Hierarchical Cooperative CoEvolution: Presentation and Assessment Study

MICHAIL MANIADAKIS and PANOS TRAHANIAS

Institute of Computer Science Foundation for Research and Technology – Hellas (FORTH)
P.O.Box 1385, Heraklion, 711 10 Crete, Greece
and
Department of Computer Science, University of Crete
P.O.Box 1470, Heraklion, 714 09 Crete, Greece
e-mail:{mmaniada,trahania}@ics.forth.gr

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

The current paper addresses the design of complex distributed systems consisting of many components by using Hierarchical Cooperative CoEvolution (HCCE), an optimization mechanism that also follows a distributed organization. The proposed coevolutionary scheme is capable of optimizing complex distributed systems, taking also into account the specialized roles of substructures. Here, we present HCCE and we compare it with ordinary Unimodal evolution and Enforced SubPopulation coevolution. The current study aims at highlighting the internal dynamics of HCCE that give rise to its effectiveness in addressing difficult distributed design problems. The results obtained attest to the validity and effectiveness of HCCE, showing that it outperforms both other schemes.

Keywords: Hierarchical Coevolution; Cooperative Coevolution; Optimization; Distributed Modelling.

1. Introduction

Evolutionary optimization methods are very effective in tackling complex design problems, mainly due to their ability to overcome local optima [10,30,18]. Despite their effectiveness, ordinary evolutionary approaches are unable to consider the internal structural properties of the underlying problems [7,38]. This is because they utilize a unified representation to encode problem solutions to the genotype, and a single fitness function to evolve the composite solution as a whole [38,7].

However, in many real-life applications, partial entities can be identified, which together compose the overall picture of the problem [5,17]. These partial entities should often follow different design objectives as it is indicated by their specialized role in the composite system. In order to effectively address the structural characteristics of the problems, an optimization mechanism that follows a distributed organization would be particularly appropriate to support the design procedure[7,35]. This is the approach followed by coevolutionary algorithms which involve two or more concurrently performed evolutionary processes with interactive performance.

Specifically, by using coevolution, separate populations are utilized to evolve partial entities of the problem [34,41]. In order to formulate a composite problem solution, individuals within different populations have to be selected, put together and operate in parallel [3,33]. Each population can use its own evolutionary parameters (e.g. encoding, genetic operators). Accordingly, increased search competencies are inherently available in coevolutionary algorithms because they decompose the overall problem domain in small and more easily explored parts.

In the current study, we concentrate on cooperative coevolution that is suitable to address problems with explicit notions of modularity [34]. In previous studies, coevolutionary approaches have been used for optimizing distributed systems consisting of a small number of components [3,9,12,14,35]. However, these approaches are not able to address very complex systems consisting of a large number of components, or systems where the components are further decomposed to other components formulating gradually more simple structures. Additionally, in the above mentioned approaches, partial evolutionary processes are driven by the same fitness function and therefore they can not sufficiently address the distinct roles of substructures in the composite system.

We have recently introduced a Hierarchical Cooperative CoEvolutionary[22,23] scheme to overcome the problems mentioned above. Hierarchical Cooperative CoEvolution (HCCE) can sufficiently address problems described by multiple levels of modularity, investigating the independent roles of partial components at various levels, and additionally enforcing their coupled operation as a globally integrated system [21,24,26].

The HCCE (see[25] for a detailed description) evolves separate populations for each component of the system, and additionally evolves populations encoding assemblies of components. These two different kinds of evolutionary processes are hierarchically organized. The evolutionary processes at lower levels investigate the structure of partial system components. They are driven by their own dynamics, trying to meet the special design objectives of each component. At the same time, the evolutionary process at the higher level, explores the integrated performance of substructures trying to identify those component structures that can successfully cooperate. This higher level assembly-formulation process, tunes lower level component design procedures favoring the structures with the best cooperative performance. This is achieved by means of a newly introduced genetic operator named "Replication" [20]. The architecture of multiple coevolutionary processes tuned by a higher level evolution, can be repeated for as many levels as necessary, being capable to explore the structure of complex distributed systems.

In the current paper we present in detail the internal mechanisms of the HCCE scheme that give rise to its effectiveness in addressing difficult distributed design problems. Additionally, HCCE is compared with Unimodal evolution and Enforced SubPopulation (ESP) coevolution [13], showing a superior performance compared to both other schemes. Furthermore, we evaluate the "Replication" genetic operator, that has been introduced to support the successful convergence of the coevolutionary

procedure, coordinating partial evolutionary processes.

The rest of the paper is organized as follows. In the next section, we present the Hierarchical Cooperative CoEvolutionary (HCCE) scheme. Then, we present the experimental procedure used for validating the robustness of HCCE. Additionally, we present the results obtained by comparing HCCE with Unimodal evolution and ESP. Finally, in the last section, we conclude the paper highlighting the most important characteristics of HCCE.

2. Hierarchical Cooperative CoEvolution (HCCE)

In order to effectively build distributed systems, a mechanism that also follows a distributed architecture would be particularly appropriate to serve the design procedure. This is because the distributed design methodology can explicitly investigate the specialties of system components, and at the same time address their coupled performance in the composite system [18,29]. Along this line, coevolutionary algorithms have been recently proposed facilitating exploration in problems consisting of many decomposable substructures [3]. They involve two or more populations with interactive performance. Initial ideas of modelling coevolutionary processes were formulated by [1,28], and further extended in [15,32]. Following the coevolutionary approach, each population evolves separately using its own evolutionary parameters, providing increased search competencies. Distinct populations are usually referred as species in the coevolutionary literature, and thus both terms will be employed henceforth interchangeably.

Most of the coevolutionary approaches presented in the literature can be classified as competitive [6,31,36], or cooperative [34,40,19]. Competitive approaches are based on an antagonistic scenario, where the success of one species implies the failure of the other. In contrast, cooperative approaches follow a synergistic scenario, where individuals are rewarded when they successfully cooperate with individuals from the other species. In the following we only consider cooperative coevolution.

The majority of cooperative coevolutionary methods underestimates the significance of collaborator choice. Usually, the individuals of a species cooperate with the best individual from the other species or a randomly selected set of cooperators (e.g. [3,14,34]). This is also the case with Pareto coevolutionary approaches [7,11,12,16] which mainly aim at balancing between different criteria in multi-objective optimization problems. The issue of collaborator selection is very important, especially when investigating large systems consisting of many components.

In order to address the collaborator selection issue, a higher level optimization process can be used that searches within species identifying the individuals with the best coupled performance. This higher level search can be implemented by means of one more evolutionary process [7,35]. The process of simultaneous evolution of partial components and assemblies of components, can be organized hierarchically, formulating a multiple level scheme consisting of gradually more complex assemblies.

In the current work we present a Hierarchical Cooperative CoEvolutionary

(HCCE) architecture capable of designing complex systems consisting of a large number of components [26,23]. Besides the evolution of species corresponding to partial components, the proposed HCCE scheme employs additional higher level evolutionary processes, to select the proper individuals from each species that can cooperatively formulate effective component assemblies [25]. These configurations are used as a basis to guide the composite coevolutionary process since individuals are more likely to be members of effective assemblies of cooperators.

The work described in [9] presents a first attempt to hierarchically coevolved species. However, compared to [9], our approach is capable of coevolving larger assemblies of cooperating species, while at the same time, emphasizes the independence of substructures utilizing multiple and potentially separate criteria to guide partial evolutionary processes. Other approaches on hierarchical problem solving investigate the design of system components sharing the same set of objectives, and thus, they can not address effectively distributed systems consisting of heterogeneous substructures [7,8,39]. Additionally, they don't utilize specialized subpopulations that have been proved to facilitate significantly the evolutionary process [14].

2.1. *Hierarchical Organization*

The Hierarchical Cooperative CoEvolutionary (HCCE) scheme employs two kinds of species encoding the configurations of either a Primitive Structure (PS) or a Coevolved Group (CG). PS species are utilized to explore the structure of system components. A CG consists of a group of cooperating PSs all of them having the same objectives. Thus, CGs specify configurations of partial solutions, encoding assemblies of individuals. The evolution of CG modulates partly the evolutionary process of its lower level PS species to enforce their cooperative performance. A CG can also be a member of another more complex CG. Consequently, several CGs can be organized hierarchically, with the higher levels enforcing the cooperation of the lower ones.

The details of the HCCE are made clear by means of a specific example. Let us assume the existence of a system consisting of two partial structures and two links facilitating the flow of information (Fig 1(a)). We assume that the components have different roles in the functionality of the overall system. Specifically, component $C1$ and link $L1$ have to support the accomplishment of partial task $T1$, while component $C2$, and link $L2$, have to support partial task $T2$. Two coevolutionary groups $CG1$ and $CG2$ are used to coevolve these structures. Additionally, we assume that all structures have to cooperate in order to accomplish a global system functionality described by task $T3$. The top level $CG3$ enforces the cooperation among the groups $CG1$, $CG2$ integrating their functionality. The HCCE scheme that designs the components of the current example, is illustrated in Fig 1(b). Four PS species are employed to evolve the components and their links, while three CG species search for the best assemblies of cooperating individuals among PS species.

A snapshot of the exemplar HCCE process is illustrated in Fig 2. All individuals

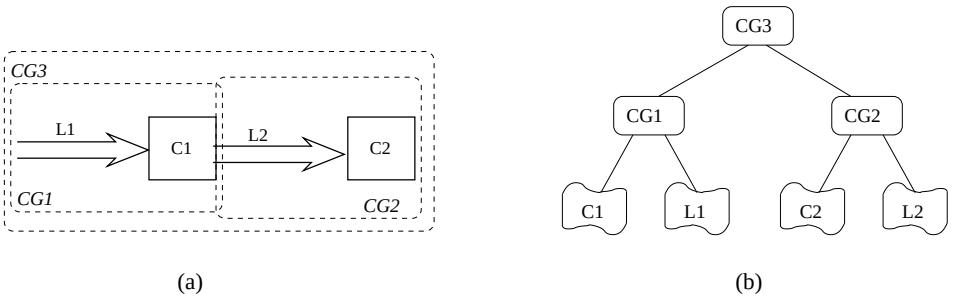

Fig. 1. The design of system substructures by cooperative coevolution. Part (a) represents graphically the internal connectivity of the system. The components are illustrated with blocks, while links are illustrated with double arrows. Part (b) represents the hierarchical coevolutionary scheme used to evolve system substructures. CGs are illustrated with rounded boxes, while PSs are represented by free shapes.

in all species are assigned an identification number which is preserved during the coevolutionary process. The identification number serves the definition of assemblies among different species. Each variable on the genome of a CG specifies the identification number of a candidate partial solution at the lower level. The arrows connecting individuals among species illustrate how the HCCE builds the proposed compound solutions. For example individual with $id = 7$ of species $CG3$ specifies a solution consisting of partial assemblies with $id = 19$ at $CG1$ and $id = 3$ at $CG2$. Analyzing further the first assembly, it consists of the individual with $id = 14$ at $C1$ species, and individual with $id = 21$ at $L1$ species. In the same way, analyzing the assembly of $CG2$, it consists of the individual with $id = 4$ at species $C2$, and individual with $id = 5$ at species $L2$. It is clear that individuals at CG species might select some components or assemblies of components multiple times, trying to identify the most successful sets of collaborators.

In order to test the performance of a complete problem solution, populations are sequentially accessed starting with the higher level. The genome values of CG individuals at various levels are used as guides to select cooperators among PS species. Then, PS individuals are decoded to specify the detailed structure of system components and links, and the performance of the proposed overall solution is tested on accomplishing the desired task.

2.2. *Partial Failure Investigation*

The HCCE scheme is capable of investigating the performance of the model in conditions of partial failure. Specifically, the deactivation of a CG together with its lower level PS species, simulates the elimination of system components. Additionally, separate fitness functions can be specified indicating the performance of the model when all substructures are present, and also indicating the performance when some partial structures are eliminated.

Turning back to the example of Fig 1, a $CG3$ individual specifies the structure

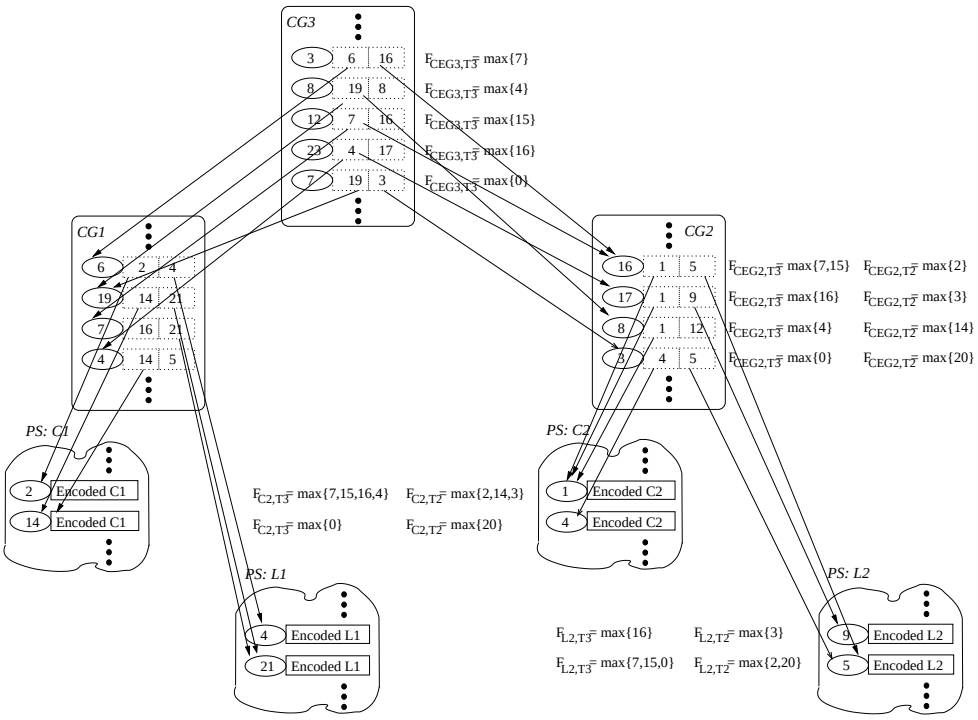

Fig. 2. An exemplar snapshot of the hierarchical coevolution of species. The arrows illustrate definition of individual assemblies. See text for details.

of the composite model aiming at the accomplishment of global task $T3$. Then, in order to simulate $C2$ failure, the components under $CG2$ are deactivated, and the remaining structures are tested on the accomplishment of partial task $T1$. Next, we simulate $C1$ failure, deactivating the components under $CG1$, and the remaining $CG2$ components are tested on the accomplishment of partial task $T2$.

By considering system performance in diverse operating conditions, important insight of the internal mechanisms of the system is provided to the designer. In particular, the exploration of system performance after the removal of components, can be very beneficial for inferring the roles of partial structures in the composite model and the dynamics of their interaction. It is worth emphasizing that the exploration of system performance in conditions of partial failure is very important for distributed systems consisting of autonomous components (e.g. agent-based systems) because it is capable of highlighting those functionalities of the system which are most affected after the collapse of one or more components.

2.3. *Fitness Assignment*

Although the majority of existing cooperative coevolutionary methods assume that all species share a common fitness function [3,19,41,13], the proposed scheme supports

the employment of separate fitness functions for different species. This is important for designing distributed systems because different objectives can be defined for the evolution of partial components.

During testing, we evaluate the coupled performance of all components. However, the fitness function of component species evaluates subjectively the overall performance, that is it evaluates the performance according to the objectives it is designed for. Specifically, for each species s, a fitness function f_s is used to drive its evolution. Different CGs can be driven by different fitness functions. However, all PS species under a CG share a common f_s. A partial fitness function $f_{s,t}$ evaluates the ability of an individual to serve task t. The composite fitness value is given by:

$$f_s = \prod_t f_{s,t} \tag{1}$$

The collaborator selection process at the higher levels of hierarchical coevolution will potentially select a lower level individual to participate in many assemblies (e.g. the case of individual 14 of PS species C1 of Fig 2). Let us assume that an individual participates in K assemblies, which means that it will get K fitness values $f_{s,t}$ regarding the accomplishment of the t-th task. The overall suitability of the individual on the task, is estimated by:

$$f_{s,t} = max_k\{f_{s,t}^k\} \tag{2}$$

where $f_{s,t}^k$ with $k \in \{1...K\}$, is the fitness value of the k-th solution formed with the membership of the individual under discussion.

The fitness assignment process is further explained by means of the working example illustrated in Fig 2. We remind the reader that according to the employed scenario, the composite model should accomplish task $T3$, the partial model of $C1$,$L1$ should accomplish task $T1$ (failure of $C2$), and the partial model of $C2$,$L2$ should accomplish task $T2$ (failure of $C1$). As a result, individuals of $CG3$ are evaluated for the accomplishment of task $T3$, individuals of $CG1$ and lower level PS species are evaluated for the accomplishment of both tasks $T3$ and $T1$, while individuals of $CG2$ and lower level PS species are evaluated for the accomplishment of both tasks $T3$ and $T2$. The assigned fitness values are illustrated in Fig 2, following the formulation introduced in eqs. (1) and (2). For the sake of brevity, we present fitness assignment only on $CG2$ and its lower level species. For the same reason we also assume that $F_{CG3,T3} = F_{CG2,T3}$, while in general they can be different.

The top level species $CG3$ is sequentially assessed and fitness values are estimated regarding the accomplishment of $T3$. Let us now examine the individual of $CG2$ with $id = 16$, which participates in two cooperator assemblies of $CG3$. Its ability to serve task $T3$ will be evaluated with the maximum of the respective fitness values. Additionally, $CG2$ individuals are assigned separate fitness values for the task $T2$ that they also serve. Thus, the individual with $id = 16$ is assigned one more partial fitness value. The same is also true for the individuals of lower level species $C2, L2$. For example, $C2$ individual with $id = 1$, has multiple participation

in the accomplishment of tasks $T3$ and $T2$ and its partial fitness values regarding the two tasks are estimated by the maxima of the respective values.

We also note the fitness assignment of the individual with $id = 4$ of $C2$. Although it receives a high score for its participation in task $T2$, it receives zero for its participation in $T3$, and consequently its aggregative score according to eq. (1) will be zero. Additionally, there are individuals which receive high aggregative score, even if none of the assemblies they participate in perform successfully in all tasks. An example illustrating this regards the individual with $id = 5$ of species $L2$. One of its cooperating assemblies receives a high score in $T1$ and a low score in $T2$, while the other receives a high score in $T2$ but a low score in $T1$. However, the individual under consideration will be assigned two high scores, because, depending on the assembly it participates in, it successfully serves both tasks. At the snapshot of the HCCE scheme shown in Fig 2, individual 5 of species $L2$ does not participate in overall effective assemblies, since none of them accomplishes both tasks. However, its high aggregative score increases the probability of participating in better assemblies in the next generation.

2.4. *Replication Operator*

A common problem for the coevolutionary approaches evolving assemblies of cooperators, is related to the multiple participation of some individuals in many different collaborator assemblies, while at the same time others are offered no cooperation at all [12,35]. A large number of multiple cooperations is generally a drawback for the coevolutionary process, because different cooperator assemblies could demand evolution of the same individual in different directions. Non-cooperating individuals can be utilized to decrease the multiplicity of cooperations for those individuals which are heavily reused. We have introduced a new genetic operator termed Replication, addressing the issue of multiple cooperations [23]. In short, for each unused individual x of a species, replication identifies the fittest individual y with more than max_c cooperations. The genome of y is then copied to x, and x is assigned $max_c - 1$ cooperations of y, by updating properly the CG population at the higher level. After replication, individuals x and y are allowed to evolve separately following independent evolutionary directions. This is illustrated in Fig 3, for the case of one CG and one PS species. Initially, individual 14 of the PS population participates in five solution assemblies, while individual 29 is offered no collaboration at all. Replication copies the chromosome of 14 to 29, and redirects two of the collaboration indicating pointers to 29. In the following evolutionary generations, individuals 14 and 29 follow separate evolutionary directions, facilitating the exploration of the search space.

2.5. *Evolutionary Step*

Evolutionary steps are performed separately but synchronously for each species of the HCCE scheme. First, individuals are sorted according to their fitness values.

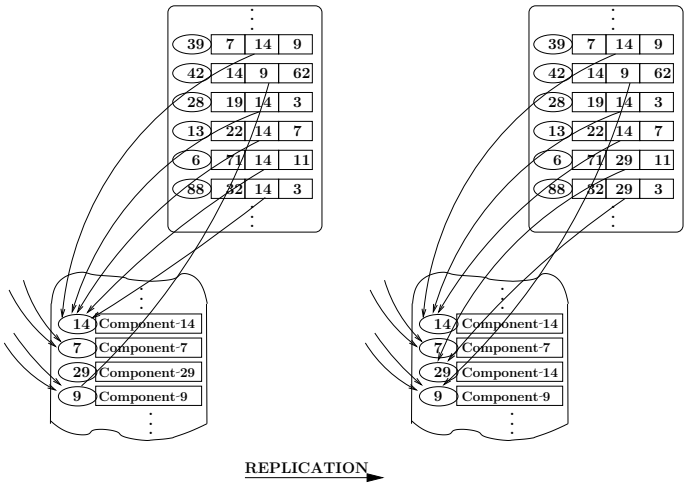

Fig. 3. Schematic representation of the replication operator ($max_c = 3$).

Then, Replication is applied to reduce multiple cooperations. Next, a predefined percentage of individuals are probabilistically crossed over. An individual selects its mate from the whole population, based on their accumulative probabilities. Finally, mutation is performed in a small percentage of the resulted population. This process is repeated for a predefined number of evolutionary epochs, driving the species exploring the structure of system components to the accomplishment of their own design objectives, and additionally driving the higher level cooperator selection processes to the formulation of successful composite structures.

3. Results

The effectiveness of the proposed coevolutionary scheme is assessed on the design of a complex distributed system. This approach is in contrast to other works employing mathematical functions as a test-bed for the study of evolutionary approaches. Particularly for the case of coevolutionary algorithms, mathematical functions based on few independent variables are usually employed [41,33], decomposing the overall problem in few and very simple entities. However, this approach can not reveal the power of each algorithm and its capability to address difficult problems. Additionally, previous works on hierarchically formulated problems investigated system components sharing common design objectives [39,8]. Thus they can not tackle systems consisting of heterogeneous substructures, as this is the case with HCCE.

In order to take a better insight of the coevolutionary procedure, the HCCE scheme is tested on the design of a complex brain-inspired cognitive system. The problem of developing brain-like computational structures fits adequately to coevolutionary approaches, because different coevolved species can be used to perform separate design decisions for the components representing brain areas, in order to

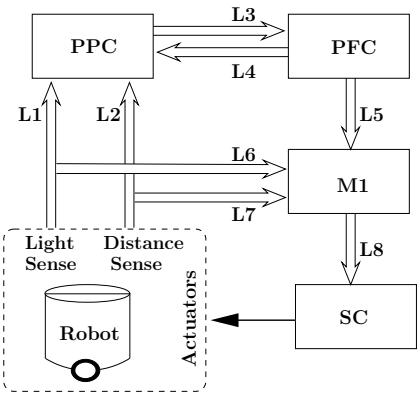

Fig. 4. A schematic overview of the computational model. Cortical agents are illustrated with blocks, while link agents are illustrated with a double arrow.

highlight their particular characteristics. An agent-based approach is followed to represent partial brain modules, addressing explicitly their autonomous role in the composite system [23]. Furthermore, following recent trends that study lesions of brain areas, the HCCE scheme supports systematic modelling of biological experiments, evaluating the performance of the model in pre- and post- lesion conditions.

During the modelling process, environmental interaction is of utmost importance, since it is difficult to investigate cognitive system functionality without embedding the model into a body to interact with its environment. Thus, a two wheeled simulated mobile robot is utilized to support environmental interaction, while at the same time the model enriches the behavioral repertory of the robot.

We note that the present series of experiments is an extension of our previous work which demonstrated the development of the cognitive working memory model [22]. Due to space limitations the problem will be described here in short, since the emphasis of the current study is to explore the dynamics of the HCCE design mechanism, rather than the biological reliability of the model.

3.1. *Problem Statement*

We study a partial brain model, which simulates posterior parietal cortex (PPC) - prefrontal cortex (PFC) - primary motor cortex (M1) - spinal cord (SC) interactions, emphasizing on working memory usage. The architecture of the model is demonstrated in Fig 4. The distributed model consists of 12 neural agents having separate self-organization dynamics. In particular 4 cortical components (represented by blocks) and 8 link components (represented by double arrows) are employed. The components have to develop different roles formulating a single, globally functional artificial system. Overall, the problem investigated here involves the specification of totaly 192 variables interacting in a highly non-linear way. Thus, the current problem can be used as an advanced test case for investigating the dynamics of HCCE, and revealing its beneficial characteristics.

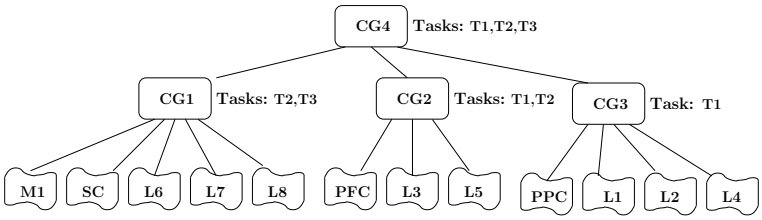

Fig. 5. A graphical illustration of the coevolutionary process.

Three tasks are designed to highlight the role of each agent in the model. The first task $T1$ accounts for developing working memory -like neural activity on PFC. The success on $T1$ task is evaluated by the fitness measure E_1. The second task T_2 aims at exploiting working memory in order to accomplish a delayed response task. This is evaluated by fitness measure E_2. The third task T_3, addresses the operation of the model in lesion conditions. Specifically, when simulated lesion is performed on PFC structure, the behavioral repertoire of the artificial organism is reduced (the accomplishment of the delayed response task is not possible any more), avoiding however collapse of the composite system. In that case, the robot is still able to drive but in a purposeless mode. This is evaluated by fitness measure E_3. Overall, the experimental process discussed above aims at reproducing computationally a biological scenario addressing pre- and post- lesion rat behavior in a T-maze [a]. The measures E_1, E_2, E_3 are described in detail in [22], and thus, they are omitted here.

3.2. *Computational Modelling*

We turn now to the design of the model by means of the HCCE. The coevolutionary scheme is demonstrated in Fig 5. According to the lesion scenario described above, each agent serves more than one tasks. This is illustrated in Fig 5, at the right side of each CG. Thus, different fitness functions guide the evolution of different components, enforcing the accomplishment of the respective tasks. In particular, in each species, individuals are assigned a combination of evaluation indexes, for the accomplishment of tasks where the composite model is performing, and the accomplishment of tasks with performance of the lesioned model.

Following the formulation introduced in eqs. (1), (2), the fitness function employed for the evolution of $CG1$ and its lower level species are:

$$\begin{aligned} &f_{CG1} = f_{CG1,T2} \cdot f_{CG1,T3} \\ &\text{with} \quad f^k_{CG1,T2} = \sqrt{E}_2, \\ &\qquad\quad f^k_{CG1,T3} = E_3 \end{aligned} \tag{3}$$

where k represents each membership of an individual in a proposed solution. The

[a]The biological relevance of the model has been discussed in [22].

fitness function employed for the evolution of $CG2$ and its lower level species are:

$$\begin{array}{l} f_{CG2} = f_{CG2,T1} \cdot f_{CG2,T2} \\ \text{with} \quad f^k_{CG2,T1} = {E_1}^2, \\ \qquad\quad f^k_{CG2,T2} = \sqrt{E_2} \end{array} \tag{4}$$

where k is as above. The fitness function employed for the evolution of $CG3$ and its lower level species are:

$$\begin{array}{l} f_{CG3} = f_{CG3,T1}, \\ \text{with} \quad f^k_{CG3,T1} = E_1 \end{array} \tag{5}$$

where k is as above. Finally the fitness function employed for the evolution of $CG4$ and its lower level species are:

$$\begin{array}{l} f_{CG4} = f_{CG4,T1} \cdot f_{CG4,T2} \cdot f_{CG4,T3}, \\ \text{with} \quad f^k_{CG4,T1} = {E_1}^2, \\ \qquad\quad f^k_{CG4,T2} = E_2, \\ \qquad\quad f^k_{CG4,T3} = \sqrt{E_3} \end{array} \tag{6}$$

where k is as above. It is reminded that all PSs share the same fitness functions with their higher level CG.

The coevolutionary process described above employed populations of 200 individuals for all PS species, 300 individuals for $CG1$, $CG2$, $CG3$, and 400 individuals for $CG4$. During the evolutionary steps, Replication threshold $max_c = 3$ is employed. Additionally, an elitist evolutionary strategy was followed in each evolutionary step with the 7 best individuals of each species, copied unchanged in the respective new generation, supporting the robustness of the evolutionary process. After 170 evolutionary epochs the process converged successfully and the cooperation of agent structures with completely different objectives, e.g. those under $CG1$ and those under $CG3$ is achieved (see eqs (3) and (5)).

3.3. *HCCE Assessment*

The problem of cognitive system design is used as a test case, investigating HCCE internal dynamics. Because of the embodiment of the cognitive system in the robotic platform and the observation of robot performance for a large number of simulation steps, the coevolutionary process demands on average ten hours to run for 170 evolutionary epochs, and it is impractical to test a large number of different runs. Therefore, we have performed six different runs of the hierarchical coevolutionary scheme. The obtained results are illustrated in Fig 6, where each column corresponds to a different run.

In the first run, the progress of the HCCE is initially slow, but after about 100 evolutionary epochs, the probabilistic search identifies a promising evolutionary direction which is efficiently exploited to identify a successful set of solutions. In the following two runs, we see that the coevolutionary process is rather unstable. Specifically, the evolution of species $CG4$ is not able to formulate successful assemblies of

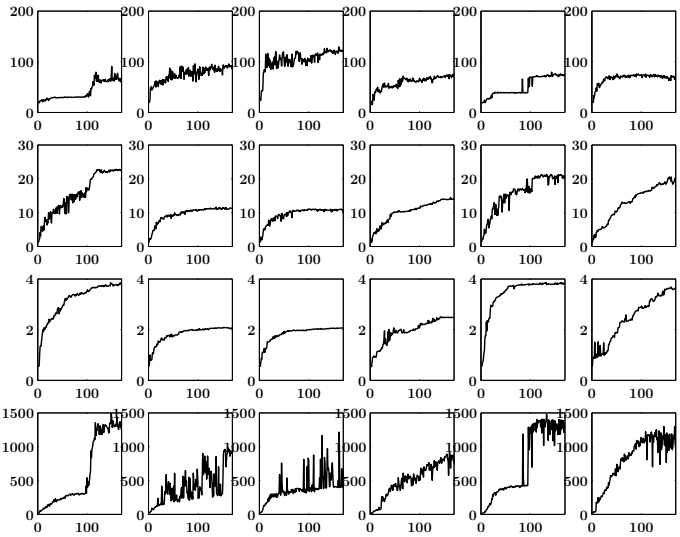

Fig. 6. Graphical illustration of the progress of six different HCCE coevolutionary runs. Each column is related to the results observed on the respective run. The lines 1-4 demonstrate the progress observed on the evolution of $CG1$, $CG2$, $CG3$, $CG4$, respectively. All plots demonstrate the fitness value of the best candidate solution in a population, against evolutionary epochs.

cooperators that will be preserved in the consecutive epochs. This fact additionally affects the progress of evolution in species $CG2$, $CG3$, which are trapped in sub-optimal solutions. In the fourth run, the progress of the composite coevolutionary scheme develops slowly, and simultaneously for all species. The coevolutionary procedure is terminated without reaching the success rate of the first run. Nevertheless, we easily observe that the progress of evolution is not stabilized, which implies that if the coevolutionary procedure could continue for more epochs, it should be able to estimate a sufficiently good result. The progress of the fifth run is similar to the first. The progress of the HCCE procedure is initially slow, when a promising assembly of cooperators is identified. After a small unstable period in the advancement of the coevolutionary procedure, an effective assembly is preserved, driving also the other individuals in an area of successful solutions. Finally, the progress of the last run is similar to the fourth. The evolution of each CG develops without rapid changes. However, in the current case, the advancement is a bit faster than the fourth run, and thus the composite procedure is able to converge in a set of solutions with a nearly optimum fitness value.

3.4. *Unimodal Evolutionary Design*

In order to get a better appreciation of HCCE effectiveness, we have also approached the problem at hand by using an ordinary evolutionary algorithm [4,30,37]. Specifically, the structure of all cortical and link agents is encoded in a single chromosome. In the

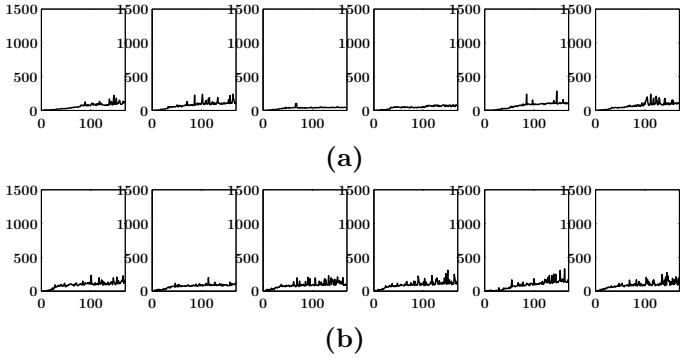

Fig. 7. Graphical illustration of the progress of Unimodal and ESP schemes. Each plot demonstrates the fitness value of the best candidate solution in a generation, against evolutionary epochs. (a) The results of six different runs of the ordinary evolutionary process. (b) The results of six different runs of the ESP scheme.

current set of experiments a population of 400 individuals is evolved for 170 epochs. The probability of applying crossover and mutation operators over the structure of a cortical or a link agent is the same with the respective probabilities of the coevolutionary scheme.

The fitness function employed to guide the evolutionary process is defined according to eq (6), similar to the function f_{CG4} that evolves the top-level CG, facilitating also direct comparison of Unimodal and HCCE processes. Specifically, the evolutionary process is driven by:

$$f = \sqrt{E_1} \cdot E_2^2 \cdot E_3 \tag{7}$$

The results of 6 independent runs of the unimodal evolutionary process are illustrated in Fig 7 (a) (compare with the last line of Fig 6). Evidently, none of the ordinary evolutionary processes was successful. Additionally, even the best of them, was not as good as the worst case of the coevolutionary scheme. These results highlight the unsuitability of unimodal evolution to design distributed structures with distinct roles of partial components, and the need for a specialized scheme able to consider explicitly the individual characteristics of substructures. All these issues are sufficiently addressed by the HCCE scheme.

3.5. *Enforced SubPopulation (ESP) Design*

Additionally, we have investigated the possibility of solving the same problem by utilizing the Enforced SubPopulation (ESP) coevolutionary scheme. In the current work, we have implemented the ESP algorithm described in [13], without however activating the stagnation check that practically re-initializes populations when the process gets stalled. Specifically, twelve different species are employed to specify the structure of the twelve components of the model. All partial populations of the ESP scheme are evolved according to the same fitness function describing the objectives

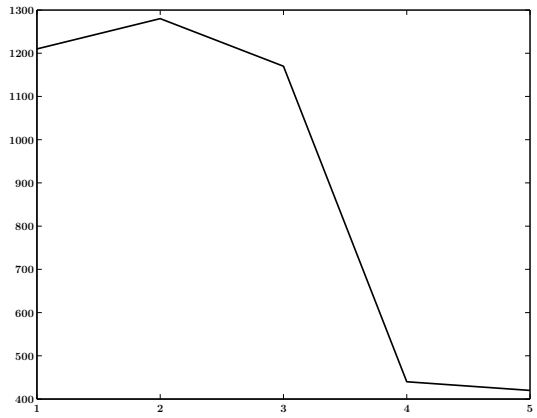

Fig. 8. The average fitness of six runs of the HCCE procedure, utilizing different values of replication threshold max_c.

of the overall system. Similar to the Unimodal evolutionary scheme, the fitness of ESP solutions is estimated by:

$$f = \sqrt{E_1} \cdot E_2^2 \cdot E_3 \tag{8}$$

Thus the progress of ESP is directly comparable with the progress of the HCCE scheme.

Similar to HCCE, each population evolving the structure of a component of the model consists of 200 individuals, while 2000 assemblies of components are randomly created, trying to identify a successful solution to the overall problem. Thus, each individual representing a candidate structure of a system component, participates in about ten complete solution assemblies. All solutions are evaluated according to eq. (8). The average fitness of individuals drives the evolution of each species.

We have performed 6 independent runs of the ESP scheme which are evolved for 170 epochs. The probability of applying crossover and mutation operators over the structure of a cortical or a link agent is the same with the respective probabilities of the HCCE scheme. The results of these processes are illustrated in Fig 7 (b). These results are directly comparable with the last line of Fig 6. Evidently, none of the ESP processes was successful. This is mainly due to the fact that ESP process is not directed towards explicitly creating successful complex assemblies. We note that in contrast to HCCE, the population of 2000 complete solution assemblies of ESP is not evolved but it is randomly generated in each evolutionary epoch [13]. In other words, it is expected that due to the large number of complex assemblies being evaluated, satisfactory distributed configurations will be randomly formulated. Unfortunately, as it is indicated by the present results, this is not the case for large problems where many components need to be coevolved.

3.6. *Replication Operator*

Additionally we have investigated the effect of the"Replication" operator, on the progress of coevolution. The effect of this operator is maximal when the value of the replication threshold max_c is low, and reduces gradually when max_c increases.

Intuitively, max_c balances the exploration versus exploitation dynamics of the coevolutionary procedure. High values of the replication threshold keeps individuals of partial species largely un-effected, being used as test cases for the individuals of the rest species. Thus, the dynamics of the coevolutionary procedure emphasize more on the exploitation of current results. In contrast, low values of replication threshold prevent individuals of partial species to participate in many cooperator assemblies, enforcing their evolution towards independent directions as it is indicated by mutation probability. In that case, the dynamics of the coevolutionary procedure emphasize more on the exploration of the search space.

We have investigated HCCE performance for five values of max_c threshold ($\{1, .., 5\}$), conducting six independent runs for each value. The average of maximum of $CG4$ fitness over the six runs is illustrated in Fig 8. Obviously, for the problem at hand, the successful convergence of the coevolutionary process is facilitated more by the exhaustive exploration of the search space. This is explained by the increased complexity of the problem and the high non-linear interaction among the partial elements of the solution.

3.7. *Discussion*

The results obtained in the current study attest to the validity and effectiveness of the HCCE scheme. Specifically, it is shown that HCCE is capable of successfully designing a system consisting of twelve complex partial components. Additionally, it is shown that HCCE significantly outperforms ordinary Unimodal evolution and the Enforced SubPopulation Coevolution. Comparing the computational demands of the methods, in each generation, HCCE and Unimodal Evolution evaluate totaly 400 candidate problem solutions, while ESP evaluates 2000 solutions (this is because the individuals encoding the structure of components have to participate in many composite assemblies, in order to obtain an average estimate of their suitability to the problem). Therefore, ESP needs considerably more processing time, because it inherently performs more evaluations. Alas, despite the increased amount of computational resources spent, the quality of the obtained results is rather poor for ESP. Clearly, HCCE is more successful than the other evolutionary methods. That is mainly for two reasons. First, HCCE is capable of explicitly addressing the autonomous characteristics of each system component, and second it is equipped with a systematic search mechanism identifying the component structures that can more effectively cooperate.

In order to obtain a complete view on the use of hierarchical coevolution as a general purpose optimization mechanism, HCCE can be further contrasted to a broader set of optimization methods, such as those based on Pareto ranking[12,16] or

Table 1. A tabular presentation of the features provided by different evolutionary approaches.

Procedural Feature	Hierarchical Cooperative CoEvolution	Pareto Evolution	Evolutionary Multimodel Partitioning
Global System Optimization	√	√	√
Single Component Optimization	√		
Partial Failure Exploration	√		
Probabilistic Fitness Function			√
Structural Identification			√
Multiple Criteria Fitness Function	√	√	√
Multiple Criteria Diversity		√	

multimodel partitioning[2,27]. The mentioned optimization methods are designed to approach different types of problems, and thus it would be unfair for them to be compared against HCCE on the problem investigated in the current study, focusing on systems with distributed architectures. Therefore, we constraint our study to a qualitative comparison of the aforementioned approaches, summarized in Table 1. The HCCE methodology is mainly targeted in addressing the distributed architecture of systems, while Pareto ranking aims at efficiently balancing population diversity based on multiple objective criteria, and evolutionary multimodel partitioning aims at structural system identification based on probabilistic fitness functions.

Due to the complementarity of the aforementioned approaches, it would be beneficial for HCCE to incorporate the advantageous features of the other methods that it is currently lacking. Unfortunately, this is not a straight forward procedure. In particular, with respect to Pareto optimality, it is not clear how fitness values should be propagated along the coevolutionary hierarchy, or how effective Pareto fronts should be formulated when partial populations have to satisfy multiple but different (for each population) fitness criteria. At the same time it seems that Pareto approaches are very time consuming when a large number of populations needs to be coevolved, because many individual combinations (the product of Pareto fronts) should be repeatedly tested.

A similar complexity problem would appear by integrating multimodel partitioning to HCCE, because many alternative neural network structures need to be explored and optimized for each system component, increasing dramatically the number of global problem solutions that need to be investigated. However, the probabilistic fitness measures used in multimodel partitioning can be rather easily integrated to HCCE. In the current work we have developed a first exploratory implementation, where all candidate solutions are tested on four random versions of the tasks $T1, T2, T3$ (the four versions correspond to statistically independent random initializations and different sequences of random sensory noise). As a probabilistic measure of a candidate solution suitability on a task, we use the average score in

the four different versions of a given task[b]. Following this approach for exploring the space of possible solutions, the amount of necessary computational resources are quadrupled compared to the original HCCE. The additional resources are exploited by HCCE to improve its robustness and stability. As it is shown in Fig 9, five out of six procedures have satisfactorily converged, resulting in successful neural network configurations. However, the more stable evolutionary process implies the lack of big exploratory steps (and the lack of big fitness jumps) preventing the identification of optimal neural network configurations (i.e. the best f_{CG4} fitness score in Fig 9 is less than the best f_{CG4} fitness score in Fig 7). In order to improve exploration we can increase the mutation rate of the evolutionary procedure, solving (at least in theory) this problem. Overall, the assimilation of multimodel partitioning features from HCCE seems to be beneficial for the coevolutionary procedure. However, the integration of the complete multimodel partitioning procedure on HCCE is an open research topic that needs to be explored further.

In the current study we have also investigated the effectiveness of the Replication operator. The obtained results demonstrated that the new operator supports the successful convergence of the coevolutionary process because it conveys information from the higher to the lower levels of the hierarchy, facilitating the coordination of partial evolutionary processes. In particular, Replication provides CG populations a means to modulate the evolution of lower level PS populations that encode the structure of system components. Thus, Replication operator can be utilized in other coevolutionary schemes that also evolve assemblies of individuals [12,35], in order to support the integration of partial components.

4. Conclusions

In the current paper we present Hierarchical Cooperative CoEvolution (HCCE), and we experimentally demonstrate that it is particularly appropriate for designing complex distributed systems. More specifically, by utilizing separate fitness functions, the proposed coevolutionary scheme addresses the specialized characteristics of system components, being capable of assigning them distinct roles. This is a clear advantage of cooperative coevolution compared to a unimodal evolutionary process that calls for a single fitness function, preventing the consideration of partial performance of substructures. Furthermore, the HCCE design mechanism facilitates the integration of autonomous components in a unified system, by means of evaluating the cooperative performance of substructures. The combination of partial autonomy and cooperative performance in a single design method, is very important for designing complex distributed systems, and thus HCCE can effectively address the distributed nature of computational systems. In summary, our study showed that HCCE is capable of:

[b]Similar to conditional probabilities, candidate solutions for each component are tested given that the rest components have a particular parameterization (see HCCE description in section 2).

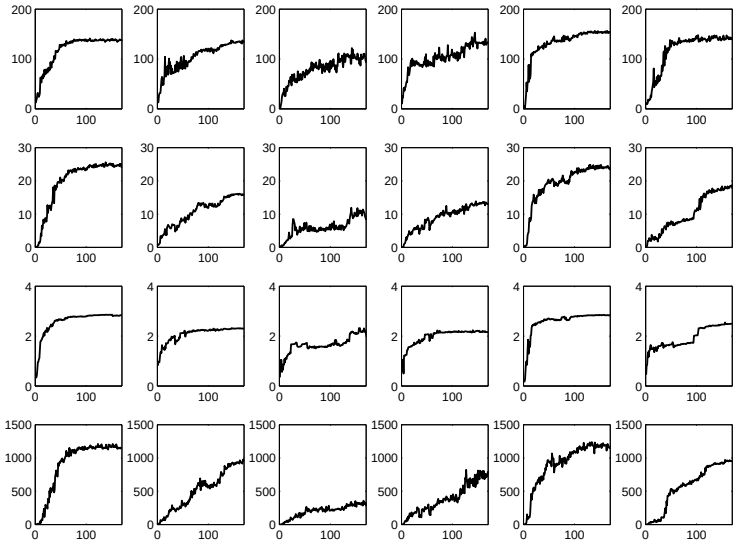

Fig. 9. The results of six coevolutionary process with probabilistic fitness functions (compare with Fig 7). Each column is related to the results observed on one run. The lines 1-4 demonstrate the progress observed on the evolution of $CG1$, $CG2$, $CG3$, $CG4$, respectively. All plots demonstrate the fitness value of the best candidate solution in a population, against evolutionary epochs.

- simultaneously coevolving a large number of partial components,
- enforcing the integration of components in a unified structure,
- considering the special role of each component in the unified system,
- investigating the performance of the overall system in conditions of partial failure.

It is worth emphasizing that despite the hierarchical organization of the coevolutionary process, the computational model is not necessary to operate in a hierarchical mode. This is because HCCE architecture does not directly correspond to the connectivity of system components that can be either hierarchical or completely parallel. Hence, the HCCE-based design method does not imply any constraints on the architecture of the distributed system.

In the future, we are planning to apply the HCCE-based design methodology in a range of different problems such as the design of cooperating robot teams, and the design of complex distributed mechanical systems consisting of different substructures. Additionally we will investigate possible directions for advancing the HCCE scheme. Research directions that seem to invite productive work concern the enrichment of the hierarchical scheme with Pareto optimality characteristics and multimodel partitioning structural optimization, as well as the design of new genetic operators specialized to improve the coevolutionary process.

Acknowledgments

The work presented in this paper has been partly supported by the European Commission funded project MATHESIS, under contract IST-027574.

References

1. R.M. Axelrod. *The evolution of cooperation*. New York: Basic Books, 1984.
2. G. Beligiannis, L. Skarlas, and S. Likothanassis. A generic applied evolutionary hybrid technique. *IEEE Signal Processing Magazine*, pages 28–388, 2004.
3. J. Casillas, O. Cordón, F. Herrera, and J.J. Merelo. Cooperative coevolution for learning fuzzy rule-based systems. In P. Collet, C. Fonlupt, J.-K. Hao, E. Lutton, and M. Schoenauer, editors, *Proceedings of the Fifth Conference on Artificial Evolution (AE-2001)*, pages 311–322. Springer Verlag, 2001.
4. Y. Chen, B. Yang, and J. Dong. Nonlinear system modelling via optimal design of neural trees. *International Journal of Neural Systems*, 14(2):1–13, 2004.
5. R. Coller. Agent factory: A framework for the engineering of agent-oriented applications. *Phd. Dissertation, University College Dublin*, 2003.
6. P. Darwen and X. Yao. Coevolution in iterated prisoner's dilemma with intermediate levels of cooperation: application to missile defence. *International Journal of Computational Intelligence and Applications*, 2(1):83–107, 2002.
7. E.D. De Jong. Representation development from pareto-coevolution. In *Proc. Genetic and Evolutionary Computation Confernce, (GECCO-2003)*, 2003.
8. A. Defaweux, T. Lenaerts, and J. vanHermet. Transition models as an incremental approach for problem solving in evolutionary algorithms. In *Proc. GECCO*, 2005.
9. M.R. Delgado, Von F.J. Zuben, and F.A.C. Gomide. Coevolutionary genetic fuzzy systems: a hierarchical collaborative approach. *Fuzzy Sets and Systems*, 141(1):89–106, 2004.
10. A.E. Eiben and M. Schoenauer. Evolutionary computing. *Information Processing Letters*, 82:1–6, 2002.
11. S.G. Ficici and J.B. Pollack. Pareto optimality in coevolutionary learning. In *Proc. of 6th European Conference on Artificial Life (ECAL-2001)*, pages 316–325, 2001.
12. N. Garcia-Pedrajas, D. Ortiz-Boyer, and C. Hervas-Martinez. Cooperative coevolution of generalized multi-layer perceptrons. *Neurocomputing*, 56:257–283, 2004.
13. F. Gomez. Robust non-linear control through neuroevolution. *PhD Thesis, AI-TR-03-303, Department of Computer Sciences, University of Texas at Austin.*, 2003.
14. F.J. Gomez and R. Miikkulainen. Solving non-markovian control tasks with neuroevolution. In *Proc. Sixteenth International Joint Conference on Artificial Intelligence - (IJCAI-1999)*, pages 1356–1361, 1999.
15. D. Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. In *Proc. Artificial Life II*, pages 313–324, 1992.
16. A.W. Iorio and X. Li. A cooperative coevolutionary multiobjective algorithm using non-dominated sorting. In *Proc. Genetic and Evolutionary Computation Confernce, (GECCO-2004)*, pages 537–548, 2004.
17. N.R. Jennings. On agent based software engineering. *Artificial Intelligence*, 117:277–296, 2000.
18. R. Kicinger, T. Arciszewski, and K. De Jong. Evolutionary computation and structural design: A survey of the state-of-the-art. *Computers & Structures*, 83:1943–1978, 2005.
19. K. Krawiec and B. Bhanu. Coevolution and linear genetic programming for visual learning. In *Proc. Genetic and Evolutionary Computation Confernce, (GECCO-2003)*, pages 332–343, 2003.

20. M. Maniadakis. Design and integration of agent-based partial brain models for robotic systems by means of hierarchical cooperative coevolution. *PhD Thesis, Department of Computer Sciences, University of Crete*, 2006.
21. M. Maniadakis, E. Hourdakis, and P. Trahanias. Modeling overlapping execution/observation brain pathways. In *Proc. Int. Joint Conference on Neural Networks, (IJCNN-07)*, pages 1255–1260, 2007.
22. M. Maniadakis and P. Trahanias. Distributed brain modelling by means of hierarchical collaborative coevolution. In *Proc. IEEE Congress on Evolutionary Computation, (CEC-2005)*, pages 2699–2706, 2005.
23. M. Maniadakis and P. Trahanias. Modelling brain emergent behaviors through coevolution of neural agents. *Neural Networks Journal*, 19(5):705–720, 2006.
24. M. Maniadakis and P. Trahanias. Assessing hierarchical cooperative coevolution. In *Proc. IEEE Int. Conference on Tools with Artificial Intelligence (ICTAI-07)*, pages 391–398, 2007.
25. M. Maniadakis and P. Trahanias. Agent-based brain modelling by means of hierarchical cooperative coevolution. *accepted for publication, ALife Journal*, 2008.
26. M. Maniadakis and P. Trahanias. Hierarchical coevolution of cooperating agents acting in the brain-arena. *accepted for publication, Adaptive Behavior Journal*, 2008.
27. G. Manioudakis, E. Demiris, and S. Likothanassis. A self-organized neural network based on the multi-model partitioning theory. *Neurocomputing*, 37:1–29, 2001.
28. S.J. Maynard. *Evolution and the theory of games.* Cambridge University Press, 1982.
29. P.B. Nair and A.J. Keane. Coevolutionary architecture for distributed optimization of complex coupled systems. *AIAA Journal*, 40(7):1434–1443, 2002.
30. V. Oduguwa, A. Tiwari, and R. Roy. Evolutionary computing in manufacturing industry: an overview of recent applications. *Applied Soft Computing*, 5:281–299, 2005.
31. B. Olsson. Co-evolutionary search in asymmetric spaces. *Information Science*, 133:103–125, 2001.
32. J. Paredis. *Artificial coevolution, explorations in artificial life.* Miller Freeman Inc., AI Expert Presents, 1995.
33. E. Popovici and K. De Jong. Understanding cooperative coevolutionary dynamics via simple fitness landscapes. In *Proc. Genetic and Evolutionary Computation Confernce, GECCO-2005*, 2005.
34. M. Potter and K. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8:1–29, 2000.
35. J. Reisinger, K.O. Stanley, and R. Miikkulainen. Evolving reusable neural modules. In *Proc. of Genetic and Evolutionary Computation Conference, (GECCO-2004)*. Springer Verlag, 2004.
36. C.D. Rosin and R.K. Belew. New methods for competitive coevolution. *Evolutionary Computation*, 5:1–29, 1997.
37. H. Surmann and M. Maniadakis. Learning feed-forward and recurrent fuzzy systems: a genetic approach. *Journal of Systems Architecture*, 47(7):649–662, 2001.
38. D. Thierens. Scalability problems of simple genetic algorithms. *Evolutionary Computation*, 7(4):331–352, 1999.
39. R.A. Watson and J.B. Pollack. A computational model of symbiotic composition in evolutionary transitions. *Biosystems*, 69(2-3):187–209, 2002.
40. R.P. Wiegand. An analysis of cooperative coevolutionary algorithms. *Phd. Dissertation, Department of Computer Science, George Mason University, USA*, 2003.
41. R.P. Wiegand, C.W. Liles, and A.K. De Jong. An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. In *Proc. of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 1235–1242. Morgan Kauf-

mann, 2001.