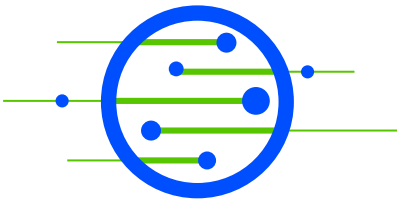

A Case for Informed Service Placement on PlanetLab

David Oppenheimer, David A. Patterson
University of California, Berkeley

Amin Vahdat
University of California, San Diego

PDN–04–025
December 2004

Status: Ongoing Draft.

A case for informed service placement on PlanetLab

David Oppenheimer, David A. Patterson, and Amin Vahdat
University of California {at Berkeley, at Berkeley, San Diego}
{davidopp,pattrsn}@cs.berkeley.edu, vahdat@cs.ucsd.edu

Abstract

We examine PlanetLab resource utilization data from the perspective of the designer of a resource discovery system. We are particularly interested in variability of, and correlations among, resource utilization attributes, both among nodes and over time. We find that, for some resources, the quantity available at a fixed time differs significantly across nodes, suggesting a potential benefit to using a resource discovery system to wisely place application instances. We further find that, for many nodes, resource availability varies over time, suggesting periodic migration of application instances could be useful. We find that some attributes are moderately correlated among nodes at the same site, but that on a single node, attributes are generally not correlated to one another. Finally, we find that for some node pairs, measured latency can reasonably predict available bandwidth based on a correlation function derived from observing latency and bandwidth across all node pairs.

1. Introduction

Shared distributed platforms such as PlanetLab[2] have become popular for evaluating and deploying wide-area distributed applications. At the same time, structured peer-to-peer overlay networks have become an attractive building block for such applications, as they provide a scalable, self-healing, self-configuring routing substrate.

One option for deploying a peer-to-peer application is to create an instance of the application on every available platform node. But because node and network links in platforms like PlanetLab are time-shared among competing users, the amount of available node and network resources varies unpredictably across nodes and over time. In this environment, the application deployer wishing to maximize application performance and predictability may decide to deploy her application only on those nodes offering desired amounts of per-node and inter-node resources, and to periodically migrate application instances as these underlying conditions change. Indeed, the strategies in OpenHash[6] allow her to be selective in choosing where to run her peer-to-peer application while still leveraging a single shared public DHT infrastructure. Or if she is building her application on an unstructured peer-to-peer network, she might deploy her application on all nodes, but choose a subset of powerful or well-connected nodes as "super-peers."

The process of locating a set of nodes that meet application requirements can be automated by a resource discovery system. In this paper we study time and space relationships among a variety of node and network resource attributes collected from PlanetLab, focusing on what these relationships tell us about the potential usefulness of employing a resource discovery system to enable informed service placement, and about certain design choices one might make when building such a system.

For example, if all nodes offer the same amount of resources at all times, then any selection of nodes is as good as any other, and intelligent node selection is unimportant. Even in the presence of resource heterogeneity at fixed points in time, if the amount of available resources on a node varies too fast and by too large a magnitude, then the set of nodes satisfying a resource discovery request will cease to satisfy the request very soon after a resource discovery mapping is computed. At the other extreme, if the amount of available resources is invariant, then *online* resource discovery is unnecessary--we can simply take a snapshot of node characteristics at one point in time and thereafter use those measurements to compute resource mappings offline. Between these extremes, a system that provides an initial deployment-time mapping and periodically suggests migrating some application instances to new nodes, is ideal. Among the goals of this paper is to determine whether the prevalence, frequency, and magnitude of resource variability on PlanetLab falls within the range that makes informed service placement useful.

Additionally, we note that if correlations exist among a set of resource attributes, then the resource discovery system can measure and store just one value from each set, using it as a surrogate for the others in the set, and thereby reducing resource usage by the resource discovery system. This paper investigates whether such correlations exist.

Thus this paper addresses the following questions that impact the usefulness and overhead of resource discovery and service placement:

- How much does the available amount of per-node resources vary among nodes at a fixed time?
- How much does the available amount of per-node resources vary over time? How much do inter-node latency and available bandwidth vary over time?
- On a given node, are any per-node attributes strongly correlated? Are inter-node latency and available bandwidth correlated?

2. Experimental environment

We used four data sources for our analyses: Ganglia[7] and CoMon[9] for per-node attributes, and all-pairs-pings (APP)[10] and Iperf[3] for inter-node latency and available bandwidth, respectively. For the first two we used data collected from October 10, 2004 to October 24, 2004. For the second two we used data collected over a one-month period ending at the same time. Although the total number of PlanetLab nodes varied during these periods, approximately 250 nodes were able to provide data on an average day.

Every five minutes we polled Ganglia's per-node report of free memory, free swap space, free disk space, one-minute load average, and 15-second average of number of network bytes sent and received per second. At the same time we polled CoMon's report of the number of "active slices" on the node. A PlanetLab *slice* is roughly equivalent to a user, and this measure tells us how many slices used at least one-tenth of one percent of the CPU in the last 5 minutes. Finally, APP measures the latency between all pairs of PlanetLab nodes every 15 minutes, and Iperf measures the bandwidth available to a bulk TCP transfer between every pair of PlanetLab nodes once to twice a week.[1]

Our Ganglia data on load comes with the following caveat. As on all other Linux systems, the one-minute load is the average number of threads that were runnable at any point in time over the last minute. But because PlanetLab nodes use proportional-share CPU scheduling, the load on a node does not directly dictate the fraction of CPU time a slice receives. Instead, the number of other slices with runnable threads (approximated by the "active slices" attribute we measured) dictates the fraction of CPU time a slice will receive. Still, some slices that were active at some point during the past five minutes will not have been active over all windows (100s of milliseconds in length) over which share proportionality is computed, so one-minute-averaged load information is also useful for obtaining a full picture of the CPU resources available on a node. Moreover, load always represents *demand* for the CPU on a node, and is thus an interesting metric.

3. Results

In this section we describe the data analysis we performed to answer the questions in Section 1, and the conclusions we draw from that analysis.

[1] Because of the relatively smaller number of measurements in the bandwidth dataset, we conducted the experiments involving latency and bandwidth on an additional dataset from the preceding month. We found similar results, so we present only the results from one of the month's datasets here.

attribute	mean	std. dev.	10th %ile	90th %ile
# of CPUs	1.0	0.0	1.0	1.0
cpu speed (MHz)	1942	572	1263	2652
total disk (GB)	127	88.5	35.1	232
total mem (MB)	1153	467	628	2017
total swap (GB)	1.0	0.0	1.0	1.0

Table 1: Variation across all measurements in the trace. Additionally, PlanetLab nodes are located in approximately 27 countries.

3.1. Resource heterogeneity

Our first experiment attempts to roughly assess the degree to which the amount of resources available varies among nodes at any fixed point in time. First we computed the mean, standard deviation, 10th percentile, and 90th percentile of each resource quantity, across all measurements from all nodes during the two week interval. Note that these attributes are not normally distributed, so although standard deviation still measures variability, it does not allow one to directly infer the percentage of values that are within various intervals of the mean, as can be done when the distribution is normal. Table 1 shows the results for five important attributes that are generally static. Because the initially-deployed set of PlanetLab nodes were identical, and subsequently-added nodes must meet certain configuration requirements, it is not surprising that some of these attributes vary little. Table 2 presents the same analysis for the seven dynamic attributes that we consider the most relevant to resource discovery. We see that all attributes vary significantly; the 90th percentile value

attribute	mean	std. dev.	10th %ile	90th %ile
1 min load avg.	6.81	20.06	1.05	11.86
free mem (MB)	62.359	125.234	13.668	105.432
free swap (MB)	755.596	178.795	524.336	1000.268
free disk (GB)	102.8	86.04	8.088	208.3
active slices	13.3	5.96	0.0	20.0
bytes/s in	50477	117023	5568	92877
bytes/s out	52543	130112	5476	96214

Table 2: Variation across all measurements in the trace, for dynamic attributes.

of an attribute is often more than 10x the 10th percentile value. To rule out the possibility that this variation is due to our incorporating all measurements from the two-weeks period, rather than because of high variability across nodes at a single point in time (*i.e.*, to rule out a scenario in which all nodes exhibit the same values for each attribute at any point in time, but those values vary in concert over time), we compute the *coefficient of variance* (CV) across all nodes for each of the attributes at each time interval. CV, defined as standard deviation divided by mean, is a normalized measure of the variability of a quantity (in this case, the variability across nodes for a particular attribute at one point in time). This data appears in Figure 1.

We see that free swap space, number of active slices, and free disk space vary the least across nodes and that the degree of this variation changes little over time. On the other hand, load, free memory, bytes in, and bytes out vary the most across nodes and the degree of this variation itself varies substantially over time. In general, because all attributes exhibit nontrivial CVs at almost all time intervals, we conclude that the quantity of available per-node resources varies sufficiently across nodes at any point in time to justify using resource discovery to choose wisely the nodes on which an application is deployed.

3.2. Variability of per-node attributes over time

Our next question relates to the variability of resources on a single node over time. Recall that very low variability suggests offline node selection is sufficient, moderate variability suggests that online resource discovery with periodic task migration is useful, and very high variability suggests that resource discovery is unlikely to be useful. To assess resource variability, we compute, for each node and attribute, over some time interval, the average of the magnitudes (absolute value) of the percentage differences in the "bad" direction for that attribute between its value at each observation in the interval and its value at the first observation in the interval. For example, if the interval is 20 minutes and a node's loads are 10, 3, 15, 20, then we compute the average of 0%, 0% (load went down), 50%, and 100%. This tells us that if an application were deployed on that node and left there for 20 minutes, it would experience an average 38% "bad" load deviation from the deployment-time load value. We perform this process using intervals of 30 minutes, 1 hour, 4 hours, and 12 hours, corresponding to leaving the application running on the node, without migration, for 30 minutes, 1 hour, 4 hours, and 12 hours. We average across all intervals of the given length starting at each observation time in the trace. From this we compute the CDF of the percentage of nodes with each possible average magnitude of percentage difference over the interval. The results for each interval length are graphed in Figure 2-4. The graph for 12-hour interval length is nearly identical to that for 4-hour length, so it is not presented here.

We make three observations about these graphs. First, the degree of variability of each attribute can be roughly ranked identically for each interval size: free disk space and free swap space vary little; network bytes sent and received vary significantly; free memory and load average vary moderately; and active slices varies moderately to significantly. We emphasize that this data presents only part of the picture: the *impact* of resource variability on an application depends on application sensitivity to that resource. For example, the impact on an application of a 10% change in the load on a node may be larger or smaller than the impact of a 10% change in the amount of free

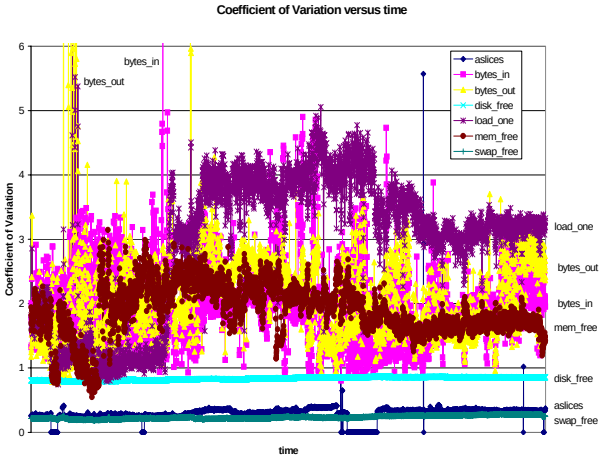

Figure 1: Coefficient of variation over time.

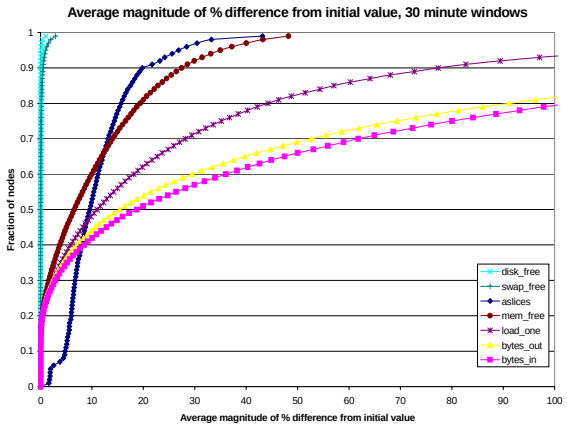

Figure 2: Attribute variability over time for 30-minute windows.

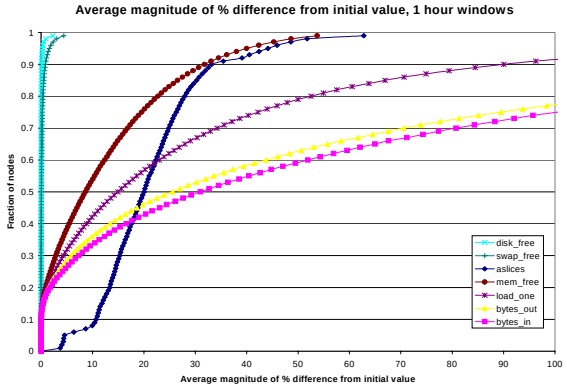

Figure 3: Attribute variability over time for 1-hour windows.

memory on that node. Nonetheless, our preliminary investigation shows that memory, load, and competition for endpoint network resources can change significantly in the "bad" direction over longer time periods, suggesting that migration could be of value applications that are sensitive to those resources.

Second, we note that the CDF curve shifts to the right as the interval length increases, suggesting that attributes vary less over short time periods than they do over longer periods. Choosing an appropriate migration interval requires finding the "sweet spot" in the curve of variability versus interval length. We have presented only three points on this curve, but we observe that reducing the interval length below 1 hour does not significantly reduce observed variation, implying that a 1-hour migration interval would capture about as much benefit as a shorter migration interval. Unfortunately even with a 30-minute migration interval, several attributes vary significantly during the interval. Nonetheless,this variation is not nearly as severe as over a 4-hour (or 12-hour) interval.

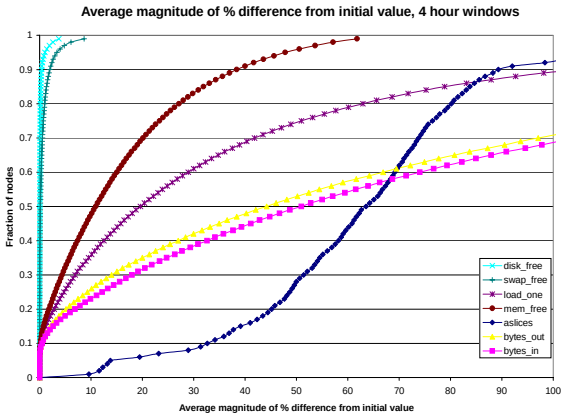

Figure 4: Attribute variability over time for 4-hour windows. .

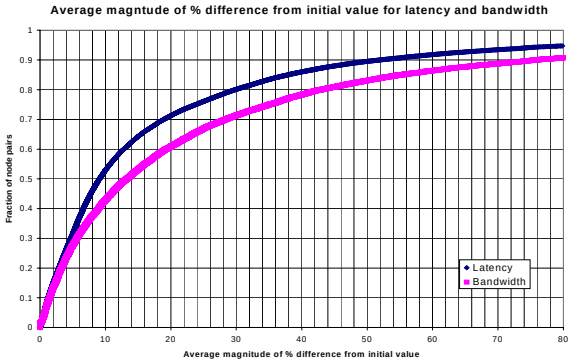

Figure 5: Inter-node latency and bandwidth variation over time.

Third, we observe that the slope of the CDF varies among attributes. A high-slope CDF suggests that all nodes have similar values of that amount of deviation from the initial value, while a low-slope CDF suggests that nodes exhibit a wide range of average deviations. We found that the slope tended to decrease as the median variability increased. This suggests that it might be possible to classify nodes as "high variability over time" or "low variability over time" nodes with respect to the high-median-variability attributes free memory, load, and network traffic, in turn allowing a user to avoid high-variability nodes if their application is sensitive to high variability in those attributes.

3.3. Variability of inter-node attributes over time

Next we computed similar CDFs for the inter-node attributes latency and available bandwidth but using the entire month of inter-node data as the window. Figure 5 shows the CDF of the percentage of node pairs with each average magnitude of percentage difference from the initial value, averaged over all starting points in the trace (all node-pair latency measurements were taken once every 15 minutes, and all node-pair bandwidth measurements were taken 1-2 times per week). We observe that latency and bandwidth exhibit about the same amount of variability over a month-long window as do per-node attributes over much shorter windows. This suggests that migration based on inter-node network characteristics may be less useful than that based on per-node resource quantities.

3.4. Correlation among per-node attributes

Our next set of analyses investigate the feasibility of reducing the overhead of a resource discovery service by using some attributes as surrogates for other attributes, and then only measuring and storing those

attributes that are serving as surrogates. We compute the correlation coefficient (r) of every pair of per-node attributes, treating each node's collection of measurements at each timestep as separate observations. Table 3 shows the correlation coefficients for the seven attributes we have been studying. Additionally, along the diagonal we record the overall correlation between the value of that attribute on the first node at each site and that of the same attribute on the second node at the same site (at least two nodes are typically installed at each PlanetLab site). Recall that a correlation coefficient of 1.0 means two attributes are perfectly correlated, a correlation coefficient of -1.0 means two attributes are perfectly inversely correlated, and a correlation coefficient of 0.0 means two attributes are completely uncorrelated. For reference, the correlation coefficient between load averaged over the last one-minute interval and load averaged over the last five-minute interval is 0.966, and the correlation coefficient between the time a system was last rebooted and the one-minute load average is 0.01.

We see that no pairs of different attributes are very strongly correlated (though bytes_in and bytes_out are somewhat correlated). From this we conclude that no pair of attributes are suitable to serve as surrogates for one another. One possible explanation for this result is that applications generally don't consume resources in equal proportions--for example, some applications are CPU-bound, consuming significant CPU but not necessarily much else--so the mix of application types on a node determines how much of each resource is used. Breaking the trace down into one-hour segments revealed some intervals with somewhat higher-than-average correlations, particularly amongst load, free swap, and active slices (r magnitudes greater than 0.4 for about 30 one-hour periods of the trace).

r	load one	mem free	swap free	disk free	actv slice	byte in	byte out
load one	*.080*						
mem free	-.050	*.627*					
swap free	-.231	.274	*.473*				
disk free	-.035	.192	.212	*.929*			
actv slice	.079	-.050	-.219	.049	*.773*		
byte in	.059	-.033	-.074	.059	.140	*.209*	
byte out	.058	-.033	-.059	.078	.137	.443	*.188*

Table 3: Correlation between pairs of attributes on the same node, except elements on the diagonal, which represent correlation between two nodes at the same site with respect to that attribute. ,

Looking between pairs of hosts at the same site instead of between pairs of attributes on a single node, the values along the diagonal reveal that for several attributes there is a correlation between the value of that attribute on the first node at a site and the value of that attribute on the second node at the same site. This suggests that depending on the user's tolerance for approximate answers, a resource discovery system could use one node at a site as a surrogate for the others at that site with respect to free disk space, number of active slices, or free memory.

3.5. Correlation among inter-node attributes

One pair of potentially correlated attributes that merit special attention is latency and bandwidth. In general, for a given loss rate, one expects bandwidth to vary roughly with 1/latency[8]. If we empirically find a strong correlation between latency and bandwidth, we might use latency as a surrogate for bandwidth, saving substantial measurement overhead.

For this experiment we annotated each pairwise bandwidth measurement collected by Iperf with the most recently measured latency between that pair of nodes as recorded by APP[1]. We graph these <latency,bandwidth> tuples in Figure 6 along with the power-law regression line derived from them. We find a correlation coefficient of -0.59, suggesting a moderate inverse power correlation. Viewed another way, using the regression equation derived from this empirical data leads to an average 233% error when attempting to estimate bandwidth using only measured latency. We conclude that latency is a suitable surrogate for bandwidth when a rough estimate of bandwidth is needed. One possible explanation for the imperfection of this correlation is that some PlanetLab nodes are artificially rate-limited (bandwidth-capped) due to cost concerns, thus artificially lowering available bandwidth below what would be predicted based on observed latency. Indeed, one can see a dense rectangular region at the bottom of Figure 6 below a horizontal line at about 1600 kb/s, where decreased latency does not correlate to increased bandwidth.

[1] APP probes every 15 minutes, so this approach allows for up to 7.5 minutes between a bandwidth measurement and the closest latency measurement.

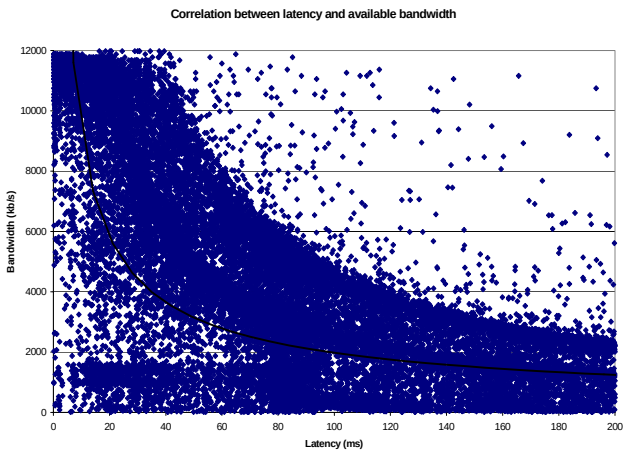

Figure 6: Latency-bandwidth correlation.

Such nodes are one possible explanation for the presence of this region.

We did find, however, that certain node pairs showed strong latency-bandwidth correlation. For example, using the regression equation obtained from the entire dataset, 17% of node pairs had bandwidths within 25% of what would be predicted given their latencies, and 56% of node pairs had bandwidths within 50% of what would be predicted. This suggests that although the latency-bandwidth correlation across the system as a whole is only moderate, we *can* use latency to estimate bandwidth reasonably effectively *for a substantial fraction of node pairs*. Once such node pairs are identified, a resource discovery system could reduce measurement overhead by inferring available bandwidth between these node pairs from measured latency between these node pairs, rather than measuring bandwidth directly.

4. Conclusion and related work

Performance and predictability requirements may dictate that a distributed application run only on nodes with sufficient resources. In examining resource availability statistics collected over two weeks from PlanetLab, we find that available resources vary significantly among nodes, suggesting that wise placement of application instances can be beneficial. Further, we find that node and network resources vary over time at a rate that suggests that periodic migration of application instances in response to changes in node resource availability may be warranted. We find that for some attributes there is a strong correlation among nodes at the same site, but on a single node, different attributes are not strongly correlated to one another. Finally, we for some node pairs, measured latency can reasonably predict available bandwidth based on a correlation function derived from observing latency and bandwidth across all node pairs

This work adds to a growing literature on measurements of resource utilization in Internet-scale systems, a subset of which we mention here. [1] studies throughput stability to many hosts from the vantage point of the 1996 Olympic Games web server, while [13] collects data from 31 pairs of hosts. [4] describes how to monitor a subset of paths to estimate loss rate and latency on all other paths in a network. [11] and [12] focus on predicting bandwidth and CPU utilization in Grid systems. Finally, [5] also studies PlanetLab nodes, but it focuses on failures and user workloads rather than on the implications of resource attribute heterogeneity, variability, and correlation for resource discovery; and it does not include data on network latency and available bandwidth.

References

[1] H. Balakrishnan, M. Stemm, S. Seshan, and R. H. Katz. Analyzing stability in wide-area network performance. *SIGMETRICS '97*, 1997.

[2] A. Bavier, L. Peterson, M. Wawrzoniak, S. Karlin, T. Spalink, T. Roscoe, D. Culler, B. Chun, and M. Bowman. Operating systems support for planetary-scale network services. *NSDI*, 2004.

[3] P. Brett. Iperf. http://www.planet-lab.org/logs/iperf/

[4] Y. Chen, D. Bindel, H. Song, and R. H. Katz, An algebraic approach to practical and scalable overlay network monitoring. *SIGCOMM '04*, 2004.

[5] B. Chun and A. Vahdat. Workload and failure characterization on a large-scale federated testbed. Intel Research Berkeley Technical Report IRB-TR-03-040, 2003.

[6] B. Karp, S. Ratnasamy, S. Rhea, and S. Shenker. Spurring adoption of DHTs with OpenHash, a public DHT service. *IPTPS '04*, 2004.

[7] M. Massie, B. Chun, and D. Culler. The Ganglia distributed monitoring system: design, implementation, and experience. *Parallel Computing*, Vol. 30, Issue 7, July 2004.

[8] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. *SIGCOMM '98*, 1998.

[9] V. Pai. CoMon. http://codeen.cs.princeton.edu/comon/

[10] J. Stribling. All-pairs pings for PlanetLab. http://www.pdos.lcs.mit.edu/~strib/pl_app/

[11] S. Vazhkudai, J. Schopf and I. Foster. Predicting the performance of wide area data transfers. *IPDPS '02*, 2002.

[12] L. Yang, I. Foster, and J. M. Schopf. Homeostatic and tendency-based CPU load predictions. *IPDPS '03*, 2003.

[13] Y. Zhang, V. Paxson, and S. Shenker. The stationarity of internet path proprerties: routing, loss, and throughput. *ACIRI Technical Report*, May 2000.