PERFORMANCE MODELLING OF NOV AND GRID PARALLEL COMPUTERS

[a]MICHAL HANULIAK

Dubnica Technical Institute, Sladkovicova 533/20, 018 41 Dubnica nad Vahom, Slovakia
email: [a]michal.hanuliak@gmail.com

This work was done within the project Modelling, optimisation and prediction of parallel computers and algorithms at University of Žilina supervised by Prof. Ing. I. Hanuliak, CSc. The authors, as the co-workers of this project, gratefully acknowledge the help of all co-workers taking part in this project.

Abstract: The paper describes development, realization and verification of analytical models for the study of the basic performance parameters of parallel computers based on connected computer systems (NOW, Grid).The suggested model considers for every node of the NOW or Grid networks one part for the own workstation´s activities and another one for node's communication channel modelling of performed data communications. In case of using multiprocessor system as modern node's communication processor the model for the own node's activities then is M/D/m system and for every node's communication channel M/D/1 system. The achieved results of the developed models were compared with the results of the common used analytical and simulation model to estimate the magnitude of their improvement.

Keywords: parallel computer, network of workstation (NOW), Grid, analytical modelling, queuing theory, performance evaluation.

1 Trends in parallel computers

In the first period of parallel computers between 1975 and 1995 dominated scientific supercomputer which was specially designed for the High performance computing (HPC). These computers have used computing model based mostly on data parallelism. Increased processor performance was caused through massive using of various parallel principles in all forms of produced processors. Parallel principles were used so in single PC´s and workstations (scalar and super scalar pipeline architecture, symmetrical multiprocessor or multicore systems (SMP) as on POWER PC or in their common using in connected network of workstations NOW (Network of workstations). The gained experience with the implementation of the parallel principles and the extension of computer networks, leads to using interconnected powerful workstations for parallel solution. This trend is characterised through downsizing of supercomputers as Cray/SGI, T3E and from other massive parallel systems (number of used processor >100) to cheaper and more universal parallel computers in the form of a network of workstations (NOW). This period we can name as the second period. Their large growth since 1980 have been stimulated by the simultaneous influence of three basic factors [7, 14]

- high performance processors and computers
- high speed interconnecting networks
- standardized tools for development of parallel algorithms.

The developing trends are actually going toward building of wide spread connected NOW networks with high computation and memory capacity (Grid). Likewise new or existed supercomputers could be a member of NOW as its workstation [20]. Conceptually Grid comes to the definition of the metacomputer. Metacomputer can be understood as the massive computer network of computing nodes built on the principle of the common use of existing processors, memories and other resources with the objective to create an illusion of one huge, powerful supercomputer. Such higher integrated forms of NOW (Grid module) named as Grid systems or metacomputers we can define as the third period in trends of parallel computers.

2Architecture of dominant parallel computers

The actual dominant asynchronous parallel computers are based on various forms of computer networks (cluster), network of workstation (NOW|) or more integrated network of NOW networks (Grid) [1, 19]. They are composed of a number of fully independent computing nodes (processors, cores or powerful workstations). From the point of programmer there is typical at developing parallel algorithms (co-operation and synchronization of parallel processes) inter process communications (IPC). According the latest trends synchronous based on PC computers (single, SMP) and asynchronous parallel computers are dominant nowadays.

2.1 Network of workstations

There has been an increasing interest in the use of networks of workstations (NOW) connected together by high speed networks [17] for solving large computation intensive problems. We illustrated at Fig. 1 integrated parallel computer consisted of NOW workstations. The used workstations are mainly extreme powerful personal workstations based on multiprocessor or multicore platform [1, 5]. This trend is mainly driven by the cost effectiveness of such systems as compared to massive multiprocessor systems with tightly coupled processors and memories (Supercomputers). Network of workstations (NOW) [8, 9] has become a widely accepted form of high performance computing (HPC). It is clear that any classical parallel computers (massive multiprocessor, supercomputers) could be a workstation of such NOW [20].

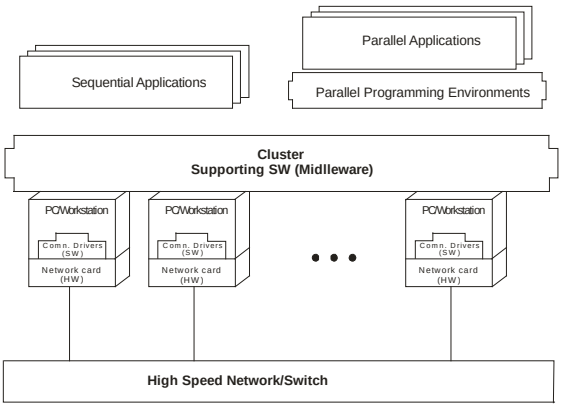

Fig. 1.Architecture of NOW.

2.2 Grid systems

In general Grids represent a new way of managing and organising of resources generally in clusters like network of NOW networks. This term define massive Grid with following basic characteristics

- wide area network of integrated all free computing resources. It is a massive number of interconnected networks, which are connected through high speed connected networks during which time whole massive system is controlled with network operation system, which makes an illusion of powerful computer system (virtual supercomputer)
- grants a function of metacomputing that means computing environment, which enables to individual applications a functionality of all system resources
- system combines distributed parallel computation with remote computing from user workstations [22].

2.3 Conventional HPC environment versus Grid environments

In Grids, the virtual pool of resources is dynamic and diverse, since the resources can be added and withdrawn at any time according to their owner's discretion, and their performance or load can change frequently over the time. The typical number of resources in the pool is of the order of several thousand or even more. For all these reasons, the user has very little or no a priori knowledge about the actual type, state and features of the resources constituting the pool.

An application in a conventional HPC parallel environment typically assumes a pool of computational nodes from (a subset of) which a virtual concurrent machine is formed [18]. The pool

consists of PC's, workstations, and possibly supercomputers, provided that the user has access (valid login name and password) to all of them. Such virtual pool of nodes for a typical user can be considered as static and this set varies in practice in the order of 10 – 100 nodes. Table 1 summarize mine differences between conventional distributed and Grid systems. From performed comparisons we can say that

- HPC environments are optimised for to provide maximal performance
- Grids are optimised to provide maximal resource capacities.

Table 1.Comparison HPC and Grid of environments.

	Conventional HPC environments	Grid environments
1.	A virtual pool of computational nodes	A virtual pool of resources
2.	A user has access (credential) to all nodes in the pool	A user has access to the pool but not to individual nodes
3.	Access to a node means access to all resources on the node	Access to a resource may be restricted
4.	The user is aware of the applications and features of the nodes	User has little or no knowledge about each resource
5.	Nodes belong to a single trust domain	Resources span multiple trust domains
6.	Elements in the pool 10 – 100, more or less static	Elements in the pool >>100, dynamic

3 Performance evaluation of parallel computer

The study of the performance of computers attempts to understand and predict the time dependent behaviour of parallel computers. It can be broadly divided into two areas – modelling and measurement. These can be further divided by objective and by technique. These two apparently disjoint approaches are in fact mutually dependent and are both required in any practical study of the performance of a real or planned system. The overall process of estimating or predicting the performance of a computer system is sometimes referred to as performance analysis or performance evaluation.

4 Performance evaluation methods

Several fundamental concepts have been developed for evaluating parallel computers. Tradeoffs among these performance factors are often encountered in real-life applications. To the performance evaluation we can use following methods

1. analytical methods
   - application of queuing theory [3, 11, 12]
   - Petri nets [4]
   - asymptotic (order) analyse [9, 10]
2. simulation methods [15]
3. experimental measurement
   - benchmarks [11, 13]
   - direct parameter measuring [16].

4.1 Analytic techniques

There is a very well developed set of techniques which can provide exact solutions very quickly, but only for a very restricted class of models. For more general models it is often possible to obtain approximate results significantly more quickly than when using simulation, although the accuracy of these results may be difficult to determine. The techniques in question belong to an area of applied mathematics known as queuing theory, which is a branch of stochastic modelling. Like simulation, queuing theory depends on the use of computers to solve its models quickly. We would like to use techniques which yield analytic solutions.

4.2 The simulation method

Simulation is the most general and versatile means of modelling systems for performance estimation. To reduce the cost of a simulation we may resort to simplification of the model which avoids explicit modelling of many features, but this increases the level of error in the results. If we need to resort to simplification of our models, it would be desirable to achieve exact results even though the model might not fully represent the system. At least then one source of inaccuracy would be removed. At the same time it would be useful if the method could produce its results more quickly than even the simplified simulation. Thus it is important to consider the use of analytic and numerical techniques before resorting to simulation. The result values of simulation model have always their discrete character, which do not have the universal form of mathematical formulas. The accuracy of simulation model depends therefore on the accuracy measure of the used simulation model for the given task. Simulation can contribute to the behaviour analyse of the parallel computers to analyse of the large modern parallel computers is very unpractical and unusable. His disadvantage is also that the achieved results are not universal. But it is very useful in these cases in which we are not able to apply no analytical method and so the simulation methods is the only analytical tool or in cases in which exist only approximate analytical methods and the simulation became the verification tool of achieved analytical results.

4.3 Asymptotic (Order) analysis

In the analysis of algorithms (serial, parallel), it is often cumbersome or impossible to derive exact expressions for parameters such as run time, speedup, efficiency, issoefficiency etc. In many cases, an approximation of the exact expression is adequate. The approximation may indeed be more illustrative of the behaviour of the function because it focuses on the critical factors influencing the parameter. We have used an extension of this method to evaluate parallel computers and algorithms in [9].

4.4 Experimental measurement

Evaluating system performance via experimental measurements is a very useful alternative for parallel systems and algorithms. Measurements can be gathered on existing systems by means of benchmark applications that aim at stressing specific aspects of the parallel systems and algorithms. Even though benchmarks can be used in all types of performance studies, their main field of application is competitive procurement and performance assessment of existing systems and algorithms. Parallel benchmarks extend the traditional sequential ones by providing a wider a wider set of suites that exercise each system component targeted workload.

5 Application of queuing theory systems

Queuing theory systems are classified according to various characteristics, which are often summarised using Kendall`s notation [6, 12]. This describes a queue as, for instance, M/M/m. The first letter describes the distribution of arrivals into the queue, the second letter describes the distribution of service times for entities which reach the front of the queue and the third number describes the number of servers for the queue. Distributions are identified by code letters, so that M means exponential times (from the name Markovian), D means constant or deterministic times, G means generally distributed (i.e. only the mean is considered significant).

5.1 Little's law

One of the most important results in queuing theory is Little's Law. This was a long standing rule of thumb in analyzing queuing systems, but gets its name from the author of the first paper which proves the relationship formally. It is applicable to the behaviour of almost any system of queues, as long as they exhibit steady state behaviour. It relates a system oriented measure - the mean number of customers in the system - to a

customer oriented measure - the mean time spent in the system by each customer (the mean end-to-end time), for a given arrival rate. Little's law says

$E(q) = \lambda . E(t_q)$ or it's following alternative

$E(w) = E(q) - m . \rho$ (m – services).

where the needed parameters are as

- λ - arrival rate at entrance to a queue
- m - number of identical servers in the queuing system
- ρ - traffic intensity (dimensionless coefficient of utilization)
- q - random variable for the number of customers in a system at steady state
- w- random variable for the number of customers in a queue at steady state
- $E(t_s)$- the expected (mean) service time of a server
- E (q)- the expected (mean) number of customers in a system at steady state
- E (w)- the expected (mean) number of customers in a queue at steady state
- $E(t_q)$- the expected (mean) time spent in system (queue + servicing) at steady state
- $E(t_w)$- the expected (mean) time spent in the queue at steady state.

5.2 Queuing networks

Continuing the examination of analytically tractable models, we look for useful results for networks of queues. These can be divided into two main groups, known as product form and non-product form. Product form networks have the property that they can be regarded as independently operating queues, where steady state can be expressed as both a set of global balance equations on customer flow in the whole network and a set of local balance equations on each queue. Local flow balance says that the mean number of customers entering any queue from all others must equal the number leaving it to go to all others, including customers which leave and rejoin the same queue immediately.

5.3 Jackson theorem

Consider the case of a network of U queue/server nodes (workstations). Customers enter the network at node j in a Poisson stream with rate γ_j. Each node has a multiple servers m (workstations based on multiprocessor with m services) and service times are distributed exponentially, with mean $1/\mu_j$, (j = l,…, U). When a customer leaves node i it goes to node j with probability r_{ij}. Customers from i leave the network with probability

$$1-\sum_{j=1}^{U} r_{ij}$$

Now let λ_i be the average total arrivals at node i, including those from outside (external input) and those from other nodes (internal inputs). If the network is in steady state, $\lambda_i$is also the rate of customers leaving i node (including intern output). Overall we can formulate a set of „flow balance equations" which express these flows.

$$\lambda_i = \gamma_i + \sum_{i=1}^{U} \lambda_i r_{ij} \quad j=1,2,\ldots, U$$

As long as the network is open, i.e. at least one γ_i is not zero, this represents a set of linear simultaneous equations with an obvious solution. Let be traffic intensity at i node

$\lambda_i / m_i \cdot \mu_i < 1$

The joint distribution of the number of customers p (k_1, k_2, ...k_U) at each of the U nodes, $p_1(k_1)$, $p_2(k_2)$, … $p_U(k_U)$, can be expressed as

$$p(k_1,k_2,\ldots,k_U) = p_1(k_1)\cdot p_2(k_2)\cdot\ldots\cdot p_U(k_U) = \prod_{i=1}^{U} p_i.k_i$$

This is Jacksom theorem for M/M/m system. The individual probabilities $p_i(k_i)$ are given as

$$p_i(k_i) = \begin{cases} p_0 \dfrac{(m\rho)^i}{i!}, & \text{pre } 1 \le i \le m \\ p_0 \dfrac{\rho^k m^m}{m!}, & \text{pre } i > m \end{cases}$$

, where $p_0 = \left[\sum_{i=0}^{m-1} \dfrac{(m\rho)^i}{i!} + \dfrac{(m\rho)^m}{m!(1-\rho)}\right]^{-1}$.

Jackson's theorem describes each node as an independent single server system with Poisson arrivals and exponential service times. The total average number of customers in the whole NOW module $E(q)_{now} = \sum_{i=1}^{U} E(q)_i$,where $E(q)_i$ is given as

$$E(q)_i = \frac{(\rho m)^{m+1}}{(m-1)!\left[\sum_{i=0}^{m} \dfrac{(m\rho)^i}{i!}\left[(m-i)^2 - i\right]\right]}$$

Then from Little's Law, total time spent by customers in the network $E(t)_q$ is $E[t_q]_{now} = \sum_{i=1}^{U} \dfrac{E(q)_i}{\lambda_i}$

Jackson theorem assumes for its applying verification of assumed independence of individual network computing nodes. Every element on its right side is a solution of isolated M/M/m geeing system with their independent average input value λ_i . We can get the intensities of this individual inputs λ_i with solving a system of linear differential equations for concrete values of extern inputs γ_i and for given transition matrix r_{ij} .

6 Modelling of the NOW and Grid

NOW is a basic module of any Grid system (network of NOW networks as for example Internet). In principle we are assumed any constraints on structure of communication system architecture. Then we are modelling one workstation as a system with two dominant overheads

- computation overheads (processor´s latency)
- communication latency.

To model these overheads through applying queuing theory we created mathematical model of one i-th computing node according Fig. 6, which models

- computation overheads (processor´s latency) as queuing theory system
- every communication channel of i-th node LI_i i=1,2, …U (Link interface) as next queuing theory systems (communication system).

Such communication network in NOW module we can represent by a weighted graph where their nodes are individual workstations (Fig. 2.).

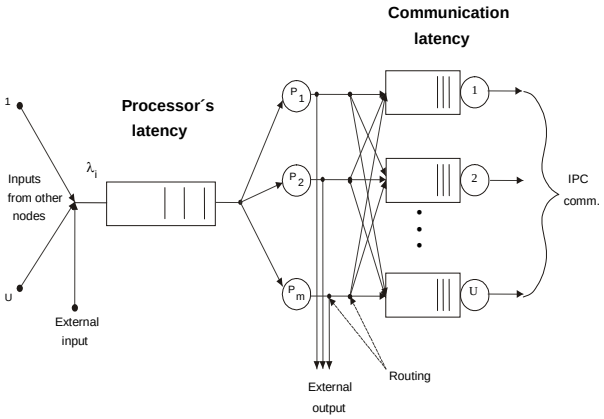

Fig. 2.Mathematical model of i–th node of NOW.

IPC data arrive at random at a source node and follow a specific route in the networks towards their destination node. Data lengths of communicated parallel processes in data units (for example in words) are considered to be random variables following distributions according Jackson theorem. Those data units are then sent independently through the communication network nodes towards the destination node. At each node a queue of incoming data units is served according to a first-come first-served (FCFS) discipline.

6.1 Suggestion and derivation of precised models

Model with M/D/m and M/D/1 systems

The used model were built on assumptions of modelling incoming demands to program queue as Poisson input stream and of the exponential inter-arrival times between communication inputs to the communication channels.

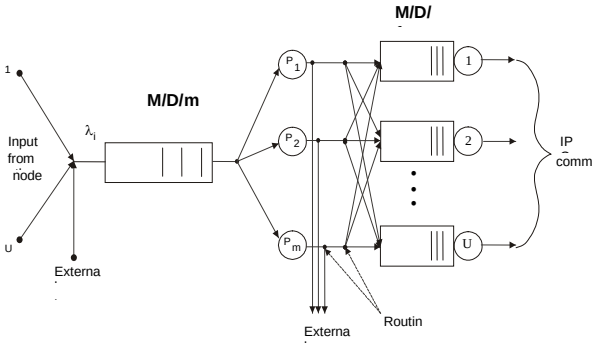

Fig. 3.Precise mathematical model of i-th node.

The idea of the previous models were the presumption of decomposition to the individual nondependent channels together with the independence presumption of the demand length, that is the demand length is derived on the basis of the probability density function $p_i = \mu\ e^{-\mu t}$ for t > 0 and f (t) = 0 for t $\leq$ 0 independent always at its input to the node. On this basis it was possible to model every used communication channel as the queuing theory system M/M/1 and derive the average value of delay individually for every channel. The whole end-to-end delay was then simply the sum of the individual delays of the every used communication channel.

These conditions are not fulfilled for every input load, for all architectures of node and for the real character of processor service time distributions. These changes could cause imprecise results. To improve the mentioned problems we suggested the behaviour analysis of the modelled NOW module improved analytical model, which will be extend the used analytical model to more precise analytical model (Fig.3.) supposing that

- we consider to model computation activities in every node of NOW network as M/D/m system
- we consider an individual communication channels in i- th node as M/D/1 systems. In this way we can take into account also the influence of real non exponential nature of the inter-arrival time of inputs to the communication channels.

These corrections may to contribute to precise behaviour analysis of the NOW network for the typical communication activities and for the variable input loads. According defined assumption to modelling of the computation processors we use the M/D/m queuing theory systems according Fig. 3. To find the average program queue delay we used the approximation formula for M/D/m queuing theory system according as

$$E(t_w)\,(M/D/m_i) = \left[1+(1-\rho_i)\cdot(m_i-1)\cdot\frac{\sqrt{45m_i}-2}{16\rho_i m_i}\cdot\frac{E(t_w)\ (M/D/1)}{E(t_w)\ (M/M/1)}\cdot E(t_w)\ (M/M/m_i)\right]$$

, in which

- ρ_i - is the processor utilization at i-th node for all used processors
- m_i - is the number of used processors at i-th node
- $E(t_w)$ (M/D/1), $E(t_w)$ (M/M/1) and $E(t_w)$ (M/M/m) are the average queue delay values for the queuing theory systems M/D/1, M/M/1 and M/M/m respectively

The chosen approximation formulae we selected from two following points

- for his simply calculation
- if the number of used processors equals one the used relation gives the exact solution, that is W(M/D/1) system
- if the number of processors is greater than one chosen relation generate a relative error, which is not greater as 1%. We verified and confirmed it through simulation experiments.

Let $\overline{x_i}$ define the fixed processing time of the i-th node processors and $E(t_w)_i$ (PQ) the average program queue delay in the i-th node. Then ρ_i of the i-th node is given as

$$\rho_i = \frac{\lambda_i . \overline{x_i}}{m_i}$$

Then the average waiting time in PQ queue $E(t_w)_i$ (M/D/m$_i$) is given through the following relations

$$E(t_w)_i(M/D/1) = \frac{\rho_i \cdot \overline{x_i}}{2(1-\rho_i)}\quad E(t_w)_i(M/M/1) = \frac{\rho_i \cdot \overline{x_i}}{1-\rho_i}$$

$$E(t_w)_i(M/M/m_i) = \frac{\frac{(m_i\cdot\rho_i)^{m_i}}{m_i!(1-\rho_i)}}{\sum_{j=0}^{m_i-1}\left[\frac{(m_i\cdot\rho_i)^j}{j!}+\frac{(m_i\cdot\rho_i)^{m_i}}{m_i!(''1-\rho_i)}\right]\cdot\frac{\overline{x_i}}{m_i}(1-\rho_i)}$$

By substituting relations for ρ_i, $E(t_w)_i$ (M/D/1), $E(t_w)_i$ (M/M/1) and $E(t_w)_i$ (M/M/m$_i$) in the relation for $E(t_w)$ (M/D/mi) we can determine $E(t_w)_i$ (PQ). Then the total average delay for the communication activities in i-th node is simply the sum of average message queue delay (MQ) plus the fixed processing time

$$E(t_w)_i = E(t_w)_i(PQ) + \overline{x_i}$$

To find the average waiting time in the queue of the communication system we consider the model of one communication queue part node as M/M/1 queuing theory system according Fig.7. Let $\overline{x_{ij}}$ determine the average servicing time for channel j at the node i. x_{ij}. Then ρ_{ij} as the utilization of the communication channel j at the node i is given as

$$\rho_{ij} = \frac{\lambda_{ij}\,\overline{x_{ij}}}{S_{ij}}$$

,where S_{ij} defines the speed of communication channel at j-th node. For simplicity we will assume that S_{ij}=1. The total incoming flow to the communication channel j at node i which is given through the value λ_{ij} and we can determine it with using of routing table and destination probability table in the same way as for a value λ_i. Let $E(t_w)_{ij}$ (LQ) be the average waiting queue time for communication channel j at the node i. Then

$$E(t_w)_{ij}(LQ) = \frac{\rho_{ij} \cdot \overline{x_{ij}}}{(1-\rho_{ij})}$$

The total average delay value is the queue $E(t_w)_{ij}$ is given then as $E(t_w)_{ij} = E(t_w)_{ij}(PQ) + \overline{x_{ij}} = \frac{\rho_{ij} \cdot \overline{x_{ij}}}{(1-\rho_{ij})} + \overline{x_{ij}}$

If we now substitute the values for $E(t_q)_i$ and $E(t_q)_{ij}$ to the relation for $E(t_q)_{now}$ we can get finally the relation for the total average delay time of whole NOW model is given as

$$E(t_q)_{now} = \frac{1}{\gamma}\left[\sum_{i=1}^{U}\left(E(t_w)_i(PQ) + \overline{x_i}\right) + \sum_{j=1}^{u_i}\left(E(t_w)_{ij}(LQ) + \overline{x_{ij}}\right)\right]$$

7 Results

The achieved results we illustrated at Fig.4. They are representing the results and relative error for the average value of the total message delay in the 5-noded communication network of classical analytical model (M/M/m + M/M/1) and developed precised analytical model ((M/D/m + M/D/1) in which we considered the fixed delay for the multiprocessor latency. The same fixed delay was included to the average communication delay at each node and in simulation model too. In both considered analytical models (M/M/m + M/M/1, M/D/m + M/D/1) decreasing of processor utilization ρ cause decreasing of total average delay in NOW module E $(t_q)_{now}$. Therefore parallel processes are waiting in the processes queues shorter time. In contrary decreasing of communication channel speeds increase channel utilization and then data of parallel processes have to wait longer in communication channel queues and increase the total node delay of parallel processes. The tested results are the part of all done tests with developed analytical models. The whole set of experimental results has proved, that the analytical model (M/D/m + M/D/1) provided best results and the analytical model (M/M/m + M/M/1) the worst ones. The deterministic time to perform parallel processes at node´s multiprocessor activities that is the servicing time of PQ queue was settled to 8 μs and the extern input flow for each node was the same. To vary the processor utilization we modified the extern input flow in the same manner for each node. The best analytical model (M/D/m + M/D/1) provides very precision results in the whole range of input workload of multiprocessors and communication channels utilization with relative error, which does not exceed 6.2% and in most cases were in the range up to 5%.

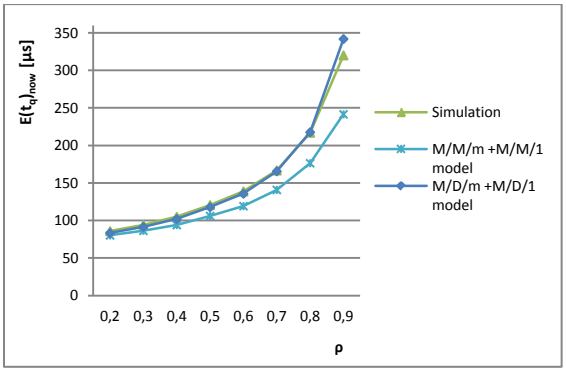

Fig. 4.Comparison of analysed models.

This is important in the range of heavily loaded network (about 80 to 90%) the accurate results are needed to avoid effectively the bottleneck congestions and other system instabilities. Comparison of this best analytical model to analytical model (M/M/m + M/M/1) according Fig. 4 show the improvements in all range of input multiprocessor loads (from 20 to 90%). The relative errors of worst analytical model are from 7 to 25%. This is due influences of processes queues delays, the nature of inter-arrival input to the communication channel in the case of high processor utilization. Developed analytical model could be applied for large NOW networks practically without any increasing of the computation time in comparison to simulation method. Simulation models require oft three orders of magnitude more computation time for testing such a massive metacomputer. Therefore limiting factor of the developed analytical models was not computation time but space complexity of memories. The needed tables RT and DPT require O (n^2) memory cells, thus limiting the network analysis to the number of computing nodes N about 100-200. In case of using system of linear equations to find λ_i and λ_{ij}, most parallel algorithms use to its solution Gauss elimination method (GEM) with its computation complexity O (n^3) [2, 21]. These values are however adequate to handle most existing communication network. In addition also for future massive metacomputers we could use a hierarchically modular architecture (decomposition).

8 Conclusion and perspectives

Performance evaluation of computers (sequential, parallel) generally used to be a very hard problem from birthday of computers. It was very hard to apply any analytical methods (queuing theory results) to performance evaluation of sequential computers because of their high number of not predictable parameters. Actually dominant using of multiprocessor and multicore parallel computers open more possibilities to apply a queuing theory results to analyse their performance. This implies the known queuing theory knowledge, that many inputs, which are inputting to queueing theory system and are generating at various independent resources by chance, could be a good approximation of Poisson distribution. Therefore we could model multiprocessor workstation as M/D/m or communication channel as M/D/1 queuing theory systems in analysed dominant parallel computers (NOW, Grid, metacomputer). In relation to it we began applying queuing theory results to existed multiprocessor systems at first as an individual workstation [11]. Then secondly in this article we have been applied queuing theory results to connected multiprocessor systems in NOW (networks of queuing theory systems) or network of NOW networks as massive Grid or metacomputer.

Then such applications of the network queuing theory systems showed paths to a very effective and practical performance analysis tool mainly for the large NOW networks or another massive number of computer networks (metacomputer, Grid). The achieved results we can apply to performance modelling of dominant parallel computers mainly in following cases

- NOW based on workstations (single processors, multiprocessors or multicores)
- Grid (network of NOW networks)
- mixed parallel computers (SMP, NOW, Grid)
- metacomputers (massive Grid etc.).

Now according current trends in parallel computers (SMP, NOW, Grid), based of powerful workstations, we are looking for flexible analytical model that will be supporting both parallel (SMP) and distributed computers (NOW, Grid, metacomputer). In such unified models we would like to study load balancing, inter-process communication (IPC), transport protocols, performance prediction etc. We would also like to analyse

- the role of adaptive routing
- to prove, or to indicate experimentally, the role of the independence assumption, if you are looking for higher moments of delay
- to verify the suggested model also for node limited buffer capacity and for other servicing algorithms than assumed FIFO (First in First out).

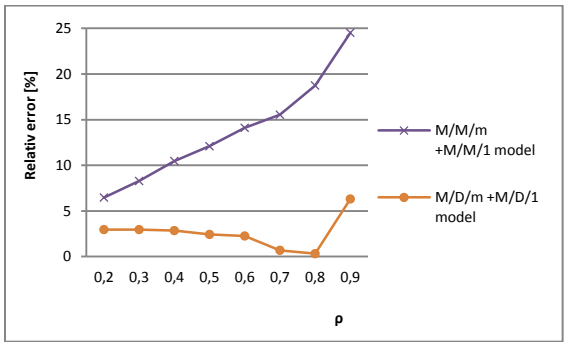

Fig. 5.Relative errors of analysed models.

Primary Paper Section: I

Secondary Paper Section: IN, JD, BA

Literature:

1. Abderazek A. B., *Multicore systems on-chip – Practical Software/Hardware design*, 200 pp., August 2010, Imperial college press
2. Arora S., Barak B., *Computational complexity - a modern approach*, Cambridge University Press, 573 pp., 2009
3. Dattatreya G. R., *Performance analysis of queuing and computer network*, 472 pp., University of Texas, Dallas, USA, 2008
4. Desel J., Esparza J., *Free Choice Petri Nets*, 256 pages, 2005, Cambridge University Press, United Kingdom
5. Dubois M., Annavaram M., Stenstrom P., *Parallel Computer Organisation and Design*, 560 pages, 2012
6. Gelenbe E., *Analysis and synthesis of computer systems*, 324 pages, April 2010, Imperial College Press
7. Hager G., Wellein G., *Introduction to High Performance Computing for Scientists and Engineers*, 356 pages. July 2010
8. Hanuliak J., Hanuliak I.,, *To performance evaluation of distributed parallel algorithms*, Kybernetes, Volume 34, No. 9/10, pp. 1633-1650, 2005, UK
9. Hanuliak P., *Analytical method of performance prediction in parallel algorithms*, The Open Cybernetics and Systemics Journal, July 2012
10. Hanuliak P., Hanuliak I., *Performance evaluation of iterative parallel algorithms*, Kybernetes, Volume 39, No.1, 2010, pp. 107- 126, UK
11. Hanuliak P., *Performance evaluation of SMP parallel computers*, AD ALTA, 2013, submitted
12. Hillston J., *A Compositional Approach to Performance Modelling*, University of Edinburg, 172 pages, 2005, Cambridge University Press, United Kingdom
13. John L. K., Eeckhout L., *Performance evaluation and benchmarking*, CRC Press, 2005
14. Kirk D. B., Hwu W. W., *Programming massively parallel processors*, Morgan Kaufmann, 280 pages, 2010
15. Kostin A., Ilushechkina L., *Modelling and simulation of distributed systems*, 440 pages, Jun 2010, Imperial College Press
16. Lilja D. J., *Measuring Computer Performance*, 280 pages, 2005, University of Minnesota, Cambridge University Press, United Kingdom
17. Kushilevitz E., Nissan N., *Communication Complexity*, 208 pages, 2006, Cambridge University Press, United Kingdom
18. Patterson D. A., Hennessy J. L., *Computer Organization and Design (4th edition)*, Morgan Kaufmann, 914 pages, 2011
19. Peterson L. L., Davie B. C., *Computer networks – a system approach*, Morgan Kaufmann, 920 pages, 2011
20. Resch M. M., *Supercomputers in Grids*, Int. Journal of Grid and HPC, No.1, p 1 - 9, January- March 2009
21. Vaníček, J., Papík, M., Pergl, R. a Vaníček T., *Theoretical principles of informatics* (In Czech), 431 pages, Kernberg Publishing, 2007, Prag, Czech republic
22. Wang L., Jie Wei., Chen J., *Grid Computing: Infrastructure*, Service, and Application, 2009, CRC Press.