3b2v7 CAG : 999– Prod.Type:typ pp.1–11(col.fig.:fig.1) ED:susan/brr PAGN: mva SCAN: lakshmi/preethy

ARTICLE IN PRESS

PERGAMON

Computers & Graphics ▮ (▮▮▮▮) ▮▮▮–▮▮▮

COMPUTERS & GRAPHICS

www.elsevier.com/locate/cag

Emergence of multiagent spatial coordination strategies through artificial coevolution

André L.V. Coelho*, Daniel Weingaertner, Ricardo R. Gudwin, Ivan L.M. Ricarte

Department of Computer Engineering and Industrial Automation (DCA), School of Electrical and Computer Engineering (FEEC), State University of Campinas (Unicamp), Zip-code 13083-970, Campinas, Brazil

Abstract

This paper describes research investigating the evolution of coordination strategies in robot soccer teams. Each player (viewed as an agent) is provided with a common set of skills and is assigned to perform over a delimited area inside a soccer field. The idea is to optimize the whole team behavior by means of a spatial coadaptation process in which new players are selected in such a way to comply with the already existing ones. The main results show that, through coevolution, we progressively create teams whose members act on complementary areas of the playing field, being capable of prevailing over a standard opponent team with a fixed formation. © 2001 Published by Elsevier Science Ltd.

Keywords: Multiagent teams; Spatial coordination; Artificial coevolution; Emergence; Simulated robot soccer

1. Introduction

One of the most compelling and challenging tasks inside the distributed artificial intelligence (DAI) field is that of suitably devising coordination protocols customized to the problems in mind. Coordination can be summarized as a property of a system of agents performing some activities in a shared environment, concerning with how to effectively orchestrate the group (inter-) actions, in time and space, for achieving coherence [1,2]. It usually incurs complexity, as there are no predefined general recipes indicating how to establish, a *priori*, the rules of group behavior in view of all possible situations/scenarios. Moreover, there is a range of aspects, such as the homogeneity/heterogeneity of the agents' skills or the environmental characteristics (static versus dynamic), that should also be regarded when one chooses the coordination mechanisms to be employed.

Soccer seems to be a rich testbed domain for the study of multiagent coordination issues. In such context, a set of players must work together in order to put the ball into the opposing goal (augmenting its score) while at the same time defending its own. This is a typical domain where cooperation [3] should take part in—the individuals have the same global objective. One important issue for a soccer team to win a game is the strategy it uses, during a game period, to place each of its components in a given region of the field (such as backfield, leftwing, attack etc). That is, how to delimit the zone at which a certain player can perform better in order to improve the capabilities of the whole group. This sort of coordination effort is referred to here as *spatial strategy*. Our primary aim is to evolve this process of team formation through a coevolutionary approach [4–8], so that a spatial strategy can emerge without human interference. In this regard, the idea is to qualitatively analyze how much the arrangement of a team may influence its overall performance. Furthermore, a secondary purpose underlying this initiative is to reveal the potentials of applying artificial life (coevolutionary-based) techniques towards complex behavior modeling in societies of artificial agents. The problem we are trying to tackle (emergence of team positioning strategies) is straightly related to the task of automatic synthesis of multiagent behavior since the organization policies adopted by a team directly constrain the possible dynamic comportment it might assume while

*Corresponding author. Tel.: +55-19-3788-3771; fax: +55-19-3289-1395.

E-mail address: acoelho@ieee.org (A.L.V. Coelho).

0097-8493/01/$ - see front matter © 2001 Published by Elsevier Science Ltd.
PII: S0097-8493(01)00155-8

facing (non-predicted) future situations presented to it by the environment.

In the sequence, we introduce the robot soccer problem and the artificial coevolutionary approach applied to multiagent spatial coordination, present our framework and solution, show the results from our experiments, and finally identify future plan of work.

2. Background and related work

Many approaches to tackle coordination problems are currently available in the literature [1,3,7]. Most of them center around the specification and implementation of high-level protocols (many times based on human social interactions) containing the actions to be taken at particular cases, either by a single agent or by the whole group. Well-known examples following such idea are Laird and others' knowledge-based coordination model and Jenning's formalism of *commitments* and *conventions*. In the latter, for instance, rules to undertake a specific course of action are conceived before the actual deployment of the team in the environment (joint commitments). In order to monitor whether these rules have been fulfilled or whether they are still valid in changing circumstances, there are also additional emergency instructions towards the dynamic adjustment of the group activities through time (social conventions).

Such kind of endeavor aiming at the conception of an *explicit* scheme of coordination seems to be only suited for a constrained class of problems, showing both performance and scalability bottlenecks when applied at more complex domains. Alternative mechanisms have been conceived in order to surpass those deficiencies, such as distributed planning and real-time (re-) planning [2,7].

A new-fashioned line of research (followed in this work) involves the pervasive use of evolutionary techniques as a means to improve both individual as well as group abilities in a concurrent manner. This methodology stipulates for group organization in a seamless and *implicit* manner; that is, there is no need for explicit pre-codified protocols. The coordination activities can now be viewed as an optimization problem whose solution(s) is (are) searched via a computational procedure that mimics the steps of the natural evolutionary process. Applying such strategy for the automatic configuration of robot soccer spatial coordination strategies constitutes the primary contribution of this work.

The CMUnited [9,10], developed at Carnegie Mellon University, has been one of the most successful physical robot soccer teams in the contests of the RoboCup world championship [11]. It encompasses a layered learning technique to first train the players basic skills (dribbling, shooting) for then building more complex capabilities (passing, positioning) upon the basic ones. The formation of the team can change in the course of the game, but the set of possible formations is determined empirically and one of them is chosen in accordance with the current situation of the match [9].

By other means, Balch [12] has used his robot soccer simulator [13] to investigate behavioral specialization in learning robot teams. In his work, all agents have a common set of skills from which they build a task achieving strategy using a Q-learning (reinforcement learning) algorithm. After playing for some time against a fixed strategy control team, the learning agents specialize into complementary roles because their reward depends on the score of the game, not on individual actions.

Some papers already report on work concerning the application of artificial learning and evolution to some soccer-related problems. For instance, the approach proposed by Agah et al. deals with the production of evolutionary cooperative strategies by means of a devised cognitive architecture based on Tropism [14], whereas Matsubara and colleagues employed a neural network approach towards players' on-line learning in how to take correct decisions (pass a ball to a peer or shoot towards the opposite goal) according to some pre-established field positioning situations [15]. Andou has already assessed the employment of reinforcement learning schemes to update players positions on the field based on where the ball has previously been located [16]. By other means, Luke et al. set out to create a completely learned team of agents using genetic programming [17]. Their approach already employed an artificial coevolutionary methodology which was conceived primarily towards behavior-based team coordination, not coping with spatial organization problems.

The approach underlying this work differs from others in several aspects. First, we do not have any predefined formation for the players, but want that the formation emerges by means of an evolutionary scheme. We do not use reinforcement learning, but also apply the result of the game as a reward function for the employed evolutionary technique (in order to calculate the level of group adaptability), so that the performance of a single player depends on the performance of the whole team. Finally, as a more adequate strategy for the soccer players progressive spatial co-adaptation, a novel memory-based, cooperative coevolutionary architecture [6,18] towards the dynamic popup (emergence) of instances of evolutionary algorithms has been designed.

3. Simulated robot soccer

Research on robot soccer has received an increasing attention through the last years. Soccer is an attractive domain for multiagent study as the success of a team depends very much on some form of coordination [19, 14]. It is also very appealing because the game is played in a dynamic, real-time, competitive and cooperative environment, from which the agents (players) percept only a small part (limited visual perspective), what typically incurs the need of world modeling, distributed learning and planning. The control of the agents is decentralized and the changes in the environment neither are fully predictable nor happen in discrete time steps. For our purposes, the game is simplified and is ruled according to the following aspects:

- Teams are composed of five players.
- The sidelines are walls—the ball bounces back instead of going out-of-bounds.
- After a scoring event, the ball is immediately placed back in the center of the field.
- Each player has accurate information about the position of the other peers and adversaries as well as of the ball.

According to Huhns and Stephens [2], there are two commonly used methods for apportioning tasks among cooperative agents. One is the *functional* distribution, in which cooperation comes as the union of the individual capacities of the players (one player is a good shooter, other is a passer, and so on). The second method, called *spatial* distribution, is a form of cooperation where the agents divide the search or performing space (in soccer, this is the field) into well-defined areas, in such a way to quicken the team performance through the sharing of goal responsibilities. In this work, the latter method was chosen for experimenting with the coevolutionary coordination of robot soccer agents.

The Java-based soccer simulator employed in our experiments [13] (Fig. 1) implements each player on a separate OS *thread* and runs the simulation in discrete steps. At each step, the robots process their sensor data before ascertaining their appropriate effector commands.

4. Coordination via coevolution

Evolutionary Computation (EC) has come out as the branch of computational intelligence research employing metaphors from natural evolutionary phenomena as a means to achieve efficient problem solving (search) techniques [3]. Its applicability has constantly increased in recent years, and many are the engineering fields that have some of their processes improved through the appliance of evolutionary-based approaches. The majority of the implementations of such approaches descend from four independent lines of research, namely *genetic algorithms* (GA), *genetic programming* (GP), *evolutionary programming* (EP), and *evolution strategies* (ES).

In the conventional GA model, a population of strings (*chromosomes*) codifying the possible solutions for the problem in hand passes through a cyclic (generation-based) process in which new candidates are constantly created and evaluated in accordance with some measure of environment adequacy known as *fitness*. Ancestors are charged by computational operators very much resembling natural evolutionary phenomena, such as reproduction, selection and mutation, being progressively replaced by more adapted newcomers. The population fitness tends to converge in the course of the process and (sub-) optimal solutions are obtained at final stages.

Some problems with this model have already been reported. First, it is very prone to the "local minima/maxima problem", as it depends very much on the configuration (search space distribution) of the initial population. Likewise, some fast convergence problems may occur if the population size is not properly set. This model has, as well, scalability problems, like how to incorporate all the knowledge about the problem and to discriminate and prioritize (possibly several) distinct factors in a unique evaluation function. In the same manner, the representation of some heterogeneity issues behind the problem may be constrained, as the phenotypic interpretation of parameters is the same for all the individuals (single species). Moreover, the model is also not adequate for the evolution of sets of interacting rules with variable sizes whose individual fitness are determined by their interactions via a simulated micro-economy. (*Classifier systems* and other related works, such as SAMUEL [20], have been devised to surpass such drawback.) Finally, it is not very suited for the representation/generation of complex structures such as those composed by many sub-entities (as it is the case of multi-agent coordination systems).

In order to tackle such deficiencies, distributed genetic algorithms [4] have been introduced. The idea is to bring about a set of genetic algorithm instances working together in a parallel/distributed environment in order to find out the best solution for a common problem. Each GA runs independently from the others. Other, more recently investigated, concepts are those of *niches* and *speciation* [21]. The first brings the idea of dynamically mounting small groups of correlated individuals that act upon a close region of a large search space. Individual niches compete for the allocation of trials. The second refers to new forms of "on-the-fly" species generation.

Extending the boundaries, there is now such a trend to apply *artificial coevolution* [5] as a more suitable technique towards complexity overcoming. Artificial coevolution has its roots in its biological counterpart. Simply put, coevolution means "any reciprocal

Fig. 1. Game start position on the Java-based robot soccer simulator.

evolutionary change in interacting species [22].” Although vague, such definition is powerful enough to comprehend any natural process in which two or more species, typically coexisting in a same environment, have their evolutionary trajectories somehow affected by the stable ecological *interactions* and *interrelationships* their members jointly promote and take part in. In the artificial realm, two or more populations of different species are optimized together, one influencing the other by some means.

Artificial coevolution seems very suited for simulating cooperation and/or competition behavior among multiagent entities. Following such premise, Puppala et al. have devised a share-memory based approach [18] to evolve cooperating individuals of two different species–painters and whitewashers—for solving a room painting problem (see Fig. 2). In this case, each of the agents has unique skills necessary to complete a job; that is, they are interdependent and the group behavior depends on the joint behavior of both components. The idea is to find pairs whose members are best adapted to each other, so the overall performance can be improved. This should be regarded as a kind of functional decomposability of a huge, high-level problem. In their scheme, an individual from the first population (codified by a rule of behavior) is assessed by mating it with other individuals of the second population (the reverse is also true). Its highest performance evaluation on all pairs that it participates is assigned as its fitness. Instead of randomly picking the individuals, the authors conceived a buffer for grabbing and remembering the most

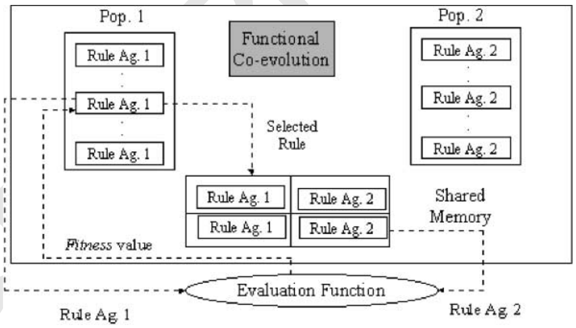


Fig. 2. Shared-memory based approach for multiagent coordination.

successful pairs achieved so far: In this case, the mating is done by selecting the N best partners from the other population which prevailed at the last generation. The memory is updated if a fitness value of a new assembled pair is higher than any of those currently stored, promoting the replacement of the stored pair with the minimum value (tail of the list) by the new one.

5. Soccer team spatial coordination

In this section, the features underlying our proposal for soccer team spatial coordination are gradually presented.

5.1. Players and regions

Since our main interest was on the formation of the team and on its influence on the result of the game, all players, from both opposing teams, were modeled with the same basic skills and control algorithm (see Appendix A). Each player was allowed to perform only inside a particular actuation area, which was characterized by three mark points: *defense* (D), *middle* (M), and *attack* (A) (Fig. 3a). In order to avoid a player choosing a too small area to play, the field was separated into 18 squares, as shown in Fig. 3b. For classification purposes, we considered nine delimited regions covering the whole field (Fig. 3b). Each player of an experimental team was bestowed with a label indicating the region to which it belongs to, in accordance with the minimum Euclidean distance between the center of its actuation area and the center of all nine regions. Table 1 shows the coordinates of these regions.

5.2. Teams

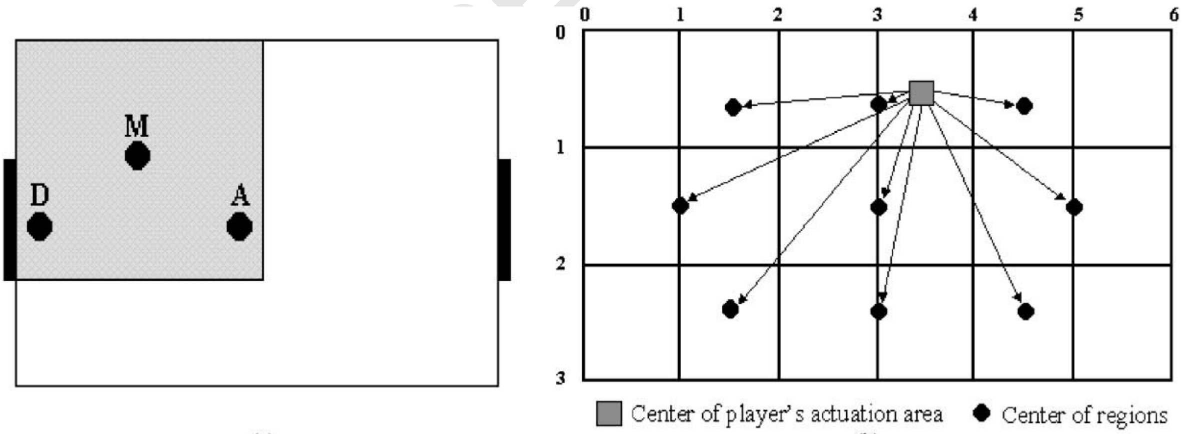
The investigation was conducted by engaging experimental teams against a fixed opponent *control* team that uses a 1:3:1 formation (Fig. 7a). The purpose was to evolve (or create) new teams that were able to defeat the control team in soccer contests, owing only to a different spatial distribution of the players. The motivation is to certify whether the task of choosing one from a range of different formation strategies has direct influence on the relative performance of an evolved team versus the control team.

Table 1
Coordinates of the region points used to classify the players according to their acting area

Region	Coordinates	Region	Coordinates
r_1	(1.5, 0.75)	r_6	(5.0, 1.5)
r_2	(3.0, 0.75)	r_7	(1.5, 2.25)
r_3	(4.5, 0.75)	r_8	(3.0, 2.25)
r_4	(1.0, 1.5)	r_9	(4.5, 2.25)
r_5	(3.0, 1.5)		

5.3. Architecture

Based on the concepts of niches, speciation and cooperative coevolution, we have designed a new architecture for multiagent spatial coordination. The most innovative idea is that of *progressively* assembling the evolving teams (niches) by allocating for each of the five possible players (positions) a promising acting region to be represented by a dynamically created species. In the beginning, all players are randomly selected from the same GA instance (named GA-0)—we could employ any other evolutionary algorithm as well. Then, in the course of generations, some players, competing against all, will prevail, spawning new offspring very akin to them. Those most adapted players and their offspring certainly will perform over similar field regions, characterizing a promising searching area for another GA instance, giving birth to a speciation process. (This is why we partitioned the field into nine logical regions.) As times passes by, new GAs are popped up and the GA-0 is restarted with another initial population if the number of players of new species (may be more than one at the same time) fires up a certain threshold. Each new spawned GA is assigned to a place (player) in all future teams formation. That is, one of its members will be selected to take place in each experimental team thereafter. The coadaptation process



(a) (b)

Fig. 3. (a) Actuation area of a player with its defense (D), middle (M) and attack (A) position. (b) Discrete field coordinates (6 × 3 rectangles). Each player is associated to the closest region, from nine defined on the field, according to the distance measured between the center of the actuation area to the center of the region.

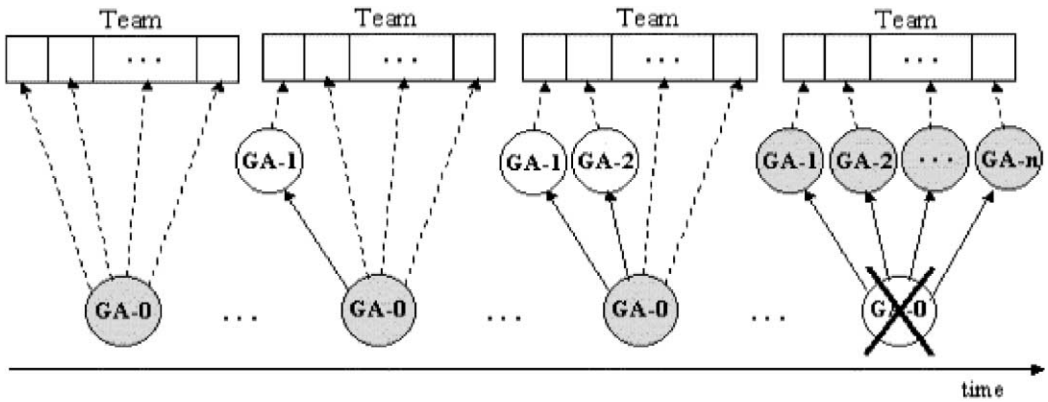


Fig. 4. Creation of the teams: on the beginning, all players are taken from the GA-0. As soon as a region of the field has enough players to form a new population, this population is copied into a new GA-1. After that, each team will have one player coming from that GA. The GA-0 is restarted and the procedure is repeated until we have one GA for each position of the team. Then the GA-0 is destroyed and the others are activated to co-evolve.

is granted as the new GAs (species) are formed by in accordance with the already existing ones. At a final step, the spawned GAs have their populations evolved synchronously during some other few cycles for fine-tuning purposes. Fig. 4 shows the details of such coevolutionary framework.

Some other considerations are worth to be mentioned:

- This approach also guides the spring of new individuals in the GA-0 population that have complementary roles from those of the already selected species, promoting for an automatic means of problem decomposition.
- The new created GA instances will not allow its individuals to evolve until each of the five field positions has an associated species. This avoids the possibility of badly influencing the formation of the new GAs with corrupted (vitiated) initial populations of the GA-0. If a GA-0 instance does not produce any novel species for a delimited number of generations it must be replaced by other instance in such a way to accelerate the search process.
- In order to give to its individuals a better chance to survive and to be selected for a new team, each new created GA instance can not have more than the half of the number of individuals in the GA-0. Only the most adapted are picked.
- The GA-0 population decreases as the number of species increases. This is because there will be less slots in a team the GA-0 individuals will struggle for.
- The architecture is also memory-based. However, in order to avoid combinatorial explosion problems (as we should have populations in the range of hundreds of elements), we did not adopt Puppala's individual fitness evaluation based on cross-mating. Instead, we opted to apply random teams assemblage (limiting the number of possible teams) for those positions that do not already have an associated species.

For implementation purposes, six classes codified in Java mainly compose this architecture; they are *Player*, *Team*, *Team_pool*, *Memory*, *Simulator*, and GA, whose interrelationship model is presented in Fig. 5. Fig. 6 brings a high-level execution flowchart for our coevolutionary approach.

5.4. Genetic algorithm

The creation and evolution of the players are controlled by a genetic algorithm that uses elitist selection, one-point crossover and mutation [23] to generate new players from a previous population. The initial population is randomly generated, in a uniform distribution. The fitness function used to reward the players is not based on their single performance, but on the score of the team where the player actuates. Eq. (1) brings such evaluation function. "S_{team}" and "$S_{control}$" are the scores of an evolved team and from the control group, respectively:

$$f(player) = \begin{cases} 3 + (S_{team} - S_{control})/10, & \text{for } S_{team} > S_{control}, \\ -1 + (S_{team} - S_{control})/10, & \text{for } S_{team} < S_{control}, \\ 1 + (S_{team}/10), & \text{for } S_{team} = S_{control} \end{cases} \quad (1)$$

It may happen that a player is chosen to play in more than one team. In this case it will keep the highest fitness of all teams it participated. In the case that a player does not play any match, it will receive the reward as being the average reward of all players on its region.

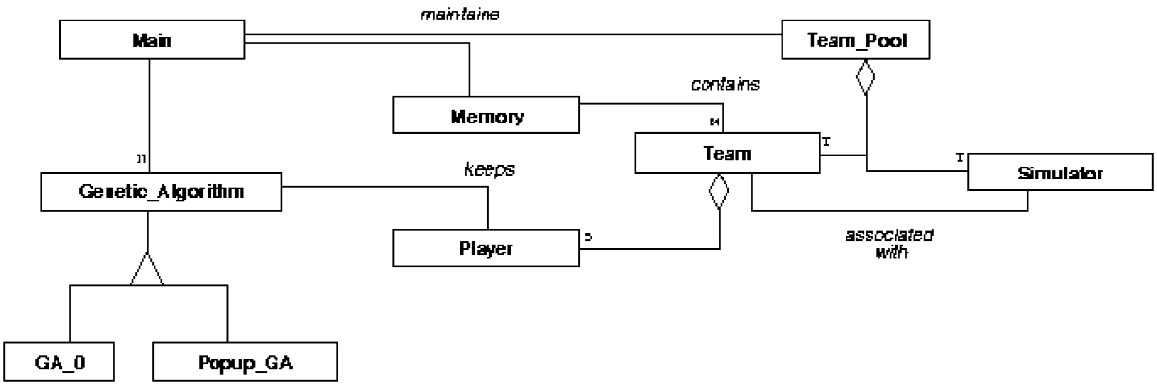


Fig. 5. Object model showing the relationships among the main Java classes. N, T, and M are parameters indicating the maximum number of GA instances (5), the number of experimental teams per generation (25) and the number of buffered teams (10).

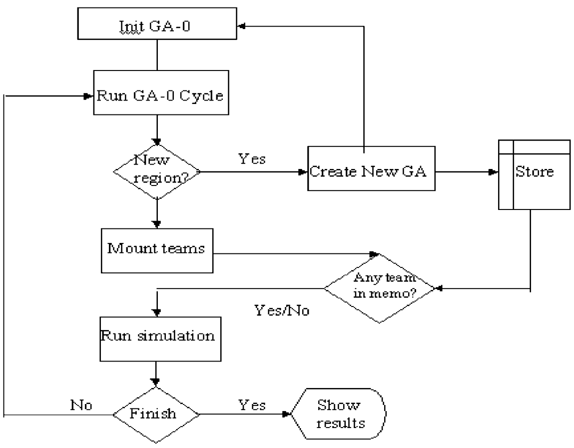


Fig. 6. Execution flow in a typical run of the proposed coevolutionary approach.

Table 2
Order in which the new GAs were created and the regions to which their initial population belonged

	GA-1	GA-2	GA-3	GA-4	GA-5
r_1					
r_2	•		•		
r_3					
r_4					
r_5					•
r_6					
r_7				•	
r_8		•			
r_9					

6. Simulation results

Experiments were conducted by running the algorithm described in Appendix B and employing Balch's Java-based soccer simulator for 50 players and 25 teams. The most important parameter settings can be found in Table 3. For each GA-0 generation, all created teams played an 8 min long match against the control team. Each simulation of a soccer match was performed on a separate Java *thread* and those simulations were the most time-consuming tasks, taking about 1 h[1] to simulate the 25 matches of an evolutionary cycle. The evaluation of the results was focused on three criteria: spring of new formations, cooperative coevolution of the agents and convergence of the genetic algorithm.

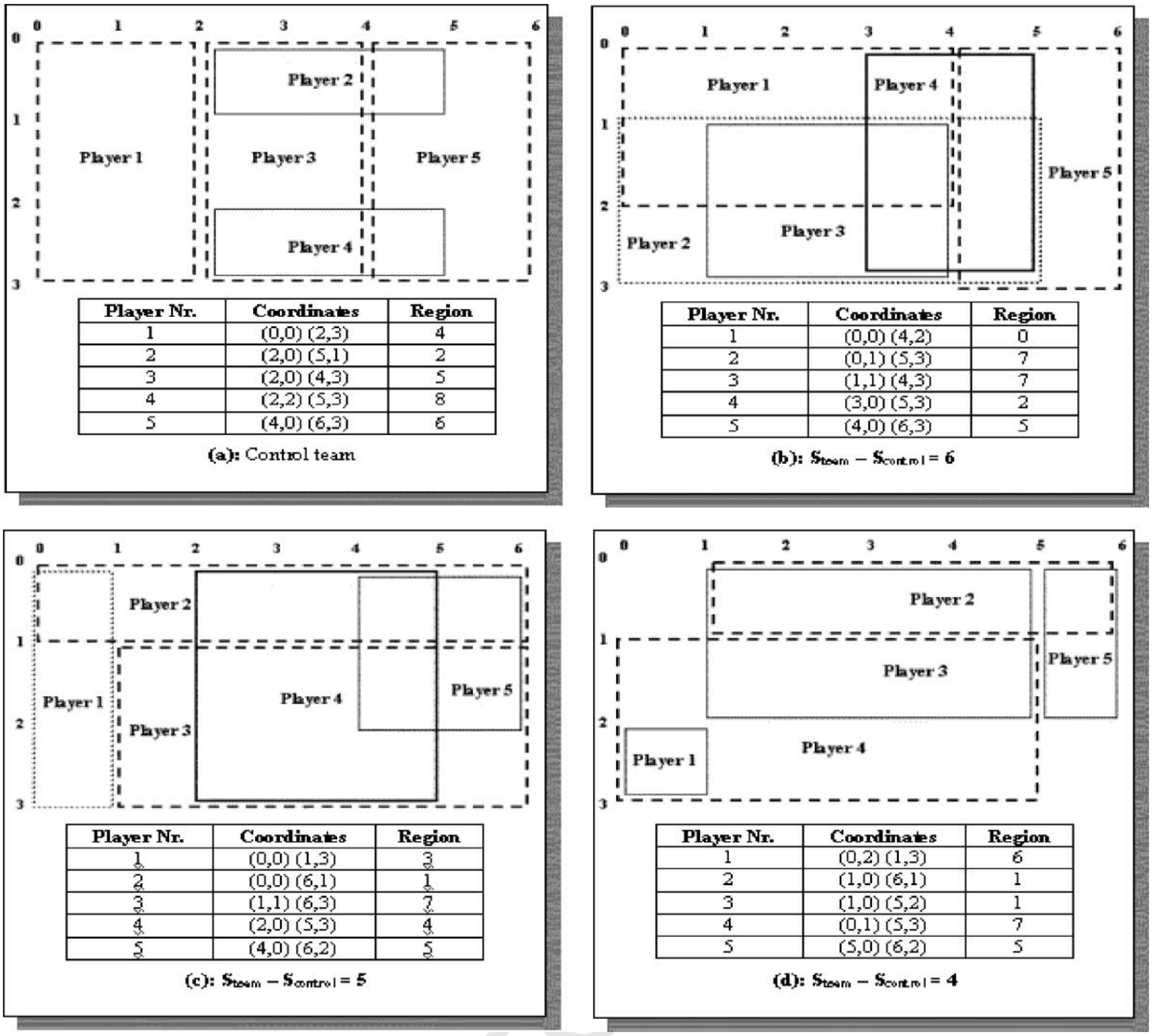
[1] The experiments were performed using an Enterprise 450 Machine with two 300-MHz Ultra Sparc-2 processors, 512MB RAM, running SunOS 5.7, and using JDK 1.2 for code implementation.

6.1. Formations

Since the only difference between the emerged teams and the control team is the formation, we could verify that it played a prominent role in robotic soccer. The 10-best evolved teams achieved for the best experimental running were able to win the control team with an average of four goals of difference. It is worth to remind that our approach created formations automatically, without human soccer expertise, and could be used to train different team formations according to its adversary. The formations of the control team and of three winning evolved teams can be seen in Fig. 7.

6.2. Coevolution

Looking at Figs. 7b–7d, we can notice that the players actuation areas that emerged were complementary. Since each of the players came from a different GA instance, and since those GAs were created in different time steps, we can conclude that the constructive



(a): Control team

Player Nr.	Coordinates	Region
1	(0,0) (2,3)	4
2	(2,0) (5,1)	2
3	(2,0) (4,3)	5
4	(2,2) (5,3)	8
5	(4,0) (6,3)	6

(b): $S_{team} - S_{control} = 6$

Player Nr.	Coordinates	Region
1	(0,0) (4,2)	0
2	(0,1) (5,3)	7
3	(1,1) (4,3)	7
4	(3,0) (5,3)	2
5	(4,0) (6,3)	5

(c): $S_{team} - S_{control} = 5$

Player Nr.	Coordinates	Region
1	(0,0) (1,3)	3
2	(0,0) (6,1)	1
3	(1,1) (6,3)	7
4	(2,0) (5,3)	4
5	(4,0) (6,2)	5

(d): $S_{team} - S_{control} = 4$

Player Nr.	Coordinates	Region
1	(0,2) (1,3)	6
2	(1,0) (6,1)	1
3	(1,0) (5,2)	1
4	(0,1) (5,3)	7
5	(5,0) (6,2)	5

Fig. 7. Team formations. (a) Formation of the control team. (b), (c) and (d) are evolved formations that were able to defeat. the control team with a goal difference of 6, 5 and 4, respectively. We verify that the players occupy complementary and overlapping parts of the field, in a way that almost all the field is covered by the team.

coadaptation approach was pivotal for the arrangement of teams whose players actuate cooperatively for the field covering. To assist the reader in this assessment, Table 2 brings the order in which the new GA instances were progressively created for the best benchmark running performed so far, displaying the respective regions to which their initial populations (emerged niches of the GA-0) belong.

6.3. Convergence of the GAs

Using the 50-players/25-teams configuration, we observed, through some experiments, that the GA-0 was very susceptible to quick convergence. Almost always, the new GA instances tended to emerge within the minimum necessary number of generations (Table 3), and their initial populations were formed by only one or two different classes of players (breeds or subspecies). This was typically a non-diversity problem. To avoid this misbehavior, we increased the number of players and teams to 500/250 and observed that the population was indeed more diverse and, thus, did not converge so fast. However, such decision incurred, as a negative side effect, unaffordable simulation cost increases in such a manner to hamper the performance assessment process. This "diversity X computational cost" tradeoff is typical in any evolutionary-based technique, being generally dealt with via the employment of empirical fine-tuning calibrations of the configuration parameters.

Table 3
System configuration parameters

Parameter	Meaning	Value
POP_LENGTH	Number of players in the initial GA-0	50
MEM_SIZE	Number of best teams that are kept on memory	10
TEAMS	Number of teams to be formed	25
PLAYERS	Number of players per team	5
MAX_NEW_GA_POP_LENGTH	Maximum number of players that a new GA may have	TEAMS/2
TIME_TO_LIVE	Maximum number of generations	120
THRESHOLD	Percentage of players a region must have to form a new GA	40%
MIN_TIME_TO_POPUP	Minimum number of generations for a new GA to be formed	5
MAX_TIME_TO_POPUP	Maximum number of generations for a new GA to be formed	20
GENE_CROSSOVER_CHANCE	Chance of crossover occurrence	25%
GENE_MUTATION_CHANCE	Chance of mutation occurrence	1%
ELITIST	Number of best players copied to next generation (elitist selection)	20%

7. Conclusion and future work

Applying coevolutionary techniques in complex problem domains has been proved to be a promising alternative strategy for achieving both performance and quality improvements. Recent research works have addressed the employment of such approach in a variety of problems, including single/multicriteria function optimization [24,4,6] and multiagent scenarios [20,18,17]. In this work, a new coevolutionary-based architecture for robot soccer teams spatial coordination was depicted and evaluated, confirming: (i) the feasibility of obtaining an automatic method for the generation of implicit coordination rules; (ii) that the spatial distribution of homogeneous players across the field may direct influence the behavior and performance of the whole team; (iii) that the approach encourages the formation of stable niches of cooperative sub-components (players) whose acting regions tend to be complementary on the field covering; and (iv) that artificial life techniques (particularly coevolutionary-based ones) are a step towards the automatic synthesis of complex behavior and control rules in societies of various autonomous entities.

Some problems were detected during the simulations execution of our approach, demanding for new design or parameter setting corrections. For instance, the fast convergence in the fitness of the new created GAs' populations surely had a great bad effect on some attained results. Increasing the size of all GAs, however, would complicate the players evaluation process and augment the computational time required at each running cycle. As a consequence, there is an intrinsic hard trade-off for configuring parameters of this sort, as well as for those relating to the GA operators (higher mutation rates shall also provide a means to overcome such fast convergence).

Another problem that deserves attention relates to the possibility of loosing the constituents of the best existing teams (those already in the memory). Although applying an elitist selection process, it is not assured that these players will be maintained in their respective GAs' populations. (What is maintained is just a copy of the whole team in the pool.) Therefore, we observed that some possibly successful teams that emerged during the generation-based process had their evolutionary "customization" hindered by the extinction of one or more of its components in a prior generation.

The formations that emerged during our tests are suitable only to play against the control team used in the experiments, performing inefficiently against teams with different formations. This is a big limitation if the team is intended to participate in a competition like the RoboCup. Therefore, an interesting extension to this work would be to train many different formations against distinct configurations of control teams, store the best formations in a run-time memory and then, during the contests, dynamically adapt the team spatial distribution in conformance with the current opponent's strategy.

Acknowledgements

CNPq and Capes have sponsored the first and second authors, respectively, through Ph.D. and MSc. scholarships.

Appendix A. Algorithm that controls the actions of each player

Compute: (A)attack, (M)middle, (D)defense
if(Player outside it's area)
Move to area;

```
    else if( Ball inside area )
    if( Closest player to ball )
            Move to ball;
              Else
          Move close to ball;
    Else//ball outside the area
  if( Ball on defense side of field )
            Move to (D);
    else if( Ball on attack field )
            Move to (A);
              Else
            Move to (M);
  if(Can kick and Is worth kicking )
            Kick the ball;
```

Appendix B. Algorithm for the creation of teams through coevolution

```
1. Create GA-0's initial population;
2. Classify players in regions;
3. Create N teams;
    4. Simulation:
        for( each team )
          Play against static team;
          Compute fitness;
            if( S_us > S_them )
              fitness ⇐ 3 + (S_team − S_control)/10;
            else if( S_team < S_control )
              fitness ⇐ −1 + (S_team − S_control)/10;
            else
              fitness ⇐ 1 + S_team/10;
            (where S_team and S_control is the score of
             each team at the end of the game)
          Set fitness of the players;
          Save M best teams;
        for( each region )
          Compute fitness of the regions (Fr);
    5. for ( G generations )
        for( GA-0 xor GA-i )
          Elitist selection;
        Crossover;
          Mutation;
        Classify players in regions;
        Create N teams;
          Take one player from each GA-i;
          Complete team with players from GA-0;
        Simulate the games;
        if( |GA| < |players per team| and G > min )
          if( |region| > Threshold )
            Create new GA-i;
            Initialize it with population of region;
            GA-0 ⇐ New initial population;
          else if( timeout )
            New GA-i ⇐ Biggest region;
            GA0  ⇐ New initial population;
    6. Print best teams
```

References

[1] Jennings N.Coordination techniques for distributed artificial intelligence, Foundations of DAI, 1996, New York John Wiley. p. 187-210.

[2] Weiss G. (editor). Multiagent systems: a modern approach to distributed artificial intelligence, Cambridge, MA: The MIT Press, 1999.

[3] Doran J, Franklin S, Jennings N, Norman T. On cooperation in multi-agent systems. The Knowledge Engineering Review 1997;12(3):309–14.

[4] Husbands P. Distributed coevolutionary genetic algorithms for multi-criteria, multi-constraint optimisation, Lecture Noles in Computer Science, vol. 865 and Berline Spring 1994. p. 150–65.

[5] Paredis J. Coevolutionary computation. Artificial Life Journal 1996;2(4).

[6] Potter M, De Jong K. Cooperative coevolution: an architecture for evolving coadapted subcomponents. Evolutionary Computation. 2000;8(1):1–29.

[7] Laird JE, Jones RM, Nielsen PE. Knowledge-based multiagent coordination. Presence—Teleoperators and Virtual Environments 1998;7(6):547–63.

[8] Stone P, Veloso M. Task decomposition, dynamic role assignment and low-bandwidth communication for real-time strategic teamwork. Artificial Intelligence 1999;110(2):241–73.

[9] Stone P, Riley P, Veloso M. The CMUnited-99 champion simulator team. In: Veloso M, Pagello E, Kitano H, editors. RoboCup-99: Robot Soccer World Cup III. Berlin: Springer, Berlin, 2000.

[10] Stone P, Veloso M, Riley P. The CMUnited-98 champion simulator team. In Asada M, Kitano H, editors. RoboCup-98: Robot Soccer World Cup II. Berlin: Springer, 1999.

[11] RoboCup Official Home Page: http://www.robocup.org.

[12] Balch T. Learning roles: behavioral diversity in robot teams. College of Computing Technical Report GIT-CC-97-12, Georgia Institute of Technology, Atlanta, Georgia, March 1997.

[13] TeamBots home page: http://www.teambots.org.

[14] Agah A, et al. Emergent cooperative strategies for robot team sports. Intelligent Automation and Soft Computing 2000;6(1):45–56.

[15] Matsubara H, Noda I, Hiraki K. Learning of cooperative actions in multi-agent systems: a case study of pass play in soccer. In: Adaptation, coevolution and learning in multiagent systems: 1996 AAAI Spring Symposium. Menlo Park, CA: AAAI Press, March 1996. p. 63–7.

[16] Andou T. Refinement of soccer agents' positions using reinforcement learning. In H. Kitano, editor, RoboCup-97; Robot Soccer World Cup I. Berlin: Springer Verlag, 1998. p. 373–88.

[17] Luke S, Hohn C, Farris J, Jackson G, Hendler J. Co-evolving soccer softbot team coordination with genetic

programming. Proceedings of the First International Workshop on RoboCup, Nagoya, Japan, August 1997. p. 115–8.

[18] Puppala N, Sen S, Gordin M. Shared memory based cooperative coevolution. Proceedings of the International Conference on Evolutionary Computation (ICEC'98). New York IEEE Press, 1998. p. 570–4.

[19] Sahota M. Real-time intelligent behavior in dynamic environments: soccer-playing robots. Master Thesis, The University of British Columbia, Canada, 1993.

[20] Potter M, De Jong K, Grefenstette J. A coevolutionary approach to learning sequential decision rules, Proceedings of the 6th International Conference on Genetic Algorithms (ICGA95), Los Altos, CAS: Morgan Kaufmann, July 1995. p. 366–72.

[21] Mahfoud SW. Niching methods for genetic algorithms. Illigal Report No. 95001, University of Illinois at Urbana-Champaign, Urbana, IL, May 1995.

[22] Thompson JN. The coevolutionary process. Chicago: The University of Chicago Press, 1994.

[23] Bäck T, Hammel U, Schwefel H-P. Evolutionary computation: comments on the history and current state, IEEE Transactions on Evolutionary Computation 1997;1(1):3–17.

[24] Husbands P, Mill F. Simulated co-evolution as the mechanism for emergent planning and scheduling. Proceedings of The Fourth International Conference on Genetic Algorithms, Morgan Kaufmann. Los Atlos, CAS 1991. p. 264–70.