Optimizing performance in heavy-tailed system: a case study

Alexander S. Rumyantsev[†], Luybov V. Potakhina[‡]

[†]Institute of Applied Mathematical Research, Karelian Research Centre, RAS

[‡]Petrozavodsk State University

E-mail: ar0@krc.karelia.ru, lpotahina@gmail.com

Abstract

This article is devoted to certain aspects of optimizing the performance of a single-server queue with heavy-tailed service time distribution, three case-studies are presented: choosing an optimal queueing policy, switching to multi-server system, choosing a task assignment policy. We discuss main statistical properties of heavy-tailed distributions and some practice examples, which illustrate evidence of heavy tails in systems. The main goal of the work is to find and describe ways, which can minimize the effect of heavy tails on system performance.

1 Introduction

The most recent advances in computer systems design show the trend of switching to multiple cores and many cores from the former single-core machine types [1, 2]. While taking into account the thermal effects and frequency limits, one may also consider the differences in the workload processes of this two types of machines. One of the goals of this work is to illustrate the advantages of using multi-core architecture instead of a single-core one.

In modern computer systems analysis one of the challenges one could face are the heavy-tailed distributions of stochastic processes involved. Recent research has highlighted that this kind of distributions can describe a behavior of some real network process better than others (e.g. exponential distribution). One could find empirical evidence and some discussion about the subject in works [3, 4, 5] and the most recent work [6]. Among the most used distributions in practice is Pareto distribution with tail

$$P(X > x) = x^{-\alpha}, \quad x > 1,\ \alpha > 1.$$

Consider some key properties of heavy tailed distributions:

- the so-called Pareto law, or *mass-count disparity*: provided heavy-tailed service time, a small fraction of tasks requires a large part of capacity and vice-versa, vast majority of small tasks requires only a little bandwidth (CPU time, traffic etc.).
- heavy-tailed distributed random values may have infinite variance (and moreover, if $\alpha < 1$, even infinite mean).
- under heavy-tailed service time, a jobs waiting time process has *burstiness*: the process has some bursts, which are much bigger than "normal" values of the process. Besides, an infinite second moment of service time leads to an infinite mean waiting time in a classical single-server system.

This effects have negative influence on system characteristics: performance, reliability, quality of service. In this paper we discuss some practical recommendations how to minimize this effect.

This paper is organized as follows. Section 2 describes the influence of a service discipline on a single-server system delays, section 3 highlights the differences in workloads of a single-server and multi-server systems. Section 4 evaluates the effect of a task assignment policy in multi-server queue, and the conclusion goes in section 5.

2 Choosing a service discipline

For a non-negative random variable X we introduce the distribution function $F(x) = P(X \leqslant x),\ x \geqslant 0$, the tail distribution $\overline{F}(x) = P(X > x)$ and the equilibrium distribution (stationary residual lifetime distribution)

$$F_r(x) = \frac{1}{EX} \int_0^x \overline{F}(y)dy, \quad x \geqslant 0.$$

Consider a M/G/1 system with heavy-tailed (Pareto) distribution of a typical task service time S:

$$\overline{B}(x) := P(S > x) = x^{-\alpha}, \quad x > 1,\ \alpha > 1.$$

One should note a well-known result connecting the moment properties of random variable X and it's equilibrium version X_r:

$$EX^{\alpha} < \infty \text{ iff } EX_r^{\alpha-1} < \infty.$$

Basically this means that a random variable X_r having residual lifetime distribution has moment properties one moment worse than X. Due to the

basic properties of Pareto random variables (among other *regularly varying* r.v.), one may conclude that asymptotically

$$\frac{P(X + X_r > x)}{P(X_r > x)} \to 1, \quad x \to \infty. \tag{2.1}$$

Here we highlight recent theoretic results of the tail asymptotics of the typical waiting time W and (which is needed in case of a preemptive discipline) sojourn time V distributions for six key service disciplines. An interested reader can find the details of the analysis in the work [7].

1. In First Come First Served (FCFS) queue the tasks are served in the order of their arrival. If an arrival finds the server busy, it has to wait at least the residual service time S_r for the task being served to complete it's request. Then, due to the property (2.1) the tail of residual service time dominates. One may prove that waiting time is asymptotically equivalent to residual service time (the details are provided in [7]).

$$P\{W > x\} \sim \frac{\rho}{1-\rho}\overline{B}_r(x), \quad x \to \infty$$

2. If there are n tasks in queue with Processor Sharing (PS) discipline, they are simultaneously served with equal part of server capacity. So, the influence of long tasks on the short tasks sojourn time is limited. Then the tail of sojourn time distribution is described asymptotically as

$$P\{V > x\} \sim \overline{B}\big((1-\rho)x\big), \quad x \to \infty$$

3. For a Last Come First Served Preemptive-Resume (LCFS-PR) discipline, an arriving task is immediately taken into service. However, this service is interrupted when another task arrives, and it is only resumed when all tasks who have arrived after it have left the system. The tail asymptotics of sojourn time in this case is as follows:

$$P\{V > x\} \sim \frac{1}{1-\rho}\overline{B}\big((1-\rho)x\big), \quad x \to \infty$$

4. On the contrary, for the Last Come First Served Non-Preemptive (LCFS-NP) queue, if an arriving task finds a busy server, it doesn't interrupt the service. So, the tail of it's waiting time is determined by the tail of a residual service time:

$$P\{W > x\} \sim \rho\overline{B}_r\big((1-\rho)x\big), \quad x \to \infty$$

5. The Foreground-Background Processor Sharing (FBPS) discipline assigns an equal part of the service capacity to the customers, which have received the least amount of service. The tail of sojourn time is the same as for the PS discipline:

$$P\{V > x\} \sim \overline{B}\big((1-\rho)x\big), \quad x \to \infty$$

6. For the Shortest Remaining Processing Time First (SRPTF) discipline the total service capacity is assigned to the task with shortest remaining processing time. This discipline is preemptive and the tail of sojourn time is the same as for the PS discipline:

$$P\{V > x\} \sim \overline{B}\big((1-\rho)x\big), \quad x \to \infty$$

Summarizing the cases above, one may conclude that for a single-server system the PS, FBPS and SRPTF disciplines have asymptotically the best moment properties for the waiting/sojourn time, while the most often used FIFO discipline is not optimal. Hence, one may consider optimizing the quality of service in such a system by changing the queueing discipline.

3 Choosing a server architecture

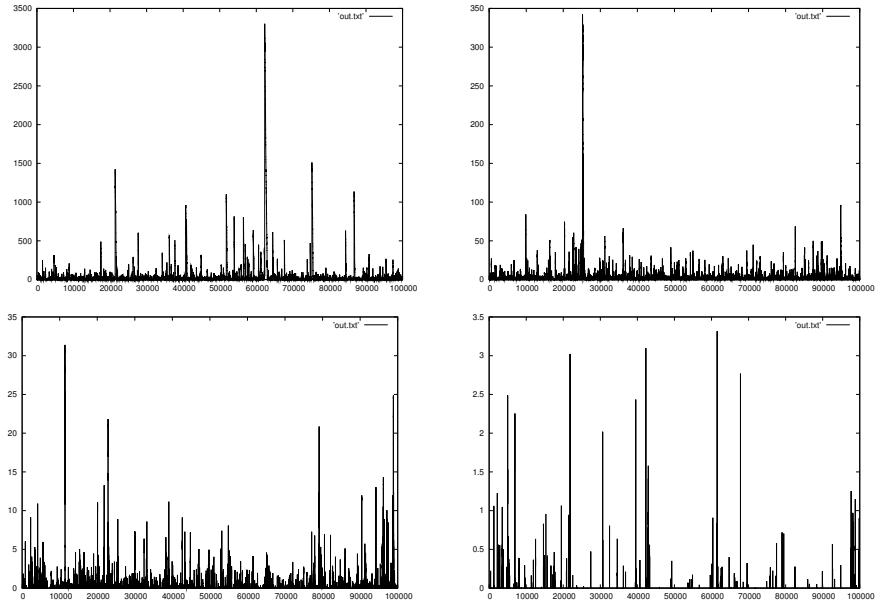
One of the trends in modern computer design is to use the low-cost power-efficient low-frequency multiple cores on a single processor instead of high-frequency systems with single core on chip. As an example one may consider the BlueGene/P systems with a PowerPC 450 cores at 850 Mhz. One of the questions arising in this regard is: which type of architecture is asymptotically best? More exactly, one may compare the moment properties of the waiting times in these two types of systems.

Consider an M/G/1 system with Pareto service times and a fast single server. Then increase the number of servers in a system with a lower speed so as to keep it's performance. (E.g. one server with a 4 GHz core, two servers with 2 GHz cores, four with 1 GHz ones, etc. One could find more details in the work [8].) For the M/G/s system one may find the following result [9]: denote $\rho = ES/ET$ the load of the system (where S and T are typical service and interarrival times respectively). Let $\rho < 1$, then one has the following moment conditions:

$$ES^{\beta} < \infty \text{ provides } EW^{s(\beta-1)} < \infty.$$

For a single server case ($s = 1$) this gives classical result. In general, these results mean that s *slow* servers provide better moment conditions for the mean waiting time than one *fast* server.

Table 1: Simulation results for waiting times in 1-, 2-, 4- and 8-core system (top-left to bottom-right).



In the numerical results below we take $\alpha = 1.5$, the total quantity of tasks is 100 000, service discipline is FIFO, and traffic intensity is $\rho = 0.3 < 1$.

At top-left picture in table 1 one can see waiting times in a simulated M/G/1 system. The maximum burst is near 3 500 time units. It is much bigger than "normal" values of the process. At the top-right the figure depicts waiting times in a system which has 2 cores. These cores are twice slower than core from 1st slide. (The identical tasks on a twice slower core in this system give service times $\hat{S}$ doubled compared to the service times S of the original system, $\hat{S} = 2S$). And one can note, that the maximum value is near 350 time units, that is an order smaller than burst at 1st slide. Bottom-left is connected with a system with 4 cores (each one with a quarter speed of the single-core server), Bottom-right is connected with system with 8 cores.

We may conclude that the increasing of core quantity can reduce mean waiting time in a system and decrease the system workload. Possibly, this is one of the results that motivates the computer system makers to use multi-core processors design.

4 Choosing a task assignment policy

Consider a system with n servers and a single queue for tasks. The queue manager (or dispatcher) should use special policy in order to assign tasks from this queue to servers, each having it's own queue. One may argue which task assignment policy is best applicable to our heavy-tailed system.

A common practical situation is when the task service times are upper and lower bounded. This leads to definition of a truncated Pareto distribution (widely used in modeling), with the density

$$f(x) = \frac{\alpha k^\alpha}{1-(k/p)^\alpha} x^{-\alpha-1}, \ where \ k \leqslant x \leqslant p$$

This distribution has all the moments finite. Nevertheless, as $p \to \infty$, this distribution approaches a standard Pareto one, which may have unbounded moments.

We assume that the dispatcher knows the size S of task. The tasks assigned to each server are served in FCFS (non-preemptive) discipline. Consider 4 key policies (the examples are based on the work [10]):

1. Random: an arriving task is sent to selected server with equal probability $1/n$.

2. Round-Robin: tasks are assigned to servers in cyclical order: i-th task being assigned to server $i \bmod n$.

3. Dynamic: Incoming task is assigned to the server with the smallest amount of remaining work time, which is the sum of the sizes of the tasks in the server's queue plus work remaining on that task currently being served.

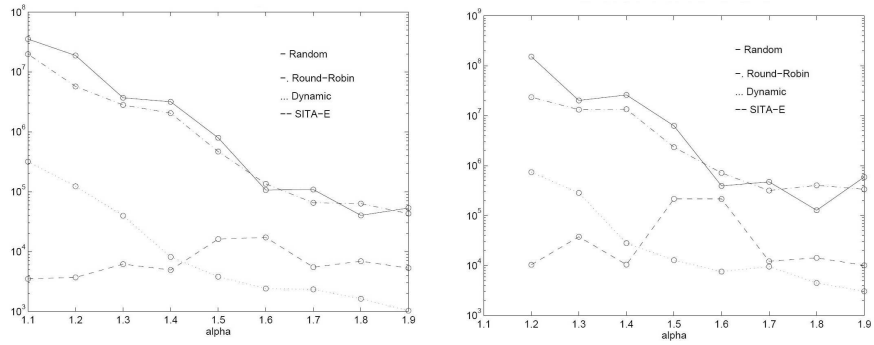
4. Size-based: so-called SITA-E (Size Interval Task Assignment with Equal Load) algorithm. The idea is to define the size range associated with each server such that the total work of each server is the same. Balancing the load this way minimizes the mean waiting time.

   Let the distribution function of task sizes be $B(x)$. Define "cutoff points" $x_i, i = 0..n, x_0 = k, x_n = p$ by the following rule:

   $$\int_{x_0}^{x_1} x dB(x) = ... = \int_{x_{n-1}}^{x_n} x dB(x) = \frac{ES}{n}$$

   and assign to the i-th core all tasks with size in the range $S \in [x_{i-1}, x_i]$ (ties defined arbitrary).

Table 2: Mean waiting time (left) and standard deviation (right) for different task assignment policies



This policies were compared via simulation in the work [10]. We consider values α from 1.1 (high variability) to 1.9 (lower variability). The figures in table 2 show mean waiting time and standard deviation for the described policies as function of parameter α.

As one can see, the mean waiting time in the system with Random and Round-Robin policies are quite similar and larger than the other two. For a large α Dynamic policy shows the best results, but if the variability of service time increases, Dynamic policy shows worse performance. In contrast, the SITA-E behavior remains quite stable, even if α tends to 1, or in other words, if the variability of service times increases.

5 Conclusion

We can conclude that a negative influence of heavy tails on system performance may be reduced in different ways. We have reviewed three practical examples on how one can decrease workload in system with heavy tailed in service times.

6 Acknowledgments

We thank prof. E. V. Morozov for useful comments.

This work is supported by Russian Foundation for Basic research, project No 10-07-00017 and done in the framework of the Strategy development Program for 2012-2016 "PetrSU complex in the research-educational space of European North: strategy of the innovation development.?

Bibliography

[1] J. Parkhurst, J. A. Darringer, B. Grundmann, *From single core to multi-core: preparing for a new exponential.* Proceedings of ICCAD'2006, San Jose, CA, 2006. pp. 67–72.

[2] Md. Tanvir Al Amin, *Multi-core: Adding a New Dimension to Computing.* Arxiv preprint arXiv10113382, 2010.

[3] E. Morozov, M. Pagano, A. Rumyantsev *Heavy-tailed distributions with applications to broadband communication systems.* Proceedings of AMICT'2007, 2008. Vol. 9, pp. 157–174.

[4] G. Samorodnitsky *Long Range Dependence* Foundations and Trends in Stochastic Systems. Vol. 1, No. 3, 2007. pp. 163–257

[5] A. Zwart, *Queueing Systems with Heavy Tails.* Eindhoven : Eindhoven University of Technology, 2001. 227 p.

[6] D. Feitelson, *Workload Modeling for Computer Systems Performance Evaluation.* Draft version. 497 p. URL: http://www.cs.huji.ac.il/ feit/wlmod/wlmod.eps

[7] S. C. Borst, O. J. Boxma, R. Nunez-Queija, *Heavy Tails: The Effect of the Service Discipline.* Proceedings of Performance TOOLS, 2002. pp. 1–30.

[8] E. V. Morozov, A. S. Rumyantsev, *Multi-server models to analyze high performance cluster.* Transactions of Karelian research center, RAS. No. 5, 2011. Pp. 75–85. (in Russian)

[9] A. Scheller-Wolf, *Further delay moment results for FIFO multiserver queues.* Queueing Systems, Vol. 34, 2000. Pp. 387–400.

[10] M. Harchol-Balter, *The Effect of Heavy-Tailed Job Size Distributions on Computer System Design.* In Proc. of ASA-IMS Conf. on Applications of Heavy Tailed Distributions in Economics, 1999.