**Renewal Request for the CIPRES Science Gateway**

##### Research Objectives

This request is for a renewal of the XSEDE allocation that sustains the CIPRES Science Gateway (CSG), which serves the evolutionary biology and systematics communities. The idea of a public portal that provides community access to computational resources was initially implemented by the CIPRES (CyberInfrastructure for Phylogenetic RESearch) project, a multi-institutional collaboration that created algorithms, tools, and computational capabilities to help researchers analyze phylogenetic relationships in a manner that scales across the entire tree of life. The CSG evolved from this implementation and is designed to simplify researcher access to community codes run on high performance computational resources, so domain users can infer evolutionary relationships on data sets that would otherwise be intractable.

Inferring evolutionary relationships from protein and DNA sequence data is computationally intensive. The process involves two distinct steps. First, ***sequence alignment*** algorithms are used to predict homology relationships between individual nucleotide or amino acid residues in a collection of homologous DNA or protein sequences. Next, ***phylogenetic trees are inferred*** using an algorithmic model for how a given set of aligned sequences evolved from a common ancestral sequence. Both steps in this process are NP hard [1, 2] and community codes for analyzing the data must use heuristics to simplify the calculations. Even with simplifying heuristics, routine phylogenetic analyses require more and more computational power every year because the amount of sequence data available for analysis is increasing exponentially [3].

The rapid increase in available data has driven a dramatic increase in the need for access to high performance computational resources for routine analyses. As a result, investigators who cannot access such resources are at a significant disadvantage relative to those who can. The powerful high-end resources of XSEDE offer a scalable solution to this problem, but the overhead for access requires submitting a convincing resource request, setting up a personal environment, learning to use the queuing systems, installing and configuring the tools, then learning to run these tools efficiently from the command line. This is beyond the capabilities of most practicing biologists. The CSG provides a point-and-click browser interface that allows users to configure and run community codes on high performance computational resources with no knowledge of the detailed implementation of these codes. This saves researcher time and effort, and dramatically expands the scope of what many researchers can do in a given time frame.

The CIPRES Science Gateway provides the following benefits for its user community:

* The browser interface allows users to run on HPC resources without requiring them to deal with the complexities of Linux, MPI, and job schedulers.
* Individuals who are resource-limited can complete analyses of larger data sets, so analyses can be tailored to the question at hand, rather than to available resources.
* Large scientific questions can be asked and answered by individuals irrespective of their local cyberinfrastructure.
* Community developers can use CSG resources to develop their codes, and then deploy new codes easily within the CIPRES Science Gateway to reach a targeted user base.
* New community codes can be deployed for a large audience in a familiar environment. This promotes communication between users and tool creators.
* The interaction between the Extended Collaborative Support group (ECS) and community code developers promotes more efficient use of existing community codes and encourages the development of new parallel codes.

##### Background

The CSG was originally released in May 2007 as the CIPRES Portal. It provided simple point-and-click access to community codes run on a modest cluster at no cost to users. The CIPRES Portal provided access to codes for sequence alignment (ClustalW, Contralign, Muscle, MAFFT, Probalign, Probcons), codes for tree inference (MrBayes, RAxML, GARLI, PAUP\*) and the POY, which combines both sequence alignment and tree inference. The CIPRES Portal allowed users to access most of the command line options for each code hosted; data and results of analyses could be stored indefinitely. The interfaces were created using a straightforward XML specification so new codes can be added as they appear within the community and existing interfaces can be adapted as community codes are modified.

The demand for computational time by CIPRES Portal users rapidly exceeded the capacity of the cluster that served as the compute engine for this resource. In response, the CIPRES Portal was converted to a Science Gateway project in 2009, to take advantage of the scalable resources then available through TeraGrid. Shortly afterward, it was renamed the CIPRES Science Gateway. The development and release of the CSG was accomplished with the following activities:

* Built infrastructure to permit TeraGrid access from a Web application using GRAM/Globus and adapted the infrastructure to the unique running environments at NCSA and TACC.
* Ran benchmarking tests and long jobs for key users to develop rules for efficiently running parallel versions of three phylogenetics codes on TeraGrid resources.
* Developed and deployed interfaces that permit users to launch parallel jobs on the TeraGrid with the number of nodes and MPI processes automatically selected according to the aforementioned rules.
* Tested the interfaces with a small group of early users and incorporated their feedback to improve portal performance and user experience.
* Created documentation and tools to help users launch their jobs on the TeraGrid, monitor job progress and completion, and interpret error messages.
* Provided TeraGrid access to the three most computationally demanding codes: MrBayes, RAxML and GARLI.
* Assisted users who had jobs that required special handling.
* Collaborated with the developers of RAxML and MrBayes to create hybrid parallel versions [4, 5] that provide dramatic improvements in speed compared to earlier versions**.**
* Deployed hybrid parallel versions for TeraGrid access via the CSG in February 2010.
* Migrated serial codes from the CIPRES cluster to the Triton cluster at SDSC, which can be used on a fee-for-service basis for such tasks.
* Supported development of the improved sequence alignment code Probalign and its evaluation in the context of other available sequence alignment codes.

Public submissions to TeraGrid resources began 12/1/2009 with an initial TRAC award of 400,000 SUs. Demand for time on the new CSG was far beyond anything we had anticipated, and we were able to meet user demand through the first allocation cycle (terminating 6/30/2010) only through generous supplements from the TRAC. In Year 2, CSG users consumed 7.5 million SUs. This need was met through a TRAC award, a supplement on the Trestles, and an advance on the request for the 2011 allocation year. In Year 3, we received an XSEDE award of nearly 10 million SUs. The CSG is currently using that award; approximately 10% of the time remains. Users are currently requesting an average of approximately 850,000 SUs per month from the CSG.

All of our CSG awards to date included support from the ASTA (now ECS) project group, with assistance in software design and implementation (Nancy Wilkins-Diehr and Doru Marcusiu) and in parallel algorithm development, implementation, and tuning (Wayne Pfeiffer). Support from this group has been essential in making our work successful.

##### Codes and Methods

Current Codes: The CSG provides access to parallel versions of five widely used phylogenetics codes on XSEDE: MrBayes (hybrid MPI/OpenMP), RAxML (hybrid MPI/Pthreads), GARLI (MPI), BEAST/BEAGLE (OpenMP), and MAFFT (Pthreads). The hybrid parallel versions of MrBayes and RAxML were implemented through an ASTA project and are the fastest versions of these codes available anywhere. Wayne Pfeiffer of SDSC was a key participant in the creation of hybrid RAxML. The codes used by CSG and their parallel implementations are described below:

MrBayes is a Bayesian tree-inference code. The CSG offers the hybrid MPI/OpenMP version of MrBayes 3.1.2 that exploits the fine-grained parallelism within each run-chain instance. As Trestles became available, we modified the portal interface to automatically configure runs correctly, based on user-entered problem specifications. Runs with the default 8 run-chain instances use one 32-core node of Trestles with 8 MPI processes @ 4 OpenMP threads each. Benchmarks show that such runs on Trestles are 5 to 10% faster than comparable runs on four 8-core nodes of (now retired) Abe. For unpartitioned data sets, runs with 4 threads are typically 2.0 - 2.7 fold faster than runs without threading. For partitioned data sets the speedups are somewhat lower.

RAxML is a maximum likelihood phylogenetics code. The CSG provides the hybrid MPI/Pthreads version (currently RAxML 7.3.0), which parallelizes over the number of searches, so multiple nodes can be used. Benchmarking studies were performed to determine the preferred numbers of MPI processes and Pthreads for Trestles. The user interface configures these numbers automatically based on user-entered problem specifications. For the recommended case of a comprehensive analysis with automatic bootstopping, two 32-core nodes of Trestles are used with 10 processes @6 threads each. Benchmarks show that such runs are 1.3-1.6 fold faster than comparable runs were on five 8-core nodes of Abe, which was the previous norm. For unpartitioned data sets, the speedups on 60 cores relative to serial runs are outstanding, with values varying from 33-59 fold for typical benchmarks. For partitioned data sets the speedups are somewhat lower.

GARLI is a maximum likelihood phylogenetics code that includes an MPI implementation. GARLI 1.0, the latest version, was recently installed on Trestles and runs efficiently using MPI. MPI is beneficial whenever there are multiple regular searches, multiple bootstrap searches, or both, one of which is almost always the case. One MPI process is assigned to whichever search is limiting, up to a maximum of 32 processes. All runs execute on a single 32-core node of Trestles, either in the shared queue if there are greater than 32 processes or in the normal queue otherwise. A standard benchmark run with 100 bootstraps is 23-fold faster on 32 cores than on a single core.

MAFFT is a multiple sequence alignment code for which a Pthreads parallel version was released recently [6]. This version became available via the CSG in February, 2011. All runs use a single 32-core node of Trestles. The BB30003 benchmark in [6] achieves a speedup relative to serial runs of between 8-11 fold on 32 cores. The run time and speedup vary because the parallel search algorithm uses random numbers in a way that is not reproducible. Thus the number of iterations and even the final alignment vary from run to run, though the resulting alignments seem to be of similar quality. Larger alignments should achieve higher speedups.

BEAST is a Bayesian Tree Inference code [7]: The BEAST (Bayesian Ecological Analysis of Statistical Trends) code became available via the CSG in September, 2011. BEAST was the single most user-requested new program for the CSG; user demand increased when the other public portal that provided access to BEAST code closed on April 15, 2011. While BEAST is a serial Java/C program that does not map well to XSEDE resources, the computationally intensive kernels of BEAST can be run using a library called BEAGLE [8], which uses threading on GPUs (CUDA) or CPUs (OpenMP). The authors reported that BEAST/BEAGLE can give 120-fold speedups relative to sequential BEAST on a GPU device. Wayne Pfeiffer installed BEAST/BEAGLE as part of our ECS collaboration, and conducted benchmarking experiments using samples of real user data from individuals who had requested the code [9]. Benchmarking experiments showed that the large speed-ups noted with BEAST/BEAGLE on GPUs was obtained only for data sets that contained partitions with a large number of characters. Data sets commonly submitted by CSG users would not receive any benefit from GPU usage, however, because they have few partitions, and the partitions have relatively low numbers of unique characters. In this situation, the benefits of running on GPUs are negated by overhead from loading and unloading data. On the other hand, running BEAST/BEAGLE using OpenMP gave 2.3 - 7 fold speed-ups when run on 8 cores on Trestles. Since the magnitude of the speed-up is determined by properties of the data set, we created interfaces that ask users about the nature of their data sets, and use this information to select the optimal configuration for that particular data set. Currently BEAST runs are conducted on 8 cores in the Trestles shared queue.

**New Codes:** We plan to continue offering new parallel codes as they appear and are requested by users. We are currently benchmarking MrBayes 3.2 for release in Spring of 2012. This newer version of MrBayes has been in development for years, and was released to the public in late 2011. MrBayes 3.2 has several new features, including restart/checkpointing capabilities that will greatly mitigate the loss of SUs seen over the past year when long running MrBayes jobs fail. The new MrBayes 3.2 also incorporates commands from the related program BEST: Bayesian Estimation of Species Trees [10]. The BEST code expands the capabilities of MrBayes 3.1.2 by implementing a Bayesian hierarchical model to jointly estimate gene trees and the species tree from multi-locus sequences. Both MrBayes 3.2 and BEST codes are highly requested in the CSG, and we look forward to releasing this code soon.

RAxML-Light [11]: This is a modified version of RAxML designed for inferring very large trees using MPI only. This code has been developed by a partnership between Wayne Pfeiffer at SDSC and Alexandros Stamatakis at Heidelberg Institute for Theoretical Studies. The code has been developed as a shell script that calls individual modules of three separate programs: RAxML-pthreads, RAxML-serial, and Parsimonator. Deploying this code has been challenging because the script deploys up a hundred separate jobs, and each job must be tracked. Moreover, since RAxML-Light jobs may require longer than the maximum time allowed by XSEDE queues, the CSG software may need to be modified to support restarts. Access to RAxML-Light has been developed with funding from the iPlant collaborative. Users will consume SUs from an independent XRAC allocation awarded to the iPlant project, while the CSG group will support the development, benchmarking, and deployment of the code.

##### Management of Resources

As noted in the accompanying progress report, resource consumption by CSG users grew substantially during the 2010 annual renewal cycle. The growth in consumption had two causes. The *primary* cause was an increase in the number of users submitting jobs to the CSG and an increase in the size of the computational jobs these users submitted. As a result, in August 2011 we implemented a policy that limited users at US institutions to 50,000 SUs per year, and users outside the US to 30,000 SUs per year (please see the attached Progress Report for additional details). A secondary cause of the growth in resource consumption was an unfortunate incompatibility between MrBayes 3.1.2 and the Lustre file system used on Trestles. This caused running MrBayes jobs to crash suddenly. This section provides detailed documentation of the effect of policies implemented to manage SU consumption by the CSG, and our strategy for managing issues that arose due to the incompatibility of MrBayes code and the Lustre system.

**Managing growth in resource demand**

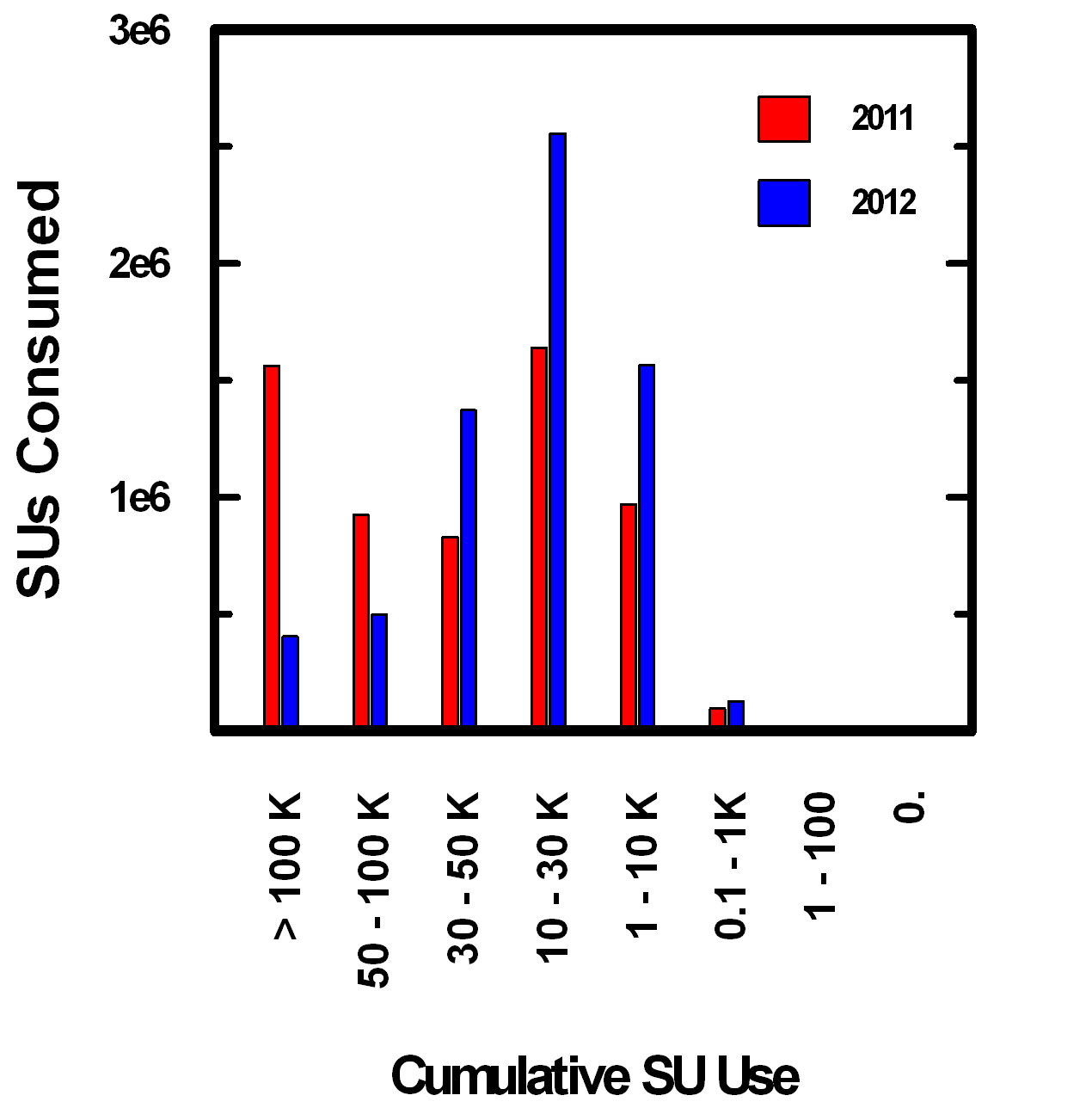
The impact of the new user management policies on CSG usage patterns was assessed by comparing usage data for after implementation of the policies in August 2011 with that of the 2010 allocation year. Figure 1 compares users binned according to their level of SU consumption in the 2011 and 2010 allocation years. The red bars represent use in each bin during the final 10 months of the 2010 allocation, while the blue bars represent use of the 2011 allocation over the 9 months that the new usage policies were in effect. Despite the shorter time period reported for 2011, there was a 6% increase in the number of CSG users relative to 2010. If normalized to a 10 month period, this rate of usage represents an 18% increase relative to 2010. This increase in users occurred entirely within the groups that consumed between 100 and 50,000 SU. On the other hand, the number of users who consumed more than 50,000 SU decreased from 27 to 10 under the new policies *(this number should theoretically have been zero according to our new policies, please see below for explanation).* Figure 2 shows the impact of the new policies within each user bin. Users consuming between 10,000 and 50,000 SUs accounted for 60% of all SUs used from the 2011 allocation, a significant increase over the 40% consumed by this group in the 2010 allocation. At the same time, consumption by high end users decreased: users who used more than 50,000 SUs consumed only 17% of total SUs from the 2011 allocation, as compared to 41% in the 2010 allocation. Thus, the new policies eliminated the disproportionate resource consumption by high end users, and the distribution of usage across all bins in the 2011 year resembled a bell-shaped curve, with the majority of SU consumption coming from users requiring between 1000 and 50000 SUs.

**2011**

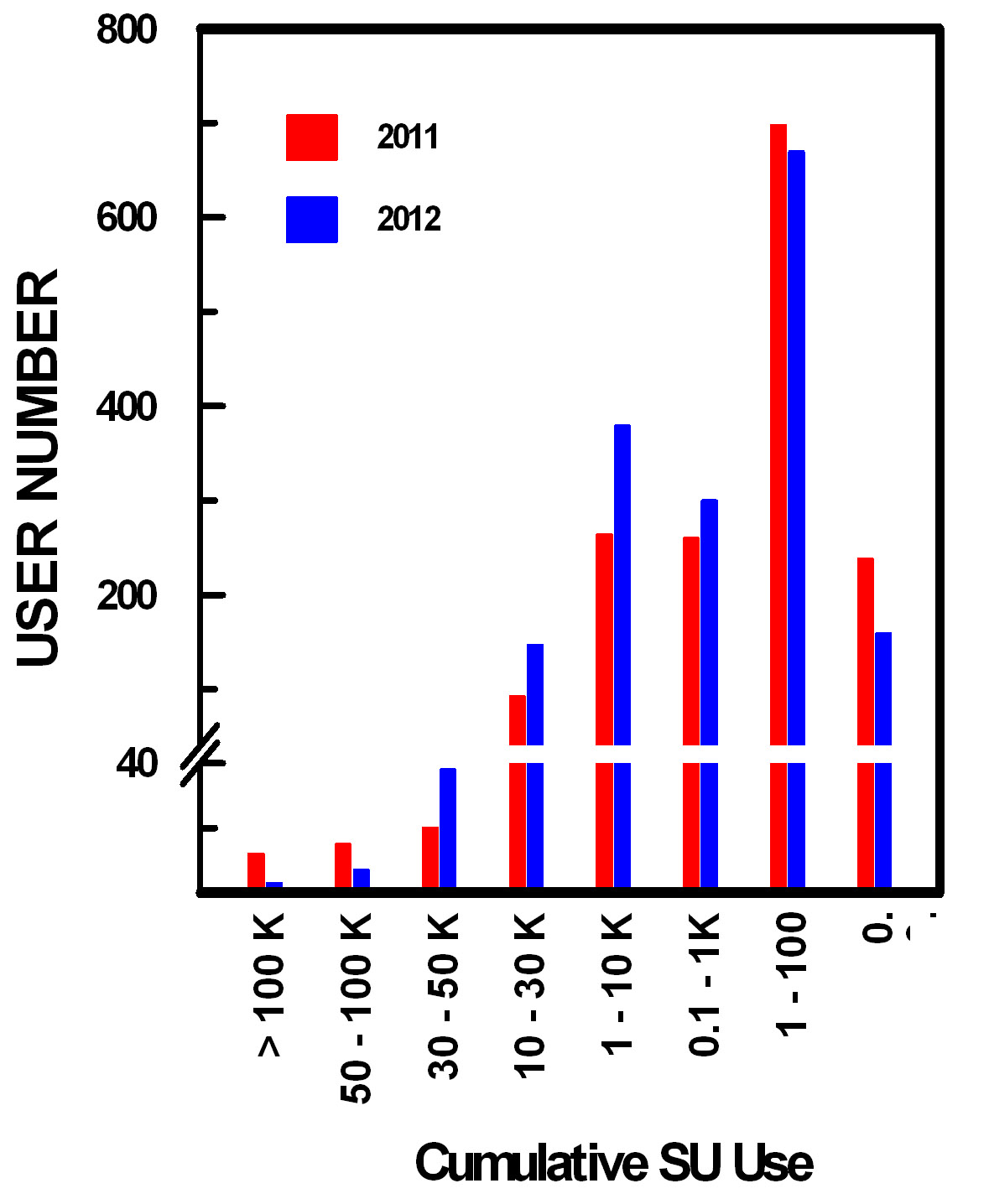
**2010**

**2011**

**2010**



**Figure 2. SU consumption by CSG users, binned according to total use.**



**Figure 1. Users of the CSG binned according to total use. The y-axis scale is discontinuous, so the number of users at larger usage levels can be viewed more clearly.**

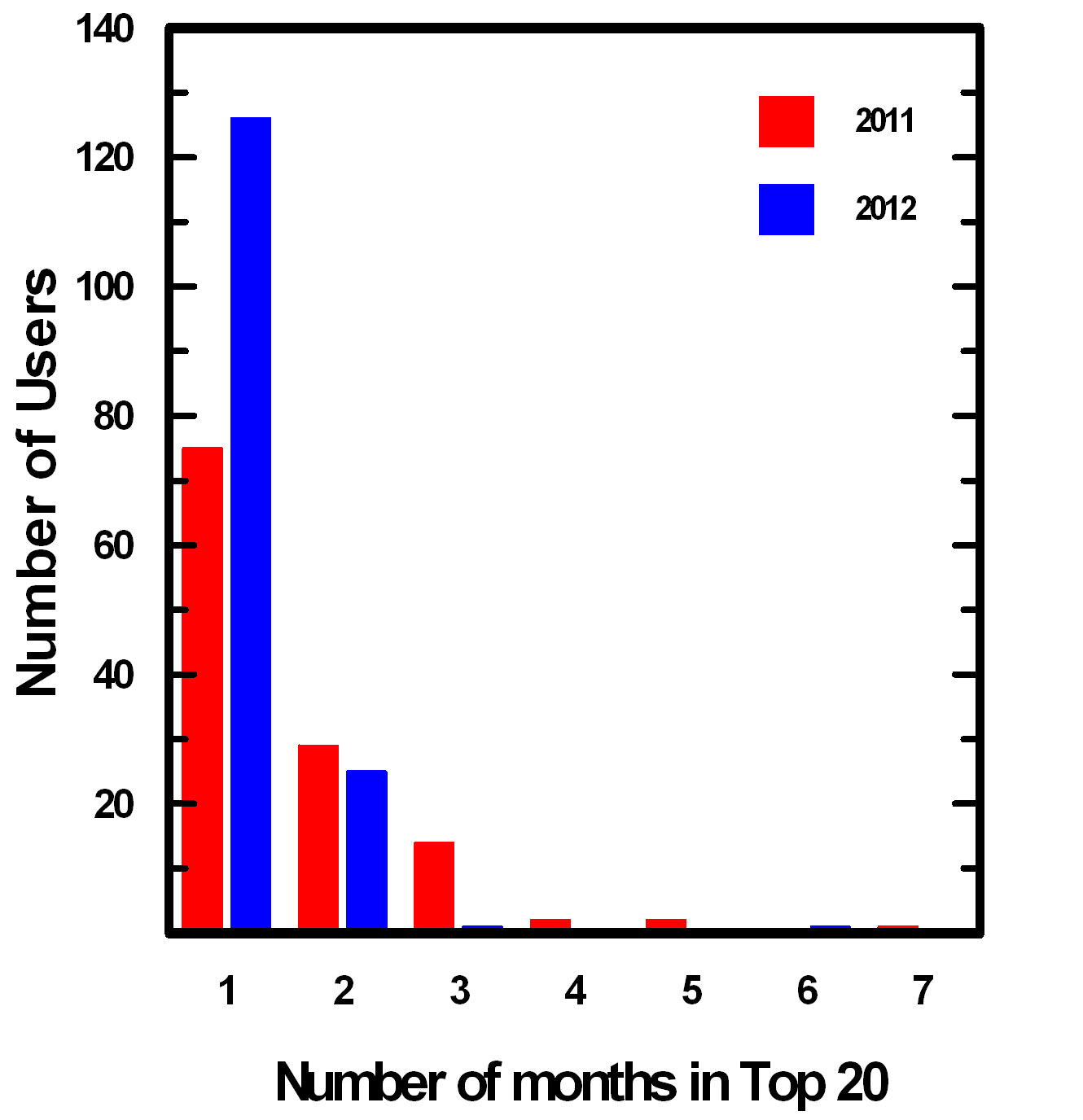
We conclude the new usage policies 1) were effective in controlling runaway usage by a small number of individuals, 2) were effective in shifting SUs to the key target constituency: users who have requirements that are beyond their local capabilities, yet are modest in the context of XSEDE resources, and 3) did not discourage new users from using the CSG.

**Why are there still users who consume more than 50,000 SUs?**

The new CSG policy dictates that individual users cannot consume more than 50,000 SUs from the community allocation, but, as the data shows, 10 users did consume more than that amount. Some of these users overshot the 50,000 SU limit because of the way user submissions are disabled. The CSG application does not automatically disable user accounts when the threshold of 50,000 SUs is passed. Rather, when a user crosses the 50,000 SU threshold, the CSG application generates an email to the user and the PI. The PI then sends a personal email message to the user to inquire about their situation. The account is disabled if the user does not reply, but there is a short time lag between notification and disabling submission from the account. The reasoning for this is as follows: if a user has nearly completed their work, and has invested 50,000 SUs into the study, it is good business practice to ask the user if their work is nearly completed. Two users exceeded the 50,000 SU limit by submitting additional jobs after they were contacted and before the account was disabled, and two were allowed to complete work that required slightly more than 50,000 SUs. Other users were allowed to exceed the 50,000 SU limit because they were victimized by the intermittent file system problems that caused long-running MrBayes jobs to crash suddenly on Trestles. The impact of this issue was predominantly on very long MrBayes job runs on 64 cores. Users impacted by a few such failures can quickly exhaust an allocation of 30,000 or even 50,000 SUs. We had no mechanism for crediting user accounts for the lost time, so we allowed users impacted by this issue to re-run jobs even though (formally) their account had already consumed 50,000 SUs. Since disabling user submissions is manual, it was simple for us to exercise our discretion in this process. Six users were allowed to exceed the 50,000 SU limit to compensate for MrBayes job losses.

**How diverse is the High End User Population?**

The potential for disproportionate use of resources by a fairly small group of users was a key concern in the review of the previous allocation request. The question the reviewers asked was: *are a few users bypassing the normal XSEDE peer-review process by submitting many large jobs to the CSG?* Given the magnitude of the allocation requested by the CSG, we feel it is important revisit this issue annually. The top 20 users (in terms of number of SUs consumed) were identified in each of 10 months during the 2010 and 2011 allocation years. We then scored the number of months that each user ranked in the top 20. Figure 3 compares the results of this analysis for 2011 with the equivalent data from the 2010 allocation year. In 2010, a total of 123 unique users appeared in the top 20 over a 10 month period. 76 of these users were in the top 20 only once, 24 appeared twice, and only 5 users appeared among the top 20 more than 3 times. In 2011, the top 20 users were even more diverse: 149 unique users appeared in the top 20, 122 (80%) of these were in the top 20 only once, 22 (18%) appeared twice, and only two individuals appeared in the top 20 more than twice. The results are consistent with the expected usage pattern for phylogenetics research: researchers have brief periods of intense resource consumption while they analyze assembled data, followed by periods of relative inactivity while they collect new data for analysis. The usage data makes it clear that there is not a small coterie of users who are using the CSG to circumvent the normal XSEDE review process, and suggests that the new policies put in place have further diversified the high end user group.



**Figure 3. Frequency of appearance among the Top 20 resource consumers over a 10 month period in 2010 and 2011.**

**2010**

**2011**

The impact of the user management policies on the total number of SUs consumed by the top twenty users is shown in Figure 4. In the 2010 allocation year, SU consumption by the top 20 users increased steadily, but usage by this group remained relatively flat during 2011. The average top 20 user consumed approximately 20,000 SU per month during the 2011 allocation year (Figure 4, left panel). Importantly, the top 20 users during the 2011 allocation year consumed a smaller proportion of total CSG resources than it did in 2010 (Figure 4, right panel), but this reflects increased consumption by users in lower usage bins, rather than a decrease in consumption by the top 20 users.

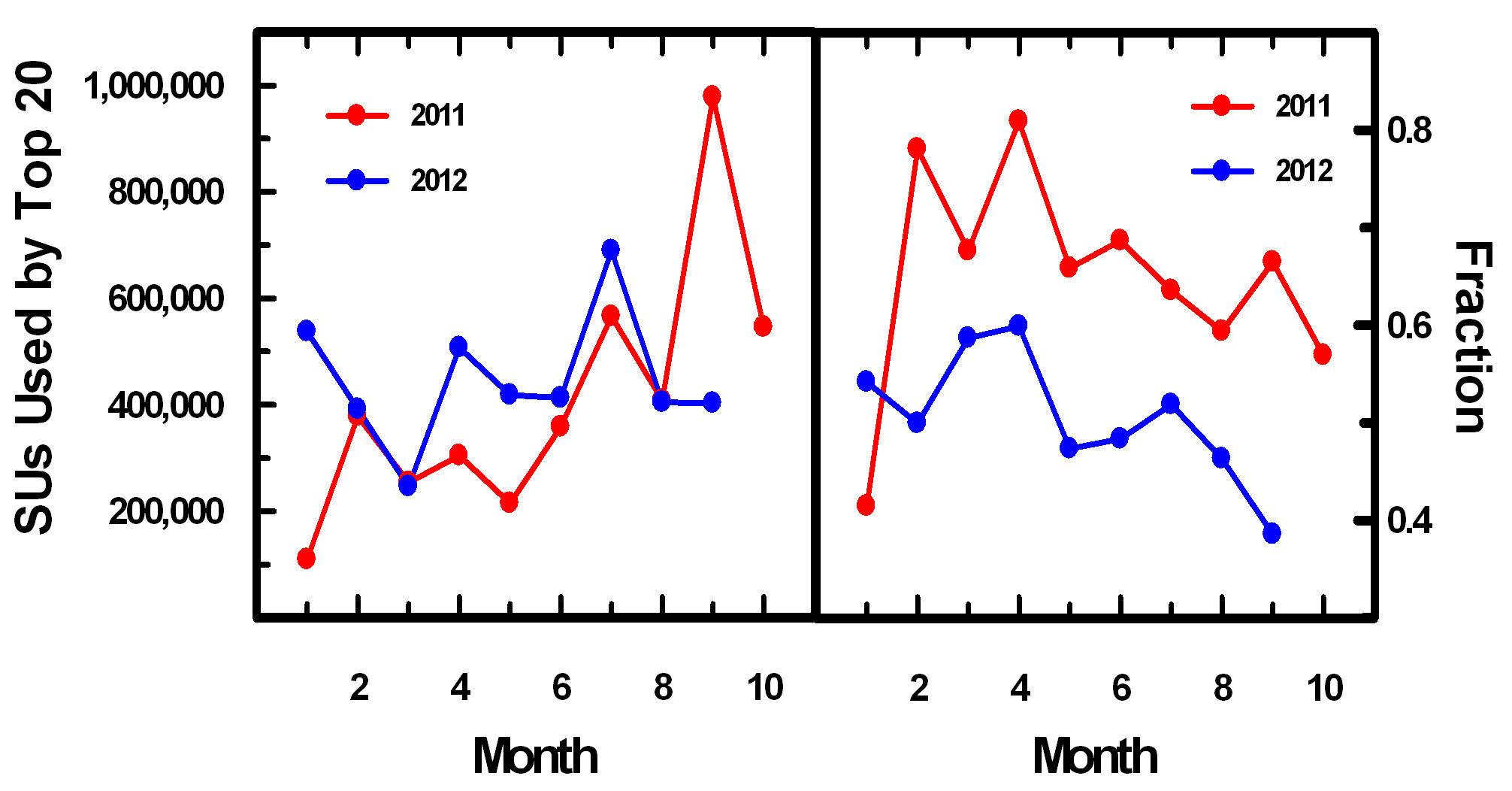
**2010**

**2011**

**2010**

**2011**

Another issue of importance in evaluating CSG resource consumption is the demographics of the user population. As in the previous year, approximately 46% of the top 20 users were employed at US institutions, and approximately 54% were at foreign institutions. The top 20 users in the US were broadly distributed across the US; they include individuals at both large research institutions and smaller colleges. Similarly, the foreign institutions represented by users in the top 20 are spread across Europe, Asia, North and South America, and Australia. At present, there are no accurate data on whether or not users at foreign institutions have collaborators in the US. In a user survey approximately half of all responding users from foreign institutions report having a US collaborator, but the extent of overlap between these respondents and the Top 20 users is unknown.



**Figure 4. Total SU consumption by the Top 20 Users over a 10 month period.**

**Managing resource loss due to job failures**

As noted above, a secondary issue that increased resource consumption by the CSG was an incompatibility between MrBayes 3.1.2 code and the Lustre file system used on Trestles. MrBayes results logging is apparently I/O intensive and at times of high usage the Lustre system is unable to complete read/write actions required by the code. This issue first became evident during May 2011 (Month 9, 2010 allocation year in Figure 4 above), when all running MrBayes jobs began to crash simultaneously at irregular times. We identified these jobs in our logs through a diagnostic error message produced by MrBayes jobs that fail for this reason, and by the simultaneous failure of several MrBayes jobs. The issue was rectified temporarily by shifting all MrBayes jobs to a separate file system in July 2011. This change ended the MrBayes job failures, and brought SU consumption back to expected levels. The problem recurred in January 2012 (Month 7, 2011 allocation year in Figure 4 above), when MrBayes jobs were shifted back to a new Lustre storage system. Although the problem was less severe in this new file system, the issue caused 22% of all submitted MrBayes jobs to fail in January 2012. In the aggregate, approximately 500,000 SUs were lost due to this issue during the 2011 allocation cycle. Because this problem is sporadic, unpredictable, and costly, we recently moved all CSG jobs to a ZFS file system mounted on Trestles. We expect this change to eliminate the issue.

##### Resource consumption under new management policies

To evaluate the effect of management policies on resource use by CSG, we first consider the impact on use by the few high end users who caused resource use to skyrocket. Based on CSG usage statistics from the 2010 allocation year, we predicted that imposing a limit of 50,000 SU would decrease annual SU consumption by about 2,000,000 SU and would impact 12 users. As a group, users who consumed more than 50,000 SUs received 1.3 million fewer SUs in the 2011 allocation year than they did for the same period in 2010. This is a significant savings, although somewhat less than predicted. Since August 2011, when the current policies were implemented, 10 user accounts have been disabled for exceeding the amount of allowed SUs. It is likely a few more will reach this threshold before the end of the 2011 allocation year.

We evaluated the impact of the new policies on overall resource use by comparing the number of SUs consumed before and after the implementation of the policy. Since the 2011 allocation cycle is still in progress, the available data were normalized to a 12-month period; the results of this calculation are shown in Table I. While consumption by users with over 50,000 SUs was significant (63%), the overall SU consumption for the 2011 allocation year is projected to be 8.7 million SUs, compared with 8.0 million SUs for the 2010 allocation year. When adjusted for loss of 500,000 SUs due to premature termination of large MrBayes jobs, the projected total consumption for the 2011 year drops to 8.2 million SUs, a 2% increase in consumption relative to the 2010 allocation year. The decrease in high end usage did not translate into a net decrease in SU consumption because of an increase in the total number of CSG users. Based on current trends, the CSG user population is projected to grow by 18% in 2011. Thus, the new policies succeeded in redistributing resources from a few high end users to a much larger number of middle level users. The policies therefore achieved their primary goal, to promote use of the resource by users with moderate computational requirements. If the user base continues to grow, we may need to re-evaluate and modify these policies to further constrain use.

**Table I. Binned Usage of the CIPRES Science Gateway**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Usage** | **Projected SU/Year** | | **Fraction Total SU** | | **Projected Total Users** | | **% Users** | |
|  | 2011 | 2012 | 2011 | 2012 | 2011 | 2012 | 2011 | 2012 |
| > 100 K | 2,081,398 | 538,279 | 0.257 | 0.0618 | 14 | 4 | 0.008 | 0.0018 |
| 50 - 100 K | 1,232,990 | 663,765 | 0.158 | 0.0762 | 18 | 9 | 0.009 | 0.0041 |
| 30 - 50 K | 1,103,531 | 1,829,005 | 0.138 | 0.2099 | 24 | 51 | 0.013 | 0.0223 |
| 10 - 30 K | 2,184,180 | 3,406,100 | 0.27 | 0.391 | 112 | 196 | 0.058 | 0.0863 |
| 1 - 10 K | 1,290,989 | 2,087,027 | 0.159 | 0.2396 | 317 | 505 | 0.165 | 0.2225 |
| 0.1 - 1K | 125,807 | 167,776 | 0.016 | 0.0193 | 312 | 400 | 0.163 | 0.1762 |
| 1 - 100 | 17,936 | 19,913 | 0.002 | 0.0023 | 838 | 893 | 0.436 | 0.3934 |
| 0 | 0 | 0 | 0 | 0 | 286 | 212 | 0.149 | 0.0934 |
| Total | 8,036,831 | 8,711,865 |  |  | 1920 | 2271 |  |  |

We believe that providing clear policies, along with tools to forecast and monitor resource consumption, will help users be strategic in their resource use. Since these usage policies were implemented, a number of users have inquired about their use levels, and view resource use as a consideration in planning their experiments. We have received several inquiries about personal allocations. To date, a single user has requested and received a personal allocation, and is running on CSG using this allocation. The tools we have created to enforce policies are flexible, and will allow us to adjust policy details as required to restrain resource consumption.

##### Justification for Resources Requested

We are requesting a renewal allocation that will meet the projected needs of CSG users in the context of the new management policies that have been implemented. We are also requesting continued support from the ECS group to help adapt, tune, and deploy new codes efficiently on XSEDE resources.

**Our allocation request is as follows:**

*11 million SUs on Trestles.* This resource request is based on extrapolation from recent usage: over the past 6 months (when file system issues did not artificially inflate use) consumption averaged approximately 850,000 SUs per month. The amount requested will provide sufficient resources to support our existing user base, plus a small amount of additional resources to accommodate growth of the user base. Over the lifetime of the CSG, an average of 25 more users submit jobs in each month. This, together with release of MrBayes 3.2/BEST is expected to increase usage by CSG users modestly over the 2012 allocation cycle.

We appreciate that this is a large request. However, when considered in the context of the number of scientists who will benefit from the allocation, and in the context of the number of CSG users versus total XSEDE users, the request is actually quite reasonable. During the past calendar year, 2705 unique users submitted jobs to XSEDE/TeraGrid via the CSG. Assuming that an equal number submit jobs between July 2012 and July 2013, the allocation requested here represents 4,066 SU per user. This is a small fraction of the limit for XSEDE start-up accounts. Moreover, in Q4 of 2011, the CSG brought 935 users to XSEDE. Over this period there were 1,389 active Gateway users, and 3,211 active XSEDE users. In other words, the CSG accounted for 66% of all active Gateway users, and 29% of all active XSEDE users. If we annualize the 385 million SUs to be allocated in the most recent quarterly XRAC meeting, the CSG request will support nearly 30% of all XSEDE users with 0.7% of the total SUs allocatable resources on XSEDE resources. As measured by the number of published scholarly works reported by our users, the 16 million SUs awarded to the CSG to date has enabled in excess of 300 publications (with 113 more in press or in preparation) between 2010 and 2012. This is more than 1 publication per 50,000 SUs consumed. We believe this is a highly effective use of computational resources.

We are requesting all 11,000,000 SUs on Trestles, based on consideration of the benefits to users. Trestles provides a significant speed-up for results delivery through a combination of factors, but primarily because the jobs are run on 32-core nodes. The increased number of cores available made for shorter wall times for individual jobs relative to previous machines. It also meant larger jobs could be run within the 168 h wall clock limit for submissions. There was also a particular benefit for RAxML jobs, which make up nearly 2/3 of all TeraGrid submissions from the CSG. As shown in Figure 5, RAxML parallel efficiency is always greater than 0.5 for runs up to 60 cores (which are actually faster than runs on all 64 cores). Importantly, the parallel efficiency improves for data sets with more patterns, and in the largest case (7400 patterns) we observe a parallel efficiency of nearly 1 at 60 cores. This is a huge benefit as the size and complexity of data sets continues to increase. Finally, an additional benefit of Trestles (though perhaps a transitory one) is the absence of queue wait times for all running jobs.



**Figure 5. Scalability of RAxML on Trestles**

*Continuing ECS support.* As noted above, the ECS support we received over the previous two years has been instrumental in the success of the CSG. We specifically request continued support from Wayne Pfeiffer, who has knowledge of our entire system, and is currently actively collaborating with developers in the systematics community to improve their codes. This has been a very productive collaboration, and we do not believe we could continue our operation without his support.

##### Availability of Other Computational Resources

The CSG will continue to use the Triton cluster at SDSC to run serial jobs. A portal that lacked these codes would be less useful. Supporting serial codes on the Triton cluster allows us to offer commonly used sequence alignment codes and file manipulation tools, while reserving XSEDE resources for jobs that use these resources efficiently. Triton offers cost-effective fee-for-service access to cycles, and we have sufficient funding to continue support of this resource for our serial codes. Juxtaposing parallel and non-parallel community codes in the CSG not only increases efficiency, but will also make it obvious to code developers why parallel versions of their code should be developed whenever possible.

##### Community Support and Management Plan

*Community Support Plan:* The Community support plan is essentially unchanged from the previous submission. The CSG allows users to report issues and request features via a commercial bug tracking software package. Documentation for appropriate use is provided in text format; flash demonstrations of job creation and data management are provided. Users are provided with links to XSEDE resource status and maintenance calendars, and warnings are posted when specific resources are unavailable. User support is provided by the PI on a greater than 8-to-5 and less than 24/7 basis, and includes all aspects of user issues, from feature requests to file format issues.

*Management Plan:* The Web application infrastructure for the portal is stable and flexible. It is designed to support changes in the toolkit with minimal developer effort. The cost of maintaining the Web presence is minimal at this time, with more than 90% of reported issues being attributable to user error or resource unavailability. The PI assumes responsibility for day-to-day operations, user support, and tracking resource availability. Software infrastructure for managing users and all middleware development for deploying will be addressed by Terri Schwartz, a senior developer with more than 20 years experience. Terri created the CSG v 1.0, and has managed all aspects of middleware development and XSEDE implementations for CSG v 2.0. Access to MrBayes3.2/BEST will be accomplished by collaboration between Wayne Pfeiffer (ECS), Terri Schwartz, and the PI.

The PI will be responsible for overseeing and tracking user consumption of resources and insuring the CSG fair use policy is enforced. He will assist users in requesting individual XSEDE allocations as that becomes necessary. He will also oversee the development and implementation of the tools required to track usage, support users in accessing personal allocations, and ensure that technical support is available to such users.

*Financial Support:* The current CSG was created with funding from two awards: the original CIPRES ITR award (NSF EF 03-31648), and the NEW Biology Workbench, NIH 5R01GM073931, which sustained the underlying framework of the CSG and allowed new features to be added. Both of these awards terminated as of March 31, 2010. Funds to sustain portal operations over the next year are available through an unrestricted gift from IBM Research. Further development of the CSG (specifically the use of RAxML Light) will be provided by a sub-contract with the iPlant Collaborative (www.iplantcollaborative.org). The iPlant team is currently supporting the development costs of adding the RAxML-Light tool to the CSG through a sub-contract to SDSC, and is also supporting integration of the capabilities of the CSG into the iPlant Discovery Environment.

*Sustainability:* We plan to continue developing proposals for NSF support for the CSG as a community resource, engage the community in development of new portal features, and explore alternate funding models that will sustain the web application for the future. We have recently collected responses from the user community to various options for obtaining funds that will sustain the service, and asking user feedback on methods that will improve sustainability by minimizing expenses in the future.