Dynamics of Co-evolutionary Learning

Hugues Juillé **Jordan B. Pollack**
Computer Science Department
Volen Center for Complex Systems
Brandeis University
Waltham, MA 02254-9110
{hugues, pollack}@cs.brandeis.edu

Abstract

Co-evolutionary learning, which involves the embedding of adaptive learning agents in a fitness environment which dynamically responds to their progress, is a potential solution for many technological chicken and egg problems, and is at the heart of several recent and surprising successes, such as Sim's artificial robot and Tesauro's backgammon player. We recently solved the two spirals problem, a difficult neural network benchmark classification problem, using the genetic programming primitives set up by [Koza, 1992]. Instead of using absolute fitness, we use a relative fitness [Angeline & Pollack, 1993] based on a competition for coverage of the data set. As the population reproduces, the fitness function driving the selection changes, and subproblem niches are opened, rather than crowded out. The solutions found by our method have a symbiotic structure which suggests that by holding niches open, crossover is better able to discover modular building blocks.

1 Introduction

Co-evolution is an ecological theory which attempts to explain how traits can evolve which are dependent between different species. In evolutionary computation however, it has been appropriated from its ecological roots to describe any iterated adaptation involving "arms-races", either between learning species or between a learner and its learning environment. Examples of co-evolutionary learning include the pioneering work by Hillis on sorting networks [Hillis, 1992], by Tesauro on self-playing Backgammon learner [Tesauro, 1992] with a recent follow up by Pollack, Blair and Land [Pollack et al., 1996], by Sims and Ray in evolving life-forms [Sims, 1994, Ray, 1992], by Angeline and Pollack on co-evolving Tic-tac-toe players [Angeline & Pollack, 1993]. In the adaptive behavior community, there is a focus developing on co-evolution in predator/prey games [Reynolds, 1994, Miller & Cliff, 1994].

Using competitive fitness in a massively parallel implementation of the genetic programming (GP) paradigm [Koza, 1992] we solved the problem of intertwined spirals, a very difficult classification benchmark from the field of neural networks. This learning problem, originated by Alexis Wieland, perhaps based on the cover of Perceptrons, has been a challenge for pattern classification algorithms and has been subject of much work in the AI community, in particular in the Neural Network field (e.g., [Lang & Witbrock, 1988, Fahlman & Lebiere, 1990, Carpenter et al., 1992]). In Neural Network classification systems, based on linear, quasi-linear, radial, or clustering basis function, the intertwined spirals problem leads to difficulty. When it is solved, the neural net solution often has a very "expansive" description of the spiral, i.e. the conjunction of many small regions, does not generalize outside the training regions, and is thus not particularly satisfying.

In this paper we compare our competitive fitness co-evolutionary approach to an absolute fitness approach to the spirals problem and find it more effective. Moreover, co-evolution in this context leads to interesting functional modularizations of the problem.

Section 2 presents a survey of the implementation of our Massively Parallel Genetic Programming (MPGP). This will help to understand the techniques that have been used in the following sections. Then, the intertwined spiral problem is described in section 3, along with its representation in the competitive fitness framework and results. Section 4 presents both theoretical and empirical analysis of our results comparing canonical and co-evolutionary optimization.

2 Massively Parallel GP

2.1 Parallel Evaluation of S-expressions

MPGP runs on a SIMD machine of 4096 processor elements (PEs), the MasPar MP-2. The individual structures that undergo adaptation in GP are represented by expression trees composed from a set of primitive functions and a set of terminals (either variables or functions of no argument). Usually, the number of functions is small, and the size of the expression trees are restricted, in order to restrict the size of the search space.

In our parallel implementation, each of the 4096 processors simulates a virtual processor. This virtual processor is a *Stack Machine* and takes the postfix representation of an S-expression as its input.

To be able to evaluate a GP expression, the following instructions are supported by the abstract machine:

- one instruction for each primitive function of the function set. At execution time, arguments for these instructions are popped from the stack into general purpose registers, the function is computed, and the result is pushed on the top of the stack.
- a `PUSH` instruction which pushes on the top of the stack the value of a terminal,
- a `IFGOTO` and a `GOTO` instruction which are necessary for branching if conditional functions are used,
- a `STOP` instruction which indicates the end of the program.

This architecture allows each PE to process efficiently a different genetic program in a MIMD-like way. The parallel interpreter of the SIMD machine reads the current postfix instruction for each virtual processor and sequentially multiplexes each instruction, *i.e*, all processors for which the current instruction is a `PUSH` become active and the instruction is performed; other processors are inactive (*idle* state). Then, the same operation is performed for each of the other instructions in the instruction set in turn. Once a `STOP` instruction is executed for a processor, that processor becomes idle, leaving the result of its evaluation on the top of the stack. When all processors have reached their `STOP` instruction, the parallel evaluation of the entire population is complete.

[Perkis, 1994] has already shown that the stack-based approach for Genetic Programming can be very efficient.

2.2 Models for Fitness Evaluation, Selection and Recombination

The MasPar MP-2 is a 2-dimensional wrap-around mesh architecture. In our implementation, the population has been modeled according to this architecture: an individual or a sub-population is assigned to each node of the mesh and, therefore, has 4 neighbors. This architecture allows us to implement different models for fitness evaluation, selection and recombination, using the kernel of the parallel GP described in the previous section.

In this paper, only a tournament style of competitive evolution has been used and compared to canonical GP. A more general presentation of the different strategies that have been implemented can be found in [Juillé & Pollack, 1996].

3 The Spiral Problem and the Competitive Evolution Paradigm

3.1 Presentation

The intertwined spiral problem consists of learning to classify points on the plane into two classes according to two intertwined spirals. The data set is composed of two sets of 97 points, on the plane between -7 and +7. These two intertwined spirals are shown as "$\times$" and "$\circ$" in figures 5 and 6.

[Koza, 1992] and [Angeline, 1995] have also investigated this problem using the Genetic Programming paradigm. We used the same setup as them to define the problem and to perform our experiments. That is, the function set is composed of: $\{+, -, *, \%, iflte, sin, cos\}$, and the terminal set is composed of: $\{x, y, \Re\}$, where $\Re$ is the ephemeral random constant.

With a population of 4096 individuals, we tried two different approaches to tackle this problem. In the first experiment, following Koza and Angeline, the fitness function was defined as the number of hits out of 194.

In the second experiment, the fitness was defined as the result of a competition among the individuals. We ignored the fact that we really knew the absolute fitness function, and set up a "game" in which only relative fitness was used as the basis for reproduction. The trivial idea would be to simply compare the absolute score of each individual and the winner would be the individual with the larger score. However, such a competition of absolute scores would simply approximate the canonical version.

Instead, we only counted a player's ability to classify those test cases which are *not* classified by its opponent. As more or less copies of a player spread through the population, their scores may rise or fall depending on how many other members of the population also "cover" the test cases. This is a form of adaptive behavior implemented dynamically in the fitness function. As a simplified view, consider a full pairwise evaluation between one weak but unique player with 25 novel hits, against 4 identical strong players all with the same 50 hits. Although they would reproduce twice as fast in an absolute fitness competition, in this modified tournament, they will only receive their 50 points for playing the weak player, who will actually receive 100! In section 4, a simplified ecological model is presented to study the dynamics of the population evolution when an absolute fitness or a competitive fitness is used to control interactions between species. We do not play all-against-all, but several rounds of a more limited tournament competition, and compute the final relative fitness of each individual as the sum of all its scores during the competition. We can of course track the absolute fitness of a population even though it is not used otherwise.

Our hypothesis is that the competitive evolution would work better because it would promote more diversity in the population, and allow subpopulations which covered different subproblems to emerge. As copies of individuals which perform well on parts of the spiral spread through the population, they will start to meet themselves in competition, and get a score of 0. This allows other individuals who may have less total hits, but cover other parts of the spiral to survive. From the recombinations between individuals of those two sub-populations one may expect the emergence of a better individual that combine the "advantages" of both.

Several approaches may be used when simulating a competitive evolution [Sims, 1994]. In this work, each generation is composed of a sequence of competition rounds in which individuals are "randomly" paired up. In fact, because of the architecture of our parallel computer which doesn't have any fast-access shared-memory, this random pairing is approximated by making all the individuals perform a "random walk" in the population. At each round, the score of individuals is the number of hits that their opponent doesn't get. At the end of each generation, individuals' fitness is calculated by summing all their scores in the competition.

Once individual fitness is evaluated, selection and recombination are performed according to a fitness proportionate rule. Details of the implementation of this model of tournament, and of selection and recombination procedures for MPGP can be found in [Juillé & Pollack, 1996].

3.2 Preliminary Results and Discussion

For the two classes of experiments, we performed 25 runs and each run was stopped after 300 generations. At each generation, 90% of the population was replaced by offsprings resulting from recombination and the remaining 10% was the result of fitness proportionate reproduction. Each individual meets 96 opponents at each generation (this number comes from the implementation of the tournament on the mesh architecture of the MasPar).

Our preliminary results concerning performance illustrate that competitive evolution outperforms the absolute fitness approach. In fact, the absolute fitness strategy works better at the beginning, with average fitness rising faster. However, this absolute fitness paradigm improves its current solution very slowly after its initial burst of optimization, and is ultimately outperformed by the competitive evolution.

There are multiple competing explanations for this performance gap. It may be that the absolute fitness is simply converging prematurely. It may be that the competitive fitness system benefits from more diversity. In sections below we analyze and try to understand these differences.

```
(sin (% (iflte (- (- (- (* _A _A)
                        (sin
                         (% (iflte  -0.52381
                                     _B
                                     (sin -0.33333)
                                     -0.33333)
                             -0.33333)))
                     (* _B _B))
                  (% _A (% -0.33333 _A)))
               -0.80952
               _B
               (sin
                (% (% _A
                      (- (cos
                          (sin
                           (* (cos
                               (sin -0.52381))
                              (% _B
                                 (% _A
                                    (- (cos
                                        -0.33333)
                                       0.04762))))))
                         0.04762))
                   (sin (sin -0.33333)))))
        -0.33333))
```

Figure 1: A 52-atom S-expression scoring 194 for the intertwined spiral problem.

If $(4 * x^2 - y^2) \; < \; 0.0$ **then**
  return $(sin(-3.0 * y))$;
else
  return $\left(sin(\frac{0.3214 * x}{0.04762 - cos(sin(\frac{y}{x} * 0.7874))}) \right)$;
endif

Figure 2: Interpretation of the solution for the intertwined spiral problem.

However, only a few runs of competitive fitness have provided us with a perfect (194 hits) solution for the intertwined spiral problem within 300 generations. We harvested some of the perfect classification solutions; One of the shortest of these S-expressions has 52 atoms and is shown in figure 1.

Because of the relatively small size of this result we were able to analyze it and simplify it mathematically, by collapsing constant calculations, removing insignificant digits, algebraic simplification, and elimination of redundant "introns". This analysis resulted in the conditional function presented in figure 2.

Basically, this solution splits the geometric plane into two domains and a different function is used for each domain. Figure 3 displays the $4x^2 - y^2$ function which multiplexes the two other functions, shown in figure 4, to create the spiral.

The resulting function is shown in figure 5, which plots

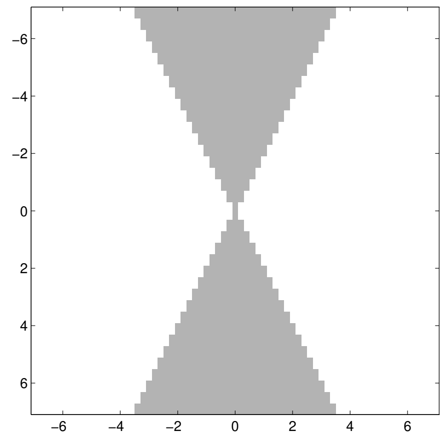

Figure 3: $4x^2 - y^2 < 0$, used to divide the plane into two domains.

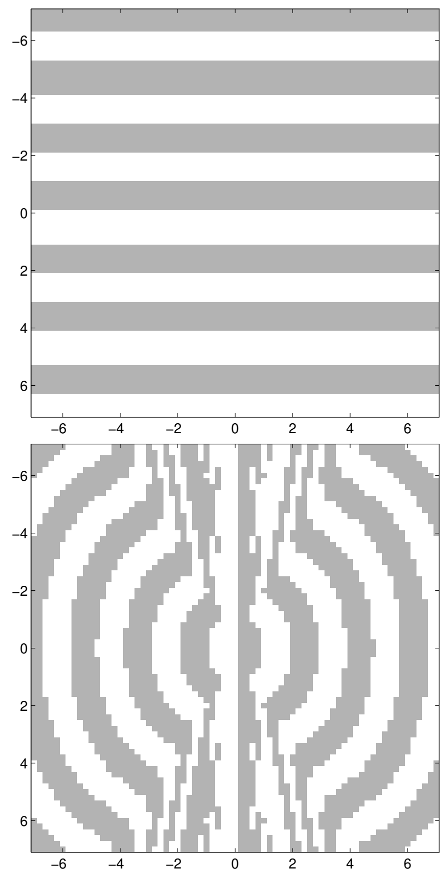

Figure 4: $sin(-3y)$ and the other function which are selectively added to make a spiral.

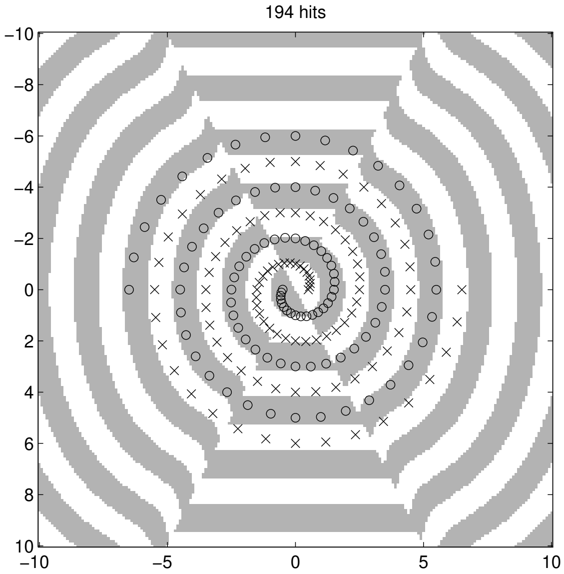

Figure 5: Perfect score generalizing classification of the two intertwined spirals.

the function (above/below 0) along with the training data on the range -10 to 10. Although it does not form a perfect spiral, it does continue to simulate a spiral way outside the original training range. In another set of experiments (limited to 100 generations) another perfect solution has been discovered (presented in figure 6). The S-expression representing this solution is composed of 161 atoms.

Furthermore, we believe that compared to neural network solutions, which are often the composition of hundreds of clusters or decision boundaries, and some of the GP solutions shown by Koza, ours is the most perspicacious to date. The fact that the spiral is composed of a symbiosis of two (or more) functions which cover separate parts of the data supports the hypothesis that the relative fitness competitive evolution strategy can be more effective than an absolute fitness function. This idea is supported by the analysis presented in the following section.

4 Co-evolution and the Dynamics of Learning

4.1 *A Theoretical model for Absolute and Relative Fitness*

To support the idea that competitive evolution allows subpopulations which cover different part of the problem to survive, contrary to an absolute fitness driven evolution, we propose the following analysis. The two-intertwined problem is a classification problem. Therefore, it can be seen as a set of test cases and the popu-

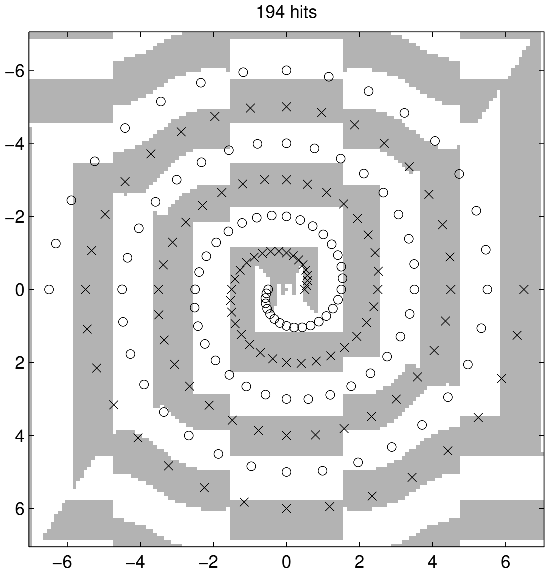

Figure 6: Another perfect score classification of the two intertwined spirals.

lation can be split up into groups (or clusters) in which individuals would cover exactly the same test cases. For the sake of clarity, let us formalize this idea. First, let us define the following terms:

- n: number of test cases,
- m: number of groups (or clusters) that compose the population,
- t_i: i^{th} test case,
- G_j: j^{th} group of individuals,
- $s_j(t)$: size (number of individuals) of group G_j at time t,
- $T(G_j)$: returns a list of booleans of size n in which the k^{th} entry indicates whether the test case t_k is covered by individuals in group G_j,
- $\mathcal{B}$: matrix whose rows are the $T(G_j)$s.

$$\mathcal{B} = \begin{pmatrix} T(G_1) \\ \vdots \\ T(G_m) \end{pmatrix}$$

Each entry $b_{i,j}$ of $\mathcal{B}$ is a 1 (*true*) if the test case t_j is covered by the group G_i, and 0 (*false*) otherwise.

For the following, let us consider an example:

- $n = 10$,
- $m = 5$,
- $T(G_1) = (0,1,1,1,0,1,0,1,0,1)$,
  $T(G_2) = (1,0,0,0,1,0,0,0,1,0)$,
  $T(G_3) = (0,1,0,1,0,0,1,1,0,1)$,
  $T(G_4) = (0,0,1,1,0,0,0,0,1,0)$,
  $T(G_5) = (0,1,0,1,1,0,1,0,0,1)$

Now, we can define the $(m \times m)$ square matrix $\mathcal{A}$ for which each entry $a_{i,j}$ equals the number of test cases correctly classified by group G_i but that group G_j doesn't. Thus, each entry of $\mathcal{A}$ is defined as follows:

$$a_{i,j} = \sum_{l=1}^{n} (b_{i,l} \wedge \neg b_{j,l})$$

With our example, $\mathcal{A}$ equals:

$$\mathcal{A} = \begin{pmatrix} 0 & 6 & 2 & 4 & 3 \\ 3 & 0 & 3 & 2 & 2 \\ 1 & 5 & 0 & 4 & 1 \\ 1 & 2 & 2 & 0 & 2 \\ 2 & 4 & 1 & 4 & 0 \end{pmatrix}$$

Now, we can define the fitness function for the two cases of study:

- *absolute fitness* for an individual of group G_j:

  $$f_a(j) = \sum_{l=1}^{n} b_{jl}$$

  For our example:
  $f_a(1) = 6; f_a(2) = 3; f_a(3) = 5; f_a(4) = 3; f_a(5) = 5$

- *relative fitness* for an individual of group G_j:

  $$f_r(j) = \sum_{l=1}^{m} (s_l(t) \times a_{j,l})$$

  According to this definition, each individual competes once against all other individuals in the population. In our experiments, we only approximate this by making each individual compete against a sample of the population.

For the sake of simplicity, we assume there are no recombination between individuals but only fitness proportionate reproduction. Indeed, what we want to show with this simplified model is that subpopulations that cover different test cases survive when competitive evolution is involved. Therefore, we want to study the dynamics of the evolution of group size with time. A simple rule for fitness proportionate reproduction, similar to the one used by [Lindgren, 1992] to model population evolution, gives us:

$$s_j(t+1) = s_j(t) \times \left(1 + \alpha \times \frac{f(j) - \overline{f}}{\overline{f}}\right)$$

where:

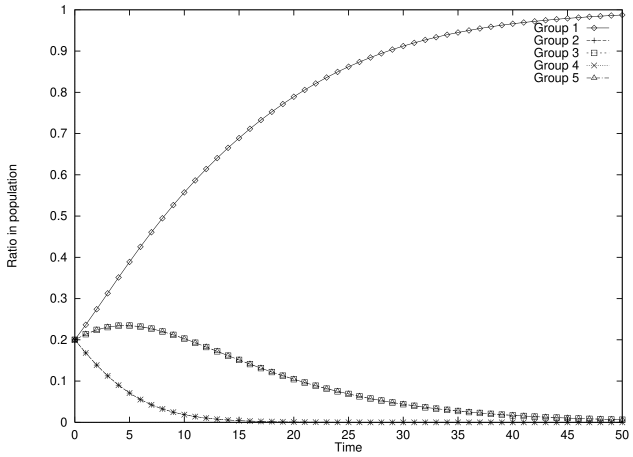

Figure 7: Evolution of the ratio for each group in the population in the case of an absolute fitness.

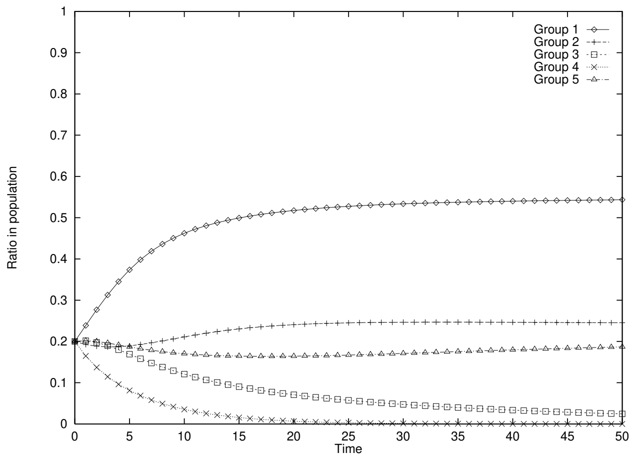

Figure 8: Evolution of the ratio for each group in the population in the case of a relative fitness.

- α is a parameter that controls the speed of the simulated evolution,
- $f(j)$ is the fitness. According to the case of study, it is replaced by $f_a(j)$ or $f_r(j)$.
- $\overline{f}$ is the average of the fitness.

A normalization step for $s_j(t+1)$ is then performed in order to keep a constant population size. If $\alpha = 1$, we get the well known expression for fitness proportionate reproduction:

$$s_j(t+1) = s_j(t) \times \frac{f(j)}{\overline{f}}$$

The graphical results of the evolution of the ratio for each group in the population for the two models of evolution are presented in figure 7 and figure 8. For our analysis, all groups have the same size at $t = 0$, and we took $\alpha = 0.5$. One can see that in the case of the absolute fitness, all the population is overcome by the first group which has the largest absolute fitness ($f_a(1) = 6$). The curves for the groups 2 and 4, and for the groups 3 and 5 overlap in this figure. On the contrary, in the case of the competitive evolution, once stability is reached, the first group takes a little more than 50% of the population and groups 2 and 5 around 20%. Group 4 disappears very quickly and group 3 takes only a tiny part of the population. It is possible to prove that these ratios at the equilibrium are independent of the initial size of the different groups (at the condition that no group has null size) and of the value of the non-null parameter α.

The aim of this analysis is to show that competitive evolution allows different subpopulation to survive, contrary to the canonical model of evolution, therefore keeping more diversity in the population. We also believe that in the case of the intertwined spiral problem, recombination of individuals from different subpopulations are at the origin of new solutions that cover some part of the problem that were specific to each of the two subpopulations. This idea is confirmed by the results of experiments presented in section 4.2.

4.2 Diversity and Useful Recombination

We realized that our co-evolution system was in fact operating to prevent convergence by increasing diversity in the population. This can also be done simply by changing parameters in a canonical genetic optimization task. So we performed more experiments to compare the evolutionary and the co-evolutionary approaches. In particular, we tried three different settings for the parameters that control the convergence rate of the search procedure and therefore control the decrease of diversity in the population. More precisely, the *normalized fitness* of individuals, which is used to control the selection process, is computed as follows:

$$raw\_fitness = score$$

$$standardized\_fitness = (max(score) - raw\_fitness) \times \alpha$$

$$adjusted\_fitness = \frac{1}{1 + standardized\_fitness}$$

$$normalized\_fitness = \frac{adjusted\_fitness}{\sum_{population} adjusted\_fitness}$$

The parameter α controls the range of the standardized fitness and therefore the distribution of the normalized fitness. Indeed, if α decreases, the difference between fit and less fit individuals for the adjusted fitness, and therefore for the normalized fitness, decreases, making the convergence slower. However, the raw fitness doesn't represent the same measure for the absolute and relative fitness approaches (the number of hits for the former and the number of hits not covered by each of the opponents for the later). Thus, the only way to compare the two methods is to try a large range for the parameter α. For

Absolute fitness	Relative fitness
$\alpha = 1.0$	$\alpha = 0.2$
$\alpha = 0.2$	$\alpha = 0.05$
$\alpha = 1.0$ $(standardized\_fitness)^2$	$\alpha = 0.01$

Table 1: Parameter setting for the experiments.

the absolute fitness, we tried two values for α and we performed one experiment for which the standardized fitness was squared, altering in another way the distribution of the normalized fitness. For the relative fitness experiments, three values were tested for α. Table 1 presents the different parameter settings for our experiments.

The tournament-like competition was implemented to be a realistic model of competition. In particular, it could be used to co-evolve game strategies [Angeline & Pollack, 1993]. However, in the case of an inductive learning problem like the intertwined spirals, the set of test cases is well-defined, fixed and of manageable size. Therefore, it is possible to efficiently implement an *all vs. all* competition as follows:

- For each test case, compute the number $\overline{S_i}$ of individuals that do *not* classify it correctly. This can be implemented in $O(\log n)$ on a parallel machine, using a form of divide-and-conquer to perform addition (*reduce* operator). Then, $\overline{S_i}$ is made available to all the individuals.

- The relative fitness of an individual j is then:

  $$\sum_{i=1}^{\#\ test\ cases} b_{j,i} \times \overline{S_i}$$

  where: $b_{j,i}$ equals 1 if individual j classifies correctly the i^{th} test case, and 0 otherwise.

The result of this process is the same as if each individual would have compete against all the other individuals in the population.

We addressed the issue of all vs. all competition here for the following reason. For most problems, the tournament competition is of more practical interest than all vs. all competition. This is the case in particular when the set of test cases is too large to allow an exhaustive evaluation of individuals with the whole set (*e.g.*, when each individual represents a game strategy). Thus, it is interesting to estimate how accurately the tournament competition approximates the all vs. all competition.

We limited the number of generations to 150. The results are presented in figure 9 where each curve corresponds to the average over 25 runs with the same value for the parameters.

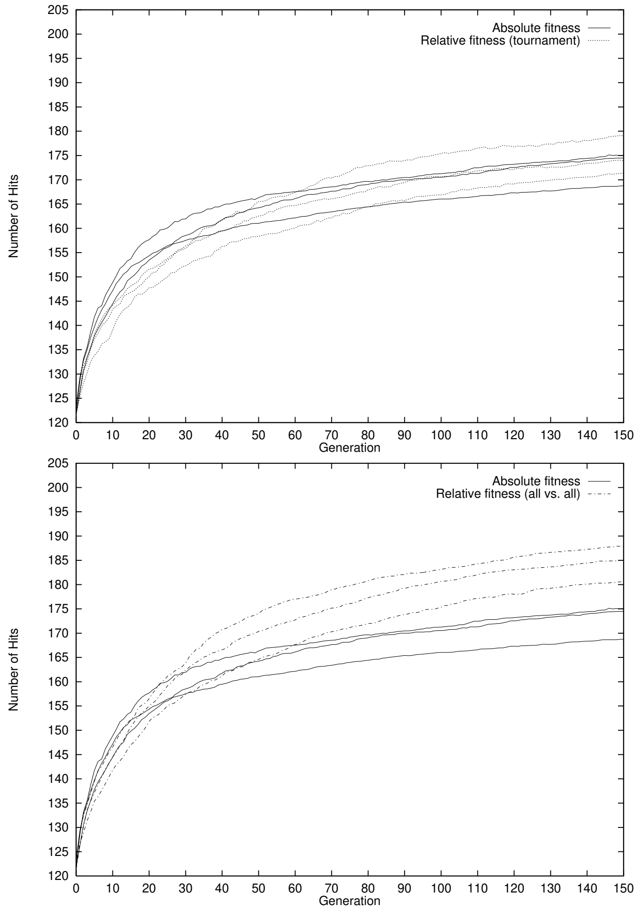

Figure 9: Absolute fitness versus tournament-like competition (top) and absolute fitness against all vs. all competition (bottom) for the intertwined spiral problem, for different parameter setting. Each curve is the average of the best individual, at each generation, over 25 runs.

The first observation is that all vs. all competition clearly outperforms canonical evolution. After 150 generations, one of the three parameter settings resulted in a perfect solution for 5 out of the 25 runs. Those runs where extended up to 200 generations and resulted in 10 perfect solutions out of the 25 runs. None of our experiments with canonical evolution resulted in a perfect solution. In the case of tournament-like competition, it is more difficult to conclude, even if a slight advantage might be given to this form of co-evolution. Only 2 out of the 75 runs resulted in a perfect solution before 150 generations.

Those experiments show that co-evolution offers a different approach to tackle the learning task and potentially works better than canonical evolution. The tournament-like competition uses only 96 rounds (compared to 4096 rounds for all vs. all) which might be seen as too small. It is difficult to extrapolate the number of rounds to achieve a given accuracy for the approximation

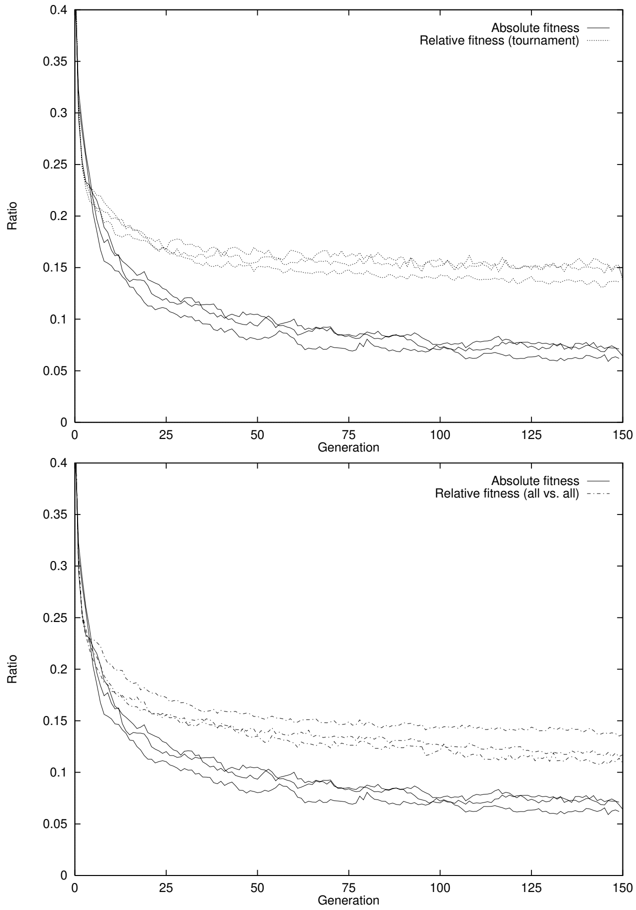

Figure 10: Comparison between absolute fitness and tournament-like competition (top) and between absolute fitness and all vs. all competition (bottom) for the evolution of the ratio of offsprings in the population that cover more test cases than their parents, for different parameter settings.

of all vs. all but it seems that 96 rounds is a good compromise regarding computer resource when the method presented in this paper to perform all vs. all cannot be applied.

For the same experiments, we also observed the evolution of another measure in order to show that learning is more efficient in the case of co-evolution. Rosca [Rosca & Ballard, 1996] defined *differential fitness* as a measure of the fitness improvement in the population. He defined this measure for offspring i as follows:

$$\begin{aligned} &\mathit{Differential\_Fitness}(i) = \\ &\qquad \mathit{Standard\_Fitness}(i) - \\ &\qquad \mathit{min}_{p \in Parents(i)} \{\mathit{Standard\_Fitness}(p)\} \end{aligned}$$

For an heuristic reason [Rosca & Ballard, 1996], the min of the parents was taken to define the differential fitness. However, in our case, one is more interested in a measure that would indicate an improvement of offsprings over both parents. Therefore, the max of the parents has been taken to define our differential fitness. Moreover, in order to compare both approaches, we defined $\mathit{Standard\_Fitness}$ as the number of hits. This definition is used only for the evaluation of the differential fitness and is independent of the previously defined standard fitness used to evaluate the normalized fitness. Thus, the differential fitness is positive only for those offsprings that cover more test cases than both parents. Figure 10 presents the evolution of the ratio of offsprings in the population for which this new differential fitness is positive. Each curve represents the average over 25 runs. Data were collected while performing the previous experiments.

One can see that the average ratio seems to be almost independent of the parameter setting for relative as well as absolute fitness. However, this ratio for co-evolution is at least 50% larger than for canonical evolution after the first 50 generations. This difference means that the probability that an offspring be better than both its parents, *i.e.* the probability of *useful* recombination or the probability of the exchange of building blocks, is significantly greater in the case of a relative fitness than for an absolute fitness. Indeed, since co-evolution maintains niches that cover different subsets of the test cases and that the relative fitness favours the covering of all the test cases by those subsets, it is more likely that recombination will occur between parents that cover different test cases. This is not the case for the absolute fitness which doesn't have this kind of bias. Moreover, *diversity* in the population is not the main reason to explain this difference since the increase of diversity for experiments with absolute fitness doesn't change significantly the ratio. This clearly shows that our model of co-evolution favors useful recombination and, ultimately, is more likely to lead to a better solution than canonical evolution.

5 Conclusion

Experiments presented in this paper show that the classification procedure for a challenging problem (namely, the intertwined spiral problem) can be significantly improved by using a relative fitness rather than an absolute fitness approach. As we look at the All versus All tournament, which is not plausible for most tasks, we can see that the competitive fitness model of co-evolution approximates fitness sharing [Goldberg & Richardson, 1987], which helps prevent premature convergence by increasing diversity in a population. Rosin and Belew's [Rosin & Belew, 1995] work on fitness sharing demonstrates this connection as well.

In summary, we achieve co-evolution within a single species by using a heuristic which dynamically adapts the fitness function to help the population cover all the test cases, rather than giving a great reproductive advantage to those individuals which perform well on the whole problem. Indeed, by giving the minority an advantage in the competition, one can expect the emergence of novel niches which are held open until they are incorporated into more successful solutions.

In evolutionary terms, the set of traits which lead to increased relative fitness of individuals or species cannot be pre-determined in a finite list as can be done for the intertwined spirals problem. However, the traits which arise over evolutionary time scales, such as strength, speed, olfaction, vision, and even cognition, are simply a competitive advantage until competitors acquire those traits or find counter-measures. Thus co-evolution, which may be viewed as an adaptive behavior on evolutionary time scales between species, can also be useful for problems for which absolute fitness is known on individual species, such as a population of genetic programs.

References

[Angeline, 1995] Angeline, P. J. (1995). Two self-adaptive crossover operations for genetic programming. In *Advances in Genetic Programming II*. MIT Press.

[Angeline & Pollack, 1993] Angeline, P. J. & Pollack, J. B. (1993). Competitive environments evolve better solutions for complex tasks. In *The Fifth International Conference on Genetic Algorithms*, pp. 264–270. Morgan Kaufmann.

[Carpenter et al., 1992] Carpenter, G., Grossberg, S., Markuzon, N., Reynolds, J., & Rosen, D. (1992). Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, 3:698–713.

[Fahlman & Lebiere, 1990] Fahlman, S. E. & Lebiere, C. (1990). The cascade-correlation learning architecture. In Touretzky (Ed.), *Advances in Neural Information Processing Systems 2*. Morgan Kauffman.

[Goldberg & Richardson, 1987] Goldberg, D. E. & Richardson, J. J. (1987). Genetic algorithms with sharing for multimodal function optimization. In Grefenstette, J. J. (Ed.), *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pp. 41–49. Lawrence Erlbaum Associates.

[Hillis, 1992] Hillis, W. D. (1992). Co-evolving parasites improve simulated evolution as an optimization procedure. In Langton, C. et al. (Eds.), *Artificial Life II*, pp. 313–324. Addison Wesley.

[Juillé & Pollack, 1996] Juillé, H. & Pollack, J. B. (1996). Massively parallel genetic programming. In Angeline & Kinnear (Eds.), *Advances in Genetic Programming II*. MIT Press.

[Koza, 1992] Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press.

[Lang & Witbrock, 1988] Lang, K. J. & Witbrock, M. J. (1988). Learning to tell two spirals apart. In *Proceedings of the 1988 Connectionist Summer Schools*. Morgan Kaufmann.

[Lindgren, 1992] Lindgren, K. (1992). Evolutionary phenomena in simple dynamics. In Langton, C. et al. (Eds.), *Artificial Life II*, pp. 295–312. Addison Wesley.

[Miller & Cliff, 1994] Miller, G. F. & Cliff, D. (1994). Protean behavior in dynamic games. In Cliff, D., Husbands, P., Meyer, J., & Wilson, S. (Eds.), *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*. MIT Press.

[Perkis, 1994] Perkis, T. (1994). Stack-based genetic programming. In *Proceedings of the 1994 IEEE World Congress on Computational Intelligence*. IEEE Press.

[Pollack et al., 1996] Pollack, J. B., Blair, A. D., & Land, M. (1996). Coevolution of a backgammon player. To appear in the proceedings of the Fifth Artificial Life Conference.

[Ray, 1992] Ray, T. S. (1992). An approach to the synthesis of life. In Langton, C. et al. (Eds.), *Artificial Life II*, pp. 371–408. Addison Wesley.

[Reynolds, 1994] Reynolds, C. W. (1994). Competition, coevolution, and the game of tag. In *Proceedings of the Fourth Artificial Life Conference*. MIT Press.

[Rosca & Ballard, 1996] Rosca, J. P. & Ballard, D. H. (1996). Discovery of subroutines in genetic programming. In Angeline & Kinnear (Eds.), *Advances in Genetic Programming II*. MIT Press.

[Rosin & Belew, 1995] Rosin, C. D. & Belew, R. K. (1995). Methods for competitive co-evolution: Finding opponents worth beating. In Eshelman, L. J. (Ed.), *Proceedings of the Sixth International Conference on Genetic Algorithms*, San Mateo, California. Morgan Kauffmann.

[Sims, 1994] Sims, K. (1994). Evolving 3d morphology and behavior by competition. In Brooks & Maes (Eds.), *Artificial Life IV*, pp. 28–39. MIT Press.

[Tesauro, 1992] Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, 8:257–277.