Appendix A6

Appendix for Chapter 6

This chapter appendix contains additional information for the design of the CDF system model in Chapter 6. Section A6.1 explores the periodic components in the workload. We determine the number of user classes to be used in the model in Section A6.2. Section A6.3 shows cluster dendrograms for the user classes. In Section A6.4, we determine the number of command classes to be used in the model, and the distribution families that will be used to represent their resource usage. The timing distributions required for the model are presented in Section A6.5.

A6.1 Periodic Workload Element Identification

We designed a graphical tool, written in the C programming language, to identify periodic components of the workload. The tool reads an input file that contains starting and elapsed times for each job. It produces jgraph code to plot a horizontal line for each job to represent the period when each job was running. The first job is plotted at the bottom of the y axis, and each job in sequence is plotted a little higher up on the y axis. Any set of jobs in the accounting records can be displayed on a single graph, as the graphical tool adjusts the vertical distance between consecutive jobs on the y axis based on the number of jobs that it has to display.

The tool is reasonably scalable, but if the input file contains too many jobs the vertical distance between jobs becomes too small to distinguish. It can, however, be used to provide insight into small subsets of the data set, such as the activity of certain users that are likely to generate periodic workload components.

Figure A6.1 shows how the tool was used to examine all jobs started by the “root” user on the workstation agrajag. There appears to be a periodic component at 5 minute intervals
on this host. There was also a long-running job that was started at approximately one hour intervals. The end of each of these long-running jobs corresponded to the beginning of the next long-running job.

Closer examination of the accounting records revealed that the periodicity at 5 minute intervals was caused by the six job script that was discussed in Chapter 3. The long-running jobs were \#Xsun jobs that corresponded to user login sessions. When one user logged out, another logged in immediately, thus there is no gap seen between the end of one session and the start of the next session. It was merely coincidental that these login sessions on agrajag were started at one hour intervals; thus, unlike the six command script, these sessions do not represent periodic components of the workload.

A C program was written to search and remove exact occurrences of the 6-command script “root” jobs in the data for each workstation. Figure A6.2 (a) shows all “root” jobs that were part of the six job script that were removed from agrajag’s data. Figure A6.2 (b) shows the jobs that remain for agrajag after these periodic jobs were removed. These non-periodic jobs in Figure A6.2 (b) will be included in the data that are used to determine the distribution-based model for the CDF system.

Figure A6.2 shows that the program was reasonably successful at removing occurrences of the six command script. The intervals in Figure A6.2(a) that did not contain exactly six commands may have been because the accounting facility does not always reliably record the actual order of command arrivals, or they may have been because our program was
confused by the occurrences of the same commands that were not part of the script.

In Figure A6.3 we show all “root” script jobs that were extracted on the 65 workstations. Each workstations is plotted one after the other in alphabetical order, with agrajag at the bottom of the graph and zarquon at the top. This graph contains the 13674 script jobs that were not included in the distribution-based model. As the scale of the graph has been increased to accommodate such a large number of jobs, each six command script appears as a single dot on the graph. There were very few “holes” in the graph, indicating that the program was quite successful at removing occurrences of the six command script.

From Figure A6.3 it is evident that the six command scripts were not synchronised
across the various CDF workstations. When this periodic component of the workload is scheduled in a simulation, this should be taken into consideration; the starting times of the scripts should be staggered.

The tool was also used to examine jobs started by system users (“sys,” “nobody,” “news”) on eddie. Figure A6.4 shows that the system workload generated by the “sys” and “nobody” users was not periodic; thus, these jobs were included in the general distribution-based model.

Figure A6.4: System Jobs started on eddie

Figure A6.5 shows all “news” jobs that were started between 1:00 pm and 3:00 pm on eddie. This graph does not indicate that the “news” user had a periodic workload component. Closer examination of the jobs started by the “news” user on eddie, however, revealed three periodic scripts that were running continuously, as well as a single \#nntpd job that was started every time a user started a news reading program (from any host).

The “news” jobs that were part of the three scripts were periodic, so they should not be included in a stochastic model of the system. These jobs were removed before the cluster analysis that was performed to determine the user classes and command classes in our model. Figure A6.6 (a) shows all news reading jobs (such as trn, xrn, and mn) that were started during the 1:00 pm to 5:00 pm period. The corresponding \#nntpd job that was started by the “news” user on eddie is shown in Figure A6.6 (b). As the arrivals of these \#nntpd jobs were not periodic, they were included in the general stochastic model.
Figure A6.5: Jobs on eddie given by the “news” User

(a) news reading jobs given on any host
(b) corresponding #nmtpd jobs (by “news” user on eddie)

Figure A6.6: Jobs Started When Network News is Read
Although marvin is not discussed in the model design study in Chapter 6, it should be noted that if this host was represented in a stochastic model of the system, it is advisable to handle the backup jobs separately. As Chapter 4 showed that the backup commands were responsible for the majority of CPU and I/O-intensive jobs on marvin, it is important that the model captures their behaviour appropriately. In addition, Figure A6.7 shows that as soon as the backup command finished on one disk partition, it was immediately started on the next disk partition. Since the time required for the backup of a particular disk partition was similar each time, distributions could be used to model the arrival pattern of the backup command.

![Figure A6.7: Jobs on marvin given by “backup” User](image)

In summary, the six command script given by “root” on the workstations and the jobs that were part of the three periodic scripts given by the “news” user on eddie were removed from the data that would be used to generate the distribution-driven model. These components of the workload were periodic, and thus should not be modelled using a stochastic distribution-driven model.
A6.2 Number of User Classes

In this section, we outline the CV graph technique that we have used to determine the number of user classes in our model. This technique is a heuristic method that can be used to forecast which numbers of classes have good potential for use in the model. Until the model is actually validated and tested, the most suitable number of classes to use in the model cannot be known.

The question of how many user classes should be used in the model presents a tradeoff. In general, as more user classes are introduced in the model, the complexity of the model increases, while the variance within each class decreases. As the complexity of the model increases, the number of parameters required for the model and the time required to execute the model generally increase, necessitating efficient storage and lookup routines in a simulation of the model.

To determine which number of user classes to use in our model, we examine $CV_{N_c}$, the average coefficient of variation (CV) for different numbers of clusters ($N_c$). We calculate the average CV for each different $N_c$ value by summing all CV values of each resource variable (CPU, blocks, IAT) in each cluster, as follows:

$$CV_{N_c} = \frac{1}{N_c} \sum_{i=1}^{N_c} \left[ \frac{1}{R} \sum_{j=1}^{R} CV_{ij} \right]$$  \hspace{1cm} (A6.1)

where $N_c$ is the number of clusters, $R$ is the number of resources, and $CV_{ij}$ is the CV for the $j^{th}$ resource in the $i^{th}$ cluster. For a particular $N_c$ value, we also calculate $CV_{c_i}$, the average CV for each cluster $c_i$, as follows

$$CV_{c_i} = \frac{1}{R} \sum_{j=1}^{R} CV_{ij},$$  \hspace{1cm} (A6.2)

and $CV_{r_j}$, the average CV for each resource $r_j$, as follows

$$CV_{r_j} = \frac{1}{N_c} \sum_{i=1}^{N_c} CV_{ij}.$$  \hspace{1cm} (A6.3)

We graphically display the values produced by these three equations, and interpret the results, as discussed in the next two sections, to determine which number of clusters will be used in our model to represent the user classes on the workstations and on eddie.

It should be noted that the MODE= parameter of the SAS CLUSTER procedure was varied for each pair of NCL= and K= parameters for the two-stage density method
[SAS90b], thus giving several cluster configurations for each $N_c$ value from which only the $N_c/k/\text{mode}$ combination with the lowest $\overline{CV}_{N_c}$ is plotted.

A6.2.1 Workstations

Figure A6.8 uses a dotted line to show the average CV values for different $N_c$ values when the two-stage density method is used to make user clusters for the workstations (when the 65 “roof” user sessions were not included). For each $N_c$ value, we use a vertical line with horizontal hash marks to indicate the CV values for each cluster, as calculated by Equation A6.2.

![Figure A6.8: $\overline{CV}_{N_c}$ and $\overline{CV}_{c_i}$ for User Clusters on Workstations](image)

Suitable $N_c$ values for the model are indicated by local minimums in the dotted line for the average CV for each $N_c$ value. Although the largest $N_c$ value ($N_c=9$) has the lowest average CV value, using 9 user classes in the model would result in a model with high parameterization. A more desirable choice for the number of clusters would be $N_c=4$ or $N_c=6$, as these local minimums represent lower variance with respect to the surrounding $N_c$ values and in addition, the average CV value is not much higher than that of $N_c=9$.

When choosing the number of clusters it is also desirable to choose an $N_c$ value that does not have one or two clusters with very high variance, as these may represent components of the model that are hard to represent. The highest cluster CV value for a particular $N_c$ value (i.e., the top of the vertical line) should not be close to the dashed line that is drawn to indicate the overall CV for the data without clustering.
Figure A6.9 is similar to Figure A6.8, except that the CV for each resource, as calculated by Equation A6.3 is shown for each $N_c$ value. Each of the three different resource values is represented on the vertical line using a different symbol. As it is desirable that there is low variance for each resource, the highest cluster CV value for a particular resource should not be close to the dashed overall CV line.

![Graph showing average coefficient of variation vs. number of clusters](image)

**Legend:**
- Overall CV Average
- Average Disk Block CV
- Average CPU Usage CV
- Average Interarrival CV
- Average CV for each $N_c$

Figure A6.9: $CV_{N_c}$ and $CV_{r_j}$ for User Clusters on Workstations

Using these guidelines, we chose four as the number of user classes on the workstations in our model. This number is small enough that it will not add too much in terms of parameterization of the model.

### A6.2.2 Eddie

In this section, we show the CV graphs that were used to determine the number of user classes for eddie. Figure A6.10 displays the CV values for each cluster, while Figure A6.11 shows the CV values for each resource. As the average CV value has a local minimum at $N_c=4$, we chose four user classes for eddie in our model. Additionally, the highest cluster CV and the highest resource CV (IAT) for this $N_c$ value were low compared to the overall CV, making it a suitable choice for the number of classes in our model.
Figure A6.10: $\overline{CV}_{N_c}$ and $\overline{CV}_{c_i}$ for User Clusters on Eddie

Figure A6.11: $\overline{CV}_{N_c}$ and $\overline{CV}_{r_j}$ for User Clusters on Eddie
A6.3 User Class Dendrograms

The dendrogram of the 305 user session observations on the 65 workstations that were grouped by the two-stage density clustering method to form four user clusters is shown in Figure A6.12. The four clusters (also referred to as modes) found by the two-stage density method are indicated by the four groupings of observations in the dendrogram. The last four groups of observations to be joined at the top of the dendrogram indicate the four user classes. The cluster fusion density for these groups of observations becomes lower as more observations are merged into modes.

Figure A6.12: Dendrogram of User Classes on Workstations

Figure A6.13 uses a dendrogram to show how the 253 user session observations on eddie were grouped into four clusters. The four user clusters can be seen clearly in this graph, separated by “white space” between each mode.
4 User Classes formed by Two-stage Density Method

Figure A6.13: Dendrogram of User Classes on Eddie

A6.4 Number of Command Classes

In this section, we outline the steps that were involved in arriving at the technique that was used to determine the number of command classes for our model. As with the user classes, the technique used to determine the number of command classes is a heuristic that can be used to narrow down the choice of potentially good numbers of clusters for the model. Although a simulation of the model may give the most insight into a desirable number of classes to use, simulations are expensive (in terms of time and resources); so heuristics that narrow down the $N_c$ choice are very practical.

We initially used the CV graph technique that was used to determine the number of users classes in Section A6.3. These graphs showed that the average CV gradually decreased and then levelled out as more clusters were chosen; however, it did not pinpoint a particular $N_c$ value that might be a better choice than the others. Figure A6.14 shows the CV graph for the workstations and Figure A6.15 shows the CV graph for Eddie.
Figure A6.14: Average Command Cluster CV per Cluster on Workstations

Figure A6.15: Average Command Cluster CV per Cluster on Eddie

Although the CV graph technique helps to identify ranges of $N_c$ values that have low variance within each cluster, these graphs are not based on any information about how the clusters will be used in the model. Bock [Boc94] advises that for applications of clustering such as model design, the concern is not so much for finding the “true” number of clusters as it is for finding the number of classes for the purpose of the application. Since for each command class, our model will store the distribution parameters that are needed to generate the resource values within the cluster, the number of command clusters should be chosen such that the resource distributions within each command class fit the actual resource data well.
We used the SAS CAPABILITY procedure to fit the data for each resource within each cluster using six different distribution families (normal, lognormal, beta, gamma, weibull, and exponential). The CAPABILITY procedure estimates the parameters of each probability distribution function (pdf) using a maximum likelihood estimate (MLE), and plots these theoretical distributions on a histogram of the actual data that uses a fixed number of intervals. Sample output from the CAPABILITY procedure is shown in Figure A6.16.

\[
\text{SAS } \chi^2_{rc} \text{ error} = \sum_{i=1}^{n} \frac{(g_i - f_i)^2}{f_i}, \quad (A6.4)
\]

for each resource \(r\) within cluster \(c\), where \(g_i\) is the observed frequency in the \(i^{th}\) interval, and \(f_i\) is the expected frequency.

The benefit of using this method is that SAS can be automated such that after it performs the clustering for a certain \(N_c\) value, it will choose the distribution family with the lowest \(\chi^2_{rc}\) error for each resource distribution within each cluster. We found that the number of intervals chosen by SAS were not always the most appropriate, thus biasing the
amount of $\chi^2_{rc}$ error reported for some clusters, as shown in Figure A6.17.

Figure A6.17: Sample of Poor Intervals chosen by SAS CAPABILITY Procedure

This demonstrates the dilemma presented by automatic techniques used to handle large data sets: the data set is so large that there is no choice but to automate the process, yet when live data are used, there are often exceptions that do not conform to the standard rules used in the automation. Although the choice of intervals by SAS is good for most clusters, the clusters for which an incorrect number of intervals was chosen were frequent enough that this technique could not be used to indicate which $N_c$ values produced the best overall fit based on the $\chi^2_{rc}$ of the resource distributions within each cluster.

Thus, we produced a new $\chi^2_{rc}$ error estimate that was not dependent on the number of histogram intervals chosen. We used the MLE parameters provided by the CAPABILITY procedure to determine the theoretical cumulative distribution function (cdf) for each distribution family, and then determined which curve had the best fit by comparing the theoretical cdf $F$ to the cdf for the actual data $G$. In this way, the number of intervals $n$ is defined by the number of distinct resource values within a particular cluster. The $\chi^2_{rc}$ error was calculated per interval as follows:

$$\chi^2_{rc}(F) \text{ error} = \frac{1}{n-1} \sum_{i=1}^{n-1} \left( \frac{G(\frac{x_i + x_{i+1}}{2}) - F(\frac{x_i + x_{i+1}}{2})}{F(\frac{x_i + x_{i+1}}{2})} \right)^2,$$  \hspace{1cm} (A6.5)
for each resource $r$ within cluster $c$, where $F$ is one of the six distribution families and $x_i$ is the left endpoint of the $i^{th}$ interval. An example of a fitted cdf curve is shown in Figure A6.18.

![Fitted cdf](image)

Legend:
- Dotted line: Observed cdf $G$
- Dashed line: Expected cdf $F$ (beta)
- Solid line: Chi-Squared (0.10991)

**Figure A6.18:** Sample of fitting theoretical cdf $F$ to actual data $G$

To determine the overall $\chi^2$ error for a particular $N_c$ value, we summed the $\chi^2_{r,c}$ value of the distribution family having the lowest $\chi^2_{r,c}$ value, for each cluster and divided by the number of clusters. This value was then weighted for each resource $r$, using weight $w_r$, and summed for each resource as follows:

$$
\text{overall } \chi^2 \text{ error} = \sum_{r=1}^{R} w_r \frac{1}{N_c} \left[ \sum_{c=1}^{N_c} \min_{F} (\chi^2_{r,c}(F)) \right],
$$

where $F$ is one of the six distribution families.

Graphs of the overall $\chi^2$ for each $N_c$ value, using uniform resource weights ($w_r = 1$), were used to determine which $N_c$ values were good choices for the number of command classes in our model. In general, intervals with lower overall $\chi^2$ values are most desirable, but the need to reduce the number of parameters in the model should also be considered.

### A6.4.1 Workstations

Figure A6.19 shows the overall $\chi^2$ error for each $N_c$ value on the workstations. The overall $\chi^2$ error is high for the low $N_c$ values, and drops steadily until approximately $N_c = 12$. For $N_c$ values greater than 12, the overall $\chi^2$ error curve levels out and gradually increases, indicating that using these larger $N_c$ values does not provide a better fit, and thus should
be avoided as they increase the parameterization of the model. If \( N_c = 12 \) produces too many parameters in the model (this will depend on the specific implementation), other reasonable choices would be in the range of \( N_c = 7 \) to \( N_c = 9 \), or if an even smaller number of parameters is required, then \( N_c = 3 \) may be a suitable choice.

We chose \( N_c = 8 \) for the number of command clusters on the workstations. Although \( N_c = 8 \) does not produce the lowest overall \( \chi^2 \) error, it produces a fairly small number of parameters in the model, while still having a reasonably low overall \( \chi^2 \) error.

![Graph](image)

Figure A6.19: Overall \( \chi^2 \) Error per \( N_c \) on Workstations

### A6.4.2 Eddie

The overall \( \chi^2 \) error for each \( N_c \) value on eddie, calculated using Equation A6.6, is shown in Figure A6.20. Reasonable choices for the number of clusters to use on eddie would be \( N_c = 14 \) or \( N_c = 19 \), as the overall \( \chi^2 \) error drops steadily until these values. If these values introduce too many parameters into the model, \( N_c = 4 \) may provide the best compromise. As to keep our model compact, we chose \( N_c = 4 \) for the number of command clusters for eddie in our model.
Distributions for Resource Usage

In this section, we present tables that show which distribution families were used to represent the resource usage of each command cluster. The distribution family with the lowest $\chi^2_{r,c}$ error (calculated by Equation A6.5) was selected for each resource $r$ in each command cluster $c$. Table A6.1 shows which distribution families were chosen for the workstations, while Table A6.2 shows which distribution families were chosen for eddie.$^1$

Notice that in most cases the lognormal distribution family was chosen to represent the CPU usage. Only in the low resource usage command classes on eddie (cluster 3 and cluster 4) do we see other distributions that are not as skewed to the right (weibull and beta).

We found that our choice of the $N_c$ value influenced the distribution family that was chosen to represent the resource usage in each cluster. When larger $N_c$ values were chosen, this generally resulted in distribution families that did not have such long tails to the right. Increasing the $N_c$ value resulted in fewer exponential and lognormal functions that were chosen to represent the CPU and disk block usage because the clusters were more defined.

$^1$"Actual value" was chosen when there were 2 or fewer distinct resource values in a cluster. In this case the probabilities and the actual values would be used to represent the resource usage, as they are more representative for the same number of parameters.
### Command Interarrival Distributions

Examination of the command interarrival times (IAT) for each user class provided insight into the CDF workload, in addition to being required in the CDF model. We commence with a discussion of the distributions required for the model.

The starting times of jobs recorded in the processing accounting records were the only data available to determine the interarrival time between consecutive pairs of commands. As the granularity of these starting times were in units of seconds, the interarrival time data that we have represents truncated data for one second intervals. We made the assumption that the data within each one second interval had an exponential distribution, as the general
shape of the data was exponential. If alternative data collection tools that provide finer
granularity for the starting times of commands are available, they can be used to verify this
assumption.

Let \( h \) be the actual data histogram function of the command interarrival times. A
particular interval \( i \) of function \( h \) contains the probability of all interarrival time values in
the interval \([i, i+1)\). Since recorded interarrival time values were truncated, all values in
the interval \([i, i+1)\) were recorded as \( i \).

We fitted an exponential function \( g(x) = ke^{-\lambda x} \) on the interval \([0, \infty)\) and we defined
\( E(\alpha) \) to be the total squared error of \( g \) over the interval \([0, \infty)\):

\[
E(\alpha) = \sum_{i=0}^{\infty} \left( \int_{i}^{i+1} (g(x) - h(x)) \, dx \right)^2,
\]

where \( h \) is the previously defined probability histogram function. We minimized this error
and examined the resulting fitted exponential distribution function \( g \) for each user class on
both the workstations and on eddie, and found that the fit was not very good because of
an overabundance of interarrival time values that were in the 0 second interval.

We therefore decided to treat the interval \([0, 1)\) separately. We fit exponential functions
\( f_0 \) and \( f_1 \) on the intervals \([0, 1)\) and \([1, \infty)\), such that

\[
f(x) = \begin{cases} 
  f_0(x) & \text{if } x \in [0, 1) \\
  f_1(x) & \text{if } x \in [1, \infty). 
\end{cases}
\]

Let \( f_0 \) be the exponential distribution function \( k_0 e^{-\lambda x} \) and similarly let \( f_1 \) be \( k_1 e^{-\lambda x} \).
Let \( p_0 \) be the percentage of interarrival times in the interval \([0, 1)\) and let \( p_1 = 1 - p_0 \). We
impose the following constraints:

\[
\int_{0}^{1} f_0(x) \, dx = p_0 \quad \text{(A6.9)}
\]

\[
\int_{1}^{\infty} f_1(x) \, dx = p_1 = 1 - p_0, \quad \text{(A6.10)}
\]

so that each part of the distribution function is weighted proportionally to the number of
interarrival time values in their respective segments, and such that the combined weight is 1.

Let us define \( E_1(\alpha) \) to be the total squared error of \( f_1 \) over the interval \([1, \infty)\):

\[
E_1(\alpha) = \sum_{i=1}^{\infty} \left( \int_{i}^{i+1} (f_1(x) - h(x)) \, dx \right)^2.
\]

\( \quad \text{(A6.11)} \)
Substituting for \( f_1 \), we get the following.

\[
E_1(\alpha_1) = \sum_{i=1}^{\infty} \left\{ \int_{i}^{i+1} k_1 e^{-\frac{x}{\alpha_1}} - h(x) \, dx \right\}^2
\]

\[
= \sum_{i=1}^{\infty} k_1 \int_{i}^{i+1} \left( e^{-\frac{u}{\alpha_1}} - h(u) \right)^2 \, du
\]

\[
= k_1 \sum_{i=1}^{\infty} \left\{ \int_{i}^{i+1} e^{-\frac{x}{\alpha_1}} \, dx - h(i) \right\}^2
\]

\[
= k_1 \sum_{i=1}^{\infty} \left\{ \left[ -\alpha_1 e^{-\frac{i}{\alpha_1}} \right]^{i+1} - h(i) \right\}^2
\]

\[
= k_1 \sum_{i=1}^{\infty} \left\{ \alpha_1 \left[ e^{-\frac{i}{\alpha_1}} - e^{-\frac{i+1}{\alpha_1}} \right] - h(i) \right\}^2
\]

\[
= k_1 \sum_{i=1}^{M} \left\{ \alpha_1 \left[ e^{-\frac{i}{\alpha_1}} - e^{-\frac{i+1}{\alpha_1}} \right] - h(i) \right\}^2,
\]

where \( M \) is the largest interarrival time value. We minimize \( E_1 \) by differentiating it with respect to \( \alpha_1 \) and solving for \( \alpha_1 \):

\[
E_1' = \frac{d}{d\alpha_1} \left\{ k_1 \sum_{i=1}^{M} \alpha_1 \left[ e^{-\frac{i}{\alpha_1}} - e^{-\frac{i+1}{\alpha_1}} \right] - h(i) \right\}^2
\]

\[
= k_1 \frac{d}{d\alpha_1} \left\{ \sum_{i=1}^{M} \alpha_1 \left[ e^{-\frac{i}{\alpha_1}} - e^{-\frac{i+1}{\alpha_1}} \right] - h(i) \right\}^2
\]

\[
= 0.
\]

(A6.12)

We can use a system like Maple to find the value of \( \alpha_1 \) that satisfies this constraint. Having found this value \( \alpha_1 \), we can then find \( k_1 \) by solving Equation A6.12.

Finally we choose the parameters \( \alpha_0 \) and \( k_0 \) such that:

\[
\int_0^1 f_0(x) \, dx = \rho_0, \quad \text{and}
\]

\[
f_0(1) = f_1(1).
\]

(A6.14)

(A6.15)

The latter condition is imposed to make \( f \) continuous at 1.

Notice that the error over the interval \([0, 1]\) is 0:

\[
E_0 = \left[ \int_0^1 (f_0(x) - h(x)) \, dx \right]^2
\]

\[
= \left[ \int_0^1 f_0(x) \, dx - h(0) \right]^2
\]

\[
= [\rho_0 - h(0)]^2
\]

\[
= 0.
\]

(A6.16)
Therefore, the total error for the combined function $f$ is simply $E_1$.

We present the fitted exponential distribution functions that resulted for user class one on the workstations in Figure A6.21. We show the single fitted exponential distribution function $g$ in Figure A6.21(a), and the combined fitted function $f$ in Figure A6.21(b). Similarly, we show user class 1 on eddie in Figure A6.22. For all user classes, using the combined function $f$ produced a lower total squared error estimate (i.e., $E_1 < E$).

![Figure A6.21: Command IAT Distribution for User Class 1 on Workstations](image1)

![Figure A6.22: Command IAT Distribution for User Class 1 on Eddie](image2)

In concluding this section, we comment on what might have accounted for the large number of 0 second interarrival times that were observed. The large number of user sessions
on eddie and the workstations is likely to have been primarily responsible for the high frequency of 0 second interarrival times. The larger the number of user sessions, the greater the probability of users starting commands at approximately the same time.

Another explanation is the presence of piped commands in the workload. Commands that are executed using the UNIX pipe system would be recorded in the process accounting records with the same starting time. If many commands are executed in this manner, we could expect a high frequency of 0 second interarrival times.

Finally, the system workload and commands that were executed in scripts may also have accounted for this high frequency of 0 second interarrival times. On both eddie and on the workstations, the root user class (class 5) contained higher frequencies of 0 second interarrival times than the other classes. As system users are more likely to run commands in scripts where the next command is started immediately after the previous one, this accounts for this higher frequency of 0 second interarrival times.

A6.5.2 Session Interarrival Distributions

The SAS CAPABILITY procedure was used to determine which distribution family should be used to model the session interarrival time (IST) distributions for each user class. The number of user class IST distributions was small enough that the SAS CAPABILITY procedure could be monitored to ensure an appropriate selection of histogram intervals.

The actual data histogram with six different fitted distributions for user class 1 on the workstations is shown in Figure A6.23, and for eddie in Figure A6.24. The SAS chi-squared error estimate in Equation A6.4 was used to determine which distribution family provided the best fit. The distribution families that provided the best fit for the remaining user classes are listed in Table A6.3.

We have not shown the distribution families for the fifth user class ("root" users) because a simpler way of putting these user sessions onto the hosts in a simulation would be used. As there was exactly one "root" user session on each host that executed commands over the full duration of our study period, a single "root" user session would be modelled on each host accordingly.
Actual Histogram with Fitted Probability Density Curves
Login Sessions for User Class 1 on Workstations

Time Between User Login Session Arrivals

Figure A6.23: IST Distribution for User Class 1 on Workstations
Actual Histogram with Fitted Probability Density Curves
Login Sessions for User Class 1 on Eddie

<table>
<thead>
<tr>
<th>Host Group</th>
<th>User Class</th>
<th>Distribution with Best Fit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eddie</td>
<td>1</td>
<td>Weibull</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Gamma</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Weibull</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Weibull</td>
</tr>
<tr>
<td>Workstations</td>
<td>1</td>
<td>Weibull</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Weibull</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Weibull</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Lognormal</td>
</tr>
</tbody>
</table>

Table A6.3: IST Distributions for each User Class
A6.5.3 Session Duration Distributions

The SAS CAPABILITY procedure and its chi-squared error estimate was also used to determine which distribution family should be used to model the user session duration time distributions for each user class. The session duration distributions are used in the model to determine for how long a particular user session will continue to issue commands.

Figure A6.25 and Figure A6.26 show the histograms with the fitted distributions for user class 1 on the workstations and on eddie, respectively. The best distribution families for each user class, as determined by the SAS chi-squared error estimate, are listed in Table A6.4.

![Figure A6.25: Duration Distribution for User Class 1 on Workstations](image-url)
Actual Histogram with Fitted Probability Density Curves
Login Sessions for User Class 1 on Eddie

Figure A6.26: Duration Distribution for User Class 1 on Eddie

<table>
<thead>
<tr>
<th>Host Group</th>
<th>User Class</th>
<th>Distribution with Best Fit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eddie</td>
<td>1</td>
<td>Beta</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Beta</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Exponential</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Exponential</td>
</tr>
<tr>
<td>Workstations</td>
<td>1</td>
<td>Beta</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Beta</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Gamma</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Gamma</td>
</tr>
</tbody>
</table>

Table A6.4: Duration Distributions for each User Class