Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 318 (2015) 5–17

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Time-inhomogeneous Population Models of a Cycle-Stealing Distributed System

Jeremy T. Bradley[a] Matthew Forshaw[b] Anton Stefanek[a] Nigel Thomas[b]

[a] *Department of Computing*
Imperial College London
London, UK

[b] *School of Computing Science*
Newcastle University
Newcastle upon Tyne, UK

Abstract

Organisations such as research institutions and universities often increase utilisation of their office workstations by deploying a high-throughput cycle-stealing distributed system. Such systems allow users to submit a large number of computing tasks into a central pool. The system observes activity of workstations and continually assigns tasks to idle machines. When a user becomes active on the machine, the scheduler interrupts the task execution. This approach can significantly increase utilisation of the resources. However, it can also lead to wastage of computing cycles if tasks get interrupted too often.
In this paper, we develop a detailed Population Continuous Time Markov Chain (PCTMC) model of the whole system that accurately captures the contention between the interactive users and high-throughput tasks. The PCTMC framework is well suited to the inherently time-inhomogeneous nature of the user behaviour and allows to capture a large number of performance and energy consumption metrics. We fit the PCTMC model to real data and propose a methodology to forecast cluster availability in the near future. We show how to use historically collected and live data to parametrise the PCTMC model and use efficient fluid analysis techniques to predict the desired metrics. Additionally, the fast analysis enables exploration of various what-if scenarios. We demonstrate a working implementation of the method using the existing GPA tool for analysis of PCTMC models. We argue that this methodology could allow the system maintainers to optimise the energy and performance parameters of the system. Moreover, it would benefit the users who could use the model forecasts to better distribute and plan their large scale computations.

Keywords: Cycle-stealing, distributed systems, Continuous time Markov chain, PCTMC

1 Introduction

Distributed high-throughput cycle-stealing clusters such as BOINC or HTCondor (previously known as Condor) [9] are a popular computation model amongst scientists, engineers and financial organisations, allowing massively distributed calculations to be spread over many thousands of otherwise idle workstations. Knowing when to initiate your job on such a cluster so that it might execute quickly depends on knowing the current state of the cluster, as well as the likely load contention at

http://dx.doi.org/10.1016/j.entcs.2015.10.016

that time of the day or week. The fact that demand for such a service fluctuates, often periodically, requires more involved modelling techniques. Similarly the size of many clusters would eliminate most traditional explicit state-space techniques, so new advances in stochastic model analysis will need to be called upon as well. For this modelling challenge we will use population-based models and an associated fluid analysis technique to deal with the large state spaces and time-inhomogeneous rates so as to capture the time-varying load of the application.

Fluid analysis of population models is a very useful technique for engineering performance in large distributed systems [5,7,15,16,17]. In the past, we might have created a simple Markov chain to capture system dynamics, but this usually had the drawback of producing models that were far too large to analyse in reasonable time. Now we have the potential to analyse Markov models by translating them to systems of differential equations which are much more tractably solved for much larger systems using tools such as GPA [13]. More recent developments in fluid analysis permit response time distributions to be extracted from population-based performance models [6]; accumulating rewards to be computed using fluid analysis [12]; and time-inhomogeneous rates [14]. Putting all these together means that we are in a position to derive service response time metrics of large distributed systems while modelling cost and energy usage and under variable load conditions.

In this paper we will briefly introduce PCTMCs as a population modelling tool in Section 2 and fluid (or ODE) analysis of PCTMCs. We will focus the remainder of the paper on the experimental setup of the HTCondor cluster deployed at the Newcastle University in Section 3 and the construction and analysis of a parametrised PCTMC model of the cluster in Section 4.

1.1 Related Work

The effects of cycle-stealing and of similar computation distribution schemes have been extensively researched in the past. Kelly *et al* [8] present a multi-class closed queueing model of a cycle-stealing system. Estrada *et al* [2] developed a detailed simulation by emulating real software components of the BOINC volunteer computing middleware to investigate different scheduling policies. Our fluid analysis approach is related to the work of Gast *et al* [3] describing a mean-field model of work-stealing algorithms in computational grids. Recently, Guenther *et al* [4] have proposed a forecasting algorithm using PCTMC models in the context of bicycle sharing schemes. They have shown a comparable performance to traditional time-series analysis techniques.

The contention between the high-throughput HTCondor jobs and interactive users is similar to the situation in data centers where where low priority "background" tasks are scheduled during idle times, without affecting performance of high priority "foreground" tasks. The framework of Mi *et al* [11] estimates the best time window to utilise idle resources while maintaining performance guarantees for the foreground tasks.

Another way to reduce wastage of computation in a HTCondor cluster is to detect "bad" tasks which are incapable of completing. McGough *et al* [10] investigate

different techniques to curtail execution of these tasks to provide more computation to "good" tasks. Their simulations predict that an order of magnitude energy saving is possible.

We believe that our method has advantages in its flexibility and the possibility of computationally cheap analysis. The PCTMC framework is convenient for various extensions and modifications of the model. The model we present in Section 4 is adaptable to different HTCondor configurations. The used ODE analysis is computationally efficient and not sensitive to the growth of the system scale. Moreover, the existing GPA tool is readily available for the analysis of PCTMC models. This gives the method the potential to be used for on-line predictions alongside a real HTCondor cluster.

2 PCTMC framework

In this section we briefly introduce the Population Continuous Time Markov Chain (PCTMC) framework and show why it is suitable for modelling the distributed system such as HTCondor. A PCTMC is a continuous time Markov chain with state space which is a subset of $\mathbb{Z}_+^N$. The N components represent positive-integer-valued *populations*. This is a compact representation for a system with a large number of components of a (relatively smaller) number of different types.

The N populations are partitioned into parts each corresponding to a different component type. The populations within each type represent different states of the component. For example, a very simple model is shown in Figure 1.

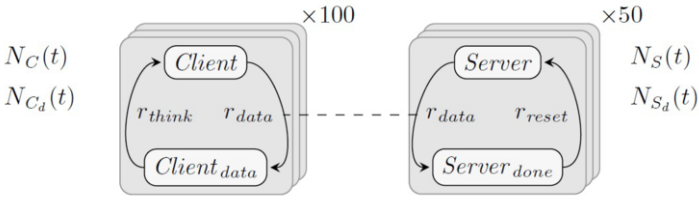

Fig. 1. A simple client/server example.

There are two component types - clients and servers, each with two possible states, *Client*, $Client_{data}$ and *Server*, $Server_{done}$ respectively. The state space at time t consists of the vector of populations of each of these four states

$$\vec{X}(t) = (N_C(t), N_{C_d}(t), N_S(t), N_{S_d}(t)) \in \mathbb{Z}_+^4$$

We assume that at each time there are 100 client and 50 server components in the system. This means that the sum of the first two populations is always 100 and the sum of last two always 50.

Behaviour of a PCTMC model is described by means of *transition classes*. Each such class c specifies a family of transitions which have an exponentially distributed delay with rate $r_c(\vec{X}(t))$ that depends on the current state $\vec{X}(t)$. Each transition, upon successful completion acts on the state by changing the populations by a

constant vector δ_c. At every point in time, a number of different transitions race for completion, with the first one determining the next evolution of the system.

For example, in the client/server model, each client can individually request data from one of the servers at rate r_{data}, after which they both change state. Therefore the corresponding transitions have a change vector $\delta_1 = (-1, 1, -1, 1)$. If we assume a so-called "bounded capacity" kinetics of the system where clients can be perfectly matched for communication with servers, we get a rate of a data transition to finish to be $r_1(\vec{X}(t)) = r_{data} \cdot \min(N_C(t), N_S(t))$. Each client then independently processes its data and returns to the initial state – this corresponds to a transition class with a change vector $\delta_2 = (1, -1, 0, 0)$ and rate $r_(\vec{X}(t)) = r_{think} \cdot N_{C_d}(t)$. Each server independently resets to the initial state – transition class $\delta_3 = (0, 0, 1, -1)$, $r_3(\vec{X}(t)) = r_{reset} \cdot N_{Server_{done}}(t)$.

There are multiple ways to make this PCTMC model more complex. Increasing the individual client and server state spaces can capture more complicated behaviour. Adding multiple client and server classes can introduce different cooperation priorities and in general make the structure of the model more heterogeneous. In such case, it is important that the model accurately captures individual connections between different types of components. This can be achieved by adding auxiliary populations which represent pairs or larger tuples of components undergoing a common "transaction".

These possible extensions give a good idea about the suitability of the PCTMC framework to our problem. In Section 4, we define a PCTMC model where components represent computers in the system, interactive users (those using the machines in usual way such as for web browsing etc.), computing tasks and the central scheduler. Each of the components has a number of states that represent the possible events in the system. Due to a non-uniform resource pool and types of tasks, the model also captures connections between different resource and task types.

2.1 Fluid analysis

Even moderately large PCTMC models with tens of components become intractable to traditional explicit state-space techniques. Often the only remaining method to analyse the model is stochastic simulation, which becomes computationally expensive as the scale of the system grows.

The fluid analysis (also called ODE analysis) addresses this problem by considering the continuous evolution of means and statistical moments of the populations instead of handling the full discrete state space of the model. The means and higher moments are shown to be well approximated by solutions to a particular system of Ordinary Differential Equations (ODEs) that can be automatically derived from the transition description of the PCTMC model.

For example, for the client/server model, the fluid analysis defines equations such as

$$\frac{d}{dt}\mathbb{E}[N_C(t)] = \cdots \frac{d}{dt}Var[N_C(t)] = \cdots$$

The computational cost of (numerically) solving the system of ODEs is much

lower than stochastic simulation, while the accuracy of the approximation is often very high.

2.2 Performance and energy metrics

The knowledge of means and higher moments of populations over time can provide the modeller with a good insight into the system behaviour. However, additional derived metrics which relate to the performance of the system are often equally or more important. For example, in the context of a HTCondor cluster, it is important to predict the expected delay before a task gets assigned to an idle computer. An extension to the ODE analysis technique allows efficient computation of various *passage time* probabilities [6].

The increased utilisation resulting from deployment of a cycle-stealing system can have significant implications on the total energy consumption of the workstations. An extension of the fluid analysis captures moments of *accumulated rewards* [12] which can represent metrics such as energy consumption. We show how the PCTMC model can be used to assess the expected increase in energy consumption when a batch of user jobs is to be submitted.

3 Collected data

In this section, we describe the structure of the data that has been collected throughout the whole of 2010 from trace logs of HTCondor deployed at the Newcastle University.

In total, around 1400 machines are taking part in the HTCondor cluster. These are workstations from various classrooms and spaces around the campus. There is a large variety of usage patterns; some machines are used for teaching and are physically inaccessible outside teaching hours, some belong to 24 hour access facilities. Some of the machines are located in halls of residence and experience higher use during weekends.

The HTCondor cluster is accessible to a large number of researchers across different departments. During the monitored period, the number of jobs submitted to the HTCondor pool has been small relative to the size of the system. This has increased since and we are hoping that a reliable forecasting tool would enable a larger variety of tasks to be run on the cluster.

HTCondor is able to keep detailed information about the state of the workstations and submitted jobs, storing every timestamped state change with additional information. Because the PCTMC framework represents the system at a more abstract level, we were able to aggregate this data into a much more compact form. Instead of storing information about individual workstations, we are only interested in the number of workstations. To capture the variety of different patterns of user behaviour and difference energy profiles of the hardware, we maintain separate populations for classes of workstations (teaching labs, library etc.).

For illustration purposes, Figure 2 shows the total number of active users (user occupied workstations) across the whole system and the number of jobs in the

system in a three week period (in October 2010).

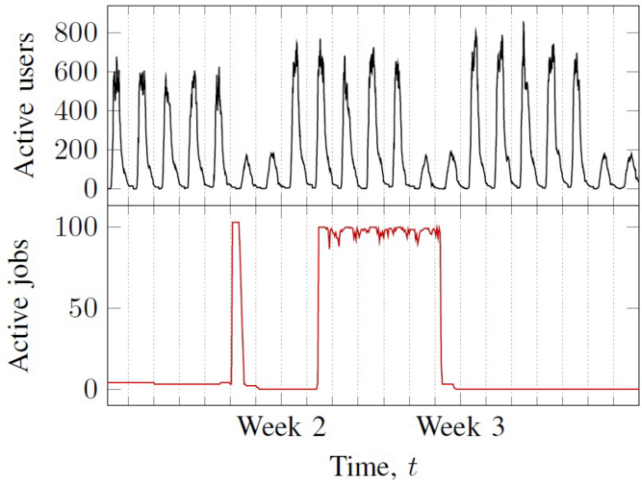

Fig. 2. A trace of the total number of active users and jobs in the system during three weeks in October 2010.

Figure 2 shows a regular periodic behaviour for the workstation usage with a period of one week. A similar pattern can be observed throughout the whole monitored period (excluding holiday times). This allows us to create a profile of an average weekly load, Figure 3.

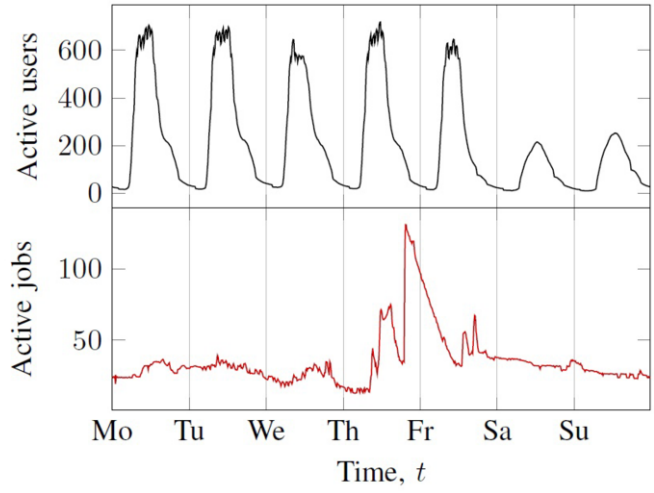

Fig. 3. Average weekly workstation usage and job submission levels.

There is a large variation in the number of jobs in the system for the sample three weeks and an average week. This reflects the fact that the job submission patterns are much less regular.

3.1 Fitting transition rates

One way to fit a PCTMC model of the user behaviour is to consider individual user arrivals and departures. These represent changes in the PCTMC populations and therefore correspond to successful transitions in the model. Each time between two successive user arrivals will be a sample from the exponential delay corresponding to the transition. We can take the samples from a time window around a time t (for example one hour) and use these to estimate the transition rate parameters for the model at time t. We can repeat this process to obtain the parameters throughout the whole period (one week).

In case of exponentially distributed delays, the parameter is the average number of arrivals. Figure 4 shows the average number of user arrivals and departures per hour for each one hour period during an average week.

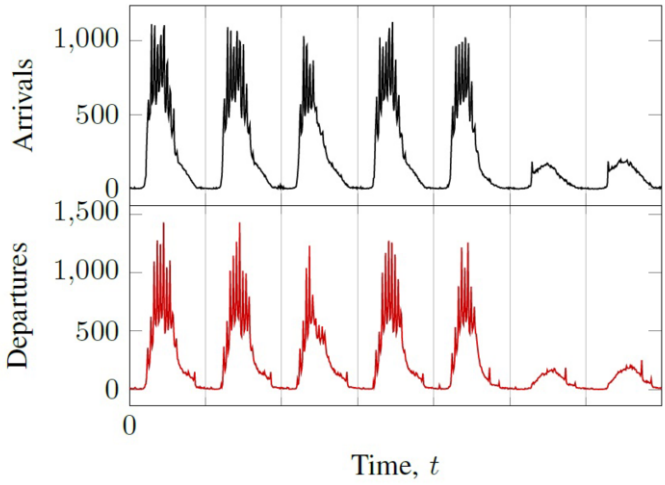

Fig. 4. The aggregate number of arrivals and departures from the system per hour during an average week.

Simple statistical tests show that the delay distributions in our data are not exponential. However, experience shows that they can be accurately approximated by multi-phase Coxian distributions. These can be included in the model in the form of new intermediate states of the PCTMC components.

4 PCTMC model

In this chapter we describe a PCTMC model of an HTCondor cluster which we fit to the data described above.

The components in the model are workstations, interactive users and HTCondor jobs - both in the system queue and assigned onto workstations. For a better fit to the data, we consider a number of different workstations/user classes based on the opening hours and teaching patterns of their respective rooms. We also consider a number of different job classes. This is important when modelling hypothetical job submissions. For example, one class can consist of normal job submissions which

are forecast from real data, whereas a second class could represent a batch of jobs a user is planning to submit.

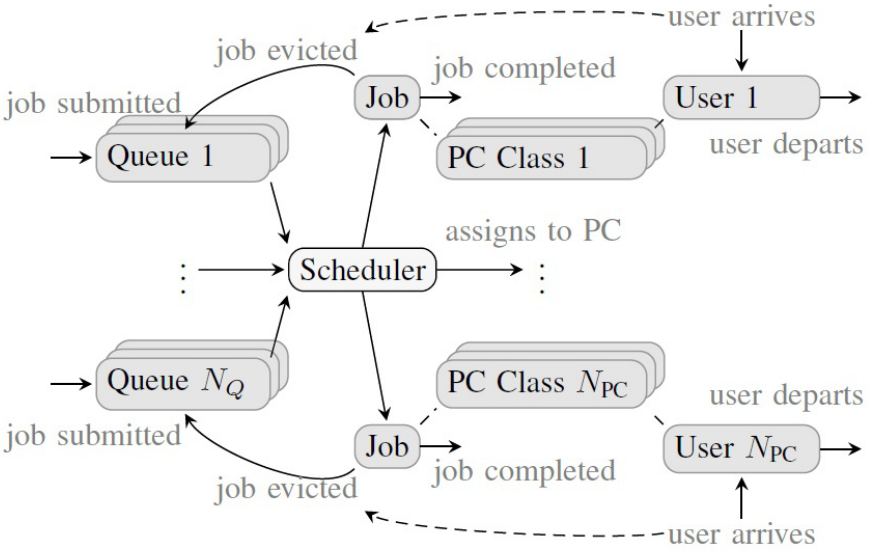

Fig. 5. Overview of the PCTMC model.

Figure 5 gives an overview of the system. Users acquire their respective workstations according to Coxian distributions fitted from the data; similarly they depart from the workstations. Jobs are submitted to the system according to the data and also depending on the modelling purposes.

Because of the time-inhomogeneous nature of the data, the parameters of the fitted Coxian distributions have to vary with time. Our aim is to fully automate this process and therefore we require a fully automated fitting algorithm. Experiments show that the expectation maximisation algorithm of the *EMpht* tool [1] gives good match to both mean and variance of the distributions.

Each machine becomes available to accept HTCondor jobs after a short delay following a user departure. Jobs are randomly assigned onto workstations. When a user acquires a machine serving a job, that job gets cancelled and moved back into the central queue. This is a simplification for illustration purposes. The model could also cope with a more realistic behaviour. For example, a standard strategy in HTCondor systems is to suspend a job rather than evict it when an interactive user acquires the machine. The job is held optimistically on the machine before being evicted. If the user leaves during this period, the job is resumed on the existing machine.

We use the model to predict the system behaviour for the subsequent short interval of time – the *forecast period*. In our examples, we set this to two hours. Using data from the previous week, we fit a time-inhomogeneous arrival and departure processes for the users and jobs. We solve the PCTMC model with the fluid analysis using the GPA tool [13]. The tool accepts a PCTMC description of the

model and is able to include the time inhomogeneous rate data. GPA then automatically generates the system of ODEs, which would otherwise be too complicated to derive manually. GPA numerically solves the ODEs and uses the solution to provide predictions of the metrics in Section 2.2 for the forecast period. We repeat this at every successive non-overlapping forecast period throughout a sample day. In real applications, this analysis could be done at any point in time to predict the immediate future for the duration of the forecast period.

Figure 6 shows the forecast for the total number of active users for regular two hour forecast periods during a sample day.

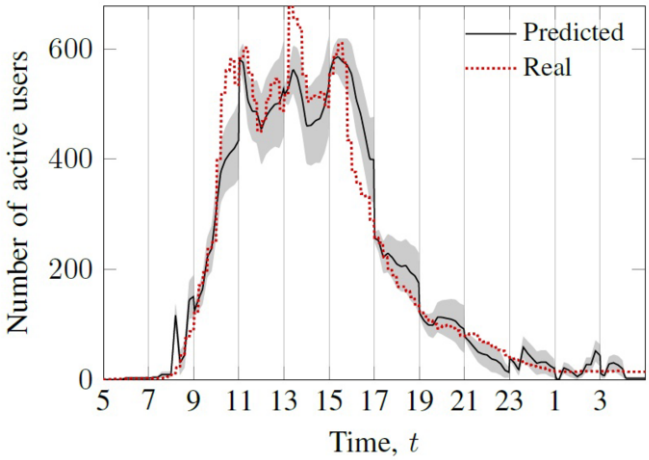

Fig. 6. Forecast model for a sample day. The grey region is the 95% interval around the predicted value.

The ODE analysis also produces higher moments which can be used to estimate the distribution of the number of active users. Figure 6 shows an interval where the population falls with probability 0.95. There is a clear qualitative agreement of the forecast, although some quantitative differences appear. However, these can be reduced by looking at a shorter time period than two hours or by using a more sophisticated approach for the forecast of arrival and departure distributions.

The forecast of the cluster occupancy would be possible by traditional time-series analysis techniques. These are known to provide a comparable accuracy of the predictions [4]. However, a major advantage of the PCTMC approach is that we can add additional hypothetical elements to the model. For example, we can submit a “probe” job at the beginning of each forecast period and evaluate the average number of evictions this job would experience, running alongside the real jobs present in the system and forecast from historical data. Figure 7 shows this for the forecasts during a sample day. As expected, the job experiences more evictions during the busier periods within the day.

Because of the possible evictions, HTCondor users normally split their computational task into a large number of jobs that are submitted in a batch. Therefore, another useful application of the PCTMC model is to predict effects of a hypothetical batch of jobs submitted at the beginning of each forecast period. For example, we can add 500 jobs to the system, with exponentially distributed time to complete

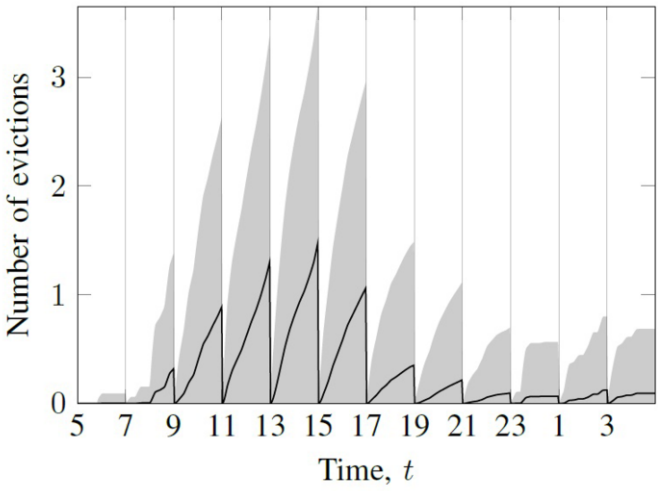

Fig. 7. Forecast for the number of evictions per job. The gray regions show the 95% intervals around the mean, estimated from the standard deviation.

with a mean of 10 minutes. The resulting PCTMC model is able to predict the throughput of the system over time. Figure 8 shows the number of finished jobs over time during each forecast period throughout a sample day.

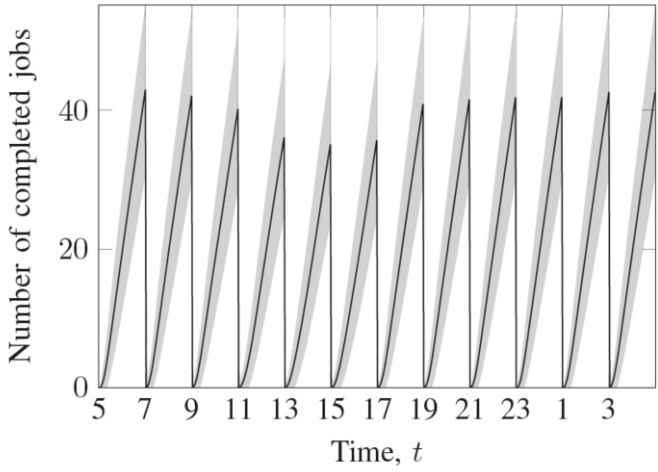

Fig. 8. Forecast for the number of finished jobs from a batch of 500 jobs with exponentially distributed processing time with mean of 10 minutes. The gray regions show the 95% intervals around the mean, estimated from the standard deviation.

Figure 9 shows the increase in energy consumption resulting from the batch of jobs. This is obtained by comparing the total energy consumption in two versions of the PCTMC model – with and without the hypothetical batch of jobs.

5 Conclusion and Future Work

In this paper we have presented preliminary work on applying Population Continuous Markov Chain (PCTMC) to modelling of cycle-stealing computing clusters. We

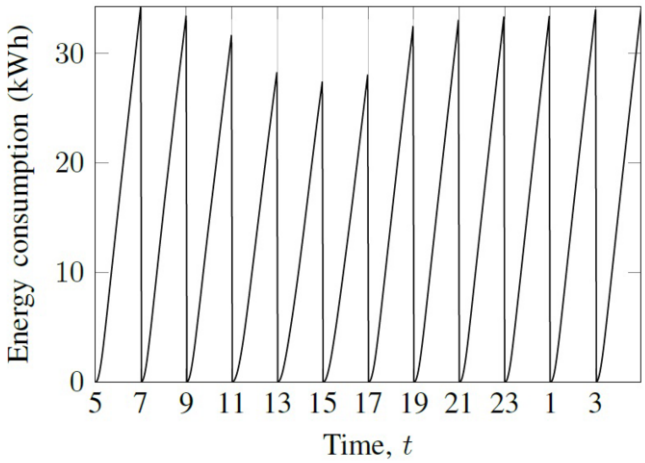

Fig. 9. Forecast of the increase in energy consumption resulting from submitting a batch of 500 jobs with an exponentially distributed processing time with mean of 10 minutes.

have shown how the PCTMC framework is well suited to the large scale nature of the system. The PCTMC models also cope well with the time-inhomogeneous user access patterns the cluster workstation experience.

We demonstrated the technique on a real-life dataset collected from a year long deployment of HTCondor at the Newcastle University. We have proposed an aggregation of HTCondor logs which can be used to fit parameters of the PCTMC model. To cope with the non-exponential nature of the system, we fitted multi-phase Coxian distributions to the arrival and departure delays. We use the PCTMC model in a simple forecasting algorithm. At each point in time when a forecast is desired, we fit parameters of the model to historical data from the same time during the previous week. This way, we obtain a time-inhomogeneous PCTMC model for the forecast period. We extend the model with hypothetical jobs that allow us to evaluate the possible effect of a user submitting a large batch of jobs. We have developed a working implementation of the technique using the existing GPA tool [12].

The results in this paper are preliminary and demonstrate the feasibility of the presented approach. We believe that a more detailed PCTMC model can be constructed to capture various crucial aspects of the real system. These include for example more complex scheduling policies, possibly in a feedback loop from the performance and energy consumption metrics, regular reset events (reboots in the system), possibility to put workstation into sleep mode, specification of dedicated HTCondor servers etc. We are certainly confident that the efficient fluid analysis will cope with the size of such models. We are planning to validate such model on the real system by submitting different batches of jobs and comparing the performance with the model's prediction.

Our parameter forecasting algorithm is very simple and relies on perfect periodicity of the usage patterns. We anticipate that a more sophisticated forecasting algorithm will be needed to accurately predict the model parameters from historical data. We are currently investigating a hybrid approach where traditional time-series

analysis techniques are used to forecast some of the parameters.

Our ultimate goal is to develop a tool that can be deployed together with the HTCondor system. The tool would regularly aggregate trace logs and automatically create and fit PCTMC models. The predictions from the models would be used to suggest to the users the best possible way to submit their jobs. For example, users might have a choice between different granularity levels of splitting their high-throughput work. The tool could suggest an optimal split based on the predicted resource availability such that the wastage caused by interruptions from interactive users is minimised. The tool would also give users an idea of the expected job duration and thus help them to better plan their computing workflow. The models can also be used by the maintainers of the system to explore different parameters and scheduling policies in order to minimise the running costs while maintaining a good throughput for the HTCondor computations.

References

[1] S. Asmussen, O. Nerman, and M. Olsson. Fitting Phase-type Distributions via the EM Algorithm. *Scandinavian Journal of Statistics*, 23(4):419 – 441, 1996.

[2] T. Estrada, M. Taufer, and D.P. Anderson. Performance Prediction and Analysis of BOINC Projects: An Empirical Study with EmBOINC. *Journal of Grid Computing*, 7(4):537–554, August 2009.

[3] N. Gast and G. Bruno. A mean field model of work stealing in large-scale systems. In *Proceedings of the ACM SIGMETRICS international conference on Measurement and modeling of computer systems - SIGMETRICS '10*, page 13, New York, New York, USA, 2010. ACM Press.

[4] M.C. Guenther and J.T. Bradley. Journey data based arrival forecasting for bicycle hire schemes. In *20th International Conference on Analytical Stochastic Modelling Techniques Applications (ASTMA'13)*, Lecture Notes in Computer Science, 2013.

[5] R.A. Hayden and J.T. Bradley. A fluid analysis framework for a Markovian process algebra. *Theoretical Computer Science*, 411(22-24):2260–2297, May 2010.

[6] R.A. Hayden, A. Stefanek, and J.T. Bradley. Fluid computation of passage-time distributions in large Markov models. *Theoretical Computer Science*, 413(1):106–141, January 2012.

[7] J. Hillston. Fluid flow approximation of PEPA models. In *Proceedings of the Second Interational Conference on the Quantitative Evaluation of SysTems (QEST)*, pages 33–42. IEEE, September 2005.

[8] W. Kelly and J. Sumitomo. What are Cycle-Stealing Systems Good For? A Detailed Performance Model Case Study. In *11th International Conference on Parallel and Distributed Systems (ICPADS'05)*, volume 2, pages 500–504. IEEE, 2005.

[9] M.J. Litzkow, M. Livny, and M.W. Mutka. Condor-a hunter of idle workstations. *8th International Conference on Distributed Computing Systems*, pages 104 – 111, 1988.

[10] A.S. McGough, M. Forshaw, C. Gerrard, and S. Wheater. Reducing the Number of Miscreant Tasks Executions in a Multi-use Cluster. In *2012 Second International Conference on Cloud and Green Computing*, pages 296–303. IEEE, November 2012.

[11] N. Mi, A. Riska, X. Li, E. Smirni, and E. Riedel. Restrained utilization of idleness for transparent scheduling of background tasks. In *Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems - SIGMETRICS '09*, page 205, New York, New York, USA, June 2009. ACM Press.

[12] A. Stefanek, R.A. Hayden, and J.T. Bradley. Fluid Analysis of Energy Consumption using Rewards in Massively Parallel Markov Models. In *ICPE'11 - Second Joint WOSP/SIPEW International Conference on Performance Engineering, Karlsruhe, Germany, March 14-16, 2011*, page 121. ACM Press, 2011.

[13] A. Stefanek, R.A. Hayden, and J.T. Bradley. GPA - A Tool for Fluid Scalability Analysis of Massively Parallel Systems. In *2011 Eighth International Conference on Quantitative Evaluation of SysTems*, pages 147–148. IEEE, September 2011.

[14] A. Stefanek, R.A. Hayden, and J.T. Bradley. Mean-field Analysis of Large Scale Markov Fluid Models with Fluid Dependent and Time-inhomogeneous Rates. *Annals of Operations Research*, 219:1–27, 2014.

[15] N. Thomas. Using ODEs from PEPA models to derive asymptotic solutions for a class of closed queueing networks. In *Proceedings of the 8th Workshop on Process Algebra and Stochastically Timed Activities*. University of Edinburgh, 2009.

[16] N. Thomas and Zhao Y. Fluid flow analysis of a model of a secure key distribution centre. In *Proceedings of the 24th Annual UK Performance Engineering Workshop*. Imperial College London, 2008.

[17] Y. Zhao and N. Thomas. Efficient solutions of a PEPA model of a key distribution centre. *Performance Evaluation*, 67(8):740–756, 2010.