Agent-based Evolutionary Multiobjective Optimisation

Krzysztof Socha
Free University of Brussles, Belgium
e-mail: krzysztof.socha@ulb.ac.be

Marek Kisiel-Dorohinicki
Department of Computer Science
University of Mining and Metallurgy, Kraków, Poland
e-mail: doroh@agh.edu.pl

Abstract - This work presents a new evolutionary approach to searching for a global solution (in the Pareto sense) to multiobjective optimisation problem. Novelty of the method proposed consists in the application of an evolutionary multi-agent system (EMAS) instead of classical evolutionary algorithms. Decentralisation of the evolution process in EMAS allows for intensive exploration of the search space, and the introduced mechanism of *crowd* allows for effective approximation of the whole Pareto frontier. In the paper the technique is described as well as preliminary experimental results are reported.

Keywords— **multiobjective optimisation, evolutionary computation, multi-agent systems.**

I. Introduction

Although both evolutionary computation and agent technology gained a lot of interest during the last decade, many aspects of their functionality still remain open. The problems become even more complicated considering systems utilising both evolutionary and agent paradigms. Building and applying such systems may be a thorny task but it often opens new possibilities for solving difficult kinds of problems. Also, as for other hybrid systems, one approach may help another in attaining its own goals. This is the case when an evolutionary algorithm is used by an agent to aid the realisation of some of its tasks, e.g. connected with learning or reasoning [13], or to support coordination of some group (team) activity, e.g. planning [10].

An evolutionary multi-agent system (EMAS) is an example of the opposite case, where a multi-agent system (MAS) helps evolutionary computation providing mechanisms allowing for the decentralisation of the solving process (evolution). The key idea of EMAS is the incorporation of evolutionary processes into MAS at a population level. It means that 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). A decisive factor of the agent's activity is its fitness, expressed by the amount of possessed non-renewable resource called *life energy*. Selection is realised in such a way that agents with high energy are more likely to reproduce, whereas a low level of energy increases possibility of death. In fact *all* decisions about actions to be performed (including death and reproduction) are made autonomously by agents, and thus EMAS may be considered as a computational technique utilising a *decentralised* model of evolution, unlike (extending) classical evolutionary computation [3]. What is more, since agents usually operate in some (virtual) space and their interactions (e.g. selection) are limited to their close neighbourhood, this model is also *distributed* like in parallel evolutionary algorithms.

Based on this model a new evolutionary approach to searching for a global solution (in the Pareto sense) to multiobjective optimisation problem may be proposed. In this particular case each agent represents a feasible solution to a given optimisation problem. By means of communication agents acquire information, which allows for the determination of the (non-)domination relation with respect to the others. Then dominated agents transfer a fixed amount of life energy to their dominants. This way non-dominated agents (representing successive approximations of the Pareto set) gain more life energy and reproduce, while dominated agents die. Additionally, the introduction of the mechanism of *crowd* allows for a uniform sampling of the whole frontier [5].

Below a more detailed description of these ideas and their implementation is presented. Preliminary experimental results show the influence of the crowding factor on the performance of the system applied to selected test problems.

II. Evolutionary techniques of multiobjective optimisation

Decision making and lots of other tasks of human activity described by many non-comparable factors may be math-

ematically formulated as multiobjective optimisation problems. The terms ”multiobjective” or ”multicriteria” indicate that a classical notion of optimality becomes ambiguous since decisions which optimise one criterion need not optimise the others. The notion of Pareto-optimality is based on domination of solutions (which corresponds to the weak-order of vectors in the evaluation space) and in a general case leads to the selection of multiple alternatives.

The shape of the multiobjective optimisation problem may be described as follows. Let the input variables be represented by a real-valued vector:

$$\vec{x} = [x_1, x_2, \ldots, x_N]^T \in \mathbb{R}^N \tag{1}$$

where N gives the number of variables. Then a subset of $\mathbb{R}^N$ of all possible (feasible) decision alternatives (options) can be defined by a system of:

- inequalities (constraints): $g_k(\vec{x}) \geq 0$, $k = 1, 2, \ldots, K$,
- equalities (bounds): $h_l(\vec{x}) = 0$, $l = 1, 2, \ldots, L$

and denoted by D. The alternatives are evaluated by a system of M functions (outcomes) denoted here by vector $F = [f_1, f_2, \ldots, f_M]^T$:

$$f_m : \mathbb{R}^N \to \mathbb{R}, \quad m = 1, 2, \ldots, M \tag{2}$$

The key issue of optimality in the Pareto sense is the relation of domination. Alternative $\vec{x}^a$ is dominated by $\vec{x}^b$ if and only if:

$$\forall m \;\; f_m(\vec{x}^a) \leq f_m(\vec{x}^b) \;\text{ and }\; \exists m \;\; f_m(\vec{x}^a) < f_m(\vec{x}^b) \tag{3}$$

The relation of domination corresponds to the weak-order of vectors in the evaluation space (given by values of F).

A solution to such-defined multiobjective optimisation problem (in the Pareto sense) means determination of all non-dominated alternatives from D – *the Pareto set* or *Pareto frontier*. In a general case (i.e. when no particular class of objective and constraint functions is considered) effective approximation of the Pareto set is hard to obtain. For specific types of criteria and constraints (e.g. of linear type) some methods are known, but even in low-dimensional cases they need much computational effort. For complex problems, involving multimodal or discontinuous criteria, disjoint feasible spaces, noisy function evaluation, etc. evolutionary approach (e.g. a genetic algorithm) may be applied.

Probably the first ideas concerning application of evolutionary algorithms to multicriteria optimisation problems came from independent work of Schaffer [14] and Fourman [7]. In the first case (*VEGA – Vector Evaluated Genetic Algorithm*) selection was realised in separate sub-populations on the basis of particular objective functions, while variation operators were applied to the whole population. In the second case selection was based on a tournament, where the objectives were assigned priorities (*lexicographic ordering*). These early approaches did not use directly the information about domination relation between solutions. Conversely, in most later presented methods some sort of ranking was used (e.g. [8], [6]), reflecting the ”degree” of domination of particular solutions. This allowed for better approximation of the Pareto frontier, especially when supported with some niching techniques (like fitness sharing), which prevented genetic drift and enabled sampling of the whole frontier. A detailed information on evolutionary multicriteria optimisation techniques may be found in [6] or [4].

III. Evolutionary multi-agent systems

While different forms of classical evolutionary computation use specific representation, variation operators, and selection scheme, they all employ a similar model of evolution – they work on a given number of data structures (population) and repeat the same cycle of processing (generation) consisting of the selection of parents and production of offspring using variation operators. Yet this model of evolution is much simplified and lacks many important features observed in organic evolution [1], e.g.:

- dynamically changing environmental conditions,
- many criteria in consideration,
- neither global knowledge nor generational synchronisation assumed,
- co-evolution of species,
- evolving genotype-phenotype mapping.

This may limit development of more and more complicated applications, especially that theory of evolutionary algorithms is not mature enough to clearly indicate the way any particular problem should be solved. To avoid at least some of these shortcomings many variations of classical evolutionary algorithms were proposed, introducing e.g. some population structure (like in *parallel evolutionary algorithms*) or specialised selection mechanisms (like *fitness sharing*). The idea of decentralised evolutionary computation realised as an *evolutionary multi-agent system* (EMAS) covers various specialised techniques in one coherent model as described below.

Following neodarwinian paradigms, two main components of the process of evolution are *inheritance* (with random changes of genetic information by means of mutation and recombination) and *selection*. They are realised by the phenomena of death and reproduction, which may be easily modelled as actions executed by agents:

1. action of *death* results in the elimination of the agent from the system,
2. action of *reproduction* is simply the production of a new agent from its parent(s).

One should notice that in an agent-based system there is no global control and no global information available [11]. In consequence all actions are performed asynchronously, as well as agents may have only incomplete and/or uncertain knowledge about the environment and other agents. At the same time agents make autonomous decisions about actions to be performed (also actions of death and reproduction). It means that the only way to achieve a population level goal – find a solution(s) to a given problem – is to adequately design the agent's behaviour.

The first component of evolutionary processes – inheritance – is to be accomplished by an appropriate definition of reproduction, which is similar to classical evolutionary algorithms. The set of parameters describing core properties of an agent (genotype) is inherited from its parent(s) – with the use of mutation and recombination. Besides, agents may possess some knowledge acquired during their life, which is not inherited. Both the inherited and acquired information determines the agent's behaviour in the system (phenotype). It rarely happens in classical evolutionary computation that inherited information does not determine the grown-up individual. If so (e.g. in evolutionary neural networks), each individual has to grow up in one step of an evolutionary algorithm, before it is evaluated. Indeed this is not a must in EMAS.

Selection is the most important and most difficult element of the model of evolution employed in EMAS. This is due to assumed lack of global knowledge (which makes it impossible to evaluate all individuals at the same time) and autonomy of agents (which causes that reproduction is achieved asynchronously). In such a situation selection mechanisms known from classical evolutionary computation cannot be used. The proposed principle of selection corresponds to its natural prototype and is based on the existence of non-renewable resource called *life energy*. The energy is gained and lost when the agent executes actions in the environment. Increase in energy is a reward for 'good' behaviour of the agent, decrease – a penalty for 'bad' behaviour (which behaviour is considered 'good' or 'bad' depends on the particular problem to be solved). At the same time the level of energy determines actions the agent is able to execute. In particular low energy level should increase possibility of death and high energy level should increase possibility of reproduction. Life energy proves to be a very comfortable tool for management of the number of agents in the population [2], as well as for realisation of niching mechanisms similar to fitness sharing or crowding (as shown below).

Like in most agent systems, agents of EMAS operate in the distributed environment and their interactions are limited to their close neighbourhood. This implies geographical population structure and allows for local selection. The topology of the living space (structure of the environment) may be fine- or coarse-grained, which is similar to diffusion and migration models of parallel evolutionary algorithms. Of course, agents may be able to move in the space, though migration may be reduced due to energetic cost of the action.

A formal description of this model and discussion of its advantages may be found in [3], [5], and other. In short, EMAS should enable the following:

- local selection allows for intensive exploration of the search space, which is similar to parallel evolutionary algorithms,
- the way phenotype (behaviour of the agent) is developed from genotype (inherited information) depends on its interaction with the environment,
- self-adaptation of the population size is possible when appropriate selection mechanisms are used.

What is more, explicitly defined living space facilitates implementation in a distributed computational environment.

IV. Flow of life energy in EMAS for multiobjective optimisation

The particular EMAS should search for a set of points which constitute the approximation of the Pareto frontier for a given multicriteria optimisation problem. The population of agents represents feasible solutions to the problem defined by a set of objective functions (each agent represents a particular solution and knows its quality with respect to each criteria). The agents act according to the above-described rules of EMAS operation. The information about the solution is inherited during reproduction – in fact this is the only component of the agent's genotype and thus the crucial element of the whole process.

Another important element of the process is the realisation of energetic reward/punishment mechanism, which should prefer non-dominated agents. At the beginning of the evolution process the initial population of agents is randomly generated and *life energy* is evenly distributed to all of them. Since this initial population is usually small comparing to its maximal possible size, the agents have a lot of life energy, which allows them to reproduce a lot of offspring (contribution to the genotype of the offspring requires also contribution in the form of life energy). At the same time they start to exchange energy based on the information about their solutions and their quality. The flow of energy between the existing agents may follow two distinctive principles. This either could be done via:

- *domination energy transfer principle* (in short: *domination principle*), or
- *crowd energy transfer principle* (in short: *crowd principle*).

Additional to these two principles there is also a flow of life energy during reproduction. Each reproduction operation

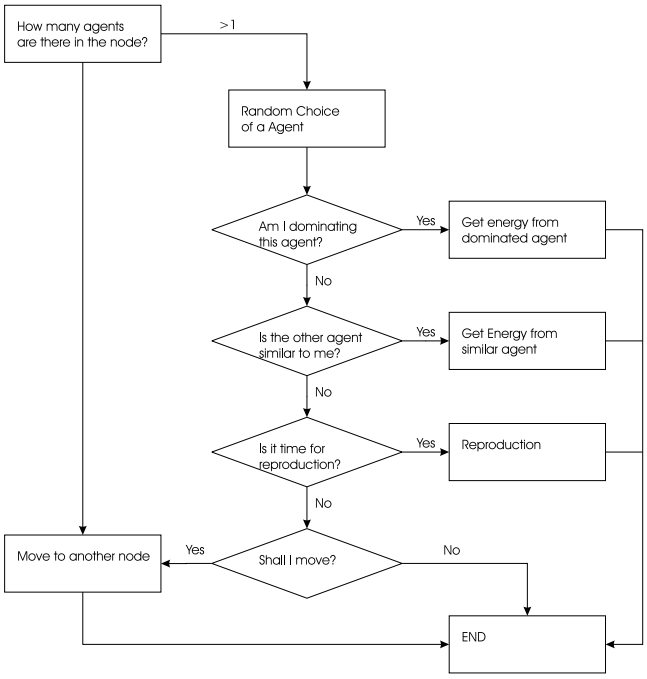

Fig. 1. The algorithm of agent's decision making

requires participation of two agents and is possible only when the total life energy of both agents (let us call them agents A and B) is greater than the triple value of the minimal energy parameter (e_{min}):

$$e_A + e_B \geq 3 \cdot e_{min} \Longrightarrow reproduction \tag{4}$$

Thus only agents possessing enough life energy are allowed to participate in the reproduction process.

The *domination principle* works by forcing dominated agents to give a fixed amount of their energy to the encountered dominants. This may happen, when two agents inhabiting one place communicate with each other and obtain information about their quality with respect to each objective function. The actual algorithm of operation of this principle is following:

1. one of the agents (agent A) initiates the communication by requesting the quality of the solution (with respect to each criteria) from another agent (agent B);
2. agent B presents the quality of its solution to the problem to agent A;
3. agent A compares the quality of its own solution with the one obtained;
4. if the solution of agent B is dominated (in the Pareto sense) by the solution of agent A, agent A will receive some energy from agent B.

Depending on the amount of energy that agent B has, it will be either amount of e_{min} (a parameter describing minimal amount of energy that could be transferred between the agents) or – if agent B has less energy than e_{min} – all its energy (in that case agent B dies):

$$\Delta e = \begin{cases} e_B & \text{if } e_B \leq e_{min} \\ e_{min} & \text{if } e_B > e_{min} \end{cases} \tag{5}$$

where: Δe – the amount of energy transferred,
e_B – initial amount of energy owned by agent B,
e_{min} – minimal amount of energy that could be transferred between the agents.

The flow of energy connected with the *domination principle* causes that dominating agents are more likely to reproduce, whereas dominated ones are more likely to die. This way, in successive generations, non-dominated agents should make up better approximations of the Pareto frontier.

The *crowd principle* has a similar mechanism of operation. There are however some important differences. It is not the quality of the solutions that counts; this time it is important how similar the solutions are. Also, the amount of energy transferred between the agents is not fixed. It is actually calculated by measuring the level of similarity between the communicating agents and comparing it to the parameter called *crowding factor*. The algorithm of operation of this mechanism is following:

1. one of the agents (agent A) initiates the communication by requesting the solution from another agent (agent B);
2. agent B presents its solution to the problem to agent A;
3. agent A then compares its solution to the one obtained and calculates the similarity level of the two solutions described as a distance (in square metric) between the two solutions:

$$d(\vec{x}^A, \vec{x}^B) = \sum_{i=1}^{N} |x_i^A - x_i^B| \tag{6}$$

where: N – number of dimensions of the problem,
x_i^A – i-th coefficient of the solution owned by agent A,
x_i^B – i-th coefficient of the solution owned by agent B.

4. agent A checks if the other solution is to be considered similar, i.e. if the distance computed in the previous step is less than the crowding factor – if so, agent A receives some energy from agent B.

The amount of energy to be transferred is calculated as the difference between the amount of energy that agent B had before the beginning of the communication and this energy multiplied by squared distance (calculated in the previous step) and divided by squared crowding factor:

$$\Delta e = \begin{cases} 0 & \text{if } d \geq \xi \\ e_B \cdot \left(1 - \frac{d^2}{\xi^2}\right) & \text{otherwise} \end{cases} \tag{7}$$

where: Δe – the amount of energy transferred,
e_B – initial amount of energy owned by the second agent,
d – calculated distance (or the similarity level),
ξ – value of the crowding factor.

The flow of energy connected with the crowd principle causes that some of the similar agents are more likely to be eliminated. This should lead to more uniform agent distribution along the Pareto frontier and prevent agent clustering around particular spots of the search space.

The idea behind introducing the mechanism of crowd was to discourage agents from creating large bunches of similar solutions at some points on the Pareto frontier (this is quite similar to the ideas presented by De Jong [12] and others). Instead they should be rather uniformly distributed over the whole frontier. Also in the case of problems for which the Pareto set consists of several disjoined parts, this mechanism should improve the ability of agents to cover a wide area of the search space, and discover other parts of the frontier.

The complete algorithm of the agent's decision making process is shown on fig. 1.

V. Experimental results

Many experiments were performed to check how the proposed technique works for different optimisation problems. Figures 2 and 3 show the results obtained for sample optimisation problems. One may notice that the solutions found are very close to the actual Pareto frontier. Surely the approximation does not cover all the points of the actual Pareto frontier, but it is due to the fact that it consists of only a finite number of agents.

$$f_1(x) = -x^4 - 3x^3 + 10x^2 + 10x + 10$$
$$f_2(x) = \frac{1}{2}x^4 + 2x^3 + 10x^2 - 10x + 5$$

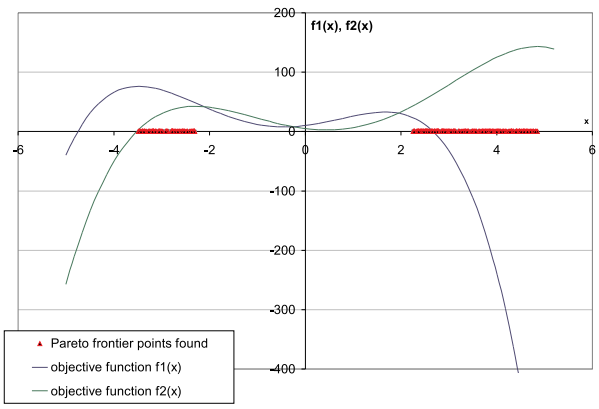

Fig. 2. Sample optimisation problem and the Pareto frontier discovered by the system

$$f_1(x,y,z,t) = -(x-2)^2 - (y+3)^2 - (z-5)^2 - (t-4)^2 + 5$$
$$f_2(x,y,z,t) = \frac{\sin x + \sin y + \sin z + \sin t}{1 + (\frac{x}{10})^2 + (\frac{y}{10})^2 + (\frac{z}{10})^2 + (\frac{t}{10})^2}$$

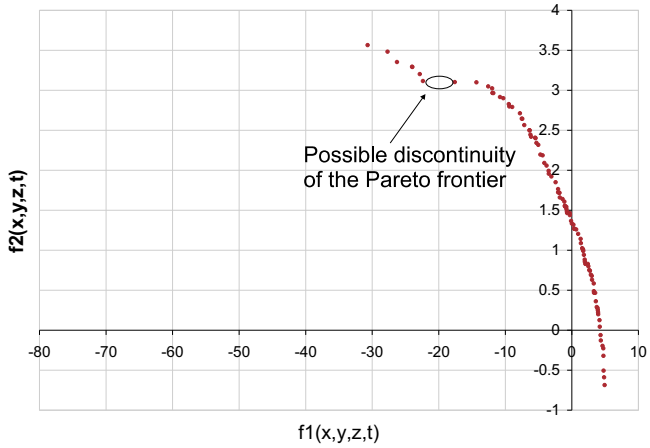

Fig. 3. The crowding factor's role in determining discontinuity of the Pareto frontier

A set of parameters is used to control EMAS behaviour and adjust it to particular problems. The parameters include (but are not limited to) the following:

- description of the criteria functions,
- description of the search domain,
- size of the initial agent population,
- value of the crowding factor,
- total amount of life energy in the system,
- the minimal amount of energy that could be transferred between agents.

These parameters were checked in order to establish the influence of the mechanism of crowd on the system performance. It was observed that:

1. Small values of the crowding factor improve the system performance for almost every test problem, as it was expected. The agents are able to find more points from the Pareto frontier, comparing to the cases, when the crowding factor equals 0 or is too large (fig. 4).
2. For problems with a fairly large number of disjoint parts of Pareto-optimal solutions a rather large value of the crowding factor (compared to the distance between separate parts of the Pareto frontier) allows the system to find these disjoint areas more efficiently (fig. 5).

As it was shown on fig. 4 and fig. 5, the crowding factor has an influence on the average minimal distance between solutions (this value was computed over the whole set of non-dominated solutions). In fact the lower value indicate that the "gaps" between solutions found, are smaller. If these gaps are

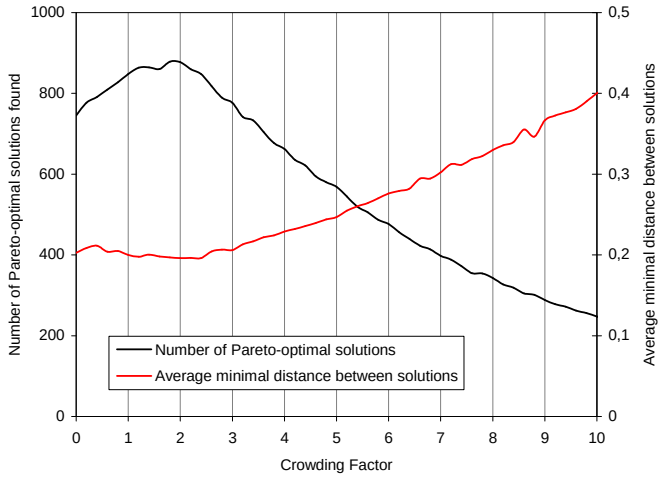

Fig. 4. The influence of the crowding factor on the performance of the system in case of the coherent Pareto frontier

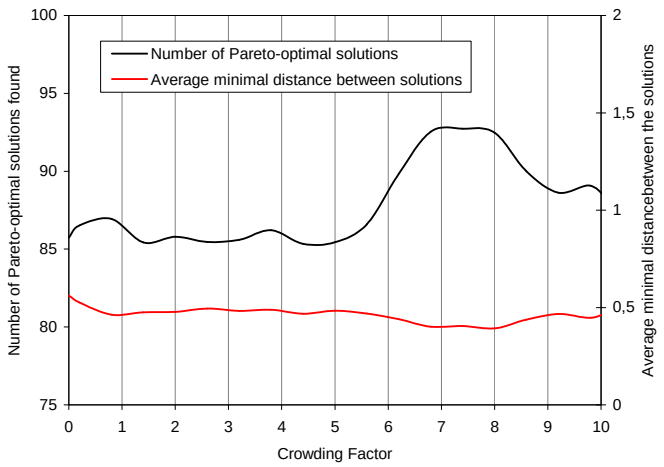

Fig. 5. The influence of the crowding factor on the performance of the system in case of Pareto frontier consisting of several disjoined parts

smaller, then the distribution is more uniform and better reflects the real Pareto frontier, which in turn makes it possible to recognize disjoint areas of Pareto frontier (fig. 3).

VI. Concluding remarks

The proposed idea of an evolutionary multi-agent system for multi-objective optimisation proved to be working in a number of tests. Up till now it is still too early to compare this method with various other heuristics supporting decision making known from literature. Yet the preliminary results show significant advantages over other techniques regarding adaptation to a particular problem, which is mainly performed by the system itself. In most of the problems investigated the introduction of the mechanism of crowd improved the system performance concerning the distribution of the solutions on the Pareto frontier. Of course, it is impossible to assure that some value of the crowding factor may give perfectly uniform distribution, and hence absolute certainty of the Pareto frontier shape. Yet, in most cases it allows to estimate it much better.

Further research should concern the effectiveness of the approach proposed, especially in the case of difficult problems (many dimensions, multimodal or discontinuous criteria, etc.). Several extensions to the evolutionary process (such as aggregation) applied to EMAS in other application domains should also be considered.

References

[1] T. Bäck, U. Hammel, and H.-P. Schwefel. Evolutionary computation: Comments on the history and current state. *IEEE Trans. on Evolutionary Computation*, 1(1), 1997.

[2] K. Cetnarowicz, G. Dobrowolski, M. Kisiel-Dorohinicki, and E. Nawarecki. Some approach to design and realisation of mass multi-agent systems. In R. Schaefer and S. Sędziwy, editors, *Advances in Multi-Agent Systems*. Jagiellonian University, 2001.

[3] K. Cetnarowicz, M. Kisiel-Dorohinicki, and E. Nawarecki. The application of evolution process in multi-agent world (MAW) to the prediction system. In M. Tokoro, editor, *Proc. of the 2nd Int. Conf. on Multi-Agent Systems (ICMAS'96)*. AAAI Press, 1996.

[4] C. A. Coello Coello. An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, editors, *Proceedings of the Congress on Evolutionary Computation*, volume 1. IEEE Press, 1999.

[5] G. Dobrowolski, M. Kisiel-Dorohinicki, and E. Nawarecki. Evolutionary multi-agent system in multiobjective optimisation. In M. Hamza, editor, *Proc. of the IASTED Int. Symp.: Applied Informatics*. IASTED/ACTA Press, 2001.

[6] C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

[7] M. P. Fourman. Compaction of symbolic layout using genetic algorithms. In Grefenstette [9].

[8] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[9] J. J. Grefenstette, editor. *Proceedings of an International Conference on Genetic Algorithms and Their Applications*. Lawrence Erlbaum Associates, 1985.

[10] T. Haynes and S. Sen. Crossover operators for evolving A team. In J. R. Koza, K. Deb, M. Dorigo, D. B. Fogel, M. Garzon, H. Iba, and R. L. Riolo, editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*. Morgan Kaufmann Publishers, 1997.

[11] N. R. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Journal of Autonomous Agents and Multi-Agent Systems*, 1(1), 1998.

[12] K. A. D. Jong. *An analysis of the behaviour of a class of genetic systems*. PhD thesis, University of Michigan, 1975.

[13] J. Liu and H. Qin. Adaptation and learning in animated creatures. In W. L. Johnson and B. Hayes-Roth, editors, *Proc. of the 1st Int. Conf. on Autonomous Agents (Agents'97)*. ACM Press, 1997.

[14] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithm. In Grefenstette [9].