Resource Availability Characteristics and Node Selection in Cooperatively Shared Computing Platforms

Vinit Padhye and Anand Tripathi
Department of Computer Science
University of Minnesota Minneapolis, 55455 Minnesota USA
Email: (padhye,tripathi)@cs.umn.edu

***Abstract*—The focus of our work is on developing techniques for selecting nodes for scheduling applications in large-scale, cooperatively pooled, shared computing platforms. In such platforms, resources at a node are allocated to competing users on fair-share basis, without any reserved resource capacities for any user. There is no platform-wide resource manager for the placement of users on different nodes. The users independently select nodes for their applications. Our study is focused on the PlanetLab environment which exemplifies such platforms. For developing node selection techniques, we first study the resource utilization characteristics of PlanetLab nodes. Our approach uses the notion of *eligibility period,* which represents a contiguous duration for which a node satisfies a given resource requirement. We study the characteristics of the eligibility periods of Planetlab nodes for various resource capacity requirements. Based on this study we develop models for identifying nodes that are likely to satisfy a given requirement for long durations. We also develop an online model for predicting the idle resource capacity that is likely to be available on a node over a short term. We evaluate and demonstrate the performance benefits of the node selection techniques and the prediction model using the PlanetLab node utilization data traces collected at different intervals over an extended period of several months.**

I. Introduction

Federated computing platforms such as Grid computing environments [8] and the PlanetLab [3] demonstrate the feasibility of using cooperatively pooled distributed resources for deploying global scale distributed applications. Even though the PlanetLab platform is primarily intended for experimental research, it demonstrates how cooperatively pooled shared computing resources across a large number of organizations can be utilized for building large-scale, distributed, shared computing platforms. In contrast to Cloud platforms such as Amazon EC2 [2] or Microsoft Azure [16], such cooperatively pooled shared computing platforms have several distinguishing characteristics. These platforms typically do not provision guaranteed levels of resource capacities to an application. Moreover, cooperatively pooled resources in such platforms are generally widely dispersed and loosely managed by the participating organizations. They generally do not utilize any centralized resource management and scheduling mechanisms, thereby putting the responsibility of node selection for application deployment and scheduling on the users. Different user applications maybe co-hosted on a node and they compete for the resources available on that node.

The focus of our work is on developing techniques and heuristics to guide application developers in selecting nodes for their application deployment. We consider here cooperatively pooled shared computing platforms with the following characteristics, as exemplified by the PlanetLab system.

- *No provision of fixed resource capacity:* These platforms do not provide any dedicated resources with fixed capacity guarantees or reservation of resource capacities for its users.
- *No central resource manager:* In such platforms, there is no central resource manager or scheduler for platform-wide resource allocation in order to balance the resource utilization and load. The users deploying the applications select the nodes to be used.
- *Fair-share based resource allocation:* The resources on a single node are allocated on fair-share basis to the competing users. For example, in PlanetLab a user is given a *slice* on each node and the resources are allocated to slices on fair-share basis. An application can consume the unused resources on a node as long as other users do not compete for it. However, the unused resource capacities on a node can fluctuate due to the changing resource demands of the applications and the number of users on the node. Due to these factors, there is no guarantee of the resource capacities available to an application.
- *Lack of guarantees for node availability:* The availability of a node is not guaranteed due to crashes or shutdowns. In such federated environments, the owner of a node has autonomous control to shutdown it at any time.

The available resource capacities on PlanetLab nodes can fluctuate significantly as shown by the study presented in [20]. It is shown there that the available resource capacity at a node may change significantly within 30-60 minutes. As observed in [20] as well as in our study of resource availability of PlanetLab nodes, a node selected to execute a task with some given resource requirements may become

This work was supported by National Science Foundation grants 0834357 and 0708604

unsuitable for hosting it in the near future due to the changes in the available resource capacity on that node. This motivates the need of supporting dynamic relocation of tasks based on resource availability. For scheduling or relocation of a task, we need to identify the nodes that satisfy the task's resource requirements. Furthermore, as observed in our study, the available resource capacities and their fluctuations vary significantly across nodes. Therefore, in order to minimize the number of migrations, it is important to discriminate among eligible nodes to identify the nodes that are most likely to satisfy a task's requirement for a long duration.

Towards developing techniques for node selection, we study the characteristics of resource availability of PlanetLab nodes using the resource utilization data that we collected at different intervals over an extended period. In this regard, our aim is not to study and characterize the long term distributions or usage patterns of PlanetLab nodes, which may change over time. Rather we are interested in understanding the node behaviors over a short term (ranging from several hours to a week) to develop techniques for selecting nodes based on their recent behavior in terms of resource availability. The techniques that we present in this paper require only about 30 minutes observation of resource usage of nodes. Thus an application deployer would require to monitor nodes only for a short time before selecting nodes for deployment or in making any online scheduling or relocation decisions.

We first study the behavior of nodes in terms of their eligibility for a task's resource requirements. The requirements of a task could be stated in terms of CPU capacity, memory, and network bandwidth. We refer to the set of nodes satisfying a given resource capacity requirement as its *eligibility set*. The *eligibility period* of a node is defined as the contiguous period for which it remains in the eligibility set. We observe the distribution of eligibility periods and set sizes for various resource requirements. The distribution of eligibility periods indicates how long a randomly selected node is likely to meet the given requirement of a task. The expected value of the eligibility set size is also an indicator of the average number of tasks of a given resource capacity requirement that can be scheduled in the system. In this study we are not concerned with node availability in terms of the MTTF and MTTR of the PlanetLab nodes as presented in [25]. Our focus is on the eligibility periods of nodes for some given resource capacity requirements; these periods typically tend to be much smaller than the MTTF values (mean 3.8 days and median 3.16 days as reported in [25]).

Our study indicates that generally a node remains eligible for smaller durations for CPU requirements as compared to memory and network bandwidth requirements. The node eligibility periods depend on the resource requirement levels for CPU and memory, whereas in case of network bandwidth the eligibility periods typically tend to be less sensitive to the requirement levels. Our study shows that for all resource types, the eligibility periods typically tend to be small with very high probability. The distribution of eligibility periods tends to be long-tailed indicating that some fraction of nodes have very high eligibility periods. We observe that recent resource usage behavior of nodes is a good indicator in selecting nodes. When a node remains eligible for a certain duration it tends to continue being eligible for a long duration. We utilize these characteristics of node resource availability in developing techniques for identifying nodes that are likely to satisfy a given resource requirement for long durations. Specifically, we develop techniques for node selection that take into account the recent eligibility period profile of a node when considering it for inclusion in the eligibility set. We present here the basis for these techniques and evaluate their performance.

When a node hosting an application becomes ineligible for its resource requirements, relocating the application to another node may not be a good option if there is a high probability for the current node to become eligible again within a short time. This can be an important consideration if the cost of relocation is high. In this regard, an important question that needs to be addressed is what is the probability that a node would become eligible again within a given duration after it becomes ineligible. To address this question, we define *ineligibility period* as the time between two successive eligibility periods of a node and observe its distribution. We find that the probability for the ineligibility periods to be under 60 seconds tends to high (close to 80%). This indicates that if an application can tolerate the ineligibility of its host for a short duration in meeting its resource requirements, then it can continue to use that node for longer periods without relocating. We present here the details of this investigation and its benefits.

Another important question that we address in this paper is how to predict the resource capacity that is likely to be available on a node in the near future. An application hosted on a node may need to estimate the resource capacity that is likely to be available beyond its minimum resource requirement. Such prediction of available resource capacity can be useful for applications such as replicated services or any distributed application which can load-balance requests or schedule their computation based on the estimated available resource capacity. Towards this we develop an online prediction model which takes into account the recent behavior of the node. We utilized this prediction model in developing techniques for building autonomically scalable services in such environments [21].

In the next section we describe the mechanisms we used for monitoring resource usage of PlanetLab nodes. Section III describes the datasets we collected for our study. Section IV presents our study of resource availability characteristics of PlanetLab nodes. Section V presents the approach for selecting nodes based on their recent availability profiles. In Section VI we present the model we developed for predicting the resource capacity likely to be available at a node. Discussion on related work is presented in Section VII and the conclusions are presented in the last section.

II. Platform-Level Resource Usage Monitoring

For this study we needed to collect the data about resource utilization of PlanetLab nodes over time. In order to obtain accurate measurements of a node's behavior in terms of

resource utilization, we wanted to collect this data at high frequency, such as at 10-20 seconds interval. We observe that *CoMon* [22], the node monitoring service provided by PlanetLab, cannot be used directly for this purpose as it provides average values of resource usage over system-defined monitoring intervals of one minute and five minutes. Thus, the resource usage information of PlanetLab nodes provided by CoMon is relatively coarse grain for our purpose. Therefore, we developed a system called Platinum for monitoring PlanetLab nodes and obtain resource usage data at configurable intervals. It collects resource usage information for various resources such as CPU, memory and network bandwidth.

The Platinum system collects data about resource utilization at each monitored node by probing its *SliceStat* [22] data at a periodic *monitoring interval* (which is set to 15 seconds in our experiments). For a monitored node, it collects the following data during each probe and maintains the average and standard deviation values over a sliding *data aggregation window* of 5 minutes which is moved at each monitoring interval.

1) CPU usage (measured in MHz)
2) Physical and virtual memory usage (in KB)
3) Average sending and receiving bandwidth usage over the past 1, 5, and 15 minutes intervals (measured in KBps).

On each probe to a node, around 4 KB data is received on average. Thus, with 15 seconds monitoring interval, for monitoring 400 nodes, the bandwidth requirement of this system would be approximately 106 KBps. Moreover, the usage data and related statistics, around 800 bytes per probe for a node, are stored in stable storage for offline data analysis.

We compute the available (i.e. unused) resource capacity at a node for a particular resource type as the difference between the node's intrinsic resource capacity and the total usage for that resource for all the slices running on that node. For example, currently available CPU capacity for a node is computed by subtracting the current CPU usage of all slices measured in MHz from the node's intrinsic CPU capacity, which is measured as the product of the number of cores and the CPU speed. A resource requirement specifies the minimum resource capacity that needs to be available on a node. For example, a 2GHz CPU requirement indicates that the available resource capacity on the selected node must always be at least 2GHz. In case of bandwidth, the requirements are specified in terms of usage rather than the available capacity. For example, 1MBps bandwidth requirement indicates that the sum of the 1-minute average bandwidth usage of all the slices must always be less than 1MBps. The requirements for network bandwidth were expressed in this way because for a significant fraction of the nodes the information about the total network bandwidth was not available.

III. Experiment Datasets

In this section we describe the datasets we collected for our study. We monitored a pool of PlanetLab nodes and collected traces of resource utilization data at different times from June 2009 to January 2012. We observed a total of 390 randomly selected PlanetLab nodes over this period. This pool represents more than a third of the PlanetLab nodes.

A. Capacity Distribution of Monitored Nodes

We first observed how the intrinsic resource capacities of the monitored nodes were distributed to understand the resource capacity variations across the observed nodes. Figures 1 and 2 show the distribution of per node CPU and memory capacities, respectively. The CPU capacity shown is the total intrinsic CPU capacity of a node, calculated as the product of number of cores and clock frequency per core of that node. As shown in Figure 1, a large fraction of monitored nodes (more than 60%) had CPU capacity in the range of 4 to 7 GHz. Only a small fraction of nodes had very high capacity (above 10 GHz). Compared to CPU capacity, node memory capacity had relatively less variation. A large fraction of nodes (more than 50%) had memory capacity between 3 to 3.5 GB. About 20% of the nodes had relatively low memory capacity (1-1.5 GB).

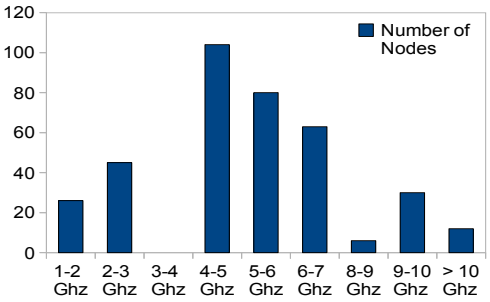

Fig. 1. Distribution of Node CPU Capacities

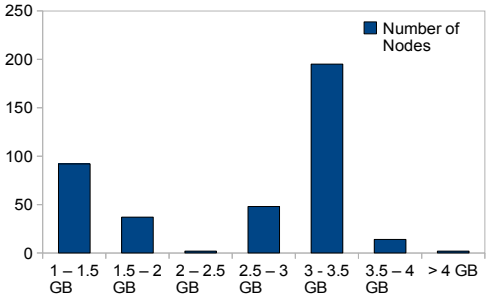

Fig. 2. Distribution of Node Memory Capacities

B. Collected Datasets

Table I describes the datasets that we collected for this study. These datasets were collected over durations ranging from 36 hours to 8 days. Our goal was to observe the resource availability of nodes over short durations (ranging from few hours to a week) to develop heuristics for selecting nodes, rather than characterizing long term usage patterns and distributions. We used datasets labeled 1, 2, and 3 to analyze the resource availability characteristics and build the models for node selection. We evaluated these models on the remaining datasets (Dataset 4-10) to verify the applicability of the developed models over different time periods. These

datasets belong to time periods both before as well as after the time periods of the collection of Datasets 1-3. The datasets used for model evaluations were at least 3.5 days long.

Dataset	Time and Duration	Number of Nodes
Datasets used to build the model		
Dataset-1	May 20-27, 2011 (7 days)	390
Dataset-2	Feb 11-12, 2011 (38 hours)	286
Dataset-3	Sep 1-2, 2010 (36 hours)	250
Datasets used for model evaluation		
Dataset-4	Jan 12-18, 2012 (6.5 days)	390
Dataset-5	Dec 9-13, 2011 (4.5 days)	390
Dataset-6	Oct 25-28, 2010 (3.5 days)	189
Dataset-7	Sep 17-22, 2010 (5.5 days)	210
Dataset-8	Mar 2-10, 2010 (8 days)	189
Dataset-9	Nov 8-12, 2009 (4 days)	303
Dataset-10	June 7-12, 2009 (5.5 days)	200

TABLE I
DATASETS AND THEIR OBSERVATION TIMES

C. Preliminary Analysis

We present here a preliminary analysis of the Datasets 1-3. We considered different resource capacity requirements, and at every probe interval in the trace (every 15 seconds) we observed the fraction of the nodes that met the given requirement. Table II shows the average and median values for this data for Datasets 1-3. A node was considered to meet the given requirement at any given point in the trace if its currently available capacity at that point was greater than the specified requirement. For CPU, we found that very few nodes satisfied the requirements above 4GHz. Therefore we considered available capacity requirements in the range of 1GHz to 4GHz. For memory, the requirement levels ranged from 0.5GB to 2GB. Although there were many nodes with total memory capacity of 3GB and above, the free memory capacity at such nodes usually tends to be less than 3GB due to memory usage of slices present on these nodes. Therefore, we did not consider requirements of 3GB and above, as very few nodes could satisfy these requirements. For network bandwidth, the requirements were expressed in terms of the sum of the usage of all the slices below some given thresholds, which were set in the range of 1.4MBps to 0.2MBps. Note that the requirement of 0.2MBps is more stringent, i.e. reflecting higher unused bandwidth, than 1.4MBps requirement.

The above data gives an indication of how many nodes can meet a given capacity requirement at a random point in time, however, an important question is how long a node continues to meet the given requirement after initially meeting the requirement. It would indicate how long an application can utilize a node selected from the pool of nodes satisfying the given requirement. Also, since the duration for which a node stays eligible would vary across nodes, it is important to select nodes that would remain eligible for long durations.

IV. CHARACTERISTICS OF NODE RESOURCE AVAILABILITY

In this section we present our study of node eligibility characteristics for the purpose of building models for node selection. We perform this study using the Datasets 1-3.

Requirement	Dataset-1		Dataset-2		Dataset-3	
	Avg.	Med.	Avg.	Med.	Avg.	Med.
1 GHz	0.35	0.42	0.61	0.70	0.42	0.47
2 GHz	0.28	0.36	0.54	0.60	0.36	0.40
3 GHz	0.21	0.29	0.43	0.52	0.22	0.29
4 GHz	0.15	0.20	0.32	0.38	0.13	0.16
0.5 GB	0.41	0.48	0.60	0.68	0.39	0.45
1 GB	0.38	0.41	0.51	0.57	0.30	0.37
2 GB	0.11	0.19	0.37	0.43	0.14	0.20
1.4MBps	0.56	0.69	0.70	0.72	0.47	0.50
1MBps	0.55	0.66	0.61	0.62	0.46	0.44
0.6MBps	0.54	0.63	0.54	0.53	0.45	0.43
0.2MBps	0.52	0.60	0.50	0.48	0.42	0.40

TABLE II
FRACTION OF NODES SATISFYING DIFFERENT RESOURCE CAPACITY REQUIREMENTS

A. Definition of Node Eligibility

We consider a node eligible for a given resource requirement based on its average available capacity for that type of resource over the current data aggregation window. The basic criterion for determining the eligibility of a node for a given requirement is as follows. If P is the average idle capacity on a node over the current data aggregation window and σ is its standard deviation, then for a given resource requirement D we select the node if it satisfies the following condition:

$$P - 2 * \sigma > D \tag{1}$$

A node is dropped from the eligibility set if the currently available capacity at that node falls below the resource requirement D, i.e. $P < D$. The criterion in equation (1) is used as a simple heuristic to select nodes by taking into account the fluctuations in their resource availability in recent past. Alternatively, one could use a simple criterion, such as $P > D$ to select nodes. However, in this case the nodes with available resource capacity fluctuating around the requirement level D would frequently enter and leave the eligibility set, remaining eligible for short durations only. Thus, to filter out such nodes we consider the standard deviation.

During a particular data trace, a node may enter and leave the eligibility set multiple times. We define the *eligibility period* of a node, which is denoted by τ, as the time between its entry in the eligibility set and its subsequent departure from the set. The *ineligibility period*, which is denoted by δ, is defined as the duration between two consecutive eligibility periods of a node. For the purpose of developing node selection techniques, we study how node eligibility periods are distributed. We first perform this study by considering the requirements of different resource types separately. Then we study node eligibility periods when requirements for multiple resources are considered together.

B. Distribution of Eligibility and Ineligibility Periods

An important question that we wanted to address is to determine how long a randomly selected node is likely to remain eligible for a given requirement. For this purpose, we

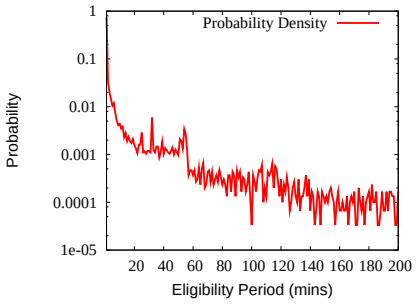

Fig. 3. PDF of Eligibility Periods for 2GHz CPU Requirement

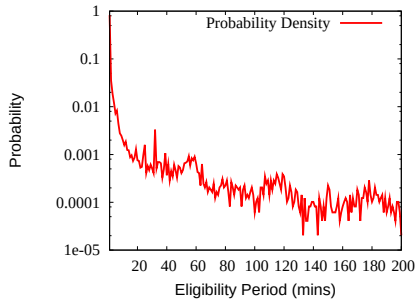

Fig. 4. PDF of Eligibility Periods for 1GB Memory Requirement

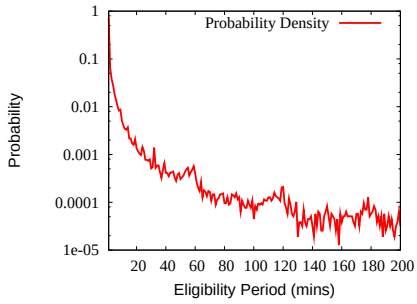

Fig. 5. PDF of Eligibility Periods for 1MBps Bandwidth Requirement

	50 percentile	70 percentile	90 percentile
1 GHz	1 - 3 mins	1 - 6 mins	10 - 17 mins
2 GHz	1 - 4 mins	1 - 6 mins	12 - 26 mins
3 GHz	2 - 4 mins	3 - 8 mins	11 - 69 mins
4 GHz	2 - 5 mins	5 - 26 mins	11 - 113 mins
0.5 GB	1 - 2 mins	1 - 4 mins	8 - 20 mins
1 GB	1 - 3 mins	1 - 4 mins	10 - 22 mins
2 GB	2 - 4 mins	2 - 9 mins	18 - 28 mins
1.4MBps	1 - 2 mins	1 - 3 mins	29 - 35 mins
1.0MBps	1 - 2 mins	1 - 3 mins	27 - 34 mins
0.6MBps	1 - 2 mins	1 - 4 mins	23 - 35 mins
0.2 MBps	2 - 3 mins	2 - 4 mins	22 - 36 mins

TABLE III
DISTRIBUTION OF ELIGIBILITY PERIODS FOR DATASET 1-3

observed the probability density of the eligibility periods. The expected duration for which a node selected randomly, at an arbitrary point in time, would remain eligible is half of the expected value of the eligibility period. We investigated eligibility period distributions separately for CPU, memory, and bandwidth requirements, for a range of resource requirement levels. As a representative example, we show, for Dataset-1, the probability density of eligibility periods for 2GHz CPU capacity, 1GB memory, and 1MBps bandwidth usage requirements in Figures 3, 4, and 5, respectively. Table III shows min-max range for 50, 70, and 90 percentile values of eligibility periods for Datasets 1-3. From this table and Figures 3, 4, and 5, we observe that the eligibility period values are typically small (50 and 70 percentile values are typically below 10 mins). Hence, there is need for further discrimination of nodes to eliminate nodes that are likely to remain eligible only for short durations.

We also investigated how ineligibility periods are distributed. The nodes which never became eligible for a given requirement were not considered in this measurement. Table IV shows the min-max range for probability mass of ineligibility period less than 15, 30 and 60 seconds across Datasets 1-3. We do not show these values separately for different requirement levels as we observed that these values did not vary significantly across different requirement levels.

We observe that with significant probability (greater than 0.5) the ineligibility period values are below 30 seconds. This indicates that there is high probability that a node is likely to become eligible again within 30 seconds after it becomes ineligible. Thus, if an application can tolerate such short ineligibility periods, it can make use of the node for a longer period.

	P[$\delta \leq$ 15 sec]	P[$\delta \leq$ 30 sec]	P[$\delta \leq$ 1 min]
CPU	0.35 - 0.52	0.50 - 0.70	0.71 - 0.84
Memory	0.42 - 0.57	0.60 - 0.79	0.80 - 0.88
Network Bandwidth	0.40 - 0.55	0.65 - 0.75	0.79 - 0.87

TABLE IV
DISTRIBUTION OF INELIGIBILITY PERIODS FOR DATASETS 1-3

C. Node Eligibility Characteristics

We investigate here how the nodes behave in terms of their eligibility periods. The distributions shown in Figures 3, 4, and 5 are the distributions of individual eligibility periods for all the nodes. As a node may become eligible multiple times during the observation period, it may have multiple values for eligibility periods. Thus, in the above distributions, a node that became eligible more number of times contributed more samples than the nodes that became eligible less number of times. Moreover, a node that tends to stay eligible for a long time is likely to contribute less number of samples than a node that tends to stay eligible for a short time but enters the eligibility set frequently. Thus, to study the individual node behavior, we consider the median value of a node's eligibility periods as its representative eligibility period. We denote the *median eligibility period* of a node by μ. In Figures 6, 7, and 8, as a representative example we show the cumulative distribution (CDF) of nodes' μ values for Dataset 1 for CPU, memory, and bandwidth requirements, respectively. Table V presents the statistics for node median eligibility periods and set sizes for this dataset.

In this table, the *unique nodes* column gives the number of nodes that became eligible during the entire duration of the observation for the corresponding resource requirement. The statistics given in the tables for a specific requirement correspond to the unique nodes for that requirement. For example, in case of 2GHz CPU capacity requirement the average eligibility period of 152 minutes is the average of 218 nodes' median eligibility periods. Similarly, in Figure 6 the CDF for 2GHZ CPU requirement is the distribution of 218 nodes' median eligibility periods. We observe that typically the median values for the nodes' median eligibility periods

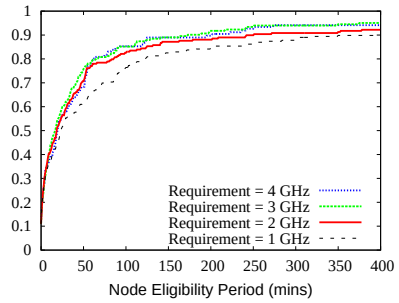

Fig. 6. CDF of Node Eligibility Periods for CPU Requirements for Dataset 1

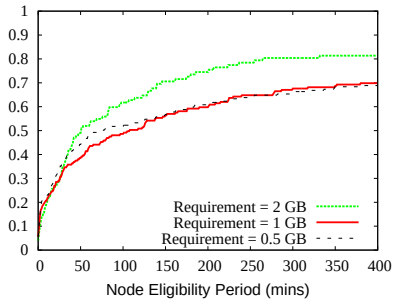

Fig. 7. CDF of Node Eligibility Periods for Memory Requirements for Dataset 1

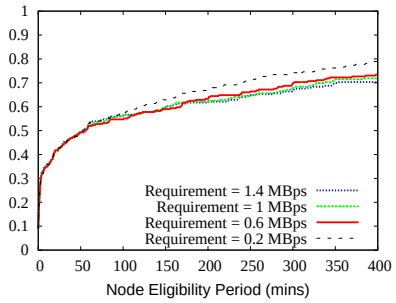

Fig. 8. CDF of Node Eligibility Periods for Network Bandwidth Requirements for Dataset 1

	Eligibility Period (μ) (minutes)			Unique Nodes	Eligibility Set Size	
	Avg	Median	Std Dev		Avg	Std Dev
CPU						
1GHz	218	36	432	246	75.1	37.86
2GHz	152	36	319	218	54.2	35
3GHz	129	30	265	182	42.5	25.1
4GHz	136	47	275	136	32.8	20.8
Memory						
0.5GB	554	147	923	199	109	51.3
1GB	540	147	919	179	91.8	44
2GB	251	73	338	102	22.1	11.9
Network bandwidth						
1.4MBps	529	103	761	256	168.8	67.8
1.0MBps	503	97	741	256	168.7	67.8
0.6MBps	450	91	651	256	168.2	67.8
0.2MBps	351	84	454	256	167	67.7

TABLE V
NODE MEDIAN ELIGIBILITY PERIOD AND SET SIZE FOR DATASET 1

tend to be always less than the average values. The standard deviation also tends to be high, comparable to the average values (coefficient of variation is between 0.89 to 2.17). This indicates that some nodes tend to exhibit significantly large eligibility periods. From these statistics, we can observe that, for all resource types, generally the median and average values for node median eligibility period (μ) tend to decrease with increase in the resource requirement levels. However, this can not be taken as a rule as one can observe that sometimes increase in the level of a requirement may lead to increase in the median or average value of μ, as in the case of 3GHz and 4GHz requirements. We find that this is because the nodes that become eligible for a lower requirement level for short durations may not qualify for a higher requirement level. Thus, sometimes in case of higher requirement levels fewer nodes may become eligible but they may remain eligible for longer durations. For all resource types and requirements, the eligibility set size always decreases with increase in the requirement level. This is to be expected as the nodes that become eligible for a higher requirement level must also be eligible for a lower requirement level. We find that typically the eligibility periods are smaller for CPU requirements as compared to memory and network bandwidth requirement. This indicates that the available capacity tends to fluctuate more for CPU than for memory and network bandwidth.

D. Relation between Node Eligibility and Ineligibility Periods

Another aspect that we investigated for characterizing nodes is how their eligibility periods and ineligibility periods are related. For this purpose, we measure the average eligibility period and average ineligibility period of a node. The ratio of a node's average eligibility period to its average ineligibility period gives the relative availability of the node. Figures 9, 10, and 11 show, separately for CPU, memory, and bandwidth, the scatter graphs of average eligibility and ineligibility periods for Dataset-1. We considered the mid-range values for requirement levels; CPU requirement of 2GHz, memory requirement of 1GB, and bandwidth requirement of 1MBps. We observe that the nodes can be classified in three groups based on their eligibility and ineligibility periods. The first group contains nodes which tend to have large ineligibility periods and small eligibility periods. The second group of nodes tend to have small eligibility and ineligibility periods, indicating that they frequently enter and leave the eligibility set. The third group of nodes, which we consider as 'high-quality' nodes, tend to have small ineligibility periods and large eligibility periods.

The scatter graph for CPU shown in Figure 9 shows that the nodes with large average eligibility periods tend to have small average ineligibility periods. There is also a large number of nodes which tend to have small eligibility periods but with large variations in ineligibility periods. This indicates that some nodes tend to have more frequent fluctuations in their CPU usage. In contrast the scatter graph, in Figure 10, for memory shows that the nodes with large eligibility periods can also have relatively large ineligibility periods. This indicates less frequent variations in memory utilization. In case of network bandwidth usage the ineligibility periods tend to be of short durations. This can be explained based on the bursty nature of network usage.

To differentiate high quality nodes from other nodes we observe the ratio of a node's average eligibility period to its average ineligibility period. We find that the nodes with average eligibility period above certain threshold tend to have higher value for this ratio than the nodes with average eligibility period below the threshold. We examined this ratio for different threshold values ranging from 5 to 60 minutes. We show the results of this study for Datasets-1 in Table VI. In this table, r_l denotes the median value of this ratio for nodes

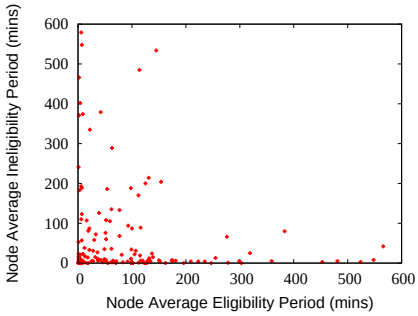

Fig. 9. Node Eligibility and Ineligibility Periods for 2GHz CPU Requirement

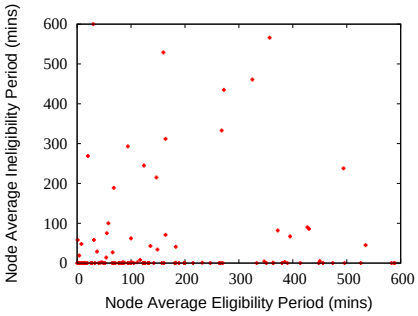

Fig. 10. Node Eligibility and Ineligibility Periods for 1GB Memory Requirement

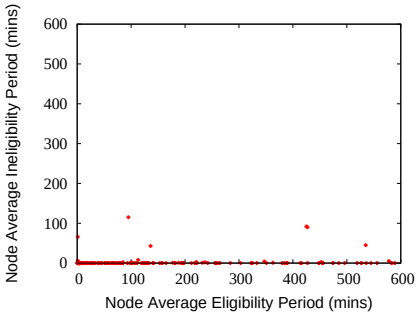

Fig. 11. Node Eligibility and Ineligibility Periods for Bandwidth Usage 1MBps

	CPU (2GHz)		Memory (1GB)		Bandwidth (1MBps)	
threshold	r_l	r_h	r_l	r_h	r_l	r_h
5 mins	0.05	4.7	0.67	94	0.82	261.1
10 mins	0.15	13	0.9	129.6	3.4	282
20 mins	0.16	22	2	133	4.1	315.1
30 mins	0.89	48	2.2	155.5	6.5	348.2
60 mins	1.37	101	3.8	193.6	20.5	347

TABLE VI
NODE DISCRIMINATION BASED ON RELATION BETWEEN NODE ELIGIBILITY AND INELIGIBILITY PERIODS

with average eligibility periods below the given threshold, and r_h denotes the median ratio for nodes above the threshold. For example, in case of 2GHz CPU capacity requirement with the threshold of 20 minutes, the nodes with average eligibility period more than 20 minutes have the median value of 22 for this ratio, and for the nodes below this threshold the median value of the ratio is 0.16.

We find that for network bandwidth, with threshold of 5 minutes one can find nodes with very large value (around 260) of r_h, i.e. their average eligibility periods are significantly larger than their average ineligibility periods. For memory one can find high quality nodes with threshold of 10 minutes. For CPU, with the threshold of 20 minutes one can find nodes that have r_h value 22, i.e. with 50% probability a selected node will have average eligibility period at least 22 times higher than its average ineligibility period. Similar trends were observed for the other two datasets.

The observations presented in this study indicate that there is a need to discriminate amongst nodes to select high quality nodes for a given requirement.

E. Residual Eligibility Period

In order to select high quality nodes, we wanted to investigate whether the recent eligibility period profile of a node can indicate the likelihood that the node would remain eligible for a long duration. As discussed earlier, Figures 3, 4, and 5 show that with significant probability the eligibility periods tend be of short durations. This motivates the need for observing how long a node has remained eligible before selecting it for inclusion in the eligibility set. We define $\lambda(t)$ as the *residual eligibility period* of a node. It is measured as the residual eligibility period after the node has been eligible for a duration of t units. This means, for a residual eligibility period $\lambda(t) = \tau - t$, i.e. $\lambda(0) = \tau$.

Table VII shows the average residual period for different values of threshold t for Dataset-1. We can observe that for all threshold values, the average residual eligibility period is much larger than the average eligibility period τ. This indicates that if a node's eligibility period exceeds certain threshold then it is likely to remain eligible for a long duration. We also observe that the average residual eligibility period increases with increase in threshold values. This indicates that the distribution of eligibility periods is not memoryless.

F. Multi-dimensional Resource Requirements

We present here characteristics of resource availability when multiple resource requirements are considered together. In this case, a node is considered eligible as long it satisfies all the specified resource requirements. A node is dropped from the eligibility set when it fails to satisfy any of the resource requirements. We consider three multi-dimensional conjoined requirements by taking low, medium and high requirement levels for each of the resource types (CPU, memory and bandwidth). Thus, in this study we consider following requirements as the representative cases.

1) low (1GHz CPU, 0.5GB memory, 1MBps bandwidth);
2) medium (2GHz CPU, 1GB memory, 0.6MBps bandwidth);
3) high (4GHz CPU, 2GB Memory, 0.2MBps bandwidth).

The node median eligibility periods for these requirements are given in Table VIII for Dataset-1.

We wanted to investigate which resource dominates the node eligibility when multiple resource requirements are considered together. For this purpose, we observed the number of times a node is dropped from the eligibility set or it is not added to the set because the requirement for one resource type is not satisfied but the requirements for other two resource types are satisfied. In this table the *resource unavailability* indicates the percentage of the times this occurs for the given resource type. This data gives indication regarding the dominating resource type in case of multi-dimensional requirements. For example, in case of low requirement (1GHz CPU, 0.5GB memory, 1MBps bandwidth), the value 60.2 for CPU resource indicates that 60.2% of times a node was dropped from

		t = 5 mins		t = 10 mins		t = 15 mins	
Requirement	Avg. $\lambda(0)$	P[$\tau \leq t$]	Avg. $\lambda(t)$	P[$\tau \leq t$]	Avg. $\lambda(t)$	P[$\tau \leq t$]	Avg. $\lambda(t)$
1GHz	10.59	0.84	60.8	0.89	83.2	0.90	93.2
2GHz	10.55	0.85	57.2	0.88	73.7	0.89	82.1
3GHz	10.52	0.86	60.39	0.90	77.7	0.91	86.3
4GHz	11.7	0.88	78	0.92	105.7	0.93	117.2
0.5GB	14.6	0.84	78.9	0.88	96.4	0.89	104.5
1GB	14.9	0.85	85.35	0.89	103.7	0.91	111.93
2GB	17.2	0.84	92.1	0.88	112.2	0.90	121.8
1.4MBps	10.64	0.85	59.4	0.90	87.2	0.92	100.7
1MBps	10.63	0.85	59.3	0.90	87.2	0.92	100.6
0.6MBps	10.64	0.85	59.2	0.90	86.7	0.92	100.1
0.2MBps	10.52	0.86	57.8	0.90	84.3	0.92	97.4

TABLE VII
RESIDUAL ELIGIBILITY PERIOD (IN MINUTES) FOR DIFFERENT THRESHOLDS

	Requirement Levels		
	low	medium	high
Node Median Eligibility Period			
Avg (Med)	68 (35)	49.5 (25)	21.9 (9)
Std. dev	8.98	6.4	12.1
Unique nodes	125	118	31
Eligibility Set Size			
Avg	113	83.5	25.1
Std. dev	48.9	39.4	16.8
Resource Unavailability (%)			
CPU	60.2	65.6	16.4
memory	29	28	77.7
bandwidth	10.8	6.4	4.9

TABLE VIII
NODE ELIGIBILITY FOR MULTI-DIMENSIONAL REQUIREMENTS

the eligibility set or was not added to the set because it did not satisfy CPU requirement but the requirements for memory and bandwidth were satisfied. From this data we can observe that typically the CPU requirements dominates the node eligibility for low and medium level of requirements. However, the memory requirements starts dominating for high level of requirements i.e. 2GB and above. We can see that the bandwidth requirements were the least dominating compared to CPU and memory.

G. Summary of Resource Availability Characteristics

We summarize here the important observations regarding node eligibility characteristics. These observations guide us in developing the node selection techniques presented in the next section.

- The eligibility periods (τ) tend to have long-tailed distributions. The nodes show wide variations in terms of their eligibility periods and some nodes tend to show significantly large eligibility periods.
- The ineligibility periods (δ) tend to be of short durations (typically 50% values are less than 30 seconds), indicating that if an application can tolerate such short ineligibility periods it may be able to use a node for a longer duration without relocating. However, it is important to understand for what fraction of its residency time at a node its resource requirement is satisfied. We address this question in the next section.
- Nodes can be distinguished based on their eligibility and ineligibility periods. Some nodes tend to have very small ineligibility periods and large eligibility periods. This indicates that there is a need to discriminate among nodes to identify high quality nodes.
- The nodes that remain eligible for certain duration tend to show long average residual eligibility periods. Thus, the nodes can be distinguished by observing how long they have remained eligible.
- When requirements for multiple resource types are considered together, the node eligibility is typically dominated by CPU requirements, but memory resource starts dominating in case of 2GB or higher requirements. Bandwidth requirement is the least dominating factor compared to CPU and memory.

V. HEURISTICS FOR NODE SELECTION AND APPLICATION PLACEMENT

We present here the techniques that we developed for selecting nodes which are likely to remain eligible for long durations for a given requirement. We utilize the node eligibility characteristics presented in the previous section to build heuristics for selecting nodes. We evaluate these techniques on Datasets 4-10 and demonstrate their benefits in selecting nodes that remain eligible for longer durations compared to the basic selection method presented in Section IV-A.

A. Profiling Based Node Selection

In the previous section, we discussed the need to discriminate nodes to identify the 'high quality' nodes which are likely to remain eligible for a long duration. For selecting such high quality nodes, we investigated the methods for discriminating nodes based on their eligibility periods. Based on these methods, we develop profiling approaches for selecting nodes for a given requirement.

The first aspect we use for discriminating nodes is their current eligibility periods. In the previous section we showed that if a node remains eligible for a certain duration then its residual eligibility period tends to be large (refer Table VII). Thus, by observing the node eligibility periods for a short duration, such as 5 minutes, one can find nodes that are likely to remain eligible for long durations. Another aspect that we consider for selecting nodes is their past eligibility periods for

a given requirement. For this purpose, we determine the conditional probability that the node's eligibility period exceeds a given threshold provided that its previous eligibility period exceeded that threshold. We measured this for various resource requirements and resource types for Datasets 1-3. We observed that for the threshold value of 30 minutes, with probability greater than 0.3 the eligibility period value exceeded the threshold given that the previous eligibility period exceeded the threshold. For threshold of 60 minutes, this conditional probability was in the range 0.4 to 0.6. This means that selecting nodes based on this criterion can give at least 30% probability of a selected node remaining eligible for at least 30 minutes.

Based on the above two criteria, we develop a profiling approach for node selection as follows. We use the basic criterion given in equation (1), in Section IV-A, for determining a node's eligibility for a given requirement. We then maintain a subset of these nodes as the *profiled set*. We add a node to the profiled set if it satisfies any of the following two conditions:

1) The node has remained eligible for a certain duration (set to 5 minutes in our experiments, using the data in Table VII as the basis).
2) The previous eligibility period of the node is greater than certain threshold (we set it to 30 minutes).

We measured the duration for which a node stays in the profiled set as the *profiled eligibility period* of that node. We observed the distribution of the profiled nodes' median eligibility periods. For the purpose of illustrating the performance benefit of the profiling approach over the basic approach, we show in Table IX the statistics for profiled node median eligibility periods for Dataset-1. We then evaluate the general applicability of this approach using Datasets 4-10. We can compare the statistics shown in Table IX to those shown in Table V for basic eligibility periods. We can observe that the profiling approach gives longer node median eligibility periods than the basic approach and the improvement ranges from a factor of 6 to 20. As expected, the average size of the profiled set is smaller than the basic eligibility set. Thus, the pool of nodes selected by the profiling approach is smaller but it contains "high quality" nodes which remain eligible for long durations.

To measure the benefit of profiling across other datasets (Datasets 4-10), for each dataset we compared the median values of the node median eligibility period (μ) using profiling-based selection and basic selection method. We measured the *profiling improvement factor (f_p)* as the ratio of the median value of μ obtained with profiling to the median value of μ obtained with basic selection method. For example, we can see from Table IX and Table V that f_p for 1GHz CPU requirement is equal to 19.6. We do not measure how eligibility period improves per node, because the profiling approach selects a smaller set of nodes and hence many of the nodes that get selected in basic selection method do not get selected in profiling method. Therefore the per node improvement can not be measured for all the nodes. To observe the effect on

	Profiled Eligibility Period (minutes)			Unique Nodes	Profiled Eligibility Set Size	
	Avg	Median	Std Dev		Avg	Std Dev
CPU						
1 GHz	1280	707	2505	177	61	27.6
2 GHz	2016	806	2807	142	54	25.4
3 GHz	1335	554	2050	105	35	18.5
4 GHz	1505	645	2156	80	28	15.6
Memory						
0.5GB	2207	1193	2661	184	72	30.6
1GB	2196	1193	2697	166	62	27.6
2GB	2028	1207	2487	63	21	9.37
Network bandwidth						
1.4MBps	2103	918	2810	247	94	37.7
1.0MBps	2075	918	2788	247	95	38.8
0.6MBps	2054	909	2800	247	94	37.7
0.2MBps	1763	829	2417	247	94	40.3

TABLE IX
PROFILED NODE MEDIAN ELIGIBILITY PERIODS AND SET SIZES FOR DATASET-1

	f_p		Set size reduction(%)	
	min	max	min	max
CPU				
1 Ghz	5.01	10.11	50	59
2 Ghz	7.89	18.01	50	55
3 Ghz	8.48	29.8	30	66
4 Ghz	7.21	11.7	50	58
Memory				
0.5GB	1.15	5.81	20	82
1 GB	1.15	8.42	30	83
2 GB	2.37	7.79	50	79
Network Bandwidth				
1.4Mbp	1.12	5.12	38	53
1Mbps	1.51	5.09	37	60
0.6Mbps	1.21	4.47	40	57
0.2Mbps	1.30	5.18	41	60

TABLE X
IMPROVEMENT ACHIEVED USING PROFILING APPROACH FOR DATASETS 4-10

eligibility set size due to profiling, we measure the percentage reduction in the set size. This is measured as the percentage decrease in the median value of the set size from basic selection to profiling-based selection. We measured the f_p and set size reduction for each of the seven datasets. Table X shows the min and max value for the above two measures across Datasets 4-10. From this data, we can see that the profiling-based node selection approach typically gives longer eligibility periods compared to the basic selection approach, confirming our earlier observation.

B. Benefit of Ineligibility Toleration

We discussed earlier the motivation for tolerating short ineligibility periods. We evaluated the benefit of this approach by observing how the eligibility periods increase by tolerating short ineligibility periods of duration Δ. For this, we used the same criterion as equation (1) to select the nodes, however, a node is dropped from the eligibility set only if the currently available resource capacity at that node remains below the specified requirement for duration greater than Δ.

Table XI shows the node eligibility period statistics using this approach for Dataset-1 to show the relative benefits in

	Node Median Eligibility Period (μ) (minutes)					
	$\Delta = 15$ secs		$\Delta = 30$ secs		$\Delta = 1$ min	
	Avg (Med)	ρ	Avg (Med)	ρ	Avg (Med)	ρ
GHz	CPU					
1	813 (104)	0.99	888 (112)	0.98	933 (116)	0.95
2	539 (50)	0.99	583 (52)	0.97	627 (55)	0.93
3	481 (40)	0.99	525 (42)	0.97	539 (44)	0.93
4	511 (63)	0.99	560 (67)	0.98	592 (72)	0.94
GB	Memory					
0.5	895 (406)	0.99	923 (430)	0.96	941 (449)	0.93
1	875 (367)	0.99	919 (389)	0.94	927 (417)	0.91
2	459 (196)	0.99	478 (202)	0.94	481 (209)	0.92
MBps	Network bandwidth					
1.4	819 (284)	0.99	837 (296)	0.98	860 (317)	0.98
1.0	793 (277)	0.99	816 (283)	0.98	848 (306)	0.98
0.6	740 (274)	0.99	801 (276)	0.98	831 (299)	0.97
0.2	731 (269)	0.99	794 (273)	0.98	822 (293)	0.97

TABLE XI
NODE MEDIAN ELIGIBILITY PERIODS (μ) WITH TOLERATION FOR DATASET-1

	f_t			ρ (median)
	25 percentile	50 percentile	75 percentile	
	CPU			
1 GHz	1.18	2.75	9.54	0.97
2 GHz	1.06	1.42	5.33	0.98
3 GHz	1.05	1.29	4.01	0.96
4 GHz	1.04	1.25	4.00	0.92
	Memory			
0.5 GB	1.87	3.70	12.27	0.97
1 GB	1.72	3.47	9.89	0.96
2 GB	1.33	2.57	7.74	0.93
	Network Bandwidth			
1.4 MBps	1.43	2.31	8.82	0.98
1 MBps	1.37	2.19	7.79	0.97
0.6 MBps	1.23	2.09	6.36	0.96
0.2 MBps	1.21	2.01	6.07	0.96

TABLE XII
IMPROVEMENT ACHIEVED USING TOLERATION APPROACH WITH $\Delta = 30$ SECONDS

comparison to the data shown in Table V for this dataset for the basic approach. Note here that the eligibility period in this case indicates the duration for which an application can use a node by tolerating ineligibility periods of durations up to Δ. The *goodness fraction*, denoted by ρ, is the fraction of the eligibility period for which the node satisfied the given resource requirement. For example, a ρ value of 0.99 indicates that the node met the given requirement for 99% of the eligibility period duration and the remaining 1% amounts to the ineligibility periods of duration less than or equal to Δ. We show in this table the average and median values for node median eligibility periods (μ) and the median value of ρ. We observe that the eligibility period values increase significantly using the toleration approach. For example, in case of 1GHz CPU capacity requirement the average μ value increased from 218 minutes (refer Table V) to 813 minutes with goodness fraction of 0.99 with toleration of just 15 seconds of ineligibility periods.

We evaluated this approach using Datasets 4-10 to validate its general applicability. For this purpose, we observed for each node, the ratio of its median eligibility period (μ) with toleration to the median eligibility period without toleration. We refer this ratio as *toleration improvement factor(f_t)*. We observed this ratio for all nodes across Datasets 4-10 to verify whether the toleration approach is beneficial in case of these datasets as well. Table XII shows the 25, 50 and 75 percentile values for this ratio. We also show the median value of *goodness fraction* (ρ) in this table. In this observation, the Δ value was set to 30 seconds. We can observe that the approach of tolerating short ineligibility periods gives longer eligibility periods. Thus, if an application can tolerate short ineligibility periods it can continue to make use of a node for a long duration without the need to relocate.

C. Utilization of Node Selection Techniques

The node selection techniques presented in this section do not require any global platform-wide monitoring service and they are also not intended for building any central scheduling and resource allocation mechanism for the underlying platform. A user can utilize these techniques for selecting nodes for deploying an application by performing short term monitoring of a set of potential nodes to identify high quality nodes. For this purpose, the users can themselves run the Platinum monitoring system in their environment. The profiling approach requires resource usage monitoring for only about 30 minutes.

The techniques presented in this section can be used for selecting nodes for deploying both applications and services. For long running services, dynamic relocation is required as the placement decisions made at the time of the initial deployment can become ill-suited after some time due to the fluctuations in resource availability at the selected nodes. However, the node selection techniques presented above can be used to select high quality nodes for replica placement to reduce the number of migrations. Moreover, if the relocation cost is high then the approach of toleration can be used to avoid unnecessary migrations. We have used these techniques in our work on building resource-aware migratory services [24] and autonomically scalable services [21] in such environments.

VI. NODE-LEVEL RESOURCE CAPACITY PREDICTION

We address here the problem of how to predict for a given resource at a node the amount of its idle capacity that is likely to be available (i.e. not used by other users) in the near future with some given probability. The prediction of available capacity can be useful for an application to estimate how much additional capacity is likely to be available beyond its resource requirements. Specifically, we address the problem that for some given confidence level C, how to predict the resource capacity R for a particular resource type such that the available capacity over some period in the near future is at least R with probability C.

We present here an online model for prediction of available resource capacity. On PlanetLab we observed that the fluctuations in the available resource capacities depend on the node's load conditions. This requires a dynamic prediction model that takes into account the node's load conditions. Our

prediction method is based on observing the fluctuations in the available resource capacities over time. To characterize such fluctuations, for each resource type we observe the average available resource capacity R_{w_o} over some period, called *observation period (w_o)*, and the average available resource capacity R_{w_p} over a period in the immediate future, called *prediction period (w_p)*. We define the *capacity modulation ratio (θ)* as

$$\theta = R_{w_p}/R_{w_o} \tag{2}$$

A capacity modulation ratio greater than 1 indicates increase in the available resource capacity by some fraction, and a value less than 1 indicates a decrease. $P[\theta \geq x]$ is the probability that the average available resource capacity over the next prediction period w_p is at least $R_{w_o} \cdot x$. Therefore, to predict the fraction of the available resource capacity that is likely to be available with a specified confidence level C, we determine x such that $P[\theta \geq x] = C$. We observe that while the available resource capacity itself may change significantly over short durations, such changes (that is the θ values) are statistically predictable over durations of several minutes (in range of 30-60 minutes). Therefore, our dynamic model for resource capacity prediction is based on observing the history of θ values over some period, called *history window (w_h)*. Our prediction model estimates the cumulative distribution (CDF) of the θ values observed over a sliding window of period w_h and calculates the value x for some given confidence level C, such that $P[\theta \geq x] = C$. This value is used to estimate the resource capacity for the next prediction period. For example, suppose that x_i is the value calculated, as described above, at the ith prediction cycle for CPU resource. Let p_i be the observed average available CPU capacity over the immediately preceding observation period of w_o duration at the ith prediction cycle. The predicted CPU capacity P_i for the following prediction period w_p is estimated as:

$$P_i = p_i \cdot x_i \tag{3}$$

The goodness of the prediction model can be determined by considering the ratio of resource capacity observed to be available in a given interval to the capacity predicted for that interval. We call it the *prediction ratio (ϕ)*. A value of ϕ close to 1 indicates that the observed capacity is close to the predicted capacity, whereas values higher or lower than 1 indicate underprediction and overprediction, respectively. Since, in equation (3), x_i is chosen such that $P[\theta \geq x_i] = C$, we expect that the resource capacity observed to be available during the immediately following prediction period w_p is at least $p_i \cdot x_i$ with probability C. Therefore, we expect that $P[\phi \geq 1] = C$. Based on this observation, the goodness of the prediction model can be evaluated based on the value of ϕ at which this required confidence C is achieved.

We evaluated the impact of the different parameters – w_h, w_o, w_p, and C – on the performance of the prediction model for Datasets 1-3. An important question is how to choose the values for these parameters. For the confidence level parameter C, a high value of C would result in underprediction, and a low value would cause overprediction.

To determine the impact of the lengths of w_o and w_p on the prediction model, we observed ϕ for these datasets with the w_o values of 1, 3, and 5 minutes, and w_p values of 0.5, 1, and 1.5 minutes. Since a ϕ value close to 1 is desirable, we determined the goodness of prediction in terms of the probability mass of ϕ values between 0.9 and 1.1. For CPU, network bandwidth, and memory, we observed that for all collected datasets w_o and w_p values of 1 minute give better prediction performance than other values, However, the sensitivity of the prediction performance to w_o and w_p parameter values was marginal.

To determine the impact of the history window (w_h) size, we observed the value of ϕ at which the required confidence level C is achieved, i.e. x such that $P[\phi \geq x] = C$. We evaluated this for w_h values ranging from 10 to 60 minutes. We observed that for all w_h values above 20 minutes this was achieved at ϕ value of approximately 0.95. This data is shown for Dataset-2 in Table XIII as a representative example. That means the model is overpredicting by about 5%, i.e. at least 95% of the predicted capacity would be available with probability C. For w_h values of 10 and 20, the required confidence level was achieved for ϕ values of 0.9 and 0.92, respectively. We also observed that w_h value of 60 minutes performs better, but only with marginal (about 4%) improvements over w_h values of 30, 40, and 50 minutes. The prediction performance for network bandwidth and memory showed similar trends.

For evaluating the effect of C, we considered the amount of underprediction in terms of the probability of ϕ being greater than 1.5. Since for all resource types, we achieve the required confidence level with approximately 5% error, the amount of overprediction is decided by the C value we set. As shown in Table XIII, for CPU resource we can observe that the amount of underprediction increases with increase in confidence level. However, for network bandwidth and memory, confidence level had relatively less impact on underprediction, since for all C values the underprediction was limited to 50%. This behavior occurs because typically the CPU usage can fluctuate significantly compared to memory and network bandwidth. The memory usage is relatively stable i.e. fluctuates less over a short time compared to CPU usage. In case of network bandwidth, the usage tends to be bursty in nature, so when there is no burst of data communication the bandwidth usage tends to be relatively stable. Therefore, for CPU capacity one can bias the prediction model towards underprediction or overprediction by choosing the confidence level whereas, for network bandwidth and memory setting higher confidence level is more desirable.

We evaluated the accuracy of our prediction model across Datasets 4-10 for various confidence levels. We fixed the model parameter values as $w_h = 30$ minutes and $w_o = w_p = 1$ minute. For each of the seven datasets, we observed the probability P[$\phi \leq 0.95$] for various confidence levels for CPU, memory and bandwidth. Table XIV shows min and max values for P[$\phi \leq 0.95$] across the seven datasets. We do not show this data separately for CPU, memory and bandwidth as we observed that for any given confidence level the value

w_h	C	P[$\phi \leq 0.95$]	P[$\phi \leq 1.5$]	P[$0.95 < \phi \leq 1.5$]
CPU				
30	70	0.32	0.88	0.56
60	70	0.31	0.89	0.57
30	80	0.22	0.76	0.54
60	80	0.21	0.75	0.54
30	90	0.13	0.66	0.53
60	90	0.11	0.66	0.53
Memory				
30	70	0.32	0.99	0.67
60	70	0.30	0.99	0.69
30	80	0.22	0.98	0.76
60	80	0.21	0.98	0.77
30	90	0.12	0.97	0.84
60	90	0.11	0.97	0.85
Network Bandwidth				
30	70	0.32	1.0	0.68
60	70	0.31	1.0	0.69
30	80	0.22	1.0	0.78
60	80	0.21	1.0	0.79
30	90	0.12	0.98	0.86
60	90	0.11	0.99	0.88

TABLE XIII
PREDICTION PERFORMANCE FOR VARIOUS VALUES OF w_h AND C

of P[$\phi \leq 0.95$] did not change significantly across different resource types. This can also be seen from Table XIII. We can observe from Table XIV that the required confidence is achieved approximately at prediction ratio of 0.95 for any specified confidence level, confirming our earlier observation.

confidence level C	P[$\phi \leq 0.95$]	
	min	max
70	0.273	0.324
75	0.249	0.259
80	0.191	0.227
85	0.139	0.165
90	0.110	0.134
95	0.067	0.071

TABLE XIV
PREDICTION ACCURACY ACROSS DATASETS 4-10

Based on these observations, we make the following conclusions:

- For any given confidence level C, our prediction model predicts the resource capacity that is likely to be available (i.e. unused) over the next interval with probability C with 95% accuracy i.e. the available resource capacity is at least 95% of the predicted capacity with probability C.
- For CPU capacity higher confidence level results in higher underprediction. However, for network bandwidth and memory, confidence level has relatively less impact on amount of underprediction. Therefore, it is desirable to set high confidence level for memory and network bandwidth.
- The sensitivity of the prediction performance to w_o and w_p values in the range of 1 to 5 minutes is marginal. Similarly, we found that for w_h values higher than 20 minutes the impact of the parameter values is marginal.

We have used this prediction model in our work on building autonomically scalable services over the PlanetLab platform [21]. The focus of that work was on dynamically adding and removing service replicas based on the estimated request handling capacities of service replicas and the observed load. In that work, we used the above prediction model for predicting the resource capacities likely to be available at a replica's host over the next 1-minute interval. A service replica periodically predicts the resource capacity likely to be available for different resource types such as CPU, memory and network bandwidth. It also monitors its request load and estimate the average resource requirement for processing a request. The predicted resource capacities and the estimated average per-request resource demands are then used to estimate the service capacity, i.e. the request handling capacity of the replica over the next 1-minute interval. Using the operational analysis technique, the replica identifies the bottleneck resource and then estimates the maximum number of requests that it can serve over the next interval. Based on the estimated request handling capacities of the replicas, the service deployment framework dynamically adds or removes the service replicas to maintain enough aggregate service capacity to handle the request load. The estimated request handling capacities of the replicas are also used for load balancing of requests across replicas. We used the node selection techniques described earlier for dynamic placement of service replicas.

VII. RELATED WORK

Characterization of resource usage and availability in cooperatively pooled shared computing platforms has been studied extensively in the past. Available capacity estimation based on idle machine availability in Desktop Grid environments have been studied in past [18], [17]. A great deal of previous work [19], [12], [13], [10], [11], [7] has provided characterization and statistical models for resource availability in Desktop Grid systems and public resource computing systems such as SETI@home. In all these previous approaches, the resource availability for task executions was defined based on host reachability, CPU availability to guest processes and keyboard/mouse activities of users. We study here resource availability characteristics based on a node's eligibility for a particular requirement based on the idle resource capacities available on that node. This way of characterizing resource availability is more appropriate for the platforms like PlanetLab because unlike Desktop Grids or volunteer computing systems, the resources on a PlanetLab node are allocated in fair-share manner to all competing users. Work presented in [20] provided a general characterization of resource usage of nodes in the PlanetLab platform in the context of service and application placement. The primary goal of our work is to develop node selection techniques for application placement and dynamic relocation and our study of resource availability characteristics was driven by this goal. The work presented in [25] provided characterization of node availability in PlanetLab based on MTTF and MTTR measures. As compared to that work, our focus is on available capacities on node and their eligibility for various requirement levels.

The work in [4] presents machine learning based techniques for grouping nodes based on similarities in their resource usage

characteristics. The main focus of that work is on scalable management of long historical resource usage information for statistical resource selection over various time intervals. In contrast to that work, the focus of our work is on selecting nodes which continue to meet given resource requirements for long durations. Our profiling method for selecting nodes requires resource usage information over a short history period of up to 30 minutes only, and hence management of resource usage information is not a significant issue.

Resource availability and capacity prediction has been studied extensively in the past [27], [28], [29], [18], [17], [6], [23], [9]. In [23] the authors provided a prediction model based on semi-Markov process to predict resource availability in fine-grained cycle sharing systems. Systems for performing online prediction such as the Network Weather Service (NWS) [29] and RPS [6] provide various prediction methods based on moving mean/median prediction, time series, and auto-regressive prediction methods. In [20] the authors also presented preliminary analysis of applying different predictions methods provided in NWS for predicting resource capacities in the PlanetLab environment. In contrast to the previous approaches, our prediction model is developed based on the observed characteristics of resource usage on PlanetLab nodes. The problem addressed in our prediction model is slightly different from the previous approaches; we address the problem of predicting the resource capacity likely to be available in the near future with some specified confidence level. By specifying the confidence level one can specify the desired reliability of the prediction. Moreover, instead of considering the actual capacity values, our prediction model is based on observing the distribution of relative change in the available capacity from one interval to the next interval.

In the past, several systems [15], [22], [1], [26], [14], [5] have been developed for resource monitoring and node selection in large-scale wide-area shared computing platforms. The primary focus of the designs of these systems was on scalability of node monitoring and resource selection techniques and efficient management of resource usage information. The system for node monitoring presented in this paper was primarily intended for experimental purpose and hence the scalability issues of the monitoring system are not the primary focus of our work.

VIII. Discussion and Conclusion

We have presented in this paper techniques for selecting nodes for application placement in a cooperatively pooled shared computing environment such as the PlanetLab platform. Our techniques are based on the characteristics of resource availability on PlanetLab nodes.

We first provided a basic selection method for determining eligibility of a node for a given requirement. We observed distribution of eligibility and ineligibility periods to develop heuristics for node selection. Our study shows that the recent eligibility period is typically a good indicator for selecting nodes and when a node remains eligible for certain duration it typically tends to continue to be eligible for a long duration. We also studied the relation between a node's eligibility period and ineligibility period. We observed that the nodes with average eligibility period above certain threshold tend to show significantly higher eligibility periods compared to their ineligibility periods. We used these observations to develop a profiling approach for selecting nodes. Our profiling-based node selection techniques select nodes that tend to remain eligible for significantly longer durations compared to the basic selection method. The profiling techniques require resource usage information for only about 30 minutes. Thus, an application deployers would need to monitor nodes only for short time before selecting nodes for deployment.

We observed that the ineligibility periods are typically of short durations and hence an application can benefit by tolerating such short lapses in its host node's ability to meet the resource requirements. Our experiments show that by tolerating ineligibility periods of up to 30 seconds, an application can continue to use a node for a long duration without the need to relocate and still have the required resource capacity available with high probability, which was observed to be over 90% in our experiments.

We presented here an online model for predicting the resource capacities likely to be available at a node in the near future with some given confidence level. Our experiments showed that for any desired confidence level, this model can predict capacity with 95% accuracy. Moreover, for CPU capacity, we observed that a higher confidence level results in higher underprediction. Hence, in case of CPU capacity, setting the confidence level is a trade-off between underprediction and overprediction. However, in case of memory and network bandwidth the underprediction is always limited. Therefore setting a high confidence level is more desirable for memory and network bandwidth. This prediction model can be used by applications and services for predicting their service capacities as demonstrated by our work in [21].

Our techniques can be used by application developers for intelligent selection of nodes for application placement in such cooperatively shared environments. We demonstrated that using these node selection techniques certain level of guarantee of resource availability can be achieved even when the underlying environment does not provide any such guarantees. Moreover, such a level of resource availability guarantee can be achieved even in the absence of any centralized resource manager, through self selection of nodes by application deployers. Thus, such cooperatively pooled environments can be built for shared computing with softer guarantees of resource availability as an alternative to using utility computing platforms such as Clouds. Such a model of shared computing is useful when the applications do not require strict capacity guarantees with the advantage that more number of users and applications can be supported through cooperative sharing of the resources.

References

[1] J. Albrecht, D. Oppenheimer, A. Vahdat, and D. A. Patterson, "Design and Implementation Tradeoffs for Wide-Area Resource Discovery," in

Proceedings of 14th IEEE Symposium on High Performance, Research Triangle Park. IEEE Computer Society, 2005, pp. 113–124.

[2] Amazon, "Amazon EC2, http://aws.amazon.com/ec2/."

[3] A. Bavier, M. Bowman, B. Chun, D. Culler, S. Karlin, S. Muir, L. Peterson, T. Roscoe, T. Spalink, and M. Wawrzoniak, "Operating System Support for Planetary-scale Network Services," in *NSDI'04: Proc.of the 1st Symp. Networked Systems Design and Implementation*, 2004, pp. 253–266.

[4] M. Cardosa and A. Chandra, "Resource Bundles: Using Aggregation for Statistical Wide-Area Resource Discovery and Allocation," in *Distributed Computing Systems, 2008. ICDCS '08. The 28th International Conference on*, June 2008, pp. 760–768.

[5] P. Costa, J. Napper, G. Pierre, and M. V. Steen, "Autonomous Resource Selection for Decentralized Utility Computing," in *International Conference on Distributed Computing Systems*, 2009, pp. 561–570.

[6] P. Dinda, "Design, implementation, and performance of an extensible toolkit for resource prediction in distributed systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 2, pp. 160 – 173, feb. 2006.

[7] P. A. Dinda, "The statistical properties of host load," *Sci. Program.*, vol. 7, pp. 211–229, August 1999.

[8] F. Douglis and I. Foster, " The Grid Grows Up," *IEEE Internet* , pp. 24–26, July-August 2003.

[9] L. Gong, X.-H. Sun, and E. F. Watson, "Performance modeling and prediction of non-dedicated network computing," *IEEE Transactions on Computers*, vol. 51, pp. 1041–1055, 2002.

[10] R. Iosup, M. Jan, O. Sonmez, and D. H. J. Epema, "On the dynamic resource availability in grids," in *Proceedings of the 8th IEEE/ACM International Conference on Grid Computing*, 2007.

[11] B. Javadi, D. Kondo, J.-M. Vincent, and D. P. Anderson, "Discovering Statistical Models of Availability in Large Distributed Systems: An Empirical Study of SETI@home," *IEEE Transactions on Parallel and Distributed Systems*, vol. 99, no. PrePrints, 2011.

[12] D. Kondo, G. Fedak, F. Cappello, A. A. Chien, and H. Casanova, "Characterizing resource availability in enterprise desktop grids," *Future Gener. Comput. Syst.*, vol. 23, pp. 888–903, August 2007.

[13] D. Kondo, M. Taufer, C. L. B. III, H. Casanova, I. Henri, C. Andrew, and A. A. Chien, "Characterizing and Evaluating Desktop Grids: An Empirical Study," in *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS04)*, 2004.

[14] B. Lowekamp, N. Miller, R. Karrer, T. R. Gross, and P. Steenkiste, "Design, Implementation, and Evaluation of the Remos Network Monitoring System," *J. Grid Computing*, vol. 1, no. 1, pp. 75–93, 2003.

[15] M. L. Massie, B. N. Chun, and D. E. Culler, "The ganglia distributed monitoring system: design, implementation, and experience," *Parallel Computing*, vol. 30, no. 5-6, pp. 817–840, 2004.

[16] Microsoft, "Microsoft Azure, http://www.microsoft.com/windowsazure/."

[17] M. Mutka and M. Livny, "The Available Capacity of a Privately Owned Workstation Environment," *Performance Evaluation*, vol. 12, no. 4, pp. 269–284, 1991.

[18] M. Mutka, "Estimating capacity for sharing in a privately owned workstation environment," *IEEE TSE*, vol. 18, no. 4, pp. 319–328, April 1992.

[19] D. Nurmi, J. Brevik, and R. Wolski, "Modeling Machine Availability in Enterprise and Wide-area Distributed Computing Environments," in *Euro-Par05*, 2003, pp. 432–441.

[20] D. Oppenheimer, B. Chun, D. Patterson, A. C. Snoeren, and A. Vahdat, "Service Placement in a Shared Wide-Area Platform," in *ATEC '06: Proceedings of the Annual Conference on USENIX '06 Annual Technical Conference*, Berkeley, CA, USA, 2006, pp. 273–288.

[21] V. Padhye and A. Tripathi, "Building Autonomically Scalable Services on Wide-Area Shared Computing Platforms," in *Proc. of the IEEE Symp. on Network Computing and Applications*, 2011, pp. 314 – 319.

[22] K. Park and V. S. Pai, "CoMon: A Mostly-scalable Monitoring System for PlanetLab," *SIGOPS Oper. Syst. Rev.*, vol. 40, no. 1, pp. 65–74, 2006.

[23] X. Ren, S. Lee, R. Eigenmann, and S. Bagchi, "Resource Availability Prediction in Fine-Grained Cycle Sharing Systems," in *HPDC*, 2006, pp. 93–104.

[24] A. Tripathi, V. Padhye, and D. Kulkarni, " Resource-Aware Migratory Services in Wide-Area Shared Computing Environments ," in *Proc. of the IEEE Symp. on Reliable Distributed Systems*, 2009, pp. 51–60.

[25] T. Warns, C. Storm, and W. Hasselbring, "Availability of Globally Distributed Nodes," in *Proceedings of the IEEE Symposium on Reliable Distributed Systems*, 2008, pp. 279–284.

[26] M. Wawrzoniak, L. Peterson, and T. Roscoe, "Sophia: An Information Plane for Networked Systems," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 1, pp. 15–20, 2004.

[27] R. Wolski, "Dynamically Forecasting Network Performance using the Network Weather Service," *Cluster Computing*, vol. 1, no. 1, pp. 119–132, 1998.

[28] R. Wolski, N. Spring, and J. Hayes, "Predicting the CPU Availability of Time-shared Unix systems on the Computational Grid," *Cluster Computing*, vol. 3, no. 4, pp. 293–301, 2000.

[29] R. Wolski, N. T. Spring, and J. Hayes, "The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing," *Future Generation Computer Systems*, vol. 15, no. 5–6, pp. 757–768, 1999.