Chapter 2
Agent-Based Computing

In the 1970s, there was a growing interest in the systems, where a task to be solved was decomposed into smaller parts (subtasks), in order to solve them separately and later integrate the solution. This approach may be described as *distributed problem solving*, and it is usually easy to implement in parallel environments such as multi-core machines, clusters or grids. In the field of *multi-agent systems* bearing a significant legacy from the distributed problem solving, distributed individuals (agents) received great attention, mainly because they were perceived as autonomous beings, being capable of interacting with their environment and other agents, bearing the features of intelligence.

In fact, during the last decades intelligent and autonomous software agents have been widely applied in various domains, such as power systems management [207], flood forecasting [128], business process management [153], intersection management [89], or solving difficult optimization problems [191], just to mention a few. The key to understand the concept of a multi-agent system (MAS) is intelligent interaction (like coordination, cooperation, or negotiation). Thus, multi-agent systems are ideally suited for representing problems that have many solving methods, involve many perspectives, and/or may be solved by many entities [299]. That is why, one of important application areas of multi-agent systems is large-scale computing [32, 286].

This chapter sketches the basic definitions of agent and agent system, then describes popular agent-based architectures particularly useful for computing systems. Next, evolutionary agent-based computing paradigm is presented along with detailed description of Evolutionary Multi-Agent System (EMAS) along with some of its variations.

A. Byrski and M. Kisiel-Dorohinicki, *Evolutionary Multi-Agent Systems*,
Studies in Computational Intelligence 680, DOI 10.1007/978-3-319-51388-1_2

2.1 Agency, Architectures, Management and Computing

Beginning of agency can be dated back to 50s of 20th century. One of the first scientists working on agents were John McCarthy and Oliver Selfridge, who proposed the notion of “agent” for the first time [161]. Agent-based techniques cannot be perceived as one, homogeneous methodology, as they originate from many different concepts and applications. Some of them develop quicker and more intensively (e.g. interface agents used for facilitating everyday computer use), other are still researched and will be evaluated properly in the future. Advantages of agent-based systems can be seen in many applications (e.g. simulation and control of transportation systems). Research on other application areas (e.g. agent-based computing) is going on.

Starting in the half of 70s of the 20th century, many scientists working on artificial intelligence began to appreciate the systems using the decomposition of the task to many smaller ones, in order to solve them one by one and later combine them into the global solution of the given problem. The approach called *distributed problem solving* focused on the possibilities of decomposing the computing process into smaller parts [103]. In the multi-agent systems a greater attention was given to the particular units of distribution, that were endowed with autonomy and could perceive their environment in order to achieve its goals.

Agents play an important role in the integration of artificial intelligence subdisciplines, which is often related to a hybrid design of modern intelligent systems [250]. In most similar applications reported in the literature (see, e.g. [62, 252] for a review), evolutionary algorithm is used by an agent to aid realization of some of its tasks, often connected with learning or reasoning, or to support coordination of some group (team) activity. In other approaches, agents constitute a management infrastructure for a distributed realization of an evolutionary algorithm [284–286].

2.1.1 Agents and Multi-agent Systems

According to one of the most popular definitions proposed by Wooldridge, an agent is a computer system situated in an environment, and capable of undertaking independent, autonomous actions in this environment in order to fulfill tasks on behalf of its user [300]. Autonomy is perceived as one of the most crucial features of an agent.

Following this definition, any computer program that manages a certain apparatus (e.g. a thermostat) or affects the state of the computer system (e.g. a Unix system daemon) may be perceived as an agent. Thus it might seem that such definition of an agent introduces a new name for some existing, well-known programming techniques. In fact, intelligent agents bring new quality crossing the borders of already existing computer systems, enhancing the notion of an object or process with additional, important features, e.g. [79, 300] helping other agents to fulfil their goals:

- reactivity: agents may perceive their environment and react to changes in that environment,
- pro-activity: agents may perform tasks based on their own initiative,
- social ability: agents are able to interact with other agents (also with users).

Another classification of agents should be referred to, in order to see how autonomous agents undertake their decisions [300]:

- logic-driven agent—utilizing deduction process [126],
- reactive agents—undertaking the decisions uses a predefined function mapping the situations into actions [35, 107],
- BDI (*ang. belief-desire-intention*) agents—manipulate the data structures representing beliefs, goals and intentions of the agents [34, 59, 60].

It is noteworthy that fulfilling the goal becomes a *raison d'être* for an agent. This is also the most important and determining factor in undertaking actions in the environment by an agent.

The notion of agent system is based directly on the notion of agent. Generally speaking, an agent system is a system, in which a key abstraction is that of an agent. Therefore, a multi-agent system is one that consists of a group of agents which interact one with another [108, 156].

Agents act in their environment, and different groups of agents may perform their tasks in different parts of the environment. In particular, their activities may overlap. As an example, the possibility of communicating between agents that are "close" in the environment may be given (of course, their closeness depends strongly on the notion of neighborhood, if it was implemented), or direct interaction with the environment (e.g. only one agent-robot may pass through the door at a time) [299].

Among the main features of multi-agent systems one may distinguish [79]:

- distribution—the agent system structure is easy to implement in distributed systems (e.g. in a local cluster),
- autonomy—each agent exists independently on the others, it can sustain in the environment without the external interaction, it undertakes the decisions based on its observations, desires and its own model of the environment,
- decentralization—there should be no global control mechanisms for the whole agent system as a consequence of the autonomy of the particular agents,
- exchange of beliefs—agents can communicate in order to exchange their information about the environment, their actions etc. (using e.g. ontologies to describe their world),
- interaction—agents can interact with other agents and with the environment by exchanging the messages using the predefined protocols or standard communication languages as e.g. ACL (Agent Communication Language) or KQML (Knowledge Query Manipulation Language) [110, 127],
- organization—decentralization and autonomy in the distributed environment creates the need to introduce a certain order of the agent population, moreover the agent usually can perceive only part of its environment and part of the neigboring agents,

- situatedness—the environment naturally sets constraints for all the possible actions and observations undertaken by the agent,
- openness—it is often difficult to set the structure of the environment that is heterogeneous and can be dynamically changed (e.g. Internet network),
- emergence—arising of holistic phenomena not programmed before, usually shown by a certain group of agents (i.e. collective intelligence) [135, 136],
- adaptation—these systems are flexible and can adapt to the changing environmental conditions,
- delegation—an important notion, following Bradshaw: "agent is that agent does",
- personalization—the agent may acquire the preferences of its user in order to ease his/hers central activities (e.g. interface agents facilitating typewriting) [195],
- accessibility—the agent systems are quite easy for description, application and adaptation to many problems, they are also very easy for implementation, providing that a certain framework is available, supporting different low-level services such as communication or monitoring.

Agents have also been used in computing systems to enhance the search and optimization capabilities with the above-mentioned agency features.

2.1.2 Architectures of Agent-Based Computing Systems

Starting with a multi-agent perspective, three types of computing agent system architectures, which form consecutive levels of increasing complexity, can be distinguished [177]. The main purpose of these architectures is to serve as a means of incorporation of computing methods into cooperating, autonomous entities—agents.

Figure 2.1 presents an example of a classical approach to hybrid computing system—an evolutionary system using a stochastic hill-climbing algorithm constituting in fact a memetic system. Each individual represents a solution, and the evolving population explores and exploits the feasible search space. The whole system is embodied in an agent utilizing this idea to adapt to the specific environment.

Evolutionary multi-agent system shown in Fig. 2.2 can be seen as next step in possible specialization [61]. In this case, evolutionary processes work at a population level—agents searching for a solution to a certain problem are able to generate new agents and may be eliminated from the system on the basis of adequately combined evolutionary operators. The predefined distributed selection mechanism increases the possibility of reproduction for the best agents (in terms of fitness function value). The result of the search is formed as a set of results obtained by single agents. The architecture of EMAS is homogeneous, which means that the agents are identical as far as an algorithm and built-in actions are considered.

EMAS may be easily hybridized with local search techniques forming a memetic variant of the base system. In this case, each agent performs local search in the course of its life in the system. This may be carried out during reproduction or at arbitrarily chosen moments.

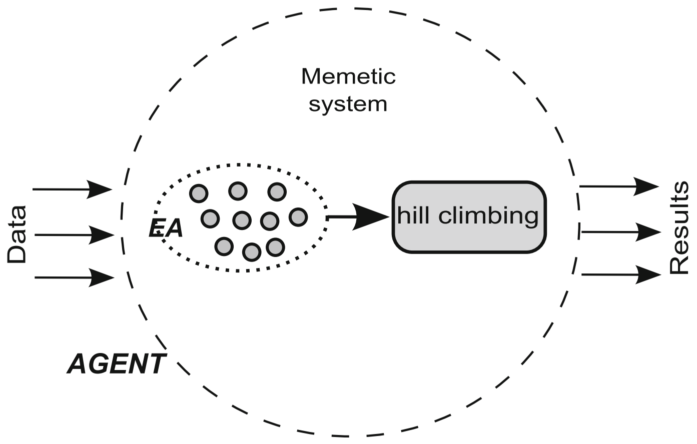

Fig. 2.1 A hybrid computing system located in a single agent

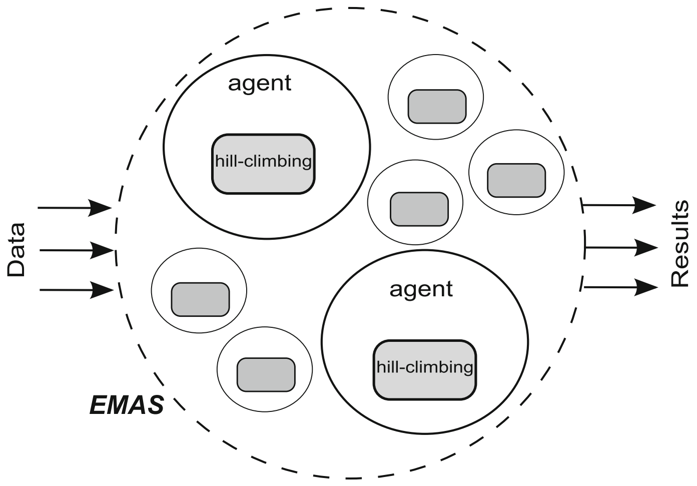

Fig. 2.2 A population of homogeneous agents

The highest specialization level is reached when the agent population is heterogeneous (see Fig. 2.3). There exist in the system different agents, focused on solving different problems or realizing of different task related to problem solving or system organization, communication etc. The result of the system is stipulated as a consequence of the outcome of the negotiation process among the agents. Many different techniques and protocols can be applied to this purpose. It may be said that these systems closely approach typical multi-agent ones operating in the computer network environment [177].

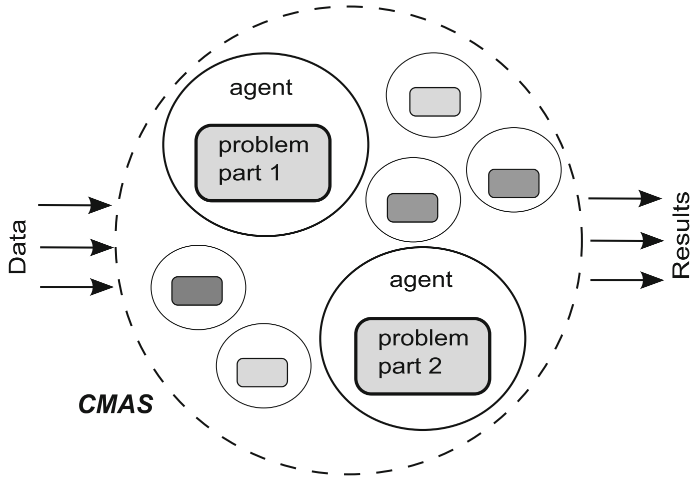

Fig. 2.3 A population of heterogeneous agents

EMAS (Sect. 2.3) and its memetic variant (Sect. 2.4.1) belong to the second class, while other variants of EMAS (Sect. 2.4) described in this monograph belong to the third class described above.

2.1.3 Computing Systems Management by Agent Means

One of the possible applications of agency in computing systems is situated at a technical level. An agent community may take care of the the management of the distributed system, utilizing their autonomy and auto-adaptation to implement, e.g. scheduling or load balancing.

Distributed management of the system deployed in a cluster or grid requires a definition of node topology, neighborhood and migration policies that are affected by a current load of nodes. Well-known standards for constructing multi-agent environments (such as FIPA [111]) do not provide such capabilities. Another important functionality missing in many platforms is the notion of distance between agents, measured as a factor of communication throughput.

Grochowski and Schaefer [284–286] proposed Smart Solid Architecture (see Fig. 2.4) that supports these requirements. This architecture is similar to EMAS architecture (cf. Sect. 2.3), where agents are also homogeneous.

However, in this approach the task is divided into subtasks and these are delegated to agents which are to solve them. Agents in this model accomplish two goals: perform computation of the task and try to find a better execution environment (computing node) for the task, based on the load and throughput information. It is

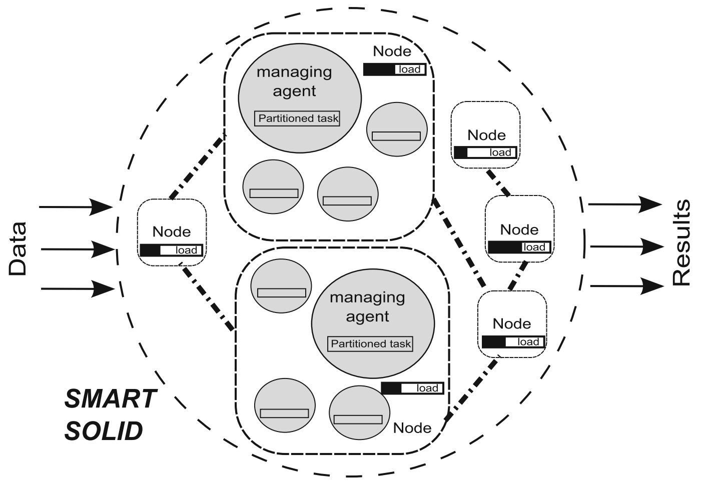

Fig. 2.4 Smart solid architecture

noteworthy that agents do interact with one another, in order to be able to solve the assigned task. Agents may undertake the following actions:

- execute the task in order to solve it and communicate the results to other agents,
- denominate the load requirements,
- based on the requirements compute the possible migration capabilities of the agent,
- further divide the task, create the child agents,
- migrate to another execution environment carrying the task.

An effective scheduling algorithm was also introduced in the discussed environment and tested by Schaefer, Grochowski and Uhruski, utilizing the natural inspiration of the diffusion mechanism [138]. This system was successfully applied e.g. in the problem of mesh generation for computed-aided engineering applications [257].

2.2 Bringing Together Agent-Based and Evolutionary Computing

Recalling the most important features of multi-agent systems (MAS) that are adopted by the researchers [108] and can be regarded as definitions of the term:

- a multi agent system is constituted by a set of agents and a sub-system called environment;
- agents perform certain actions that are characteristic to them, resulting in changes of themselves or the environment;

- some of elementary agents' actions form mechanisms perceived as specific in multi-agent systems in general, e.g. negotiation, communication, exchange of resources etc.;
- providing that there is a structure embedded in the environment (a graph) reflecting spatial phenomena of the system, it forms a base for migration of agents;
- an agent that is treated as a black box (observed from the outside) may have some human attributes like autonomy, intelligence etc.

one can consider building an agent-based system, focused on solving computing tasks, incorporating the above-mentioned features. However, one must keep in mind, that computing systems are aimed at producing tangible results (e.g. an optimal value of certain function, an optimal set of certain parameters), thus they need certain mechanisms for synchronization of their work. At the same time any synchronization realized in agent-based systems is suspicious, as it may be perceived as a means for global control. Agent-based systems leverage the notions of autonomy, thus bringing up new techniques for control of the agents that are aimed at producing one common goal are necessary.

2.2.1 Non-renewable Resources

The most important problem of the outlined concept is that in agent systems using centralized mechanisms of computing process is impossible. A specific approach must be created, because of:

- lack of global knowledge and global synchronization of actions,
- autonomy of particular agents in undertaking their actions.

Thus the concept of controlling the evolution process based on non-renewable resources was proposed [169, 174].

The developed mechanisms demand defining a certain number of resources, that describe the state of the environment and agents. The resources can be used to indicate the quality or efficiency of the agents solving particular task. The agents rich in resources may perform certain actions and survive in the environment. It is usually realized by defining of a certain threshold of the resource, required for realizing particular action. The algorithms of the agent behavior may also be defined by the probability distributions dependent on the amount of resources owned. To very important issues belong the boundary conditions of the of the agent resources, conditioning its participation in the processes realized in the population, thus affecting the behavior of the whole system. On exceeding these conditions the agent is removed from the population and its remaining resources are returned to the environment or distributed among the other agents.

Each agent during creation in the system obtains certain amounts of the resources, either from other agents or from the environment. Realization of certain actions may make the agent lose or gain resources. This can become a mechanism for assessing the quality of the agent—increasing of the resource amount may be treated as a reward for "good" behavior, and decreasing as a punishment. The most natural way of realization of this mechanism is introducing a cost for realization of any action and rewards for achieving good results of the realized task. The resources may be assigned using the environmental mechanisms on the local level or by the dedicated agents based on information about the quality of the reached solutions, supplied from outside.

2.2.2 Integrating Evolutionary Paradigm with Agency

A starting point for the idea of *agent-based evolutionary computing* was the observation that many newly constructed problem classes calls requires new metaheuristics, in particular evolutionary approaches, on the other hand classic evolutionary computing techniques do not follow many essential aspects of natural evolutionary processes [14]. The basic evolutionary algorithm is often enhanced in order to realize specific mechanisms observed in nature, in particular in order to maintain useful diversity of the population. At the same time, one of the most neglected fact of classic evolutionary computing is that the whole evolution process is centralized and fully synchronized. Thus the objects of evolution become simple, inactive data structures, and they are processed by a common "manager" which has complete and total "knowledge" about the whole population [58].

New possibilities should be given by an evolution model, that will enrich the individuals in capabilities of autonomous acting and interaction in a common environment. As certain substitutes of such approach may be treated several coevolutionary models or parallel evolutionary algorithms, in particular their hybrid variants, making the realization of the evolutionary processes dependent on the particular context of the individual or of the (sub-)population. It seems, however, that using the concept of agency may bring new quality into the computing systems by creating a new computing model. Thus, embedding the evolutionary processes into agent system will make possible full decentralization of the evolution process. Independence of the environment perception and the actions undertaken by the agents makes possible their mutual interaction, leading to opening new possibilities of introducing completely new elements into the evolution process. Such an approach, that may be called *decentralized evolutionary computation* should help in eliminating some of the above-mentioned limitations of the classic evolution model. Similar assumptions are shared by a concept of Multi-Agent Genetic Algorithm [304], however this approach assumes a very limited autonomy of the agents localized in predefined, constant positions of two-dimensional grid (similarly to cellular model of evolution described in Sect. 1.2.4).

A key concept of agent-based evolutionary computing is introducing of evolutionary mechanisms into agent environment, in other words subjecting the agent population to the evolution process. As a result, a system capable of self-adaptation to the changing conditions both on the particular components level (agents) or their organizational structure [84, 221]. It can be said, that the evolution process is "managed" or "lead" to a certain extent by the population of autonomous agents, representing individuals that undergo the evolution process. These individuals may belong to different species or groups.

2.2.3 *Logical and Physical System Structure*

A very important feature for the evolution process conducted in the agent system is the fact of co-existence of the agents in a common, possibly spatial environment. The agents and resources can be localized in certain places of the environment (regions, cells) where they can be observed, and affected by the other agents (cf. Fig. 2.5). Each individual may access only a certain part of the information and resources, lying in its range. Moreover, the agent can interact, observe and communicate with a limited number of other agents, found in its neighborhood (determined using a predefined topology). An effect of such local perception and local interaction is limited access of the agents to certain resources and information that can be found in the system, thus the evolutionary processes are conducted with higher diversity, making possible implementation of parapatric or allopatric speciation.

Limitations in reachability of resources and information may result in creating somehow privileged areas in the system space (objectively or subjectively dependent on the current state and/or agent needs). This may be also caused by the structure of the system space itself, but also presence (or absence) of agents with certain features. The agents themselves may also want to change their placement in the system, realizing the migration according to the rules imposed by the system space structure. Possibility of migration may also be used for realization of load balancing mechanisms, in the case of the distributed computing environment.

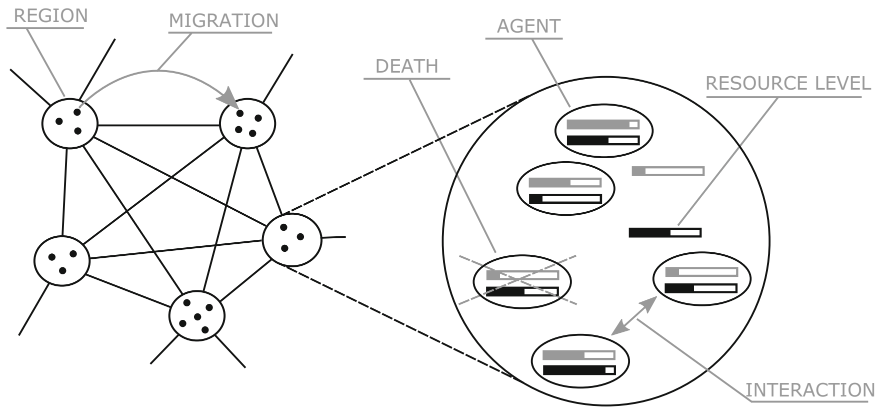

Fig. 2.5 Concept of space and resources in agent-based computing

Pursuing analogies to parallel evolutionary algorithms, one can consider two space models: fine-grained (multi-dimensional neighborhood structure, so-called “cells” with a predefined metric, e.g. “Manhattan” setting the interaction range) or coarse-grained (so-called “regions” or “islands”, completely isolated, where the interactions are possible only within one region). In the latter case, the migration range may be limited by the topology of the connections among the regions, similarly to multi-population evolutionary algorithms. Here coarse-grained or hybrid (coarse grained with fine grained inside the regions) models are preferred because of relatively easy implementation and natural mapping of the logical and physical spaces of the agent system.

2.2.4 Representation of the Tasks and Solutions

In the discussed concept, the population of agents is located in a common environment which has certain properties (e.g. space structure, availability of certain resources, placement of agents and resources, possibilities of cooperation etc.) which represent the current state of the real world or certain task to be solved. A significant difference may be perceived here when comparing to classic evolutionary algorithm, where the task is represented only by the fitness function, while the environment of particular individuals is not considered in any way.

Of course agents may observe the environment and the other agents, they may also realize actions affecting their own state, as well as the state of the environment or even other agents. In any possible moment, the state of the agent (e.g. its resources or information), its location in the system space or the outcome of its action (e.g. placement of the resources in the environment) may become a basis for setting its solution of the given problem. The solution may also be represented by the state of the behavior of a group of agents that makes also a significant improvement, when comparing to classic evolutionary algorithms.

The basic variant of the considered model is so-called **evolutionary multi-agent system—EMAS** where the agent is considered as a basic evolution entity [169]. The agent is equipped in the ability of reproduction, that may be accompanied by random changes in the inherited features of the generated agent (mutation, recombination). Moreover, it is necessary to introduce certain mechanisms for elimination of the agents representing low quality solutions. In effect, an asynchronous and decentralized process of evolution is constituted. If it is properly parametrized, it should lead to autonomous adaptation of the system to the state needed in particular situation (i.e. a proper configuration of the parameters).

An interesting alternative to the above-described models is **flock-based multi-agent system—FMAS** [173], where the agent represents a group of individuals (a flock). Similarly to *migrational* model of evolutionary algorithm, in the case of each flock an evolution process is realized (e.g. classic evolutionary algorithm), enhanced by the possibility of migration of individuals. However in this case, the individuals migrate between the flocks within one evolutionary island, while the flocks, that can be treated as another level of system organization, may migrate themselves between

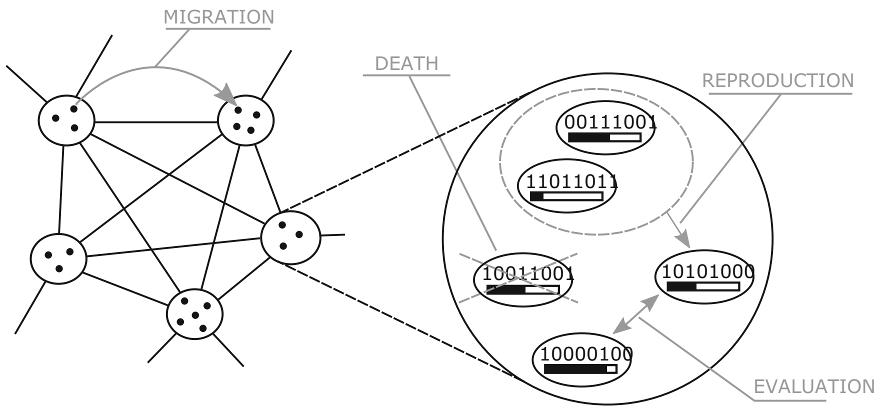

Fig. 2.6 A concept of evolutionary multi-agent system (EMAS)

the islands. Moreover it is possible to split or merge the flocks depending on their state (or rather the state of the individuals constituting the flock), leading to possibility of dynamic reconfiguration of the whole system in the course of the computing process. The presented concept has all the advantages of the parallel evolutionary algorithm, at the same time allowing for dynamic adaptation of the computing system to the problem being solved [172].

2.3 Evolutionary Multi-agent System

The core idea of the evolutionary multi-agent system is treating each agent as an individual in the population, thus equipping it in the information (in the genetic model of evolution—a genotype) allowing for generation (*ontogenesis*) the solution of the given task (*a phenotype*). Generally speaking, the work of EMAS is based on processing the population of agents cooperating in the common environment, according to the inheritance and natural selection rules. However because of the specific features of the agent environment, the evolution must be realized in a different way than in the classic evolutionary algorithm. The key concept of this process will be of course, as it is in the natural processes, the phenomenon of death and the capability of reproduction, that must be reflected in the design of the agents (see Fig. 2.6).

2.3.1 Reproduction and Inheritance

One of the two of the most crucial elements of the evolution process is reproduction, i.e. introducing new offspring individuals, similar to the parents. In evolutionary multi-agent system, each agent is equipped with the ability of reproduction. It consists in creating of the offspring agent and passing to it (inheritance) of certain features of the parent, or certain combination of the features of parents in the case of sexual reproduction (recombination) with possibility of small random changes (mutation). Of course besides the inherited features, the agent may possess certain information that are not inherited and that may be changed in the course of its life (e.g. the knowledge about the surrounding environment).

Depending on the assumed evolution model, the features inherited by the agent may be called:

- *genetic material* (the evolution model is similar to the one used in genetic algorithms),
- *individual features* (the evolution model is similar to the one used in evolution strategies),
- *species features* (the evolution model is similar to the one used in evolutionary programming).

In the first case, the basic parameters depicting the agent behavior become its genotype—they are inherited from its parents in the moment of creation and remain unchanged during all of its life. In this case, the complete information—inherited and acquired—describes to the full extent the behavior of the agent in the environment—it phenotype. In the second and the third case the inherited features describe the agent behavior in full.

At the same time the reproduction is directly connected with the position of the agent in the environment:

- the newly created agent is placed in the nearest vicinity of the parent or parents,
- in the case of recombination the parents must be located in their neighborhood,
- the evaluation of the individual may be realized only in comparison to its neighboring agents.

The possibilities of migration, meant as searching for appropriate location for agent in the environment, will have in EMAS a significant influence on the course of evolutionary processes.

2.3.2 Selection

The biggest problem in the evolution process assumed in EMAS is the way of realization of the second crucial element of the evolutionary processes—the natural selection rule. Because of autonomy of the individuals (agents), one cannot simply realize the mechanisms known in the classic evolutionary algorithms. At the same time, a very important aspect of the evolutionary agent system that must be considered during the realization of the selection mechanisms is the dynamics of the population. The selection process cannot lead to significant reduction of the agents, at the same time should not lead to their excessive growth in number. The first problem leads to stagnation of the evolution process, or even to the extinction of population, the second one to decreasing of the efficiency of the system.

In the approach discussed here, following the mechanisms observed in nature, the selection process is realized based on the described in the previous section mechanisms based on non-renewable resources. In the simplest case, one kind of such resource is used, called *life energy*. Each of agents is assigned certain levels of the life energy, that is necessary for performing certain actions, in particular connected to the evolution process, such as reproduction. The agent may undertake the decision to realize the reproduction action, when the level of its energy exceeds the *reproduction threshold*. Thanks to this only the best individuals, on the basis on their energetic status, have the highest probability of reproduction. At the same time, agents having low energy should be successively eliminated from the population. In order to realize this in the basic model of EMAS, a *death threshold* is set. If the energy level of an agent falls below the death threshold, the agent is removed from the population—a typical value of the death threshold is 0, so the agent is removed from the population when it looses the whole life energy. As a consequence, in the system there remain only the agents with the best quality (from the point of view of the problem to be solved), and by the manipulation of the available energy, the size of the population can be controlled.

In the beginning of the evolution process, each agent receives the same energy portion $e_i[t_0] = e_{init}$ (e_i means here the energetic level of the i-th agent, t_0 means the moment of its creation). This value along with the population size are very crucial parameters describing the features of the system. Similarly, a certain energy portion (dependent on the parent or the environment) is passed to the agent created in the process of reproduction. This value should be of course lower than the reproduction threshold, and higher than the death threshold ($e_{repr} > e_i[t_0] > e_{death}$). Later, in the course of agent's life, its energy rises or falls down caused by the actions it realizes:

$$e_i[t+1] = e_i[t] + \Delta e_i^+[t] - \Delta e_i^-[t] \tag{2.1}$$

where $\Delta e_i^+[t]$ is the sum of the energy gained and $\Delta e_i^-[t]$ is the sum of energy lost by the i-th agent in the t-th step.

As a consequence, the agent can gain or loose the possibility of realizing of certain actions, in particular the action of reproduction, or even, when the energy level becomes lower than the death threshold, it will be removed from the system. Assuming that each change of the energetic level of the agent is accompanied by a reversed change of the energy of a certain agent or the environment, one can achieve the effect of maintaining the constant value of the total energy in the whole system, forming so called *energy preserving rule*:

$$\sum_{i=1}^{N} e_i[t] = const \tag{2.2}$$

where N stands for the number of the agents in the population at the time t.

This feature makes possible (using the parameters of the dynamics of the evolutionary processes—death and reproduction thresholds) assessing the limit of the agent population.

The most important mechanism of the inter-agent energy flow is the meeting strategy. Each agent in each time step undertakes (usually randomly) a decision of choosing another agent in order to compare the quality of their solutions (from the optimization criterion point of view), and if such comparison is possible, the better agent receives a predefined portion of energy from the worse agent (such setting resembles somehow the tournament selection well-known in the classic evolutionary algorithms). Thus the agents representing the best solutions in the population in the subsequent steps will increase their energy level, overtaking the other agents' energy, that in the consequence would lead to reaching certain energy level and to reproduction. At the same time the agents representing the lowest quality solutions in the population will successively loose their energy, leading to reaching the predefined death threshold, and to their elimination from the system. Thus the meeting strategy along with the death and reproduction actions realize the goal of the selection, i.e. preference of the best solutions from the point of view of the optimization criterion.

The energy must be also localized in the environment, that may participate in the process of the energy exchange among the agents. In the simplest case the environment may acquire the energy from the removed agents and pass them to the newly created agents. It can also participate in the evaluation mechanisms and assign the agents the energy in such way, that the better agents get more than the worse ones.

2.3.3 Computational Features

In the case of applying evolutionary multi-agent system as a technique for solving search and optimization system, as a method closer to the natural evolution model, one can expect the following advantages of the mechanism applied:

- local selection allowing for creation of geographical niches, meaning more intensive exploration of different search space parts at the same time,
- evolving of phenotype during the life of the individual (agent) based on its interaction with the environment,
- self-adaptation of the population size to the solved problem, applying appropriate selection mechanisms, allowing for efficient management of the population dynamics.

Introduction of the environment space along with the migration mechanisms makes easier the implementation of the system in the distributed environment, that should ease its efficiency.

Local Selection

As it was said before, one of possibility brought by the EMAS model is distribution of the evolution process in the environment of the agent system, and as a consequence, realization of actions by the agents in the context of other agents and resources in certain location only. Thus the realization of the selection mechanisms in EMAS is a consequence of the assumption of the lack of global knowledge and synchronization, that must be applied in each agent system (cf. Sect. 2.2.1). It makes impossible the evaluation of all the agents at the same time, followed by the creation of certain niches, where the subpopulation may concentrate in different parts of the search space.

This mechanism reminds of co-evolutionary techniques or parallel evolutionary algorithms (cf. Sect. 1.2.4), one can expect here similar advantages as offered by them. First of all the diversity of the population should increase, followed by the so-called convergence reliability [12], that becomes very important in the practical applications of EMAS as a technique of solving the problems with unknown characteristics. The mechanisms supporting of the creation of niches make this approach particularly useful for solving multi-modal and multi-criteria problems (cf. [244]).

Evolving of Phenotype

In the case of applying the genetic evolution model (similar to genetic algorithms), one can see the subsequent characteristic feature of EMAS. The agent becomes here the object of the evolution process and the entity functioning in its environment, that makes possible its adaptation (learning) in the course of its life to the encountered conditions, making possible development of not-inherited features, being the part of phenotype. As an example of using this feature, one can consider applying of local optimization techniques, enhancing the solution owned by the agent in the course of its life [49]. One can also consider this as a feature particularly useful for the problems, based on searching for the optimal parameters of the algorithms that may be realized at the same time as evolution process (e.g. search for optimal parameters of neural network architecture, where the genotype consists of the network parameters and the phenotype—the network trained in the course of agent's life).

2.3.4 Maintaining Population Diversity

The greatest advantage of the above-described, basic variant of EMAS is the simplicity of its realization, accompanied by its rich characteristics in the case of solving search and optimization problems. Similarly as in the case of classic evolutionary computing, one can identify a broad spectrum of problems, for which these features are insufficient. One can point out the applications requiring higher adaptive abilities, such as in the case of non-stationary environment, which characteristics changes in time, or in the case of multi-criteria optimization. It turns out that the EMAS model can be treated as a convenient base for natural implementation of many mechanisms observed in nature, which—sometimes with great difficulty—are used in the classic evolutionary computing and other population-based techniques of computational intelligence [186].

An example of a simple realization of a classic concept from the group of niching and speciation techniques, is the implementation of the crowding, using the meeting strategies and the energy transfer. It makes a counterweight to the observed in all evolutionary techniques preference to group the solutions in the local extrema basins of attraction, making the main cause for loosing of diversity of the solutions in the population. This is realized by decreasing of the energy of the individuals representing very similar solutions, also these potentially populating the same ecological niche (i.e. the basin of attraction of the same local extremum).

In the considered variant of the meeting strategy, besides the comparison of the quality, also the distance between the individuals is computed, according to the assumed metric, e.g. for the continuous problem realized as follows:

$$d(x^A, x^B) = \sum_{i=1}^{N} |x_i^A - x_i^B| \tag{2.3}$$

where $x^A = [x_1^A, \ldots, x_N^A]$ i $x^B = [x_1^B, \ldots, x_N^B]$ are solutions represented by the meeting agents. The work of the mechanism of the *energetic crowding* depends on the value of the parameter ξ called *crowding factor*. If the similarity (distance) between the solutions exceeds certain value of this parameter, the agent initiating the meeting overtakes the energy from the encountered agent in the value of Δe depending on the owned energy and on the level of similarity between the solutions:

$$\Delta e = \begin{cases} 0 & \text{if } d \geq \xi \\ e_B \cdot \left(1 - \frac{d^2}{\xi^2}\right) & \text{in other cases} \end{cases} \tag{2.4}$$

where e_B is the energy level owned by the encountered agent and d is the computed similarity (distance) between the solutions.

The energy flow connected with the meeting mechanism makes possible, that too similar (considering the solutions owned) agents will loose their life energy in their subsequent life steps, leading to restricting their reproduction ability and even leading to their removal from the population. Indirectly it may result in increasing the probability of reproduction of the other agents, assuming they will represent good solutions and they will obtain energy in the basic mechanism of meeting strategy (i.e. by comparing their quality). In the single-criterion optimization, this mechanism can be too weak for efficiently maintain the local population in different basins of attraction of the local extrema, that differ to a great extent with the fitness value, but it should work well in the multi-criteria problems (see Sect. 2.4.4).

2.4 Other Variants of EMAS

In this section other, more sophisticated versions of evolutionary multi-agent systems are presented, namely immunological, co-evolutionary and multi-criteria and memetic ones.

2.4.1 Memetic EMAS

EMAS may be held up as an example of a cultural algorithms, where evolution is performed at the level of relations among agents, and cultural knowledge is acquired from the energy-related information. This knowledge makes it possible to state which agent is better and which is worse, justifying the decision about reproduction. Therefore, the energy-related knowledge serves as situational knowledge (see Sect. 1.3.2). Memetic variants of EMAS may be easily introduced by modifying evaluation or variation operators (by adding an appropriate local-search method).

The idea of memetic EMAS consist in putting together a local search technique and the evaluation or variation operators utilized in EMAS (see Fig. 2.7). Therefore, implementation of Baldwinian and Lamarckian memetics in EMAS may be easily carried out in the following way:

- Baldwinian memetics: this implementation is done in much the same way as in classical evolutionary computing where in the course of evaluation of a a certain individual, the actual returned fitness is computed for one of its potential descendants (after running the local search procedure in the genotype domain, starting from the evaluated individual). Usually this is an iterative process that involves

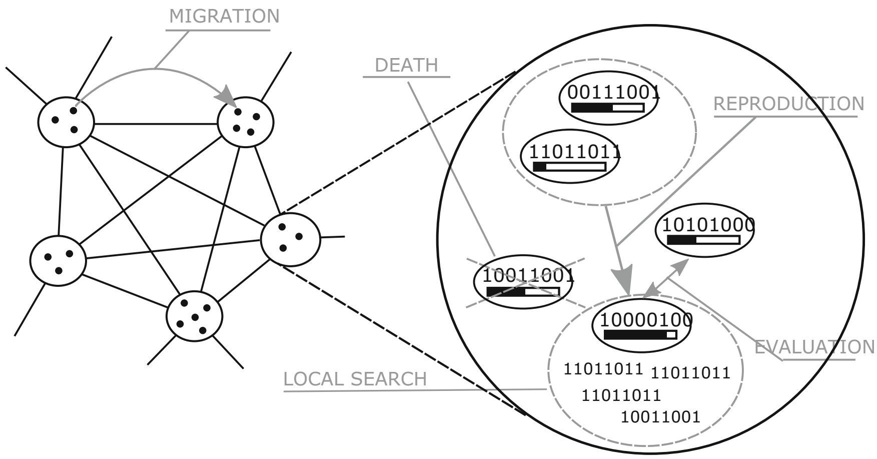

Fig. 2.7 Concept of memetic evolutionary multi-agent system (memEMAS)

many mutations, or using any other dedicated local search technique. The result returned is the fitness of the best encountered genotype, while the genotype of the evaluated individual remains unchanged.

- Lamarckian memetics: agent may improve its genotype (by running a local search procedure in the genotype domain, starting from the evaluated individual, similarly as in the case of Baldwinian memetics). Usually this is an iterative process, involving many mutations, or using any other dedicated local search technique. The result returned is the best encountered genotype. In the case of Lamarckian memetics, the genotype of the evaluated individual is changed. This procedure may be performed either during the reproduction or at the arbitrarily chosen moments of agent's life, depending solely on agent's own decision.

2.4.2 Immunological EMAS

Solving difficult search problems with population-based approaches, especially those with costly evaluation of their candidate solutions (e.g. inverse problems [9]), requires looking for techniques that may make it possible to increase the search efficiency. One method can be reducing the number of fitness function calls. It may be done in several ways, e.g. by applying tabu search [7] or, strictly in a technical layer, by caching fitness values generated for identical (or similar) genotypes. In this section, an immunological selection mechanism designed for EMAS is discussed.

The main idea of applying immunological inspirations to speed up the process of selection in EMAS is based on the assumption that "bad" phenotypes come from "bad" genotypes. Thus, a new group of agents (acting as lymphocyte T-cells) may be introduced (see works of Byrski and Kisiel-Dorohinicki [44, 45] and Ph.D. thesis of Byrski [39]). They are responsible for recognizing and removing agents with genotypes similar to the genotype pattern possessed by these lymphocytes. Another approach may introduce specific penalty applied by T-cells for recognized agents (certain amount of the agent's energy disappears) instead of removing them from the system. The general structure of iEMAS (immunological EMAS) is presented in Fig. 2.8.

Of course, there must exist some predefined affinity (lymphocyte-agent matching) function which may be based, e.g. on the percentage difference between corresponding genes. Lymphocytes are created in the system after the action of death. The late agent genotype is transformed into lymphocyte patterns by means of mutation operator, and the new lymphocyte (or a group of lymphocytes) is introduced into the system.

Antibodies in the real immune system of the vertebrates are subjected to the *negative selection* process in thymus, where they test other cells belonging to the organism—if during this time they recognize the "self" cell, they are considered as infeasible and removed. This mechanism has to counteract too quick realization of the removing of the antigens, that may turn out to be "self"' cells indeed. Within a specified period of time, the affinity of immature lymphocytes' patterns towards

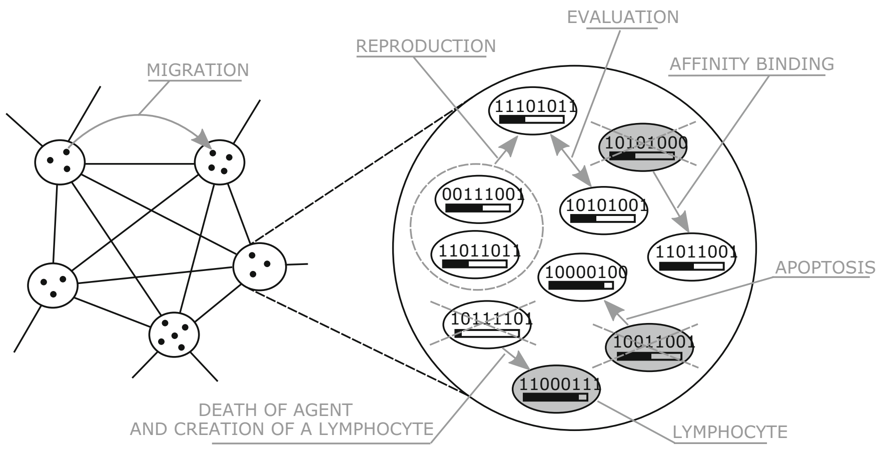

Fig. 2.8 Structure and the principle of work of immunological variant of EMAS (iEMAS)

"good" agents (possessing relatively high amount of energy) is tested. If it is high (lymphocytes recognize "good" agents as non-self), they are removed from the system. If the affinity is low, it is assumed that they will be able to recognize non-self individuals ("bad" agents) leaving agents with high energy intact.

The antibodies are subject of the *apoptosis* process, that is implemented in order to retain the capability of forgetting and adaptation by the immune system. A similar process in realized in the described agent system. The lifespan of lymphocytes is controlled by specific, renewable resource (strength), used as a counter by the lymphocyte agent, and with its depletion the immune cell is removed from the system.

Summing up, the cycle of the lymphocyte agent (T-cell) is realized as follows:

1. During removal of the computing agent, in the system there are created one or more of the T-cells. Each of these cells obtains the pattern of the solution of the removed agent (it can be mutated though). Thanks to this the T-cells become a vessel for potentially "bad" solutions.
2. Immature T-cells test the agents for some time (realizing negative selection), and if they recognize an agent with relatively high level of the energy, they are removed from the system. After finishing this process they work in an unconstrained manner.
3. T-cell owns a pattern of a potentially "bad" solution of the problem (its structure is identical as the structure of the genotype) and uses certain probability measure seeking similar solutions in the system, in order to remove them or to penalize the agents that own these solutions (in the simplest case this "similarity" or "affinity" function may be realized as Euclidean distance in the domain of continuous optimization, or Hamming distance in discrete optimization).
4. T-cell after finishing its negative selection process may work in the system during a certain period of time. After finishing this, it is removed from the system, however

this time may be extended by rewarding the T-cell for proper recognitions of the agents.

iEMAS is an example of a cultural algorithm, which is an extension of EMAS. However, based on the agent energy-related information the population of lymphocytes is modified (they are considered mature or not). Therefore, the agent energy-related knowledge serves as situational knowledge (see Sect. 1.3.2). Lymphocyte energy serves as temporal knowledge, so the lymphocyte may be removed after a certain period of time.

Removing from the population the solutions infeasible or "bad" (from the point of view of the current goal) can be perceived as similar to the tabu search algorithms [8]. However it should be noted, that in the proposed approach there is no a globally accessible set of "tabu" solutions, instead the introduced T-cells act locally, searching for infeasible solutions in its neighborhood. This can be perceived as a somehow "softened" tabu search.

The early removing of "bad" solutions and a decrease in the number of the individuals in the computing populations (see the computing results later in this section) makes iEMAS a weapon of choice to deal with problems, where a complex fitness function is used. Indeed, an interesting optimization task was approached with iEMAS, namely the evolution of neural network architecture [44, 47] and benchmark function optimization [45].

2.4.3 Co-evolutionary Multi-agent Systems

The concept of introducing into EMAS man co-evolving species (or sexes) [90, 91] is aimed at maintaining a significant diversity of the population by developing sub-population localized in the basins of attraction of the local extrema, as it is realized in the niching and speciation techniques. This approach can also be used for solving multi-modal optimization problems.

The starting point for the further considerations on *co-evolutionary* multi-agent systems (CoEMAS) is the observation, that similarly to the selection mechanism, it is quite difficult to apply classic co-evolutionary techniques in EMAS, because of the implicit characteristics of the multi-agent systems (cf. Sect. 2.2.1). At the same time one can use the existing concepts connected with existence of the system space and resources in order to introduce allopatric speciation and competition for the limited resources, and decentralized characteristics of the interaction among the agents may easily depict the existing influences between the species (or sexes).

The simplest solution leading to of allopatric speciation is introduction of high cost (meant as loosing the life energy) of migration among the islands [91]. The speciation process is based in this case on the geographical isolation of sub-populations, that may have chance to group the solutions from the neighborhoods of the basins of attraction of different local extrema, though in this case there exists no mechanism that could counteract localization of the same extrema by different populations.

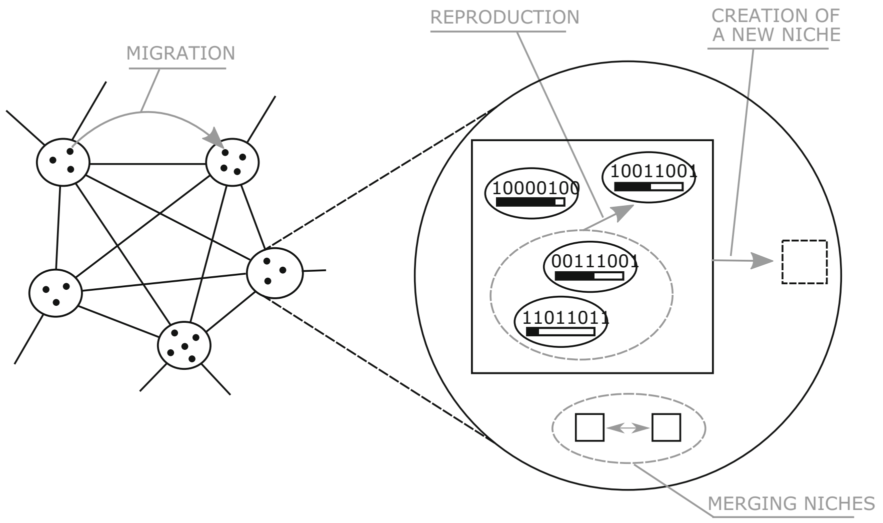

Fig. 2.9 Co-evolutionary multi-agent system (CoEMAS) with co-evolution of species

In the CoEMAS with *co-evolution of species* there exist two types of agents and in fact, two organization levels of the system: agents representing solutions live "inside" the *niche* agents, so they divide the population into two isolated (from the reproduction point of view) populations (see Fig. 2.9). These niches interact among themselves (a competition for limited resources may occur, migration among the islands and merging of niches) and the solutions (by the migration of solutions among the niches), so their solutions are localized in the basins of attraction of different local extrema.

CoEMAS with *a sexual selection* allows for existing of agents of many species and inside the species—many sexes on the same organizational level of the system. The competition for the limited resources along with the phenomenon of conflict and co-evolution of sexes (higher reproduction costs in the case of females that prefer the partners localized in the basin of attraction of the same local extremum) lead to sympatric speciation—creation of groups of agents (species), carrying solutions from the basins of attraction of different local extrema [93].

Application of *mutualism* mechanism (cooperation) leads to introducing into the system several cooperating species of individuals, which cooperation may consist in e.g. solving sub-problems of the same problem. Thus the complete group of all individual belonging to all species may produce the complete solution of the considered problem [94, 98].

In the co-evolutionary multi-agent system with *predator-prey* mechanism, there exist two species of agents: the prey have encoded in the genotypes the solution of the problem, while the predator eliminate from the system the prey having relatively low level of energy [100].

In the all above-referenced variants, the mechanisms applied support stable maintaining of the higher diversity of the solutions, giving similar effects as the ones observed in classic co-evolutionary techniques.

2.4.4 EMAS in Multi-criteria Optimization

A very interesting application of the concept of evolutionary multi-agent system is the task of multi-criteria optimization in the Pareto sense. The goal of the system is in this case finding a set of solutions that may be treated as a certain approximation of the Pareto front for the considered poly-optimization problem. In the simplest case, evolving population of agent may represent the set of feasible solutions of the problem described by the embedded into the system criteria functions. The most crucial element of the evolution process in this case is the meeting strategy, realized in the case of poly-optimization according to the dominance relation. In this way the agents representing the non-dominated solutions gain the energy and may undertake the reproduction process with higher probability. At the same time the dominated agents loose their energy, and in the course of time are eliminated from the system. Because of that, in the subsequent evolution stages, the set of the non-dominated agents should represent subsequent approximations of the Pareto front of the considered problem [83, 176].

Unfortunately, EMAS, similarly to other evolutionary algorithms, requires additional mechanism for maintaining diversity of the population on such level, that possibly uniform sampling of the whole Pareto front could be realized. One of ways that can be applied is taking advantage of the crowding technique (sf. Sect. 2.3.4). This mechanism is effective especially effective in the case of solving the problems with non-uniform Pareto front, because of more intensive exploration of the search space [170, 178, 268].

At the same time the described process may lead to the stagnation phenomenon (lack of directed evolution), when after passing a certain time, the population is composed of nearly agents representing nearly identical (mutually non-dominated) solutions, causing loss of energy flow during meetings. In this case one can try to apply elitism mechanisms that can be realized in EMAS in the variant described as archive.

Elitist Variant

It is noteworthy, that in the context of general understanding of elitism as an algorithmical feature, consisting in preserving the already found solutions, the above-described basic EMAS variant may be perceived as elitist. This is because the selection based on resources allows any agent to remain in the system, until it encounters sufficient number of better agents, that during the energy transfers its energy level will fall down below the death threshold. In order to differentiate the presented elitist variants, this one is called *semi-elitist* one.

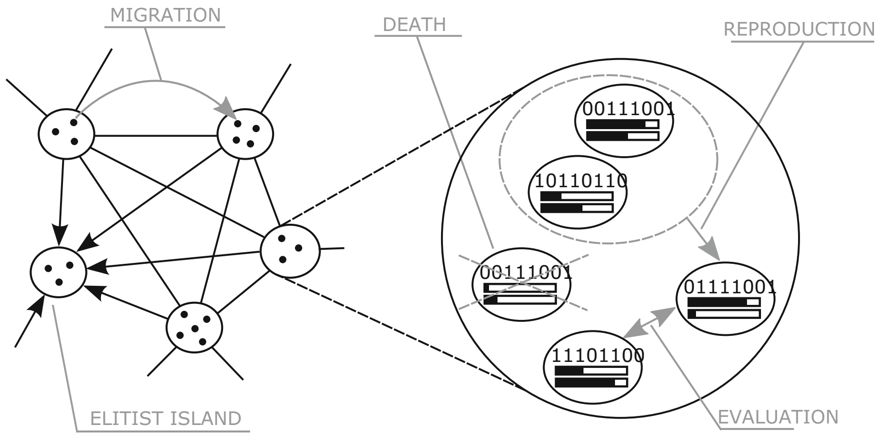

Fig. 2.10 Concept of elitist evolutionary multi-agent system (elEMAS)

The concept of *elitist* evolutionary multi-agent system (elEMAS) consists in introducing so-called *elitits* island [262], into the computing system structure, that becomes a certain archive (see Fig. 2.10). On the elitist island the agents do not confer to "regular" evolution rules, realizing only a special, predefined meeting strategy. If during the meeting it turns out, that the solution represented by one of the agents dominates the solution of the other agent, the dominated agent performs the death action at once. Thus one can assume that on the elitist island, only non-dominated agents may be found, i.e. representing the approximation of the Pareto front.

A crucial assumption which must me made here is that only the agents representing high-quality solutions are transferred to the elitist island. In order to make such assessment possible, in elEMAS an additional resource was introduced, so-called *prestige*, that is gained by the agent if it dominates another agent during the meeting realized on one of regular islands. If the agent passes so-called elitist threshold (measured by the prestige resource) it means that compared to other agents, its solution is valuable and it can be transferred to the elitist island. The final decision on this transfer is undertaken by the agent based on the knowledge about other similar solutions—the migration is possible only if the agent has identified sufficient number of similar solutions (this may be compared to crowding mechanism. A side-effect of this decision is elimination of such agent from the evolution process, thus increasing the chances of other agents that should enhance the diversity of the population [264–266].

Co-evolutionary Techniques

The applications of the EMAS to multiobjective optimization encompass also the previously mentioned co-evolutionary variants—using parasitism [96], mutualism [101], predator-prey [97] and sexual selection [95, 99].

2.5 Summary

In this chapter a review of the basic issues connected with structure and behavior of the described class of computing systems hybridizing the evolutionary and agent-oriented paradigms were presented. The authors aimed at showing the simplicity and features of the basic version of evolutionary multi-agent system (EMAS), and at the same time, its plethora of proposing potential extensions, following different nature-based inspirations. Because of that, besides basic EMAS, in this section different its extensions were presented (immunological, co-evolutionary, elitist and memetic).

The developed new metaheuristic algorithm has been present in publication from 1996. However after recent influential paper by Sörensen [270]. We would like to claim that EMAS directly hybridizes agent-based and evolutionary paradigms, utilizes in full the already present terminology and tackles many different areas and problems. Therefore as far as we know the makers of EMAS (to whom we also belong) did their best not to fall into category of so-called “novel” metaheuristics identified and fought against by Sörensen.

Decentralization of the evolution process can yield considerable advantages in application to solving certain problem classes and using particular mechanisms [43]. However besides the presented computational features of the techniques, the authors are far from claiming that they are the best, universal optimization “solvers”. One has to still remember about the well-known *no free lunch theorem* [297], stating, that the best solutions are obtained by using a method dedicated for solving certain classes of problems.

The presented outline of different agent-based evolutionary computing concepts does not cover many aspects of the total characteristics of the presented problems. On this stage the main goal was to formulate general principles of their structure and behavior and sketching out the domain of their applications. At the same time many additional issues may arise, connected with analysis and design of such complex systems (such as implementation features or formal analysis). These will be tackled in the next sections.