1-1-2012

A Multimodal Problem for Competitive Coevolution

Philip Hingston

Tirtha Ranjeet
Edith Cowan University

Chiou Peng Lam
Edith Cowan University

Martin Masek
Edith Cowan University

10.1007/978-3-642-35101-3_29

A Multimodal Problem for Competitive Coevolution

Philip Hingston, Tirtha Ranjeet, Chiou Peng Lam, and Martin Masek

Edith Cowan University, 2 Bradford St, Mt Lawley 6050 Western Australia
p.hingston@ecu.edu.au

Abstract. Coevolutionary algorithms are a special kind of evolutionary algorithm with advantages in solving certain specific kinds of problems. In particular, competitive coevolutionary algorithms can be used to study problems in which two sides compete against each other and must choose a suitable strategy. Often these problems are multimodal — there is more than one strong strategy for each side. In this paper, we introduce a scalable multimodal test problem for competitive coevolution, and use it to investigate the effectiveness of some common coevolutionary algorithm enhancement techniques.

Keywords: coevolution, multimodal, diversity

1 Introduction

Competitive coevolutionary algorithms are an important class of evolutionary algorithm, in which there is no externally defined objective fitness function. Instead, fitness is defined in a relative way, based on interactions between several coevolving populations. For this reason, competitive coevolutionary algorithms can suffer convergence "pathologies", and techniques have been developed to address these. In this paper, we focus on multimodality in coevolution, a problem feature that is known to cause convergence problems in evolutionary algorithms.

Coevolutionary algorithms may be either cooperative, in which members of each population combine to solve a problem, or competitive, in which members of each population compete against each other. One class of problem for which coevolutionary algorithms seem especially suited is the problem of determining good strategies for the opposing parties in an adversarial situation. There is one population for each party, in which each member of the population represents a possible strategy for that party. The relative fitness of each strategy in the population depends on the outcomes of conflicts with strategies from the other population(s). Examples of problems that can be approached in this way are games, negotiations and tactical planning. Often these problems appear to be multimodal, i.e. there is more than one strong strategy for each side.

In order to study the effects of multimodality in coevolution, we introduce a scalable multimodal test problem, and use it to investigate the effectiveness of some common coevolutionary algorithm enhancement techniques in improving an algorithm's ability to solve multimodal problems.

In the next section, we briefly review related work, before introducing our test problem. We then describe a simple coevolutionary algorithm and some commonly used enhancement techniques. In the following section, we describe our experiments, in which we test the simple version of the algorithm as well as variations that use these enhancements. Finally, we present a series of plots summarising the results of our experimentation and draw our conclusions.

2 Related Work

With regard to evolutionary algorithms, multimodality has long been recognised as an important issue, and something that often occurs in real world problems. Accordingly, there have been many studies testing various evolutionary algorithms on a range of multimodal test problems (e.g. [9, 17, 13, 22, 18, 28, 29]). Techniques have been developed to enhance evolutionary algorithms for multimodal problems, such as crowding [25], fitness sharing [23], derating [2] and speciation [15]. However, we have been unable to locate any similar work on multimodal test problems for competitive coevolution, or on testing the effectiveness of these special techniques in the context of competitive coevolution.

Coevolutionary algorithms have been used to solve multimodal function optimisation problems (e.g. [10, 16, 27]), generally by subdividing the problem, assigning subpopulations to different subproblems. Our interest here is different – there is no external objective function to optimise, instead, the multimodality arises from the interaction between two competing, coevolving populations. There are many examples of competitive coevolution being used to solve such problems, for example in game playing (e.g. [20, 12, 5]) and red teaming (e.g. [24, 14]), but we have not located any work specifically addressing multimodality in these applications.

3 A Multimodal Test Problem

In this section, we introduce a multimodal test problem for coevolutionary algorithms with two competing sides. What does multimodality mean in an adversarial problem? Intuitively, the idea is that a problem is multimodal if there is more than one strong strategy for each side, but how can this idea be operationalised? For an evolutionary algorithm, multimodalty means that there is more than one local optimum in the fitness landscape. But in a coevolutionary algorithm, fitness landscapes are constantly changing as the compositions of the populations change.

For the purposes of this study, we replace the usual fitness landscape with what we will call a *generalisation landscape*. The *generalisation performance* of a solution is based on its expected performance against a randomly selected opponent. This notion has been formalised by Chong et al. [6, 7], who proposed a suitable set of related measures for generalisation performance, and provided methods to estimate the values of these measures. We discuss these measures in more detail in Section 5. A generalisation landscape is then defined as the

surface generated by mapping generalisation performance over a search space. We will consider a problem to be multimodal if its generalisation surface has multiple local optima.

We will call our multimodal test problem an *n-peaks* problem, as there are n equally good strategies (corresponding to n peaks in the generalisation landscape) for each side. The challenge for a coevolutionary algorithm is to locate as many of these peaks as possible.

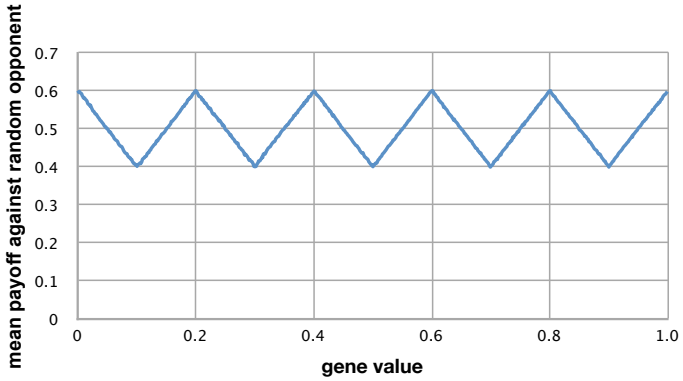


Fig. 1. Mean payoffs against random opponents for solutions to the 5-peaks problem with H= 1 and L = 1. An individual near 0, for example, will get a payoff of H (i.e. 1) against most opponents in the first interval (0 to 0.2), the third interval (0.4 to 0.6), and the fifth interval (0.8 to 1.0), and a payoff of L (i.e. also 1) against only a few opponents, in the second and fourth intervals, giving a mean payoff of nearly 0.6.

The problem is symmetric (the domain and task are the same for both sides). The domain for each side is the interval $[0, 1]$. The problem is parameterised by n, and by two payoff values $H > 0$ and $L > 0$. When two individuals x and y compete, the outcome is determined as in Equations (1)–(6).

$$i_x = \lfloor (x \times n) \rfloor \tag{1}$$

$$i_y = \lfloor (y \times n) \rfloor \tag{2}$$

$$v_x = |0.5 - (x \times n) + i_x| \tag{3}$$

$$v_y = |0.5 - (y \times n) + i_y| \tag{4}$$

$$gap = \mathrm{mod}(i_x - i_y, n) \tag{5}$$

$$score(x, y) = \begin{cases} H, \, if & gap\%2 = 0 \; and \; v_x > v_y \\ L, \, if & gap\%2 = 1 \; and \; v_x < v_y \\ 0 \quad otherwise & \end{cases} \tag{6}$$

Intuitively, we picture the domain as divided into n equal intervals. Figure 1 illustrates the case $n = 5$, $H = L = 1$. When x and y compete, the outcome

depends on which intervals they belong to (Equations 1 and 2), and on the distances from the centres of their intervals (Equations 3 and 4). If x and y are in the same interval, then x gets a payoff of H if it is *further from* the centre of the interval than y is (otherwise 0). If y is in the next interval to the right of x, then x gets a payoff of L if it is *closer to* the centre of its interval than y is to the centre of its interval. For this purpose, the "next interval to the right" of the rightmost interval is considered to be the leftmost interval - i.e. the domain wraps around. If y is two intervals to the right of x, then x gets a payoff if it is furthest from the centre of its interval. This pattern continues, with wrapping if necessary, so that the domain is actually circular, rather than linear. If y is an even number of intervals to the right of x, then it is good for x to be nearer its boundary (for a payoff of H), while if y is an odd number of intervals to the right, then it is good for x to be nearer the centre of its interval (for a payoff of L).

In the case of the problem in Figure 1, those individuals close to the interval boundaries get a high payoff against about 60% of opponents randomly selected from the domain (for an average payoff of 0.6), while those near the middle of their interval only get a high payoff against about 40% of opponents (average payoff 0.4). Thus, there are n local optima or peaks in the generalisation landscape (counting 0 and 1 as the same individual, so that there is half a peak near 0 and the other half of it is near 1).

Although, in this paper, we study only the 5-peaks problem in Figure 1, the picture is similar for other values of n. By setting the values of H and L, the difference between peak and trough values can be manipulated. By changing the definition slghtly and using more than one H and/or L value, the heights of individual peaks could also be controlled. It is also straightforward to extend the idea to higher dimensions, by subdividing a hypercube into cells, and using a kind of Manhatten distance between cells in place of the *gap* value.

4 Algorithm and Variations

In order to illustrate the difficulties posed by multimodality, we carried out experiments to test the performance of a simple competitive coevolutionary algorithm, along with some popular variations, on an *n-peaks* problem. In this section we describe the algorithm and variations that we used.

4.1 CEAN - a naïve coevolutionary algorithm

As a base case, we use a simple, naïve, competitive coevolutionary algorithm which we call *CEAN*. We then define variations on *CEAN* which include a fitness sharing mechanism, or a Hall of Fame, or both, and we also vary the mutation rate. Algorithm 1 describes the algorithm in pseudocode. Note that we have not included crossover (but it could easily be added) – we don't use crossover here because the genome for our problem is a single real number. The parameter μ is the mutation rate and the procedure *Mutate* mutates an individual population

member. The procedure *Select* selects one individual from a population, based on the fitness values of the population members. Finally, the procedure *CalculateFitness* assigns fitness values to the members of both populations, based on competition between members of the two populations.

```
   Input: Two initial populations P_1^0 and P_2^0
   Output: Two final populations P_1^f and P_2^f
 1 begin
 2     t ← 0;
 3     while t < f do
 4         CalculateFitness(P_1^t, P_2^t);
 5         for i ∈ 1..2 do
 6             P_i^{t+1} ← {};
 7             while P_i^{t+1} is not full do
 8                 s ← Select(P_i^{t+1});
 9                 with probability μ, s ← Mutate(s);
10                 P_i^{t+1} ← P_i^{t+1} ∪ {s};
11             end
12         end
13         t ← t + 1;
14     end
15 end
```

Algorithm 1: CEAN

For the naïve algorithm, *CalculateFitness* assigns a fitness value for each population member as the mean payoff achieved in competition with the members of the other population. This is presented in pseudocode in Algorithm 2.

4.2 Variants

Even when solving unimodal problems, we know that coevolutionary algorithms often need special care to avoid coevolutionary pathologies such as cycling, loss of gradient, and so on [1, 11, 23, 8, 3]. Two common remedies are the use of an archive (to prevent evolutionary forgetting), and diversity maintenance techniques (to prevent loss of diversity). We therefore created variations on CEAN that include an archive and/or a diversity maintenance mechanism.

First let us consider diversity maintenance. A simple, explicit way to maintain diversity is to use a high mutation rate, but this also has the disadvantage of disrupting evolutionary learning in a random, uncontrolled way. Among the available implicit diversity maintenance techniques, we chose to use competitive fitness sharing [23]. This works by penalising population members that are similar to others in the population. The simple fitness of an individual is calculated in the normal way, and is then divided by a quantity called the *niche count* to determine its *shared fitness*. Selection is then carried out using shared fitness rather than simple fitness. Modifying CEAN to use this selection procedure gives an algorithm variant we call CEAFS.

```
Input: Two populations P_1 and P_2
begin
    for x ∈ P_1 do
        f ← 0;
        for y ∈ P_2 do
            f ← f + score(x, y);
        end
        fitness(x) = f / |P_2|;
    end
    for y ∈ P_2 do
        f ← 0;
        for x ∈ P_1 do
            f ← f + score(y, x);
        end
        fitness(y) = f / |P_1|;
    end
end
```

Algorithm 2: Calculating fitness for CEAN

Equations 7 and 8 are used to calculate the niche count, where x_i is the i^{th} individual in the population, and u is the genome length (so $d_{i,j}$ is the Euclidean distance between x_i and x_j). c_i is the niche count for x_i, τ is a constant that determines the shape of the sharing function, n_r is a constant (*niche radius*) and N is the population size.

$$c_i = \sum_{j=1}^{N} \begin{cases} 1 - (\frac{d_{i,j}}{n_r})^{\tau} & \text{if } d_{i,j} \leq n_r \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$d_{i,j} = \sqrt{\sum_{m=1}^{u} (x_{i,m} - x_{j,m})^2} \tag{8}$$

As an archive mechanism, we implemented a Hall of Fame (HOF) [23]. For each population, we maintain an archive, known as a Hall of Fame, consisting of fittest individuals from each earlier generation. In this CEAHOF variant of CEAN, the fitness calculation given in Algorithm 2 is modified to calculate the average payoff of the individual in question against members of the opposing population *as well as* the members of the archive. After each generation, the fittest individual from each population is added to the archive.

Thus we have three variants: the naïve algorithm, CEAN, a variant that uses fitness sharing, CEAFS, and a variant that uses a Hall of Fame, CEAHOF. Finally, we also created a fourth variant which uses both fitness sharing and a Hall of Fame, CEACFH. In this variant, fitness values are calculated using the Hall of Fame as for CEAHOF, and then these values are adjusted to obtain shared fitness values as for CEAFS.

5 Performance Measures

One aspect of performance is generalisation performance - that is, how well do solutions found for one side in a contest, learned via a coevolutionary algorithm, generalise to compete well against arbitrary strategies for the other side?

We use Chong et al.'s notion of generalisation performance [6, 7]. They describe their methods in terms of a population attempting to learn general solutions to perform well against a large space of test cases. They begin by defining generalization performance as the mean score of a solution in all possible test cases. This intuitively appealing definition poses several practical difficulties. First, for many problems of interest, the space of possible test cases could be very large, or even infinite, and there may be no way to compute a mean score analytically. Therefore, they propose a statistical approximation approach, in which a mean score is computed for a suitable sample of test cases. The second difficulty is to decide what probability distribution should be used over the space of test cases. In many cases, scores against "high quality" test cases might be considered more important, as they would be more likely to be chosen by an opponent, for example. Chong et al. therefore propose two different methods for sampling the space of test cases: unbiased sampling (which is purely random) and biased sampling (which favours higher quality test cases). In this paper, we use biased sampling to measure algorithm performance. The procedure to obtain a biased sample for testing is described in detail in [6, 7].

In addition, they consider several different summary values to describe the overall generalisation performance of a population of solutions: average, best and ensemble. We consider only the "best" figure. Equations 9 and 10 describe how this is calculated. Here *TestSet* is a biased sample of test cases, P is a population of solutions, and $best(P, k)$ is the set consisting of the k members of P with the highest simple fitness values. Intuitively, $best(P, k)$ is what the coevolutionary algorithm "thinks" is the best k solutions found, and $BestGP(P)$ is the generalisation performance of the best generaliser amongst them.

$$GP(x) = \frac{1}{|TestSet|} \sum_{y \in TestSet} score(x, y) \tag{9}$$

$$BestGP(P) = \max_{x \in best(P,k)} GP(x) \tag{10}$$

In the case of a multimodal problem, another relevant aspect of performance is how well an algorithm does at locating *as many peaks as possible* – that is, can the algorithm locate many different representative solutions with high generalisation performance, rather than simply *any of them*.

One way to quantify this aspect of performance is to calculate what proportion of peaks the algorithm finds on average, and how often it succeeds in finding all peaks. These are summarised by the two measures *peak ratio* (Equation 11) and *success ratio* (Equation 12) [26].

$$\text{peak ratio} = \frac{\text{total peaks found}}{\text{number of peaks} \times \text{number of runs}} \tag{11}$$

$$\text{success ratio} = \frac{\text{number of times all peaks found}}{\text{number of runs}} \tag{12}$$

In addition to these ratios, we also calculated the *circular earth mover's distance* (CEMD). CEMD was introduced by Rabin et al. [21] for comparing two histograms, and has been widely used in image processing for comparing images. Since we know the true location of the peaks in our test problem, we can construct an "ideal" distribution for an evolved population, in the form of a histogram in which all buckets not containing a peak are empty, and all buckets containing a peak contain equal numbers of solutions. We can then compare the actual histogram with this ideal histogram. *Earth mover's distance* is the minimum total amount of movement that would be required to make the two histograms identical. For the case of a circular domain, there is a simple way to calculate this, given in Equation 13.

Here the two histograms are F and G, and N is the number of buckets. F_k is the cumulative histogram derived from F, starting at bucket k and wrapping at the right hand edge of the domain (and likewise for G_k). In our experiments we used 40 equally spaced buckets for this calculation.

$$CEMD = \min_{k \in \{1,2,\ldots,N\}} \{\frac{1}{N} \sum_{i=1}^{N} |F_k[i] - G_k[i]|\} \tag{13}$$

6 Experiments and Results

To investigate the effects of diversity maintenance via fitness sharing and/or mutation, and of an archive in the form of a Hall of Fame, on the 5-peaks problem, we executed each algorithm 60 times for each mutation rate from 2.5% to 100% in steps of 2.5%. In each case we used the fixed parameter settings as in Table 1. The values of niche radius and τ were set on the basis of preliminary empirical tests. For each execution, in each generation, we recorded diversity (genotypic diversity), generalisation performance (best GP), and peak finding ability (CEMD, peak ratio and success ratio).

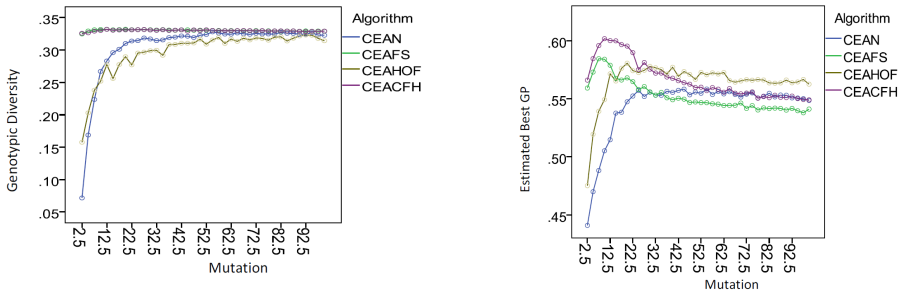
Table 1. Fixed Algorithm Parameters

Parameter	Value
Mutation	Gaussian, with wrapping, $\sigma = 0.1$
Selection	Stochastic universal sampling
Population size	50 in each population
Generations	300
Niche radius	0.2
τ	1.0
HOF size	50

Figures 2 to 4 present the results in the form of a series of profile plots. Each data point is an average over 60 executions of the mean value for the figure in question over the final 60 generations. There is a data point for each algorithm variant and mutation rate. (Here we report data for the first side, but the problem is symmetric and the data for the other side is entirely similar, as expected.)

For example, in terms of diversity, in Figure 2(a), we see that the variants that use fitness sharing have the highest diversity, and that this diversity is not sensitive to the mutation rate (with a slight peak at a mutation rate of about 12.5%). The two variants without fitness sharing have lower diversity, increasing with mutation rate, with CEAHOF performing worst in terms of diversity.

Turning next to generalisation performance, Figure 2(b) shows the equivalent plot for best generalisation performance. For reference, assuming that both populations are reasonably diverse, generalisation performance should be in the range 0.4 to 0.6. We can see from the figure that CEAN, the naïve algorithm, is the worst performer, with a maximum of only about 0.56, for mutation rates above 27.5%. Fitness sharing improves performance to about 0.58 for CEAFS with mutation rates between about 7.5% and 12.5%, which is also the range that gives slightly better diversity with this variation. CEAHOF achieves almost the same level for mutation rates between about 17.5% and 57.5%. But the best performance is for the combined variant CEACFH, with above 0.59 for mutation rates between 5% and 12.5%.



(a) diversity (b) generalisation

Fig. 2. Profile plots of diversity and best generalisation performance (mean over the final 60 generations) versus mutation rate for each of the 4 algorithm variants.

From this, we draw a tentative conclusion that Hall of Fame provides a benefit to generalisation performance, but this may be limited by the drawback that it reduces diversity. Combining a Hall of Fame with a diversity mechanism solves the diversity loss problem, and fitness sharing is a more effective mechanism for this than simply increasing the mutation rate.

The last three plots address the question of the ability of the algorithm variants to find the multiple peaks of our multimodal problem. Figure 3 shows this in terms of the circular earth mover's distance. Recall that this measures how similar the distribution of the population is to an 'ideal' distribution, so smaller values are considered better. Clearly the minimum value is achieved by the combined variant (closely followed by the fitness sharing one), with low mutation rates of about 2.5% to 12.5%. This picture is confirmed by the results for peak ratio (Figure 4(a)) and success ratio (Figure 4(b)). The two fitness sharing variants with low mutation rates are clearly superior, with the combined variant slightly shading the plain fitness sharing variant. For mutation rates above about 35%, all the variants perform equally poorly.

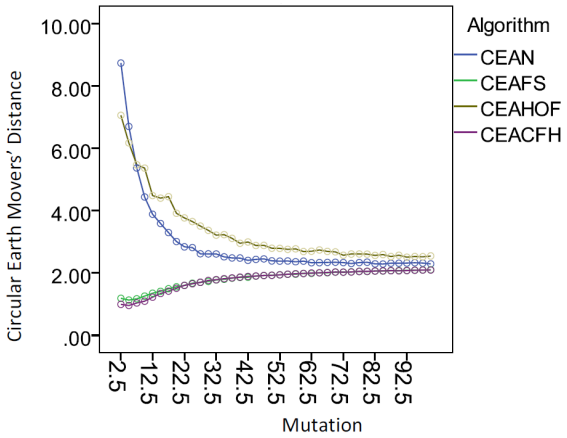
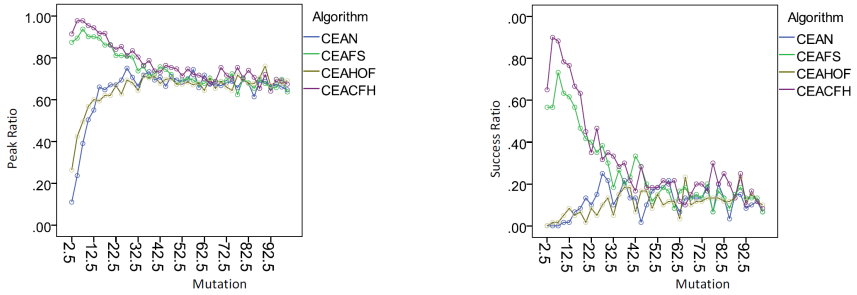


Fig. 3. Profile plot of the circular earth mover's distance (mean over the final 60 generations) versus mutation rate for each of the 4 algorithm variants.



(a) peak ratio (b) success ratio

Fig. 4. Profile plots for peak finding performance (mean over the final 60 generations) versus mutation rate for each of the 4 algorithm variants.

7 Conclusion

In this paper, we have examined the performance of a competitive coevolutionary algorithm on a multimodal problem. We created the *n-peaks* problem, a scalable multimodal test problem in which the number and amplitude of the peaks in the fitness landscape can be manipulated.

We then used an instance of the problem to test a naïve competitive coevolutionary algorithm, as well as several variants incorporating an archive (Hall of Fame) and a diversity maintenance mechanism (competitive fitness sharing), in terms of their generalisation ability, and peak finding ability. We found that, for this problem, best results in terms of both criteria were obtained with the combination of an archive and diversity maintenance, with a moderately low level of mutation.

In future work, it remains to investigate other instances of the problem with different generalisation landscapes, and in higher dimensions. In addition, other methods for handling multimodality can be tested.

References

1. R. A.Watson and J. B. Pollack. Coevolutionary dynamics in a minimal substrate. In *Proceedings of the 2001 Genetic and Evolutionary Computation Conference(GECCO'01)*, pages 702–709, 2001.
2. D. Beasley, D. R. Bull, and R. R. Martin. A sequential niche technique for multimodal function optimization. *Evolutionary Computation*, 1(2):101–125, 1993.
3. J. Cartlidge and S. Bullock. Combating coevolutionary disengagement by reducing parasite virulence. *Evolutionary Computation*, 12(2):193–222, 2004.
4. K. Chellapilla and D. Fogel. Evolving neural networks to play checkers without expert knowledge. *IEEE Transactions on Neural Networks*, 10(6):1382–1391, 1999.
5. R. Chiong and M. Kirley. Iterated N-Player games on small-world networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2011)*, Dublin, Ireland, July 2011, pages 1123-1130.
6. S. Chong, P. Tino, and X. Yao. Measuring generalisation performance in coevolutionary learning. *IEEE Transactions on Evolutionary Computation*, 12(4):479–505, 2008.
7. S. Chong, P. Tino, and X. Yao. Relationship between generalization and diversity in coevolutionary learning. *IEEE Transactions on Computational Intelligence and AI in Games*, 1(3):214–232, 2009.
8. E. D. de Jong and J. B. Pollack. Ideal evaluation from coevolution. *Evolutionary Computation*, 12(2):159–192, 2004.
9. K. Deb and D. Goldberg. An investigation of niche and species formation in genetic function optimization. In *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, 1989.
10. R. Drezewski. A co-evolutionary multi-agent system for multi-modal function optimization. In *Proceedings of the International Conference on Computational Science, Krakow, Poland*, pages 664–661, 2004.
11. S. G. Ficici. *Solution concepts in coevolutionary algorithms*. PhD thesis, Brandeis Univ., Waltham, MA, 2004.

12. D. Fogel. *Blondie24: playing at the edge of AI.* Morgan Kaufmann Publishers Inc., 2002.
13. N. Hansen and S. Kern. Evaluating the cma evolution strategy on multimodal test functions. In *Proceedings of Parallel Problem Solving from Nature*, pages 1–10, 2004.
14. P. Hingston and M. Preuss. Red teaming with coevolution. In *2011 IEEE Congress on Evolutionary Computation (CEC)*, pages 1155–1163, June 2011.
15. J.-P. Li, G. T. P. M. E. Balazs, and P. J. Clarkson. A species conserving genetic algorithm for multimodal function optimization. *Evolutionary Computation*, 10(3):207–234, 2002.
16. Y. Liu, X. Yao, and Q. Zhao. Scaling up fast evolutionary programming with cooperative coevolution. In *Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, Korea*, pages 1101–1108, 2001.
17. B. Miller and M. Shaw. Genetic algorithms with dynamic niche sharing for multimodal function optimization. In *Proceedings of IEEE International Conference on Evolutionary Computation*, 1995.
18. K. Parsopoulos and M. Vrahatis. A generator for multimodal test functions with multiple global optima. *IEEE Transactions on Evolutionary Computation*, 8(3):211–224, 2004.
19. A. Petrowski. A clearing procedure as a niching method for genetic algorithms. In *Proceedings of the 3rd IEEE International Conference on Evolutionary Computation*, pages 798–803, 1996.
20. J. Pollack and A. Blair. Co-evolution in the successful learning of backgammon strategy. *Machine Learning*, 32:225–240, 2998.
21. J. Rabin, J. Delon, and Y. Gousseau. Circular earth mover's distance for the comparison of local features. In *Proceedings of the International Conference on Pattern Recognition*, 2008.
22. J. Rönkkönen, X. Li, V. Kyrki and J. Lampinen. A framework for generating tunable test functions for multimodal optimization *Soft Computing*, 15(9): 1689-1706, 2011.
23. C. Rosin and R. Belew. New methods for competitive coevolution. *Evolutionary Computation*, 5(1):1–29, 1997.
24. C. C. Seng, C. C. Lian, L. K. M. Spencer, and O. W. S. Darren. A co-evolutionary approach for military operational analysis. In *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, GEC '09, pages 67–74, New York, NY, USA, 2009. ACM.
25. S.W.Mahfoud. Crowding and preselection revisited. In *Proceedings of Parallel Problem Solving from Nature 2*, pages 27–36, 1992.
26. R. Thomsen. Multimodal optimization using crowding-based differential evolution. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1382–1389, 2004.
27. S. Xuhua and Y. Haizhen. An exploring coevolution multi-agent system for multimodal function optimization. In *Intelligent Systems and Applications, 2009. ISA 2009. International Workshop on*, pages 1–4, May 2009.
28. T. Weise, S. Niemczyk, H. Skubch, R. Reichle and Kurt Geihs. A tunable model for multi-objective, epistatic, rugged, and neutral fitness landscapes In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008)*, Atlanta, GA, USA, July 2008, pages 795–802.
29. E. Yu and P. Suganthan. Ensemble of niching algorithms. *Information Sciences*, 180:2815–2833, 2010.