OPTIMISATION OF SERVICE CAPACITY BASED ON QUEUEING THEORY

MICHAL DORDA, DUSAN TEICHMANN, VOJTECH GRAF

VSB – Technical University of Ostrava, Faculty of Mechanical Engineering, Institute of Transport, Ostrava, the Czech Republic

DOI: 10.17973/MMSJ.2019_10_201889

e-mail: michal.dorda@vsb.cz

Queueing theory is a mathematical tool which can be applied for capacity planning and optimisation of production, manufacturing or logistics systems. One of the possible applications of queueing theory is service capacity optimisation. Let us consider that an engineering company operates *m* homogeneous machines. We assume that the machines are successively operating and down and times between failures and times to repair are exponentially distributed. The broken-down machines are repaired by *n* repairmen; we assume that *n* < *m*. In the article a mathematical model of the problem is presented; the model can be used for optimisation of the number of the repairmen with respect to costs of the system. Results obtained by the mathematical model are compared with simulation results; a simulation model of the problem is based on coloured Petri nets.

KEYWORDS

Service capacity, optimisation, queueing theory, simulation, coloured Petri nets, Matlab, CPN Tools

1 INTRODUCTION

Queueing theory is focused on modelling systems which are designated for providing service to customers entering the system. We can encounter queueing systems in many sectors – most often queueing systems are applied in manufacturing, logistics, transport, telecommunications and informatics. Queueing theory offers tools which enable the prediction of the behaviour of such systems – for example we can estimate the mean number of customers waiting in the queue or their mean waiting time with respect to the number of service facilities (known as servers) and so on. Based on such pieces of information, we can design the service systems optimally – we can estimate the optimal number of servers which should be in the system in order to minimise economic losses.

There are many mathematical models of queueing systems that have been developed. The models differ in their assumptions – arrival patterns of the customers, the number of the servers and their mutual arrangement, the queue capacity (if customers are allowed to wait in the system) and service discipline (for example First in first out or Last in first out, etc.) and so on. A summary of basic queueing models can be found in many available sources – for example [Pesko 1999], [Adan 2002] and [Cooper 1981].

In the article we present an example of a practical application of queueing theory in the area of mechanical engineering – the optimisation of a service capacity. Let us consider that a manufacturing company operates some homogeneous machines which produce some products. We assume that the machines are successively operating and down due to random failures. The broken-down machines are repaired by a group of repairmen which the company employs in order to keep the machines in a failure-free state.

In the article, a mathematical model of the problem is presented. The model can be used for the optimisation of the number of repairmen with respect to the system costs. The results obtained by the mathematical model are compared with simulation results; a simulation model of the problem is based on coloured Petri nets.

2 PROBLEM FORMULATION AND MATHEMATICAL MODEL

Let us consider a manufacturing company which operates a group of homogeneous machines. Let us assume that the number of the machines is equal to *m*. The machines are successively operating (failure-free) and down (out of order). In the failure-free state, each machine produces some products and, therefore, brings a profit to the company. On the other hand, when the machine is broken-down it cannot produce anything until is repaired. That means that each failure of the machine brings a loss in production of the company.

With two possible states of the machines, two continuous random variables are associated. The first random variable models the time between failures (TBF) of the machines and the second random variable corresponds to the time to repair (TTR). We assume in the model that both random variables are exponentially distributed. The exponential random variable can be defined by the probability density function in the following form:

$$f(t) = \lambda \cdot e^{-\lambda \cdot t} \tag{1}$$

for $\lambda > 0, t > 0$, where λ is the parameter of the exponential distribution of probability. It holds for the exponential distribution of probability that its mean value *ET* is the reciprocal value of its parameter:

$$ET = \frac{1}{\lambda}. \tag{2}$$

To differentiate between the random variables, we assume that the times between failures are exponentially distributed with the parameter λ, whilst the times to repair are exponentially distributed with the parameter μ. Based on formula (2) we can write the mean time between failures (MTBF) as follows:

$$MTBF = \frac{1}{\lambda} \tag{3}$$

and the mean time to repair (MTTR):

$$MTTR = \frac{1}{\mu}. \tag{4}$$

The service company employs a group of repairmen who repair the broken-down machines. Let us assume that the number of the repairmen equals *n*, where *n* < *m*. Let us assume that the number of the broken-down machines at time *t* equals *k*, where $k \in \{0, 1, \dots, m-1, m\}$. The values $k(t)$ define the individual states of the queueing system at time *t* we use for modelling our problem.

Based on the assumption that two possible events may occur when a machine has broken down at time t:

- if it holds that $k(t) < n$, then at least one repairman is idle and any idle repairman starts to repair the machine immediately.
- if it holds that $n \le k(t) < m$, then no repairman is idle (all the repairmen are busy) and $k(t) - n$ machines are already waiting in the queue to be repaired; the machine which has just broken down must join the queue.

The problem described in the previous text can be modelled by a finite-source queueing model. A bibliography of finite-source queueing systems, including their possible applications, can be found in [Sztrik 2005].

When modelling queueing systems, we can apply two possible approaches. The first one studies the behaviour of the queueing system with respect to time, known as a transient analysis. The transient analysis gives information on the results of the model (probabilities of the individual states, performance measures such as the mean number of customers waiting in the queue or the mean waiting time of the customers and so on) depending on time. The transient analysis is quite difficult even for simple queueing systems. However, if we want to obtain time dependences of the results for more complex queueing models, we can employ simulation methods.

The second way is to study a queueing system in steady-state, that means for $t \to \infty$. The steady-state analysis gives results that are not dependent on time. In the article, we carry out a steady-state analysis of the studied problem. Note that the results of the steady-state analysis can also be compared with simulation results.

To model the system in steady-state, we illustrate the individual states k, where $k \in \{0,1,\ldots,m-1,m\}$, and the possible transitions between them graphically in a state transition diagram (see Figure 1). The vertices of the diagram represent the individual states (the number of the broken-down machines) and the directed edges indicate the transitions with the corresponding rates.

In steady-state, we can apply the global balance principle which states that for each subset X of the set S of all the states of the system, the flow out of set X is equal to the flow into the set X – see for example [Adan 2002].

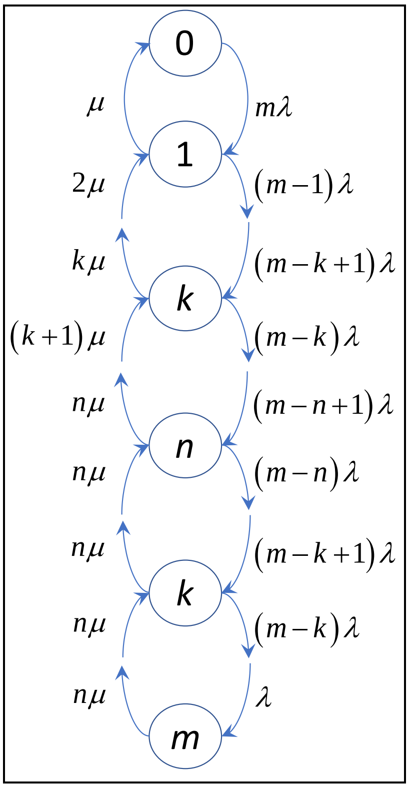


Figure 1. The state transition diagram of the system

Table 1 present a summary of mathematical notation which is used in the mathematical model presented in the following section of the article.

Symbol	Its meaning
λ (or "lambda" in the Matlab code)	The mean arrival rate of the Poisson input process [customers per unit time]
μ (or "mi" in the Matlab code)	The mean service rate [customers per unit time]
n	The number of the repairmen [-]
m	The number of the machines [-]
k	The number of the broken-down machines [-]
P_k	The probability that *k* machines are broken-down [-]
ES	The mean number of the machines under repair [-]
EL	The mean number of the machines waiting in the queue to be repaired [-]
EK	The mean number of the broken-down machines [-]
ER	The mean number of the failure-free machines [-]
EW	The mean time a machine must wait in the queue to be repaired [unit time]
ET	The mean time a machine must spend in the system [unit time]

Table 1. Summary of notation used in the model

Applying the principle, we obtain the following linear equation system:

$$m\lambda \cdot P_0 = \mu \cdot P_1, \tag{5}$$

$$\left[(m-1)\lambda+\mu\right]\cdot P_1 = m\lambda \cdot P_0 + 2\mu \cdot P_2, \tag{6}$$

$$\left[(m-k)\lambda+k\mu\right]\cdot P_k = (m-k+1)\lambda \cdot P_{k-1} + $$

$$+(k+1)\mu \cdot P_{k+1} \text{ for } k=2,\ldots,n-1, \tag{7}$$

$$\left[(m-n)\lambda+n\mu\right]\cdot P_n = (m-n+1)\lambda \cdot P_{n-1} + n\mu \cdot P_{n+1}, \tag{8}$$

$$\left[(m-k)\lambda+n\mu\right]\cdot P_k = (m-k+1)\lambda \cdot P_{k-1} + $$

$$+n\mu \cdot P_{k+1} \text{ for } k=n+1,\ldots,m-1, \tag{9}$$

$$n\mu \cdot P_m = \lambda \cdot P_{m-1}. \tag{10}$$

Note that equation (5) was obtained for $X=\{0\}$, equation (6) for $X=\{1\}$, equation (7) for $X=\{k\}$, where $k=2,\ldots,n-1$, equation (8) for $X=\{n\}$, equation (9) for $X=\{k\}$, where $k=n+1,\ldots,m-1$ and the last equation (10) for $X=\{m\}$.

The linear equation system (5) – (10) consists of $m+1$ equations with $m+1$ unknown steady-state probabilities P_k, which express the probability that k machines are broken-down (or the fraction of time in which k machines are broken-down). However, one equation of the system is a linear combination of the others. To obtain a unique solution of the system we must omit an equation, for example equation (10), and the equation must be replaced by the normalisation condition which states that the sum of all the probabilities is equal to 1:

$$\sum_{k=0}^{m} P_k = 1 . \qquad (11)$$

If we do not need to express closed-form formulas for the individual steady-state probabilities, then equation system (5) – (9) and (11) can be solved numerically using a suitable software tool, for example Matlab.

To indicate how the closed-form formulas for the steady-state probabilities of the individual system states can be derived, let us assume that $X = \{0,1,\ldots,k-1\}$ for $k < n$. Then we can write that:

$$(m-k+1)\lambda \cdot P_{k-1} = k\mu \cdot P_k . \qquad (12)$$

It can be easily proved that formula (12) holds for $k = 1,\ldots,n$.

Now we assume that $X = \{0,1,\ldots,n,\ldots k-1\}$ for $k > n$. Then the global balance principle yields:

$$(m-k+1)\lambda \cdot P_{k-1} = n\mu \cdot P_k . \qquad (13)$$

Formula (13) holds for $k = n,\ldots,m$.

From formula (12) we can directly derive that:

$$P_k = \frac{(m-k+1)}{k} \cdot \frac{\lambda}{\mu} \cdot P_{k-1} \text{ for } k = 1,\ldots,n \qquad (14)$$

and from formula (13) we easily obtain:

$$P_k = \frac{(m-k+1)}{n} \cdot \frac{\lambda}{\mu} \cdot P_{k-1} \text{ for } k = n,\ldots,m . \qquad (15)$$

Based on formula (14) we can recursively derive that:

$$P_k = \binom{m}{k} \cdot \left(\frac{\lambda}{\mu}\right)^k \cdot P_0 \text{ for } k = 1,\ldots,n \qquad (16)$$

and formula (15) yields:

$$P_k = \frac{(m-k+1)\cdot(m-k+2)\cdots(m-n)}{n^{k-n}} \cdot \binom{m}{n} \cdot \left(\frac{\lambda}{\mu}\right)^k \cdot P_0 \text{ for}$$

$$k = n,\ldots,m . \qquad (17)$$

To apply formulas (16) and (17), the value of P_0 must be known. A formula for calculating it can be derived from the normalisation condition (11). We can write that:

$$P_0 = \left[\sum_{k=0}^{n} \binom{m}{k} \cdot \left(\frac{\lambda}{\mu}\right)^k + \right.$$

$$\left. + \sum_{k=n+1}^{m} \frac{(m-k+1)\cdot(m-k+2)\cdots(m-n)}{n^{k-n}} \cdot \binom{m}{n} \cdot \left(\frac{\lambda}{\mu}\right)^k \right]^{-1} . \qquad (18)$$

More detailed information about the process of derivation of closed-form formulas (16) – (18) can be found, for example, in [Kluvanek 1981].

The steady-state probability that no machine is broken-down (or all the machines are working) equals P_0. The steady-state probability that at least one machine is broken-down equals $1-P_0$. The steady-state probability that at least *r* machines are working (no more than $r-1$ machines are broken-down) is $\sum_{k=0}^{r-1} P_k = 1 - \sum_{k=r}^{m} P_k$, where $r < m$. The steady-state probability that all the repairmen are busy equals $\sum_{k=n}^{m} P_k$.

Based on the steady-state probabilities, we can calculate the performance measures, which give information about the steady-state behaviour of the modelled system. To express formulas for the individual performance measures, we apply the well-known formula for the mean value of a discrete random variable and Little's law, which expresses the relationship between the mean number of the customers found in a queue (or in a system) and the mean waiting time of the customers (or the mean time spent in the system) – see for example [Adan 2002].

For the mean number of machines under repair *ES,* it holds that:

$$ES = \sum_{k=1}^{n-1} k \cdot P_k + n \cdot \sum_{k=n}^{m} P_k . \qquad (19)$$

The mean number of machines waiting in the queue to be repaired *EL* is equal to:

$$EL = \sum_{k=n+1}^{m} k \cdot P_k . \qquad (20)$$

The mean number of broken-down machines *EK* is the sum of *ES* and *EL* and therefore we can write:

$$EK = ES + EL = \sum_{k=1}^{m} k \cdot P_k . \qquad (21)$$

The mean number of the failure-free machines *ER* can be calculated using the formula:

$$ER = m - EK . \qquad (22)$$

The mean time a machine must wait in the queue to be repaired *EW* can be calculated based on Little's law and the value of *EL*:

$$EW = \frac{EL}{\lambda \cdot ER} . \qquad (23)$$

and finally, the mean time spent in the system *ET* is the sum of the mean waiting time and the mean time to repair (MTTR):

$$ET = \frac{EL}{\lambda \cdot ER} + \frac{1}{\mu} . \qquad (24)$$

A Matlab function which solves the linear equation system (5) – (9) and (11) and calculates the performance measures based on formulas (19) – (24) was created – see the code below. Note that "*lambda*" in the function corresponds to λ and "*mi*" to μ:

```
function [ES,EL,EK,ER,EW,ET] = MM_code(lambda,mi,n,m)
P=sparse(m+1,m+1);
P(1,1)=-(m*lambda); P(1,2)=mi;
if n>1;
for k=2:n;
   P(k,k-1)=(m-k+2)*lambda; P(k,k)=-((m-k+1)*lambda+(k-
1)*mi);
   P(k,k+1)=k*mi;
end
end
for k=n+1:m;
   P(k,k-1)=(m-k+2)*lambda; P(k,k)=-((m-k+1)*lambda+n*mi);
   P(k,k+1)=n*mi;
end
if m>n;
for k=1:m+1;
   P(m+1,k)=1;
end
end
A=sparse(m+1,1);
A(m+1,1)=1;
X=sparse(m+1,1);
X=P\A;
ES=0;
for k=1:n+1;
   ES=ES+(k-1)*X(k,1);
end
for k=n+2:m+1;
   ES=ES+n*X(k,1);
end
EL=0;
for k=n+2:m+1;
   EL=EL+(k-1-n)*X(k,1);
```

```
end
EK=0;
for k=2:m+1;
   EK=EK+(k-1)*X(k,1);
end
ES, EL, EK, ER=m-EK, EW=EL/(lambda*ER)
ET=EW+(1/mi)
end
```

3 SIMULATION MODEL

To validate the results we obtain by the solution of the mathematical model, we created a simulation model of the same queueing system. To create the simulation model, we employed coloured Petri nets. Detailed information about them can be found for example in [Jensen 2013].

The simulation model was created using the software tool CPN Tools. Each Petri net in CPN Tools consists of two parts:

- a graph of the coloured Petri net which is formed by places depicted as ellipses or circles, transitions depicted as rectangles and oriented arcs that connect places with transitions and vice versa.
- a text part called inscription which comprises, for example, arc inscriptions or place inscriptions.

Let us describe briefly the simulation model and its individual elements. Figure 2 presents the simulation model; the depicted model corresponds to an example which will be presented in the next section. The model consists of 7 places and 4 transitions with the following meaning:

- the place "*Initialisation*" contains the number of tokens in the initial marking which corresponds to the value of *m* – the tokens represent the individual machines. At simulation time 0, the transition "*Generating first TBF*" is fired *m* times in order to generate the first value of the time between failures for each machine. This is done through the arc expression of the arc which connects the transition with the place "*Working machines*". Thanks to the arc expression, the timestamp of each token modelling the machines is increased by an exponentially distributed value with the mean value of 100.0 hours in our case.
- The place "*Working machines*" contains the tokens which model the machines that are working (failure-free) – that means the number of the tokens found in the place corresponds to the number of the working machines. Each token found in the place has the value of its timestamp, which is greater than the actual simulation time. When the timestamp of any token is equal to the simulation time, the transition "*Machine is broken down*" is fired and the token is removed from the place.
- The place "*Queue*" contains only a single token in each marking of the net. The token represents a list of integer values, where the individual integer values are equal to points in time when the individual machines that are currently waiting to be repaired have broken-down and entered the system. It is obvious that the number of the integer values in the list corresponds to the number of the machines that are waiting in the queue.
- Because the place "*Queue*" contains only a single token in each marking, the place cannot be used to estimate the mean number of the machines waiting to be repaired. In order to get this value, the auxiliary place "*Machines waiting for repair*" was added. The number of the tokens found in the place corresponds to the number of the broken-down machines waiting for repair in each marking of the presented Petri net.
- The place "*Broken-down machines*" has a similar meaning to the place "*Machines waiting for repair*", the place was created to monitor the number of the machines that are out of order.
- If a repairman is idle – that means a token is found in the place "*Idle repairmen*" – then the repair of a machine waiting in the queue can start, this is realised through firing the transition "*Begin repair*". Note that the initial marking of the place "*Idle repairmen*" corresponds to the parameter *n* of the mathematical model. When the transition is fired, the value corresponding to the machine which is first in the queue is removed from the list of the waiting machines, which is represented by the token found in the place "*Queue*" (and a token is also removed from the place "*Machines waiting for repair*" of course) and the variable "*machine*" is bound to the value.
- The tokens that are found in the place "*Machines under

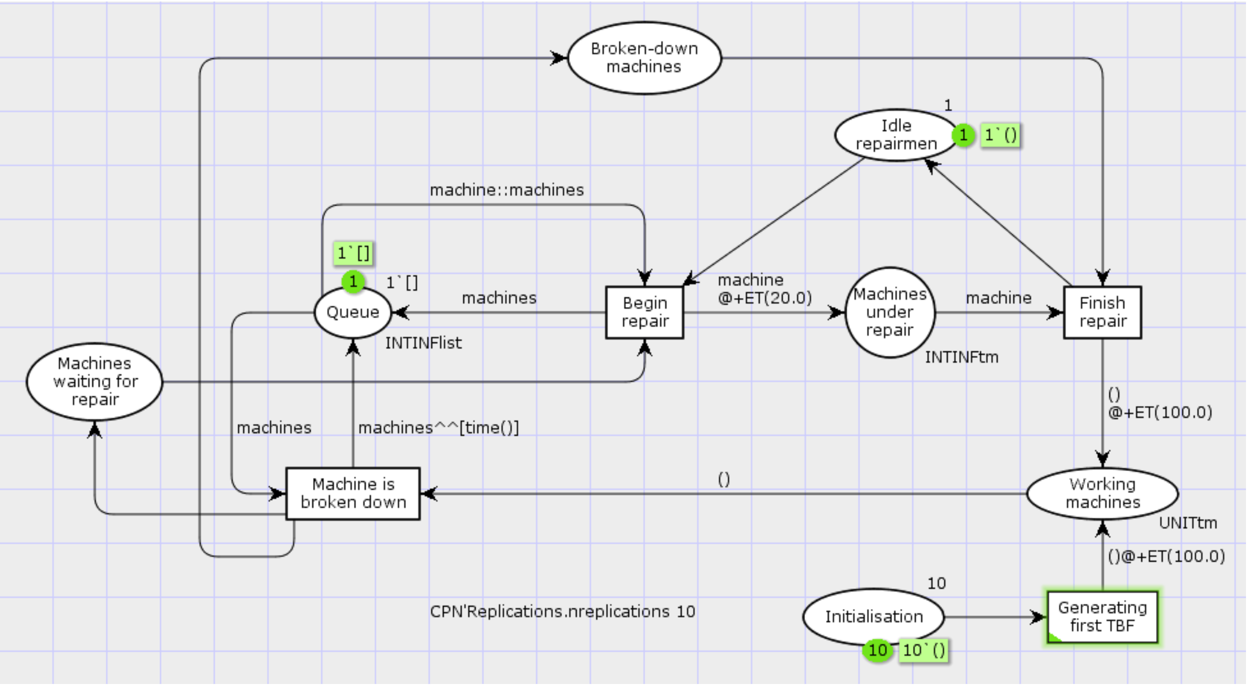


Figure 2. The simulation model in CPN Tools

repair" model the machines that are being repaired. Because it is necessary that the tokens must spend exponentially distributed times in the place to model their repair times, in the arc expression of the arc connecting the transition "*Begin repair*" to the place "*Machines under repair*" an increase of the timestamp is declared.

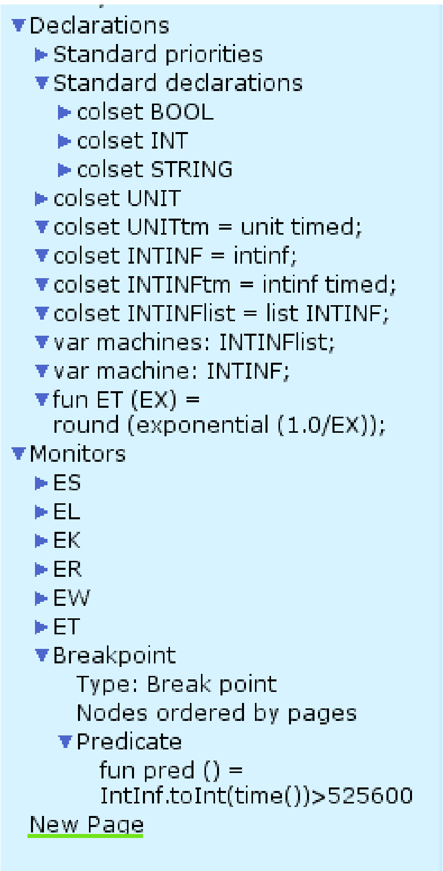
- When the timestamp of any token found in the place "*Machines under repair*" equals the simulation time, then the transition "*Finish repair*" is fired – this means the token which models the machine is removed from the place "*Machines under repair*", the repairman who has finished his/her job is idle now (a token is added in the place "*Idle repairmen*") and the token modelling the machine is moved into the place "*Working machines*", where it spends an exponentially distributed time because its timestamp is increased when moving along the arc connecting the transition "*Finish repair*" to the place "*Working machines*".

To estimate the individual performance measures by simulation experiments with the presented model, some monitoring functions must be defined:

- the monitoring function "*ES*" estimates the mean number of the machines under repair; the function is associated with the place "*Machines under repair*".
- to estimate the mean number of the machines waiting in the queue to be repaired, the monitoring function "*EL*" was defined. The function is associated with the place "*Machines waiting for repair*".
- The mean number of the broken-down machines is estimated by the function "*EK*" by monitoring individual markings of the place "*Broken-down machines*".
- To estimate the mean number of the failure-free machines, the monitoring function "*ER*" was created and the function is associated with the place "*Working machines*".
- The mean time a machine must wait in the queue to be repaired can be estimated thanks to the monitoring function "*EW*", which is associated with the transition "*Begin repair*". Whenever the transition is fired, the value of the variable "*machine*" (the value represents the simulation time when the machine has entered the queue) is subtracted from the current simulation time and this value represents the waiting time of the machine.
- the mean time spent in the system can be estimated by the monitoring function "*ET*", which is connected with the transition "*Finish repair*". The time a machine spends in the system is calculated in the same way as the waiting time.

Figure 3 shows a summary of the monitoring functions as defined in the model. In addition, one can see colour sets (in our case the colour set "*UNIT*", "*UNITtm*", "*INTINF*" and "*INTINFlist*" are used, where the colour set "*UNIT*" is predefined by the programmers of CPN Tools and the others must be defined by a user), variables (the user must define the variables "*machines*" and "*machine*" in our case) and a function (the user must define the function "*ET (EX)*" which is used for generating the exponentially distributed times between failures and the times to repair) that are used in the model – known as declarations.

By simulation, we can obtain interval estimations of the observed performance measures. To do so, the auxiliary text "*CPN'Replications.nreplications 10*" must be added to the model. After evaluating the text, 10 independent replications are run. To stop each replication after reaching a predefined simulation time, it is necessary to define a breakpoint function. In our model, we defined the function "*Breakpoint*" which stops each simulation run after reaching the simulation time of 525 600 hours (24 years). Such a simulation time is large enough to compare the simulation results with the steady-state values obtained by solving the mathematical model.



```
▼Declarations
  ▶Standard priorities
  ▼Standard declarations
    ▶colset BOOL
    ▶colset INT
    ▶colset STRING
  ▶colset UNIT
  ▼colset UNITtm = unit timed;
  ▼colset INTINF = intinf;
  ▼colset INTINFtm = intinf timed;
  ▼colset INTINFlist = list INTINF;
  ▼var machines: INTINFlist;
  ▼var machine: INTINF;
  ▼fun ET (EX) =
    round (exponential (1.0/EX));
▼Monitors
  ▶ES
  ▶EL
  ▶EK
  ▶ER
  ▶EW
  ▶ET
  ▼Breakpoint
     Type: Break point
     Nodes ordered by pages
    ▼Predicate
       fun pred () =
       IntInf.toInt(time())>525600
 New Page
```

Figure 3. Declarations and monitors of the simulation model

4 EXPERIMENTS AND THEIR RESULTS

In the previous sections, we presented the mathematical and simulation model which can be used for solving the problem of service capacity optimisation. Let us apply both models to an example. Let us consider that a group of repairmen services 10 homogeneous machines, for which we know that the mean time between failures is equal to 100 hours and the mean time to repair is 20 hours. Note that the values are not based on reality, the values are fictional to present the approach.

Table 2 presents the results obtained by solving the mathematical model using the Matlab function presented in the previous text. In the table the results for 1 up to 8 repairmen are presented. Note that the results for 9 repairmen are almost identical with the results for 8 repairmen when rounding up to 5 digits.

We can see that the value of *ES* is increasing with the number of the repairmen. Based on the value of *ES* we can calculate the utilisation *κ* of the repairmen. It holds that:

$$\kappa = \frac{ES}{n}\,. \qquad (25)$$

Based on formula (25), we can calculate that if we only employ a single repairman, then his/her utilisation is almost 100%, while for 8 repairmen their utilisation is about 21%. From the value of *ES,* we can obtain another interesting piece of information. The value of $n - ES$ gives information about the mean number of the idle repairmen. That means that for 1 repairman, the value tends to 0 whilst for 8 repairmen only 1.7 repairmen are busy on average and 6.3 repairmen are idle on average. That means we pay most of them for doing nothing.

n	1	2	3	4
ES	0.98162	1.51926	1.63914	1.66212
EL	4.11031	0.88447	0.16516	0.02726
EK	5.09192	2.40372	1.80430	1.68938
ER	4.90808	7.59628	8.19570	8.31062
EW [h]	83.74578	11.64341	2.01517	0.32796
ET [h]	103.74578	31.64341	22.01517	20.32796
n	5	6	7	8
ES	1.66607	1.66661	1.66666	1.66667
EL	0.00360	0.00035	0.00002	0.00000
EK	1.66967	1.66696	1.66669	1.66667
ER	8.33033	8.33304	8.33331	8.33333
EW [h]	0.04325	0.00424	0.00029	0.00001
ET [h]	20.04325	20.00424	20.00029	20.00001

Table 2. Results obtained by the mathematical model

By observing *ER,* we can conclude that for the given values of MTBF and MTTR, even with 8 repairmen only 8.3 machines are working on average. This could be a problem if we requested, for example, that at least 9 machines must work on average. We can see that for our parameters it is impossible to achieve this. Then it is necessary to increase the number of the machines and some of them are held in reserve.

Regarding the waiting times, we can see that they are decreasing very quickly with the increasing number of repairmen. For a single repairman, the mean waiting time is about 3.5 days while for 2 repairmen the mean waiting time is less than 0.5 day.

The results obtained by the simulation model are presented in Table 3. In this case we can see that two values are presented for each performance measure. This is because we can calculate confidence intervals for the simulation results. The upper values in the table correspond to the lower bounds of two-sided 95% confidence intervals and lower values to upper bounds of the two-sided 95 % confidence intervals. Thanks to the great value of the simulation time, when the simulation runs were stopped, the confidence intervals are narrow.

By comparing the corresponding results in both tables, we can say that the results obtained by the mathematical model lie in the confidence intervals or are close to them. Therefore, we can state that the results are mutually equivalent.

4.1 *Economic optimisation*

Our task is to define the optimal number of repairmen. It is obvious that each failure of the machine brings a loss in production because the machine does not work until is repaired. If we have only a few repairmen, then the total repairman costs (their wages) are low but the production losses may be significant. On the other hand, if we have a lot of repairmen, then the total repairman costs are higher (as stated earlier most of them are paid for doing nothing) but the production losses are not so significant.

Let *a* be the repairman wage per an hour and *b* the production loss per an hour per broken-down machine. Then the total repairman costs are equal to $a \cdot n$ and the production losses $b \cdot EK$. The total costs *TC* are then equal to the sum of the total repairman costs and the productions losses:

$$TC = a \cdot n + b \cdot EK \, . \qquad (26)$$

It is more than obvious that we want to minimise the total costs of the system. Let us assume that $a = 200$ CZK/h and $b = 400$ CZK/h. Based on the values of *EK* obtained by the mathematical model, we can estimate the total costs of the system – see Table 4.

n	1	2	3	4
ES	0.98148	1.51587	1.62896	1.66005
	0.98303	1.52416	1.64237	1.67274
EL	4.09352	0.88049	0.15758	0.02674
	4.15876	0.89666	0.16583	0.02808
EK	5.07553	2.39722	1.78705	1.68729
	5.14126	2.41997	1.80769	1.70032
ER	4.85874	7.58003	8.19231	8.29968
	4.92447	7.60278	8.21295	8.31271
EW [h]	83.44028	11.59362	1.92373	0.32214
	85.12962	11.82035	2.02057	0.33701
ET [h]	103.44967	31.56822	21.81626	20.29557
	105.25272	31.89904	22.02780	20.43960
n	5	6	7	8
ES	1.66176	1.66210	1.65986	1.66285
	1.67513	1.67273	1.67023	1.67729
EL	0.00329	0.00029	0.00001	0.00000
	0.00389	0.00042	0.00004	0.00001
EK	1.66530	1.66240	1.65988	1.66285
	1.67876	1.67312	1.67026	1.67729
ER	8.32125	8.32688	8.32974	8.32271
	8.33470	8.33760	8.34013	8.33715
EW [h]	0.03948	0.00342	0.00013	0.00000
	0.04658	0.00497	0.00046	0.00014
ET [h]	19.96730	19.93581	19.94192	19.98037
	20.14215	20.05138	20.04158	20.10435

Table 3. Results obtained by the simulation model

n	Repairman costs [CZK/h]	Production losses [CZK/h]	Total costs [CZK/h]
1	200	2037	2237
2	400	961	1361
3	600	722	1322
4	800	676	1476
5	1000	668	1668
6	1200	667	1867
7	1400	667	2067
8	1600	667	2267
9	1800	667	2467

Table 4. Total costs of the system

The individual cost functions, including the total cost function, can be depicted graphically – see Figure 4. We can see that the economically optimal number of the repairmen is 3 in this example.

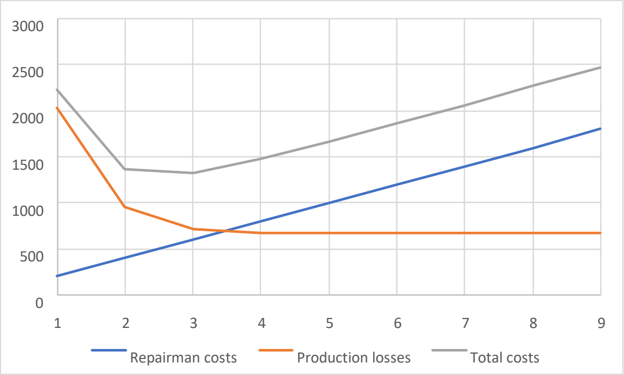


Figure 4. Graphical representation of the cost functions

5 CONCLUSIONS

The article presented the queueing approach to the optimisation of a service capacity. To optimise the number of repairmen the company should employ to repair the machines, the finite-source queueing model was applied.

The queueing model for the steady-state analysis can be expressed by two forms. The first form is represented by the linear equation system, which can be easily solved, for example, in Matlab – the Matlab code of the function is demonstrated in the article. The function solves the equation system numerically. The second method is an expression of the closed-form formulas for the steady-state probabilities of the individual states.

Using both methods gives information about the steady-state probabilities that are necessary to calculate the performance measures of the service system. The performance measures can also be estimated based on simulation experiments. In the article, we presented the coloured Petri net model of the system. We found out that the results are comparable.

Optimisation of the number of repairmen can be realised by the means of the cost function. In the article, we demonstrated the cost function which considers the repairman costs and the production losses. Our task was to minimise the total costs with respect to the number of repairmen.

REFERENCES

[Adan 2002] Adan, I.J.B.F., Resing, J. Queueing Theory. Eindhoven University of Technology, 2002.

[Cooper 1981] Cooper, R. B. Introduction to Queueing Theory, New York: Elsevier North Holland, 1981.

[Jensen 2013] Jensen, K. Coloured Petri nets: basic concepts, analysis methods and practical use. Vol. 1. Springer Science & Business Media, 2013. ISBN 978-3-642-08243-6.

[Kluvanek 1981] Kluvanek, P., Brandalik, F. Operations Research I (in Czech). Bratislava: ALFA Bratislava, 1981.

[Pesko 1999] Pesko, S., Smiesko, J. Stochastic models of Operations Research (in Slovak). Zilina: Zilinska univerzita v Ziline, 1999. ISBN 80-7100-570-3.

[Sztrik 2005] Sztrik, J. Finite-source queueing systems and their applications. In M. Ferenczi, A. Pataricza, L. Rnyai, eds. Formal Methods in Computing, Akadémia Kiadó, Budapest, Hungary 2005, pp. 311-356.

CONTACTS:

doc. Ing. Michal Dorda, Ph.D.
doc. Ing. Dusan Teichman, Ph.D.
Ing. Vojtech Graf
VSB – Technical University of Ostrava, Faculty of Mechanical Engineering, Institute of Transport
17. listopadu 15, Ostrava-Poruba, 708 33, the Czech Republic
+420 597 325 754, michal.dorda@vsb.cz, www.id.vsb.cz