Flipping the priority: effects of prioritising HTC jobs on energy consumption in a multi-use cluster

Matthew Forshaw
School of Computing Science
Newcastle University
Newcastle upon Tyne, UK
matthew.forshaw@ncl.ac.uk

A. Stephen McGough
School of Engineering and Computing Sciences
Durham University
Durham, DH1 3LE, UK
stephen.mcgough@durham.ac.uk

ABSTRACT

High Throughput Computing (HTC) through the use of volunteer computing provides a compelling opportunity to perform large volumes of computation without the need to invest in computational resources. This relies on the good will of computer owners who volunteer their computer idle time. In scenarios when there is contention the HTC system will relinquish the computer – normally achieved through termination or suspension. This often leads to longer turn-around times for the HTC jobs and an increase in energy consumption when the work is restarted elsewhere. However, within large organisations this clear distinction of who is the owner of a computer and who should take priority is not always clear. If a user enters a large cluster room (of identical computers) should they be allowed to use a computer which is servicing HTC jobs when other computers are idle? Should HTC jobs be able to delay the rebooting of computers thus allowing the jobs to complete? In this work we relax some of the common policies used in management of computers in large organisations in order to evaluate if alternative policies which may have impact on the primary users of the computers could save enough energy to make this impact tolerable. We evaluate our approach through the use of the HTC-Sim simulation framework. We demonstrate a potential energy saving of 12.4%, along with overhead reductions of 20-74%, and up to 48% reduction in job terminations by re-directing only 13% of users away from computers servicing HTC jobs.

Categories and Subject Descriptors

H.3.4 [**Information Systems**]: Systems and Software—*Performance evaluation (efficiency and effectiveness)*; I.6.8 [**Simulation and Modelling**]: Types of Simulation—*discrete event*

General Terms

Simulation, Energy, HTC, Volunteer Computing, Optimisation

1. INTRODUCTION

The energy impact of IT infrastructures is a significant factor for many organisations. Approaches to improve the energy-efficiency of datacentre computing have been studied extensively [3], however, energy impact of computers used by end-users (forming the so-called *Desktop estate*) have received only superficial attention. Many organisations seek to achieve energy savings by adopting aggressive power management polices applied across these computers – such as automatic shutdown after relatively short idle periods.

High-Throughput Computing (HTC) systems are a popular choice for making use of idle computers within existing infrastructure, offering significant computing without the capital expense. They have been extensively used to exploit desktop estates [2, 10]. Traditionally these systems work in a manner which prioritises the main user of the computer – if the user of the system tries to use the computer then the HTC system will cease activity – a consequence of who has purchased the hardware. Likewise the cluster may be rebooted regularly (e.g. nightly) to 'freshen' computer states and ensure the latest patches / software is available to the interactive users. However, this simple rule of priority may not be the most sensible – a cluster of computers may be purchased for both HTC and interactive users, the computers may be powerful enough to run HTC and user activity at the same time, computer reboots could be delayed to allow HTC jobs to complete or given a cluster room full of computers it would seem reasonable to direct interactive users to computers unused by HTC before interfering with HTC jobs. It is the last two cases that we focus on in this paper.

In an environment where multiple computers exist within the same physical location – a cluster – with each computer having equivalent functionality, a user entering a cluster will select a computer based on some personal preferences. This may be preferential locations (heating, cooling, light), close to friends, or in an area unoccupied by others. If the selected computer is running an HTC job (referred to here as a job) the normal course of action is to relinquish the computer to this user. This may be achieved through job eviction, suspension or the migration of the job to an alternate computer. This will have a detrimental impact on the submitter of the job and the owner of the cluster. For the HTC submitter this will increase the overheads (time in excess of the execution time for a job within the HTC system) whilst for the cluster owner this will increase the energy consumption through partial runs.

Given that the cluster room is not full – a situation under which HTC workload would not normally be present – the user could be informed that the computer they have selected is not available for use. This could be achieved through the use of a display screen indicating such or some form of information board displayed at some location within the cluster. Users could either select an alternative computer or be guided to a specific computer. This policy would, obviously, need to relinquish computers to the user given the situation where all computers are now either serving jobs or have interactive users. Again, which computers should have their HTC workload terminated is a critical decision. As our focus here is an evaluation of the potential energy and overhead reductions possible we do not concern ourselves with the practical implication details here. We model an organisational scale system in which a number of clusters exist, assuming that users will tolerate being moved to another computer within the same cluster but would not tolerate being moved to a different cluster.

The regular – and forced – rebooting of computers within a cluster has many benefits from the interactive user's point of view. Many erroneous states that the computer may enter due to user activity can be cleared and the system can be set to perform automatic updates and install new software immediately upon reboot. Thus offering a more reliable and consistent service to the interactive users. However, this can have a detrimental impact on the HTC users. Reboots are scheduled for times of the day when it is anticipated that there will be low (if any) interactive users present – normally the early hours of the morning. During these hours the reboots are likely to have significant impact on jobs as lack of interactive users equates to times when jobs can run. If we can defer these reboots till just before cluster openings we can remove the impact on the jobs whilst still maintaining the quality of service for interactive users. Likewise it may be preferable to reboot an idle computer as early as possible to allow jobs which arrive later to keep running – even into the time when the cluster is open.

In this work we extend our HTC simulation system HTC-Sim [7] in order to investigate the impact on the system, in terms of energy saved and reduction of average overheads, by re-directing users away from computers running HTC workload and delaying nightly reboots. We acknowledge that there will be an impact on the interactive users due to the 'annoyance' caused by not being able to log into the first computer that they choose. However, as this requires an understanding of how much this will effect the user we see this as out of scope for our work here. We position this work to evaluate if such a further study would prove worthwhile.

The remainder of this paper is structured as follows. In Section 2 we present motivational evidence for performing this work. Section 3 discusses related work while in Section 4 we present our simulation model. Section 5 discusses the metrics we use in this paper to evaluate the policies we present in Section 6. We introduce our trace-driven simulation environment in Section 7. Experimental results are shown in Section 8. Conclusions and plans for future work are presented in Section 9.

2. MOTIVATION

In this section we analyse the characteristics of the HTCondor system at Newcastle University, based on our logs from 2010, in order to identify potential scenarios where moving users or moving reboots would show a benefit.

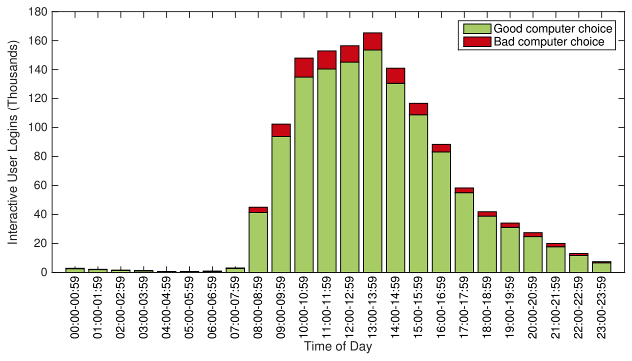

Figure 1: Interactive user logins by the hour

2.1 Moving interactive users

In order to motivate the policy of moving users between computers to reduce the impact on jobs we will investigate here the impact of user logins classifying them as either:

Bad computer choice: where a user selects a computer running a job but there is at least one computer in the cluster which is not servicing a user or job.

Good computer choice: where a user selects a computer which is either not running a job, or the computer is running a job but there are no computers in the cluster which are not servicing users or jobs.

Although we cannot determine this information from our trace-logs of user interaction and HTC workload – as the HTCondor logs only capture the final successful computer for a job – we are able to use HTC-Sim with default configuration in order to determine the impact. Figure 1 illustrates the total number of logins which occurred during each hour of the day during 2010. Of these the vast majority are 'good' computer choices with only a small proportion of 'bad' choices. There is a clear pattern of user activity with the quietest time being 5-6am, with only 67 'bad' choices, whilst 1-2pm is the busiest time with 11,838 'bad' choices made during this hour in 2010. Figure 2 indicates the percentage of 'bad' user choices made during the day. The highest values occur as the time approaches 8-9am as this is the time when users start to arrive while the HTC system

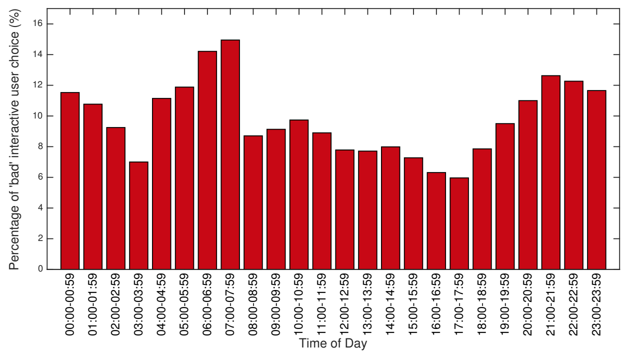

Figure 2: Percentage of 'bad' interactive user choice

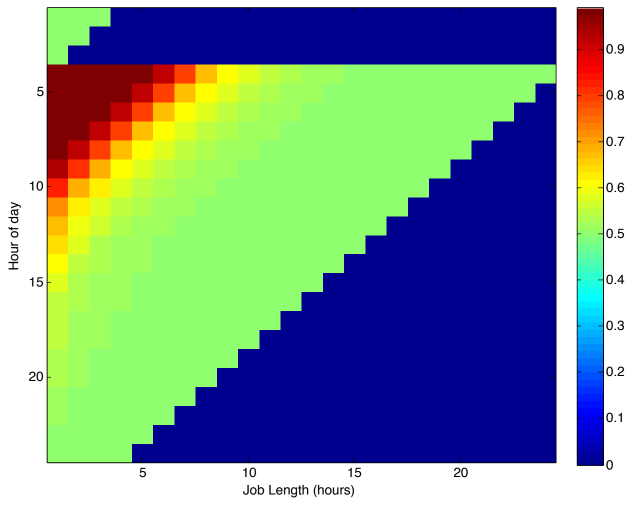

Figure 3: Probability job will complete per hour

has been able to exploit these computers overnight so will have a large number of jobs running. Steering users away from computers running jobs would give these computers a chance to complete the jobs before the main influx of users. Likewise, after 6-7pm the number of 'bad' choices increases most likely a consequence of lower cluster use allowing HTC jobs to start, then when users arrive they interrupt these jobs. Again since the number of interactive users is much lower than other times of the day moving the users around could give a great advantage.

In all hour intervals shown in Figure 2 the percentage of 'bad' choices is relatively small – ranging between 6% and 15%. Thus implementing a policy where users are guided to more preferable computers would seem to make sense as this could reduce energy consumption whilst at the same time would only affect a minority of the users. It should be noted that saving a particular job from a user does not now guarantee that the job will run to completion as future users may still cause the job to be evicted. However, if we prioritise to 'keep' longer-running jobs then this ensures the best chance for these jobs.

2.2 Moving reboots

Figure 3 shows the probability that a job of length x hours will complete given that it is submitted during hour y of the day. Note that this is assuming that no other jobs are running at the time and should therefore be considered as an overestimate of the probability. As all computers are rebooted at 3am this leads to the diagonal cut-off within the heat map going from a 50% chance of completion to 0% in the lower right hand side of the figure. There is only one hour slot under which a 24 hour job can complete – when started immediately after a computer reboot at 3am. The highest chance of short jobs completing successfully being between 3am and 8am – as this is the most idle time of the day for interactive users.

This 3am cut-off time would seem highly detrimental to jobs - especially since midnight through till 8am would appear to be the most ideal time for jobs to be executed. Placing a forced reboot in the middle of this interval is effectively halving the runtime for potential jobs.

3. RELATED WORK

A HTC system requires the careful coordination of a number of competing polices where each policy set reflects the wants and desires of different entities within the system. Livny and Raman [11] identify six different layers of policy within a HTC system: local, owner, system, customer, application resource management and application. The policies we seek to modify within this work relate to the owner layer.

Liang *et al* [9] explore the management of a dual-boot HPC cluster based on multi-use student clusters at the University of Huddersfield, such that computers may be booted into Windows for use by students, or into Linux to service HPC workloads. Students arriving to computers which are booted into a Linux instance would be subject to delays while the machine reboots back into a Windows mode. Therefore, the selection of resources to run HPC jobs is important not only to the cluster owner but also to interactive users of the cluster. We see such a dual-boot approach as being complimentary to our own, and a class of policies which we could model and evaluate trivially within HTC-Sim.

In [12] we explore the impact of a number of cluster management policies governing the powering up and down of computers within clusters. We go further to evaluate the performance and energy consumption of an HTCondor high-throughput system deployed upon these resources.

Li *et al* [8] demonstrate through some 30 experiments that allowing HTC workloads to co-exist on the same hardware as interactive users can save between 33% and 52% energy without significant impact to the interactive users when using modern multi-core systems. We see this as a good approach and have shown similar results in previous work [12].

University College London uses a thin-client system for its open access computers though runs the client on a full-spec computer. This allows them to exploit the unused computing power of the computers as part of their high-throughput computing system. Though in itself this doesn't reduce power consumption the ability to use computers for multiple purposes simultaneously helps them cut down on capital expense and maintenance.

Stefanek *et al* [14] demonstrate a quadratic relationship between the CPU load of a desktop computer and the power consumed, they go further to postulate how this could be used for scheduling in a HTCondor environment. We see this as complementary work to our own.

Both Bouguerra *et al* [4] and Forshaw *et al* [6] propose strategies for when to take checkpoints within HTC systems. We see this as an appropriate solution given that a user cannot be re-routed to a different computer.

4. SYSTEM MODEL

We provide a brief overview of the HTC-Sim system and our extensions allowing the movement of interactive users between computers. A more complete description of HTC-Sim can be found in [7]. Figure 4 depicts the overall architecture of our system. Two different types of user interact with the system – Interactive users and High-throughput users. Interactive users will interact with a computer directly whilst a high-throughput user will interact through a high-throughput management service. Computers within the system are logically grouped into clusters of equivalent specification and location. Computers may be sent to sleep if

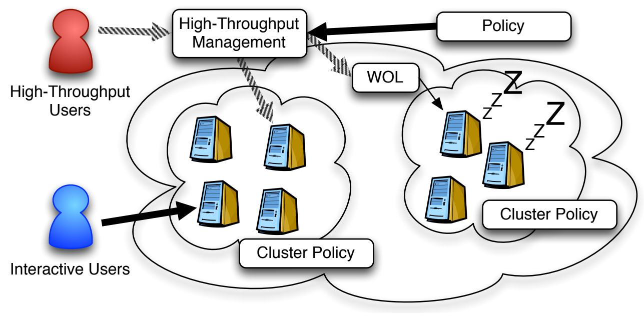

Figure 4: The HTC-Sim Architecture

idle for a period of time. Both iterative users and the high-throughput management system are able to wake sleeping computers. Policies which enact the wants and desires of different entities within the system are present through different policy sets acting over parts of the system. Here we can vary a policy set in order to evaluate how it changes the performance of the system.

The HTC-Sim model comprises of four main entities: Computers, Interactive Users, HTC Jobs and the HTC management system:

Computers are realised as entities within the model, they are associated with a *cluster* within the organisation. Clusters are a mechanism to group together computers which are co-located and share the same physical parameters – often being provisioned at the same time. Computers can be in a number of states, each with associated energy consumptions: sleep, idle or active. Where active indicates use by an interactive user or executing a HTC job – in either of these cases the computer will be linked with the Interactive User entity or Job entity. Computers within a cluster will share a set of policies governing how they are used – such as idle time before going to sleep or when to perform reboots. All energy consumption statistics for computers are collected globally.

Interactive Users are the primary users of the system and are represented by a tuple $\langle s_i, c_i, u_i, e_i \rangle$, where s_i and e_i are the login and logout timestamps respectively, c_i is the name of the computer, and u_i is a hash of the interactive users identity.

Interactive User entities will link themselves with the appropriate computer and in general will evict any job entity linked with that computer – though the simulation allows policies where Jobs are not evicted due to interactive users.

In this work we allow the value of c_i to be changed at the time a user arrives. Different policies can be enacted at the time of a user arriving which may change the selected computer. However, as computers are part of a cluster we restrict movement to other computers within the cluster.

HTC jobs are entities and are represented by the tuple $\langle j, b, q, d, h, e, u, d \rangle$, where j is the identifier of a job (or batch of jobs), b is the identifier of a job within a batch (if present), q is the time the job was submitted into the system, d was the job duration, h is the hash of the user who submitted the job, e is the HTC result state of running the job (either 'success' or 'terminated') and u, d represent the data transfer to and from the resource which ran the job. Note that if a job was terminated then d represents the time that the job termination was submitted. The HTC job will be managed by the HTC management entity and may be associated with computers at various stages.

HTC management One HTC management entity will take in each HTC job at the time of submission and attempt to execute it on computers within the system. Policies govern when a job can be executed on a computer (such as when the computer does not have an interactive user) and when to evict a job (such as when an interactive user logs in). The HTC system will repeatedly attempt to execute jobs on computers until either the job obtains enough time on a computer to complete, the job is terminated or the job violates some policy associated with the HTC management system (such as being re-submitted to computers for execution too many times). The HTC management system captures statistics on jobs run.

5. METRICS

In evaluating our policies we report on a number of metrics for both the users of the system and the HTCondor environment.

Average job overhead is defined as the time difference between the job entering and departing the system, and the actual job execution time:

$$\frac{\sum_{j}^{j \in J} (f_j - s_j - r_j)}{|J|}$$

for a set of jobs J, where s_j, f_j and r_j are the arrival time, departure time and runtime of job j respectively.

Proportion of interactive user migration: In this work we seek to minimise the impact of our policies on the interactive users. For each of our policies, we record the proportion of user sessions who were re-assigned to a computer other than the computer specified by c_i in their tuple, against those where we were able to use the original computer. We acknowledge that further metrics should be included here which take into account the degree of inconvenience which is incurred by an interactive user who is moved onto a different computer and see this as future work.

Energy consumption Here we concern ourselves only with the amount of energy consumed through the execution of HTC jobs within the system. This includes energy consumed for productive jobs – which complete execution on a computer – and unproductive jobs – where jobs are evicted or the job is terminated by the HTC user or the system administrator. By also including the energy incurred through data transfer we compute the total energy impact on the system from running a HTC system. We compute the HTC energy impact as:

$$\sum_{c=0}^{n} t_c E_c \tag{1}$$

where n is the number of computers, t_c is the time spent by computer c servicing HTC jobs and E_c is the energy consumed by computer c in an active state.

Job evictions is the number of HTC jobs evicted by either the arrival of interactive users, or a computer reboot.

6. POLICIES

In this section we discuss a number of policies which can be applied to manage a multi-use cluster. These policies govern the approach to rebooting machines, assigning newly arriving users to computers within a cluster and to the power-management policies for the computers.

6.1 Reboot Policies

We outline a number of reboot policies which we have evaluated as part of this work:

RB1 Machine reboots occur according to the cluster management policies enacted in 2010, with all cluster machines rebooting between 3am and 5am.

RB2 Machine reboots are scheduled to occur when clusters close for the night. This maximises the potential time for HTC jobs to run though does have a detrimental impact on jobs running at cluster close time. In the case of clusters which do not have a close time this policy schedules reboots at midnight.

RB3(n, r) As with RB2, but if an HTC job is currently running on the machine the reboot will be deferred until n minutes before the cluster reopens. In a real system many simultaneous reboots could impose significant load on the network and storage nodes. In order to mitigate these potential effects, we introduce a random component in the reboot scheduling, varying the reboot time of each computer by η, where η is uniformly distributed on $[-r, r]$. As the value of r increases, the system will become less susceptible to large number of simultaneous reboots, but will lead to a greater impact on HTC jobs running on these resources.

RB4 As part of its Green IT strategy, Newcastle University now make available *baseline* power saving scripts [1] which may be deployed to computers as part of the managed desktop estate. These policies are intended as a starting point for use by computing officers managing departmental clusters within different Schools and Faculties. Under this policy, active machines are polled every 10 minutes and are suspended if there is no interactive user present, and the CPU is determined to be idle. Computers are scheduled to reboot (and wake from sleep if required) at a random time between 01:00 and 06:59, before returning to sleep. These policies are indented to be *deliberately conservative* in their hibernation of computers, and computing officers are encouraged to lower thresholds as appropriate for the environment they are deployed in. We evaluate these baseline policies across all clusters without taking into account particular preferences or customisations made on a cluster-by-cluster basis, rather to evaluate the suitability of these recommendations for widespread deployment.

6.2 User allocation policies

We present a number of user allocation policies for where to allocate users. Note that for all policies we only allow users to be moved around within the same physical location:

U1 Exact. Users arrive to the computer specified in our trace data for 2010. This is represented by c_i in the interactive tuple.

U2 Random. Users are allocated to their original computer choice (c_i) if this computer is not currently occupied with a job or interactive user. Alternatively, an idle or sleeping computer is selected at random from the same cluster. If no idle or sleeping computer is available then a computer running a job is selected at random. Note that as we don't allow users to be moved to a different cluster at no point can we fail to find a computer to place the user. This provides a benchmark against which other policies are evaluated.

U3(n) The user is allocated to the computer they specify (c_i) if the computer is idle, sleeping, or has a HTC job with a runtime less than n minutes. If they are re-assigned they are placed onto a computer which is idle or sleeping, if available. If no available computer is found they are placed onto the computer with the HTC job that has the shortest accumulated running time. Thus users should have less impact on HTC jobs and if they do it will be against jobs that have accrued only a short amount of execution time.

U4 In our implementation environment there are a number of clusters which are defined as being discrete, but which are based in the same physical location. These include large flat-floor teaching spaces which are further divided into *zones* – used to distinguish discrete teaching areas. This policy extends Policy **U3**(n) by relaxing the requirement that a user can only be re-assigned to a computer within the same cluster and instead allow users to be re-assigned to computers within the same physical location. In these situations moving between adjacent clusters can be as quick as moving within a cluster.

6.3 Computer Power management

Power management of computers covers when a computer can be awake (active or idle) and when the computer can sleep. Originally investigated in [12], the four power management policies evaluated here are:

P1 Computers are permanently awake. This was the default policy used by most high-throughput computing installations before the introduction of power saving. This policy can lead to large amounts of wasted energy when a computer is idle though as computers are always available it minimises overheads.

P2 Computers are on during cluster opening times or powered off otherwise with no ability to wake up. If the computer is servicing jobs at cluster close time it remains active until this and any currently queued jobs are completed. This can have significant impact on overheads for jobs which arrive while computers are powered down.

P3(n) Computers sleep after n minutes of inactivity with no wakeup for high-throughput jobs. Initially we used a value of one hour as the resume time from shutdown was significant. However, the reliable sleep feature of Microsoft Windows 7 has made this process almost instantaneous though at present we still use the one hour time interval. This again can lead to significant overhead increases for HTC jobs.

P4(n) Computers sleep after n minutes of inactivity with HTC being made aware of their availability. This is an extension of policy P3 allowing the HTC system to wake up computers when needed. This policy now mitigates the effect on overheads by allowing computers to be powered on.

7. IMPLEMENTATION

We extend our HTC-Sim simulation framework to support a new class of policy decision, one which governs the selection of the computer for an arriving interactive user. This is implemented through the use of a Java class interface allowing for easy development of new policy implementations. Each of the user selection policies outlined in Section 6.2 have been implemented using this interface. Likewise an interface has been implemented for the reboot policies allowing each of the reboot policies outlined in Section 6.1. The power management interfaces already existed within the codebase. Along with the use of a configuration script which selects which policy implementations to use allows us to quickly evaluate different policy configuration scripts for our test environment.

7.1 HTCondor at Newcastle University

We drive our trace-driven simulation based on our HTCondor and interactive user traces from Newcastle University in 2010 – Figures 5 and Figure 6 respectively. Full details of these traces can be found in [5, 7]. The traces are based on 1,359 student access computers, distributed around the Newcastle campus in 35 clusters, some clusters dedicated for teaching and others open access, with each cluster exhibiting a different interactive usage patterns. Computer energy consumption is based on a tuple $\langle a, i, s \rangle$, where a, i and s are the energy rates for active, idle and sleep. Three computer types were present at the time as depicted in Table 1. Normal computers were the low energy computers purchased for most clusters, while high-end computers had more computational and graphics capability for CAD work. A number of clusters still comprised of older computers – due to the policy of replacing computers on a four-year rolling cycle. As these computers pre-dated the university plan to reduce energy consumption they tended to have higher energy consumptions. As our focus is the comparison of different polices for reducing energy we ignore performance differences between computers and assume the execution time will match the original execution time.

Type	Cores	Speed	Power Consumption		
			Active	*Idle*	*Sleep*
Normal	2	~3Ghz	57W	40W	2W
High End	4	~3Ghz	114W	67W	3W
Legacy	2	~2Ghz	100-180W	50-80W	4W

Table 1: Computer Types

These computer clusters are provisioned for the needs of the primary (interactive) users of the system. Students generally demand Windows-based machines so the proportion of resources capable of checkpointing (i.e. Linux) is limited. At Newcastle University, Linux-based machines constitute only ~5% of resources available to HTCondor.

The HTC-Sim models the bandwidth available between computers, imposing time delays on data ingress/egress. Further details on our bandwidth modelling can be found in [13].

8. EXPERIMENTAL RESULTS

Figure 7 *(a)* shows the impact of each policy on task overheads. We observe that policies P1 (Computers are permanently awake) and P4 (Computers sleep after n minutes with HTC being made aware of their availability) provide significantly lower overheads for HTC jobs compared to policies P2 and P3, which lack the ability to wake computers to service HTC jobs. In each case we find RB2 and RB3 provide lower average overheads compared to the reboot strategies employed in 2010. With the exception of Policy P3, the greatest overhead reduction is seen by RB3; however RB2 outperforms RB3 under policy P3. This is likely to be a consequence of the rebooting making all computers in the cluster which were sleeping and unavailable to the HTC system now available. Compelling overhead reductions of 18.7-74.7% are demonstrated by varying reboot policy, with the largest saving sought by policy combination P3+U1+RB2.

Furthermore, we find Newcastle University's baseline power saving policies (RB4) which reboots randomly between 01:00 and 06:59 to be roughly equivalent with RB1; highlighting a general issue with policies which reboot during the night, and consequently interrupt long idle periods suitable for running long-running tasks.

Figure 7 *(b)* shows the energy consumption associated

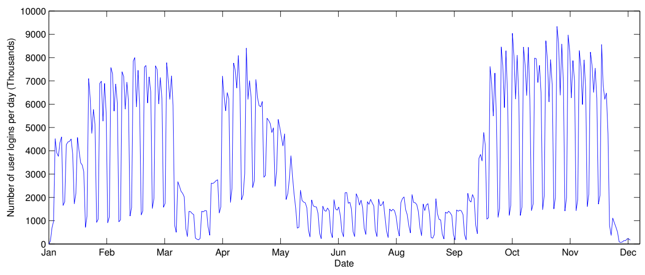

Figure 5: Interactive user activity trace for 2010

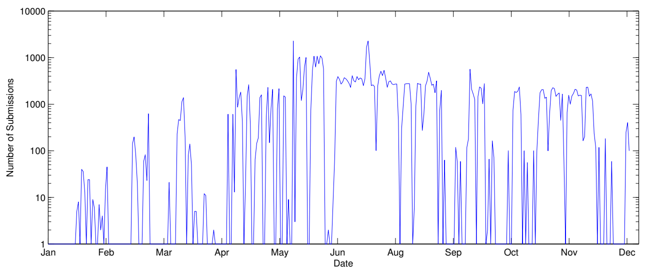

Figure 6: HTCondor workload trace for 2010

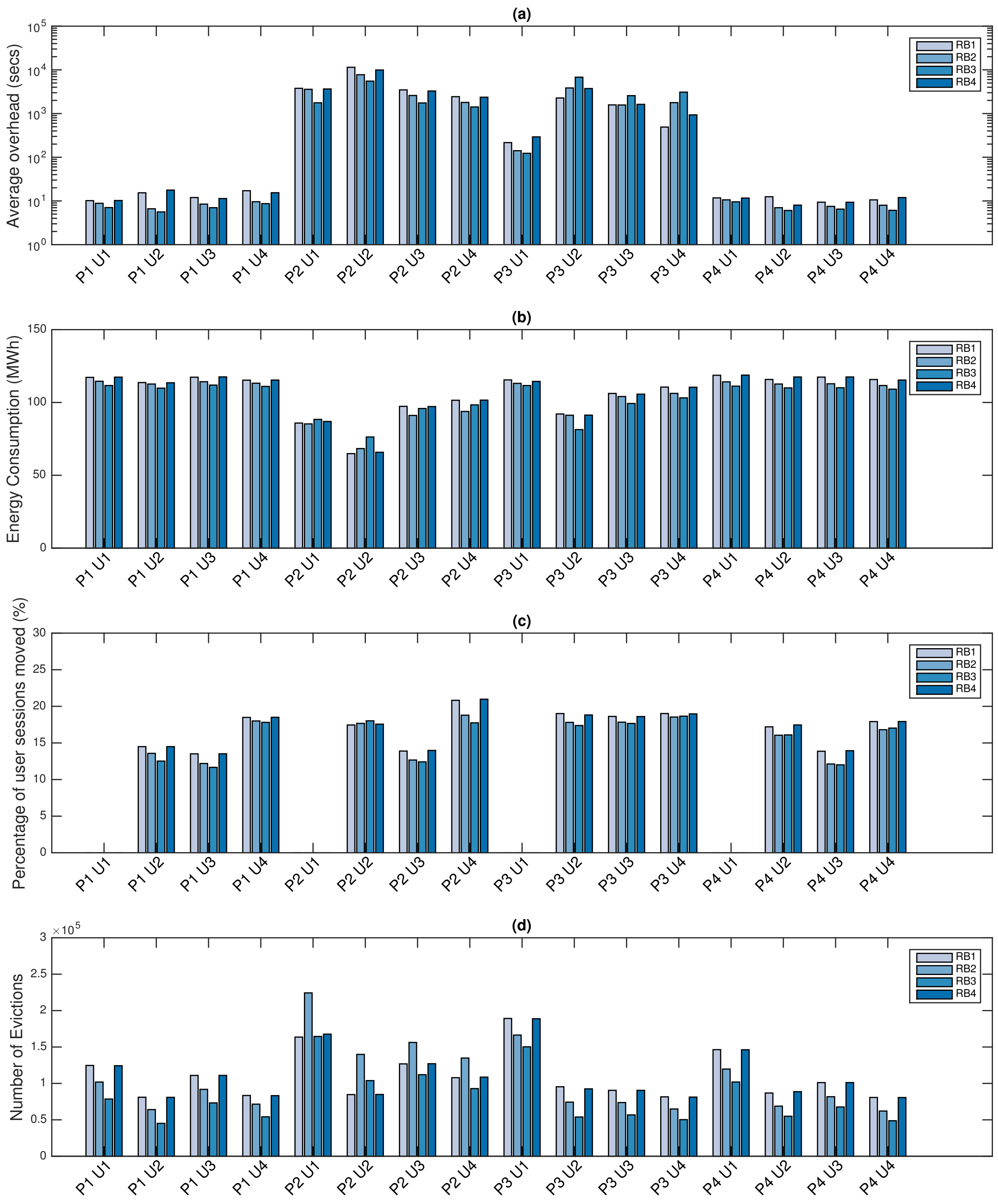

Figure 7: Impact of Reboot, User allocation, HTC availability and Computer Power management policies on average HTC job overheads, energy consumption, percentage of user sessions moved and number of HTC job evictions.

with each of our policies. Policies P2 and P3 perform significantly better, but it is evident from Figure 7 *(a)* that the inability for the policies to wake machines imposes a significant detrimental effect on overheads for the HTC jobs. Reboot policies RB2 and RB3 demonstrate improvements up to 7.6% in many cases, with the largest saving of 12.4% for policy combination P3+U3+RB2.

Figure 7 *(c)* shows the impact of each policy on the number of user sessions which are moved as a consequence of our policy, i.e. user sessions whose destination machine differs from c_i specified in the session tuple. Intuitively, Policy U1 results in zero user sessions moved in all cases, but we see from Figure 7 *(d)* that policy U1 always results in a significantly greater number of HTC job evictions. Policies U2 and U3 are shown to have a moderate impact on user sessions, with an average of 15.6% and 13.6% respectively. Policy U4, which allows the movement of user sessions between co-located clusters, has a much more significant effect, with an average of 18% of user sessions moved.

We see Reboot policies have an impact on the percentage of user sessions moved. Policies RB1 and RB4 exhibit similar levels of user sessions moved, which are relatively higher than for the other policies because the reboot scheduled through the night allows HTC jobs to be allocated to computers. The level of user sessions moved is lower for RB2 which schedules reboots for the closing of clusters, as this makes available a significant idle period through the night which aids longer-running HTC jobs in completing. Policy RB3 exhibits the lowest levels of user interruption in all cases, as it performs reboots immediately before clusters open, clearing computers of HTC jobs which may otherwise have caused the user to be moved. For Policy RB3 we introduce a random component r to mitigate the impact of large numbers of reboots occurring at the same time. We observe negligible impact varying r and in aid of space omit figures from this paper.

Figure 7 *(d)* shows the number of jobs which are evicted from computers. While Policy U1 (Exact) has exhibited low impact on the interactive users of the system, here we see the number of evictions to be significantly higher for this policy. The impact of reboot policies RB1 and RB4 is shown to be similar in all cases. RB2 offers an improvement in the number of job evictions, due to the lower likelihood of an HTC job running at cluster closing times, owing to prior contention over computers with students. RB3 offers further improvements by deferring the reboot in the presence of an HTC job. RB2 and RB3 are shown to outperform both RB1 and RB4 in most cases, with the exception of under policy P2 where policy P2 will mean most computers have gone to sleep, after completing jobs, before reboots caused by RB1 and RB4 occur – reducing job evictions, whilst for RB2 any job running at cluster close time will be evicted.

9. CONCLUSIONS

In this paper we have explored, through trace-driven simulation, the impact of relaxing commonly adopted policies governing the operation of volunteer HTC clusters; those of steering users to computers not running HTC jobs – to reduce job terminations, and moving the scheduling of routine machine reboots.

Our experimental results demonstrates significant reductions in the overheads incurred by HTC jobs (20-74%) and reduction of energy consumption (12.4%) by the HTC system may be obtained, with relatively little impact on the interactive users of the computers. It should be noted that these improvements can be used simultaneously with other energy saving policies with cumulative savings.

We highlight the importance of communication among the operators of campus clusters and high-throughput computing system managers. This motivates future work in the area of operating policies for HTC systems which reconcile the different, and often opposing, demands of the cluster owner, HTC submitter and interactive user.

10. REFERENCES

[1] Newcastle University Green IT Reporting Tools. `http://www.ncl.ac.uk/itservice/green/reporting-tools/`.

[2] D. P. Anderson. Boinc: A system for public-resource computing and storage. In *Grid Computing, 2004*, pages 4–10. IEEE, 2004.

[3] A. Beloglazov, R. Buyya, Y. C. Lee, A. Zomaya, et al. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in computers*, 82(2):47–111, 2011.

[4] M. Bouguerra, D. Kondo, and D. Trystram. On the scheduling of checkpoints in desktop grids. In *IEEE/ACM CCGrid*, pages 305–313, 2011.

[5] M. Forshaw. *Operating Policies for Energy Efficient Large Scale Computing*. PhD thesis, Newcastle University, UK, 2015.

[6] M. Forshaw, A. S. McGough, and N. Thomas. Energy-efficient checkpointing in high-throughput cycle-stealing distributed systems. *ENTCS*, 310(0):65–90, 2015.

[7] M. Forshaw, N. Thomas, and S. McGough. Trace-driven simulation for energy consumption in high throughput computing systems. In *IEEE/ACM DS-RT*, 2014.

[8] J. Li, A. Deshpande, J. Srinivasan, and X. Ma. Energy and performance impact of aggressive volunteer computing with multi-core computers. In *IEEE MASCOTS '09*, pages 1–10, Sept 2009.

[9] S. Liang, V. Holmes, and I. Kureshi. Hybrid computer cluster with high flexibility. In *IEEE CLUSTER 2012*, pages 128–135, Sept 2012.

[10] M. Litzkow, M. Livney, and M. W. Mutka. Condor-a hunter of idle workstations. In *ICDCS '88*, pages 104–111, 1988.

[11] M. Livny and R. Raman. High-throughput resource management. *The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann*, pages 311–337, 1999.

[12] A. S. McGough, M. Forshaw, C. Gerrard, P. Robinson, and S. Wheater. Analysis of power-saving techniques over a large multi-use cluster with variable workload. *Concurrency and Computation: Practice and Experience*, 25(18):2501–2522, 2013.

[13] A. S. McGough, M. Forshaw, C. Gerrard, S. Wheater, B. Allen, and P. Robinson. Comparison of a cost-effective virtual cloud cluster with an existing campus cluster. *FGCS*, 41(0):65–78, 2014.

[14] A. Stefanek, U. Harder, and J. T. Bradley. Energy consumption in the office. In *Computer Performance Engineering*, pages 224–236. Springer, 2013.