Int j simul model 12 (2013) 1, 50-61
ISSN 1726-4529 Original scientific paper

MULTI-OBJECTIVE BUFFER SPACE ALLOCATION WITH THE CROSS-ENTROPY METHOD

Bekker, J.
Department of Industrial Engineering, Stellenbosch University, Private Bag X1,
Matieland 7602, South Africa
E-Mail: jb2@sun.ac.za

Abstract
The buffer allocation problem (BAP) has been widely studied by researchers while pursuing diverse research goals. Similarly, the cross-entropy method has been applied to a variety of optimisation problems with single objectives. In this article it is extended to the multi-objective case and proposed as a computationally economic approach to optimise at least two conflicting objectives of the BAP, namely throughput rate and allocated buffer space, while using computer simulation as evaluation function of small to large stochastic queuing networks of unreliable resources. No assumptions are made regarding the service time, time-to-failure and repair time distributions, and a general solution for obtaining the network-related Pareto front is proposed. The results for test networks indicate that reasonable Pareto fronts can be obtained via a low number of multi-objective solution evaluations using the modified cross-entropy method (CEM).

(Received in April 2012, accepted in September 2012. This paper was with the author 1 month for 1 revision.)

Key Words: Cross-Entropy, Simulation, Multi-Objective Optimisation, Buffer Allocation

1. INTRODUCTION

In this article we study the multi-objective optimisation (MOO) of finite buffer queuing networks with the cross-entropy method (CEM). The CEM was developed for *Importance Sampling* by [1] and has been extended by [2] for application in continuous and combinatorial optimisation with single-objective functions. Recently, it has been adapted for continuous multi-objective optimisation by [3] and tested on some benchmark problems from [4]. In this article we adapt the cross-entropy multi-objective optimisation algorithm for the discrete case. We evaluate the proposed algorithm using a typical problem in queuing networks, namely the buffer allocation problem (BAP).

Finite queuing networks are associated with many practical systems through which discrete or continuous flow occurs, such as manufacturing systems and telecommunication networks. These networks often exhibit flow variation or asynchronous part movement, hence the need for buffer space in the network. One of the network design priorities is to maximise the network throughput rate, which increases with more buffer space [5]. However, buffer space may be costly for several reasons in commercial projects, and costs must be minimised. This gives rise to the BAP. The BAP is usually formulated as a stochastic, non-linear, integer mathematical programming problem and is computationally hard to solve [6]. The problem has several variations, see [6] and [7].

The BAPs used in this study have series and general topologies with discrete, constrained buffer sizes. The problem is similar to those used in studies by [2] and [8], and we briefly describe it here: a production line consists of a series of m machines with $m-1$ buffers. There are B_i spaces available per buffer, while the spaces in front of the first and the last machine are infinite. Generally, the machines have exponentially distributed processing and repair times with mean rates μ_i and r_i respectively. The machine failures of our models are

DOI:10.2507/IJSIMM12(1)5.228

operation-dependent failures (ODF), which are more realistic than time-based failures [9]. A network with a series topology is shown schematically in Fig. 1.

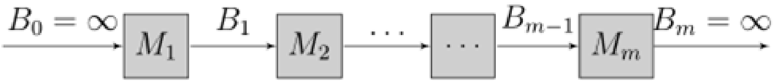

Figure 1: A typical series queuing network.

Researchers have studied the BAP for several decades, see for example [6], [8], [10], [11], [12], [13], and recently, [14] and [15]. Tabu search and simulation were applied to the BAP by [16], while [17] applied a genetic algorithm (GA) to a BAP with exponential failure and repair rates but deterministic processing times.

In MOO, the decision maker evaluates two or more conflicting objectives, often with non-commensurable units of measurement. Specific combinations of decision variables yield different combinations of objective function values. A subset of the latter forms a *Pareto front*, which is a set of non-dominated solutions [4]. This means that no solution in the Pareto set is "better" than any other solution and the decision maker can select from various solution candidates.

The MOO approach has been applied to the BAP, see for example [18], and [5] studied buffer allocation and throughput rate trade-offs in M|G|1|K queuing networks. In [19], an informative classification of 50 articles in the literature on the BAP is given.

Our research objective is as follows: Computer simulation is widely used to evaluate objective functions of complex, dynamic stochastic real-world systems. When solving practical engineering problems, it is often required to optimise more than one objective. Judged by the literature, the CEM seems to converge relatively fast when solving single-objective optimisation problems. We propose adapting the CEM for multi-objective simulation-optimisation studies, with the idea to do as *few objective evaluations as possible, while still finding a satisfactory Pareto front*. The BAP is chosen as a test bench because it is well researched with reference solutions available and also presents problems that are very hard to solve. We show that the CEM as a multi-objective optimiser can minimise the total buffer space by appropriately allocating space to each buffer while maximising throughput rate.

We briefly present the CEM next, followed by a discussion of experiments and results.

2. THE CROSS-ENTROPY METHOD AND THE BAP

The CEM for optimisation is discussed in [2], on the CEM website [20], in a complete journal issue on the CEM [21], and recently, in [22]. [23] apply the CEM to the BAP and maximise the throughput rate of various production line configurations. The discrete flow BAP, as stated earlier, is a combinatorial optimisation problem (COP). For convenience, we briefly state the cross-entropy approach to the BAP; for detail see [23].

A possible buffer allocation is the vector $\mathbf{x} = (x_1, ..., x_{m-1})$ in the (finite) set of all possible buffer allocations $\mathcal{X} = \{(x_1, \dots, x_{m-1}): x_i \in \{0,1,\dots,n\}, i = 1,\dots,m-1\}$. Let $S(\mathbf{x})$ be the steady state throughput rate for the buffer allocation $\mathbf{x}$, then the optimisation problem is:

$$\text{maximise } S(\mathbf{x}) \text{ over } \mathbf{x} \in \mathcal{X} \qquad (1)$$

The estimation formulation is:

$$\text{maximise } \hat{S}(\mathbf{x}) \text{ over } \mathbf{x} \in \mathcal{X} \qquad (2)$$

The CEM requires that an estimation problem be associated with the optimisation problem of (1), and to do so, one defines a collection of indicator functions $I_{\{S(\mathbf{x})\geq\gamma\}}$ on $\mathcal{X}$ for different values of the threshold $\gamma \in \mathbb{R}$. Let $\{f(\cdot,\mathbf{v}), \mathbf{v} \in \mathcal{V}\}$ be a family of discrete probability mass functions (pmfs) on $\mathcal{X}$ that are parameterised by a real-value vector $\mathbf{v}$. To solve the problem associated with (1), assume $\mathbf{u} \in \mathcal{V}$ and estimate the probability $l = \mathbb{P}_{\mathbf{u}}\{S(\boldsymbol{X}) \geq \gamma\} = \sum_{\mathbf{x}} I_{\{S(\mathbf{x}) \geq \gamma\}} f(\mathbf{x};\mathbf{u}) = \mathbb{E}_{\mathbf{u}} I_{\{S(\mathbf{x}) \geq \gamma\}}$ with $f(\mathbf{x};\mathbf{u})$ being the probability mass function (pmf) on $\mathcal{X}$ and γ some chosen level. Suppose now γ is equal to γ^*, then $l = f(\mathbf{x}^*;\mathbf{u})$, which is a very small probability. It can be estimated with *Importance Sampling* by taking a random sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ from a different pmf g and estimate l via:

$$\hat{l} = \frac{1}{N} \sum_{k=1}^{N} I_{\{S(\mathbf{X}) \geq \gamma\}} \frac{h(\mathbf{X}_k;\mathbf{u})}{g(\mathbf{X}_k)} \tag{3}$$

This is the unbiased *importance sampling estimator* of l. The optimal way to estimate l is to use the change of measure with a different pmf:

$$g^*(\mathbf{x}) = \frac{I_{\{S(\mathbf{x}) \leq \gamma\}} h(\mathbf{x};\mathbf{u})}{l} \tag{4}$$

Since this optimal pmf is generally difficult to obtain and depends on the unknown l, one chooses g such that the *cross-entropy* or *Kullback-Leibler distance* between g and g^* is minimal. The Kullback-Leibler distance between two pmfs g^* and h is defined as:

$$D(g^*, h) = \mathbb{E}_{g^*} \ln \frac{g^*(\mathbf{X})}{h(\mathbf{X})} = \sum_{\mathbf{x}} g^*(\mathbf{x}) \log g^*(\mathbf{x}) - \sum_{\mathbf{x}} g^*(\mathbf{x}) \log h(\mathbf{x}) \tag{5}$$

Because $I_{\{S(\mathbf{X}) \geq \gamma\}}$ is non-negative, and the pmf f is parameterised by a finite dimensional vector $\mathbf{v}$, i.e. $f(\mathbf{x}) = f(\mathbf{x};\, \mathbf{v})$, $g(\mathbf{x}) = f(\mathbf{x};\, \mathbf{v})$ for some reference parameter $\mathbf{v}$ [22]. To estimate l, we choose $\mathbf{v}$ such that $D(g^*, f(\cdot;\tilde{\mathbf{v}}))$ is minimal. That means $\mathbb{E}_{\mathbf{v}} I_{\{S(\mathbf{x}) \geq \gamma\}} \log f(\boldsymbol{X}, \tilde{\mathbf{v}})$ should be maximal. It has been shown that if the parameterised distribution is Bernoulli, the elements of the corresponding probability matrix can be estimated with:

$$\hat{\mathrm{p}}_{ij} = \frac{\sum_{k=1}^{N} I_{\{\hat{S}(X_k) \geq \gamma\}} I_{\{X_{ki} = j\}}}{\sum_{k=1}^{N} I_{\{\hat{S}(X_k) \geq \gamma\}}} \tag{6}$$

A smoothing update rule is recommended by [2] and shown in (7) for $\hat{\mathbf{v}}_t$, with $0 \leq \alpha \leq 1$:

$$\hat{\mathbf{P}}_t = \alpha \hat{\mathbf{P}}_t + (1 - \alpha) \hat{\mathbf{P}}_{t-1} \tag{7}$$

The single-objective BAP optimisation with the cross-entropy method is shown in Algorithm 1 [23] for a Bernoulli vector P.

Algorithm 1: CEM algorithm for the stochastic BAP

```
1: Set all elements in $\hat{P}_0 = 0.5$. Set $t = 0$.
2: Generate $\boldsymbol{X}_1, \ldots, \mathbf{X}_N$ using $\hat{P}_{t-1}$, and compute the sample $1 - \varrho$ quantile $\hat{\gamma}_t$ of the
   performance function using $\hat{\gamma}_t = S_{(\lceil 1-\varrho \rceil N)}$.
3: Use the same sample and update $\hat{P}_t$ using (6).
4: Smooth $\hat{P}_t$ as follows: $\hat{P}_t \leftarrow \alpha \hat{P}_t + (1 - \alpha) \hat{P}_{t-1}$.
5: If for some $t \geq \delta$, say $\delta = 5$, $\hat{\gamma}_t = \hat{\gamma}_{t-1} = \cdots \hat{\gamma}_{t-\delta}$, then stop, otherwise set $t \leftarrow t + 1$
   and return to Step 2.
```

Next, we present the multi-objective optimisation formulation for solving the BAP with the CEM.

3. FORMULATION OF THE BAP FOR THE CEM

Researchers usually attempt to minimise the total number of buffer spaces while maximising the throughput rate $T_R(\mathbf{x})$ of a given queuing network. We do not minimise the total number of buffer spaces directly, but define a modified second objective which is more practical: we consider the total number of buffer slots assigned, but measure the actual system occupation due to *observed* work in progress (WIP). Since the WIP varies over time, the time duration of the WIP at each level (0, 1, 2, ...) is recorded, and after a finite time *T*, the p_qth percentile of the WIP is observed. Let T_i denote the total time that the WIP was at level WIP_i during the simulation run, and n_{p_q} is the WIP level of the n_pth time percentile, then n_{p_q} is determined by:

$$\text{Minimise } \mathrm{n}_{\mathrm{p_q}} \tag{8}$$

$$\text{Subject to } \sum_{i=0}^{n_{pq}} [T_i \times WIP_i] / \sum_{i=0}^{WIP_M} [T_i \times WIP_i] \geq p_q \tag{9}$$

In (9), WIP_M is the maximum WIP level observed during *T*. The reason for using a *time-based* percentile is to consider the *intensity* of each WIP level. If only the values of the WIP level are used, a few high values may occur for a short time during the simulation run, and more buffer spaces will be allocated by the algorithm. These spaces will then hardly be used once the real system is implemented, resulting in wasted space. The time-based percentile rules out buffer size extremities which exist for short time periods, or will include them if they are significant. This objective is denoted by $W_P(\mathbf{x})$ and must be minimised. Note that, since a percentile is used, the system will not be able to accommodate WIP for $(1 - p_q)$ 100 % of the time, and the throughput rate will be reduced. Let n_i be the maximum allowable number of buffer slots of buffer *i*, then the problem formulation is:

$$\text{Minimise } S_B(\mathbf{x}) := [-T_R(\mathbf{x}),\ W_P(\mathbf{x})] \tag{10}$$

$$\text{Subject to } \mathbf{x} \in \mathcal{X} \tag{11}$$

$$x_i \leq n_i,\ i = 1, \dots, m-1 \tag{12}$$

The values of the objectives are estimated with simulation models of the BAPs. Since the number of buffer spaces per buffer (niche) can be any integer from zero to infinity, the search must be contained. Also, the probability matrix *P* of the CEM requires a finite number of values. We suggest that a preliminary run of the simulation be done with infinite buffers, and estimate the maximum individual buffer occupations, then assign these to the n_i. These values are not guaranteed to be the absolute maximum, but they provide a starting point. Since the probability matrix *P* can become large if the number of buffers is large, together with a large number allowed per buffer, we modified the probability structure to be represented by truncated Poisson distributions. The Poisson distribution is generally defined on the integer range 0, 1, ..., ∞, but in this study, the range of each buffer *i* will be limited to $[0,\ n_i]$. The literature does not prescribe a specific discrete distribution for the CEM, and usually uniform discrete distributions are used on algorithm initialisation. The advantage of the truncated Poisson approach is, however, that it requires only a vector of Poisson rates to be maintained, with the number of elements equal to $m - 1$. For the BAP, we shall use the parameter vector $\mathbf{v} = \mathbf{\Lambda} = \{\lambda_1, \dots, \lambda_{\mathrm{m}-1}\}$. Similar to (6), the λ_{i} can be estimated with:

$$\hat{\lambda}_i = \frac{\sum_{k=1}^{N} I_{\{\hat{S}(X_k) \geq \gamma\}} X_{ki}}{\sum_{k=1}^{N} I_{\{\hat{S}(X_k) \geq \gamma\}}} \tag{13}$$

To determine the truncated Poisson distribution with rate λ_i and limit n_i, we first determine the cumulative Poisson value on the range $[0,\ n_i]$, as follows:

$$F_i(x, \lambda_i, n_i) = \sum_{x=0}^{n_i} \frac{e^{-\lambda_i} \lambda_i^x}{x!} \tag{14}$$

The truncated Poisson distribution on the given range $[0,\ n_i]$ is obtained by normalising the Poisson distribution, as follows:

$$f_i(x, \lambda_i, n_i) = \frac{e^{-\lambda_i} \lambda_i^x}{F_i(x, \lambda_i, n_i) \times x!} \tag{15}$$

Sampling from this distribution is simple and a buffer value x on the defined range is easily obtained. We state the proposed algorithm for multi-objective optimisation using the cross-entropy method (MOO CEM) as Algorithm 2, which will be applied to instances of the BAP using discrete-event simulation for evaluation of decision sets. The elements of $\boldsymbol{\Lambda}$ are arbitrarily initialised as $\lambda_i = n_i \cdot U(0,1)$ and we followed the histogram approach developed in [3] to guide the search. In this approach, each iteration of the algorithm forms an elite vector of solutions with Pareto ranking. Then, the values of each decision variable (DV) in this vector are grouped into a vector $\boldsymbol{C}_i = \{c_{i1}, c_{i2}, \dots, c_{ir_i}\}$ of subclasses that cover the definition range of the DV. We used $r_i = n_i/2 + 1$ subclasses, and the frequencies of values per DV are stored in vector $\boldsymbol{R}_i = \{\tau_{i1}, \tau_{i2}, \dots, \tau_{ir_i}\}$. During a given iteration, new values for the DVs are sampled from those subclasses, and the frequencies determine the number of values allowed in the sample on that range. The elite vector implies I in (13).

Algorithm 2: CEM multi-objective buffer allocation algorithm

```
Initialise $t = 0$, $\boldsymbol{\Lambda}_0 = n_i \cdot U(0,1)$, $i = 1,\dots, m-1$. Set $\boldsymbol{E} = \emptyset$.
For each DV $X_i$, set $r_i = n_i/2 + 1$ and create $\boldsymbol{C}_i$. Initialise $\boldsymbol{R}_i = 1$.
Generate a population $\boldsymbol{P}_N$ of $N$ solutions $\mathbf{x}_1,\dots,\mathbf{x}_N$ using $\boldsymbol{\Lambda}_t$ and (15) then
  evaluate each.
Rank the solutions and assign the non-dominated solutions to $\boldsymbol{E}$.
Repeat
  For Each DV $X_i$
    Determine the frequency elements of $\boldsymbol{R}_i$ by incrementing $\tau_{i\kappa}$ if
    $\boldsymbol{E}(i,\kappa) \in [c_{i\kappa}, c_{i\kappa+1})$, for $1 \le \kappa \le r_i$.
    Set $\boldsymbol{X}_i = \emptyset$.
    For $j := 1$ to $r_i$
      Let $\lambda_{ij} = U(0,1) \cdot (c_{ij+1} - c_{ij})$.
      Generate $\boldsymbol{Y} \sim \text{Poisson}(\lambda_{ij})$ on $[0, n_i]$ with $|\boldsymbol{Y}| = \tau_{ij}$, and append to $\boldsymbol{X}_i$.
    End For
  End For
  Evaluate each vector $c$ in the population $\boldsymbol{P}_N$ using DES and return
  estimations $\hat{T}_R(\mathbf{x}_c)$ and $\widehat{W}_P(\mathbf{x}_c)$.
  Form $\boldsymbol{E}' \leftarrow P_N \cup \boldsymbol{E}$ and set $\boldsymbol{E} = \emptyset$.
  Form $\boldsymbol{E}$ via Pareto ranking of $\boldsymbol{E}'$ and set $N_E$ = Number of rows in $\boldsymbol{E}$.
  Increment $t$.
  Estimate $\boldsymbol{\Lambda}_t = \sum_{j=1}^{N_E} \mathbf{E}(j,i)/N_E$, $N_E > 0$ and all DVs $X_j$ (see (13)).
  Update $\boldsymbol{\Lambda}_t$ as follows: $\boldsymbol{\Lambda}_t \leftarrow (1-\alpha)\boldsymbol{\Lambda}_{t-1} + \alpha\boldsymbol{\Lambda}_t$.
Until $\boldsymbol{\Lambda}_t$ has converged.
Return $\boldsymbol{E}$.
```

Convergence is checked using the development of the variance per DV in $\boldsymbol{E}$. As soon as consecutive values of each variance value do not differ by more than a preset deviation δ, say $\delta = 10^{-2}$, the algorithm terminates. The decision maker thus has to specify the population size N, the smoothing parameter α, the value for p_q (typically 0.95), and the termination value of δ. The ranking in Algorithm 2 is based on the Pareto ranking algorithm of [24]. Next, we describe the experimental setup to apply the multi-objective CEM to a number of BAPs.

4. EXPERIMENTAL SETUP

We propose the following methodology for multi-objective optimisation of the BAP with the CEM, assuming that a valid simulation model exists:

1. Determine the duration of the simulation transient period, as well as the required number of replications to obtain sufficient confidence intervals for the output parameters.
2. With the simulation model, estimate the maximum buffer sizes for the unrestricted case of infinite buffers.
3. Decide on values for p_q, N, α and δ. Typical values for these are p_q = 0.95, N = 20, $\alpha = 0.7$ and $\delta = 5 \times 10^{-2}$.
4. Execute Algorithm 2.

We studied five BAPs with various properties to investigate the CEM for multi-objective optimisation. Some problem parameters are summarised in Table I. The BAPs studied were:

1. BAP1: A linear manufacturing line with five machines, with exponential processing and failure rates. All machines fail according to a Poisson distribution with rate 20, while the downtime is exponentially distributed with a mean of two hours. This is a standard problem in the literature [23].
2. BAP2: A manufacturing line similar to BAP1, but with realistic real-world processing times: the times are offset by a minimum (a task cannot be executed in less time), and the distributions are truncated (a task has a finite duration). All the distributions are defined on the positive domain, since processing time cannot be negative. The lognormal distribution was found to be a satisfactory descriptor of the processing times.
3. BAP3: A manufacturing line similar to BAP2, but each workstation can rework a product just finished if it fails a quality inspection, or a product produced by the workstation can be completely rejected (Table I). The number of finished products leaving the system is thus generally fewer than the number entered.
4. BAP4: A 16-node network with failure-free machines has been developed and studied by [6]. We include this network to determine if the CEM can solve non-linear systems.
5. BAP5: This problem has the same configuration as BAP4, but we modify the problem by introducing operation-dependent machine failures as in BAP1–3. A schematic of BAP4 (and BAP5), the 16-node network with branches, is shown in Fig. 2.

Table I: Model parameters for BAP1–3.

	Processing times		Rejection rate BAP3	
Machine	BAP1	BAP2, 3	Quality	Final
1	1.0	1.090+logn(0.080, 0.21)/0.41	3 %	2 %
2	1.1	1.080+logn(0.075, 0.17)/0.48	2 %	2 %
3	1.2	1.071+logn(0.068, 0.17)/0.58	2 %	2 %
4	1.3	1.069+logn(0.059, 0.17)/0.67	2 %	2 %
5	1.4	1.058+logn(0.058, 0.17)/0.74	2 %	2 %

The discrete-event simulation models of the queuing networks were implemented in the Arena simulation package [25] while the optimisation algorithm was coded in Matlab R2007b. These two components were integrated to form the simulation-optimisation model. We followed the replication-deletion approach for the output analysis [26], and estimated the duration of the transient phase and the number of replications required for 95 % confidence with preliminary runs. An appropriate transient duration was found to be 800 time units, with total simulation duration of 5800 time units. Ten pseudo-independent replications were required. We chose the cross-entropy smoothing parameter $\alpha = 0.7$ while the population size was chosen to be 20 for BAP1–3 and 25 for BAP4–5. These numbers were determined via informal experimentation: large population sizes are preferred, but then the number of evaluations increases, while very small population sizes do not afford the algorithm sufficient information. The simulation-optimisation model was executed on an IBM Lenovo laptop with Intel Core i5 processor running at 2.40 GHz.

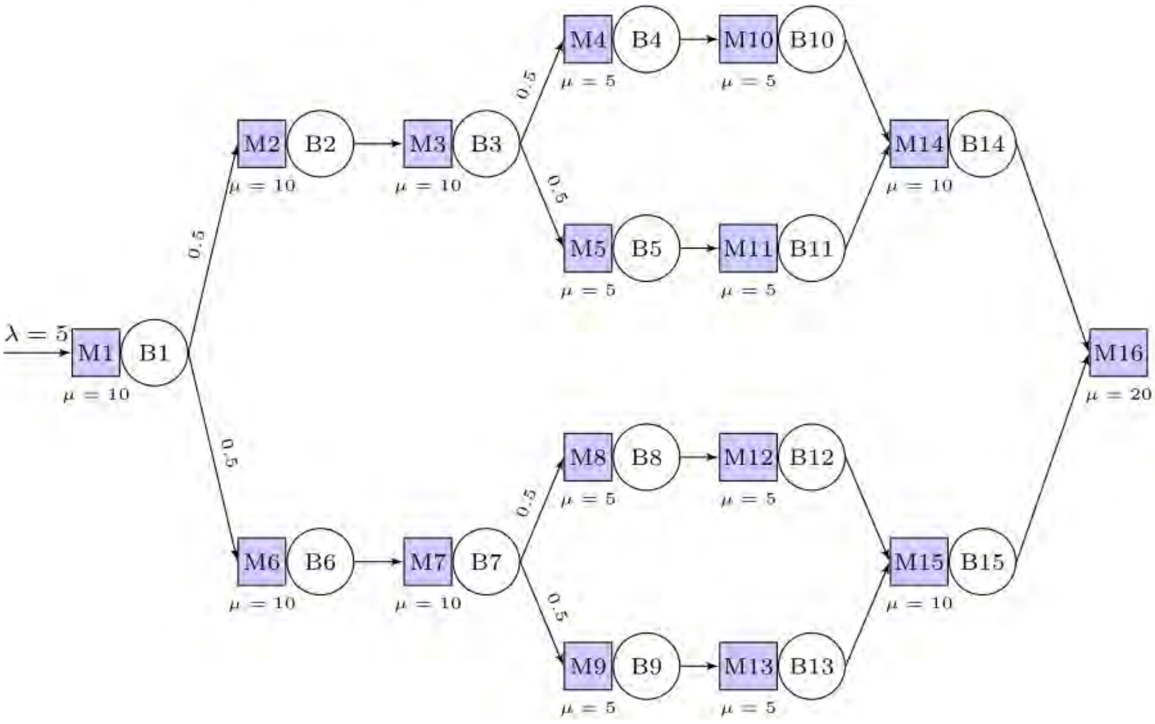

Figure 2: Sixteen-node network: BAP4 and BAP5 [6].

5. EXPERIMENTAL RESULTS

The results for the five BAPs are shown in Table II. The buffer sizes allowed are the maximum buffer occupation estimated via an independent experiment. The approximation set to the Pareto front (PF) for BAP1 is shown in Fig. 3, and some solutions obtained for BAP1 are briefly discussed. In Fig. 3, three labels (A, B and C) with solutions are shown. The values in the first pair of parentheses show the buffer sizes allocated between each machine. The second pair of parentheses contains the estimated values for the WIP percentile and the throughput rate. The label marked with an 'A' in Fig. 3 shows the buffer allocations for the lowest throughput rate, and a WIP percentile of 0. This value is zero because all buffers sizes are zero, and the worst T_R is 0.484. If the decision maker requires a higher T_R, the solution shown by label 'B' can be considered: the T_R is 0.768 while the 95th WIP percentile is 12.8. The near-maximum throughput rate is 0.89, and the WIP percentile is 33.7, shown by label 'C'. If the decision maker requires the WIP percentile to be no more than 10, it follows from the detail solution set (not shown) that an allocation of (4, 3, 2, 5) must be implemented, with an expected T_R of 0.727.

Similar plots for BAP2, BAP3, BAP4 and BAP5 are shown in Figs. 4 to 7 respectively. The extreme minimums and maximums were estimated for each problem, i.e. the T_R for the case of all buffers set to zero, and for all buffers infinitely large. The latter case is indicated with a solid line. It is encouraging to see that for each problem, both values of these extremes were found by the algorithm.

Table II: Simulated results for the five BAPs, for given maximum buffer sizes.

Problem	Buffer size B_i allowed	Problem size	Number of evaluations[1]	Size estimated Pareto set	Max. estimated T_R
BAP1	35, 30, 25, 25	656 250	340	55	0.905
BAP2	30, 25, 20, 20	300 000	280	38	0.719
BAP3	25, 20, 15, 15	112 500	520	44	0.620
BAP4	15 for all i	$> 10^{17}$	1 050	34	5.023
BAP5	15 for all i	$> 10^{17}$	400	59	4.043

[1] To obtain estimated Pareto set

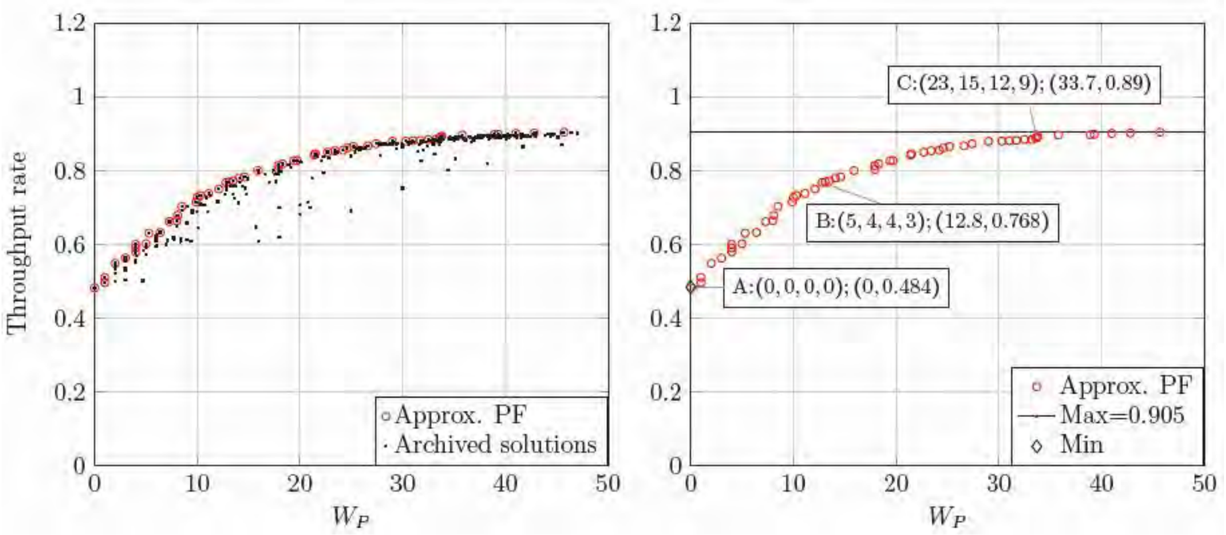

Figure 3: Approximate Pareto front and archive for BAP1.

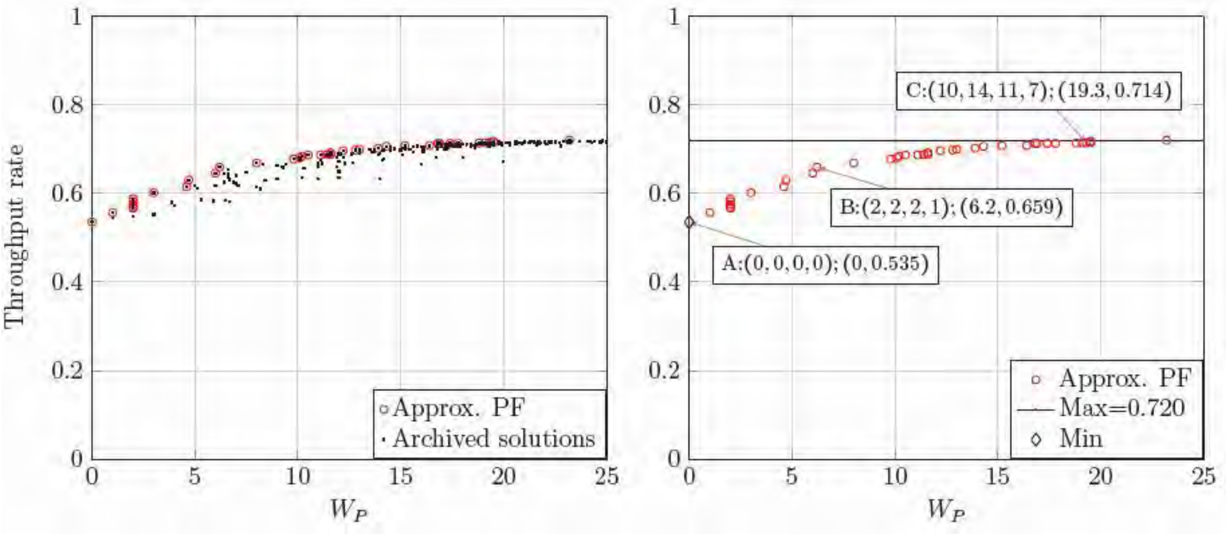

Figure 4: Approximate Pareto front and archive for BAP2.

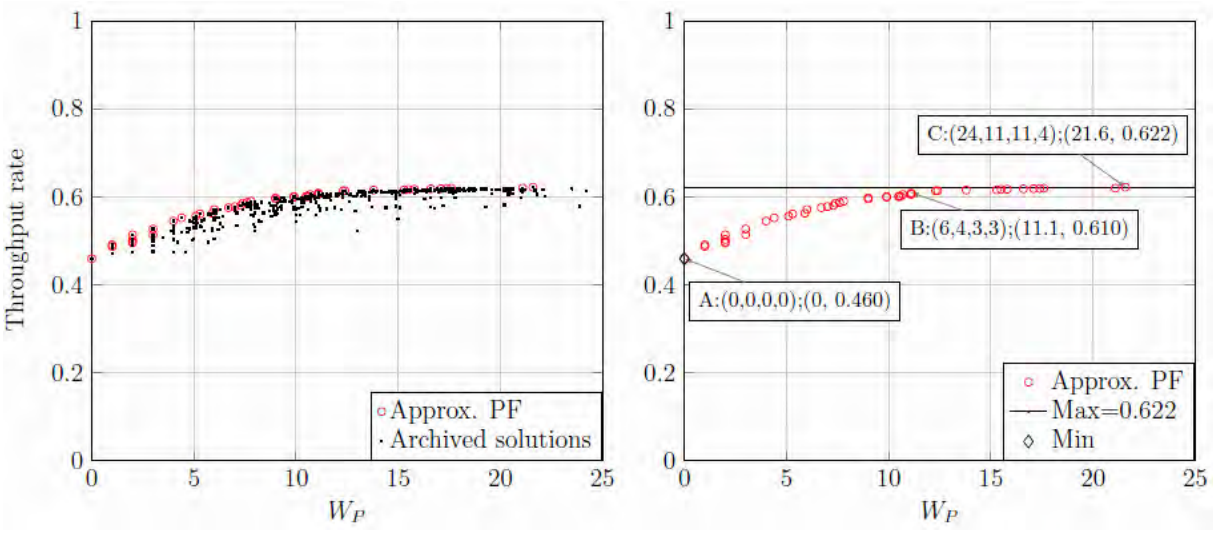

Figure 5: Approximate Pareto front and archive for BAP3.

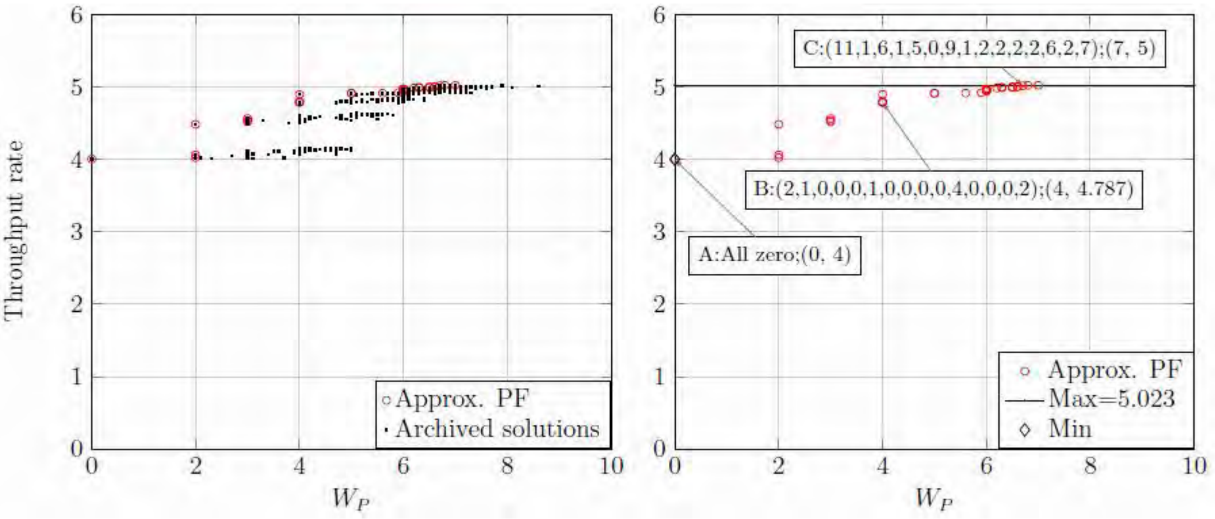

Figure 6: Approximate Pareto front and archive for BAP4.

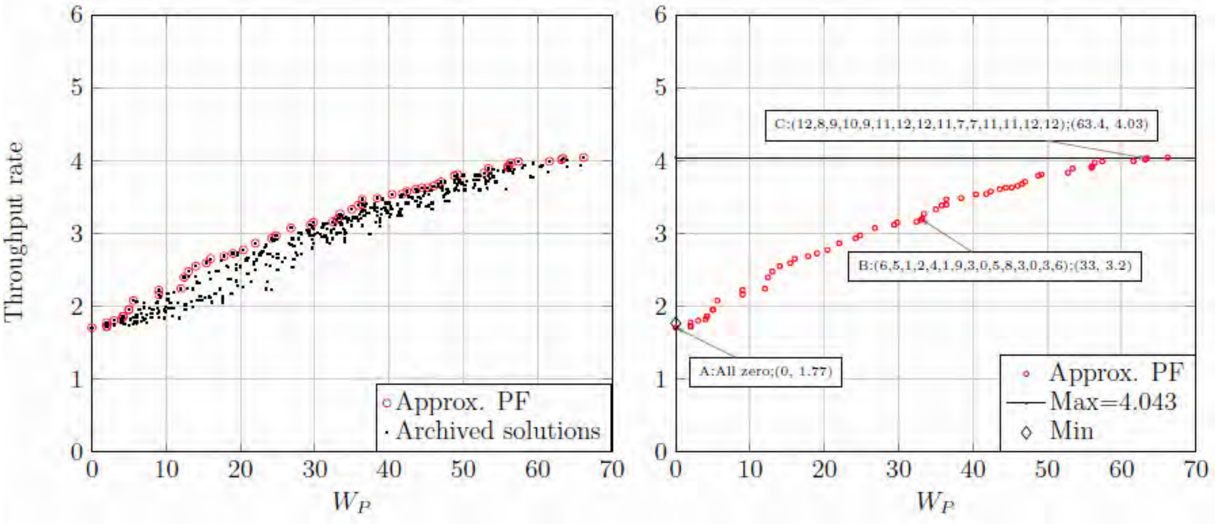

Figure 7: Approximate Pareto front and archive for BAP5.

We have no proof that these Pareto sets contain the complete sets of true solutions, but the graphic shapes of the fronts found in the experiments make sense and are in line with the findings of others, for example [5]. Furthermore, fewer evaluations are needed than the numbers reported by other researchers. [5] applied the well-known non-dominated sorting genetic algorithm (NSGA-II) of [27] to the BAP and used a population size of 80 with the maximum number of generations equal to 1000, thus allowing up to 80000 evaluations. For the 16-node problem (BAP4), their minimum average number of generations in the various experiments is approximately 250, which required 2000 evaluations. We thus claim that the Pareto fronts of multi-objective problems having many solutions can be estimated via fewer evaluations when using the MOO CEM.

6. SUMMARY AND CONCLUSIONS

We presented a multi-objective optimisation algorithm (MOO CEM) for problems with discrete solution spaces using the CEM on five different queuing networks. It is often only possible to evaluate proposed solutions of discrete, dynamic stochastic processes with computer simulation, which can be time consuming, especially when the solution space is large. The MOO CEM found good solutions (approximate Pareto sets) for the various BAPs using fairly few computationally expensive simulations for solution evaluations, which was our primary research objective. The MOO CEM algorithm is simple and fast, and its execution time is dominated by the runtime of the simulation models.

Our approach is pragmatic and aimed at practical solutions: the Matlab code for the MOO CEM is simple to use, and the decision maker has to provide a valid simulation model. We made no assumptions regarding the time and failure distributions of the simulated processes. We showed that the CEM as a multi-objective optimiser can minimise the total buffer space by appropriately allocating space to each buffer while maximising throughput rate. A possible drawback of our approach is execution time, as it still took between 10 and 30 minutes to generate solutions for each network. However, when designing complex, expensive systems, time durations of this magnitude are negligible.

We suggest further research be done by applying the proposed MOO CEM algorithm to other types of queuing networks that have more than two objectives. Also, combining the MOO CEM algorithm, which is population-based, with a local search method might increase the convergence speed. The performance of the proposed approach should also be compared to that of commercial packages like OptQuest (www.optek.com).

In engineering, optimisation or near-optimisation of parts of the whole is not sufficient. In practical applications of the proposed method, the engineering design team should also consider the parts of the whole in an integrated manner. Specific resource optimisation is required, for example in a machining operation, which is one of several operations in a typical manufacturing queuing network. In [28], for example, optimisation of a steel-turning process is demonstrated where the cutting speed, feed rate and depth of cut are decision variables, and surface roughness, cutting time and removal rate are the multi-objectives. To achieve the optimisation, the Global Criterion Method and Principal Components Analysis are combined. Once a queuing network is operational, it must be further optimised through its management principles, for example how the workload is scheduled. This in itself is a difficult task, and typical work in that domain has been done by [29]. They solve MOO problems using a hybrid of fuzzy logic and particle swarm optimisation.

Integration of these different aspects of queuing networks is a continuous challenge.

REFERENCES

[1] Rubinstein, R. Y. (1997). Optimization of computer simulation models with rare events, *European Journal of Operations Research*, Vol. 99, No. 1, 89-112, doi:10.1016/S0377-2217(96)00385-2

[2] Rubinstein, R. Y.; Kroese, D. P. (2004). *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning*, Springer, New York

[3] Bekker, J.; Aldrich, C. (2010). The cross-entropy method in multi-objective optimisation: An assessment, *European Journal of Operational Research*, Vol. 211, No. 1, 112-121, doi:10.1016/j.ejor.2010.10.028

[4] Coello Coello, C. A.; Lamont, G. B.; Van Veldhuizen, D. A. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed., Springer, New York

[5] Cruz, F. R. B.; Van Woensel, T.; MacGregor Smith, J. (2010). Buffer and throughput trade-offs in M|G|1|K queuing networks: A bi-criteria approach, *International Journal of Production Economics*, Vol. 125, No. 2, 224-234, doi:10.1016/j.ijpe.2010.02.017

[6] Cruz, F. R. B.; Duarte, A. R.; Van Woensel, T. (2008). Buffer allocation in general single-server queuing networks, *Computers & Operations Research*, Vol. 35, No. 11, 3581-3598, doi:10.1016/j.cor.2007.03.004

[7] Papadopoulos, H. T.; Heavey, C. (1996). Queuing theory in manufacturing systems analysis and design: A classification of models for production and transfer lines, *European Journal of Operational Research*, Vol. 92, No. 1, 1-27, doi:10.1016/0377-2217(95)00378-9

[8] Vouros, G. A.; Papadopoulos, H. T. (1998). Buffer allocation in unreliable production lines using a knowledge-based system, *Computers & Operations Research*, Vol. 25, No. 12, 1055-1067, doi:10.1016/S0305-0548(98)00034-3

[9] Yang, S.; Wu, C.; Hu, S. J. (2000). Modeling and analysis of multi-stage transfer lines with unreliable machines and finite buffers, *Annals of Operations Research*, Vol. 93, No. 1-4, 405-421, doi:10.1023/A:1018944411591

[10] Hillier, F. S.; So, K. C. (1991). The effect of machine breakdowns and interstage storage on the performance of production line systems, *International Journal of Production Research*, Vol. 29, No. 10, 2043-2055, doi:10.1080/00207549108948066

[11] Gershwin, S. B.; Schor, J. E. (2000). Efficient algorithms for buffer space allocation, *Annals of Operations Research*, Vol. 93, No. 1, 117-144, doi:10.1023/A:1018988226612

[12] Heavey, C.; Papadopoulos, H. T.; Browne, J. (1993). The throughput rate of multi-station unreliable production lines, *European Journal of Operational Research*, Vol. 68, No. 1, 69-89, doi:10.1016/0377-2217(93)90077-Z

[13] MacGregor Smith, J.; Cruz, F. R. B. (2005). The buffer allocation problem for general finite buffer queuing networks, *IIE Transactions*, Vol. 37, No. 4, 343-365, doi:10.1080/07408170590916986

[14] Can, B.; Heavey, C. (2011). Comparison of experimental designs for simulation-based symbolic regression of manufacturing systems, *Computers & Industrial Engineering*, Vol. 61, No. 3, 447-462, doi:10.1016/j.cie.2011.03.012

[15] Cao, Y.; Subramaniam, V.; Chen, C. (2012). Performance evaluation and enhancement of multistage manufacturing systems with rework loops, *Computers & Industrial Engineering*, Vol. 62, No. 1, 161-176, doi:10.1016/j.cie.2011.09.004

[16] Lutz, C. M.; Davis, K. R., Sun, M. (1998). Determining buffer location and size in production lines using tabu search, *European Journal of Operational Research*, Vol. 106, No. 2-3, 301-316, doi:10.1016/S0377-2217(97)00276-2

[17] Dolgui, A.; Eremeev, A.; Kolokolov, A.; Sigaev, V. (2002). A genetic algorithm for the allocation of buffer storage capacities in a production line with unreliable machines, *Journal of Mathematical Modelling and Algorithms*, Vol. 1, No. 2, 89-104, doi:10.1023/A:1016560109076

[18] Malakooti, B. (1991). A multiple criteria decision making approach for the assembly line balancing problem, *International Journal of Production Research*, Vol. 29, No. 10, 1979-2001, doi:10.1080/00207549108948063

[19] Battini, D.; Persona, A.; Regattieri, A. (2009). Buffer size design linked to reliability performance: A simulative study, *Computers & Industrial Engineering*, Vol. 56, No. 4, 1633-1641, doi:10.1016/j.cie.2008.10.020

[20] CEM Website, from *http://www.cemethod.org*, accessed on 02-04-2012

[21] Kroese, D. P.; Rubinstein, R. Y. (Eds.), (2005). Special issue: The Cross-Entropy Method for Combinatorial Optimization, Rare Event Simulation and Neural Computation, *Annals of Operations Research*, Vol. 134, No. 1

[22] Kroese, D. P. (2010). Cross-entropy method, Cochran, J. J.; Cox, L. A.; Keskinocak, P.; Kharoufeh, J. P.; Cole Smith, J. (Eds.), *Wiley Encyclopaedia of Operations Research and Management Science*, John Wiley & Sons, Inc., New York

[23] Alon, G.; Kroese, D. P.; Raviv, T.; Rubinstein, R. Y. (2005). Application of the cross-entropy method to the buffer allocation problem in a simulation-based environment, *Annals of Operations Research*, Vol. 134, No. 1, 137-151, doi:10.1007/s10479-005-5728-8

[24] Goldberg, D. E. (1989). *Genetic Algorithms in search, optimization and machine learning*, Addison-Wesley Publishing Company, Boston

[25] Kelton, W. D.; Sadowski, R.; Sturrock, D. (2007*). Simulation with Arena*, 3rd ed., McGraw-Hill, New York

[26] Law, A. M.; Kelton, W. D. (2000). *Simulation modeling and analysis*, 3rd ed., McGraw-Hill, New York

[27] Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, 182-197, doi:10.1109/4235.996017

[28] Gomes, J. H. F.; Salgado Jr., A. R.; De Paiva, A. P.; Ferreira, J. R.; Da Costa, S. C.; Balestrassi, P. P. (2012). Global Criterion Method Based on Principal Components to the Optimization of Manufacturing Processes with Multiple Responses, *Strojniski vestnik – Journal of Mechanical Engineering*, Vol. 58, No. 5, 345-353, doi:10.5545/sv-jme.2011.136

[29] Galzina, V.; Lujic, R.; Saric, T. (2012). Adaptive fuzzy particle swarm optimization for flow-shop scheduling problem, *Technical Gazette*, Vol. 19, No. 1, 151-157