EngOpt 2012 - International Conference on Engineering Optimization
Rio de Janeiro, Brazil, 1-5 July 2012.

Multiobjective Optimization of Finite Queueing Networks

N. L. C. Brito*, A. R. Duarte†, J. H. Ferreira‡, and F. R. B. Cruz§

*Departamento de Ciências Exatas, Universidade Estadual de Montes Claros,
39401-089 - Montes Claros - MG, Brazil
E-mail: nilson.brito@unimontes.br
†Departamento de Matemática, Universidade Federal de Ouro Preto,
35400-000 - Ouro Preto - MG, Brazil
E-mail: anderson@iceb.ufop.br
‡Departamento de Engenharia Elétrica, Centro Federal de Educação Tecnológica de Minas Gerais,
30510-000 - Belo Horizonte - MG, Brasil
E-mail: jhissa@des.cefetmg.br
§Departamento de Estatística, Universidade Federal de Minas Gerais,
31270-901 - Belo Horizonte - MG, Brazil
E-mail: fcruz@est.ufmg.br

1. Abstract
In this paper a multi-objective algorithm is presented to simultaneously optimize the total number of buffers, the overall service rate, and the throughput of a general-service finite queueing network. These conflicting objectives are optimized by means of a multi-objective genetic algorithm, designed to produce solutions for more than one objective. Some computational experiments that were conducted will be shown, in order to determine the efficacy and efficiency of the approach. Some insights are given.

2. Keywords: Genetic algorithms, network of queues, multi-objective optimization, throughput maximization, allocation.

3. Introduction
The focus here is on single-server queueing networks with exponentially distributed inter-arrival times and generally distributed service times, configured in an arbitrary acyclic topology. More specifically, the focus is on networks of $M/G/1/K$ queues, which in Kendall [1] notation stands for **M**arkovian arrivals, **G**enerally distributed service times, a single server, and the total capacity of K items, *including* the item in service. A realistic example of this type of network is shown in Fig. 1.

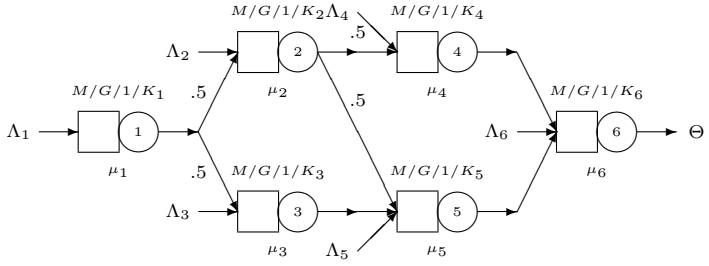


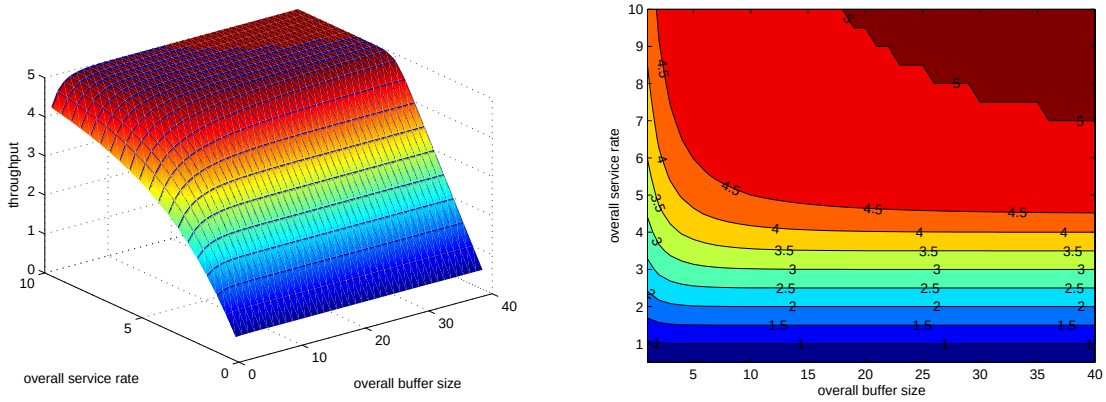
Figure 1: An $M/G/1/K$ queueing network

The goal is to obtain *simultaneously* the maximum throughput (Θ) using the minimum number of buffers ($\mathbf{K} = \{K_1, K_2, \ldots, K_n\}$) and the minimum service rates ($\boldsymbol{\mu} = \{\mu_1, \mu_2, \ldots, \mu_n\}$), given the topology and the external arrival rates ($\boldsymbol{\Lambda} = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_n\}$). Potential users of these queueing models include computer scientists and engineers. Indeed, these models may help to understand and to improve various real-life systems, including manufacturing [2], production [3] and health [4] systems, urban or pedestrian traffic [5], computer and communication systems [6], and web-based applications with tiered configurations [7].

There is a trade-off between the overall number of buffers and service rates and the resulting throughput. Because buffers and services can be very expensive, the overall buffer and service capacity should not be large. On the other hand, the highest possible network throughput should be reached. Unfortunately, the throughput is directly affected by the number of buffers allocated and the service rates.

Indeed, if the buffer and service capacity reduces there will be in general an undesirable reduction in the throughput. It is possible to observe this behavior in Fig. 2, which shows Θ for a single $M/G/1/K$ queue with $cv^2 = 1.5$ (squared coefficient of variation of the service time) and $\Lambda = 5$ users per time unit (external arrival rate), as a function of several values for buffer size, K, and service rate, μ (see Equations 4 and 10), as well as the respective contour plot.

Similar throughput behavior is also observed in a network of queues, as one shall see shortly. Notice the smoothness of the surface of the plot shown in Fig. 2. Convexity is also suggested. Similar results were reported for simple queueing networks [8]. However, the top surface flatness represents trouble for the traditional optimization methods. Indeed, Smith & Cruz [9] reported a successful optimization algorithm coupling Powell method with multiple starts to avoid premature convergence to locally optimal solutions.



(a) throughput Θ *versus* service rate μ and buffer size K (b) contour plot

Figure 2: Results of a single $M/G/1/K$ queue for arrival rate $\Lambda = 5.0$

In this study, an optimization approach is presented to simultaneously optimize the total number of buffers, the overall service rate, and the throughput of networks of $M/G/1/K$ queues. The proposed method produces a set of efficient solutions for more than one objective in the objective function [10]. With the proposed approach, the decision maker is able to evaluate the effect of solution replacement. Moreover, the multi-objective approach also allows the user to increase one objective (*e.g.*, throughput) while simultaneously reducing another objective (*e.g.*, buffer and service rate allocation).

This paper is organized as follows. A multi-objective evolutionary algorithm specifically developed to multi-objective optimization is presented in Sec. 4, along with the GEM, a performance evaluation tool used to approximate the throughput. In Sec. 5, the results of a comprehensive set of computational experiments are presented to show the efficiency of the approach. Finally, Sec. 6 concludes this paper with final remarks and suggestions for future research in the area.

4. Algorithms

4.1. Mathematical Programming Formulation

From a modeling point of view, the throughput maximization problem can be defined by a mixed-integer mathematical programming formulation, in which the total buffer and server costs are minimized and the throughput is maximized subject to integer buffer allocations and non-negative service rates. By defining a queueing network as a digraph $G(N, A)$, where N is a finite set of nodes (queues) and A is a finite set of arcs (pair of connected queues), a possible formulation is [11]:

$$\text{minimize } F(\mathbf{K}, \boldsymbol{\mu}), \tag{1}$$

subject to

$$K_i \in \{1, 2, \ldots\}, \ \forall i \in N, \tag{2}$$

$$\mu_i \geq 0, \ \forall i \in N, \tag{3}$$

where the decision variables K_i and μ_i indicate the total capacity of the service and the service rate for the ith $M/G/1/K$ queue, respectively. The objective functions, $F(\mathbf{K}, \boldsymbol{\mu}) \equiv \Big(f_1(\mathbf{K}), f_2(\boldsymbol{\mu}), -f_3(\mathbf{K}, \boldsymbol{\mu})\Big)$, are the total buffer allocation, $f_1(\mathbf{K}) = \sum_{\forall i \in N} K_i$, the overall service allocation, $f_2(\boldsymbol{\mu}) = \sum_{\forall i \in N} \mu_i$, and the overall throughput, $f_3(\mathbf{K}, \boldsymbol{\mu}) = \Theta(\mathbf{K}, \boldsymbol{\mu})$.

Notice that in the literature throughput is often modeled as a constraint that must be greater than a threshold value (Θ_τ) rather than as an objective that must be maximized (see, for instance, Andriansyah *et al.* [3]). The problem is that to solve this single-objective version of the problem the throughput constraint must be relaxed and to establish an appropriate Θ_τ is not a trivial task. Moreover, often a small decrease in the throughput results in a significant reduction in the buffer and service allocation. Such a trade-off between throughput and the number of buffers and service rates unfortunately will not be apparent in an *equivalent* single-objective formulation (which usually combines the multiple-objective formulation into a single-objective formulation by means of a vector of weights, $\boldsymbol{\omega}$). Additionally the determination of vector $\boldsymbol{\omega}$ is difficult and often leads to arbitrary single-objective formulations.

In this paper, a multi-objective evolutionary algorithm (MOEA) is used in combination with a generalized expansion method (GEM), which is a well-known method for obtaining accurate approximations of queueing network performance [12]. MOEAs are particularly suitable for multi-objective problems and have been shown to perform well in similar multi-objective problems of networks (*e.g.*, see Carrano *et al.* [13] and references therein). The algorithms will be presented in two parts. Initially, the performance evaluation algorithm will be described. Then, the proposed optimization algorithm will be detailed.

4.2. Performance Evaluation - Single Queues

When the interest is on single queues (not exactly the case here), the throughput $\Theta(\mathbf{K}, \boldsymbol{\mu})$ is:

$$\Theta(\mathbf{K}, \boldsymbol{\mu}) = \lambda(1 - p_K), \tag{4}$$

where λ is the external arrival rate and p_K is the called blocking probability, which is the probability that an item finds the system full (that is, the number of items in the systems is equal to the total capacity K). Thus, the problem of finding $\Theta(\mathbf{K}, \boldsymbol{\mu})$ reduces to determining p_K.

For the special case of pure Markovian systems (*i.e.*, $M/M/1/K$ queues), the blocking probability expression may be easily found in the queueing theory literature (*e.g.*, Gross *et al.*, [14]):

$$p_K = \frac{(1-\rho)\rho^K}{1-\rho^{K+1}}. \tag{5}$$

valid for $\rho < 1$, where $\rho \equiv \lambda/\mu$ is the system utilization. Relaxing the integrality constraint of K, it is possible to express in closed form the optimal buffer allocation for $M/M/1/K$ queues in terms of ρ and p_K:

$$K_{\mathrm{M}} = \left\lceil \frac{\ln\left(\frac{p_K}{1-\rho+p_K\rho}\right)}{\ln(\rho)} \right\rceil, \tag{6}$$

where $\lceil x \rceil$ is the smallest integer not superior to x. Consequently, it is possible to derive the optimal buffer allocation for $M/M/1/K$ queues:

$$x_{\mathrm{M}} = K_{\mathrm{M}} - 1. \tag{7}$$

For general-service multi-server queues $M/G/c/K$, the blocking probability must be derived by approximate techniques. In particular, a previous paper [9] has shown that a two-moment approximation based on the Markovian expression, Eq. (7), is quite effective:

$$x_\epsilon(cv^2) = x_{\mathrm{M}} + \mathrm{INT}\left[\frac{(cv^2-1)\sqrt{\rho}}{2} x_{\mathrm{M}}\right], \tag{8}$$

where $\mathrm{INT}[x]$ is the integer part of x. In particular, for single-server queues, $M/G/1/K$, given ρ and cv^2, the optimal buffer allocation may be written as:

$$x_\epsilon = \frac{\left[\ln\left(\frac{p_K}{1-\rho+p_K\rho}\right) + \ln(\rho)\right]\left(2+\sqrt{\rho}cv^2-\sqrt{\rho}\right)}{2\ln(\rho)}. \tag{9}$$

Finally, one can isolate p_K and determine a closed-form expression for the blocking probability in $M/G/1/K$ queues, as a function of K (note that for $M/G/1/K$ queues, $K = 1 + x_\epsilon$):

$$p_K = \frac{(1-\rho)\rho^{\left(\frac{2+\sqrt{\rho}cv^2-\sqrt{\rho}+2(K-1)}{2+\sqrt{\rho}cv^2-\sqrt{\rho}}\right)}}{1-\rho^{\left(2\frac{2+\sqrt{\rho}cv^2-\sqrt{\rho}+(K-1)}{2+\sqrt{\rho}s^2-\sqrt{\rho}}\right)}}. \tag{10}$$

4.3. Performance Evaluation - Networks of Queues

For networks of queues, the estimation of the throughput is made by means of the generalized expansion method (GEM), which is an algorithm that has been successfully used to estimate the performance of arbitrarily configured, finite queueing, acyclic networks [12]. The method is a combination of node-by-node decomposition and repeated trials, in which each queue is analyzed separately, and corrections are made to account for interrelated effects between network queues.

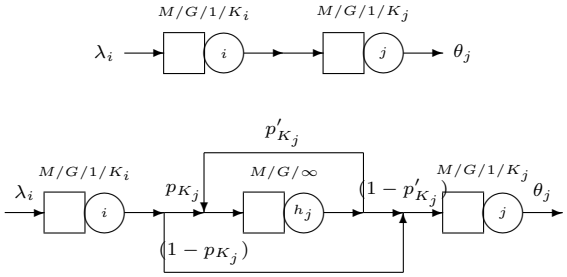


Figure 3: Generalized expansion method

As described in details by Kerbache & Smith [12], the GEM creates for each finite node j an auxiliary vertex (h_j) that is modeled as a $M/G/\infty$ queue, as seen in Fig. 3. For each entity placed into the system, vertex j may be blocked (with probability p_{K_j}), or may be unblocked (with probability $1 - p_{K_j}$). When blocking occurs, the entities are rerouted to vertex h_j and are delayed while node j is busy. Vertex h_j records the time an entity has to wait before entering vertex j and computes the effective arrival rate to vertex j.

The ultimate goal of GEM is to provide an approximation procedure that updates the service rates of upstream nodes and takes into account blocking in services caused by downstream nodes. The approximation below is based on the corrected blocking probabilities ($\tilde{p}_{K_i}$) of all nodes, which will provide an accurate estimation for the overall throughput (Θ).

$$\tilde{\mu}_i^{-1} = \mu_i^{-1} + p_{K_j}(\mu'_h)^{-1}. \tag{11}$$

Notice that the performance evaluation process must be conducted in a specific order. The performance evaluation of the network under study, defined as digraph $G(N, A)$, is presented in Fig. 4. The algorithm accounts for blocked services at upstream nodes, resulting in effective service rates that are reduced, in accordance with Eq. (11). Note that the performance evaluation algorithm is a variant of Dijkstra's labeling algorithm for the determination of shortest paths [15]. For instance, in the network illustrated in Fig.1, a valid evaluation sequence is $1 \to 2 \to 4 \to 3 \to 5 \to 6$. Specifically, the sequence must make it sure a node only will be assessed after all of its predecessors. Assuming that circuits are not present in $G(N, A)$, the GEM has a running time complexity of $\mathcal{O}(N^2)$, which is in accordance with Dijkstra's algorithm.

4.4. Optimization Algorithm

For the network under consideration, MOEAs seems to be a suitable choice for the multi-objective maximization of throughput. MOEAs are optimization algorithms that perform an approximate global search based on information obtained from the evaluation of several points in the search space [16].

```
algorithm
    read graph, G(N, A)
    read routing probabilities, p_[ij], ∀ (i, j) ∈ A
    read external arrival rates and service rates, Λ_i, μ_i, ∀ i ∈ N
    initialize set of labeled nodes, P ← ∅
    while P ≠ V
        choose j such that (j ∈ N) and (j ∉ P)
        if {i| (i, j) ∈ A} ⊆ P then
            /* compute performance measures */
            compute p_Kj θ_j
            /* forward information to successors */
            for ∀ k ∈ {k'| (j, k') ∈ A} then
                λ_k ← λ_k + θ_j p_[jk]
            end for
            /* label node as pre-evaluated */
            P ← P ∪ {j}
        end if
    end while
end algorithm
```

Figure 4: Performance evaluation algorithm

The population of points that converge to an optimal value are obtained through the application of the genetic operators, *mutation*, *crossover*, *selection*, and *elitism*.

Each one of these operators characterizes an instance of a MOEA and can be implemented in several different ways. Additionally, MOEA convergence is guaranteed by assigning a value of fitness to each population member and preserving diversity. In fact, recent successful applications of GAs were reported for single-objective applications [17] and for multiple-objective applications [13]. The instance of MOEA used in this study is based upon the elitist non-dominated sorting genetic (NSGA-II) algorithm of Deb *et al.* [18], which is shown in Fig. 5. In the application of GAs for multi-objective optimization, the selection and elitism operators must be specifically structured to correctly identify optimal conditions as it will be shown shortly.

```
algorithm
    read graph, arrival, service rates, G(N, A), Λ_i ∀ i ∈ N
    P_1 ← GenerateInitialPopulation(popSize)
    for i = 1 until numGen do
        /* generate offspring by crossover and mutation */
        Q_i ← MakeNewPop(P_i)
        /* combine parent and offspring */
        R_i ← P_i ∪ Q_i
        /* find non-dominated fronts F = (F_1, F_2, ...) */
        F ← FastNonDominatedSort(R_i)
        /* find new population by */
        /* the crowding-distance-assignment */
        P_{i+1} ← GenerateNewPopulation(R_i)
    end for
    P_{numGen+1} ← ExtractParetoSet(P_numGen)
    write P_{numGen+1}
end algorithm
```

Figure 5: Elitist multi-objective genetic algorithm (NSGA-II)

Elitism is based on the concept of dominance. Point $\mathbf{x}_i = (x_{i_1}, x_{i_2}, \ldots, x_{i_n})$ dominates point $\mathbf{x}_j = (x_{j_1}, x_{j_2}, \ldots, x_{j_n})$ if $\mathbf{x}_i$ is superior to $\mathbf{x}_j$ in one objective ($f_k(\mathbf{x}_i) < f_k(\mathbf{x}_j)$, for minimization) and is not inferior in any other objective ($f_\ell(\mathbf{x}_i) \not> f_\ell(\mathbf{x}_j)$, for minimization). To perform elitism, the fast non-dominated sorting algorithm is employed [18]. This algorithm separates the individuals in the population into several layers (or fronts) $\mathcal{F}_i$, such that the solutions in $\mathcal{F}_1$ are non-dominated, and every solution

in a given front $\mathcal{F}_i$, $i > 1$, is dominated by at least one solution in $\mathcal{F}_{i-1}$, and not by any solution in $\mathcal{F}_j$, $j \geq i$. This can be achieved in $\mathcal{O}(n \log n)$ time [18].

Selection is performed by sequentially selecting points from each non-dominated front $(\mathcal{F}_1, \mathcal{F}_2, \ldots)$ until the number of required individuals for the next iteration is obtained. Some decision must be made if the maximum number of individuals is exceeded after the addition of a group of individuals from front $\mathcal{F}_i$. One possibility is to compute a measure of diversity, such as the crowding distance defined by Deb *et al.* [18], to ensure the highest diverse population. Thus, only the points with the largest crowding distance are kept for future iterations.

Crossover and mutation are dependent on the application, as well known. For the problem at hand, the *uniform crossover* mechanism was selected [19], which is popular in multivariable encodings due to its efficiency in identifying, inheriting, and protecting common genes, as well as re-combining non-common genes [20]. In this mechanism, crossover is performed for each variable with a probability (`rateCro`), in accordance with the crossover operator. The crossover operator used in the algorithm is the *simulated binary crossover operator* (SBX) [21]. SBX is quite convenient for real-coded GAs because of its ability to simulate *binary crossover operators* avoiding re-encoding the variables. The children $(x_{i,(\bullet,t+1)})$ are calculated from the parents $(x_{i,(\bullet,t)})$ according to the following equations

$$x_{i,(1,t+1)} = 0.5\Big[(1+\beta)x_{i,(1,t)} + (1-\beta)x_{i,(2,t)}\Big], \tag{12}$$

$$x_{i,(2,t+1)} = 0.5\Big[(1-\beta)x_{i,(1,t)} + (1+\beta)x_{i,(2,t)}\Big], \tag{13}$$

where β is a random variable with the following density function:

$$f(\beta) = \begin{cases} 0.5(\eta+1)\beta^{\eta}, & \text{if } \beta \leq 1, \\ 0.5(\eta+1)\frac{1}{\beta^{\eta+2}}, & \text{otherwise,} \end{cases} \tag{14}$$

noticing that Equations (12) and (13) are designed to create children solutions that posses a similar search power to a single-point crossover of binary-coded GAs [22]. By adjusting η, several different weights (β) can be generated to produce children that are more (small η) or less (large η) similar to their parents.

For each individual gene (each decision variables K_i or μ_i), the mutation scheme occurs with a specific probability (`rateMut`). As suggested by Deb & Agrawal [22], Gaussian perturbations were added to the decision variables, $K_i + \varepsilon_i$ and $\mu_i + \varepsilon_{N+i}$, for all $i \in N$, with $\varepsilon_i \sim \mathcal{N}(0,1)$, $i \in \{1, 2, \ldots, 2N\}$.

Finally, to ensure feasibility of constraints (2) and (3) after crossover and mutation, the integer variables values must be rounded accordingly and all the variables readjusted by applying reflection operators as follows

$$K_{\mathrm{rfl}_i} = K_{\mathrm{lowlim}} + |K_i - K_{\mathrm{lowlim}}|, \tag{15}$$

and

$$\mu_{\mathrm{rfl}_i} = \mu_{\mathrm{lowlim}_i} + |\mu_i - \mu_{\mathrm{lowlim}_i}|, \tag{16}$$

where K_{lowlim} is the lower limit of buffer allocation (*i.e.*, $K_{\mathrm{lowlim}} = 1$) and μ_{lowlim_i} is the lower limit of service allocation (to ensure that $\rho < 1$ holds). Notice that K_i and μ_i are the resulting values after crossover and mutation, and K_{rfl_i} and μ_{rfl_i} are the results after reflection. The proposed scheme always generates feasible solutions without avoiding or favoring any particular solution.

4.5. Convergence Issues

Recently, the stopping criterion of multi-objective optimization evolutionary algorithms has been analyzed in detail. Evidently, the maximum number of generations (`numGen`) plays an important role in the quality of the solutions. However, increasing the number of generation may not be ideal because computational time is wasted when many iterations do not lead to a significant improvement. Thus, Rudenko & Schoenauer [23] suggested that a superior stopping criterion is obtained when a fixed number of iterations are performed without improvement. To demonstrate the complexity of the issue, Rudenko

& Schoenauer [23] conducted a comprehensive set of computational experiments. Their results revealed that an obvious stopping criterion, such as the entire population possessing a rank of 1, did not indicate that evolution should be terminated. Rudenko & Schoenauer [23] proposed a local stopping criterion that computes a measure of the stability of non-dominated solutions after each iteration based on the stabilization of the maximal crowding distance, d_l, measured over L generations and calculated by the following standard deviation:

$$\sigma_L = \sqrt{\frac{1}{L}\sum_{l=1}^{L}(d_l - \bar{d}_L)^2}, \tag{17}$$

in which $\bar{d}_L$ is the average of d_l over L generations and the criterion $\sigma_L < \delta_{\text{lim}}$ should indicate when MOEA should stop. Rudenko & Schoenauer [23] suggested that L and δ_{lim} should be set to 40 and 0.02, respectively, which leads to a stopping criteria that is $\sigma_{40} \leq 0.02$.

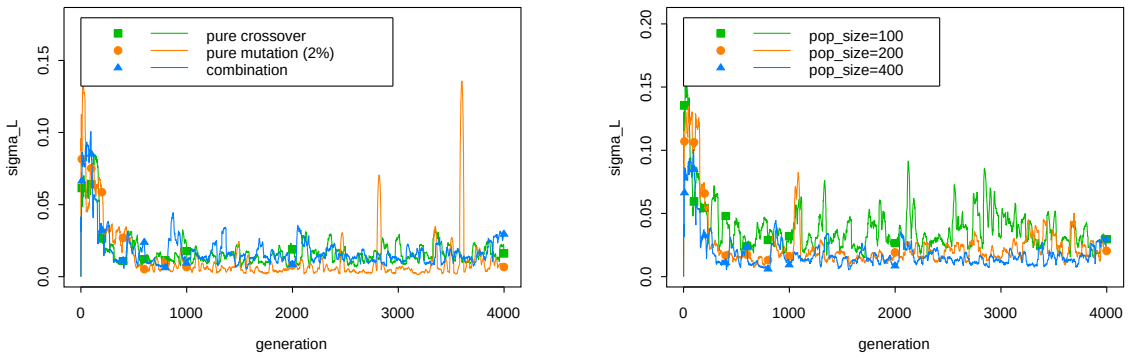
5. Results and Discussion

In order to use an implementation of GEM [9] based on the International Mathematics and Statistics Library (IMSL), the optimization algorithm was implemented in FORTRAN and the code is available from the authors upon request for educational and research purposes. Firstly, the computational experiments were conducted to discover the sub-optimal set of parameters that guarantee rapid convergence. Finally, a detailed analysis of a queueing network was performed.

5.1 Parameter Setup

As indicated by previous studies on GAs, the sub-optimal set of parameters to ensure rapid convergence with a minimal amount of computational effort must be determined by trial and error. Only results obtained for the network from Fig. 1 are presented, although different topologies of acyclic similarly sized networks were also tested. The results (not presented) were similar.

Convergence is monitored in Fig. 6-a in terms of σ_L along the generations, for combinations of crossover and mutation. It is remarkable that pure mutation could solve the problem but using SBX operator (crossover) may remove instabilities from σ_L. Thus, combining mutation and SBX may be advantageous regardless of the number of queues in the network. Figure 6-b discloses that the population size (`popSize`) affects algorithm convergence, although the population size cannot be arbitrarily increased, as it may risk the processing time.

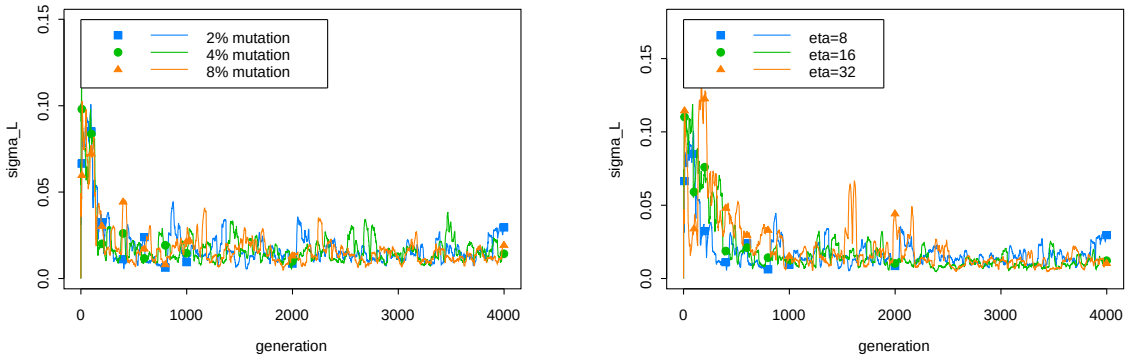


(a) crossover and mutation

(b) population size

Figure 6: Effects of crossover and mutation and population size

The standard deviation as function of `rateMut` is displayed in Fig. 7-a disclosing that an increase in the mutation rate may speed up the convergence but after a certain rate no further improvements are obtained. Then, mutation rates between up to 2% seemed to respond for the best results, as shown by the experimental results provided. In Fig. 7-b the standard deviation is presented as a function of parameter η which responds for the SBX operator performance. From these experiments, it is possible to conclude that values above 8 are not effective in terms of stability.



(a) mutation rate (b) η

Figure 7: Effect of the mutation rate and η

As a final word concerning the best group of parameters for the algorithm, one could use the following combination: (i) combined use of SBX and mutation, with (ii) a mutation rate below 2%, (iii) although greater the better the population, 400 individuals seem to be enough, and (iv) the dispersion parameter, η, should not go above 8. To ensure a finite computation time, a maximum number of generations `numGen` was set to 4,000. Fortunately, MOEAs are robust enough to perform well in a broad range of problems, as confirmed by the experiments run (not shown).

5.2 Network Analysis

The network presented in Fig. 1 was analyzed with the proposed method. Two different squared coefficients of variation were analyzed, $cv^2 = 0.5$ and 1.5, with arrival rate ($\Lambda_1 = 5.0$). First, the convergence speed of the genetic algorithm was confirmed to be robust for this type of problem. The experimental set-up was identical to the previous analysis. However, the results indicated that convergence was stable at 2,000 iterations.

Figure 8 shows the results. It is possible to see the final population and the respective contour plot. It is remarkable the resemblance between these two contour plots and the exact contour plot for a single queue, Fig. 2-b. The results suggest that the *networks* of queues seem to behave such as an equivalent *single* queue. Unfortunately, it is unknown whether or not it would be possible to derive some sort of algorithm to predict the parameters of such an equivalent single queue.

6. Conclusions

In order to optimize the throughput, the buffer sizes, and the service rates of single server, general-service queueing networks, a multi-objective approach was presented. The generalized expansion method (GEM) was coupled with a multi-objective genetic algorithm (MOGA) to make it possible to derive insightful Pareto curves displaying the trade-off between throughput and the allocation of buffers and service rates.

Topics for future investigation in this area include extensions to networks of multi-server queues and networks of general-arrival queues, possibly by means of kernels [6]. Also interesting is to consider different performance measures, such as the WIP, sojourn time, and so on. These are only few examples of possible topics for research.

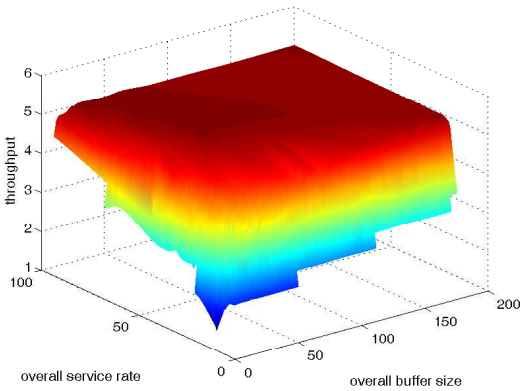
7. Acknowledgments

This research is partially funded by CNPq, CAPES, and FAPEMIG.

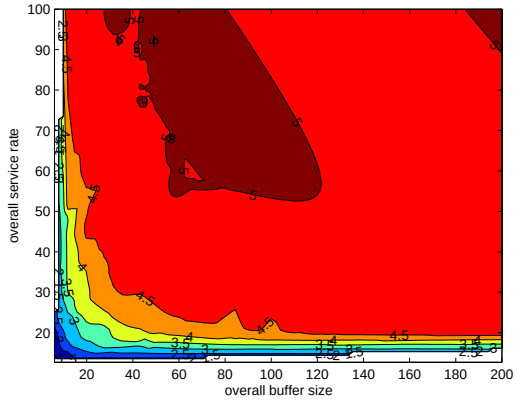
8. References

[1] D. G. Kendall, Stochastic processes occurring in the theory of queues and their analysis by the method of embedded Markov chains, *Annals Mathematical Statistics*, 1953, 24, 338–354.

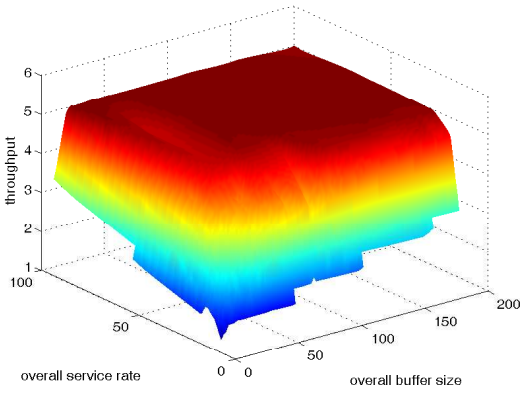
[2] A. M. Youssef, H. A. ElMaraghy, Performance analysis of manufacturing systems composed of mod-



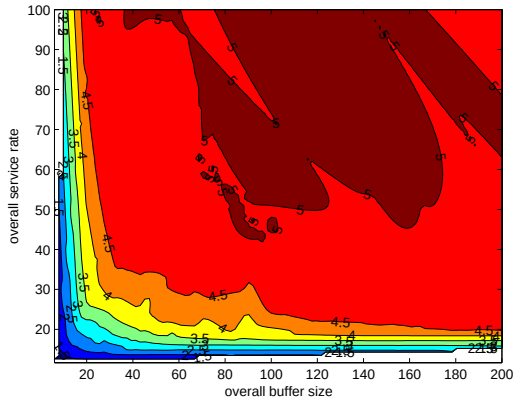
(a) final surface for $s^2 = 0.5$



(b) contour plot for $s^2 = 0.5$



(c) final surface for $s^2 = 1.5$



(d) contour plot for $s^2 = 1.5$

Figure 8: Final results for the network from Fig. 1

ular machines using the universal generating function, *Journal of Manufacturing Systems*, 2008, 27 (2), 55–69.

[3] R. Andriansyah, T. van Woensel, F. R. B. Cruz, L. Duczmal, Performance optimization of open zero-buffer multi-server queueing networks, *Computers & Operations Research*, 2010, 37 (8), 1472–1487.

[4] C. Osorio, M. Bierlaire, An analytic finite capacity queueing network model capturing the propagation of congestion and blocking, *European Journal of Operational Research*, 2009, 196 (3), 996–1007.

[5] F. R. B. Cruz, T. van Woensel, J. M. Smith, K. Lieckens, On the system optimum of traffic assignment in $M/G/c/c$ state-dependent queueing networks, *European Journal of Operational Research*, 2010, 201 (1), 183–193.

[6] G. M. Gontijo, G. S. Atuncar, F. R. B. Cruz, L. Kerbache, Performance evaluation and dimensioning of $GIX/M/c/N$ systems through kernel estimation, *Mathematical Problems in Engineering*, 2011, Article ID 348262, 20 p.

[7] K. Chaudhuri, A. Kothari, R. Pendavingh, R. Swaminathan, R. Tarjan, Y. Zhou, Server allocation algorithms for tiered systems, *Algorithmica*, 2007, 48 (2), 129–146.

[8] L. E. Meester, J. G. Shanthikumar, Concavity of the throughput of tandem queueing systems with finite buffer storage space, *Advances in Applied Probability*, 1990, 22 (3), 764–767.

[**9**] J. M. Smith, F. R. B. Cruz, The buffer allocation problem for general finite buffer queueing networks, *IIE Transactions*, 2005, 37 (4), 343–365.

[**10**] V. Chankong, Y. Y. Haimes, *Multiobjective Decision Making: Theory and Methodology*, Elsevier, Amsterdam, The Netherlands, 1983.

[**11**] F. R. B. Cruz, G. Kendall, L. While, A. R. Duarte, N. L. C. Brito, Throughput maximization of queueing networks with simultaneous minimization of service rates and buffers, *Mathematical Problems in Engineering*, 2012, Article ID 348262, 19 p.

[**12**] L. Kerbache, J. M. Smith, The generalized expansion method for open finite queueing networks, *European Journal of Operational Research*, 1987, 32, 448–461.

[**13**] E. G. Carrano, L. A. E. Soares, R. H. C. Takahashi, R. R. Saldanha, O. M. Neto, Electric distribution network multiobjective design using a problem-specific genetic algorithm, *IEEE Transactions on Power Delivery*, 2006, 21 (2), 995–1005.

[**14**] D. Gross, J. F. Shortle, J. M. Thompson, C. M. Harris, *Fundamentals of Queueing Theory*, 4th Ed., Wiley-Interscience, New York, NY, USA, 2009.

[**15**] E. W. Dijkstra, A note on two problems in connection with graphs, *Numerical Mathematics*, 1959, 1, 269–271.

[**16**] K. Deb, *Multi-objective Optimisation using Evolutionary Algorithms*, Wiley, 2001.

[**17**] F.-T. Lin, Solving the knapsack problem with imprecise weight coefficients using genetic algorithms, *European Journal of Operational Research*, 2008, 185 (1), 133–145.

[**18**] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, 2002, 6 (2), 182–197.

[**19**] T. Bäck, D. Fogel, Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation*, Institute of Physics Publishing and Oxford University Press, 1997.

[**20**] X.-B. Hu, E. Di Paolo, An efficient genetic algorithm with uniform crossover for the multi-objective airport gate assignment problem. *Proceedings of IEEE Congress on Evolutionary Computation, CEC 2007*, 2007, Singapore, 55–62.

[**21**] K. Deb, H.-G. Beyer, *Self-adaptive genetic algorithms with simulated binary crossover*, Technical report no. CI-61/99, Department of Computer Science/XI, University of Dortmund, 44221 Dortmund, 1999, Germany. URL `citeseer.ist.psu.edu/deb99selfadaptive.html`

[**22**] K. Deb, R. B. Agrawal, Simulated binary crossover for continuous search space, *Complex Systems*, 1995, 9, 115–148. URL `citeseer.ist.psu.edu/deb95simulated.html`

[**23**] O. Rudenko, M. Schoenauer, A steady performance stopping criterion for Pareto-based evolutionary algorithms. *Proceedings of the 6th International Multi-Objective Programming and Goal Programming Conference*, 2004, Hammamet, Tunisia.