Evolutionary Algorithms Based on Game Theory and Cellular Automata with Coalitions

Article *in* Intelligent Systems Reference Library · January 2013

DOI: 10.1007/978-3-642-30504-7_19

CITATIONS
11

READS
28

4 authors:

Bernabe Dorronsoro
Universidad de Cádiz
179 PUBLICATIONS **3,358** CITATIONS
SEE PROFILE

Juan Carlos Burguillo
University of Vigo
192 PUBLICATIONS **1,487** CITATIONS
SEE PROFILE

A. Peleteiro
Zalando Dublin, Ireland
33 PUBLICATIONS **470** CITATIONS
SEE PROFILE

Pascal Bouvry
University of Luxembourg
556 PUBLICATIONS **6,129** CITATIONS
SEE PROFILE

Evolutionary Algorithms based on Game Theory and Cellular Automata with Coalitions

Bernabé Dorronsoro, Juan C. Burguillo, Ana Peleteiro, and Pascal Bouvry

Abstract .

Cellular genetic algorithms (cGAs) are a kind of genetic algorithms (GAs) with decentralized population in which interactions among individuals are restricted to the closest ones. The use of decentralized populations in GAs allows to keep the population diversity for longer, usually resulting in a better exploration of the search space and, therefore in a better performance of the algorithm. However, the use of decentralized populations supposes the need of several new parameters that have a major impact on the behavior of the algorithm. In the case of cGAs, these parameters are the population and neighborhood shapes. Hence, in this work we propose a new adaptive technique based in Cellular Automata, Game Theory and Coalitions that allow to manage dynamic neighborhoods. As a result, the new adaptive cGAs (EACO) with coalitions outperform the compared cGA with fixed neighborhood for the selected benchmark of combinatorial optimization problems.

1 Introduction

Evolutionary algorithms (EA) are well-known population based metaheuristics [BFM97, GK03, OZ06]. They work on a set of solutions (called *population*), evolving them simultaneously towards (hopefully) better ones by applying some

Bernabé Dorronsoro
Interd. Centre for Security, Reliability and Trust, Luxembourg e-mail: bernabe.dorronsoro@uni.lu

Juan Carlos Burguillo
ETSET. Universidad de Vigo. 36310-Vigo, Spain e-mail: J.C.Burguillo@det.uvigo.es

Ana Peleteiro
ETSET. Universidad de Vigo. 36310-Vigo, Spain e-mail: apeleteiro@gti.uvigo.es

Pascal Bouvry
Université du Luxembourg, Luxembourg e-mail: pascal.bouvry@uni.lu

stochastic operators (typically called *evolutionary operators*, e.g., selection, recombination, and mutation). However, it is well accepted in the literature that EAs perform a fast convergence, getting stuck in local optimal solutions, when dealing with complex problems. In order to avoid this fast convergence, it is usual to decentralize the population, what helps in keeping the diversity of solutions for longer [AT02a, AT02b].

There are two main ways for decentralizing the population in EAs, namely island or coarse-grained EAs (dEAs) [AT02a], and cellular or fine-grained EAs (cEAs) [AD08]. On the one hand, in dEAs, the population is split into several smaller sub-populations that are independently evolved by EAs, and exchanging some information (typically the best solution found so far) with other sub-populations. Thus, the EAs in the different islands will perform a fast convergence to hopefully distinct regions of the search space, preserving this way the overall population diversity. The information exchange among islands allows them to benefit from the exploration performed by the others and to introduce diversity in the local sub-populations. On the other hand, in the case of cEAs, the individuals composing the population are arranged in a usually 2-dimensional toroidal mesh, and only next individuals can interact in the breeding loop. Diversity is preserved thanks to the isolation by distance introduced with the use of neighborhoods. The effect is the formation of niches in the population exploring different regions of the search space. At the same time, these niches are not isolated, since due to the neighborhood overlapping, individuals located at the borders of the niches can interact and exchange information.

Therefore, the goal of structuring the population in EAs is to somehow preserve the population diversity for longer, typically at the cost of slowing down the convergence speed of the algorithm. This could be a good strategy for complex multimodal and/or epistatic problems for which a too exploitative behavior results in quick diversity loss in the population, thus the algorithm is stuck in some local optimum from which it cannot scape. However, for some other problems, this fast convergence speed could be desirable, so EAs with panmictic populations would find better solutions in shorter times with respect to other EAs with decentralized populations.

This chapter proposes several contributions considering recent approaches in the areas of Evolutionary Algorithms, Spatial Cellular Automata, Evolutionary Game Theory and Coalitions. First, we provide an introduction to all these areas describing the current state of the art. Then, we consider the integration of all these topics in a way to obtain a synergy in the development of Evolutionary Algorithms, with the advantages of dEAs and cEAs, but avoiding their particular drawbacks and removing their typically required parameters, like the neighborhood to use (cEAs) or the islands connectivity topology, the migration frequency, and the policies to exchange and discard individuals (in dEAs). This can be done thanks to the use of spatial cellular approaches with neighborhoods, and allowing the formation of coalitions among cells as a way to create islands of evolution in order to preserve diversity. Besides, we rely on Evolutionary Game Theory to consider every cell as a player of a game arranged in a two dimensional torus. Cells will be able to evolve depending

on their payoff with respect to their neighbors, and have also the support provided by their coalition members. This approach allows the payoff of a given solution to be defined in terms of how much such solution has improved in the last generations. The idea is to speed up the evolution of individuals grouped in high-quality coalitions that are quickly converging to promising solutions.

The chapter is structured as follows. Sect. 2 introduces the concept of population topology in Evolutionary Algorithms. Sect. 3 provides a short introduction of Game Theory and Coalitions. Then, Sect. 4 introduces Evolutionary Algorithms, and describes our new EACO algorithm, which is the main contribution of this chapter. Afterwards, Sect. 5 presents a set of complex combinatorial problems we used in our experiments to compare EACO versus the canonical cGA. Sect. 6 presents the promising results obtained by EACO, and in Sect. 7 we outline some conclusions and future work.

2 Population Topologies for Evolutionary Algorithms

In this section we review the main existing decentralized population topologies that have been proposed for EAs. There is a large number of papers in this topic, and it is not the scope of this section to revise all of them, but only some of the most outstanding ones.

The influence of using fine and coarse-grained populations in EAs has been deeply investigated in the literature [Ste93, CP00, AT02a, Alb05, NAd06, Tal06, AD08, Tom05]. As we already mentioned, they are only two boundary cases of decentralized populations. Recently, several works appeared studying new population topologies that share properties of both models. They are discussed below.

2.1 Enhanced Cellular Topologies

In this section we address several papers that propose enhancements on cGAs that modify the algorithm dynamics, but keeping the original cellular population topology. Simoncini et al. [SVCC06] propose the use of an anisotropic selection that is giving higher priority to some of the individuals in the neighborhood to be chosen against others. Specifically, those individuals at the north and south positions will be more likely to be chosen than the individuals in the east and west locations. Ishibuchi et al. [ISTN11] propose a cGA with two different neighborhood structures: one for selection, as in the case of a canonical cGA, and the second one for replacement. This way, the new offspring is considered to be inserted in the whole replacement neighborhood, instead of considering just the current individual. This model is accelerating the algorithm convergence speed, while Simoncini et al. one is reducing it.

In 2002, Li and Sutherland presented in [LS02] a variation of cGA called prey/predator algorithm, where the preys (corresponding to the individuals representing potential solutions to the problem) move freely around the positions of the grid, mating with neighbor preys in each generation. Moreover, there exists a number of predators which are continuously displacing around the population, and they kill the weakest prey of their neighbor in each generation.

Finally, Alba et al. propose in [AMD+06] a new Estimation of Distribution Algorithm (EDA) with a cellular population in which small subpopulations are located in every location of the lattice, instead of only one single individual. This is done because the EDA needs large populations to get enough information to estimate the distribution of the solutions of the variables, therefore adding small subpopulations in every lattice will multiply the number of solutions in the neighborhood.

2.2 Hierarchical Populations

Janson et al. proposed in [JM05] a hierarchical Particle Swarm Optimization method (H-PSO) in which individuals are arranged in a tree hierarchy. Therefore, better solutions move towards the highest levels of the hierarchy, exchanging their position with worse ones. In this hierarchy, particles are influenced by their personal best solution and by its parent in the hierarchy. Later, in [JADM06], the authors propose a new cellular Genetic Algorithm (cGA) with a pyramidal hierarchy into the population, such that the best individuals will be placed in the highest levels of the hierarchy. Therefore, individuals interact in this case with more than one individual from the hierarchy, as determined by the neighborhood defined in the cellular population. The effect is that the exploitation of the most promising solutions is enhanced, since they are located next to each other in the population thanks to the hierarchy, and the cellular population promotes the interaction of neighboring individuals. At the same time, the diversity of solutions in the population is maintained due to the evolution of worse individuals at the lower levels of the hierarchy, promoting the exploration of other regions of the search space different than the ones where the most promising current solutions are.

2.3 Population Structures Based on Social Networks

There exist some works analyzing new fine-grained topologies that are not as connected as the panmictic population (which is fully connected), but have shorter characteristic path length (i.e., the maximum distance between any two solutions) than the cellular model. In particular, it is worth mentioning the studies made by Giacobini et al. to both theoretically [GTT05] and empirically [MG06] analyze the behavior of different GAs using random, scale-free, and small-world topologies. Additionally, Payne et al. addressed in [PE06] another theoretical study on the be-

havior of GAs with scale-free and small-world topologies, and they later extended it in [PE08] to analyze the effects of some characteristics of these kinds of networks, like the scale and assortativity (the degree of connections of nodes). They arrived to the conclusion that increasing the assortativity leads to shorter convergence times, while high scale networks provide longer convergence rates. However, the main conclusion of these studies by Giacobini et al. and Payne et al. is that small-world populations are competitive, but the potential utility of scale-free population structures is still unclear. In [DB11c], the influence on the behavior of the algorithm under different small-world topologies generated in several different ways is analyzed.

Despite random graphs were not the best performing ones in the studies made by Giacobini et al. for GAs [GTT05, MG06], they have become popular for PSO algorithms. Indeed, this is the population model used in the Standard PSO 2007 (SPSO 07)[Sta]. Kennedy and Mendes [KM02, KM06] have deeply investigated on the use of fine-grained population topologies (e.g., ring, Von Neumann, random, pyramid, or star graphs, to name a few) for PSO algorithms. As a conclusion of their studies, they recommend the use of Von Neumann topologies. In [MKN04], the same authors present a new fully informed PSO algorithm in which the information of all the neighbors is taken into account in the generation of new solutions.

In [DB11b], Dorronsoro and Bouvry compare the behavior of several Differential Evolution (DE) algorithms with a number of different population topologies, like panmictic, cellular, island, hierarchical, random, or small-world. Among the compared topologies, the island and small-world populations were the best performing ones for the considered continuous problems.

2.4 Dynamic Topologies

There also exist several approaches using dynamic topologies. In this sense, Sunganthan presented in 1999 [Sug99] probably the first EA with a dynamic population topology: a PSO algorithm with variable neighborhood sizes. The idea is that every particle starts with a neighborhood of size one (the particle itself), and then the size of the neighborhood is increasing during the run. One year later, Kennedy [Ken00] proposed splitting the population into clusters (groups of swarms), where the centroid of every cluster is the one influencing the other particles in its cluster. A similar concept is the clubs-based PSO [EEB07], but in this case particles can belong to different clubs (or clusters), and they can also join to or quit from a club. A third algorithm we would like to mention here is the species-based PSO (SPSO), proposed by Li in 2004 [Li04]. In this case, the population is split into swarms of the same species, i.e., containing particles with some similarities. Only those particles of the same species interact in the creation of the next generation of solutions.

In 2008, Whitacre et al. proposed in [WSP08] a dynamic population structure for GAs that automatically evolves by following a set of simple rules to mimic the interaction networks of complex systems. Individuals are dynamically created or

removed from the population in terms of their quality by following some rules. The population is modeled as a graph, and therefore its structure is self-evolving when adding or removing solutions. As a result, this kind of population performs longer convergence times compared to cGAs. However, as a consequence, they can provide better results after a high number of evaluations. Similar topologies were proposed for PSO by Godoy et al. [GV09] and Clerc [Cle06].

Dorronsoro et al. proposed different self-adaptive cGAs that automatically modify the population topology to a more suitable one according to the convergence speed of the population. In [AD05], the shape of the population is changed based on the principle that narrower populations provide with more exploration capabilities than square ones, that are more exploitative. More recently, a new cGA that is using different neighborhood sizes according to the quality of individuals was proposed in [DB11a].

Finally, there are two recent papers [YML09, CG10] that incorporate a novel idea in which the evolution process of the cGA is guided by a cellular automaton (CA). In particular, the CA is used to activate or deactivate the evolution of the individuals in the population topology at each generation according to the cellular automaton state. This is done to slow down the convergence speed of the algorithm, thus keeping diversity for longer.

3 Game Theory and Coalitions

In this section, we provide a brief introduction to some relevant concepts of Game Theory and then we introduce coalitions, that became a relevant research topic derived from Game Theory, but nowadays they are used in many contexts.

3.1 Game Theory

Game Theory [Bin94] provides useful mathematical tools to understand the possible strategies that individuals may follow when competing or collaborating in games. This branch of applied mathematics is used nowadays in the social sciences (mainly economics), biology, engineering, political science, international relations, computer science and philosophy. Game theory was developed extensively in the 1950s by many scholars and it was later explicitly applied to biology from the 1970s [Smi82], although similar developments go back at least as far as the 1930s [Fis30].

Initially, game theory was developed to analyze competitions in which one individual does better at another's expense: zero sum games [Mv47]. From that moment, traditional applications of game theory attempt to find equilibria in these games. In any equilibrium each player of the game adopts a strategy that they are unlikely to change. Many equilibrium concepts have been developed; among them we find the famous Nash equilibrium [Nas50].

Game theory has implications and uses for many real-world domains, including those involving automated agents. These domains encompass electronic commerce, auctions, and general resource allocation scenarios. As a result of the desire to embed game theoretic principles into agent systems, computational aspects of game theory and social choice have been extensively studied in recent years [BR08]. We could divide game theory in two main branches: non cooperative [Eht97] and cooperative game theory [Owe68, Kal91].

Non cooperative game theory, or competitive games [Nas51], assume that each participant acts independently, without collaboration or communication with the others. The player chooses its strategy for improving its own benefit. This has many applications as resource allocation [HL08] and congestion control [AB05] among others.

Cooperative game theory studies the behavior of the agents when they cooperate. This kind of games has been widely explored in different disciplines as economics or political science, but they are also used in other domains as in networking paradigms. Within cooperative games, we find coalition games, in which a set of players seek to form cooperative groups to improve their performance. Coalitions enable agents to accomplish goals they may not be able to accomplish independently. In [SZD+09], the authors group coalitional games into three major classes: canonical coalitional games, where the grand coalition of all users is an optimal structure and is of major importance; coalition formation games, where the network structure that forms depends on gains and costs from cooperation; and coalitional graph games, where the players' interactions are governed by a communication graph structure.

Evolutionary game theory (EGT) [Smi82] models the application of interaction dependent strategies in populations along generations. EGT differs from classical game theory by focusing on the dynamics of strategy change more than the properties of strategy equilibrium. EGT has become of increased interest to economists, sociologists, anthropologists, and philosophers [Bin94].

In game theory and behavioral ecology, an evolutionarily stable strategy (ESS) [Smi82] is a strategy which, if adopted by a population of players, cannot be invaded by any alternative strategy. An ESS is a Nash equilibrium which is "evolutionarily" stable, meaning that once it is fixed in a population, natural selection alone is sufficient to prevent alternative (mutant) strategies from successfully invading.

In evolutionary games, participants do not posses unfailing Bayesian rationality. Instead they play with limited computing and memory resources. The only requirement is that the players learn by trial and error, incorporate what they learn in future behavior, and die or somehow change if they do not.

3.2 Coalitions

Coalitions are an important way of cooperation through which multi-agent systems improve their performance, accomplish their assignments, or increase their bene-

fits, among others. Coalition formation is useful as it may increase the ability of agents to accomplish tasks and achieve their goals [Li07], and it is important for distributed applications ranging from electronic business to mobile and ubiquitous computing, where adaptation to changing resources and environments is crucial. It is used in domains as electronic commerce [FRA05], sensor networks [GSS08] or communication networks [SZH+11].

Coalition formation has been treated from the sociological point of view, since it is a pervasive aspect of social life [Gam61]. The formation and stability of coalitions depend on the rules of coalition formation proposed [Yi97]. For example, an infinite-horizon game in which a coalition forms if and only if all the members agree to form the coalition is proposed in [Blo95, Blo96]. In [RV97], the coalitions can break up into smaller coalitions only, and in [Yi92], non-members can join an existing coalition without the permission of the existing members.

Coalition formation is also addressed in game theory. However, the game theoretic approach is normally centralized and computationally intractable. In [SAK10], the authors re-examine the computational complexity of the different coalition formation problems when assuming that the number of agent types is fixed. Besides, regarding the computation tractability of algorithms, the authors study in [SSJ98] the implementation of distributed coalition formation algorithms within a real-world multi-agent system, presenting the problems that arise when attempting to utilize the theoretical coalition formation algorithms for a real-world system.

In [LXY+09], the authors classify the typical algorithms of coalition formation in multi-robot systems into three kinds: first ones are the deterministic search algorithms, as for example in Shehory [SK98]. The second ones regard the task allocation algorithms, as in Parker's ALLIANCE [Par98]. The third approach is based in Evolutionary Algorithms [LC06, SGW06]

As we said before, coalition formation can become intractable since all possible shapes of coalitions depend exponentially on the number of agents. Finding the optimal partition of agents set by checking the whole space may be too expensive. One possible solution could be using Genetic Algorithms (GA), as in [YL07], where they present a GA-based algorithm for coalition structure formation which aims at achieving goals of high performance, scalability, and fast convergence rate simultaneously. Other possibility is to use Evolutionary Algorithms, as in [GK08], that introduces the solution of Coalition Formation Problem (CFP) in Multi-Agent Systems (MAS). It consists on the use of a evolutionary algorithm by developing an evolutionary based algorithm for creation of coalitions of agent for solving assumed tasks. In [LXY+09], the Quantum Evolutionary Algorithm is proposed for solving coalition formation in dynamic environments, where a skillful Quantum probability representation of chromosome coding strategy is designed to adapt to the complexity of the multi-robot coalition formation problem. In [SGW06], they present a coalition structure optimization algorithm in MAS based on Particle Swarm Optimization (PSO). They try to maximize the summation of the coalition values, and to search for an optimized coalition structure in a minimal searching range.

Optimizing the coalition structure n-skill games is also another way to find the best coalition structure to improve gains when each agent has to perform a task. The

Coalitional Skill Games (CSGs) are a simple model of cooperation among agents. This is a restricted form of coalitional games, where each agent has a set of skills that are required to complete various tasks. Each task requires a set of skills in order to be completed, and a coalition can accomplish the task only if the coalition's agents cover the set of required skills for the task. In [BR08], the authors consider the computational complexity of several problems in CSGs. Also they study the problem of coalition structures in CSGs in [BMJK10], and they propose a Fixed Parameter Tractable (FPT) algorithm.

The coalition formation is also imported in multi-robot cooperation domain. For example in [LC06], the agent coalition cooperation in MAS (Multi-Agent System) is applied in multi-robot system with Genetic Algorithm for the formation of multi-robot coalition and coalition structure in order to gain the maximum possible coalition value during task execution. Also, a reasoning system that enables a group of heterogeneous robots to form coalitions to accomplish a multi-robot task using tightly coupled sensor sharing is presented in [PT06]. [CYCS10] investigates the Multi-robot Task Allocation (MRTA) problem for a group of heterogeneous mobile robots using a leader-follower based coalition methodology.

Implementing cooperation in large scale communication networks faces several challenges such as adequate modeling, efficiency, complexity, and fairness. Coalitional games prove to be a very powerful tool for designing fair, robust practical and efficient cooperation strategies among networks. In [SZD+09], the authors present the state-of-the-art of research contributions, from game theory and communications, that addresses the major opportunities and challenges in applying coalitional games to the understanding and designing of modern communication systems, with emphasis on both new analytical techniques and novel application scenarios.

Traditional models assume that each agent participates in exactly one coalition. However, it is common that in real-life one agent can participate in various groups and perform one task in each of them. Overlapping Coalition Formation (OCF) games are cooperative games where the players can simultaneously participate in several coalitions. In [SK96], the authors present algorithms for iterative formation of an overlapping coalition, and they show that agents can benefit from it. In [DDRJ06], the authors consider overlapping coalition formation in sensor networks. In these two papers, it is assumed that the agents are fully cooperative, which is not the case in general, since agents tend to maximize their own benefit. For that, Chalkiadakis et al. propose in [CEMJ08, CEM+10] a game theoretic model for overlapping coalition formation. In [ZE11], the authors propose a unified framework for the study of stability in the OCF setting.

4 Evolutionary Algorithms with Coalitions

We propose a new EA with Coalitions (EACO), designed on a dynamic topology that is taking advantage of both cellular and island population models. It is basically a cellular EA in which coalitions of individuals are automatically formed and main-

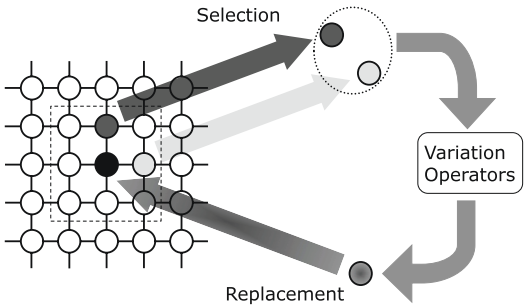

Fig. 1 In cellular EAs, individuals are only allowed to interact with their neighbors

tained. Individuals in the population will choose the coalition to join at every time in terms of some rewards they can expect from the coalition. Neighborhoods are defined by these coalitions. Therefore, coalitions can be understood as small panmictic subpopulations (i.e., islands) embedded in the cellular topology.

Section 4.1 briefly introduces the working principles of cellular EAs, while Section 4.2 presents a detailed description of the new EACO algorithm.

4.1 Cellular Evolutionary Algorithms

Cellular EAs [AD08, Whi93, MS89] are structured population algorithms with a high explorative capacity. The individuals composing their population are arranged into a (usually) two dimensional toroidal mesh, and only neighbor individuals (i.e., the closest ones measured in Manhattan distance) are allowed to interact during the breeding loop (see Fig. 1). This way, we are introducing some kind of isolation in the population that depends on the distance between individuals. Hence, the genetic information of a given individual can be spread slowly through the grid (since neighborhoods are overlapped), and it will need a high number of generations to reach distant individuals (thus preventing the population from premature convergence). Structuring the population in this way we achieve a good exploration/exploitation trade off on the search space, thus improving the capacity of the algorithm for solving complex problems [AT02a].

A canonical cEA follows the pseudo-code included in Algorithm 1. In this basic cEA, the population is usually structured in a regular grid of d dimensions ($d = 1, 2, 3$), and a neighborhood is defined on it. The algorithm iteratively considers as current each individual in the grid (line 3), and individuals may only interact with individuals belonging to their neighborhood (line 4), so parents are chosen among the neighbors (line 5) with a given criterion. Crossover and mutation operators are applied to the individuals in lines 6 and 7, with probabilities P_c and P_m, respectively. Afterwards, the algorithm computes the fitness value of the new offspring individual (or individuals) (line 8), and inserts it (or one of them) instead of the current

individual in the population (line 9) following a given replacement policy. This loop is repeated until a termination condition is met (line 2).

The cEA described here is asynchronous, since the population is updated with the next generation individuals just after creating them. This way, these new individuals can interact with those belonging to their parent's generation. On the contrary, there is also the possibility of storing all the offspring individuals in an auxiliary population, and then replace all the individuals in the population at the same time. This last version matches with the synchronous cEA model. As it was studied in [AD08, ADGT06], the use of asynchronous policies allows faster convergence of the population than in the case of the synchronous one.

Algorithm 1 Pseudocode for a canonical cEA

```
 1: //Algorithm parameters in 'cea'
 2: while ! StopCondition() do
 3:    for individual ← 1 to cea.popSize do
 4:       n_list←Get_Neighborhood(cea,position(individual));
 5:       parents←Selection(n_list);
 6:       offspring←Recombination(cea.Pc,parents);
 7:       offspring←Mutation(cea.Pm,offspring);
 8:       Evaluation(offspring);
 9:       Add(position(individual),offspring,cea);
10:    end for
11: end while
```

4.2 Description of EACO

As introduced before, EA with Coalitions (EACO) is a new class of EAs with dynamic population topology that aims at taking profit of the benefits of the two main existing population structures, namely cellular and island populations. This is achieved by introducing the concept of coalitions among individuals. Individuals are arranged in a toroidal lattice, as in cellular EAs, and they form coalitions according to some policies that will be defined next. All the individuals belonging to the same coalition can interact among them, like in the subpopulations of an island EA.

Individuals can only belong to one single coalition, and they will join to those coalitions from which they expect to get a maximum profit. In the same way, they can leave their current coalition if the benefit of belonging to it is low. When an individual leaves a coalition, it can either join another existing one or create a completely new one, depending on the expected gain. Therefore, individuals are considered to be selfish entities that are able to collaborate with others to look for benefits.

In order to further evolve to better solutions, individuals will be interested in mating with diverse solutions of good quality. Therefore, it would be beneficial for them to belong to coalitions composed by a large number of high quality individuals,

and representing a diverse set of solutions at the same time. The quality of a coalition is consequently evaluated in terms of its size, the average quality of the solutions forming it, and their diversity. This way, belonging to a high quality coalition will generally be beneficial for individuals.

To summarize, we list below the main characteristics of our model with coalitions:

- Larger coalitions are desirable, as cells have more candidates for mating in the next generation.
- Coalitions with higher diversity of solutions are desirable, as they have a higher genetic richness.
- Coalitions with better average fitness of solutions are desirable, as it indicates that they have a better genetic quality.
- Every coalition or independent cell has a global valuation, depending on the previous values, that determines its "quality".
- Solutions from a given coalition can interact with any other solution in the same coalition, so the coalition behaves like a panmictic island.
- Solutions can leave a coalition and join other neighboring ones or create a new coalition, according to their selfish behavior to maximize the expected benefit.
- Solutions can only belong to one single coalition, except if they are in the neighborhood of an independent cell. In this case, they behave as frontier cells and they can mate with this cell.
- Independent cells behave as in classical cEAs, i.e., they have the classical Von Neumann neighborhood. For the simplicity of the algorithmic model, we will consider this independent cell as a potential coalition, with only one member, that mates with its neighbors.
- There will be a parameter coefficient of independence ($Ind_c \in [0,1]$), that models the desire that cells have to remain independent.

In EACO, all the coalitions will evolve for a maximum number of iterations at every generation of the algorithm. Therefore, better coalitions will be rewarded with longer evolution processes. Inside a coalition, the evolution is performed as in regular EAs with panmictic populations; typically, the parents are selected from the population using some selection scheme, and then some variation operators are applied to them in order to generate a set of offspring solutions that are inserted into the population following some policy.

As commented before, every coalition (C_i) has a valuation, which is determined as follows,

$$Valuation(C_i) = \alpha \cdot Size(C_i) + \beta \cdot Var(C_i) + \gamma \cdot Avg(C_i) \ , \quad (1)$$

where $\alpha + \beta + \gamma = 1$ and these coefficients model the weight given to the size of the coalitions, the variance and the average quality of the solutions, respectively. This valuation will be used by cells to evaluate the quality of a coalition and move from one coalition to another.

Algorithm 2 describes the pseudocode of our EACO algorithm, which is relatively similar to the one described before for the cEAs case. In EACO, as happens

Algorithm 2 Pseudocode for EACO

```
1:  //Algorithm parameters in 'eaco'
2:  while ! StopCondition() do
3:      coa_value_list←Coalition_Valuation(coa_list);
4:     for individual ← 1 to eaco.popSize do
5:         n_list←Get_Coalition_Neighborhood(eaco,position(individual));
6:         parents←Selection(n_list);
7:         offspring←Recombination(eaco.Pc,parents);
8:         offspring←Mutation(eaco.Pm,offspring);
9:        Add(position(individual),offspring,eaco);
10:        coa_individuals←Change_Coalition(eaco,position(individual),coa_value_list, Ind_c);
11:    end for
12: end while
```

in cEA, the population is structured in a regular grid of d dimensions ($d = 1,2,3$), and a neighborhood is defined on it. The algorithm first evaluates the valuation of every coalition (line 3). At the beginning, all the cells are independent, so their valuation differences will be based only in the quality of those cell solutions. Then, the algorithm iteratively considers as current each individual in the grid (line 4), and individuals may only interact with individuals belonging to their Von Neumman neighborhood (line 5), in the independent case, or within the coalition neighborhood. Then, parents are chosen among those neighbors (line 6) with a given criterion. Crossover and mutation operators are also applied to the individuals in lines 7 and 8, with probabilities P_c and P_m, respectively. Afterwards, the algorithm computes the fitness value of the new offspring individual (or individuals) (line 9), and inserts it (or one of them) instead of the current individual in the population (line 9) following a given replacement policy. Finally, the cell can change from one coalition to another (line 10) depending on its present valuation and the independence coefficient (Ind_c). This loop is repeated until a termination condition is met (line 2).

5 Set of Problems

In this section, we present the set of problems chosen for this study. The benchmark is representative because it contains many different interesting features in optimization, such as epistasis, multimodality, deceptiveness, use of constraints, parameter identification, and problem generators. These are important ingredients in any work trying to evaluate algorithmic approaches with the objective of getting reliable results, as stated by Whitley et al. in [WRDM97].

Initially, we will experiment with the reduced set of problems studied in [AT00], which includes the massively multimodal deceptive problem (MMDP), the frequency modulation sounds (FMS), and the multimodal problem generator P-PEAKS. Next we will extend this basic three-problem benchmark with COUNTSAT (an instance of MAXSAT), error correcting code design (ECC), maximum cut of a graph (MAXCUT), and the minimum tardy task problem (MTTP). The election of

this set of problems is justified by both their difficulty and their application domains (combinatorial optimization, continuous optimization, telecommunications, scheduling, etc.). This let us guarantee a high level of confidence in the results, although the evaluation of conclusions will result more laborious than with a small test suite.

The problems selected for this benchmark are explained in subsections 5.1 to 5.5. We include the explanations in this paper to make it self-contained and to avoid the typical small lacks that could preclude other researchers from reproducing the results.

5.1 Massively Multimodal Deceptive Problem (MMDP)

The MMDP is a problem that has been specifically designed to be difficult for an EA [GDH92]. It is made up of k deceptive subproblems (s_i) of 6 bits each one, whose value depends on the number of ones (*unitation*) a binary string has (see Fig. 2). It is easy to see (graphic of Fig. 2) that these subfunctions have two global maxima and a deceptive attractor in the middle point.

Unitation	Subfunction value
0	1.000000
1	0.000000
2	0.360384
3	0.640576
4	0.360384
5	0.000000
6	1.000000

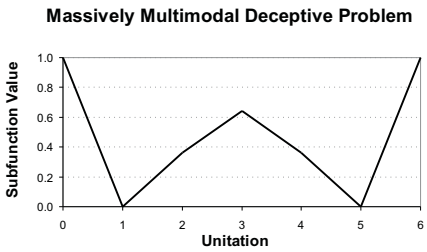

Fig. 2 Basic deceptive bipolar function (s_i) for MMDP.

In MMDP, each subproblem s_i contributes to the fitness value according to its *unitation* (Fig. 2). The global optimum has a value of k and it is attained when every subproblem is composed of zero or six ones. The number of local optima is quite large (22^k), while there are only 2^k global solutions. Therefore, the degree of multimodality is regulated by the k parameter. We use here a considerably large instance of $k = 40$ subproblems. The instance we try to maximize for solving the problem is shown in Eq. 2, and its maximum value is 40.

$$f_{MMDP}(\mathbf{s}) = \sum_{i=1}^{k} fitness_{s_i} \tag{2}$$

5.2 Multimodal Problem Generator (P-PEAKS)

The P-PEAKS problem [DPS97] is a multimodal problem generator. A problem generator is an easily parameterizable task which has a tunable degree of epistasis, thus admitting to derive instances with growing difficulty. Also, using a problem generator removes the opportunity to hand-tune algorithms to a particular problem, therefore allowing a larger fairness when comparing algorithms. With a problem generator we evaluate our algorithms on a high number of random problem instances, since a different instance is solved each time the algorithm runs, then the predictive power of the results for the problem class as a whole is increased.

The idea of P-PEAKS is to generate P random N-bit strings that represent the location of P peaks in the search space. The fitness value of a string is the number of bits the string has in common with the nearest peak in that space, divided by N (as shown in Eq. 3). By using a small/large number of peaks we can get weakly/strongly epistatic problems. In this paper we have used an instance of $P = 100$ peaks of length $N = 100$ bits each, which represents a medium/high epistasis level [AT00]. The maximum fitness value for this problem is 1.0.

$$f_{P-PEAKS}(\mathbf{x}) = \frac{1}{N} \max_{1 \leq i \leq p} \{N - HammingD(\mathbf{x}, Peak_i)\} \tag{3}$$

5.3 Error Correcting Code Design Problem (ECC)

The ECC problem was presented in [MS77]. We will consider a three-tuple (n, M, d), where n is the length of each codeword (number of bits), M is the number of codewords, and d is the minimum Hamming distance between any pair of codewords. Our objective will be to find a code which has a value for d as large as possible (reflecting greater tolerance to noise and errors), given previously fixed values for n and M. The problem we have studied is a simplified version of that in [MS77]. In our case we search for half of the codewords ($M/2$) that will compose the code, and the other half is made up by the complement of the codewords computed by the algorithm.

The fitness function to be maximized is:

$$f_{ECC}(C) = \frac{1}{\sum_{i=1}^{M} \sum_{\substack{j=1 \\ i \neq j}}^{M} \frac{1}{d_{ij}^2}}, \tag{4}$$

where d_{ij} represents the Hamming distance between codewords i and j in the code C (made up of M codewords, each of length n). We consider in the present paper an instance where $M = 24$ and $n = 12$. The search space is of size $\binom{4096}{24}$, which is approximately 10^{87}. The optimal solution for $M = 24$ and $n = 12$ has a fitness value of 0.0674 [CFW98].

5.4 Maximum Cut of a Graph (MAXCUT)

The MAXCUT problem is to look for a partition of the set of vertices (V) of a weighted graph $G = (V, E)$ into two disjoint subsets V_0 and V_1 so that the sum of the weights of the edges with one endpoint in V_0 and the other one in V_1 is maximized. For encoding the problem we use a binary string $(x_1, x_2, \ldots, x_n)$ of length n where each digit corresponds to a vertex. If a digit is 1 then the corresponding vertex is in set V_1; if it is 0 then the corresponding vertex is in set V_0. The function to be maximized [KBH94] is:

$$f_{MAXCUT}(\mathbf{x}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_{ij} \cdot \left[x_i \cdot (1 - x_j) + x_j \cdot (1 - x_i) \right] \tag{5}$$

Note that w_{ij} contributes to the sum only if nodes i and j are in different partitions. We have considered in this study three different instance graphs. Two of them are randomly generated graphs of moderate sizes: a sparse one "cut20.01", and a dense one "cut20.09"; both of them are made up of 20 vertices. The other instance is a scalable weighted graph of 100 vertices. The maximum fitness values for these instances are 10.119812 for "cut20.01", 56.740064 in the case of "cut20.09", and 1077 for "cut100".

5.5 Minimum Tardy Task Problem (MTTP)

MTTP [Sti87] is a task-scheduling problem wherein each task i from the set of tasks $T = \{1, 2, \ldots, n\}$ has a length l_i —the time it takes for its execution—, a deadline d_i —before which a task must be scheduled, and its execution completed—, and a weight w_i. The weight is a penalty that has to be added to the objective function in the event that the task remains unscheduled. The lengths, weights, and deadlines of tasks are all positive integers. Scheduling the tasks of a subset S of T is to find the starting time of each task in S, such as at most one task at time is performed and such that each task finishes before its deadline.

We characterize a one-to-one scheduling function g defined on a subset of tasks $S \subseteq T : S \mapsto \mathbb{Z}^+ \bigcup \{0\}$, so that for all tasks $i, j \in S$ has the following properties:

1. A task can not be scheduled before any previous one has finished: $g(i) < g(j) \Rightarrow g(i) + l_i \leq g(j)$.
2. Every task finishes before its deadline: $g(i) + l_i \leq d_i$.

The objective function for this problem is to minimize the sum of the weights of the unscheduled tasks. Therefore, the optimal scheduling minimizes Eq. 6:

$$f_{MTTP}(\mathbf{x}) = \sum_{i \in T - S} w_i \tag{6}$$

The schedule of tasks S can be represented by a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ containing all the tasks ordered by its deadline. Each $x_i \in \{0, 1\}$, where if $x_i = 1$ then task i

is scheduled in S, while if $x_i = 0$ means that task i is not included in S. The fitness function is the inverse of Eq. 6 , as described in [KBH94]. We have used in this study two different instances [KBH94] for analyzing the behavior of our algorithms with this function: "mttp100", and "mttp200", with sizes 100 and 200, and maximum fitness values of 0.005 and 0.0025, respectively.

6 Results

We present in Table 1 the results obtained in our experiments for every problem[1]. We compare the performance of EACO versus the canonical cellular GA, which was demonstrated to outperform other GAs with panmictic and decentralized populations in [AD08]. The results were obtained after performing 100 independent runs of every algorithm for all the problems. We did an exhaustive evaluation of the EACO algorithm, and we heuristically found that mainly it works better using: $Ind_c = 0.05$, $\alpha = 0.1$, $\beta = 0.1$ and $\gamma = 0.8$. This means that the coefficient with bigger weight in the valuation is the average functional value of the coalition.

For the two algorithms, in the table we show the best result found (or the percentage of runs in which the optimum was found), the average best solution found (unless the optimum is found in every run), and the average number of generations required to find the optimum. The best results are emphasized in **bold font**. In the last two columns, we present the results of the Wilcoxon unpaired signed rank sum statistical test in the comparison of both algorithms for the results found and the number of generations required. Those values lower than 0.05 (emphasized in **bold font in the table**) mean that differences between the algorithms are significant with 95% confidence. Those cases when EACO is significantly outperforming cGA according to this test are emphasized with dark grey background in Table 1, while light grey background stands for significantly better behavior of cGA.

We can see in the table that the two algorithms find always the optimal solution for ECC, MAXCUT20_01, MAXCUT20_09, MTTP100, and P-PEAKS. For the other 3 problems, EACO is more effective for MAXCUT100, but worse for the other two ones. However, if we pay attention to the average best solutions reported by the algorithms in the 100 independent runs, we can see that EACO is significantly more accurate than cGA for MAXCUT100, and there are not statistically significant differences for the other two problems.

Regarding the number of generations required to find the optimum, EACO is faster than cGA with statistical significance for most problems. The exceptions are MAXCUT100, since we did not find significant differences between the two algorithms, and MMDP, for which cGA is statistically faster.

Summarizing, the new EACO algorithm clearly outperforms the cGA in terms of efficiency (i.e., the number of generations required to find the optimum) and accuracy (i.e., the quality of solutions when the optimum is not found), while the

[1] An applet version of our CellNet simulator, implementing the EACO algorithm over the problem set, can be accessed at the URL: http://www.det.uvigo.es/~jrial/Sim/CellNet

two algorithms perform similarly in terms of efficacy (i.e., the percentage of runs in which the optimum is found).

Table 1 Computational Results

Problem	cGA			EACO			Wilcoxon test	
	Best	Avg. Result	Avg. Gen.	Best	Avg. Result	Avg. Gen.	Results	Gen.
ECC	**100.0%**	—	1.3103$E2$ ±2.3662$E1$	**100.0%**	—	**1.1256E2** ±**3.4095E1**	—	**3.858E − 8**
MAXCUT100	(52.0%)	4.6656$E-4$ ±8.6721$E-6$	**1.9892E2** ±**1.9213E2**	(**78.0**%)	**2.1297E − 4** ±**9.2710E−6**	3.5442$E2$ ±2.6644$E2$	**3.162E − 2**	0.1631
MAXCUT20_01	**100.0%**	—	7.8700 ±3.0966	**100.0%**	—	**5.0600** ±**1.8793**	—	**2.335E − 11**
MAXCUT20_09	**100.0%**	—	1.3530$E1$ ±4.5002	**100.0%**	—	**8.3700** ±**3.1096**	—	**2.2E − 16**
MMDP	(**2.0**%)	**2.5484E − 2** ±**0.0**	**3.6800E0** ±**3.2527E1**	2.5227$E-2$ (0.0%)	2.6026$E-2$ ±0.0	— —	0.6894	—
MTTP100	**100.0%**	—	**1.3733E2** ±**1.9499E1**	**100.0%**	—	1.7657$E2$ ±5.4288$E1$	—	**3.774E − 10**
MTTP200	(**88.0**%)	**6.0900E1** ±**0.0**	2.5469$E2$ ±3.2804$E1$	(46.0%)	2.7156$E2$ ±0.0	**1.5520E2** **8.5397E1**	0.4963	**2.194E − 3**
P-PEAKS	**100.0%**	—	5.7400$E1$ ±4.2545	**100.0%**	—	**4.2210E1** ±**5.0578**	—	**2.2E − 16**

7 Conclusions

In this chapter we have presented an introduction to the areas of Evolutionary Algorithms, Spatial Cellular Automata, Evolutionary Game Theory and Coalitions. We considered the integration of all these topics in a way to obtain a synergy in the development of Evolutionary Algorithms, with the advantages of dEAs and cEAs, but avoiding their particular drawbacks and removing their typically required parameters, like the neighborhood to use (cEAs) or the islands connectivity topology, the migration frequency, and the policies to exchange and discard individuals (in dEAs).

The main contribution of the chapter is the EACO algorithm, which uses spatial cellular approaches with neighborhoods, allowing the formation of coalitions among cells as a way to create islands of evolution in order to preserve diversity. Besides, we rely on Evolutionary Game Theory to consider every cell as a player of a game arranged in a two dimensional torus. Cells are able to evolve depending on their payoff with respect to their neighbors, and also the support provided by their coalition members. This approach allows the payoff of a given solution to be defined in terms of how much such solution has improved in the last generations. The idea speeds up the evolution of individuals grouped in high-quality coalitions that are quickly converging to promising solutions.

As the reader can see in the results section, EACO and canonical cGA perform similarly concerning the efficacy (understood as how many times the optimum is found). However, EACO is more accurate than the cGA for all the problems and also

more efficient (understood as faster to find the optimum) for all the complex problems selected, except for MAXCUT100. We have also provided evidences about the statistical significance of all these results.

This is still our first version of EACO, and there is much room for improvement, like performing an exhaustive parameter tuning, or studying different policies to compute the benefits for an individual after joining or leaving a coalition.

References

[AB05] T. Alpcan and T. Basar. A globally stable adaptive congestion control scheme for internet-style networks with delay. *IEEE/ACM Trans. Netw.*, 13:1261–1274, December 2005.

[AD05] E. Alba and B. Dorronsoro. The exploration/exploitation tradeoff in dynamic cellular evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 9(2):126–142, April 2005.

[AD08] E. Alba and B. Dorronsoro. *Cellular Genetic Algorithms*. Operations Research/Compuer Science Interfaces. Springer-Verlag Heidelberg, 2008.

[ADGT06] E. Alba, B. Dorronsoro, M. Giacobini, and M. Tomassini. *Handbook of Bioinspired Algorithms and Applications*, chapter Decentralized Cellular Evolutionary Algorithms, pages 103–120. CRC Press, 2006.

[Alb05] E. Alba. *Parallel Metaheuristics: A New Class of Algorithms*. Wiley, October 2005.

[AMD+06] E. Alba, J. Madera, B. Dorronsoro, A. Ochoa, and M. Soto. Theory and practice of cellular UMDA for discrete optimization. In T.P. Runarsson et al., editor, *Proc. of the International Conference on Parallel Problem Solving from Nature IX (PPSN-IX)*, volume 4193 of *LNCS*, pages 242–251, Reykjavik, Iceland, September 2006. Springer.

[AT00] E. Alba and J. M. Troya. Cellular evolutionary algorithms: Evaluating the influence of ratio. In M. Schoenauer et al., editor, *PPSN-6*, volume 1917 of *Lecture Notes in Computer Science*, pages 29–38. Springer-Verlag, 2000.

[AT02a] E. Alba and M. Tomassini. Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 6(5):443–462, October 2002.

[AT02b] E. Alba and J. M. Troya. Improving flexibility and efficiency by adding parallelism to genetic algorithms. *Soft Computing*, 12(2):91–114, 2002.

[BFM97] T. Bäck, D. B. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation*. Oxford University Press, 1997.

[Bin94] K. Binmore. *Game theory*. Mc Graw Hill, 1994.

[Blo95] F. Bloch. Endogenous structures of association in oligopolies. *RAND Journal of Economics*, 26(3):537–556, Autumn 1995.

[Blo96] F. Bloch. Sequential formation of coalitions in games with externalities and fixed payoff division. *Games and Economic Behavior*, 14(1):90–123, May 1996.

[BMJK10] Y. Bachrach, R. Meir, K. Jung, and P. Kohli. Coalitional structure generation in skill games. In *Association for the Advancemnt of Artificial Intelligence*, 2010.

[BR08] Y. Bachrach and J. S. Rosenschein. Coalitional skill games. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 2*, AAMAS '08, pages 1023–1030, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

[CEM+10] G. Chalkiadakis, E. Elkind, E. Markakis, M. Polukarov, and N. Jennings. Cooperative games with overlapping coalitions. *Journal of Artificial Intelligence Research (JAIR)*, 39:179–216, September 2010.

[CEMJ08] G. Chalkiadakis, E. Elkind, E. Markakis, and N. R. Jennings. Overlapping coalition formation. In *Proceedings of the 4th International Workshop on Internet and Network Economics*, WINE '08, pages 307–321, Berlin, Heidelberg, 2008. Springer-Verlag.

[CFW98] H. Chen, N. S. Flann, and D. W. Watson. Parallel genetic simulated annealing: A massively parallel SIMD algorithm. *IEEE Transactions on Parallel and Distributed Systems*, 9(2):126–136, 1998.

[CG10] G. Cantor and J. Gómez. Maintaining genetic diversity in fine-grained parallel genetic algorithms by combining cellular automata, cambrian explosions and massive extinctions. In *Proc. IEEE International Conference on Evolutionary Computation (CEC)*, pages 1–8, 2010.

[Cle06] M. Clerc. *Particle Swarm Optimization*. ISTE (International Scientific and Technical Encyclopedia), 2006.

[CP00] E. Cantú-Paz. *Efficient and Accurate Parallel Genetic Algorithms*, volume 1 of *Book Series on Genetic Algorithms and Evolutionary Computation*. Kluwer Academic Publishers, 2nd edition, 2000.

[CYCS10] J. Chen, X. Yan, H. Chen, and D. Sun. Resource constrained multirobot task allocation with a leader-follower coalition method. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5093 –5098, oct. 2010.

[DB11a] B. Dorronsoro and P. Bouvry. Adaptive neighborhoods for cellular genetic algorithms. In *Nature Inspired Distributed Computing (NIDISC) sessions of the International Parallel and Distributed Processing Symposium (IPDPS) 2011 Workshop*, pages 383–389, 2011.

[DB11b] B. Dorronsoro and P. Bouvry. Improving classical and decentralized differential evolution with new mutation operator and population topologies. *IEEE Transactions on Evolutionary Computation*, 15(1):67–98, 2011.

[DB11c] B. Dorronsoro and P. Bouvry. On the use of small-world population topologies for genetic algorithms. In *EVOLVE 2011, A bridge between Probability, Set Oriented Numerics and Evolutionary Computation*, pages e–proceedings, 2011.

[DDRJ06] V. D. Dang, R. K. Dash, A. Rogers, and N. R. Jennings. Overlapping coalition formation for efficient data fusion in multi-sensor networks. In *In 21st National Conference on AI (AAAI*, pages 635–640, 2006.

[DPS97] K. A. De Jong, M. A. Potter, and W. M. Spears. Using problem generators to explore the effects of epistasis. In T. Bäck, editor, *Proceedings of the 7th International Conference of Genetic Algorithms*, pages 338–345. Morgan Kaufman, 1997.

[EEB07] W. Elshamy, H. M. Emara, and A. Bahgat. Clubs-based particle swarm optimization. In *Proceedings of the IEEE Swarm Intelligence Symposium (SIS)*, pages 289–296, 2007.

[Eht97] H. Ehtamo. Dynamic noncooperative game theory : Tamer basar and geert jan olsder, 2nd ed. (academic press, san diego, ca, 1995) isbn 0-12-080221-x. *Journal of Economic Dynamics and Control*, 21(6):1113–1116, June 1997.

[Fis30] R. Fisher. *The genetical theory of natural selection*. Oxford: Clarendon Press, 1930.

[FRA05] P. Faratin and J. A. Rodríguez-Aguilar, editors. *Agent-Mediated Electronic Commerce VI, Theories for and Engineering of Distributed Mechanisms and Systems, AAMAS 2004 Workshop, AMEC 2004, New York, NY, USA, July 19, 2004, Revised Selected Papers*, volume 3435 of *Lecture Notes in Computer Science*. Springer, 2005.

[Gam61] W. A. Gamson. A theory of coalition formation. *American Sociological Review*, 26(3):pp. 373–382, 1961.

[GDH92] D. Goldberg, K. Deb, and J. Horn. Massively multimodality, deception, and genetic algorithms. In *Proc. Int. Conf. Parallel Prob. Solving from Nature II*, pages 37–46, 1992.

[GK03] F. W. Glover and G. A. Kochenberger, editors. *Handbook of Metaheuristics*. International Series in Operations Research Management Science. Kluwer, 2003.

[GK08] W. Gruszczyk and H. Kwasnicka. Coalition formation in multi-agent systems; an evolutionary approach. In *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on*, pages 125 –130, oct. 2008.

[GSS08] R. Glinton, P. Scerri, and K. Sycara. Agent-based sensor coalition formation. In *Information Fusion, 2008 11th International Conference on*, pages 1 –7, 30 2008-july 3 2008.

[GTT05] M. Giacobini, M. Tomassini, and A. Tettamanzi. Takeover time curves in random and small-world structured populations. In *Proc. of the Genetic and Evolutionary Computation COnference (GECCO)*, pages 1333–1340, Washington D.C. USA, June 25–29 2005. ACM Press.

[GV09] A. Godoy and F. J. Von Zuben. A complex neighborhood based particle swarm optimization. In *Proc. IEEE International Conference on Evolutionary Computation (CEC)*, pages 720–727, 2009.

[HL08] Z. Han and K. J. R. Liu. *Resource Allocation for Wireless Networks: Basics, Techniques, and Applications*. Cambridge University Press, New York, NY, USA, 2008.

[ISTN11] H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima. Implementation of cellular genetic algorithms with two neighborhood structures for single-objective and multi-objective optimization. *Soft Computing*, 15(9):1749–1767, 2011.

[JADM06] S. Janson, E. Alba, B. Dorronsoro, and M. Middendorf. Hierarchical cellular genetic algorithm. In J. Gottlieb and G.R. Raidl, editors, *Evolutionary Computation in Combinatorial Optimization (EvoCOP)*, volume 3906 of *Lecture Notes in Computer Science (LNCS)*, pages 111–122, Budapest, Hungary, April 2006. Springer.

[JM05] S. Janson and M. Middendorf. A hierarchical particle swarm optimizer and its adaptive variant. *IEEE Systems, Man and Cybernetics - Part B*, 35(6):1272–1282, 2005.

[Kal91] E. Kalai. Game theory: Analysis of conflict : By roger b. myerson, harvard univ. press, cambridge, ma, 1991. 568 pp. *Games and Economic Behavior*, 3(3):387–391, August 1991.

[KBH94] S. Khuri, T. Bäck, and J. Heitkötter. An evolutionary approach to combinatorial optimization problems. In *Proc. of the "ACM Press" Computer Science Conference*, pages 66–73, Phoenix, Arizona, 1994. ACM Press.

[Ken00] J. Kennedy. Stereotyping: improving particle swarm performance with cluster analysis. In *Proc. IEEE International Conference on Evolutionary Computation (CEC)*, volume 2, pages 1507–1512, 2000.

[KM02] J. Kennedy and R. Mendes. Population structure and particle swarm performance. In *Proc. IEEE International Conference on Evolutionary Computation (CEC)*, pages 1671–1676. IEEE Press, 2002.

[KM06] J. Kennedy and R. Mendes. Neighborhood topologies in fully informed and best-of-neighborhood particle swarms. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, 36(4):515–519, 2006.

[LC06] H.-Y. Liu and J.-F. Chen. Multi-robot cooperation coalition formation based on genetic algorithm. In *Machine Learning and Cybernetics, 2006 International Conference on*, pages 85 –88, aug. 2006.

[Li04] X. Li. Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization. In *Proc. of the Genetic and Evolutionary Computation COnference (GECCO)*, volume 3102 of *Lecture Notes in Computer Science (LNCS)*, pages 105–116. Springer, 2004.

[Li07] X. Li. Improving multi-agent coalition formation in complex environments, 2007.

[LS02] X. Li and S. Sutherland. A cellular genetic algorithm simulating predator-prey interactions. In *Proc. of the Third International Conference on Genetic Algorithms (ICGA)*, pages 416–421. Morgan Kaufmann, 2002.

[LXY+09] Z. Li, B. Xu, L. Yang, J. Chen, and K. Li. Quantum evolutionary algorithm for multi-robot coalition formation. In *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, GEC '09, pages 295–302, New York, NY, USA, 2009. ACM.

[MG06] M. Tomassini M. Giacobini, M. Preuss. Effects of scale-free and small-world topologies on binary coded self-adaptive CEA. In *Evolutionary Computation in Combinatorial Optimization (EvoCOP)*, volume 3906 of *Lecture Notes in Computer Science (LNCS)*. Springer, 2006.

[MKN04] R. Mendes, J. Kennedy, and J. Neves. The fully informed particle swarm: Simpler, maybe better. *IEEE Transactions on Evolutionary Computation*, 8(3):204–210, 2004.

[MS77] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, Amsterdam, 1977.

[MS89] B. Manderick and P. Spiessens. Fine-grained parallel genetic algorithm. In J.D. Schaffer, editor, *Third Int. Conf. on Genetic Algorithms ICGA-3*, pages 428–433. Morgan-Kaufmann, 1989.

[Mv47] O. Morgenstern and J. von Neumann. *The theory of games and economic behavior*. Princeton University Press, 1947.

[NAd06] N. Nedjah, E. Alba, and L. de Macedo Mourelle. *Parallel Evolutionary Computations*. Studies in Computational Intelligence. Springer, 2006.

[Nas50] J. Nash. Equilibrium points in n-person games. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 36, pages 48–49, 1950.

[Nas51] J. Nash. Non-cooperative games. *The Annals of Mathematics*, 54(2):pp. 286–295, 1951.

[Owe68] G. Owen. *Game theory*. Saunders, 1968.

[OZ06] S. Olariu and A. Y. Zomaya, editors. *Handbook of Bioinspired Algorithms and Applications*. CRC Press, 2006.

[Par98] L. E. Parker. Alliance: an architecture for fault tolerant multirobot cooperation. *Robotics and Automation, IEEE Transactions on*, 14(2):220 –240, apr 1998.

[PE06] J. L. Payne and M. J. Eppstein. Emergent mating topologies in spatially structured genetic algorithms. In *Proc. of the Genetic and Evolutionary Computation COnference (GECCO)*, pages 207–214, Seattle, Washington, USA, 2006. ACM Press.

[PE08] J. L. Payne and M. J. Eppstein. The influence of scaling and assortativity on takeover times in scale-free topologies. In *Proc. of the Genetic and Evolutionary Computation COnference (GECCO)*, pages 241–248, Atlanta, Georgia, USA, 2008. ACM Press.

[PT06] L. E. Parker and F. Tang. Building multirobot coalitions through automated task solution synthesis. *Proceedings of the IEEE*, 94(7):1289 –1305, july 2006.

[RV97] D. Ray and R. Vohra. Equilibrium binding agreements,. *Journal of Economic Theory*, 73(1):30–78, March 1997.

[SAK10] T. Shrot, Y. Aumann, and S. Kraus. On agent types in coalition formation problems. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1 - Volume 1*, AAMAS '10, pages 757–764, Richland, SC, 2010. International Foundation for Autonomous Agents and Multiagent Systems.

[SGW06] Y. Shen, B. Guo, and D. Wang. Optimal coalition structure based on particle swarm optimization algorithm in multi-agent system. In *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, volume 1, pages 2494 –2497, 0-0 2006.

[SK96] O. Shehory and S. Kraus. Formation of overlapping coalitions for precedence-ordered task-execution among autonomous agents. In *ICMAS-96*, pages 330–337, December 1996.

[SK98] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1):165–200, May 1998.

[Smi82] J. Maynard Smith. *Evolution and the theory of games*. Cambridge University Press, 1982.

[SSJ98] O. Shehory, K. Sycara, and S. Jha. Multi-agent coordination through coalition formation. In Munindar Singh, Anand Rao, and Michael Wooldridge, editors, *Intelligent Agents IV Agent Theories, Architectures, and Languages*, volume 1365 of *Lecture Notes in Computer Science*, pages 143–154. Springer Berlin,Heidelberg, 1998. 10.1007/BFb0026756.

[Sta] Standard Particle Swarm Optimization, Particle Swarm Central website.

[Ste93] J. Stender. *Parallel Genetic Algorithms: Theory and Applications*. IOS Press, Amsterdam, The Netherlands, 1993.

[Sti87] D. R. Stinson. *An Introduction to the Design and Analysis of Algorithms*. The Charles Babbage Research Center, Winnipeg, Manitoba, Canada, 1985 (second edition, 1987).

[Sug99] P. N. Suganthan. Particle swarm optimiser with neighborhood operator. In *Proc. IEEE International Conference on Evolutionary Computation (CEC)*, volume 3, pages 1958–1962, 1999.

[SVCC06] D. Simoncini, S. Verel, P. Collard, and M. Clergue. Anisotropic selection in cellular genetic algorithms. In *Proc. of the Genetic and Evolutionary Computation COnference (GECCO)*, pages 559–566, Seattle, Washington, USA, 2006. ACM Press.

[SZD+09] W. Saad, H. Zhu, M. Debbah, A. Hjorungnes, and T. Basar. Coalitional game theory for communication networks. *Signal Processing Magazine, IEEE*, 26(5):77 –97, september 2009.

[SZH+11] W. Saad, H. Zhu, A. Hjorungnes, D. Niyato, and E. Hossain. Coalition formation games for distributed cooperation among roadside units in vehicular networks. *Selected Areas in Communications, IEEE Journal on*, 29(1):48 –60, january 2011.

[Tal06] E.-G. Talbi. *Parallel Combinatorial Optimization*. John Wiley & Sons, 2006.

[Tom05] M. Tomassini. *Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time*. Natural Computing Series. Springer, 2005.

[Whi93] D. Whitley. Cellular genetic algorithms. In S. Forrest, editor, *Fifth Int. Conf. on Genetic Algorithms ICGA-5*, page 658, California, CA, USA, 1993. Morgan-Kaufmann.

[WRDM97] D. Whitley, S. Rana, J. Dzubera, and K. E. Mathias. Evaluating evolutionary algorithms. *Artificial Intelligence*, 85:245–276, 1997.

[WSP08] J. M. Whitacre, R. A. Sarker, and T. T. Pham. The self-organization of interaction networks for nature-inspired optimization. *IEEE Transactions on Evolutionary Computation*, 12(2):220–230, 2008.

[Yi92] S.-S. Yi. *Endogenous formation of coalitions in oligopoly*. Working paper series. Harvard University, 1992.

[Yi97] S.-S. Yi. Stable coalition structures with externalities. *Games and Economic Behavior*, 20(2):201 – 237, 1997.

[YL07] J. Yang and Z. Luo. Coalition formation mechanism in multi-agent systems based on genetic algorithms. *Applied Soft Computing*, 7(2):561 – 568, 2007.

[YML09] L. Yumind, L. Ming, and L. Ling. Cellular genetic algorithms with evolutional rule. In *International Workshop on Intelligent Systems and Applications (ISA)*, pages 1–4. IEEE, 2009.

[ZE11] Y. Zick and E. Elkind. Arbitrators in overlapping coalition formation games. In *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2001)*, 2011.