Computer Science • Vol. 11 • 2010

Aleksander Byrski*, Marek Kisiel-Dorohinicki*,
Marco Carvalho**

A CRISIS MANAGEMENT APPROACH TO MISSION SURVIVABILITY IN COMPUTATIONAL MULTI-AGENT SYSTEMS

In this paper we present a biologically-inspired approach for mission survivability (considered as the capability of fulfilling a task such as computation) that allows the system to be aware of the possible threats or crises that may arise. This approach uses the notion of resources used by living organisms to control their populations. We present the concept of energetic selection in agent-based evolutionary systems as well as the means to manipulate the configuration of the computation according to the crises or user's specific demands.

Keywords: *agent systems, crisis management, soft computing*

UTRZYMANIE KRYTYCZNYCH ZADAŃ OPARTE NA ZARZĄDZANIU KRYZYSOWYM W OBLICZENIOWYCH SYSTEMACH WIELOAGENTOWYCH

W artykule prezentujemy biologicznie inspirowany mechanizm wspomagający utrzymanie krytycznych zadań (tzw. mission survivability) który umożliwia wykrywanie oraz przeciwdziałanie wybranym zagrożeniom. Przedstawione podejście wzorowane jest na wykorzystywaniu przez żywe organizmy zasobów do kontroli populacji. Prezentujemy koncepcje selekcji energetycznej mającej zastosowanie w ewolucyjnych systemach wieloagentowych (EMAS) oraz sposoby konfiguracji obliczenia w celu przeciwdziałania sytuacjom kryzysowym, według preferencji użytkownika.

Słowa kluczowe: *systemy agentowe, zarządzanie kryzysowe, soft computing*

1. Introduction

The distributed and ad hoc nature of tactical environments (military or not) often requires the application of unconventional monitoring and control strategies, capable to adapt to runtime changes in system resources and requirements [12, 13]. In such

* Department of Computer Science, AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Kraków, Poland, {olekb,doroh}@agh.edu.pl
** Institute for Human & Machine Cognition, 40 South Alcaniz Street Pensacola, FL 32502, USA, mcarvalho@ihmc.us

environments, the occurrence of specific hardware or software oriented crises should be envisaged and appropriately handled in order to ensure the survivability of the mission (e.g. completing the computation, simulation, etc.).

To better support the challenging requirements of distributed computations, such as evolutionary or memetic optimization (e.g. EA-based multi-deme algorithms [7]), distributed algorithm implementations can be merged with a multi-agent based approach, where randomized search strategies happen at the agent level, and information sharing occurs through the local interaction between agents. Most often, agents are associated with platforms and services, using explicit message exchange (or environmental and system-level monitoring) to infer the state of peer agents (i.e. nodes and services) [15, 16].

From a Multi-Agent System (MAS) perspective, the challenges for distributed coordination generally include the stability of the system (with respect to an appropriate size of the population of agents), the capability of responding to the emergence situations occurring in the hardware (breakdown of the computing nodes) as well as in the user requirements (e.g. change of the computation parameters). In the context of this work, these situations are treated as crisis conditions that must be effectively mitigated for mission continuity.

In this paper we propose a mechanism to maintain and control distributed MAS-based (or equivalent) coordination algorithms for mission critical system management and optimization. Leveraging a concept of non-renewable resources-driven selection mechanism [8], our crises management framework regulates the energy (computational resource) and time spent for distributed computational tasks to properly accommodate the requirements of the mission, so as to fulfill the needs for globally optimal, accurate, or acceptable solutions for different tasks and operation tempos.

First, we provide a basic review on crisis management for MAS, which is a key reference concept for system stability measure. Then, we introduce an illustrative biologically-inspired analogy based on evolutionary algorithms applied to computational multi-agent systems. Finally, we conclude our work with a brief discussion on some preliminary experimental results.

2. Crises management in MAS

General problems of crises management in MAS are discussed here following the approach presented by Nawarecki, Kisiel-Dorohinicki, and Dobrowolski in [21]. In their work, the authors analytically studied the problem and proposed a formal model of MAS where the monitoring of such a system was considered [18, 10].

A critical situation in MAS is recognized as a particular state or sequence of states that violate or lead to the violation of local (or global) system goals. Local critical situations generally concern a single agent, while global situations involve a group of agents. The emergence of a local crisis may entail the transition to a global crisis, however, the functional characteristics of distributed multi-agent systems tend to mitigate such effects. This phenomenon results directly from the distributed nature of

multi-agent systems. Based on these characteristics, an argument could be made that anti-crisis mechanisms are already incorporated into MAS, however, as we describe in this work, the threat of a global crisis is possible, although it does require specific conditions and mechanisms mechanisms [18].

A crisis among a group of agents is treated in this work as a global crisis due to the similarities of the state description, and the fact that such a crisis must emerge with respect to a partial or side-goal of a system. Hence, two types of critical situations can be distinguished: direct and indirect.

A direct critical situation refers to the threat of loosing operability of the system as a consequence of the unavailability of some of the agents actions. On the other hand, an indirect critical situation is primarily caused by the lack of resources (violation of the appropriate balance) that, in turn, gives a deficit of functionality. The detection of both types of critical situations can be realized by a monitoring sub-system based on individual evaluations that determine the loss of functionality. It can also be achieved by observations of the distribution of some of the resources that are crucial to the agents or system activity.

The above characteristics allow to define general conditions of critical situations management:

1. Possibility of observation (monitoring) of the system state based on individual observations of the agents' states;
2. Adoption of adequate methods to perform a state evaluation in order to achieve the operational criteria for the recognition of critical situations;
3. Availability of appropriate anti-crisis mechanisms.

The degree of realization of the above postulates can be regarded as a determinant of the system immunity against a crisis. As it has been indicated, a flexible MAS may have, by nature, some elements of the anti-crisis mechanisms already implemented, either as part of the agents' algorithms or derived from how the communication or organization of the system (or sub-system) is achieved.

In this paper we focus on multi-agent computational systems. This type of systems brings new possibilities into the world of computation by hybridizing different approaches such as distributed computing, and biological and social inspirations. Moreover, we rely on specific mechanisms designed to deal with certain crises, instead of introducing complex monitoring solutions for MAS.

3. Biological and ecological system survivability

Homeostasis is the property of an open system, especially living organisms, to regulate its internal environment, so as to maintain a stable condition through multiple dynamic equilibrium adjustments controlled by interrelated regulation mechanisms. Complex systems, such as the human body or the biocenosis of living organisms interacting within a specific biotope, must be homeostatic in order to maintain stability and survive. Moreover, these systems must adapt themselves to the changes occurring in the environment [22].

In ecology, the Gaia hypothesis, proposed by James Lovelock, states that the entire mass of living matter on Earth (or any planet with life) functions as a vast organism that actively modifies its environment to produce another one that suits its needs. In this view, the entire planet maintains homeostasis. Whether this sort of system is present on Earth is still open to debate. However, some relatively simple homeostatic mechanisms are generally accepted. For example, when atmospheric carbon dioxide levels rise, plants are able to grow better and thus remove more carbon dioxide from the atmosphere [19].

In biology, human body exhibits homeostatic functions affecting the suitability of our body fluids to sustain life; these include properties like temperature, salinity, acidity (carbon dioxide), and the concentrations of nutrients and wastes (urea, glucose, various ion, oxygen). Since these properties affect the chemical reactions that keep bodies alive, there are built-in physiological mechanisms to maintain them at desirable levels. This control is achieved through functions performed by the various organs in the body (e.g. thermal regulation: the skeletal muscles can shiver to produce heat if the body temperature is too low, chemical regulation: the pancreas produces insulin and glucagon to control blood-sugar concentration). Most of these organs are controlled by hormones secreted from the pituitary gland, which in turn is controlled by the hypothalamus [20].

From the plethora of parameters and interactions, we choose one as inspiration of our regulatory mechanism that helps in dealing with crises in a particular class of multi-agent computational system. That is the introduction of a limited resource in the system called "life energy" [8]. Life energy cannot be directly translated to the resources found in real systems. However, it can be perceived as food and exchanged among the elements of a system where such as resource exists. In biological systems, homeostasis of living creatures can be maintained thanks to the existence of food chains [22].

4. Evolutionary multi-agent system survivability

We consider mission survivability as an ability to continue computation in an Evolutionary Multi-Agent System. The idea of EMAS came from Cetnarowicz [8] and was further enhanced by Byrski and Kisiel-Dorohinicki (see e.g. [1, 2]), and proved to be effective in many difficult single and multi-objective (see e.g. [11, 24]) problems. The system is complex and has many degrees of freedom what poses a serious challenge for optimal configuring according to the specific task.

Encouraged by the promising experimental results Byrski and Schaefer conducted research aimed at proving variant asymptotic features of EMAS in order to legitimize the significant effort that must be put into configuring and running these systems. First results allowed to construct Markovian model of EMAS [6, 23], subsequent research proved the asymptotic guarantee of success for these systems (publication pending).

In Evolutionary Multi-Agent Systems (EMAS), besides interaction mechanisms typical for agent-based systems (such as communication), agents are able to *reproduce* (generate new agents) and may *die* (be eliminated from the system) (Fig. 1). Agent inheritance may be accomplished by an appropriate model of reproduction (with mutation and recombination), which is similar to classical evolutionary algorithms.

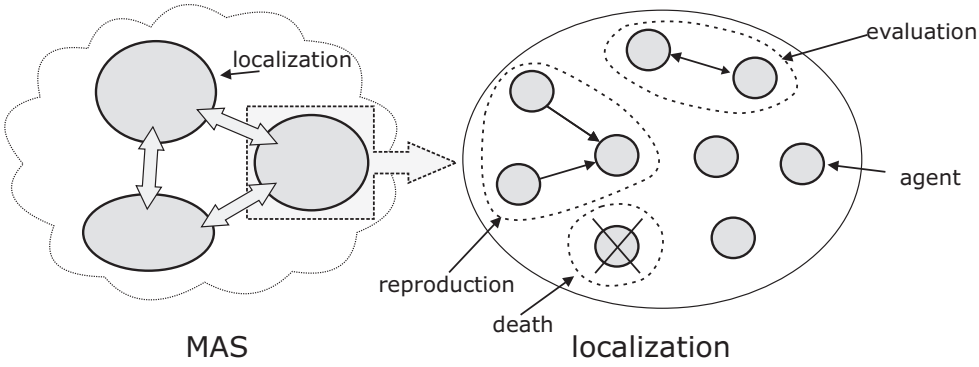


Fig. 1. Structure and behavior of Evolutionary Multi-Agent System (EMAS)

Unfortunately, selection mechanisms from classical evolutionary computation algorithms cannot be used in EMAS because of the assumed lack of global knowledge (which makes it impossible to evaluate all individuals at the same time), and the autonomy of agents (which causes reproduction to be achieved asynchronously). The resource-based (energetic) selection scheme assumes that agents are rewarded for "good" behavior, and penalized for "bad" behavior (which behavior is considered "good" or "bad" depends on the particular problem to be solved) [17].

In the simplest case, the evaluation of an agent (its phenotype) is based on the idea of agent rendezvous. Assuming some neighborhood structure (the simplest case would be population decomposition along with allopatric speciation model [7], see 1) in the environment, agents evaluate their neighbors, and exchange energy. Worse agents (considering their fitness) are forced to transfer a fixed amount of energy to their better neighbors. As the population evolves, this flow of energy allows to obtain agents that represent better approximations of the solution [9].

The limited amount of this energetic resource and the capability of constant exchange allows for achieving homeostasis of the population of agents. Throughout the course of the selection (which is decentralized, not as in classical evolutionary algorithms[14]) the agents exchange their energy during meetings (a better agent – in the means of certain fitness function – takes a portion of the energy of a worse agent). In this way, the energy flows gradually from worse to better agents allowing them to reproduce (while worse agents are compelled to die).

We identify the following crises that may affect the EMAS computation and may be caused by the computation itself:

- overcrowding – there is not a predefined maximum number of agents for the system, and so the system might become overcrowded and affect the performance of the hardware (slowdown, e.g. when solving optimization problems with complex, time consuming fitness functions);
- extinction – the population of agents may become extinct as a result of the death of multiple agents.

Fortunately, these crises can be addressed effectively thanks to the existence of the limited energetic resource:

- when the number of agents rises, their average energy level falls down. Many of the agents in the population end up with lower energy levels, which increases the probability of death. Hence, the energy limit causes the number of agents to decrease in a situation where overcrowding is expected;
- when the number of agents decreases, their average energy level increases. Small populations of agents end up with a higher probability of reproduction (regardless of the quality of the solution acquired by the agents). Hence, the energy limit causes the number of agents to increase in a situation where extinction is expected.

Other critical situations may be caused by a user:

- the need of the user to obtain a sub-optimal result as soon as possible – the mission of the system is to find the solution quickly;
- the need of the user to obtain the most accurate result as possible – the mission of the system is to find the most precise solution.

These critical situations may be handled by modifying the energetic parameters of the selection mechanism:

- when the user needs a solution quickly, the energy transfer rate should be increased to reduce the total number of agents in the population, so the computation can run faster, even though the accuracy may fall;
- when the user needs a precise solution, the energy transfer rate should be decreased, so the accuracy can rise as the problem space will be searched by more agents, even though the computation may take longer.

5. Experimental results

An EMAS was implemented using the AgE platform[1] and applied to the problem of global optimization.

The system consisted of three fully connected subpopulations. In the beginning, 20 agents were placed on each of the subpopulations.

[1] AgE platform `http://age.iisg.agh.edu.pl` is an open-source component-oriented agent-based distributed computation environment developed at AGH University of Science and Technology.

The energetic selection mechanism had the following parameters:

e_0 – the starting energy of every agent is $e_0 = 30$, so the total sum of energy present in the system (regardless of the number of agents) is $e_T = 1800$;

e_{get} – during each rendezvous, agents exchange a predefined amount of energy $e_{get} = 1$;

e_{die} – an agent dies when its energy falls below the level $e_{die} = 0$;

e_{repr} – an agent may reproduce when its energy level exceeds $e_{repr} = \beta_{repr} \cdot \frac{avgen+besten}{2}$ (where *avgen* is the average energy of agents in the subpopulation, *besten* is the energy of the best agent in the population, β_{repr} is certain parameter. In the experiments presented below $\beta_{repr} \in [0.5, 1.5]$);

e_{migr} – an agent may migrate (move to another subpopulation) when its energy exceeds $e_{migr} = 1.2 \cdot e_{repr}$.

Each agent contained the real-value vector (solution of the global optimization problem). The fitness function was computed according to well-known benchmark problems (such as Rastrigin, Schwefel, Griewank e.a.). The results presented below were obtained for the 10-dimensional Rastrigin problem. Variation operators of discrete crossover and uniform mutation with small probability of macro-mutation were used.

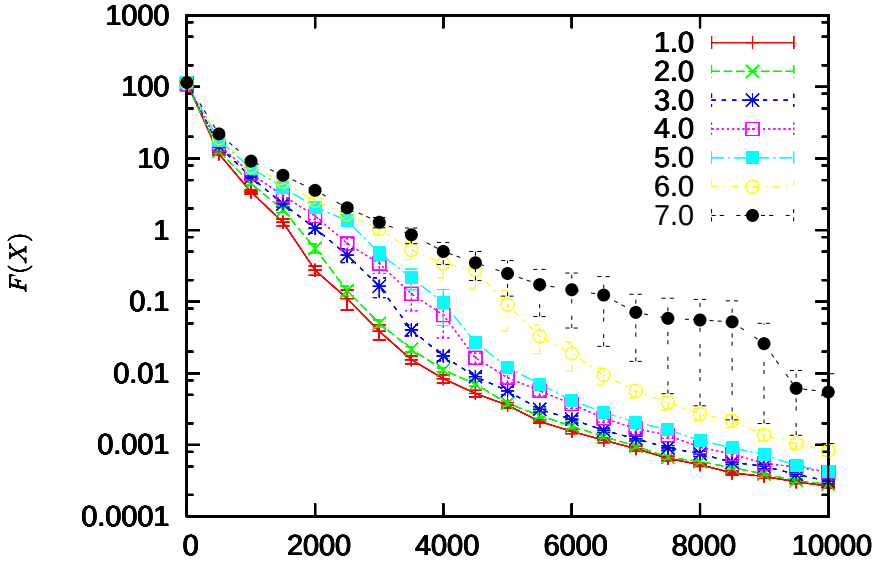
Figures 2–5 show the characteristics of best fitness and agent count for different values of energetic parameters, population dynamics and average energy depending on the step of system's work. The results described next were obtained from 10 repetitions of the experiments (in the graphs standard deviation is shown).

Graphs 2(a) and 2(b) present the best fitness in the population and the number of agents for different values of the energy exchanged during rendezvous. It may be clearly seen that low values of the e_{get} cause better capabilities of localizing the optimum (high values cause the opposite). For all values of e_{get} the population is stable. This parameter seems to be one of the most important parameters of the selection, because it affects the efficiency of the system in a clearly visible way.

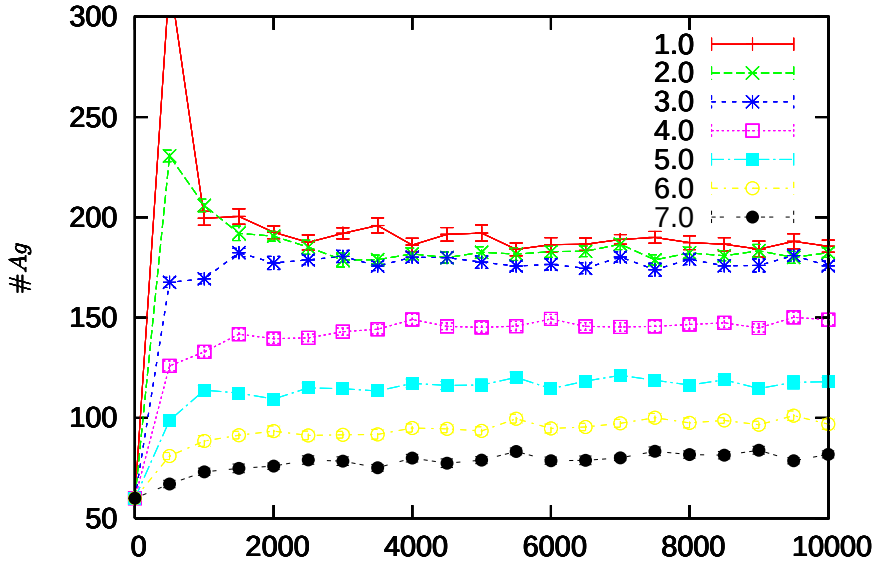
The best fitness in the population and the number of agents for different values of the starting energy is shown in Graphs 3(a) and 3(b). This parameter does not affect the search capabilities and the number of agents in the population, yet the population seems stable for any examined value of e_0.

Another important parameter, β_{repr}, is examined in Graphs 4(a) and 4(b). This parameter affects greatly the search capabilities of the system, because when wrongly chosen (too high) the agents cannot reproduce, so after several rendezvous only one agent remains in the population, carrying all possible energy. Moreover, it seems that there is a certain value of $\beta_{repr} = 1.0$ that turned out to be optimal in our experiments. This value of the parameter allows the system to find the best suboptimal solution.

Graphs 5(a) and 5(b) present the population dynamics – current number of agents in the population, newly created, and removed agents in each of the steps of system's work; and the average energy level in the system, respectively.

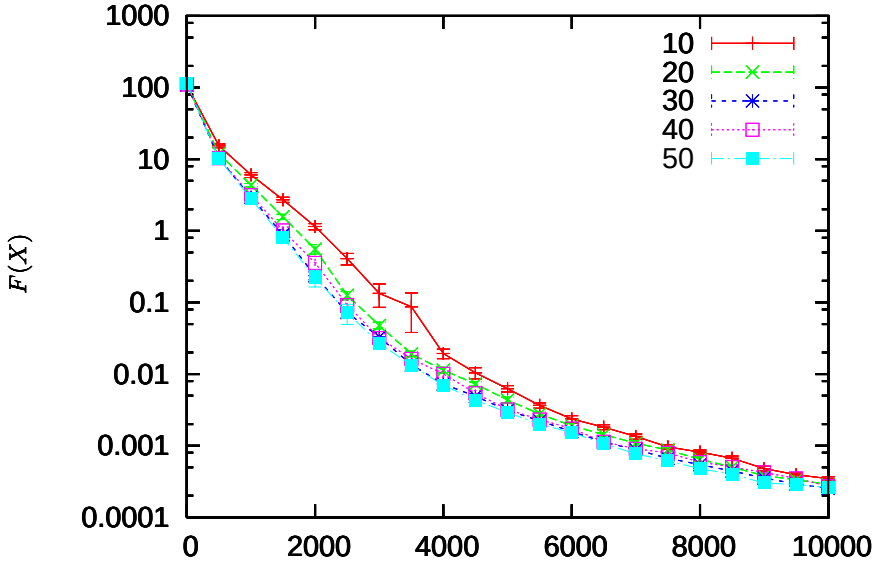


(a) Best fitness

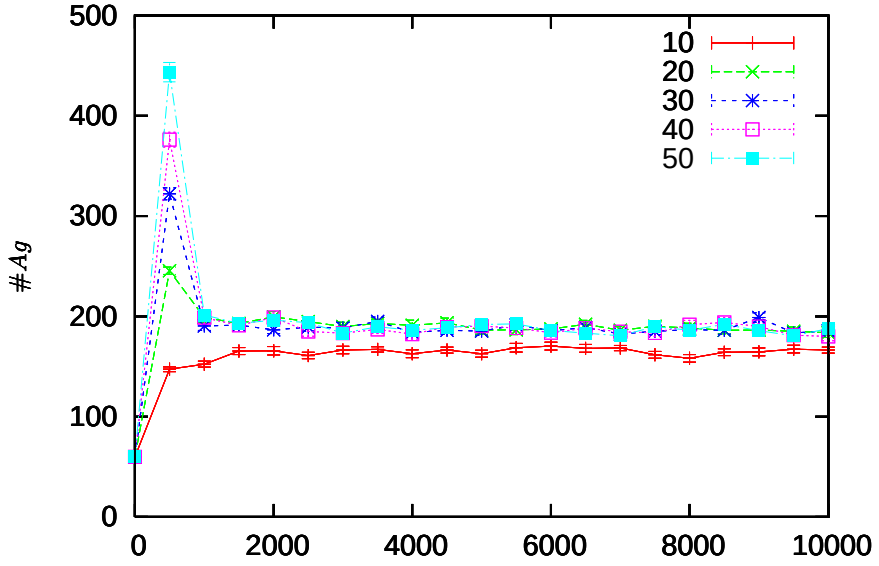


(b) Agent count

Fig. 2. Best fitness and agent count for different values of energy transfer quantum ($e_{get} = 1, \ldots, 7$)

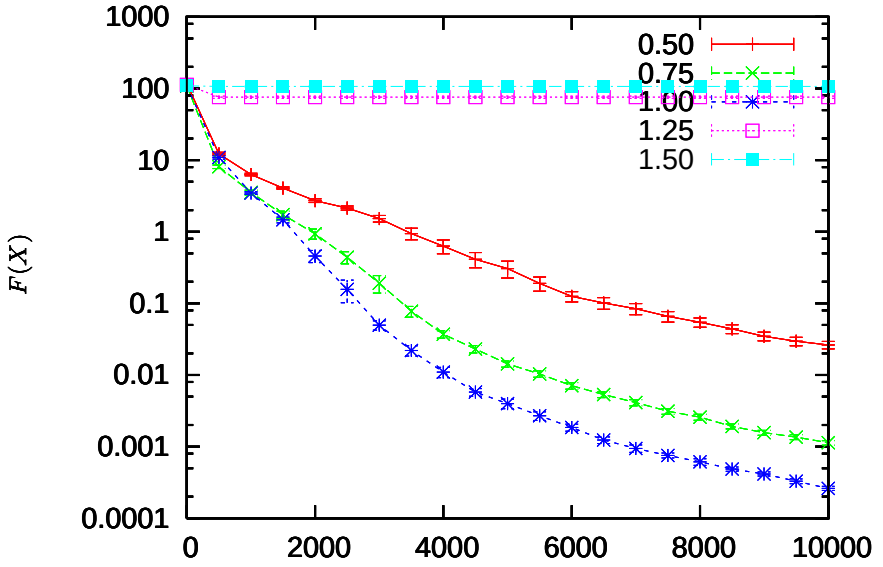


(a) Best fitness

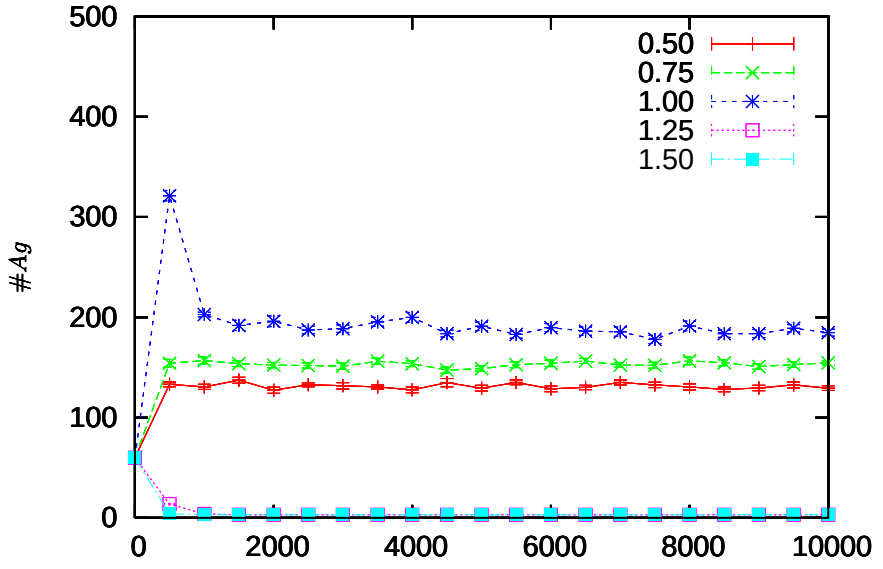


(b) Agent count

Fig. 3. Best fitness and agent count for different values of starting energy ($e_0 = 10, \ldots, 50$)

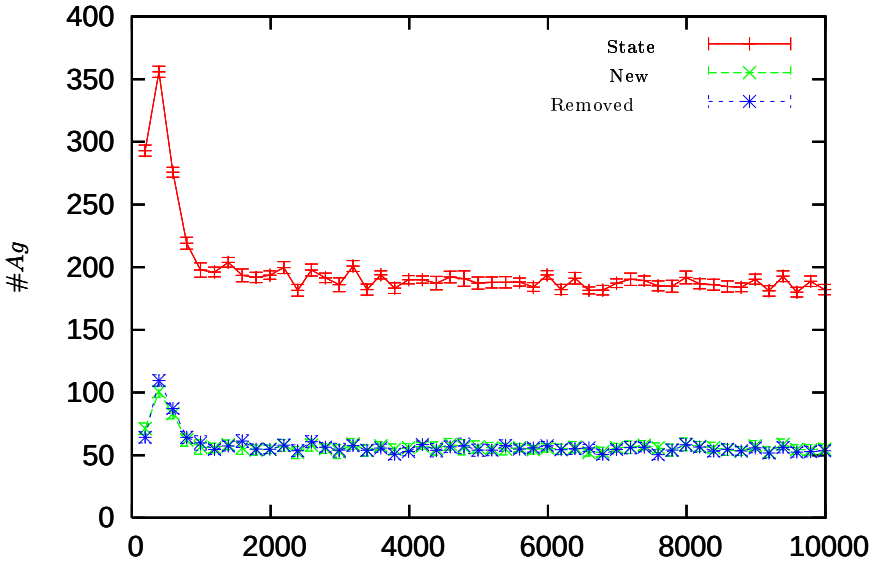


(a) Best fitness

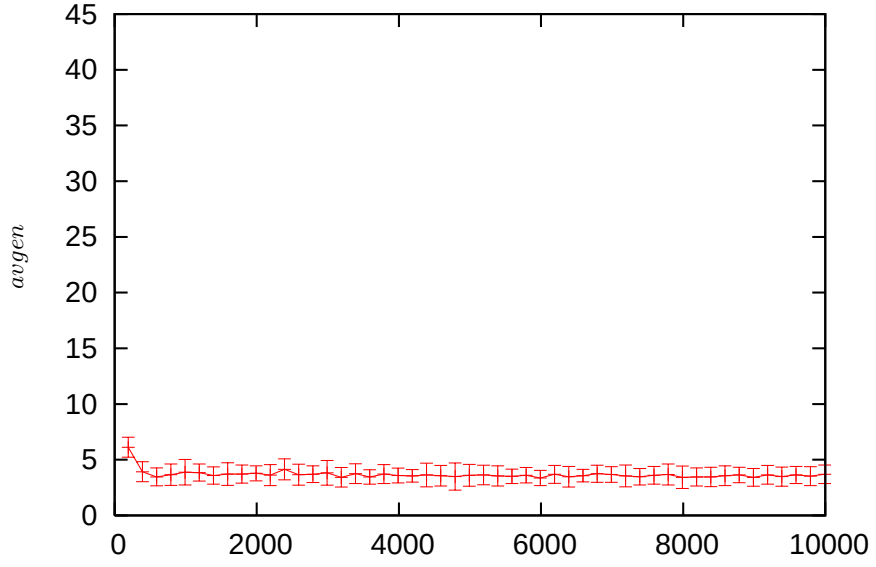


(b) Agent count

Fig. 4. Best fitness and agent count for different values of reproduction energy ($\beta_{repr} = 0.5, \ldots, 1.5$)



(a) Population dynamics



(b) Average energy

Fig. 5. Population dynamics and average energy depending on the step of system's work

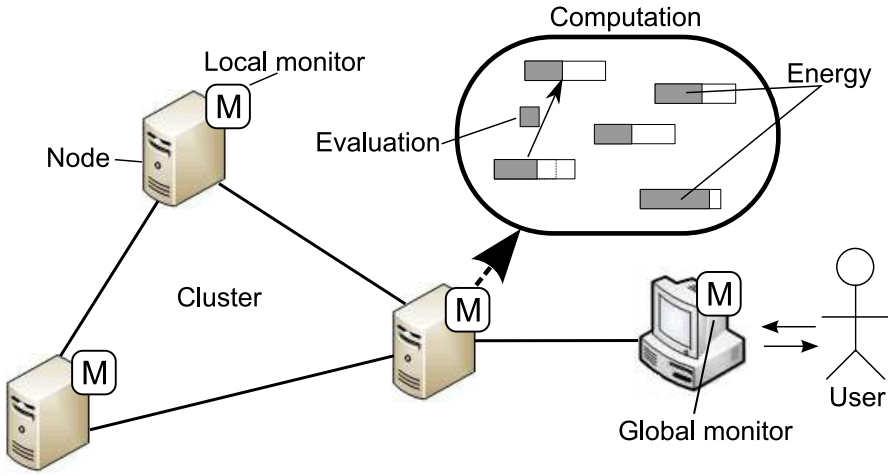


Fig. 6. Monitoring of distributed computational system (EMAS)

It is important to note that the average birth-rate is balanced by the mortality of the agents, which causes the population to be stable. The stable dynamics of the population is caused by the balance in energy levels when energy is distributed among agents.

These preliminary results let to draw some conclusions on the survivability of the computation regardless of the occurrence of the crises mentioned in 4:

- There is a possibility of causing overcrowding and extinction by using the wrong values of the system parameters. For example, in Figure 4(b) the extinction occurs when $\beta_{repr} \geq 1.25$. Such situations should be predicted and prevented, either by the analysis of experimental runs of the system or a formal model of such systems (models that will become capable of such analysis are under construction [4, 5]).
- The need for obtaining sub-optimal results as soon as possible – increasing the value of e_{get} (see Figure 2(b)) reduces the number of agents in the population (the system works faster), but it also degrades the quality of the solutions found (see Figure 2(a)). However, when e_{get} is appropriately chosen, the decrease in the quality of the solutions may be considered acceptable by the user.
- The need for obtaining the most accurate results as possible – decreasing the value of e_{get} (see Figure 2(b)) increases the number of agents in the population (the system works slower), but it also improves the quality of the solutions found (see Figure 2(a)). Again, however, when e_{get} is appropriately chosen, the decrease in the speed of the computation may be considered acceptable by the user.
- Hardware fault and decrease of computing power – the reaction to these crises depends on the configuration of the local and global monitors. A global monitor may help to recover from hardware faults by modifying e_{get} (decreasing its value so the number of agents rises), and a local monitor may dynamically ma-

nipulate e_{get} in a similar way (e.g. increasing it so the number of agents in the subpopulation falls down and the hardware is more efficiently used).

6. Distributed computational systems survivability

The systems under consideration are usually implemented as distributed systems and run on computing clusters (see the structure of EMAS in Figure 6). The subpopulations are distributed among the nodes of the cluster (there may be more than one subpopulation in one node). Every node has a local monitor and these monitors communicate with global monitors in order to gather the results of computation and to react when crises occur. Users may affect the whole computation by changing its parameters (e.g. when mission critical situations occur) using the monitoring system [3]. The global monitor depicted in the Fig. 6 is used solely for technical issues such as observing of performance of particular nodes or logging the information regarding the solutions of the given problem found out by the system, that though being distributed in nature, should deliver its outcome to a particular receiver.

The following crises may be caused by the computation environment:

- hardware fault – one of more cluster nodes are turned off, which has an effect in the decreasing of the subpopulations count – this crisis often cannot be avoided because of its strictly external nature;
- hardware computing power decrease – caused by system features, lack of memory etc., it affects the speed of the computation in one or more subpopulations – this crisis may be caused by overcrowding mentioned in the Section 4.

These types of faults are easily handled because of the autonomy of the system:

- when a portion of the system becomes inoperative, a global monitor may order the recreation of this section in another node (if necessary and when new nodes are available), or it may modify again the energetic parameters of the system to increase the number of agents in the subpopulations. It is important to note that, regardless of the occurrence of this fault, the computation in EMAS will be unaffected because the total energy, though decreased, still remains constant;
- when a portion of the system decreases its computing power, a local monitor may modify the computation only on this node (again, the modification of the energetic parameters seems to be an obvious solution to this problem).

7. Conclusions

In this paper we presented a biologically-inspired, non-renewable resource based selection algorithm that can be used for distributed evolutionary computation in an agent environment. Besides its capabilities that allow for using it in an agent-based environment (with lack of global control), its parameters may be easily used to prevent several threats that may occur in such systems. For example, a system may continue the computation after a hardware failure is detected in a section of the cluster, and

the whole computation will be unaffected (at most some of its parameters may be adapted, e.g. the rate of energy exchange to automatically prevent extinction and overcrowding in this new situation).

Other crises that may occur involves the sudden changes of user's requirements such as the need for obtaining quick sub-optimal results, or the most accurate results as possible. These needs may be addressed by modifying (during system operation) the parameters of the selection algorithm.

The system has also self-sustaining capabilities, e.g. the restricted total energy that is present in the system prevents the population of agents from overcrowding, as well as from extinction (regardless of energetic parameters, in a reasonable range, the population of agents is stable).

In the future we plan to broaden the scope of the tests in a bigger distributed environment (larger clusters or several clusters may be considered). We also plan to generate all identified crisis situations, as well as to look for possible system configurations that can deal with them.

References

[1] Byrski A., Kisiel-Dorohinicki M.: *Immunological selection mechanism in agent-based evolutionary computation.* Proc. of Information Processing and Web Mining IIPWM'05, Gdansk, Poland, 2005.

[2] Byrski A., Kisiel-Dorohinicki M.: *Agent-based evolutionary and immunological optimization.* Proc. of International Conference on Computational Science ICCS 2007, Beijing, China, 2007.

[3] Byrski A., Kisiel-Dorohinicki M.: *User-assisted management of agent-based evolutionary computation.* Proc. of International Conference on Computational Science ICCS 2008, Krakow, Poland, 2008.

[4] Byrski A., Schaefer R.: *Immunological mechanism for asynchronous evolutionary computation boosting.* Proc. of European Workshop on Intelligent Computational Methods and Applied Mathematics ICMAM 2008, Krakow, Poland, 2008.

[5] Byrski A., Schaefer R.: *Formal model for agent-based asynchronous evolutionary computation.* Proc. of IEEE World Congress on Computational Intelligence, Trondheim, Norway, 2009.

[6] Byrski A., Schaefer R.: *Stochastic model of evolutionary and immunological multi-agent systems: Mutually exclusive actions.* Fundamenta Informaticae, vol. 94, 2009.

[7] Cantú-Paz E.: *A summary of research on parallel genetic algorithms.* IlliGAL Report No. 95007. University of Illinois, 1995.

[8] Cetnarowicz K., Kisiel-Dorohinicki M., Nawarecki E.: *The application of evolution process in multi-agent world (MAW) to the prediction system.* Proc. of the 2nd International Conference on Multi-Agent Systems ICMAS'96, Kyoto, Japan, 1996.

[9] Dobrowolski G., Kisiel-Dorohinicki M.: *Management of evolutionary MAS for multiobjective optimization.* Proc. of Evolutionary Methods in Mechanics, 2004.

[10] Dobrowolski G., Kisiel-Dorohinicki M., Nawarecki E.: *Monitoring as a means for discovery of crises in MAS.* Proc. of 12th IFAC symposium on INformation COntrol problems in Manufacturing INCOM, 2006.

[11] Dreżewski R.: *A co-evolutionary multi-agent system for multi-modal function optimization.* Proc. of International Conference on Computational Science, Kraków, Poland, 2004.

[12] Ellison R. J., Fischer D. A. et al.: *Survivable network systems: An emerging discipline.* Technical Report, Carnegie Mellon University, 1999.

[13] Ellison R. J., Goodenough J., et al.: *Survivability assurance for system of systems.* Technical report, Carnegie Mellon University, 2008.

[14] Goldberg D.: *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Professional, 1989.

[15] Jennings N. R., Sycara. K., Wooldrdige M. J.: *A roadmap of agent research and development.* Journal of Autonomous Agents and Multi-Agent Systems, vol. 1, 1998, pp. 7–38.

[16] Jennings N. R., Wooldridge M. J.: *Software agents.* IEE Review, 1996, pp. 17–20.

[17] Kisiel-Dorohinicki M.: *Agent-oriented model of simulated evolution.* Proc. of Theory and Practice of Informatics SofSem 2002, Milovy, Czech Republic, 2002.

[18] Kisiel-Dorohinicki M.: *Monitoring in multi-agent systems: two perspectives.* Monitoring, security and rescue techniques in multiagent systems. Springer Verlag, Advances in Soft Computing, 2005.

[19] Lovelock J. E.: *Gaia as seen through the atmosphere.* Atmospheric Environment, vol. 6, 1972.

[20] Marieb E. N., Hoehn K.: *Human Anatomy & Physiology.* Pearson Benjamin Cummings, 2007.

[21] Nawarecki E., Kisiel-Dorohinicki M., Dobrowolski G.: *Architecture for discovery of crises in MAS.* Fundamenta Informaticae, vol. 71, 2006.

[22] Putman R., Wratten S. D.: *Principles of Ecology.* University of California Press, 1992.

[23] Schaefer R., Byrski A., Smołka M.: *Stochastic model of evolutionary and immunological multi-agent systems: Parallel execution of local actions.* Fundamenta Informaticae, vol. 94, 2009.

[24] Siwik L., Dreżewski R.: *Agent-based multi-objective evolutionary algorithms with cultural and immunological mechanisms.* Evolutionary computation, In-Teh, 2009.