J Heuristics (2017) 23:1–30
DOI 10.1007/s10732-016-9322-9

Reference sharing: a new collaboration model for cooperative coevolution

Min Shi[1] · **Shang Gao**[2]

Received: 30 September 2015 / Revised: 16 October 2016 / Accepted: 28 December 2016 /
Published online: 25 January 2017

Abstract Cooperative coevolutionary algorithms have been a popular and effective learning approach to solve optimization problems through problem decomposition. However, their performance is highly sensitive to the degree of problem separability. Different collaboration mechanisms usually have to be chosen for particular problems. In the paper, we aim to design a collaboration model that can be successfully applied to a wide range of problems. We present a novel collaboration mechanism that offers this type of potential, along with a new sorting strategy for individuals that are assigned multiple fitness values. Furthermore, we demonstrate and analyze our algorithm through comparison studies with other popular cooperative coevolutionary models on a suite of standard function optimization problems.

Keywords Cooperative coevolution · Collaboration method · Fitness measurement · Function optimization

1 Introduction

Although approaches for evolving coadapted subcomponents have existed for several decades (Giordana et al. 1994; Holland 1986; Husbands and Mill 1991; Moriarty and Miikkulainen 1997; Paredis 1995), a general architecture of components-coevolved

✉ Min Shi
mins@met.no

✉ Shang Gao
shang.gao@oru.se

[1] Department of Computer and Information Science, Norwegian University of Science and Technology, Sem Sælands vei 9, 7491 Trondheim, Norway

[2] Department of Informatics, School of Business, Örebro University, 70182 Örebro, Sweden

algorithms, called cooperative coevolutionary algorithms (CCEAs), was not proposed until a decade ago (Potter and Jong 2000). A typical way to apply CCEAs to a problem is to decompose it into components, and then solve each component semi-independently in order to achieve the whole solution of the problem. There are three major steps involved in this procedure:

1. *Problem decomposition* This step determines how to divide a problem into components with an appropriate granularity. The division is carried out based on the structure of the problem solutions. Most of the current methods implement a natural decomposition where each component represents one or multi-dimensions of the optimized structure. The one dimension could be a single variable in a function optimization (Bucci and Pollack 2005; Potter and De Jong 1994; Potter 1997), or a hidden neuron in an evolved artificial neural network (Gomez 2003; Moriarty and Miikkulainen 1997).
2. *Components evolution* Each component is assigned to a population. A certain evolutionary algorithm (EA) is used, either homogeneous or inhomogeneous, to evolve each component. The same EA is applied between different components in homogeneous form, while different EAs might be employed in homogeneous form. Each population implements the evolutionary processes of reproduction and replacement independently of each other.
3. *Components coadaptation* During the above evolutionary process, collaborative relationships are built between different components during fitness assessment. When an individual of one population is evaluated, a collaboration is established by combining the individual with individuals selected from either other populations or a collaborator pool. The performance of the collaboration will be assigned to the individual as fitness. In the end CCEAs always output the combination of individuals that achieves the best collaboration as a final solution of the problem.

In step 1, problem decomposition can be static or dynamic. The first case decomposes, usually manually, a problem before starting the evolutionary process, and it does not alter the decomposed components afterwards (Bucci and Pollack 2005; Panait and Luke 2005; Potter and Jong 2000). The second case predecomposes a problem at the beginning, but components are able to be self-adaptively tuned to proper interaction levels during the evolutionary process (Ray and Yao 2009; Weicker and Weicker 1999; Yang et al. 2008a, b; Omidvar et al. 2014). In step 2, there are two main patterns to evolve components: sequentially and in parallel. In the sequential pattern, each population takes turns to evolve generation by generation (Potter 1997). In one generation only one population is active to execute evaluation, reproduction and replacement procedures, and the other populations are frozen. The active population is frozen in next generation, and another one is active and so forth. In the parallel pattern, evaluation is performed after all populations execute reproduction and replacement in each generation (Gomez 2003; Wiegand 2004). A master–slave architecture (Parsopoulos 2012) can be applied to the parallel pattern to reduce the total computation time of the entire coevolutionary system. The third step is a crucial step in CCEAs. "Survival of the fittest" is the underlying principle of evolution. How to determine if an individual is fit or unfit, however, is neither direct nor definite in CCEAs, because the individual

has to collaborate with others. An individual could receive high fitness when in one collaboration, but a low one in another. Good or bad is not absolute, but relative to its references. Many practitioners have addressed this issue, which will be discussed in the next section. This paper also proposes a new collaboration and evaluation model used in step 3; the algorithm introduced in this work can be applied to both sequential and parallel CCEAs, regardless of the decomposition strategies. The experiments of this paper use a sequential CCEA.

CCEAs have been applied to a great number of optimization problems with varying success, including function optimization (Bucci and Pollack 2005; Potter and De Jong 1994), evolving artificial neural networks (García-Pedrajas et al. 2003; Gomez 2003; Shi and Wu 2008), learning fuzzy systems (Casillas et al. 2002; Pena-Reyes and Sipper 2000). A major characteristic of CCEAs is the ability to simplify the complexities of a problem through decomposition. However, everything has two sides. Decomposition could make a problem easier to solve if the interaction between decomposed components is weak. And it could also make a problem harder to solve if there is strong linkage between decomposed components. To simplify our description, we would like to define the former problem to be a separable problem, and the latter one to be a nonseparable problem.

For a nonseparable problem, a high degree of interaction between components (a.k.a. epistasis) exists such that the fitness contribution of one gene is highly dependent upon other genes, and, in general, optimality is less absolute and more inclined to exhibit a Nash equilibrium (Nash 1951). This linkage can occur between genes of the same component or different components. Standard CCEAs, combining static decomposition in step 1 with sequential evolution in step 2 and a greedy collaboration strategy (to be explained in Sect. 2) in step 3, do not consider the Nash equilibrium when evaluating individuals. The performance of standard CCEAs has been reported to lag behind that of traditional EAs, evolving the whole solution in one population, when a high degree interaction exists between components (Potter 1997; Sofge et al. 2002; Weicker and Weicker 1999). Watson and Pollack (2005) discussed the evolvability of systems with significant inter-module dependencies. They found that these systems are evolvable under certain evolutionary scenarios (e.g., compositional evolution). This also leaded to a different understanding of the impact of inter-module interactions on evolvability. Some efforts have been made to analyze the effects on the performance of the CCEAs on this kind of problem (Popovici and De Jong 2005, 2006; Wiegand et al. 2002). A few approaches have been proposed to improve CCEA models for handling epistatic problems.

Potter (1997) proposed an alternative collaboration strategy, called the *less greedy* strategy, for his cooperative coevolutionary model. In this strategy, an individual collaborated with both the best individuals and random individuals chosen from other populations, and was assigned the best fitness of both evaluations. For some problems, this collaboration achieves a slight improvement over CCEAs with a greedy strategy, but it is still inferior to traditional EAs for problems with a high degree of epistasis.

A blended population algorithm, proposed by Sofge et al. (2002), combined a CCEA with an EA. Their algorithm built both multiple populations evolved by the CCEA and one population evolved by the EA. Individuals were allowed to migrate

from the CCEA to the EA over the evolutionary process. The blended population algorithm handles epistasis slightly better. However, they only reported their algorithm on function optimization in two dimensions.

Weicker and Weicker (1999) developed an adaptive coevolutionary algorithm. At the beginning their algorithm worked like standard CCEAs where components were coevolved in separate populations. During the coevolutionary process, the decomposed components could be gradually reduced through population combination once enough epistatic links were observed. Through the self-adaptive process, their algorithm was able to achieve the best performance between traditional EAs and standard CCEAs for problems with and without epistasis.

In contrast to Weicker, Ray and Yao (2009) introduced an algorithm to self-adaptively segment the components of a problem. The algorithm is called Cooperative Coevolutionary Algorithm with Correlation based Adaptive Variable Partitioning (CCEA-AVP). They applied their algorithm on a set of function optimization problems. The CCEA-AVP started from an EA; it dynamically decomposed a problem into components according to a predefined correlation coefficient between variables. The correlation based variable partition was repeated at every subsequent generation until a predefined maximum number of components was reached. Similar to the algorithm proposed by Weicker and Weicker (1999), CCEA-AVP achieved a tradeoff between EAs and CCEAs for both separable and nonseparable problems.

A recent research by Omidvar et al. (2014) proposed an automatic decomposition strategy called differential grouping. Their method at first detects the underlying interaction structure of decision variables, and then form subcomponent based on the detection such that the interdependence between the variables is kept to a minimum. An automatic near optimal decomposition of decision variables is implemented in this method, which is also helpful to handle epistasis. They especially demonstrated their method to be beneficial in solving large-scale global optimization on both separable and nonseparable problems.

This work proposes an improvement to CCEAs based on a new collaboration model known as *Reference Sharing (RS)*. The modified algorithm is called CCEA-RS in this paper. The most distinctive feature of CCEA-RS is the construction of collaborations. Instead of selecting collaborators from other populations, all individuals of populations cooperate with the references in an archive and they all share a single archive. An individual could receive multiple fitness values depending on the number of references in the archive. To measure individuals with multi-fitness values, we also describe a new sorting algorithm, even-distribution sorting. Both the collaboration model and the fitness measurement could be applied to other cooperative coevolutionary models. We evaluate our algorithm on a suite of test functions including both separable and nonseparable problems, and compare its performance with CCEAs using two different collaboration strategies. Interestingly, our algorithm achieves pretty good results in all the cases. For nonseparable problems, CCEAs were claimed to be attractive only when the problems have a large number of variables (Ray and Yao 2009; Yang et al. 2008a, b). However, CCEA-RS can tackle epistasis in smaller-dimensional problems as well.

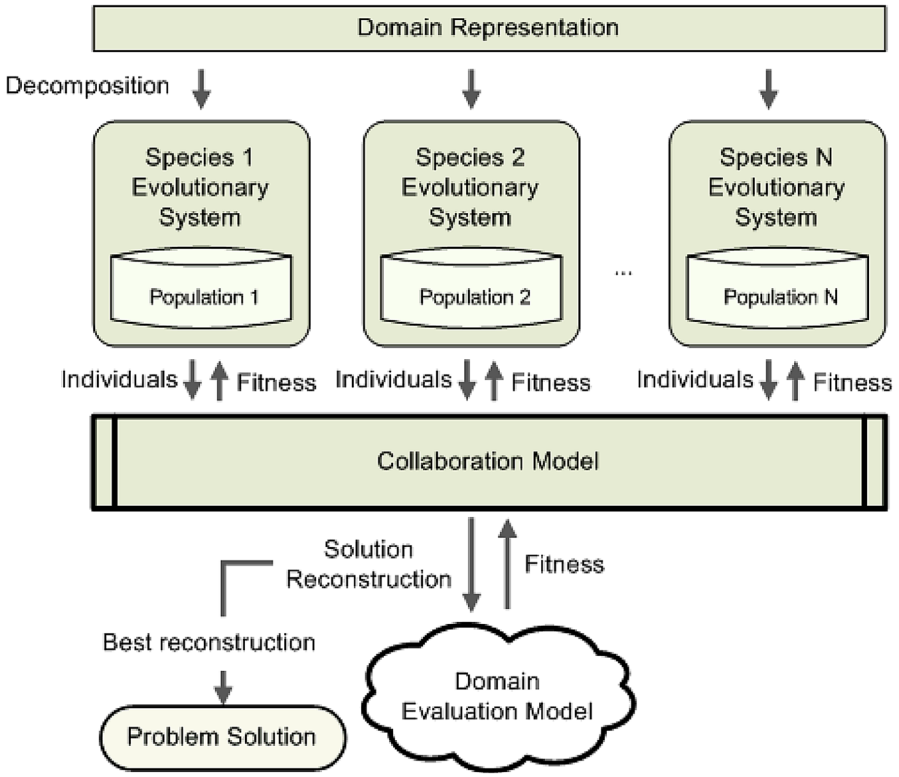

Fig. 1 A general framework of CCEAs

1.1 New architecture of cooperative coevolution

This work includes modifications to the generalized architecture of cooperative coevolution proposed by Potter and Jong (2000). The new architecture is shown in Fig. 1. There are three major improvements in our framework.

1. The original architecture of cooperative coevolution does not illustrate the decomposition process of a domain. The first step, domain decomposition, is indispensable in cooperative coevolution. The decomposition is carried out on the representation of the domain. This procedure decides how to divide the domain into components and assigns each component to an evolutionary system, although the decomposed components could be changed during the evolutionary process in some algorithms. Each component is regarded as a species to be evolved.
2. Instead of taking turns to activate one evolutionary system and freeze others, all the evolutionary systems are shown in the same level in our framework. Thus, practitioners are free to design their patterns, either sequential or parallel, as introduced previously, when evolving species. Some more complex patterns can also be applied, such as a sequential pattern with various interaction frequencies (Popovici and De Jong 2006).

3. Our architecture adds a collaboration model that connects evolutionary systems with a shared domain. Nowadays more and more collaboration schemes have been proposed for cooperative coevolution. Individuals of one population collaborate with representatives selected from other populations, as described by the original CCEA architecture. Unfortunately, this is not sufficiently general to handle the majority of collaboration schemes. Through the utilization of the collaboration model, we are able to introduce different collaboration methods into the algorithms. The collaboration model not only decides how to select collaborators, how to establish collaborations, how many collaborators to use, and when interactions among populations happen, but also decides how to accumulate the outcomes of the collaborations evaluated by the shared domain and to distribute that fitness back to the individuals.

This paper is organized as follows. Section 2 introduces some existing collaboration models and discusses some important issues addressed by different models. Section 3 describes our algorithm, including the new collaboration model and three sorting strategies used to measure individuals in our model. Our test domains are presented in Sect. 4. Section 5 conducts an empirical comparison between our method and others, where we find that CCEA-RS is more robust for problems displaying a wide range of separability. The analysis is carried out in Sect. 6. Section 7 summarizes the work and presents topics for future research.

2 Collaboration models in cooperative coevolution

2.1 Existing methods

Although exhaustive testing of each individual's collaborative potential with every other individual in every other population may facilitate global optimization, it is also exponential in the number of individuals, and thus computationally expensive. This has motivated the design of a wide variety of collaboration models that use only a small, but effective, subset of these combinations.

One of the earliest collaboration models proposed by Potter for the generalized architecture of CCEAs can be called a *1 + 1 collaboration model* (Fig. 2a) (Potter and Jong 2000). In this model, every individual of a population undergoing evaluation cooperates with individuals selected from other populations, one from each. Those selected collaborators are called *representative*. One collaboration constructs a complete problem solution, which can then be evaluated in the problem domain. It is important to note that in this model, only one collaboration is built for evaluating each individual. The performance of the collaboration will only be assigned as fitness to the evaluated individual, but not to its collaborators. To choose the representatives, Potter suggested a greedy collaboration in which the current best individual of each population is the representative for some cases, while alternative strategies, such as random selection of the representatives, can be used for other cases. In contrast to Iorio and Li (2004) select a representative randomly from the best non-domination level of each population. A detailed explanation of non-domination can be found in Sect. 3.2.

For handling nonseparable problems, a *1 + N collaboration model* (Fig. 2b) was employed to try to decrease susceptibility in Nash equilibria (Potter 1997). Instead of performing an evaluation based on one collaboration, N-collaborations are established for each individual per generation in this model. Different strategies, such as greedy, random strategy and others can be simultaneously introduced to select collaborators from other populations. Multiple collaborations produce multiple fitness values. Potter applied sizes 2 to N and greedily assigned the best performance between the two collaborations as the fitness. Bucci and Pollack (2005) employed the same collaboration size and the same selection strategies, but used a Pareto dominance mechanism to evaluate individuals. An empirical study analyzed which selection strategies, select pressure and collaboration size were appropriate for a particular problem using the 1 + N collaboration model (Wiegand et al. 2001). More recently, varying the numbers of collaborators overtime was demonstrated to be better than fixed collaboration schemes (Panait and Luke 2005).

An archive-based collaboration model (Panait et al. 2006), a variant of 1 + N collaboration model, maintains collaborators in archives, one per population. The archives preserve useful information in past generations. Those individuals who mostly help individuals from other populations to improve themselves are intended to be good collaborators and encouraged to collaborate with new individuals in the next generation. Panait et al. (2006) designed their archives with a dynamic size. Initially, the archive of each population was a copy of the population itself. The size was reduced by removing individuals who were not able to raise the rank of other individuals. The purpose was to build minimal archives while guiding accurate evaluation.

In the research field of evolving artificial neural networks (EANNs), ESP (Gomez 2003) uses cooperative coevolution with an *N + N collaboration model* (Fig. 2c). This model constructs N collaborations for evaluating all individuals of all populations per generation. One collaboration is formed by randomly selecting an individual from each population; the performance of the collaboration is accumulated from every individual who takes part in this collaboration. After this collaboration pattern has been repeated N times, each individual obtains an average fitness of the collaborations that it participated in. In addition, a shuffling process is introduced to guarantee that each individual gets chances to participate in collaborations (Hoverstad 2007). In the revised version, the order of individuals in each population is shuffled before evaluation, after which the ith individual is selected from each population to form the ith collaboration. This method calls the shuffling process again when the last individual of a population has been taken. The one-to-one matching scheme is repeated for all N collaborations.

Another technique introduced in EANNs can be called an *evolutionary collaboration model* (Fig. 2d) (García-Pedrajas et al. 2003; Moriarty and Miikkulainen 1997). Instead of predefining a collaboration scheme, this model searches for the optimal collaboration of individuals using an evolutionary algorithm. Thus, besides evolving components, an additional population of *blueprints* evolves the combinations of the collaborative individuals of the components in parallel. Each individual of the blueprint population represents one combination of collaborative individuals. At the beginning of blueprint evolution, combinations are created randomly. Effective combinations can be maintained and new combinations forms can be explored by evolving the blueprint

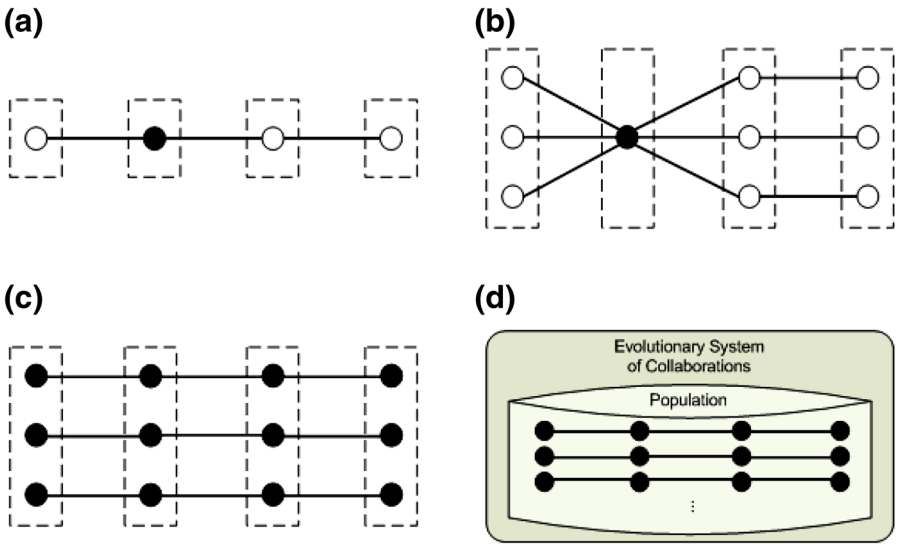

Fig. 2 Collaboration models. *Black circles* are individuals undergoing evaluation; they will be assigned fitness after finishing all collaborations shown in the models per generation. *White circles* are collaborators selected from other populations; they merely participate in collaborations, and will not receive fitness. In (**a**–**c**), all nodes in the same dashed square are individuals from the same population. Although these figures illustrate the collaboration models with four populations and N = 3, the actual number of populations and the size of N can vary. In (**b**), collaborators, marked by *white circles*, can be selected from other populations or the archives of the populations. In (**d**), the *black circles* are individuals evolved in parallel by the component evolutionary systems; or they can stem from a single population, as in SANE (Moriarty and Miikkulainen 1997) or multi-populations, as in COVNET (García-Pedrajas et al. 2003). **a** 1 + 1 Collaboration mode, **b** 1 + N collaboration model, **c** N + N collaboration model and **d** evolutionary collaboration model

population. During the evaluation phase of this model, each individual is assigned an average fitness of the collaborations throughout the blueprints that the individual participated in, as in ESP (Gomez 2003).

Some researchers have investigated the problem of collaboration model under the perspective of genetic programming (GP). For example, Doucette et al. (2012) proposed a genetic programming-based learning algorithm, called Symbiotic bid-based (SBB) GP for cooperatively evolving GP teams. It coevolves three populations: A point population, a team population and a learner population. The learner population represents a set of symbionts (learners), which associate a GP-bidding behavior with an action. Further, Hierarchical task decomposition through symbiosis in reinforcement learning has been applied in reinforcement learning (Doucette et al. 2012).

Thomason and Soule (2007) proposed an approach called orthogonal evolution of teams (OET). This approach overcomes the weaknesses of island and team approaches by applying evolutionary pressure at both team and individual levels during selection and replacement. And Wu and Banzhaf (2010) proposed a new computational multilevel selection framework. This framework extends evolution from individuals to multiple group levels, through which cooperative solutions can be hierarchically built out of simple ones. Furthermore, Wu and Banzhaf (2011) introduced a multilevel genetic programming (MLGP) system base on computational multilevel selection

framework to tackle the evolution of cooperation. The applicability of MLGP is also demonstrated under the context of GP classification.

When it comes to the approach targeting mechanisms for collaboration formulation, Kim et al. (2001) find that symbiotic evolution tends to strengthen the parallel search capability of an evolutionary algorithm, whereas endosymbiotic evolution is effective in speeding up the solution convergence. And Watson and Pollack (2003) have investigated the use of coevolution as a problem solving technique.

2.2 Issues involving collaboration models

By decomposing the representation of a problem into pieces, the search space of the problem solution is decomposed into sub-spaces as well. Optimal search algorithms cannot reach the optimal solution of the problem simply through isolated optimal searches on each of the sub-spaces due to the linkage of fitness landscapes between the sub-spaces. Collaboration models build connections between the fitness landscapes of the components when implementing evaluations. Still, different problems could be caused by different collaboration models including over-specialization, over-generalization and incomplete linkage.

Over-specialization is known as a focusing problem, which has been widely discussed in competitive coevolution (Watson and Pollack 2001; Bucci and Pollack 2003; Jong and Pollack 2004), but rarely discussed in cooperative coevolution. In competitive coevolution, over-specialization implies that algorithms focus on achieving partial underlying objectives, such as only beating the weaknesses of opponents, rather than evolving general solutions to meet all objectives. The focusing problem could also happen in cooperative coevolution when evaluating individuals based on collaborations with over-specialized collaborators. Obviously, the $1+1$ collaboration model is one that most often encounters this problem. Only selecting one individual from a collaborative population to be the representative of the population typically losses much information about the component. Such a case sometimes leads to premature convergence. This is a good reason for adding an extra random collaboration, or other less-greedy collaboration.

Over-generalization indicates an exactly opposite situation, where individuals collaborate with over-generalized collaborators. The over-generalized collaborators usually contain a fairly large percentage of randomly-selected individuals. Over-generalization often occurs when evaluated individuals are assigned average or maximum fitness produced from such collaboration groups. As a result, CCEAs tend to find solutions that are robust under partial solution changes (Wiegand 2004). To alleviate this problem, multi-fitness measurement techniques have been introduced to evaluate individuals who receive multi-fitness in coevolution (Bucci and Pollack 2005; Iorio and Li 2004; Jong and Pollack 2004). Non-dominated sorting (Deb et al. 2000) is one of the most widely used sorting algorithms used for achieving this purpose.

The collaboration models introduced in the previous section build connections between decomposed components, but do not completely rebuild the linkage of these components. The linkage represents intra- or inter-component gene interaction. When all components are represented and evolved in the same population, the interaction

among the components is represented at an expected level due to unbroken linkages. However, the interaction will be degraded when these components or genes are evolved in separate populations, where the linkage is severed. In such a case, the interaction could be represented at a far from expected level depending on the separabilities of problems. A nonseparable problem definitely has stronger interaction among components than a separable problem. Linkage cannot be simply rebuilt by connecting the components from different populations when implementing evaluations. This is a crucial reason that the performance of standard CCEAs could lag behind that of EAs for problems with a high degree of epistasis.

In summary, the following recommendations should be taken into consideration when designing a robust collaboration model for CCEAs: (1) evaluate individuals based on multi-collaboration model as much as possible; (2) pay careful attention to time-consumption when designing multi-collaboration models; (3) avoid assigning only a single fitness value when assessing an individual in a multi-collaboration model, and (4) reconstruct linkage between decomposed components or genes.

3 The CCEA-RS

Our CCEA-RS implementation preserves the principal architecture of the standard CCEA developed by Potter (1997). There are two main modifications over the standard CCEA. First, during the evaluation stage, a new collaboration mechanism is used for measuring the fitness of an individual. The evaluated individual does not collaborate with individuals selected from other populations, but with members of an archive, known as *references*. After evaluation, each individual receives one or more fitness values, depending on the number of references in the archive. Section 3.1 explains the collaboration formation in detail. Second, we assess whether or not one individual is superior to another through various multi-fitness assessment strategies instead of via a single fitness value. Various sorting strategies can be chosen; we introduce three of them in Sect. 3.2. A general framework for our algorithm is shown in Fig. 3.

In our implementation, we generate a new population by preserving the top n individuals of the previous generation, and replace the worst $2n$ individuals with sexual reproduction of random individuals from the top n. Other selection and replacement mechanisms can also be applied to this general framework.

3.1 The reference sharing collaboration

All the populations in the CCEA-RS share one archive. This archive stores references for evaluation. Each individual of one population cooperates with every reference in the archive. The size of the archive is a predefined parameter in our algorithm. Each reference in the archive represents a complete solution and thus has a fitness value. To initialize the archive, every reference is formed by concatenating randomly selected chromosomes from each initial population. A buffer context vector shows similar collaboration mechanism was applied in cooperative particle swarm optimization (Li et al. 2015; Parsopoulos 2012). In their work, this context vector was employed to save the global optimal information from each subswarm. Then, each

```
gen=0
for each species s do begin
    Pop_s(gen) = randomly initialized population
    end
for each reference r in the archive do begin
    Archive_r(gen) = randomly initialized collaborators
    evaluate fitness of Archive_r(gen)
    end
for each species s do begin
    evaluate fitness of Pop_s(gen) based on Archive(gen)
    update Archive(gen)
    end
while termination = false do begin
    gen = gen +1
    for each species s do begin
        sort Pop_s(gen-1) using multi-fitness measurement
        generate Pop_s(gen) from Pop_s(gen-1)
        apply mutation to Pop_s(gen)
        evaluate fitness of Pop_s(gen) based on Archive(gen)
        update Archive(gen)
        end
    end
```

Fig. 3 A general framework of the CCEA-RS algorithm

subswarm was evaluated by using the context vector to complement the missing components. In our collaboration model, multiple references can be applied. We also propose a new measurement strategy to evaluate individuals who receive multiple fitness.

The formation of collaborations is illustrated in Fig. 4. Assume that there are n populations coevolved for one problem and the size of the archive is m. Then m collaborations will be formed when an individual collaborates with each (of the m) references in the archive. We will evaluate the individual based on all the m collaborations, and assign all the m fitness values to the individual. For example, when individual i of population p collaborates with reference j of the archive, the chromosome of individual i initially merges into the chromosome of the reference j by replacing the *pth* chromosome segment with the chromosome of individual i (see Fig. 4). We then evaluate the resultant solution and assign the result to the individual i as its *jth* multi-fitness entry. After evaluating all the collaborations, each individual will receive m fitness values. The archive is updated online. When we measure the collaborations during evaluation, the *pth* chromosome segment of the reference j in the archive is replaced with the chromosome of the individual i as long as the fitness of the *jth* collaboration is better than the fitness of the reference j. An example of how a collaboration is implemented and when the archive is updated have been shown in Fig. 5.

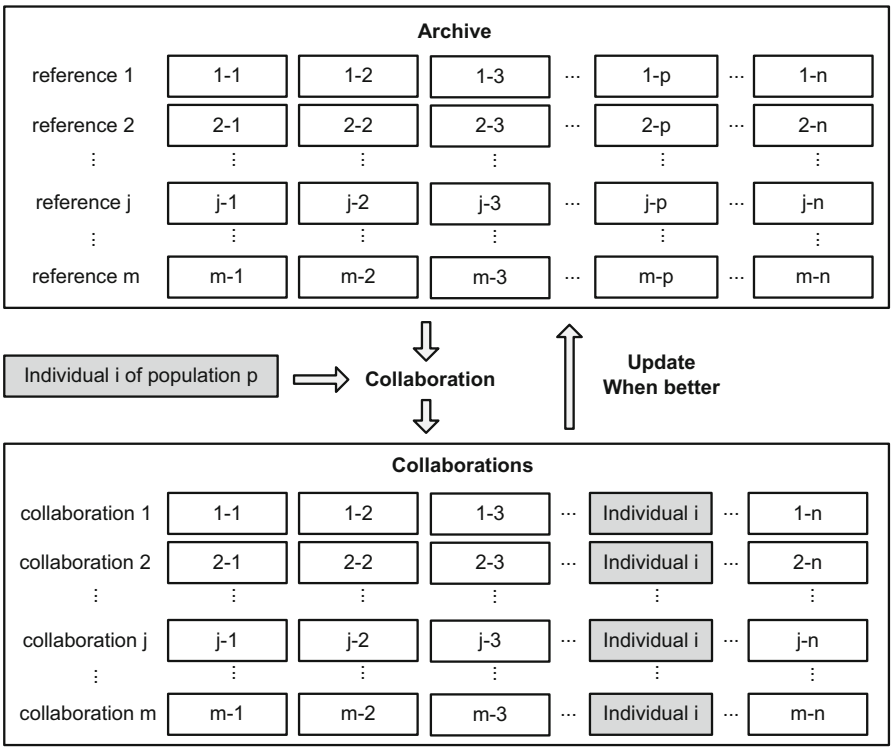

Fig. 4 Outline of the collaboration framework in CCEA-RS. *White blocks* denote the chromosome segments of references in the archive; *grey blocks* denote the chromosome of an individual in populations. There are two numbers in each *white block*. The *first* indicates the index of the reference/collaboration, while the *second* denotes the index of the chromosome segment of that reference/collaboration. The length of the *pth* chromosome segment of a reference equals the chromosome length of an individual in the *pth* population

3.2 Multi-fitness measurement

Traditionally, in evolutionary and coevolutionary algorithms, a single fitness value is assigned to an individual (Angeline and Pollack 1993; Hillis 1990; DeJong and Spears 1991; Potter and Jong 2000; Rosin and Belew 1997; Spears et al. 1993). The comparison of fitness among individuals is straightforward: the closer to the objective value the better the fitness. More recently, multi-fitness evaluation has received growing interest, especially in coevolution. The idea originates from Multi-Objective Evolutionary Algorithms (MOEA) (Fonseca and Fleming 1993; Horn et al. 1994; Srinivas and Deb 1994; Zitzler and Thiele 1998; Deb et al. 2000), wherein each individual receives a fitness value for each objective and individuals are sorted (for selection) using various ranking strategies.

Further, as described in Sect. 2.2, we have found some related works on multi-fitness measurement. For example, Thomason and Soule (2007) assume an orthogonality scheme. Kim et al. (2001) assume a pairwise replacement algorithm. Watson and Pollack (2003) define a pairwise dominance relation at the genome level. Doucette

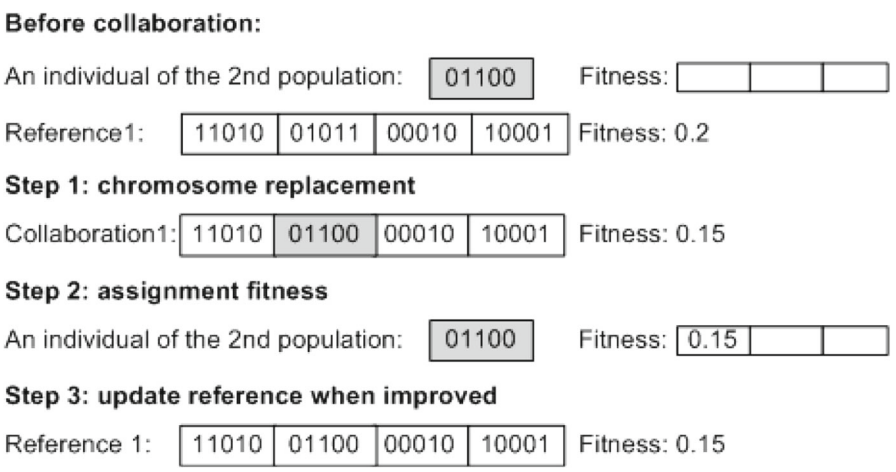

Fig. 5 An example of a collaboration formed between an individual of the second population and the first reference of the archive. In the first step, we replace the second chromosome segment of the reference 1 with the chromosome of the individual. In the second step, the collaboration fitness is assigned to the individual as its first fitness. In step 3, we update reference 1 if the collaboration fitness is better than its previous value, where the smaller the value the better the fitness in this case

et al. (2012) independently evolve teams under a variable length representation with fitness sharing as applied to Pareto archiving or goal function. Moreover, fitness is only ever associated with a team of individuals, thus side stepping biases created by estimating fitness at the level of the individual. Wu and Banzhaf (2011) adopt a multilevel selection scheme with fitness potentially being associated with individuals or groups depending on the 'level' of selection.

In this research, we would like to complement the existing research on multi-fitness measurement. In this section, we describe three sorting strategies for multi-fitness measurement conducted in this study. Selection and replacement will be implemented based on the measurement. We now define some notations that will be used for describing the sorting algorithms. After evaluating all individuals of population P, the performance of all the collaborations is saved in an $M \times N$ payoff matrixes G, where N is the size of population P and M is the size of archive R. Matrix entry $G_{i,\mathrm{j}}$ is the payoff received by individual i of the population when it collaborates with reference j of the archive.

3.2.1 Greedy sorting

Greedy sorting is the simplest and the most straightforward way to perform multi-fitness measurement. Each individual x is ranked in front of individual y if the best fitness value of x is better than the best fitness value of y(i.e. $best(G_{x,1}, G_{x,2}, \ldots, G_{x,M})$ is better than $best(G_{y,1}, G_{y,2}, \ldots, G_{y,M})$, where the best could be max function in maximum optimization problems or min function in minimum optimization problems). If both individuals have the same best fitness value, then we compare the second best one, and so on.

3.2.2 Non-dominated sorting

We implemented a fast non-dominated sorting, developed by Deb et al. (2000). In this algorithm we measure the success of collaborations based on Pareto dominance in cooperative coevolution (Bucci and Pollack 2005):

- Individual x *Pareto dominates* individual y relative to the set of references in archive R, denoted as $x \succ y$, iff $\forall w \in R : G_{x,w} \geq G_{y,w}$ and $\exists u \in R : G_{x,u} > G_{y,u}$.
- Individuals x and y are *mutually non-dominating*, denoted as $x \diamond y$, iff $\exists w, u \in R$: $G_{x,w} > G_{y,w}$ and $G_{x,u} < G_{y,u}$.
- The *Pareto layer*, denoted as F^i, identifies the Pareto domination level of individuals. All the individuals in layer F^i are dominated by all the individuals in layer F^{i-1}, but Pareto dominate all the individuals in layer F^{i+1}. F^0, called the *Pareto front*, is the subset of the best non-dominated individuals in population P.

Crowding distance (Deb et al. 2000) is a common metric for sorting individuals within the same Pareto layer, where solutions with higher crowding distance are better since they contribute to a more uniform distribution along the non-dominated front. The goal of our algorithm, however, is not to find a single solution to achieve multi-objective optimization, but to find a combination of chromosomal segments to achieve the best performance for a single objective. So we employ the greedy strategy, described above, to rank individuals within the same Pareto layer.

Non-dominated sorting (Deb et al. 2000) is one of the most prevalent strategies employed in MOEAs. The multi-fitness measurement using non-dominated sorting has also been introduced in coevolution, and has been demonstrated to enhance performance (Ficici and Pollack 2001; Noble and Watson 2001; Jong and Pollack 2004; Bucci and Pollack 2005; Iorio and Li 2004).

3.2.3 Even-distributed sorting

After analyzing the mechanism of non-dominated sorting, it is not difficult to observe that not all individuals in dominated layers, especially in layer F^1, are really bad. Some of them might have high performance but just are dominated by one of the individuals in the Pareto front. For example, note that individual x_4 is dominated by x_6 in Fig. 7c. Similarly, not all the individuals in the Pareto front, F^0, are good. Some of them might be in the Pareto front only because they have slightly better performance than another individual relative to one of the references in the archive; but the dominated fitness value is actually the worst one of that individual (see individual x_3 in Fig. 7c). In competitive coevolution and multi-objective evolution, this evaluation mechanism is applicable to preserve an individual that could be poor to achieve most objectives but good to achieve one objective which other individuals cannot. However, in cooperative coevolution, we are not interested in finding individuals that generally have good collaborations with all references in the archive, but in those that have exceptional collaborations with one or a few references. In our implementation, all reproducing individuals are selected from the top n, so a potentially good individual in dominated layers could be eliminated when the size of the Pareto front is bigger than n.

```
Even-distributed population P'= { }
for each reference i do begin
   sortingPopulation[i] = sorting population P according to the ith reference
  end for
while size( P')<size( P )
  for each reference i do begin
     q = the best individual in sortingPopulation [i]
     P'= P'<< q
     for each soringPopulation j do begin
        sortingPopulation [j]= sortingPopulation [j] \q
       end for
     end for
   end while
```

Fig. 6 Pseudo code of even-distributed sorting

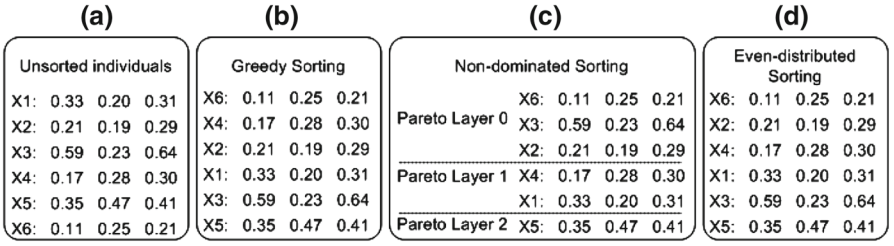

Fig. 7 An example of individuals sorted by three sorting algorithms

For the purpose of preserving those potentially good individuals of dominated layers, we propose a new sorting strategy, called *even-distributed sorting*. This algorithm begins by performing M different sorts of population P, with the *ith* sort based on the fitness of each individual in collaboration with the *ith* reference. (i.e. $sort(G_{1,i}, G_{2,i}, \ldots, G_{N,i})$ where $1 \leq i \leq M$). Next, individuals are ranked according to the M sorts, where the best individual is selected from each sorting list, in turn, and the selected individual is removed from all sorting lists. The pseudo code for this algorithm is shown in Fig. 6. The order of the M references in the archive only affects the order of individuals evaluated in the same round, where selecting the best individual from the first sorting population to the last sorting population is regarded as one round evaluation. Since M is normally much smaller than the individual size of P, the affection can be ignored on the whole.

Figure 7 illustrates an example of individuals with multi-fitness values sorted by the three different sorting algorithms. Note that the best and the worst individuals are identical for all three sorting algorithms, but the others are different. Obviously, the rank mechanism of the non-dominated sorting is quite different from the other two. While the rank mechanism of the even-distributed sorting can be regarded as a variation of the greedy sorting, in which the indices of the fitness values will be taken into consideration when selecting the best fitness value. However, the essential difference

of the two sorting strategies could cause entirely different evolutionary behavior. The greedy sorting only focuses on which individual has the best single fitness, but does not care which reference contributes the fitness. In contrast, even-distributed sorting focuses on the references in turn, selecting the individual that achieves the best fitness when collaborating with the reference. By considering each of the references in the archive, even-distributed sorting avoids any strong bias induced by, for example, one reference that most individuals collaborate with well. In such reference-biased cases, good individuals can be favored by a greedy strategy, but other individuals who are potentially good for collaborating with other references could be deprived of reproductive opportunities. Consequently, the CCEA-RS can end up with an archive in which only one of the references plays a significant role in fitness assessment and the other references are ignored, even through several are normally predefined in the algorithm. The even-distributed strategy, however, is able to balance this situation by assigning equal preference to each reference.

4 Test problems

Since they have useful properties such as linearity, separability, multimodality, and complex fitness-landscape topography, function optimization problems are often used to analyze new CCEA approaches. We choose two separable functions, Rastrigin and Schwefel, and four nonseparable functions, Trid, Rosenbrock, Booth and Powell, in this test suite. Each function has its own characteristic fitness landscape. For each, the global minimum is the target/optimal value.

The first test function, Rastrigin and is expressed by the following equation:

$$f(\vec{x}) = nA + \sum_{i=1}^{n} x_i^2 - A\cos(2\pi x_i),$$

where $n = 20$, and $A = 3$, and $-5.12 \leq x_i \leq 5.12$. The Rastrigin is a typical nonlinear and multimodal function; it has many regularly distributed local minima. The external variable A is used for controlling the multimodal amplitude and frequency. The only global minimum of zero is at the point *(0, 0,..., 0)*.

The second test function is Schwefel, as defined by:

$$f(\vec{x}) = 418.9829n + \sum_{i=1}^{n} x_i \sin\left(\sqrt{|x_i|}\right),$$

where $n = 10$ and $-500.0 \leq x_i \leq 500.0$. The Schwefel is also a nonlinear and multimodal function; its landscape consists of a great number of peaks and basins. The global minimum of zero is close to the corners of the domain at the point *(−420.9687, −420.9687,...)*. An interesting characteristic of this function is that the next best minimum is far from the global one: the landscape is very deceptive.

The third is Neumaier no.3, also called theTrid function:

$$f(\vec{x}) = \sum_{i=1}^{n} (x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1},$$

where $n = 10$ and $-n^2 \leq x_i \leq n^2$. Unlike the first two functions in the test suite, the Trid function has no local minimum, only the global one, which, for the 10-dimensional problem, is -210 at *(10, 18, 24, 28, 30, 30, 28, 24, 18, 10)*. A primary characteristic of this function is strong coupling between the variables, which causes difficulties for genetic algorithms (Deep 2007).

Rosenbrock, the fourth function, is highly non-linear and defined as:

$$f(\vec{x}) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right],$$

where $n = 20$ and $-2.048 \leq x_i \leq 2.048$. The landscape of the Rosenbrock contains a very narrow parabolic valley. It is trivial to find the valley, but difficult to locate the minimum within it. A two dimensional Rosenbrock is unimodal, but in higher dimensions, it is not. The global minimum of zero is at the point *(1, 1,..., 1)*.

The fifth function is Booth, defined as:

$$f(\vec{x}) = \sum_{i=1}^{n-1} \left[(x_i + 2x_{i+1} - 7)^2 + (2x_i + x_{i+1} - 5)^2 \right],$$

where n*=10* and $-100 \leq x_i \leq 100$. The landscape surface of Booth is similar to, but flatter than, that of Trid, and it has several local minima, unlike Trid. Booth's global minimum of zero is at *(1, 3,..., 1, 3)*.

The equation of the sixth function, Powell, is:

$$f(\vec{x}) = \sum_{i=1}^{n/4} \Big[(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4 \Big],$$

where $n = 12$ and $-4 \leq x_i \leq 4$. The global minimum of zero is at the point *(3, −1, 0, 1,..., 3, −1, 0, 1)*. Powell also has strong coupling between the variables.

5 Experiments

In this section, we first investigate the archive size of CCEA-RS and then compare the three sorting algorithms on the test suite. Next, we evaluate the CCEA-RS by comparing its performance with that of two other popular CCEAs.

In all experiments, we use a population size of 100, two-point crossover, and bit-flipping mutation with rate 0.05 per bit. Each variable x_i of these functions is encoded by an m-bit binary string and evolved in an independent population, where $m = 16$in our experiments. We transform the binary string into a floating-point value via the following standard procedure:

$$x_i = \frac{d_i}{2^m}(x_{i,\max} - x_{i,\min}) + x_{i,\min},$$

where d_i is the integer value of the binary string of x_i, where $x_{i,max}$ and $x_{i,min}$ are the upper and lower bounds (respectively) of x_i.

One of the main purposes of these experiments is to analyze the efficiency achieved by different sorting algorithms and collaboration models. Thus, we employ the same selection and replacement mechanisms in all versions of our CCEAs in order to eliminate the effect caused by this difference. For each algorithm, the 60 worst individuals are replaced with offspring produced by the 30 best individuals in each generation. The experimental results shown in our diagrams were generated from an average of 50 runs each. For the results, statistical significance has been verified using Student's two-tailed t test, assuming unequal variances at 95% confidence.

5.1 Archive size

The archive size is a predefined parameter in CCEA-RS; it indicates the number of references in the evaluation model. In the first group of experiments, we use even-distribution sorting and evaluate the performance of CCEA-RS with archive sizes ranging from 1 to 10.

A separable problem, Schwefel, and a nonseparable problem, Trid, are used. Figure 8 illustrates boxplots and statistical significance of the performance with 10 different archive sizes over 50 independent runs. The boxplots show six statistics: maximum, minimum, median, mean, the first quartile and the third quartile. We have cut some plots and zoomed in on a different fitness interval for each plot in order to highlight the differences of statistics. For the sake of assessing the statistical significance of the performance between each pair of archive sizes, a grid map for each problem is plotted on the right side of Fig. 8. The color in each grid cell indicates the statistical significance of the performance between the two corresponding archive sizes: (1) gray, there was no statistical significance between the performance of the different archive sizes; (2) black, the performance difference was statistically significant, and the bigger archive size performed better; (3) white, the performance difference was statistically significant, and the smaller archive size performed better.

For the separable problem, Schwefel, the size of the archive appears less important. The quartiles, median and minimum of the performance for all difference archive sizes are at almost the same height. Although the differences of the mean values appear a bit bigger, the corresponding plot of the statistical significance for Schwefel shown on the right side indicates that none of these differences is statistically significant. For the nonseparable problem, Trid, the bigger archive sizes achieve better statistical values, as shown on the bottom left. And the bigger the size difference of a pairwise archive,

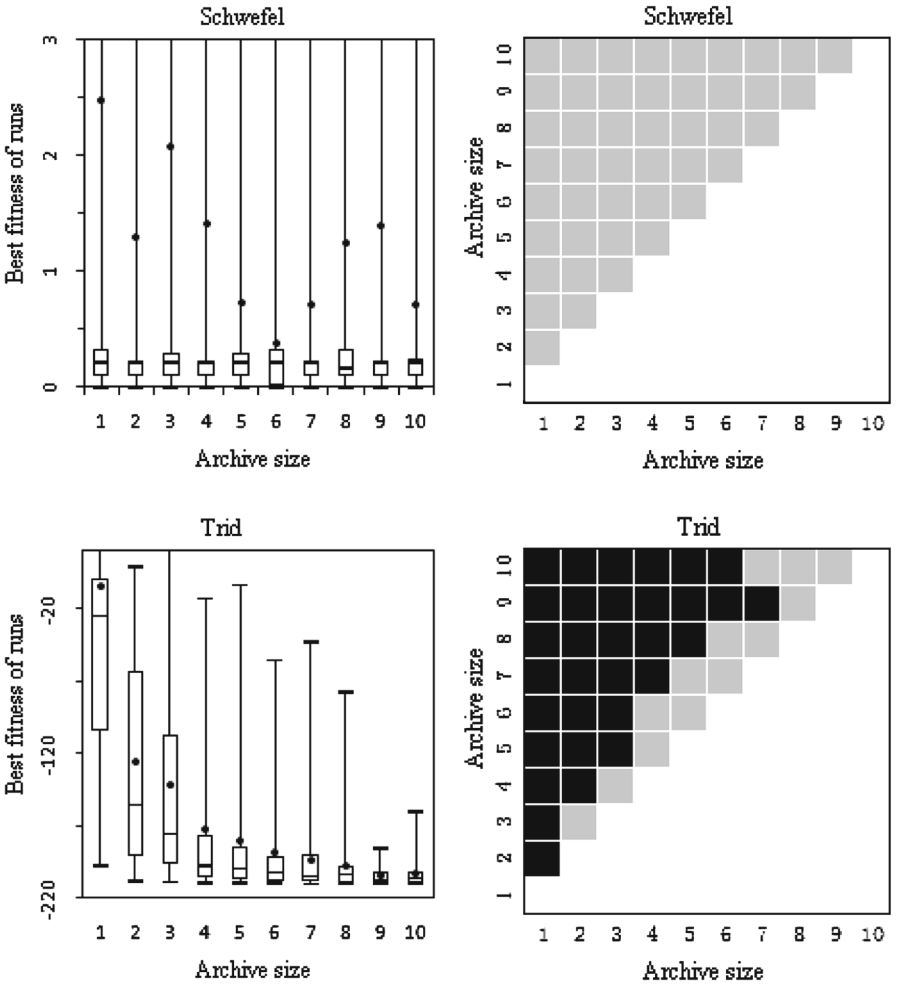

Fig. 8 *Left* the boxplots of the statistical performance of CCEA-RS with 10 different sizes of archive on two functions (and 50 runs each). The two tips of the *whiskers* are the best and the worst values respectively; the *boxes* show the inter-quartile ranges; in each box, the *line* denotes the median, and the *dot* is the mean. *Right* the statistical significance of the performance difference between each pair of archive sizes. *Black* indicates that the bigger archive size performed better; *gray* indicates no significant difference

the more statistically significant the difference of the performance. The gray squares shown along the diagonal of the right plot also signify that there is no distinguishable performance difference when the archive sizes vary only slightly.

On the two right plots, note that no grid cells are colored white. Thus, when there is statistical significance, it always favors the larger archive. Unfortunately, in CCEA-RS, each individual of one population cooperates with every reference in the archive, so the bigger the archive, the more time-consuming the algorithm. After taking several factors, including performance ratio and time-consumption, into consideration, we chose an intermediate archive size of 5 in the remaining experiments.

Table 1 Comparison of average performance of even-distributed sorting with the other two different sorting strategies, 50 runs for each result

Function	Greedy			Non-dominated			Even-distributed
	Average fitness	*P* value	Friedman	Average fitness	*P* value	Friedman	Average fitness
Rastrigin	0.1779	0.6070	2.88	0.1180	0.8007	2	0.1377
Schwefel	0.9109	0.8338	0.32	1.6263	0.4159	**6.48**	0.7278
Trid	−140.732	**0.0022**	**18**	−146.613	**0.0229**	**8**	−180.316
Rosenbrock	19.4118	**1.165E−04**	**11.52**	18.0828	**8.811E−06**	**25.92**	8.3020
Booth	41.466	**2.797E−04**	**25.92**	36.643	**0.0091**	**20.48**	25.968
Powell	1.5064	**1.06E−04**	**23.12**	0.4514	0.0424	**8**	0.2508

The *P* values evaluate the statistical significance using student's two-tailed *t* test assuming unequal variances at 95% confidence. Friedman values evaluate the statistical significance using Friedman test. The upper critical value of Friedman test is 6.04 for group number = 2, subject number = 50 and alpha = 0.05
P value is smaller than 0.05, marked in bold, indicates the performance difference is statistical significance between even-distributed sorting and the evaluated method

5.2 Sorting strategies for multi-fitness measurement

In the second group of experiments, we investigate how the three multi-fitness measurements affect the performance of CCEA-RS on the entire test suite. In addition to evaluating the comparison results using Student's two-tailed *t* test, Friedman test is also applied in this section. Our purpose is to assess the statistical significance of average performance between even-distributed sorting and each of the other two sorting algorithm, greedy sorting and non-dominated sorting, so the performance difference between greedy sorting and non-dominated sorting is not stated here.

By optimizing both separable and nonseparable problems, the following runs show the differing abilities of the CCEA-RS sorting strategies to handle epistasis. The results give a strong indication of the advantages of even-distributed sorting.

Table 1 illustrates the average performance of CCEA-RS using three different sorting strategies over 500 generations for 50 runs each. The *P* values shown in this table are from a two-tailed Student's *t* test that compares even-distributed sorting with the corresponding sorting strategy. When we evaluate the three sorting algorithms merely according to the average best fitness achieved, obviously, the greedy sorting is not a good strategy to measure multi-fitness of individuals in most of cases. The non-dominated sorting performed better than the even-distributed sorting for the Rastrigin problem. We at first evaluate the performance difference by using Student's *t* test. The statistical difference between even-distributed sorting and the other two is not significant for the first two problems. However, for the four nonseparable problems the even-distributed sorting achieved significantly better performance than the other two, in which only the *P* value 0.0424 for the Powell problem is a bit higher than the threshold value 0.025 to reach a 95% confidence level. Then we take a look at the evaluation from Friedman test. Friedman test gives an even stronger indication of the advantages of even-distributed sorting than Student's test. The upper critical value of

Friedman test is 6.04 for group number is 2, subject number is 50 at 95% confidence. A higher value than the upper critical bound shows significant of the difference. So besides of the four nonseparable problems, the performance of even-distributed sorting is also significantly better than non-dominated sorting for Schwefel. In general, even-distributed sorting works best for both separable and nonseparable problems.

5.3 Collaboration models

It is still not clear if the new collaboration model is superior to other models for CCEAs, so this section compares the CCEA-RS to two other versions of CCEAs, both employing a 1 + N collaboration model, which is a popular choice for optimizing nonseparable functions (Potter 1997; Wiegand et al. 2001). In these experiments, two methods (one greedy and one not) are used for selecting the N collaborators of an individual:

CCEA-1: choose the best N individuals from each of the collaborative populations, defined according to the fitness evaluation in the previous generation, and assign the fitness of the best collaboration to the individual. This is a greedy strategy.

CCEA-2: choose the best individual plus N − 1 random individuals from each of the collaborative populations, and again, assign an individual's fitness as that of its best collaboration. This is a much less greedy strategy.

In order to compare the algorithms as fairly as possible, N is set to 5 to enable the same number of collaborations in the CCEA-RS as in the other CCEAs. The average performance of CCEA-1, CCEA-2 and CCEA-RS for the six functions in the test suite is presented in Fig. 9. Each number shown in the brackets is the *P* value from a two-tailed Student's *t* test comparing the final best results of CCEA-RS with that of the other CCEA over 50 runs. CCEA-RS employs even-distributed sorting. All functions were optimized over 500 generations in each independent run.

Both the Rastrigin and Schwefel are separable problems, so, as expected the CCEA with the greedy selection strategy (CCEA-1) converged faster than CCEA-2. An empirical analysis conducted by Wiegand et al. has already reported the same conclusion: a greedy strategy to select collaborators is warranted if a problem is separable (Wiegand et al. 2001). Notice that the less greedy selection strategy also includes a greedy implementation, so CCEA-2 gradually achieves similar results to the CCEA-1 in the end. By using reference sharing collaboration, CCEA-RS converges as fast as CCEA-1. Although the convergence curves for the Schwefel problem in Fig. 9 show that the CCEA-RS achieves better performance than the other two, the *P* values in the brackets clarify that there is no statistically significant difference among the final best results of these algorithms for both separable problems. Thus, the result only indicates that the CCEA-RS is able to achieve the same performance as the CCEA-1 for separable problems.

We now shift attention to the four nonseparable problems, in which the interaction among variables is much stronger. For nonseparable problems, researchers have argued that CCEAs with less greedy collaboration strategies perform better than their greedy counterparts (Potter and De Jong 1994; Wiegand et al. 2001). This point is demonstrated again in our experiments (see the graphs for the nonseparable problems

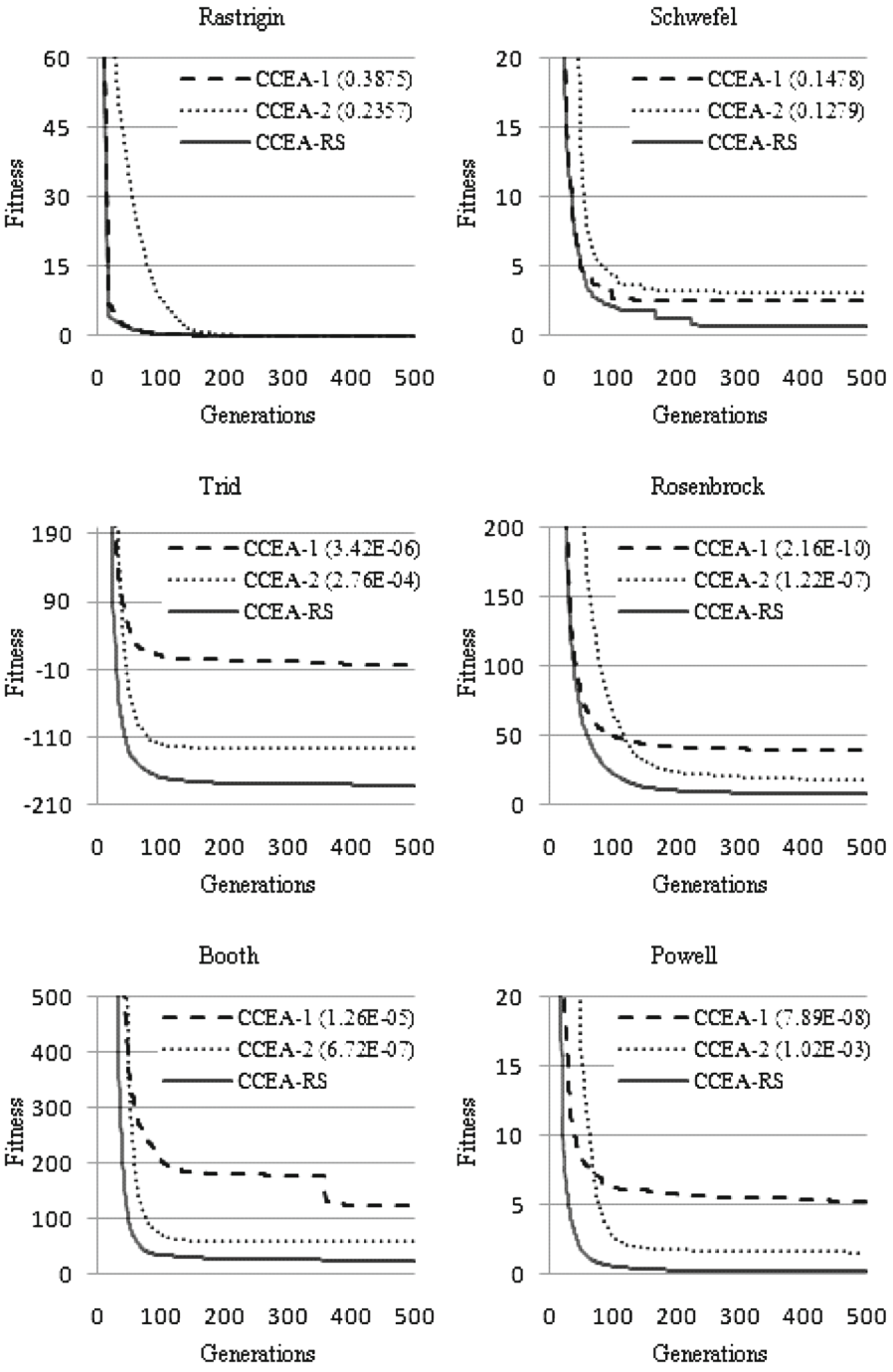

Fig. 9 Average convergence curves of three different CCEAs. The *numbers* in the brackets are *P* values generated from two-tailed Student's *t* tests comparing final best results of the CCEA-RS with those of the other two CCEAs, based on 50 runs

in Fig. 9), where the CCEA-2 performs better than the CCEA-1 on all four. The performance difference between CCEA-1 and CCEA-2 is especially significant for the Trid problem, because the Trid has very strong coupling (i.e. high degree of epistatic) among the variables. The convergence curves for these problems demonstrate that the CCEA-RS clearly outperformed the other two algorithms, both in terms of convergence speed and maximum attained fitness, as clearly supported by the *P* values.

The above studies clearly promote the CCEA-RS as a general-purpose algorithm for both separable and nonseparable problems.

6 Analysis

As shown above (and in many other studies), the separabilities of problems affect the performance of CCEAs. Wiegand et al. (2001) imply that different degrees of separability dictate different collaboration strategies. However, our experiments show that reference-sharing collaboration tackles many problems across the separability spectrum, probably because it has the capability to rebuild linkages between variables.

To further investigate linkage, its effects upon CCEAs, and the ability of CCEA-RS to handle it, we implemented three types of decomposition bias: full, half and bipartite:

- *Full decomposition* A function of n variables partitioned into n components, where each component represents a function variable of this function. Here, all linkages are broken.
- *Half decomposition* A function of n variables is decomposed into $n/2$ components of 2 variables each. Here, half the linkages are broken.
- *Bipartite decomposition* A function of n variables is decomposed into 2 components, each composed of $n/2$ variables. Here, only one linkage is broken.

By breaking the linkage between variables into different levels, we should be able to observe how CCEAs can rebuild linkages, or, in other words, how the performance of CCEAs is affected by the broken linkage. Because the linkage between variables for nonseparable problems is stronger than that for separable problems, two nonseparable problems, Rosenbrock and Booth, are chosen for these studies. We analyze three collaboration models as the basis for three different CCEAs:

- CCEA-G: a CCEA with a greedy 1 + 1 collaboration model. This is also a variation of CCEA-1 where N is equal to 1.
- CCEA-LG: a CCEA with a 1 + N collaboration model, which is less greedy than CCEA-G. This is the same as CCEA-2 with N = 2.
- CCEA-RS: a CCEA with reference-sharing collaboration and even-distributed sorting; the archive size is 3.

The three collaboration methods carry out different numbers of evaluation for each individual per generation: one for CCEA-G, two for CCEA-LG, and three for CCEA-RS. For fairness of comparison, all CCEAs end their runs after 100,000 function evaluations. Here, the focus is not on which CCEA performs best, but on the performance difference of the decomposition biases for the same CCEA on the same problem.

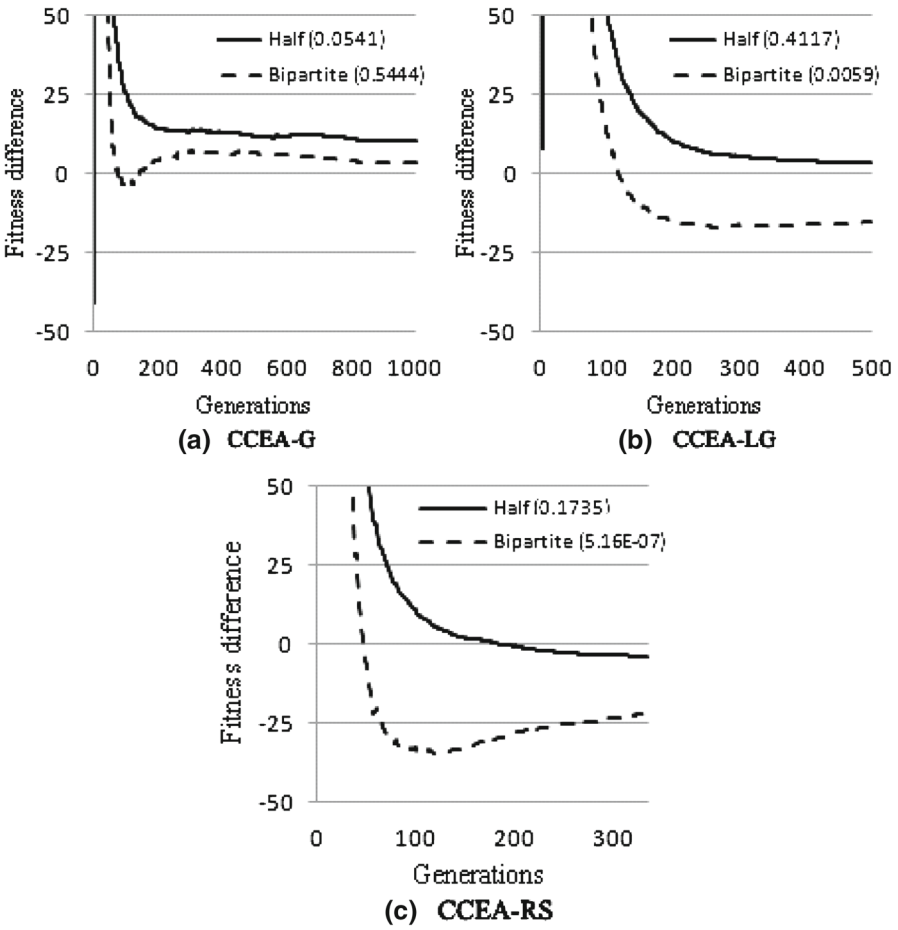

Fig. 10 Performance differences of using half and bipartite decomposition versus using the full decomposition for Rosenbrock. The *numbers* shown in the brackets are the *P* values from two-tailed Student's *t* tests of comparing the performance of using the full decomposition with that of using the corresponding decomposition method

Figures 10 and 11 illustrate the average fitness differences of using the three decomposition biases over 50 runs for the two problems, where they axis represents the difference between the performance of using half (or bipartite) decomposition and that of using full decomposition. Thus, y = 0 denotes situations where the given decomposition gives the same result as does full decomposition, and curves above (below) the y axis indicate better (worse) performance than the corresponding full-decomposition scenario. To make the diagrams easier to read, not all the fitness differences during generation are shown. We zoomed in on a difference fitness interval where the algorithms have achieved relatively steady convergences.

Clearly, the CCEAs with bipartite decomposition perform worst in all cases, but Fig. 11a is unique in that both variants out-perform full decomposition. A primary reason is that the 1 + 1 collaboration model has poorer capacity to rebuild linkages

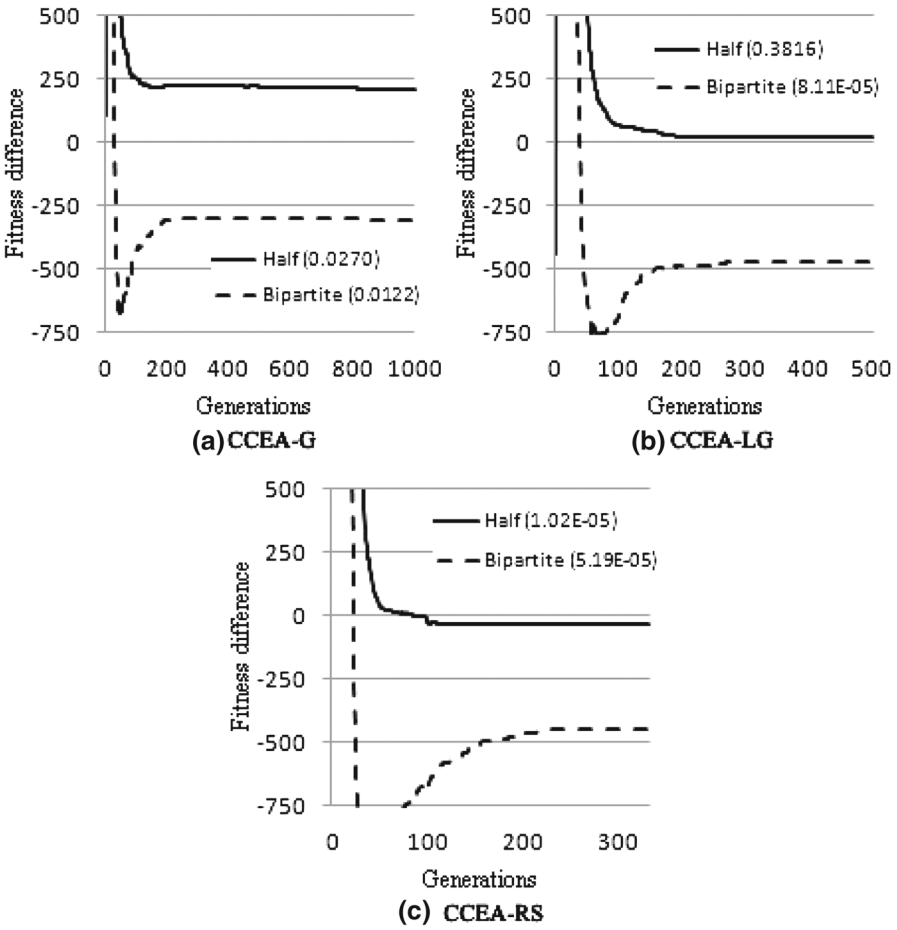

Fig. 11 Performance differences of using half and bipartite decomposition from using the full decomposition for Booth The *numbers* shown in the brackets are the *P* values from Student's two-tails *t* test of comparing the performance of using the full decomposition with that of using the corresponding decomposition methods

than do other models. When the linkage between variables is very strong, reducing the number of broken linkages is a feasible way to improve the performance of CCEA-G. However, a too coarse decomposition degrades the search efficiency of CCEAs; that could explain why the decomposition with an intermediate size, that is half decomposition, achieved better results than the bipartite decomposition, and under all the conditions. For the CCEA-LG, the half decomposition was superior to the full decomposition, which indicates that the capacity of rebuilding the broken linkage for full decomposition is worse than that for half decomposition in CCEA-LG.

Although their performance difference is not statistically significant for a 95% confidence level, these results are typical for our experiments. The most impressive results are from the CCEA-RS. With full decomposition, CCEA-RS achieved the best results among the three decomposition biases for both problems. Especially, the

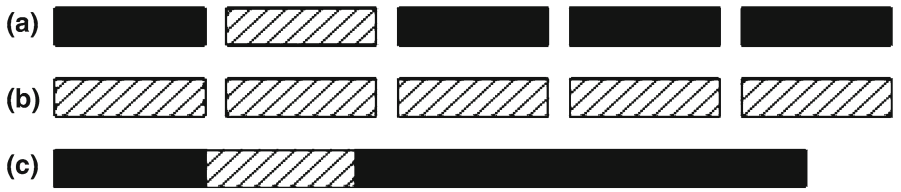

Fig. 12 Methods to form a solution in different collaboration models. The *black blocks* are unevolved components when forming a solution for evaluation. The *blocks* with *diagonal lines* are components undergoing evolution. Both (**a**, **b**) use split-join techniques with **a** 1+1, 1+N collaboration models and **b** N+N and evolutionary collaboration models. **c** The segmentation and replacement technique used in CCEA-RS

performance differences between CCEA-RS with full decomposition and the other two are statistically significant for the Booth problem. These results imply that the CCEA-RS is less sensitive to resolution when decomposing a problem. This indicates that it has good capacity to rebuild broken linkages.

The most important difference between reference sharing collaboration and other traditional collaboration methods introduced in Sect. 2.1 is that they use two different techniques when forming a solution for evaluation, which are presented graphically in Fig. 12. The traditional collaboration methods assemble a solution through split-and-join techniques, where the decomposed components of the solution are filled in with splitting individuals from different populations. These individuals are joined together temporarily to form a solution in each generation for evaluation. In such process, the linkage status between components from last generation is not kept when implementing next evaluation, therefore cannot guarantee to improve in next generation. In Fig. 12a, b we use gaps to indicate the split-and-join process to form solutions happening in each generation. Conversely, reference sharing collaboration forms solutions at the beginning and then gradually modifies them via piecewise replacement. There are no gaps between components in Fig. 12c, because a solution is always treated as a whole. A chromosome segment of a solution is replaced only when the replacement improves the solution. Linkage status is always kept from last generation to the next, therefore, improvement can be guaranteed. Intuitively, the latter approach that gradually improves solutions should handle linkage more effectively than classic split-and-join methods. Although reference sharing collaboration still cannot guarantee an ideal mending to achieve the optimal solution of a problem, this can be improved by increasing the number of candidate solutions in the archive, as demonstrated in Sect. 5.1.

7 Conclusion and future work

In the literature (Potter 1997; Wiegand et al. 2002; Iorio and Li 2004), a high degree of interaction between components has been known to destroy the decomposability of a problem. This is particularly difficult issue for CCEAs, since decomposition is central to these approaches. Decomposing a problem into fewer numbers of components, of course, can preserve more linkage between components, but this degrades the CCEAs explorative capabilities. Of course, explorative operators, crossover and mutation, operate on the entire chromosome of a component, regardless of search-

space size space, but there is no doubt that the explorative ability of CCEAs with fine decomposition is higher than that of with coarse decomposition. A fatal disadvantage, however, is that fine decomposition also introduces more broken linkage between components. If they fail to rebuild the broken linkages, CCEAs with fine decomposition often perform worse than those with coarse decomposition.

Our new algorithm, the CCEA-RS, is able to exert the explorative ability of the coevolutionary model and meanwhile reconstruct the linkage through the reference sharing collaborations. The capability of the linkage reconstruction can be adjusted by the archive size. Of course, larger archive size reflects higher capabilities, but it is more time-consuming for evaluation. By using even-distribution sorting, every reference receives equal collaboration opportunities. In this way, the algorithm is able to utilize all the references of the archive most efficiently. Moreover, our experimental results show that no prior knowledge of the separabilities of a problem is needed; CCEA-RS performs fairly well compared to CCEAs with greedy and less greedy collaboration strategies, regardless of the degree of problem separability. Our preliminary results indicate that new collaboration methods can greatly improve CCEAs.

However, we are also aware of some limitations of this research. Firstly, we need to demonstrate our method on more suites of benchmark problems, especially for large scale optimization (Li et al. 2013). Secondly, we mainly discuss the strengths of the proposed algorithm, the CCEA-RS. Some additional evaluations are needed to assess its capabilities. Last but not least, the applicability of the CCEA-RS to non-functional problems is not explored. We plan to carry out future research on this.

In reference sharing collaboration, the archive references not only represent solutions, but also guide the CCEA-RS to evolve the components of the solutions. After problems decomposition, each component evolutionary system is blind to the entire search landscape. Reference sharing collaboration regards the decomposed components as a whole and supervises each component in evolving to contribute to the whole solution. Each reference in the archive works like a guide. However, there could exist that two or more references have similar chromosome representation. These guides, thus, could be exploring the same area of the whole search space, therefore debase the efficiency of the algorithm. Future work includes the consideration of the reference diversity in the archive and tests on decomposition in dynamic tasks.

Acknowledgements The author would like to thank Keith Downing, Professor at the Department of Computer and Information Science at NTNU, for his valuable comments.

References

Angeline, P.J., Pollack, J.B.: Competitive environments evolve better solutions for complex tasks. In: ICGA, pp. 264–270 (1993)

Bucci, A., Pollack, J.B.: Focusing versus intransitivity geometrical aspects of co-evolution. In: Genetic and Evolutionary Computation—GECCO 2003, pp. 250–261. Springer (2003)

Bucci, A., Pollack, J.B.: On identifying global optima in cooperative coevolution. In: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, pp. 539–544. ACM (2005)

Casillas, J., Cordón, O., Herrera, F., Merelo, J.: Cooperative coevolution for learning fuzzy rule-based systems. In: Artificial Evolution, pp. 311–322. Springer (2002)
De Jong, E.D., Pollack, J.B.: Ideal evaluation from coevolution. Evol. Comput. **12**(2), 159–192 (2004)
Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. Lect. Notes Comput. Sci. **1917**, 849–858 (2000)
Deep, K.: A new hybrid self organizing migrating genetic algorithm for function optimization. In: IEEE Congress on Evolutionary Computation, 2007. CEC 2007, pp. 2796-2803. IEEE (2007)
DeJong, K.A., Spears, W.M.: Learning concept classification rules using genetic algorithms. In: The Twelfth International Joint Conference on Artificial Intelligence (1991)
Doucette, J.A., Lichodzijewski, P., Heywood, M.I.: Hierarchical task decomposition through symbiosis in reinforcement learning. In: Proceedings of the Fourteenth International Conference on Genetic and Evolutionary Computation Conference, pp. 97–104. ACM (2012)
Doucette, J.A., McIntyre, A.R., Lichodzijewski, P., Heywood, M.I.: Symbiotic coevolutionary genetic programming: a benchmarking study under large attribute spaces. Genet. Program. Evolvable Mach. **13**(1), 71–101 (2012)
Ficici, S.G., Pollack, J.B.: Pareto optimality in coevolutionary learning. In: Kelemen, J., Sosík, P. (eds.) Advances in Artificial Life, pp. 316–325. Berlin, Heidelberg Springer (2001)
Fonseca, C.M., Fleming, P.J.: Genetic algorithms for multiobjective optimization: formulation discussion and generalization. In: ICGA, pp. 416–423 (1993)
García-Pedrajas, N., Hervás-Martínez, C., Muñoz-Pérez, J.: COVNET: a cooperative coevolutionary model for evolving artificial neural networks. IEEE Trans. Neural Netw. **14**(3), 575–596 (2003)
Giordana, A., Saitta, L., Zini, F.: Learning disjunctive concepts by means of genetic algorithms. In: ICML, pp. 96–104 (1994)
Gomez, F.J.: Robust non-linear control through neuroevolution. Ph.D. Thesis, Computer Science Department, University of Texas at Austin (2003)
Hillis, W.D.: Co-evolving parasites improve simulated evolution as an optimization procedure. Phys. D Nonlinear Phenom. **42**(1), 228–234 (1990)
Holland, J.H.: Escaping brittleness: the possibilities of general purpose learning algorithms applied to parallel rule-based systems. In: Michalski, R., Carbonell, J., Mitchell, T. (eds.) Machine Learning: An Artificial Intelligence Approach, vol. 2, pp. 593–623. Morgan Kaufmann (1986)
Horn, J., Nafpliotis, N., Goldberg, D.E.: A niched Pareto genetic algorithm for multiobjective optimization. In: Proceedings of the First IEEE Conference on Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence, vol. 81, pp. 82–87 (1994)
Hoverstad, B.A.: Revisiting the personal satellite assistant: neuroevolution with a modified enforced subpopulations algorithm. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, pp. 2021–2028. ACM (2007)
Husbands, P., Mill, F.: Simulated co-evolution as the mechanism for emergent planning and scheduling. In: ICGA, pp. 264–270 (1991)
Iorio, A.W., Li, X.: A cooperative coevolutionary multiobjective algorithm using non-dominated sorting. In: Genetic and Evolutionary Computation–GECCO 2004, pp. 537-548. Springer (2004)
Kim, J.Y., Kim, Y., Kim, Y.K.: An endosymbiotic evolutionary algorithm for optimization. Appl. Intell. **15**(2), 117–130 (2001)
Li, X., Tang, K., Omidvar, M.N., Yang, Z., Qin, K.: Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization, Technical Report, Evolutionary Computation and Machine Learning Group, RMIT University, Australia (2013)
Li, Y., Jiao, L., Shang, R., Stolkin, R.: Dynamic-context cooperative quantum-behaved particle swarm optimization based on multilevel thresholding applied to medical image segmentation. Inf. Sci. **294**, 408–422 (2015)
Moriarty, D.E., Miikkulainen, R.: Forming neural networks through efficient and adaptive coevolution. Evol. Comput. **5**(4), 373–399 (1997)
Nash, J.: Non-cooperative games. Ann. Math. **54**, 286–295 (1951)
Noble, J., Watson, R.A.: Pareto coevolution: using performance against coevolved opponents in a game as dimensions for Pareto selection. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001) (2001)
Omidvar, M.N., Li, X., Mei, Y., Yao, X.: Cooperative co-evolution with differential grouping for large scale optimization. IEEE Trans. Evol. Comput. **18**(3), 378–393 (2014)

Panait, L., Luke, S.: Time-dependent collaboration schemes for cooperative coevolutionary algorithms. In: Proceedings of the 2005 AAAI Fall Symposium on Coevolutionary and Coadaptive Systems (2005)
Panait, L., Luke, S., Harrison, J.F.: Archive-based cooperative coevolutionary algorithms. In: Proceedings of the 8th annual Conference on Genetic and Evolutionary Computation, pp. 345–352. ACM (2006)
Paredis, J.: The symbiotic evolution of solutions and their representations. In: Proceedings of the 6th International Conference on Genetic Algorithms, pp. 359–365. Morgan Kaufmann Publishers Inc. (1995)
Parsopoulos, K.E.: Parallel cooperative micro-particle swarm optimization: a master-slave model. Appl. Soft Comput. **12**(11), 3552–3579 (2012)
Pena-Reyes, C.A., Sipper, M.: Applying Fuzzy CoCo to breast cancer diagnosis. In: Proceedings of the 2000 Congress on Evolutionary Computation, 2000, pp. 1168-1175. IEEE (2000)
Popovici, E., De Jong, K.: A dynamical systems analysis of collaboration methods in cooperative coevolution. In: AAAI Fall Symposium on Coevolutionary and Coadaptive Systems (2005)
Popovici, E., De Jong, K.: The effects of interaction frequency on the optimization performance of cooperative coevolution. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp. 353–360. ACM (2006)
Potter, M.A.: The design and analysis of a computational model of cooperative coevolution. Ph.D. Thesis, George Mason University (1997)
Potter, M.A., De Jong, K.A.: Cooperative coevolution: an architecture for evolving coadapted subcomponents. Evol. Comput. **8**(1), 1–29 (2000)
Potter, M.A., De Jong, K.A.: A cooperative coevolutionary approach to function optimization. In: Parallel Problem Solving from Nature—PPSN III, pp. 249–257. Springer (1994)
Ray, T., Yao, X.: A cooperative coevolutionary algorithm with correlation based adaptive variable partitioning. In: IEEE Congress on Evolutionary Computation, 2009. CEC'09, pp. 983–989. IEEE (2009)
Rosin, C.D., Belew, R.K.: New methods for competitive coevolution. Evol. Comput. **5**(1), 1–29 (1997)
Shi, M., Wu, H.: Evolving efficient connection for the design of artificial neural networks. In: Artificial Neural Networks—ICANN 2008, pp. 909–918. Springer (2008)
Sofge, D., De Jong, K., Schultz, A.: A blended population approach to cooperative coevolution for decomposition of complex problems. In: Proceedings of the World on Congress on Computational Intelligence, pp. 413–418. IEEE (2002)
Spears, W.M., De Jong, K.A., Bäck, T., Fogel, D.B., De Garis, H.: An overview of evolutionary computation. In: Machine Learning: ECML-93, pp. 442–459. Springer (1993)
Srinivas, N., Deb, K.: Muiltiobjective optimization using nondominated sorting in genetic algorithms. Evol. Comput. **2**(3), 221–248 (1994)
Thomason, R., Soule, T.: Novel ways of improving cooperation and performance in ensemble classifiers. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, pp. 1708–1715. ACM (2007)
Watson, R.A., Pollack, J.B.: Coevolutionary dynamics in a minimal substrate. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001), pp. 702–709. Morgan Kaufmann (2001)
Watson, R.A., Pollack, J.B.: A computational model of symbiotic composition in evolutionary transitions. Biosystems **69**(2), 187–209 (2003)
Watson, R.A., Pollack, J.B.: Modular interdependency in complex dynamical systems. Artif. Life **11**(4), 445–457 (2005)
Weicker, K., Weicker, N.: On the improvement of coevolutionary optimizers by learning variable interdependencies. In: Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99. IEEE (1999)
Wiegand, R.P.: An analysis of cooperative coevolutionary algorithms. Ph.D. Thesis, George Mason University (2004)
Wiegand, R.P., Liles, W.C., De Jong, K.A.: The effects of representational bias on collaboration methods in cooperative coevolution. In: Parallel Problem Solving from Nature—PPSN VII, pp. 257–268. Springer (2002)
Wiegand, R.P., Liles, W.C., De Jong, K.A.: An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pp. 1235–1245 (2001)
Wu, S.X., Banzhaf, W.: A hierarchical cooperative evolutionary algorithm. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, pp. 233–240. ACM (2010)

Wu, S.X., Banzhaf, W.: Rethinking multilevel selection in genetic programming. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, pp. 1403–1410. ACM (2011)
Yang, Z., Tang, K., Yao, X.: Large scale evolutionary optimization using cooperative coevolution. Inf. Sci. **178**(15), 2985–2999 (2008)
Yang, Z., Tang, K., Yao, X.: Multilevel cooperative coevolution for large scale optimization. In: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), pp. 1663–1670. IEEE (2008)
Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms—a comparative case study. In: Parallel Problem Solving from Nature—PPSN V, pp. 292–301. Springer (1998)