Recent Possibilities of Intelligent Agents in Distributed Systems

Marcin Woźniak*
*Institute of Mathematics, Silesian University of Technology,
Kaszubska 23, 44-100 Gliwice, Poland
Email: Marcin.Wozniak@polsl.pl

***Abstract*—The article is to discuss recent advances in various aspects of intelligent agents that perform control functions in workflow management and data processing. Cloud-Computing brings various possibilities of novel approach to data management and efficient computer systems, however the process of distribution must be managed not only to increase efficiency but also lower energy consumption. This is a task for intelligent agents, that can play crucial role in modern computer science. In this article recent possibilities for these type of computer systems are discussed.**

I. Introduction

Development in computational technology sets demands in front of software engineers. The systems and solutions applied in various branches of industry must become more and more intelligent. These solutions are based on intelligent agents that have control functions over systems they manage.

In modern economy we want computers to manage efficiently enough not only to increase the income into the budgets but also to help in environment protection, what means lower energy consumption and more flexible positioning of the controlled systems. All these aspects demand intelligent technology, that will be able to optimize controlled objects. Therefore current research concern two important fields of computer science: Computational Intelligence and Software Engineering. This two, if combined, can give Agents: the intelligent software and systems.

Agents are very important for Cloud-Computing. This service is becoming more popular in the recent years, because of it construction that enables users to use software installed in machines in remote locations. One server can efficiently service many users. The idea for this type of servicing is based on economic calculations. Mainly it is applied in large corporations that have many branches placed in remote locations. Second type of Cloud-Computing is available for everybody in popular services like document sharing, social networks, multimedia streaming and more. First type has mainly economic reasons. For a large corporation it is inefficient to have similar software departments in every branch. Commonly the software is used by some workers in different shifts. Moreover if the corporation is international, time zones play an important role. While users are working with the system in one branch, somewhere else on the other continent the other team is having a break. Therefore proper configuration of a Cloud-Computing increase efficient usage of the software, decrease energy consumption what helps to protect environment and improves budget [1], [2], [3], [4].

Tailored system composition and application of managing agents can improve the overall efficiency [5], [6]. According to research conducted for the European Commission, Cloud-Computing can decrease amount of money spend on IT by 20% and drastically lower energy consumption. Cloud-Computing is usually available for users in one of three ways:

- IaaS (Infrastructure as a Service): special infrastructure with dedicated components.
- PaaS (Platform as a Service): special platform of virtual machines and operation systems.
- SaaS (Software as a Service): users get an access to applications without integrating into the system, which is the most popular model.

However mainly some compositions of the mentioned above are more practical. In this article an idea for intelligent agents managing wrokflows and knowledge distribution will be discussed.

A. Related Works

Intelligent Agents assist in processes where computer systems need proper managing. One can point various aspects of these situations, however the article is to discuss workflow management in various systems i.e. with knowledge retrieval.

Examinations of distributed systems for workflow management can be devoted to proper positioning for request management [7], [8], [9], [10]. Other aspects like intelligent software architectures [11] and efficient management [12] can also improve the overall performance. Since new technology is still introduced all the time, new possibilities that were not available before can be of concern now. Intelligent Agents help in sorting of big data [13], [14]. Application of intelligent technologies helps to increase speed of reasoning over incomplete data and data mining [15], [16]. Image processing also can be improved by advanced software [17], [18], [19], [20].

II. Intelligent Workflow Management

Request processing in Cloud-Computing systems (Fig. 1) is commonly managed by two (or more) separate agents. Each client machine sends a request to the system. These requests are ordered by first managing agent and then proceed to next managing agent. This agent composition has a role of the office, similar to offices in authority or local agencies, where each income is indexed and then send to proper department. Similarly in the agent based system, first managing server is

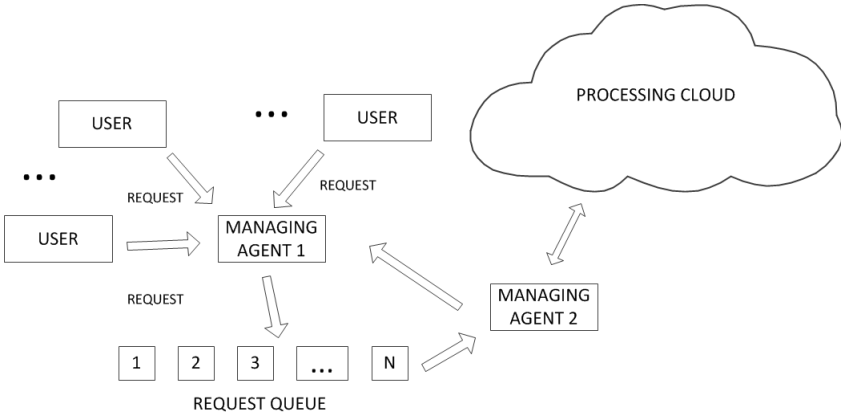

Fig. 1. A common model of requests processing in Cloud-Computing system.

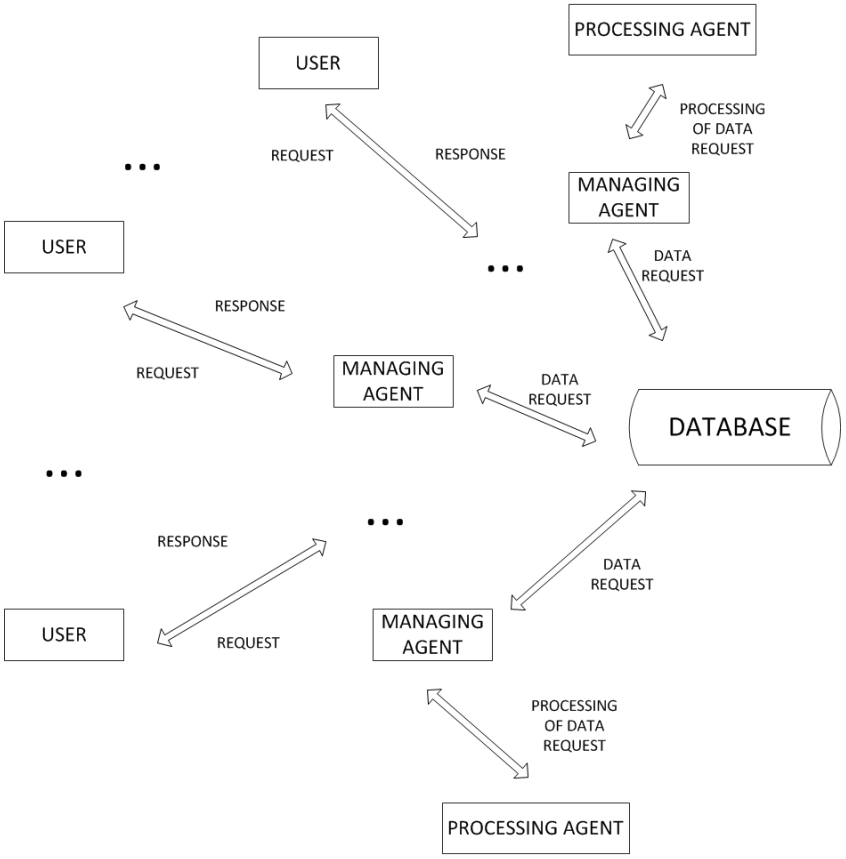

Fig. 2. A common model of knowledge processing for Cloud-Computing system.

indexing incoming requests and then sends them to proper agents responsible for different tasks. These agents proceed indexed incomes using resources or processing units connected to the cloud. These type of Cloud-Computing services is used in various verification systems, where requests go through managing agent to the processing server, which is using connection to database to assists verification. Mainly the system is also equipped in backup server, which is to store data backup. It ensures that any operation in the system is easily erased and the knowledge can be restored if any demand can happen. In Cloud-Computing systems knowledge is stored in a server that via connection with other agents is able to process information (Fig. 2). This processing depends on the requests. Mainly in each type of Cloud-Computing system there is a database (one o more), where all the information is stored. If a user requests information, a signal is addressed to one of managing agents. After confirming the rights to obtain requested data, managing unit starts to proceed. If the agent is asked to provide simple data query the operation is not complicated, from the data base is selected a portion of the data and then it is returned to the user. However if a user requested detailed information or analysis of the data, very often a new process begins. In this case managing agent is sending portion of the data for processing to a dedicated agent which is specialized in this

type of operation. In this time user request is indexed and waits for turn, meanwhile the agent can service another request from the queue. This request has similar service. Simple data query is processed ad-hoc, other is sent with data to processing agent. When processing agent has a result of analysis, it is sent to managing agent. Here according to indexes in the queue it is returned to user. All these types of processing can be modeled to optimize service costs.

A. Service model

Workflow managing is important for Quality of Service (QoS). Service description for cost optimization defines $T_{service}$, T_{income} and $T_{vacation}$, which describe average time of service, average income time and average vacation time (backup, conservation and etc.). See detailed description in [8]. Classical cost structure is considered in [21]. While in [22] and [23] are presented most important aspects of positioning and cost optimization. Various queueing models for applied type of the server are investigated in [24], [25], [26] and [27]. Please see also [28] and [29] for a review of important results on modeling and positioning.

Research on similar objects [30], [31], [32], [33], [9] with it's analytical model for traffic are applied in [8]. In the research a finite-buffer $H_2/M/1/N$-type QS, similar to server traffic modeling functions discussed in [34] and [35] was used. Let it be here presented only a brief description, for details please see [8]. Incoming requests describes 2-order distribution function:

$$F(t) = p_1\big(1 - e^{-\lambda_1 t}\big) + p_2\big(1 - e^{-\lambda_2 t}\big), \quad t > 0, \qquad (1)$$

where $\lambda_i > 0$ for $i = 1, 2$ and $p_1, p_2 \geq 0$. Inter-arrival times are mixed of two exponential distributions with parameters λ_1 and λ_2, which are being "chosen" with probabilities p_1 and p_2. In the system, there are $(N - 1)$ places in queue and one for packet in the service. System starts working at $t = 0$ with at least one packet present. After busy period the server begins vacation which is modeled with 2-order hyper exponential distribution function:

$$V(t) = q_1\big(1 - e^{-\alpha_1 t}\big) + q_2\big(1 - e^{-\alpha_2 t}\big), \quad t > 0. \qquad (2)$$

Interpretation of parameters α_i, $i = 1, 2$ and q_1, q_2 is similar to that for λ_i, $i = 1, 2$ and p_1 and p_2. If at the end of vacation there is no packet present in the system, the server is on standby and waits for first arrival to start service process. If there is at least one packet waiting for service in the buffer at the end of vacation, the service process starts immediately and new busy period begins.

Functions $F(\cdot)$ and $V(\cdot)$ help to simulate inter-arrival times and vacation defined in (1) and (2). In the research optimal values for parameters λ_i, p_i, μ and α_i are calculated to optimize amount of resources to perform all operations. This is modeled in $r_n(c_1)$ defined as:

$$r_n(c_1) = \frac{Q_n(c_1)}{\mathbf{E}_n(c_1)} = \frac{r(\tau_1)\mathbf{E}_n\tau_1 + r(\delta_1)\mathbf{E}_n\delta_1}{\mathbf{E}_n\tau_1 + \mathbf{E}_n\delta_1}, \qquad (3)$$

where the symbols are: $r(\tau_1)$-fixed unit operation costs during busy period τ_1, $r(\delta_1)$-fixed unit operation costs during idle time δ_1, $\mathbf{E}_n\tau_1$-means of busy period τ_1 and $\mathbf{E}_n\delta_1$-idle time δ_1 on condition that system starts with n packets present. In (3) are used means of busy period and vacation (idle) time. The explicit formula with detailed information and description for conditional joint characteristic functions of τ_1, δ_1 and $h(\tau_1)$ is presented in [8] and [36]. General equation to calculate these values is:

$$B_n(s, \varrho, z) = \mathbf{E}\{e^{-s\tau_1 - \varrho\delta_1} z^{h(\tau_1)} \mid X(0) = n\}, 2 \leq n \leq N, \qquad (4)$$

where $s \geq 0$, $\varrho \geq 0$ and $|z| \leq 1$, $n \geq 1$. Details on this equation are discussed in [8], [13] and [36] where using it we can define components of (3) to model total cost of work:

$$\mathbf{E}_n e^{-s\tau_1} = \mathbf{E}\{e^{-s\tau_1} \mid X(0) = n\} = B_n(s, 0, 1), \qquad (5)$$

then for model traffic finally we have:

$$\mathbf{E}_n\tau_1 = -\frac{\partial}{\partial s} B_n(s, 0, 1)\Big|_{s=0}, \qquad (6)$$

similarly we have:

$$\mathbf{E}_n\delta_1 = -\frac{\partial}{\partial \varrho} B_n(0, \varrho, 1)\Big|_{\varrho=0}. \qquad (7)$$

B. Applied Harmony Search Algorithm

Harmony Search Algorithm (HSA) was presented in 2001 by Zong Woo Geem. It's main application is to optimize and position various objects. However the first idea was to apply the HSA to compose artificial music. The method in the beginning was to perform a music by creating tones according to the rules of applied music type. The algorithm is using natural adaptation of sounds to compose tones and finally music theme. It is all based on harmony between sounds that create the music nice in theme for the people. Musician selects the sounds from the scale that best work together. HSA works analogously in the optimization. Each object variable has a specified range. When choosing the best of the possible numbers to have harmony in the process of optimization we get the solution.

HSA can be applied to optimize various objects. If we take parameters values as the sounds that we must adapt to compose a music theme we can have efficient optimization method. HSA will search the space $D = [a_1, b_1] \times \cdots \times [a_n, b_n]; f : D \to R^n$ for optimal values $a_i \leq x_i \leq b_i; i = 1, \ldots, n$ that optimize the object according to fitness condition $f(x_i) \to optimum$. In it's simplicity HSA does not demand any special restriction for $f(\cdot)$ function. It only must be possible to calculate it's value at any point of D. In the HSA we define:

- HM (Harmony Memory) - harmony memory that stores the best harmonies used for music composition:
$$HM = \begin{bmatrix} \mathbf{x_1} = (x_1^1, \ldots, x_1^n) | f(\mathbf{x_1}) \\ \cdots \\ \mathbf{x_{HMS}} = (x_{HMS}^1, \ldots, x_{HMS}^n) | f(\mathbf{x_{HMS}}) \end{bmatrix}$$
As each harmony we understand a set of values representing positioned object. If the new vector of harmony (state of the object) is better than any other among the previous HM vectors, this new vector replaces the worst one in the HM. This procedure is repeated until the stopping criterion is met,
- HMS (Harmony Memory Size) - number of harmonies stored in the memory,

- HMCR (Harmony Memory Considering Rate) - the coefficient of vector choice for memory in the range $(0, 1)$. It helps to decide whether, the new component will be selected from the memory of harmony HM, or will it be the new value of the accepted range of variation variables,
- PAR (Pitch Adjusting Rate) - the tone adjustment factor in the range $(0, 1)$.

Algorithm 1 HSA applied to position workflow traffic

1: Define coefficients: HMS, HMCR, PAR and $generations$–number of harmony search,
2: Dedicated criterion function: lowest cost of operation (3),
3: Create at random initial set HM,
4: t:=0,
5: **while** $t \leq generations$ **do**
6: with probability equal HMCR among all existing harmonies in HM $x_i^j \in \mathbf{x_I^J}$, where $I = 1, \ldots, i, \ldots, HMS$ and $J = 1, \ldots, j, \ldots, n$ compose new harmony vector $\mathbf{x_{new}}$,
7: with a probability equal to the value of PAR change $x_i^j = x_i^j + \alpha$, where $\alpha = b \cdot u$ and $b \in [0.01, 0.001]$ and $u \in [-1, 1]$,
8: with probability equal to 1 - HMCR take randomly new harmony vector $\mathbf{x'_{new}}$ variables $a_i \leq x_i \leq b_i$,
9: **while** $i \leq HMS$ **do**
10: **if** $f(\mathbf{x_{new}})$ is better then f($\mathbf{x}_*$) **then**
11: change $\mathbf{x_{new}}$ with worst harmony vector $\mathbf{x}_*$,
12: **end if**
13: **if** $f(\mathbf{x'_{new}})$ is better then f($\mathbf{x}_*$) **then**
14: change $\mathbf{x'_{new}}$ with worst harmony vector $\mathbf{x}_*$,
15: **end if**
16: **end while**
17: Sort harmonies in HM memory,
18: Next $generation$: $t++$,
19: **end while**
20: Best harmony vector in HM memory is potential optimum.

III. Research Results

Application of HSA to traffic modeling will help to position operation time and optimize service cost $r_n(c_1)$ for system under-load, critical load or overload. HSA simulations were performed for $r(\tau_1) = 0.5$ and $r(\delta_1) = 0.5$. It means that modeled workflow management is simulated for 0.5 energy unit consumption each vacation and work period. These values may be changed in (3). Presented modeling results are averaged of 100 samplings for HMS $= 50$ harmonies in 80 $generations$ with HMCR and PAR taken randomly at each generation, where times in the system have equations:

- Average service time: $T_{service} = \frac{1}{\mu}$,
- Average time between packages income into the system: $T_{income} = \frac{p_1}{\lambda_1} + \frac{p_2}{\lambda_2}$,
- Average vacation time: $T_{vacation} = \frac{q_1}{\alpha_1} + \frac{q_2}{\alpha_2}$,
- Examined system size: N = buffer size +1.

Scenario 1.
In Table I are optimum values for all parameters that affect lowest cost of system operation workflow.

TABLE I. Optimal parameters $\mu, \lambda_i, \alpha_i, p_i, q_i$ for $i = 1, 2$ and lowest value of (3).

λ_1	λ_2	α_1	α_2	p_1	p_2	q_1	q_2
3.1	2.1	1.5	0.3	1.82	1.4	5.8	3.7
μ	0.7			$r_n(c_1)$	0.32		
	$T_{service}$		T_{income}		$T_{vacation}$		
$[sec]$	1.42		1.25		16.2		

Scenario 2.
Request service is set to $T_{service} = 2[sec]$. Research results are shown in Table II.

TABLE II. Optimal parameters $\mu, \lambda_i, \alpha_i, p_i, q_i$ for $i = 1, 2$ and lowest value of (3).

λ_1	λ_2	α_1	α_2	p_1	p_2	q_1	q_2
2.4	3.7	0.6	0.5	64.56	0.91	2.6	13.20
μ	0.45			$r_n(c_1)$	0.43		
	$T_{service}$		T_{income}		$T_{vacation}$		
$[sec]$	2.22		27.14		30.73		

Scenario 3.
Positioning of heavy traffic i.e. when $T_{service} = 0.5[sec]$. Research results with system positioning are shown in Table III.

TABLE III. Optimal parameters $\mu, \lambda_i, \alpha_i, p_i, q_i$ for $i = 1, 2$ and lowest value of (3).

λ_1	λ_2	α_1	α_2	p_1	p_2	q_1	q_2
41.3	25.2	107.3	1.8	2.1	1.4	52.21	15.7
μ	2.34			$r_n(c_1)$	0.27		
	$T_{service}$		T_{income}		$T_{vacation}$		
$[sec]$	0.42		0.1		9.2		

Scenario 4.
If we position peculiar incoming traffic $T_{income} = 2[sec]$. Research results are shown in Table IV.

TABLE IV. Optimal parameters $\mu, \lambda_i, \alpha_i, p_i, q_i$ for $i = 1, 2$ and lowest value of (3).

λ_1	λ_2	α_1	α_2	p_1	p_2	q_1	q_2
3.2	4.8	1.11	1.72	5.73	2.92	14.6	6.71
μ	0.43			$r_n(c_1)$	0.34		
	$T_{service}$		T_{income}		$T_{vacation}$		
$[sec]$	2.32		2.39		17.05		

IV. Final Remarks

In Cloud-Computing or distributed systems where the amount of data to be transfered over the network is large optimal managing can significantly influence workflow and lower traffic. Presented approach to simulation and positioning can be a great advantage for distributed systems. Proposed positioning and modeling will accelerate operations and it is not burdened by typical workflow simulation restrictions.

In the article a novel approach to workflow simulation and modeling is presented. Proposed novel method is easy to implement with possibility to improve. Moreover it can be implemented in Cloud-Computing where data packages influence workflow stability and performance.

REFERENCES

[1] F. Bonanno, G. Capizzi, G. Lo Sciuto, C. Napoli, G. Pappalardo, and E. Tramontana, "A novel cloud-distributed toolbox for optimal energy dispatch management from renewables in igss by using wrnn predictors and gpu parallel solutions," in *Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), 2014 International Symposium on*. IEEE, 2014, pp. 1077–1084.

[2] F. Bonanno, G. Capizzi, and C. Napoli, "Some remarks on the application of rnn and prnn for the charge-discharge simulation of advanced lithium-ions battery energy storage," in *Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), 2012 International Symposium on*. IEEE, 2012, pp. 941–945.

[3] G. Capizzi, F. Bonanno, and C. Napoli, "Recurrent neural network-based control strategy for battery energy storage in generation systems with intermittent renewable energy sources," in *International conference on clean electrical power (ICCEP)*. IEEE, 2011, pp. 336–340.

[4] G. Capizzi, F. Bonanno, and C. Napoli, "A new approach for lead-acid batteries modeling by local cosine," in *Power Electronics Electrical Drives Automation and Motion (SPEEDAM), 2010 International Symposium on*. IEEE, 2010, pp. 1074–1079.

[5] C. Napoli, G. Pappalardo, E. Tramontana, and G. Zappalà, "A cloud-distributed gpu architecture for pattern identification in segmented detectors big-data surveys," *The Computer Journal*, p. bxu147, 2014.

[6] C. Napoli, G. Pappalardo, and E. Tramontana, "Improving files availability for bittorrent using a diffusion model," in *23rd IEEE International WETICE Conference*. IEEE, 2014, pp. 191–196.

[7] M. Gabryel, M. Woźniak, and R. Damaševičius, "An application of differential evolution to positioning queueing systems," *Lecture Notes in Artificial Intelligence - ICAISC'2015*, vol. 9120, pp. 379–390, 2015, DOI: 10.1007/978-3-319-19369-4_34.

[8] M. Woźniak, W. M. Kempa, M. Gabryel, and R. K. Nowicki, "A finite-buffer queue with single vacation policy - analytical study with evolutionary positioning," *International Journal of Applied Mathematics and Computer Science*, vol. 24, no. 4, pp. 887–900, 2014, DOI: 10.2478/amcs-2014-0065.

[9] M. Woźniak, "On positioning traffic in nosql database systems by the use of particle swarm algorithm," in *Proceedings of XV Workshop DAGLI OGGETTI AGLI AGENTI - WOA'2014*, vol. 1260. 25-26 September, Catania, Italy: CEUR Workshop Proceedings (CEUR-WS.org), RWTH Aachen University, 2014.

[10] M. Woźniak, "On applying cuckoo search algorithm to positioning $GI/M/1/N$ finite-buffer queue with a single vacation policy," in *Proceedings of the 12th Mexican International Conference on Artificial Intelligence - MICAI'2013*. 24-30 November, Mexico City, Mexico: IEEE, 2013, pp. 59–64, DOI: 10.1109/MICAI.2013.12.

[11] G. Borowik, M. Woźniak, A. Fornaia, R. Giunta, C. Napoli, G. Pappalardo, and E. Tramontana, "A software architecture assisting workflow executions on cloud resources," *International Journal of Electronics and Telecommunications*, vol. 61, no. 1, pp. 17–23, 2015, DOI: 10.1515/eletel-2015-0002.

[12] G. Ćwikła, A. Sekala, and M. Woźniak, "The expert system supporting design of the manufacturing information acquisition system (MIAS) for production management," *Advanced Materials Research*, vol. 1036, pp. 852–857, 2014, DOI: 10.4028/www.scientific.net/AMR.1036.852.

[13] M. Woźniak, Z. Marszałek, M. Gabryel, and R. K. Nowicki, "Modified merge sort algorithm for large scale data sets," *Lecture Notes in Artificial Intelligence - ICAISC'2013*, vol. 7895, pp. 612–622, 2013, DOI: 10.1007/978-3-642-38610-7_56.

[14] M. Woźniak and Z. Marszałek, *Extended Algorithms for Sorting Large Data Sets*. Poland: Silesian University of Technology Press, 2014.

[15] R. K. Nowicki, B. Nowak, and M. Woźniak, "Rough k nearest neighbours for classification in the case of missing input data," in *Proceedings of the 9th International Conference on Knowledge, Information and Creativity Support Systems*, G. A. Papadopoulos, Ed. 6-8 November, Limassol, Cyprus: University of Cyprus Press, 2014, pp. 196–207.

[16] B. Nowak, R. K. Nowicki, M. Woźniak, and C. Napoli, "Multi-class nearest neighbor classifier for incomplete data handling," *Lecture Notes in Artificial Intelligence - ICAISC'2015*, vol. 9119, pp. 465–476, 2015.

[17] M. Woźniak and Z. Marszałek, "An idea to apply firefly algorithm in 2D images key-points search," *Communications in Computer and Information Science - ICIST'2014*, vol. 465, pp. 312–323, 2014, DOI: 10.1007/978-3-319-11958-8_25.

[18] M. Woźniak and D. Połap, "Basic concept of cuckoo search algorithm for 2D images processing with some research results : An idea to apply cuckoo search algorithm in 2d images key-points search," in *SIGMAP 2014 - Proceedings of the 11th International Conference on Signal Processing and Multimedia Applications, Part of ICETE 2014 - 11th International Joint Conference on e-Business and Telecommunications*. 28-30 August, Vienna, Austria: SciTePress, 2014, pp. 157–164, DOI: 10.5220/0005015801570164.

[19] C. Napoli, G. Pappalardo, E. Tramontana, Z. Marszałek, D. Połap, and M. Woźniak, "Simplified firefly algorithm for 2D image key-points search," in *IEEE SSCI 2014 - 2014 IEEE Symposium Series on Computational Intelligence - CIHLI 2014: 2014 IEEE Symposium on Computational Intelligence for Human-Like Intelligence, Proceedings*. 9-12 December, Orlando, Florida, USA: IEEE, 2014, pp. 118–125, DOI: 10.1109/CIHLI.2014.7013395.

[20] M. Woźniak, D. Połap, M. Gabryel, R. K. Nowicki, C. Napoli, and E. Tramontana, "Can we preprocess 2d images using artificial bee colony?" *Lecture Notes in Artificial Intelligence - ICAISC'2015*, vol. 9119, pp. 660–671, 2015, DOI: 10.1007/978-3-319-19324-3_59.

[21] J. Teghem, "Control of the service process in a queueing system," *Europ. Jour. of Operations Research*, vol. 1, no. 23, pp. 141–158, 1986.

[22] O. Kella, "Optimal control of the vacation scheme in an $M/G/1$ queue," *Operations Research Journal*, vol. 4, no. 38, pp. 724–728, 1990.

[23] R. Lillo, "Optimal operating policy for an $M/G/1$ exhaustive server-vacation model," *Methodology and Computing in Applied Probability*, vol. 2, no. 2, pp. 153–167, 2000.

[24] U. C. Gupta, A. D. Banik, and S. Pathak, "Complete analysis of $MAP/G/1/N$ queue with single (multiple) vacation(s) under limited service discipline," *Journal of Applied Mathematics and Stochastic Analysis*, no. 3, pp. 353–373, 2005.

[25] U. C. Gupta and K. Sikdar, "Computing queue length distributions in $MAP/G/1/N$ queue under single and multiple vacation," *Applied Mathematics and Computation*, vol. 2, no. 174, pp. 1498–1525, 2006.

[26] Z. Niu and Y. Takahashi, "A finite-capacity queue with exhaustive vacation/close-down/setup times and markovian arrival processes," *Queueing Systems*, vol. 1, no. 31, pp. 1–23, 1999.

[27] Z. Niu, T. Shu, and Y. Takahashi, "A vacation queue with setup and close-down times and batch markovian arrival processes," *Performance Evaluation Journal*, vol. 3, no. 54, pp. 225–248, 2003.

[28] H. Takagi, *Queueing Analysis, vol. 1: Vacation and Priority Systems, vol. 2. Finite Systems*. Amsterdam: North-Holland, 1993.

[29] N. Tian and Z. G. Zhang, *Vacation queueing models. Theory and applications*. Berlin, Heidelberg: Springer - Verlag, 2006.

[30] W. M. Kempa, "$GI/G/1/\infty$ batch arrival queuing system with a single exponential vacation," *Mathematical Methods of Operations Research*, vol. 1, no. 69, pp. 81–97, 2009.

[31] W. M. Kempa, "Characteristics of vacation cycle in the batch arrival queuing system with single vacations and exhaustive service," *Int. Jour. of Applied Mathematics*, vol. 4, no. 23, pp. 747–758, 2010.

[32] W. M. Kempa, "Some new results for departure process in the $M^X/G/1$ queuing system with a single vacation and exhaustive service," *Stochastic Analysis and Applications*, vol. 1, no. 28, pp. 26–43, 2009.

[33] W. M. Kempa, "On departure process in the batch arrival queue with single vacation and setup time," *Annales UMCS Informatica*, vol. 1, no. 10, pp. 93–102, 2010.

[34] D. Hongwei, Z. Dongfeng, and Z. Yifan, "Performance analysis of wireless sensor networks of serial transmission mode with vacation on fire prevention," *ICCET'10 IEEE CPS*, pp. 153–155, 2010.

[35] V. Mancuso and S. Alouf, "Analysis of power saving with continuous connectivity," *Comp. Networks*, vol. 56, no. 10, pp. 2481–2493, 2012.

[36] M. Woźniak, W. M. Kempa, M. Gabryel, R. K. Nowicki, and Z. Shao, "On applying evolutionary computation methods to optimization of vacation cycle costs in finite-buffer queue," *Lecture Notes in Artificial Intelligence - ICAISC'2014*, vol. 8467, pp. 480–491, 2014, DOI: 10.1007/978-3-319-07173-2_41.