A Co-Plot Analysis of Logs and Models of Parallel Workloads

DAVID TALBY, DROR G. FEITELSON and ADI RAVEH
The Hebrew University of Jerusalem

We present a multivariate analysis technique called Co-plot that is especially suitable for few samples of many variables. Co-plot embeds the multi-dimensional samples in two dimensions, in a way that allows key variables to be identified and relations between both variables and observations to be analyzed together. When applied to the workloads on parallel supercomputers, we find two stable perpendicular axes of highly correlated variables, one representing individual job attributes and the other representing multi-job attributes. The different workloads, on the other hand, are rather different from one another, and may also change over time. Synthetic models for workload generation are also analyzed, and found to be reasonable in the sense that they span the same range of variable combinations as the real workloads. However, the spread of real workloads implies that a single model cannot be similar to all of them. This leads us to construct a parameterized model, with parameters that correspond to the two axes identified above. We also find that existing models do not model the temporal structure of the workload well, and hence are wanting for tasks such as comparing schedulers, and that the common methodology for load manipulation of workloads is problematic.

Categories and Subject Descriptors: I.6 [**Computing Methodologies**]: Simulation and Modeling – *Model Validation and Analysis,* I.6 [**Computing Methodologies**]: Simulation and Modeling – *Model Development*; C.4 [**Computer Systems Organization**]: Performance of Systems – *Modeling Techniques*

General Terms: Measurement, Design

Additional Key Words and Phrases: Co-Plot, Multivariate Analysis, Workload Modeling, Parallel Workloads, Parametric Model, Load Manipulation, Non-Stationary Workload, Synthetic Workload

1. INTRODUCTION

A characterization of the workload a system will face is necessary in order to evaluate schedulers, processor allocators, and make many other design decisions. Two kinds of workloads are typically used: A trace of a real production workload, or a synthetic workload produced by a statistical model. For concreteness, we will consider traces and models of parallel supercomputer workloads in this paper.

Production logs have the advantage of being more realistic, as they are a direct recording of a workload that has occurred in practice. However, they may suffer from various problems. For example, the trace may contain errors or otherwise unreliable data: mysterious jobs that exceeded the system's limits, undocumented downtime, dedication of the system to certain users, and patterns of activity that are not generally representative [Koldinger et al., 1991; Windisch et al., 1996; Downey and Feitelson, 1999; Feitelson and Tsafrir, 2006]. In addition, different workloads can be highly variable. Such

This research was supported in part by the Israel Science Foundation (grant no. 167/03). The third author was partially supported by the Racannati Foundation.

Authors' addresses: David Talby and Dror G. Feitelson, School of Computer Science and Engineering, The Hebrew University, Givat Ram, Jerusalem 91904, Israel; davidt@cs.huji.ac.il, feit@cs.huji.ac.il. Adi Raveh, School of Business Administration, The Hebrew University, Mt. Scopus, Jerusalem 91905, Israel; msraveh@olive.mscc.huji.ac.il.

problems limit the degree to which we can draw conclusions from past workloads to predict future ones, or infer from one installation – one hardware configuration, user base, and scheduler – about other ones. It is therefore necessary to map invariants that are common to multiple workloads and can be relied upon to be representative.

The alternative to using production traces is to generate synthetic models [Ferrari, 1972; Calzarossa and Serazzi, 1993], and several such models have been proposed for parallel workloads [Feitelson, 1996; Downey, 1997; Jann et al., 1997; Cirne and Berman, 2001; Lublin and Feitelson, 2003]. Such models are typically based on measurements of real workloads [Agrawala et al., 1976; Feitelson and Tsafrir, 2006]. Models have the advantage over production logs of putting all the assumptions "on the table", and of being more flexible, by allowing their user to easily vary the model's parameters.

The problem with models is that they need to be representative of real workloads. We compare eleven production workloads with the generated output of six statistical models, and test how the models measure up to reality. Our production logs include at least 25,000 jobs and are at least six months long, to ensure that they represent real usage in a production environment at some supercomputing center. This is the first time a data set of this size and diversity has been analyzed, enabling us to present new insights about the commonalities and differences among workloads in production sites.

Based on our analysis of real parallel workloads, we try to identify desired properties of workload models: Which variables should form the backbone of the model? What can we tell about the distributions of these variables? What can we tell about the correlation of these variables and other variables? Can a single model represent all observed workloads? And how should the model be adjusted to match a desired level of load, number of processors, and so forth?

In order to reach the answers, we use a new statistical method called Co-Plot, which is well suited for situations in which few observations but many variables about them are available. Co-Plot works by mapping the observations into a 2-dimensional space, such that relative distances are preserved, and that key variables can be identified with different directions. We find that for our data such a mapping is indeed possible, indicating that a 2-dimensional model should be sufficient. Based on correlations between the different variables, we can assign meanings to the two axes that define such a model: one axis represents the properties of single jobs (e.g. its parallelism and runtime), while the other represents the properties of the workload as a whole (e.g. the arrival rate). We then select a representative variable from each axis, and show how to construct a parametric model that can fit a desired location in the 2-dimensional workload space.

Section 2 presents Co-plot, and Section 3 presents the data set used for the analysis. Sections 4 to 8 analyze the production logs from several angles, leading to an evaluation of the synthetic models available today in Section 9, and considerations for building new models in Sections 10 through 12. Limitations are mentioned in Section 13. Conclusions and future work are presented in Section 14.

2. CO-PLOT

Classical multivariate analysis methods, such as cluster analysis and principal component analysis, analyze variables and observations separately. Co-Plot is a new technique which analyzes them simultaneously. This means that we'll be able to see, in the same analysis, clusters of observations (workloads in our case), clusters of variables, the relations between clusters (correlation between variables, for example), and a characterization of observations (as being above average in certain variables and below in others). The technique has been used before mostly in the area of economics [Lipshitz and Raveh, 1994; Raveh, 2000].

Co-plot is especially suitable for tasks in which there are few observations and relatively many variables – as opposed to regression based techniques, in which the

number of observations must be an order of magnitude larger than the number of variables. This is crucial in our case, in which there are few workloads (eleven production ones and six synthetic ones), and a similar number of variables. We denote the number of observations by *n*, and the number of variables (the dimensionality) by *p*.

Co-plot's output is a visual display of its findings. It is based on two graphs that are superimposed on each other. The first graph maps the *n* observations into a two-dimensional space. This mapping, if it succeeds, conserves relative distance: observations that are close to each other in *p* dimensions are also close in two dimensions, and vice versa. The second graph consists of *p* arrows, representing the variables, and shows the direction of the gradient for each one.

Given an input matrix $Y_{n\times p}$ of *p* variable values for each of *n* observations (see for example Table 2), the analysis consists of four stages. The first is to normalize the variables, which is needed in order to be able to relate them to each other, although each has different units and scale. This is done in the usual way. If $\overline{Y}_j$ is the *j*'th variable's mean, and D_j is its standard deviation, then Y_{ij} is normalized into Z_{ij} by:

$$Z_{ij} := (Y_{ij} - \overline{Y}_j) / D_j \qquad (1)$$

In the second stage, we compute a measure of dissimilarity $S_{ik} \geq 0$ between each pair of observations (rows of $Z_{n\times p}$). A symmetric n×n matrix is produced from all the pairs of observations. In our analysis we use city-block distance – the sum of absolute deviations – as the measure of dissimilarity:

$$S_{ik} = \sum_{j=1}^{p} \left| Z_{ij} - Z_{kj} \right| \qquad (2)$$

In stage three, the *n* observations are mapped by means of a multidimensional scaling (MDS) method from the original *p*-dimensional space into a two dimensional Euclidean space, such that 'close' observations (with a small dissimilarity between them) are close to each other on the map, while 'distant' ones are also distant on the map. Note, however, that we are mainly interested in relative distances. This can be expressed as a relationship between the map distances d_{ik} (which are just the Euclidean distance in 2D) and the corresponding dissimilarity metrics S_{ik}, where we require that

$$S_{ik} < S_{lm} \text{ if and only if } d_{ik} < d_{lm}$$

The MDS we use is Guttman's Smallest Space Analysis, or SSA [Guttman, 1968]. SSA is an iterative mapping technique, in which the need for additional iterations is guided by the coefficient of alienation Θ. The smaller it is, the better that the mapping reflects the original dissimilarities, and values below 0.15 are considered good. To evaluate Θ we first evaluate another metric μ which directly measures the correlation between the dissimilarity measures and the map distances:

$$\mu = \frac{\sum_{i,k,l,m} (S_{ik} - S_{lm})(d_{ik} - d_{lm})}{\sum_{i,k,l,m} \left| S_{ik} - S_{lm} \right| \left| d_{ik} - d_{lm} \right|} \qquad (3)$$

Thus μ can attain the maximal value of 1. This is then used to define Θ as follows:

$$\Theta = \sqrt{1 - \mu^2} \qquad (4)$$

This functional form does not decrease much for mediocre values of μ, but drops sharply when μ approaches 1, so it is only low for very good correlations. Full details of the SSA algorithm are given in [Guttman, 1968]. It is a widely used method in social sciences, and several examples and intuitive descriptions can be found in [Maital, 1978].

If the third stage results in a high coefficient of alienation, then the data does not fit well into two dimensions, and a different technique is required. On the other hand, if the coefficient of alienation is low, we know that the 2D map indeed captures the salient features of the data.

In the fourth stage of the Co-plot method, p arrows are drawn on the two dimensional Euclidean space obtained in the previous stage. Each variable j is represented by an arrow emerging from the center of gravity of the n points. The direction of each arrow is chosen so that the correlation between the actual values of the variable j and their projections on the arrow is maximized. Therefore, observations with a high value in this variable should be in the part of the space the arrow points to, while observations with a low value in this variable will be on the other side of the map. The length of the arrow is proportional to this correlation.

As a result of this construction, arrows associated with highly correlated variables will point in about the same direction, while arrows associated with uncorrelated variables will tend to be orthogonal. Thus the cosines of angles between these arrows are approximately proportional to the correlations between their associated variables.

The quality of the graph generated by the Co-plot technique is assessed by two types of measures, one for stage 3 and another for stage 4. In stage 3, a single measure – the coefficient of alienation – is used to determine the quality of the two-dimensional map. In stage 4, p separate measures – one for each variable – are given. These are the magnitudes of the p maximal correlations, which measure the goodness of fit of the p regressions. These correlations help in deciding whether to eliminate or add variables: Variables that do not fit into the graphical display, namely, have low correlations, should be removed, because they do not relate well to the principal features of the data as identified by the 2D mapping. The removal of a variable requires a re-computation of the Co-Plot, since it affects the earlier stages (SSA) as well. Note that since each variable's arrow is computed separately, there is no need to fit all the 2^p subsets of variables as in other methods that use a general coefficient of goodness-of-fit. The higher the variable's correlation, the better the variable's arrow represents the common direction and order of the projections of the n points along the axis it is on.

3. THE DATA SET

Traces of real production workloads were available from the Parallel Workloads Archive [Feitelson, 1999]. The full data is given in Table 2. The traces are from ten machines, where the San Diego Paragon is analyzed as two logs – one for 1995 and one for 1996; the two logs identify users differently, and so can't be safely merged.

Table 1: Parallel Computers in our data set

	Log	**Location**	**Machine Type**	**Period**
1	NASA	NASA Ames	iPSC/860	Oct 1993 – Dec 1993
2	PAR5	San Diego Supercomputing Center	Paragon	Dec 1994 – Dec 1995
3	PAR6	San Diego Supercomputing Center	Paragon	Dec 1995 – Dec 1996
4	BLUE	San Diego Supercomputing Center	Blue Horizon	Apr 2000 – Jan 2003
5	SDSP	San Diego Supercomputing Center	IBM SP/2	Apr 1998 – Apr 2000
6	CTC	Cornell Theory Center	IBM SP/2	Jun 1996 – May 1997
7	KTH	Swedish Institute of Technology	IBM SP/2	Sep 1996 – Aug 1997
8	LACM	Los Alamos National Lab	CM-5	Oct 1994 – Sep 1996
9	O2K	Los Alamos National Lab	Origin 2000	Nov 1999 – Apr 2000
10	LLNL	Lawrence Livermore National Lab	Cray T3D	Jun 1996 – Sep 1996
11	OSC	Ohio Supercomputing Center	Linux Cluster	Jan 2000 – Nov 2001

The following variables were measured for each workload:

1. The **number of processors** in the system.
2. **Number of jobs per day**. Since some logs are longer than others, it is required to normalize the total number of jobs in the log by the log's duration, to obtain a comparable statistic.
3. **Runtime Load**, or the percent of available node seconds that were actually allocated to jobs. This is calculated as the sum over all jobs of each job's runtime multiplied by its number of processors, divided by the product of the number of processors in the machine multiplied by the log duration.
4. **CPU Load**, which is the percent of actual CPU work out of the total available CPU time during the log's lifetime. CPU times are the part of runtime in which the job actually processed.
5. **Number of users per thousand jobs**. This measures the number of unique users that used the machine, normalized by the total number of jobs that were executed.
6. **Number of executables per thousand jobs**. Number of unique executables used on the machine, normalized as above. A lower number indicates more repeated requests to run the same executable. This data was missing from some logs, so it couldn't be used in some of our tests.
7. **Runtimes** of jobs. This is represented by the median and 90% interval of the distribution of runtimes.
8. **Degree of parallelism**, i.e. the number of processors used by each job, again represented by the median and 90% interval of the distribution.
9. **Normalized degree of parallelism** (median and 90% interval of the distribution). This gives the number of processors that would be used out of a 128-processor machine. It is calculated as the percent of available processors that jobs used, multiplied by 128. It enables the decoupling of conclusions about the effect of machine size and parallelism.
10. **Total CPU work** over all processors of the job (median and 90% interval).
11. **Inter-arrival times** (median and 90% interval of the distribution).

As shown in [Downey and Feitelson, 1999], the average and standard deviation of some of these fields are extremely unstable due to the very long tail of the involved distributions. Removing the top 0.1% jobs from a workload, for example, could change the average by 5% and the CV by 40%. These findings follow similar ones in [Lazowska, 1977], and mean that the very big jobs must never be removed from workloads as outliers. On the other hand, these extreme values are often the most suspect in terms of possible data errors (what do you do with a job that lasted more than the system is supposed to allow?). Therefore, it is preferable to use order statistics, such as the median and intervals. In our analysis, the 90% interval – the difference between the 95% and 5% percentiles – was used; the 50% interval was also tested, and gave virtually the same results.

Since not all workload traces had all the required fields, missing values were approximated. These are all the assumptions that were made:

1. In the NASA and KTH logs, total CPU work wasn't given and was approximated by the product of runtime and degree of parallelism. In the LLNL log, the opposite was done: The runtime was approximated by the total work divided by parallelism.
2. If one of CPU load and runtime load were missing, the other of the two fields was used. This was done in the NASA, KTH and LLNL workloads.
3. If the submit time of jobs was not known but the time the job started running (after possibly being in a queue) was, the inter-arrival time was based on this start time. This was necessary in NASA, LLNL and the interactive workloads.

Table 2: Data of production workloads

Variable:	Sign:	CTC 1	KTH 2	LACM5 3	O2K 4	LLNL 5	NASA 6	BLUE 7	PAR5 8	PAR6 9	SDSP2 10	OSC 11
Processors in Machine	TN	512	100	1024	2048	256	128	1152	416	416	128	57
Jobs per Day	JD	233.76	83.83	279.32	885.65	183.82	459.59	255.41	209.66	104.45	99.84	119.20
Runtime Load	RL	0.556	0.690	0.735	0.640	0.616	0.467	0.762	0.628	0.611	0.829	0.431
CPU Load	CL	0.464	0.690	0.478	0.391	0.616	0.467	0.627	0.667	0.685	0.734	0.590
Users per KJobs	UJ	8.56	7.51	1.06	2.76	7.18	1.63	1.87	1.28	1.55	5.95	2.63
Executables per KJobs	EJ	155.29	N/A	19.74	N/A	32.88	11.67	N/A	N/A	N/A	896.50	0.01
Runtime Median	Rm	946	847	68	569	36	19	210	25	174	318	383
Runtime Interval	Ri	57226	47861	9063	32243	9143	1170	21738	26535	29924	41583	64089
Processors Median	Pm	2	3	64	15	8	1	8	4	8	2	1
Processors Interval	Pi	37	31	224	127	62	31	128	63	63	48	5
Norm. Procs. Median	Nm	0.5	3.8	8.0	0.9	4.0	1.0	0.9	1.2	2.5	2.0	2.2
Norm. Procs. Interval	Ni	9.3	39.7	28.0	7.9	31.0	31.0	14.2	19.4	19.4	48.0	11.2
CPU Work Median	Cm	559	847	6	21	36	19	30	24	129	264	646
CPU Work Interval	Ci	54794	47861	3658	9624	9143	1170	15990	25788	29640	41079	92838
Inter-Arrival Median	Am	79	192	59	26	119	59	102	73	124	135	64
Inter-Arrival Interval	Ai	1487	3810	1356	357	1660	446	1256	1786	3838	3823	2496

Several other variables were also computed and tested, but removed from the rest of this paper once we found out that they had consistent low correlations. The removed variables were:

1. The un-normalized forms of number of jobs, users, and executables.
2. The duration of each log.
3. The percent of completed jobs, and the percent of cancelled jobs.
4. A measure of the processor allocation flexibility of each machine.
5. A measure of the scheduler's flexibility and sophistication on each machine.
6. The means and standard deviations of the runtime, parallelism, CPU work and inter-arrival time.

4. PRODUCTION WORKLOADS: OBSERVATIONS

Running the Co-Plot algorithm on a given data set is done in several iterations. At first, all observations and variables are used. Then outlier observations and low-correlation variables are removed, and runs on different combinations of the remaining data are done as well. Each run produces a different plot, which enables the analyst to see which observations are stable, and which tend to change. As an example, the two figures in this section and the next use different sets of variables. The conclusions presented in this paper are only ones that proved stable.

Figure 1 includes all the observations – none of them is an extreme outlier. The variables used are those that had the highest correlations, which facilitates the creation of a slightly more accurate 2-D map of the observations (the full set of variables will be analyzed in the next section). The coefficient of alienation of this Co-Plot is 0.10, and the average correlation of variables is 0.85. These are generally considered as excellent goodness-of-fit values [Borg and Groenen, 1997].

The most obvious conclusion that can be drawn about the logs from this map is that they are not clustered. Architecture does not imply proximity – the CTC, KTH and SDSP logs are all IBM SP2 machines of similar size. Neither duration nor the year in which logs were recorded affects the distance between them. Even the two paragon logs are not as close as could be expected, which means that the workload changed over time. The

location has a negligible effect as well – the SDSP, PAR5, PAR6 and BLUE logs are all from the San Diego Supercomputing Center, and are in the same vicinity; however, the Los Alamos CM5 and O2K log are far-off from one another. Also, the San Diego logs range over six years (1995-2000 inclusive), during which many other factors could change as well.

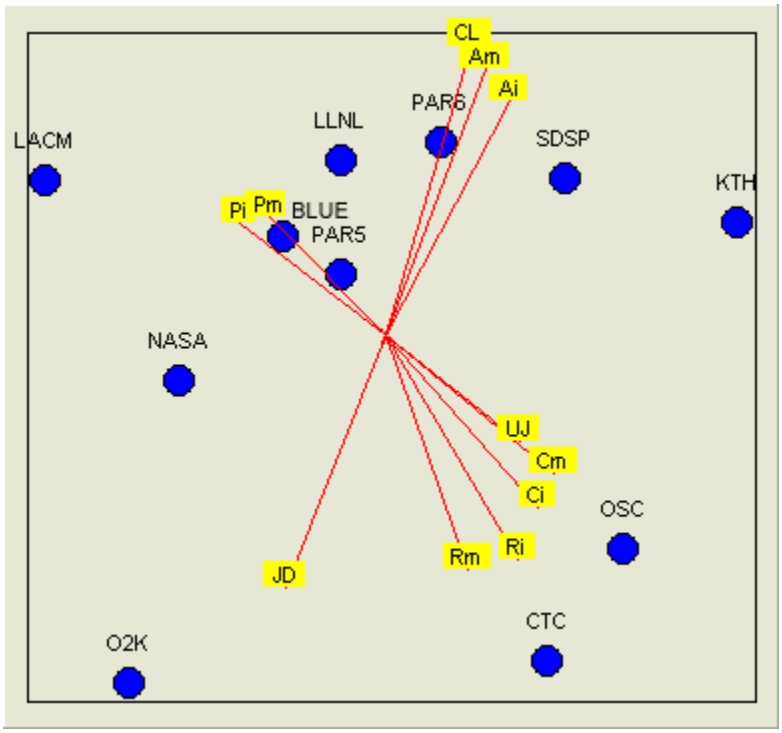

Figure 1: Co-plot output of all production workloads
See Table 2 for variable signs

Since Co-plot analyzes observations and variables together, it is not only possible to see clusters of observations, but also to identify their nature. For example, the OSC and CTC logs are characterized by a relatively high number of different users (UJ), very long runtimes (Rm) and high total CPU work (Cm), but relatively little parallelism (Pm); indeed, the OSC machine had only 57 nodes in total, and CTC has an abnormally high fraction of serial jobs. Such facts are deduced in Co-plot from the arrows: The projection of a point on a variable's arrow should be proportional to its distance from the variable's average, where above average is in the direction of the arrow and vice versa.

In the same manner, the LACM workload (upper left) has a very high degree of parallelism, but below average runtimes. The Blue Horizon has similar characteristics, but not as extreme. The Paragon 1995 log is near-average, but in 1996 the workload there has a high CPU load (CL), accompanied by high inter-arrival times (Am) on one hand and a low number of jobs (JD) on the other. The LLNL and the SDSP logs are similar to the PAR6 log – but the SDSP has more users, higher runtimes, and more CPU work, while the LLNL log has smaller values on all these variables, but higher parallelism. The KTH log has similar attributes as the SDSP log, but taken to the extreme. The O2K log stands out with many jobs, very low (relatively) inter-arrival times, and a low CPU load.

Note that although we can see that the workloads are 'far' from each other, and use terms such as 'high' and 'low' values, this notion of distance is always relative to the other observations in this analysis. This happens because all variables in a Co-Plot analysis must be normalized – otherwise we can't compare relations between them – and

means that we should beware of attaching real distance to Co-plot's output. On the other hand, this observation map, especially because we have no outliers, defines an intuitive notion of the "parallel workloads space". Since our sample includes a variety of architectures, locations, and time frames, the borders of this spaces – as defined by the minimal and maximal values in table 2 – describe what a "normal" parallel workload is like.

5. PRODUCTION WORKLOADS: VARIABLES

The following figure adds to the one from the previous section four more variables: The total number of nodes (TN), the runtime load (RL), and the median and interval of the normalized number of processors (Nm and Ni). The coefficient of alienation is 0.10, and the average of correlations is 0.80. Three variables had correlations below 0.70 – the median of normalized parallelism (Nm), the runtime load (RL), and the normalized number of users (UJ). Removing these variables had no effect on the Co-Plot, so they do not interfere; but conclusions about them should be made with caution.

The map tells a lot about the types of distributions that should be used for modeling workloads. The runtimes, the degree of parallelism, and total CPU work exhibit a high positive correlation between the median and interval of the distribution. This means that systems where jobs have higher runtimes and parallelism also exhibit a more varied (or more probably, skewed) stream of requests. In some cases this may occur because common administrative tools, such as limiting the maximal runtime, affect both the median and interval of the observed runtime in the same way.

Inter-arrival times and the normalized parallelism, however, may require a different treatment. Although their median and interval are positively correlated, the correlation is not full. Repeated analyses of these variables shows, that in some plots their correlation is less than that shown in Figure 2, although it's always positive. This is important for building models – all models include inter-arrival times, and every model that takes the number of processors of the target machine as a parameter needs to take this parameter's effect into account.

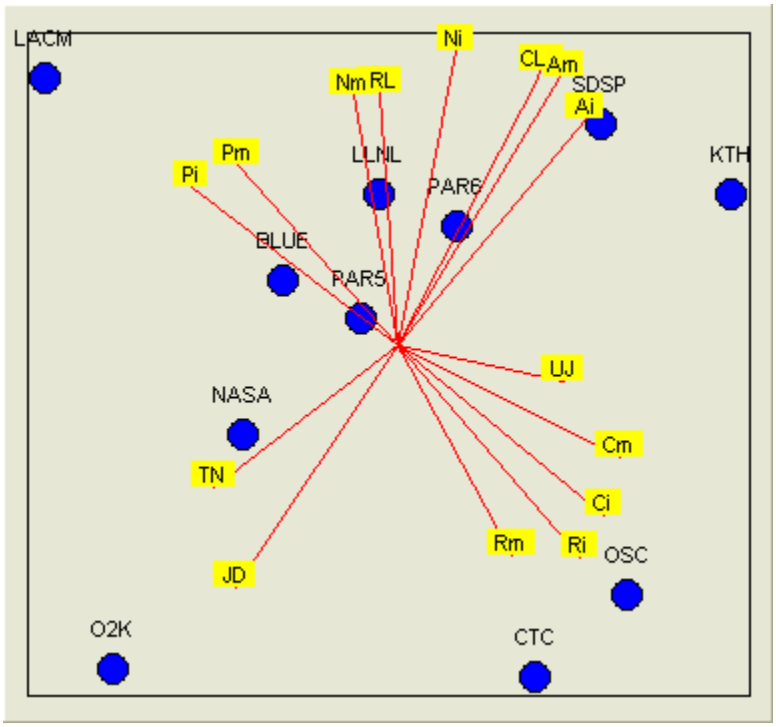

Figure 2: Co-plot output of all production workloads using all variables

Perhaps more important than analyzing individual variables, the Co-Plot analysis enables us to understand the correlations between all the variables at once, and to define clusters of variables and the relations between them. In order to do so, Figure 3 includes a subset of the arrows from Figure 2 (those that appear in Figure 1), plus two imaginary perpendicular dotted axes, to make our explanation easier.

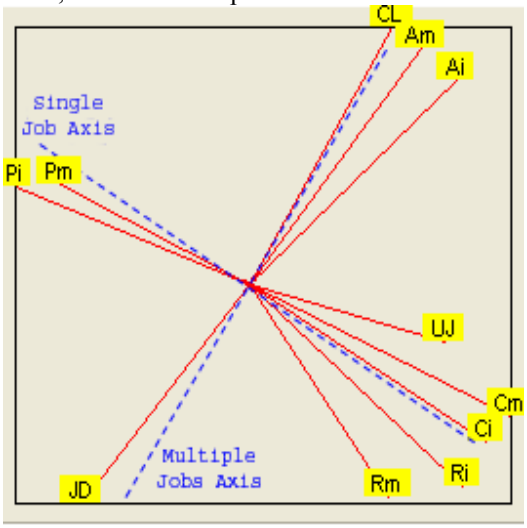

Figure 3: Variable arrows from Figure 1

The two dotted axes are perpendicular – meaning not correlated in the plot. All the variables along each axis are highly correlated – for example, the CPU load (CL) and normalized number of jobs (JD) are very highly correlated, although that correlation is negative. This correlation does not have to be linear – note the definition of distance and alienation from section 2 – but it means that the variables grow together, by some relationship. The two axes were drawn since most of the variable arrows fall very close to one of them, and they define the two main variable clusters of this data set:

1. **The Multiple Jobs Axis** – includes the number of jobs per day (JD), the median and interval of the inter-arrival times (Am and Ai), and the CPU load (CL). The interval of the normalized degree of parallelism (Ni) and the total number of nodes in the machine (TN) are also members of this axis (this conclusion is drawn from other runs of the analysis as well), although their correlation with the axis is not full.
2. **The Single Job Axis** – includes the median and interval of runtimes (Rm and Ri), the median and interval of total CPU work (Cm and Ci), and the median and interval of parallelism (Pm and Pi). The normalized number of users (UJ) also belongs to this cluster; its correlation to the axis is not full, but actually stronger in most runs than it appears in Figure 3.

The runtime load (RL) and the median of normalized parallelism (Nm) seem to be between the two axes. However, as mentioned above, these two variables have a low correlation in this Co-Plot, so it's hard to conclude anything about them. Since the runtime load is an important parameter in many studies in this area, it should be noted that the CPU load – which exhibits high correlation in this Co-Plot – can possibly be used to represent it, as the two variables are highly correlated, in the usual linear sense of correlation.

The names of the two axes reflect the fact that one axis gathers the variables that define a single job, while the other axis gathers variables that define the relations between jobs. Although we are making a generalization here, as some important variables are not fully correlated with either axis, we can still say that these two axes define the two-dimensional workload space. The low coefficient of alienation (0.10 and below) achieved in

our analyses means that the workloads space can be hosted in a 2-D space without much loss of information. The two axes we identified clarify what each dimension stands for.

The Single Job Axis provides us with the following information. Logs with high runtimes (on average, relative to other logs) also have high total CPU work, and in contrast have low parallelism. This is true for both the median and the interval of the three distributions involved. Also, logs with high runtimes have more users per job. This is very important, because the number of users is one of the only variables that can be estimated in advance, before a system is built. The strong correlation between the number of users and the medians and intervals of runtimes or degrees of parallelism means that these variables too can now be predicted in advance. This is why the number of users was normalized by the number of jobs, and not by the job's duration (like the JD variable): this was required in order to decouple it – i.e. make it uncorrelated – to the Multiple Jobs Axis. The number-of-users-per-day (UD) variable was also computed: it enters the Co-Plot with a high correlation, and is usually drawn between the two axes.

The Multiple Jobs Axis also provides valuable information about its variables. First, logs with many jobs per day (JD) are usually found on machines with a lot of processors (TN). The TN variable is not fully correlated to this axis, since – as expected – it has a small positive correlation with parallelism (it is expected that machines with more nodes will have higher parallelism, although this effect seems to be weak). As expected, logs with many jobs per day have very low inter-arrival times (more jobs over the same time frame must be temporally closer to one another). On the other hand, they also have very low CPU loads and runtime loads, which is counterintuitive.

The median and interval of the normalized number of processors are between axes. The median of the normalized parallelism has an obvious positive correlation to the (un-normalized) parallelism, and is uncorrelated to the total number of nodes in the machine (as expected by its definition). The interval is somewhat closer to the Multiple Jobs Axis, but the correlation is far from full, so it's safest not to conclude anything about it.

One should be careful not to misinterpret these findings – note that they relate to whole workloads, not jobs. For example, the negative correlation between the degree of parallelism and runtime means that systems with high average parallelism exhibit lower runtimes, not that jobs that use many processors are shorter (the opposite of which was indicated in [Feitelson and Rudolph, 1998; Downey and Feitelson, 1999). The reason for this may be, for example, that systems with more processors tend to enforce tighter runtime limits on jobs. This would fit in with the conjecture that systems with fewer processors may try to compensate for this by offering more flexible policies.

6. FLURRIES

A flurry [Tsafrir and Feitelson, 2006] is a burst of very high activity by a single user. In contrast to normal active work, a flurry is an extreme case in which the load created by a user, over a short period of time, is orders of magnitude higher than usual, and affects the entire workload in a significant way. Sometimes the runtimes or CPU work used in a flurry is above the declared limits of the system, which makes them even more questionable. The following figure shows two examples of this phenomenon, both of which caused by an unusually high number of jobs by a user over a short period of time. On the right, from the SDSC SP2 log, a single user (out of 428 users) created a stream of jobs in one week, that is about seven times higher than the maximum number of jobs per week created by all other users, anytime in that log. On the left, from the LANL CM-5 log, 3 out of 213 users create similar short-term streams of a very large number of jobs – five to six times the maximum of all other users during the log.

As argued in [Feitelson and Tsafrir, 2006; Tsafrir and Feitelson, 2006], a workload model should in general not include flurries, since they are non-representative behavior, and

should be modeled separately. Therefore, like any other case of outliers in a statistical data set, it is recommended to remove flurries from a data set before analyzing or modeling it.

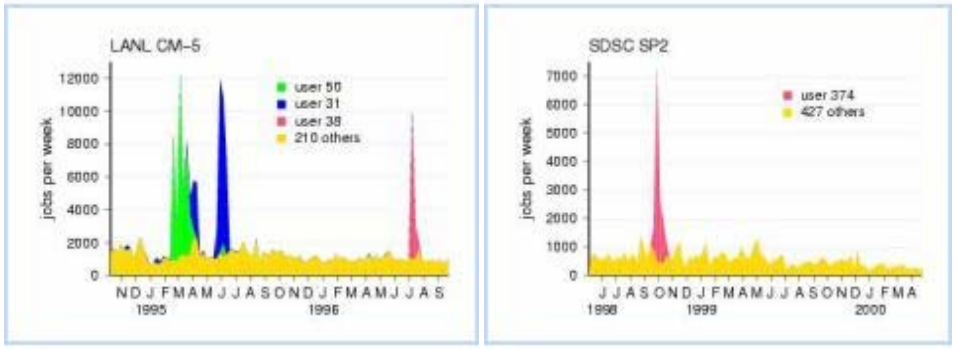

Figure 4: Jobs-per-week flurries in the LANL CM-5 and SDSC SP2 logs

Table 3: Data of flurry-free production workloads

Variable:	Sign:	CTC 1	OSC 2	LACM5 3	O2K 4	SDSP2 5	NASA 6	BLUE 7	PAR5 8	PAR6 9
Processors in Machine	TN	512	57	1024	2048	128	128	1152	416	416
Jobs per Day	JD	227.63	119.18	169.29	885.65	81.13	198.34	248.14	147.19	86.68
Runtime Load	RL	0.556	0.428	0.734	0.640	0.827	0.466	0.762	0.627	0.611
CPU Load	CL	0.464	0.583	0.477	0.391	0.732	0.466	0.627	0.666	0.685
Users per KJobs	UJ	8.79	2.63	1.75	2.76	7.32	3.78	1.92	1.82	1.87
Executables per KJobs	EJ	159.48	0.01	32.58	N/A	1103.21	27.03	N/A	N/A	N/A
Runtime Median	Rm	1114	382	414	569	237	86	219	38	207
Runtime Interval	Ri	57562	63952	11202	32243	47463	3716	22688	30478	31048
Processors Median	Pm	3	1	32	15	4	4	8	8	8
Processors Interval	Pi	39	5	480	127	64	63	128	63	63
Norm. Procs. Median	Nm	0.8	2.3	4.0	0.9	4.0	4.0	0.9	2.5	2.5
Norm. Procs. Interval	Ni	9.8	11.2	60.0	7.9	64.0	63.0	14.2	19.4	19.4
CPU Work Median	Cm	669	645	64	21	170	86	35	39	156
CPU Work Interval	Ci	55184	92580	7826	9624	47102	3716	16545	30305	30765
Inter-Arrival Median	Am	85	64	176	26	201	90	104	113	212
Inter-Arrival Interval	Ai	1527	2496	2048	357	4738	1217	1292	2467	4363

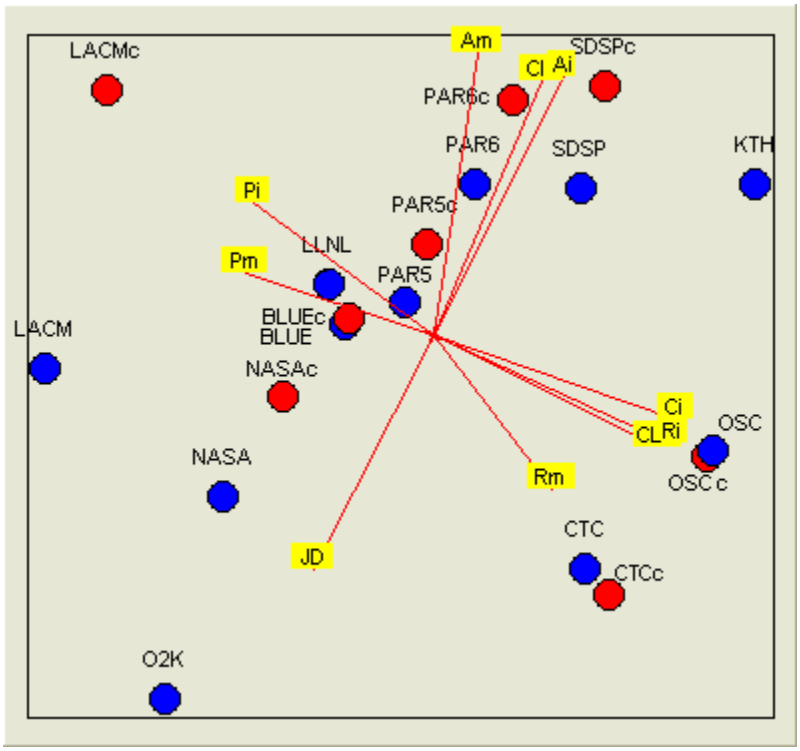

Figure 5: Co-Plot of original and clean production workloads

To examine the effect of non-representative behavior on our previous results, all variables were re-computed for the "clean" versions[2] of each log we used, and the Co-Plot analysis was repeated with the recomputed variables. Of all the logs we used, only KTH, LLNL, and O2K do not have a separate clean version. The NASA log didn't have flurries, but was affected by a large number of system jobs which were not user-generated behavior. The SDSC Paragon logs had both flurries and repetitive system scripts. All other logs had at least one flurry.

Table 3 summarizes the data for all the cleaned affected logs. Figure 5 includes all the original production logs, all the cleaned logs, and all the variables used in Figure 1 except UJ (which was removed due to low correlation). The coefficient of alienation is 0.09, and the average correlation is 0.83.

First, it is clear that cleaning the logs had no affect on our previous findings. The observations map and the variables map remain the same. The same results are also obtained when only the cleaned logs are used to build the plot – the cleaned logs appear very close to where the original logs were placed, and their interactions with all the variables remain essentially the same. This result was stable across many other runs and configurations, with different sets of variables, or mixed sets of cleaned and original logs.

The cleaned logs are very close to their corresponding original logs in almost all cases. In the NASA, Paragon and San Diego SP2 logs, the cleaned logs move along the Multiple Jobs Axis (up and slightly to the right, opposite the direction of the jobs-per-day variable), because cleaning these logs involved removing many thousands of jobs. However, all the other features of the logs were retained. The Los Alamos CM5 cleaned log moved the farthest from its original – in this log, 79,329 out of its 201,387 jobs were part of three large flurries. So relative to the fact that almost 40% of the log's jobs were

[2] The parallel workloads archive, from which the logs were taken, includes both raw and cleaned versions.

removed, its move on the observations map is not dramatic (the log remained at the same part of the map). This may be explained by the fact that our variables are relatively stable and resistant to outliers, as they are based on the median and range of values, not on the mean and standard deviation [Lazowska, 1977; Downey and Feitelson, 1999].

7. WORKLOADS OVER TIME

Logs of workloads from production work in recent years are the main source of data for evaluations of new system designs, which may be used for production work in the near future. But this practice is of questionable merit, since workload patterns may change with time [Hotovy, 1996]: users learn the system as time passes and adjust their behavior to maximize their benefits, administrators fine-tune the system continuously, and the dominant projects on machines may change every few months.

Co-Plot allows us to test the degree to which workloads change with time, by mapping several consecutive periods of logged work on the same machine, and comparing them with each other and with other workloads. If recent workloads are indeed good predictors of the near future, we would expect the workloads from consecutive periods on the same machine to be mapped close to each other. Three workloads in our sample were long enough to test this – the LANL CM5 log, the SDSC SP2 log, and the Blue Horizon log were each divided into four equal periods.

The resulting Co-Plot, including these twelve quarters and the original three full workloads, is shown in Figure 6. The variables used are the same as before, except the median of runtimes, which was removed due to a low correlation. The coefficient of alienation is 0.06, and the average correlation is 0.88. The data for the LACM and SDSP logs, which exhibit the highest variability, is given in Table 4.

Table 4: Data production workloads separated to quarters

	LANL CM5				SDSC SP2			
	Oct4-Mar5	Apr5-Sep5	Oct5-Mar6	Apr6-Sep6	Apr8-Sep8	Oct8-Mar9	Apr9-Sep9	Oct6-Apr0
Jobs per Day	395.91	347.35	142.02	206.34	161.31	105.80	76.70	51.12
Runtime Load	0.719	0.817	0.679	0.724	0.737	0.810	0.883	0.888
CPU Load	0.436	0.522	0.468	0.485	0.604	0.720	0.780	0.834
Users per KJobs	2.22	2.93	7.99	5.73	14.54	21.97	25.46	45.49
Executables per KJobs	13.55	31.63	117.26	106.90	2110.40	3312.17	2749.84	6020.51
Runtime Median	62	65	629	84	1061	115	206	231
Runtime Interval	7028	7343	10961	11106	32629	46288	40300	45043
Processors Median	64	32	64	128	1	4	4	8
Processors Interval	224	224	480	480	32	56	64	64
Norm. Procs. Median	8.0	4.0	8.0	16.0	1.0	4.0	4.0	8.0
Norm. Procs. Interval	28.0	28.0	60.0	60.0	32.0	56.0	64.0	64.0
CPU Work Median	3	7	122	8	1075	45	140	157
CPU Work Interval	2923	3491	8595	7689	30724	46640	40755	43191
Inter-Arrival Median	39	44	228	92	72	142	220	360
Inter-Arrival Interval	833	992	2412	1888	2242	3422	4860	7868

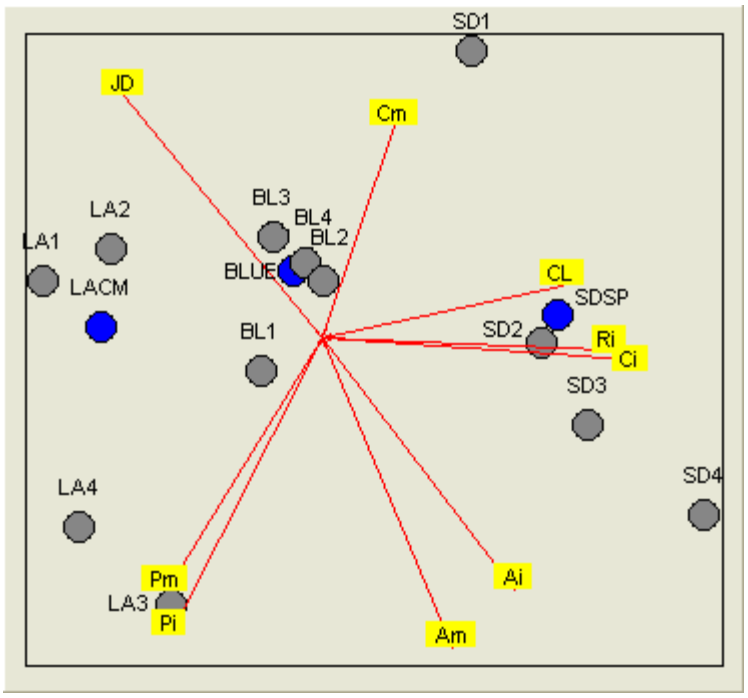

Figure 6: Co-Plot of three workloads divided to quarters

As the plot shows, logs can change significantly over time. The Blue Horizon was the most stable, with the possible exception of the first seven months. This may be due to a slow start-up phase of the machine, or because only the first four months of the log include interactive jobs that were not submitted through LoadLeveler. The administrative change to the used queues that was performed towards the end of 2001 does not seem to have changed the workload in a noticeable way.

In contrast, the LACM and SDSP logs show significant changes in workload between periods. Regarding the LACM log, a query with the person who provided the log from LANL indeed revealed that at the end of 1995 there was a significant change of policy regarding the CM-5. It approached the end of its life for grand challenge jobs, and most users were moved to a newer machine; only a few groups continued to use the CM-5 during 1996, mainly to complete projects that were hard to port.

In the SDSC SP2, the first period of six months is very different from the rest of the log. This may have been caused because the log collection began before the machine was fully operational, or before it was in wide use. From their location on the Co-Plot, it seems that in each subsequent period this machine experienced higher inter-arrival times and less jobs, while maintaining roughly the same CPU load, indicating that users submitted fewer but larger jobs.

Apart from the evolution of each specific log, it is instructive to look at the entire map as well. The three full logs retain their place on the map as they were in the previous sections. Most variables remain the same as well – except the CL, Ci and Ri variables which moved close to one another, between the two axes. Another change in variables is the low correlation of the median of runtimes. This should not be taken to reflect on our previous findings, since this Co-Plot is based on only three logs, and so is highly biased towards their specific features.

The above analysis indicates that workloads indeed change over time. It is dangerous to rely on the past to predict a future workload – even for the very same machine. It is therefore recommended to perform such an analysis for each new workload log that is

obtained. This will reveal whether the log is homogeneous, and whether it contains time intervals in which work on the logged machine had unusual patterns.

8. INTERACTIVE VERSUS BATCH WORKLOADS

The work people do on parallel computers is usually divided into interactive jobs – very short test runs in which the user sits by the computer and waits for the result – and batch jobs, which may take hours or even days to complete. In two of the logs in our sample – the LANL CM5 and the SDSC Blue Horizon – we know exactly the type of each job, since different jobs were submitted using different queues, and these queues are well documented. Since it is obvious that interactive and batch jobs are very different from one another, it is interesting to plot them together, and learn how they are characterized, and how they should be modeled. Table 5 summarizes the data gathered for each job type in each log, for both the original and flurry-cleaned versions.

The Co-Plot in Figure 7 is the result of running the above eight workloads, together with the original and cleaned versions of the LACM and BLUE logs. All the other logs were not used in this analysis, which therefore focuses on batch versus interactive logs only. This was reflected in the variables map, which is entirely new, both in terms of the variables themselves and in terms of the correlations between them. The Co-Plot has a coefficient of alienation of 0.06, and an average correlation of 0.92. The variables of jobs per day, users per job, and both the median and interval of inter-arrival times had low correlations, and were removed. The users-per-job variable, when added to the analysis, is drawn in the direct opposite direction of the CPU load (CL) variable, which means that they are negatively correlated (i.e. the interactive log portions have more users, and lower loads).

Table 5: Interactive and Batch portions of workloads

	Original Logs				Cleaned Logs			
	LACM int	LACM bat	BLUE int	BLUE bat	LACM int	LACM bat	BLUE int	BLUE bat
Jobs per Day	109.77	153.63	96.21	176.64	109.77	153.63	93.27	171.78
Runtime Load	0.017	0.734	0.008	0.756	0.017	0.734	0.008	0.756
CPU Load	0.003	0.475	0.004	0.624	0.003	0.475	0.004	0.624
Users per KJobs	2.64	1.92	6.06	2.70	2.64	1.92	6.25	2.78
Executables per KJobs	50.17	35.90	N/A	N/A	50.17	35.90	N/A	N/A
Runtime Median	57	554	153	355	57	554	146	393
Runtime Interval	268	11465	1688	34212	268	11465	1706	35408
Processors Median	32	32	8	8	32	32	8	8
Processors Interval	96	480	64	256	96	480	64	256
Norm. Procs. Median	4.0	4.0	0.9	0.9	4.0	4.0	0.9	0.9
Norm. Procs. Interval	12.0	60.0	7.1	28.4	12.0	60.0	7.1	28.4
CPU Work Median	3	98	6	81	3	98	6	98
CPU Work Interval	54	8465	876	26108	54	8465	881	26935
Inter-Arrival Median	16	207	188	126	16	207	196	135
Inter-Arrival Interval	281	2225	2117	1963	281	2225	2184	2010

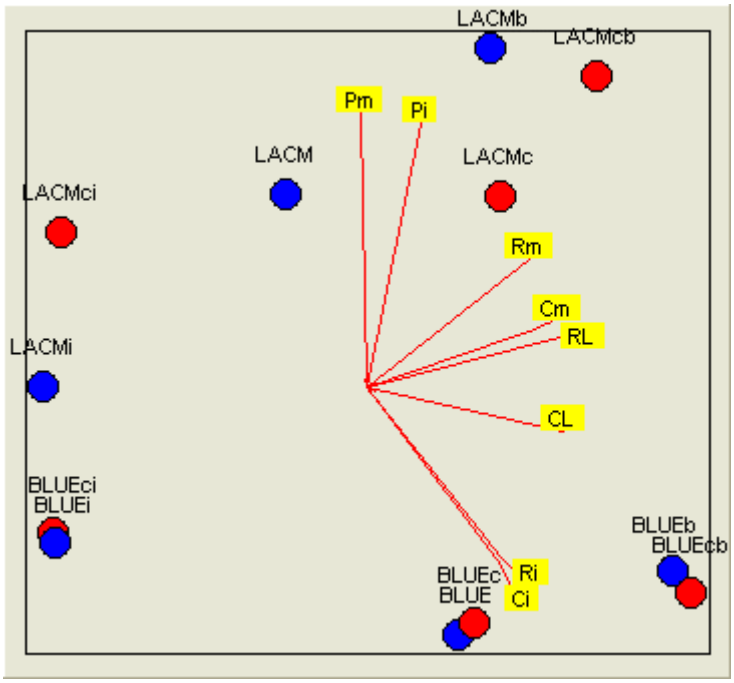

Figure 7: Co-Plot of interactive versus batch workloads

The variables that remained have created a high-quality Co-Plot, in terms of goodness-of-fit. The observations map clearly separates the batch logs from the interactive logs, as expected. This also happens when the batch and interactive jobs are analyzed together with all the other logs that were used in figures 1 and 2: the interactive logs cluster together, and are so separated from the other (full and batch) logs that they seem to be outliers. Note that in this analysis, the full logs are mapped between their batch and interactive components, as may be expected. The full logs are closer to their batch logs in both cases, since the batch jobs are the majority (in terms of number of jobs and total work) in both cases.

The variables map shows, again as expected, that the batch jobs are characterized by higher runtimes and total CPU work, as well as much higher loads (both runtime load and CPU load). The intervals of the runtime and CPU work are nearly orthogonal to their medians, rather than being highly correlated as before; this is probably because the intervals are higher in BLUE than in LANL, and the fact that only these two logs are used here. Likewise, parallelism is not a differentiator between batch and interactive logs, but it is a differentiator between LANL (higher parallelism) and BLUE (lower).

The cleaned logs are very close to their origins, and do not affect any of the above results – using only the original logs or only the cleaned logs causes virtually no change in the resulting plot.

9. SYNTHETIC WORKLOADS

Having performed a detailed examination of the production logs, we now turn to inspect the statistical workload models that are currently available. As these models are each based on data from one or more logs, we would expect a joint Co-Plot of the logs and models to reflect this: each model should be close to the log on which it is based. The results are that this is not always the case, in part because a model may be based on more than one log. However, the models are not too far off from the logs, and none is an outrageous outlier.

The eight synthetic models available are the following. The first model was proposed by Feitelson in [1996] (shown as FE6). This model is based on observations from several workload logs. Its main features are the hand-tailored distribution of job sizes (i.e. the number of processors used by each job), which emphasizes small jobs and powers of two, a correlation between job size and running time, and the repetition of job executions. In principle such repetitions should reflect feedback from the scheduler, as jobs are assumed to be re-submitted only after the previous execution terminates. Here we deal with a pure model, so we assume they run immediately and are resubmitted after their running time. The second model (FE7) is a modification from '97 [Feitelson and Jette, 1997].

The model by Downey (DOW) is based mainly on an analysis of the SDSC Paragon log [Downey, 1997 & 1997b]. It uses a novel log-uniform distribution to model service times (that is, the total computation time across all nodes) and average parallelism. This is supposed to be used to derive the actual runtime based on the number of processors allocated by the scheduler. Again, as we are dealing with a pure model here, we instead use the average parallelism as the number of processors, and divide the service time by this number to derive the running time.

Jann's model (JAN) is based on a careful analysis of the CTC SP2 workload [Jann et al., 1997]. Both the running time and inter-arrival times are modeled using hyper Erlang distributions of common order. A separate distribution is used for different ranges of number of processors, with the parameters derived by matching the first three moments of the empirical distribution from the log.

The model by Lublin [Lublin and Feitelson, 2003] (LUB) is based on a statistical analysis of four logs. It includes a model of the number of processors used which emphasizes powers of two, a model of running times that correlates with the number of processors used by each job, and a model of inter-arrival times. While superficially similar to the Feitelson models, Lublin based the choice of distributions and their parameters on better statistical procedures in order to achieve a better representation of the original data.

The last and most recent model is by Cirne and Berman [2001]. This is a comprehensive model for generating moldable jobs, based on the analysis of four SP2 logs. It is composed of two parts: A model for generating a stream of rigid jobs; and a model for turning rigid jobs into moldable ones. The model also addresses the issues of requested times for a job, and the possibility of job cancellation. Here we test the basic features of the model – the generation of arrival time, runtime, and parallelism for each job. The model takes into account workday cycles, and its inter-arrivals pattern can be adjusted to match each of the logs used to build it (we tested against the KTH, CTC, and SDSP2 versions of the model, denoted CBk, CBc, and CBs, respectively).

In order to produce test data for the models, the following methodology was used. Using code available for each model about 200,000 jobs from each model were generated. From these jobs, all the per-log variables that we analyzed so far were computed. We also experimented with generating additional job sets for each model, in order to test the stability of the results. The result was that the Co-Plot did not change. This is expected, since all the models use stationary distributions, and therefore tend to be more stable than real workloads.

Figure 8 is the result of running Co-Plot on all production logs and on all the models. The coefficient of alienation of this plot is 0.09, and the average correlation of the variables is 0.85. The variables used in this plot are the medians and intervals of the runtimes, CPU work, and normalized parallelism, and the median of inter-arrival times. The runtime and CPU load variables are in the plot, although their correlations were lower than in the previous ones – 0.77 and 0.79 respectively. The un-normalized parallelism (Pm and Pi) variables had low correlations, and were removed. The number of jobs per day had a very low correlation and was removed – the models were not designed with this variable in mind, and indeed don't model it well. For example, the

Downey and Feitelson models produce more than ten times the average number of jobs per day compared to the production logs. A similar phenomenon occurs with the interval of inter-arrival times, which had a very low correlation – in some of the models this value is an order of magnitude lower than in the production workloads.

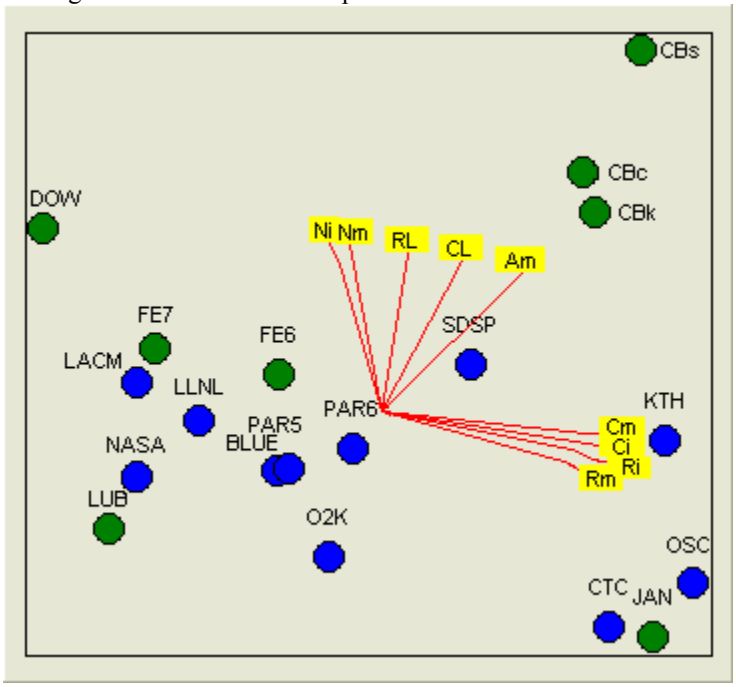

Figure 8: Co-Plot of logs versus models

The Cirne and Berman models seem to be outliers in Figure 8 – they are clustered far from the rest of the logs and models. Using only one of the three variations of this model did not change this result. It seems that although three of the four logs used to build this model are used in our analysis too, the distributions weren't designed to match the median and interval statistics on which we rely here.

Jann's model is very close to CTC, which is expected, since it was designed specifically to match the CTC workload, using a sophisticated framework which matched the first three moments of the major distributions. The OSC log is similar to the CTC log, so it is also well modeled by Jann's model.

Feitelson's models and Lublin's model are close to a group of several production workloads – LLNL, NASA, LACM, BLUE, and the Paragon. Lublin's model is placed closer to the center of this group of logs in other runs. Downey's model is somewhat separated from the pack in the plot, although it is not an extreme outlier.

The KTH, SDSP, and O2K logs have no model close to them. The LACM log also separates from its group in some of the runs. This reflects the general problem of finding an appropriate model for a given machine. Since logs are not clustered according to location, machine type, size, or any other such criteria, it is impossible to match one of the existing models to a new machine in order to predict its workload in advance.

The above experiment was repeated with the addition of the batch and interactive workloads, to see how they are modeled. The batch workload of LACM5 was an outlier – no model (or other log) was close to it. The Blue Horizon batch workload was not an outlier – it was drawn mid-way between where the KTH and PAR6 logs stand in figure 8 – but still no model was very close to it. Both interactive logs were clustered very close to each other, and distant from the other logs. With the exception of Lublin's model, none of the existing models is designed specifically for interactive logs, though some may

implicitly model only batch jobs, and this shows in the analysis. Since the interactive portions of the logs in our data set are very similar, a simple non-parametric model should suffice to model them. The interactive version of Lublin's model does this well – it was plotted with good proximity to the interactive logs.

Another interesting question is how the models compare to the cleaned logs – the same workloads but without flurries and other anomalies. Intuitively, cleaned logs should provide a better match to the models, since the distributions that describe them are better behaved. Figure 9 is the result of running Co-Plot on all the production logs, using the cleaned versions whenever possible, and using all models except the Cirne and Berman models. The coefficient of alienation is 0.09, and the average of correlations is 0.84.

The Cirne and Berman models were removed because they were extreme outliers – about half of the plot's area separated them from the rest of the observations. After removing them (and the two load variables, although this didn't change the plot much), both the resulting observations map and variables map are close to our previous analysis. Lublin's model appears close to the center, and this repeats in other runs of Co-Plot as well.

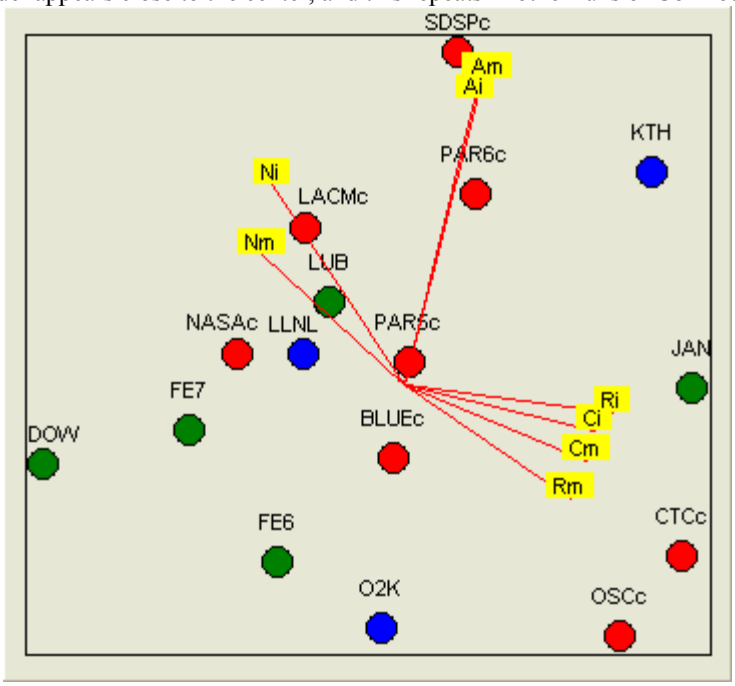

Fig. 9. Co-Plot of cleaned logs versus models

As this model is usually near the center of the map it seems to be the best choice for someone wishing to generate an "average" sample workload. However, this model is only a close fit to 3-4 of our production logs.

A quick look at the variable arrows of Figures 8 and 9 is also worthy. It is almost the same as that of figure 2, apart from a slight shift of the Am variable from the Multiple Jobs Axis in Figure 8, and a shift of the normalized parallelism variables towards the Single Jobs Axis in Figure 9. This is good news: At least for variables that remained in the plot, the synthetic workload models do not distort the picture by assuming wildly incorrect distributions or correlations.

10. IMPLICATIONS: BUILDING A PARAMETRIC WORKLOAD META-MODEL

Having concluded our data analysis using the Co-Plot method, the next three sections explore three major implications of the presented results. This both translates our findings into more concrete tools for further research, as well as highlights the usefulness of the

results that the Co-Plot method provides. The three issues are the construction of a parametric workload meta-model; considerations for manipulating the load of given models; and guidelines for extending the parametric model into a full model that improves upon current models.

One of the first results of our analysis was the fact that real workloads are rather different from one another, so a single model cannot represent all systems. This is supported by Figure 1, which compares the production workloads. Except maybe the LLNL, Blue Horizon, and Paragon workloads, the other logs are as far from one another as possible. As any specific model will also occupy a given place in the "workloads space" represented by the Co-Plot, it too will be far from most of the logs.

The good news is that another result of the Co-Plot analysis was the identification of variables that can be used to parameterize a workload model. Figures 2 and 3 show that the analyzed workloads fit well into a two-dimensional space, and that each can be placed in this space by defining its position on two axes: The Single Job Axis, and the Multiple Jobs Axis. The variables on each axis are highly correlated with each other, so that once one is given, the others can be well approximated. Moreover, the two axes are largely independent, so there is no need for multiple regression, for example. It is therefore clear that a parametric workload model should take one parameter from each axis.

The main criterion for selecting a representative parameter from each axis is that this should be a variable that can be estimated for a future system, before that system is built. This is required because a model of a system's workload is most useful before it is built and deployed. Specifically, we found that the Multiple Jobs axis can be represented by the number of jobs per day (JD), or to a lesser degree (but trivial to know in advance) the total number of nodes in the machine. The Single Jobs Axis does not contain an easy-to-predict variable, except for the users per job parameter, to which it is only weakly correlated. Experimenting with different possible lead variables led to the conclusion that

Table 6: Fitting equations and coefficients to the *JD* and *Cm* variables

Equation	R^2
$Ci = 14052 \times \ln(Cm) - 31068$	0.75
$Rm = 0.8346 \times Cm + 131.0012$	0.60
$Ri = 10423 \times \ln(Cm) - 14441$	0.76
$Pm = -1.531 \times \ln(Cm) + 12.261$	0.28
$Pi = 74.69 \times e^{-0.002 \times Cm}$	0.44
$UJ = 0.7633 \times Cm^{0.3073}$	0.45
$CL = 2.2024 \times JD^{-0.2534}$	0.75
$Am = 2764.4 \times JD^{-0.6582}$	0.74
$Ai = 580400 \times JD^{-1.1079}$	0.96

the highest correlations are achieved when the Single Jobs Axis is represented by the median of CPU work (Cm). Table 6 summarizes the equations and R^2 values for fitting the other main variables to the two selected representative variables. These values were generated based on all the data from the production workloads (Table 2).

Note that the correlation is not always linear, as this is not the type of correlation that Co-Plot uses (see the definition of distance and alienation from Section 2). If variables are plotted along the same axis, it means that the variables grow together, by some relationship. Limiting this relationship to be linear would make Co-Plot far less useful in practice, since far fewer datasets would fit into a two-dimensional space under such a limitation. The imperfection of the R^2 values in Table 6 are caused by a combination of the imperfect plot they are based upon (a coefficient of alienation of 0.10 and not 0.0), and our choice to use only simple fitting equations, to keep the model simple and avoid over-fitting.

In order to test whether this meta-model is good enough for practical use, we test our ability to generate parameter combinations that will locate a model in a specific place in the Co-Plot map. For this we generated four synthetic observations, using combinations of two pairs of "low" and "high" values of the two proposed parameters, as shown in Table 7.

Table 7: Fitting equations and coefficients to the *JD* and *Cm* variables

Observation	JD	Cm
Gen1	100	25
Gen2	300	25
Gen3	100	600
Gen4	300	600

The resulting Co-Plot is shown in Figure 10, using the same variable set as in Figure 1, and the same dataset augmented by Gen1 to Gen4. The coefficient of alienation is 0.11 and the average of correlations is 0.85. The results are very positive in two respects. First, the Figure itself is not affected at all by the addition of the four new observations: All the observations and all the variables are exactly where they were in Figure 1, and retain all their relative positions and directions.

Second, the four synthetic observations are each placed in a different quadrant of the plot, as marked by the center and directions of the two axes. Moreover, each is placed in the quarter it is expected to belong to, as dictated by Table 6. We therefore conclude that the meta-model specified by Table 6 is a good representation of the current known world of production parallel workloads.

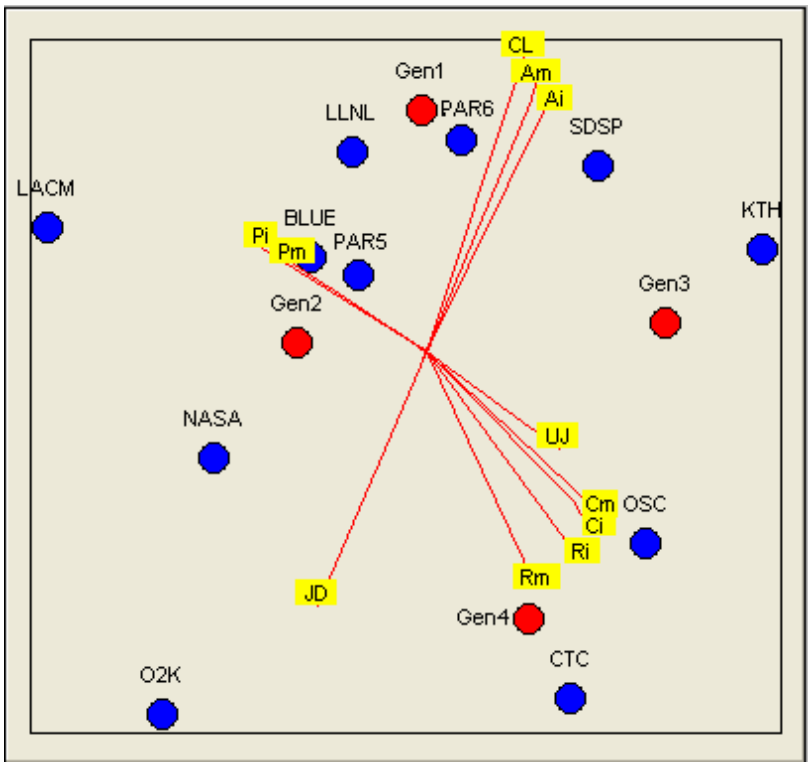

Figure 10: Verifying the Parametric JD/Cm Meta-Model via Synthetic Observations

11. IMPLICATIONS: LOAD MANIPULATION

Testing a new scheduling algorithm or comparing algorithms is often done by simulating how they schedule the same workload, over a range of loads. A graph of performance versus load is then drawn and analyzed. Such a test is usually carried out by taking a single workload, and manipulating its load [Majumdar et al., 1988; Lo et al., 1998].

There are three basic ways to raise a workload's load: Lowering the inter-arrival time, raising the runtimes, and raising the degree of parallelism. The most common [Majumdar et al., 1988; Lo et al., 1998] technique is to expand or condense the distribution of one of these three fields by a constant factor. Note that by doing so the median and interval (any interval) is also multiplied by the same factor. The choice of which field to alter in order to change a system's load should depend on the correlations between the runtime load and these three variables. We would choose lowering inter-arrival times if it were negatively correlated with load, and raising runtimes or parallelism if they were positively correlated with it. By doing so, we minimize the side effects of raising the load on other features of the workload.

Regrettably, this logical criterion does not seem to match the data. First, from Figure 2 it is clear that systems with a higher average runtime load (RL) have a higher inter-arrival time median (Am), not a lower one. Likewise, the median of jobs runtimes (Rm) is negatively correlated to the load, rather than being positively correlated. As for the degree of parallelism, while it is indeed correlated with the runtime load, this is only a weak correlation. In addition, raising parallelism is almost never possible, since it breaks the dominance of powers-of-2 requests, which completely alters the behavior of many algorithms for which the workloads are needed (schedulers are the most obvious example).

These correlations mean that a correct way to raise a system's load would end up with somewhat higher inter-arrival times, a somewhat higher degree of parallelism, and shorter runtimes. None of the three simplistic ways to alter the load satisfy these requirements; rather, they contradict them. Thus varying a given workload model's load is not as simple as it looks.

A possible solution to this problem is based on the parametric meta-model presented in the previous section. Although the runtime load is not fully correlated to either axis, the CPU load (CL) is highly correlated to the Multiple Jobs Axis, and it can replace the jobs-per-day variable as the lead variable representing that axis in the meta-model. Note that current models do not differentiate between runtime load and CPU load – they implicitly assume that jobs use 100% of the available CPU during their entire run. Table 8 defines the meta-model equations based on the CL variable rather than the JD variable; only equations for CL are given, since the ones for Cm remain as they were in Table 6.

Table 8: Fitting equations and coefficients to the *CL* variables in stead of the *JD* variable

Equation	R^2
$JD = 39.309 * CL^{-2.9549}$	0.75
$Am = 285.72 * CL^{2.208}$	0.72
$Ai = 10103 * CL^{3.3032}$	0.73

Note that under this meta-model, changing the load affects the runtimes, parallelism, and inter-arrival times as expected. This is not a recipe for manipulating a given workload to achieve a desired load level; but a model that will be based on this meta-model will handle load manipulations correctly. Again we validate this model using four synthetically generated observations, as done in Table 7 for the JD/Cm meta-model. The low and high values used for the CPU load were 0.45 and 0.65 respectively, and the Co-Plot which merges Figure 1 with the four generated observations is given in Figure 11. Its coefficient of alienation is 0.11, and its average of correlations is 0.85. As the Figure

shows, the CL/Cm meta-model is as successful as the JD/Cm meta-model in representing the known workloads space.

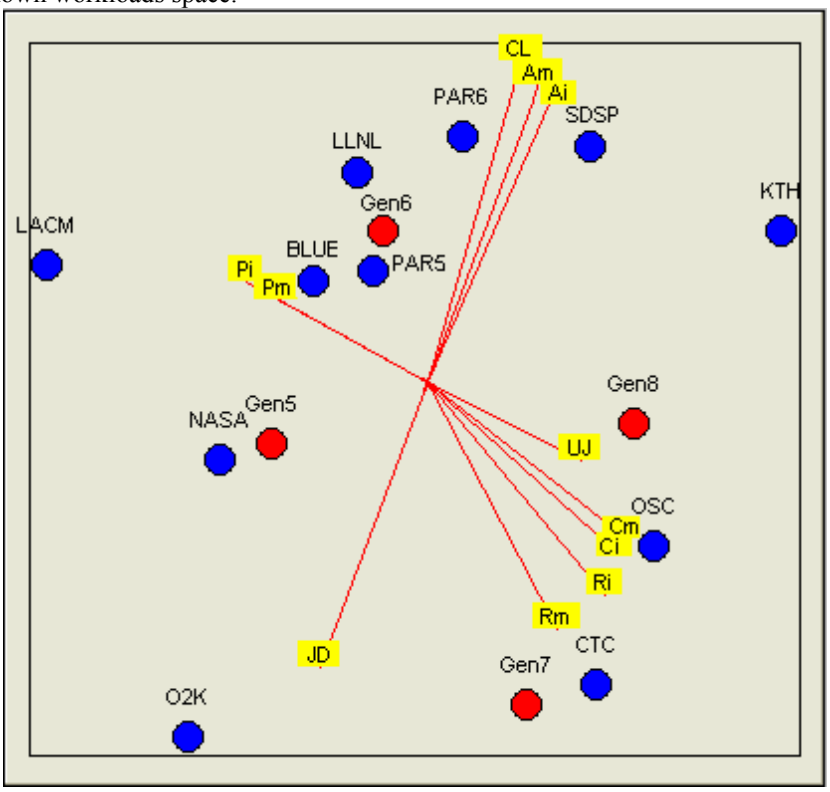

Figure 11: Verifying the Parametric CL/Cm Meta-Model via Synthetic Observation

12. IMPLICATIONS: GUIDELINES FOR WORKLOAD MODELING

The Co-Plot analysis can also be used to provide a set of concrete guidelines for the construction of future workload models. Note that the parametric models presented in the previous two sections are only meta-models: They do not provide a complete model that can be used to create a stream of parallel jobs. This requires traditional distributing fitting techniques, which are beyond the scope of the current paper. As we saw, the results of our Co-Plot analysis can help by providing guidelines for a complete model, by identifying the two major axes of correlated variables, and which variables to use as representatives. In addition, we can suggest the following observations.

1. Current models do a good job modeling the size of jobs – runtime, parallelism and CPU work – but not a very good job for modeling the temporal structure of the workload – inter-arrival times, jobs per day, and in a lesser sense the load.

The most interesting variables in figures 8 and 9 are those that aren't there. The number of jobs per day, the inter-arrival times interval (and in some runs the median), and both kinds of load, had a much lower correlation than when analyzed on the production workloads alone. This can only mean that the models have the "wrong" values in these variables. If the values of all observations for some variable don't grow in the same direction along some axis in the Co-Plot, then it is impossible to superimpose that variable's arrow with an adequate correlation on the plot. This means that the removed variables are not modeled well by most of the analyzed models. On the other hand, the variables related to the size of a job – runtime, parallelism and total CPU work – kept their high correlations, and thus seem to be well modeled today.

This means that the current models may be ill-suited for design decisions that are sensitive to the temporal structure of the workload. In particular, the multiple and delicate interactions between the workload and the scheduler [Feitelson, 2002 & 2003] mean that more work is needed when modeling a workload intended to be used to select a scheduler.

2. Production workloads are not stationary but change over time. Design decisions and workload models need to adjust over time accordingly.

Figure 1 shows that architecture, a site's user population, and even time do not guarantee similarity between workloads. Figure 6 shows that production workloads may change over time. There is no guarantee that a design decision for a computer based on the past – even its own past – is the right one to make. A parametric model, based on the expected use of the computer in the near future, may provide a better answer. For example, if an administrator of a production system plans to cancel a limit on the runtimes of jobs, and expects the system's median runtime to grow as a result, then our results can be used to predict the direction of change to other variables of the workload as well.

The models that are available today are similar in this issue to the production workloads, since they are non-parametric. This means that workloads generated by each model will have a single position (Co-Plot makes this visual, but the claim holds metaphorically as well). There is no reason to believe that an arbitrary future production workload will match that specific position. The Cirne-Berman model support for adjusting the inter-arrival pattern had little affect on its overall position – but this is expected, since changing only one variable of a workload can only have a limited effect. In addition, future models should be non-stationary if they are used to produce a workload that is more than a few months long. One approach to modeling such phenomena is using long-range statistical processes [Beran, 1994]; another approach may be a multi-class workload model, in which for example the arrival and departure of users can be modeled explicitly [Calzarossa and Serazzi, 1994].

3. The median and interval are the preferred statistics to use when comparing logs, because they are stable to flurries and to outliers in general. Model the runtime, degree of parallelism, inter-arrival times, and total CPU work by a distribution that has an almost full correlation between the median and interval.

As figure 5 shows, the median and interval statistics are very stable to changes in the log. In particular, they are robust to outliers, most notably in the shape of flurries. This is highly important to the analysis and use of workloads, since in most production logs there are many kinds of questionable entries. For example, if a job lasted ten times more than the machine's documented maximal runtime, it is impossible to tell whether this is a mistake in the log, or a real case in which the machine was dedicated to a high-priority special project. As shown in [Lazowska, 1977; Downey and Feitelson, 1999; Feitelson and Tsafrir, 2006], such unusual jobs can have a strong effect on the mean and standard deviation of an entire workload's runtimes, but have negligible effects on the runtime median and interval. It is impossible to reach any firm conclusions about workloads if the wrong statistics are used.

Moreover, figures 1-3 demonstrate that the correlation between the median and interval of most of our major variables is positive and high. This requires the use of the right distributions to model them; note that this requirement is different from correlation between the mean and standard deviation.

13. LIMITATIONS OF THE CO-PLOT METHOD

Co-Plot has several unique advantages as a statistical analysis method: In particular, it is applicable even when there are many variables and few observations, it analyses variables and observations together, and it does not assume that variables are

independent. However, Co-Plot has several inherent limitations as well, which one must consider when deciding whether to apply it to other datasets.

First, Co-Plot presents a relative, rather than absolute, picture of the analyzed observations. For example, when looking at Figure 1, we can say that it represents the differences between the analyzed logs, but we cannot tell whether these are "small" or "large" differences. It is possible that the addition of several logs which would be very dissimilar to this dataset will cause the current logs to cluster together, and the new logs to cluster at the opposite side of the plot. To minimize this effect, we have used a relatively large dataset from a variety of unrelated locations and architectures. But still, as in every statistical analysis, our conclusions are limited by the available information.

Second, Co-Plot places the arrows layer of the plot so that they are optimal with respect to the first layer of observations, but the first layer itself is not perfectly laid out. As explained in section 2, the arrow for each variable is placed after the observations are placed and independently of the other arrows, such that its angle best reflects the growth of values of the associated variable. However, since the observations are usually placed with a coefficient of alienation that is larger than zero, than the arrows are actually optimized to fit this skewed picture rather than the original data. This means that conclusions drawn from the arrows, for example about correlations between variables, may be skewed as well.

Co-Plot works this way because it is a precondition to analyzing both the observations and the variables in the same plot together. Two things must be done to minimize this effect. First, one must be careful to ignore plots with high coefficients of alienation, or low correlations for the fit of specific variables. The values we have given for the plots we used in this paper are considered adequate. Second, one should not rely on a single plot, but instead analyze many variations of the data – using slightly different observation and variable sets each time – and observe which features of the resulting analysis are stable.

A third, more general limitation of Co-Plot is that it is only useful when the data naturally fits into a two-dimensional space. If the data cannot be reduced (approximately) to a 2D space, a high coefficient of alienation will be measured, so there is no danger of unknowingly missing this phenomenon. If this happens a different technique will be required. The principles of Co-Plot are general and can be used with data in any dimensionality, not just 2D, but this makes visualization more difficult, and is therefore not supported by the current software implementations of Co-Plot.

A fourth limitation of Co-Plot is that it does not gracefully handle data with missing values. In our dataset, we have dealt with this issue by using approximations – for example, when the CPU load wasn't given, the runtime load was used to estimate it. However, this may be a more difficult problem in other datasets.

14. SUMMARY AND FUTURE RESEARCH

This paper makes two main contributions. First, it presents the Co-plot technique, a multivariate statistical method suitable for the demands of the workload modeling field. It works well given few observations and many variables, as happens in our case. And it does not require the variables to be independent for the analysis to be valid. In contrast, it analyses the variables together with the observations, and reports which variables are significant, and how they interact with one another and with the observations.

Second, this paper provides new insights about the majority of production workloads and synthetic models available today for parallel supercomputers. This is the largest and most diverse data set of workloads analyzed so far – our eleven observations are from ten machines, eight architectures and seven locations. This critical mass enables us to characterize the "workloads space" for real-world large production supercomputers, and to identify its important variables and their relations.

Future work extending this research is planned along two paths. The first is the use of the Co-Plot method for additional variables – testing for self-similarity, locality, cycles and so forth – and comparing the production and synthetic workloads on these questions. The second is to construct a new workload model, based on the meta-model and guidelines given in the previous sections, and to show examples in which its use is indeed required to make the right design decisions.

ACKNOWLEDGEMENTS

This work would not be possible without the help of many people who supplied logs, source codes, and explanations. We'd like to personally thank Bill Nitzberg, Reagan Moore, Allen Downey, Lars Malinowsky, Curt Canada, Moe Jette, and Joefon Jann.

We'd be happy to share the data sets and tools we used. The production workloads in standard workload format and source codes of synthetic models are available online at the Parallel Workloads Archive [Feitelson, 1999]. A Co-Plot program and a workload analysis program, both under Windows, are also available online at [Talby, 1999].

REFERENCES

AGRAWALA, A.K., MOHR, J.M. AND BYRANT, R.M. 1976. An approach to the workload characterization problem. *Computer 9(6)*, 18-32.

BERAN, JAN 1994. *Statistics for Long-Memory Processes* (*Monographs on Statistics and Applied Probability)*, Chapman and Hall, New York, NY.

BORG, INGWER AND GROENEN, PATRICK 1997. *Modern Multidimensional Scaling – Theory and Applications*. Springer Series in Statistics, Springer, pp.29-48 and pp. 199-205.

CALZAROSSA, MARIA AND SERAZZI, GIUSEPPE 1993. Workload Characterization: A Survey. In *Proceedings of IEEE 81(8)*, Aug 1993, 1136-1150.

CALZAROSSA, MARIA AND SERAZZI, GIUSEPPE 1994. Construction and Use of Multiclass Workload Models. *Performance Evaluation 19(4)*, 341-352.

CIRNE, WALFREDO AND BERMAN, FRANCINE 2001. A Model for Moldable Supercomputer Workloads. In *Proceedings of 15th Intl. Parallel & Distributed Processing Symposium*, Apr 2001.

DOWNEY, ALLEN B. 1997. A Parallel Workload Model and Its Implications for Processor Allocation. In *Proceedings of 6th Intl. Symposium on High Performance Distributed Computing*, Aug 1997.

DOWNEY, ALLEN B. 1997b. Using Queue Time Predictions for Processor Allocation. In *Job Scheduling Strategies for Parallel Processing 1997*, D. G. FEITELSON AND L. RUDOLPH, Eds., Springer-Verlag, Lecture Notes in Computer Science vol. 1291, 35-57.

DOWNEY, ALLEN B. AND FEITELSON, DROR G. 1999. The Elusive Goal of Workload Characterization. *Performance Evaluation Review 26(4)*, 14-29.

FEITELSON, DROR G. 1996. Packing schemes for gang scheduling. In *Job Scheduling Strategies for Parallel Processing 1996*, D. G. FEITELSON AND L. RUDOLPH, Eds., Springer-Verlag, Lecture Notes in Computer Science vol. 1162, 89-110.

FEITELSON, D. G. 1999. *The Parallel Workloads Archive*. Available from World Wide Web: (http://www.cs.huji.ac.il/labs/parallel/workload)

FEITELSON, DROR G. 2002. The forgotten factor: Facts; On performance evaluation and its dependence on workloads. In *Proceedings of EuroPar 2002*, Aug 2002, Springer-Verlag, Lecture Notes in Computer Science vol. 2400, 49-60.

FEITELSON, DROR G. 2003. Metric and workload effects on computer systems evaluation. *Computer 36(9)*, 18-25.

FEITELSON, DROR G. AND JETTE, MORRIS A. 1997. Improved Utilization and Responsiveness with Gang Scheduling. In *Job Scheduling Strategies for Parallel Processing 1997*, D. G. FEITELSON AND L. RUDOLPH, Eds., Springer-Verlag, 1997, Lecture Notes in Computer Science vol. 1291, 238-261.

FEITELSON, DROR G. AND NITZBERG, Bill 1995. Job characteristics of a production parallel scientific workload on the NASA Ames iPSC/860. In *Job Scheduling Strategies for Parallel Processing 1995*, D. G. FEITELSON AND L. RUDOLPH, Eds., Springer-Verlag, 1995, Lecture Notes in Computer Science vol. 949, 337-360.

FEITELSON, DROR G. AND RUDOLPH, LARRY 1998. Metrics and Benchmarking for Parallel Job Scheduling. In *Job Scheduling Strategies for Parallel Processing 1998*, D. G. FEITELSON AND L. RUDOLPH, Eds., Springer-Verlag, 1998, Lecture Notes in Computer Science vol. 1459, 1-24.

FEITELSON, DROR G. AND TSAFRIR, DAN 2006. Workload sanitation for performance evaluation. In *Proceedings of IEEE Intl. Symposium on Performance Analysis of Systems and Software*, Mar 2006.

FERRARI, D. 1972. Workload characterization and selection in computer performance measurement. *Computer 5(4)*, 18-24.

GUTTMAN, L. 1968. A general non-metric technique for finding the smallest space for a configuration of points. *Psychometrica 33*, 479-506.

HOTOVY, STEVEN 1996. Workload Evolution on the Cornell Theory Center IBM SP2. In *Job Scheduling Strategies for Parallel Processing 1996*, D. G. FEITELSON AND L. RUDOLPH, Eds., Springer-Verlag, 1996, Lecture Notes in Computer Science vol. 1162, 27-40.

JANN, JOEFON; PATTNAIK, PRATAP; FRANKE, HUBERTUS; WANG, FANG; SKOVIRA, JOSEPH AND RIODAN, JOSEPH 1997. Modeling of Workload in MPPs. In *Job Scheduling Strategies for Parallel Processing 1997*, D. G. FEITELSON AND L. RUDOLPH, Eds., Springer-Verlag, 1997, Lecture Notes in Computer Science vol. 1291, 95-116.

KOLDINGER, E.J.; EGGERS, S.J., AND LEVY, H.M. 1991. On the validity of trace-driven simulation for multiprocessors. In *Proceeding of 18th Ann. Intl. Symp. Computer Architecture*, May 1991, 244-253,.

LAZOWSKA, E.D. 1977. The use of percentiles in modeling CPU service time distributions. In *Computer Performance*, K.M. CHANDY AND M. REISER, Eds., North Holland, 53-66.

LIPSHITZ, G., AND RAVEH, A. 1994. Applications of the Co-plot method in the study of socioeconomic differences among cities: A basis for a differential development policy. *Urban Studies 31*, 123-135.

LO, V.; MACHE, J. AND WINDISCH, K. 1998. A comparative study of real workload traces and synthetic workload models for parallel job scheduling. In *Job Scheduling Strategies for Parallel Processing 1998*, D. G. FEITELSON AND L. RUDOLPH, Eds., Springer Verlag, 1998. Lecture Notes in Computer Science vol. 1459, 25-46.

LUBLIN, URI AND FEITELSON, DROR G. 2003. The workload on parallel supercomputers: modeling the characteristics of rigid jobs. *Journal of Parallel & Distributed Computing 63(11)*, 1105-1122.

MAITAL, S. 1978. Multidimensional Scaling: Some Econometric Applications. *Journal of Econometrics 8*, 33-46.

MAJUMDAR S.; EAGER, D.L. AND BUNT, R.B. 1988. Scheduling in multiprogrammed parallel systems. In *SIGMETRICS Conf. Measurement & Modeling of Computer Systems*, May 1988, pp. 104-113.

RAVEH, ADI 2000. The Greek banking system: Reanalysis of performance. *European Journal of Operational Research 120*, 525-534.

TALBY, D. 1999. *Visual Co-Plot*. Available from World Wide Web: (http://www.cs.huji.ac.il/~davidt/vcoplot)

TSAFRIR, DAN AND FEITELSON, DROR G. 2006. Instability in parallel job scheduling simulation: the role of workload flurries. In, *20th Intl. Parallel & Distributed Processing Symposium* Apr 2006.

WINDISCH, K.; LO, V.; MOORE, R.; FEITELSON, D. AND NITZBERG, B. 1996. A comparison of workload traces from two production parallel machines. In *Proceedings of 6th Symposium on Frontiers Massively Parallel Computing*, Oct 1996, 319-326.