Evaluating the Stability of Supercomputer Workload Model

Alexander Rumyantsev

***Abstract*—A multi-server model with simultaneous service and concurrent server release is considered. A stability condition of the model (under exponential assumptions) is presented. A method for fast evaluation of the stability criterion is obtained. The condition allows to verify the stability of a supercomputer at design time, provided that the governing sequence of tasks is known beforehand. The `hpcwld` package that provides R [1] functions for the workload (unfinished work) evaluation as well as stability condition evaluation is discussed. The package can be obtained via Comprehensive R Archive Network [2]. The results may be of practical interest to the designers and owners of supercomputers.**

***Index Terms*—Supercomputer workload, stochastic modeling, stability, dimension reduction, CRAN, R.**

I. Introduction

HIGH performance clusters (HPC) are powerful tools for the speedup of computations in the tasks requiring heavy calculations (e. g. the Grand Challenges [3]). However, development and improvement of an HPC is rather expensive. Building a very powerful system, which highly overlaps the users need for computational power, may lead to waste of the resources. At the same time, a system which lacks computational power, may lead to loss of stability, which in practice means the growth of delays in the queue and unsatisfied users. When performance is an issue and the delays have a trend to grow, a long term consequence will be increasing migration of users to another, more satisfactory HPC site and (in case of commercial use) income decrease. The compromise may be successfully reached by performing a stability analysis of HPC at the design time.

Models of HPC workload, as well as simulation of HPC processing with workload traces, are widely used for the design and deployment of scheduling software [4]–[11]. However, the majority of aforementioned papers consider models that allow mostly numerical evaluation, and do not suit to perform the stability analysis.

Multi-server systems have been intensively studied in literature, for references see e. g. the reviews [12]–[19]. In such models customers are being served by a number of (potentially different) servers. Apart from classical multi-server models, in which each customer is served by only one of the servers, a number of works are dedicated to the analysis of a particular subclass of multi-server systems with the so-called simultaneous service (see [20]–[27]). The distinctive feature of these models is that a (random) number of servers are simultaneously occupied by one customer and may become free either simultaneously (the so-called *concurrent service* systems [28]), or independently (the *independent service* systems [28]). The former of these two system models captures the essential feature of an HPC, where each customer (or the so-called *rigid job* [27]) may occupy at once a number of CPUs, and release them simultaneously upon the completion of its service. Application of the multi-server models with simultaneous service to the analysis of real HPC systems is straightforward. We stress that such models may not in general be reduced to classical multiserver models.

A multi-server concurrent service model for the workload process (unfinished work on each server) was presented in [29]. A modification of the so-called Kiefer–Wolfowitz recursion [30], [31] was performend in such a way to incorporate the simultaneous server occupation/release property. The main difficulty in analysis of supercomputer workload model is that the discipline is not work-conserving, which means that there might be available processors at the same time with non-empty queue [20], [21]. This greatly increases the complexity of stability analysis of the model, as the workload turns out to be non work-conserving. However, the model in [29] due to its relative simplicity is analytically tractable, and the stability condition for the model has been represented in an explicit form by means of matrix-analytic method [32].

The considered model was implemented as an extension package `hpcwld` for R statistical software [1]. The choice of R as a target platform was motivated by the potential power of statictical methods available as extensions that allow to perform a rich statistical study over the results of simulation. The presented package contains function to evaluate the stability condition for the concurrent service workload model by means of characteristics of the input flow of customers.

This work is organized as follows. In section II, the stability of a concurrent service model based on the modified Kiefer–Wolfowitz recursion is discussed. Next, computational issues are summarized. The functionality of the `hpcwld` package for the R system [1] is briefly described. To conclude, we add a discussion about the points for future work.

II. Stability of an HPC workload model

Consider an open system with s identical servers working in parallel. Customers arrive at random epochs $t_i,\ i \geqslant 1$ (with exponentially distributed interarrival times $T_i := t_{i+1} - t_i > 0$)

Manuscript received December 05, 2014; accepted December 10, 2014. Date of online publication: December 28, 2014.

The work is supported by grant of the Russian Foundation for Basic Research (projects 14-07-31007, 13-07-98801).

A. Rumyantsev is with the Institute of Applied Mathematical Research of Karelian Research Centre, Russian Academy of Sciences, 11, Pushkinskaya str., 185910 Petrozavodsk, Karelia, Russia; e-mail:ar0@krc.karelia.ru.

into a single queue (First Come First Served queueing discipline). A customer i needs to occupy a (random) number N_i of servers for (indentical, exponentially distributed) time S_i simultaneously. In case of insufficient resources, the customer i has to spend time $D_i \geqslant 0$ in the queue. Define a customer i to be of class-k if $N_i = k$. The required number of servers for each customer has a discrete distribution

$$p_k := \mathsf{P}\{N = k\}, \quad k = 1, \dots, s.$$

(The indices are omitted for generic elements of a stochastic sequence.)

The workload vector of the system under consideration follows the modification of a Kiefer–Wolfowitz recursion:

$$W_{i+1} = R\big(\overbrace{W_{i,N_i} + S_i - T_i, \dots, W_{i,N_i} + S_i - T_i}^{N_i \text{ components}}, W_{i,N_i+1} - T_i, \dots, W_{i,s} - T_i\big)^+, \quad (1)$$

where W_i is the vector of unfinished work on each of s processors placed in the ascending order (as seen by customer i at the arrival epoch t_i), $R(\cdot)$ places components of a vector in an ascending order and $(\cdot)^+ = \max(0, \cdot)$ componentwise. The delay of i-th customer is then defined as follows:

$$D_i := W_{i,N_i},\ i \geqslant 1. \quad (2)$$

Note that the driving sequence $\{T_i, S_i, N_i\}$ for $i \geqslant 0$ may be extracted from the logfile of the queue management software of an HPC (a number of available workload traces may be found in [33]). Distribution sampling may be also used in case the characteristics of the input flow are known [34].

The *necessary and sufficient stability condition* for the considered system (where stability means that the delay D_i converges weakly to a stationary delay D as $i \to \infty$) is as follows [32]

$$\rho := \lambda C \mathsf{E} S < 1, \quad (3)$$

where $\lambda := 1/\mathsf{E}T$ is the arrival intensity,

$$C := \sum_{m \in \mathcal{M}} \frac{\prod_{i=1}^s p_{m_i}}{\max\{i : \sum_{j=1}^i m_j \leqslant s\}}, \quad (4)$$

and $\mathcal{M} = \{1, \dots, s\}^s$. The complete proof of this stability result is to appear in a separate paper. Intuitively, the set $\mathcal{M}$ is a phase space of the system, where a "phase" is treated as the vector of classes of the first s customers in the system (in the order of arrival) present at some instance t. In this regard, the constant C may be treated as the intensity of clients processing simultaneously for a given phase space $\mathcal{M}$. This means that sC may be intuitively treated as mean number of servers occupied by each customer, given phase space $\mathcal{M}$, and the criterion (3) may be treated as $\lambda \mathsf{E}(N|M)\mathsf{E}S < s$, where $\mathsf{E}(\cdot|\mathcal{M})$ is the mean taken conditionally on M. However, a direct use of the criterion (3) is limited by the need of performing enumeration over the set $\mathcal{M}$ of capacity s^s to calculate (4). This means that evaluation of C for $s > 10$ is computationally limited.

A. Dimension reduction method

Transform the set $\mathcal{M}$ into a set $\mathcal{N}$ as follows: for each $m = (m_1, \dots, m_s) \in \mathcal{M}$ define a vector function $n^{(m)} = (n_1^{(m)}, \dots, n_s^{(m)}) \in \mathcal{N}$ by the rule

$$n_i^{(m)} = \#\{m_j = i,\ 1 \leqslant j \leqslant ||m||\}, \quad 1 \leqslant i \leqslant s, \quad (5)$$

where $||m|| := \max\{i : \sum_{j=1}^i m_j \leqslant s\}$. Note that $n^{(m)}$ is a vector representation of an integer partition of the value $\sum_{i=1}^{||m||} m_i = \sum_{j=1}^s j n_j^{(m)}$ by the set $\{i \leqslant s : n_i^{(m)} > 0\}$. It is readily seen, that the transform $n^{(m)} : \mathcal{M} \to \mathcal{N}$ is a surjection, and $\mathcal{N}$ is the set of integer partitions of the numbers $1, \dots, s$. Denote $\mathcal{N}_k$ the set of integer partitions of $k \leqslant s$. Thus, $\mathcal{N} = \bigcup_{k=1}^s \mathcal{N}_k$.

Fix $\hat{n} \in \mathcal{N}_k$. Note that $\hat{n}$ is an integer partition of k. Then define a subset $\mathcal{M}(\hat{n}) := \{m \in \mathcal{M} : n^{(m)} = \hat{n}\}$. By definition (5) for $m \in \mathcal{M}(\hat{n})$ the following holds:

$$||m|| = \sum_{i=1}^s \hat{n}_i. \quad (6)$$

Moreover,

$$\prod_{i=1}^{||m||} p_{m_i} = \prod_{i=1}^s p_i^{\hat{n}_i}, \quad m \in \mathcal{M}(\hat{n}).$$

Thus, by definition (5), for $\mathcal{M}(\hat{n})$ the values $(m_1, \dots, m_{||m||})$ are a permutation of a multiset $\{i \leqslant s : \hat{n}_i > 0\}$ with multiplicities $n_i, i \geqslant 0$. The number of such permutations equals

$$\frac{(\sum_{i=1}^s \hat{n}_i)!}{\prod_{i=1}^s \hat{n}_i!}. \quad (7)$$

It remains to note that one has $m_{||m||+1} \geqslant s - k + 1$, whereas $m_{||m||+i},\ i > 1$ may be arbitrary, where $m \in \mathcal{M}(\hat{n})$. This allows to deduce

$$\sum_{m \in \mathcal{M}(\hat{n})} \prod_{i=||m||+1}^s p_{m_i} = \sum_{i=s-k+1}^s p_i \sum_{m \in \mathcal{M}(\hat{n})} \prod_{i=||m||+2}^s p_{m_i} = \sum_{i=s-k+1}^s p_i.$$

Then, recalling (7), get

$$\sum_{m \in \mathcal{M}(\hat{n})} \frac{\prod_{i=1}^s p_{m_i}}{||m||} = \frac{(\sum_{i=1}^s \hat{n}_i)!}{\prod_{i=1}^s \hat{n}_i!} \prod_{i=1}^s p_i^{n_i} \sum_{j=s-k+1}^s p_j. \quad (8)$$

With (8), recalling (6), from (4) deduce

$$C = \sum_{k=1}^s \sum_{n \in \mathcal{N}_k} \frac{(\sum_{i=1}^s \hat{n}_i - 1)!}{\prod_{i=1}^s \hat{n}_i!} \prod_{i=1}^s p_i^{n_i} \sum_{j=s-k+1}^s p_j. \quad (9)$$

Note that the capacity of the set $\mathcal{N}$ of integer partitions is asymptotically $s^{-1} \exp(\sqrt{s})$ [35], which allows one to perform computation of the value C using (9) for $s = O(10^3)$. Intuitively, this transformation of the set $\mathcal{M}$ to $\mathcal{N}$ relies on the observation that for given $\hat{n}$ the phases that transform to $\hat{n}$ are in some sense similar. The first $||m||$ customers (that are actually served) are the same up to sample, the customer $||m|| + 1$ can be only of a limited subset of classes, and the options for customers $||m|| + 2, \dots, s$ together are exhaustive

events. This allows one to sufficiently reduce the complexity of the calculations required to evaluate the stability constant C.

Moreover, as the summation is done over all the distinct sets of partitions of integers $1, \ldots, s$, the algorithms of parallel partition generation may be applied (see [36], [37]). This allows to further extend the upper bound on s. Moreover, consider also the following approximation of C using Monte-Carlo simulation:

$$C \approx \frac{1}{|\mathcal{M}'|} \sum_{m \in \mathcal{M}'} \frac{1}{||m||}, \tag{10}$$

where $\mathcal{M}' \subseteq \mathcal{M}$ is a random subset and $|\mathcal{M}'|$ is the capacity of the set $\mathcal{M}'$.

B. The hpcwld package

The `hpcwld` package for HPC model workload evaluation can be obtained via CRAN [2] for R version 2.15.0 or higher. The package includes documentation with usage examples as well as sample dataset. The latest development version can be obtained via R-Forge [38]. All the functions are implemented in R. The package allows evaluation of the workload recursion (1) for given driving sequences $\{T_i, N_i, S_i\}$ (the `Wld` function). The aforementioned sequences may be imported from a so-called standart workload format file, a number of such files may be obtained via the workload archive [33]. The corresponding import and export functions are included in the package (functions `FromSWF` and `ToSWF`). The function for evaluation of stability constant C using equation (9) as well as Monte-Carlo version (10) are implemented in the package (function `StabC`).

III. Discussion

In this paper we presented a method for evaluation of the stability criterion for an HPC workload model, that belongs to class of simultaneous concurrent service multi-server models. The condition allows to verify the stability of an HPC at design time, provided that the governing sequence of tasks is known beforehand.

One of the drawbacks of the model (1) is the assumption of the FCFS service discipline, which is not so oftenly used in practice [34]. Nevertheless, the model is still useful as an estimation of a real system. Moreover, one may easily see that in case of no difference in priorities and deadlines of the Backfill discipline, the customers are served according to FCFS discipline. Note also that the stability criterion has been strictly proved only in exponential case. However, we conjecture that the criterion is true also in case of a general service time distribution.

Acknowledgment

Author thanks Dr. Evsey V. Morozov for valuable comments.

References

[1] R Core Team, "R: A Language and Environment for Statistical Computing," 2014. [Online]. Available: http://www.r-project.org/

[2] CRAN team, "The Comprehensive R Archive Network," 2014. [Online]. Available: http://cran.r-project.org/

[3] Wikimedia Foundation, "Grand Challenges - Wikipedia, the free encyclopedia," 2014. [Online]. Available: http://en.wikipedia.org/wiki/Grand_Challenges

[4] D. G. Feitelson and L. Rudolph, "Metrics and benchmarking for parallel job scheduling," in *Job Scheduling Strategies for Parallel Processing*. Springer, 1998, pp. 1–24.

[5] J. Jann, P. Pattnaik, H. Franke, F. Wang, J. Skovira, and J. Riordan, "Modeling of workload in MPPs," in *Job Scheduling Strategies for Parallel Processing, IPPS'97 Workshop, Geneva, Switzerland, April 5, 1997, Proceedings*, ser. Lecture Notes in Computer Science, vol. 1291. Springer, 1997, pp. 95–116.

[6] H. Li, D. Groep, and L. Wolters, "Workload characteristics of a multi-cluster supercomputer," in *Job Scheduling Strategies for Parallel Processing*. Springer, 2005, pp. 176–193.

[7] V. Lo, J. Mache, and K. Windisch, "A comparative study of real workload traces and synthetic workload models for parallel job scheduling," in *Job Scheduling Strategies for Parallel Processing*. Springer, 1998, pp. 25–46.

[8] E. Medernach, "Workload analysis of a cluster in a grid environment," in *Job scheduling strategies for parallel processing*. Springer, 2005, pp. 36–61.

[9] B. Song, C. Ernemann, and R. Yahyapour, "Parallel computer workload modeling with markov chains," in *Job Scheduling Strategies for Parallel Processing*. Springer, 2005, pp. 47–62.

[10] A. Krampe, J. Lepping, and W. Sieben, "A hybrid markov chain model for workload on parallel computers," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, ser. HPDC '10. New York, NY, USA: ACM, 2010, pp. 589–596.

[11] E. Shmueli and D. G. Feitelson, "On simulation and design of parallel-systems schedulers: are we doing the right thing?" *Parallel and Distributed Systems, IEEE Transactions on*, vol. 20, no. 7, pp. 983–996, 2009.

[12] J. Sztrik, "Finite-source queueing systems and their applications: A bibliography," 2002. [Online]. Available: http://irh.inf.unideb.hu/user/jsztrik/research/fsqreview.pdf

[13] O. Boxma, G. Koole, and Z. Liu, "Queueing-theoretic solution methods for models of parallel and distributed systems," in *Performance Evaluation of Parallel and Distributed Systems Solution Methods*, 1994, pp. 1–24.

[14] M. Satyanarayanan, "Multiprocessing: An annotated bibliography," *Computer*, vol. 13, no. 5, pp. 101–116, 1980.

[15] H. Takagi and L. Boguslavsky, "A supplementary bibliography of books on queueing analysis and performance evaluation," *Queueing Systems*, vol. 8, no. 1, pp. 313–322, 1991.

[16] D. Worthington, "Reflections on queue modelling from the last 50 years," *J Oper Res Soc*, vol. 60, no. S1, pp. S83–S92, 2009.

[17] N. U. Prabhu, "A bibliography of books and survey papers on queueing systems: Theory and applications," *Queueing Syst. Theory Appl.*, vol. 2, no. 4, pp. 393–398, 1988.

[18] D. Terekhov, D. G. Down, and J. C. Beck, "Queueing-theoretic approaches for dynamic scheduling: A survey." [Online]. Available: http://www.cas.mcmaster.ca/ downd/dariasurvey.pdf

[19] O. Boxma, G. Koole, and Z. Liu, "Queueing-theoretic solution methods for models of parallel and distributed systems," in *Performance Evaluation of Parallel and Distributed Systems Solution Methods*, 1994, pp. 1–24.

[20] L. Green, "A queueing system in which customers require a random number of servers," *Operations Research*, vol. 28, no. 6, pp. 1335–1346, 1980.

[21] ——, "Comparing operating characteristics of queues in which customers require a random number of servers," *Management Science*, vol. 27, no. 1, pp. 65–74, 1980.

[22] P. Brill and L. Green, "Queues in which customers receive simultaneous service from a random number of servers: A system point approach," *Management Science*, vol. 30, no. 1, pp. 51–68, 1984.

[23] W. Whitt, "Blocking when service is required from several facilities simultaneously," *AT&T Technical Journal*, vol. 64, no. 8, pp. 1807–1856, 1985.

[24] S. Kim, "M/m/s queueing system where customers demand multiple server use," Ph.D. dissertation, Southern Methodist University, 1979.

[25] G. Y. Fletcher, H. Perros, and W. Stewart, "A queueing system where customers require a random number of servers simultaneously," *European Journal of Operational Research*, vol. 23, pp. 331–342, 1986.
[26] D. Filippopoulos and H. Karatza, "An m/m/2 parallel system model with pure space sharing among rigid jobs," *Mathematical and Computer Modelling*, vol. 45, no. 5–6, pp. 491–530, 2007.
[27] S. Chakravarthy and H. Karatza, "Two-server parallel system with pure space sharing and markovian arrivals," *Computers & Operations Research*, vol. 40, no. 1, pp. 510 – 519, 2013.
[28] N. M. Van Dijk, "Blocking of finite source inputs which require simultaneous servers with general think and holding times," *Oper. Res. Lett.*, vol. 8, no. 1, pp. 45–52, 1989.
[29] E. V. Morozov and A. S. Rumyantsev, "Stability analysis of a multiprocessor model describing a high performance cluster," in *XXIX International Seminar on Stability Problems for Stochastic Models and V International Workshop "Applied Problems in Theory of Probabilities and Mathematical Statistics related to modeling of information systems", Book of Abstracts*. Moscow: Institute of Informatics Problems, RAS, 2011, pp. 82–83.
[30] J. Kiefer and K. Wolfowitz, "On the theory of queues with many servers," *Transactions of the American Mathematical Society*, vol. 78, no. 1, pp. 1–18, 1955.
[31] ——, "On the characteristics of the general queueing process, with applications to random walk," *The Annals of Mathematical Statistics*, vol. 27, no. 1, pp. 147–161, 1956.
[32] A. S. Rumyantsev, "Stabilization of a high performance cluster model," in *Proceedings of 2014 International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), Saint-Petersburg, 06-08 Oct. 2014*. Saint-Petersburg: IEEE, 2014, pp. 1–4.
[33] D. G. Feitelson, "Parallel Workloads Archive: Logs," 2014. [Online]. Available: http://www.cs.huji.ac.il/labs/parallel/workload/logs.html
[34] ——, "Workload modeling for computer systems performance evaluation (web draft)," http://www.cs.huji.ac.il/ feit/wlmod/wlmod.pdf, 2014.
[35] P. Erdős, "On an elementary proof of some asymptotic formulas in the theory of partitions," *Annals of Mathematics. Second Series*, vol. 43, pp. 437–450, 1942.
[36] L. A. Sanchis and M. B. Squire, "Parallel algorithms for counting and randomly generating integer partitions," *Journal of Parallel and Distributed Computing*, vol. 34, no. 1, pp. 29–35, 1996, WOS:A1996UJ38500003.
[37] K. Yamanaka, S.-i. Kawano, Y. Kikuchi, and S.-i. Nakano, "Constant time generation of integer partitions," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E90-A, no. 5, pp. 888–895, 2007.
[38] R Core Team, "R-Forge," 2014. [Online]. Available: http://r-forge.r-project.org/