Autonomous Hybridization of Agent-Based Computing

Mateusz Godzik(✉), Michał Idzik, Kamil Pietak, Aleksander Byrski, and Marek Kisiel-Dorohinicki

AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Krakow, Poland
{godzik,miidzik,kpietak,olekb,doroh}@agh.edu.pl

Abstract. Using agent-based systems for computing purposes, where agent becomes not only driver for realizing computing task, but a part of the computing itself is an interesting paradigm allowing for easy yet robust design of metaheuristics, making possible easy parallelization and developing new efficient computing methods. Such methods as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) or Evolutionary Multi Agent-System (EMAS) are examples of such algorithms. In the paper novel approach to hybridization of such computing systems is presented. A number of agents doing their computing task can agree to run other algorithm (similarly to high level hybrid proposed by Talbi). The paper focuses on presenting the background and the idea of such algorithm along with firm experimental results.

Keywords: Agent-based computing · Hybrid metaheuristics · Nature-inspired algorithms.

1 Introduction

Wolpert and MacReady have confirmed [30] the necessity for tuning the existing metaheuristics in order to solve the difficult computing problems. Such tuning usually comprises of choosing the parameters, following different well-known methodologies to make sure the chosen values make the algorithm run really efficacious (cf. iRace [19]). However, adaptation of the algorithms to a greater extent, following e.g. hybridization (cf. Talbi [27]) may allow not only for finding better metaheuristics, but also creating algorithms which will be easily parallelized or run in hybrid environments. In particular, well-researched computing methods, for which particular formal proofs were conducted (e.g. simple genetic algorithm [29] or EMAS [2]), may become a good basis for further hybridizations.

EMAS is present in the state of the art since 1996 [6] and since then many different versions and hybrids of this algorithm were proposed, yielding interesting results [4]. This computing method consists in putting together evolutionary and agent-based computing paradigm, creating a system where agent becomes a

N. T. Nguyen et al. (Eds.): ICCCI 2020, LNAI 12496, pp. 139–151, 2020.
https://doi.org/10.1007/978-3-030-63007-2_11

part of the computing process and undergoes such processes as death and reproduction in order to participate in decentralized selection and produce offspring that will be used for exploring and exploiting the search space. Minding that EMAS was thoroughly analyzed from the theoretical point of view [2], it may be viewed a good starting point for introducing hybrid versions.

Recently new hybrids of EMAS were proposed, comprising the already researched, agent- and evolution-based synergetic metaheuristic with swarm algorithm [21] and Differential Evolution (DE) [13]. Based on the experiences gathered during conducting of this research, a more general approach to hybridization of EMAS was proposed, making the agents responsible for choosing a new metaheuristic, using it for improvement of their solution. Such an autonomous approach for hybridization of metaheuristics is a main contribution of this paper.

In the following sections, after presenting the basic structure and principles of EMAS and related hybrid metaheuristics, the concept of autonomous hybrid agent-based metaheuristic is presented, supported by experimental results and their discussion. Finally the conclusion is given and the future work is sketched out.

2 Evolutionary Multi Agent-Systems

EMAS [7] is metaheuristic which accurateness was proven with a proper formal background, moreover can be treated as interesting and quite effective [2]. Therefore, this algorithm was chosen to solve the problem presented in this article.

Because evolutionary processes are by their very nature decentralized, they can easily be implemented in a multi-agent process at the population level. This means that agents can *reproduce* - in cooperation with other agents or be killed (*die*) as a result of rivalry between agents (selection). A congruous model with narrow autonomy of agents deployed in the planned positions on some lattice (as in a model built from cellular of parallel evolutionary algorithms) was developed by Zhong et al. [33]. However, the decentralized model of evolution in EMAS [15] was created to ensure to give agents full independence.

Such a system is built of a big number of simple and homogeneous agents, each of whom is trying to develop his solution to a common problem. The low computational complexity and the ability to create separate subsystems (subpopulations) means that such systems can be efficiently used in large-scale distributed environments (see, e.g. [3]).

Each agent in EMAS can be seen as a representation of a single solution to a given problem, while the islands on which the agents are located represent a distributed computational structure. The island is a local environment in which agents can interact with each other. However, agents are not trapped on the island - they can change location so that information and resources can be exchanged throughout the entire system [15].

In EMAS, the main evolutionary factors - inheritance and selection - are implemented through agents' activities related to *death* and *reproduction* (see

Fig. 1). Inheritance is the result of a properly defined reproduction - similar to classic evolutionary algorithms. The agent's basic features are recorded in its genotype, which inherits from the parent(s) as a result of mutation and recombination (variation operators). An agent can also gain knowledge (phenotype) during his existence. Such knowledge is not inherited by his descendants, but along with the genotype affects the behavior of the agent. It is worth noting here that it is relatively easy to influence the increase of diversity in EMAS by introducing algorithms such as allotropic speciation (cf. [5]). It introduces the population distribution and allows the agent to move from one evolutionary island to another (migration) (see Fig. 1). Assuming the lack of public knowledge and the automation of agents, a selection mechanism was introduced, which is based on the acquisition and exchange of non-renewable raw materials [7]. As a result, the quality of the solution presented by the agent can be expressed by the number of non-renewable resources owned by it. In other words, the agent gains resources as a result of effective ("good") actions or loses them as a result of wrong decisions ("bad actions"). "Bad" or "good" actions can be understood here as an attempt to find a good enough solution. Based on the amount of resources, selection is carried out - agents with more resources have a better chance to *reproduce*, while those who have gained little will increase their likelihood of death. Following the classic taxonomy of Franklin and Graesser - EMAS agents can be qualified as Artificial Life Agents (this is a kind of Computational Agents) [12].

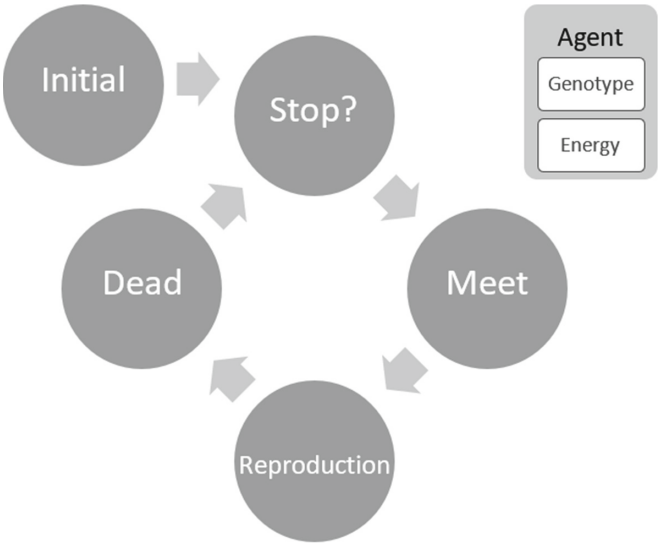


Fig. 1. Evolutionary multi-agent system (EMAS)

Many of the optimization problems that have been solved using EMAS and its versions have yielded better results than some classic approaches. They con-

cern, for example, financial optimization, optimization of multiple goals and optimization of the neural network architecture. In this way, it was proved that in practice EMAS is applicable as a comprehensive optimization tool.

A summary of EMAS-related review has is given in [4]. EMAS can serve as an example of a cultural algorithm in which evolution is possible thanks to the interaction between agents and cultural knowledge is obtained from information related to non-renewable resources (energy). This approach makes it possible to determine which agent is better and which is worse, thus justifying decisions on which genes should remain in the pool. Therefore, knowledge about energy (non-renewable resources) is situational knowledge. Adding an appropriate local-search method to operators, EMAS easily change to memetic version.

3 Hybrid Agent-Based Computing Methods

PSO can be referred as agent-based systems. Their results are obtained by a cooperation of particles (agents), which form a population. There are also many examples of PSO and Genetic Algorithm (GA) hybridization. Basic GA-PSO approach was proposed in [14], where part of population is processed with evolutionary operators and then PSO method is used to adjust output set of individuals. Other GA/PSO solutions include hybrid for solving combinatorial problems, e.g. MPSO [1] (designed to solve Traveling Salesman Problem) or the algorithm for combinatorial vehicle routing optimization problem (VRP) [31]. An overview of PSO hybridizations is presented in [28]. It shows capabilities of PSO combined with multiple approaches like DE, evolutionary programming, ACO, tabu search, simulated annealing, sequential quadratic programming, gradient descend, etc.

PSO hybridization is also present in multi-objective optimization. Standard GA is replaced with MOEA (Multi-Objective Evolutionary Algorithm) and further adjustments are performed. OMOPSO [25] algorithm combines PSO technique, two-layered archive and Pareto dominance relation. Its results are competitive to other classical MOEA approaches. Improvements of this method were proposed in SMPSO [20], including better control of particles' velocity and incorporating polynomial mutation as turbulence factor. There were also attempts to combine PSO and many-objective solutions in order to solve problems with large set of objectives, where classical MOEA approaches are not sufficient. An example of such hybridization is MaOPSO [11], an algorithm using core ideas of well-known many-objective methods, such as NSGAIII.

Another set of agent-based computing methods is ACO. These methods are also very often combined with GA. In [23] an ACO algorithm is used to create an initial population for the subsequent phase, which uses EA operators to develop a solution (HRH). A hybrid incorporating both an ACO and a DE algorithm was proposed in [32]. DE is being run between ACO iterations, optimizing the pheromone trail deposited into the environment (LRH). ACO-EA hybrids for continuous optimization problems have also been designed. As an example, [8] proposes such an algorithm, where ACO_R and CGA_R (Conditionally Breeding Genetic Algorithm) execute their iterations and generations interchangeably, whilst sharing a population.

More generic GA hybridization can be achieved with multi-deme metaheruristics. Hierarchical Genetic Strategy (HGS) introduced in [24] is an example of such model. HGS can be used to divide calculations into tree-like structure. Nodes closer to the tree root are responsible for more general calculations, while leaves evaluate detailed aspects of most promising search space areas. Sprout nodes are added to the tree when satisfactory results are found on parent node. Meanwhile, old or redundant nodes are cut and removed from the tree. HGS was combined with classical multi-objective algorithms (MOHGS [9]). This promising direction was further investigated in [17]. MO-mHGS, improved meta-model is able to connect with any single-deme MOEA algorithm as its driver. New nodes are created basing on progress ratio of popular MOEA quality indicator – hypervolume. In addition, fitness function can be adjusted to speed up calculations on lower tree levels. It was shown that MO-mHGS can significantly improve single-deme algorithm performance. Moreover, HGS model has natural capabilities to be treated as agent-based system and run in parallel environment.

EMAS was hybridized many times and different directions of such endeavors were undertaken. E.g. one of first EMAS hybrids were immunological-EMAS (proposed by Byrski) where the notion of immunological cells introduced among the evolutionary agents was used to remove non-promising individuals, speeding up the whole computing process. Other significant directions of hybridizing emas were co-evolutionary MAS (developed by Drezewski, introducing many sexes, niching and speciation mechanisms, aiming at improving the diversity and solving multi-modal optimiation problems). Elitist-EMAS was proposed by Siwik and was aimed at solving multi-criteria optimization problems by using a notion of elitist evolutionary island inside regular multi-deme population structure of EMAS. Those hybrids were summarized in [4].

Korczynski worked on memetic version EMAS where a dedicated buffering mechanism for fitness evaluation was constructed [16], so high-dimensional optimization problems could be addressed.

Finally, Godzik et al. worked on hybrids of evolutionary algorithms, in particular EMAS with PSO [21] and DE [13]. After abstracting of the mechanism uses for hybridizations in those papers, the higher-level hybridization was considered and a relevant algorithm is a main contribution of this paper.

4 Autonomous Hybrid Agent-Based Metaheuristic

Autonomous hybrid agent-based metaheuristic is a type of modified EMAS algorithm. It consists of the same steps as the base algorithm, except for one additional step. As shown in Fig. 2, an additional hybrid step is placed as the last step of the algorithm loop. In this step, three stages can be distinguished: checking the start condition, running support algorithms (e.g. PSO, DE) and adjusting the energy level of agents using redistribution. The steps are described in detail below.

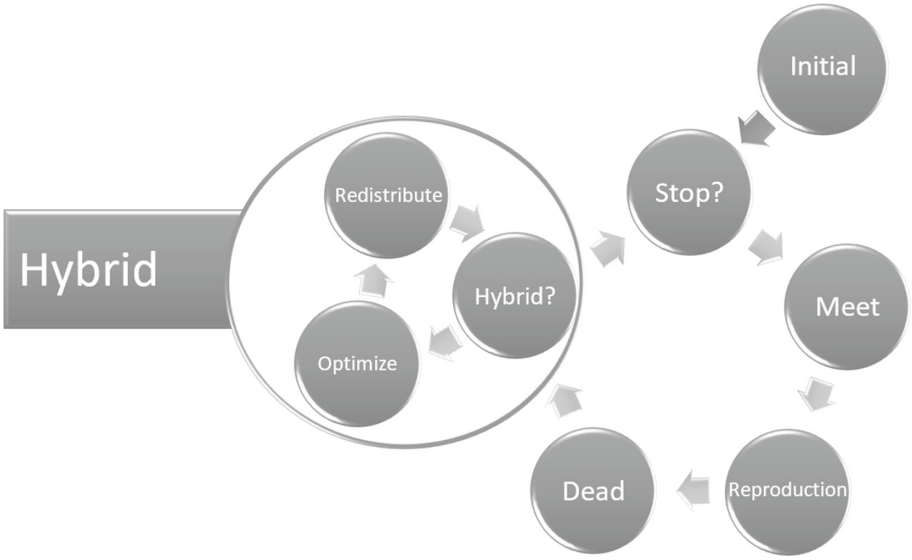


Fig. 2. Hybrid Evolutionary multi-agent system

4.1 Conditions for Running the Optimization

Agents can decide autonomously which optimization they would like to take part in. They may also decide not to participate in any of the optimizations. The terms of these decisions can be any and different for each optimization method and different for each agent. They may depend on the value of energy (e.g. agents with low energy scared to die; agents with average energy wanting to look for another area of solutions; agents with high energy wanting to improve their solution before reproduction). The decision may also depend on the random factor, the agent's life span or calculation time. During their life, agents can freely change their decisions about the desire to optimize.

Because running optimization algorithms is expensive (it consumes both a lot of computation time and calls to the evaluation function), hybrids are not run every algorithm cycle. Sometimes also running a given algorithm does not make sense because of the properties of the algorithm itself. Very often it is not correct to run the algorithm with only one or more willing agents. Therefore, at this stage it is checked if the remaining stages of this step will be started. Launch conditions include the number of cycles since the last launch, variety of solutions or the aforementioned condition of the minimum number of willing agents for a given hybrid. These are just examples, you can try others. Conditions can also be combined with each other (both by conjunction and alternative).

If the start condition is met, the next stage follows (Optimization algorithms). If not, the algorithm goes to the next step (checking the end of the algorithm). In both cases, the agents' decisions are not changed, and if agents do not change them themselves, they will again be considered in the next hybridization step.

4.2 Optimization Algorithms

If the algorithm requires creating new solutions (agents) and killing existing ones, it is worth considering when including such an algorithm. It is recommended to modify the steps of such an algorithm so as not to create/delete agents, but only to replace their solutions. Thanks to this, we will not lose energy and other parameters belonging to agents.

In our article we present the concept of the EMAS algorithm combined with PSO and DE. Previous attempts have already been made to improve EMASA by each of these algorithms separately. The results were promising, which is why we continued this direction of research and combined both ideas into one algorithm in which EMAS agents can choose between many algorithms. For the purposes of this paper, we used two algorithms to be able to examine the impact of this solution on results more easily. In further studies, you can try to use more algorithms. You can also try different algorithms or parameters, e.g. startup frequency or length of operation.

4.3 Redistribution Operator

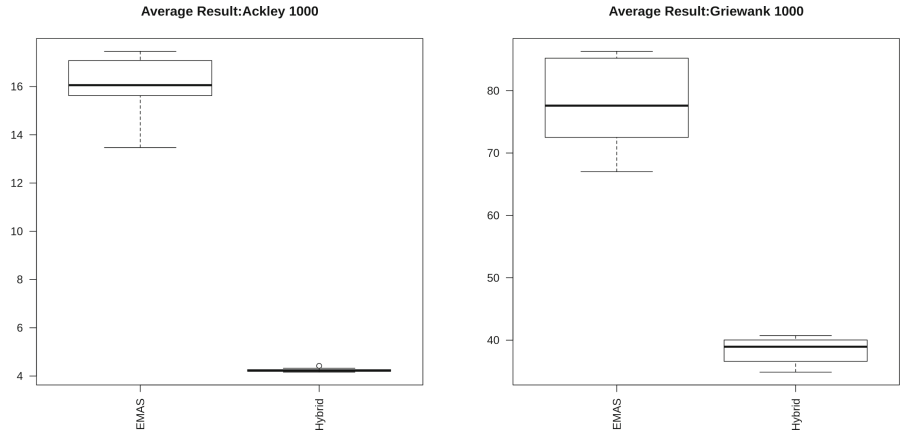
The algorithms used in the hybrid step do not use agent energy. Therefore, after leaving these algorithms, agents have an energy level inadequate to the quality of the solution. To repair this condition, agents leave the energy redistribution operator after leaving the algorithms. You can use different redistribution operators. This article uses the proportional redistribution operator. It sets energy in proportion to agent solutions. Example: we have agent A and B. Agent A has twice the solution than agent B. As a result of the operator's action, agent A will have twice as much energy as agent B. The sum of the agents' energy before and after redistribution is constant.

5 Experimental Results

In this section, the experimental results obtained for EMAS and the proposed hybrid algorithm are presented and discussed.

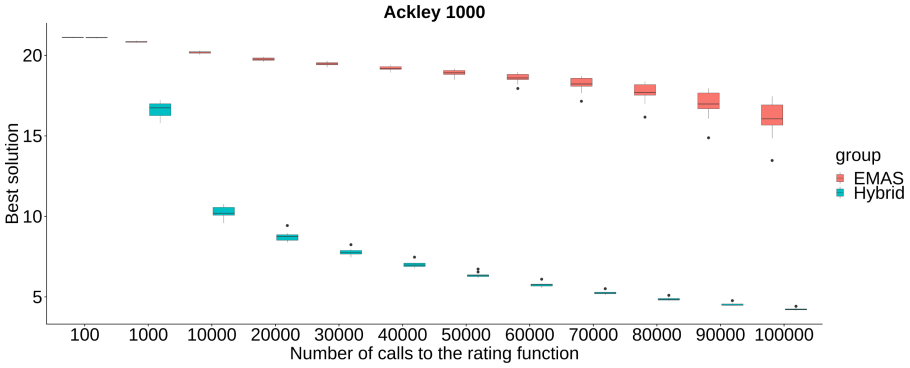
All experiments were realized using a laptop with Intel i7-6700HQ 2.60 GHz processor and 32 GB of RAM. 64-bit Ubuntu system (ver. 18.04). For experiments, the jMetal platform[1] was used. This platform has been modified and improved by dr Leszek Siwik. On this platform a lot of calculations have been made such as [22] and [26]. All algorithms, problem definitions and other components come from this platform and can be found here: https://bitbucket.org/lesiwik/modelowaniesymulacja2018. The tests were realized using jMetal version 5.6 and Java 11.0.4.

[1] jMetal [10] is an object-oriented Java-based framework aimed at the development, experimentation, and study of metaheuristics for solving optimization problems. http://jmetal.github.io/jMetal/.

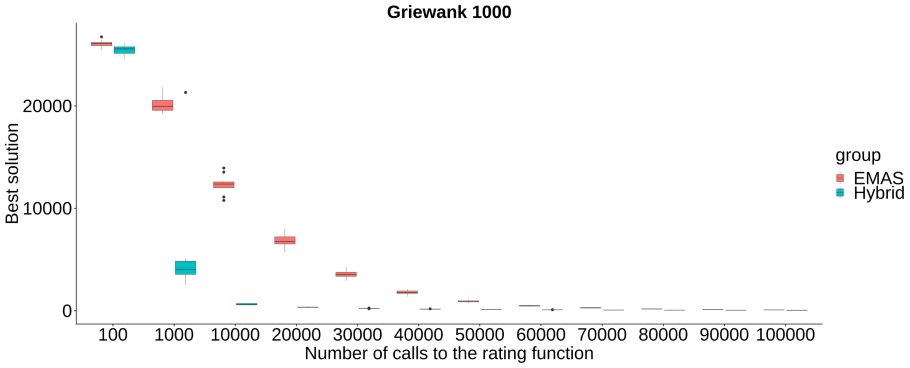


(a) 1000-dimensional Ackley function (b) 1000-dimensional Griewank function

Fig. 3. Best fitness values for the selected benchmark functions.



(a) 1000-dimensional Ackley function



(b) 1000-dimensional Griewank function

Fig. 4. Best fitness values in the time domain for the selected benchmark functions.

5.1 Benchmarks

In order to compare basic algorithm and hybrid was used implementation of problems from jmetal (Ackley, Griewank, Rastrigin) in two sizes (500D and 1000D) [10].

5.2 Configuration

The most important parameters set for the compared systems are bellow:

- EMAS parameters: Population size: 50; Initial energy: 10; Reproduction predicate: energy above 20; Death predicate: energy equal to 0; Crossover operator: SBXCrossover; Mutation operator: PolynomialMutation; Mutation probability: 0.01; Reproduction energy transfer: 10; Fight energy transfer: 1;
- Hybrid parameters: Hybrid predicate: algorithm calls frequency - every 500 EMAS cycles; Redistribution operator: Proportional redistribute operator;
- PSO parameters: Max iterations: 3; Optymalization predicate: energy below 3; Minimal population size: 20;
- DE parameters: Max iterations: 3; Optimilization predicate: energy more than 17; Minimal population size: 20;

For each variant of dimension and algorithm (EMAS or hybrid), optimization tests were carried out 10 times, and the stopping conditions were related to the number of calls of the evaluation function. Namely each experiment could cause a maximum of 100 * size of the problem (for 500 D it was 50,000) calls.

5.3 Discussion of the Results

Each of the charts presented in Fig. 3 consists a box plot describing the best values of an objective function obtained for all individuals in the population after calculations. On both charts left boxplot is for EMAS and right boxplot is from Hybrid algorith results. For both the 1000 dimensional Ackley function and the 1000 dimensional Griewank function the results of the Hybrid algorithm are better than the results of EMAS.

Figure 4 shows the successive stages of obtaining these results. Red color is the results of EMAS, blue of Hybrid algorithm. It can be seen that at the very beginning the values of the drawn solutions are comparable. Differences, however, appear from further stages. Two more things follow from this chart. The first is greater repeatability of hybrid results relative to EMAS. The hybrid achieves a significant improvement at the very beginning of the search, and then systematically improves the result. The quality of the solution can be controlled by extending or shortening the calculation time. The results of given stages and the shape of the function they make up are different for different problems. It depends on the problem and its difficulties.

Results of all problems and median, mean, standard deviation, maximum and minimum of results can be found in Table 1. From the value it contains, it can be seen that for each of the problems presented and its size, the hybrid algorithm finds better solutions. Often it is even an order of magnitude better solution.

Table 1. Results of EMAS and Hybrid algorithm for tested problems.

EMAS					
	Mean	Median	SD	Minimum	Maximum
Griewank 500	36,53959	36,47595	2,465202	32,42908	40,19382
Griewank 1000	77,72416	77,59668	6,799436	67,00974	86,30626
Rastrigin 500	1230,133	1234,579	53,72975	1140,196	1312,001
Rastrigin 1000	2623,715	2624,648	80,49499	2497,007	2739,022
Ackley 500	13,58948	13,70353	2,301543	9,254563	17,67479
Ackley 1000	15,99848	16,05875	1,130237	13,47086	17,46292
Hybrid					
	Mean	Median	SD	Minimum	Maximum
Griewank 500	24,43789	2,52E + 01	2,637139	21,17531	29,13334
Griewank 1000	38,32668	38,94534	2,002169	34,87153	40,72877
Rastrigin 500	879,7538	893,0913	73,14583	715,2792	972,8848
Rastrigin 1000	1622,948	1647,153	73,70861	1497,864	1741,93
Ackley 500	4,636582	4,611668	0,163013	4,330747	4,90088
Ackley 1000	4,240177	4,221788	0,075062	4,160407	4,42008

6 Conclusion

In this paper a concept of autonomous, agent-based hybrid metaheuristic algorithm rooted in EMAS was presented. The agents decide in an autonomous way, in which possible hybridization they would like to participate, and depending on their choice (e.g. based on the level of their energy) their solutions undergo optimization by one of possible hybrid algorithms. In this paper PSO and DE algorithms were used, while it is possible to extend this list so the proposed concept of hybridization is open. One has to remember that after completion of this hybrid step, the computing continues inside EMAS, therefore proper redistribution of energy is required, depending on the quality of the improved solutions.

The experiments conducted, presented in this paper, tackled selected benchmark problems and tested the efficacy of the introduced hybridizations. For three difficult benchmarks (Griewank, Rastrigin and Ackley) set in 500 and 1000 dimensions the experiments showed that the hybrid versions improve significantly the results obtained by classic, non-hybrid ones. This encourages us to further research the proposed metaheuristics and broaden the experiments and delve into detailed testing of different hybridization intrincacies, e.g. the mechanism of redistribution of energy or the mechanism of autonomous decision undertaken by the agents, whether to participate (or not) in a hybrid step.

The proposed metaheuristic can be further investigated by extending research area to multiobjective optimization problems. Standard, EMAS and PSO algorithms can be replaced with MOEA solutions and hybrids. Incorporating MO-

mHGS [18] multi-deme model into multi-agent system environment is also worth considering.

Acknowledgments. The work presented in this paper was supported by Polish National Science Centre PRELUDIUM project no. 2017/25/N/ST6/02841.

References

1. Borna, K., Khezri, R.: A combination of genetic algorithm and particle swarm optimization method for solving traveling salesman problem. Cogent Math. **2**(1) (2015). http://doi.org/10.1080/23311835.2015.1048581
2. Byrski, A., Schaefer, R., Smołka, M.: Asymptotic guarantee of success for multi-agent memetic systems. Bull. Pol. Acad. Sci. Tech. Sci. **61**(1), 257–278 (2013)
3. Byrski, A., Debski, R., Kisiel-Dorohinicki, M.: Agent-based computing in an augmented cloud environment. Comput. Syst. Sci. Eng. **27**(1), 7–18 (2012)
4. Byrski, A., Dreżewski, R., Siwik, L., Kisiel-Dorohinicki, M.: Evolutionary multi-agent systems. Knowl. Eng. Rev. **30**(2), 171–186 (2015). https://doi.org/10.1017/S0269888914000289
5. Cantú-Paz, E.: A summary of research on parallel genetic algorithms. IlliGAL Report No. 95007. University of Illinois (1995)
6. Cetnarowicz, K., Kisiel-Dorohinicki, M., Nawarecki, E.: The application of evolution process in multi-agent world (MAW) to the prediction system. In: Tokoro, M. (ed.) Proceedings of the 2nd International Conference on Multi-Agent Systems (ICMAS 1996). AAAI Press (1996)
7. Cetnarowicz, K., Kisiel-Dorohinicki, M., Nawarecki, E.: The application of evolution process in multi-agent world (MAW) to the prediction system. In: Tokoro, M. (ed.) Proceedings of the 2nd International Conference on Multi-Agent Systems (ICMAS 1996), pp. 26–32. AAAI Press (1996)
8. Chen, Z., Wang, R.: GA and ACO-based hybrid approach for continuous optimization. In: 2015 International Conference on Modeling, Simulation and Applied Mathematics. Atlantis Press (2015). https://doi.org/10.2991/msam-15.2015.81
9. Ciepiela, E., Kocot, J., Siwik, L., Dreżewski, R.: Hierarchical approach to evolutionary multi-objective optimization. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2008. LNCS, vol. 5103, pp. 740–749. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-69389-5_82
10. Durillo, J.J., Nebro, A.J.: jMetal: a Java framework for multi-objective optimization. Adv. Eng. Softw. 42, 760–771 (2011). http://www.sciencedirect.com/science/article/pii/S0965997811001219. https://doi.org/10.1016/j.advengsoft.2011.05.014
11. Figueiredo, E.M., Ludermir, T.B., Bastos-Filho, C.J.: Many objective particle swarm optimization. Inf. Sci. **374**, 115–134 (2016)
12. Franklin, S., Graesser, A.: Is It an agent, or just a program?: a taxonomy for autonomous agents. In: Müller, J.P., Wooldridge, M.J., Jennings, N.R. (eds.) ATAL 1996. LNCS, vol. 1193, pp. 21–35. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0013570. http://dl.acm.org/citation.cfm?id=648203.749270
13. Godzik, M., Grochal, B., Piekarz, J., Sieniawski, M., Byrski, A., Kisiel-Dorohinicki, M.: Differential evolution in agent-based computing. In: Nguyen, N.T., Gaol, F.L., Hong, T.-P., Trawiński, B. (eds.) ACIIDS 2019. LNCS (LNAI), vol. 11432, pp. 228–241. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-14802-7_20

14. Kao, Y.T., Zahara, E.: A hybrid genetic algorithm and particle swarm optimization for multimodal functions. Appl. Soft Comput. **8**(2), 849–857 (2008). http://dx.doi.org/10.1016/j.asoc.2007.07.002
15. Kisiel-Dorohinicki, M.: Agent-oriented model of simulated evolution. In: Grosky, W.I., Plášil, F. (eds.) SOFSEM 2002. LNCS, vol. 2540, pp. 253–261. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36137-5_19
16. Korczynski, W., Byrski, A., Kisiel-Dorohinicki, M.: Buffered local search for efficient memetic agent-based continuous optimization. J. Comput. Sci. **20**, 112–117 (2017). https://doi.org/10.1016/j.jocs.2017.02.001
17. Lazarz, R., Idzik, M., Gadek, K., Gajda-Zagorska, E.: Hierarchic genetic strategy with maturing as a generic tool for multiobjective optimization. J. Comput. Sci. **17**, 249–260 (2016)
18. Lazarz, R., Idzik, M., Gadek, K., Gajda-Zagórska, E.: Hierarchic genetic strategy with maturing as a generic tool for multiobjective optimization. J. Comput. Science **17**, 249–260 (2016). https://doi.org/10.1016/j.jocs.2016.03.004
19. López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L.P., Birattari, M., Stützle, T.: The irace package: Iterated racing for automatic algorithm configuration. Oper. Res. Perspect. **3**, 43–58 (2016). https://doi.org/10.1016/j.orp.2016.09.002. http://www.sciencedirect.com/science/article/pii/S2214716015300270
20. Nebro, A.J., Durillo, J.J., Garcia-Nieto, J., Coello, C.C., Luna, F., Alba, E.: SMPSO: a new PSO-based metaheuristic for multi-objective optimization. In: IEEE symposium on Computational Intelligence in Multi-Criteria Decision-Making, 2009. MCDM 2009, pp. 66–73. IEEE (2009)
21. Placzkiewicz, L., et al.: Hybrid swarm and agent-based evolutionary optimization. In: Shi, Y., et al. (eds.) ICCS 2018. LNCS, vol. 10861, pp. 89–102. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93701-4_7
22. Podsiadło, K., Łoś, M., Siwik, L., Woźniak, M.: An algorithm for tensor product approximation of three-dimensional material data for implicit dynamics simulations. In: Shi, Y., et al. (eds.) Computational Science - ICCS 2018, pp. 156–168. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93701-4_12
23. Rajappa, G.P.: Solving combinatorial optimization problems using genetic algorithms and ant colony optimization. Ph.D. thesis, University of Tennessee (2012). https://trace.tennessee.edu/utk_graddiss/1478
24. Schaefer, R., Kolodziej, J.: Genetic search reinforced by the population hierarchy. Found. Genet. Algorithms **7**, 383–401 (2002)
25. Sierra, M., Coello, C.: Improving PSO-based multi-objective optimization using crowding, mutation and e-dominance. In: Evolutionary Multi-Criterion Optimization, pp. 505–519 (2005)
26. Siwik, L., Los, M., Kisiel-Dorohinicki, M., Byrski, A.: Hybridization of isogeometric finite element method and evolutionary multi-agent system as a tool-set for multiobjective optimization of liquid fossil fuel reserves exploitation with minimizing groundwater contamination. Procedia Comput. Sci. **80**, 792–803 (2016). https://doi.org/10.1016/j.procs.2016.05.369. http://www.sciencedirect.com/science/article/pii/S1877050916308444. International Conference on Computational Science 2016, ICCS 2016, 6–8 June 2016, San Diego, California, USA
27. Talbi, E.G.: A taxonomy of hybrid metaheuristics. J. Heuristics **8**, 541–564 (2002)
28. Thangaraj, R., Pant, M., Abraham, A., Bouvry, P.: Particle swarm optimization: Hybridization perspectives and experimental illustrations. Appl. Math. Comput. **217**(12), 5208–5226 (2011). https://doi.org/10.1016/j.amc.2010.12.053. http://www.sciencedirect.com/science/article/pii/S0096300310012555

29. Vose, M.: The Simple Genetic Algorithm: Foundations and Theory. MIT Press, Cambridge, MA, USA (1998)
30. Wolpert, D., Macready, W.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **67**(1), 67–82 (1997)
31. Xu, S.H., Liu, J.P., Zhang, F.H., Wang, L., Sun, L.J.: A combination of genetic algorithm and particle swarm optimization for vehicle routing problem with time windows. Sensors **15**(9), 21033–21053 (2015). https://doi.org/10.3390/s150921033. http://www.mdpi.com/1424-8220/15/9/21033
32. Zhang, X., Duan, H., Jin, J.: DEACO: hybrid ant colony optimization with differential evolution. In: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2008, 1–6 June 2008, Hong Kong, China, pp. 921–927 (2008). https://doi.org/10.1109/CEC.2008.4630906
33. Zhong, W., Liu, J., Xue, M., Jiao, L.: A multiagent genetic algorithm for global numerical optimization. IEEE Trans. Syst. Man Cybern. Part B Cybern. **34**(2), 1128–1141 (2004)