separable models with a few customer classes, and accurate approximate solutions of models with many customer classes, can be obtained with modest machine resources.

The major advantage of multiple class models over single class models is also the main drawback. By identifying distinct workload components, output performance measures for each can be given separately. At the same time, input parameter values are required for each' individual class.

2.1 Bounding Analysis on Queueing Network Models

2.1.1 Introduction

We begin with the simplest useful approach to computer system analysis using queueing network models: bounding analysis. With very little computation determining upper and lower bounds on system throughput and response time as functions of the system workload intensity is possible. We describe techniques to compute two classes of performance bounds: asymptotic bounds and balanced system bounds. Asymptotic bounds hold for a wider class of systems than do balanced system bounds. They also are simpler to compute. The offsetting advantage of balanced system bounds is that they are tighter, and thus provide more precise information than asymptotic bounds.

There are several characteristics of bounding techniques. The development of these techniques provides valuable insight into the primary factors affecting the performance of computer systems (Muntz & Wong 1974). The bounds can be computed quickly, even by hand. Bounding analysis therefore is suitable as a first cut modelling technique that can be used to eliminate inadequate alternatives at an early stage of a study. Often, a number of alternatives can be treated together, with a single bounding analysis providing useful information about them all.

Bounding techniques are most useful in system sizing studies. Such studies involve long-range planning, and consequently often are based on preliminary estimates of system characteristics. With such imprecision in knowledge of the system, quick bounding studies may be more appropriate than more detailed analyses leading to specific estimates of performance measures. System sizing studies typically involve consideration of a large number of candidate configurations. Often a single resource (such as the CPU) is the dominant concern, because the remainder of the system can be configured to match the power of this resource. Bounding analysis permits considering as one alternative a group of candidate configurations that have the same critical resource but differ with respect to the pattern of demands at the other service centers. Bounding techniques also can be used to estimate the potential performance gain of alternative upgrades to existing systems.

The discussion of bounding analysis is restricted to the single class case. Multiple class generalizations exist, but they are not used widely. One reason for this is those bounding techniques are most useful for capacity studies of the bottleneck center, for which single class models suffice (Denning & Kahn 1975). Additionally, a major

attraction of bounding techniques in practice is their simplicity, which would be lost if multiple classes were included in the models.

The models can be described by the following parameters:

- K, the number of service centers;
- D_{max}, the largest service demand at any single center;
- D, the sum of the service demands at the centers;
- the type of the customer class (batch, terminal, or transaction);
- Z, the average think time (if the class is of terminal type).

For models with transaction type workloads, the throughput bounds show the maximum customer arrival rate that can be processed by the system. The response time bounds reflect the largest and smallest possible response times that these customers could experience as a function of the system arrival rate. For models with batch or terminal type workloads, the bounds show the maximum and minimum possible system throughputs and response times as functions of the number of customers in the system. Throughput upper and response time lower bounds are cited as optimistic bounds and throughput lower and response time upper bounds are cited as pessimistic bounds.

2.1.2 Asymptotic Bounds

Asymptotic bounding analysis provides optimistic and pessimistic bounds on system throughput and response time in single class queueing networks. As their name suggests, they are derived by considering the (asymptotically) extreme conditions of light and heavy loads. The validity of the bounds depends on only a single assumption: that the service demand of a customer at a center does not depend on how many other customers currently are in the system, or at which service centers they are located. The type of information provided by asymptotic bounds depends on whether the system workload is open (transaction type) or closed (batch or terminal type).

2.1.2.1. Transaction Workloads

For transaction workloads, the throughput bound shows the maximum possible arrival rate of customers that the system can process successfully. If the arrival rate exceeds this bound, a backlog of unprocessed customers grows continually as jobs arrive. Thus, in the long run, an arriving job has to wait an indefinitely long. Here we say that the system is saturated. The throughput bound thus is the arrival rate that separates feasible

processing from saturation. The key to determining the throughput bound is the utilization law:

$U_k = X_k S_k$ for each center k. If we denote the arrival rate to the system as λ, then $X_k = \lambda V_k$, and the utilization law can be rewritten as $U_k = \lambda D_k$, where D_k is the service demand at center k. To derive the throughput bound, we simply note that as long as all centers have unused capacity, an increased arrival rate can be accommodated. However, when any of the centers becomes saturated, the entire system becomes saturated, since no increase in the arrival rate of customers can be handled successfully. Thus, the throughput bound is the smallest arrival rate λ_{sat} at which any center saturates. Clearly, the center that saturates at the lowest arrival rate is the bottleneck center - the center with the largest service demand. Let max be the index of the bottleneck center. Then:

$$U_{\max}(\lambda) = \lambda D_{\max} \leq 1$$

so:

$$\lambda_{sat} = \frac{1}{D_{\max}}$$

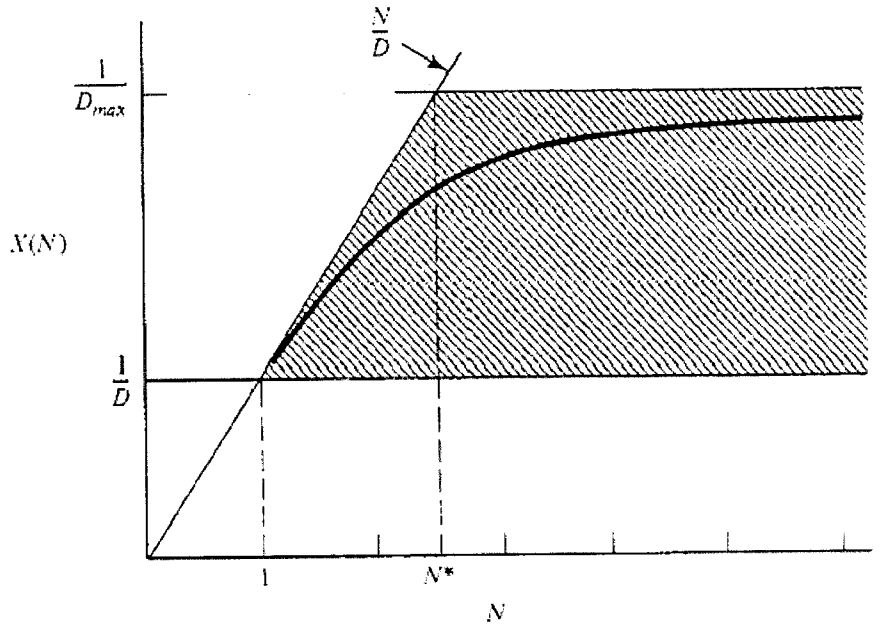
Thus, for arrival rates greater than or equal to $1/D_{max}$ the system is saturated, while the system is capable of processing arrival rates less than $1/D_{max}$.

Asymptotic response time bounds indicate the largest and smallest possible response times experienced by customers when the system arrival rate is h. Because the system is unstable if $\lambda > \lambda_{sat}$ we limit our investigation to the case where the arrival rate is less than the throughput bound. There are two extreme situations. In the best possible case, no customer ever interferes with any other, so that no queueing delays are experienced. In that case the system response time of each customer is simply the sum of its service demands, which we denote by D. In the worst possible case, n customers arrive together every n/λ time n units (the system arrival rate is $\frac{n}{n/\lambda} = \lambda$). Customers at the end of the batch are forced to queue for customers at the front of the batch, and thus experience large response times. As the batch size n increases, more and more customers are waiting an increasingly long time. Thus, for any postulated pessimistic bound on response times for system arrival rate h, it is possible to pick a batch size n sufficiently large that the bound is exceeded. We conclude that there is no pessimistic bound on response times, regardless of how small the arrival rate λ might be.

2.1.2.2. Batch and Terminal Workloads

Figures 2.1.la and 2.1.lb show the general form of the asymptotic bounds on throughput and response time for batch and terminal workloads, respectively. The bounds indicate that the precise values of the actual throughputs and response times must lie in the shaded portions of the figures. The general shapes and positions of these values are indicated by the curves in the figures.

Batch Throughput:



Batch Response Time:

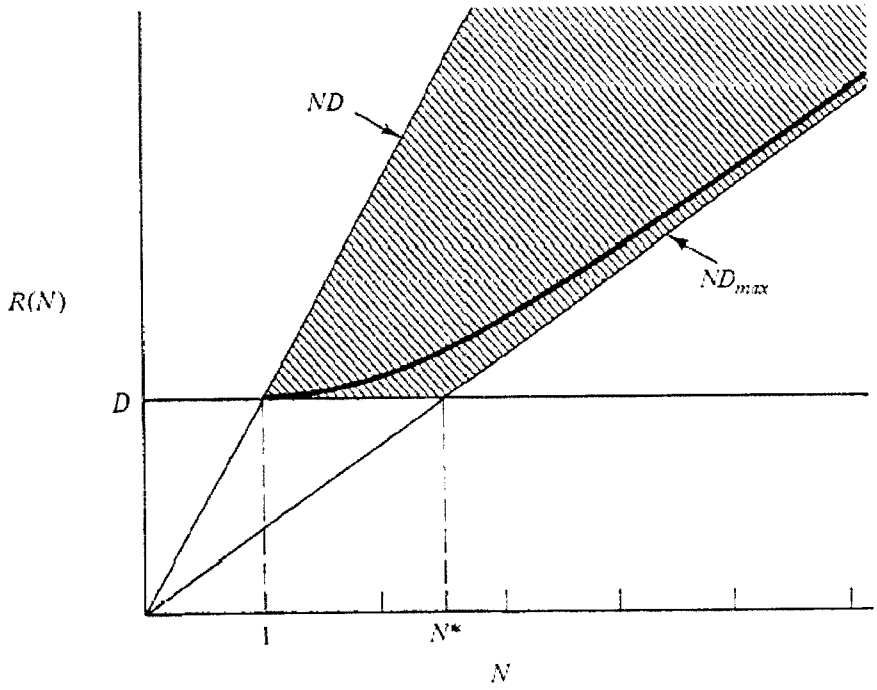
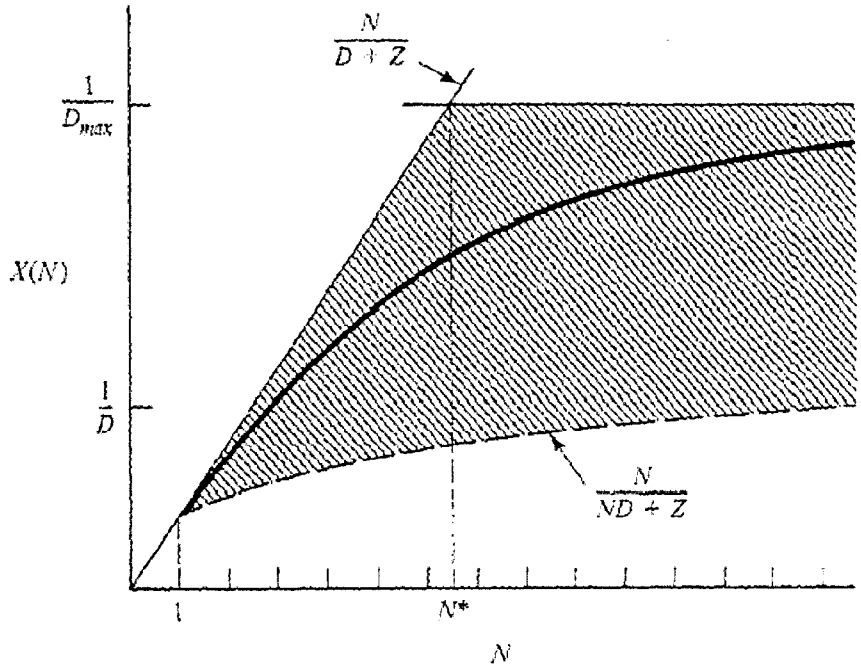


Figure 2.1.la - Asymptotic Bounds on Performance (Beizer 1978)

Terminal Throughput:



Terminal Response Time:

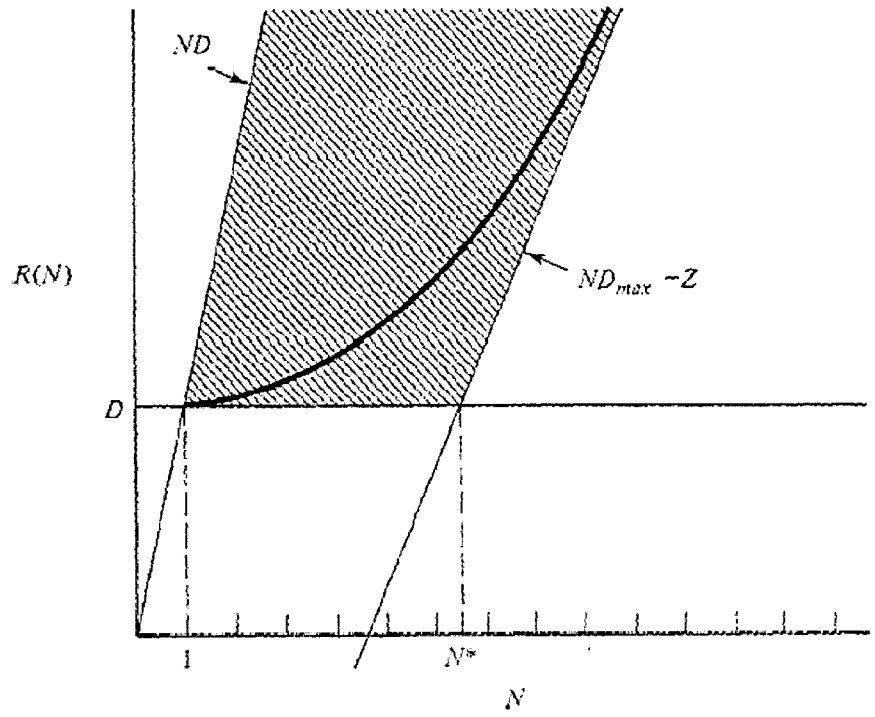


Figure 2.1.lb - Asymptotic Bounds on Performance (Beizer 1978)

To derive the bounds shown in the figures, we first consider the bounds on throughput, and then use Little's law to transform them into corresponding bounds on response time. By taking the think time, Z, to be zero, we obtain results for batch workloads.

We begin with the heavy load (many customer) situation. As the number of customers in the system (N) becomes large, the utilizations of all centers grow, but clearly no utilization can exceed one. From the utilization law we have for each center k that:

$$U_k(N) = X(N)\, D_k \leq 1$$

Each center limits the maximum possible throughput that the system can achieve. Since the bottleneck center (max) is the first to saturate, it restricts system throughput most severely. We conclude that:

$$X(N) \leq 1/D_{max}$$

Intuitively this is clear, because if each customer requires on average D_{max} time units of service at the bottleneck center, then in the long run customers certainly cannot be completed any faster than one every D_{max} time units.

Next consider the light load (few customers) situation. At the extreme, a single customer alone in the system attains a throughput of $1 / (D+ Z)$, since each K interaction consists of a period of service (of average length $D = \sum_{k=1}^{k} D_k$) and a think time (of average length Z).

As more customers are added to the system there are two bounding situations:

- The smallest possible throughput occurs when each additional customer is forced to queue behind all other customers already in the system. In this case, with N customers in the system, $(N- 1)D$ time units are spent queued behind other customers, D time units are spent in service, and Z time units are spent thinking, so that the throughput of each customer is 1/(ND + Z). Thus, system throughput is $N/(ND + Z)$.
- The largest possible throughput occurs when each additional customer is not delayed at all by any other customers in the system. In this case no time is spent queueing, D time units are spent in service, and Z time units are spent thinking. Thus, the throughput of each customer is $1/(D+Z)$, and system throughput is $N/(D+Z)$.

The above observations can be summarized as the asymptotic bounds on system throughput:

$$\frac{N}{ND+Z} \leq X(N) \leq \min\left(\frac{1}{D_{max}}, \frac{N}{D+Z}\right)$$

Note that the optimistic bound consists of two components, the first of which applies under heavy load and the second of which applies under light load. As illustrated by Figure 2.1.1, there is a particular population size N* such that for all N less than N* the light load optimistic bound applies, while for all N larger than N* the heavy load bound applies.

This crossover point occurs where the values of the two bounds are equal:

$$N^* = \frac{D+Z}{D_{max}}$$

We can obtain bounds on response time R(N) by transforming our throughput bounds using Little's law. We begin by rewriting the previous equation:

$$\frac{N}{ND+Z} \le \frac{N}{R(N)+Z} \le \min(\frac{1}{D_{max}}, \frac{N}{D+Z})$$

Inverting each component to express the bounds on R (N) yields:

$$\max(D_{max}, \frac{D+Z}{N} - Z) \le \frac{R(N)+Z}{N} \le \frac{ND+Z}{N}$$

or:

$$\max(D, ND_{max} - Z) \le R(N) \le ND$$

2.1.3. Using Asymptotic Bounds

In this section we present three applications of asymptotic bounds: an assessment of the effect of alleviating a bottleneck.

	workload type	bound
X	batch	$\frac{1}{D} \le X(N) \le \min(\frac{N}{D}, \frac{1}{D_{max}})$
	terminal	$\frac{N}{ND+Z} \le X(N) \le \min(\frac{N}{D+Z}, \frac{1}{D_{max}})$
	transaction	$X(\lambda) \le 1/D_{max}$
R	batch	$\max(D, ND_{max}) \le R(N) \le ND$
	terminal	$\max(D, ND_{max} - Z) \le R(N) \le ND$
	transaction	$D \le R(\lambda)$

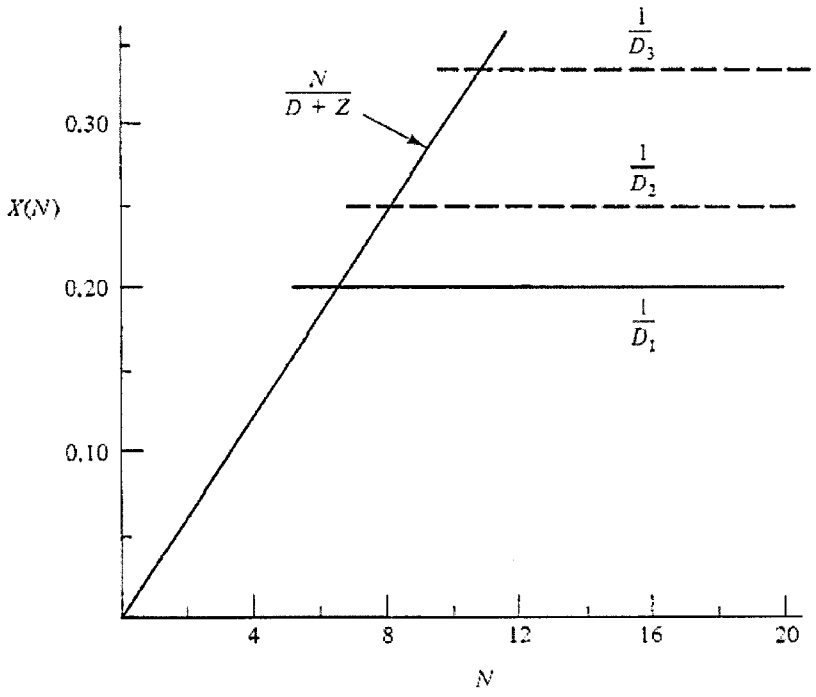
Table 2.1.1 - Summary of Asymptotic Bounds

2.1.3.1. Effect of Bottleneck Removal

So far we have been most concerned with the bottleneck center, which constrains throughput to be at most $1/D_{max}$. What happens if we alleviate that bottleneck, either by replacing the device with a faster one or by shifting some of the work to another device? In either case, D_{max} is reduced and so the throughput optimistic bound, $1/D_{max}$, increases. A limit to the extent of this improvement is imposed by the center with the second highest service demand originally. We call this center the secondary bottleneck, as contrasted with the primary bottleneck (Beizer 1978).

An important lesson to be learned is the futility of improving any center but the bottleneck with respect to enhancing performance at heavy load. Reducing the service demand at centers other than the bottleneck improves only the light load asymptote, and the improvement usually is insignificant. Figure 2.1.2b compares the effects on the asymptotic bounds of independently doubling the speed (halving the service demand) at the primary and secondary bottlenecks for this example system. Observe that, at heavy load, performance gains only are evident when the demand at the primary bottleneck is reduced.

Throughput:



Response Time:

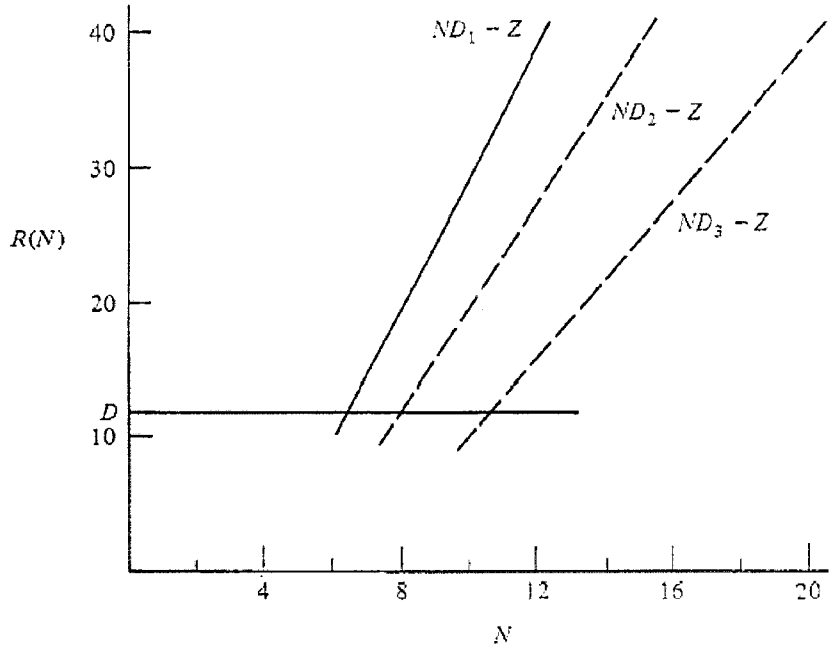


Figure 2.1.2a - Secondary and Tertiary Asymptotic Bounds (Beizer 1978)

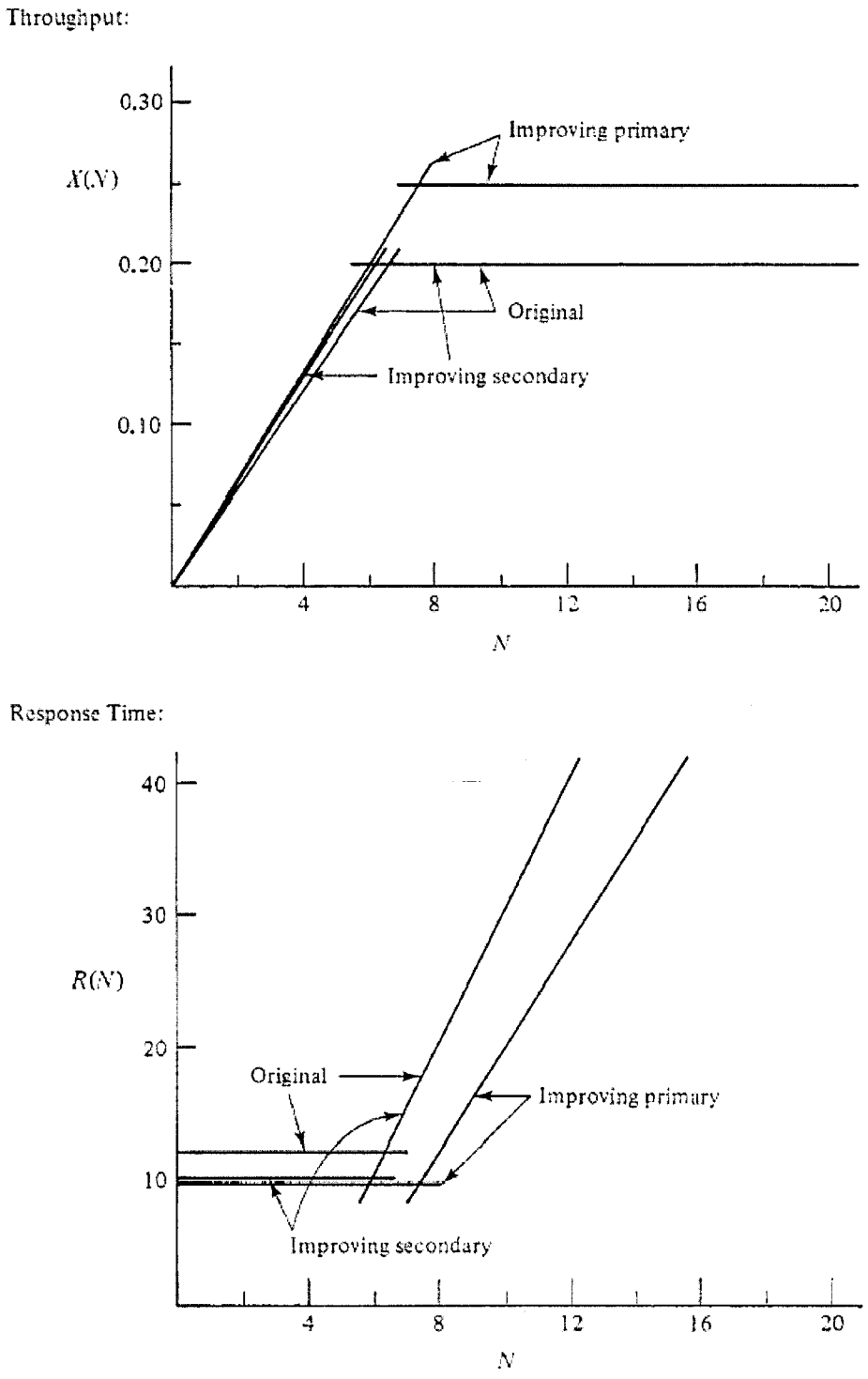


Figure 2.1.2 b - Relative Effects of Reducing Various Service Demands (Beizer 1978)

2.1.4. Balanced System Bounds

With a modest amount of computation beyond that required for asymptotic bounds, tighter bounds can be obtained (Zahorjan et al. 1982). These bounds are called balanced system bounds because they are based upon systems that are "balanced" in the

sense that the service demand at every center is the same, i.e., $D_1=D_2=D_3= \ldots =D_k$. Figures 2.1.3a and 2.1.3b show the general form of balanced system bounds (together with the asymptotic bounds) for batch (2.1.3a) and terminal (2.1.3b) workloads.

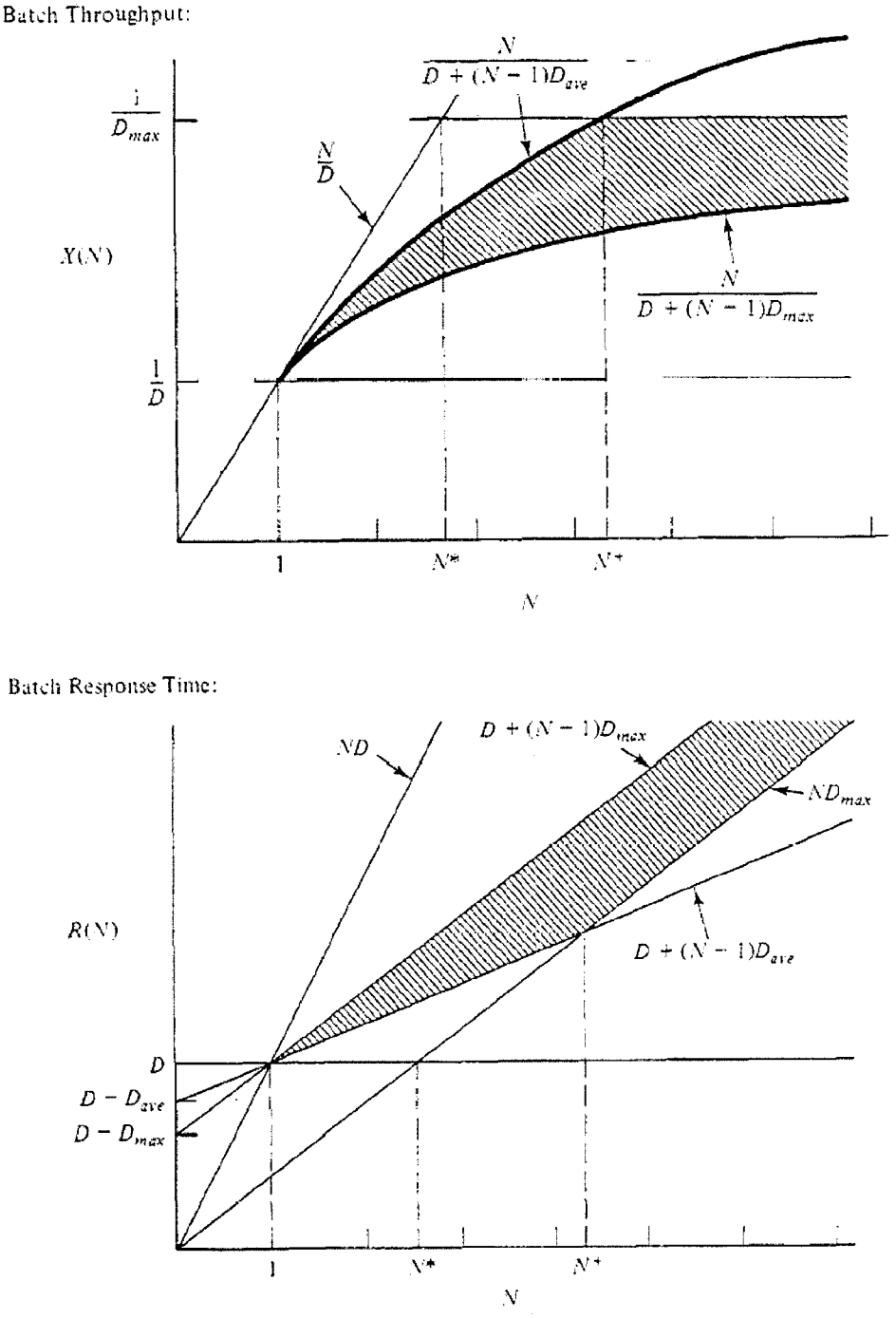


Figure 2.1.3a - Balanced System Bounds on Performance (Beizer 1978)

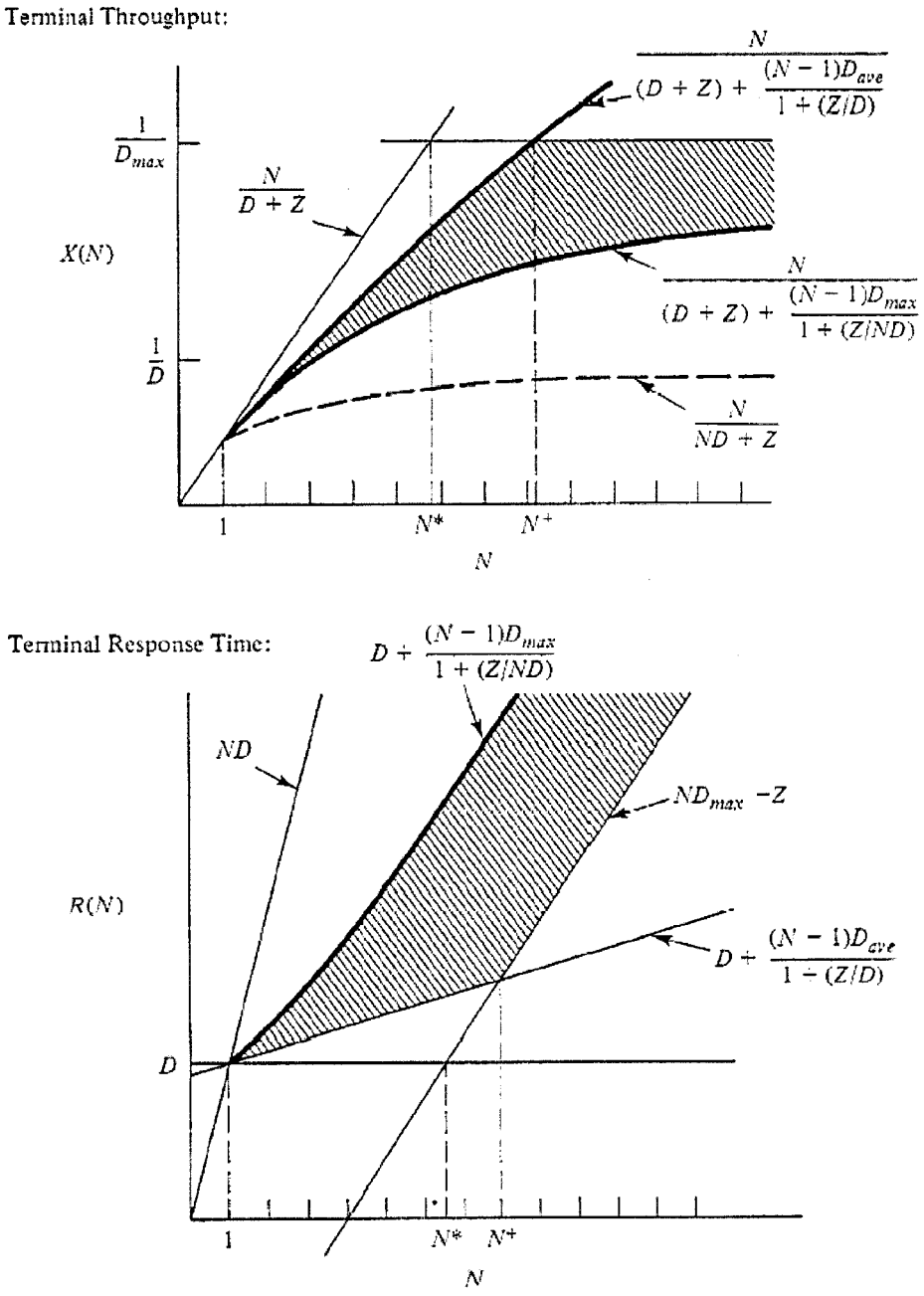


Figure 2.1.3b - Balanced System Bounds on Performance (Beizer 1978)

Formally, this analysis requires that various assumptions be made about the system being modelled. This is in contrast to asymptotic bounds, which require only that the service demand of a customer at a center does not depend on how many other customers are currently in the system or at which centers they are located.

For balanced systems, the techniques to be presented have a particularly simple form. The utilization of every service center is given by:

$$U_k(N) = \frac{N}{N+K-1}$$

By the utilization law, system throughput is then:

$$X(N) = \frac{U_k}{D_k} = \frac{N}{N+K-1} \times \frac{1}{D_k}$$

where D_k is the service demand at every center.

Let D_{max} , D_{ave}, , and D_{min}, denote respectively the maximum, average, and minimum of the service demands at the centers of the model we wish to evaluate. We bound the throughput of that system by the throughputs of two related balanced systems: one with service demand D_{min} at every center, and the other with service demand D_{max} at every center:

$$\frac{N}{N+K-1} \times \frac{1}{D_{\max}} \le X(N) \le \frac{N}{N+K-1} \times \frac{1}{D_{\max}}$$

These inequalities hold because, of all systems with K centers, N customers, and maximum service demand D_{max}, the one with the lowest throughput is the balanced system with demand D_{max} at each center. Similarly, of all systems with K centers, N customers, and minimum demand D_{min}, the one with the highest throughput is the balanced system with demand D_{min} at each center. Corresponding bounds on average response times are:

$$(N+K-1)D_{\min} \le D_{\min} \le (N+K-1)D_{\max}$$

Tighter balanced system bounds can be obtained by constraining not only the maximum service demand, D_{max} but also the total demand, D. Of all systems with a given total service demand $D = \sum_{k=1}^{k} D_k$, the one with the highest throughput is the one in which all service demands are equal (i.e., $D_k = D/K$, $k = 1, \ldots, K$). This confirms our intuition that the increase in delay resulting from an increase in load is greater than the decrease in delay resulting from an equivalent decrease in load. Therefore, optimistic bounds are given by:

$$X(N) \le \frac{N}{N+K-1} \times \frac{1}{d_{AVE}} = \frac{N}{D+(N-1)D_{ave}}$$

and:

$$D+(N-1)D_{ave} \le R(N)$$

Note that the optimistic balanced system bound intersects the heavy load component of the optimistic asymptotic bound (at a point that we will denote by N^*). Beyond this point, the balanced system bound is defined to coincide with the asymptotic bound.

Analogously, of all systems with total demand D and maximum demand D_{max}, the one with the lowest throughput has D/D_{max} centers with demand D_{max} and zero demand at the remaining centers. (The fact that D/D_{max} may not be an integer hampers intuition, but not the validity of the bounds.) Therefore, pessimistic bounds are:

$$\frac{N}{N+\frac{D}{D_{\max}}-1}\times\frac{1}{D_{\max}}=\frac{N}{D+(N-1)D_{\max}}\le X(N)$$

and:

$$R(N)\le D+(N-1)D_{\max}$$

	workload type	bound
X	batch	$\frac{N}{D+(N-1)D_{\max}}\le X(N)\le \min(\frac{1}{D_{\max}},\frac{N}{D+(N-1)D_{\max}})$
	terminal	$\frac{N}{D+Z+\frac{(N-1)D_{\max}}{1+Z/(ND)}}\le X(N)\le \min(\frac{1}{D_{\max}},\frac{N}{D+Z+\frac{(N-1)D_{\max}}{1+Z/(ND)}})$
	transaction	$X(\lambda)\le 1/D_{\max}$
R	batch	$\max(ND_{\max}, D+(N-1)D_{\max})\le R(N)\le D+(N-1)D_{max}$
	terminal	$\max(ND_{\max}-Z, D+\frac{(N-1)D_{\max}}{1+Z/D})\le R(N)\le D+\frac{(N-1)D_{\max}}{1+Z/(ND)}$
	transaction	$\frac{D}{1-\lambda D_{\max}}\le R(\lambda)\le \frac{D}{1-\lambda D_{\max}}$

Table 2.1.2 - Summary of Balanced System Bounds

2.2. Models with One Job Class

2.2.1. Introduction

Single class models are refinements of bounding models that provide estimates of performance measures, rather than simply bounds. In single class models, the customers are assumed to be indistinguishable from one another. Although single class models always are simplifications, they nonetheless can be accurate representations of real systems. There are a number of situations. The first one is increase information. The results of a bounding study might not provide sufficiently detailed information. Single class models are the next step in a progression of increasingly detailed models. The

second one is single workload of interest. The computer system under consideration may be running only a single workload of significance to its performance. Therefore, it may not be necessary to represent explicitly the other workload components such as homogeneous workloads (Jackson 1963). The various workload components of a computer system may have similar service demands. A reasonable modelling abstraction is to consider them all to belong to a single customer class.

Conversely, there are a number of situations in which it might be inappropriate to model a computer system workload by a single customer class. These situations typically arise either because distinct workload components exhibit markedly differing resource usage, or because the aim of the modelling study requires that inputs or outputs be specified in terms of the individual workload components rather than in terms of the aggregate workload. Typical instances of each are: multiple distinct workloads, class dependent model inputs, class dependent model outputs (Gordon & Newell 1967). Thus a multiple class model is required. Systems having workloads with substantially differing characteristics, may be modelled more reasonably by multiple class than by single class queueing networks.

Establishing the workload intensity has two aspects: selecting an appropriate workload type (transaction, batch, or terminal), and setting the appropriate workload intensity parameter(s) for that type. Selecting an appropriate workload type typically is straightforward, since the three workload types of queueing network models correspond directly to the three predominant workload types of computer systems. One technical distinction that arises is that between open models and closed model. Since the number of customers that may be in an open model at any time is unbounded, while the number of customers that may be in a closed model is bounded by the population of the closed class, the response times of open models tend to be larger than those of corresponding closed models with the same system throughput. This occurs because in open models the potential for extremely large queue lengths exists, while in closed models, because of the finite population, it does not. This difference usually is significant only when some device in the system is near saturation.

In queueing network models, the workload intensity is a fixed. In contrast, in a computer system the workload intensity may vary. Despite this discrepancy, queueing network models are useful in a wide variety of situations:

- heavy load assumption - It may be interesting to study the behavior of a system under the maximum possible load. By hypothesis, the load is sufficiently heavy that there always are jobs waiting to enter memory.
- non-integer workload intensity - The measurement data for a system might show that the average multiprogramming level is not an integer. Some algorithms for evaluating queueing network models allow non-integer customer populations.
- workload intensity distribution - Measurement data might provide a distribution of observed workload intensities, e.g., proportions of time P[N=n] that there were n active terminal users on the system
- sizing studies - Because the solutions of single class models can be obtained extremely quickly, it is feasible to evaluate a model for a large number of workload intensities.
- robustness studies - Similarly, since it often is the case that workload growth cannot be forecast accurately, it generally is useful to evaluate a model for a range of workload intensities surrounding the expected one.

The base CPU service requirement per job was estimated by dividing the total CPU busy time over the measurement interval by the number of jobs that completed in the interval. Thus, system CPU overhead was allocated equally among all jobs. Parameterizing the I/O subsystem was more complicated. In the model, the channel service demands were set to the sum of the average latency and data transfer times, while the disk service demands were set to the average seek time. Thus, all components of I/O service time were represented exactly once.

Measurement of the system showed that the average multiprogramming level varied significantly during the measurement interval. To account for this variability, the model was evaluated once for each observed multiprogramming level. Performance projections were obtained by weighting the distinct solutions by the percentage of time the corresponding multiprogramming levels were observed in the system.

2.2.2. Solution Techniques

The solution of a queueing network model is a set of performance measures that describe the time averaged (or long term) behavior of the model. Computing these measures for general networks of queues is quite expensive and complicated (Buzen 1973). However, for separable queueing network models, solutions can be obtained simply.

2.2.2.1. Open Model Solution Technique

For open models (those with transaction workloads), one of the key output measures, system throughput, is given as an input. Because of this, the solution technique for these models is especially simple. We list here the formulae that apply for each performance measure of interest.

- processing capacity

  The processing capacity of an open model, λ_{sat} is the arrival rate at which it saturates. This is given by:

$$\lambda_{sat} = \frac{1}{\max_k D_k} = \frac{1}{D_{\max}}$$

  In the derivations that follow, we assume that $\lambda < \lambda_{sat}$.

- throughput

  By the forced flow law, if λ customers/second enter the network, then the system output rate must also be λ customers/second. Similarly, if each customer requires on average V_k visits to device k, the throughput at device k must be λV_k, visits/second. Thus:

$$X_k(\lambda) = \lambda V_k$$

- utilization

  By the utilization law, device utilization is equal to throughput multiplied by service time. Thus:

$$U_k(\lambda) = X_k(\lambda) S_k = \lambda D_k$$

  (In the case of delay centers, the utilization must be interpreted as the average number of customers present.)

- residence time

  The residence time at center k, $R_k(\lambda)$, is the total time spent at that center by a customer, both queueing and receiving service. For service centers of delay type, there is no queueing component, so $R_k(\lambda)$ is simply the service time multiplied by the number of visits:

  $R_k(\lambda) = V_k S_k = D_k$ (delay centers)

  For queueing centers, R_k is the sum of the total time spent in service and the total time spent waiting for other customers to complete service. The former component is $V_k S_k$ .The latter component is the time spent waiting for customers already in the queue

when a customer arrives. Letting $A_k(\lambda)$ designate the average number of customers in queue as seen by an arriving customer, the queueing component is $V_k \{A_k(\lambda)\ S_k\}$. Thus, for queueing centers the residence time is given by:

$$R_k(\lambda) = V_k[S_k + S_k A_k(\lambda)]$$
$$= D_k[1 + A_k(\lambda)]$$

An implication of the assumptions made in constructing separable networks is that the queue length seen upon arrival at center k, $A_k(\lambda)$ is equal to the time averaged queue length $Q_k(\lambda)$ giving:

$$R_k(\lambda) = D_k[1 + Q_k(\lambda)]$$

which, using Little's law to re-express Q_k, is:

$$R_k(\lambda) = D_k[1 + \lambda R_k(\lambda)]$$
$$= \frac{D_k}{1 - U_k(\lambda)} \quad \text{(queueing centers)}$$

This equation exhibits the intuitively appealing property that as

$$U_k(\lambda) \to 0, R_k(\lambda) \to D_k, \textit{and as} \quad U_k(\lambda) \to 1, R_k(\lambda) \to \infty$$

- queue length

  By Little's law:

$$Q_k(\lambda) = \lambda R_k(\lambda)$$
$$= \begin{cases} U_k & (\textit{delay centers}) \\ \dfrac{U_k(\lambda)}{1 - U_k(\lambda)} & (\textit{queueing centers}) \end{cases}$$

- system response time

  System response time is the sum of the residence times at all service centers :

$$R_k(\lambda) = \sum_{k=1}^{K} R_k(\lambda)$$

- average number in system

  The average number in system can be calculated using Little's law, or by summing the queue lengths at all centers:

$$Q(\lambda) = \lambda R(\lambda) = \sum_{k=1}^{K} Q_k(\lambda)$$

2.2.2.2. Closed Model Solution Techniques

The technique we use to evaluate closed queueing networks (those with terminal or batch classes) is known as mean value analysis (MVA). It is based on three equations:

- Little's law applied to the queueing network as a whole :

$$X(N) = \frac{N}{Z + \sum_{k=1}^{K} R_k(N)} \qquad (2.2.1)$$

where X(N) is the system throughput and R_k (N) the residence time at center k, when there are N customers in the network. (As usual, if the customer class is batch type. we take Z = 0.) Note that system throughput can be computed from input parameter data if the device residence times R_k (N) are known.

- Little's law applied to the service centers individually :

$$Q_k(N) = X(N)R_k(N) \qquad (2.2.2)$$

Once again, the residence times must be known before Little's law can be applied to compute queue lengths.

- The service center residence time equations :

$$R_k(N) = \begin{cases} D_k & (delay\ centers) \\ D_k[1 + A_k(N)] & (queueing\ centers) \end{cases} \qquad (2.2.3)$$

where A_k(N) is the average number of customers seen at center k when a new customer arrives.

Note that, as with open networks, the key to computing performance measures for closed networks is the set of A_k(N). If these were known, the R_k(N) could be computed, followed by X(N) and the Q_k(N). In the case of open networks we were able to substitute the time averaged queue lengths, Q_k(N), for the arrival instant queue lengths, A_k(N). In the case of closed networks, this substitution is not possible. To see that A_k(N) does not equal Q_k(N) in closed networks, consider the network consisting of two queueing service centers and a single customer with a service demand of 1 second at each center. Since there is only one customer, the time averaged queue lengths at the service centers are simply their utilizations, so Qi(l) = Q,(l) = l/2. However, the arrival instant queue lengths Al(l) and A,(l) both are zero, because with a single customer in the network no customers could possibly be in queue ahead of an arriving customer. In general, the key distinction is that the arrival instant queue lengths are computed conditioned on the fact that some

customer is arriving to the center, while the time averaged queue lengths are computed over randomly selected moments.

2.2.2.3. Exact Solution Technique

The exact MVA solution technique is important for two reasons:

- It is the basis from which the approximate technique is derived.
- There are no known bounds on the inaccuracy of the approximate technique. While typically it is accurate to within a few percent relative to the true solution, it cannot be guaranteed that in any particular situation the results will not be worse.

The exact solution technique involves computing the arrival instant queue lengths Ak (N) exactly (Reiser & Lavenberg 1980), then applying equations (2.2.1)-(2.2.3). The characteristic of closed, separable networks that makes them amenable to this approach is that the $A_k(N)$ have a particularly simple form:

$$A_k(N) = Q_k(N-1) \qquad (2.2.4)$$

In other words, the queue length seen at arrival to a queue when there are N customers in the network is equal to the time averaged queue length there with one less customer in the network. This equation has an intuitive justification. At the moment a customer arrives at a center, it is certain that this customer itself is not already in queue there. Thus, there are only N- 1 other customers that could possibly interfere with the new arrival. The number of these that actually are in queue, on average, is simply the average number there when only those N- 1 customers are in the network.

2.2.2.4. Approximate Solution Technique

The key to the exact MVA solution technique is equation (2.2.4), which computes the arrival instant queue length for population n based on the time averaged queue length with population n- 1 (Sevcik & Mitrani 1981). The nature of the algorithm is a direct consequence of this relationship. By replacing equation (2.2.4) with an approximation:

$$A_k(N) \approx h[Q_k(N)]$$

for some suitable function h, a more efficient, iterative algorithm can be obtained. For instance, the approximation we will propose shortly also depends on N. However, we use this notation for simplicity, and to suggest that the key requirement is the value of Q_k (N).) The accuracy of the algorithm depends, of course, on the accuracy of the function h that is used. Crucial to this faster solution technique is the function h. Unfortunately, no function h is known that is exact for all separable networks. Instead, an approximation must be used. A particularly simple and reasonably accurate approximation is:

$$A_k(N) = Q_k(N-1) \approx h[Q_k(N)] \equiv \frac{N-1}{N} Q_k(N) \quad (2.2.5)$$

Equation (2.2.5) estimates the arrival instant queue length by approximating its exact value, the queue length with one fewer customer. This $Q_k(N)/N$ and approximation is based on the assumption that the ratios Q_k (N-1)/(N-1) are equal for all k, i.e., that the amount that each queue length is diminished by the removal of a single customer is equal to the amount that customer contributes to the queue length. In general, this assumption is quite accurate. In particular, it is asymptotically correct for very large N, and trivially correct for models with only a single customer. Thus, the approximation is guaranteed to be good at the two extremes. Experience with the technique has demonstrated that it also gives remarkably good results for intermediate populations, Since this error is well within the bounds of other discrepancies inherent in the computer system analysis process, the approximate MVA technique is satisfactory as a general solution technique.

2.2.2.5. Theoretical Foundations

Separable queueing network models are a subset of the general class of queueing network models obtained by imposing restrictions on the behavior of the service centers and customers (Chandy & Neuse 1982). The name "separable" comes from the fact that each service center can be separated from the rest of the network, and its solution evaluated in isolation. The solution of the entire network then can be formed by combining these separate solutions. In an intuitive sense, a separable network has the property that each service center acts (largely) independently of the others.

There are five assumptions about the behavior of a model that, if satisfied, guarantee that the model is separable. These are:

- service center flow balance - Service center flow balance is the extension of the flow balance assumption to each individual service center: the number of arrivals at each center is equal to the number of completions there.
- one step behavior - One step behavior asserts that no two jobs in the system "change state" at exactly the same time. Real systems almost certainly display one step behavior.

The remaining three assumptions are called homogeneity assumptions. This name is derived from the fact that in each case the assumption is that some quantity is the same regardless of the current locations of some or all of the customers in the network.

- routing homogeneity - To this point we have characterized the behavior of customers in the model simply by their service demands. A more detailed characterization would include the routing patterns of the jobs, that is, the patterns of centers visited.
- device homogeneity - The rate of completions of jobs from a service center may vary with the number of jobs at that center, but otherwise may not depend on the number or placement of customers within the network.
- homogeneous external arrivals - The times at which arrivals from outside the network occur may not depend on the number or placement of customers within the network.

These assumptions are sufficient for the network to be separable, and thus to be evaluated efficiently. However, the specific solution algorithms we have presented thus far require one additional assumption, which is a stronger form of the device homogeneity assumption:

- service time homogeneity - The rate of completions of jobs from a service center, while it is busy, must be independent of the number of customers at that center, in addition to being independent of the number or placement of customers within the network.

The weaker of the two assumptions, device homogeneity, permits the rate of completions of jobs from a center to vary with the queue length there. Centers with this characteristic are called load dependent centers. A delay center is a simple example of a load dependent center, since the rate of completions increases in proportion to the number of customers at the center. Service time homogeneity asserts that the rate of completions is independent of the queue length. Centers with this characteristic are called load independent. The queueing centers we have described so far are examples of load independent centers. The particular versions of the MVA algorithms presented in this chapter are applicable only to networks consisting entirely of load independent and delay centers.

2.3. Models with Multiple Job Classes

2.3.1. Introduction

Multiple class models, like single class models, provide estimates for performance measures such as utilization, throughput, and response time. There are the advantages of multiple class models over single class models. Outputs are given in terms of the individual customer classes. For example, in modelling a transaction processing system, response times for each of a number of transaction types could be obtained by including each type as a separate class. With a single class model, only a single estimate for response time representing the average over all transaction types could be obtained (Baskett et al. 1975). For systems in which the jobs being modelled have significantly different behaviors, such as systems with a mixture of CPU and I/O bound jobs, a multiple class model can provide more accurate results. This means that some systems can be modelled adequately only by multiple class models, since the single class assumption that jobs are indistinguishable is unacceptable.

There are the disadvantages of multiple class models relative to single class models. Since there are multiple customer classes in the model, multiple sets of input parameters are required. The data gathering portion of the modelling process therefore is more tedious. Most current measurement tools do not provide sufficient information to determine the input parameters appropriate to each customer class with the same accuracy as can be done for single class models. Multiple class solution techniques are somewhat more difficult to implement, and require more machine resources, than single class techniques.

For the most part, these disadvantages result from inadequate modeling support software, and thus should become less significant as queueing network modelling becomes more widespread. The first two disadvantages can be eliminated by measurement tools that are designed with knowledge of the information required to establish a model. The third disadvantage is significant only if one is developing queueing network modelling software. Commercially available software packages are capable of evaluating multiple class models. Thus, once the model inputs have been obtained, it is no more difficult to deal with a multiple class model than with a single class model.

2.3.2. Workload Representation

The major additional consideration is the specification of scheduling disciplines. Since customers in single class models are indistinguishable, the scheduling disciplines at the various service centers are characterized entirely as being either delay or queueing (Chandy et al. 1977). However, in multiple class models, customers are distinguishable, and so the choice of scheduling discipline can be important.

There are a large number of scheduling disciplines that can be represented in (separable) multiple class queueing network models. For practical purposes, however, the following disciplines have proven to be sufficient:

- First-come-first-served (FCFS) - Under FCFS scheduling, customers are served in the order in which they arrive. Although this is the simplest of scheduling disciplines to implement, it is difficult to model analytically. To do so, it is necessary to impose the restriction that all customer classes have the same service requirement at each visit to the service center in question ($S_{c,k}$). It is possible, however, for different customer classes to require different total numbers of visits to the service center ($V_{c,k}$), thus providing for distinct service demands there ($D_{c,k}$). A FCFS center might be appropriate to represent a disk containing user files for a number of classes. Since the basic operations performed at the device by the various classes are the same, it is reasonable to assume that the average service times across classes are nearly equal. The actual number of file accesses for a customer of each class can be represented in the model by appropriate values of the $V_{c,k}$ for each class c.
- processor sharing (PS) - Processor sharing is an idealization of round robin (RR) scheduling. Under RR, control of the processor circulates among all jobs in the queue. Each job receives a quantum of service before it must relinquish control to the next job in the queue, rejoining the queue at its tail. Under PS, the length of the quantum is effectively zero, so that control of the processor circulates infinitely rapidly among all jobs. The effect is that jobs are served simultaneously, but each of the n jobs in service receives only 1/n-th of the full power of the processor.
- last-come-first-served preemptive-resume (LCFS) - Under this discipline an arriving job preempts the job in service (if any) and immediately begins service itself. When a job completion occurs, the most recently preempted job resumes service at the point at which it was interrupted. LCFS might be used to model a CPU in a system where the frequency with which high priority system tasks are dispatched is high enough that LCFS is a reasonable approximation.

- delay - As in single class models, multiple class delay centers are used to represent devices at which residence time consists entirely of service (there is no queueing delay).

Although the first three disciplines seem quite different, the performance measures obtained from a model will be the same regardless of which is used.

2.3.3. Solution Techniques

The solution techniques for multiple class models yield values for performance measures such as utilization, throughput, response time, and queue length, for each individual customer class (Lindzey & Browne 1979). These techniques are natural extensions of the single class solution techniques. As in the single class case, the details of the solution technique depend on the types of the workloads (open or closed). This dictates the organization of our discussion

2.3.3.1. Open Model Solution Technique

Let C be the number of classes in the model. Each class c is an open class with arrival rate λ_c. We denote the vector of- arrival rates by $\bar{\lambda} \equiv (\lambda_1, \lambda_2, ..., \lambda_c)$. Because the throughputs of the classes in open models are part of the input specification, the solution technique for these models is quite simple. The formulae to calculate performance measures of interest are:.

- processing capacity

A system is said to have sufficient capacity to process a given offered load x if it is capable of doing so when subjected to the workload over a long period of time. For multiple class models, sufficient capacity exists if the following inequality is satisfied:

$$\max_k \left\{ \sum_{c=1}^{C} \lambda_c D_{c,k} \right\} < 1$$

This simply ensures that no service center is saturated as a result of the combined loads of all the classes. In the derivations that follow, we will assume that this inequality is satisfied.

- throughput

By the forced flow law the throughput of class c at center k as a function of $\bar{\lambda}$ is:

$$X_{c,k}(\vec{\lambda}) = \lambda_c V_{c,k}$$

- utilization

From the utilization law:

$$U_{c,k}(\vec{\lambda}) = X_{c,k}(\vec{\lambda})S_{c,k} = \lambda_c D_{c,k}$$

- residence time

As with single class models, residence time is given by:

$$R_{c,k}(\bar{\lambda}) = \begin{cases} D_{c,k} & (delay\ centers) \\ D_{c,k}\left[1 + A_{c,k}(\bar{\lambda})\right] & (queueing\ centers) \end{cases}$$

where $A_{c,k}(\bar{\lambda})$ is the average number of customers seen at center k by an arriving class c customer. The intuition behind this formula is similar to that for single class models. For delay centers, a job's residence time consists entirely of its service demand there, $V_{c,k}$ $S_{c,k}$. The explanation of the formula for queueing centers depends on the scheduling discipline used. For FCFS centers, the residence time is simply the sum of an arriving job's own service time, $V_{c,k}$ $S_{c,k}$ and the service times of the jobs already present at the arrival instant, $V_{c,k}$ $[A_{c,k}(\bar{\lambda})S_{c,k}]$, since at FCFS centers all classes must have the same service time at each visit. For PS centers, the residence time is the basic service requirement, V, kSc,k, "inflated" by a factor representing the service rate degradation due to other jobs competing in the same queue, $1 + A_{ck}(\bar{\lambda})$. For LCFS centers the equation has no simple intuitive explanation, but nonetheless is valid.

An implication of the assumptions made in constructing separable networks is that the queue length seen on average by an arriving customer must be equal to the time averaged queue length. Thus, for queueing centers :

$$R_{c,k}(\bar{\lambda}) = D_{c,k}\left[1 + Q_k(\bar{\lambda})\right]$$

where $Q_k(\bar{\lambda})$ is the time averaged queue length at center k (the sum over all classes). Applying Little's law:

$$R_{c,k}(\bar{\lambda}) = D_{c,k}\left[1 + \sum_{j=1}^{C} \lambda_j R_{j,k}(\bar{\lambda})\right]$$

Notice now that the right hand side of the above equation depends on the particular class c only for the basic service demand $D_{c,k}$. Thus,

$\frac{R_{c,k}(\bar{\lambda})}{R_{j,k}(\bar{\lambda})}$ must equal $\frac{D_{c,k}}{D_{j,k}}$, giving $R_{j,k}(\bar{\lambda}) = \frac{D_{j,k}}{D_{c,k}} R_{c,k}(\bar{\lambda})$. Substituting into the equation above and re-writing, we have:

$$R_{c,k}(\vec{\lambda}) = \frac{D_{c,k}}{1 - \sum_{j=1}^{C} U_{j,k}(\vec{\lambda})} \qquad \text{(queueing centers)}$$

- queue length

Applying Little's law to the residence time equation above, the queue length of class c at center k, $Q_{ck}(\vec{\lambda})$, is:

$$Q_{c,k}(\vec{\lambda}) = \lambda_c R_{c,k}(\vec{\lambda})$$

$$= \begin{cases} U_{c,k}(\vec{\lambda}) & \textit{(delay centers)} \\ \dfrac{U_{c,k}(\vec{\lambda})}{1 - \sum_{j=1}^{C} U_{j,k}(\vec{\lambda})} & \textit{(queueing centers)} \end{cases}$$

- system response time

The response time for a class c customer, $R_c(\vec{\lambda})$, is the sum of its residence times at all devices:

$$R_c(\vec{\lambda}) = \sum_{k=1}^{K} R_{c,k}(\vec{\lambda})$$

- average number in system

The average number of class c customers in system can be calculated using Little's law, or by summing the class c queue lengths at all centers :

$$Q_c(\vec{\lambda}) = \lambda_c R_c(\vec{\lambda}) = \sum_{k=1}^{K} Q_{c,k}(\vec{\lambda})$$

2.3.3.2. Closed Model Solution Techniques

A closed, multiple class model consists of C classes, each of which has a fixed population. We denote the workload intensity by $\vec{N} \equiv (N_1, \ldots\ldots, N_C)$, where N_C, is the class C population size. Because the throughputs of closed classes are not provided as inputs, obtaining solutions for closed models is somewhat more complicated than for open models (Sanguinetti & Billington 1980).

The solution technique used is an extension of the single class mean value analysis (MVA) algorithm. Like its single class counterpart, multiple class MVA relies on three key equations:

- For each class, Little's law applied to the queueing network as a whole

$$X_c(\bar{N}) = \frac{N_c}{Z_c + \sum_{k=1}^{K} R_{c,k}(\bar{N})}$$

- For each class, Little's law applied to the service centers individually

$$Q_{c,k}(\bar{N}) = X_c(\bar{N}) R_{c,k}(\bar{N})$$

It also is useful to consider the total queue length at center k:

$$Q_k(\bar{N}) = \sum_{c=1}^{C} Q_{c,k}(\bar{N})$$

- For each class, the service center residence time equations

$$R_{c,k}(\bar{N}) = \begin{cases} D_{c,k} & (\textit{delar centers}) \\ D_{c,k}\left[1 + A_{c,k}(\bar{N})\right] & (\textit{queueing centers}) \end{cases}$$

where $A_{c,k}(\vec{N})$ is the arrival instant queue length at center k seen by an arriving class c customer. We note that performance measures can be computed using the above equations once the $A_{ck}(\vec{N})$ are known.

As with single class models, there are two approaches to the evaluation of closed models, exact and approximate. (We emphasize again that the word "exact" refers to how the solution relates to the model, not how the solution of the model relates to the system being modelled.) As with the single class MVA algorithms, the two methods differ in how the arrival instant queue lengths are computed.

2.3.3.3. Exact Solution Technique

To obtain an exact solution of a closed model, one must compute the values of the $A_{ck}(\vec{N})$ exactly. Given these values, equations can be applied to compute the full solution of the model. The key to the exact MVA solution technique is the multiple class generalization of the relationship used in the single class case:

$$A_{c,k}(\bar{N}) = Q_k(\bar{N} - \bar{1}_c)$$

where $(\bar{N} - \bar{1}_c)$ is population $\bar{N}$ with one class c customer removed. Intuitively, the queue length seen upon arrival to a center is equal to the time averaged queue length at the center with the arriving customer removed from the network.

Beginning from the trivial solution of the network with the empty population $\bar{O}(Q_k(\bar{O}) = 0$ for all centers k), equation (2.3.4) can be used, along with equations, to construct iteratively the solutions for increasing populations, culminating in performance

measures for the population of interest, $\bar{N}$. Note that in general the solution for each population $\bar{n}$ requires as input C solutions, one for each population n-l, c = 1,2,..., C.

2.3.3.4. Approximate Solution Technique

Because the exact solution technique can require excessive time and space for large numbers of classes, the approximate solution technique often is the only one that can be used in practice. Moreover, since the approximate technique is quite accurate, it is useful as a general technique, even for networks that could be solved exactly.

The multiple class approximate solution technique is a straightforward extension of the single class approximation. The estimates are obtained based on the time averaged queue lengths at the service centers with the full customer population. Thus, the approximate solution technique does not require that one first compute solutions for all populations between the zero population and the full population, but instead iterates at the full population. An initial guess for time averaged queue lengths is made to start the iteration. The approximating function is applied to this guess, and the resulting approximate arrival instant queue lengths. Applications of equations result in new estimates for time averaged queue lengths, which then can be used to begin the next step of the iteration. The iteration continues until successive estimates of time averaged queue lengths are sufficiently close.

The significant advantage of this method over the exact technique is that it iterates on solutions of the network with the full customer population $(\vec{N})$, rather than building up from the solution for the empty network. The approximation therefore requires much less storage than the exact technique, since it maintains the solution of the network for only one population $(\vec{N})$. In particular, the storage requirement is proportional to the product of C and K. The savings in time are harder to quantify because of the iterative nature of the approximate algorithm, although empirically these savings are considerable. The number of operations required per iteration is proportional to the product of C and K. (In other words, the populations of the classes are not a consideration.) Less than two dozen iterations typically are required for convergence to less than a 0.1% change in queue lengths. The accuracy of the technique typically is within a few percent of the exact solution for throughputs and utilizations, and within 10% for queue lengths and residence times.

As noted, the approximate solution technique is built upon estimates for the arrival instant queue lengths at each device for each class that depend only on information obtained from the solution of the network with the full population, A particular estimate for the function h, that has been used successfully is:

$$\begin{aligned} A_{c,k}(\vec{N}) &= Q_k(\overrightarrow{N-1_c}) \\ &\approx h_c\left[Q_{1,k}(\vec{N}),\ldots,Q_{C,k}(\vec{N})\right] \\ &\equiv \left[\frac{N_c-1}{N_c}Q_{c,k}(\vec{N})\right]+\sum_{\substack{j=1\\ j\neq c}}^{C}Q_{j,k}(\vec{N}) \end{aligned}$$

Comparing equation to the exact formula, it is evident that the assumption made in the approximation is that the removal of a customer from the network does not affect the placement of customers in other classes, and reduces queue lengths in its own class in proportion to their original size. Equation has worked well in practice. More sophisticated estimates also have been used, although these are somewhat more difficult to implement and require more machine resources, in terms of both time and space.

An important benefit of the approximate technique is that non-integer multiprogramming levels easily are incorporated in the model. One simply sets N, to the (non-integer) multiprogramming level and applies the approximation. No interpolation between separate integer solutions is required.

Chapter III

Models for Computer Systems

In Chapter III, we discuss the parameterization of models, which we discussed in Chapter II. Parameterization is the heart of the modelling process. Our presentation is divided into three parts. In section 3.1 we discuss the construction of baseline models of existing systems. The inputs required by queueing network models can be divided into three groups: the customer description, the center description, and the service demands. The information required to determine the values of these inputs is obtained from a system description and data recorded and reported by various monitors. Many of the input values can be determined in a straightforward manner from this information. Other values, however, must be inferred.

In section 3.2 we discuss modification analysis: the process of adjusting parameter values to project performance for modified environments. we have indicated, how to modify the parameters of a baseline model to represent various common changes to the workload, to the hardware, and to the system software and operating policies. A key point in conducting a modification analysis, especially as part of a study in which a large number of alternatives must be considered, is the need to identify those effects of the modification that are primary, and those that are secondary. Primary effects typically are easy to anticipate and to represent. Secondary effects typically are less easy to anticipate, and even once anticipated, less easy to quantify and represent.

In section 3.3 we discuss the use of queueing network models to project the performance of future systems - systems for which baseline models cannot be constructed and validated. The process of designing a new system involves continuous tradeoffs between cost and performance. Queueing network models can help to quantify performance, and thus to guide the entire design process. We began with a description of several early experiences in projecting the performance of future systems. We then discussed various aspects of a general approach to the problem.

3.1 Basic Systems

3.1.1. Introduction

We will discuss the construction of baseline models of existing systems. This activity relies on knowledge of the hardware, software, workload, and monitoring tools associated with the system under study. It also requires access to information recorded by accounting and software monitors during system operation. Here, we describe general approaches applicable to a variety of systems. We have divided the inputs of queueing network models into three groups: the customer description, the center description, and the service demands.

3.1.2. Types and Sources of Information

The information required to specify parameter values for a queueing network model of an existing system includes static information about the system configuration and dynamic information extracted from records produced during system operation by various monitoring packages. Some information is recorded for purposes of accounting, while other information is recorded explicitly for performance evaluation purposes. Software packages of varying degrees of sophistication are available for storing, analyzing, and reporting the information recorded during system operation. Our intention is not to be comprehensive, but rather to highlight points of particular relevance to the construction and use of queueing network models.

One type of information required is a description of the hardware and software of the system. With respect to hardware, this information includes an enumeration of the components of the system (processors, channels, storage devices, communication devices, etc.) and an indication their interconnections (e.g., the paths over which data can be moved from a particular storage device to memory). With respect to software, this information includes the operating system in use, and the values of parameters that influence resource allocation. Examples of such parameters include CPU scheduling priorities for various workload components, placement of files on storage devices, etc.

This system description is relatively static, in that it changes only week to week or month to month. The information it provides about the hardware suggests what resources should be represented as centers in the model. The information it provides about the software and operating policies suggests appropriate modelling assumptions and helps in the interpretation of measurement data.

Another type of information that is required is recorded dynamically during system operation by various monitors. Accounting monitors write records at the termination of batch jobs or interactive sessions, indicating the system resources consumed by the job or session (CPU seconds, I/O operations, memory residence time, connect time, etc.). Software performance monitors write records describing resource usage and performance status from another point of view. At specified intervals, queue lengths or device status indicators may be sampled and the results written in a record. Also, certain events that are considered significant (such as swapping a customer out of main memory) may be documented in a record.

When using a reporting routine to obtain information, it is necessary to specify the interval of time over which information is to be gathered. Generally, it is appropriate to run the monitor during peak loads, as these present the most significant performance problems. The duration of the observation interval should be long enough that end effects do not significantly affect the accuracy of the measurements. End effects are measurement errors caused by the fact that some customers are processed partly within and partly outside of the observation interval. In particular, it is typical to assume that the system operates in flow balance over the measurement interval, so that the job arrival and completion rates are equal. However, because some jobs arrive but do not complete during the interval, and other jobs arrive before but complete during the interval, flow balance may not hold. Clearly, measurements obtained from longer observation intervals are affected less by these end' effects than are shorter intervals. Typically, observation intervals of thirty to ninety minutes are appropriate for obtaining software monitor data. If monitoring overhead is a concern, shorter intervals can be used, but the danger of anomalies is increased.

Other sources of useful information include hardware monitors and monitors specialized for particular application subsystems (such as data-base or telecommunications subsystems). Hardware monitors, because they are "external observers" of the system, obtain accurate measurements and do not perturb system operation. They are incapable, however, of associating resource usage with workload components. The specialized application subsystem monitors are helpful in assessing the performance of subsystems whose autonomy from the host operating system prevents standard monitors from being able to record their activity. Table 3.1.1 summarizes the information typically available from various sources.

type	information provided
system description	hardware configuration operating system (and version) resource allocation and scheduling strategies tuning parameter values
accounting monitor	CPU usage, by workload component logical I/O operation count, by workload component customer completions. by workload component
software monitor	measured busy time, by device physical I/O operation count, by device average queue length, by device throughput, by workload component average response time, by workload component
hardware monitor	observed busy time, by device

Table 3.1.1 Sources of Information

Enhanced awareness of the problems of configuration management and capacity planning has led recently to some encouraging progress in the use and management of system measurement data. First, special reporting routines tailored to the requirements of queueing network modelling have been developed for some systems. These routines analyze records produced by existing accounting and software monitors. Some are capable of defining a queueing network in a format directly acceptable by particular queueing network modelling software packages. While these routines are a great aid, intervention by an analyst still is necessary in most cases to obtain a validated model. This is true because of inadequacies in the measurement data, and the fact that the analyst's knowledge of the system is not available to the automated routine.

Second, some of the newer reporting routines have been generalized to be capable of using and contributing to a performance database. The records written by various monitors constitute a rudimentary performance database. Merely organizing the records according to their types and source makes them easier to use. The utility of the database is further enhanced, however, if it is extended to include aggregated information produced by reporting routines. There are several advantages to maintaining such a performance database. For one, long-term trends can be examined if information aggregated on a month by month basis is included in the database. Also, information intended for management planning can be isolated from the more technically oriented information intended for system tuning. Finally, by having various aggregations of monitoring information available in a database, the need for regular printed reports is substantially reduced.

3.1.3. Customer Description

Most large computer systems have workloads consisting of several identifiable components. Performance studies often are intended to assess performance of each workload component, since system-wide average values for throughput and response time have little significance in systems that include such diverse workload components as background batch and foreground transaction processing. There are several goals to meet in deciding how to assign the workload components of the system to the customer classes of a queueing network model:

- Classes should consist of customers whose service demands are of comparable magnitude and similar balance across service centers, since input parameters to the model for all customers in the same class are identical..
- Classes must distinguish workload components for which independent performance projections are desired as outputs of the model.
- Classes may be made to correspond to accounting and performance groups. This facilitates the calculation of various parameter values, since accounting data is organized by accounting group.
- Classes may be used to distinguish work generated by various organizational units. This permits unit specific performance projections, and facilitates later modification analysis

A first step in identifying customer classes is to group portions of the workload according to whether they are best represented as batch, terminal, or transaction types. Often, the nature of a workload component suggests an appropriate type: if requests arrive at a constant rate, then transaction; if requests are generated by a set of users that await the completion of service to one request before generating another, then terminal; if the number of active requests is constant, then batch. Variations are possible, though, especially in conducting a modification analysis. As one example, a workload component might in fact consist of users at terminals, but for planning purposes its intensity might be described in terms of a request arrival rate. In this case, the use of a transaction type might be appropriate. As another example, a system might have many workload components, only a few of which are of interest. The presence of the other components might be reflected in the model by a single "aggregate" class of transaction type

Within each type of customer class, further separation of workload components may be desirable. Batch work of different priorities may be represented as distinct

classes. Different interactive systems may be treated as separate terminal classes. If trivial transactions can be distinguished from substantive transactions, then different classes can be used to distinguish the two groups.

Having identified each workload component to be represented as a distinct customer class and determined the type of that class, the next step is to establish the workload intensity of each class. For a transaction class, the workload intensity is the transaction arrival rate. Over a reasonably long observation interval in a system that is not saturated, the arrival rate is essentially the same as the completion rate. Consequently, an estimate for the arrival rate of class c is:

$$\lambda_c = \frac{measured\ completions\ of\ class\ c}{length\ of\ measurement\ interval}$$

For a batch class, the workload intensity is given by the average number of batch customers active. An estimate for N_c, the number of class c customers, can be obtained in several ways:

- If jobs are processed in a fixed number of regions and memory queueing times are high then N, is the number of processing regions.
- If the software monitor provides an estimate of the average multiprogramming level of the class over the observation interval by sampling, then N, can be taken to be that estimate.
- If accounting data provides the residence time of each job in the central subsystem, then N, can be estimated by:

$$N_c = \frac{\sum_{\substack{class\ c \\ jobs}} measured\ job\ residence\ time}{length\ of\ measurement\ interval}$$

For a terminal class, workload intensity is specified by the number of active terminals, N_c, along with the average think time, Z_c. Three possibilities for estimating N_c for terminal classes correspond directly to the three methods used for batch classes:

- If terminals connect to the system through a limited number of ports, and if all ports are busy throughout most of the observation interval, then N_c is the number of ports.
- If the software monitor provides the average number of active terminals over the observation interval, then N_c can be taken to be that number.

- If accounting data includes session lengths, then N, can be estimated (over an observation interval that is long relative to average session length in order to restrict end effects) by:

$$N_c = \frac{\sum_{\substack{class\ c \\ sessions}} measured\ session\ length}{length\ of\ measurement\ interval}$$

The average think time of a terminal class often is one of the most difficult input parameters to estimate. There are several reasons. First, there are differing views of when think time starts and ends. We will adopt the one in which it starts with the arrival of the first character of a response from the system, and ends when the last character of the next request to the system is entered. Second, some systems allow a stream of commands to be entered without awaiting responses. Such systems can cause think times to be negative! Third, some think times become so long that they actually represent a loss of an active terminal. Fourth, average think time seldom is measured directly by performance monitors. Consequently, the best estimate of think time often is obtained by estimating 2, from the response time law:

$$Z_c = \frac{N_c}{X_c} - R_c$$

where N_c is estimated as described above, and X_c and R_c are measured values. Because there often is less confidence in the estimate of think time than in the estimates of other parameters, it may be desirable to test the sensitivity of the model to this value.

When memory constraints are imposed on transaction or terminal classes, it is necessary to specify the capacity associated with each domain so that the modelling approach can be used. The capacity of each domain typically is known from the system description. Whether or not the domain was filled to capacity in a particular measurement interval is revealed by comparing the average number active among classes assigned to the domain to the domain capacity.

3.1.4. Center Description

The service centers of a queueing network model correspond to significant points of congestion or delay in the system. There are many ways of representing system resources by a set of service centers. Here we suggest only the most widely accepted methods, which have proven successful in a large number of modelling studies. For systems with single CPUs and for tightly-coupled multiprocessors, a single service center

is used to represent the CPU(s) in the queueing network model. Loosely-coupled multiprocessors are modelled by including one service center per processor. Front end communications processors and back end database machines also may be represented as separate service centers.

The representation of disk subsystems can be done in a variety of ways. A number of components are involved in each disk I/O operation. The modelling approach that has proven most successful, however, is to use a single service center to represent each disk. Congestion due to other I/O subsystem components is represented by calculating an appropriate effective service demand for each center.

Other peripheral devices can be represented more simply than disks. Unit record equipment typically is ignored in constructing queueing network models. This is justified in many systems because spooling makes the use of unit record devices asynchronous. Similarly, terminal controllers typically are not represented. If delays in the communications front end are thought to be important in a particular study, then a special approach must be used. This might involve a hierarchical model in which a conventional central subsystem model is evaluated, and then the delays due to communication are represented in a high-level model that includes an FESC representing the central subsystem.

3.1.5. Service Demands

The final set of values needed to parameterize a queueing network model are the service demands at each center of the customers belonging to each class. Obtaining these values can be a difficult and time consuming process. As a practical consideration, it is important to concentrate on obtaining accurate estimates for the most heavily utilized centers, because a small error in estimating the service demands at the bottleneck center will affect performance projections more than a much larger error at a lightly utilized center. In estimating service demands, the three center types (delay, FESC, and queueing) are treated differently.

Delay centers have service demands that represent a delay that is not caused by congestion. It usually is not difficult to determine appropriate values for delay centers. In addition, errors in the service demands at delay centers are not "magnified" by queueing delay calculations when the model is evaluated.

For FESCs, the load dependent service rates can be determined in many ways. Two major approaches are evaluating low-level queueing network models and considering hardware characteristics.

The estimating service demands for queueing centers is straightforward: at the conclusion of the measurement interval, the measured busy time for each class at each device is divided by the number of system completions for the class. In practice, however, two difficulties arise:

- In the multiple class case, the available data frequently is insufficient to apportion the measured busy time among the classes with certainty. The reasons and the remedies differ for various devices and various systems.
- A portion of the busy time attributed to each class is intrinsic to that class: its basic processing and I/O requirements. The remainder consists partly of service demand inflation and partly of overhead. Service demand inflation, introduced in Chapter 10, is the component of measured disk busy times due to contention in the I/O subsystem. Overhead is work done by the operating system "on behalf of" the customers of the class. Part of the overhead component is fixed, in that it does not depend on system congestion, and part of it is variable and typically increases with system load. In a baseline model these distinctions do not matter, but in conducting a modification analysis they can be crucial, for the service demand inflation and variable overhead components of the model usually change in a new environment.

3.1.5.1. Estimating Processor Service Demands

Since the CPU typically is a heavily utilized resource, it is important to determine accurately the service demands of the various classes there. As noted in Table 3.1.1, monitor data often includes the CPU usage and the number of customer completions for each workload component. Unfortunately, the quotient of these quantities turns out in practice to yield a poor estimate of CPU service demand. The reason is that the CPU usage reported on a per class basis often fails to capture significant amounts of CPU activity. More specifically, the sum of the CPU busy times reported on a per class basis is likely to be considerably less than the total CPU busy time reported by a monitor that does not attempt to distinguish among classes. The ratio of attributed CPU usage for a class to the total CPU busy time due to activities initiated by that class is known as the capture ratio. Capture ratios typically range from .85 down to .40 for various systems and various workload components. For a particular system, the overall capture ratio can be

estimated as suggested above: by dividing the sum of the CPU busy times reported on a per class basis by the total CPU busy time reported by a monitor that does not attempt to distinguish among classes (often by a software monitor). In the case of single class models, dividing the estimate of total CPU busy time from software monitor data by the estimate of total customer completions from either accounting or software monitor data will yield a good estimate for CPU service demand. In the case of multiple class models, though, techniques must be devised to apportion the unattributed CPU busy time among classes.

This process has three steps:

- calculate the unattributed busy time during the interval
- decide how much to attribute to each class
- compute how much to attribute to each customer of each class

The second of these steps is the interesting one, and will be addressed in the paragraphs that follow.

Consider a system with a workload consisting of two components: batch jobs and interactive users. Assume that information comparable to that listed in Table 3.1.1 has been obtained. Let f_{BATCH} and f_{INTER} be (unknown) factors by which the attributed CPU busy time for each class must be multiplied so that all measured CPU busy time is attributed to some class. (Observe that f_c is the inverse of the capture ratio for class c.> This leads to the equation:

$$B_{CPU} = f_{BATCH} \times A_{BATCH,CPU} + f_{INTER} \times A_{INTER,CPU}$$

where $A_{c,CPU}$ is the CPU usage attributed to class c, and $B_{CUP}$is the total measured CPU busy time.

To determine unique values for f_{BATCH} and f_{INTER} we must establish a relationship between them in addition to this equation. Several possibilities exist:

- Assume that the ratio of total CPU time to attributed CPU time is the same for each class, yielding:

$$f_{BATCH} = f_{INTER} = \frac{B_{CPU}}{\left[A_{INTER,CPU} + A_{BATCH,CPU}\right]}$$

- Since the unattributed CPU busy time is likely to be overhead, use class based information on activities likely to cause CPU overhead to determine a relative measure of total overhead for each class. For instance, assuming that overhead is due

almost entirely to page fault handling, and letting OK, be the measured number of pages transferred because of class c faults, we have:

$$f_{INTER} = 1 + \frac{\frac{OV_{INTER}}{OV_{INTER} + OV_{BATCH}} \times [B_{CPU} - [A_{INTER,CPU} + A_{BATCH,CPU}]]}{A_{INTER,CPU}}$$

The second approach is the more reasonable. Unfortunately, more than one factor inevitably

$$OV_C = \sum_{all\ factors\ i} weight\ i \times factor\ i_C$$

contributes to overhead. Thus, OV_c is better defined as the weighted sum of several factors:

When one attempts to apply this approach in practice, two common problems are apt to be encountered:

- Even for a single measurement interval, it may be difficult to determine which factors to consider, and what weights to assign to these factors. Iteration inevitably is required: estimate weights, calculate service demands, evaluate model, re-estimate weights, etc.
- If one truly is to have confidence in the weights selected, then data from a number of measurement intervals must be considered, and weights must be found that yield good model results when applied to each set of data. An ad hoc approach can be adopted, or linear regression techniques can be used.

Once f_{BATCH} and f_{INTER} have been determined, the service demands of the two classes can be estimated by the equation:

$$D_{c,CPU} = \frac{f_c \times A_{c,CPU}}{measured\ class\ c\ completions}$$

Note that the service demands determined in this way include intrinsic service, fixed overhead, and an amount of variable overhead that reflects the degree of system congestion in the interval covered by the measurement data.

3.1.5.2. Estimating I/O Service Demands

I/O activity in most current computer systems is dominated by operations on direct access storage devices (fixed head, movable head, and electronic disks). Tape I/O and I/O for staging data to and from mass storage devices plays a secondary role. Other

types of peripheral devices typically are inconsequential with respect to performance. Our discussion in this section focuses on disk I/O, reflecting its importance.

We described how the lengths of certain portions of disk service requirements (seek, latency, rotation, and transfer) could be established from system knowledge and measurement data. We assumed that both the visit counts and the service times per visit for each class at each disk were known. In this section, we suggest a method for determining these quantities. First we consider the visit counts, then the service times.

We distinguish two ways of viewing I/O operations. Physical I/O operations correspond to activations of I/O subsystem components to transfer data to or from peripherals. Logical I/O operations correspond to operating system calls by customers requesting access to blocks of information. For a number of reasons physical and logical I/O operations do not correspond directly to one another. Sometimes, a logical I/O operation may not result in a physical I/O operation; for example, a logical I/O operation may request access to a block of information that already is in memory. Sometimes, a logical I/O operation may result in several 288 Parameterization: Existing Systems physical I/O operations; for example, errors detected in reading or writing a block may cause operations to be retried.

It is the physical I/O operations that correspond to the visit counts, but physical operations seldom are reported on a per class basis. Typically, logical I/O operations are broken down by class but not by device (often by an accounting monitor), while physical I/O operations are broken down by device but not by class (often by a software monitor). The first step in confronting this situation is to estimate the ratio of physical to logical I/OS for each class. We define g, to be the ratio of physical to logical I/OS for class c. (The assumption that the ratio depends on class but not device is realistic in most systems.) Estimating the g, is a problem analogous to estimating the f, in the case of the CPU. Possible approaches include:

- Assume that g_c is the same for each class, so that:

$$g_c = \frac{\sum_{\text{all disks } k} P_k}{\sum_{\text{all classes } j} L_j}$$

- Use generally accepted ratios for standard types of workloads for the architectural family of the system.

- For a number of observation intervals, determine the values for the g_c that best satisfy the set of equations:

$$\sum_{all\ dusks\ k} P_k(i) = \sum_{all\ classes\ c} g_c \times L_c(i)$$

where (i) denotes values obtained during the i-th observation interval.

Once these g_c have been estimated, we proceed to determine the visit counts. In essence, we must satisfy the equations:

$$P_k = \sum_{all\ classes\ c} (measured\ class\ c\ completions) \times V_{c,k}$$

$$L_c = (measured\ class\ c\ completions) \times \sum_{all\ disks\ k} \frac{V_{c,k}}{g_c}$$

Consequently, we must use additional information to specify the $V_{c,k}$ values uniquely. Alternatives include:

- The simplest assumption, which can be used in the absence of any other information, is that all classes use the various disks in the same proportions:

$$\frac{V_{c,k}}{V_{c,k'}} = \frac{V_{c',k}}{V_{c',k'}} \quad for\ classes\ c\ and\ c',\ and\ disks\ k\ and\ k'$$

- The software configuration portion of the system description frequently indicates the location of various key data sets: paging files, swapping files, catalogs, files devoted to various applications, etc. If a particular class is known not to use a device, then its visit count there can be set to zero. If a particular class is known to be the exclusive user of a device, then its visit count there can be set to the measured physical I/O count of the device divided by the measured number of completions of the class. The remaining visit counts can be resolved in a series of stages. At each stage, the distribution of I/OS for the class for which the least flexibility remains is determined.
- In some systems there are software monitors capable of observing directly the number of physical I/OS broken down by both class and device. Although such monitors cause too much overhead to be used continuously, they can be used over short intervals (e.g., 10 minutes) to obtain an indication of the distribution of physical I/OS by class and device.
- Occasionally, the breakdown of logical I/OS by device as well as by class is known. This additional information makes it possible to proceed with greater confidence. In particular, if we can assume that the ratio of physical I/OS to logical I/OS is the same

for each class, then the physical I/OS at a particular device can be attributed to classes in the same proportions as are the logical I/OS.

3.1.6. Validating the Model

Model validation involves comparing these estimates with the measured values of the corresponding quantities. A model can be considered "validated" when it has been demonstrated that, in several (or many) measurement intervals, the differences between the estimates produced by the model and the measured quantities are sufficiently small.

In choosing observation intervals for use in validating the model, it is desirable to look ahead to the types of system changes to be investigated with the model. If the model is to be used to investigate the effect of an increased workload intensity, then the model should be validated on observation intervals representing a range of workload intensities. Similarly, if an increase in the size of main memory is to be considered, it is beneficial to validate the model on several different memory sizes. This could be done in a number of ways. Scheduling parameters could be adjusted to keep the number of active customers artificially low (thus underutilizing the memory). Alternatively, a portion of the memory could be disabled during an observation interval.

The correspondence between model estimates and measured quantities depends on several factors. Single class models can be validated with higher precision than multiple class models because their input parameter values can be determined from measurement data with greater accuracy. Some performance measures can be matched more easily than others. In validating multiple class models, it seldom is possible to reflect the behavior of every class at every device accurately. Clearly, it is desirable to have the model represent most accurately the behavior of the critical (mostly heavily used) resources. Similarly, if one class of customers is of particular interest in a modelling study, then validation of the model should place special emphasis on the performance measures of that class. An important point to note is that queueing network models typically project percentage changes in performance with more accuracy than absolute levels of performance.

Often, even in well conceived and well executed modelling studies, an initial model will not satisfy the validation criterion. In such cases, reasonable modifications of the assumptions used in estimating input parameters (especially service demands) should be attempted. For example, by noting which classes have throughputs underestimated, the analyst may be guided in a reassessment of how overhead should be attributed to the various classes. This review is repeated until the model can be validated. It is not unusual

for several iterations to be required at this stage. In some cases, however, no reasonable technique for estimating inputs yields acceptable results. This is a sign that some important aspect of the system's behavior has not been captured in the model. In many such cases, accuracy can be improved by adding more detail to the model.

It is important to realize the significance of validating a model successfully. If information from measurement data is used to establish values of model inputs, then the fact that the model outputs match the measurement data is, at first glance, not surprising. After a little thought, however, one realizes that success in validation carries the significant implication that the numerous assumptions made in establishing the model are acceptable in the context of the particular system under study. With a validated model, we are prepared to proceed to the modification analysis and performance projection, the subjects of the next chapter.

3.2. System Modifications

3.2.1. Introduction

We will discuss how to represent such modifications by alterations to the inputs of the validated model. The accuracy and utility of the resulting performance projections depend on three factors:

- how well the baseline model..
- how accurately the modifications are forecast.
- how well the anticipated modifications are represented as changes to the model inputs.

In general, system modifications have both primary and secondary effects. For example, a CPU upgrade has the primary effect of reducing the CPU service requirement of each user (in seconds, rather than instructions), and may have one or more secondary effects, such as changing the number of times that each user is swapped, on the average. We will see that for many modifications it is relatively easy to anticipate and represent the primary effects, but harder to anticipate, and thus to quantify and represent, the secondary effects. For this reason, successful performance projection studies in which several alternatives are being considered often take the following form:

- Initially, each alternative is investigated by representing only its primary effects. This can be done quickly.

- The results may reveal that some of the alternatives are not worthy of further consideration. These alternatives are discarded.
- The remaining alternatives are investigated in more detail, with attention paid to secondary as well as primary effects.

3.2.2. Changes to the Workload

The workload presented to a computer system can change in several ways. First, the intensities of workload components can change. Second, the character of workload components can change. Third, the number of workload components can change. The following three subsections describe how the effects of each of these changes can be represented by adjustments to the inputs of a validated model. Both in this section and in the ones to follow, we will indicate modified input parameter values as primed quantities.

3.2.2.1. Changes in Workload Intensities

The most frequently studied workload changes are changes in intensity. Naturally, the primary effect of such a change is reflected by modifying the appropriate workload intensity input parameters.

For a transaction class, a typical workload forecast would be "a 30% increase in transaction volume". This can be represented in the model by $\lambda'_c \leftarrow 1.3\ \lambda_c$.

For a terminal class, a typical workload forecast would be "a 50% increase in the number of active users". This can be represented in the model by $N'_c \leftarrow 1.5\ N_c$.

In the case of both transaction and terminal classes, increased competition for main memory will result from an increase in workload intensity. If the baseline model included a memory constraint, then we may assume that the same constraint still applies. If no such constraint were present in the baseline model, then the analyst must decide whether or not the increased central subsystem population that results from the parameter modification is realistic in light of the amount of memory available. If not, an appropriate memory constraint should be imposed. In either case, the variable component of overhead may increase.

For a batch class, it is unusual for a workload forecast to be phrased in terms of the multiprogramming level, N_c. Additional complexity arises from the fact that the value of this parameter in the baseline model can be due to several factors. At one extreme, there may be a persistent backlog of batch jobs, so that N_c reflects a memory constraint. In this case, an increase in the availability of batch jobs would only result in a larger

backlog. At the other extreme, if sufficient memory is available to activate most batch jobs immediately when they arrive, the value of N_c is not related to a memory constraint. In this case, an increase in the availability of batch jobs would allow N_c to increase. Typically, a workload forecast for a batch class will be phrased in terms of throughput. The analyst must adjust N_c to achieve the forecast throughput, and then consider whether or not the increased central subsystem population is realistic with respect to the available memory.

3.2.2.2. Changes in the Character of Workload Components

Changes to application programs may lead to changes in the resource requirements of customers. Such changes would be represented in a model by adjusting service demands. Three examples are given in the following paragraphs.

It is future to modify an application program to do more checking of the validity and consistency of the input data it receives. The change is projected to increase the CPU path length of a transaction by 20%. The primary effect of this modification can be represented in the model by increasing the CPU service demand of transactions by 20%.

It is proposed to introduce data compression techniques to reduce the space occupied by a file that is processed sequentially by an application. The data transferred by the application will decrease, while its CPU requirements will increase. To represent this modification in the model, the data transfer component of the service demand at the appropriate disk should be decreased, while the service demand at the CPU should be increased.

3.2.2.3. Changes in the Number of Workload Components

The primary effect of removing a workload component from a system is represented easily in the model by eliminating the corresponding customer class. The result will be a decrease in the activity at various devices, and a corresponding improvement in the performance of the remaining workload components.

Similarly, the primary effect of adding a workload component is represented by adding a new class. The result will be an increase in the activity at various devices, and a potential degradation in the performance of the original workload components. Of course, the workload intensity and service demands of the new class must be determined and specified. If a similar application runs at some installation with a similar hardware and

software configuration, then measured service demands can be used. For a new application that cannot be measured, estimating service demands is much harder.

Both the removal and the addition of workload components have a number of effects which, although of lesser importance than changes in device congestion, still can have considerable impact on performance. When a workload component is removed, memory becomes available for allocation to the remaining components. Knowledge of the operating policies of the system is required to determine how to represent this. When a component is added, it may be necessary to obtain memory at the expense of other components. Again, system knowledge is required.

3.2.3. Changes to the Hardware

New hardware products based on recent technological developments are announced with great frequency. This makes capacity planning and configuration management a continuing challenge. Fortunately, queueing network models are well suited to quickly evaluating configuration modifications. In the subsections to follow we describe how CPU upgrades, memory expansions, and I/O subsystem modifications can be represented as modifications to model input parameter values.

3.2.3.1. CPU Upgrades

Perhaps the most common configuration change is the upgrade of a CPU within a family of processors of the same architecture. Fortunately, this also is one of the easiest changes to evaluate using queueing network models. The relative instruction execution rates among processors within a family generally are known and publicized by vendors and user groups.

Consequently, the primary parameter change is to multiply the CPU service demand by the ratio of old CPU's processing rate (r_{OLD}) to that of the new (r_{NEW}):

$$D'_{c,CPU} \leftarrow \frac{r_{OLD}}{r_{NEW}} \times D_{c,CPU} \quad \textit{for each class } c$$

A common secondary effect of a CPU upgrade is a change in variable overhead. Additional memory or I/O equipment often accompanies such an upgrade.

Rather than acquiring a faster CPU, it sometimes is possible to acquire a second processor to form a tightly coupled multiprocessor system. As we have discussed the primary corresponding change to model parameters would be to represent the processor complex as an FESC with service rate approximately twice as great with two or more

customers present as with only one customer present. An important secondary effect is the interference between the processors in accessing memory or shared data structures. This interference causes the capacity of a dual processor to be considerably less than twice the capacity of a single processor. If appropriate measurement data is available, the service rates of the FESC can be set to reflect the degree of interference.

3.2.3.2. Memory Expansions

Since additional memory can be allocated in a number of ways, representing the effect of a memory expansion requires knowledge of the operating policies of the system. The most common way to employ additional memory is to permit an increase in the central subsystem population of various classes. For batch classes, the parameter N, would be changed. For transaction or terminal classes, the memory constraint would be adjusted upwards. The key, of course, is to estimate the extent to which each class will be affected. To some extent, this is under the control of installation-dependent tuning Parameterization: Systems parameters. A few, well chosen experiments with smaller memory sizes can help to determine the effect of operating policies.

Additional memory also can be used to permit workload components to run more efficiently at existing central subsystem populations. In this case, the entire effect of the memory upgrade would be felt as a decrease in variable overhead. A third use of additional memory is to make frequently accessed files permanently resident in memory. Examples include system routines or indices. If measurement data indicates frequency of use for these files, then disk service demands can be decreased by an appropriate amount to represent fixing them in memory.

As a final example, additional memory can be used to increase the size of the disk cache employed by many operating systems. Experimentation with a few different cache sizes would indicate the relationship between disk cache size and disk cache hits (and thus I/O activity).

3.2.3.3. I/O Subsystem Modifications

Each generation of disks can be characterized by basic quantities such as capacity, seek time, latency time, and transfer rate. From these characteristics it is possible to estimate the changes in disk service demands that will result from replacing one type of disk with another. The exact speed ratio depends on block size, seek pattern, and I/O subsystem contention.

A secondary effect to consider in this case is the fact that there is a temptation to reduce the number of drives as part of a conversion effort. The resulting change in seek patterns may cause the average seek distance to increase, making it more difficult to forecast service demands.

An I/O subsystem can be upgraded by increasing the numbers of channels and controllers or by changing the interconnections among existing components, as well as by adding storage devices. Changes of this sort would be expected to reduce contention in the I/O subsystem by creating alternate paths between the CPU and the disks. Consequently, the contention component of effective disk service demands would be reduced. The techniques suggested in Chapter 10 are oriented towards assessing the effect of this sort of modification.

3.2.4. Changes to the Operating Policies and System

Operating systems typically leave a great deal of flexibility to installations with respect to certain operating policies that can have a significant influence on performance: placement of files on devices, assignment of workload components to memory domains, setting of scheduling priorities, etc. The first three subsections that follow discuss the representation of modifications to such operating policies in queueing network models. The fourth subsection discusses the representation of the effect of operating system upgrades.

3.2.4.1. File Placement

Performance often can be improved by altering the assignment of flies to devices, with the objective of balancing the load across disks and other I/O subsystem components. The parameter changes to represent such modifications in the model are straightforward. If the disks involved are identical, then the primary effect can be represented (in the case of three disks) by:

$$D'_{each\ disk} \leftarrow \frac{D_{Disk\,1} + D_{Disk\,2} + D_{Disk\,3}}{3}$$

If a decrease in the contention component of the effective disk service demands is expected, with the analyst balancing the seek, latency, and transfer components as above.

If the devices involved differ in speed, more effort is required. The service demands in the baseline model must be viewed as the product of visit counts and service

times per visit. The service demand at each of k disks after balancing is given by the equations:

$$D'_{each\ disk} = V'_{Disk\ 1} S_{Disk\ 1} = \cdots = V'_{Disk\ k} S_{Disk\ k}$$

and:

$$\sum_{j=1}^{k} V'_{Disk\ j} = \sum_{j=1}^{k} V_{Disk\ j}$$

Thus, we assume that balancing the load does not change the service times at the devices substantially, and that the total number of physical I/O operations does not change. The service demand for each disk will be:

$$D'_{each\ disk} = \frac{\sum_{j=1}^{k} V_{Disk\ j}}{\sum_{j=1}^{k} (1/S_{Disk\ j})}$$

Thus, we would seek an assignment of files to disks such that capacity constraints are not exceeded and the visit count-to files assigned to each disk approximately satisfy:

$$V'_{Disk\ j} = \frac{D'_{each\ disk}}{S_{Disk\ j}}$$

The approach described above generally will succeed only in approximately balancing the I/O load. The service times at the various disks in fact will change due to altered seek patterns and other secondary effects. Also, carefully balancing the I/O load according to access patterns observed during one period of the day will not lead to a balanced load throughout the day. Consequently, in doing I/O balancing, peak load periods should be given most consideration, but implications for other periods should be considered.

When representing the addition of disks to a configuration, it is appropriate to attempt I/O load balancing at the same time. An example in Section 4.2.6 illustrates the evaluation of the effect of I/O load balancing through altering the placement of user files.

3.2.4.2. Memory Allocation

The allocation of memory is critical to performance. An operating system typically requires substantial memory for its own use, devoted to resident code and data structures, transient routines, and I/O buffers.

The remaining memory is allocated to user programs. As described above, it is typical to define domains with limited capacities and to assign workload components to these domains. This approach regulates competition for memory so that thrashing does not occur. The primary effect of altering the allocation of memory can be represented by changing the domain capacities in the model. The problems that arise are similar to those that arise in modelling the addition of memory, which were discussed in an earlier section. Especially in a virtual memory system, it can be difficult to determine the number of jobs that can be accommodated in a specific amount of memory. Limited benchmarking can be of assistance in determining how the rate of paging depends on the amount of main memory available for each active customer.

3.2.4.3. Tuning Parameters

In most operating systems, many of the scheduling and resource allocation activities are controlled by tuning parameters. Among other things, such parameters control the dispatching and initiation priorities of various workload components, and the amount of service guaranteed to customers before they are eligible to be swapped out. Queueing network models can be used to gain an understanding of the effects of changing certain tuning parameters. The major benefit of such studies is to estimate the extent to which performance might be affected by a particular parameter.

3.2.4.4. Operating System Upgrades

Operating systems provide certain services to the programs that execute under them. The variety of services available and the efficiency with which they are delivered differs from one system to another. The operating systems for most major computer systems evolve continually. Each version (or "release") typically provides some new functions, and possibly improves the efficiency with which earlier functions are delivered.

To model the effect of an operating system upgrade, the analyst must determine the relative efficiency of various functions by relying either on 306 Parameterization: Systems statements by the vendor or on experience of early users. Given this information, modification of the model is straightforward. For example, if it is claimed that CPU path lengths for user I/O processing will be decreased by a factor of two, the analyst first must determine this overhead component of CPU service demand for the workload on the existing system, then divide it by two to represent the effect of the new release.

Operating system efficiency also is of importance when comparing various systems under consideration for the support of a new workload. In this case, it is necessary to translate a workload description in system independent terms into service demands for each candidate system. In the case of CPU service demands, for example, the relative CPU execution rates of the various systems tell only part of the story: the efficiency of operating software can have a dramatic effect on performance.

3.2.5. Secondary Effects of Changes

Previous sections have concentrated on the representation of the primary effects of system changes. In the present section we consider the representation of certain secondary effects that are common to a number of the modifications we have discussed.

We showed one approach to estimating the change in swapping activity that would accompany various system modifications. We also showed how variability in paging activity could be incorporated in a model. We developed algorithms to estimate path contention in complex I/O subsystems as a function of other system characteristics. We mentioned the representation of the CPU overhead that accompanies all other activities.

In this section, we will talk in more general terms about techniques to forecast the level of CPU and I/O overhead present in a system, Our approach will be one that was suggested in earlier chapters: to extrapolate from the results of a few measurement intervals.

3.2.5.1. Changes in Variable Overhead

Almost every contemplated change to a system will, as a secondary effect, change the variable overhead incurred in system operation. The most significant examples of this in many systems are changes in paging and swapping rates, which involve both CPU and I/O activity. CPU upgrades, memory expansions, increases in workload intensities, even changes in the priority structure among classes, all have the secondary effect, of changing paging and swapping rates.

As we have noted before, when a model is used to project performance for relatively minor modifications, changes in variable overhead need not be considered. The more significant the modification under consideration, the more important it is to attempt to quantify these changes. This is a difficult task; in some cases it will be necessary to employ a sensitivity analysis to indicate the range of anticipated performance.

Assume that variable overhead increases linearly with workload intensity, an appropriate estimate for the service demand at device k for a new workload intensity I' is given by:

$$D'_k = D_k^{(1)} + (I' - I^{(1)}) \times \left[\frac{D_k^2 - D_k^1}{I^{(2)} - I^{(1)}} \right]$$

Approximating the dependence of variable overhead on workload intensity by more complex curves typically yields slightly greater accuracy, particularly if workload intensity changes are large, but this gain may not justify the added complexity.

Careful treatment of variable overhead is more difficult in multiple class models. There are several issues involved:

- In the multiple class case, more observation intervals are necessary, because the workload intensity now is. a vector. For example, if the workload consists of two major components, interactive and batch, we might consider four observation intervals: heavy batch and heavy interactive, heavy batch and light interactive, light batch and heavy interactive, and light batch and light interactive.
- Within each observation interval, it is difficult to attribute variable overhead to the classes accurately, because of the inadequacy of measurement tools.
- Where the single class case involved fitting a curve through some points, the analogous procedure for the multiple class case with C classes involves fitting a C-dimensional surface. Such multidimensional surface fitting, however, is too complex to be justified considering other limitations on the accuracy of this technique. In almost all cases, a sequence of one-dimensional extrapolations based on changes to one workload component at a time will suffice.

From the preceding discussion, it should be apparent that estimating changes in variable overhead is difficult, and cannot be done with high confidence. Consequently, it often is appropriate to evaluate the model under both optimistic and pessimistic assumptions in order to assess the importance of accurately estimating overhead in projecting performance.

For example, when memory size is increased, paging and swapping activity typically are reduced. Because it is difficult to determine the extent of this reduction, we might evaluate the model once assuming no change in paging and swapping activity, and again assuming that all paging and swapping activity is eliminated.

3.2.5.2. Changes in I/O Service Times

Many modifications have the secondary effect of changing the seek, transfer, and contention components of effective disk service time. Relocating files from one disk to another can cause the seek patterns to change on each disk. Typically, the average seek time will increase on the disk to which the file is moved and will decrease on the other. If all files do not have the same block size, then the average transfer times at both disks also will be altered.

3.3. Proposed Systems

3.3.1. Introduction

The preceding two sections have discussed the parameterization of queueing network models of basic systems and developing systems. In this section we consider models of proposed systems: major new systems and subsystems.

The process of design and implementation involves continual tradeoffs between cost and performance. Quantifying the performance implications of various alternatives is central to this process. It also is extremely challenging. In the case of basic systems, measurement data is available.

In the case of evolving systems, contemplated modifications often are straightforward, and limited experimentation may be possible in validating a baseline model. In the case of proposed systems, these advantages do not exist. For this reason, it is tempting to rely on seat-of-the-pants performance projections, which all too often prove to be significantly in error. The consequences can be serious, for performance, like reliability, is best designed in, rather than added on.

Recently, progress has been made in evolving a general framework for projecting the performance of proposed systems. There has been a confluence of ideas from software engineering and performance evaluation, with queueing network models playing a central role.

3.3.2. Background

User satisfaction with a new application system depends to a significant extent on the system's ability to deliver performance that is acceptable and consistent. In this section we describe some early attempts at assessing the performance of large systems during the design stage.

In the mid 1960s GECOS III was being designed by General Electric as an integrated batch and timesharing system. After the initial design was complete, two activities began in parallel: one team began the implementation, while another developed a simulation model to project the effects of subsequent design and implementation decisions.

The simulation modelling team came out second best. The model was not debugged until several months after a skeletal version of the actual system was operational. Thus, many of the design questions that might have been answered by the model were answered instead by the system. The model could not be kept current. The projections of the model were not trusted, because the system designers lacked confidence in the simulation methodology. This attempt to understand the interactions among design decisions throughout the project lifetime failed. Other attempts have been more successful.

In the late 1960s TSO was being developed as a timesharing subsystem for IBM's batch-oriented MVT operating system. During final design and initial implementation of the final system, an earlier prototype was measured in a test environment, and a queueing network model was parameterized from these measurements and from detailed specifications of the final design.

The average response time projected by the model was- significantly lower than that measured for prototype. However, the design team had confidence in the model because a similar one had been used successfully for MIT's CTSS system. The team checked the prototype for conformance with specifications and detected a discrepancy: the scheduler had been implemented with an unnecessary locking mechanism that created a software bottleneck. When this was corrected, the projections of the model and the behavior of the prototype were compatible.

In the early 1970s MVS was being designed and developed as a batch-oriented operating system for IBM's new family of virtual memory machines. A simulation model was developed for an early version of this system, OS/VS2 Release 2. The model's purpose was to provide performance information for system designers. Model validation was a problem. In the design stage, key model parameters were represented only as ranges of values. Performance projections were checked for reasonableness, to ensure that the model represented the functional flow of work through the system. This type of sensitivity analysis compensated for the lack of precise parameter values.

The system was changing constantly during design and implementation. To reduce this problem, the model builders maintained a close working relationship with the system designers and implementors. This modelling effort was considered to be a success, because several of its recommendations had direct, beneficial effects on the design of the system.

In the mid 1970s the Advanced Logistics System (ALS) was under development for the U.S. Air Force. After the design was completed, during initial implementation, a modelling study was undertaken to determine the bottlenecks in the design and to recommend alternate designs yielding better performance. Hierarchical modelling, as described in Chapter 8, was applied. Four major subsystems were identified in ALS: CPU and memory, system disks, database disks, and tapes. A hierarchical model was structured along these lines, dividing the modelling task into manageable components. Parameter values came from a combination of measurements and detailed specifications.

Both analytic and simulation solutions of the model were obtained. Most ALS features could be captured in the analytic solution. Simulation was used to validate the analytic results and to explore certain system characteristics in more detail. The modelling study predicted that as the workload. increased, the first bottleneck would be encountered in the system disk subsystem, and the next in the CPU and memory subsystem. Both predictions were verified in early production operation, Thus, the study was judged a success.

Each successful project that we have described used a different underlying approach: an analytic model for TSO, a-simulation model for MVS, and hierarchical analytic and simulation models for ALS. However, these projects shared a number of underlying principles.

3.3.3. A General Framework

Unfortunately, it is not common to attempt to quantify the performance of the systems. There are two major reasons for this:

- Manpower devoted to performance projection is viewed as manpower that otherwise could be devoted to writing code and delivering the system on time.
- There is no widely accepted approach to integrating performance projections with a system design project.

The first of these points is interpreted invalid by the false sense of economy on which it is based: the implications of misguided design decisions for the ultimate cost of a system

can be enormous. The second of these points is becoming less significant as aspects of a general framework begin to emerge.

3.3.3.1. The Methodology

Performance is not the domain of a single group. Thus, performance projection is best done in a team environment, with representation from groups such as intended users, software designers, software implementors, configuration planners, and performance analysts. By analogy to software engineering, the team would begin its task by conducting a performance walkthrough of a future design. A typical walkthrough would consist of the following steps:

- The intended users would describe anticipated patterns of use of the system. In queueing network modelling terms, they would identify the workload components, and the workload intensities of the various components.
- The software designers would identify, for a selected subset of the workload components, the path through the software modules of the system that would be followed in processing each component: which modules would be invoked, and how frequently.
- The software implementors would specify the resource requirements for each module in system-independent terms: software path lengths, I/O volume, etc.
- The configuration planners would translate these systemindependent resource requirements into configuration-dependent terms.
- The performance analysts would synthesize the results of this process, constructing a queueing network model of the system.

Various parts of this process would be repeated as the performance analysts seek additional information, as the design evolves, and as the results of the analysis indicate specific areas of concern. An important aspect of any tool embodying this methodology is the support that it provides for this sort of iteration and successive refinement.

It should be clear that what has been outlined is a methodical approach to obtaining queueing network model inputs, method that could be of value in any modelling study, not just an evaluation of a suggested system.

3.3.3.2. Other Considerations

The design stage of a suggested system has received most of our attention. This is where the greatest leverage exists to change plans. However, it is important to continue

the performance projection effort during the life of the project. Implementation, testing, and maintenance/evolution follow design.

Given the desirability of tracking performance over the software lifetime, it is useful to maintain a repository of current information about important aspects of the project. If the repository is automated in database form, software designers and implementors are more likely to keep it current. Budgeting time and manpower for performance projection may lengthen the development schedule somewhat, but the benefits can be significant. A final important factor is the ability to turn this general framework into specific working strategies.

Chapter IV

Experimental study

4.1. General Trends in the Use of ICT in Education

As the world is gradually becoming more and more connected, special emphasis is put on the use and integration of online resources.

- Generally, full integration of ICT in education is still very rare. Highly interactive multimedia or hypermedia are not yet widely used. Online activities involving an intranet or the Internet are used for information and communication purposes rather than tools for interactive education.
- New, mixed modes of learning are emerging: Face-to-face and online learning activities, lectures, videos, multimedia and telecommunication tools support the various learning processes, sometimes in a hybrid manner and sometimes in a more integrated manner.
- Distance education is now being delivered in two different ways, namely in a synchronous mode where participants are using ICTs to communicate at the same time and in an asynchronous mode where participants are learning/communicating independently, ie at different times whenever they are online (anytime-anywhere learning).

It was found that face-to-face meetings or synchronous interaction in real time are still required to supplement asynchronous and independent learning if more effective learning is to take place. ICTs facilitate a high level of interaction among students, the instructor, and the computer-mediated material. Communication can be dynamic and as variable as the teachers and students desire, and it can take place through a variety of modes, such as e-mail, chat, bulletin board, and video conferencing.

- ICTs have become a driving force of educational reform and they are an integrative part of national education policies and plans.

Growing evidence shows how more and more countries have started to equip their schools with computers to achieve school reform or school improvement efforts or even to give their schools a semblance of being modern and technologised. However, at this

point, many educators who see online technology as an enabler of new teaching, learning, and governance practice, may only have scarce information on the potential and authentic use of ICT in education.

The introduction of technology in school undergoes three phases, namely a substitution phase where traditional practices still occur but new technologies are used; a transition phase where new practices begin to appear and well-established practices are being questioned; and a transformation phase where technologies enable new practices and some old ones become obsolete. If educators insist on using ICT as substitute for current practices, they may not contribute to solving the educational problems they are encountering now.

- The introduction of ICTs in schools has brought about a more positive attitude to school among learners. Since ICT and web-based learning offers greater diversity of learning goals, projects, activities, and exercises than traditional classroom offerings, student interest and motivation have increased substantially.

Teachers and students are stimulated because teaching becomes more dynamic which expands their vision as well as access to high quality materials and educational software. Moreover, teachers seem to be motivated to teach more creatively. Portals link teachers to an array of lesson plans, teacher guides, and student exercises that are posted on the Internet by government agencies, NGOs, and educational institutions.

- Online classrooms tend to be more successful if ICT is combined with an appropriate pedagogy. The educational arena of online learning is still in its infancy. While there are many institutions that offer online courses, in-depth understanding of the pedagogical issues related to online education remains unexplored.

Many online courses are nothing but web pages combined with e-mail and chat rooms without any pedagogical foundation. There has been a decrease in teacher-led activities as well as a decrease in the amount of frontal instruction and a move toward more project activities and independent learning as a result of ICT use.

- Online learning enables learners to have more control over educational content and activities. Online environments put the learner at the centre of the educational experience. In traditional teaching, repetitions are used frequently by presenting very similar information in different forms or by asking the same question worded differently. Many learners do not like repetitive exercises.

The Internet encourages learners to dig for information and practical examples by themselves. Hypermedia and multimedia facilitate an array of approaches that have never been possible in traditional teaching and learning.

The Internet promotes an alternative type of learning by doing where students are asked to undertake projects that are related to real life situations. Technology delivers information with emphasis on active creation and exploration of knowledge rather than one-way information transfer, which allows the learner to make full use of their own multiple cognitive abilities.

- The interactive feature of learning resources enables learners to become increasingly engaged in the construction of content and thus contribute to a more authentic learning situation. For instance, students can access virtual libraries worldwide. Thus they have access to vast amounts of information and resources that are unattainable in a single instructional setting.

As far as teachers are concerned, a wealth of teaching resources posted on the Internet everyday has helped teachers handle day-to-day teaching challenges. Teachers can exchange lesson plans, pedagogical techniques, and strategies dealing with issues and common problems.

- Online learning provides built-in technical tools that make learning easier. For example, the language used for searching information and materials are intuitive and immediate. It does not have to be learnt by the user and can be adopted with minimal effort. Basic syntax and grammar can be used as instruments for navigation and retrieval of information.

Integration of communication and authoring tools, along with the click-to-connect interface has succeeded in significantly streamlining the process of checking email, accessing shared data, and setting up conferencing connections. Simulations or visualization technologies are likely to help students to learn complex systems in more concrete ways. Computer mediated communication chat (CMC) and bulletin board tools can supplement face to face sessions.

- The education and training of educators now includes just-in-time and collaborative learning. ICTs open a whole world of lifelong upgrading through distance education, asynchronous learning, and training on demand. ICTs are flexible enough to introduce new courses in direct response to emerging demands.

- ICTs help to break the professional isolation from which many teachers suffer. With ICT, they can easily connect with other professionals, colleagues, and mentors, with universities and centres of expertise, and with sources of teaching materials. Teachers are now publishing their instructional materials on the Internet and sharing their successful teaching practices with other teachers.
- The use of networked computers to promote group learning activities is becoming more and more popular. Computer technologies in education are moving from individualized self-learning to distance-delivered group learning methods. Using computer-mediated communication tools, and web-based group space, students can apply knowledge by combining their efforts to construct a resource activity or project. Cooperative learning through computers has positive effects on group task performance, individual achievement, and attitudes towards collaborative learning.
- Universities are entering into partnerships with the private sector, particularly the IT industry, in order to help maintain operation and financial viability of ICT-based education programmes. More and more schools are realising that linking with the business sector will not necessarily threaten the school systems. Others see an advantage in capitalising on their education services and products. Learning alliances in the delivery of products can offer multiple benefits, such as reduced training development costs, shared R&D costs, or shared organisational content databases and libraries.

ICTs are altering the function of libraries and are intrinsically changing the role of librarians. Schools need not continue to suffer from a lack of library support from isolation from a wealth of learning resources that are readily available on the Internet.

4.2. The Status of e-education

A growing number of organizations are now delivering training and education over the Internet, including colleges and universitis, corporations, military institutions, and even secondary schools.

There are an estimated ten million courses online, and the U.S. alone reports about 700 e-learning companies. Some companies or institutions offer online tutoring to students at specific grade levels, ranging from primary through university; others offer courses only for corporations; some offer courses for individuals in career development

and/or personal development; and many offer training in various management, finance and ICT-related skills. Increasingly, training and support for teachers is occurring online.

4.2.1. General Objectives of e-Education in Myanmar

The general objectives of initiating e-Education in Myanmar are outlined below:

1. To create an academic environment that is endowed with dynamic knowledge and utilizing the technology that will emerge according to the times.
2. To realize the transformation of the working force into a learning force
3. To transform Myanmar into a knowledge dominated society
4. To strive for Myanmar society to become a learning society
5. To raise Myanmar education to international standard

4.2.2. Specific Objectives of e-Education

The specific objectives of initiating e-Education in Myanmar are:

1. To promote a more efficient and effective education
2. To facilitate the emergence of a more accessible education
3. To increase greater public access to education-related information
4. To develop procedures that are more citizen-centric.

4.2.3. Implementation of e-Education in Myanmar

As an endeavour to enhance learning opportunities that transcend the limitations of place and time, e-Education was launched in Myanmar on 1 January 2001. By harnessing information and communication technologies, learning opportunities are being widened for citizens wherever they may be located in the country, regardless of age, sex, experience and educational qualifications thereby facilitating Myanmar to become a learning society able to face the challenges posed by the Knowledge Age.

Information and communication technology, which is changing and developing with the times is being harnessed as a teaching technology and learning technology, and an academic environment that is endowed with dynamic knowledge is being created by utilizing the technology that is emerging in accordance with the times in order to be in line with the education vision of creating an education system that will generate a learning society capable of facing the challenges of the Knowledge Age. The following

programmes are being implemented in the higher education sub-sector for the harnessing of technology:

1. Enhancement of skill in communication technology and electronic technology, with priority given at all levels of undergraduate and postgraduate studies, especially in continuing education programmes.
2. Inclusion of microtechnology from the undergraduate level to give priority to carrying out Transformation of libraries in higher education institutions into Electronic Information Resource Centres.
3. Establishment of Electronic Resource Centres as Myanmar Electronic Information Network (MEIR_Net).
4. Establishment of the Myanmar Higher Education Network (MHE_Net) that will link all the higher education institutions with the Internet.

A variety of measures has been undertaken for the realization of the programmes initiated to harness technology to provide better educational services. One of these is the drive to make all graduates of higher education technology-literate. 50 hours of computer training per academic year is being provided to all undergraduate students and upon graduation, students would have acquired 150 hours of computer training. The number of ICT programmes at both graduate and postgraduate levels are being increased and offered in the form of formal courses as well as that of Human Resource Development Programmes and are open to student learners as well as teaching and administrative staff.

Higher education institution libraries are being modernized using ICT to transform them into Electronic Information Resource Centres with the long-term aim of making the services accessible to the user from his/her home. As a first step, fibre optic LAN has been installed connecting the academic departments at Yangon University and the Universities' Central Library where Intranet services are available.

The Resource Centre for Ancient Myanmar Manuscripts in Yangon is digitizing and scanning Myanmar ancient manuscripts, parabeiks, rare books and stone inscriptions to make ancient Myanmar manuscripts, more accessible to researchers.

As part of the programme to link Myanmar Higher Education Network with international education networks, a number of steps have been taken. Higher education institutions and their academic departments have been connected with servers at the respective Departments of Higher Education. Academic departments have also constructed databases and websites to provide information about the courses and services

they offer and to contribute to the construction of the MEIR_Net. Internet and e-mail service have also been made available to departments and higher education institutions under the Ministry of Education, facilitating linkages with institutions within the country as well as with international education networks. Provision of access to the Internet will have a major impact on researchers and teachers at higher education institutions who will now be able to gain information from anywhere anytime in the world and further improve the quality of their research and teaching as well as make their findings known to the world.

A computerization system has also been introduced in the management of higher education for the systematic storage, retrieval and utilization of higher education management data and information to improve and expedite services.

4.3. MOE Networks

With a population of over 50 millions in Myanmar, dissemination of computer knowledge to the vast majority of the student population is one of the steps which some of our neighbouring countries have even yet to complete.

There are a total of 66 Colleges, Degree Colleges and Universities and Institutes under the Ministry of Education. The Ministry of Education is implementing the Education WAN System in Myanmar. The achievements or results expected of this project are for the free flow and exchange of information first within each University, College or Institute and then within all the institutions in the Ministry of Education. This will enhance their performance as well as facilitate and promote better planning, co-ordination and control at the Ministerial level. With the access to the Worldwide Web through the Internet, free flow of knowledge to and fro from international education as well as other communities is also expected with immense potential benefits to Myanma's education.

VSAT System

System known as Very Small Aperture Terminal (VSAT) is used in satellite communication system. The VSAT links the computer to the satellite. A computer network based on the VSAT system can be established and this network can be utilized to relay not only printed but also audio-visual information. Hence the VSAT can be

exploited for both interactive teaching and video conferencing. Since it can be utilized without requiring any special training, it is a very effective and uncomplicated system.

The MOE Intranet, utilizing VSAT was established at the Ministry of Education in October 2001. The first stage involved the establishment of the intranet system with the use of forty VSAT Systems installed at universities, degree colleges, colleges, departments, and office of the Minister, the Ministry of Education. Within the intranet system of the Ministry of Education, Yangon University, Yangon Institute of Economics, Universities Historical Research Centre, Department of Higher Education (Lower Myanmar) and Myanmar Education Research Bureau are linked with LAN, making use of fibre optic cable. The installation of the VSAT System has enabled the Ministry of Education to form a multimedia society for the Higher Education Sector that is in tandem with the 21st century society. Central to the VSAT network of the Ministry of Education is Bagan Teleport, which is responsible for the administration and monitoring of the intranet system of the Ministry.

Every university, college and department has now been installed with a VSAT together with a computer server. Through the use of the server, the connection among the network of universities, degree colleges, colleges, and departments are being expanded and exchange of information and knowledge is enhanced.

The data broadcasting system previously installed at e-Education Learning Centres is a one-way system. During live transmission and discussions the telephone was the only means to clarify what is unclear to viewers. With the introduction of VSAT at universities, degree colleges, colleges and higher education departments, the two systems can be linked enabling the upgrading of the one-way system into an interactive system.

The VSAT Network has the capability to provide the following services:

1. Interactive classes, joint classes and seminars.
2. Study groups, study meetings, and administrative meetings.
3. Symposiums and conferences.

In FY 2001-2002, the Ministry of Education set up 40 VSAT systems at the following locations and they are now being used for both management and teaching purposes. In FY 2002-2003, the Ministry of Education extend 4 VSAT systems at new estiblish universities and they are in process.

4.4. Models of Computation

The particular model of computation can influence the choice and design of a load distribution scheme, and as such, requires clarification. In order to place distributed computation in its correct context, a short section on the centralised model is included. The centralised system consists of a set of terminals connected to a single central computer. The central computer is typically a timesharing system which switches frequently between processes to provide a share of the system resources to each. In this way, the timesharing system can provide an interactive environment for all users. The major drawback of this scheme is the need for a very powerful central computer. The primary advantage is that the entire capacity of the central computer is available.

4.4.1. Feature of Networks

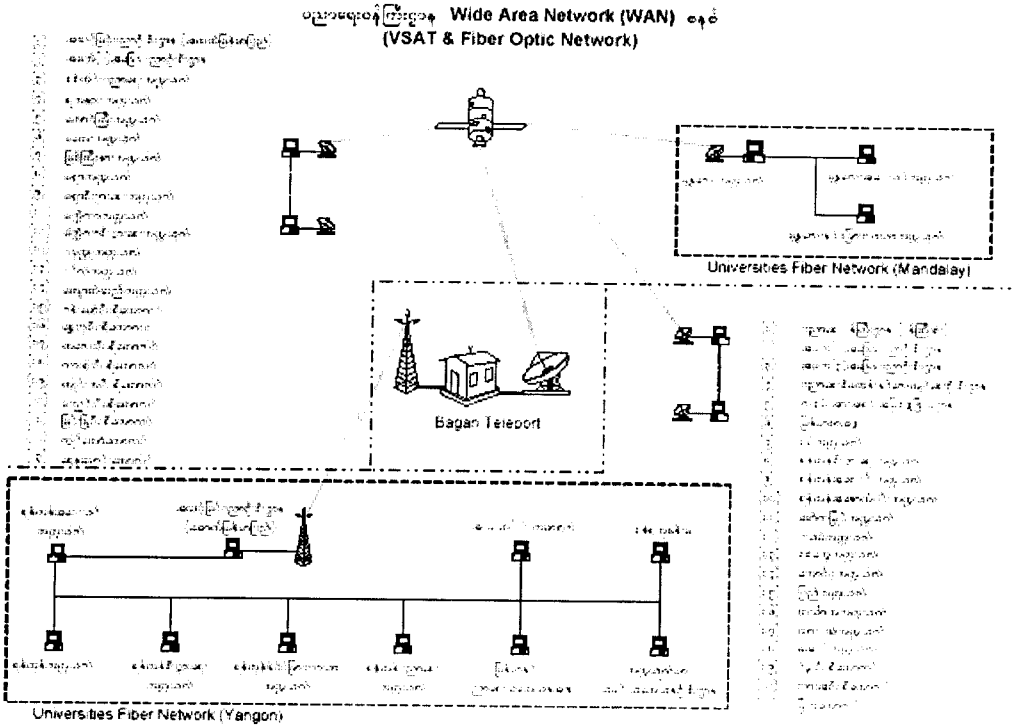


Figure 4.1. MOE WAN

MOE WAN consists of two fibre backbones, VSAT terminals, Radio Links, IPStar terminals and dial-up connections. For Internet browsing, central Web server is located at Bagan Cyber. Web browsing request link through VSAT and download requested web pages direct from MOE Internet Server located at Bagan Cyber. Even though, each bandwidth of VSAT are not large enough, the congestion is not large because of very few numbers of Internet users from the Universities and Colleges. One fibre backbone LAN is located at Mandalay and only three Universities are connected. Another fibre backbone LAN is located at Yangon and there are five Universities and Deaprtment of Higher Education (Lower Myanmar) connected. Main Internet Security and Acceleration (ISA) Servers was located at Department of Higher Education (Lower Myanmar) and link with Bagan Cyber by way of radio link and VSAT link.

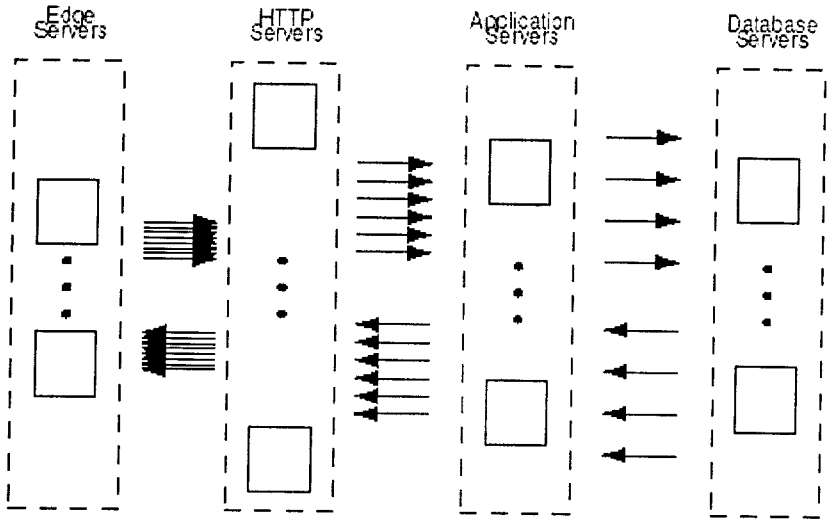


Figure 4.2 Multi-tiered DHE(L) Network system

Figure 4.2. displays the general structure of DHE(L) Network system such as those that provide on-line storefronts on the Internet. End-to-end service levels (e.g., response times) depend on the flow of requests across multiple tiers of servers, each of which has its own complex structure. The system is organized into multiple groups of servers, called tiers. The first tier, the Edge Servers, accept in-coming requests and routes them to the Hypertext Transfer Protocol (HTTP) Servers (the second tier) where requests are interpreted. Some fraction of these requests require more sophisticated processing and so are forwarded to an Application Server (the third tier). The programs that execute here may require access to structured data in a Database Server (the fourth tier). Each software element (e.g., Edge Server, HTTP Server, Application Server, Database Server) has a complex structure.

4.4.2. DHE(L) Fibre Network

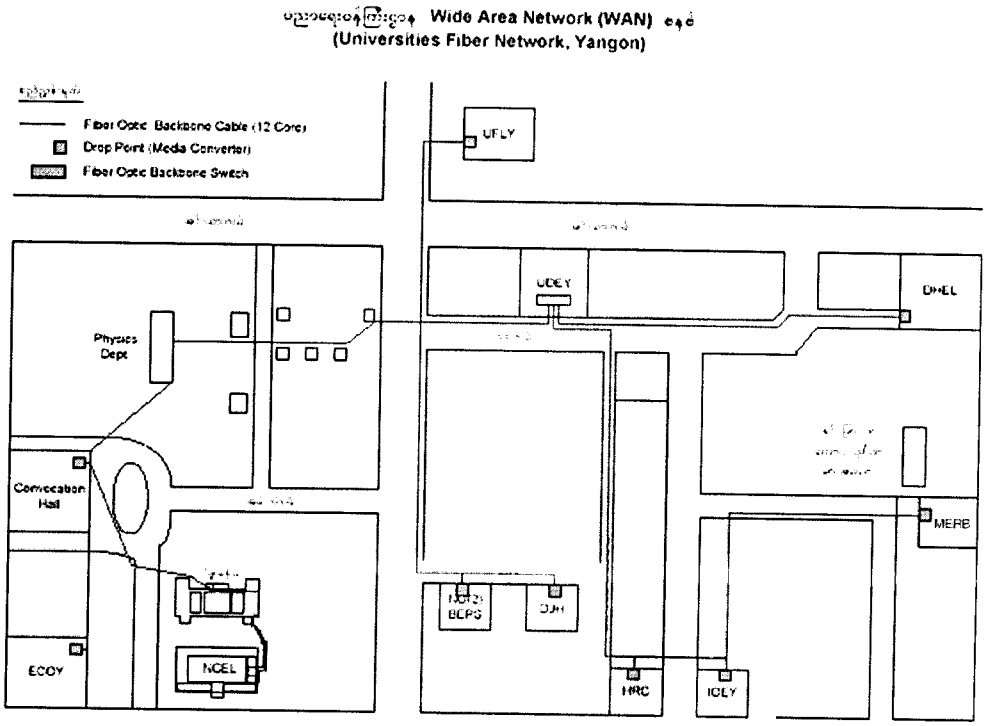


Figure 4.3 Department of Higher Education(Lower Myanmar) Fibre Backbone LAN.

Deaprtment of Higher Education (Lower Myanmar) is conected with five Universities, Myanmar Education Research Bereau (MERB), Historical Research Centre, National Centre for English Language and Diamond Jublee Hall. Each University has their own LAN system . Department of Higher Education (Lower Myanmar) Fibre Backbone LAN is of Xterminal type LAN system.

4.4.3. Gathering measurements

Sr.	Organization	No. of Server	No. of Client	MOE		Internet	
				E-Mail	Web	E-Mail	Web
1	Yangon University	2	50	26	50	26	50
2	University of Distance Education Yangon	1	14	2	14	3	14
3	Institute of Economics Yangon	2	29	2	29	4	29
4	Institute of Education Yangon	1	7	1	7	3	7
5	University of Foreign Language Yangon	1	4	3	4	3	4
	Total	7	104	34	104	39	104

Table 4.4.1 DHE(L) Network's Site List and Usage (by using Fiber Optic Network)

4.4.3.1. Network Design

The Xterminal-Server model is a hybrid of the workstation and timesharing models, as shown in figure 4.4.

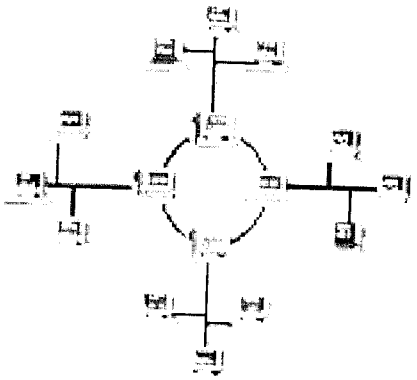


Figure 4.4: ***The Xterminal-Server model consisting of a set of compute servers each of which is responsible for a set of Xterminals.***

This is the model of distributed system in use at the Computer Science Department at Victoria University, and it is this system from which all workload traces were recorded. As a consequence, it is also the model used in the testbed simulation, and in the design of the load distribution policies.

MOE Network is the Xterminal-Server model design and it is a hybrid of each University's LAN system. This is the model of distributed system in use at the DHE(L) Network, and this system from which all workload traces were recorded. As a consequence, it is also the model used in the workloads simulation, and in the design of the load distribution policies. Currently, Internet browsing and e-mail service are the main jobs for the Fibre Optic LAN. Database usage is still scarce in the Ministry of Education. Downloading the teaching materials, data for research requirements and news are the major usages. Job flow for the DHE(L) is as following Figure 4.5.

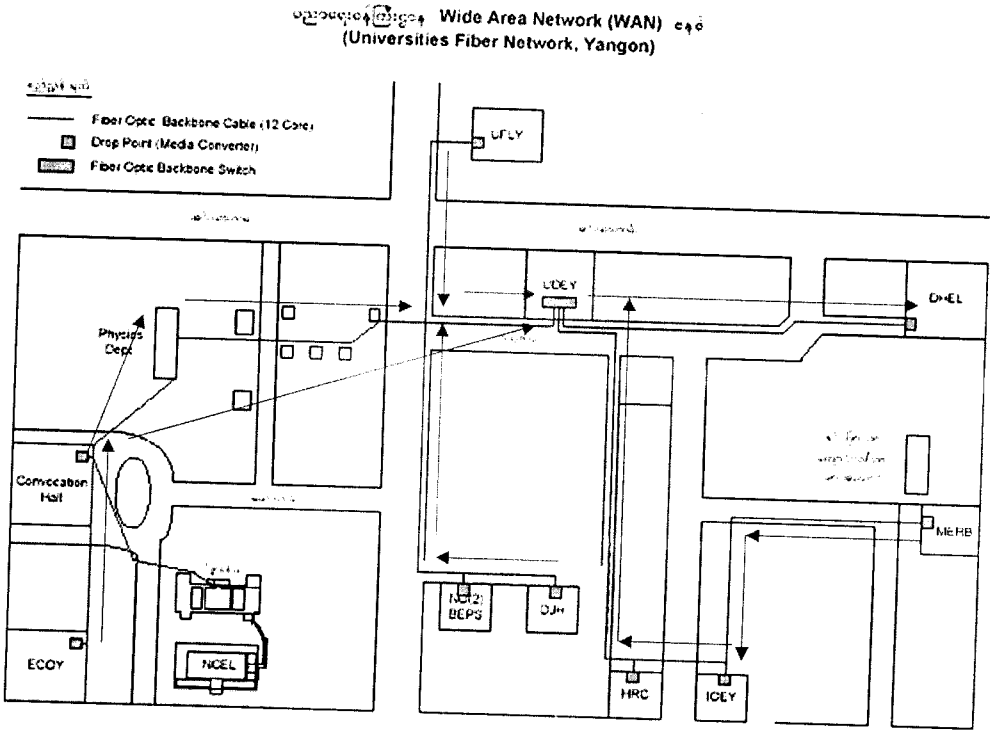


Figure 4.5. Job flow for the DHE(L) Network

According to the configuration, Internet Security and Acceleration (ISA) Server computer at DHE(L) was directly connected by the clients of the Universities. Normally, every request was forward to Web cache server at Bagan Cyber. ISA Server maintains a centralized cache of frequently requested Internet content that can be accessed by any Web browser in our network. More than hundred users can access to Web browsing but, not more than sixty percent of the clients use Web browsing at the crowded hours. According to the increasing requests for Web browsing, we expect increasing demand will become three to five times. For simulation of network performance we use open source simulation programs of *Graphical Spreadsheet Queueing Simulation*, by Armann

Ingolfsson and Tom Grossman, published in volume 2, number 2 of INFORMS Transactions on Education. These spreadsheet queueing templates (or "queueing engines") are simple models of queues with 1 to 12 servers, including queues with balking, reneging, or both. All the programs are written in Visual Basic. By using these open source queueing engines current utilization of ISA Server and response to clients are calculated in Appendix Table (A-1) through Appendix Table(A-8) according to the assumptions. If the demand increase three time the utilization of ISA Server and response to clients are also calculated. Very simply, we can imagine that if the demand increase there will be more waiting time or delay times for Web browsing. Our suggestion is that the current ISA Server is not feasible for increasing users. Not to be very crowded or not to be too much waiting time we need to find the solution. As mentioned before we can improve our hardwares to run efficiently. But, we need more expenses.

4.4.3.2. Proposed Design

According to networking technologies, we want to propose the new network design model for increasing users without any expenses for hardwares. That is to build ISA Server at each University and grouped together in arrays. Computers running ISA Servers can be grouped together in arrays. In ISA Server, an array is a group of ISA Server computers used to provide fault tolerance, loading balancing, and distributed caching. Arrays allow a group of ISA Server computers to be treated and managed as a single, logical entity. An array installation also means increased performance and bandwidth savings. Grouping each university ISA Server computers in an array allows the client requests to be distributed among multiple ISA Servers computers, thereby improving response time for clients. Because load is distributed across all the servers in the array, we can achieve improved performance even with current moderate hardware.

When the internet boowsing request come in, the ISA Server forward the request to Web cache server. ISA Server maintains a centralized cache of frequently requested Internet content that can be accessed by any Web browser in our network. This improves clients' browsers performance, decrease response time, and reduces bandwidth consumption on internet connections (Microsoft 2001).

ISA Server can act as a Web server by fulfilling incoming client requests for Web content from its cache and forwarding requests to the Web server only when the requests cannot be served from its cache.

ISA Server extends our caching performance with a customizable cache download feature. By using the ISA Server Scheduled Content Download feature, everybody can download the HTTP content directly to the ISA Server cache, either upon request or as scheduled. We can update the ISA Server cache with HTTP content that we anticipate will be requested by clients in our University. This content is then available for access directly from the ISA Server cache, rather than from the Internet.

We can download a single URL, multiple URLs, or an entire Web site. When we schedule a cache content download job, we can limit which content should be downloaded, for example by limiting the download to a single domain or a certain number of links to be followed. We can also limit the download to text content only. When we schedule content download, we can configured dynamic content to be cached by configuring the ISA Server cache to store the objects, even if the HTTP cache control header indicate that they are not necessarily cacheable. The download occurs according to a preconfigured and optionally, recurring schedule.

Scheduled content download jobs are also configurable for outgoing Web requests and for incoming Web request. For outgoing Web requests, we determine which objects on the Internet users most often request. We then schedule jobs that retreive the objects from the Internet and Load them into the cache. For incoming Web request, we can schedule content download jobs that will retrieve content from our Internet Web servers and maintain the content in the ISA cache.

ISA Server can be configured to automatically update objects in a cache. With active caching enable, ISA Server analyzes objects that are in the cache to determine which are most frequently accessed. When popular objects in the cache get ready to expire, ISA Server automatically refreshes the content in the cache.

Active caching is a way to keep objects fresh in the cache but verifying them with the origin Web server before the object actually expires and is accessed by a client. The goal is to speed up those client accesses that would normally require a round trip to the origin server to revalidate the data. Because this involves some expense (in both proxy processing and network bandwidth), the goal is to refresh only the objects that are likely to be accessed in the future by the client.

With ISA Server we can support chained, or hierarchical, caching. The term chaining refers to hierarchical connection between individual ISa Server computers, or arrays, of ISA Server computers. Our client requests are sent upstream through the chain

of cache servers until the requested object is found. When the object is located on a upstream server, it is cached at every server's cache, until the object is returned to the client. Chaining is an effective means of distributing server load and fault tolerance.

So, there will be six ISA Server in DHE(L) fibre backbone LAN system. When there is no change in the numbers of Internet users, the utilization of ISA Servers and response to clients are calculated in Appendix Table (A-3) according to the assumptions by using the open source queueing simulation engines. Assuming that there is change in the numbers of Internet users, the utilization of ISA Servers and response to clients are calculated in Appendix Table (A-5) through Table (A-8). The same computations were carried out for an array, a group of ISA Server computers used to provide fault tolerance, loading balancing, and distributed caching.

4.4.3.3. Input Parameters

As mentioned above there are 104 client terminal users in the system. Based on the users' record for the year 2004, we tried to find the pattern of request arrivals and the request completions. We found out that average request arrival time is three seconds in normal time and it is less than one second between 11:00 A.M. and 3:00 P.M. Also, We found out that the average completion time is less than five seconds. Minimum request completion time is one second and maximum request completion time is more than 180 seconds. This information indicates that the network is quite busy between 11:00 A.M. and 3:00 P.M. We expect that number of users will increase up to three times in this network. Then, waiting time will also increase. Therefore, simulation is made under four conditions. For Case I, we assumed that request arrival is as small as normal time and also workloads are small. For Case II, we assumed that request arrival is as small as normal time and workloads are larger. Case III, we assumed that request arrival is three times of normal condition and workloads are small. Case IV, we assumed that request arrival is three times the normal condition and workloads are larger. Simulation is made for one ISA Server and for the proposed six array ISA Servers. Assumptions of Poisson arrivals, exponentially distributed interarrival times are used. For the open source queueing simulation engines we used the following interarrival and service time probability distributions to represent four possible cases.

Case I: Current condition with small file-size downloading.

Interarrival Time Probability Distribution

Probability	Lower Bound	Upper Bound	Interarrival Time (sec)
0.45	0	0.45	3
0.25	0.45	0.7	9
0.1	0.7	0.8	15
0.2	0.8	1	30

Service Time Probability Distribution

Probability	Lower Bound	Upper Bound	Service Time (sec)
0.5	0	0.5	3
0.35	0.5	0.85	6
0.15	0.85	1	9

Case II: Current condition with large file-size downloading.

Interarrival Time Probability Distribution

Probability	Lower Bound	Upper Bound	Interarrival Time (sec)
0.45	0	0.45	3
0.25	0.45	0.7	9
0.1	0.7	0.8	15
0.2	0.8	1	30

Service Time Probability Distribution

Probability	Lower Bound	Upper Bound	Service Time (sec)
0.5	0	0.5	30
0.35	0.5	0.85	60
0.15	0.85	1	90

Case III: Increasing Users with small file-size downloading.

Interarrival Time Probability Distribution

Probability	Lower Bound	Upper Bound	Interarrival Time (sec)
0.45	0	0.45	1
0.25	0.45	0.7	3
0.1	0.7	0.8	5
0.2	0.8	1	10

Service Time Probability Distribution

Probability	Lower Bound	Upper Bound	Service Time (sec)
0.5	0	0.5	3
0.35	0.5	0.85	6
0.15	0.85	1	9

Case IV: Increasing Users with large file-size downloading.

Interarrival Time Probability Distribution

Probability	Lower Bound	Upper Bound	Interarrival Time (sec)
0.45	0	0.45	1
0.25	0.45	0.7	3
0.1	0.7	0.8	5
0.2	0.8	1	10

Service Time Probability Distribution

Probability	Lower Bound	Upper Bound	Service Time (sec)
0.5	0	0.5	30
0.35	0.5	0.85	60
0.15	0.85	1	90

4.4.3.4. Simulation Model Results

Simulation results under the queueing network model with respective assumptions for request arrivals and request completions are shown in Appendix Table (A-1) to (A-4).

The average waiting times are found to be 2 seconds in Case I, 1 hour 43 minutes and 23 seconds in Case II, 10 minutes and 40 seconds in Case III and 6 hours 10 minutes and 24 seconds in Case IV, respectively.

These results are for one ISA Server. The results for the proposed six arrayed ISA Servers are shown in Appendix Table (B-1) to (B-4). The average waiting times are 0 second in Case I, 17 seconds in Case II, 0 seconds in Case III and 36 minutes and 59 seconds in Case IV, respectively.

It can be seen that the proposed system yields much lower average waiting times. So it is much faster than the current system responding to the requests.

4.4.3.5. Additional Investigation of System Performance with the Help of PDQ Solver

We also use Pretty Dam Quick (PDQ) open source queueing model solver and library functions written in C language. This open source programs use specially for analyzing the queueing network's performance. It uses queueing theory paradigms to represent a computer system. Hardware and software resources are represented by queues in PDQ by making calls to the appropriate library functions. PDQ can be solved instantaneously by calling the PDQ_Solve() function. This, in turn, generates a report of all the corresponding performance metrics such as system throughput, and system response time, as well as details about queue lengths and utilizations at each of the defined queues. DHE(L) Network is an open type network. PDQ can solve the service disciplines for (first come first serves) FCFS, (infinite server) ISRV, (last come first serves) LCFS, and (processor sharing) PSHR. We use our queueing model service disciplines of FCFS (first-come first-served). PDQ can also solve the workload streams for batch, terminal and transaction. A batch class workload, which is defined to be one with zero thinktime, is only consistent in the context of a closed queueing circuit to distinguish from terminal class. A terminal class workload, which is defined to be one with non-zero thinktime, is also consistent in the context of a closed queueing circuit to

distinguish from batch. A transaction class workload, which is defined by an arrival rate rather than a thinktime, is consistent in the context of an open queueing circuit. This workload streams are used in open circuit queueing model and different solution methods can be used with PDQ open sources which include the following solution methods: (i) approximate MVA solution technique of solving a closed queueing circuit, (ii) the canonical solution technique consistent in the context of an open queueing circuit and (iii) the iterative MVA (Mean Value Analysis) method for up to three workload classes of a closed queueing circuit. The method of computation used in queueing network modelling is the canonical solution technique. The results obtained from this simulation technique are used to compare current DHE(L) Network performance and proposed DHE(L) Network performance. The results are presented in Table 4.4.2. In calculating the performance of a network, the PDQ uses all the hard-wares' qualities available.

Investigation of performance of the two network systems is carried out with the help of PDQ. The PDQ is not a simulator but a queueing model solver. Accuracy of the PDQ can be concentrated rather than accuracy of the solution. The PDQ is a flexible library function rather than a hard-wired binary applications. However, the flexibility of the PDQ requires the performance model to be expressed in the C language.

The performance of the current network system and proposed network system are compared based on the results obtained from using the PDQ queueing model solver. For this queueing model solver, the required discipline is first-come first-served (FCFS), used with PDQ-cerate mode. There are three workload systems for the PDQ solver. In the current investigation the transaction workload, which is defined by an arrival rate rather than a thinktime, is employed since the transaction workload is consistent in the context of an open queueing circuit. Concerning the solution methods, the canonical solution method is employed in this comparative investigation. The performance of the current and proposed network systems because the canonical solution technique is appropriate and consistent in the context of an open queueing circuit.

Current system		Proposed system	
System Performance			
Mean Throughput	8.3333 Job/Sec	Mean Throughput	0.0667 Job/Sec
Response Time	0.1372 Sec	Response Time	0.1164 Sec
Bounds Analysis:		Bounds Analysis:	
Max Demand	32.5521 Job/Sec	Max Demand	37.6674 Job/Sec
Max Throughput	32.5521 Job/Sec	Max Throughput	37.6674 Job/Sec
Resource Performance			
PC		PC	
Throughput	8.3333 Job/Sec	Throughput	0.0667 Job/Sec
Utilization	22.1235 Percent	Utilization	0.177 Percent
Queue Length	0.2212 Job	Queue Length	0.0018 Job
Residence Time	0.0265 Sec	Residence Time	0.0265 Sec
FS		FS	
Throughput	8.3333 Job/Sec	Throughput	0.0667 Job/Sec
Utilization	4.6829 Percent	Utilization	0.0375 Percent
Queue Length	0.0468 Job	Queue Length	0.0004 Job
Residence Time	0.0056 Sec	Residence Time	0.0056 Sec
GW		GW	
Throughput	8.3333 Job/Sec	Throughput	0.0667 Job/Sec
Utilization	25.6 Percent	Utilization	0.128 Percent
Queue Length	0.256 Job	Queue Length	0.0013 Job
Residence Time	0.0307 Sec	Residence Time	0.0192 Sec
Resource Utilization			
Node	*% Utilization*	*Node*	*% Utilization*
Xterminal	30.9333	Xterminal	24.7467
Desktop PC	0.5942	Desktop PC	0.5942
ISA Server	58.7503	ISA Server	23.5001
Gateway SNA	61.8667	Gateway SNA	77.3333

Table 4.4.2 Performance Comparison between Current and Proposed Systems

The results obtained from the use of the PDQ queueing model solver are presented in Table 4.4.2. In investigating of the performance of the two network systems; with the help of PDQ queueing model solver, it is assumed that (i) the arrival rate of jobs is one job per second and (ii) average service demand for a job is 30 seconds. The performance of the two network systems can be compared from the standpoints of (i) system performance (ii) resource performance and (iii) resource utilization. In the Table 4.4.2.

the abbreviations' PC stands for Personal Computer, FS stands for ISA Server (File Server) and GW stands for the gateway.

All possible outputs described in section 1.4.3. are calculated. The values of these outputs depend upon the values of all of the model inputs of the current network system and proposed network system. Among these outputs, some will be discussed. System response time, *R*, corresponds to our intuitive notion of response time or the interval between submitting a request and receiving a response on an interactive system. Obviously, system response time is the sum of the residence times at the various centers. Average system response time for the proposed network system is 0.0108 seconds less than the current network system. That means the proposed system can more quickly respond to the customers. Also the system throughput of the proposed network system is 8.2 seconds less than the current network system. Because of six arrayed ISA Servers, a job in the queuing network can be more quickly served. Besides, the proposed system yields higher maximum bounds for demand as well as throughput. As to the resource utilization, the proposed system's utilization is 4.65 percent less than the current network system. This is because, the number of ISA Servers is six times larger in the proposed system than the current system and ISA Server can share the workload. Therefore, the proposed system is less busy than the current network system. Also, queue length of the proposed system is much shorter than the current network system. From the user point of view, queue length for proposed network system is much less than the current network system. The current network system uses only one gateway but proposed network system uses six arrayed ISA Servers. Therefore, the gateway utilization of the proposed network system is 25 percent less than the current network system utilization. Because of smaller queue size, the waiting time is shorter and the customer receives the service more quickly. Therefore, the utilization of the PC by a customer for the proposed network system is 22 percent less than the current network system. The results in Table 4.4.2, from the angles of (i) system performance (ii) resource performance and (iii) resource utilization, the proposed network system is clearly better than the current network system.

Chapter V

Conclusion and Future Research Area

5.1. Conclusions

The work in this thesis involves the investigation of networking model, with the intention of extending the efficiency of performance of a computer system. We laboured much in the presentation about queueing network modeling in the first three chapters. The reason is simple: the effectiveness with which such software can be applied is multiplied many times by an understanding of the principles and techniques upon which it is based. First, we discussed about Little's law and its relatives, which provide the technical foundation of queueing network modelling. This knowledge, along with an awareness of the widespread success of performance studies using queueing network models, provides confidence in the approach. Then, we discussed the techniques used to evaluate separable queueing network models, and the assumptions upon which these techniques rely. The robustness of these techniques with respect to the assumptions was shown. This knowledge again builds confidence in the approach and, it indicates the range of applicability of queueing network models. It also provides insight into the ways in which detailed models of specific subsystems can be constructed using separable queueing network models as a basis.

We discussed a collection of techniques to evaluate detailed models of specific subsystems, where representing the effects of system characteristics is often necessary. This knowledge helps to understand the homogeneity assumptions made by queueing network modelling packages in specific cases, so that it can be known if these assumptions should be a source of concern in a particular performance study. We also discussed how to parameterize queueing network models to conduct studies of basic, developing, and proposed systems.

Finally, we discussed how queueing network models can be applied in MOE WAN. As computer systems continue to develop, recognizing that the applicability of queueing network technology is important, and of existing queueing network modelling

software, extends well beyond the confines of centralized systems with simple characteristics.

To repeat some comments made in the introduction, queueing network models, while not a panacea, are the appropriate tool in a wide variety of applications. Computer system analysis using queueing network models is a blend of art and science, requiring both knowledge and experience. By using queueing network models it might be expected that we would successfully contribute to measuring the performance of computer network as a whole.

Network array model applied to DHE(L) Network system was found to be effective for improving the performance of DHE(L) Network system. This model performed in very simple ways, yielding dramatic performance improvements. However, there may be other models using complex softwares that will perform sufficiently better to warrant further consideration. The advantages of considering the available system resources and matching those to the resources required by jobs are clear. In particular, whilst the improvement of performance of communication hardware is not large, it is nonetheless consistent, the additional state and complexity of communication hardware is the minimal.

It has already been stated that the process of design and implementation involves continual tradeoffs between cost and performance. Therefore, performance as well as cost of the system needs to be considered. However, in this study only the performance of the proposed network system was able to be investigated and cost of the system could not be investigated.

5.2. Future Research Area

The results presented in chapter four showed that forming an array of ISA Servers offers advantages over one ISA Server, and therefore reducing the service times. In comparison with more complex load distribution policies such as those using averages and maintaining the process mix constitutes complexity, and it is sufficiently promising to warrant further study. Our simulation engines use Poisson arrivals. Other incoming distributions can be considered for future experiments. The experimental work in this thesis has only considered a Xterminal-Server model with six servers. These results need to be investigated with respect to networks of different types, different sizes and configurations. Networks of different types may respond differently to the maintenance of

the *performance*. This aspect should be explored as a future research area, although it seems that the technique might be successful with a small to medium number of servers.

We applied to the possible completion of jobs within the update period. Instead the system assumes that the update period is sufficiently short to provide a corrected update content from the Web Server. We would like to suggest that the availability of predicted resource would allow longer times for content updates before performance is degraded, but this would require a better means of estimating the inter update load, and was not investigated in detail. Many issues can be considered in the design of computer communication networks: network bandwidth, multiplexing and concentration strategies, network protocol layers, flow control policies, routing strategies, and buffering strategies. Finally, in the future research concerning application of the queueing networking system to MOE network, performance as well as cost should be considered to have an optimal system with a tradeoff between cost and performance.

APPENDIX

Table (A-1)
Queueing Simulation under condition I

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
start		9:00					
1	9	9:00:09	6	9:00:09	9:00:15	0:00:00	0:00:06
2	3	9:00:12	3	9:00:15	9:00:18	0:00:03	0:00:06
3	30	9:00:42	3	9:00:42	9:00:45	0:00:00	0:00:03
4	30	9:01:12	9	9:01:12	9:01:21	0:00:00	0:00:09
5	9	9:01:21	6	9:01:21	9:01:27	0:00:00	0:00:06
6	9	9:01:30	9	9:01:30	9:01:39	0:00:00	0:00:09
7	3	9:01:33	6	9:01:39	9:01:45	0:00:06	0:00:12
8	15	9:01:48	3	9:01:48	9:01:51	0:00:00	0:00:03
9	9	9:01:57	3	9:01:57	9:02:00	0:00:00	0:00:03
10	15	9:02:12	3	9:02:12	9:02:15	0:00:00	0:00:03
11	3	9:02:15	9	9:02:15	9:02:24	0:00:00	0:00:09
12	30	9:02:45	9	9:02:45	9:02:54	0:00:00	0:00:09
13	3	9:02:48	9	9:02:54	9:03:03	0:00:06	0:00:15
14	3	9:02:51	3	9:03:03	9:03:06	0:00:12	0:00:15
15	3	9:02:54	3	9:03:06	9:03:09	0:00:12	0:00:15
16	30	9:03:24	3	9:03:24	9:03:27	0:00:00	0:00:03
17	3	9:03:27	9	9:03:27	9:03:36	0:00:00	0:00:09
18	9	9:03:36	3	9:03:36	9:03:39	0:00:00	0:00:03
19	15	9:03:51	3	9:03:51	9:03:54	0:00:00	0:00:03
20	9	9:04:00	9	9:04:00	9:04:09	0:00:00	0:00:09
21	3	9:04:03	9	9:04:09	9:04:18	0:00:06	0:00:15
22	9	9:04:12	6	9:04:18	9:04:24	0:00:06	0:00:12
23	3	9:04:15	3	9:04:24	9:04:27	0:00:09	0:00:12
24	3	9:04:18	6	9:04:27	9:04:33	0:00:09	0:00:15
25	30	9:04:48	3	9:04:48	9:04:51	0:00:00	0:00:03
26	9	9:04:57	6	9:04:57	9:05:03	0:00:00	0:00:06
27	30	9:05:27	3	9:05:27	9:05:30	0:00:00	0:00:03
28	3	9:05:30	6	9:05:30	9:05:36	0:00:00	0:00:06
29	30	9:06:00	6	9:06:00	9:06:06	0:00:00	0:00:06
30	30	9:06:30	3	9:06:30	9:06:33	0:00:00	0:00:03
31	3	9:06:33	9	9:06:33	9:06:42	0:00:00	0:00:09
32	3	9:06:36	3	9:06:42	9:06:45	0:00:06	0:00:09
33	9	9:06:45	6	9:06:45	9:06:51	0:00:00	0:00:06
34	3	9:06:48	3	9:06:51	9:06:54	0:00:03	0:00:06
35	30	9:07:18	6	9:07:18	9:07:24	0:00:00	0:00:06
36	15	9:07:33	3	9:07:33	9:07:36	0:00:00	0:00:03
37	9	9:07:42	3	9:07:42	9:07:45	0:00:00	0:00:03
38	3	9:07:45	3	9:07:45	9:07:48	0:00:00	0:00:03
39	3	9:07:48	3	9:07:48	9:07:51	0:00:00	0:00:03
40	30	9:08:18	6	9:08:18	9:08:24	0:00:00	0:00:06
41	3	9:08:21	3	9:08:24	9:08:27	0:00:03	0:00:06
42	3	9:08:24	3	9:08:27	9:08:30	0:00:03	0:00:06
43	3	9:08:27	3	9:08:30	9:08:33	0:00:03	0:00:06
44	15	9:08:42	6	9:08:42	9:08:48	0:00:00	0:00:06
45	3	9:08:45	6	9:08:48	9:08:54	0:00:03	0:00:09
46	3	9:08:48	3	9:08:54	9:08:57	0:00:06	0:00:09
47	15	9:09:03	3	9:09:03	9:09:06	0:00:00	0:00:03
48	15	9:09:18	3	9:09:18	9:09:21	0:00:00	0:00:03
49	3	9:09:21	3	9:09:21	9:09:24	0:00:00	0:00:03
50	3	9:09:24	6	9:09:24	9:09:30	0:00:00	0:00:06
51	3	9:09:27	6	9:09:30	9:09:36	0:00:03	0:00:09
52	9	9:09:36	6	9:09:36	9:09:42	0:00:00	0:00:06
53	3	9:09:39	6	9:09:42	9:09:48	0:00:03	0:00:09
54	3	9:09:42	3	9:09:48	9:09:51	0:00:06	0:00:09
55	30	9:10:12	9	9:10:12	9:10:21	0:00:00	0:00:09

Table (A-1)
Queueing Simulation under condition I

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
56	3	9:10:15	3	9:10:21	9:10:24	0:00:06	0:00:09
57	15	9:10:30	3	9:10:30	9:10:33	0:00:00	0:00:03
58	3	9:10:33	6	9:10:33	9:10:39	0:00:00	0:00:06
59	9	9:10:42	3	9:10:42	9:10:45	0:00:00	0:00:03
60	3	9:10:45	3	9:10:45	9:10:48	0:00:00	0:00:03
61	9	9:10:54	9	9:10:54	9:11:03	0:00:00	0:00:09
62	30	9:11:24	3	9:11:24	9:11:27	0:00:00	0:00:03
63	9	9:11:33	3	9:11:33	9:11:36	0:00:00	0:00:03
64	3	9:11:36	3	9:11:36	9:11:39	0:00:00	0:00:03
65	3	9:11:39	9	9:11:39	9:11:48	0:00:00	0:00:09
66	3	9:11:42	6	9:11:48	9:11:54	0:00:06	0:00:12
67	3	9:11:45	3	9:11:54	9:11:57	0:00:09	0:00:12
68	3	9:11:48	3	9:11:57	9:12:00	0:00:09	0:00:12
69	3	9:11:51	3	9:12:00	9:12:03	0:00:09	0:00:12
70	3	9:11:54	6	9:12:03	9:12:09	0:00:09	0:00:15
71	3	9:11:57	3	9:12:09	9:12:12	0:00:12	0:00:15
72	3	9:12:00	3	9:12:12	9:12:15	0:00:12	0:00:15
73	9	9:12:09	3	9:12:15	9:12:18	0:00:06	0:00:09
74	3	9:12:12	6	9:12:18	9:12:24	0:00:06	0:00:12
75	3	9:12:15	9	9:12:24	9:12:33	0:00:09	0:00:18
76	9	9:12:24	3	9:12:33	9:12:36	0:00:09	0:00:12
77	3	9:12:27	3	9:12:36	9:12:39	0:00:09	0:00:12
78	9	9:12:36	6	9:12:39	9:12:45	0:00:03	0:00:09
79	15	9:12:51	6	9:12:51	9:12:57	0:00:00	0:00:06
80	3	9:12:54	6	9:12:57	9:13:03	0:00:03	0:00:09
81	3	9:12:57	3	9:13:03	9:13:06	0:00:06	0:00:09
82	3	9:13:00	3	9:13:06	9:13:09	0:00:06	0:00:09
83	3	9:13:03	3	9:13:09	9:13:12	0:00:06	0:00:09
84	3	9:13:06	9	9:13:12	9:13:21	0:00:06	0:00:15
85	3	9:13:09	3	9:13:21	9:13:24	0:00:12	0:00:15
86	30	9:13:39	3	9:13:39	9:13:42	0:00:00	0:00:03
87	9	9:13:48	6	9:13:48	9:13:54	0:00:00	0:00:06
88	9	9:13:57	6	9:13:57	9:14:03	0:00:00	0:00:06
89	15	9:14:12	3	9:14:12	9:14:15	0:00:00	0:00:03
90	30	9:14:42	9	9:14:42	9:14:51	0:00:00	0:00:09
91	3	9:14:45	3	9:14:51	9:14:54	0:00:06	0:00:09
92	3	9:14:48	3	9:14:54	9:14:57	0:00:06	0:00:09
93	9	9:14:57	9	9:14:57	9:15:06	0:00:00	0:00:09
94	30	9:15:27	3	9:15:27	9:15:30	0:00:00	0:00:03
95	3	9:15:30	3	9:15:30	9:15:33	0:00:00	0:00:03
96	3	9:15:33	6	9:15:33	9:15:39	0:00:00	0:00:06
97	30	9:16:03	3	9:16:03	9:16:06	0:00:00	0:00:03
98	3	9:16:06	6	9:16:06	9:16:12	0:00:00	0:00:06
99	9	9:16:15	6	9:16:15	9:16:21	0:00:00	0:00:06
100	3	9:16:18	3	9:16:21	9:16:24	0:00:03	0:00:06
101	3	9:16:21	3	9:16:24	9:16:27	0:00:03	0:00:06
102	30	9:16:51	3	9:16:51	9:16:54	0:00:00	0:00:03
103	30	9:17:21	6	9:17:21	9:17:27	0:00:00	0:00:06
104	9	9:17:30	3	9:17:30	9:17:33	0:00:00	0:00:03
105	30	9:18:00	9	9:18:00	9:18:09	0:00:00	0:00:09
106	30	9:18:30	3	9:18:30	9:18:33	0:00:00	0:00:03
107	9	9:18:39	6	9:18:39	9:18:45	0:00:00	0:00:06
108	3	9:18:42	3	9:18:45	9:18:48	0:00:03	0:00:06
109	9	9:18:51	6	9:18:51	9:18:57	0:00:00	0:00:06
110	15	9:19:06	9	9:19:06	9:19:15	0:00:00	0:00:09
111	3	9:19:09	6	9:19:15	9:19:21	0:00:06	0:00:12
112	3	9:19:12	9	9:19:21	9:19:30	0:00:09	0:00:18

Table (A-1)
Queueing Simulation under condition I

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
113	3	9:19:15	9	9:19:30	9:19:39	0:00:15	0:00:24
114	9	9:19:24	3	9:19:39	9:19:42	0:00:15	0:00:18
115	3	9:19:27	6	9:19:42	9:19:48	0:00:15	0:00:21
116	30	9:19:57	6	9:19:57	9:20:03	0:00:00	0:00:06
117	30	9:20:27	3	9:20:27	9:20:30	0:00:00	0:00:03
118	30	9:20:57	6	9:20:57	9:21:03	0:00:00	0:00:06
119	9	9:21:06	6	9:21:06	9:21:12	0:00:00	0:00:06
120	9	9:21:15	3	9:21:15	9:21:18	0:00:00	0:00:03
121	9	9:21:24	3	9:21:24	9:21:27	0:00:00	0:00:03
122	30	9:21:54	9	9:21:54	9:22:03	0:00:00	0:00:09
123	3	9:21:57	3	9:22:03	9:22:06	0:00:06	0:00:09
124	30	9:22:27	6	9:22:27	9:22:33	0:00:00	0:00:06
125	30	9:22:57	3	9:22:57	9:23:00	0:00:00	0:00:03
126	30	9:23:27	6	9:23:27	9:23:33	0:00:00	0:00:06
127	3	9:23:30	3	9:23:33	9:23:36	0:00:03	0:00:06
128	30	9:24:00	3	9:24:00	9:24:03	0:00:00	0:00:03
129	3	9:24:03	6	9:24:03	9:24:09	0:00:00	0:00:06
130	9	9:24:12	6	9:24:12	9:24:18	0:00:00	0:00:06
131	3	9:24:15	6	9:24:18	9:24:24	0:00:03	0:00:09
132	3	9:24:18	6	9:24:24	9:24:30	0:00:06	0:00:12
133	15	9:24:33	3	9:24:33	9:24:36	0:00:00	0:00:03
134	3	9:24:36	6	9:24:36	9:24:42	0:00:00	0:00:06
135	3	9:24:39	6	9:24:42	9:24:48	0:00:03	0:00:09
136	3	9:24:42	3	9:24:48	9:24:51	0:00:06	0:00:09
137	30	9:25:12	9	9:25:12	9:25:21	0:00:00	0:00:09
138	3	9:25:15	3	9:25:21	9:25:24	0:00:06	0:00:09
139	3	9:25:18	3	9:25:24	9:25:27	0:00:06	0:00:09
140	3	9:25:21	6	9:25:27	9:25:33	0:00:06	0:00:12
141	30	9:25:51	3	9:25:51	9:25:54	0:00:00	0:00:03
142	9	9:26:00	3	9:26:00	9:26:03	0:00:00	0:00:03
143	30	9:26:30	3	9:26:30	9:26:33	0:00:00	0:00:03
144	3	9:26:33	3	9:26:33	9:26:36	0:00:00	0:00:03
145	3	9:26:36	3	9:26:36	9:26:39	0:00:00	0:00:03
146	3	9:26:39	3	9:26:39	9:26:42	0:00:00	0:00:03
147	3	9:26:42	3	9:26:42	9:26:45	0:00:00	0:00:03
148	3	9:26:45	3	9:26:45	9:26:48	0:00:00	0:00:03
149	9	9:26:54	6	9:26:54	9:27:00	0:00:00	0:00:06
150	3	9:26:57	3	9:27:00	9:27:03	0:00:03	0:00:06
151	30	9:27:27	3	9:27:27	9:27:30	0:00:00	0:00:03
152	9	9:27:36	6	9:27:36	9:27:42	0:00:00	0:00:06
153	30	9:28:06	3	9:28:06	9:28:09	0:00:00	0:00:03
154	30	9:28:36	6	9:28:36	9:28:42	0:00:00	0:00:06
155	3	9:28:39	3	9:28:42	9:28:45	0:00:03	0:00:06
156	3	9:28:42	6	9:28:45	9:28:51	0:00:03	0:00:09
157	30	9:29:12	3	9:29:12	9:29:15	0:00:00	0:00:03
158	30	9:29:42	9	9:29:42	9:29:51	0:00:00	0:00:09
159	9	9:29:51	6	9:29:51	9:29:57	0:00:00	0:00:06
160	3	9:29:54	3	9:29:57	9:30:00	0:00:03	0:00:06
161	30	9:30:24	6	9:30:24	9:30:30	0:00:00	0:00:06
162	3	9:30:27	3	9:30:30	9:30:33	0:00:03	0:00:06
163	30	9:30:57	9	9:30:57	9:31:06	0:00:00	0:00:09
164	9	9:31:06	3	9:31:06	9:31:09	0:00:00	0:00:03
165	3	9:31:09	3	9:31:09	9:31:12	0:00:00	0:00:03
166	3	9:31:12	9	9:31:12	9:31:21	0:00:00	0:00:09
167	9	9:31:21	3	9:31:21	9:31:24	0:00:00	0:00:03
168	9	9:31:30	6	9:31:30	9:31:36	0:00:00	0:00:06
169	9	9:31:39	3	9:31:39	9:31:42	0:00:00	0:00:03

Table (A-1)
Queueing Simulation under condition I

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
170	3	9:31:42	9	9:31:42	9:31:51	0:00:00	0:00:09
171	9	9:31:51	3	9:31:51	9:31:54	0:00:00	0:00:03
172	9	9:32:00	6	9:32:00	9:32:06	0:00:00	0:00:06
173	9	9:32:09	3	9:32:09	9:32:12	0:00:00	0:00:03
174	3	9:32:12	3	9:32:12	9:32:15	0:00:00	0:00:03
175	3	9:32:15	3	9:32:15	9:32:18	0:00:00	0:00:03
176	3	9:32:18	3	9:32:18	9:32:21	0:00:00	0:00:03
177	9	9:32:27	6	9:32:27	9:32:33	0:00:00	0:00:06
178	30	9:32:57	6	9:32:57	9:33:03	0:00:00	0:00:06
179	9	9:33:06	6	9:33:06	9:33:12	0:00:00	0:00:06
180	3	9:33:09	9	9:33:12	9:33:21	0:00:03	0:00:12
181	9	9:33:18	3	9:33:21	9:33:24	0:00:03	0:00:06
182	15	9:33:33	3	9:33:33	9:33:36	0:00:00	0:00:03
183	3	9:33:36	6	9:33:36	9:33:42	0:00:00	0:00:06
184	3	9:33:39	3	9:33:42	9:33:45	0:00:03	0:00:06
185	3	9:33:42	9	9:33:45	9:33:54	0:00:03	0:00:12
186	30	9:34:12	6	9:34:12	9:34:18	0:00:00	0:00:06
187	9	9:34:21	9	9:34:21	9:34:30	0:00:00	0:00:09
188	9	9:34:30	6	9:34:30	9:34:36	0:00:00	0:00:06
189	30	9:35:00	3	9:35:00	9:35:03	0:00:00	0:00:03
190	30	9:35:30	3	9:35:30	9:35:33	0:00:00	0:00:03
191	9	9:35:39	6	9:35:39	9:35:45	0:00:00	0:00:06
192	3	9:35:42	3	9:35:45	9:35:48	0:00:03	0:00:06
193	9	9:35:51	3	9:35:51	9:35:54	0:00:00	0:00:03
194	30	9:36:21	3	9:36:21	9:36:24	0:00:00	0:00:03
195	15	9:36:36	3	9:36:36	9:36:39	0:00:00	0:00:03
196	3	9:36:39	6	9:36:39	9:36:45	0:00:00	0:00:06
197	3	9:36:42	9	9:36:45	9:36:54	0:00:03	0:00:12
198	3	9:36:45	3	9:36:54	9:36:57	0:00:09	0:00:12
199	3	9:36:48	3	9:36:57	9:37:00	0:00:09	0:00:12
200	3	9:36:51	3	9:37:00	9:37:03	0:00:09	0:00:12
201	3	9:36:54	6	9:37:03	9:37:09	0:00:09	0:00:15
202	15	9:37:09	3	9:37:09	9:37:12	0:00:00	0:00:03
203	3	9:37:12	3	9:37:12	9:37:15	0:00:00	0:00:03
204	3	9:37:15	6	9:37:15	9:37:21	0:00:00	0:00:06
205	15	9:37:30	3	9:37:30	9:37:33	0:00:00	0:00:03
206	30	9:38:00	3	9:38:00	9:38:03	0:00:00	0:00:03
207	3	9:38:03	3	9:38:03	9:38:06	0:00:00	0:00:03
208	3	9:38:06	3	9:38:06	9:38:09	0:00:00	0:00:03
209	30	9:38:36	6	9:38:36	9:38:42	0:00:00	0:00:06
210	9	9:38:45	3	9:38:45	9:38:48	0:00:00	0:00:03
211	15	9:39:00	3	9:39:00	9:39:03	0:00:00	0:00:03
212	3	9:39:03	6	9:39:03	9:39:09	0:00:00	0:00:06
213	15	9:39:18	9	9:39:18	9:39:27	0:00:00	0:00:09
214	9	9:39:27	3	9:39:27	9:39:30	0:00:00	0:00:03
215	3	9:39:30	3	9:39:30	9:39:33	0:00:00	0:00:03
216	15	9:39:45	6	9:39:45	9:39:51	0:00:00	0:00:06
217	3	9:39:48	6	9:39:51	9:39:57	0:00:03	0:00:09
218	3	9:39:51	3	9:39:57	9:40:00	0:00:06	0:00:09
219	3	9:39:54	6	9:40:00	9:40:06	0:00:06	0:00:12
220	3	9:39:57	6	9:40:06	9:40:12	0:00:09	0:00:15
221	3	9:40:00	3	9:40:12	9:40:15	0:00:12	0:00:15
222	3	9:40:03	9	9:40:15	9:40:24	0:00:12	0:00:21
223	3	9:40:06	3	9:40:24	9:40:27	0:00:18	0:00:21
224	3	9:40:09	3	9:40:27	9:40:30	0:00:18	0:00:21
225	3	9:40:12	3	9:40:30	9:40:33	0:00:18	0:00:21
226	15	9:40:27	3	9:40:33	9:40:36	0:00:06	0:00:09

Table (A-1)
Queueing Simulation under condition I

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
227	9	9:40:36	6	9:40:36	9:40:42	0:00:00	0:00:06
228	3	9:40:39	3	9:40:42	9:40:45	0:00:03	0:00:06
229	9	9:40:48	3	9:40:48	9:40:51	0:00:00	0:00:03
230	30	9:41:18	3	9:41:18	9:41:21	0:00:00	0:00:03
231	9	9:41:27	3	9:41:27	9:41:30	0:00:00	0:00:03
232	3	9:41:30	3	9:41:30	9:41:33	0:00:00	0:00:03
233	9	9:41:39	3	9:41:39	9:41:42	0:00:00	0:00:03
234	30	9:42:09	6	9:42:09	9:42:15	0:00:00	0:00:06
235	9	9:42:18	3	9:42:18	9:42:21	0:00:00	0:00:03
236	9	9:42:27	3	9:42:27	9:42:30	0:00:00	0:00:03
237	15	9:42:42	6	9:42:42	9:42:48	0:00:00	0:00:06
238	3	9:42:45	6	9:42:48	9:42:54	0:00:03	0:00:09
239	9	9:42:54	3	9:42:54	9:42:57	0:00:00	0:00:03
240	3	9:42:57	6	9:42:57	9:43:03	0:00:00	0:00:06
241	3	9:43:00	3	9:43:03	9:43:06	0:00:03	0:00:06
242	30	9:43:30	3	9:43:30	9:43:33	0:00:00	0:00:03
243	30	9:44:00	3	9:44:00	9:44:03	0:00:00	0:00:03
244	3	9:44:03	6	9:44:03	9:44:09	0:00:00	0:00:06
245	3	9:44:06	9	9:44:09	9:44:18	0:00:03	0:00:12
246	9	9:44:15	6	9:44:18	9:44:24	0:00:03	0:00:09
247	15	9:44:30	3	9:44:30	9:44:33	0:00:00	0:00:03
248	30	9:45:00	6	9:45:00	9:45:06	0:00:00	0:00:06
249	9	9:45:09	6	9:45:09	9:45:15	0:00:00	0:00:06
250	3	9:45:12	3	9:45:15	9:45:18	0:00:03	0:00:06
251	9	9:45:21	6	9:45:21	9:45:27	0:00:00	0:00:06
252	3	9:45:24	6	9:45:27	9:45:33	0:00:03	0:00:09
253	3	9:45:27	6	9:45:33	9:45:39	0:00:06	0:00:12
254	3	9:45:30	6	9:45:39	9:45:45	0:00:09	0:00:15
255	3	9:45:33	6	9:45:45	9:45:51	0:00:12	0:00:18
256	9	9:45:42	3	9:45:51	9:45:54	0:00:09	0:00:12
257	3	9:45:45	6	9:45:54	9:46:00	0:00:09	0:00:15
258	30	9:46:15	3	9:46:15	9:46:18	0:00:00	0:00:03
259	3	9:46:18	3	9:46:18	9:46:21	0:00:00	0:00:03
260	3	9:46:21	6	9:46:21	9:46:27	0:00:00	0:00:06
261	3	9:46:24	3	9:46:27	9:46:30	0:00:03	0:00:06
262	3	9:46:27	3	9:46:30	9:46:33	0:00:03	0:00:06
263	9	9:46:36	3	9:46:36	9:46:39	0:00:00	0:00:03
264	3	9:46:39	6	9:46:39	9:46:45	0:00:00	0:00:06
265	3	9:46:42	3	9:46:45	9:46:48	0:00:03	0:00:06
266	3	9:46:45	3	9:46:48	9:46:51	0:00:03	0:00:06
267	3	9:46:48	3	9:46:51	9:46:54	0:00:03	0:00:06
268	3	9:46:51	3	9:46:54	9:46:57	0:00:03	0:00:06
269	9	9:47:00	3	9:47:00	9:47:03	0:00:00	0:00:03
270	3	9:47:03	3	9:47:03	9:47:06	0:00:00	0:00:03
271	15	9:47:18	3	9:47:18	9:47:21	0:00:00	0:00:03
272	30	9:47:48	6	9:47:48	9:47:54	0:00:00	0:00:06
273	30	9:48:18	6	9:48:18	9:48:24	0:00:00	0:00:06
274	15	9:48:33	6	9:48:33	9:48:39	0:00:00	0:00:06
275	15	9:48:48	3	9:48:48	9:48:51	0:00:00	0:00:03
276	30	9:49:18	6	9:49:18	9:49:24	0:00:00	0:00:06
277	9	9:49:27	6	9:49:27	9:49:33	0:00:00	0:00:06
278	9	9:49:36	3	9:49:36	9:49:39	0:00:00	0:00:03
279	3	9:49:39	6	9:49:39	9:49:45	0:00:00	0:00:06
280	30	9:50:09	3	9:50:09	9:50:12	0:00:00	0:00:03
281	15	9:50:24	9	9:50:24	9:50:33	0:00:00	0:00:09
282	15	9:50:39	6	9:50:39	9:50:45	0:00:00	0:00:06
283	30	9:51:09	3	9:51:09	9:51:12	0:00:00	0:00:03

Table (A-1)
Queueing Simulation under condition I

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
284	9	9:51:18	3	9:51:18	9:51:21	0:00:00	0:00:03
285	9	9:51:27	3	9:51:27	9:51:30	0:00:00	0:00:03
286	3	9:51:30	9	9:51:30	9:51:39	0:00:00	0:00:09
287	9	9:51:39	6	9:51:39	9:51:45	0:00:00	0:00:06
288	15	9:51:54	6	9:51:54	9:52:00	0:00:00	0:00:06
289	3	9:51:57	3	9:52:00	9:52:03	0:00:03	0:00:06
290	3	9:52:00	9	9:52:03	9:52:12	0:00:03	0:00:12
291	3	9:52:03	3	9:52:12	9:52:15	0:00:09	0:00:12
292	30	9:52:33	6	9:52:33	9:52:39	0:00:00	0:00:06
293	30	9:53:03	3	9:53:03	9:53:06	0:00:00	0:00:03
294	9	9:53:12	3	9:53:12	9:53:15	0:00:00	0:00:03
295	3	9:53:15	6	9:53:15	9:53:21	0:00:00	0:00:06
296	3	9:53:18	6	9:53:21	9:53:27	0:00:03	0:00:09
297	30	9:53:48	6	9:53:48	9:53:54	0:00:00	0:00:06
298	3	9:53:51	6	9:53:54	9:54:00	0:00:03	0:00:09
299	3	9:53:54	3	9:54:00	9:54:03	0:00:06	0:00:09
300	9	9:54:03	3	9:54:03	9:54:06	0:00:00	0:00:03
301	3	9:54:06	3	9:54:06	9:54:09	0:00:00	0:00:03
302	15	9:54:21	6	9:54:21	9:54:27	0:00:00	0:00:06
303	30	9:54:51	6	9:54:51	9:54:57	0:00:00	0:00:06
304	30	9:55:21	3	9:55:21	9:55:24	0:00:00	0:00:03
305	9	9:55:30	3	9:55:30	9:55:33	0:00:00	0:00:03
306	3	9:55:33	9	9:55:33	9:55:42	0:00:00	0:00:09
307	3	9:55:36	3	9:55:42	9:55:45	0:00:06	0:00:09
308	9	9:55:45	6	9:55:45	9:55:51	0:00:00	0:00:06
309	3	9:55:48	3	9:55:51	9:55:54	0:00:03	0:00:06
310	9	9:55:57	6	9:55:57	9:56:03	0:00:00	0:00:06
311	3	9:56:00	3	9:56:03	9:56:06	0:00:03	0:00:06
312	30	9:56:30	3	9:56:30	9:56:33	0:00:00	0:00:03
313	3	9:56:33	6	9:56:33	9:56:39	0:00:00	0:00:06
314	3	9:56:36	9	9:56:39	9:56:48	0:00:03	0:00:12
315	3	9:56:39	3	9:56:48	9:56:51	0:00:09	0:00:12
316	15	9:56:54	3	9:56:54	9:56:57	0:00:00	0:00:03
317	9	9:57:03	6	9:57:03	9:57:09	0:00:00	0:00:06
318	15	9:57:18	3	9:57:18	9:57:21	0:00:00	0:00:03
319	9	9:57:27	6	9:57:27	9:57:33	0:00:00	0:00:06
320	3	9:57:30	3	9:57:33	9:57:36	0:00:03	0:00:06
321	9	9:57:39	3	9:57:39	9:57:42	0:00:00	0:00:03
322	3	9:57:42	9	9:57:42	9:57:51	0:00:00	0:00:09
323	9	9:57:51	6	9:57:51	9:57:57	0:00:00	0:00:06
324	3	9:57:54	3	9:57:57	9:58:00	0:00:03	0:00:06
325	3	9:57:57	9	9:58:00	9:58:09	0:00:03	0:00:12
326	9	9:58:06	9	9:58:09	9:58:18	0:00:03	0:00:12
327	3	9:58:09	3	9:58:18	9:58:21	0:00:09	0:00:12
328	30	9:58:39	6	9:58:39	9:58:45	0:00:00	0:00:06
329	3	9:58:42	3	9:58:45	9:58:48	0:00:03	0:00:06
330	9	9:58:51	9	9:58:51	9:59:00	0:00:00	0:00:09
331	3	9:58:54	6	9:59:00	9:59:06	0:00:06	0:00:12
332	9	9:59:03	6	9:59:06	9:59:12	0:00:03	0:00:09
333	9	9:59:12	6	9:59:12	9:59:18	0:00:00	0:00:06
334	30	9:59:42	3	9:59:42	9:59:45	0:00:00	0:00:03
335	3	9:59:45	3	9:59:45	9:59:48	0:00:00	0:00:03
336	3	9:59:48	3	9:59:48	9:59:51	0:00:00	0:00:03
337	3	9:59:51	3	9:59:51	9:59:54	0:00:00	0:00:03
closed	15	10:00:06					

Table (A-2)
Queueing Simulation under condition II

Cust. #	Interarriva Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
start		9:00					
1	3	9:00:03	30	9:00:03	9:00:33	0:00:00	0:00:30
2	3	9:00:06	30	9:00:33	9:01:03	0:00:27	0:00:57
3	30	9:00:36	90	9:01:03	9:02:33	0:00:27	0:01:57
4	3	9:00:39	30	9:02:33	9:03:03	0:01:54	0:02:24
5	9	9:00:48	60	9:03:03	9:04:03	0:02:15	0:03:15
6	9	9:00:57	30	9:04:03	9:04:33	0:03:06	0:03:36
7	15	9:01:12	30	9:04:33	9:05:03	0:03:21	0:03:51
8	3	9:01:15	30	9:05:03	9:05:33	0:03:48	0:04:18
9	30	9:01:45	30	9:05:33	9:06:03	0:03:48	0:04:18
10	9	9:01:54	90	9:06:03	9:07:33	0:04:09	0:05:39
11	30	9:02:24	30	9:07:33	9:08:03	0:05:09	0:05:39
12	9	9:02:33	60	9:08:03	9:09:03	0:05:30	0:06:30
13	9	9:02:42	30	9:09:03	9:09:33	0:06:21	0:06:51
14	3	9:02:45	30	9:09:33	9:10:03	0:06:48	0:07:18
15	3	9:02:48	30	9:10:03	9:10:33	0:07:15	0:07:45
16	9	9:02:57	30	9:10:33	9:11:03	0:07:36	0:08:06
17	9	9:03:06	30	9:11:03	9:11:33	0:07:57	0:08:27
18	9	9:03:15	60	9:11:33	9:12:33	0:08:18	0:09:18
19	15	9:03:30	30	9:12:33	9:13:03	0:09:03	0:09:33
20	9	9:03:39	30	9:13:03	9:13:33	0:09:24	0:09:54
21	15	9:03:54	60	9:13:33	9:14:33	0:09:39	0:10:39
22	15	9:04:09	30	9:14:33	9:15:03	0:10:24	0:10:54
23	30	9:04:39	30	9:15:03	9:15:33	0:10:24	0:10:54
24	30	9:05:09	60	9:15:33	9:16:33	0:10:24	0:11:24
25	9	9:05:18	60	9:16:33	9:17:33	0:11:15	0:12:15
26	3	9:05:21	90	9:17:33	9:19:03	0:12:12	0:13:42
27	9	9:05:30	60	9:19:03	9:20:03	0:13:33	0:14:33
28	30	9:06:00	90	9:20:03	9:21:33	0:14:03	0:15:33
29	3	9:06:03	60	9:21:33	9:22:33	0:15:30	0:16:30
30	3	9:06:06	60	9:22:33	9:23:33	0:16:27	0:17:27
31	15	9:06:21	60	9:23:33	9:24:33	0:17:12	0:18:12
32	3	9:06:24	30	9:24:33	9:25:03	0:18:09	0:18:39
33	30	9:06:54	30	9:25:03	9:25:33	0:18:09	0:18:39
34	9	9:07:03	90	9:25:33	9:27:03	0:18:30	0:20:00
35	15	9:07:18	90	9:27:03	9:28:33	0:19:45	0:21:15
36	30	9:07:48	60	9:28:33	9:29:33	0:20:45	0:21:45
37	30	9:08:18	30	9:29:33	9:30:03	0:21:15	0:21:45
38	3	9:08:21	90	9:30:03	9:31:33	0:21:42	0:23:12
39	3	9:08:24	30	9:31:33	9:32:03	0:23:09	0:23:39
40	3	9:08:27	90	9:32:03	9:33:33	0:23:36	0:25:06
41	9	9:08:36	90	9:33:33	9:35:03	0:24:57	0:26:27
42	3	9:08:39	60	9:35:03	9:36:03	0:26:24	0:27:24
43	15	9:08:54	90	9:36:03	9:37:33	0:27:09	0:28:39
44	3	9:08:57	30	9:37:33	9:38:03	0:28:36	0:29:06
45	9	9:09:06	30	9:38:03	9:38:33	0:28:57	0:29:27
46	3	9:09:09	30	9:38:33	9:39:03	0:29:24	0:29:54
47	9	9:09:18	90	9:39:03	9:40:33	0:29:45	0:31:15
48	3	9:09:21	90	9:40:33	9:42:03	0:31:12	0:32:42
49	30	9:09:51	30	9:42:03	9:42:33	0:32:12	0:32:42
50	30	9:10:21	30	9:42:33	9:43:03	0:32:12	0:32:42
51	3	9:10:24	30	9:43:03	9:43:33	0:32:39	0:33:09
52	3	9:10:27	30	9:43:33	9:44:03	0:33:06	0:33:36
53	30	9:10:57	60	9:44:03	9:45:03	0:33:06	0:34:06

Table (A-2)
Queueing Simulation under condition II

Cust. #	Interarriva Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
54	3	9:11:00	60	9:45:03	9:46:03	0:34:03	0:35:03
55	3	9:11:03	30	9:46:03	9:46:33	0:35:00	0:35:30
56	30	9:11:33	30	9:46:33	9:47:03	0:35:00	0:35:30
57	3	9:11:36	30	9:47:03	9:47:33	0:35:27	0:35:57
58	9	9:11:45	30	9:47:33	9:48:03	0:35:48	0:36:18
59	9	9:11:54	30	9:48:03	9:48:33	0:36:09	0:36:39
60	15	9:12:09	60	9:48:33	9:49:33	0:36:24	0:37:24
61	3	9:12:12	90	9:49:33	9:51:03	0:37:21	0:38:51
62	9	9:12:21	60	9:51:03	9:52:03	0:38:42	0:39:42
63	30	9:12:51	30	9:52:03	9:52:33	0:39:12	0:39:42
64	15	9:13:06	90	9:52:33	9:54:03	0:39:27	0:40:57
65	9	9:13:15	30	9:54:03	9:54:33	0:40:48	0:41:18
66	9	9:13:24	60	9:54:33	9:55:33	0:41:09	0:42:09
67	15	9:13:39	30	9:55:33	9:56:03	0:41:54	0:42:24
68	3	9:13:42	30	9:56:03	9:56:33	0:42:21	0:42:51
69	9	9:13:51	30	9:56:33	9:57:03	0:42:42	0:43:12
70	9	9:14:00	30	9:57:03	9:57:33	0:43:03	0:43:33
71	3	9:14:03	90	9:57:33	9:59:03	0:43:30	0:45:00
72	3	9:14:06	30	9:59:03	9:59:33	0:44:57	0:45:27
73	15	9:14:21	90	9:59:33	10:01:03	0:45:12	0:46:42
74	9	9:14:30	90	10:01:03	10:02:33	0:46:33	0:48:03
75	30	9:15:00	60	10:02:33	10:03:33	0:47:33	0:48:33
76	3	9:15:03	30	10:03:33	10:04:03	0:48:30	0:49:00
77	15	9:15:18	30	10:04:03	10:04:33	0:48:45	0:49:15
78	3	9:15:21	30	10:04:33	10:05:03	0:49:12	0:49:42
79	30	9:15:51	60	10:05:03	10:06:03	0:49:12	0:50:12
80	9	9:16:00	30	10:06:03	10:06:33	0:50:03	0:50:33
81	30	9:16:30	30	10:06:33	10:07:03	0:50:03	0:50:33
82	30	9:17:00	30	10:07:03	10:07:33	0:50:03	0:50:33
83	3	9:17:03	90	10:07:33	10:09:03	0:50:30	0:52:00
84	3	9:17:06	30	10:09:03	10:09:33	0:51:57	0:52:27
85	9	9:17:15	30	10:09:33	10:10:03	0:52:18	0:52:48
86	30	9:17:45	30	10:10:03	10:10:33	0:52:18	0:52:48
87	3	9:17:48	30	10:10:33	10:11:03	0:52:45	0:53:15
88	30	9:18:18	60	10:11:03	10:12:03	0:52:45	0:53:45
89	3	9:18:21	30	10:12:03	10:12:33	0:53:42	0:54:12
90	9	9:18:30	30	10:12:33	10:13:03	0:54:03	0:54:33
91	30	9:19:00	90	10:13:03	10:14:33	0:54:03	0:55:33
92	9	9:19:09	60	10:14:33	10:15:33	0:55:24	0:56:24
93	9	9:19:18	60	10:15:33	10:16:33	0:56:15	0:57:15
94	9	9:19:27	90	10:16:33	10:18:03	0:57:06	0:58:36
95	3	9:19:30	30	10:18:03	10:18:33	0:58:33	0:59:03
96	30	9:20:00	30	10:18:33	10:19:03	0:58:33	0:59:03
97	3	9:20:03	30	10:19:03	10:19:33	0:59:00	0:59:30
98	3	9:20:06	60	10:19:33	10:20:33	0:59:27	1:00:27
99	9	9:20:15	30	10:20:33	10:21:03	1:00:18	1:00:48
100	3	9:20:18	30	10:21:03	10:21:33	1:00:45	1:01:15
101	30	9:20:48	30	10:21:33	10:22:03	1:00:45	1:01:15
102	9	9:20:57	30	10:22:03	10:22:33	1:01:06	1:01:36
103	3	9:21:00	30	10:22:33	10:23:03	1:01:33	1:02:03
104	15	9:21:15	90	10:23:03	10:24:33	1:01:48	1:03:18
105	30	9:21:45	30	10:24:33	10:25:03	1:02:48	1:03:18
106	3	9:21:48	60	10:25:03	10:26:03	1:03:15	1:04:15
107	30	9:22:18	90	10:26:03	10:27:33	1:03:45	1:05:15
108	3	9:22:21	30	10:27:33	10:28:03	1:05:12	1:05:42

Table (A-2)
Queueing Simulation under condition II

Cust. #	Interarriva Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
109	3	9:22:24	90	10:28:03	10:29:33	1:05:39	1:07:09
110	3	9:22:27	30	10:29:33	10:30:03	1:07:06	1:07:36
111	3	9:22:30	60	10:30:03	10:31:03	1:07:33	1:08:33
112	9	9:22:39	30	10:31:03	10:31:33	1:08:24	1:08:54
113	9	9:22:48	30	10:31:33	10:32:03	1:08:45	1:09:15
114	30	9:23:18	60	10:32:03	10:33:03	1:08:45	1:09:45
115	9	9:23:27	30	10:33:03	10:33:33	1:09:36	1:10:06
116	9	9:23:36	30	10:33:33	10:34:03	1:09:57	1:10:27
117	30	9:24:06	60	10:34:03	10:35:03	1:09:57	1:10:57
118	30	9:24:36	30	10:35:03	10:35:33	1:10:27	1:10:57
119	3	9:24:39	60	10:35:33	10:36:33	1:10:54	1:11:54
120	30	9:25:09	90	10:36:33	10:38:03	1:11:24	1:12:54
121	3	9:25:12	30	10:38:03	10:38:33	1:12:51	1:13:21
122	15	9:25:27	60	10:38:33	10:39:33	1:13:06	1:14:06
123	30	9:25:57	90	10:39:33	10:41:03	1:13:36	1:15:06
124	3	9:26:00	30	10:41:03	10:41:33	1:15:03	1:15:33
125	3	9:26:03	30	10:41:33	10:42:03	1:15:30	1:16:00
126	30	9:26:33	30	10:42:03	10:42:33	1:15:30	1:16:00
127	3	9:26:36	90	10:42:33	10:44:03	1:15:57	1:17:27
128	30	9:27:06	60	10:44:03	10:45:03	1:16:57	1:17:57
129	3	9:27:09	60	10:45:03	10:46:03	1:17:54	1:18:54
130	9	9:27:18	60	10:46:03	10:47:03	1:18:45	1:19:45
131	3	9:27:21	60	10:47:03	10:48:03	1:19:42	1:20:42
132	9	9:27:30	60	10:48:03	10:49:03	1:20:33	1:21:33
133	3	9:27:33	90	10:49:03	10:50:33	1:21:30	1:23:00
134	9	9:27:42	30	10:50:33	10:51:03	1:22:51	1:23:21
135	3	9:27:45	90	10:51:03	10:52:33	1:23:18	1:24:48
136	15	9:28:00	60	10:52:33	10:53:33	1:24:33	1:25:33
137	3	9:28:03	60	10:53:33	10:54:33	1:25:30	1:26:30
138	3	9:28:06	90	10:54:33	10:56:03	1:26:27	1:27:57
139	30	9:28:36	30	10:56:03	10:56:33	1:27:27	1:27:57
140	3	9:28:39	30	10:56:33	10:57:03	1:27:54	1:28:24
141	30	9:29:09	60	10:57:03	10:58:03	1:27:54	1:28:54
142	3	9:29:12	30	10:58:03	10:58:33	1:28:51	1:29:21
143	3	9:29:15	30	10:58:33	10:59:03	1:29:18	1:29:48
144	9	9:29:24	30	10:59:03	10:59:33	1:29:39	1:30:09
145	3	9:29:27	90	10:59:33	11:01:03	1:30:06	1:31:36
146	9	9:29:36	30	11:01:03	11:01:33	1:31:27	1:31:57
147	9	9:29:45	30	11:01:33	11:02:03	1:31:48	1:32:18
148	3	9:29:48	60	11:02:03	11:03:03	1:32:15	1:33:15
149	9	9:29:57	60	11:03:03	11:04:03	1:33:06	1:34:06
150	3	9:30:00	90	11:04:03	11:05:33	1:34:03	1:35:33
151	15	9:30:15	30	11:05:33	11:06:03	1:35:18	1:35:48
152	3	9:30:18	90	11:06:03	11:07:33	1:35:45	1:37:15
153	3	9:30:21	60	11:07:33	11:08:33	1:37:12	1:38:12
154	3	9:30:24	90	11:08:33	11:10:03	1:38:09	1:39:39
155	3	9:30:27	60	11:10:03	11:11:03	1:39:36	1:40:36
156	15	9:30:42	30	11:11:03	11:11:33	1:40:21	1:40:51
157	30	9:31:12	60	11:11:33	11:12:33	1:40:21	1:41:21
158	3	9:31:15	60	11:12:33	11:13:33	1:41:18	1:42:18
159	15	9:31:30	60	11:13:33	11:14:33	1:42:03	1:43:03
160	30	9:32:00	30	11:14:33	11:15:03	1:42:33	1:43:03
161	30	9:32:30	90	11:15:03	11:16:33	1:42:33	1:44:03
162	3	9:32:33	60	11:16:33	11:17:33	1:44:00	1:45:00
163	3	9:32:36	30	11:17:33	11:18:03	1:44:57	1:45:27

Table (A-2)
Queueing Simulation under condition II

Cust. #	Interarriva Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
164	9	9:32:45	60	11:18:03	11:19:03	1:45:18	1:46:18
165	9	9:32:54	60	11:19:03	11:20:03	1:46:09	1:47:09
166	3	9:32:57	30	11:20:03	11:20:33	1:47:06	1:47:36
167	3	9:33:00	90	11:20:33	11:22:03	1:47:33	1:49:03
168	9	9:33:09	60	11:22:03	11:23:03	1:48:54	1:49:54
169	30	9:33:39	60	11:23:03	11:24:03	1:49:24	1:50:24
170	3	9:33:42	60	11:24:03	11:25:03	1:50:21	1:51:21
171	3	9:33:45	30	11:25:03	11:25:33	1:51:18	1:51:48
172	9	9:33:54	60	11:25:33	11:26:33	1:51:39	1:52:39
173	30	9:34:24	30	11:26:33	11:27:03	1:52:09	1:52:39
174	15	9:34:39	30	11:27:03	11:27:33	1:52:24	1:52:54
175	3	9:34:42	60	11:27:33	11:28:33	1:52:51	1:53:51
176	3	9:34:45	60	11:28:33	11:29:33	1:53:48	1:54:48
177	3	9:34:48	60	11:29:33	11:30:33	1:54:45	1:55:45
178	3	9:34:51	60	11:30:33	11:31:33	1:55:42	1:56:42
179	15	9:35:06	60	11:31:33	11:32:33	1:56:27	1:57:27
180	15	9:35:21	30	11:32:33	11:33:03	1:57:12	1:57:42
181	3	9:35:24	30	11:33:03	11:33:33	1:57:39	1:58:09
182	3	9:35:27	30	11:33:33	11:34:03	1:58:06	1:58:36
183	9	9:35:36	90	11:34:03	11:35:33	1:58:27	1:59:57
184	3	9:35:39	30	11:35:33	11:36:03	1:59:54	2:00:24
185	9	9:35:48	60	11:36:03	11:37:03	2:00:15	2:01:15
186	3	9:35:51	60	11:37:03	11:38:03	2:01:12	2:02:12
187	30	9:36:21	60	11:38:03	11:39:03	2:01:42	2:02:42
188	3	9:36:24	90	11:39:03	11:40:33	2:02:39	2:04:09
189	9	9:36:33	30	11:40:33	11:41:03	2:04:00	2:04:30
190	3	9:36:36	60	11:41:03	11:42:03	2:04:27	2:05:27
191	3	9:36:39	60	11:42:03	11:43:03	2:05:24	2:06:24
192	9	9:36:48	30	11:43:03	11:43:33	2:06:15	2:06:45
193	9	9:36:57	30	11:43:33	11:44:03	2:06:36	2:07:06
194	9	9:37:06	90	11:44:03	11:45:33	2:06:57	2:08:27
195	3	9:37:09	30	11:45:33	11:46:03	2:08:24	2:08:54
196	9	9:37:18	60	11:46:03	11:47:03	2:08:45	2:09:45
197	9	9:37:27	30	11:47:03	11:47:33	2:09:36	2:10:06
198	30	9:37:57	60	11:47:33	11:48:33	2:09:36	2:10:36
199	30	9:38:27	90	11:48:33	11:50:03	2:10:06	2:11:36
200	3	9:38:30	30	11:50:03	11:50:33	2:11:33	2:12:03
201	9	9:38:39	60	11:50:33	11:51:33	2:11:54	2:12:54
202	15	9:38:54	90	11:51:33	11:53:03	2:12:39	2:14:09
203	9	9:39:03	30	11:53:03	11:53:33	2:14:00	2:14:30
204	3	9:39:06	30	11:53:33	11:54:03	2:14:27	2:14:57
205	30	9:39:36	60	11:54:03	11:55:03	2:14:27	2:15:27
206	30	9:40:06	30	11:55:03	11:55:33	2:14:57	2:15:27
207	9	9:40:15	60	11:55:33	11:56:33	2:15:18	2:16:18
208	9	9:40:24	30	11:56:33	11:57:03	2:16:09	2:16:39
209	30	9:40:54	90	11:57:03	11:58:33	2:16:09	2:17:39
210	9	9:41:03	30	11:58:33	11:59:03	2:17:30	2:18:00
211	3	9:41:06	90	11:59:03	12:00:33	2:17:57	2:19:27
212	3	9:41:09	30	12:00:33	12:01:03	2:19:24	2:19:54
213	3	9:41:12	60	12:01:03	12:02:03	2:19:51	2:20:51
214	9	9:41:21	90	12:02:03	12:03:33	2:20:42	2:22:12
215	15	9:41:36	30	12:03:33	12:04:03	2:21:57	2:22:27
216	3	9:41:39	90	12:04:03	12:05:33	2:22:24	2:23:54
217	3	9:41:42	30	12:05:33	12:06:03	2:23:51	2:24:21
218	3	9:41:45	30	12:06:03	12:06:33	2:24:18	2:24:48

Table (A-2)
Queueing Simulation under condition II

Cust. #	Interarriva Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
219	30	9:42:15	60	12:06:33	12:07:33	2:24:18	2:25:18
220	30	9:42:45	90	12:07:33	12:09:03	2:24:48	2:26:18
221	3	9:42:48	30	12:09:03	12:09:33	2:26:15	2:26:45
222	9	9:42:57	30	12:09:33	12:10:03	2:26:36	2:27:06
223	3	9:43:00	60	12:10:03	12:11:03	2:27:03	2:28:03
224	30	9:43:30	30	12:11:03	12:11:33	2:27:33	2:28:03
225	3	9:43:33	60	12:11:33	12:12:33	2:28:00	2:29:00
226	30	9:44:03	30	12:12:33	12:13:03	2:28:30	2:29:00
227	3	9:44:06	60	12:13:03	12:14:03	2:28:57	2:29:57
228	3	9:44:09	30	12:14:03	12:14:33	2:29:54	2:30:24
229	3	9:44:12	30	12:14:33	12:15:03	2:30:21	2:30:51
230	3	9:44:15	60	12:15:03	12:16:03	2:30:48	2:31:48
231	3	9:44:18	30	12:16:03	12:16:33	2:31:45	2:32:15
232	9	9:44:27	30	12:16:33	12:17:03	2:32:06	2:32:36
233	9	9:44:36	60	12:17:03	12:18:03	2:32:27	2:33:27
234	3	9:44:39	30	12:18:03	12:18:33	2:33:24	2:33:54
235	30	9:45:09	60	12:18:33	12:19:33	2:33:24	2:34:24
236	9	9:45:18	60	12:19:33	12:20:33	2:34:15	2:35:15
237	3	9:45:21	60	12:20:33	12:21:33	2:35:12	2:36:12
238	3	9:45:24	60	12:21:33	12:22:33	2:36:09	2:37:09
239	3	9:45:27	60	12:22:33	12:23:33	2:37:06	2:38:06
240	9	9:45:36	60	12:23:33	12:24:33	2:37:57	2:38:57
241	9	9:45:45	30	12:24:33	12:25:03	2:38:48	2:39:18
242	9	9:45:54	60	12:25:03	12:26:03	2:39:09	2:40:09
243	9	9:46:03	30	12:26:03	12:26:33	2:40:00	2:40:30
244	3	9:46:06	90	12:26:33	12:28:03	2:40:27	2:41:57
245	3	9:46:09	60	12:28:03	12:29:03	2:41:54	2:42:54
246	3	9:46:12	30	12:29:03	12:29:33	2:42:51	2:43:21
247	30	9:46:42	30	12:29:33	12:30:03	2:42:51	2:43:21
248	9	9:46:51	30	12:30:03	12:30:33	2:43:12	2:43:42
249	9	9:47:00	60	12:30:33	12:31:33	2:43:33	2:44:33
250	9	9:47:09	90	12:31:33	12:33:03	2:44:24	2:45:54
251	3	9:47:12	30	12:33:03	12:33:33	2:45:51	2:46:21
252	30	9:47:42	30	12:33:33	12:34:03	2:45:51	2:46:21
253	9	9:47:51	60	12:34:03	12:35:03	2:46:12	2:47:12
254	30	9:48:21	30	12:35:03	12:35:33	2:46:42	2:47:12
255	3	9:48:24	30	12:35:33	12:36:03	2:47:09	2:47:39
256	3	9:48:27	60	12:36:03	12:37:03	2:47:36	2:48:36
257	9	9:48:36	60	12:37:03	12:38:03	2:48:27	2:49:27
258	15	9:48:51	60	12:38:03	12:39:03	2:49:12	2:50:12
259	3	9:48:54	30	12:39:03	12:39:33	2:50:09	2:50:39
260	3	9:48:57	30	12:39:33	12:40:03	2:50:36	2:51:06
261	30	9:49:27	30	12:40:03	12:40:33	2:50:36	2:51:06
262	3	9:49:30	60	12:40:33	12:41:33	2:51:03	2:52:03
263	30	9:50:00	60	12:41:33	12:42:33	2:51:33	2:52:33
264	15	9:50:15	60	12:42:33	12:43:33	2:52:18	2:53:18
265	9	9:50:24	30	12:43:33	12:44:03	2:53:09	2:53:39
266	9	9:50:33	30	12:44:03	12:44:33	2:53:30	2:54:00
267	3	9:50:36	60	12:44:33	12:45:33	2:53:57	2:54:57
268	3	9:50:39	60	12:45:33	12:46:33	2:54:54	2:55:54
269	3	9:50:42	30	12:46:33	12:47:03	2:55:51	2:56:21
270	3	9:50:45	30	12:47:03	12:47:33	2:56:18	2:56:48
271	15	9:51:00	30	12:47:33	12:48:03	2:56:33	2:57:03
272	30	9:51:30	90	12:48:03	12:49:33	2:56:33	2:58:03
273	3	9:51:33	30	12:49:33	12:50:03	2:58:00	2:58:30

Table (A-2)
Queueing Simulation under condition II

Cust. #	Interarriva Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
274	3	9:51:36	30	12:50:03	12:50:33	2:58:27	2:58:57
275	3	9:51:39	30	12:50:33	12:51:03	2:58:54	2:59:24
276	9	9:51:48	30	12:51:03	12:51:33	2:59:15	2:59:45
277	9	9:51:57	30	12:51:33	12:52:03	2:59:36	3:00:06
278	3	9:52:00	30	12:52:03	12:52:33	3:00:03	3:00:33
279	9	9:52:09	30	12:52:33	12:53:03	3:00:24	3:00:54
280	3	9:52:12	30	12:53:03	12:53:33	3:00:51	3:01:21
281	30	9:52:42	60	12:53:33	12:54:33	3:00:51	3:01:51
282	3	9:52:45	60	12:54:33	12:55:33	3:01:48	3:02:48
283	15	9:53:00	90	12:55:33	12:57:03	3:02:33	3:04:03
284	30	9:53:30	30	12:57:03	12:57:33	3:03:33	3:04:03
285	15	9:53:45	90	12:57:33	12:59:03	3:03:48	3:05:18
286	15	9:54:00	60	12:59:03	13:00:03	3:05:03	3:06:03
287	3	9:54:03	90	13:00:03	13:01:33	3:06:00	3:07:30
288	9	9:54:12	30	13:01:33	13:02:03	3:07:21	3:07:51
289	3	9:54:15	60	13:02:03	13:03:03	3:07:48	3:08:48
290	9	9:54:24	60	13:03:03	13:04:03	3:08:39	3:09:39
291	3	9:54:27	60	13:04:03	13:05:03	3:09:36	3:10:36
292	15	9:54:42	30	13:05:03	13:05:33	3:10:21	3:10:51
293	9	9:54:51	60	13:05:33	13:06:33	3:10:42	3:11:42
294	30	9:55:21	30	13:06:33	13:07:03	3:11:12	3:11:42
295	3	9:55:24	30	13:07:03	13:07:33	3:11:39	3:12:09
296	15	9:55:39	60	13:07:33	13:08:33	3:11:54	3:12:54
297	9	9:55:48	30	13:08:33	13:09:03	3:12:45	3:13:15
298	3	9:55:51	90	13:09:03	13:10:33	3:13:12	3:14:42
299	15	9:56:06	60	13:10:33	13:11:33	3:14:27	3:15:27
300	9	9:56:15	90	13:11:33	13:13:03	3:15:18	3:16:48
301	3	9:56:18	60	13:13:03	13:14:03	3:16:45	3:17:45
302	3	9:56:21	30	13:14:03	13:14:33	3:17:42	3:18:12
303	3	9:56:24	30	13:14:33	13:15:03	3:18:09	3:18:39
304	3	9:56:27	30	13:15:03	13:15:33	3:18:36	3:19:06
305	9	9:56:36	60	13:15:33	13:16:33	3:18:57	3:19:57
306	9	9:56:45	30	13:16:33	13:17:03	3:19:48	3:20:18
307	30	9:57:15	30	13:17:03	13:17:33	3:19:48	3:20:18
308	3	9:57:18	60	13:17:33	13:18:33	3:20:15	3:21:15
309	3	9:57:21	30	13:18:33	13:19:03	3:21:12	3:21:42
310	3	9:57:24	30	13:19:03	13:19:33	3:21:39	3:22:09
311	9	9:57:33	30	13:19:33	13:20:03	3:22:00	3:22:30
312	15	9:57:48	60	13:20:03	13:21:03	3:22:15	3:23:15
313	3	9:57:51	90	13:21:03	13:22:33	3:23:12	3:24:42
314	9	9:58:00	30	13:22:33	13:23:03	3:24:33	3:25:03
315	15	9:58:15	30	13:23:03	13:23:33	3:24:48	3:25:18
316	30	9:58:45	30	13:23:33	13:24:03	3:24:48	3:25:18
317	3	9:58:48	30	13:24:03	13:24:33	3:25:15	3:25:45
318	15	9:59:03	60	13:24:33	13:25:33	3:25:30	3:26:30
319	30	9:59:33	30	13:25:33	13:26:03	3:26:00	3:26:30
320	3	9:59:36	60	13:26:03	13:27:03	3:26:27	3:27:27
321	9	9:59:45	30	13:27:03	13:27:33	3:27:18	3:27:48
closed	15	10:00:00					

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
start		9:00					
1	1	9:00:01	3	9:00:01	9:00:04	0:00:00	0:00:03
2	3	9:00:04	3	9:00:04	9:00:07	0:00:00	0:00:03
3	10	9:00:14	3	9:00:14	9:00:17	0:00:00	0:00:03
4	1	9:00:15	3	9:00:17	9:00:20	0:00:02	0:00:05
5	1	9:00:16	6	9:00:20	9:00:26	0:00:04	0:00:10
6	1	9:00:17	6	9:00:26	9:00:32	0:00:09	0:00:15
7	5	9:00:22	6	9:00:32	9:00:38	0:00:10	0:00:16
8	1	9:00:23	6	9:00:38	9:00:44	0:00:15	0:00:21
9	10	9:00:33	6	9:00:44	9:00:50	0:00:11	0:00:17
10	3	9:00:36	6	9:00:50	9:00:56	0:00:14	0:00:20
11	1	9:00:37	3	9:00:56	9:00:59	0:00:19	0:00:22
12	1	9:00:38	9	9:00:59	9:01:08	0:00:21	0:00:30
13	1	9:00:39	3	9:01:08	9:01:11	0:00:29	0:00:32
14	1	9:00:40	3	9:01:11	9:01:14	0:00:31	0:00:34
15	1	9:00:41	3	9:01:14	9:01:17	0:00:33	0:00:36
16	10	9:00:51	3	9:01:17	9:01:20	0:00:26	0:00:29
17	1	9:00:52	3	9:01:20	9:01:23	0:00:28	0:00:31
18	1	9:00:53	9	9:01:23	9:01:32	0:00:30	0:00:39
19	1	9:00:54	3	9:01:32	9:01:35	0:00:38	0:00:41
20	1	9:00:55	6	9:01:35	9:01:41	0:00:40	0:00:46
21	5	9:01:00	3	9:01:41	9:01:44	0:00:41	0:00:44
22	3	9:01:03	6	9:01:44	9:01:50	0:00:41	0:00:47
23	10	9:01:13	3	9:01:50	9:01:53	0:00:37	0:00:40
24	5	9:01:18	3	9:01:53	9:01:56	0:00:35	0:00:38
25	1	9:01:19	9	9:01:56	9:02:05	0:00:37	0:00:46
26	1	9:01:20	3	9:02:05	9:02:08	0:00:45	0:00:48
27	10	9:01:30	3	9:02:08	9:02:11	0:00:38	0:00:41
28	1	9:01:31	3	9:02:11	9:02:14	0:00:40	0:00:43
29	10	9:01:41	3	9:02:14	9:02:17	0:00:33	0:00:36
30	5	9:01:46	6	9:02:17	9:02:23	0:00:31	0:00:37
31	1	9:01:47	6	9:02:23	9:02:29	0:00:36	0:00:42
32	1	9:01:48	3	9:02:29	9:02:32	0:00:41	0:00:44
33	1	9:01:49	9	9:02:32	9:02:41	0:00:43	0:00:52
34	1	9:01:50	3	9:02:41	9:02:44	0:00:51	0:00:54
35	3	9:01:53	6	9:02:44	9:02:50	0:00:51	0:00:57
36	3	9:01:56	6	9:02:50	9:02:56	0:00:54	0:01:00
37	5	9:02:01	3	9:02:56	9:02:59	0:00:55	0:00:58
38	1	9:02:02	3	9:02:59	9:03:02	0:00:57	0:01:00
39	1	9:02:03	6	9:03:02	9:03:08	0:00:59	0:01:05
40	1	9:02:04	6	9:03:08	9:03:14	0:01:04	0:01:10
41	1	9:02:05	6	9:03:14	9:03:20	0:01:09	0:01:15
42	1	9:02:06	9	9:03:20	9:03:29	0:01:14	0:01:23
43	10	9:02:16	3	9:03:29	9:03:32	0:01:13	0:01:16
44	3	9:02:19	9	9:03:32	9:03:41	0:01:13	0:01:22
45	5	9:02:24	9	9:03:41	9:03:50	0:01:17	0:01:26
46	5	9:02:29	3	9:03:50	9:03:53	0:01:21	0:01:24
47	5	9:02:34	3	9:03:53	9:03:56	0:01:19	0:01:22
48	1	9:02:35	3	9:03:56	9:03:59	0:01:21	0:01:24
49	3	9:02:38	3	9:03:59	9:04:02	0:01:21	0:01:24
50	10	9:02:48	3	9:04:02	9:04:05	0:01:14	0:01:17
51	3	9:02:51	3	9:04:05	9:04:08	0:01:14	0:01:17
52	1	9:02:52	3	9:04:08	9:04:11	0:01:16	0:01:19
53	1	9:02:53	9	9:04:11	9:04:20	0:01:18	0:01:27
54	1	9:02:54	3	9:04:20	9:04:23	0:01:26	0:01:29
55	10	9:03:04	3	9:04:23	9:04:26	0:01:19	0:01:22
56	1	9:03:05	6	9:04:26	9:04:32	0:01:21	0:01:27

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
57	1	9:03:06	6	9:04:32	9:04:38	0:01:26	0:01:32
58	5	9:03:11	6	9:04:38	9:04:44	0:01:27	0:01:33
59	1	9:03:12	3	9:04:44	9:04:47	0:01:32	0:01:35
60	1	9:03:13	3	9:04:47	9:04:50	0:01:34	0:01:37
61	5	9:03:18	3	9:04:50	9:04:53	0:01:32	0:01:35
62	1	9:03:19	9	9:04:53	9:05:02	0:01:34	0:01:43
63	10	9:03:29	3	9:05:02	9:05:05	0:01:33	0:01:36
64	5	9:03:34	9	9:05:05	9:05:14	0:01:31	0:01:40
65	1	9:03:35	3	9:05:14	9:05:17	0:01:39	0:01:42
66	3	9:03:38	3	9:05:17	9:05:20	0:01:39	0:01:42
67	5	9:03:43	9	9:05:20	9:05:29	0:01:37	0:01:46
68	1	9:03:44	6	9:05:29	9:05:35	0:01:45	0:01:51
69	5	9:03:49	6	9:05:35	9:05:41	0:01:46	0:01:52
70	1	9:03:50	9	9:05:41	9:05:50	0:01:51	0:02:00
71	1	9:03:51	3	9:05:50	9:05:53	0:01:59	0:02:02
72	10	9:04:01	3	9:05:53	9:05:56	0:01:52	0:01:55
73	10	9:04:11	6	9:05:56	9:06:02	0:01:45	0:01:51
74	1	9:04:12	6	9:06:02	9:06:08	0:01:50	0:01:56
75	1	9:04:13	6	9:06:08	9:06:14	0:01:55	0:02:01
76	1	9:04:14	9	9:06:14	9:06:23	0:02:00	0:02:09
77	3	9:04:17	3	9:06:23	9:06:26	0:02:06	0:02:09
78	5	9:04:22	3	9:06:26	9:06:29	0:02:04	0:02:07
79	5	9:04:27	3	9:06:29	9:06:32	0:02:02	0:02:05
80	5	9:04:32	6	9:06:32	9:06:38	0:02:00	0:02:06
81	3	9:04:35	3	9:06:38	9:06:41	0:02:03	0:02:06
82	1	9:04:36	6	9:06:41	9:06:47	0:02:05	0:02:11
83	3	9:04:39	6	9:06:47	9:06:53	0:02:08	0:02:14
84	1	9:04:40	3	9:06:53	9:06:56	0:02:13	0:02:16
85	5	9:04:45	6	9:06:56	9:07:02	0:02:11	0:02:17
86	10	9:04:55	3	9:07:02	9:07:05	0:02:07	0:02:10
87	1	9:04:56	6	9:07:05	9:07:11	0:02:09	0:02:15
88	3	9:04:59	3	9:07:11	9:07:14	0:02:12	0:02:15
89	1	9:05:00	6	9:07:14	9:07:20	0:02:14	0:02:20
90	1	9:05:01	3	9:07:20	9:07:23	0:02:19	0:02:22
91	1	9:05:02	3	9:07:23	9:07:26	0:02:21	0:02:24
92	3	9:05:05	3	9:07:26	9:07:29	0:02:21	0:02:24
93	5	9:05:10	6	9:07:29	9:07:35	0:02:19	0:02:25
94	10	9:05:20	6	9:07:35	9:07:41	0:02:15	0:02:21
95	5	9:05:25	9	9:07:41	9:07:50	0:02:16	0:02:25
96	3	9:05:28	6	9:07:50	9:07:56	0:02:22	0:02:28
97	1	9:05:29	6	9:07:56	9:08:02	0:02:27	0:02:33
98	1	9:05:30	6	9:08:02	9:08:08	0:02:32	0:02:38
99	1	9:05:31	3	9:08:08	9:08:11	0:02:37	0:02:40
100	3	9:05:34	6	9:08:11	9:08:17	0:02:37	0:02:43
101	1	9:05:35	3	9:08:17	9:08:20	0:02:42	0:02:45
102	5	9:05:40	6	9:08:20	9:08:26	0:02:40	0:02:46
103	1	9:05:41	3	9:08:26	9:08:29	0:02:45	0:02:48
104	5	9:05:46	3	9:08:29	9:08:32	0:02:43	0:02:46
105	3	9:05:49	6	9:08:32	9:08:38	0:02:43	0:02:49
106	1	9:05:50	3	9:08:38	9:08:41	0:02:48	0:02:51
107	3	9:05:53	6	9:08:41	9:08:47	0:02:48	0:02:54
108	1	9:05:54	3	9:08:47	9:08:50	0:02:53	0:02:56
109	10	9:06:04	9	9:08:50	9:08:59	0:02:46	0:02:55
110	10	9:06:14	3	9:08:59	9:09:02	0:02:45	0:02:48
111	1	9:06:15	3	9:09:02	9:09:05	0:02:47	0:02:50
112	1	9:06:16	3	9:09:05	9:09:08	0:02:49	0:02:52
113	1	9:06:17	6	9:09:08	9:09:14	0:02:51	0:02:57
114	3	9:06:20	6	9:09:14	9:09:20	0:02:54	0:03:00

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
115	1	9:06:21	6	9:09:20	9:09:26	0:02:59	0:03:05
116	1	9:06:22	6	9:09:26	9:09:32	0:03:04	0:03:10
117	1	9:06:23	9	9:09:32	9:09:41	0:03:09	0:03:18
118	1	9:06:24	3	9:09:41	9:09:44	0:03:17	0:03:20
119	3	9:06:27	6	9:09:44	9:09:50	0:03:17	0:03:23
120	3	9:06:30	6	9:09:50	9:09:56	0:03:20	0:03:26
121	5	9:06:35	3	9:09:56	9:09:59	0:03:21	0:03:24
122	1	9:06:36	3	9:09:59	9:10:02	0:03:23	0:03:26
123	10	9:06:46	9	9:10:02	9:10:11	0:03:16	0:03:25
124	3	9:06:49	3	9:10:11	9:10:14	0:03:22	0:03:25
125	1	9:06:50	6	9:10:14	9:10:20	0:03:24	0:03:30
126	3	9:06:53	6	9:10:20	9:10:26	0:03:27	0:03:33
127	3	9:06:56	6	9:10:26	9:10:32	0:03:30	0:03:36
128	1	9:06:57	3	9:10:32	9:10:35	0:03:35	0:03:38
129	1	9:06:58	3	9:10:35	9:10:38	0:03:37	0:03:40
130	1	9:06:59	9	9:10:38	9:10:47	0:03:39	0:03:48
131	1	9:07:00	9	9:10:47	9:10:56	0:03:47	0:03:56
132	3	9:07:03	3	9:10:56	9:10:59	0:03:53	0:03:56
133	3	9:07:06	3	9:10:59	9:11:02	0:03:53	0:03:56
134	1	9:07:07	9	9:11:02	9:11:11	0:03:55	0:04:04
135	1	9:07:08	3	9:11:11	9:11:14	0:04:03	0:04:06
136	10	9:07:18	3	9:11:14	9:11:17	0:03:56	0:03:59
137	10	9:07:28	9	9:11:17	9:11:26	0:03:49	0:03:58
138	5	9:07:33	3	9:11:26	9:11:29	0:03:53	0:03:56
139	5	9:07:38	6	9:11:29	9:11:35	0:03:51	0:03:57
140	3	9:07:41	3	9:11:35	9:11:38	0:03:54	0:03:57
141	3	9:07:44	3	9:11:38	9:11:41	0:03:54	0:03:57
142	1	9:07:45	3	9:11:41	9:11:44	0:03:56	0:03:59
143	5	9:07:50	9	9:11:44	9:11:53	0:03:54	0:04:03
144	10	9:08:00	6	9:11:53	9:11:59	0:03:53	0:03:59
145	10	9:08:10	3	9:11:59	9:12:02	0:03:49	0:03:52
146	10	9:08:20	9	9:12:02	9:12:11	0:03:42	0:03:51
147	10	9:08:30	3	9:12:11	9:12:14	0:03:41	0:03:44
148	5	9:08:35	6	9:12:14	9:12:20	0:03:39	0:03:45
149	1	9:08:36	3	9:12:20	9:12:23	0:03:44	0:03:47
150	1	9:08:37	6	9:12:23	9:12:29	0:03:46	0:03:52
151	3	9:08:40	3	9:12:29	9:12:32	0:03:49	0:03:52
152	1	9:08:41	9	9:12:32	9:12:41	0:03:51	0:04:00
153	1	9:08:42	6	9:12:41	9:12:47	0:03:59	0:04:05
154	1	9:08:43	3	9:12:47	9:12:50	0:04:04	0:04:07
155	5	9:08:48	6	9:12:50	9:12:56	0:04:02	0:04:08
156	5	9:08:53	9	9:12:56	9:13:05	0:04:03	0:04:12
157	3	9:08:56	6	9:13:05	9:13:11	0:04:09	0:04:15
158	1	9:08:57	3	9:13:11	9:13:14	0:04:14	0:04:17
159	1	9:08:58	9	9:13:14	9:13:23	0:04:16	0:04:25
160	3	9:09:01	9	9:13:23	9:13:32	0:04:22	0:04:31
161	1	9:09:02	6	9:13:32	9:13:38	0:04:30	0:04:36
162	10	9:09:12	9	9:13:38	9:13:47	0:04:26	0:04:35
163	5	9:09:17	3	9:13:47	9:13:50	0:04:30	0:04:33
164	1	9:09:18	3	9:13:50	9:13:53	0:04:32	0:04:35
165	1	9:09:19	3	9:13:53	9:13:56	0:04:34	0:04:37
166	1	9:09:20	6	9:13:56	9:14:02	0:04:36	0:04:42
167	10	9:09:30	6	9:14:02	9:14:08	0:04:32	0:04:38
168	1	9:09:31	3	9:14:08	9:14:11	0:04:37	0:04:40
169	1	9:09:32	9	9:14:11	9:14:20	0:04:39	0:04:48
170	1	9:09:33	9	9:14:20	9:14:29	0:04:47	0:04:56
171	1	9:09:34	9	9:14:29	9:14:38	0:04:55	0:05:04
172	1	9:09:35	9	9:14:38	9:14:47	0:05:03	0:05:12

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
173	1	9:09:36	6	9:14:47	9:14:53	0:05:11	0:05:17
174	10	9:09:46	9	9:14:53	9:15:02	0:05:07	0:05:16
175	1	9:09:47	6	9:15:02	9:15:08	0:05:15	0:05:21
176	1	9:09:48	3	9:15:08	9:15:11	0:05:20	0:05:23
177	3	9:09:51	9	9:15:11	9:15:20	0:05:20	0:05:29
178	10	9:10:01	6	9:15:20	9:15:26	0:05:19	0:05:25
179	3	9:10:04	9	9:15:26	9:15:35	0:05:22	0:05:31
180	1	9:10:05	6	9:15:35	9:15:41	0:05:30	0:05:36
181	1	9:10:06	3	9:15:41	9:15:44	0:05:35	0:05:38
182	1	9:10:07	3	9:15:44	9:15:47	0:05:37	0:05:40
183	1	9:10:08	9	9:15:47	9:15:56	0:05:39	0:05:48
184	1	9:10:09	3	9:15:56	9:15:59	0:05:47	0:05:50
185	1	9:10:10	3	9:15:59	9:16:02	0:05:49	0:05:52
186	1	9:10:11	6	9:16:02	9:16:08	0:05:51	0:05:57
187	1	9:10:12	3	9:16:08	9:16:11	0:05:56	0:05:59
188	10	9:10:22	6	9:16:11	9:16:17	0:05:49	0:05:55
189	5	9:10:27	6	9:16:17	9:16:23	0:05:50	0:05:56
190	3	9:10:30	9	9:16:23	9:16:32	0:05:53	0:06:02
191	3	9:10:33	3	9:16:32	9:16:35	0:05:59	0:06:02
192	1	9:10:34	3	9:16:35	9:16:38	0:06:01	0:06:04
193	5	9:10:39	9	9:16:38	9:16:47	0:05:59	0:06:08
194	1	9:10:40	3	9:16:47	9:16:50	0:06:07	0:06:10
195	3	9:10:43	3	9:16:50	9:16:53	0:06:07	0:06:10
196	3	9:10:46	9	9:16:53	9:17:02	0:06:07	0:06:16
197	1	9:10:47	3	9:17:02	9:17:05	0:06:15	0:06:18
198	3	9:10:50	3	9:17:05	9:17:08	0:06:15	0:06:18
199	10	9:11:00	6	9:17:08	9:17:14	0:06:08	0:06:14
200	1	9:11:01	3	9:17:14	9:17:17	0:06:13	0:06:16
201	3	9:11:04	6	9:17:17	9:17:23	0:06:13	0:06:19
202	1	9:11:05	3	9:17:23	9:17:26	0:06:18	0:06:21
203	10	9:11:15	3	9:17:26	9:17:29	0:06:11	0:06:14
204	1	9:11:16	3	9:17:29	9:17:32	0:06:13	0:06:16
205	1	9:11:17	9	9:17:32	9:17:41	0:06:15	0:06:24
206	1	9:11:18	3	9:17:41	9:17:44	0:06:23	0:06:26
207	3	9:11:21	3	9:17:44	9:17:47	0:06:23	0:06:26
208	1	9:11:22	6	9:17:47	9:17:53	0:06:25	0:06:31
209	10	9:11:32	9	9:17:53	9:18:02	0:06:21	0:06:30
210	10	9:11:42	6	9:18:02	9:18:08	0:06:20	0:06:26
211	3	9:11:45	3	9:18:08	9:18:11	0:06:23	0:06:26
212	10	9:11:55	6	9:18:11	9:18:17	0:06:16	0:06:22
213	1	9:11:56	3	9:18:17	9:18:20	0:06:21	0:06:24
214	3	9:11:59	6	9:18:20	9:18:26	0:06:21	0:06:27
215	10	9:12:09	6	9:18:26	9:18:32	0:06:17	0:06:23
216	5	9:12:14	3	9:18:32	9:18:35	0:06:18	0:06:21
217	10	9:12:24	3	9:18:35	9:18:38	0:06:11	0:06:14
218	3	9:12:27	6	9:18:38	9:18:44	0:06:11	0:06:17
219	1	9:12:28	3	9:18:44	9:18:47	0:06:16	0:06:19
220	1	9:12:29	6	9:18:47	9:18:53	0:06:18	0:06:24
221	1	9:12:30	6	9:18:53	9:18:59	0:06:23	0:06:29
222	10	9:12:40	6	9:18:59	9:19:05	0:06:19	0:06:25
223	3	9:12:43	6	9:19:05	9:19:11	0:06:22	0:06:28
224	10	9:12:53	3	9:19:11	9:19:14	0:06:18	0:06:21
225	3	9:12:56	3	9:19:14	9:19:17	0:06:18	0:06:21
226	1	9:12:57	3	9:19:17	9:19:20	0:06:20	0:06:23
227	1	9:12:58	9	9:19:20	9:19:29	0:06:22	0:06:31
228	10	9:13:08	9	9:19:29	9:19:38	0:06:21	0:06:30
229	10	9:13:18	3	9:19:38	9:19:41	0:06:20	0:06:23
230	1	9:13:19	3	9:19:41	9:19:44	0:06:22	0:06:25

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
231	3	9:13:22	6	9:19:44	9:19:50	0:06:22	0:06:28
232	1	9:13:23	3	9:19:50	9:19:53	0:06:27	0:06:30
233	5	9:13:28	6	9:19:53	9:19:59	0:06:25	0:06:31
234	3	9:13:31	6	9:19:59	9:20:05	0:06:28	0:06:34
235	1	9:13:32	9	9:20:05	9:20:14	0:06:33	0:06:42
236	3	9:13:35	3	9:20:14	9:20:17	0:06:39	0:06:42
237	1	9:13:36	9	9:20:17	9:20:26	0:06:41	0:06:50
238	3	9:13:39	3	9:20:26	9:20:29	0:06:47	0:06:50
239	3	9:13:42	3	9:20:29	9:20:32	0:06:47	0:06:50
240	5	9:13:47	6	9:20:32	9:20:38	0:06:45	0:06:51
241	3	9:13:50	6	9:20:38	9:20:44	0:06:48	0:06:54
242	1	9:13:51	3	9:20:44	9:20:47	0:06:53	0:06:56
243	1	9:13:52	3	9:20:47	9:20:50	0:06:55	0:06:58
244	10	9:14:02	3	9:20:50	9:20:53	0:06:48	0:06:51
245	10	9:14:12	9	9:20:53	9:21:02	0:06:41	0:06:50
246	10	9:14:22	3	9:21:02	9:21:05	0:06:40	0:06:43
247	1	9:14:23	6	9:21:05	9:21:11	0:06:42	0:06:48
248	1	9:14:24	6	9:21:11	9:21:17	0:06:47	0:06:53
249	3	9:14:27	9	9:21:17	9:21:26	0:06:50	0:06:59
250	3	9:14:30	3	9:21:26	9:21:29	0:06:56	0:06:59
251	1	9:14:31	6	9:21:29	9:21:35	0:06:58	0:07:04
252	1	9:14:32	6	9:21:35	9:21:41	0:07:03	0:07:09
253	1	9:14:33	6	9:21:41	9:21:47	0:07:08	0:07:14
254	1	9:14:34	3	9:21:47	9:21:50	0:07:13	0:07:16
255	3	9:14:37	6	9:21:50	9:21:56	0:07:13	0:07:19
256	3	9:14:40	6	9:21:56	9:22:02	0:07:16	0:07:22
257	5	9:14:45	6	9:22:02	9:22:08	0:07:17	0:07:23
258	10	9:14:55	6	9:22:08	9:22:14	0:07:13	0:07:19
259	1	9:14:56	3	9:22:14	9:22:17	0:07:18	0:07:21
260	3	9:14:59	6	9:22:17	9:22:23	0:07:18	0:07:24
261	10	9:15:09	3	9:22:23	9:22:26	0:07:14	0:07:17
262	1	9:15:10	6	9:22:26	9:22:32	0:07:16	0:07:22
263	10	9:15:20	3	9:22:32	9:22:35	0:07:12	0:07:15
264	3	9:15:23	3	9:22:35	9:22:38	0:07:12	0:07:15
265	3	9:15:26	6	9:22:38	9:22:44	0:07:12	0:07:18
266	1	9:15:27	6	9:22:44	9:22:50	0:07:17	0:07:23
267	1	9:15:28	3	9:22:50	9:22:53	0:07:22	0:07:25
268	1	9:15:29	9	9:22:53	9:23:02	0:07:24	0:07:33
269	10	9:15:39	6	9:23:02	9:23:08	0:07:23	0:07:29
270	1	9:15:40	3	9:23:08	9:23:11	0:07:28	0:07:31
271	3	9:15:43	3	9:23:11	9:23:14	0:07:28	0:07:31
272	3	9:15:46	9	9:23:14	9:23:23	0:07:28	0:07:37
273	1	9:15:47	3	9:23:23	9:23:26	0:07:36	0:07:39
274	1	9:15:48	3	9:23:26	9:23:29	0:07:38	0:07:41
275	3	9:15:51	3	9:23:29	9:23:32	0:07:38	0:07:41
276	10	9:16:01	6	9:23:32	9:23:38	0:07:31	0:07:37
277	1	9:16:02	6	9:23:38	9:23:44	0:07:36	0:07:42
278	10	9:16:12	3	9:23:44	9:23:47	0:07:32	0:07:35
279	1	9:16:13	6	9:23:47	9:23:53	0:07:34	0:07:40
280	3	9:16:16	9	9:23:53	9:24:02	0:07:37	0:07:46
281	1	9:16:17	6	9:24:02	9:24:08	0:07:45	0:07:51
282	1	9:16:18	6	9:24:08	9:24:14	0:07:50	0:07:56
283	10	9:16:28	3	9:24:14	9:24:17	0:07:46	0:07:49
284	1	9:16:29	3	9:24:17	9:24:20	0:07:48	0:07:51
285	1	9:16:30	9	9:24:20	9:24:29	0:07:50	0:07:59
286	1	9:16:31	3	9:24:29	9:24:32	0:07:58	0:08:01
287	1	9:16:32	3	9:24:32	9:24:35	0:08:00	0:08:03
288	1	9:16:33	3	9:24:35	9:24:38	0:08:02	0:08:05

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
289	10	9:16:43	3	9:24:38	9:24:41	0:07:55	0:07:58
290	10	9:16:53	6	9:24:41	9:24:47	0:07:48	0:07:54
291	1	9:16:54	3	9:24:47	9:24:50	0:07:53	0:07:56
292	1	9:16:55	6	9:24:50	9:24:56	0:07:55	0:08:01
293	1	9:16:56	3	9:24:56	9:24:59	0:08:00	0:08:03
294	1	9:16:57	6	9:24:59	9:25:05	0:08:02	0:08:08
295	10	9:17:07	6	9:25:05	9:25:11	0:07:58	0:08:04
296	10	9:17:17	3	9:25:11	9:25:14	0:07:54	0:07:57
297	1	9:17:18	3	9:25:14	9:25:17	0:07:56	0:07:59
298	10	9:17:28	6	9:25:17	9:25:23	0:07:49	0:07:55
299	10	9:17:38	6	9:25:23	9:25:29	0:07:45	0:07:51
300	1	9:17:39	3	9:25:29	9:25:32	0:07:50	0:07:53
301	3	9:17:42	3	9:25:32	9:25:35	0:07:50	0:07:53
302	10	9:17:52	3	9:25:35	9:25:38	0:07:43	0:07:46
303	3	9:17:55	6	9:25:38	9:25:44	0:07:43	0:07:49
304	1	9:17:56	3	9:25:44	9:25:47	0:07:48	0:07:51
305	1	9:17:57	3	9:25:47	9:25:50	0:07:50	0:07:53
306	10	9:18:07	3	9:25:50	9:25:53	0:07:43	0:07:46
307	10	9:18:17	3	9:25:53	9:25:56	0:07:36	0:07:39
308	10	9:18:27	6	9:25:56	9:26:02	0:07:29	0:07:35
309	3	9:18:30	6	9:26:02	9:26:08	0:07:32	0:07:38
310	5	9:18:35	9	9:26:08	9:26:17	0:07:33	0:07:42
311	1	9:18:36	9	9:26:17	9:26:26	0:07:41	0:07:50
312	10	9:18:46	3	9:26:26	9:26:29	0:07:40	0:07:43
313	3	9:18:49	6	9:26:29	9:26:35	0:07:40	0:07:46
314	1	9:18:50	3	9:26:35	9:26:38	0:07:45	0:07:48
315	5	9:18:55	6	9:26:38	9:26:44	0:07:43	0:07:49
316	5	9:19:00	3	9:26:44	9:26:47	0:07:44	0:07:47
317	1	9:19:01	9	9:26:47	9:26:56	0:07:46	0:07:55
318	10	9:19:11	3	9:26:56	9:26:59	0:07:45	0:07:48
319	1	9:19:12	3	9:26:59	9:27:02	0:07:47	0:07:50
320	10	9:19:22	3	9:27:02	9:27:05	0:07:40	0:07:43
321	3	9:19:25	3	9:27:05	9:27:08	0:07:40	0:07:43
322	1	9:19:26	3	9:27:08	9:27:11	0:07:42	0:07:45
323	1	9:19:27	9	9:27:11	9:27:20	0:07:44	0:07:53
324	1	9:19:28	3	9:27:20	9:27:23	0:07:52	0:07:55
325	1	9:19:29	3	9:27:23	9:27:26	0:07:54	0:07:57
326	1	9:19:30	3	9:27:26	9:27:29	0:07:56	0:07:59
327	1	9:19:31	3	9:27:29	9:27:32	0:07:58	0:08:01
328	3	9:19:34	6	9:27:32	9:27:38	0:07:58	0:08:04
329	1	9:19:35	3	9:27:38	9:27:41	0:08:03	0:08:06
330	3	9:19:38	3	9:27:41	9:27:44	0:08:03	0:08:06
331	10	9:19:48	6	9:27:44	9:27:50	0:07:56	0:08:02
332	3	9:19:51	3	9:27:50	9:27:53	0:07:59	0:08:02
333	1	9:19:52	3	9:27:53	9:27:56	0:08:01	0:08:04
334	10	9:20:02	6	9:27:56	9:28:02	0:07:54	0:08:00
335	3	9:20:05	3	9:28:02	9:28:05	0:07:57	0:08:00
336	1	9:20:06	6	9:28:05	9:28:11	0:07:59	0:08:05
337	1	9:20:07	3	9:28:11	9:28:14	0:08:04	0:08:07
338	3	9:20:10	6	9:28:14	9:28:20	0:08:04	0:08:10
339	3	9:20:13	6	9:28:20	9:28:26	0:08:07	0:08:13
340	1	9:20:14	3	9:28:26	9:28:29	0:08:12	0:08:15
341	10	9:20:24	6	9:28:29	9:28:35	0:08:05	0:08:11
342	5	9:20:29	6	9:28:35	9:28:41	0:08:06	0:08:12
343	10	9:20:39	3	9:28:41	9:28:44	0:08:02	0:08:05
344	1	9:20:40	6	9:28:44	9:28:50	0:08:04	0:08:10
345	3	9:20:43	3	9:28:50	9:28:53	0:08:07	0:08:10
346	10	9:20:53	3	9:28:53	9:28:56	0:08:00	0:08:03

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
347	1	9:20:54	3	9:28:56	9:28:59	0:08:02	0:08:05
348	3	9:20:57	6	9:28:59	9:29:05	0:08:02	0:08:08
349	1	9:20:58	9	9:29:05	9:29:14	0:08:07	0:08:16
350	1	9:20:59	3	9:29:14	9:29:17	0:08:15	0:08:18
351	3	9:21:02	3	9:29:17	9:29:20	0:08:15	0:08:18
352	10	9:21:12	3	9:29:20	9:29:23	0:08:08	0:08:11
353	5	9:21:17	3	9:29:23	9:29:26	0:08:06	0:08:09
354	1	9:21:18	3	9:29:26	9:29:29	0:08:08	0:08:11
355	1	9:21:19	9	9:29:29	9:29:38	0:08:10	0:08:19
356	10	9:21:29	6	9:29:38	9:29:44	0:08:09	0:08:15
357	3	9:21:32	9	9:29:44	9:29:53	0:08:12	0:08:21
358	10	9:21:42	3	9:29:53	9:29:56	0:08:11	0:08:14
359	1	9:21:43	6	9:29:56	9:30:02	0:08:13	0:08:19
360	10	9:21:53	6	9:30:02	9:30:08	0:08:09	0:08:15
361	3	9:21:56	3	9:30:08	9:30:11	0:08:12	0:08:15
362	1	9:21:57	6	9:30:11	9:30:17	0:08:14	0:08:20
363	10	9:22:07	3	9:30:17	9:30:20	0:08:10	0:08:13
364	10	9:22:17	3	9:30:20	9:30:23	0:08:03	0:08:06
365	1	9:22:18	6	9:30:23	9:30:29	0:08:05	0:08:11
366	1	9:22:19	6	9:30:29	9:30:35	0:08:10	0:08:16
367	3	9:22:22	6	9:30:35	9:30:41	0:08:13	0:08:19
368	1	9:22:23	3	9:30:41	9:30:44	0:08:18	0:08:21
369	3	9:22:26	9	9:30:44	9:30:53	0:08:18	0:08:27
370	3	9:22:29	6	9:30:53	9:30:59	0:08:24	0:08:30
371	5	9:22:34	6	9:30:59	9:31:05	0:08:25	0:08:31
372	10	9:22:44	3	9:31:05	9:31:08	0:08:21	0:08:24
373	3	9:22:47	6	9:31:08	9:31:14	0:08:21	0:08:27
374	1	9:22:48	6	9:31:14	9:31:20	0:08:26	0:08:32
375	10	9:22:58	3	9:31:20	9:31:23	0:08:22	0:08:25
376	3	9:23:01	3	9:31:23	9:31:26	0:08:22	0:08:25
377	1	9:23:02	6	9:31:26	9:31:32	0:08:24	0:08:30
378	1	9:23:03	9	9:31:32	9:31:41	0:08:29	0:08:38
379	1	9:23:04	6	9:31:41	9:31:47	0:08:37	0:08:43
380	10	9:23:14	6	9:31:47	9:31:53	0:08:33	0:08:39
381	3	9:23:17	6	9:31:53	9:31:59	0:08:36	0:08:42
382	1	9:23:18	9	9:31:59	9:32:08	0:08:41	0:08:50
383	5	9:23:23	6	9:32:08	9:32:14	0:08:45	0:08:51
384	1	9:23:24	3	9:32:14	9:32:17	0:08:50	0:08:53
385	1	9:23:25	3	9:32:17	9:32:20	0:08:52	0:08:55
386	3	9:23:28	3	9:32:20	9:32:23	0:08:52	0:08:55
387	5	9:23:33	9	9:32:23	9:32:32	0:08:50	0:08:59
388	3	9:23:36	3	9:32:32	9:32:35	0:08:56	0:08:59
389	1	9:23:37	3	9:32:35	9:32:38	0:08:58	0:09:01
390	1	9:23:38	3	9:32:38	9:32:41	0:09:00	0:09:03
391	10	9:23:48	3	9:32:41	9:32:44	0:08:53	0:08:56
392	1	9:23:49	6	9:32:44	9:32:50	0:08:55	0:09:01
393	1	9:23:50	3	9:32:50	9:32:53	0:09:00	0:09:03
394	1	9:23:51	9	9:32:53	9:33:02	0:09:02	0:09:11
395	1	9:23:52	3	9:33:02	9:33:05	0:09:10	0:09:13
396	5	9:23:57	9	9:33:05	9:33:14	0:09:08	0:09:17
397	5	9:24:02	9	9:33:14	9:33:23	0:09:12	0:09:21
398	1	9:24:03	3	9:33:23	9:33:26	0:09:20	0:09:23
399	10	9:24:13	3	9:33:26	9:33:29	0:09:13	0:09:16
400	5	9:24:18	9	9:33:29	9:33:38	0:09:11	0:09:20
401	1	9:24:19	6	9:33:38	9:33:44	0:09:19	0:09:25
402	1	9:24:20	9	9:33:44	9:33:53	0:09:24	0:09:33
403	10	9:24:30	3	9:33:53	9:33:56	0:09:23	0:09:26
404	1	9:24:31	9	9:33:56	9:34:05	0:09:25	0:09:34

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
405	3	9:24:34	3	9:34:05	9:34:08	0:09:31	0:09:34
406	3	9:24:37	3	9:34:08	9:34:11	0:09:31	0:09:34
407	3	9:24:40	3	9:34:11	9:34:14	0:09:31	0:09:34
408	10	9:24:50	3	9:34:14	9:34:17	0:09:24	0:09:27
409	10	9:25:00	9	9:34:17	9:34:26	0:09:17	0:09:26
410	10	9:25:10	6	9:34:26	9:34:32	0:09:16	0:09:22
411	5	9:25:15	3	9:34:32	9:34:35	0:09:17	0:09:20
412	1	9:25:16	3	9:34:35	9:34:38	0:09:19	0:09:22
413	3	9:25:19	6	9:34:38	9:34:44	0:09:19	0:09:25
414	1	9:25:20	9	9:34:44	9:34:53	0:09:24	0:09:33
415	1	9:25:21	6	9:34:53	9:34:59	0:09:32	0:09:38
416	10	9:25:31	6	9:34:59	9:35:05	0:09:28	0:09:34
417	5	9:25:36	6	9:35:05	9:35:11	0:09:29	0:09:35
418	1	9:25:37	3	9:35:11	9:35:14	0:09:34	0:09:37
419	3	9:25:40	6	9:35:14	9:35:20	0:09:34	0:09:40
420	1	9:25:41	6	9:35:20	9:35:26	0:09:39	0:09:45
421	3	9:25:44	9	9:35:26	9:35:35	0:09:42	0:09:51
422	3	9:25:47	6	9:35:35	9:35:41	0:09:48	0:09:54
423	1	9:25:48	3	9:35:41	9:35:44	0:09:53	0:09:56
424	1	9:25:49	3	9:35:44	9:35:47	0:09:55	0:09:58
425	3	9:25:52	3	9:35:47	9:35:50	0:09:55	0:09:58
426	1	9:25:53	3	9:35:50	9:35:53	0:09:57	0:10:00
427	1	9:25:54	3	9:35:53	9:35:56	0:09:59	0:10:02
428	10	9:26:04	3	9:35:56	9:35:59	0:09:52	0:09:55
429	1	9:26:05	6	9:35:59	9:36:05	0:09:54	0:10:00
430	10	9:26:15	6	9:36:05	9:36:11	0:09:50	0:09:56
431	1	9:26:16	6	9:36:11	9:36:17	0:09:55	0:10:01
432	3	9:26:19	6	9:36:17	9:36:23	0:09:58	0:10:04
433	3	9:26:22	3	9:36:23	9:36:26	0:10:01	0:10:04
434	10	9:26:32	3	9:36:26	9:36:29	0:09:54	0:09:57
435	3	9:26:35	6	9:36:29	9:36:35	0:09:54	0:10:00
436	3	9:26:38	9	9:36:35	9:36:44	0:09:57	0:10:06
437	10	9:26:48	3	9:36:44	9:36:47	0:09:56	0:09:59
438	1	9:26:49	3	9:36:47	9:36:50	0:09:58	0:10:01
439	1	9:26:50	3	9:36:50	9:36:53	0:10:00	0:10:03
440	3	9:26:53	6	9:36:53	9:36:59	0:10:00	0:10:06
441	1	9:26:54	3	9:36:59	9:37:02	0:10:05	0:10:08
442	1	9:26:55	6	9:37:02	9:37:08	0:10:07	0:10:13
443	3	9:26:58	6	9:37:08	9:37:14	0:10:10	0:10:16
444	3	9:27:01	6	9:37:14	9:37:20	0:10:13	0:10:19
445	1	9:27:02	6	9:37:20	9:37:26	0:10:18	0:10:24
446	5	9:27:07	6	9:37:26	9:37:32	0:10:19	0:10:25
447	1	9:27:08	3	9:37:32	9:37:35	0:10:24	0:10:27
448	10	9:27:18	6	9:37:35	9:37:41	0:10:17	0:10:23
449	5	9:27:23	3	9:37:41	9:37:44	0:10:18	0:10:21
450	10	9:27:33	3	9:37:44	9:37:47	0:10:11	0:10:14
451	1	9:27:34	3	9:37:47	9:37:50	0:10:13	0:10:16
452	1	9:27:35	9	9:37:50	9:37:59	0:10:15	0:10:24
453	1	9:27:36	6	9:37:59	9:38:05	0:10:23	0:10:29
454	3	9:27:39	3	9:38:05	9:38:08	0:10:26	0:10:29
455	3	9:27:42	3	9:38:08	9:38:11	0:10:26	0:10:29
456	3	9:27:45	6	9:38:11	9:38:17	0:10:26	0:10:32
457	1	9:27:46	3	9:38:17	9:38:20	0:10:31	0:10:34
458	3	9:27:49	9	9:38:20	9:38:29	0:10:31	0:10:40
459	3	9:27:52	6	9:38:29	9:38:35	0:10:37	0:10:43
460	10	9:28:02	9	9:38:35	9:38:44	0:10:33	0:10:42
461	1	9:28:03	6	9:38:44	9:38:50	0:10:41	0:10:47
462	1	9:28:04	6	9:38:50	9:38:56	0:10:46	0:10:52

Table (A-3)
Queueing Simulation under condition III

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
463	1	9:28:05	9	9:38:56	9:39:05	0:10:51	0:11:00
464	3	9:28:08	3	9:39:05	9:39:08	0:10:57	0:11:00
465	5	9:28:13	6	9:39:08	9:39:14	0:10:55	0:11:01
466	1	9:28:14	6	9:39:14	9:39:20	0:11:00	0:11:06
467	3	9:28:17	3	9:39:20	9:39:23	0:11:03	0:11:06
468	1	9:28:18	3	9:39:23	9:39:26	0:11:05	0:11:08
469	3	9:28:21	3	9:39:26	9:39:29	0:11:05	0:11:08
470	3	9:28:24	3	9:39:29	9:39:32	0:11:05	0:11:08
471	1	9:28:25	9	9:39:32	9:39:41	0:11:07	0:11:16
472	1	9:28:26	9	9:39:41	9:39:50	0:11:15	0:11:24
473	10	9:28:36	6	9:39:50	9:39:56	0:11:14	0:11:20
474	1	9:28:37	3	9:39:56	9:39:59	0:11:19	0:11:22
475	1	9:28:38	3	9:39:59	9:40:02	0:11:21	0:11:24
476	3	9:28:41	3	9:40:02	9:40:05	0:11:21	0:11:24
477	3	9:28:44	3	9:40:05	9:40:08	0:11:21	0:11:24
478	1	9:28:45	3	9:40:08	9:40:11	0:11:23	0:11:26
479	10	9:28:55	6	9:40:11	9:40:17	0:11:16	0:11:22
480	3	9:28:58	9	9:40:17	9:40:26	0:11:19	0:11:28
481	1	9:28:59	6	9:40:26	9:40:32	0:11:27	0:11:33
482	3	9:29:02	9	9:40:32	9:40:41	0:11:30	0:11:39
483	3	9:29:05	3	9:40:41	9:40:44	0:11:36	0:11:39
484	1	9:29:06	6	9:40:44	9:40:50	0:11:38	0:11:44
485	10	9:29:16	9	9:40:50	9:40:59	0:11:34	0:11:43
486	10	9:29:26	3	9:40:59	9:41:02	0:11:33	0:11:36
487	3	9:29:29	3	9:41:02	9:41:05	0:11:33	0:11:36
488	5	9:29:34	3	9:41:05	9:41:08	0:11:31	0:11:34
489	10	9:29:44	6	9:41:08	9:41:14	0:11:24	0:11:30
490	1	9:29:45	3	9:41:14	9:41:17	0:11:29	0:11:32
491	5	9:29:50	3	9:41:17	9:41:20	0:11:27	0:11:30
492	1	9:29:51	3	9:41:20	9:41:23	0:11:29	0:11:32
493	1	9:29:52	6	9:41:23	9:41:29	0:11:31	0:11:37
494	5	9:29:57	6	9:41:29	9:41:35	0:11:32	0:11:38
495	1	9:29:58	3	9:41:35	9:41:38	0:11:37	0:11:40
496	10	9:30:08	6	9:41:38	9:41:44	0:11:30	0:11:36
497	1	9:30:09	3	9:41:44	9:41:47	0:11:35	0:11:38
498	3	9:30:12	6	9:41:47	9:41:53	0:11:35	0:11:41
499	3	9:30:15	3	9:41:53	9:41:56	0:11:38	0:11:41
500	10	9:30:25	6	9:41:56	9:42:02	0:11:31	0:11:37
501	1	9:30:26	6	9:42:02	9:42:08	0:11:36	0:11:42
502	1	9:30:27	6	9:42:08	9:42:14	0:11:41	0:11:47
503	3	9:30:30	3	9:42:14	9:42:17	0:11:44	0:11:47
504	1	9:30:31	3	9:42:17	9:42:20	0:11:46	0:11:49
505	10	9:30:41	9	9:42:20	9:42:29	0:11:39	0:11:48
506	1	9:30:42	6	9:42:29	9:42:35	0:11:47	0:11:53
507	10	9:30:52	3	9:42:35	9:42:38	0:11:43	0:11:46
508	1	9:30:53	6	9:42:38	9:42:44	0:11:45	0:11:51
509	3	9:30:56	6	9:42:44	9:42:50	0:11:48	0:11:54
510	5	9:31:01	3	9:42:50	9:42:53	0:11:49	0:11:52
511	10	9:31:11	9	9:42:53	9:43:02	0:11:42	0:11:51
512	1	9:31:12	6	9:43:02	9:43:08	0:11:50	0:11:56
513	3	9:31:15	3	9:43:08	9:43:11	0:11:53	0:11:56
514	1	9:31:16	3	9:43:11	9:43:14	0:11:55	0:11:58
515	3	9:31:19	3	9:43:14	9:43:17	0:11:55	0:11:58
516	3	9:31:22	6	9:43:17	9:43:23	0:11:55	0:12:01
517	5	9:31:27	6	9:43:23	9:43:29	0:11:56	0:12:02
518	1	9:31:28	3	9:43:29	9:43:32	0:12:01	0:12:04
519	3	9:31:31	3	9:43:32	9:43:35	0:12:01	0:12:04
520	1	9:31:32	3	9:43:35	9:43:38	0:12:03	0:12:06

#	Time (min)	Time (hr:min)	Time (min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
start		9:00															
1	3	9:00:03	3	9:00:03	9:00:06											0:00:00	0:00:03
2	3	9:00:06	3	9:00:06	9:00:09											0:00:00	0:00:03
3	3	9:00:09	3	9:00:09	9:00:12											0:00:00	0:00:03
4	9	9:00:18	9	9:00:18	9:00:27											0:00:00	0:00:09
5	3	9:00:21	3			9:00:21	9:00:24									0:00:00	0:00:03
6	15	9:00:36	3	9:00:36	9:00:39											0:00:00	0:00:03
7	9	9:00:45	6	9:00:45	9:00:51											0:00:00	0:00:06
8	3	9:00:48	3			9:00:48	9:00:51									0:00:00	0:00:03
9	3	9:00:51	6	9:00:51	9:00:57											0:00:00	0:00:06
10	3	9:00:54	9			9:00:54	9:01:03									0:00:00	0:00:09
11	3	9:00:57	6	9:00:57	9:01:03											0:00:00	0:00:06
12	3	9:01:00	6					9:01:00	9:01:06							0:00:00	0:00:06
13	15	9:01:15	6	9:01:15	9:01:21											0:00:00	0:00:06
14	3	9:01:18	3			9:01:18	9:01:21									0:00:00	0:00:03
15	9	9:01:27	3	9:01:27	9:01:30											0:00:00	0:00:03
16	3	9:01:30	3	9:01:30	9:01:33											0:00:00	0:00:03
17	30	9:02:00	3	9:02:00	9:02:03											0:00:00	0:00:03
18	9	9:02:09	3	9:02:09	9:02:12											0:00:00	0:00:03
19	9	9:02:18	3	9:02:18	9:02:21											0:00:00	0:00:03
20	9	9:02:27	3	9:02:27	9:02:30											0:00:00	0:00:03
21	9	9:02:36	3	9:02:36	9:02:39											0:00:00	0:00:03
22	3	9:02:39	3	9:02:39	9:02:42											0:00:00	0:00:03
23	3	9:02:42	3	9:02:42	9:02:45											0:00:00	0:00:03
24	9	9:02:51	9	9:02:51	9:03:00											0:00:00	0:00:09
25	3	9:02:54	3			9:02:54	9:02:57									0:00:00	0:00:03
26	15	9:03:09	3	9:03:09	9:03:12											0:00:00	0:00:03
27	9	9:03:18	6	9:03:18	9:03:24											0:00:00	0:00:06
28	3	9:03:21	9			9:03:21	9:03:30									0:00:00	0:00:09
29	3	9:03:24	3	9:03:24	9:03:27											0:00:00	0:00:03
30	3	9:03:27	3	9:03:27	9:03:30											0:00:00	0:00:03
31	30	9:03:57	6	9:03:57	9:04:03											0:00:00	0:00:06
32	30	9:04:27	3	9:04:27	9:04:30											0:00:00	0:00:03
33	3	9:04:30	3	9:04:30	9:04:33											0:00:00	0:00:03
34	30	9:05:00	9	9:05:00	9:05:09											0:00:00	0:00:09
35	9	9:05:09	3	9:05:09	9:05:12											0:00:00	0:00:03
36	9	9:05:18	3	9:05:18	9:05:21											0:00:00	0:00:03
37	9	9:05:27	9	9:05:27	9:05:36											0:00:00	0:00:09
38	3	9:05:30	6			9:05:30	9:05:36									0:00:00	0:00:06
39	3	9:05:33	3					9:05:33	9:05:36							0:00:00	0:00:03
40	3	9:05:36	3	9:05:36	9:05:39											0:00:00	0:00:03
41	30	9:06:06	9	9:06:06	9:06:15											0:00:00	0:00:09
42	9	9:06:15	6	9:06:15	9:06:21											0:00:00	0:00:06
43	3	9:06:18	3			9:06:18	9:06:21									0:00:00	0:00:03
44	3	9:06:21	9	9:06:21	9:06:30											0:00:00	0:00:09
45	3	9:06:24	6			9:06:24	9:06:30									0:00:00	0:00:06
46	30	9:06:54	6	9:06:54	9:07:00											0:00:00	0:00:06
47	9	9:07:03	6	9:07:03	9:07:09											0:00:00	0:00:06
48	9	9:07:12	3	9:07:12	9:07:15											0:00:00	0:00:03

	[redacted] (min)	[redacted] (hr:min)	[redacted] (min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	Time (hr:min)	Time (hr:min)
49	9	9:07:21	6	9:07:21	9:07:27										0:00:00	0:00:06
50	3	9:07:24	3			9:07:24	9:07:27								0:00:00	0:00:03
51	15	9:07:39	3	9:07:39	9:07:42										0:00:00	0:00:03
52	3	9:07:42	3	9:07:42	9:07:45										0:00:00	0:00:03
53	30	9:08:12	3	9:08:12	9:08:15										0:00:00	0:00:03
54	30	9:08:42	3	9:08:42	9:08:45										0:00:00	0:00:03
55	9	9:08:51	6	9:08:51	9:08:57										0:00:00	0:00:06
56	9	9:09:00	6	9:09:00	9:09:06										0:00:00	0:00:06
57	3	9:09:03	6			9:09:03	9:09:09								0:00:00	0:00:06
58	15	9:09:18	3	9:09:18	9:09:21										0:00:00	0:00:03
59	3	9:09:21	9	9:09:21	9:09:30										0:00:00	0:00:09
60	9	9:09:30	6	9:09:30	9:09:36										0:00:00	0:00:06
61	3	9:09:33	6			9:09:33	9:09:39								0:00:00	0:00:06
62	3	9:09:36	3	9:09:36	9:09:39										0:00:00	0:00:03
63	9	9:09:45	3	9:09:45	9:09:48										0:00:00	0:00:03
64	3	9:09:48	9	9:09:48	9:09:57										0:00:00	0:00:09
65	3	9:09:51	9			9:09:51	9:10:00								0:00:00	0:00:09
66	3	9:09:54	3					9:09:54	9:09:57						0:00:00	0:00:03
67	30	9:10:24	6	9:10:24	9:10:30										0:00:00	0:00:06
68	9	9:10:33	3	9:10:33	9:10:36										0:00:00	0:00:03
69	3	9:10:36	6	9:10:36	9:10:42										0:00:00	0:00:06
70	9	9:10:45	9	9:10:45	9:10:54										0:00:00	0:00:09
71	3	9:10:48	3			9:10:48	9:10:51								0:00:00	0:00:03
72	9	9:10:57	6	9:10:57	9:11:03										0:00:00	0:00:06
73	3	9:11:00	3			9:11:00	9:11:03								0:00:00	0:00:03
74	9	9:11:09	9	9:11:09	9:11:18										0:00:00	0:00:09
75	30	9:11:39	3	9:11:39	9:11:42										0:00:00	0:00:03
76	30	9:12:09	3	9:12:09	9:12:12										0:00:00	0:00:03
77	3	9:12:12	3	9:12:12	9:12:15										0:00:00	0:00:03
78	3	9:12:15	6	9:12:15	9:12:21										0:00:00	0:00:06
79	9	9:12:24	3	9:12:24	9:12:27										0:00:00	0:00:03
80	3	9:12:27	6	9:12:27	9:12:33										0:00:00	0:00:06
81	3	9:12:30	3			9:12:30	9:12:33								0:00:00	0:00:03
82	3	9:12:33	6	9:12:33	9:12:39										0:00:00	0:00:06
83	3	9:12:36	3			9:12:36	9:12:39								0:00:00	0:00:03
84	3	9:12:39	9	9:12:39	9:12:48										0:00:00	0:00:09
85	9	9:12:48	3	9:12:48	9:12:51										0:00:00	0:00:03
86	3	9:12:51	9	9:12:51	9:13:00										0:00:00	0:00:09
87	9	9:13:00	6	9:13:00	9:13:06										0:00:00	0:00:06
88	9	9:13:09	3	9:13:09	9:13:12										0:00:00	0:00:03
89	9	9:13:18	3	9:13:18	9:13:21										0:00:00	0:00:03
90	30	9:13:48	6	9:13:48	9:13:54										0:00:00	0:00:06
91	3	9:13:51	6			9:13:51	9:13:57								0:00:00	0:00:06
92	9	9:14:00	6	9:14:00	9:14:06										0:00:00	0:00:06
93	3	9:14:03	6			9:14:03	9:14:09								0:00:00	0:00:06
94	9	9:14:12	3	9:14:12	9:14:15										0:00:00	0:00:03
95	15	9:14:27	6	9:14:27	9:14:33										0:00:00	0:00:06
96	9	9:14:36	3	9:14:36	9:14:39										0:00:00	0:00:03
97	3	9:14:39	6	9:14:39	9:14:45										0:00:00	0:00:06

#	Time (min)	Time (hr:min)	Time (min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
98	3	9:14:42	3			9:14:42	9:14:45									0:00:00	0:00:03
99	30	9:15:12	3	9:15:12	9:15:15											0:00:00	0:00:03
100	30	9:15:42	9	9:15:42	9:15:51											0:00:00	0:00:09
101	30	9:16:12	3	9:16:12	9:16:15											0:00:00	0:00:03
102	3	9:16:15	6	9:16:15	9:16:21											0:00:00	0:00:06
103	9	9:16:24	6	9:16:24	9:16:30											0:00:00	0:00:06
104	3	9:16:27	9			9:16:27	9:16:36									0:00:00	0:00:09
105	9	9:16:36	9	9:16:36	9:16:45											0:00:00	0:00:09
106	30	9:17:06	9	9:17:06	9:17:15											0:00:00	0:00:09
107	3	9:17:09	9			9:17:09	9:17:18									0:00:00	0:00:09
108	15	9:17:24	6	9:17:24	9:17:30											0:00:00	0:00:06
109	3	9:17:27	6			9:17:27	9:17:33									0:00:00	0:00:06
110	15	9:17:42	3	9:17:42	9:17:45											0:00:00	0:00:03
111	3	9:17:45	9	9:17:45	9:17:54											0:00:00	0:00:09
112	3	9:17:48	6			9:17:48	9:17:54									0:00:00	0:00:06
113	3	9:17:51	9					9:17:51	9:18:00							0:00:00	0:00:09
114	3	9:17:54	6	9:17:54	9:18:00											0:00:00	0:00:06
115	3	9:17:57	6			9:17:57	9:18:03									0:00:00	0:00:06
116	3	9:18:00	9	9:18:00	9:18:09											0:00:00	0:00:09
117	3	9:18:03	3			9:18:03	9:18:06									0:00:00	0:00:03
118	30	9:18:33	3	9:18:33	9:18:36											0:00:00	0:00:03
119	30	9:19:03	6	9:19:03	9:19:09											0:00:00	0:00:06
120	15	9:19:18	3	9:19:18	9:19:21											0:00:00	0:00:03
121	9	9:19:27	3	9:19:27	9:19:30											0:00:00	0:00:03
122	30	9:19:57	3	9:19:57	9:20:00											0:00:00	0:00:03
123	15	9:20:12	9	9:20:12	9:20:21											0:00:00	0:00:09
124	3	9:20:15	6			9:20:15	9:20:21									0:00:00	0:00:06
125	3	9:20:18	3					9:20:18	9:20:21							0:00:00	0:00:03
126	3	9:20:21	3	9:20:21	9:20:24											0:00:00	0:00:03
127	15	9:20:36	3	9:20:36	9:20:39											0:00:00	0:00:03
128	30	9:21:06	3	9:21:06	9:21:09											0:00:00	0:00:03
129	9	9:21:15	3	9:21:15	9:21:18											0:00:00	0:00:03
130	15	9:21:30	3	9:21:30	9:21:33											0:00:00	0:00:03
131	3	9:21:33	3	9:21:33	9:21:36											0:00:00	0:00:03
132	15	9:21:48	6	9:21:48	9:21:54											0:00:00	0:00:06
133	3	9:21:51	9			9:21:51	9:22:00									0:00:00	0:00:09
134	3	9:21:54	3	9:21:54	9:21:57											0:00:00	0:00:03
135	30	9:22:24	6	9:22:24	9:22:30											0:00:00	0:00:06
136	30	9:22:54	3	9:22:54	9:22:57											0:00:00	0:00:03
137	30	9:23:24	3	9:23:24	9:23:27											0:00:00	0:00:03
138	3	9:23:27	9	9:23:27	9:23:36											0:00:00	0:00:09
139	30	9:23:57	6	9:23:57	9:24:03											0:00:00	0:00:06
140	3	9:24:00	6			9:24:00	9:24:06									0:00:00	0:00:06
141	3	9:24:03	3	9:24:03	9:24:06											0:00:00	0:00:03
142	3	9:24:06	3	9:24:06	9:24:09											0:00:00	0:00:03
143	9	9:24:15	3	9:24:15	9:24:18											0:00:00	0:00:03
144	9	9:24:24	3	9:24:24	9:24:27											0:00:00	0:00:03
145	15	9:24:39	3	9:24:39	9:24:42											0:00:00	0:00:03
146	30	9:25:09	6	9:25:09	9:25:15											0:00:00	0:00:06

	[illegible] (min)	[illegible] (hr:min)	[illegible] (min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	Time (hr:min)	Time (hr:min)
147	3	9:25:12	3			9:25:12	9:25:15								0:00:00	0:00:03
148	30	9:25:42	9	9:25:42	9:25:51										0:00:00	0:00:09
149	15	9:25:57	6	9:25:57	9:26:03										0:00:00	0:00:06
150	3	9:26:00	6			9:26:00	9:26:06								0:00:00	0:00:06
151	9	9:26:09	6	9:26:09	9:26:15										0:00:00	0:00:06
152	9	9:26:18	6	9:26:18	9:26:24										0:00:00	0:00:06
153	9	9:26:27	6	9:26:27	9:26:33										0:00:00	0:00:06
154	3	9:26:30	3			9:26:30	9:26:33								0:00:00	0:00:03
155	9	9:26:39	6	9:26:39	9:26:45										0:00:00	0:00:06
156	3	9:26:42	3			9:26:42	9:26:45								0:00:00	0:00:03
157	3	9:26:45	3	9:26:45	9:26:48										0:00:00	0:00:03
158	3	9:26:48	3	9:26:48	9:26:51										0:00:00	0:00:03
159	30	9:27:18	6	9:27:18	9:27:24										0:00:00	0:00:06
160	9	9:27:27	3	9:27:27	9:27:30										0:00:00	0:00:03
161	3	9:27:30	6	9:27:30	9:27:36										0:00:00	0:00:06
162	30	9:28:00	3	9:28:00	9:28:03										0:00:00	0:00:03
163	30	9:28:30	6	9:28:30	9:28:36										0:00:00	0:00:06
164	9	9:28:39	9	9:28:39	9:28:48										0:00:00	0:00:09
165	9	9:28:48	3	9:28:48	9:28:51										0:00:00	0:00:03
166	3	9:28:51	3	9:28:51	9:28:54										0:00:00	0:00:03
167	3	9:28:54	9	9:28:54	9:29:03										0:00:00	0:00:09
168	15	9:29:09	3	9:29:09	9:29:12										0:00:00	0:00:03
169	9	9:29:18	9	9:29:18	9:29:27										0:00:00	0:00:09
170	30	9:29:48	6	9:29:48	9:29:54										0:00:00	0:00:06
171	3	9:29:51	3			9:29:51	9:29:54								0:00:00	0:00:03
172	3	9:29:54	3	9:29:54	9:29:57										0:00:00	0:00:03
173	3	9:29:57	3	9:29:57	9:30:00										0:00:00	0:00:03
174	3	9:30:00	3	9:30:00	9:30:03										0:00:00	0:00:03
175	3	9:30:03	3	9:30:03	9:30:06										0:00:00	0:00:03
176	3	9:30:06	6	9:30:06	9:30:12										0:00:00	0:00:06
177	3	9:30:09	6			9:30:09	9:30:15								0:00:00	0:00:06
178	9	9:30:18	9	9:30:18	9:30:27										0:00:00	0:00:09
179	3	9:30:21	3			9:30:21	9:30:24								0:00:00	0:00:03
180	3	9:30:24	6			9:30:24	9:30:30								0:00:00	0:00:06
181	30	9:30:54	6	9:30:54	9:31:00										0:00:00	0:00:06
182	30	9:31:24	6	9:31:24	9:31:30										0:00:00	0:00:06
183	3	9:31:27	3			9:31:27	9:31:30								0:00:00	0:00:03
184	3	9:31:30	9	9:31:30	9:31:39										0:00:00	0:00:09
185	9	9:31:39	3	9:31:39	9:31:42										0:00:00	0:00:03
186	9	9:31:48	9	9:31:48	9:31:57										0:00:00	0:00:09
187	3	9:31:51	3			9:31:51	9:31:54								0:00:00	0:00:03
188	9	9:32:00	6	9:32:00	9:32:06										0:00:00	0:00:06
189	30	9:32:30	6	9:32:30	9:32:36										0:00:00	0:00:06
190	9	9:32:39	6	9:32:39	9:32:45										0:00:00	0:00:06
191	30	9:33:09	3	9:33:09	9:33:12										0:00:00	0:00:03
192	9	9:33:18	6	9:33:18	9:33:24										0:00:00	0:00:06
193	15	9:33:33	3	9:33:33	9:33:36										0:00:00	0:00:03
194	9	9:33:42	6	9:33:42	9:33:48										0:00:00	0:00:06
195	3	9:33:45	9			9:33:45	9:33:54								0:00:00	0:00:09

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Time (hr:min)	Time (hr:min)
																0:00:00	0:00:06
196	30	9:34:15	6	9:34:15	9:34:21											0:00:00	0:00:03
197	30	9:34:45	3	9:34:45	9:34:48											0:00:00	0:00:03
198	9	9:34:54	3	9:34:54	9:34:57											0:00:00	0:00:06
199	9	9:35:03	6	9:35:03	9:35:09											0:00:00	0:00:03
200	30	9:35:33	3	9:35:33	9:35:36											0:00:00	0:00:09
201	3	9:35:36	9	9:35:36	9:35:45											0:00:00	0:00:03
202	30	9:36:06	3	9:36:06	9:36:09											0:00:00	0:00:09
203	30	9:36:36	9	9:36:36	9:36:45											0:00:00	0:00:03
204	30	9:37:06	3	9:37:06	9:37:09											0:00:00	0:00:03
205	3	9:37:09	3	9:37:09	9:37:12											0:00:00	0:00:03
206	3	9:37:12	3	9:37:12	9:37:15											0:00:00	0:00:06
207	3	9:37:15	6	9:37:15	9:37:21											0:00:00	0:00:06
208	9	9:37:24	6	9:37:24	9:37:30											0:00:00	0:00:03
209	3	9:37:27	3			9:37:27	9:37:30									0:00:00	0:00:03
210	30	9:37:57	3	9:37:57	9:38:00											0:00:00	0:00:03
211	30	9:38:27	3	9:38:27	9:38:30											0:00:00	0:00:09
212	3	9:38:30	9	9:38:30	9:38:39											0:00:00	0:00:06
213	15	9:38:45	6	9:38:45	9:38:51											0:00:00	0:00:06
214	9	9:38:54	6	9:38:54	9:39:00											0:00:00	0:00:06
215	30	9:39:24	6	9:39:24	9:39:30											0:00:00	0:00:09
216	3	9:39:27	9			9:39:27	9:39:36									0:00:00	0:00:03
217	9	9:39:36	3	9:39:36	9:39:39											0:00:00	0:00:09
218	3	9:39:39	9	9:39:39	9:39:48											0:00:00	0:00:06
219	9	9:39:48	6	9:39:48	9:39:54											0:00:00	0:00:03
220	9	9:39:57	3	9:39:57	9:40:00											0:00:00	0:00:06
221	3	9:40:00	6	9:40:00	9:40:06											0:00:00	0:00:09
222	3	9:40:03	9			9:40:03	9:40:12									0:00:00	0:00:09
223	3	9:40:06	9	9:40:06	9:40:15											0:00:00	0:00:03
224	30	9:40:36	3	9:40:36	9:40:39											0:00:00	0:00:03
225	30	9:41:06	3	9:41:06	9:41:09											0:00:00	0:00:03
226	3	9:41:09	3	9:41:09	9:41:12											0:00:00	0:00:06
227	30	9:41:39	6	9:41:39	9:41:45											0:00:00	0:00:09
228	9	9:41:48	9	9:41:48	9:41:57											0:00:00	0:00:06
229	3	9:41:51	6			9:41:51	9:41:57									0:00:00	0:00:06
230	3	9:41:54	6					9:41:54	9:42:00							0:00:00	0:00:09
231	3	9:41:57	9	9:41:57	9:42:06											0:00:00	0:00:03
232	3	9:42:00	3			9:42:00	9:42:03									0:00:00	0:00:03
233	3	9:42:03	3			9:42:03	9:42:06									0:00:00	0:00:06
234	30	9:42:33	6	9:42:33	9:42:39											0:00:00	0:00:09
235	30	9:43:03	9	9:43:03	9:43:12											0:00:00	0:00:03
236	3	9:43:06	3			9:43:06	9:43:09									0:00:00	0:00:06
237	3	9:43:09	6			9:43:09	9:43:15									0:00:00	0:00:03
238	9	9:43:18	3	9:43:18	9:43:21											0:00:00	0:00:09
239	30	9:43:48	9	9:43:48	9:43:57											0:00:00	0:00:03
240	9	9:43:57	3	9:43:57	9:44:00											0:00:00	0:00:03
241	3	9:44:00	3	9:44:00	9:44:03											0:00:00	0:00:03
242	3	9:44:03	3	9:44:03	9:44:06											0:00:00	0:00:03
243	3	9:44:06	3	9:44:06	9:44:09											0:00:00	0:00:06
244	3	9:44:09	6	9:44:09	9:44:15												

	[redacted] (min)	[redacted] (hr:min)	[redacted] (min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)	[redacted] (hr:min)
245	3	9:44:12	6			9:44:12	9:44:18									0:00:00	0:00:06
246	15	9:44:27	6	9:44:27	9:44:33											0:00:00	0:00:06
247	3	9:44:30	6			9:44:30	9:44:36									0:00:00	0:00:06
248	30	9:45:00	3	9:45:00	9:45:03											0:00:00	0:00:03
249	9	9:45:09	3	9:45:09	9:45:12											0:00:00	0:00:03
250	30	9:45:39	3	9:45:39	9:45:42											0:00:00	0:00:03
251	9	9:45:48	9	9:45:48	9:45:57											0:00:00	0:00:09
252	3	9:45:51	6			9:45:51	9:45:57									0:00:00	0:00:06
253	9	9:46:00	3	9:46:00	9:46:03											0:00:00	0:00:03
254	3	9:46:03	3	9:46:03	9:46:06											0:00:00	0:00:03
255	9	9:46:12	3	9:46:12	9:46:15											0:00:00	0:00:03
256	3	9:46:15	3	9:46:15	9:46:18											0:00:00	0:00:03
257	15	9:46:30	6	9:46:30	9:46:36											0:00:00	0:00:06
258	30	9:47:00	6	9:47:00	9:47:06											0:00:00	0:00:06
259	3	9:47:03	6			9:47:03	9:47:09									0:00:00	0:00:06
260	3	9:47:06	3	9:47:06	9:47:09											0:00:00	0:00:03
261	30	9:47:36	6	9:47:36	9:47:42											0:00:00	0:00:06
262	9	9:47:45	6	9:47:45	9:47:51											0:00:00	0:00:06
263	3	9:47:48	6			9:47:48	9:47:54									0:00:00	0:00:06
264	3	9:47:51	3	9:47:51	9:47:54											0:00:00	0:00:03
265	9	9:48:00	6	9:48:00	9:48:06											0:00:00	0:00:06
266	3	9:48:03	3			9:48:03	9:48:06									0:00:00	0:00:03
267	9	9:48:12	3	9:48:12	9:48:15											0:00:00	0:00:03
268	9	9:48:21	3	9:48:21	9:48:24											0:00:00	0:00:03
269	9	9:48:30	3	9:48:30	9:48:33											0:00:00	0:00:03
270	3	9:48:33	6	9:48:33	9:48:39											0:00:00	0:00:06
271	3	9:48:36	3			9:48:36	9:48:39									0:00:00	0:00:03
272	3	9:48:39	9	9:48:39	9:48:48											0:00:00	0:00:09
273	30	9:49:09	3	9:49:09	9:49:12											0:00:00	0:00:03
274	30	9:49:39	3	9:49:39	9:49:42											0:00:00	0:00:03
275	15	9:49:54	3	9:49:54	9:49:57											0:00:00	0:00:03
276	30	9:50:24	3	9:50:24	9:50:27											0:00:00	0:00:03
277	15	9:50:39	3	9:50:39	9:50:42											0:00:00	0:00:03
278	30	9:51:09	3	9:51:09	9:51:12											0:00:00	0:00:03
279	3	9:51:12	3	9:51:12	9:51:15											0:00:00	0:00:03
280	9	9:51:21	3	9:51:21	9:51:24											0:00:00	0:00:03
281	3	9:51:24	3	9:51:24	9:51:27											0:00:00	0:00:03
282	9	9:51:33	9	9:51:33	9:51:42											0:00:00	0:00:09
283	30	9:52:03	3	9:52:03	9:52:06											0:00:00	0:00:03
284	3	9:52:06	9	9:52:06	9:52:15											0:00:00	0:00:09
285	9	9:52:15	3	9:52:15	9:52:18											0:00:00	0:00:03
286	9	9:52:24	6	9:52:24	9:52:30											0:00:00	0:00:06
287	9	9:52:33	9	9:52:33	9:52:42											0:00:00	0:00:09
288	30	9:53:03	9	9:53:03	9:53:12											0:00:00	0:00:09
289	3	9:53:06	3			9:53:06	9:53:09									0:00:00	0:00:03
290	30	9:53:36	3	9:53:36	9:53:39											0:00:00	0:00:03
291	9	9:53:45	3	9:53:45	9:53:48											0:00:00	0:00:03
292	3	9:53:48	6	9:53:48	9:53:54											0:00:00	0:00:06
293	9	9:53:57	3	9:53:57	9:54:00											0:00:00	0:00:03

#	Time (min)	Time (hr:min)	Time (min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
294	15	9:54:12	3	9:54:12	9:54:15											0:00:00	0:00:03
295	30	9:54:42	9	9:54:42	9:54:51											0:00:00	0:00:09
296	30	9:55:12	3	9:55:12	9:55:15											0:00:00	0:00:03
297	3	9:55:15	6	9:55:15	9:55:21											0:00:00	0:00:06
298	9	9:55:24	6	9:55:24	9:55:30											0:00:00	0:00:06
299	9	9:55:33	9	9:55:33	9:55:42											0:00:00	0:00:09
300	9	9:55:42	3	9:55:42	9:55:45											0:00:00	0:00:03
301	3	9:55:45	6	9:55:45	9:55:51											0:00:00	0:00:06
302	3	9:55:48	3			9:55:48	9:55:51									0:00:00	0:00:03
303	3	9:55:51	6	9:55:51	9:55:57											0:00:00	0:00:06
304	3	9:55:54	3			9:55:54	9:55:57									0:00:00	0:00:03
305	3	9:55:57	3	9:55:57	9:56:00											0:00:00	0:00:03
306	3	9:56:00	6	9:56:00	9:56:06											0:00:00	0:00:06
307	15	9:56:15	3	9:56:15	9:56:18											0:00:00	0:00:03
308	9	9:56:24	3	9:56:24	9:56:27											0:00:00	0:00:03
309	3	9:56:27	9	9:56:27	9:56:36											0:00:00	0:00:09
310	9	9:56:36	6	9:56:36	9:56:42											0:00:00	0:00:06
311	15	9:56:51	6	9:56:51	9:56:57											0:00:00	0:00:06
312	3	9:56:54	3			9:56:54	9:56:57									0:00:00	0:00:03
313	9	9:57:03	3	9:57:03	9:57:06											0:00:00	0:00:03
314	9	9:57:12	3	9:57:12	9:57:15											0:00:00	0:00:03
315	3	9:57:15	3	9:57:15	9:57:18											0:00:00	0:00:03
316	3	9:57:18	3	9:57:18	9:57:21											0:00:00	0:00:03
317	3	9:57:21	6	9:57:21	9:57:27											0:00:00	0:00:06
318	3	9:57:24	3			9:57:24	9:57:27									0:00:00	0:00:03
319	3	9:57:27	3	9:57:27	9:57:30											0:00:00	0:00:03
320	3	9:57:30	6	9:57:30	9:57:36											0:00:00	0:00:06
321	3	9:57:33	6			9:57:33	9:57:39									0:00:00	0:00:06
322	9	9:57:42	6	9:57:42	9:57:48											0:00:00	0:00:06
323	3	9:57:45	6			9:57:45	9:57:51									0:00:00	0:00:06
324	9	9:57:54	3	9:57:54	9:57:57											0:00:00	0:00:03
325	9	9:58:03	6	9:58:03	9:58:09											0:00:00	0:00:06
326	3	9:58:06	6			9:58:06	9:58:12									0:00:00	0:00:06
327	3	9:58:09	6	9:58:09	9:58:15											0:00:00	0:00:06
328	3	9:58:12	3			9:58:12	9:58:15									0:00:00	0:00:03
329	3	9:58:15	6	9:58:15	9:58:21											0:00:00	0:00:06
330	30	9:58:45	9	9:58:45	9:58:54											0:00:00	0:00:09
331	3	9:58:48	6			9:58:48	9:58:54									0:00:00	0:00:06
332	9	9:58:57	9	9:58:57	9:59:06											0:00:00	0:00:09
333	15	9:59:12	3	9:59:12	9:59:15											0:00:00	0:00:03
334	30	9:59:42	9	9:59:42	9:59:51											0:00:00	0:00:09
335	15	9:59:57	3	9:59:57	10:00:00											0:00:00	0:00:03
closed	3	10:00:00															

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
start		9:00					
1	3	9:00:03	30	9:00:03	9:00:33	0:00:00	0:00:30
2	3	9:00:06	30	9:00:33	9:01:03	0:00:27	0:00:57
3	10	9:00:16	90	9:01:03	9:02:33	0:00:47	0:02:17
4	3	9:00:19	60	9:02:33	9:03:33	0:02:14	0:03:14
5	10	9:00:29	30	9:03:33	9:04:03	0:03:04	0:03:34
6	10	9:00:39	60	9:04:03	9:05:03	0:03:24	0:04:24
7	5	9:00:44	60	9:05:03	9:06:03	0:04:19	0:05:19
8	5	9:00:49	90	9:06:03	9:07:33	0:05:14	0:06:44
9	10	9:00:59	30	9:07:33	9:08:03	0:06:34	0:07:04
10	10	9:01:09	60	9:08:03	9:09:03	0:06:54	0:07:54
11	3	9:01:12	60	9:09:03	9:10:03	0:07:51	0:08:51
12	1	9:01:13	60	9:10:03	9:11:03	0:08:50	0:09:50
13	3	9:01:16	60	9:11:03	9:12:03	0:09:47	0:10:47
14	3	9:01:19	60	9:12:03	9:13:03	0:10:44	0:11:44
15	5	9:01:24	60	9:13:03	9:14:03	0:11:39	0:12:39
16	1	9:01:25	30	9:14:03	9:14:33	0:12:38	0:13:08
17	1	9:01:26	60	9:14:33	9:15:33	0:13:07	0:14:07
18	1	9:01:27	60	9:15:33	9:16:33	0:14:06	0:15:06
19	10	9:01:37	30	9:16:33	9:17:03	0:14:56	0:15:26
20	10	9:01:47	90	9:17:03	9:18:33	0:15:16	0:16:46
21	3	9:01:50	30	9:18:33	9:19:03	0:16:43	0:17:13
22	1	9:01:51	60	9:19:03	9:20:03	0:17:12	0:18:12
23	10	9:02:01	30	9:20:03	9:20:33	0:18:02	0:18:32
24	5	9:02:06	90	9:20:33	9:22:03	0:18:27	0:19:57
25	10	9:02:16	90	9:22:03	9:23:33	0:19:47	0:21:17
26	1	9:02:17	90	9:23:33	9:25:03	0:21:16	0:22:46
27	1	9:02:18	60	9:25:03	9:26:03	0:22:45	0:23:45
28	1	9:02:19	60	9:26:03	9:27:03	0:23:44	0:24:44
29	1	9:02:20	60	9:27:03	9:28:03	0:24:43	0:25:43
30	1	9:02:21	30	9:28:03	9:28:33	0:25:42	0:26:12
31	1	9:02:22	60	9:28:33	9:29:33	0:26:11	0:27:11
32	3	9:02:25	30	9:29:33	9:30:03	0:27:08	0:27:38
33	3	9:02:28	60	9:30:03	9:31:03	0:27:35	0:28:35
34	3	9:02:31	60	9:31:03	9:32:03	0:28:32	0:29:32
35	1	9:02:32	90	9:32:03	9:33:33	0:29:31	0:31:01
36	1	9:02:33	90	9:33:33	9:35:03	0:31:00	0:32:30
37	1	9:02:34	30	9:35:03	9:35:33	0:32:29	0:32:59
38	10	9:02:44	30	9:35:33	9:36:03	0:32:49	0:33:19
39	10	9:02:54	60	9:36:03	9:37:03	0:33:09	0:34:09
40	3	9:02:57	90	9:37:03	9:38:33	0:34:06	0:35:36
41	1	9:02:58	90	9:38:33	9:40:03	0:35:35	0:37:05
42	5	9:03:03	30	9:40:03	9:40:33	0:37:00	0:37:30
43	1	9:03:04	30	9:40:33	9:41:03	0:37:29	0:37:59
44	1	9:03:05	60	9:41:03	9:42:03	0:37:58	0:38:58
45	1	9:03:06	30	9:42:03	9:42:33	0:38:57	0:39:27
46	1	9:03:07	30	9:42:33	9:43:03	0:39:26	0:39:56
47	10	9:03:17	30	9:43:03	9:43:33	0:39:46	0:40:16
48	5	9:03:22	60	9:43:33	9:44:33	0:40:11	0:41:11
49	3	9:03:25	30	9:44:33	9:45:03	0:41:08	0:41:38
50	1	9:03:26	30	9:45:03	9:45:33	0:41:37	0:42:07
51	1	9:03:27	90	9:45:33	9:47:03	0:42:06	0:43:36
52	1	9:03:28	90	9:47:03	9:48:33	0:43:35	0:45:05
53	10	9:03:38	60	9:48:33	9:49:33	0:44:55	0:45:55
54	1	9:03:39	30	9:49:33	9:50:03	0:45:54	0:46:24
55	5	9:03:44	30	9:50:03	9:50:33	0:46:19	0:46:49

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
56	1	9:03:45	30	9:50:33	9:51:03	0:46:48	0:47:18
57	1	9:03:46	60	9:51:03	9:52:03	0:47:17	0:48:17
58	1	9:03:47	60	9:52:03	9:53:03	0:48:16	0:49:16
59	1	9:03:48	30	9:53:03	9:53:33	0:49:15	0:49:45
60	5	9:03:53	60	9:53:33	9:54:33	0:49:40	0:50:40
61	1	9:03:54	30	9:54:33	9:55:03	0:50:39	0:51:09
62	1	9:03:55	30	9:55:03	9:55:33	0:51:08	0:51:38
63	3	9:03:58	60	9:55:33	9:56:33	0:51:35	0:52:35
64	1	9:03:59	30	9:56:33	9:57:03	0:52:34	0:53:04
65	1	9:04:00	30	9:57:03	9:57:33	0:53:03	0:53:33
66	10	9:04:10	60	9:57:33	9:58:33	0:53:23	0:54:23
67	3	9:04:13	30	9:58:33	9:59:03	0:54:20	0:54:50
68	3	9:04:16	30	9:59:03	9:59:33	0:54:47	0:55:17
69	10	9:04:26	30	9:59:33	10:00:03	0:55:07	0:55:37
70	1	9:04:27	60	10:00:03	10:01:03	0:55:36	0:56:36
71	1	9:04:28	90	10:01:03	10:02:33	0:56:35	0:58:05
72	10	9:04:38	90	10:02:33	10:04:03	0:57:55	0:59:25
73	3	9:04:41	30	10:04:03	10:04:33	0:59:22	0:59:52
74	1	9:04:42	90	10:04:33	10:06:03	0:59:51	1:01:21
75	3	9:04:45	30	10:06:03	10:06:33	1:01:18	1:01:48
76	1	9:04:46	30	10:06:33	10:07:03	1:01:47	1:02:17
77	10	9:04:56	60	10:07:03	10:08:03	1:02:07	1:03:07
78	3	9:04:59	90	10:08:03	10:09:33	1:03:04	1:04:34
79	1	9:05:00	30	10:09:33	10:10:03	1:04:33	1:05:03
80	5	9:05:05	30	10:10:03	10:10:33	1:04:58	1:05:28
81	3	9:05:08	30	10:10:33	10:11:03	1:05:25	1:05:55
82	3	9:05:11	60	10:11:03	10:12:03	1:05:52	1:06:52
83	1	9:05:12	60	10:12:03	10:13:03	1:06:51	1:07:51
84	10	9:05:22	60	10:13:03	10:14:03	1:07:41	1:08:41
85	1	9:05:23	30	10:14:03	10:14:33	1:08:40	1:09:10
86	1	9:05:24	30	10:14:33	10:15:03	1:09:09	1:09:39
87	3	9:05:27	60	10:15:03	10:16:03	1:09:36	1:10:36
88	1	9:05:28	90	10:16:03	10:17:33	1:10:35	1:12:05
89	3	9:05:31	30	10:17:33	10:18:03	1:12:02	1:12:32
90	1	9:05:32	30	10:18:03	10:18:33	1:12:31	1:13:01
91	1	9:05:33	90	10:18:33	10:20:03	1:13:00	1:14:30
92	10	9:05:43	30	10:20:03	10:20:33	1:14:20	1:14:50
93	3	9:05:46	30	10:20:33	10:21:03	1:14:47	1:15:17
94	1	9:05:47	30	10:21:03	10:21:33	1:15:16	1:15:46
95	3	9:05:50	30	10:21:33	10:22:03	1:15:43	1:16:13
96	1	9:05:51	60	10:22:03	10:23:03	1:16:12	1:17:12
97	3	9:05:54	90	10:23:03	10:24:33	1:17:09	1:18:39
98	1	9:05:55	30	10:24:33	10:25:03	1:18:38	1:19:08
99	1	9:05:56	30	10:25:03	10:25:33	1:19:07	1:19:37
100	1	9:05:57	60	10:25:33	10:26:33	1:19:36	1:20:36
101	3	9:06:00	90	10:26:33	10:28:03	1:20:33	1:22:03
102	10	9:06:10	30	10:28:03	10:28:33	1:21:53	1:22:23
103	10	9:06:20	60	10:28:33	10:29:33	1:22:13	1:23:13
104	3	9:06:23	90	10:29:33	10:31:03	1:23:10	1:24:40
105	1	9:06:24	60	10:31:03	10:32:03	1:24:39	1:25:39
106	3	9:06:27	60	10:32:03	10:33:03	1:25:36	1:26:36
107	1	9:06:28	30	10:33:03	10:33:33	1:26:35	1:27:05
108	3	9:06:31	30	10:33:33	10:34:03	1:27:02	1:27:32
109	3	9:06:34	30	10:34:03	10:34:33	1:27:29	1:27:59
110	1	9:06:35	90	10:34:33	10:36:03	1:27:58	1:29:28
111	3	9:06:38	30	10:36:03	10:36:33	1:29:25	1:29:55
112	1	9:06:39	90	10:36:33	10:38:03	1:29:54	1:31:24

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
113	3	9:06:42	60	10:38:03	10:39:03	1:31:21	1:32:21
114	1	9:06:43	60	10:39:03	10:40:03	1:32:20	1:33:20
115	5	9:06:48	30	10:40:03	10:40:33	1:33:15	1:33:45
116	1	9:06:49	30	10:40:33	10:41:03	1:33:44	1:34:14
117	10	9:06:59	30	10:41:03	10:41:33	1:34:04	1:34:34
118	1	9:07:00	60	10:41:33	10:42:33	1:34:33	1:35:33
119	10	9:07:10	30	10:42:33	10:43:03	1:35:23	1:35:53
120	3	9:07:13	30	10:43:03	10:43:33	1:35:50	1:36:20
121	10	9:07:23	60	10:43:33	10:44:33	1:36:10	1:37:10
122	1	9:07:24	60	10:44:33	10:45:33	1:37:09	1:38:09
123	10	9:07:34	60	10:45:33	10:46:33	1:37:59	1:38:59
124	1	9:07:35	60	10:46:33	10:47:33	1:38:58	1:39:58
125	3	9:07:38	60	10:47:33	10:48:33	1:39:55	1:40:55
126	3	9:07:41	30	10:48:33	10:49:03	1:40:52	1:41:22
127	1	9:07:42	30	10:49:03	10:49:33	1:41:21	1:41:51
128	1	9:07:43	30	10:49:33	10:50:03	1:41:50	1:42:20
129	10	9:07:53	60	10:50:03	10:51:03	1:42:10	1:43:10
130	1	9:07:54	30	10:51:03	10:51:33	1:43:09	1:43:39
131	1	9:07:55	30	10:51:33	10:52:03	1:43:38	1:44:08
132	1	9:07:56	30	10:52:03	10:52:33	1:44:07	1:44:37
133	1	9:07:57	60	10:52:33	10:53:33	1:44:36	1:45:36
134	1	9:07:58	90	10:53:33	10:55:03	1:45:35	1:47:05
135	5	9:08:03	30	10:55:03	10:55:33	1:47:00	1:47:30
136	10	9:08:13	60	10:55:33	10:56:33	1:47:20	1:48:20
137	3	9:08:16	30	10:56:33	10:57:03	1:48:17	1:48:47
138	5	9:08:21	30	10:57:03	10:57:33	1:48:42	1:49:12
139	10	9:08:31	30	10:57:33	10:58:03	1:49:02	1:49:32
140	3	9:08:34	60	10:58:03	10:59:03	1:49:29	1:50:29
141	1	9:08:35	30	10:59:03	10:59:33	1:50:28	1:50:58
142	10	9:08:45	60	10:59:33	11:00:33	1:50:48	1:51:48
143	1	9:08:46	60	11:00:33	11:01:33	1:51:47	1:52:47
144	1	9:08:47	30	11:01:33	11:02:03	1:52:46	1:53:16
145	1	9:08:48	90	11:02:03	11:03:33	1:53:15	1:54:45
146	10	9:08:58	90	11:03:33	11:05:03	1:54:35	1:56:05
147	10	9:09:08	30	11:05:03	11:05:33	1:55:55	1:56:25
148	1	9:09:09	60	11:05:33	11:06:33	1:56:24	1:57:24
149	3	9:09:12	30	11:06:33	11:07:03	1:57:21	1:57:51
150	10	9:09:22	30	11:07:03	11:07:33	1:57:41	1:58:11
151	10	9:09:32	60	11:07:33	11:08:33	1:58:01	1:59:01
152	10	9:09:42	90	11:08:33	11:10:03	1:58:51	2:00:21
153	1	9:09:43	90	11:10:03	11:11:33	2:00:20	2:01:50
154	1	9:09:44	90	11:11:33	11:13:03	2:01:49	2:03:19
155	1	9:09:45	60	11:13:03	11:14:03	2:03:18	2:04:18
156	1	9:09:46	30	11:14:03	11:14:33	2:04:17	2:04:47
157	3	9:09:49	90	11:14:33	11:16:03	2:04:44	2:06:14
158	1	9:09:50	30	11:16:03	11:16:33	2:06:13	2:06:43
159	1	9:09:51	60	11:16:33	11:17:33	2:06:42	2:07:42
160	10	9:10:01	30	11:17:33	11:18:03	2:07:32	2:08:02
161	3	9:10:04	30	11:18:03	11:18:33	2:07:59	2:08:29
162	1	9:10:05	60	11:18:33	11:19:33	2:08:28	2:09:28
163	5	9:10:10	60	11:19:33	11:20:33	2:09:23	2:10:23
164	1	9:10:11	90	11:20:33	11:22:03	2:10:22	2:11:52
165	1	9:10:12	90	11:22:03	11:23:33	2:11:51	2:13:21
166	1	9:10:13	90	11:23:33	11:25:03	2:13:20	2:14:50
167	1	9:10:14	30	11:25:03	11:25:33	2:14:49	2:15:19
168	1	9:10:15	90	11:25:33	11:27:03	2:15:18	2:16:48
169	10	9:10:25	30	11:27:03	11:27:33	2:16:38	2:17:08

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
170	3	9:10:28	30	11:27:33	11:28:03	2:17:05	2:17:35
171	1	9:10:29	30	11:28:03	11:28:33	2:17:34	2:18:04
172	10	9:10:39	60	11:28:33	11:29:33	2:17:54	2:18:54
173	10	9:10:49	30	11:29:33	11:30:03	2:18:44	2:19:14
174	10	9:10:59	30	11:30:03	11:30:33	2:19:04	2:19:34
175	10	9:11:09	30	11:30:33	11:31:03	2:19:24	2:19:54
176	1	9:11:10	30	11:31:03	11:31:33	2:19:53	2:20:23
177	3	9:11:13	30	11:31:33	11:32:03	2:20:20	2:20:50
178	1	9:11:14	30	11:32:03	11:32:33	2:20:49	2:21:19
179	10	9:11:24	60	11:32:33	11:33:33	2:21:09	2:22:09
180	1	9:11:25	30	11:33:33	11:34:03	2:22:08	2:22:38
181	1	9:11:26	30	11:34:03	11:34:33	2:22:37	2:23:07
182	10	9:11:36	60	11:34:33	11:35:33	2:22:57	2:23:57
183	3	9:11:39	30	11:35:33	11:36:03	2:23:54	2:24:24
184	3	9:11:42	30	11:36:03	11:36:33	2:24:21	2:24:51
185	1	9:11:43	60	11:36:33	11:37:33	2:24:50	2:25:50
186	3	9:11:46	30	11:37:33	11:38:03	2:25:47	2:26:17
187	10	9:11:56	30	11:38:03	11:38:33	2:26:07	2:26:37
188	1	9:11:57	30	11:38:33	11:39:03	2:26:36	2:27:06
189	1	9:11:58	30	11:39:03	11:39:33	2:27:05	2:27:35
190	3	9:12:01	60	11:39:33	11:40:33	2:27:32	2:28:32
191	1	9:12:02	30	11:40:33	11:41:03	2:28:31	2:29:01
192	3	9:12:05	60	11:41:03	11:42:03	2:28:58	2:29:58
193	3	9:12:08	90	11:42:03	11:43:33	2:29:55	2:31:25
194	10	9:12:18	30	11:43:33	11:44:03	2:31:15	2:31:45
195	1	9:12:19	30	11:44:03	11:44:33	2:31:44	2:32:14
196	5	9:12:24	60	11:44:33	11:45:33	2:32:09	2:33:09
197	1	9:12:25	30	11:45:33	11:46:03	2:33:08	2:33:38
198	5	9:12:30	90	11:46:03	11:47:33	2:33:33	2:35:03
199	5	9:12:35	60	11:47:33	11:48:33	2:34:58	2:35:58
200	1	9:12:36	30	11:48:33	11:49:03	2:35:57	2:36:27
201	1	9:12:37	60	11:49:03	11:50:03	2:36:26	2:37:26
202	1	9:12:38	60	11:50:03	11:51:03	2:37:25	2:38:25
203	3	9:12:41	30	11:51:03	11:51:33	2:38:22	2:38:52
204	3	9:12:44	60	11:51:33	11:52:33	2:38:49	2:39:49
205	1	9:12:45	30	11:52:33	11:53:03	2:39:48	2:40:18
206	1	9:12:46	30	11:53:03	11:53:33	2:40:17	2:40:47
207	1	9:12:47	60	11:53:33	11:54:33	2:40:46	2:41:46
208	10	9:12:57	30	11:54:33	11:55:03	2:41:36	2:42:06
209	1	9:12:58	60	11:55:03	11:56:03	2:42:05	2:43:05
210	3	9:13:01	90	11:56:03	11:57:33	2:43:02	2:44:32
211	10	9:13:11	30	11:57:33	11:58:03	2:44:22	2:44:52
212	3	9:13:14	90	11:58:03	11:59:33	2:44:49	2:46:19
213	1	9:13:15	60	11:59:33	12:00:33	2:46:18	2:47:18
214	10	9:13:25	30	12:00:33	12:01:03	2:47:08	2:47:38
215	1	9:13:26	90	12:01:03	12:02:33	2:47:37	2:49:07
216	1	9:13:27	90	12:02:33	12:04:03	2:49:06	2:50:36
217	1	9:13:28	30	12:04:03	12:04:33	2:50:35	2:51:05
218	10	9:13:38	30	12:04:33	12:05:03	2:50:55	2:51:25
219	1	9:13:39	30	12:05:03	12:05:33	2:51:24	2:51:54
220	1	9:13:40	30	12:05:33	12:06:03	2:51:53	2:52:23
221	1	9:13:41	60	12:06:03	12:07:03	2:52:22	2:53:22
222	3	9:13:44	90	12:07:03	12:08:33	2:53:19	2:54:49
223	5	9:13:49	90	12:08:33	12:10:03	2:54:44	2:56:14
224	3	9:13:52	60	12:10:03	12:11:03	2:56:11	2:57:11
225	1	9:13:53	90	12:11:03	12:12:33	2:57:10	2:58:40
226	1	9:13:54	30	12:12:33	12:13:03	2:58:39	2:59:09

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
227	10	9:14:04	60	12:13:03	12:14:03	2:58:59	2:59:59
228	1	9:14:05	90	12:14:03	12:15:33	2:59:58	3:01:28
229	1	9:14:06	30	12:15:33	12:16:03	3:01:27	3:01:57
230	1	9:14:07	30	12:16:03	12:16:33	3:01:56	3:02:26
231	1	9:14:08	60	12:16:33	12:17:33	3:02:25	3:03:25
232	10	9:14:18	60	12:17:33	12:18:33	3:03:15	3:04:15
233	1	9:14:19	30	12:18:33	12:19:03	3:04:14	3:04:44
234	1	9:14:20	90	12:19:03	12:20:33	3:04:43	3:06:13
235	1	9:14:21	30	12:20:33	12:21:03	3:06:12	3:06:42
236	1	9:14:22	60	12:21:03	12:22:03	3:06:41	3:07:41
237	5	9:14:27	90	12:22:03	12:23:33	3:07:36	3:09:06
238	1	9:14:28	90	12:23:33	12:25:03	3:09:05	3:10:35
239	1	9:14:29	30	12:25:03	12:25:33	3:10:34	3:11:04
240	3	9:14:32	60	12:25:33	12:26:33	3:11:01	3:12:01
241	1	9:14:33	30	12:26:33	12:27:03	3:12:00	3:12:30
242	1	9:14:34	30	12:27:03	12:27:33	3:12:29	3:12:59
243	1	9:14:35	60	12:27:33	12:28:33	3:12:58	3:13:58
244	1	9:14:36	30	12:28:33	12:29:03	3:13:57	3:14:27
245	1	9:14:37	60	12:29:03	12:30:03	3:14:26	3:15:26
246	10	9:14:47	30	12:30:03	12:30:33	3:15:16	3:15:46
247	1	9:14:48	90	12:30:33	12:32:03	3:15:45	3:17:15
248	10	9:14:58	60	12:32:03	12:33:03	3:17:05	3:18:05
249	3	9:15:01	60	12:33:03	12:34:03	3:18:02	3:19:02
250	10	9:15:11	30	12:34:03	12:34:33	3:18:52	3:19:22
251	10	9:15:21	60	12:34:33	12:35:33	3:19:12	3:20:12
252	10	9:15:31	30	12:35:33	12:36:03	3:20:02	3:20:32
253	1	9:15:32	60	12:36:03	12:37:03	3:20:31	3:21:31
254	10	9:15:42	30	12:37:03	12:37:33	3:21:21	3:21:51
255	3	9:15:45	30	12:37:33	12:38:03	3:21:48	3:22:18
256	1	9:15:46	30	12:38:03	12:38:33	3:22:17	3:22:47
257	1	9:15:47	90	12:38:33	12:40:03	3:22:46	3:24:16
258	10	9:15:57	30	12:40:03	12:40:33	3:24:06	3:24:36
259	10	9:16:07	30	12:40:33	12:41:03	3:24:26	3:24:56
260	1	9:16:08	90	12:41:03	12:42:33	3:24:55	3:26:25
261	1	9:16:09	30	12:42:33	12:43:03	3:26:24	3:26:54
262	1	9:16:10	30	12:43:03	12:43:33	3:26:53	3:27:23
263	1	9:16:11	60	12:43:33	12:44:33	3:27:22	3:28:22
264	5	9:16:16	30	12:44:33	12:45:03	3:28:17	3:28:47
265	3	9:16:19	60	12:45:03	12:46:03	3:28:44	3:29:44
266	3	9:16:22	60	12:46:03	12:47:03	3:29:41	3:30:41
267	10	9:16:32	30	12:47:03	12:47:33	3:30:31	3:31:01
268	1	9:16:33	30	12:47:33	12:48:03	3:31:00	3:31:30
269	5	9:16:38	60	12:48:03	12:49:03	3:31:25	3:32:25
270	1	9:16:39	60	12:49:03	12:50:03	3:32:24	3:33:24
271	1	9:16:40	30	12:50:03	12:50:33	3:33:23	3:33:53
272	1	9:16:41	60	12:50:33	12:51:33	3:33:52	3:34:52
273	1	9:16:42	90	12:51:33	12:53:03	3:34:51	3:36:21
274	1	9:16:43	30	12:53:03	12:53:33	3:36:20	3:36:50
275	10	9:16:53	30	12:53:33	12:54:03	3:36:40	3:37:10
276	10	9:17:03	90	12:54:03	12:55:33	3:37:00	3:38:30
277	1	9:17:04	30	12:55:33	12:56:03	3:38:29	3:38:59
278	1	9:17:05	30	12:56:03	12:56:33	3:38:58	3:39:28
279	1	9:17:06	30	12:56:33	12:57:03	3:39:27	3:39:57
280	10	9:17:16	60	12:57:03	12:58:03	3:39:47	3:40:47
281	1	9:17:17	90	12:58:03	12:59:33	3:40:46	3:42:16
282	3	9:17:20	60	12:59:33	13:00:33	3:42:13	3:43:13
283	5	9:17:25	30	13:00:33	13:01:03	3:43:08	3:43:38

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
284	3	9:17:28	30	13:01:03	13:01:33	3:43:35	3:44:05
285	10	9:17:38	30	13:01:33	13:02:03	3:43:55	3:44:25
286	1	9:17:39	90	13:02:03	13:03:33	3:44:24	3:45:54
287	3	9:17:42	60	13:03:33	13:04:33	3:45:51	3:46:51
288	1	9:17:43	30	13:04:33	13:05:03	3:46:50	3:47:20
289	5	9:17:48	90	13:05:03	13:06:33	3:47:15	3:48:45
290	3	9:17:51	90	13:06:33	13:08:03	3:48:42	3:50:12
291	3	9:17:54	30	13:08:03	13:08:33	3:50:09	3:50:39
292	10	9:18:04	30	13:08:33	13:09:03	3:50:29	3:50:59
293	1	9:18:05	30	13:09:03	13:09:33	3:50:58	3:51:28
294	3	9:18:08	60	13:09:33	13:10:33	3:51:25	3:52:25
295	10	9:18:18	90	13:10:33	13:12:03	3:52:15	3:53:45
296	3	9:18:21	30	13:12:03	13:12:33	3:53:42	3:54:12
297	1	9:18:22	60	13:12:33	13:13:33	3:54:11	3:55:11
298	1	9:18:23	60	13:13:33	13:14:33	3:55:10	3:56:10
299	1	9:18:24	60	13:14:33	13:15:33	3:56:09	3:57:09
300	3	9:18:27	90	13:15:33	13:17:03	3:57:06	3:58:36
301	3	9:18:30	60	13:17:03	13:18:03	3:58:33	3:59:33
302	1	9:18:31	30	13:18:03	13:18:33	3:59:32	4:00:02
303	10	9:18:41	30	13:18:33	13:19:03	3:59:52	4:00:22
304	5	9:18:46	30	13:19:03	13:19:33	4:00:17	4:00:47
305	1	9:18:47	30	13:19:33	13:20:03	4:00:46	4:01:16
306	1	9:18:48	60	13:20:03	13:21:03	4:01:15	4:02:15
307	1	9:18:49	30	13:21:03	13:21:33	4:02:14	4:02:44
308	1	9:18:50	60	13:21:33	13:22:33	4:02:43	4:03:43
309	3	9:18:53	30	13:22:33	13:23:03	4:03:40	4:04:10
310	1	9:18:54	30	13:23:03	13:23:33	4:04:09	4:04:39
311	10	9:19:04	60	13:23:33	13:24:33	4:04:29	4:05:29
312	10	9:19:14	60	13:24:33	13:25:33	4:05:19	4:06:19
313	5	9:19:19	90	13:25:33	13:27:03	4:06:14	4:07:44
314	1	9:19:20	90	13:27:03	13:28:33	4:07:43	4:09:13
315	3	9:19:23	30	13:28:33	13:29:03	4:09:10	4:09:40
316	3	9:19:26	30	13:29:03	13:29:33	4:09:37	4:10:07
317	1	9:19:27	30	13:29:33	13:30:03	4:10:06	4:10:36
318	1	9:19:28	60	13:30:03	13:31:03	4:10:35	4:11:35
319	5	9:19:33	30	13:31:03	13:31:33	4:11:30	4:12:00
320	5	9:19:38	60	13:31:33	13:32:33	4:11:55	4:12:55
321	1	9:19:39	30	13:32:33	13:33:03	4:12:54	4:13:24
322	1	9:19:40	30	13:33:03	13:33:33	4:13:23	4:13:53
323	10	9:19:50	30	13:33:33	13:34:03	4:13:43	4:14:13
324	1	9:19:51	60	13:34:03	13:35:03	4:14:12	4:15:12
325	3	9:19:54	30	13:35:03	13:35:33	4:15:09	4:15:39
326	3	9:19:57	60	13:35:33	13:36:33	4:15:36	4:16:36
327	3	9:20:00	30	13:36:33	13:37:03	4:16:33	4:17:03
328	3	9:20:03	90	13:37:03	13:38:33	4:17:00	4:18:30
329	5	9:20:08	90	13:38:33	13:40:03	4:18:25	4:19:55
330	10	9:20:18	30	13:40:03	13:40:33	4:19:45	4:20:15
331	1	9:20:19	30	13:40:33	13:41:03	4:20:14	4:20:44
332	3	9:20:22	60	13:41:03	13:42:03	4:20:41	4:21:41
333	10	9:20:32	30	13:42:03	13:42:33	4:21:31	4:22:01
334	1	9:20:33	60	13:42:33	13:43:33	4:22:00	4:23:00
335	5	9:20:38	60	13:43:33	13:44:33	4:22:55	4:23:55
336	1	9:20:39	60	13:44:33	13:45:33	4:23:54	4:24:54
337	1	9:20:40	30	13:45:33	13:46:03	4:24:53	4:25:23
338	1	9:20:41	90	13:46:03	13:47:33	4:25:22	4:26:52
339	10	9:20:51	90	13:47:33	13:49:03	4:26:42	4:28:12
340	3	9:20:54	60	13:49:03	13:50:03	4:28:09	4:29:09

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
341	3	9:20:57	60	13:50:03	13:51:03	4:29:06	4:30:06
342	10	9:21:07	60	13:51:03	13:52:03	4:29:56	4:30:56
343	3	9:21:10	60	13:52:03	13:53:03	4:30:53	4:31:53
344	1	9:21:11	30	13:53:03	13:53:33	4:31:52	4:32:22
345	3	9:21:14	60	13:53:33	13:54:33	4:32:19	4:33:19
346	10	9:21:24	30	13:54:33	13:55:03	4:33:09	4:33:39
347	3	9:21:27	30	13:55:03	13:55:33	4:33:36	4:34:06
348	1	9:21:28	60	13:55:33	13:56:33	4:34:05	4:35:05
349	3	9:21:31	30	13:56:33	13:57:03	4:35:02	4:35:32
350	1	9:21:32	60	13:57:03	13:58:03	4:35:31	4:36:31
351	3	9:21:35	30	13:58:03	13:58:33	4:36:28	4:36:58
352	5	9:21:40	60	13:58:33	13:59:33	4:36:53	4:37:53
353	3	9:21:43	60	13:59:33	14:00:33	4:37:50	4:38:50
354	1	9:21:44	30	14:00:33	14:01:03	4:38:49	4:39:19
355	5	9:21:49	90	14:01:03	14:02:33	4:39:14	4:40:44
356	1	9:21:50	90	14:02:33	14:04:03	4:40:43	4:42:13
357	10	9:22:00	90	14:04:03	14:05:33	4:42:03	4:43:33
358	10	9:22:10	60	14:05:33	14:06:33	4:43:23	4:44:23
359	10	9:22:20	60	14:06:33	14:07:33	4:44:13	4:45:13
360	3	9:22:23	30	14:07:33	14:08:03	4:45:10	4:45:40
361	1	9:22:24	30	14:08:03	14:08:33	4:45:39	4:46:09
362	1	9:22:25	60	14:08:33	14:09:33	4:46:08	4:47:08
363	1	9:22:26	30	14:09:33	14:10:03	4:47:07	4:47:37
364	10	9:22:36	30	14:10:03	14:10:33	4:47:27	4:47:57
365	1	9:22:37	30	14:10:33	14:11:03	4:47:56	4:48:26
366	3	9:22:40	30	14:11:03	14:11:33	4:48:23	4:48:53
367	5	9:22:45	60	14:11:33	14:12:33	4:48:48	4:49:48
368	10	9:22:55	60	14:12:33	14:13:33	4:49:38	4:50:38
369	5	9:23:00	30	14:13:33	14:14:03	4:50:33	4:51:03
370	3	9:23:03	30	14:14:03	14:14:33	4:51:00	4:51:30
371	1	9:23:04	30	14:14:33	14:15:03	4:51:29	4:51:59
372	3	9:23:07	60	14:15:03	14:16:03	4:51:56	4:52:56
373	1	9:23:08	60	14:16:03	14:17:03	4:52:55	4:53:55
374	3	9:23:11	90	14:17:03	14:18:33	4:53:52	4:55:22
375	1	9:23:12	90	14:18:33	14:20:03	4:55:21	4:56:51
376	3	9:23:15	30	14:20:03	14:20:33	4:56:48	4:57:18
377	1	9:23:16	30	14:20:33	14:21:03	4:57:17	4:57:47
378	3	9:23:19	30	14:21:03	14:21:33	4:57:44	4:58:14
379	3	9:23:22	90	14:21:33	14:23:03	4:58:11	4:59:41
380	5	9:23:27	90	14:23:03	14:24:33	4:59:36	5:01:06
381	3	9:23:30	30	14:24:33	14:25:03	5:01:03	5:01:33
382	1	9:23:31	30	14:25:03	14:25:33	5:01:32	5:02:02
383	3	9:23:34	60	14:25:33	14:26:33	5:01:59	5:02:59
384	1	9:23:35	30	14:26:33	14:27:03	5:02:58	5:03:28
385	10	9:23:45	60	14:27:03	14:28:03	5:03:18	5:04:18
386	10	9:23:55	30	14:28:03	14:28:33	5:04:08	5:04:38
387	5	9:24:00	90	14:28:33	14:30:03	5:04:33	5:06:03
388	10	9:24:10	30	14:30:03	14:30:33	5:05:53	5:06:23
389	3	9:24:13	90	14:30:33	14:32:03	5:06:20	5:07:50
390	1	9:24:14	30	14:32:03	14:32:33	5:07:49	5:08:19
391	3	9:24:17	30	14:32:33	14:33:03	5:08:16	5:08:46
392	1	9:24:18	30	14:33:03	14:33:33	5:08:45	5:09:15
393	10	9:24:28	60	14:33:33	14:34:33	5:09:05	5:10:05
394	10	9:24:38	60	14:34:33	14:35:33	5:09:55	5:10:55
395	1	9:24:39	60	14:35:33	14:36:33	5:10:54	5:11:54
396	3	9:24:42	30	14:36:33	14:37:03	5:11:51	5:12:21
397	5	9:24:47	60	14:37:03	14:38:03	5:12:16	5:13:16

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
398	1	9:24:48	30	14:38:03	14:38:33	5:13:15	5:13:45
399	10	9:24:58	60	14:38:33	14:39:33	5:13:35	5:14:35
400	10	9:25:08	90	14:39:33	14:41:03	5:14:25	5:15:55
401	1	9:25:09	60	14:41:03	14:42:03	5:15:54	5:16:54
402	5	9:25:14	60	14:42:03	14:43:03	5:16:49	5:17:49
403	10	9:25:24	60	14:43:03	14:44:03	5:17:39	5:18:39
404	10	9:25:34	30	14:44:03	14:44:33	5:18:29	5:18:59
405	1	9:25:35	90	14:44:33	14:46:03	5:18:58	5:20:28
406	10	9:25:45	60	14:46:03	14:47:03	5:20:18	5:21:18
407	1	9:25:46	30	14:47:03	14:47:33	5:21:17	5:21:47
408	3	9:25:49	60	14:47:33	14:48:33	5:21:44	5:22:44
409	3	9:25:52	60	14:48:33	14:49:33	5:22:41	5:23:41
410	1	9:25:53	60	14:49:33	14:50:33	5:23:40	5:24:40
411	1	9:25:54	30	14:50:33	14:51:03	5:24:39	5:25:09
412	3	9:25:57	30	14:51:03	14:51:33	5:25:06	5:25:36
413	1	9:25:58	60	14:51:33	14:52:33	5:25:35	5:26:35
414	3	9:26:01	60	14:52:33	14:53:33	5:26:32	5:27:32
415	1	9:26:02	30	14:53:33	14:54:03	5:27:31	5:28:01
416	3	9:26:05	90	14:54:03	14:55:33	5:27:58	5:29:28
417	10	9:26:15	30	14:55:33	14:56:03	5:29:18	5:29:48
418	3	9:26:18	30	14:56:03	14:56:33	5:29:45	5:30:15
419	1	9:26:19	60	14:56:33	14:57:33	5:30:14	5:31:14
420	1	9:26:20	90	14:57:33	14:59:03	5:31:13	5:32:43
421	10	9:26:30	60	14:59:03	15:00:03	5:32:33	5:33:33
422	3	9:26:33	30	15:00:03	15:00:33	5:33:30	5:34:00
423	1	9:26:34	30	15:00:33	15:01:03	5:33:59	5:34:29
424	1	9:26:35	30	15:01:03	15:01:33	5:34:28	5:34:58
425	1	9:26:36	30	15:01:33	15:02:03	5:34:57	5:35:27
426	1	9:26:37	30	15:02:03	15:02:33	5:35:26	5:35:56
427	1	9:26:38	60	15:02:33	15:03:33	5:35:55	5:36:55
428	1	9:26:39	90	15:03:33	15:05:03	5:36:54	5:38:24
429	3	9:26:42	60	15:05:03	15:06:03	5:38:21	5:39:21
430	1	9:26:43	90	15:06:03	15:07:33	5:39:20	5:40:50
431	1	9:26:44	60	15:07:33	15:08:33	5:40:49	5:41:49
432	1	9:26:45	60	15:08:33	15:09:33	5:41:48	5:42:48
433	3	9:26:48	60	15:09:33	15:10:33	5:42:45	5:43:45
434	1	9:26:49	30	15:10:33	15:11:03	5:43:44	5:44:14
435	10	9:26:59	90	15:11:03	15:12:33	5:44:04	5:45:34
436	3	9:27:02	60	15:12:33	15:13:33	5:45:31	5:46:31
437	1	9:27:03	60	15:13:33	15:14:33	5:46:30	5:47:30
438	3	9:27:06	60	15:14:33	15:15:33	5:47:27	5:48:27
439	10	9:27:16	30	15:15:33	15:16:03	5:48:17	5:48:47
440	10	9:27:26	90	15:16:03	15:17:33	5:48:37	5:50:07
441	3	9:27:29	30	15:17:33	15:18:03	5:50:04	5:50:34
442	1	9:27:30	60	15:18:03	15:19:03	5:50:33	5:51:33
443	1	9:27:31	60	15:19:03	15:20:03	5:51:32	5:52:32
444	10	9:27:41	30	15:20:03	15:20:33	5:52:22	5:52:52
445	5	9:27:46	60	15:20:33	15:21:33	5:52:47	5:53:47
446	1	9:27:47	30	15:21:33	15:22:03	5:53:46	5:54:16
447	10	9:27:57	60	15:22:03	15:23:03	5:54:06	5:55:06
448	3	9:28:00	60	15:23:03	15:24:03	5:55:03	5:56:03
449	3	9:28:03	30	15:24:03	15:24:33	5:56:00	5:56:30
450	1	9:28:04	30	15:24:33	15:25:03	5:56:29	5:56:59
451	3	9:28:07	30	15:25:03	15:25:33	5:56:56	5:57:26
452	1	9:28:08	60	15:25:33	15:26:33	5:57:25	5:58:25
453	10	9:28:18	90	15:26:33	15:28:03	5:58:15	5:59:45
454	1	9:28:19	30	15:28:03	15:28:33	5:59:44	6:00:14

Table (A-4)
Queueing Simulation under condition IV

Cust. #	Interarrival Time (sec)	Arrival Time (hr:min)	Service Time (sec)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
455	3	9:28:22	60	15:28:33	15:29:33	6:00:11	6:01:11
456	3	9:28:25	30	15:29:33	15:30:03	6:01:08	6:01:38
457	3	9:28:28	30	15:30:03	15:30:33	6:01:35	6:02:05
458	1	9:28:29	30	15:30:33	15:31:03	6:02:04	6:02:34
459	1	9:28:30	30	15:31:03	15:31:33	6:02:33	6:03:03
460	1	9:28:31	90	15:31:33	15:33:03	6:03:02	6:04:32
461	1	9:28:32	60	15:33:03	15:34:03	6:04:31	6:05:31
462	10	9:28:42	30	15:34:03	15:34:33	6:05:21	6:05:51
463	1	9:28:43	60	15:34:33	15:35:33	6:05:50	6:06:50
464	10	9:28:53	30	15:35:33	15:36:03	6:06:40	6:07:10
465	3	9:28:56	30	15:36:03	15:36:33	6:07:07	6:07:37
466	1	9:28:57	60	15:36:33	15:37:33	6:07:36	6:08:36
467	5	9:29:02	60	15:37:33	15:38:33	6:08:31	6:09:31
468	1	9:29:03	60	15:38:33	15:39:33	6:09:30	6:10:30
469	3	9:29:06	30	15:39:33	15:40:03	6:10:27	6:10:57
470	3	9:29:09	30	15:40:03	15:40:33	6:10:54	6:11:24
471	1	9:29:10	30	15:40:33	15:41:03	6:11:23	6:11:53
472	1	9:29:11	30	15:41:03	15:41:33	6:11:52	6:12:22
473	1	9:29:12	30	15:41:33	15:42:03	6:12:21	6:12:51
474	3	9:29:15	30	15:42:03	15:42:33	6:12:48	6:13:18
475	10	9:29:25	30	15:42:33	15:43:03	6:13:08	6:13:38
476	1	9:29:26	60	15:43:03	15:44:03	6:13:37	6:14:37
477	3	9:29:29	60	15:44:03	15:45:03	6:14:34	6:15:34
478	1	9:29:30	60	15:45:03	15:46:03	6:15:33	6:16:33
479	10	9:29:40	60	15:46:03	15:47:03	6:16:23	6:17:23
480	3	9:29:43	30	15:47:03	15:47:33	6:17:20	6:17:50
481	10	9:29:53	90	15:47:33	15:49:03	6:17:40	6:19:10
482	5	9:29:58	90	15:49:03	15:50:33	6:19:05	6:20:35
483	1	9:29:59	30	15:50:33	15:51:03	6:20:34	6:21:04
484	3	9:30:02	30	15:51:03	15:51:33	6:21:01	6:21:31
485	1	9:30:03	90	15:51:33	15:53:03	6:21:30	6:23:00
486	5	9:30:08	30	15:53:03	15:53:33	6:22:55	6:23:25
487	3	9:30:11	90	15:53:33	15:55:03	6:23:22	6:24:52
488	10	9:30:21	90	15:55:03	15:56:33	6:24:42	6:26:12
489	3	9:30:24	90	15:56:33	15:58:03	6:26:09	6:27:39
490	1	9:30:25	90	15:58:03	15:59:33	6:27:38	6:29:08
491	1	9:30:26	30	15:59:33	16:00:03	6:29:07	6:29:37
492	1	9:30:27	30	16:00:03	16:00:33	6:29:36	6:30:06
493	1	9:30:28	90	16:00:33	16:02:03	6:30:05	6:31:35
494	1	9:30:29	30	16:02:03	16:02:33	6:31:34	6:32:04
495	3	9:30:32	60	16:02:33	16:03:33	6:32:01	6:33:01
496	1	9:30:33	60	16:03:33	16:04:33	6:33:00	6:34:00
497	5	9:30:38	90	16:04:33	16:06:03	6:33:55	6:35:25
498	10	9:30:48	30	16:06:03	16:06:33	6:35:15	6:35:45
499	1	9:30:49	30	16:06:33	16:07:03	6:35:44	6:36:14
500	5	9:30:54	30	16:07:03	16:07:33	6:36:09	6:36:39
501	1	9:30:55	60	16:07:33	16:08:33	6:36:38	6:37:38
502	1	9:30:56	60	16:08:33	16:09:33	6:37:37	6:38:37
503	3	9:30:59	60	16:09:33	16:10:33	6:38:34	6:39:34
504	10	9:31:09	60	16:10:33	16:11:33	6:39:24	6:40:24
505	10	9:31:19	30	16:11:33	16:12:03	6:40:14	6:40:44
506	1	9:31:20	30	16:12:03	16:12:33	6:40:43	6:41:13
507	3	9:31:23	30	16:12:33	16:13:03	6:41:10	6:41:40
508	5	9:31:28	30	16:13:03	16:13:33	6:41:35	6:42:05
509	3	9:31:31	30	16:13:33	16:14:03	6:42:02	6:42:32
510	10	9:31:41	30	16:14:03	16:14:33	6:42:22	6:42:52
511	1	9:31:42	30	16:14:33	16:15:03	6:42:51	6:43:21

#	Time (min)	Time (hr:min)	Time (min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
start		9:00														0:00:00	0:00:30
1	30	9:00:30	30	9:00:30	9:01:00											0:00:00	0:01:00
2	30	9:01:00	60	9:01:00	9:02:00											0:00:00	0:01:00
3	15	9:01:15	60			9:01:15	9:02:15									0:00:00	0:00:30
4	9	9:01:24	30					9:01:24	9:01:54							0:00:00	0:00:30
5	15	9:01:39	30							9:01:39	9:02:09					0:00:00	0:01:00
6	9	9:01:48	60									9:01:48	9:02:48			0:00:00	0:00:30
7	15	9:02:03	30	9:02:03	9:02:33											0:00:00	0:01:00
8	15	9:02:18	60			9:02:18	9:03:18									0:00:00	0:00:30
9	15	9:02:33	30	9:02:33	9:03:03											0:00:00	0:01:00
10	30	9:03:03	60	9:03:03	9:04:03											0:00:00	0:00:30
11	3	9:03:06	30					9:03:06	9:03:36							0:00:00	0:01:30
12	3	9:03:09	90							9:03:09	9:04:39					0:00:00	0:01:00
13	3	9:03:12	60									9:03:12	9:04:12			0:00:00	0:00:30
14	9	9:03:21	30			9:03:21	9:03:51									0:00:00	0:00:30
15	30	9:03:51	30			9:03:51	9:04:21									0:00:00	0:01:00
16	9	9:04:00	60					9:04:00	9:05:00							0:00:00	0:01:00
17	3	9:04:03	60	9:04:03	9:05:03									9:04:06	9:05:36	0:00:00	0:01:30
18	3	9:04:06	90									9:04:12	9:04:42			0:00:03	0:00:33
19	3	9:04:09	30													0:00:09	0:00:39
20	3	9:04:12	30			9:04:21	9:04:51									0:00:12	0:00:42
21	15	9:04:27	30							9:04:39	9:05:09					0:00:00	0:00:30
22	30	9:04:57	30			9:04:57	9:05:27									0:00:00	0:00:30
23	30	9:05:27	30	9:05:27	9:05:57											0:00:00	0:01:00
24	9	9:05:36	60			9:05:36	9:06:36									0:00:00	0:01:30
25	3	9:05:39	90					9:05:39	9:07:09							0:00:00	0:00:30
26	30	9:06:09	30	9:06:09	9:06:39											0:00:00	0:01:30
27	30	9:06:39	90	9:06:39	9:08:09											0:00:00	0:00:30
28	15	9:06:54	30			9:06:54	9:07:24									0:00:00	0:01:00
29	30	9:07:24	60			9:07:24	9:08:24									0:00:00	0:00:30
30	15	9:07:39	30					9:07:39	9:08:09							0:00:00	0:01:00
31	3	9:07:42	60							9:07:42	9:08:42					0:00:00	0:01:00
32	3	9:07:45	60									9:07:45	9:08:45			0:00:00	0:00:30
33	9	9:07:54	30											9:07:54	9:08:24	0:00:00	0:00:30
34	15	9:08:09	30	9:08:09	9:08:39											0:00:00	0:01:00
35	30	9:08:39	60	9:08:39	9:09:39											0:00:00	0:00:30
36	3	9:08:42	30			9:08:42	9:09:12									0:00:00	0:01:00
37	9	9:08:51	60					9:08:51	9:09:51							0:00:00	0:00:30
38	3	9:08:54	30							9:08:54	9:09:24					0:00:00	0:01:30
39	3	9:08:57	90									9:08:57	9:10:27			0:00:00	0:00:30
40	3	9:09:00	30											9:09:00	9:09:30	0:00:00	0:01:30
41	30	9:09:30	90			9:09:30	9:11:00									0:00:00	0:00:30
42	30	9:10:00	30	9:10:00	9:10:30											0:00:00	0:00:30
43	30	9:10:30	30	9:10:30	9:11:00											0:00:00	0:00:30
44	9	9:10:39	30					9:10:39	9:11:09							0:00:00	0:00:30
45	9	9:10:48	30							9:10:48	9:11:18					0:00:00	0:01:30
46	30	9:11:18	90	9:11:18	9:12:48											0:00:00	0:00:30
47	3	9:11:21	30			9:11:21	9:11:51										

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
48	9	9:11:30	60					9:11:30	9:12:30							0:00:00	0:01:00
49	3	9:11:33	60							9:11:33	9:12:33					0:00:00	0:01:00
50	9	9:11:42	60									9:11:42	9:12:42			0:00:00	0:01:00
51	9	9:11:51	60			9:11:51	9:12:51									0:00:00	0:01:00
52	9	9:12:00	30											9:12:00	9:12:30	0:00:00	0:00:30
53	3	9:12:03	30					9:12:30	9:13:00							0:00:27	0:00:57
54	30	9:12:33	30							9:12:33	9:13:03					0:00:00	0:00:30
55	30	9:13:03	60	9:13:03	9:14:03											0:00:00	0:01:00
56	9	9:13:12	60			9:13:12	9:14:12									0:00:00	0:01:00
57	9	9:13:21	60					9:13:21	9:14:21							0:00:00	0:01:00
58	3	9:13:24	30							9:13:24	9:13:54					0:00:00	0:00:30
59	3	9:13:27	60									9:13:27	9:14:27			0:00:00	0:01:00
60	30	9:13:57	30							9:13:57	9:14:27					0:00:00	0:00:30
61	3	9:14:00	90											9:14:00	9:15:30	0:00:00	0:01:30
62	3	9:14:03	60	9:14:03	9:15:03											0:00:00	0:01:00
63	30	9:14:33	90			9:14:33	9:16:03									0:00:00	0:01:30
64	9	9:14:42	30					9:14:42	9:15:12							0:00:00	0:00:30
65	30	9:15:12	90	9:15:12	9:16:42											0:00:00	0:01:30
66	9	9:15:21	30					9:15:21	9:15:51							0:00:00	0:00:30
67	3	9:15:24	60							9:15:24	9:16:24					0:00:00	0:01:00
68	3	9:15:27	60									9:15:27	9:16:27			0:00:00	0:01:00
69	3	9:15:30	30											9:15:30	9:16:00	0:00:00	0:00:30
70	3	9:15:33	30					9:15:51	9:16:21							0:00:18	0:00:48
71	30	9:16:03	30			9:16:03	9:16:33									0:00:00	0:00:30
72	30	9:16:33	30			9:16:33	9:17:03									0:00:00	0:00:30
73	3	9:16:36	90					9:16:36	9:18:06							0:00:00	0:01:30
74	3	9:16:39	30							9:16:39	9:17:09					0:00:00	0:00:30
75	30	9:17:09	60	9:17:09	9:18:09											0:00:00	0:01:00
76	3	9:17:12	90			9:17:12	9:18:42									0:00:00	0:01:30
77	3	9:17:15	90							9:17:15	9:18:45					0:00:00	0:01:30
78	3	9:17:18	30									9:17:18	9:17:48			0:00:00	0:00:30
79	9	9:17:27	60											9:17:27	9:18:27	0:00:00	0:01:00
80	3	9:17:30	30									9:17:48	9:18:18			0:00:18	0:00:48
81	3	9:17:33	60					9:18:06	9:19:06							0:00:33	0:01:33
82	3	9:17:36	90	9:18:09	9:19:39											0:00:33	0:02:03
83	3	9:17:39	30									9:18:18	9:18:48			0:00:39	0:01:09
84	9	9:17:48	30											9:18:27	9:18:57	0:00:39	0:01:09
85	3	9:17:51	60			9:18:42	9:19:42									0:00:51	0:01:51
86	15	9:18:06	30							9:18:45	9:19:15					0:00:39	0:01:09
87	30	9:18:36	30									9:18:48	9:19:18			0:00:12	0:00:42
88	3	9:18:39	30											9:18:57	9:19:27	0:00:18	0:00:48
89	9	9:18:48	60					9:19:06	9:20:06							0:00:18	0:01:18
90	3	9:18:51	90							9:19:15	9:20:45					0:00:24	0:01:54
91	3	9:18:54	30									9:19:18	9:19:48			0:00:24	0:00:54
92	30	9:19:24	90											9:19:27	9:20:57	0:00:03	0:01:33
93	9	9:19:33	60	9:19:39	9:20:39											0:00:06	0:01:06
94	3	9:19:36	90			9:19:42	9:21:12									0:00:06	0:01:36
95	9	9:19:45	30									9:19:48	9:20:18			0:00:03	0:00:33
96	30	9:20:15	30					9:20:15	9:20:45							0:00:00	0:00:30

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Server #7 Start (hr:min)	Server #7 End (hr:min)
97	15	9:20:30	60									9:20:30	9:21:30			0:00:00	0:01:00
98	15	9:20:45	30	9:20:45	9:21:15											0:00:00	0:00:30
99	3	9:20:48	30					9:20:48	9:21:18							0:00:00	0:00:30
100	15	9:21:03	60							9:21:03	9:22:03					0:00:00	0:01:00
101	3	9:21:06	30											9:21:06	9:21:36	0:00:00	0:00:30
102	9	9:21:15	30	9:21:15	9:21:45											0:00:00	0:00:30
103	9	9:21:24	90			9:21:24	9:22:54									0:00:00	0:01:30
104	9	9:21:33	90					9:21:33	9:23:03							0:00:00	0:01:30
105	30	9:22:03	30	9:22:03	9:22:33											0:00:00	0:00:30
106	30	9:22:33	30	9:22:33	9:23:03											0:00:00	0:00:30
107	3	9:22:36	30							9:22:36	9:23:06					0:00:00	0:00:30
108	30	9:23:06	60	9:23:06	9:24:06											0:00:00	0:01:00
109	9	9:23:15	90			9:23:15	9:24:45									0:00:00	0:01:30
110	3	9:23:18	30					9:23:18	9:23:48							0:00:00	0:00:30
111	3	9:23:21	30							9:23:21	9:23:51					0:00:00	0:00:30
112	30	9:23:51	30					9:23:51	9:24:21							0:00:00	0:00:30
113	3	9:23:54	30							9:23:54	9:24:24					0:00:00	0:00:30
114	9	9:24:03	30									9:24:03	9:24:33			0:00:00	0:00:30
115	15	9:24:18	30	9:24:18	9:24:48											0:00:00	0:00:30
116	30	9:24:48	60	9:24:48	9:25:48											0:00:00	0:01:00
117	3	9:24:51	60			9:24:51	9:25:51									0:00:00	0:01:00
118	30	9:25:21	90					9:25:21	9:26:51							0:00:00	0:01:30
119	9	9:25:30	30							9:25:30	9:26:00					0:00:00	0:00:30
120	9	9:25:39	60									9:25:39	9:26:39			0:00:00	0:01:00
121	30	9:26:09	60	9:26:09	9:27:09											0:00:00	0:01:00
122	3	9:26:12	60			9:26:12	9:27:12									0:00:00	0:01:00
123	3	9:26:15	60							9:26:15	9:27:15					0:00:00	0:01:00
124	9	9:26:24	60											9:26:24	9:27:24	0:00:00	0:01:00
125	30	9:26:54	30					9:26:54	9:27:24							0:00:00	0:00:30
126	9	9:27:03	30									9:27:03	9:27:33			0:00:00	0:00:30
127	9	9:27:12	60	9:27:12	9:28:12											0:00:00	0:01:00
128	3	9:27:15	60			9:27:15	9:28:15									0:00:00	0:01:00
129	9	9:27:24	60					9:27:24	9:28:24							0:00:00	0:01:00
130	3	9:27:27	30							9:27:27	9:27:57					0:00:00	0:00:30
131	30	9:27:57	60							9:27:57	9:28:57					0:00:00	0:01:00
132	15	9:28:12	30	9:28:12	9:28:42											0:00:00	0:00:30
133	15	9:28:27	60			9:28:27	9:29:27									0:00:00	0:01:00
134	3	9:28:30	30					9:28:30	9:29:00							0:00:00	0:00:30
135	3	9:28:33	60									9:28:33	9:29:33			0:00:00	0:01:00
136	30	9:29:03	30	9:29:03	9:29:33											0:00:00	0:00:30
137	30	9:29:33	30	9:29:33	9:30:03											0:00:00	0:00:30
138	9	9:29:42	60			9:29:42	9:30:42									0:00:00	0:01:00
139	30	9:30:12	30	9:30:12	9:30:42											0:00:00	0:00:30
140	3	9:30:15	30					9:30:15	9:30:45							0:00:00	0:00:30
141	15	9:30:30	30							9:30:30	9:31:00					0:00:00	0:00:30
142	3	9:30:33	60									9:30:33	9:31:33			0:00:00	0:01:00
143	30	9:31:03	30	9:31:03	9:31:33											0:00:00	0:00:30
144	3	9:31:06	60			9:31:06	9:32:06									0:00:00	0:01:00
145	3	9:31:09	90					9:31:09	9:32:39							0:00:00	0:01:30

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)	[illegible] (hr:min)
										9:31:12	9:32:12					0:00:00	0:01:00
146	3	9:31:12	60													0:00:00	0:01:00
147	30	9:31:42	60	9:31:42	9:32:42							9:31:51	9:33:21			0:00:00	0:01:30
148	9	9:31:51	90											9:32:00	9:33:30	0:00:00	0:01:30
149	9	9:32:00	90													0:00:00	0:01:00
150	9	9:32:09	60			9:32:09	9:33:09									0:00:00	0:01:00
151	3	9:32:12	60							9:32:12	9:33:12					0:00:24	0:01:24
152	3	9:32:15	60					9:32:39	9:33:39							0:00:24	0:01:24
153	3	9:32:18	60	9:32:42	9:33:42											0:00:48	0:02:18
154	3	9:32:21	90			9:33:09	9:34:39									0:00:36	0:01:06
155	15	9:32:36	30							9:33:12	9:33:42					0:00:42	0:01:42
156	3	9:32:39	60									9:33:21	9:34:21			0:00:48	0:01:18
157	3	9:32:42	30											9:33:30	9:34:00	0:00:54	0:02:24
158	3	9:32:45	90					9:33:39	9:35:09							0:00:48	0:01:48
159	9	9:32:54	60	9:33:42	9:34:42											0:00:39	0:01:09
160	9	9:33:03	30							9:33:42	9:34:12					0:00:54	0:01:24
161	3	9:33:06	30											9:34:00	9:34:30	0:00:57	0:01:27
162	9	9:33:15	30							9:34:12	9:34:42					0:01:03	0:02:03
163	3	9:33:18	60									9:34:21	9:35:21			0:01:09	0:02:09
164	3	9:33:21	60											9:34:30	9:35:30	0:01:03	0:02:03
165	15	9:33:36	60			9:34:39	9:35:39									0:00:57	0:01:57
166	9	9:33:45	60	9:34:42	9:35:42											0:00:54	0:01:54
167	3	9:33:48	60							9:34:42	9:35:42					0:01:12	0:02:42
168	9	9:33:57	90					9:35:09	9:36:39							0:01:21	0:02:21
169	3	9:34:00	60									9:35:21	9:36:21			0:01:00	0:02:30
170	30	9:34:30	90											9:35:30	9:37:00	0:01:00	0:02:00
171	9	9:34:39	60			9:35:39	9:36:39									0:00:54	0:02:24
172	9	9:34:48	90	9:35:42	9:37:12											0:00:45	0:01:15
173	9	9:34:57	30							9:35:42	9:36:12					0:01:06	0:02:06
174	9	9:35:06	60							9:36:12	9:37:12					0:01:12	0:02:12
175	3	9:35:09	60									9:36:21	9:37:21			0:01:27	0:01:57
176	3	9:35:12	30					9:36:39	9:37:09							0:01:24	0:01:54
177	3	9:35:15	30			9:36:39	9:37:09							9:37:00	9:38:30	0:01:30	0:03:00
178	15	9:35:30	90					9:37:09	9:38:09							0:01:36	0:02:36
179	3	9:35:33	60													0:01:06	0:01:36
180	30	9:36:03	30			9:37:09	9:37:39									0:00:54	0:01:24
181	15	9:36:18	30	9:37:12	9:37:42											0:00:51	0:01:51
182	3	9:36:21	60							9:37:12	9:38:12					0:00:51	0:01:21
183	9	9:36:30	30									9:37:21	9:37:51			0:00:39	0:02:09
184	30	9:37:00	90			9:37:39	9:39:09									0:00:27	0:00:57
185	15	9:37:15	30	9:37:42	9:38:12											0:00:33	0:01:03
186	3	9:37:18	30									9:37:51	9:38:21			0:00:42	0:02:12
187	9	9:37:27	90					9:38:09	9:39:39							0:00:36	0:01:36
188	9	9:37:36	60	9:38:12	9:39:12											0:00:27	0:01:27
189	9	9:37:45	60							9:38:12	9:39:12					0:00:27	0:00:57
190	9	9:37:54	30									9:38:21	9:38:51			0:00:33	0:01:03
191	3	9:37:57	30											9:38:30	9:39:00	0:00:45	0:01:15
192	9	9:38:06	30									9:38:51	9:39:21			0:00:45	0:02:15
193	9	9:38:15	90											9:39:00	9:40:30	0:00:39	0:01:09
194	15	9:38:30	30			9:39:09	9:39:39										

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] Time (hr:min)	[illegible] Time (hr:min)
195	15	9:38:45	60	9:39:12	9:40:12					9:39:12	9:39:42					0:00:27	0:01:27
196	3	9:38:48	30									9:39:21	9:39:51			0:00:24	0:00:54
197	9	9:38:57	30					9:39:39	9:40:09							0:00:24	0:00:54
198	30	9:39:27	30			9:39:42	9:40:12									0:00:12	0:00:42
199	15	9:39:42	30							9:39:45	9:40:45					0:00:00	0:00:30
200	3	9:39:45	60													0:00:00	0:01:00
201	30	9:40:15	90	9:40:15	9:41:45											0:00:00	0:01:30
202	3	9:40:18	30			9:40:18	9:40:48									0:00:00	0:00:30
203	3	9:40:21	30					9:40:21	9:40:51							0:00:00	0:00:30
204	3	9:40:24	30									9:40:24	9:40:54			0:00:00	0:00:30
205	15	9:40:39	30											9:40:39	9:41:09	0:00:00	0:00:30
206	9	9:40:48	30			9:40:48	9:41:18									0:00:00	0:01:30
207	9	9:40:57	90					9:40:57	9:42:27							0:00:00	0:01:00
208	3	9:41:00	60							9:41:00	9:42:00					0:00:00	0:00:30
209	3	9:41:03	30									9:41:03	9:41:33			0:00:00	0:00:30
210	30	9:41:33	30			9:41:33	9:42:03									0:00:00	0:01:30
211	3	9:41:36	90									9:41:36	9:43:06			0:00:00	0:00:30
212	3	9:41:39	30											9:41:39	9:42:09	0:00:00	0:01:30
213	30	9:42:09	90	9:42:09	9:43:39											0:00:00	0:00:30
214	3	9:42:12	30			9:42:12	9:42:42									0:00:00	0:01:30
215	3	9:42:15	90							9:42:15	9:43:45					0:00:00	0:00:30
216	3	9:42:18	30											9:42:18	9:42:48	0:00:00	0:00:30
217	9	9:42:27	30					9:42:27	9:42:57							0:00:00	0:00:30
218	3	9:42:30	30			9:42:42	9:43:12									0:00:12	0:00:42
219	9	9:42:39	60											9:42:48	9:43:48	0:00:09	0:01:09
220	15	9:42:54	30					9:42:57	9:43:27							0:00:03	0:00:33
221	3	9:42:57	30									9:43:06	9:43:36			0:00:09	0:00:39
222	30	9:43:27	60			9:43:27	9:44:27									0:00:00	0:01:00
223	9	9:43:36	60					9:43:36	9:44:36							0:00:00	0:01:00
224	30	9:44:06	30	9:44:06	9:44:36											0:00:00	0:00:30
225	3	9:44:09	60							9:44:09	9:45:09					0:00:00	0:01:00
226	3	9:44:12	90									9:44:12	9:45:42			0:00:00	0:01:30
227	3	9:44:15	30											9:44:15	9:44:45	0:00:00	0:00:30
228	15	9:44:30	60			9:44:30	9:45:30									0:00:00	0:01:00
229	9	9:44:39	90	9:44:39	9:46:09											0:00:00	0:01:30
230	15	9:44:54	30					9:44:54	9:45:24							0:00:00	0:00:30
231	15	9:45:09	30							9:45:09	9:45:39					0:00:00	0:00:30
232	3	9:45:12	60											9:45:12	9:46:12	0:00:00	0:01:00
233	9	9:45:21	30					9:45:24	9:45:54							0:00:03	0:00:33
234	9	9:45:30	30			9:45:30	9:46:00									0:00:00	0:00:30
235	3	9:45:33	60							9:45:39	9:46:39					0:00:06	0:01:06
236	3	9:45:36	90									9:45:42	9:47:12			0:00:06	0:01:36
237	30	9:46:06	90			9:46:06	9:47:36									0:00:00	0:01:30
238	30	9:46:36	60	9:46:36	9:47:36											0:00:00	0:01:00
239	30	9:47:06	60					9:47:06	9:48:06							0:00:00	0:01:00
240	3	9:47:09	90							9:47:09	9:48:39					0:00:00	0:01:30
241	9	9:47:18	30									9:47:18	9:47:48			0:00:00	0:00:30
242	15	9:47:33	60											9:47:33	9:48:33	0:00:00	0:01:00
243	30	9:48:03	30	9:48:03	9:48:33											0:00:00	0:00:30

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	Time (hr:min)	Time (hr:min)
																0:00:00	0:00:30
244	3	9:48:06	30			9:48:06	9:48:36									0:00:00	0:01:00
245	3	9:48:09	60					9:48:09	9:49:09							0:00:00	0:01:00
246	30	9:48:39	60	9:48:39	9:49:39											0:00:00	0:00:30
247	30	9:49:09	30			9:49:09	9:49:39									0:00:00	0:01:00
248	30	9:49:39	60	9:49:39	9:50:39											0:00:00	0:01:00
249	9	9:49:48	60			9:49:48	9:50:48									0:00:00	0:01:00
250	3	9:49:51	60					9:49:51	9:50:51							0:00:00	0:00:30
251	3	9:49:54	30							9:49:54	9:50:24					0:00:00	0:01:30
252	9	9:50:03	90									9:50:03	9:51:33			0:00:00	0:00:30
253	15	9:50:18	30											9:50:18	9:50:48	0:00:03	0:01:03
254	3	9:50:21	60							9:50:24	9:51:24					0:00:15	0:00:45
255	3	9:50:24	30	9:50:39	9:51:09											0:00:21	0:00:51
256	3	9:50:27	30			9:50:48	9:51:18									0:00:12	0:01:12
257	9	9:50:36	60											9:50:48	9:51:48	0:00:12	0:01:12
258	3	9:50:39	60					9:50:51	9:51:51							0:00:27	0:01:27
259	3	9:50:42	60	9:51:09	9:52:09											0:00:33	0:02:03
260	3	9:50:45	90			9:51:18	9:52:48									0:00:36	0:01:36
261	3	9:50:48	60							9:51:24	9:52:24					0:00:42	0:01:42
262	3	9:50:51	60									9:51:33	9:52:33			0:00:48	0:01:18
263	9	9:51:00	30											9:51:48	9:52:18	0:00:42	0:01:42
264	9	9:51:09	60					9:51:51	9:52:51							0:00:57	0:01:27
265	3	9:51:12	30	9:52:09	9:52:39											0:01:03	0:01:33
266	3	9:51:15	30							9:52:24	9:53:54			9:52:18	9:52:48	0:01:06	0:02:36
267	3	9:51:18	90									9:52:33	9:53:03			0:01:12	0:01:42
268	3	9:51:21	30													0:01:15	0:02:45
269	3	9:51:24	90	9:52:39	9:54:09											0:01:21	0:02:21
270	3	9:51:27	60			9:52:48	9:53:48							9:52:48	9:53:18	0:01:12	0:01:42
271	9	9:51:36	30					9:52:51	9:54:21							0:01:12	0:02:42
272	3	9:51:39	90									9:53:03	9:54:03			0:00:54	0:01:54
273	30	9:52:09	60											9:53:18	9:54:48	0:01:06	0:02:36
274	3	9:52:12	90			9:53:48	9:54:18									0:01:06	0:01:36
275	30	9:52:42	30							9:53:54	9:55:24					0:00:57	0:02:27
276	15	9:52:57	90									9:54:03	9:55:03			0:01:03	0:02:03
277	3	9:53:00	60													0:01:00	0:01:30
278	9	9:53:09	30	9:54:09	9:54:39											0:01:00	0:01:30
279	9	9:53:18	30			9:54:18	9:54:48									0:00:48	0:02:18
280	15	9:53:33	90					9:54:21	9:55:51							0:01:03	0:02:03
281	3	9:53:36	60	9:54:39	9:55:39											0:01:09	0:01:39
282	3	9:53:39	30			9:54:48	9:55:18							9:54:48	9:55:48	0:01:06	0:02:06
283	3	9:53:42	60									9:55:03	9:56:03			0:01:12	0:02:12
284	9	9:53:51	60			9:55:18	9:55:48									0:01:24	0:01:54
285	3	9:53:54	30							9:55:24	9:56:54					0:01:27	0:02:57
286	3	9:53:57	90													0:01:12	0:01:42
287	30	9:54:27	30	9:55:39	9:56:09											0:01:12	0:01:42
288	9	9:54:36	30			9:55:48	9:56:18							9:55:48	9:56:18	0:01:09	0:01:39
289	3	9:54:39	30					9:55:51	9:56:51							0:01:09	0:02:09
290	3	9:54:42	60									9:56:03	9:56:33			0:01:18	0:01:48
291	3	9:54:45	30													0:01:15	0:02:45
292	9	9:54:54	90	9:56:09	9:57:39												

Queuing Simulation [illegible]

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	Time (hr:min)	Time (hr:min)
293	30	9:55:24	90			9:56:18	9:57:48							9:56:18	9:57:48	0:00:54	0:02:24
294	30	9:55:54	90									9:56:33	9:57:03			0:00:24	0:01:54
295	15	9:56:09	30					9:56:51	9:57:21							0:00:24	0:00:54
296	15	9:56:24	30							9:56:54	9:57:54					0:00:27	0:00:57
297	9	9:56:33	60									9:57:03	9:57:33			0:00:21	0:01:21
298	3	9:56:36	30					9:57:21	9:58:21							0:00:27	0:00:57
299	9	9:56:45	60									9:57:33	9:58:33			0:00:36	0:01:36
300	30	9:57:15	60													0:00:18	0:01:18
301	9	9:57:24	30	9:57:39	9:58:09											0:00:15	0:00:45
302	3	9:57:27	30			9:57:48	9:58:18									0:00:21	0:00:51
303	15	9:57:42	30											9:57:48	9:58:18	0:00:06	0:00:36
304	9	9:57:51	90							9:57:54	9:59:24					0:00:03	0:01:33
305	30	9:58:21	30	9:58:21	9:58:51											0:00:00	0:00:30
306	15	9:58:36	60			9:58:36	9:59:36									0:00:00	0:00:30
307	3	9:58:39	60					9:58:39	9:59:39							0:00:00	0:01:00
308	30	9:59:09	30	9:59:09	9:59:39											0:00:00	0:01:00
309	9	9:59:18	30									9:59:18	9:59:48			0:00:00	0:00:30
310	3	9:59:21	30											9:59:21	9:59:51	0:00:00	0:00:30
311	30	9:59:51	30	9:59:51	10:00:21											0:00:00	0:00:30
closed	15	10:00:06															

#	Time (min)	Time (hr:min)	Time (min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
start		9:00															
1	10	9:00:10	6	9:00:10	9:00:16											0:00:00	0:00:06
2	1	9:00:11	6			9:00:11	9:00:17									0:00:00	0:00:06
3	3	9:00:14	9					9:00:14	9:00:23							0:00:00	0:00:09
4	3	9:00:17	3	9:00:17	9:00:20											0:00:00	0:00:03
5	3	9:00:20	6	9:00:20	9:00:26											0:00:00	0:00:06
6	1	9:00:21	6			9:00:21	9:00:27									0:00:00	0:00:06
7	5	9:00:26	3	9:00:26	9:00:29											0:00:00	0:00:03
8	1	9:00:27	9			9:00:27	9:00:36									0:00:00	0:00:09
9	1	9:00:28	6					9:00:28	9:00:34							0:00:00	0:00:06
10	1	9:00:29	6	9:00:29	9:00:35											0:00:00	0:00:06
11	1	9:00:30	3							9:00:30	9:00:33					0:00:00	0:00:03
12	1	9:00:31	3									9:00:31	9:00:34			0:00:00	0:00:03
13	10	9:00:41	9	9:00:41	9:00:50											0:00:00	0:00:09
14	1	9:00:42	3			9:00:42	9:00:45									0:00:00	0:00:03
15	1	9:00:43	3					9:00:43	9:00:46							0:00:00	0:00:03
16	5	9:00:48	6			9:00:48	9:00:54									0:00:00	0:00:06
17	1	9:00:49	3					9:00:49	9:00:52							0:00:00	0:00:03
18	10	9:00:59	3	9:00:59	9:01:02											0:00:00	0:00:03
19	3	9:01:02	3	9:01:02	9:01:05											0:00:00	0:00:03
20	1	9:01:03	6			9:01:03	9:01:09									0:00:00	0:00:06
21	3	9:01:06	6	9:01:06	9:01:12											0:00:00	0:00:06
22	10	9:01:16	3	9:01:16	9:01:19											0:00:00	0:00:03
23	3	9:01:19	6	9:01:19	9:01:25											0:00:00	0:00:06
24	3	9:01:22	3			9:01:22	9:01:25									0:00:00	0:00:03
25	10	9:01:32	3	9:01:32	9:01:35											0:00:00	0:00:03
26	1	9:01:33	3			9:01:33	9:01:36									0:00:00	0:00:03
27	1	9:01:34	3					9:01:34	9:01:37							0:00:00	0:00:03
28	10	9:01:44	3	9:01:44	9:01:47											0:00:00	0:00:03
29	3	9:01:47	3	9:01:47	9:01:50											0:00:00	0:00:03
30	3	9:01:50	3	9:01:50	9:01:53											0:00:00	0:00:03
31	1	9:01:51	3			9:01:51	9:01:54									0:00:00	0:00:03
32	10	9:02:01	3	9:02:01	9:02:04											0:00:00	0:00:03
33	1	9:02:02	6			9:02:02	9:02:08									0:00:00	0:00:06
34	1	9:02:03	3					9:02:03	9:02:06							0:00:00	0:00:03
35	10	9:02:13	9	9:02:13	9:02:22											0:00:00	0:00:09
36	1	9:02:14	9			9:02:14	9:02:23									0:00:00	0:00:09
37	10	9:02:24	6	9:02:24	9:02:30											0:00:00	0:00:06
38	1	9:02:25	3			9:02:25	9:02:28									0:00:00	0:00:03
39	10	9:02:35	3	9:02:35	9:02:38											0:00:00	0:00:03
40	3	9:02:38	9	9:02:38	9:02:47											0:00:00	0:00:09
41	10	9:02:48	9	9:02:48	9:02:57											0:00:00	0:00:09
42	3	9:02:51	3			9:02:51	9:02:54									0:00:00	0:00:03
43	1	9:02:52	6					9:02:52	9:02:58							0:00:00	0:00:06
44	10	9:03:02	3	9:03:02	9:03:05											0:00:00	0:00:03
45	5	9:03:07	3	9:03:07	9:03:10											0:00:00	0:00:03
46	10	9:03:17	6	9:03:17	9:03:23											0:00:00	0:00:06
47	10	9:03:27	6	9:03:27	9:03:33											0:00:00	0:00:06
48	1	9:03:28	3			9:03:28	9:03:31									0:00:00	0:00:03
49	3	9:03:31	3			9:03:31	9:03:34									0:00:00	0:00:03

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Time (hr:min)	Time (hr:min)
50	10	9:03:41	6	9:03:41	9:03:47											0:00:00	0:00:06
51	1	9:03:42	3			9:03:42	9:03:45									0:00:00	0:00:03
52	1	9:03:43	3					9:03:43	9:03:46							0:00:00	0:00:03
53	3	9:03:46	3			9:03:46	9:03:49									0:00:00	0:00:03
54	10	9:03:56	6	9:03:56	9:04:02											0:00:00	0:00:06
55	1	9:03:57	9			9:03:57	9:04:06									0:00:00	0:00:09
56	1	9:03:58	3					9:03:58	9:04:01							0:00:00	0:00:03
57	3	9:04:01	6					9:04:01	9:04:07							0:00:00	0:00:06
58	1	9:04:02	3	9:04:02	9:04:05											0:00:00	0:00:03
59	1	9:04:03	3							9:04:03	9:04:06					0:00:00	0:00:03
60	1	9:04:04	6									9:04:04	9:04:10			0:00:00	0:00:06
61	3	9:04:07	3	9:04:07	9:04:10											0:00:00	0:00:03
62	3	9:04:10	6	9:04:10	9:04:16											0:00:00	0:00:06
63	3	9:04:13	6			9:04:13	9:04:19									0:00:00	0:00:06
64	10	9:04:23	6	9:04:23	9:04:29											0:00:00	0:00:06
65	1	9:04:24	3			9:04:24	9:04:27									0:00:00	0:00:03
66	10	9:04:34	3	9:04:34	9:04:37											0:00:00	0:00:03
67	1	9:04:35	3			9:04:35	9:04:38									0:00:00	0:00:03
68	10	9:04:45	3	9:04:45	9:04:48											0:00:00	0:00:03
69	1	9:04:46	6			9:04:46	9:04:52									0:00:00	0:00:06
70	1	9:04:47	9					9:04:47	9:04:56							0:00:00	0:00:09
71	1	9:04:48	3	9:04:48	9:04:51											0:00:00	0:00:03
72	5	9:04:53	3	9:04:53	9:04:56											0:00:00	0:00:03
73	1	9:04:54	9			9:04:54	9:05:03									0:00:00	0:00:09
74	1	9:04:55	6							9:04:55	9:05:01					0:00:00	0:00:06
75	1	9:04:56	3	9:04:56	9:04:59											0:00:00	0:00:03
76	1	9:04:57	3					9:04:57	9:05:00							0:00:00	0:00:03
77	1	9:04:58	3									9:04:58	9:05:01			0:00:00	0:00:03
78	5	9:05:03	3	9:05:03	9:05:06											0:00:00	0:00:03
79	1	9:05:04	3			9:05:04	9:05:07									0:00:00	0:00:03
80	1	9:05:05	3					9:05:05	9:05:08							0:00:00	0:00:03
81	5	9:05:10	3	9:05:10	9:05:13											0:00:00	0:00:03
82	1	9:05:11	6			9:05:11	9:05:17									0:00:00	0:00:06
83	3	9:05:14	3	9:05:14	9:05:17											0:00:00	0:00:03
84	1	9:05:15	6					9:05:15	9:05:21							0:00:00	0:00:06
85	10	9:05:25	3	9:05:25	9:05:28											0:00:00	0:00:03
86	3	9:05:28	3	9:05:28	9:05:31											0:00:00	0:00:03
87	1	9:05:29	3			9:05:29	9:05:32									0:00:00	0:00:03
88	5	9:05:34	3	9:05:34	9:05:37											0:00:00	0:00:03
89	1	9:05:35	3			9:05:35	9:05:38									0:00:00	0:00:03
90	10	9:05:45	3	9:05:45	9:05:48											0:00:00	0:00:03
91	1	9:05:46	3			9:05:46	9:05:49									0:00:00	0:00:03
92	10	9:05:56	3	9:05:56	9:05:59											0:00:00	0:00:03
93	3	9:05:59	6	9:05:59	9:06:05											0:00:00	0:00:06
94	1	9:06:00	3			9:06:00	9:06:03									0:00:00	0:00:03
95	10	9:06:10	3	9:06:10	9:06:13											0:00:00	0:00:03
96	1	9:06:11	3			9:06:11	9:06:14									0:00:00	0:00:03
97	1	9:06:12	6					9:06:12	9:06:18							0:00:00	0:00:06
98	5	9:06:17	6	9:06:17	9:06:23											0:00:00	0:00:06
99	1	9:06:18	3			9:06:18	9:06:21									0:00:00	0:00:03
100	1	9:06:19	3					9:06:19	9:06:22							0:00:00	0:00:03

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
101	1	9:06:20	3							9:06:20	9:06:23					0:00:00	0:00:03
102	1	9:06:21	9			9:06:21	9:06:30									0:00:00	0:00:09
103	5	9:06:26	6	9:06:26	9:06:32											0:00:00	0:00:06
104	3	9:06:29	3					9:06:29	9:06:32							0:00:00	0:00:03
105	10	9:06:39	6	9:06:39	9:06:45											0:00:00	0:00:06
106	5	9:06:44	6			9:06:44	9:06:50									0:00:00	0:00:06
107	3	9:06:47	6	9:06:47	9:06:53											0:00:00	0:00:06
108	3	9:06:50	6			9:06:50	9:06:56									0:00:00	0:00:06
109	3	9:06:53	6	9:06:53	9:06:59											0:00:00	0:00:06
110	1	9:06:54	3					9:06:54	9:06:57							0:00:00	0:00:03
111	1	9:06:55	3							9:06:55	9:06:58					0:00:00	0:00:03
112	3	9:06:58	9			9:06:58	9:07:07									0:00:00	0:00:09
113	5	9:07:03	9	9:07:03	9:07:12											0:00:00	0:00:09
114	10	9:07:13	6	9:07:13	9:07:19											0:00:00	0:00:06
115	1	9:07:14	6			9:07:14	9:07:20									0:00:00	0:00:06
116	1	9:07:15	3					9:07:15	9:07:18							0:00:00	0:00:03
117	1	9:07:16	6							9:07:16	9:07:22					0:00:00	0:00:06
118	5	9:07:21	3	9:07:21	9:07:24											0:00:00	0:00:03
119	10	9:07:31	3	9:07:31	9:07:34											0:00:00	0:00:03
120	1	9:07:32	6			9:07:32	9:07:38									0:00:00	0:00:06
121	5	9:07:37	6	9:07:37	9:07:43											0:00:00	0:00:06
122	10	9:07:47	6	9:07:47	9:07:53											0:00:00	0:00:06
123	10	9:07:57	3	9:07:57	9:08:00											0:00:00	0:00:03
124	10	9:08:07	6	9:08:07	9:08:13											0:00:00	0:00:06
125	1	9:08:08	3			9:08:08	9:08:11									0:00:00	0:00:03
126	3	9:08:11	3			9:08:11	9:08:14									0:00:00	0:00:03
127	1	9:08:12	3					9:08:12	9:08:15							0:00:00	0:00:03
128	1	9:08:13	3	9:08:13	9:08:16											0:00:00	0:00:03
129	5	9:08:18	3	9:08:18	9:08:21											0:00:00	0:00:03
130	3	9:08:21	3	9:08:21	9:08:24											0:00:00	0:00:03
131	1	9:08:22	3			9:08:22	9:08:25									0:00:00	0:00:03
132	3	9:08:25	3	9:08:25	9:08:28											0:00:00	0:00:03
133	10	9:08:35	6	9:08:35	9:08:41											0:00:00	0:00:06
134	1	9:08:36	6			9:08:36	9:08:42									0:00:00	0:00:06
135	1	9:08:37	6					9:08:37	9:08:43							0:00:00	0:00:06
136	1	9:08:38	6							9:08:38	9:08:44					0:00:00	0:00:06
137	3	9:08:41	3	9:08:41	9:08:44											0:00:00	0:00:03
138	3	9:08:44	3	9:08:44	9:08:47											0:00:00	0:00:03
139	3	9:08:47	6	9:08:47	9:08:53											0:00:00	0:00:06
140	5	9:08:52	3			9:08:52	9:08:55									0:00:00	0:00:03
141	5	9:08:57	6	9:08:57	9:09:03											0:00:00	0:00:06
142	1	9:08:58	6			9:08:58	9:09:04									0:00:00	0:00:06
143	1	9:08:59	6					9:08:59	9:09:05							0:00:00	0:00:06
144	1	9:09:00	3							9:09:00	9:09:03					0:00:00	0:00:03
145	3	9:09:03	6	9:09:03	9:09:09											0:00:00	0:00:06
146	1	9:09:04	3			9:09:04	9:09:07									0:00:00	0:00:03
147	1	9:09:05	3					9:09:05	9:09:08							0:00:00	0:00:03
148	1	9:09:06	6							9:09:06	9:09:12					0:00:00	0:00:06
149	10	9:09:16	6	9:09:16	9:09:22											0:00:00	0:00:06
150	1	9:09:17	6			9:09:17	9:09:23									0:00:00	0:00:06
151	1	9:09:18	6					9:09:18	9:09:24							0:00:00	0:00:06

#	Time (min)	Time (hr:min)	Time (min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
152	3	9:09:21	3							9:09:21	9:09:24					0:00:00	0:00:03
153	3	9:09:24	9	9:09:24	9:09:33											0:00:00	0:00:09
154	10	9:09:34	3	9:09:34	9:09:37											0:00:00	0:00:03
155	3	9:09:37	3	9:09:37	9:09:40											0:00:00	0:00:03
156	1	9:09:38	3			9:09:38	9:09:41									0:00:00	0:00:03
157	10	9:09:48	6	9:09:48	9:09:54											0:00:00	0:00:06
158	10	9:09:58	3	9:09:58	9:10:01											0:00:00	0:00:03
159	1	9:09:59	6			9:09:59	9:10:05									0:00:00	0:00:06
160	3	9:10:02	6	9:10:02	9:10:08											0:00:00	0:00:06
161	1	9:10:03	6					9:10:03	9:10:09							0:00:00	0:00:06
162	10	9:10:13	3	9:10:13	9:10:16											0:00:00	0:00:03
163	1	9:10:14	6			9:10:14	9:10:20									0:00:00	0:00:06
164	1	9:10:15	9					9:10:15	9:10:24							0:00:00	0:00:09
165	1	9:10:16	3	9:10:16	9:10:19											0:00:00	0:00:03
166	3	9:10:19	9	9:10:19	9:10:28											0:00:00	0:00:09
167	3	9:10:22	9			9:10:22	9:10:31									0:00:00	0:00:09
168	10	9:10:32	6	9:10:32	9:10:38											0:00:00	0:00:06
169	1	9:10:33	3			9:10:33	9:10:36									0:00:00	0:00:03
170	3	9:10:36	6			9:10:36	9:10:42									0:00:00	0:00:06
171	1	9:10:37	9					9:10:37	9:10:46							0:00:00	0:00:09
172	1	9:10:38	3	9:10:38	9:10:41											0:00:00	0:00:03
173	3	9:10:41	6	9:10:41	9:10:47											0:00:00	0:00:06
174	5	9:10:46	3			9:10:46	9:10:49									0:00:00	0:00:03
175	1	9:10:47	3	9:10:47	9:10:50											0:00:00	0:00:03
176	5	9:10:52	6	9:10:52	9:10:58											0:00:00	0:00:06
177	1	9:10:53	3			9:10:53	9:10:56									0:00:00	0:00:03
178	1	9:10:54	9					9:10:54	9:11:03							0:00:00	0:00:09
179	1	9:10:55	9							9:10:55	9:11:04					0:00:00	0:00:09
180	10	9:11:05	9	9:11:05	9:11:14											0:00:00	0:00:09
181	3	9:11:08	6			9:11:08	9:11:14									0:00:00	0:00:06
182	3	9:11:11	3					9:11:11	9:11:14							0:00:00	0:00:03
183	10	9:11:21	3	9:11:21	9:11:24											0:00:00	0:00:03
184	1	9:11:22	3			9:11:22	9:11:25									0:00:00	0:00:03
185	10	9:11:32	9	9:11:32	9:11:41											0:00:00	0:00:09
186	3	9:11:35	3			9:11:35	9:11:38									0:00:00	0:00:03
187	1	9:11:36	6					9:11:36	9:11:42							0:00:00	0:00:06
188	3	9:11:39	3			9:11:39	9:11:42									0:00:00	0:00:03
189	3	9:11:42	6	9:11:42	9:11:48											0:00:00	0:00:06
190	1	9:11:43	6			9:11:43	9:11:49									0:00:00	0:00:06
191	1	9:11:44	9					9:11:44	9:11:53							0:00:00	0:00:09
192	1	9:11:45	3							9:11:45	9:11:48					0:00:00	0:00:03
193	3	9:11:48	3	9:11:48	9:11:51											0:00:00	0:00:03
194	1	9:11:49	6			9:11:49	9:11:55									0:00:00	0:00:06
195	1	9:11:50	9							9:11:50	9:11:59					0:00:00	0:00:09
196	1	9:11:51	6	9:11:51	9:11:57											0:00:00	0:00:06
197	1	9:11:52	3									9:11:52	9:11:55			0:00:00	0:00:03
198	3	9:11:55	3			9:11:55	9:11:58									0:00:00	0:00:03
199	1	9:11:56	3					9:11:56	9:11:59							0:00:00	0:00:03
200	10	9:12:06	9	9:12:06	9:12:15											0:00:00	0:00:09
201	1	9:12:07	9			9:12:07	9:12:16									0:00:00	0:00:09
202	1	9:12:08	9					9:12:08	9:12:17							0:00:00	0:00:09

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server A Start (hr:min)	Server A End (hr:min)	Server B Start (hr:min)	Server B End (hr:min)	Server C Start (hr:min)	Server C End (hr:min)	Server D Start (hr:min)	Server D End (hr:min)	Server E Start (hr:min)	Server E End (hr:min)	Server F Start (hr:min)	Server F End (hr:min)	Time in System (hr:min)	Time in Queue (hr:min)
203	3	9:12:11	6							9:12:11	9:12:17					0:00:00	0:00:06
204	1	9:12:12	3									9:12:12	9:12:15			0:00:00	0:00:03
205	1	9:12:13	3											9:12:13	9:12:16	0:00:00	0:00:03
206	1	9:12:14	6	9:12:15	9:12:21											0:00:01	0:00:07
207	10	9:12:24	6	9:12:24	9:12:30											0:00:00	0:00:06
208	3	9:12:27	3			9:12:27	9:12:30									0:00:00	0:00:03
209	3	9:12:30	3	9:12:30	9:12:33											0:00:00	0:00:03
210	1	9:12:31	9			9:12:31	9:12:40									0:00:00	0:00:09
211	1	9:12:32	6					9:12:32	9:12:38							0:00:00	0:00:06
212	1	9:12:33	3	9:12:33	9:12:36											0:00:00	0:00:03
213	1	9:12:34	3							9:12:34	9:12:37					0:00:00	0:00:03
214	1	9:12:35	3									9:12:35	9:12:38			0:00:00	0:00:03
215	1	9:12:36	6	9:12:36	9:12:42											0:00:00	0:00:06
216	1	9:12:37	3							9:12:37	9:12:40					0:00:00	0:00:03
217	1	9:12:38	3					9:12:38	9:12:41							0:00:00	0:00:03
218	1	9:12:39	6									9:12:39	9:12:45			0:00:00	0:00:06
219	1	9:12:40	6			9:12:40	9:12:46									0:00:00	0:00:06
220	3	9:12:43	3	9:12:43	9:12:46											0:00:00	0:00:03
221	1	9:12:44	6					9:12:44	9:12:50							0:00:00	0:00:06
222	1	9:12:45	6							9:12:45	9:12:51					0:00:00	0:00:06
223	1	9:12:46	6	9:12:46	9:12:52											0:00:00	0:00:06
224	1	9:12:47	9			9:12:47	9:12:56									0:00:00	0:00:09
225	10	9:12:57	3	9:12:57	9:13:00											0:00:00	0:00:03
226	3	9:13:00	3	9:13:00	9:13:03											0:00:00	0:00:03
227	3	9:13:03	6	9:13:03	9:13:09											0:00:00	0:00:06
228	1	9:13:04	3			9:13:04	9:13:07									0:00:00	0:00:03
229	10	9:13:14	3	9:13:14	9:13:17											0:00:00	0:00:03
230	3	9:13:17	6	9:13:17	9:13:23											0:00:00	0:00:06
231	1	9:13:18	6			9:13:18	9:13:24									0:00:00	0:00:06
232	1	9:13:19	3					9:13:19	9:13:22							0:00:00	0:00:03
233	3	9:13:22	3					9:13:22	9:13:25							0:00:00	0:00:03
234	3	9:13:25	3	9:13:25	9:13:28											0:00:00	0:00:03
235	3	9:13:28	6	9:13:28	9:13:34											0:00:00	0:00:06
236	1	9:13:29	3			9:13:29	9:13:32									0:00:00	0:00:03
237	3	9:13:32	6			9:13:32	9:13:38									0:00:00	0:00:06
238	1	9:13:33	6					9:13:33	9:13:39							0:00:00	0:00:06
239	10	9:13:43	6	9:13:43	9:13:49											0:00:00	0:00:06
240	1	9:13:44	9			9:13:44	9:13:53									0:00:00	0:00:09
241	10	9:13:54	9	9:13:54	9:14:03											0:00:00	0:00:09
242	10	9:14:04	6	9:14:04	9:14:10											0:00:00	0:00:06
243	1	9:14:05	6			9:14:05	9:14:11									0:00:00	0:00:06
244	5	9:14:10	6	9:14:10	9:14:16											0:00:00	0:00:06
245	3	9:14:13	6			9:14:13	9:14:19									0:00:00	0:00:06
246	3	9:14:16	6	9:14:16	9:14:22											0:00:00	0:00:06
247	1	9:14:17	6					9:14:17	9:14:23							0:00:00	0:00:06
248	1	9:14:18	6							9:14:18	9:14:24					0:00:00	0:00:06
249	3	9:14:21	3			9:14:21	9:14:24									0:00:00	0:00:03
250	5	9:14:26	3	9:14:26	9:14:29											0:00:00	0:00:03
251	1	9:14:27	9			9:14:27	9:14:36									0:00:00	0:00:09
252	1	9:14:28	3					9:14:28	9:14:31							0:00:00	0:00:03
253	3	9:14:31	9	9:14:31	9:14:40											0:00:00	0:00:09

#	Time (min)	Time (hr:min)	Time (min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Time (hr:min)	Time (hr:min)
254	10	9:14:41	9	9:14:41	9:14:50											0:00:00	0:00:09
255	1	9:14:42	3			9:14:42	9:14:45									0:00:00	0:00:03
256	1	9:14:43	3					9:14:43	9:14:46							0:00:00	0:00:03
257	1	9:14:44	3							9:14:44	9:14:47					0:00:00	0:00:03
258	3	9:14:47	6			9:14:47	9:14:53									0:00:00	0:00:06
259	5	9:14:52	9	9:14:52	9:15:01											0:00:00	0:00:09
260	1	9:14:53	6			9:14:53	9:14:59									0:00:00	0:00:06
261	10	9:15:03	3	9:15:03	9:15:06											0:00:00	0:00:03
262	5	9:15:08	3	9:15:08	9:15:11											0:00:00	0:00:03
263	1	9:15:09	3			9:15:09	9:15:12									0:00:00	0:00:03
264	1	9:15:10	6					9:15:10	9:15:16							0:00:00	0:00:06
265	3	9:15:13	3	9:15:13	9:15:16											0:00:00	0:00:03
266	3	9:15:16	3	9:15:16	9:15:19											0:00:00	0:00:03
267	1	9:15:17	6			9:15:17	9:15:23									0:00:00	0:00:06
268	1	9:15:18	6					9:15:18	9:15:24							0:00:00	0:00:06
269	1	9:15:19	3	9:15:19	9:15:22											0:00:00	0:00:03
270	1	9:15:20	3							9:15:20	9:15:23					0:00:00	0:00:03
271	3	9:15:23	3	9:15:23	9:15:26											0:00:00	0:00:03
272	1	9:15:24	3			9:15:24	9:15:27									0:00:00	0:00:03
273	5	9:15:29	3	9:15:29	9:15:32											0:00:00	0:00:03
274	1	9:15:30	6			9:15:30	9:15:36									0:00:00	0:00:06
275	5	9:15:35	3	9:15:35	9:15:38											0:00:00	0:00:03
276	1	9:15:36	6			9:15:36	9:15:42									0:00:00	0:00:06
277	1	9:15:37	3					9:15:37	9:15:40							0:00:00	0:00:03
278	10	9:15:47	6	9:15:47	9:15:53											0:00:00	0:00:06
279	1	9:15:48	3			9:15:48	9:15:51									0:00:00	0:00:03
280	1	9:15:49	3					9:15:49	9:15:52							0:00:00	0:00:03
281	1	9:15:50	3							9:15:50	9:15:53					0:00:00	0:00:03
282	10	9:16:00	3	9:16:00	9:16:03											0:00:00	0:00:03
283	10	9:16:10	3	9:16:10	9:16:13											0:00:00	0:00:03
284	10	9:16:20	3	9:16:20	9:16:23											0:00:00	0:00:03
285	10	9:16:30	6	9:16:30	9:16:36											0:00:00	0:00:06
286	3	9:16:33	3			9:16:33	9:16:36									0:00:00	0:00:03
287	10	9:16:43	6	9:16:43	9:16:49											0:00:00	0:00:06
288	3	9:16:46	9			9:16:46	9:16:55									0:00:00	0:00:09
289	3	9:16:49	6	9:16:49	9:16:55											0:00:00	0:00:06
290	5	9:16:54	3					9:16:54	9:16:57							0:00:00	0:00:03
291	3	9:16:57	6	9:16:57	9:17:03											0:00:00	0:00:06
292	10	9:17:07	3	9:17:07	9:17:10											0:00:00	0:00:03
293	3	9:17:10	3	9:17:10	9:17:13											0:00:00	0:00:03
294	5	9:17:15	3	9:17:15	9:17:18											0:00:00	0:00:03
295	1	9:17:16	3			9:17:16	9:17:19									0:00:00	0:00:03
296	1	9:17:17	6					9:17:17	9:17:23							0:00:00	0:00:06
297	1	9:17:18	9	9:17:18	9:17:27											0:00:00	0:00:09
298	1	9:17:19	3			9:17:19	9:17:22									0:00:00	0:00:03
299	1	9:17:20	6							9:17:20	9:17:26					0:00:00	0:00:06
300	1	9:17:21	3									9:17:21	9:17:24			0:00:00	0:00:03
301	1	9:17:22	3			9:17:22	9:17:25									0:00:00	0:00:03
302	3	9:17:25	9			9:17:25	9:17:34									0:00:00	0:00:09
303	1	9:17:26	9					9:17:26	9:17:35							0:00:00	0:00:09
304	1	9:17:27	3	9:17:27	9:17:30											0:00:00	0:00:03

	(min)	(hr:min)	(min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)
305	3	9:17:30	9	9:17:30	9:17:39											0:00:00	0:00:09
306	3	9:17:33	3							9:17:33	9:17:36					0:00:00	0:00:03
307	5	9:17:38	3			9:17:38	9:17:41									0:00:00	0:00:03
308	10	9:17:48	6	9:17:48	9:17:54											0:00:00	0:00:06
309	5	9:17:53	3			9:17:53	9:17:56									0:00:00	0:00:03
310	1	9:17:54	6	9:17:54	9:18:00											0:00:00	0:00:06
311	3	9:17:57	6			9:17:57	9:18:03									0:00:00	0:00:06
312	1	9:17:58	3					9:17:58	9:18:01							0:00:00	0:00:03
313	1	9:17:59	6							9:17:59	9:18:05					0:00:00	0:00:06
314	3	9:18:02	6	9:18:02	9:18:08											0:00:00	0:00:06
315	1	9:18:03	6			9:18:03	9:18:09									0:00:00	0:00:06
316	1	9:18:04	6					9:18:04	9:18:10							0:00:00	0:00:06
317	10	9:18:14	6	9:18:14	9:18:20											0:00:00	0:00:06
318	1	9:18:15	9			9:18:15	9:18:24									0:00:00	0:00:09
319	5	9:18:20	3	9:18:20	9:18:23											0:00:00	0:00:03
320	1	9:18:21	3					9:18:21	9:18:24							0:00:00	0:00:03
321	5	9:18:26	9	9:18:26	9:18:35											0:00:00	0:00:09
322	1	9:18:27	3			9:18:27	9:18:30									0:00:00	0:00:03
323	10	9:18:37	3	9:18:37	9:18:40											0:00:00	0:00:03
324	3	9:18:40	6	9:18:40	9:18:46											0:00:00	0:00:06
325	3	9:18:43	3			9:18:43	9:18:46									0:00:00	0:00:03
326	1	9:18:44	3					9:18:44	9:18:47							0:00:00	0:00:03
327	1	9:18:45	3							9:18:45	9:18:48					0:00:00	0:00:03
328	1	9:18:46	3	9:18:46	9:18:49											0:00:00	0:00:03
329	5	9:18:51	9	9:18:51	9:19:00											0:00:00	0:00:09
330	3	9:18:54	6			9:18:54	9:19:00									0:00:00	0:00:06
331	3	9:18:57	3					9:18:57	9:19:00							0:00:00	0:00:03
332	5	9:19:02	3	9:19:02	9:19:05											0:00:00	0:00:03
333	10	9:19:12	3	9:19:12	9:19:15											0:00:00	0:00:03
334	1	9:19:13	3			9:19:13	9:19:16									0:00:00	0:00:03
335	1	9:19:14	3					9:19:14	9:19:17							0:00:00	0:00:03
336	1	9:19:15	3	9:19:15	9:19:18											0:00:00	0:00:03
337	3	9:19:18	3	9:19:18	9:19:21											0:00:00	0:00:03
338	1	9:19:19	3			9:19:19	9:19:22									0:00:00	0:00:03
339	1	9:19:20	6					9:19:20	9:19:26							0:00:00	0:00:06
340	1	9:19:21	3	9:19:21	9:19:24											0:00:00	0:00:03
341	3	9:19:24	3	9:19:24	9:19:27											0:00:00	0:00:03
342	3	9:19:27	9	9:19:27	9:19:36											0:00:00	0:00:09
343	3	9:19:30	9			9:19:30	9:19:39									0:00:00	0:00:09
344	3	9:19:33	6					9:19:33	9:19:39							0:00:00	0:00:06
345	1	9:19:34	9							9:19:34	9:19:43					0:00:00	0:00:09
346	3	9:19:37	6	9:19:37	9:19:43											0:00:00	0:00:06
347	1	9:19:38	3									9:19:38	9:19:41			0:00:00	0:00:03
348	3	9:19:41	3			9:19:41	9:19:44									0:00:00	0:00:03
349	10	9:19:51	6	9:19:51	9:19:57											0:00:00	0:00:06
350	10	9:20:01	6	9:20:01	9:20:07											0:00:00	0:00:06
351	1	9:20:02	3			9:20:02	9:20:05									0:00:00	0:00:03
352	10	9:20:12	3	9:20:12	9:20:15											0:00:00	0:00:03
353	1	9:20:13	9			9:20:13	9:20:22									0:00:00	0:00:09
354	3	9:20:16	6	9:20:16	9:20:22											0:00:00	0:00:06
355	1	9:20:17	6					9:20:17	9:20:23							0:00:00	0:00:06

	[redacted] (min)	[redacted] (hr:min)	[redacted] (min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)	(hr:min)
356	3	9:20:20	6							9:20:20	9:20:26					0:00:00	0:00:06
357	3	9:20:23	6	9:20:23	9:20:29											0:00:00	0:00:06
358	10	9:20:33	9	9:20:33	9:20:42											0:00:00	0:00:09
359	3	9:20:36	9			9:20:36	9:20:45									0:00:00	0:00:09
360	10	9:20:46	9	9:20:46	9:20:55											0:00:00	0:00:09
361	3	9:20:49	3			9:20:49	9:20:52									0:00:00	0:00:03
362	3	9:20:52	6			9:20:52	9:20:58									0:00:00	0:00:06
363	5	9:20:57	3	9:20:57	9:21:00											0:00:00	0:00:03
364	3	9:21:00	3	9:21:00	9:21:03											0:00:00	0:00:03
365	1	9:21:01	6			9:21:01	9:21:07									0:00:00	0:00:06
366	3	9:21:04	3	9:21:04	9:21:07											0:00:00	0:00:03
367	10	9:21:14	3	9:21:14	9:21:17											0:00:00	0:00:03
368	3	9:21:17	6	9:21:17	9:21:23											0:00:00	0:00:06
369	1	9:21:18	3			9:21:18	9:21:21									0:00:00	0:00:03
370	5	9:21:23	9	9:21:23	9:21:32											0:00:00	0:00:09
371	3	9:21:26	9			9:21:26	9:21:35									0:00:00	0:00:09
372	3	9:21:29	6					9:21:29	9:21:35							0:00:00	0:00:06
373	5	9:21:34	9	9:21:34	9:21:43											0:00:00	0:00:09
374	1	9:21:35	3			9:21:35	9:21:38									0:00:00	0:00:03
375	1	9:21:36	3					9:21:36	9:21:39							0:00:00	0:00:03
376	5	9:21:41	6			9:21:41	9:21:47									0:00:00	0:00:06
377	1	9:21:42	6					9:21:42	9:21:48							0:00:00	0:00:06
378	5	9:21:47	6	9:21:47	9:21:53											0:00:00	0:00:06
379	1	9:21:48	3			9:21:48	9:21:51									0:00:00	0:00:03
380	3	9:21:51	6			9:21:51	9:21:57									0:00:00	0:00:06
381	1	9:21:52	3					9:21:52	9:21:55							0:00:00	0:00:03
382	10	9:22:02	3	9:22:02	9:22:05											0:00:00	0:00:03
383	1	9:22:03	3			9:22:03	9:22:06									0:00:00	0:00:03
384	10	9:22:13	6	9:22:13	9:22:19											0:00:00	0:00:06
385	1	9:22:14	3			9:22:14	9:22:17									0:00:00	0:00:03
386	3	9:22:17	3			9:22:17	9:22:20									0:00:00	0:00:03
387	3	9:22:20	3	9:22:20	9:22:23											0:00:00	0:00:03
388	3	9:22:23	6	9:22:23	9:22:29											0:00:00	0:00:06
389	1	9:22:24	6			9:22:24	9:22:30									0:00:00	0:00:06
390	1	9:22:25	3					9:22:25	9:22:28							0:00:00	0:00:03
391	10	9:22:35	6	9:22:35	9:22:41											0:00:00	0:00:06
392	10	9:22:45	3	9:22:45	9:22:48											0:00:00	0:00:03
393	10	9:22:55	9	9:22:55	9:23:04											0:00:00	0:00:09
394	1	9:22:56	6			9:22:56	9:23:02									0:00:00	0:00:06
395	3	9:22:59	3					9:22:59	9:23:02							0:00:00	0:00:03
396	10	9:23:09	6	9:23:09	9:23:15											0:00:00	0:00:06
397	1	9:23:10	3			9:23:10	9:23:13									0:00:00	0:00:03
398	3	9:23:13	9			9:23:13	9:23:22									0:00:00	0:00:09
399	3	9:23:16	6	9:23:16	9:23:22											0:00:00	0:00:06
400	1	9:23:17	3					9:23:17	9:23:20							0:00:00	0:00:03
401	10	9:23:27	9	9:23:27	9:23:36											0:00:00	0:00:09
402	3	9:23:30	3			9:23:30	9:23:33									0:00:00	0:00:03
403	1	9:23:31	6					9:23:31	9:23:37							0:00:00	0:00:06
404	1	9:23:32	3							9:23:32	9:23:35					0:00:00	0:00:03
405	1	9:23:33	3			9:23:33	9:23:36									0:00:00	0:00:03
406	1	9:23:34	3									9:23:34	9:23:37			0:00:00	0:00:03

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] Start (hr:min)	[illegible] End (hr:min)	[illegible] Time (hr:min)	[illegible] Time (hr:min)
start		9:00															
1	5	9:00:05	60	9:00:05	9:01:05											0:00:00	0:01:00
2	3	9:00:08	90			9:00:08	9:01:38									0:00:00	0:01:30
3	3	9:00:11	60					9:00:11	9:01:11							0:00:00	0:01:00
4	3	9:00:14	30							9:00:14	9:00:44					0:00:00	0:00:30
5	5	9:00:19	60									9:00:19	9:01:19			0:00:00	0:01:00
6	3	9:00:22	60											9:00:22	9:01:22	0:00:00	0:01:00
7	10	9:00:32	60							9:00:44	9:01:44					0:00:12	0:01:12
8	1	9:00:33	30	9:01:05	9:01:35											0:00:32	0:01:02
9	10	9:00:43	30					9:01:11	9:01:41							0:00:28	0:00:58
10	5	9:00:48	90									9:01:19	9:02:49			0:00:31	0:02:01
11	1	9:00:49	60											9:01:22	9:02:22	0:00:33	0:01:33
12	10	9:00:59	60	9:01:35	9:02:35											0:00:36	0:01:36
13	1	9:01:00	30			9:01:38	9:02:08									0:00:38	0:01:08
14	1	9:01:01	30					9:01:41	9:02:11							0:00:40	0:01:10
15	1	9:01:02	90							9:01:44	9:03:14					0:00:42	0:02:12
16	5	9:01:07	60			9:02:08	9:03:08									0:01:01	0:02:01
17	1	9:01:08	90					9:02:11	9:03:41							0:01:03	0:02:33
18	1	9:01:09	60											9:02:22	9:03:22	0:01:13	0:02:13
19	10	9:01:19	30	9:02:35	9:03:05											0:01:16	0:01:46
20	10	9:01:29	30									9:02:49	9:03:19			0:01:20	0:01:50
21	3	9:01:32	60	9:03:05	9:04:05											0:01:33	0:02:33
22	1	9:01:33	30			9:03:08	9:03:38									0:01:35	0:02:05
23	1	9:01:34	30							9:03:14	9:03:44					0:01:40	0:02:10
24	10	9:01:44	60									9:03:19	9:04:19			0:01:35	0:02:35
25	1	9:01:45	60											9:03:22	9:04:22	0:01:37	0:02:37
26	1	9:01:46	30			9:03:38	9:04:08									0:01:52	0:02:22
27	3	9:01:49	90					9:03:41	9:05:11							0:01:52	0:03:22
28	1	9:01:50	30							9:03:44	9:04:14					0:01:54	0:02:24
29	1	9:01:51	30	9:04:05	9:04:35											0:02:14	0:02:44
30	1	9:01:52	30			9:04:08	9:04:38									0:02:16	0:02:46
31	1	9:01:53	60							9:04:14	9:05:14					0:02:21	0:03:21
32	1	9:01:54	30									9:04:19	9:04:49			0:02:25	0:02:55
33	1	9:01:55	90											9:04:22	9:05:52	0:02:27	0:03:57
34	1	9:01:56	60	9:04:35	9:05:35											0:02:39	0:03:39
35	3	9:01:59	30			9:04:38	9:05:08									0:02:39	0:03:09
36	1	9:02:00	60									9:04:49	9:05:49			0:02:49	0:03:49
37	1	9:02:01	30			9:05:08	9:05:38									0:03:07	0:03:37
38	1	9:02:02	60					9:05:11	9:06:11							0:03:09	0:04:09
39	5	9:02:07	30							9:05:14	9:05:44					0:03:07	0:03:37
40	10	9:02:17	30	9:05:35	9:06:05											0:03:18	0:03:48
41	1	9:02:18	30			9:05:38	9:06:08									0:03:20	0:03:50
42	1	9:02:19	30							9:05:44	9:06:14					0:03:25	0:03:55
43	1	9:02:20	90									9:05:49	9:07:19			0:03:29	0:04:59
44	10	9:02:30	60											9:05:52	9:06:52	0:03:22	0:04:22
45	1	9:02:31	90	9:06:05	9:07:35											0:03:34	0:05:04
46	10	9:02:41	30			9:06:08	9:06:38									0:03:27	0:03:57
47	1	9:02:42	30					9:06:11	9:06:41							0:03:29	0:03:59
48	3	9:02:45	30							9:06:14	9:06:44					0:03:29	0:03:59

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
49	10	9:02:55	60			9:06:38	9:07:38									0:03:43	0:04:43
50	1	9:02:56	60					9:06:41	9:07:41							0:03:45	0:04:45
51	3	9:02:59	60							9:06:44	9:07:44					0:03:45	0:04:45
52	1	9:03:00	60											9:06:52	9:07:52	0:03:52	0:04:52
53	1	9:03:01	30									9:07:19	9:07:49			0:04:18	0:04:48
54	3	9:03:04	30	9:07:35	9:08:05											0:04:31	0:05:01
55	1	9:03:05	30			9:07:38	9:08:08									0:04:33	0:05:03
56	5	9:03:10	60					9:07:41	9:08:41							0:04:31	0:05:31
57	1	9:03:11	60							9:07:44	9:08:44					0:04:33	0:05:33
58	1	9:03:12	60									9:07:49	9:08:49			0:04:37	0:05:37
59	10	9:03:22	30											9:07:52	9:08:22	0:04:30	0:05:00
60	10	9:03:32	30	9:08:05	9:08:35											0:04:33	0:05:03
61	1	9:03:33	60			9:08:08	9:09:08									0:04:35	0:05:35
62	1	9:03:34	60											9:08:22	9:09:22	0:04:48	0:05:48
63	1	9:03:35	30	9:08:35	9:09:05											0:05:00	0:05:30
64	3	9:03:38	30					9:08:41	9:09:11							0:05:03	0:05:33
65	1	9:03:39	60							9:08:44	9:09:44					0:05:05	0:06:05
66	1	9:03:40	60									9:08:49	9:09:49			0:05:09	0:06:09
67	10	9:03:50	30	9:09:05	9:09:35											0:05:15	0:05:45
68	1	9:03:51	30			9:09:08	9:09:38									0:05:17	0:05:47
69	3	9:03:54	30					9:09:11	9:09:41							0:05:17	0:05:47
70	3	9:03:57	30											9:09:22	9:09:52	0:05:25	0:05:55
71	10	9:04:07	60	9:09:35	9:10:35											0:05:28	0:06:28
72	3	9:04:10	60			9:09:38	9:10:38									0:05:28	0:06:28
73	3	9:04:13	30					9:09:41	9:10:11							0:05:28	0:05:58
74	1	9:04:14	30							9:09:44	9:10:14					0:05:30	0:06:00
75	3	9:04:17	90									9:09:49	9:11:19			0:05:32	0:07:02
76	10	9:04:27	30											9:09:52	9:10:22	0:05:25	0:05:55
77	1	9:04:28	30					9:10:11	9:10:41							0:05:43	0:06:13
78	5	9:04:33	60							9:10:14	9:11:14					0:05:41	0:06:41
79	1	9:04:34	90											9:10:22	9:11:52	0:05:48	0:07:18
80	1	9:04:35	30	9:10:35	9:11:05											0:06:00	0:06:30
81	1	9:04:36	30			9:10:38	9:11:08									0:06:02	0:06:32
82	3	9:04:39	30					9:10:41	9:11:11							0:06:02	0:06:32
83	3	9:04:42	90	9:11:05	9:12:35											0:06:23	0:07:53
84	1	9:04:43	60			9:11:08	9:12:08									0:06:25	0:07:25
85	5	9:04:48	30					9:11:11	9:11:41							0:06:23	0:06:53
86	1	9:04:49	30							9:11:14	9:11:44					0:06:25	0:06:55
87	3	9:04:52	60									9:11:19	9:12:19			0:06:27	0:07:27
88	3	9:04:55	60					9:11:41	9:12:41							0:06:46	0:07:46
89	10	9:05:05	30							9:11:44	9:12:14					0:06:39	0:07:09
90	1	9:05:06	90											9:11:52	9:13:22	0:06:46	0:08:16
91	3	9:05:09	60			9:12:08	9:13:08									0:06:59	0:07:59
92	3	9:05:12	30							9:12:14	9:12:44					0:07:02	0:07:32
93	1	9:05:13	60									9:12:19	9:13:19			0:07:06	0:08:06
94	3	9:05:16	90	9:12:35	9:14:05											0:07:19	0:08:49
95	1	9:05:17	90					9:12:41	9:14:11							0:07:24	0:08:54
96	1	9:05:18	30							9:12:44	9:13:14					0:07:26	0:07:56
97	10	9:05:28	60			9:13:08	9:14:08									0:07:40	0:08:40
98	1	9:05:29	90							9:13:14	9:14:44					0:07:45	0:09:15

Queueing Simulation under condition 1

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
99	3	9:05:32	90									9:13:19	9:14:49			0:07:47	0:09:17
100	3	9:05:35	60											9:13:22	9:14:22	0:07:47	0:08:47
101	1	9:05:36	30	9:14:05	9:14:35											0:08:29	0:08:59
102	1	9:05:37	30			9:14:08	9:14:38									0:08:31	0:09:01
103	1	9:05:38	30					9:14:11	9:14:41							0:08:33	0:09:03
104	1	9:05:39	60											9:14:22	9:15:22	0:08:43	0:09:43
105	3	9:05:42	30	9:14:35	9:15:05											0:08:53	0:09:23
106	10	9:05:52	30			9:14:38	9:15:08									0:08:46	0:09:16
107	1	9:05:53	30					9:14:41	9:15:11							0:08:48	0:09:18
108	1	9:05:54	60							9:14:44	9:15:44					0:08:50	0:09:50
109	1	9:05:55	60									9:14:49	9:15:49			0:08:54	0:09:54
110	10	9:06:05	30	9:15:05	9:15:35											0:09:00	0:09:30
111	1	9:06:06	60			9:15:08	9:16:08									0:09:02	0:10:02
112	3	9:06:09	30					9:15:11	9:15:41							0:09:02	0:09:32
113	1	9:06:10	30											9:15:22	9:15:52	0:09:12	0:09:42
114	1	9:06:11	60	9:15:35	9:16:35											0:09:24	0:10:24
115	1	9:06:12	60					9:15:41	9:16:41							0:09:29	0:10:29
116	1	9:06:13	90							9:15:44	9:17:14					0:09:31	0:11:01
117	1	9:06:14	60									9:15:49	9:16:49			0:09:35	0:10:35
118	1	9:06:15	60											9:15:52	9:16:52	0:09:37	0:10:37
119	3	9:06:18	60			9:16:08	9:17:08									0:09:50	0:10:50
120	3	9:06:21	30	9:16:35	9:17:05											0:10:14	0:10:44
121	1	9:06:22	60					9:16:41	9:17:41							0:10:19	0:11:19
122	3	9:06:25	60									9:16:49	9:17:49			0:10:24	0:11:24
123	1	9:06:26	30											9:16:52	9:17:22	0:10:26	0:10:56
124	5	9:06:31	30	9:17:05	9:17:35											0:10:34	0:11:04
125	10	9:06:41	30			9:17:08	9:17:38									0:10:27	0:10:57
126	1	9:06:42	60							9:17:14	9:18:14					0:10:32	0:11:32
127	1	9:06:43	90											9:17:22	9:18:52	0:10:39	0:12:09
128	5	9:06:48	60	9:17:35	9:18:35											0:10:47	0:11:47
129	1	9:06:49	30			9:17:38	9:18:08									0:10:49	0:11:19
130	10	9:06:59	60					9:17:41	9:18:41							0:10:42	0:11:42
131	1	9:07:00	60									9:17:49	9:18:49			0:10:49	0:11:49
132	1	9:07:01	30			9:18:08	9:18:38									0:11:07	0:11:37
133	3	9:07:04	30							9:18:14	9:18:44					0:11:10	0:11:40
134	1	9:07:05	30	9:18:35	9:19:05											0:11:30	0:12:00
135	3	9:07:08	90			9:18:38	9:20:08									0:11:30	0:13:00
136	5	9:07:13	90					9:18:41	9:20:11							0:11:28	0:12:58
137	1	9:07:14	60							9:18:44	9:19:44					0:11:30	0:12:30
138	10	9:07:24	60									9:18:49	9:19:49			0:11:25	0:12:25
139	3	9:07:27	30											9:18:52	9:19:22	0:11:25	0:11:55
140	3	9:07:30	60	9:19:05	9:20:05											0:11:35	0:12:35
141	1	9:07:31	30											9:19:22	9:19:52	0:11:51	0:12:21
142	1	9:07:32	30							9:19:44	9:20:14					0:12:12	0:12:42
143	1	9:07:33	30									9:19:49	9:20:19			0:12:16	0:12:46
144	1	9:07:34	30											9:19:52	9:20:22	0:12:18	0:12:48
145	3	9:07:37	30	9:20:05	9:20:35											0:12:28	0:12:58
146	1	9:07:38	90			9:20:08	9:21:38									0:12:30	0:14:00
147	10	9:07:48	60					9:20:11	9:21:11							0:12:23	0:13:23
148	10	9:07:58	30							9:20:14	9:20:44					0:12:16	0:12:46

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
149	5	9:08:03	60									9:20:19	9:21:19			0:12:16	0:13:16
150	3	9:08:06	30											9:20:22	9:20:52	0:12:16	0:12:46
151	1	9:08:07	90	9:20:35	9:22:05											0:12:28	0:13:58
152	10	9:08:17	60							9:20:44	9:21:44					0:12:27	0:13:27
153	3	9:08:20	30											9:20:52	9:21:22	0:12:32	0:13:02
154	3	9:08:23	90					9:21:11	9:22:41							0:12:48	0:14:18
155	3	9:08:26	60									9:21:19	9:22:19			0:12:53	0:13:53
156	1	9:08:27	30											9:21:22	9:21:52	0:12:55	0:13:25
157	3	9:08:30	30			9:21:38	9:22:08									0:13:08	0:13:38
158	1	9:08:31	60							9:21:44	9:22:44					0:13:13	0:14:13
159	3	9:08:34	90											9:21:52	9:23:22	0:13:18	0:14:48
160	3	9:08:37	30	9:22:05	9:22:35											0:13:28	0:13:58
161	3	9:08:40	30			9:22:08	9:22:38									0:13:28	0:13:58
162	1	9:08:41	30									9:22:19	9:22:49			0:13:38	0:14:08
163	1	9:08:42	30	9:22:35	9:23:05											0:13:53	0:14:23
164	3	9:08:45	30			9:22:38	9:23:08									0:13:53	0:14:23
165	1	9:08:46	90					9:22:41	9:24:11							0:13:55	0:15:25
166	3	9:08:49	30							9:22:44	9:23:14					0:13:55	0:14:25
167	1	9:08:50	30									9:22:49	9:23:19			0:13:59	0:14:29
168	5	9:08:55	30	9:23:05	9:23:35											0:14:10	0:14:40
169	3	9:08:58	60			9:23:08	9:24:08									0:14:10	0:15:10
170	10	9:09:08	90							9:23:14	9:24:44					0:14:06	0:15:36
171	10	9:09:18	90									9:23:19	9:24:49			0:14:01	0:15:31
172	5	9:09:23	60											9:23:22	9:24:22	0:13:59	0:14:59
173	1	9:09:24	30	9:23:35	9:24:05											0:14:11	0:14:41
174	10	9:09:34	30	9:24:05	9:24:35											0:14:31	0:15:01
175	5	9:09:39	60			9:24:08	9:25:08									0:14:29	0:15:29
176	1	9:09:40	60					9:24:11	9:25:11							0:14:31	0:15:31
177	1	9:09:41	30											9:24:22	9:24:52	0:14:41	0:15:11
178	1	9:09:42	30	9:24:35	9:25:05											0:14:53	0:15:23
179	1	9:09:43	30							9:24:44	9:25:14					0:15:01	0:15:31
180	3	9:09:46	90									9:24:49	9:26:19			0:15:03	0:16:33
181	10	9:09:56	60											9:24:52	9:25:52	0:14:56	0:15:56
182	3	9:09:59	60	9:25:05	9:26:05											0:15:06	0:16:06
183	1	9:10:00	30			9:25:08	9:25:38									0:15:08	0:15:38
184	3	9:10:03	90					9:25:11	9:26:41							0:15:08	0:16:38
185	1	9:10:04	60							9:25:14	9:26:14					0:15:10	0:16:10
186	3	9:10:07	30			9:25:38	9:26:08									0:15:31	0:16:01
187	1	9:10:08	60											9:25:52	9:26:52	0:15:44	0:16:44
188	5	9:10:13	30	9:26:05	9:26:35											0:15:52	0:16:22
189	10	9:10:23	30			9:26:08	9:26:38									0:15:45	0:16:15
190	3	9:10:26	60							9:26:14	9:27:14					0:15:48	0:16:48
191	3	9:10:29	30									9:26:19	9:26:49			0:15:50	0:16:20
192	1	9:10:30	60	9:26:35	9:27:35											0:16:05	0:17:05
193	1	9:10:31	30			9:26:38	9:27:08									0:16:07	0:16:37
194	1	9:10:32	60					9:26:41	9:27:41							0:16:09	0:17:09
195	10	9:10:42	90									9:26:49	9:28:19			0:16:07	0:17:37
196	5	9:10:47	90											9:26:52	9:28:22	0:16:05	0:17:35
197	1	9:10:48	90			9:27:08	9:28:38									0:16:20	0:17:50
198	1	9:10:49	60							9:27:14	9:28:14					0:16:25	0:17:25

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1		Server #2		Server #3		Server #4		Server #5		Server #6		Wait Time (hr:min)	Total Time (hr:min)
				Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)	Start (hr:min)	End (hr:min)		
199	3	9:10:52	60	9:27:35	9:28:35											0:16:43	0:17:43
200	3	9:10:55	90					9:27:41	9:29:11							0:16:46	0:18:16
201	3	9:10:58	60							9:28:14	9:29:14					0:17:16	0:18:16
202	10	9:11:08	60									9:28:19	9:29:19			0:17:11	0:18:11
203	1	9:11:09	30											9:28:22	9:28:52	0:17:13	0:17:43
204	1	9:11:10	90	9:28:35	9:30:05											0:17:25	0:18:55
205	5	9:11:15	30			9:28:38	9:29:08									0:17:23	0:17:53
206	3	9:11:18	30											9:28:52	9:29:22	0:17:34	0:18:04
207	1	9:11:19	60			9:29:08	9:30:08									0:17:49	0:18:49
208	10	9:11:29	90					9:29:11	9:30:41							0:17:42	0:19:12
209	10	9:11:39	60							9:29:14	9:30:14					0:17:35	0:18:35
210	3	9:11:42	60									9:29:19	9:30:19			0:17:37	0:18:37
211	1	9:11:43	30											9:29:22	9:29:52	0:17:39	0:18:09
212	5	9:11:48	30											9:29:52	9:30:22	0:18:04	0:18:34
213	1	9:11:49	60	9:30:05	9:31:05											0:18:16	0:19:16
214	10	9:11:59	30			9:30:08	9:30:38									0:18:09	0:18:39
215	1	9:12:00	30							9:30:14	9:30:44					0:18:14	0:18:44
216	1	9:12:01	30									9:30:19	9:30:49			0:18:18	0:18:48
217	3	9:12:04	90											9:30:22	9:31:52	0:18:18	0:19:48
218	1	9:12:05	30			9:30:38	9:31:08									0:18:33	0:19:03
219	3	9:12:08	30					9:30:41	9:31:11							0:18:33	0:19:03
220	1	9:12:09	60							9:30:44	9:31:44					0:18:35	0:19:35
221	1	9:12:10	60									9:30:49	9:31:49			0:18:39	0:19:39
222	1	9:12:11	60	9:31:05	9:32:05											0:18:54	0:19:54
223	10	9:12:21	90			9:31:08	9:32:38									0:18:47	0:20:17
224	3	9:12:24	90					9:31:11	9:32:41							0:18:47	0:20:17
225	10	9:12:34	30							9:31:44	9:32:14					0:19:10	0:19:40
226	1	9:12:35	60									9:31:49	9:32:49			0:19:14	0:20:14
227	1	9:12:36	30											9:31:52	9:32:22	0:19:16	0:19:46
228	3	9:12:39	60	9:32:05	9:33:05											0:19:26	0:20:26
229	10	9:12:49	30							9:32:14	9:32:44					0:19:25	0:19:55
230	1	9:12:50	30											9:32:22	9:32:52	0:19:32	0:20:02
231	10	9:13:00	60			9:32:38	9:33:38									0:19:38	0:20:38
232	5	9:13:05	30					9:32:41	9:33:11							0:19:36	0:20:06
233	1	9:13:06	60							9:32:44	9:33:44					0:19:38	0:20:38
234	1	9:13:07	60									9:32:49	9:33:49			0:19:42	0:20:42
235	10	9:13:17	60											9:32:52	9:33:52	0:19:35	0:20:35
236	3	9:13:20	90	9:33:05	9:34:35											0:19:45	0:21:15
237	1	9:13:21	30					9:33:11	9:33:41							0:19:50	0:20:20
238	3	9:13:24	90			9:33:38	9:35:08									0:20:14	0:21:44
239	3	9:13:27	90					9:33:41	9:35:11							0:20:14	0:21:44
240	1	9:13:28	60							9:33:44	9:34:44					0:20:16	0:21:16
241	3	9:13:31	60									9:33:49	9:34:49			0:20:18	0:21:18
242	3	9:13:34	90											9:33:52	9:35:22	0:20:18	0:21:48
243	1	9:13:35	60	9:34:35	9:35:35											0:21:00	0:22:00
244	1	9:13:36	60							9:34:44	9:35:44					0:21:08	0:22:08
245	1	9:13:37	60									9:34:49	9:35:49			0:21:12	0:22:12
246	3	9:13:40	30			9:35:08	9:35:38									0:21:28	0:21:58
247	10	9:13:50	30					9:35:11	9:35:41							0:21:21	0:21:51
248	1	9:13:51	30											9:35:22	9:35:52	0:21:31	0:22:01

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
																0:21:43	0:22:13
249	1	9:13:52	30	9:35:35	9:36:05											0:21:45	0:22:15
250	1	9:13:53	30			9:35:38	9:36:08									0:21:47	0:22:17
251	1	9:13:54	30					9:35:41	9:36:11							0:21:40	0:22:10
252	10	9:14:04	30							9:35:44	9:36:14					0:21:42	0:23:12
253	3	9:14:07	90									9:35:49	9:37:19			0:21:42	0:22:12
254	3	9:14:10	30											9:35:52	9:36:22	0:21:52	0:23:22
255	3	9:14:13	90	9:36:05	9:37:35											0:21:52	0:22:22
256	3	9:14:16	30			9:36:08	9:36:38									0:21:54	0:22:54
257	1	9:14:17	60					9:36:11	9:37:11							0:21:56	0:22:56
258	1	9:14:18	60							9:36:14	9:37:14					0:21:54	0:22:54
259	10	9:14:28	60											9:36:22	9:37:22	0:22:07	0:22:37
260	3	9:14:31	30			9:36:38	9:37:08									0:22:32	0:23:32
261	5	9:14:36	60			9:37:08	9:38:08									0:22:34	0:23:34
262	1	9:14:37	60					9:37:11	9:38:11							0:22:36	0:24:06
263	1	9:14:38	90							9:37:14	9:38:44					0:22:38	0:23:08
264	3	9:14:41	30									9:37:19	9:37:49			0:22:40	0:23:10
265	1	9:14:42	30											9:37:22	9:37:52	0:22:52	0:23:52
266	1	9:14:43	60	9:37:35	9:38:35											0:23:01	0:23:31
267	5	9:14:48	30									9:37:49	9:38:19			0:22:54	0:23:24
268	10	9:14:58	30											9:37:52	9:38:22	0:23:00	0:24:00
269	10	9:15:08	60			9:38:08	9:39:08									0:23:02	0:23:32
270	1	9:15:09	30					9:38:11	9:38:41							0:23:09	0:23:39
271	1	9:15:10	30									9:38:19	9:38:49			0:23:11	0:24:11
272	1	9:15:11	60											9:38:22	9:39:22	0:23:21	0:23:51
273	3	9:15:14	30	9:38:35	9:39:05											0:23:24	0:23:54
274	3	9:15:17	30					9:38:41	9:39:11							0:23:17	0:23:47
275	10	9:15:27	30							9:38:44	9:39:14					0:23:21	0:23:51
276	1	9:15:28	30									9:38:49	9:39:19			0:23:32	0:24:32
277	5	9:15:33	60	9:39:05	9:40:05											0:23:34	0:24:34
278	1	9:15:34	60			9:39:08	9:40:08									0:23:36	0:24:06
279	1	9:15:35	30					9:39:11	9:39:41							0:23:38	0:24:08
280	1	9:15:36	30							9:39:14	9:39:44					0:23:42	0:24:12
281	1	9:15:37	30									9:39:19	9:39:49			0:23:42	0:24:12
282	3	9:15:40	30											9:39:22	9:39:52	0:24:00	0:24:30
283	1	9:15:41	30					9:39:41	9:40:11							0:24:02	0:25:02
284	1	9:15:42	60							9:39:44	9:40:44					0:24:04	0:24:34
285	3	9:15:45	30									9:39:49	9:40:19			0:24:06	0:24:36
286	1	9:15:46	30											9:39:52	9:40:22	0:24:09	0:24:39
287	10	9:15:56	30	9:40:05	9:40:35											0:24:07	0:24:37
288	5	9:16:01	30			9:40:08	9:40:38									0:24:09	0:24:39
289	1	9:16:02	30					9:40:11	9:40:41							0:24:16	0:24:46
290	1	9:16:03	30									9:40:19	9:40:49			0:24:18	0:25:18
291	1	9:16:04	60											9:40:22	9:41:22	0:24:30	0:25:30
292	1	9:16:05	60	9:40:35	9:41:35											0:24:23	0:24:53
293	10	9:16:15	30			9:40:38	9:41:08									0:24:25	0:25:25
294	1	9:16:16	60					9:40:41	9:41:41							0:24:27	0:24:57
295	1	9:16:17	30							9:40:44	9:41:14					0:24:31	0:25:01
296	1	9:16:18	30									9:40:49	9:41:19			0:24:49	0:25:19
297	1	9:16:19	30			9:41:08	9:41:38									0:24:54	0:25:24
298	1	9:16:20	30							9:41:14	9:41:44						

Queueing Simulation under condition IV

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
												9:41:19	9:41:49			0:24:58	0:25:28
299	1	9:16:21	30											9:41:22	9:41:52	0:24:56	0:25:26
300	5	9:16:26	30													0:25:06	0:25:36
301	3	9:16:29	30	9:41:35	9:42:05											0:25:06	0:25:36
302	3	9:16:32	30			9:41:38	9:42:08									0:24:59	0:25:59
303	10	9:16:42	60					9:41:41	9:42:41							0:24:52	0:25:22
304	10	9:16:52	30							9:41:44	9:42:14					0:24:56	0:25:56
305	1	9:16:53	60									9:41:49	9:42:49			0:24:58	0:25:28
306	1	9:16:54	30											9:41:52	9:42:22	0:25:10	0:25:40
307	1	9:16:55	30	9:42:05	9:42:35											0:25:12	0:25:42
308	1	9:16:56	30			9:42:08	9:42:38									0:25:17	0:26:17
309	1	9:16:57	60							9:42:14	9:43:14					0:25:22	0:25:52
310	3	9:17:00	30											9:42:22	9:42:52	0:25:34	0:26:04
311	1	9:17:01	30	9:42:35	9:43:05											0:25:36	0:26:06
312	1	9:17:02	30			9:42:38	9:43:08									0:25:36	0:26:06
313	3	9:17:05	30					9:42:41	9:43:11							0:25:43	0:26:43
314	1	9:17:06	60									9:42:49	9:43:49			0:25:45	0:26:15
315	1	9:17:07	30											9:42:52	9:43:22	0:25:57	0:26:27
316	1	9:17:08	30	9:43:05	9:43:35											0:25:57	0:26:57
317	3	9:17:11	60			9:43:08	9:44:08									0:25:57	0:26:57
318	3	9:17:14	60					9:43:11	9:44:11							0:25:50	0:27:20
319	10	9:17:24	90							9:43:14	9:44:44					0:25:57	0:26:57
320	1	9:17:25	60											9:43:22	9:44:22	0:26:05	0:27:05
321	5	9:17:30	60	9:43:35	9:44:35											0:26:09	0:26:39
322	10	9:17:40	30									9:43:49	9:44:19			0:26:27	0:26:57
323	1	9:17:41	30			9:44:08	9:44:38									0:26:25	0:27:25
324	5	9:17:46	60					9:44:11	9:45:11							0:26:30	0:27:30
325	3	9:17:49	60									9:44:19	9:45:19			0:26:23	0:26:53
326	10	9:17:59	30											9:44:22	9:44:52	0:26:33	0:27:33
327	3	9:18:02	60	9:44:35	9:45:35											0:26:33	0:28:03
328	3	9:18:05	90			9:44:38	9:46:08									0:26:36	0:27:06
329	3	9:18:08	30							9:44:44	9:45:14					0:26:39	0:28:09
330	5	9:18:13	90											9:44:52	9:46:22	0:26:55	0:27:55
331	3	9:18:16	60					9:45:11	9:46:11							0:26:55	0:27:55
332	3	9:18:19	60							9:45:14	9:46:14					0:26:57	0:27:57
333	3	9:18:22	60									9:45:19	9:46:19			0:27:12	0:28:12
334	1	9:18:23	60	9:45:35	9:46:35											0:27:40	0:29:10
335	5	9:18:28	90			9:46:08	9:47:38									0:27:42	0:28:12
336	1	9:18:29	30					9:46:11	9:46:41							0:27:35	0:28:05
337	10	9:18:39	30							9:46:14	9:46:44					0:27:39	0:29:09
338	1	9:18:40	90									9:46:19	9:47:49			0:27:41	0:28:41
339	1	9:18:41	60											9:46:22	9:47:22	0:27:53	0:28:53
340	1	9:18:42	60	9:46:35	9:47:35											0:27:56	0:28:26
341	3	9:18:45	30					9:46:41	9:47:11							0:27:58	0:28:28
342	1	9:18:46	30							9:46:44	9:47:14					0:28:15	0:28:45
343	10	9:18:56	30					9:47:11	9:47:41							0:28:15	0:28:45
344	3	9:18:59	30							9:47:14	9:47:44					0:28:18	0:29:18
345	5	9:19:04	60											9:47:22	9:48:22	0:28:30	0:29:30
346	1	9:19:05	60	9:47:35	9:48:35											0:28:30	0:29:00
347	3	9:19:08	30			9:47:38	9:48:08									0:28:32	0:29:32
348	1	9:19:09	60					9:47:41	9:48:41								

Queueing Simulation under condition IV

Cust. #	Interarrival Time (min)	Arrival Time (hr:min)	Service Time (min)	Server #1 Start (hr:min)	Server #1 End (hr:min)	Server #2 Start (hr:min)	Server #2 End (hr:min)	Server #3 Start (hr:min)	Server #3 End (hr:min)	Server #4 Start (hr:min)	Server #4 End (hr:min)	Server #5 Start (hr:min)	Server #5 End (hr:min)	Server #6 Start (hr:min)	Server #6 End (hr:min)	Wait Time (hr:min)	Total Time (hr:min)
349	3	9:19:12	60							9:47:44	9:48:44					0:28:32	0:29:32
350	1	9:19:13	60									9:47:49	9:48:49			0:28:36	0:29:36
351	5	9:19:18	60			9:48:08	9:49:08									0:28:50	0:29:50
352	1	9:19:19	30											9:48:22	9:48:52	0:29:03	0:29:33
353	1	9:19:20	60	9:48:35	9:49:35											0:29:15	0:30:15
354	10	9:19:30	90					9:48:41	9:50:11							0:29:11	0:30:41
355	1	9:19:31	60							9:48:44	9:49:44					0:29:13	0:30:13
356	3	9:19:34	60									9:48:49	9:49:49			0:29:15	0:30:15
357	1	9:19:35	30											9:48:52	9:49:22	0:29:17	0:29:47
358	1	9:19:36	60			9:49:08	9:50:08									0:29:32	0:30:32
359	10	9:19:46	90											9:49:22	9:50:52	0:29:36	0:31:06
360	1	9:19:47	60	9:49:35	9:50:35											0:29:48	0:30:48
361	1	9:19:48	30							9:49:44	9:50:14					0:29:56	0:30:26
362	1	9:19:49	90									9:49:49	9:51:19			0:30:00	0:31:30
363	10	9:19:59	30			9:50:08	9:50:38									0:30:09	0:30:39
364	3	9:20:02	60					9:50:11	9:51:11							0:30:09	0:31:09
365	1	9:20:03	60							9:50:14	9:51:14					0:30:11	0:31:11
366	3	9:20:06	60	9:50:35	9:51:35											0:30:29	0:31:29
367	1	9:20:07	60			9:50:38	9:51:38									0:30:31	0:31:31
368	1	9:20:08	30											9:50:52	9:51:22	0:30:44	0:31:14
369	3	9:20:11	90					9:51:11	9:52:41							0:31:00	0:32:30
370	10	9:20:21	60							9:51:14	9:52:14					0:30:53	0:31:53
371	1	9:20:22	30									9:51:19	9:51:49			0:30:57	0:31:27
372	1	9:20:23	30											9:51:22	9:51:52	0:30:59	0:31:29
373	3	9:20:26	30	9:51:35	9:52:05											0:31:09	0:31:39
374	3	9:20:29	30			9:51:38	9:52:08									0:31:09	0:31:39
375	10	9:20:39	60									9:51:49	9:52:49			0:31:10	0:32:10
376	10	9:20:49	90											9:51:52	9:53:22	0:31:03	0:32:33
377	1	9:20:50	30	9:52:05	9:52:35											0:31:15	0:31:45
378	1	9:20:51	30			9:52:08	9:52:38									0:31:17	0:31:47
379	1	9:20:52	30							9:52:14	9:52:44					0:31:22	0:31:52
380	1	9:20:53	60	9:52:35	9:53:35											0:31:42	0:32:42
381	3	9:20:56	60			9:52:38	9:53:38									0:31:42	0:32:42
382	3	9:20:59	90					9:52:41	9:54:11							0:31:42	0:33:12
383	10	9:21:09	30							9:52:44	9:53:14					0:31:35	0:32:05
384	3	9:21:12	60									9:52:49	9:53:49			0:31:37	0:32:37
385	3	9:21:15	30							9:53:14	9:53:44					0:31:59	0:32:29
386	1	9:21:16	30											9:53:22	9:53:52	0:32:06	0:32:36
387	1	9:21:17	60	9:53:35	9:54:35											0:32:18	0:33:18
388	1	9:21:18	30			9:53:38	9:54:08									0:32:20	0:32:50
389	1	9:21:19	60							9:53:44	9:54:44					0:32:25	0:33:25
390	5	9:21:24	60									9:53:49	9:54:49			0:32:25	0:33:25
391	10	9:21:34	60											9:53:52	9:54:52	0:32:18	0:33:18
392	10	9:21:44	60			9:54:08	9:55:08									0:32:24	0:33:24
393	1	9:21:45	30					9:54:11	9:54:41							0:32:26	0:32:56
394	1	9:21:46	30	9:54:35	9:55:05											0:32:49	0:33:19
395	3	9:21:49	30					9:54:41	9:55:11							0:32:52	0:33:22
396	1	9:21:50	30							9:54:44	9:55:14					0:32:54	0:33:24
397	1	9:21:51	90									9:54:49	9:56:19			0:32:58	0:34:28
398	1	9:21:52	30											9:54:52	9:55:22	0:33:00	0:33:30

References

1 Allen, A. O., Probability, Statistics, and Queueing Theory with Computer Science Applications. Academic Press, 1978.

2 Anderson, E., A Method for the Estimation of Resource Use for Queueing Models. Proc. CMG X International Conference (1979), 157-164.

3 Artis, H. P., A Technique for Establishing Resource Limited Job Class Structures. Proc. CMG X International Conference (1979), 249 - 253.

4 Artis, H. Pat., Estimating Latent Demand for Random Arrival Batch Workloads. Computer Performance 2,1 (March 1981), 26-29.

5 Bard, Y., Some Extensions to Multiclass Queueing Network Analysis. In M. Arato, A. Butrimenko, and E. Gelenbe (eds.), Performance of Computer Systems. North-Holland, 1979.

6 Baskett, Forest., Chandy, K. M., Muntz, R.R., and Palacios, F. G., Open, Closed, and Mixed Networks of Queues with Different Classes of Customers. JACM 22,2 (April 1975), 248-260.

7 Beizer, B., Micro-Analysis of Computer System Performance. Van Nostrand Reinhold, 1978.

8 BGS., CRYSTAL/IMS Modeling Support Library User's Guide. BGS Systems, Inc., Waltham, MA, 1982.

9 BGS., CRYSTALKICS Modeling Support Library User's Guide. BGS Systems, Inc., Waltham, MA, 1982.

10 Buzen, J.P., Computational Algorithms for Closed Queueing Networks with Exponential Servers. CACM 16,9 (September 1973, 527- 531.

11 Buzen, J.P., Fundamental Operational Laws of Computer System Performance. Acta Znformatica 7,2 (1976), 167-182.

12 Buzen, J.P., A Queueing Network Model of MVS. Computing Surveys 10,3 (September 1978), 3 19-33 1.

13 Campbell, D.J. and Heffner, W.J., Measurement and Analysis of Large Operating Systems During System Development. 1968 Fall Joint Computer Conference Proceedings, AFIPS Volume 37 (1968)) AFIPS Press, 903-914.

14 Chandy, K. M., Howard, J.H. and Towsley, D. F., Product Form and Local Balance in Queueing Networks. JACM 24,2 (April 1977), 250-263.

15 Chandy, K.M., Herzog, U. and Woo, L.S., Parametric Analysis of Queueing Networks. IBM Journal of Research and Development 19,1 (January 1975), 50-57.

16 Cooper, J.C. , A Capacity Planning Methodology. IBM Systems Journal 19,1 (1980), 28-45.

17 Cooper, J.C., A Capacity Planning Methodology. IBM Systems Journal 19,1 (1980), 28-35.

18 Cox, D.R. and Miller, H.D., The Theory of Stochastic Processes. Wiley, 1965.

19 Denning, P. J. and Buzen, J. P. , The Operational Analysis of Queueing Network Models. Computing Surveys 10,3 (September 1978), 225-261.

20 Dowdy, L. W. and Breitenlohner, H. J. , A Model of Univac 1100/42 Swapping. Proc. ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems (19811, 36-47. Copyright 1981 by the Association for Computing Machinery.

21 Dowdy, L. W. and Budd, R.M. , File Placement Using Predictive Queueing Models. In R.L. Disney and T.J. Ott (eds.), Applied Probability - Computer Science: The Interface, Vol. II. Birkhauser, 1982, 459-476.

22 Ferrari , D., Computer Systems Performance Evaluation. Prentice-Hall, 1978.

23 Ferrari, D., Serrazi, G. and Zeigner, A, Measurement and Tuning of Computer Systems. Prentice-Hall, 1983.

24 Ferrari, D., Computer Systems Performance Evaluation. Prentice-Hall, 1978.

25 Ferrari, D., Serrazi, G., and Zeigner, A., Measurement and Tuning of Computer Systems. Prentice-Hall, 1983.

26 Gelenbe, E. and Mitrani, I., Analysis and Synthesis of Computer Systems. Academic Press, 1980.

27 Gordon, W.J. and Newell, G.F., Closed Queueing Networks with Exponential Servers. Operations Research 15 (1967), 244-265.

28 Taha, H.A., Operation Research, An Introduction, Sixth edition, Eastern Economy edition, Prentice-Hall India, 2000.

29 Ingolf, S., Introduction to Simulation with GPSS, Prentice-Hall, 1998.

30 Kienzle, M.G. and Sevcik, K.C. , A Systematic Approach to the Performance Modelling of Computer Systems. Proc. IFIP W.G. 7.3 International Symposium on Computer Performance Modelling, Measurement and Evaluation (1979), 3-27.

31 Kienzle, M.G. and Sevcik, K. C., Survey of Analytic Queueing Network Models of Computer Systems. Proc. ACM SIGMETRICS Conference on Simulation, Measurement and Modeling of Computer Systems (1979), 113-129.

32 Kleinrock, L., Queueing Systems - Volume II: Computer Applications. John Wiley & Sons, 1976.

33 Kobayashi, H., Modeling and Analysis - An Introduction to System Performance Evaluation Methodology. Addison-Wesley, 1978.

34 Lassettre, E.R. and Scherr, A.L., Modeling the Performance of the OS/360 Time-Sharing Option (TSO). In Walter Freiberger (ed.), Statistical Computer Performance Evaluation. Academic Press, 1972, 57-72.

35 Lavenberg, S.S. and Reiser, M. , Stationary State Probabilities of Arrival Instants for Closed Queueing Networks with Multiple Types of Customers. Journal of Applied Probability (December 1980).

36 Lavenberg, S.S., Computer Performance Modeling Handbook. Academic Press, 1983.

37 Lazowska, E. D., The Use of Analytic Modelling in System Selection. Proc. CMG XI International Conference (1980), 63-69.

38 Levy, A.I. , Capacity Planning with Queueing Network Models: An IMS Case Study. Proc. CMG X International Conference (1979), 227- 232.

39 Lindzey, G.E. Jr. and Browne, J.C. , Response Analysis of a MultiFunction System. Proc. ACM SIGMETRICS Conference on Simulation, Measurement and Modeling of Computer Systems (1979), 19-26. Copyright 1979 by the Association for Computing Machinery.

40 Lipsky, L. and Church, J.D., Applications of a Queueing Network Model for a Computer System. Computing Surveys 9, 3 (September 1977) 205-222. Copyright 1977 by the Association for Computing Machinery.

41 Little, J.D.C., A Proof of the Queueing Formula L = A W. Operations Research 9 (1961), 383-387.

42 Lo, T.L., Computer Capacity Planning Using Queueing Network Models. Proc. IFIP W.G. 7.3 International Symposium on Computer Performance Modelling, Measurement, and Evaluation (1980), 145-152.

43 Microsoft, Microsoft Internet Security ans Acceleration Server 2000, Microsoft Press, 2001.

44 Microsoft, Microsoft Windows 2000, Microsoft Press, 2002.

45 Muntz, R.R. and Wong, J.W., Asymptotic Properties of Closed Queueing Network Models. Proc. 8th Princeton Conference on Information Sciences and Systems (1974).

46 owne, J.C., Chandy, K.M., Brown, R.M. , Keller, T.W., Towsley, D.F., and Dissley, C.W., BrHierarchical Techniques for Development of Realistic Models of Complex Computer Systems. Proc. IEEE 63,4 (June 1975), 966-975.

47 Patrick. S, and Steven G., Client/ Server Computing, Second edition, Prentice-Hall India, 1997.

48 Reiser, M. and Lavenberg, S.S., Mean Value Analysis of Closed Multichain Queueing Networks. JACM27,2 (April 1980), 313-322.

49 Rose, C.A., A Measurement Procedure for Queueing Network Models of Computer Systems. Computing Surveys 10,3 (September 1978), 263-280.

50 Sanguinetti, John and Billington. Richard., A Multi-Class Queueing Network Model of an Interactive System. Proc. CMG XI International Conference (1980), 50-55.

51 Sauer, C.H. and Chandy, K.M., Computer Systems Performance Modeling. Prentice-Hall, 1981.

52 Sauer, C.H. and Chandy, KM.,Approximate Analysis of Central Server Models. IBM Journal of Research and Development 19,3 (May 1975), 301-313.

53 Sauer, C. H. and Chandy, K. M., Computer Systems Performance Modeling. Prentice-Hall, 1981.

54 Saxton, H. E. and Lamont, G. B., Validation of a DEC-10 Closed Queueing Network Model. Proc. CMG IX International Conference (1978), 143-151.

55 Schweitzer, P., Approximate Analysis of Multiclass Closed Networks of Queues. Proc. International Conference on Stochastic Control and Optimization (1979).

56 Schwetman, H.D., Hybrid Simulation Models of Computer Systems. CACM 21,9 (September 1978), 71 S-723.

57 Sevcik, K.C. and Mitrani, I., The Distribution of Queueing Network States at Input and Output Instants. JaCM 28,2 (April 1981), 358 - 371.

58 Smith, C., Increasing Information Systems Productivity by Software Performance Engineering. Proc. CMG XII International Conference (1981).

59 Smith, C. and Browne, J.C. ,Performance Engineering of Software Systems: A Design-Based Approach. To be published, 1983.

60 Smith, C. and Browne, J.C.,Performance Engineering of Software Systems: A Case Study. 1982 National Computer Conference Proceedings, AFIPS Volume 51 (1982), AFIPS Press, 217-244.

61 Svobodova, Liba., Computer Performance Measurement and Evaluation Methods: Analysis and Applications. North-Holland, 1976.

62 Tibbs, R. W. and Kelly, J. C.,The Application of Analytic and Simulation Models to Size a Large Computer System. Proc. 18th CPEUG Meeting (1982), 231-257.

63 Tolopka, S.J. and Schwetman, H.D., Mix-Dependent Job Scheduling - An Application of Hybrid Simulation. 1979 National Computer Conference Proceedings, AFIPS Volume 48 (1979), AFIPS Press, 45-49.

64 Trivedi, K. S., Probability and Statistics with Reliability, Queuing, and Computer Science Applications. Prentice-Hall, 1982.

65 Zahorjan, J., Sevcik, K.C., Eager, D.L. and Galler, B.I., Bound Analysis of Queueing Networks. CACM25,2 (February 1982), 134-141.

66 Gunther, N. J., The Practical Performance Analyst, iUniverse.com Inc., Lincoln, Nebraska, October 2000.

67 Dumke, R., Rautenstrauch, C., Schmietendorf, A., and Scholz, A., Performance Engineering: State of the Art and Current Trends, Lecture Notes in Computer Science, Heidelberg, Germany, October 2001.

68 Gunther, N. J., Analyzing Computer System Performance with Perl::PDQ, Springer Professional Computing Series, Heidelberg, Germany, 2004.

Web Resources

1. Andrew Ross' links to queueing software.
   http://www.lehigh.edu/~amr5/q/software.html
2. (entered June 8, 2004) Discrete Simulation software list 1.
   http://www.topology.org/soft/sim.html
3. (entered June 8, 2004) Discrete Simulation software list 2.
   http://dmoz.org/Science/Software/Simulation/
4. (entered June 5, 2004) SIMPY. Discrete Event Simulation Language. Public domain.
   http://simpy.sourceforge.net/
5. (entered May 7, 2004) "We have developed Java applets for queueing formulas" Janos Strzik. PQTJ (Practical Queueing Theory in Java)
   http://irh.inf.unideb.hu/user/jsztrik/education/09/english/index.html
6. (entered October 24,2003) Queueing ToolPak 4.0: (by A. Ingolfsson, U of Alberta)
   http://www.bus.ualberta.ca/aingolfsson/QTP/
7. (entered July 29, 2003)
   PDQ (Pretty Damn Quick) written in the C language and open-sourced under GPL.
   http://www.perfdynamics.com/Tools/PDQcode.html
8. (entered April 8, 2003) Open Directory Project (dmoz) SIMULATION. Editor Stanislaw Raczynski.
   http://dmoz.org/Science/Software/Simulation/
9. (entered March 4, 2003) Q1.0. A program for analysing queues. "Q was written by Marko Boon and Michel Vollebregt for the Faculty of Mathematics and Computer Science of the Eindhoven University of Technology. It was written for education purposes."
   http://www.win.tue.nl/cow/Q/html/
10. (entered March 3, 2003)
    Queueing program in MAPLE. Available from the web site for the text book "Discrete-Event System Simulation" (third ed.) by Banks, Carson, Nelson, and Nicol. Click on Source Code. http://www.bcnn.org/
11. (entered March 3, 2003) MATLAB code for queues, by Andrew Ross.
    http://www.lehigh.edu/~amr5/q/matlab.html
12. (entered Nov 5, 2002)
    QLib library. "The library implements a number of functions for solving several

queueing problems encountered in the performace analysis of modern broadband communications networks. " http://keskus.hut.fi/tutkimus/com2/Qlib/

13. (entered Sept. 23, 2002) MAM Solver. (Matrix Analytic Methods Solver). http://www.cs.wm.edu/MAMSolver/
14. (entered on August 10, 2002) From: "Jim Thompson" http://www.geocities.com/qtsplus
15. (entered April 19, 2002) Queuing Model Simulator (QMS). by Stanislaw Raczynski P.O.Box 22-783. 14000 Mexico D.F.Mexico. http://www.raczynski.com/pn/qms.htm
16. (entered dec. 26, 2001) PRISM: Probabilistic Symbolic Model Checker (Version 1.2) http://www.cs.bham.ac.uk/~dxp/prism/
17. (entered on Nov. 11, 2001) Demo-version of SIRIUS+. From Alexander Dudin. Available at his web site http://dudin.iatp.unibel.by/
18. (entered Oct. 16, 2001) Queueing Add-on for Excel to accompany Operations Research Models and Methods by Paul A. Jensen & Jonathan F. Bard. http://www.me.utexas.edu/~jensen/ORMM/computation/unit/que_add/nm_queues.html
19. (entered Oct. 6, 2001) OMNeT++ and Queues. (The queueing tutorial is by Nicky van Foreest. http://www.hit.bme.hu/phd/vargaa/omnetpp.htm
20. (entered Oct. 1, 2001) MARCA (Markov Chain Analyzer) by William Stewart. http://www.csc.ncsu.edu/faculty/WStewart/MARCA/marca.html
21. (entered Sept. 21, 2001) SAS(Statistical Analysis System) has an Operations Rersearch module which includes a queueing simulation tool called QSIM. http://support.sas.com/rnd/app/or/qsim.html
22. (entered Sept 7, 2001) Tom Grossman's Spreadsheet Queueing Simulation Templates in EXCEL. http://www.ucalgary.ca/~grossman/simulation/
23. (entered April, 2001) JPQ - Java Powered Queueing. Version 1.0 (Beta). Developed by Muhammad El-Taha and Bacel Maddah http://www.usm.maine.edu/math/JPQ/
24. (information updated December 10, 2002) Telpack Version 2 http://www.sice.umkc.edu/telpack/

25. QNAT Queueing Network Analysis Tool
Date: Nov. 12, 1999.
26. http://poisson.ecse.rpi.edu/~hema/qnat/
http://poisson.ecse.rpi.edu/~hema/qnat/qnat_download.html
27. (entered June 28, 2000). Software Stochast, of Hank Tijms.
http://www.econ.vu.nl/medewerkers/tijms/default.htm
28. (entered June 28, 2000). The Queueing Theory Cookbook of Samuel Baker gives spreadsheet formulas for some queueing measures.
http://hadm.sph.sc.edu/Courses/J716/qcookbook/index.html
29. (information updated January 11, 2002) Some (basic) queueing tools written by Andrew Ross in MATLAB, available at
http://www.lehigh.edu/~amr5/q/matlab.html
30. (entered June 17, 2000.) There is some EXCEL code to accompany Gross and Harris' queueing text at
ftp://ftp.wiley.com/public/sci_tech_med/queueing_theory/
31. (information added Feb. 2000) Extend at
http://www.imaginethatinc.com/
32. (information added January 14, 1999)
http://www.multimania.com/mocanu/
33. Brian Fox. 1996. Quick Q. For Windows 3.x and Windows 95. Shareware version available from Statlib. http://www.Lib.stat.cmu.edu/DOS/general
34. Explorations with Mathcad: Queuing Theory Applications and Examples
http://www.mathcad.com/addons/library_queuing.asp
35. Mike Tanner. 1995. Practical Queueing Analysis (Book and disk) McGraw Hill.
http://ourworld.compuserve.com/homepages/MITAN