Chapter 1

Introduction

A *distributed system* is composed of individual hosts that are connected via a network. Distributed systems are designed to accommodate a number of users, such that each user has a fully functional view of the system and its resources. Communication among the linked computers (hosts) and access to shared resources are often made transparent to the users in a distributed system. The hosts in the system share resources, such as printers, scanners, disks, and the network, as in some centralized systems. Hosts in a distributed system may have different hardware components and run different versions of the operating system. The scale and complex integration of the components in a distributed system add to its practicality, while at the same increasing the difficulty in designing and understanding it.

1.1 Motivation

Distributed systems have been steadily increasing in number and scale over the past two decades. This increase has been motivated by a decrease in hardware costs, and an increase in hardware speed. This increasing trend intensifies the need for *workload characterization* and *system modelling* in this area. Workload characterization is an important preliminary step in any model design study, as it can be used to identify model parameters and to determine the interval on which the model will be based. It also provides insight into a system's performance under the current system conditions, and it can aid in detecting potential system bottlenecks or other performance limitations. System modelling is useful for studies involving capacity planning, determining hardware upgrades, performance prediction, load sharing considerations, and designing new systems. Depending on the type of model, the user community within a particular system may also be studied and predicted. The confidence of the results determined by the model will largely depend upon the accu-

racy of the model. This stresses the importance of developing tools to aid in the design of representative system models.

The increasing size of distributed systems creates the need for compact models and techniques to efficiently analyze massive amounts of data, which is often characteristic of medium to large scale systems. Methods must be developed that can be automated to take the task of data manipulation out of the hands of the system model designer. Of particular interest are tools and statistical techniques that can massage a large set of data to provide meaningful insight into the overall data set and into the system itself. These tools can be used to reduce the data into a more compact form which is appropriate for inclusion in a synthetic system model.

Another motivation for the work in this thesis is the lack of studies in the area of full-scale distributed system models. Most other distributed system workload modelling studies choose to model only isolated components of the system, such as requests that are intercepted by the file server. In this thesis, the workload is considered as the sum of the individual workstations on the network, and a model is designed to this effect.

1.2 Stages of Workload Model Design

When the final goal is a system simulation or other type of study that requires building a model for a system, the steps involved must be carefully thought through. There are five paramount steps that are essential in the design of a workload model. These steps are becoming the standard and can be found in whole or in part in a number of references, including [Fer78], [Fer81], [FSZ83] and [Bod90]. It is important to realize that the process of designing models is quite often continuous; results in one stage of the workload model design may indicate a need to revisit an earlier stage in the design cycle. The directional arrows shown in the diagram for the workload model design stages in Figure 1.1 indicate possible orders of succession for the stages.

In the first step, *outline model*, a rough draft of the model is formulated based on the intended purpose of the model. Issues, such as measurement session, modelling level, basic workload components, and parameters, are examined and resolved. In the *data collection* step, all data that have been identified as important for the model are collected. These data are then analyzed in the preliminary *workload characterization* step, to determine the specific measurement interval that will be used in the study. Various methods are used to

examine the values and distributions of the parameters. The parameters (or feature set) to comprise the model are determined based on this analysis. The workload characterization can also be used to detect potential problems in the system and to study the operation of the system under the current workload. The next step is *model design*. In this step, the representative components of the workload are extracted using a statistical technique such as cluster analysis. The representative parameter values are assigned to the model components, and then the mix of these components is constructed.

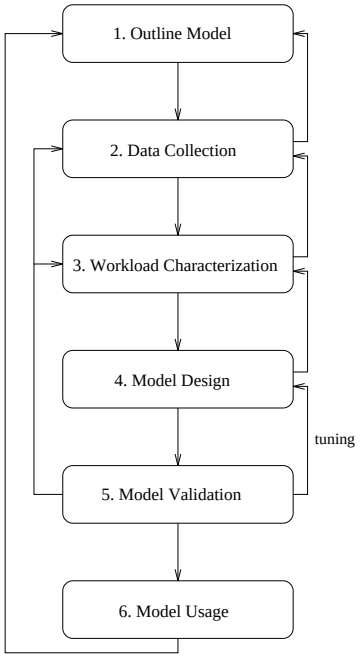

Figure 1.1: Workload Model Design Stages

In the *model validation* phase, the model is run and the representativeness evaluation is performed. If it is determined that the model is not representative, then the parameters or the mixes must be modified, and the model must be run again until the representativeness evaluation meets the desired criteria (tuning). Once it has been determined that the model is representative, the model is ready to be used. The final step, which is called *model usage*, can take many different forms, depending on the type of workload model (real, synthetic, or artificial). The model may be used for different purposes, some of which are performance, prediction, or capacity evaluation studies.

1.3 Thesis Direction

In this section, we discuss our original thesis goals and the transformation of the focus of our thesis. Our original intentions were to design and to carry out a full-fledged simulation of a distributed system model; however, we experienced several problems (both avoidable and nonavoidable) that made a simulation of the model developed for the original goals of this thesis impossible. In the following three sections, we will discuss the original thesis topic, the problems that contributed to the shift in our thesis topic, and the new thesis direction.

1.3.1 Original Thesis Goals

The original goal of this thesis was to examine a scalable hierarchical load sharing policy in a large-scale distributed computing environment, which was similar in scale to CMU's campus-wide Andrew computing system [How88]. As we did not have access to a system as large as Andrew (approximately 5000 hosts), and because of the massive amount of data that would be required to collect data from an Andrew-like system, we chose to design a model for a smaller 71-host system,[1] which would later be extrapolated to a large-scale Andrew-like system. The model for the 71-host system would be validated and replicated such that each instance would represent a separate network component (sub-network) in the large-scale Andrew-like system.

The hierarchical load sharing policy to be examined was based on the hierarchy of the sub-networks. Each sub-network contained a host that served as the load information manager (LIM). This host was responsible for broadcasting load information at periodic intervals and for collecting and updating load information for its local sub-network. The load information that was available to a particular host indicated which hosts were suitable for remote execution on the local ring, as well as progressively less information about the suitability of hosts at each higher level in the hierarchy. Our intent was to examine several load sharing policies that varied in terms of the hosts that were eligible for remote execution (global eligibility or only local sub-network eligibility) and in terms of the amount of load information that was maintained.

Although our intentions were good, it was confirmed that our original study could

[1] The study system is the CDF academic computing system, for which more detailed information is provided in Chapter 3.

not be carried out for two primary reasons. The first reason is that the initial workload characterization of our 71-host system showed that there was little potential for load sharing in our study system. Unless artificial "hot-spots" of activity were induced into the workload, there would rarely be a need for remote placement of jobs, and thus the potential of our hierarchical load sharing policy could not be determined with confidence. The second reason is that the model that we designed for the 71-host system was impossible to validate due to essential data that were not collected. Unfortunately these problems were only realized towards the end of our study, and could not be overcome without recollecting our data and modifying the system data collection software.

1.3.2 Trials and Tribulations of Real Data

The amount of time and effort required to collect and analyze data for distributed systems is non-trivial, and therefore a lot of extra work will be induced if the data must be recollected. As the task of having to recollect data can be quite stress-inducing for data analysts, we have compiled a "data analyst's survival checklist," in the hopes that other data analysts will not have to learn the hard way as we have.

√ 1. Read the manual pages of the data collection tools very carefully, and study all command options.

√ 2. Determine all hardware components (such as caches, disks, I/O controllers) that exist in the system and understand their functionality (such that they could be simulated if required).

√ 3. Study the granularity of measurements and determine if this level of granularity is appropriate for the study at hand.

√ 4. Collect information about to which disk(s) each I/O request should be scheduled in a simulation.

√ 5. Determine if there are any background processes that contribute to the system workload that have not been collected by the basic data collection tool that you are using.

√ 6. Consider the effect of your data collection on the observed system workload, and determine how to minimize and compensate for its effect.

√ 7. If the data collection tools are known to have inaccuracy problems, determine if these limitations are acceptable for the purpose of the study at hand.

√ 8. Determine if the necessary level of resource interaction is represented by the data that will be collected.

√ 9. Depending on the level of representation of the components in the model, it may be necessary to collect additional information that indicates where time is spent in network transfers.

√ 10. Spend a lot of time planning your study and carefully consider every element that might potentially be needed. When in doubt, it is better to collect too much data than not enough.

More detailed information about these ten checklist items is provided in Section A1.1 of Appendix A1. The subsection number in Section A1.1 corresponds to the number in the above checklist.

Many potential tribulations can be avoided by carefully planning your study and by considering potential problem areas in advance. No matter how careful you are, though, collection and analysis of massive amounts of live data may generate problems that are not easily handled. For example, the occasional user may submit outlier jobs with extremely high resource usage. Other users may generate jobs at periodic intervals that are inappropriate for modelling using the conventional stochastic distribution-based techniques. Problems such as these must be considered when selecting techniques that are appropriate to be used in the processes of workload characterization analysis, modelling, and simulation.

1.3.3 Revised Thesis Direction

As time constraints did not allow us sufficient time to recollect and reanalyze our data, we chose to continue our study on the 71-host system, but to shift the focus of our study from the system simulation to the workload characterization and system model design. With respect to the workload model design stages shown in Figure 1.1, our thesis will now concentrate on the first four steps. In particular, we concentrate on the various techniques and tools that are appropriate for distributed system workload characterization and model design.

Information about the type, design goals, and intended usage of the model designed for our study follows. This discussion is intended to give a general overview of the considerations that have governed the design decisions for our model.

Workload Model Type

Static workload models assume that a system is operating at equilibrium and attempt to capture the behaviour of the system at a specific instant in time. Assuming that a

system is operating at equilibrium can result in a model that makes it difficult reproduce fluctuations in the resource demand pattern over time. Such fluctuations are very typical of computer systems that have a lot of variation in their resource demands and arrival patterns. Unlike static workload models, *dynamic* workload models are able to capture the dynamic properties of the workload as time passes. Although dynamic workload models may be better at reproducing the time-varying aspects of a system, they also require significantly more data and parameters in the model. In this thesis, the focus is on the static aspects of workload model design, although some dynamic properties are examined briefly.

Unlike the majority of models that use the mean of the representative model components, the model designed in this thesis is a *stochastic distribution-driven* model. In general, distribution-driven models have larger storage overhead, but are usually more representative. Ideally a model should be examined to determine if the extra overhead required to store component distributions is merited in terms of increased accuracy on the chosen performance indices of the model.

Model Design Goals

There are a number of general goals that are vital in the actual design of system models. These goals are particularly important during the first four phases of the workload design cycle. The general model design goals are summarised in Table 1.1. As an actual simulation of the model that is designed in this thesis was not carried out, the model design is guided by only the first three goals in Table 1.1: compactness, flexibility, and completeness. The results obtained from running the model are usually used to determine the representativeness and reproducibility of the model.

Model Usage

Although the model does not get used until the final stages of the workload model design process, it is important to keep the intended use of the model in mind during all stages of model design. The intended use of the model designed in this thesis is to study the system performance for a load sharing or capacity planning study. To accommodate the requirements of a load sharing study, the model is designed so that jobs could be placed on remote hosts at initiation or migrated at a later time. To accomplish this, we have designed the model so that it is able to produce resource requirements on a per job basis. To be able

Design Goal	Description
compactness	The amount of data and parameters in the model should be smaller than those collected from the actual system.
flexibility	The workload model should be able to produce workloads that are easy to modify for experimentation purposes.
completeness	The model should represent all parameters and data that is known to be of importance in the study.
representativeness	The behaviour of the system under the workload model must be similar to the behaviour of the system under the actual workload.
reproducibility	The workload model should consistently generate workloads that produce similar characteristics, under the same parametric settings.

Table 1.1: General Model Design Goals

to use our model in a capacity planning study, our model is made scalable in terms of the number of users in the system. It is designed so that the number of users in the system is a configurable model parameter that will allow us to study the system's performance under a heavier or lighter user workload. The values of the model parameters for the resource distributions could also be adjusted to test the system under different workload conditions.

1.4 Contributions of the Work

The task of analyzing massive amounts of "live" data is inherently difficult. The sheer complexity of large data sets, combined with the unpredictable nature of users, complicates the task of efficiently handling data sets from large systems. This task is further complicated by the lack of simple all-encompassing tools that can be used to aid in the processes. This results in data analysis studies that require a lot of time and effort on the part of the data analyst. In this thesis, we use graphical and statistical methods to reduce the complexity and to provide meaningful insight into large amounts of workload data that were collected from a distributed system.

Our study furthers the use of cluster analysis in workload characterization studies. Unlike other studies that haphazardly choose a clustering method, we examine ten different clustering methods and choose the method that is most appropriate for our data type as well as for the purpose of our study. We use cluster analysis in our workload characterization study to summarise our wealth of data (dissection) and to provide insightful information about the types of users and commands in our study system.

This thesis also shows how cluster analysis can be used to classify the interactive users and jobs in an academic computer system, to provide the skeleton of a workload model. Unlike other studies that seemingly randomly select the number of components to be used in the model, we examine techniques to determine a suitable number of clusters based on the purpose of the model. We explore techniques that can be automated to reduce the amount of supervision that is required by the data analyst.

The tools and methodology that are used in our measurement collection, workload characterization, and model design phases are outlined in detail throughout this thesis. In addition to the specific results that we present for these processes, the collective wisdom provided by the discussion of the methodology throughout this thesis provides valuable insight for others in the same area of research.

1.5 Thesis Organization

The remainder of this work is organized as follows: Chapter 2 gives a brief overview of the existing literature that relates to this thesis. Chapter 3 establishes the framework of the study by describing the CDF system, the data collection procedure, and other background information. Chapter 4 describes the workload in the CDF system with respect to what is important for the model, and for a distributed system in general. The workload characterization determines the interval and parameters to be included in the model, and examines the current operation of the system. Chapter 5 uses cluster analysis for a workload characterization analysis of the study system. Chapter 6 outlines the workload model and discusses the techniques that were used to devise the user and command classes in the model. It also presents the distributions for the timing of the components in the model. The concluding remarks and directions for future research can be found in Chapter 7.

Chapter-based appendices follow the body of Chapters 1, 3, 4, 5, 6, and 7. These appendices contain additional information for the reader who requires more detailed information, but generally are not required for a first reading of the work.