Optimization of Performance of Queuing Systems With Long–Tailed Service Times

Bartłomiej Jacek Kubica[1] and Krzysztof Malinowski[1,2]

[1] Warsaw University of Technology, Institute of Control and Computation Engineering, ul. Nowowiejska 15/19, 00–665 Warsaw, Poland, email: bkubica@elka.pw.edu.pl
[2] Research and Academic Computer Network (NASK), ul. Wąwozowa 18, 02–796 Warsaw, Poland, email: K.Malinowski@ia.pw.edu.pl

AbstractThis paper considers the problem of the performance optimization of a queuing system with long–tailed service time. Due to nonconvexity of the problem global optimization methods have to be used. Interval methods and genetic algorithms are considered. Approximation of the Laplace transform of the probability distribution function of the service time arises as a subproblem. A new method to do it is developed.

1 Introduction

Consider a queuing system, representing e.g. an Internet server.

We want to solve the following problem: find the arrival rate λ and the service rate μ of this queuing system, to optimize some performance measure for users waiting for the completion of their tasks. It can be set as the following optimization problem (e.g. [2], [9], [12]):

$$\begin{aligned}
&\max_{\lambda,\mu} \Big(Q = V(\lambda) - \lambda \cdot G(\lambda,\mu) - C(\mu)\Big) \\
&\text{s.t.} \\
&0 \le \lambda \le \Lambda \ , \\
&\lambda < \mu \ .
\end{aligned}$$

The meaning of the above notions is as follows:

- $V(\lambda)$ – is the aggregated utility of the users, assumed to be increasing, concave and strictly differentiable
- $G(\lambda,\mu)$ – is the delay cost of the user,
- $C(\mu)$ – is the capacity cost (usually a linear structure is assumed $C(\mu) = c \cdot \mu$).

To get rid of the strict inequality in the ergodicity condition $\lambda < \mu$, we replace it by the inequality:

$$\lambda - \mu + \varepsilon \le 0 \ ,$$

where ε is a small positive number.

The paper [11] considers more details and other notions mentioned in the above optimization problem.

What about the delay cost G ? In [2] a few measures are proposed: linear cost, polynomial cost, etc. However, in the case of Pareto–distributed service time (with $\alpha < 2$; see below, Section 2) most of them are useless: they assume infinite values, regardless the values of parameters λ and μ (proof given in [12], Subsection 4.1.4). The only useful measure of the delay cost is the exponential one:

$$G = \frac{v}{k} \cdot \left(1 - \widetilde{w}(k)\right) ,$$

where $v > 0$ and $k > 0$ are some parameters, estimation of which is beyond our interest and $\widetilde{w}(k)$ is the Laplace transform of the PDF (probability density function) of sojourn time.

So, what we need to measure performance is the $\mathcal{L}$–transform of the sojourn time, which (for an $M/GI/1$ queuing system) is defined by the so–called Pollaczek–Khinchin formula (see e.g. [1]):

$$\widetilde{w}(s) = \frac{(1-\rho)\cdot \widetilde{b}(s) \cdot s}{\lambda \cdot \widetilde{b}(s) + s - \lambda} , \tag{1}$$

where $\rho = \frac{\lambda}{\mu}$ and $\widetilde{b}(s)$ is the $\mathcal{L}$–transform of PDF of the service time

The Laplace transform of the PDF $b(t)$ of a random variable is defined as follows (see e.g. [3], [13]):

$$\widetilde{b}(s) = \mathcal{L}\{b(t)\} = \int\limits_0^\infty e^{-st} b(t)\, dt . \tag{2}$$

Now, what distributions are suitable to model service times in computer networks ?

2 Long–tailed distributions in queuing modeling

Many researchers (e.g. [3], [13]) claim that service times (and several other quantities) in computer networks should be modeled by long–tailed distributions, i.e. distributions, the tails of which decay slower than exponentially:

$$\forall\, a > 0 \qquad \lim_{x\to\infty} e^{a\cdot x} F^c(x) = +\infty , \tag{3}$$

where $F^c(x) = 1 - F(x)$ is the tail (complementary cumulative distribution function) of the random variable.

A subclass of long–tailed distributions are power–tailed ones. Their tails decay hyperbolically:

$$\exists\, a > 0\, \exists\, c > 0 \qquad \lim_{x\to\infty} x^a \cdot F^c(x) = c . \tag{4}$$

Good examples of those are: Pareto distribution (power–tailed), lognormal and Weibull distribution (both long–tailed, but not power–tailed).

In the remaining part we restrict our attention to Pareto distribution. Its most commonly encountered form makes use of two parameters: the shaping parameter $\alpha > 0$ and the location parameter $\beta > 0$. A Pareto–distributed variable X has the CDF (cumulative distribution function) $F_X(x) = 1 - \left(\frac{\beta}{x}\right)^\alpha$ (for $x \geq \beta$; otherwise $F_X(x) = 0$) and PDF $f_X(x) = \frac{\beta^\alpha}{x^{\alpha+1}}$ (also for $x \geq \beta$).

2.1 Laplace transforms for long–tailed distributions

The $\mathcal{L}$–transform, defined by Equation (2), is well–defined and finite for the PDF of each random variable. Unfortunately, for some probability distribution functions the Laplace transform does not have an analytic form. According to e.g. [13], this is the case for all power–tailed distributions (e.g. Pareto distribution) and most other long–tailed ones (including lognormal and Weibull distributions).

The Laplace transforms, useful e.g. in $M/GI/1$ and $GI/M/1$ queuing systems analysis (see e.g. [1], have to be approximated somehow. Below, we present a popular method to approximate such transforms.

2.2 TAM – Transform Approximation Method

TAM (Transform Approximation Method) is described e.g. in [3], [13].

Let us consider a random variable B with the PDF $b(t)$ and CDF $F_B(t)$. We are trying to compute the $\mathcal{L}$–transform, i.e. the integral (2), approximately.

The essence of TAM is very simple: we discretize the set of values of the random variable B – that is contained in the interval $[0, +\infty[$ – obtaining n points: $b_1 < b_2 < \ldots < b_n$.

Let us denote the CDF's values in these points in the following way:

$$y_i = F(b_i) \qquad i = 1, \ldots, n \ .$$

We associate some probability masses with these points:

$$p_1 = \frac{y_1 + y_2}{2} \ , \quad p_n = 1 - \frac{y_{n-1} + y_n}{2} \ , \tag{5}$$

$$p_i = \frac{y_{i+1} - y_{i-1}}{2} \qquad \text{for } i = 2, \ldots, n-1 \ . \tag{6}$$

Then we can approximate the $\mathcal{L}$–transform $\widetilde{f}(s)$ of the PDF of X by a finite sum:

$$\breve{b}(s) = \sum_{i=1}^{n} p_i \cdot e^{-s \cdot b_i} \ . \tag{7}$$

The above description does not specify how to choose points b_i (or y_i). There are several approaches to do it (see below), but how to do it to get the best accuracy remains an open problem.

Possible parameterizations. One, the first, method was developed in 1998 by Gross and Harris. This first version used the formula $\breve{b}(s) = \frac{1}{n} \cdot \sum_{i=1}^{n} e^{-s \cdot b_i}$, where $b_i = F^{-1}(\frac{i}{n+1})$.

Such an approach is called *uniform–TAM*, or shortly UTAM. Currently, more widely used is the GTAM (*geometric–TAM*), which sets: $y_i = 1 - q^i$ (for some q such that $0 < q < 1$) and $b_i = F^{-1}(y_i)$.

Independently of the used parametrization, the classical TAM has some disadvantages:

- we must use some artificial probability masses (5)–(6), different from actual probability distribution,

- there are several sources of error: discretization, truncation, etc, that are difficult to estimate.

To overcome these faults, let us use the interval methods.

3 Basics of interval computations

3.1 Interval arithmetic

Now, we shall define some basic notions of intervals and their arithmetic. We follow a wide literature here, like the article [5] or books [4], [6], to name just a few examples.

We define the (closed) interval $[\underline{x}, \overline{x}]$ as a set $\{x \in \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}\}$. We denote all intervals by brackets; open ones will be denoted as $]\underline{x}, \overline{x}[$ and partially open as: $[\underline{x}, \overline{x}[$, $]\underline{x}, \overline{x}]$. (We prefer this notation to using the parenthesis that are used also to denote sequences, vectors, etc.)

We also use boldface lowercase letters to denote interval variables, e.g. $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$.

Following [7], $\mathbb{IR}$ denotes the set of all real intervals and $\mathbb{IC}_{rect}$ – the set of "rectangular" complex intervals (i.e. pairs of intervals for real and imaginary parts).

We design arithmetic operations on intervals so that the following condition was fulfilled: if we have $\odot \in \{+, -, \cdot, /\}$, $a \in \boldsymbol{a}$, $b \in \boldsymbol{b}$, then $a \odot b \in \boldsymbol{a} \odot \boldsymbol{b}$. We omit the actual formulae for arithmetic operations; they can be found elsewhere, e.g. ([4], [5], [6]).

Now, let us define a notion to set links between real and interval functions.

Definition 3.1. A function $\mathsf{f} \colon \mathbb{IR} \to \mathbb{IR}$ is an <u>inclusion function</u> of $f \colon \mathbb{R} \to \mathbb{R}$, if for every interval $\boldsymbol{x}$ within the domain of f the following condition is satisfied:

$$\{f(x) \mid x \in \boldsymbol{x}\} \subseteq \mathsf{f}(\boldsymbol{x}) \ . \tag{8}$$

The definition is analogous for functions $f \colon \mathbb{R}^n \to \mathbb{R}^m$.

4 Interval Transform Approximation Method

To introduce the interval analog of TAM (Section 2.2) let us consider a real–valued random variable B with the PDF $b(t)$ and CDF $F_B(t)$. In other papers, e.g. [10], [12] (see also [8], we use the theory of so–called *interval random variables* to develop the bounds on the $\mathcal{L}$–transform of $b(t)$, i.e. the integral (2). It leads to the following formula:

$$\breve{\mathsf{b}}(\boldsymbol{s}) = \sum_{i=1}^{n} p_i \cdot e^{-\boldsymbol{s} \cdot \boldsymbol{b}_i} \ , \tag{9}$$

where $\boldsymbol{b}_i = [b_{i-1}, b_i]$, $b_0 < b_1 < \cdots < b_n = +\infty$, $\boldsymbol{s} = \left[\underline{s}, \overline{s}\right]$ is an interval complex variable and $p_i = P\Big(\{B \in \boldsymbol{b}_i\}\Big)$.

In (9) $\breve{\mathsf{b}}$ is a function mapping the set of complex intervals $\mathbb{IC}_{rect}$ into itself. In [12] the proof is given that (9) is an inclusion function of $\widetilde{b}(s)$, if only $s \in \boldsymbol{s}$ and $\operatorname{Re} \underline{s} > 0$.

The essence of the method Let us now refer to TAM, described in Section 2.2 and develop its interval analog. We may call it "Interval TAM" or ITAM for brevity.

Not to get too involved in details (concerning the interval random variables theory), the essence of ITAM is to use Formula (9), instead of (7), to approximate the Laplace transform.

The advantages of such approach in comparison with the traditional TAM are obvious:

- we use correct probabilities associated with the intervals, not probability masses quite arbitrarily associated with chosen points, as in (5)–(6),
- we can naturally bound the discretization error and truncation error,
- as in other interval methods, we can bound the numerical error (see e.g. [4], [5], [6]).

5 Global optimization methods

As was shown in [14] (see also [9], [11] [12]), the optimization problem from Section 1 is a nonconvex one. This is why global optimization methods should be used to find its solution.

Two methods were used to solve the optimization problem: an interval branch and bound method and a genetic algorithm.

5.1 Branch and bound global optimization method

The general schema of the method (see e.g. [4], [5], [6]) can be expressed by the following pseudocode:

`IBB` $(\boldsymbol{x}^{(0)}; \mathtt{f}, \nabla\mathtt{f}, \nabla^2\mathtt{f}, \ldots; \mathtt{g}_1, \nabla\mathtt{g}_1, \nabla^2\mathtt{g}_1, \ldots, \mathtt{g}_m, \nabla\mathtt{g}_m, \nabla^2\mathtt{g}_m, \ldots)$
// $\boldsymbol{x}^{(0)}$ is the initial box
// $\mathtt{f}(\cdot)$ is the interval extension of the objective function $f(\cdot)$
// $\nabla\mathtt{f}(\cdot)$ and $\nabla^2\mathtt{f}(\cdot)$ are interval extensions of gradient and Hessian of $f(\cdot)$
// $\mathtt{g}_i(\cdot)$ are interval extensions of the constraints, etc.
// L_{sol} is the list of solutions
$[\underline{y}^{(0)}, \overline{y}^{(0)}] = \mathtt{f}(\boldsymbol{x}^{(0)})$;
compute f_{min} = the upper bound on the global minimum (e.g. objective value in a feasible point)
$L = \{(\boldsymbol{x}^{(0)}, \underline{y}^{(0)})\}$;
$L_{sol} = \emptyset$;
`while` $(L \neq \emptyset)$ `do`
    $\boldsymbol{x}$ = the element of L with the lowest function value underestimation ;
    compute the values of interval extensions of the constraint functions ;
    `if` ($\boldsymbol{x}$ is infeasible) `then` discard $\boldsymbol{x}$;
    update f_{min} if possible ;
    perform other rejection/reduction tests on $\boldsymbol{x}$;
    `if` ($\boldsymbol{x}$ is verified to contain a unique critical point or $\boldsymbol{x}$ is small and not infeasible) `then`
        add $\boldsymbol{x}$ to L_{sol} ;
    `else`
        bisect $\boldsymbol{x}$ to subboxes $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$;
        compute lower bounds $\underline{y}^{(1)}$ and $\underline{y}^{(2)}$ on the function value in the obtained boxes ;
        delete $\boldsymbol{x}$ from L ;
        `for` $i = 1, 2$ `do`
            put $(\boldsymbol{x}^{(i)}, \underline{y}^{(i)})$ on the list L preserving the increasing order of the lower bounds ;

```
        end for
        delete from L boxes with y_(i) > f_min ;
    end if
end while
delete from L_sol the boxes with y_(i) > f_min ;
return L_sol ;
end IBB
```

The above pseudocode mentions some "rejection/reduction tests" that may be used in conjunction with the IBB algorithm. There are several such tests. Most important of them are: several kinds of Newton operators, constraint propagation steps and, probably oldest of them all, monotonicity tests.

We do not describe them, as they are widely available in literature, e.g. [5] or in books [4], [6], etc.

5.2 Genetic algorithm (GA)

The GA used for the global optimization was based on the schema of [15]). It has the following features:

- the individuals are represented as vectors of floating point numbers taking value in [0, 1],
- the range selection, based on Formula (10; see below) is used (with $c = 0$),
- the uniform crossover is applied, i.e. two random individuals are combined linearly ($z_i = a_i \cdot x_i + (1 - a_i) \cdot y_i$) with the coefficient a_i uniformly distributed on [0, 1],
- the uniform mutation is used, done by replacing a given component of a vector describing an individual, by a random number uniformly distributed on the interval [0, 1],
- the population consists of 10 individuals,
- the best individual is guaranteed to survive if no better one is found (the elitist succession) – it replaces the worst point of the new generation then,
- the computations are finished when the best individual remains unchanged for 100 generations,
- a specific succession procedure, preserving the population diversity is used – see below.

The following probabilities are used in the rank succession:

$$p_1 = \frac{2-c}{n} \ , \quad p_n = \frac{c}{n} \ , \tag{10}$$
$$p_i = p_n + \frac{i-1}{n-1} \cdot (p_1 - p_m) \ , \ \text{for } i = 2, \ldots, n-1 \ ,$$

where $c \in [0, 1)$ is a parameter. Smaller values of c give better efficiency, but lower robustness.

The succession acts in the following way (see [15]):

1. check whether a better individual than the best of the old generation is found;
2. if it is not, replace the worst individual of the new generation by the "champion";

3. starting from the second–best individual do the following:
   a) compare the corresponding individuals from the old and new generation;
   b) if the one of them is both better than the other and more distant (in the sense of the Euclidean distance) from the "champion" then this individual survives;
   c) otherwise
   compute the distances form the champion (d_o and d_n) and (minimized) objective functions (f_o and f_n) for the old and the new individual;
   if ($d_n \cdot f_o > d_o \cdot f_n$) then the new one survives;
   otherwise the old one or the new one survive with probability $\frac{1}{2}$.

6 Numerical experiments

The lack of space means the authors can present only a limited number of experiments. We consider an $M/P/1$ queuing system, i.e. a single queue with one server, exponentially distributed interarrival time and Pareto–distributed service time. The Pareto distribution has the shape parameter $\alpha = 1.1$, in all cases.

Table 1 shows the performance of the IBB algorithm, setting the parameters of the queuing system. It uses ITAM to provide an enclosure of values of the $\mathcal{L}$–transform. The column "n.o. boxes" presents the number of boxes, containing the solution, that are returned by the branch–and–bound algorithm.

Hence, Tables 2, 3 and 4 show the performance of the GA, using the real–valued TAM.

Table 1. Results of the interval branch–and–bound for the single $M/P/1$ queue, capacity cost $c = 1.0$ and exponential delay cost with different values of v and k; ITAM with 100 discretization points

v	k	exec. time	f. eval.	n.o. boxes	λ	μ
10	10	0.66 sec.	73	7	[0.350004, 0.350006]	[0.350504, 0.350507]
10	2	25.94 sec.	3491	121	[0.150201, 0.150202]	[0.150701, 0.150703]
5	2	2.78 sec.	374	68	[0.275082, 0.275084]	[0.275582, 0.275585]
0.1	0.4	0.79 sec.	93	7	[0.387505, 0.387507]	[0.388005, 0.388008]

Table 2. Results of the GA for the single $M/P/1$ queue with $c = 1.000$; exponential delay cost with $v = 10$ and $k = 10$; TAM with 100 discretization points

No.	iter	f.eval.	exec. time	λ	μ	f_{opt}
1	592	5920	0.820 sec.	0.349933	0.402276	-1.172657
2	1008	10080	1.490 sec.	0.350136	0.352065	-1.223070
3	475	4750	0.690 sec.	0.349778	0.360653	-1.214125
4	1307	13070	1.870 sec.	0.350150	0.366449	-1.208701
5	825	8250	1.200 sec.	0.350001	0.358384	-1.216617
6	621	6210	0.910 sec.	0.350004	0.365069	-1.209935
7	962	9620	1.410 sec.	0.350144	0.368917	-1.206227
8	626	6260	0.930 sec.	0.350000	0.350906	-1.224094
9	545	5450	0.780 sec.	0.349723	0.370158	-1.204564
10	681	6810	0.970 sec.	0.350000	0.382247	-1.192753

Table 3. Results of the GA for the single $M/P/1$ queue with $c = 1.000$; exponential delay cost with $v = 10$ and $k = 2$; TAM with 100 discretization points

No.	iter	f.eval.	exec. time	λ	μ	$f_{\rm opt}$
1	422	4220	0.610 sec.	0.150345	0.153552	-0.221793
2	376	3760	0.570 sec.	0.150015	0.213225	-0.161931
3	720	7200	1.060 sec.	0.150001	0.151823	-0.223179
4	615	6150	0.870 sec.	0.149996	0.152541	-0.222455
5	380	3800	0.570 sec.	0.150876	0.164338	-0.211536
6	1346	13460	1.950 sec.	0.150306	0.156384	-0.218922
7	801	8010	1.190 sec.	0.150001	0.151720	-0.223281
8	704	7040	1.020 sec.	0.150008	0.150745	-0.224263
9	696	6960	1.040 sec.	0.150390	0.156966	-0.218425
10	754	7540	1.070 sec.	0.150482	0.156728	-0.218753

Table 4. Results of the GA for the single $M/P/1$ queue with $c = 1.000$; exponential delay cost with $v = 5$ and $k = 2$; TAM with 100 discretization points

No.	iter	fun. eval.	exec. time	λ	μ	$f_{\rm opt}$
1	390	3900	0.560 sec.	0.275408	0.280162	-0.751536
2	878	8780	1.240 sec.	0.275181	0.284501	-0.747017
3	885	8850	1.280 sec.	0.275066	0.281781	-0.749595
4	803	8030	1.140 sec.	0.275040	0.279522	-0.751807
5	553	5530	0.780 sec.	0.275733	0.327297	-0.705522
6	671	6710	0.990 sec.	0.275062	0.282095	-0.749281
7	615	6150	0.900 sec.	0.275066	0.282014	-0.749364
8	1231	12310	1.790 sec.	0.275038	0.279503	-0.751824
9	869	8690	1.280 sec.	0.275117	0.280306	-0.751107
10	1005	10050	1.460 sec.	0.275136	0.275691	-0.755700

7 Conclusions

Two algorithms to set optimal values of queuing system parameters were presented in the paper. Both performed moderately well.

The interval method was slow for some cases ($v = 10$ and $k = 2$ in Table 1, but it is more precise.

Both algorithms may still be improved, e.g. by using other approximations of the Laplace transform in IBB or using more advanced genetic approaches (like Hierarchical Genetic Strategy) in GA.

Bibliography

[1] I. Adan, J. Resing, "*Queueing Theory*", 2001, the book downloadable from: `http://www.cs.duke.edu/~fishhai/misc/queue.pdf` .

[2] S. Dewan, H. Mendelson, "User Delay Costs and Internal Pricing for a Service Facility", *Management Science*, Vol. 36, No. 12 (1990), pp. 1502–1517.

[3] M. J. Fischer, D. Gross, D. M. B. Masi, J. F. Shortle, "Analyzing the Waiting Time Process in Internet Queueing Systems With the Transform Approximation Method",

The Telecommunications Review, No. 12 (2001), pp. 21-32.

[4] E. Hansen, "*Global Optimization Using Interval Analysis*", Marcel Dekker, New York, 1992.

[5] R. B. Kearfott, "Interval Computations: Introduction, Uses, and Resources", *Euromath Bulletin*, Vol. 2, No. 1 (1996), pp. 95–112, available on the web at `http://interval.louisiana.edu/preprints/survey.ps` .

[6] R. B. Kearfott, "*Rigorous Global Search: Continuous Problems*", Kluwer Academic Publishers, Dordrecht, 1996.

[7] R. B. Kearfott, M. T. Nakao, A. Neumaier, S. M. Rump, S. P. Shary, P. van Hentenryck, "Standardized notation in interval analysis", *Reliable Computing*, No. 10 (2004); available on the web at `http://www.mat.univie.ac.at/~neum/software/int/notation.ps.gz` .

[8] B. J. Kubica, "*Estimating Utility Functions of Network Users – A Quasi–Bayesian Algorithm, Evolutionary Strategies and Interval Computations*", KAEiOG'03 Conference Proceedings (Konferencja Algorytmy Ewolucyjne i Optymalizacja Globalna), Łagów, 2003.

[9] B. J. Kubica, K. Malinowski, "*The Queuing Models of Mendelson – Review, Analysis and Some Generalizations*", Proceedings of the SPECTS 2004 Multiconference (International Symposium on Performance Evaluation of Computer and Telecommunications Systems), San Jose, 2004, pp. 478–483.

[10] B. J. Kubica, K. Malinowski, "*Przedziałowe przybliżenia zmiennych losowych i ich zastosowania*", KZM'04 (Konferencja Zastosowań Matematyki), 2005; the abstract available on the web at `http://www.impan.gov.pl/~zakopane/34/Kubica.pdf` .

[11] B. J. Kubica, K. Malinowski, "An Interval Global Optimization Algorithm Combining Symbolic Rewriting and Componentwise Newton Method Applied to Control a Class of Queueing Systems", *Reliable Computing*, Vol. 11, No. 5 (2005), pp. 393–411.

[12] B. J. Kubica, "*Optimization of Admission Control for Systems with Uncertain Parameters*", PhD Thesis, 2005, Institute of Control and Computational Engineering, Faculty of Electronics and Information Technology, Warsaw University of Technology.

[13] J. F. Shortle, M. J. Fischer, D. Gross, D. M. B. Masi, "Using the Transform Approximation Method to Analyze Queues with Heavy–Tailed Service", *Journal of Probability and Statistical Science*, Vol. 1, No. 1 (2003), pp. 17–30.

[14] S. Stidham Jr., "Pricing and Capacity Decisions for a Service Facility: Stability and Multiple Local Optima", *Management Science*, Vol. 38, No. 8 (1992), pp. 1121–1139.

[15] R. Yang, I. Douglas, "Simple Genetic Algorithm with Local Tuning: Efficient Global Optimization Technique", *Journal of Optimization Theory and Applications*, Vol. 98, No. 2 (1998), pp. 449–465.