A NEW APPROACH TO COMPUTER ANALYSIS OF QUEUING SYSTEMS WITHOUT PROGRAMMING

Zoran Ž. Avramović, Radomir Z. Radojičić, Saša D. Mirković

University of Belgrade, zoran.avramovic@sf.bg.ac.rs

Contribution to the state of the art

DOI: **10.7251/JIT1501025A** UDC: **662.62:519.245**

Abstract. The paper presents original object oriented programming system ***ARS*** for modelling and simulation queuing systems. Programming system was developed in programming language C++. It establishes connection with intrinsic, but also with other Windows programming packages, in a simple way, through object oriented environment. Basic characteristics and possibilities of programming system, as well as comparative analysis of simulators: mathematical model (analytical solution) - GPSS/H - ***ARS***, on the example of closed queuing network in the paper is given. The significant application for computer performance evaluation is reported.

Keywords: simulation, queuing system, programming system, computer performance evaluation.

INTRODUCTION

Since the invention of computer the growing importance is attached to virtual experiments on the computer. In view that the subject of attention in the research activities are more and more frequently the complex systems, where the mathematical or statistical analyses are either too complex or do not give results, the computer simulation has been obtaining a growing importance [6,7].

The methodological approach to modelling is directly related to the language choice for the system simulation. The experience in development of the theory and practice of simulation indicates that the most efficient means of the simulation models programming are the specialised simulation languages.

ARS is a simulation system which, in a simple way, by defined graphical models, creates the model (specifying a sequence of activities and precisioning the operations performed by their implementation over the attributes of the model objects).

The philosophy of the ***ARS*** programming system is based on the object approach to the modelling and simulation process.

THE CONCEPT OF THE SYSTEM

The ***ARS*** has been conceived as a general programming system intended for stochastic modelling and to processes oriented discrete simulation. It has been accomplished on an object approach, establishing a close, natural relationship between the analysed system and the simulation model - alleviating the modelling and simulation process to the user.

The ***ARS*** programming system enables the modelling and simulation of single- and multiphase of queuing systems, either single- or multi-channel ones (with equivalent or non-equivalent servers). It is also possible to model and simulate both open and closed networks, but also the queuing networks with and without the feedback.

The motive for development and implementation of this system is the authors' wish to expand and improve the capabilities of modern simulation languages and simplify their use and application.

Basic Characteristics of the System

The ***ARS*** programming system represents an object approach to the simulation modelling, which enables a simple modelling procedure, a high detail level of real processes, updating and use of the results in all steps of operation. It is enough to have a basic knowledge on the computer operation for its use. The users' requests to model the systems and laws of their functioning as truly as possible for the requirements of various investigations, are met by ***ARS*** through the flexible graphic environment available and a wide range of basic and derived "objects".

The basic elements present in modelling by the ***ARS*** programming system are static and dynamic objects (transactions). Any change in the object status results from the occurrence of an event, which may represent either starting or ending an activity.

The static objects direct, hold, update and partly control the dynamic objects, or change their parameters' values. One of the basic static objects in ***ARS*** is a server, representing the object by which the activity duration or delay is modelled.

The dynamic objects are created in input points of the model by the basic object for generation of objects, or defined by an initial state of the system. These objects are moved through the model bearing their characteristics - transaction parameters. As long as they are present in the model, the dynamic objects are in interaction with other objects in various forms.

During the simulation, it is possible to achieve the interaction and communication, in all variants, between static and dynamic objects (either directly or indirectly), with a possibility of using additional variables and functions from a wide range of the offered ones, as well as those derived there from.

The objects may be basic and derived. The basic objects are defined in advance and they cannot change their intended function, but their specified parameters can be changed.

The ***ARS*** programming system incorporates a capacity of modelling the queuing system by user through the application of offered or independently designed objects. By designing of the own objects of an arbitrary complexity and through the definition of their parameters, it becomes possible for the user to better imitate the computer system functioning so that the desired level of detailing in an accessible way is obtained.

The Modelling and Simulation

The model of the queuing system in ***ARS*** is a series of static objects through which the transactions pass during the simulation [8].

The simulation of queuing system operation in the ***ARS*** system is performed by a simple procedure: after creating a graphic model of the system and defining the work load and system resources parameters, the interactive simulation is activated and implemented [1,2]. During the simulation and on its completion, the user has at disposal (current and final) simulation results - statistic indicators of basic and derived performances of the queuing model.

The user has at disposal a large number of different probability distributions of arrivals of open network transactions, various distributions of service time and standard scheduling algorithms (FCFS, LCFS, PRI, RSS, and RR).

The modelling is implemented with the graphic editor's support, by simply creating a state-transition diagram, using the offered objects, and on the basis of already created models in the form of the derived objects. It would be possible to use either the mouse or develop a model by means of the keyboard.

The visual monitoring of the transactions flow in all stages of creation and modelling, as well as in the phase of simulation, represents an important innovation in relation to the program packets of similar nature.

The monitoring and analysis of the transient response and steady state are achieved by storing and erasing the selected statistic indicators, dynamic objects and their attributes. When detecting the steady state (by monitoring the steadiness of relevant, preselected indicators of system performances), it is possible to automatically stop the simulation.

By building a more detailed model and defining the derived objects based on the requirements, creating the own functions of distributions on the basis functions, by definition of users' functions to be used during the simulation, and based on the appearance of working environment, the adjustment of the system is made according to the users' requirements.

Working Environment

The ***ARS*** programming system communicates and uses, in a very flexible way, both internal and other Windows program packets having the required inter-relation with it. The system is furnished with the original program packet for a curve-fitting, representing an important supplementary characteristic. The use of a wide set of approximate functions and deriving of conclusions on the basis of numerical processing and graphical presentation of simulation results brings an advantage over the existing program packets of a similar application.

The user has in front of him a possibility of a dialogue with the computer in all steps of modelling and simulation.

The interactive simulation enables the experimenting during which, in randomly selected moments, or after the meeting of a particular condition, an arbitrary object in the system can be approached, watched and its current status monitored, the values of some variables can be changed, the status of the system also can be changed by introducing taking out or updating of the object parameters or even modify the initial diagram and continue the simulation thereafter.

A particular advantage of the interactive operation is shown in detecting, locating and correcting of errors.

The system comprises a standard HELP giving the required information in all steps of model elaboration and in all stages of simulation. There is also a possibility of a multi-media presentation of the programming system, with an animation and accompanying vocal presentations, supplementing the basic elements of the simulation system. Programming system was developed in programming language C++ [9,12].

Simple Queuing Model of the Closed Network (the Comparative Analysis)

The comparative analysis of queuing system (mathematical model - GPSS model - ***ARS*** model), has been made on a simple example of a closed network with central and three parallel connected servers with an exponential distribution of serving time (Figure 1). The results related to selected servers in relation to time and number of transactions in the system is presented, namely:
- mean serving time,
- mean utilisation of server, and
- mean turnaround time of transaction in the central server and its queue.

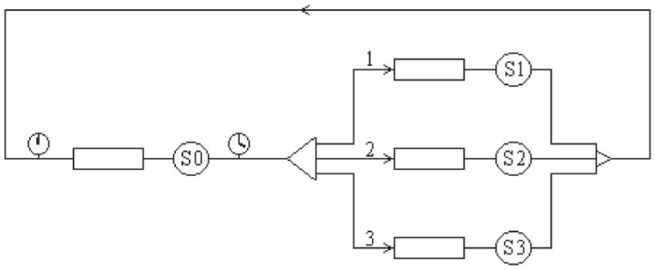

Figure 1. Queuing model of the closed network with the central and three parallel servers

Mathematical model - Analytical solution

As an indicator of performances of the analysed system, the server's utilisation has been observed. The analytical solution, through the application of Buzen's algorithm [5], gives following utilisation of the j-th server:

$$U_j = x_j \cdot G(n-1) / G(n), \quad j = 1, 2, \dots, k \qquad (1)$$

where the «normalising denominator»

$$G(n) = g(n, k) \qquad (2)$$

was derived as limes from a recurrent equation:

$$g(i, j) = g(i, j-1) + x_j \cdot g(i-1, j), \quad (i > 0, j > 0), \qquad (3)$$

with boundary conditions:

$$g(i, 0) = 0, \qquad i = 1,2,\dots,n \qquad (4)$$

$$g(0, j) = 1, \qquad j = 1,2,\dots,k \qquad (5)$$

where:

n - number of transactions circulating through the system,

k - number of servers, and

x_j - normalised request for the j-th server.

Simulation model in GPSS/H language

The listing of the program implemented in the simulation language GPSS/H is elementary [3,4,11].

Simulation model in the *ARS* programming system

The queuing model of the implemented solution of the closed network, in the programming system ***ARS*** is shown in Figure 1. The basic static objects used in this network:

Chronoscope (Object for time measuring - punctual measuring of small time particles)

Queue (Object for waiting for serving; finite queue; n-queue capacity)

Server (Variable delay object for serving or processing)

Collector (Merging point. Object for putting the transactions together)

Distributor (Branching point. Object for probabilistic branching of transactions)

Table 1. The server's basic parameters

Server	Server's name	Mean serving time
S0	Central	5 ms
S1	First parallel	25 ms
S2	Second parallel	25 ms
S3	Third parallel	10 ms

Table 2. The probabilistic branching element's basic parameters

Channel	Probability of transfer
1	2/11
2	2/11
3	7/11

Review and analysis of the results

Figure 2 shows the mean serving time of serving by the first parallel server in relation to the number of transactions in the model. The comparison with the value from Table 1 shows that the ***ARS*** programming system gives a lower and more stable relative error of serving time dispersion. For the illustration, we should mention that the mean relative error in the concrete case is: for ***ARS*** 2.35%, and for GPSS/H 4.37%.

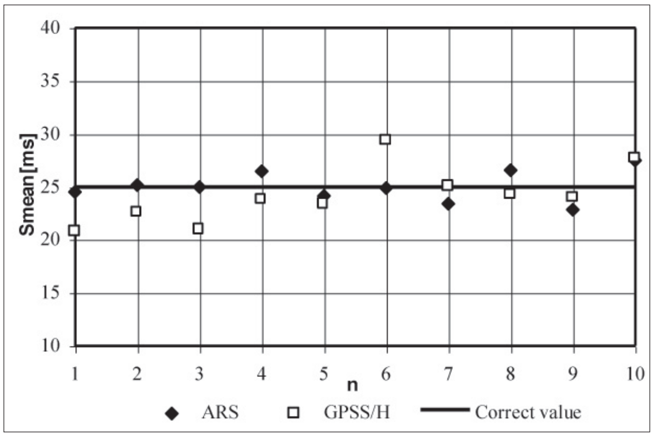

Figure 2. Mean serving time for the first parallel server in relation to the number of transactions in the model

The results showing the dependence of the central server's utilisation from the number of transactions in the model (Figure 3) indicate that the ***ARS*** programming system is significantly more convergent to results obtained by the analytical method from the mathematical model. In the case in question the relative error mean value for ***ARS*** is 3.28%, and for GPSS/H 5.91% (whereas their maximum values are 7.68% for ***ARS*** and 18.10% for GPSS/H).

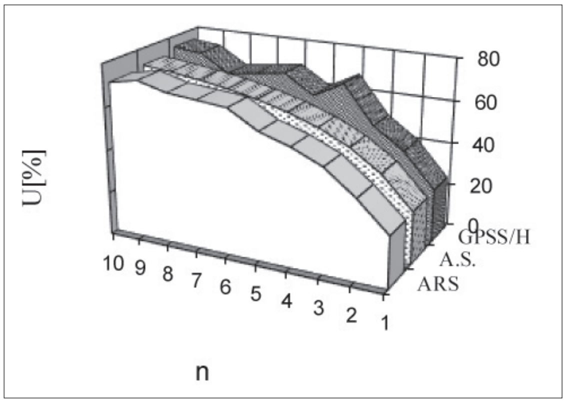

Figure 3. Utilisation of the central server in relation to the number of transactions in the model (A.S. - Analytical solution)

Of a particular interest are the results obtained by observation of the central server utilisation in relation to the simulation time. It is not difficult to notice in Figure 4 that utilisation offered by the ***ARS*** programming system, during the transient regime of simulation, has an unstable value which, as the simulation continues, converts into steady, close to the theoretical one. As the opposed to this, GPSS/H demonstrates, for the same parameter, a higher dispersion during the entire period under consideration. It has been found that during the whole period under consideration, the mean value of a relative error for the ***ARS*** programming system is 3.01%, and for GPSS/H 3.61%.

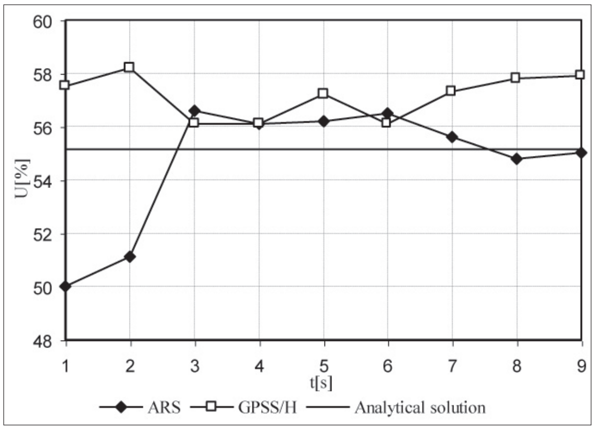

Figure 4. The simulator behavior in the transient time (on an example of the central server utilization) (for 4 transactions in the model)

The mean time of transactions' holding in the central server and its queue (Figure 5), and the mean time of transactions' queuing at the central server (Figure 6), in relation to the number of transactions in the model, for the programming system ***ARS*** and GPSS/H have approximately same law of change.

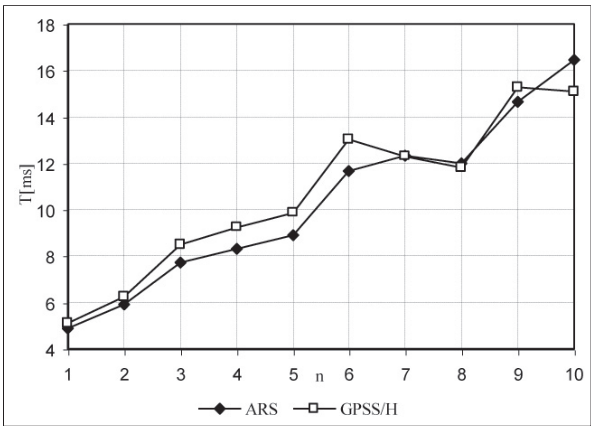

Figure 5. Turnaround time in the central server and its queue in relation to the number of transactions in the model

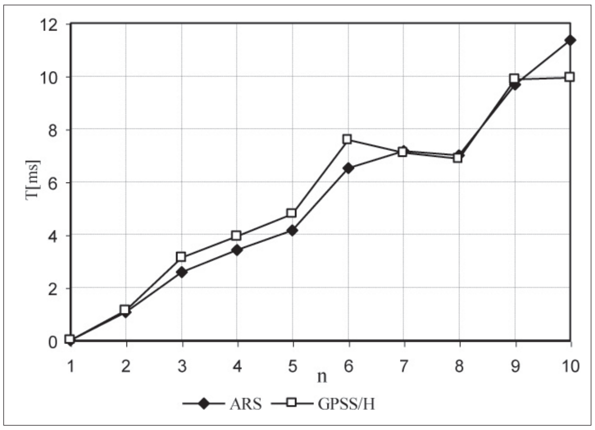

Figure 6. Turnaround time in the central server's queue in relation to the number of transactions in the model

The ***ARS*** programming system, using the own least-squares curve-fitting program, has established the approximate function for the mean time of transactions' queuing in relation to the number of transactions in the model:

$$Tq\,[ms] = a\cdot \ln(1+n) + b / \ln(1+n) + c \cdot \sqrt{n} + d / \sqrt{n} + e / n + f, \qquad n=1,2,... \qquad (6)$$

Values of coefficients in equation are:

$$a = -341.63,\ b = -38541.22,\ c = 640.82,\ d = 1048.67,\ e = 38190.69,\ f = 18087.80. \qquad (7)$$

For the found approximate dependence, the correlation coefficient is 0.991.

SIMULATION MODEL OF THE COMPUTER SYSTEM

The simulation analysis of the computer system model functioning has been made on the example of a closed multi-processor multi-processes network (Figure 7) with three processors, two disks, paging storage consisting of 512 pages, five priority queues for the processor, three priority queues for storage allocation and ten processes moving through the model, respecting the dependence of the processes activities (as shown in Figure 8). The attributes of all processes are presented in a tabular form (Table 3).

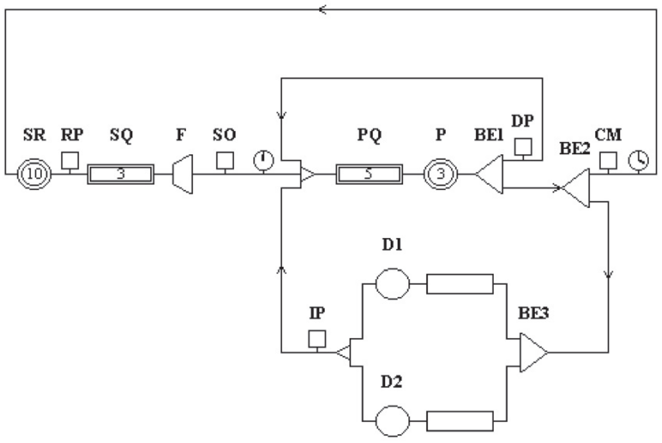

Figure 7. Queuing model of the multi-processor multi-processes computer system

Legend:

SR	- State of rest (10)	P	- Processors (3)
RP	- Request for the processes	DP	- Decreasing of priority
SQ	- Storage queue (3)	IP	- Increasing of priorities
F	- Filter	CM	- Clearing of memory
SO	- Storage occupation	D	- Disks (2)
PQ	- Processor queue (5)	BE	- Branching elements (3)

Table 3. Dynamic objects (processes) of the system with attributes and initial values

Process	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11
1	18	5	5	5	0.01	0	1000	1.00	10	2	0
2	32	4	3	4	0.02	0	2000	0.50	20	5	0
3	40	3	2	4	0.05	0	3000	0.00	20	8	0
4	46	3	2	4	0.05	0	4000	0.50	5	10	0
5	74	3	1	4	0.10	0	5000	0.50	10	50	0
6	88	3	1	4	0.10	0	6000	0.00	40	40	0
7	92	3	1	4	0.10	4	7000	0.30	50	60	0
8	144	3	1	4	0.20	2	8000	0.50	100	15	0
9	166	3	1	4	0.30	2	9000	1.00	200	17	0
10	200	3	1	4	0.40	3	10000	0.80	250	30	0

Legend: A1 - number of pages, A2 - current priority, A3 - the lowest priority, A4 - the highest priority, A5 - probability of transfer to the state of rest, A6 - request for process, A7 - mean turnaround time of the state of rest, A8 - probability of transfer to the first disk, A9 - mean time serving for disk, A10 - mean time of serving for the processor, A11 - remaining processing time (established after the interruption of the process due to the lapse of the processor time quantum)

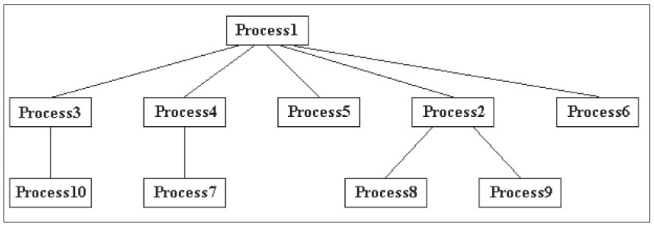

Figure 8. Tree presenting the dependence of processes by presence in the storage (when a process is active, all preceding ones must be stored)

The static objects in this network and their basic parameters are (serving times by all servers are the exponentially distributed):

SR: Multi-channel server (10 channels). It obtains the mean serving time through its attribute process (A7). While receiving the request from object RP, object SR releases the requested processes, in case they are served within it.

RP: The object for events generation and updating of variable systems, transaction attributes and object parameters. It generates the event to object SR for release to the first process and the process determined by attribute A6 of the current process.

SQ: A multiple queuing for storage. It consists of three queues. The highest priority queue comprises the processes with more than 100 pages, the medium priority - processes with the number of pages between 50 and 100 and the lowest priority - other processes.

F: The object letting in the transactions when the condition required is met. If the coming process requires more pages than available at the moment, it is on admitted until the required number of pages is available.

SO: By processing through this object, the value of the variable SR (the number of pages available in the storage, the global variable of the system) decreases by the number of pages of the current process.

PQ: Five queues accepting the processes by their priority attribute.

P: Multi-channel server with three processors. The mean serving time is obtained from the process attribute (A10). The upper limit of the serving time is specified - the processor time quantum. (The remaining serving time is recorded in attribute A11 of the process, and the process is released. If on the process occupation attribute A11 is other than zero, this value is added to the current serving time).

BE1: The process is forwarded to the first branch if attribute A11 is more than zero.
BE2: The branching is implemented according to the leaving probability (the first branch is approached with the probability given by attribute A5).
DP: It decreases the priority to the process by 1 (if it is higher than the minimum value for that process).
CM: The storage is clear. Variable SR increases by the number of required pages for the current process.
BE3: Divides the processes according to probabilities obtained from attribute A8 of the current process (disk D1 is approached with the probability given by attribute A8).
D1 and **D2**: Servers with mean serving time according to attribute A9 of the process.
IP: It increases the priority to the process (if it is lower than the maximum value for that process).

Computer model exploitation

The simulation starts from the state where all processes are inactive (meaning that the storage is quite clear). After the expiration of the phase of rest of the process, the storage begins to be fed, gradually. By watching the transient time up to the twentieth second (Figure 9); one can see that the storage utilisation grows up to the steady value of 80%, when all processes are active.

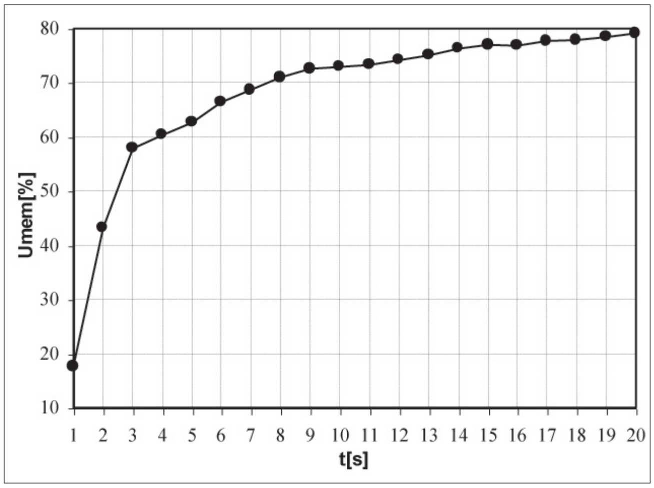

Figure 9. Storage utilisation in the transient time (for the processor time quantum q = 9 ms)

Using its own least-squares curve-fitting program, the ***ARS*** has established the approximate function for storage utilisation in relation to time:

$U\,[\%] = a \cdot \ln(t) + b \cdot t^2 + c \cdot t + d\sqrt{t} + e,\qquad (t > 0, t[s])$. (8)

The values of coefficients in equation are:
$a = 105.97,\ b = -0.13,\ c = 17.44,\ d = -153.83,\ e = 154.04.$ (9)

For the obtained approximate dependence, the correlation coefficient is 0.998.

The particularly interesting results have been obtained for the total times of process staying in the subsystem (from occupation of storage to cessation of further activities and clearance of the occupied storage - between two chronoscopes) in relation to time (Figure 10). The first process is needed by all other processes (if any process is activated, then the first one must also be activated, Figure 8), so that it practically does not leave the subsystem. With the third process, the rectilinear dependence disappears, and the curve is more inclined to abscissa (since it is conditional for one process only - the tenth). With the tenth process the curve is much closer to abscissa (as expected, since that process is not conditional for any other one and its mean rest time is considerably longer).

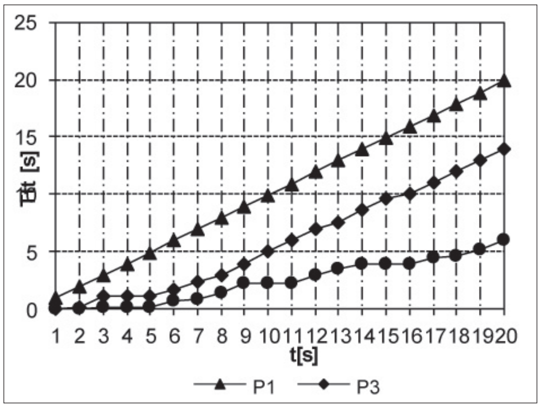

Figure 10. Turnaround time of the processes between two chronoscopes (for the processor time quantum q = 9 ms)

The process priority changes in relation to whether it addresses the disk (the priority increases), or interrupts due to the processor time quantum expired (the priority decreases). Since the priority change depends on the time quantum, hence the total turnaround time in the priority queues depends on quantum (as the quantum grows, the number of process interruptions decreases, and therefore the priority of the processes decreases less frequently - which results in higher priorities, so that the turnaround time in the higher priority queues is longer). Figure 11 shows that the total turnaround time in priority 4 queue is significantly longer than turnaround time in priority 1 queue with the longer quantum of the processor time.

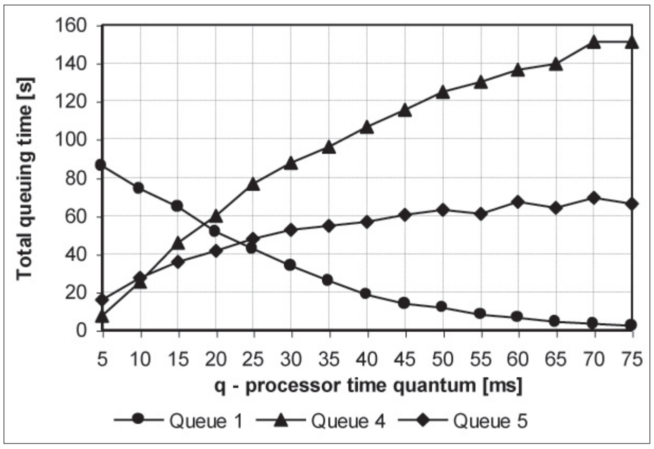

Figure 11. Total queuing times for the processor in relation to the processor time quantum

Conclusion

The paper presents the original programming system ***ARS*** for simulation of the queuing system. The presented programming system is applicable for the simulation of open and closed queuing networks of an arbitrary complexity. It enables a simple modelling, interactive simulation, adequate presentation and numerical processing of results, arbitrary adjustment of working environment and system resources and other advantages, aiming at a more efficient designing.

The programming system offers to the user a possibility of introducing his own objects, enabling a more successful modelling of complex systems. The own system for numerical processing of the simulation results is an efficient tool for collecting the statistic indicators of the relevant system performances. The interactive simulation and a possibility of changing the structure of the analysed of queuing system makes a basis for a successful detection and removal of errors found during the modelling.

Considering the before specified properties of the ***ARS*** programming system, we are of the opinion that it represents an adequate tool in the field of modelling the discrete queuing systems.

References

[1] Avramović, Z.Ž., Radojičić, R.Z., "ARS - new programming system for computer performance analysis" (in Serbian), *ETRAN*, Budva, 1996, 47-50.

[2] Avramović, Z.Ž., Radojičić, R.Z., Savkić, D.S., "The modelling and simulation operation of a railway station in the ARS programming system" (in Serbian), *JUŽEL*, Niš, 1996, 31-34.

[3] Avramović, Z.Ž., Radojičić, R.Z., Savkić, D.S., "Basic characteristics of object oriented programming system ARS for simulation queuing systems" (in Serbian), *YuInfo*, Brezovica, 1996, 107.

[4] Banks, J., Carson, J. S., Sy, J.N., *Getting Started with GPSS/H*, Wolverine Software Corporation, Annadale, 1989.

[5] Buzen, J.P., "Computational algorithms for closed queuing networks with exponential servers", *Comm. ACM*, 16 (1973) 527-531.

[6] Gordon, G., *System Simulation*, Prentice-Hall, Englewood Cliffs, Nj, 1978.

[7] Graham, R.L., Knuth, D.E., Patashnik, O., *Concrete mathematics: a foundation for computer science*, Addison-Wesley Publishing Company, 1988.

[8] Kleinrock, L., *Queuing theory, Volume I: Theory*, John Wiley & Sons, Nj, 1975.

[9] Nutaro, J. J. (2011). Building software for simulation: theory and algorithms, with applications in C++. John Wiley & Sons.

[10] Pritsker, A.A.B., *Introduction to Simulation and SLAM II*, Systems Publishing Corporation, West Lafayette, Indiana, 1984.

[11] Schriber, T.J., *Simulation using GPSS*, John Wiley, 1974.

[12] Varga, A. (2001, June). *The OMNeT++ discrete event simulation system*. In Proceedings of the European simulation multiconference (ESM'2001) (Vol. 9, No. S 185, p. 65). sn.

Submitted: April 4, 2015.
Accepted: May 29, 2015.