Archive-based Cooperative Coevolutionary Algorithms

Liviu Panait
lpanait@cs.gmu.edu

Sean Luke
sean@cs.gmu.edu

Joseph F. Harrison
jharri1@gmu.edu

Department of Computer Science
George Mason University
4400 University Dr, MSN 4A5
Fairfax, VA 22030 USA

ABSTRACT

Archive-based cooperative coevolutionary algorithms attempt to retain a set of individuals which act as good collaborators for other coevolved individuals in the evolutionary system. We introduce a new archive-based algorithm, called iCCEA, which compares favorably with other cooperative coevolutionary algorithms. We explain the current problems with cooperative coevolution which have given rise to archive methods, detail the iCCEA algorithm, compare it against other traditional and archive-based methods on basic problem domains, and discuss the reasons behind the performance of various algorithms.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems

General Terms

Experimentation

Keywords

Cooperative Coevolution, Archive-based Coevolution

1. INTRODUCTION

Cooperative coevolution [5, 15] tries to simplify the search space of a problem by breaking the structure of a candidate solution into subcomponents, each evolved in a separate population. The fitness of an individual is assessed by testing it in combination with individuals from the other populations to form complete solutions, and (typically) taking the maximum result from those combinations.

One notional use of cooperative coevolutionary algorithms (CCEAs) is to take advantage of decomposable problems to simplify the search space. Though in some cases CCEAs may not behave in this fashion [6], we believe this heuristic to be the primary impetus for using a CCEA. To begin, imagine the situation where each component's contribution to the joint fitness turns out to be entirely independent of one another. In this "trivial" case, cooperative coevolution can reduce an $O(a^n)$ search space into n separate component spaces each $O(a)$ in size, assuming the components all have the same length. Of course, in this case one could just as well perform a separate evolutionary run on each of the subspaces. But in more interesting problems, components will have dependencies among them, but some — hopefully much — of the space may consist of relatively independent regions. If the problem is decomposed into components along these "fissure" lines, cooperative coevolution promises to exploit the decomposition to achieve much of the aforementioned search space reduction when separate per-subspace runs are not feasible.

Unfortunately, by projecting the search space into separate component search spaces, CCEAs lose a great deal of information. The fitness of component individuals is sensitive to the components (*collaborators*) with which they are teamed. As a result, CCEAs may not just get caught in but *gravitate towards* suboptimal solutions represented by Nash equilibria in the joint search space [16].

One way to counter this is to endow the individuals with more information about which collaborators would show the individuals at their best. Imagine if one had an oracle function $f(i)$ which, for a given individual i, provided that collaborator which helped i achieve its maximum possible fitness. If we knew f, we could reduce the problem to the aforementioned "trivial" case. f is of course theoretical fiction. But a real-world heuristic estimate of f can be helpful even if noisy and inaccurate: significant performance improvements have been demonstrated even when f is only partly available [11].

The obvious approximation of f is to test i with a large number of collaborators and to take the maximum. Earlier work [2, 3, 17] has demonstrated the efficacy of taking the maximum of N evaluations with random collaborators from the other population, plus perhaps the best performing collaborator of the previous generation, rather than the average or minimum of them. The advantages of using more collaborators were graphically illustrated in [13], and the benefits of varying the collaboration scheme over time were demonstrated in [10]. But using large numbers of collaborators makes for expensive fitness evaluations, and at some point, testing with more collaborators may start hurting more than it helps.

iCCEA attempts to reduce the number of fitness evaluations by maintaining an archive of good collaboration choices for each of the populations. The idea is to intelligently identify which members of the collaborating population are likely to be good collaborators, and only test individuals against that small set rather than against the full population itself or against a large number of random collaborators. This is similar to maintaining information about useful past collaborations, as used in [9, 18]. iCCEA selects these collaborators from among those members of the previous population which helped individuals improve themselves the most. iCCEA

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.

further reduces the archive size (and thus the number of evaluations) by taking advantage of the ranked nature of tournament selection: if a collaborator improves an individual, but not enough to change its rank (and thus its evolutionary viability), it is not considered for the archive. Further, iCCEA maintains diversity in the archive by ensuring that it doesn't contain individuals whose maximal collaborations are too close to one another in solution space.

In this paper we will detail the iCCEA archive algorithm, and we will compare it against a related algorithm, pCCEA [1]. We will also compare these algorithms against three ordinary CCEAs (cCCEA, rCCEA and rCCEA-Perm, described later) on several two-population CCEA test problems. Overall, the results will show that iCCEA significantly outperforms its competitors in most experiments we performed.

2. COOPERATIVE COEVOLUTION

CCEAs use multiple populations to optimize a final joint solution. Each population evolves individuals representing a component of the final solution: a full candidate solution is formed by joining together an individual chosen from each population.

In a CCEA, an individual is evaluated by joining it with one or more individuals (collaborators) from the other population and assessing the performance of each of the results. Generally (and in all cases in this paper) the fitness of the individual is the maximum reward over these evaluations. In other respects, the evolutionary process within a population is typical. In this paper we will restrict ourselves to two populations, each using a generational EA.

There are three common kinds of CCEAs. In the *sequential CCEA* [15], only one population is evaluated and bred per generation (i.e. the populations take turns). In *parallel CCEAs* [16], both populations are simultaneously evaluated and bred. In some parallel CCEAs, the individuals are assessed by combining them with collaborators from the previous generation, while in others, the current generations' collaborators are used. All algorithms presented in this paper will use this last case.

How should collaborators be chosen for an individual? One way, which we term the cCCEA algorithm, is to evaluate an individual against *every single collaborator* in the other population. cCCEA is guaranteed to converge to the global optimum if the population size is sufficiently large [11]. An alternative is to sample from the other population. In the rCCEA algorithm, an individual's fitness is the maximum reward over evaluations with six individuals: five random individuals, plus the fittest individual in the collaborator population from the previous generation. A variant of rCCEA, which we will call rCCEA-Perm, chooses random collaborators by shuffling each population and then pairing individuals in the populations. In rCCEA-PERM, both the individual and its collaborator count the evaluation towards their tally, and so the number of evaluations is almost halved.

Bucci and Pollack [1] applied recent advances from competitive coevolution research to improve CCEAs, resulting in the pCCEA algorithm. pCCEA employs an elitist archive in each population which maintains those individuals in the population which, as collaborators, were effective in assisting *some* individual from the other population. The algorithm defines a domination relationship among individuals: individual i dominates individual j if, for all collaborators x, i's reward joined with x is never worse than j's reward joined with x, and there exists at least one collaborator y for which i's joint reward is higher than j's. Each generation, pCCEA first evaluates every individual with every possible collaborator, then assembles the set of individuals dominated by no one else. This set forms the archive, and is automatically copied to the next generation. pCCEA then fills the rest of the next generation by iteratively selecting two random individuals and comparing them. If neither dominates the other, both are selected for breeding. If one is dominated by the other, only the dominating individual is selected.

It is important to note some possible drawbacks of this algorithm. First, pCCEA does not use the archive to define a small set of collaborators. Instead, it uses the archive only to promote certain collaborators to the next generation where they, along with new population members, will be used to evaluate individuals. Thus it maintains a set of "informative" collaborators but does not try to use that set to minimize the number of necessary evaluations. This is both positive and negative, as it requires many evaluations, but provides diversity in evaluations by adding newly-generated collaborators to the evaluation mix. Second, pCCEA does not use fitness to select individuals. An individual may be selected if it collaborates better than others with even a single partner from the other population, even if that collaboration result is very poor. Third, pCCEA only compares individuals to other individuals and not to the group. Thus an individual may be selected for the archive if it beats each individual in *some* collaboration scenario even if it is never the best choice in the population for *any* collaboration scenario. Both of these features may tend to slow evaluation as more individuals are promoted to the next generation. Fourth, and importantly, our experiments in Section 3 indicate that pCCEA's archive tends to converge to the Pareto frontier, which may be unfortunately infinite in even simple cooperative multiagent domains. In such situations, the archive will rapidly consume the entire population, and search will stagnate.

2.1 The iCCEA Algorithm

The iCCEA algorithm also maintains an archive of fruitful individuals, but it constructs and uses this archive in a manner different from pCCEA. Notably, rather than testing individuals with the full population of collaborators, iCCEA tests them only with the members of the archive, plus additional randomly-chosen collaborators if the archive size does not provide enough evaluations. iCCEA's approach to archive construction tends to result in very small archives, even in pathologically bad situations for pCCEA.

Like pCCEA, iCCEA promotes this archive to the next generation, and uses tournament selection to fill in the rest of the new population. Tournament selection is based on the fitnesses of the individuals rather than pareto dominance. Because tournament selection is used, we may consider just the rank ordering among the individuals rather than their actual fitness. iCCEA tries to build a small archive of collaborators which produce the same rank-ordering of fitnesses among individuals in the other population as they would receive were they tested with the full population of collaborators in the previous generation. Each archive member helped *some* individual receive its highest reward, and thus provides information about an "interesting" part of the collaboration space.

The size of the archive is an important factor. A large archive is expensive to evaluate against, but it provides more information about the complexity of the search space. An archive size of 1 reduces to a common evaluation approach for CCEAs [15, 17]: using the best individual (from the previous generation) plus some individuals chosen at random from the other population.

Evaluation. At the beginning of a run, the archive ($Archive_p$) of each population p is simply set to the population itself. This means that in the first generation, each individual of a population will be evaluated against all the individuals of the other population. This is expensive but it is intended to provide good initial collaborator information before the archive adjusts to a more efficient size.

Evaluation is in three parts. First, individuals are evaluated against the other population's archive members. Second, if more evaluations are desired (if the number of evaluations per individual has not yet reached $MaxEvals$), individuals in populations are repeatedly paired off with individuals in the other population and evaluated together. Third, each individual's fitness is set to the maximum reward it obtained over all its evaluations this generation. The pseudocode for evaluation process is:

```
iCCEA-Evaluation
  Parameter MaxEvals: maximum evaluations per individual
  For each population p
    p' = the other population
    For each individual i in p
      For each individual j in p'
        F_i^j = -∞
      For each individual a in Archive_p'
        F_i^a = Reward_p(i,a)
        F_a^i = Reward_p'(i,a)
  MaxArchive = max_p |Archive_p|
  Repeat for max(0, MaxEvals - MaxArchive) times
    For each population p
      Shuffle p
    For i from 1 to PopSize
      a_1 = individual in population p^1 with index i
      b_1 = individual in population p^2 with index i
      F_{a_1}^{b_1} = Reward_{p^1}(a_1, b_1)
      F_{b_1}^{a_1} = Reward_{p^2}(a_1, b_1)
  For each population p
    p' = the other population
    For each individual i in p
      Fitness(i) = max_{j∈p'} F_i^j
```

Breeding and Archive Selection. The breeding and population reassembly phase of iCCEA proceeds similarly to the one in pCCEA: the archive members are selected from the old population and are copied directly into the new population, and the remainder of the new population is filled with children bred using standard evolutionary computation algorithms applied to the old population (including the old archive). The entire previous population (including the archive) competes for breeding. The pseudocode is straightforward:

```
iCCEA-Breeding
  For each population
    Select its new archive with iCCEA-Archive-Selection
    Copy the archive into the new population
    Fill the rest of the new population using standard EC breeding
```

Archive selection is intended to select those individuals which revealed features of the projected joint space useful to the other population. Specifically, we aim to select a minimal archive of individuals from population p such that when assessing the fitness of individuals in the other population p', testing them against the full set of individuals in p would not change their rank ordering beyond just testing them against the individuals in p's archive. The hope is that this archive would provide an accurate evaluation and ranking of the individuals in p' in the next generation as well.

We want this archive to be minimal because each individual in p' is being evaluated against every single individual in p's archive. Large archives imply an $O(n^2)$ evaluation cost per generation. Therefore we add individuals to the archive only if they cause individuals in the other population to improve significantly enough so as to effect the ranking. Of the various individuals which change this ranking, we will select the ones which do so by raising fitnesses to the highest levels.

```
iCCEA-Archive-Selection
  Parameter MinDist: minimum distance requirement
  For each population p
    p' = the other population
    Archive_p = ∅
    Ineligible_p = ∅
    Repeat forever
      For each individual i in p - Archive_p
        For each individual x in p'
          Fit1_x = max_{j∈Archive_p} F_x^j
          Fit2_x^i = max(Fit1_x, F_x^i)
        For each individual x in p'
          For each individual y in p'
            Fit3_{x,y}^i = { Fit2_x^i   if Fit1_x ≤ Fit1_y and Fit2_x^i > Fit2_y^i
                           { -∞         otherwise
        For each individual i in p - (Archive_p ⋃ Ineligible_p)
          MaxFit_i = max_{x,y∈p'} Fit3_{x,y}^i
        a = arg max_i MaxFit_i
        If MaxFit_a = -∞
          Break from repeat loop
        Select y such that MaxFit_a = max_x Fit3_{xy}^a
        If min_{i∈Archive_p} distance(⟨a,y⟩, ⟨i,Collaborator_i⟩) < MinDist
          Ineligible_p = Ineligible_p ⋃ (a)}
        Else
          Add a to Archive_p
          Collaborator_a = y
```

The archive selection process starts from the empty set and proceeds iteratively. For each individual i not yet in the archive, and for each individual x in the other population, we first compute $Fit1_x$, the fitness of x if evaluated in combination with all individuals currently in the archive. We then compute $Fit2_x^i$, the fitness of x if i were part of the archive. Note that $Fit2_x^i \geq Fit1_x$. We use three criteria to determine if i should be added to the archive. First: does there exist a pair of individuals x and y in the other population whose relative ranks change when i is added to the archive? That is, is it true that $\exists x,y : Fit1_x \leq Fit1_y$ and $Fit2_x^i > Fit2_y^i$? Second, is the individual "eligible"? An individual i is ineligible if there already exists an individual in the archive whose joint solution (with the collaborator whose rank it significantly improved) is sufficiently "close" in genotype space to the joint solution formed by i and the collaborator whose rank i improves most (we use euclidian distance). Third, and finally, of all individuals i that meet the first two criteria, we add to the archive the one that changed the ranking by raising the fitness of its x to the highest level. Note that the first individual to be selected for the archive is always the one with the highest fitness. The naive algorithmic description of iCCEA-Archive-Selection was chosen for clarity but is $O(n^3)$. It is relatively straightforward to design algorithms that take advantage of the relatively small archive size in order to significantly reduce the asymptotic complexity.

3. EXPERIMENTS

In [1], Bucci compared pCCEA against certain traditional coevolutionary algorithms. We continue that study with a further comparison of pCCEA, cCCEA, rCCEA, rCCEA-Perm, and iCCEA. For iCCEA, we set $MaxEvals = 5$, but we used different settings for $MinDist$: 0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.5, 0.75, and 1.0. We call

each iCCEA-m, where m is the *MinDist* parameter used.

We tested these algorithms over three different kinds of problem domains. The MTQ problem family, Rastrigin, and Griewangk problems are multimodal problems (they have local optima) with a discrete number of Nash Equilibria. The OneRidge, Rosenbrock, and Booth problems are unimodal but have an infinite number of Nash Equilibria. The SMTQ problem family is both multimodal and has infinite Nash Equilibria. Infinite Nash equilibria often appear in the form of diagonal ridges or curves in the joint landscape, where change in a single collaborator — expressed as a horizontal or vertical move in the landscape — heads off the fitness ridge and drops in fitness. To improve, *both* collaborators must change simultaneously, resulting in a diagonal move along the ridge.

Some of the problems were inverted from their original form to be used as maximization problems. All experiments employed two populations of 32 individuals each. Individuals encoded real-valued numbers between 0 and 1. Parents were selected via tournament selection of size 2, and children were created via mutation by adding to the parent's value a number generated randomly from a normal distribution with mean 0 and standard deviation 0.01, rejecting and reselecting the value if it fell outside [0...1]. The fittest individual was copied automatically to the next population for cCCEA, rCCEA, and rCCEA-Perm. The entire archive survived automatically from one generation to the next for iCCEA and pCCEA.

The experiments were performed using the ECJ package [8]. Given that the algorithms required different numbers of evaluations per generation, we gave each a budget of 51200 evaluations (plus or minus a few extra to complete the last generation). This led to 50 generations for pCCEA and pCCEA, 134 generations for rCCEA, 240 generations for rCCEA-Perm, and a variable number of generations for iCCEA. Each experiment was repeated 250 times for statistical significance. Given that results often did not have a normal distribution, we reported the 95% confidence interval for the median of the results (as recommended in [7]). Statistical significance was verified via non-parametric pairwise t-tests. Each such test was performed at a 99.999% confidence level (approximated via the Bonferroni inequality) to provide an overall confidence level of 95% over all tests.

3.1 Multimodal Domains

Multimodal domains are challenging for cooperative coevolution because both populations need to coordinate to explore multiple peaks. We start our exploration in a simple two-peak domain used previously in [16] to emphasize the relative overgeneralization pathology associated with certain cooperative coevolutionary algorithms. Following this, we analyze the performance of the methods in two traditional optimization benchmark problems: Griewangk and Rastrigin. As a summary of our findings, we observe that iCCEA performs best, especially for relatively small values of *MinDist* (usually for *MinDist* ≤ 0.5). At the other extreme, pCCEA appears to have the worst performance. The specifics of these experiments follow.

3.1.1 The Maximum of Two Quadratics Domains

The maximum of two quadratics (MTQ) class of domains includes a global optimum and a local suboptimum, where the suboptimum covers a much wider range of the search space and is thus difficult to escape. The problems have been used before by [1, 12].

The function for the MTQ class is defined as

$$\text{MTQ}(x,y) \leftarrow \max \begin{cases} H_1 * (1 - \frac{16*(x-X_1)^2}{S_1} - \frac{16*(y-Y_1)^2}{S_1}) \\ H_2 * (1 - \frac{16*(x-X_2)^2}{S_2} - \frac{16*(y-Y_2)^2}{S_2}) \end{cases}$$

Table 1: 95% confidence interval for the median performance of the methods in the MTQ domain instance with $H_1 = 50$

Method	Lower Bound	Median	Upper Bound
pCCEA	149.7024	149.79412	149.8466
cCCEA	149.99997	149.99997	149.99998
iCCEA-0.0	150	150	150
iCCEA-0.05	150	150	150
iCCEA-0.1	150	150	150
iCCEA-0.15	150	150	150
iCCEA-0.2	150	150	150
iCCEA-0.25	150	150	150
iCCEA-0.5	150	150	150
iCCEA-0.75	150	150	150
iCCEA-1.0	150	150	150
rCCEA	50	50	149.99998
rCCEA-Perm	50	50	150

where x and y may take values ranging between 0 and 1. Different settings for H_1, H_2, X_1, Y_1, X_2, Y_2, S_1, and S_2 affect the difficulty of the problem domain in one of the following aspects. H_1 and H_2 affect the heights of the two peaks: higher peaks may increase the chances that the algorithm converges there. S_1 and S_2 affect the area that the two peaks cover: higher values for one of them result in a wider coverage of the specific peak. This makes it more probable that the coevolutionary search algorithm will converge to this peak, even though it may be suboptimal. Different values for X_1, Y_1, X_2, and Y_2 result in changes in the locations of the centers of the two quadratics, which also affect the relatedness of the two peaks: similar values of the x or y coordinates for the two centers imply higher overlaps of the projections along one or both axes (the projections of the joint space for one or both agents may retain more information about the globally optimal solution even if the other agent's population starts to converge to the suboptimal solution). In these experiments, we set $S_1 = \frac{16}{10}$, $X_1 = \frac{3}{4}$, $Y_1 = \frac{3}{4}$, $H_2 = 150$, $S_2 = \frac{1}{32}$, $X_2 = \frac{1}{4}$, and $Y_2 = \frac{1}{4}$. H_1 was varied across experiments, but it was always less than 125.

First, we set H_1 to 50 to have a wide difference between the heights of the two peaks. In this case, coevolution may have difficulties finding the global optimum primarily because the optimum's coverage is significantly smaller than that of the suboptimal peak. Table 1 presents the performance of the methods in the MTQ domain. The average run of iCCEA lasted about 262 generations for each setting of *MinDist*. We observe that the iCCEA algorithm converges to the optimal answer in most runs. Non-parametric statistical tests indicate that iCCEA is better than all other methods for any setting of *MinDist*.

Similar to the experiments in [1], we set H_1 to 125 to create a more deceiving domain instance: the individuals on the suboptimal peak have higher fitness and are thus more likely to be selected. This also reduces the size of the area where the optimal peak is superior to the suboptimal one, making the problem harder than when $H_1 = 50$.

The results (summarized in Table 2) indicate that iCCEA with *MinDist* smaller or equal to 0.5 is the top tier performer, and it significantly outperforms the other methods. The second tier consists of cCCEA, pCCEA, and iCCEA with *MinDist* greater than 0.5. Last, rCCEA and rCCEA-Perm have significantly worse results than all other methods. The difference in heights leads to a slight increase in archive size for iCCEA. As a consequence, the average run of iCCEA lasted around 253 generations, down from an average of 262 generations when $H_1 = 50$ (this decrease is significant at the 99.999% confidence level).

Table 2: 95% confidence interval for the median performance of the methods in the MTQ domain instance with $H_1 = 125$

Method	Lower Bound	Median	Upper Bound
pCCEA	148.97783	149.4233	149.65022
cCCEA	149.99985	149.99991	149.99995
iCCEA-0.0	150	150	150
iCCEA-0.05	150	150	150
iCCEA-0.1	150	150	150
iCCEA-0.15	150	150	150
iCCEA-0.2	150	150	150
iCCEA-0.25	150	150	150
iCCEA-0.5	150	150	150
iCCEA-0.75	125	125	149.99998
iCCEA-1.0	125	125	125
rCCEA	125	125	125
rCCEA-Perm	125	125	125

Table 3: 95% confidence interval for the median performance of the methods in the Griewangk domain

Method	Lower Bound	Median	Upper Bound
pCCEA	-0.009099687	-0.008524607	-0.008080815
cCCEA	-2.3027173e-08	-1.6132691e-08	-1.06091615e-08
iCCEA-0.0	-1.2595542e-09	-9.4223985e-10	-7.1052053e-10
iCCEA-0.05	-6.868509e-10	-5.39539185e-10	-4.6161386e-10
iCCEA-0.1	-7.2518436e-10	-5.51555305e-10	-4.5181736e-10
iCCEA-0.15	-7.856029e-10	-6.1868948e-10	-4.777243e-10
iCCEA-0.2	-7.931591e-10	-6.166706e-10	-4.8393356e-10
iCCEA-0.25	-8.377967e-10	-6.9733245e-10	-5.179104e-10
iCCEA-0.5	-7.4639095e-10	-5.629196e-10	-4.5529336e-10
iCCEA-0.75	-0.0073960405	-2.74472505e-09	-1.2845356e-09
iCCEA-1.0	-0.0073960405	-2.56566515e-09	-8.832689e-10
rCCEA	-0.007396041	-0.0073960405	-0.0073960405
rCCEA-Perm	-0.0073960423	-0.0073960414	-0.007396041

Table 4: 95% confidence interval for the median performance of the methods in the Rastrigin domain

Method	Lower Bound	Median	Upper Bound
pCCEA	-0.0001342016	-4.20476935e-05	-2.75237e-05
cCCEA	-3.373491e-05	-1.92212125e-05	-1.20936875e-05
iCCEA-0.0	-0.99495906	-7.32915735e-07	-3.9109466e-07
iCCEA-0.05	-0.99495906	-1.44978885e-06	-4.859953e-07
iCCEA-0.1	-1.5460682e-06	-6.5733175e-07	-3.8390752e-07
iCCEA-0.15	-0.99495906	-1.46234045e-06	-7.4304603e-07
iCCEA-0.2	-0.99495906	-9.1238677e-07	-4.893715e-07
iCCEA-0.25	-0.99495906	-8.07612e-07	-4.2249255e-07
iCCEA-0.5	-9.465858e-07	-5.692597e-07	-3.5596943e-07
iCCEA-0.75	-0.99495906	-8.1733685e-07	-5.13917e-07
iCCEA-1.0	-2.7152691e-06	-6.213839e-07	-4.274801e-07
rCCEA	-0.9949591	-5.13458285e-06	-2.3090972e-06
rCCEA-Perm	-1.700194e-06	-9.64721415e-07	-5.1975854e-07

Table 5: 95% confidence interval for the median performance of the methods in the OneRidge domain

Method	Lower Bound	Median	Upper Bound
pCCEA	1.47526	1.4772663	1.4805671
cCCEA	1.9103878	1.91416505	1.9197007
iCCEA-0.0	1.5467398	1.5517363	1.5576606
iCCEA-0.05	2	2	2
iCCEA-0.1	2	2	2
iCCEA-0.15	2	2	2
iCCEA-0.2	2	2	2
iCCEA-0.25	2	2	2
iCCEA-0.5	2	2	2
iCCEA-0.75	2	2	2
iCCEA-1.0	2	2	2
rCCEA	2	2	2
rCCEA-Perm	2	2	2

3.1.2 The Griewangk Domain

The Griewangk function is defined as

$$Griewangk(x,y) \leftarrow -1 - \frac{\bar{x}^2}{4000} - \frac{\bar{y}^2}{4000} + \cos(\bar{x})\cos(\frac{\bar{y}}{\sqrt{2}})$$

where $\bar{x} = 10.24x - 5.12$, $\bar{y} = 10.24y - 5.12$, and x and y are encoded in individuals as real-valued numbers between 0 and 1. The function has a maximum value equal to 0 for $x = y = 0$, and several suboptimal peaks surrounding it.

The results are summarized in Table 3. iCCEA performs significantly better than all other methods when $MinDist \leq 0.5$. Of the iCCEA settings, $MinDist = 0.0$ appears to be slightly worse (with confidence around 99%, lower than the desired 99.999%). This is because iCCEA-0.0 has significantly higher archive sizes, yielding a lower numbers of generations per run (161 generations for $MinDist = 0.0$, compared to 233 generations for $MinDist = 0.2$). The second tier of performers consists of cCCEA, rCCEA, rCCEA-Perm, iCCEA-0.75 and iCCEA-1.0. Finally, pCCEA is significantly worse than all other methods.

3.1.3 The Rastrigin Domain

The Rastrigin function is defined as

$$Rastrigin(x,y) \leftarrow -20 - \bar{x}^2 + 10\cos(2\pi\bar{x}) - \bar{y}^2 + 10\cos(2\pi\bar{y})$$

where $\bar{x} = 10.24x - 5.12$, $\bar{y} = 10.24y - 5.12$, and x and y encoded in individuals as real-valued numbers between 0 and 1. The function has a maximum value equal to 0 for $x = y = 0$, and many suboptimal peaks surrounding it.

The results are summarized in Table 4. The first tier of performers consists of iCCEA (for any value of *MinDist*) and rCCEA-Perm, which are not significantly worse than any other method. rCCEA is worse then iCCEA for four settings of *MinDist*. Last, cCCEA and pCCEA are dominated by ten and eleven other methods respectively. Surprisingly, iCCEA had about 263 generations per run, indicating that the archive size is very low. While this contradicts our expectations for a highly multimodal peak, we believe this effect is due to the fact that the Rastrigin function describes essentially a quadratic curve ($1 - \bar{x}^2 - \bar{y}^2$), accompanied by a plethora of smaller peaks. As a consequence, iCCEA appears to exploit the underlying quadratic function to achieve a small archive at each generation.

3.2 Domains with Infinite Nash Equilibria

Optimal and suboptimal peaks are usually Nash equilibria. Suboptimal Nash equilibria can distort the perspective each population has over the joint search space to the point of completely eliminating all information about the globally optimal solution [10]. iCCEA is able to cope better with this issue because its archive stores information about multiple peaks simultaneously.

What happens in domains where there are just too many Nash equilibria? While one might believe that such domains would have a plethora of peaks that could trick most algorithms, that is not usually the case. Instead, problems containing diagonal up-the-hill ridges are also characterized by an infinite number of Nash equilibria! One such simple domain is OneRidge, proposed in [14]. The Rosenbrock and Booth functions create domains with an infinite number of Nash equilibria as well. The results of experiments (de-

Table 6: 95% confidence interval for the median performance of the methods in the Rosenbrock domain

Method	Lower Bound	Median	Upper Bound
pCCEA	-0.1084212	-0.086038075	-0.07215236
cCCEA	-5.4309703e-06	-3.97336365e-06	-3.095801e-06
iCCEA-0.0	-7.0464353e-06	-5.6810313e-06	-4.386692e-06
iCCEA-0.05	-2.9603286e-06	-2.0364071e-06	-1.484208e-06
iCCEA-0.1	-1.5654674e-06	-1.05892685e-06	-6.922747e-07
iCCEA-0.15	-9.347807e-07	-7.36914e-07	-5.559006e-07
iCCEA-0.2	-7.9747747e-07	-5.1362627e-07	-4.3587843e-07
iCCEA-0.25	-1.1045466e-06	-8.6313633e-07	-5.565543e-07
iCCEA-0.5	-1.5428313e-06	-1.1737227e-06	-7.397234e-07
iCCEA-0.75	-9.623004e-07	-6.5651085e-07	-4.610377e-07
iCCEA-1.0	-9.135398e-07	-6.48494135e-07	-4.043417e-07
rCCEA	-1.45301265e-05	-1.1222902e-05	-8.5360825e-06
rCCEA-Perm	-5.950367e-07	-4.9335307e-07	-4.3179807e-07

Table 7: 95% confidence interval for the median performance of the methods in the Booth domain

Method	Lower Bound	Median	Upper Bound
pCCEA	-2.2757607e-05	-1.78912995e-05	-1.3247128e-05
cCCEA	-4.1714014e-07	-3.2144021e-07	-2.7121857e-07
iCCEA-0.0	-0.00010195026	-8.06368125e-05	-5.630301e-05
iCCEA-0.05	-5.884756e-07	-4.18359055e-07	-3.5493673e-07
iCCEA-0.1	-3.606497e-07	-2.82521425e-07	-2.2286326e-07
iCCEA-0.15	-4.869323e-07	-3.895216e-07	-3.1035532e-07
iCCEA-0.2	-3.2798286e-07	-2.77684885e-07	-2.2034128e-07
iCCEA-0.25	-4.5106958e-07	-3.7739077e-07	-2.7794223e-07
iCCEA-0.5	-3.0225107e-07	-2.44586035e-07	-2.1388162e-07
iCCEA-0.75	-3.2224608e-07	-2.5254693e-07	-1.9835213e-07
iCCEA-1.0	-4.3931948e-07	-3.6072215e-07	-2.66255e-07
rCCEA	-1.5708204e-07	-1.31100655e-07	-1.02842776e-07
rCCEA-Perm	-4.172578e-07	-2.9863297e-07	-2.3673356e-07

tailed next) indicate that restricting the diversity of iCCEA's archive is crucial for good performance in these three domains; otherwise, the archive usually becomes too large and interferes with the exploration of the space.

3.2.1 The OneRidge Domain

The OneRidge problem maximizes the function

$$OneRidge(x,y) \leftarrow 1 + 2\min(x,y) - \max(x,y)$$

where x and y range between 0 and 1. OneRidge is particularly difficult for concurrent learners because it contains a very large number of Nash equilibria: for any value v between 0 and 1, (v,v) is a Nash equilibrium. This implies that for almost any Nash equilibrium (except for the global optimum $(1,1)$) there are an infinite number of better Nash equilibria; unfortunately, both populations must simultaneously change to a new equilibrium in order to improve. To better study the algorithms' capacity to follow this ridge to the global optimum, we randomly initialized the populations of individuals to only values smaller than 0.5.

The results are summarized in Table 5. rCCEA and rCCEA-Perm find the global optimum in every single run. iCCEA achieves identical optimal performance when *MinDist* is greater than 0. Among the remaining methods, cCCEA is significantly better, followed by iCCEA-0.0, then pCCEA. Increases in *MinDist* put additional constraints on archive inclusion, resulting in longer runs (from 134 generations for iCCEA-0.0, to 247 generations for iCCEA-0.05, to 264 generations for iCCEA-1.0).

3.2.2 The Rosenbrock Domain

Next, we examined the performance of the methods in the Rosenbrock domain, which is defined by

$$Rosenbrock(x,y) \leftarrow -(100(\bar{x}^2 - \bar{y})^2 + (1-\bar{x})^2)$$

where $\bar{x} = 10.24x - 5.12$, $\bar{y} = 10.24y - 5.12$, and x and y are encoded in individuals as real-valued numbers between 0 and 1. Similarly to OneRidge, Rosenbrock is particularly difficult for concurrent learners because optimization requires both populations to simultaneously follow a narrow ridge up-the-hill; in addition, the ridge in the Rosenbrock domain has a non-linear shape.

The results of the methods are presented in Table 6. The non-parametric statistical tests indicate that rCCEA-Perm and iCCEA (with *MinDist* ranging from 0.1 to 1.0) are the top-tier performers. The second tier consists of cCCEA, iCCEA-0.0, and iCCEA-0.05. rCCEA and pCCEA have the worst performance; in particular, pCCEA is significantly worse than all other methods. The difference among the iCCEA variations again highlights the importance of putting some restrictions on minimum distance between members of the archive: iCCEA-0.0 evaluates populations with a large archive, resulting in around 74 generations per run, while iCCEA-0.2 uses a moderately-sized archive that allows it 254 generations per run.

3.2.3 The Booth Domain

Similarly to the OneRidge domain, the Booth problem also creates a narrow ridge that can be pursued only by simultaneous exploration from both populations. The Booth function is defined as:

$$Booth(x,y) \leftarrow -(\bar{x} + 2\bar{y} - 7)^2 - (2\bar{x} + \bar{y} - 5)^2$$

where $\bar{x} = 10.24x - 5.12$, $\bar{y} = 10.24y - 5.12$, and x and y encoded in individuals as real-valued numbers between 0 and 1. The Booth function creates a squashed peak along a diagonal axis, usually resulting in penalties for miscoordinated explorations.

The results, summarized in Table 7, indicate that rCCEA significantly outperforms all other methods. iCCEA falls in the second tier for $MinDist > 0$, together with cCCEA and rCCEA-Perm. Last, pCCEA is worse than every other method but iCCEA-0.0 (which is significantly worse than everything else). As in the OneRidge and Rosenbrock domain, iCCEA-0.0 used larger archives and exhausted its computational budget in fewer generations (around 160), while even small values of *MinDist* reduced the archive size significantly (runs lasted for about 261 generations when $MinDist > 0$). We are still exploring the causes for rCCEA's impressive performance in this problem domain.

3.3 Multimodal Domains with Infinite Nash Equilibria

Finally, we created a new class of problem domains that combines the difficulties associated with both multimodal search spaces and domains with infinite numbers of Nash equilibria along diagonal ridges. We started with the MTQ class of problems, and changed each peak to have an ellipsoid shape aligned diagonally. This resulted in diagonal ridges towards the optima, and thus infinite numbers of Nash equilibria along those ridges. The new class of problems, which we termed SMTQ, is defined as

$$\text{SMTQ}(x,y) \leftarrow \max \begin{cases} H_1 * (1 - \frac{32(x_1^r - X_1)^2}{S_1} - \frac{8(y_1^r - Y_1)^2}{S_1}) \\ H_2 * (1 - \frac{32(x_2^r - X_2)^2}{S_2} - \frac{8(y_2^r - Y_2)^2}{S_2}) \end{cases}$$

Table 8: 95% confidence interval for the median performance of the methods in the SMTQ domain instance with $H_1 = 50$

Method	Lower Bound	Median	Upper Bound
pCCEA	130.5234	135.36403	139.55858
cCCEA	149.99992	149.99994	149.99995
iCCEA-0.0	149.99994	149.99994	149.99995
iCCEA-0.05	149.99998	149.99998	149.99998
iCCEA-0.1	149.99998	149.99998	149.99998
iCCEA-0.15	149.99998	149.99998	150
iCCEA-0.2	149.99998	149.99998	149.99998
iCCEA-0.25	149.99998	149.99998	150
iCCEA-0.5	149.99998	149.99998	150
iCCEA-0.75	149.99998	149.99998	149.99998
iCCEA-1.0	149.99998	149.99998	149.99998
rCCEA	50	149.99981	149.99994
rCCEA-Perm	149.99995	149.999975	149.99998

Table 9: 95% confidence interval for the median performance of the methods in the SMTQ domain instance with $H_1 = 125$

Method	Lower Bound	Median	Upper Bound
pCCEA	133.09122	137.50821	140.33017
cCCEA	149.99974	149.99985	149.9999
iCCEA-0.0	149.9999	149.99991	149.99995
iCCEA-0.05	149.99997	149.99998	149.99998
iCCEA-0.1	149.99997	149.99998	149.99998
iCCEA-0.15	149.99995	149.99997	149.99998
iCCEA-0.2	149.99997	149.99998	149.99998
iCCEA-0.25	149.99997	149.99998	149.99998
iCCEA-0.5	149.99997	149.99998	149.99998
iCCEA-0.75	149.99994	149.99997	149.99997
iCCEA-1.0	125	149.99995	149.99997
rCCEA	125	125	125
rCCEA-Perm	125	125	125

where x_1^r, y_1^r, x_2^r, and y_2^r are the original x and y values (which ranged between 0 and 1) rotated around the centers of the two peaks by $\pi/4$:

$$x_1^r = (x - X_1)\cos(\pi/4) + (y - Y_1)\sin(\pi/4) + X_1$$
$$y_1^r = (x - X_1)\cos(\pi/4) - (y - Y_1)\sin(\pi/4) + Y_1$$
$$x_2^r = (x - X_2)\cos(\pi/4) + (y - Y_2)\sin(\pi/4) + X_2$$
$$y_2^r = (x - X_2)\cos(\pi/4) - (y - Y_2)\sin(\pi/4) + Y_2$$

We used the same values for H_2, X_1, Y_1, X_2, Y_2, S_1, and S_2 as for the MTQ class.

Tables 8–9 present the results of the methods in the SMTQ domains for $H_1 = 50$ and $H_1 = 125$. iCCEA with $MinDist \in \{0.05, ..., 0.5\}$ performs significantly better than pCCEA, cCCEA, rCCEA, and rCCEA-Perm in both domains. This was expected given the results in the previous domains: no distance restrictions ($MinDist = 0.0$) leads to large archives that interfere with following ridges of equilibria in domains such as in Section 3.2, while large values of $MinDist$ tamper with iCCEA's attempts to search multiple peaks concurrently in multimodal domains (Section 3.1).

4. DISCUSSION AND CONCLUSIONS

In [1], Bucci and Pollack argue that pCCEA generally outperforms traditional CCEA methods: but in many of our experiments, this is not the case. It's important to understand why this is so.

All of the problem domains we tested against, although simple, are ones with an infinite number of Pareto-non-dominated points. This is not unreasonable as many real-world problems have this attribute. pCCEA must include all such points in its archive as it finds them. As pCCEA moves the entire archive forward into the next population, and as more non-dominated points are found, the archive grows until it consumes the entire population, at which point evolution stagnates. This happens even in the simple MTQ domain. There exist a few problem domains (notably OneRidge) for which the theoretical number of eligible archive members is the same for both iCCEA and pCCEA. But even in this case, the ranking used in iCCEA tends to result in a much smaller overall archive (even with no constraints on diversity).

There is another more controversial reason why iCCEA outperforms pCCEA in this paper. Our statistical comparison methodology used a nonparametric population-difference test. Usually, one would compare the means of two samples and verify statistical significance using a t-test. However, this presumes that the samples come from normal distributions. Generally, our multimodal problem domains produce best-of-run samples which are not only non-normal, but contain multiple peaks. These samples are completely inappropriate for a t-test. The standard way to deal with such problems is to use a non-parametric test; in essence, one would combine the results of the two distributions, rank all of them, and then use the rank values instead of the raw values as the statistic in a t-test.

This nonparametric comparison will occasionally produce fairly different (and in our opinion, better) results than a t-test. But such a nonparametric test does not compare difference in mean: it roughly compares difference in *median*. And while pCCEA's median is often worse, its mean is occasionally better than others. For example, the bulk of iCCEA final solutions were near the optimum peak, but occasionally iCCEA converged to the suboptimum. This pulled down the mean, though the median was near-optimum. pCCEA instead would rarely perform as well as iCCEA, but also would rarely find itself in the suboptimum by accident. This was the case in the MTQ and SMTQ domains, where pCCEA was not dominated in terms of mean performance (it actually outperformed all other methods for $H_1 = 50$). When analyzing the mean performance, there was also no significant difference among the methods in the Rastrigin domain, and pCCEA was outperformed by only rCCEA in the Booth domain. However, our analysis in all these cases indicates that the distributions of results were significantly different from one another and from the normal distribution, which in turn indicates that nonparametric tests might better reflect the differences among the methods.

We have two hypotheses as to why iCCEA would find itself caught in local suboptima more often than pCCEA. First, iCCEA primarily compares individuals in conjunction with archive members, whereas pCCEA compares against the whole population. As a result, pCCEA tends to do more exploration initially. Second, pCCEA is much less picky about which individuals are permitted into the archive, allowing more diverse individuals to be preserved to the next generation. Reducing the selection pressure in iCCEA may alleviate these problems. We hope to examine these hypotheses in future work.

Another area of future work is improving the number of evaluations. At present iCCEA (and pCCEA) compare the population against the *entire archive*. If the archive is large, this is expensive. It is plausible to compare instead against a randomly- or intelligently-selected subset of the archive of some fixed size. The disadvantage is that we no longer necessarily use the archive to produce the best collaboration found so far. It is not clear what the trade-off means in real terms.

Recent literature in the area of *competitive* coevolution has also examined the usefulness of informative evaluations, though some algorithms in this literature tend, quite naturally, to be designed only for competition. Still, with the two fields reaching common ground here and in other areas, the time is ripe to better identify those features the fields hold in common in order to enable further cross-pollination. pCCEA's inspiration is a good example of this.

In this paper we presented the iCCEA archive algorithm and showed that it performed well against other CCEA algorithms in basic test problems. It is unknown at this time whether or not this will scale to complex domains. But we believe archive methods may go a long way towards countering the loss of information inherent in cooperative coevolution, and in doing so will, with any luck, help CCEAs live up to their promise.

5. REFERENCES

[1] A. Bucci and J. Pollack. On identifying global optima in cooperative coevolution. In Hans-Georg Beyer et al. [4], pages 539–544.

[2] L. Bull. Evolutionary computing in multi-agent environments: Partners. In T. Back, editor, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 370–377. Morgan Kaufmann, 1997.

[3] L. Bull. Evolutionary computing in multi-agent environments: Operators. In D. W. V W Porto, N Saravanan and A. E. Eiben, editors, *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, pages 43–52. Springer Verlag, 1998.

[4] Hans-Georg Beyer et al., editor. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2005*. ACM, 2005.

[5] P. Husbands and F. Mill. Simulated coevolution as the mechanism for emergent planning and scheduling. In R. Belew and L. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 264–270. Morgan Kaufmann, 1991.

[6] T. Jansen and R. P. Wiegand. The cooperative coevolutionary (1+1) ea. *Evolutionary Computation*, 12(4):405–434, 2004.

[7] E. Lehmann. *Nonparametrics: Statistical Methods Based on Ranks*. McGraw-Hill, 1975.

[8] S. Luke. ECJ 13: A Java EC research system. Available at http://cs.gmu.edu/∼eclab/projects/ecj/, 2005.

[9] D. E. Moriarty. *Symbiotic Evolution of Neural Networks in Sequential Decision Tasks*. PhD thesis, Department of Computer Science, University of Texas at Austin, 1997.

[10] L. Panait and S. Luke. Time-dependent collaboration schemes for cooperative coevolutionary algorithms. In *Proceedings of the 2005 AAAI Fall Symposium on Coevolutionary and Coadaptive Systems*, 2005.

[11] L. Panait, R. P. Wiegand, and S. Luke. Improving coevolutionary search for optimal multiagent behaviors. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 653–658, Acapulco, Mexico, 2003. Morgan Kaufmann.

[12] L. Panait, R. P. Wiegand, and S. Luke. A sensitivity analysis of a cooperative coevolutionary algorithm biased for optimization. In Kalyanmoy Deb et al., editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2004*, page (to appear). Springer, 2004.

[13] L. Panait, R. P. Wiegand, and S. Luke. A visual demonstration of convergence properties of cooperative coevolution. In *Parallel Problem Solving from Nature — PPSN-2004*. Springer, 2004.

[14] E. Popovici and K. D. Jong. Understanding cooperative co-evolutionary dynamics via simple fitness landscapes. In Hans-Georg Beyer et al. [4], pages 507–514.

[15] M. Potter. *The Design and Analysis of a Computational Model of Cooperative CoEvolution*. PhD thesis, George Mason University, Fairfax, Virginia, 1997.

[16] R. P. Wiegand. *Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, Department of Computer Science, George Mason University, 2003.

[17] R. P. Wiegand, W. Liles, and K. De Jong. An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. In L. Spector, E. D. Goodman, A. Wu, W. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2001*, pages 1235–1242. Morgan Kaufmann, 2001.

[18] R. P. Wiegand and J. Sarma. Spatial embedding and loss of gradient in cooperative coevolutionary algorithms. In X. Yao, E. Burke, J. A. Lozano, J. Smith, J. J. Merelo Guervós, J. A. Bullinaria, J. Rowe, P. Tino, A. Kaban, and H.-P. Schwefel, editors, *Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, pages 912–922. Springer-Verlag, 2004.