Coevolution of Neural Networks using a Layered Pareto Archive

German A. Monroy
Dept. of Computer Sciences
The Univ. of Texas at Austin
Austin, TX 78712 USA
gmonroy@ieee.org

Kenneth O. Stanley[1]
Dept. of Computer Sciences
The Univ. of Texas at Austin
Austin, TX 78712 USA
kstanley@cs.ucf.edu

Risto Miikkulainen
Dept. of Computer Sciences
The Univ. of Texas at Austin
Austin, TX 78712 USA
risto@cs.utexas.edu

ABSTRACT
The Layered Pareto Coevolution Archive (LAPCA) was recently proposed as an effective Coevolutionary Memory (CM) which, under certain assumptions, approximates monotonic progress in coevolution. In this paper, a technique is developed that interfaces the LAPCA algorithm with NeuroEvolution of Augmenting Topologies (NEAT), a method to evolve neural networks with demonstrated efficiency in game playing domains. In addition, the behavior of LAPCA is analyzed for the first time in a complex game-playing domain: evolving neural network controllers for the game Pong. The technique is shown to keep the total number of evaluations in the order of those required by NEAT, making it applicable to complex domains. Pong players evolved with a LAPCA and with the Hall of Fame (HOF) perform equally well, but the LAPCA is shown to require significantly less space than the HOF. Therefore, combining NEAT and LAPCA is found to be an effective approach to coevolution.

Categories and Subject Descriptors
I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *Intelligent Agents.*

General Terms: Algorithms, Experimentation, Theory.

Keywords
Coevolution, Neural Networks, Pong, Hall of Fame, Layered Pareto Coevolution Archive, Neuroevolution of Augmenting Topologies.

1. INTRODUCTION
Coevolution is a good learning method for adversarial games for two reasons. First, evaluators are generated automatically and need not be externally supplied. Second, coevolution is open-ended. Since the pool of evaluators is not permanent but improves continuously, the evolving players have to keep up with increased levels of performance, leading to an "arms race" [14]. A sustained arms race might not occur, however, if the evaluators are only taken from the current generation. Good evaluators from past generations may have been lost, and have to be rediscovered, in what is called coevolutionary forgetting [10]. To prevent forgetting and encourage progress, it is necessary to have a Coevolutionary Memory (CM) that retains the most valuable evaluators from previous generations.

Coevolutionary algorithms have been applied to the evolution of neural network controllers in competitive domains like the robotic predator-prey interaction [11] and the robot duel [19]. In both cases, the CM used was the Hall of Fame (HOF). The HOF contains the single fittest individual from every opponent generation; the individuals in the HOF are then used as evaluators for future generations [14]. The HOF is a good heuristic CM because it is very simple to implement: the fitness information required to choose an individual for inclusion into the memory is already provided by the evolutionary algorithm. However, the selective pressure provided by the HOF is likely to be suboptimal because it may be missing useful evaluators produced during evolution that were not the fittest of their generations. Those missed evaluators could have made the evaluation set more *pedagogical* [14]. Besides, elements are never removed from the HOF, so it may contain players that are no longer useful as evaluators.

The theoretical properties of an *ideal evaluator set* [1,7] for coevolution have been studied in the context of Evolutionary Multi-Objective Optimization. Under this approach, the evolving players are called learners and the evaluators are called testers. Defeating each tester is considered a separate objective and learners are compared in their capacity to accomplish multiple objectives. Whenever there is at least one objective that learner A accomplishes but learner B does not, and all of the objectives accomplished by B are also accomplished by A, A is said to Pareto-dominate B. In the resulting *Pareto coevolution* [9,22] learners are not compared in direct competition but in terms of Pareto dominance with respect to a set of testers. The Pareto front is the set of learners that are not Pareto-dominated by other learners. Therefore, the Pareto front contains either the single best or the multiple "better" strategies discovered by Pareto coevolution.

Pareto coevolution has been implemented in a sequence of algorithms of progressive sophistication: DELPHI [6], IPCA [4] and LAPCA [5]. All three algorithms were benchmarked in a game domain called Discretized Compare-on-One that compares discretized numeric vectors by their biggest component. The Layered Pareto Coevolution Archive (LAPCA) algorithm was reported to make faster progress, have a smaller archive size, and result in fewer fitness evaluations [4,5]. Besides, a tester set that makes every possible outcome distinction between learners was mathematically proven to determine *all* underlying objectives of the problem domain, provided that each and every possible candidate is presented (considered for inclusion into the archive) with a non-zero probability [7]. Therefore, the tester set in a Pareto Archive approaches an ideal evaluator set.

The LAPCA algorithm has been demonstrated in a simple game that compares numeric vectors. It has not been applied, for example, to the evolution of neural networks, which are a promising approach to

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

[1] Current affiliation: School of Electrical Engineering and Computer Science. University of Central Florida. Orlando, FL 32816 USA

constructing agents for complex games [15]. On the other hand, implementing the LAPCA in a practical domain using neural networks could require a prohibitive number of evaluations. If the domain has too many underlying objectives, coevolution could cause an explosive archive growth, and the archive update procedure would need too many game evaluations. Hence, the first question that this research intends to address is: can the LAPCA be used to coevolve neural networks using a reasonable number of evaluations?

Moreover, the LAPCA contains a close approximation to the ideal evaluator set, whereas the HOF is mostly a heuristic. Therefore, the LAPCA should outperform the HOF: coevolution with LAPCA should occur faster and yield better individuals. Since this issue has not been studied before, the second question is: how do LAPCA and HOF compare?

To investigate these two questions, this research implements the LAPCA algorithm as the CM for the Neuroevolution of Augmenting Topologies (NEAT) algorithm created by Stanley and Miikkulainen [16]. NEAT was chosen as the evolutionary algorithm because it has been successfully used in the continual coevolution of neural networks [21] and because it has a built-in speciation mechanism that was deemed useful to select candidates. The third question is, therefore: how can NEAT be implemented effectively with LAPCA?

The game domain is a discrete version of the game Pong [23] in which each player is controlled by a neural network. The performance of different variations of the HOF and LAPCA memories is analyzed statistically from multiple runs of the coevolutionary algorithm with different initial conditions. Representative players of every run play against each other and the results of players evolved with the same CM are averaged.

The main result is that it is indeed practical to use a LAPCA instead of the HOF in neuroevolution. In order to achieve this result, a method was developed that keeps the number of evaluations required by LAPCA low. This method exploits the speciation mechanism of NEAT to select the generational candidates for the archive: only the fittest player in each species can be a candidate. Selecting multiple candidates according to this criterion is shown equivalent or superior to choosing the fittest member of the generation or a random set of generation candidates. In general, the LAPCA grows to be smaller than the HOF and its growth can be controlled by adjusting the number of candidates per generation. The slow growth of the LAPCA and the ability to have its size under control makes LAPCA feasible in terms of number of evaluations.

No statistically significant difference was found in the performance of the players evolved with the two CMs. This result can be interpreted in two ways: either the two CMs evolve players with the same performance (at least in the Pong domain) because their evaluators are equally good, or the Pong domain is not very sensitive to the CM used. Additional experiments using smaller CMs showed that the latter is the case. When the only evaluator in the CM was the fittest player of the generation immediately before, only a small amount of regression or coevolutionary forgetting was measured. Thus, the question of whether LAPCA outperforms HOF is still open, and additional experiments in more complex domains are required to measure the impact of the CMs in playing performance.

2. BACKGROUND AND RELATED WORK

This section describes the problem of forgetting in coevolution and the solutions implemented for comparison: the HOF and the LAPCA CMs. The problem is put in context first by introducing coevolution as a generalization of evolution. In addition, the neuroevolution algorithm is reviewed and the Pong game is presented.

2.1 Evolution vs. Coevolution

The final result in traditional evolution is the solution with the highest fitness, according to a given fitness function, among all generations. Coevolution is a more general case of evolution in which, instead of being fixed, the fitness function for one population is determined by another population that is also evolving [2]. Competitive coevolution is a particular case of coevolution in which the fitness of individuals in one population is determined by the outcomes of competitive interactions against individuals in the other population. Competitive coevolution has been successfully applied to games like Nim and 3D Tic-Tac-Toe [14] and Poker [12].

There are three reasons why coevolution is better suited than traditional evolution for two-player games. First, the only way that traditional evolution could be used to evolve players is by already having a pool of good players. This approach implies that another method has to provide such players in the first place, which may be difficult. Second, once the maximum level of play against the fixed pool of players is reached, evolution stops. However, it could improve further if the newly evolved players were to be added to the pool, as is the case in coevolution. Third, in some kinds of games the pool must contain mediocre players for traditional evolution to work. The reason is that if the players in the pool are too good, none of the individuals in the first generations would ever win against them and there would be no selection and in consequence slow or no evolution (only variation). Coevolution, on the contrary, evaluates fitness against players with a variety of skill levels that form a "pedagogical series" [14].

2.2 Need for a CM

In natural coevolution (e.g. in an actual predator-prey situation between two animal species) the populations that are coevolving must be alive at the same time. In other words, the pool of evaluators and the players being evaluated must belong to the same generation. This limitation causes two problems:

- Losing a good trait by collusion. If the populations collude to reduce the selective pressure that they exert on each other, they may lose the performance they once had.
- Rediscovering past strategies. The populations might be stuck in a loop, re-evolving traits they had in the past but that they lost because such traits were not useful to defeat recent generations of the opponent population.

These two problems are instances of a more general problem called forgetting. Forgetting can be avoided by using a CM. A CM is a collection of former players that is representative of all the strategies that have been developed over the course of evolution. To prevent forgetting, instead of only drawing evaluators from the latest opponent generation, evaluators are taken from the opponent CM, potentially from any generation in the past.

A CM is defined by two policies: how to introduce candidates and how to extract evaluators. Candidates are the members of every new generation that are considered for introduction to the CM. Evaluators are the elements of the CM that get chosen to measure the fitness of opponent players. The two CMs analyzed experimentally in this research, the HOF and the LAPCA, are described next.

2.2.1 HOF: A Best-of-Generation CM

The HOF was proposed by Rosin and Belew as a technique to ensure progress in coevolution [14]. The HOF is a CM that preserves the fittest individual of every generation for future use as a fitness evaluator.

A Best of Generation (BOG) CM, as the term will be used here, is more general than the HOF because it can admit more than one individual per generation. To take advantage of the speciation mechanism of NEAT (described in section 2.3), only the fittest individuals from different species are considered for inclusion to a BOG memory. For example, BOG-3 accepts the three fittest individuals of every generation that belong to different species. BOG-1 is the same as the HOF. The candidate introduction policy in a BOG CM is straightforward: all the players presented to the CM are retained forever. The evaluator extraction policy is straightforward as well: evaluators are typically drawn from a uniform sample of the CM.

Due to its simple implementation that does not require additional game evaluations, the HOF has been a common practice in the competitive coevolution of neural network controllers. In particular, it has been implemented in domains like the robotic predator-prey interaction [11] and the robot duel [19]. The HOF is a useful heuristic that forms a baseline on which to improve.

2.2.2 LAPCA as a CM

Pareto coevolution is the interpretation of coevolution as Evolutionary Multi-Objective Optimization [9,22]. Under this view, the evolving individuals are called learners. In Pareto coevolution any two learners are compared by their results against other individuals called testers. Hence, the testers are the multiple objectives being optimized by coevolution and the goal of the learners is to defeat the biggest number of testers.

The Pareto front is the set of learners not dominated in a Pareto sense by any other learner. For a given set of testers, a learner A dominates learner B in a Pareto sense if there is no tester for which B obtains a better score than A but there is at least one tester against which A obtains a better score than B. The Pareto front contains the best players discovered by coevolution and is thus the *solution concept* of Pareto coevolution [8].

The Pareto Archive is the union of learners and testers. De Jong [4] showed that monotonic progress in coevolution can be guaranteed in a particular version called Incremental Pareto-Coevolution Archive (IPCA), provided that every possible individual is generated with a non-zero probability. The Layered Pareto-Coevolution Archive (LAPCA) is a practical approximation of IPCA: progress is not guaranteed but the archive grows slower than in IPCA, needs fewer evaluations, and makes faster progress [5].

In LAPCA, every generation an archive update procedure receives a set of learner candidates and a set of tester candidates. Its operation can be roughly assimilated to a sieve. Normally during the update procedure some learner candidates are retained in the learner set and some tester candidates are retained in the tester set, while the rest are immediately discarded. When new individuals join the archive, usually the dominance structure changes and some older members of the archive are eliminated. The number of game-outcome evaluations that are required by the update procedure is approximately proportional to the product between the number of candidates and the size of the archive.

The LAPCA algorithm receives its name because the learners are structured in non-dominated layers, resembling the peeling an onion. The first non-dominated layer is the Pareto front. Once the first non-dominated layer has been removed, the set of non-dominated learners remaining constitutes the second layer, and so on. This layered structure for the learners is useful for two reasons. First, it provides a useful criterion for the update procedure: a tester is retained only if it can discriminate learners that belong to the same or consecutive layers. Second, the size of the archive can be adjusted by the experimenter by retaining only the first *n* non-dominated layers. In order to maximize the quality of the LAPCA, in this research all layers are retained. Since new testers are retained according to whether they distinguish between existing learners and new learners are retained according to the objectives established by the existing testers, a mutual dependency develops between learners and testers.

The candidate introduction policy for the LAPCA CM is determined by the update procedure. However the experimenter decides which individuals in the evolving population are presented as learner candidates and which ones as tester candidates. The evaluator extraction policy for the LAPCA CM is also up to the experimenter, who decides whether evaluators are drawn from the learner set, the tester set, the first non-dominated layer of learners or from the whole archive.

2.3 NEAT: Evolution of Neural Networks

Many methods of training and evolving neural networks search for an optimum set of weights once the researcher has provided a candidate topology. Choosing such a fixed topology is a big problem in itself and there is always the risk of using a suboptimal number of nodes and weights. Stanley and Miikkulainen's NeuroEvolution of Augmenting Topologies (NEAT) searches both the topology *and* the weight spaces simultaneously, starting from a minimal configuration with no hidden nodes [16]. In addition to the conventional weight mutation operators that produce variation, NEAT also mutates the topology by adding hidden nodes within existing links and by adding weighted links between unlinked nodes. This process of topology growth over generations is called "complexification" [20].

NEAT has been shown to achieve efficient reinforcement learning of neural networks in discrete-time controller tasks. In particular, it has been reported to require a record low number of evaluations in the double pole balancing benchmark [18]. NEAT has also been applied to the coevolution of controllers for the robot duel domain [21]. The robot duel is an open ended problem with opportunity for many different winning strategies. The complexification of NEAT was found responsible for the discovery of new and more powerful strategies over the course of coevolution. The speciation mechanism of NEAT was credited with optimizing previously found strategies [19].

Because NEAT has been successfully applied to a competitive coevolution domain and because it provides speciation and complexification, it was chosen to investigate the impact of different CMs.

2.4 Test Domain: a Modified Pong Game

Poppleton was probably the first to analyze Coevolution in the game of Pong [13]. Instead of using a CM, he used the fittest player in the most recent generation (Last Elite Opponent) of up to four genetically isolated populations, to compute the fitness of every new generation of players. Poppleton concluded that his experimental results did not confirm nor disprove that such method of fitness evaluation offered an advantage.

The experiments in this research are also based on Pong. The main advantage offered by Pong is that the board can be represented as a grid of discrete positions. The collision detection algorithm is simple, leading to fast evaluations and hence short simulations. Another advantage is that the evaluation time can be modified arbitrarily by changing the size of the grid.

The player's ability to move the paddle forward and backward and to have independent control of the ball deflection allowed increasing the complexity of the strategy space. A more complex strategy space, in turn, was deemed useful to take full advantage of the CMs. The specifics of the game domain are described in the following

section, along with the comparison methodology used in the experiments.

3. COMPARISON METHODOLOGY

The experiments described in this research assess the relative performance of different variants of coevolution by applying two comparison methodologies to the evolved players: Best of Run and Best of Generation. Each comparison has advantages and disadvantages but they complement each other well. All the evaluations required by evolution and by the comparison methodologies take place in the Pong domain, detailed next.

3.1 Parameters of the Game Domain

The board grid used in the experiments is 15 units tall and 21 units wide, with the ball occupying a single unit and the paddles an area of five units by one unit. The paddles are free to move horizontally within the first 7 units of their side of the field, and vertically with no limitations. The paddles can move at most one unit in each direction (both vertical and horizontal) per time step. The ball is allowed to move twice as fast as the players, to force the players to predict the vertical position of the ball instead of just following it.

The neural network controller for each player has 6 inputs and 6 outputs. The 6 inputs are three pairs of absolute coordinates: the player's paddle, the ball and the opponent's paddle. The 6 outputs are three pairs of binary values, corresponding to vertical motion, horizontal motion and ball deflection. Since the player knows the location of the opponent, it can have an advantage by deflecting the ball away from it.

Each game consists of two serves, one for each side, to make the game symmetrical. A player wins a serve when the opponent misses the ball. If no player has missed the ball after 200 time steps, the serve is considered a tie. Fitness is computed by letting every player of a generation compete in 10 games against the same set of opponents, and averaging the scores. The set of opponents is uniformly sampled from the CM. For all experiments the population size is 100 and coevolution lasts 100 generations.

The update procedure of the LAPCA uses evaluations of direct dominance between learners and testers to determine the Pareto dominance between two learners. The dominance relationship between a learner and a tester in turn is computed by playing all 30 possible serves between them. If the learner wins more serves than it loses (independently of the number of ties) it is considered to dominate the tester.

3.2 Best of Run Comparison

"Best of Run" is defined as the most successful player of the 10,000 originated in a particular run of the coevolutionary algorithm (100 generations times 100 individuals per generation) and is *the* solution to the problem. For all experiments in this research, the Best of Run is the winner of a *master tournament* [11] among the fittest players (champions) of every generation.

The Best of Run Comparison determines if one of two methods of coevolution is better by measuring the performance of 50 Best of Run players from each method, which requires a total of 100 coevolution runs. Each of the 100 individuals plays *all* possible games (15 serves in each direction) against each of the other 99 individuals and gets scored by the number of players that it dominates. Thus, the score is an integer between 0 and 99. A player dominates another if it wins at least one more serve than the opponent in all possible games of direct competition. The scores of all players from the same method are averaged and the means of the two methods are tested for one-sided significance using a two-sample z statistic [3].

The main advantage of the Best of Run Comparison is that it measures the real output of coevolution. All that matters for a practical application is the single best player evolved. A disadvantage is that it does not show the speed of coevolution. Even if the comparison does not demonstrate that one method is significantly better than the other, one can still be faster, i.e., reach the final level of performance in fewer generations.

3.3 Best of Generation Comparison

The Best of Generation Comparison attempts to solve the limitation of the Best of Run Comparison by approximating an absolute fitness function and using it to measure progress. The comparison requires 50 runs of each method being compared. First, the best players of each generation (champions) are stored, for all generations, all runs and all methods. Second, the Best of Run individuals of all methods are collected together in an evaluation pool. Third, each stored champion plays against a different sample of 25 players drawn from the evaluation pool. Fourth, for each method and each generation, the number of wins, ties and losses is averaged over the 50 runs.

The comparison methods complement each other: the Best of Run comparison determines the CM that evolves the best players, whereas the Best of Generation shows coevolutionary progress over generations. Both comparisons are applied to different types of CMs in the next section.

4. EXPERIMENTS

The specific questions addressed by this research are:

- Does the LAPCA scale up to a complex domain like Pong?
- What is the best way to introduce candidates from NEAT into the LAPCA?
- Is the LAPCA a better CM than the HOF?

The experiment to answer the first question uses the HOF as the CM while a LAPCA is used to monitor the progress of coevolution. To answer the other questions, a LAPCA is implemented as the CM. First, the advantage of introducing random versus fittest candidates to the archive is evaluated. Then, the LAPCA and the BOG memories (which include the HOF) are compared.

4.1 Analyzing the Growth of the LAPCA

Before using the LAPCA as a CM, it is helpful to analyze how it grows when applied to monitor an already evolving population. A controlled growth is important because the size of the archive determines the number of evaluations and the storage space required by the LAPCA algorithm. If the archive grows too fast its applicability would be restricted, due to an excessive number of evaluations.

Using the HOF as the CM, after every generation three players were presented as tester candidates and three different players were presented as learner candidates to a LAPCA. The tester candidates were drawn randomly from the population, whereas the learner candidates were the three fittest players belonging to three different species in the population.

The average size of the learner and the tester sets of the archive at every generation are shown in Figure 1. The first interesting result is that the learner set is consistently bigger than the tester set (from around 50% in early generations to about 100% in the last ones). This fact has also been reported by De Jong for the Compare-on-one problem [5]. Another observation is that the archive is an effective filter, since by generation 100 it has retained a minority (an average of 37.6 with standard deviation 6.3) of the 600 players presented as candidates. Finally, the fact that the rate of archive growth decreases over generations suggests that the update procedure detects the stagnation in the quality of the candidates presented (as shown in

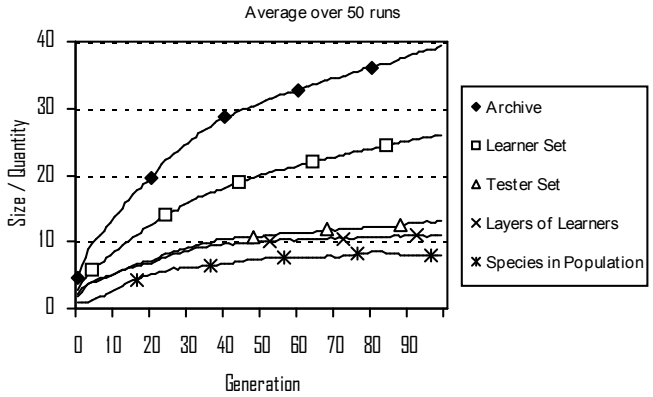

Figure 1. Growth of the LAPCA and its components. The LAPCA is not used as a CM in this experiment, but to monitor the progress of coevolution. The small size of the archive and its controlled growth prevent the archive update procedure from requiring an excessive number of evaluations.

Figure 2 and Figure 3, most of the learning occurs in the first 40 generations).

4.2 Introducing Candidates to the LAPCA

After having analyzed the growth rate of the LAPCA as a monitor, the next experiment introduces the LAPCA as a CM. The whole archive (the union of the learner and tester sets) was used to extract evaluators, i.e. to measure the fitness of the evolving individuals.

To use a LAPCA in the most effective way as a CM, it is necessary to determine which individuals from the population, when presented as candidates, produce the best archive (i.e. one that results in higher selective pressure and therefore superior players). It makes sense to present candidate individuals from different species to exploit the diversity in the speciation mechanism of NEAT. However, selecting only the fittest players of different species could produce a bias towards strategies that win on average, which may in turn prevent idiosyncratic players from joining the archive. Such lost idiosyncratic players could have had strategies that worked

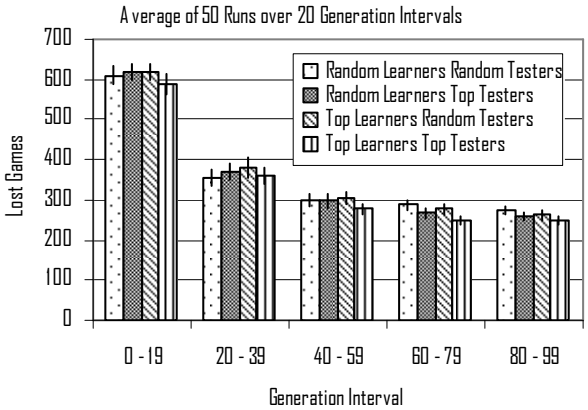

Figure 2. Best of Generation comparison between different methods of selecting candidates to the LAPCA CM. Fewer losses mean better learning; losses are averaged over 20 generation intervals. The vertical marks on top of the bars correspond to standard error. "Top Learners Top Testers" is significantly better ($p < 0.02$) than "Random Learners Random Testers" in the interval 60-79. "Top Learners Top Testers" is also better ($p < 0.05$) than "Top Learners Random Testers" in the same interval. The fittest individuals from different species are better candidates

Table 1. Best of Run pairwise comparison between different methods of presenting candidates to the LAPCA CM (N=50). Both the learner and tester candidates can be either the three fittest individuals in the Population belonging to three different species (Top), or just three individuals chosen at random (Random). The first two comparisons are statistically significant (z-test, $p < 0.05$). The fittest individuals from different species are better candidates.

Comp.	Candidate Introduction		Dominated Players (Average)	Dominated Players (Std Dev)	Diff.	p value
	Learners	Testers				
1	**Top**	**Top**	49.30	15.81	5.32	**0.040**
	Random	Random	43.98	14.51		
2	**Random**	**Top**	48.82	15.27	5.00	**0.047**
	Random	Random	43.82	14.59		
3	Top	Random	48.40	17.26	3.54	0.133
	Random	Random	44.86	14.48		
4	Top	Top	48.00	15.06	3.08	0.162
	Top	Random	44.92	16.20		
5	Top	Top	47.64	14.68	2.58	0.197
	Random	Top	45.06	15.51		
6	Top	Random	47.26	16.56	1.78	0.286
	Random	Top	45.48	14.85		

extremely well in specific situations and would have raised the selective pressure of the archive. In other words, fitness alone might not be the best criterion to present individuals to the archive, since it could be diversity that matters most.

The second experiment compares two kinds of candidate introduction: "Random" and "Top". "Random" corresponds to choosing three individuals from a uniform sample of the population, disregarding species and fitness. "Top" corresponds to choosing the three fittest individuals in the population that belong to different species. Since these two types of candidate introduction can be applied independently to the tester and the learner sets, there are four combinations. The six possible pairwise comparisons between the four types of candidate introduction appear on Table 1, using the Best of Run method. There are only two statistically significant comparisons: "Random Learners and Random Testers" are

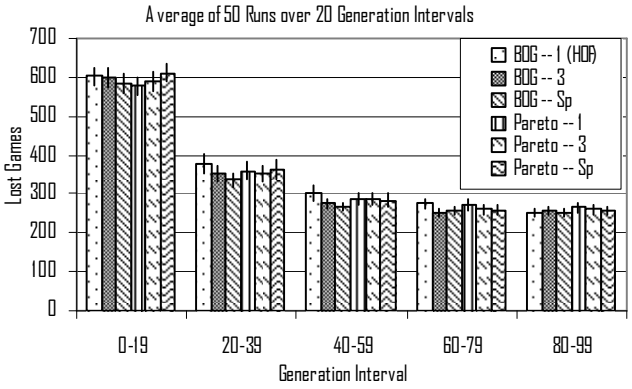

Figure 3. Best of Generation comparison between the BOG and LAPCA CMs. Fewer losses mean better learning; losses are further averaged over 20 generation intervals. The population candidates introduced to each memory (1, 3, Sp) are the same as in Table 2. "BOG -- Sp" is better ($p < 0.05$) than "BOG -- 1 (HOF)" in the generation interval 40-59. For the rest of the cases, the performance of evolved players is similar.

Table 2. Best of Run pairwise comparison between the BOG and the LAPCA CMs (N=50). The candidates are the fittest individual from each generation ("1"), the top 3 fittest individuals belonging to different species ("3"), or the fittest individual of every species ("Sp"). The LAPCA with a candidate from every species (from approximately 8 in the last 40 generations) is better ($p < 0.06$) than the BOG memory with the top 3 fittest individuals from different species. For the rest of the cases, the performance of evolved players is similar.

Comp.	Coevolutionary Memory	Candidates per Generation	Players Dominated (Average)	Players Dominated (Std Dev)	Diff.	p value
1	**LAPCA**	**Sp**	48.46	13.98	4.60	**0.051**
	BOG	3	43.86	14.22		
2	LAPCA	Sp	47.72	13.73	2.86	0.163
	BOG	1 (HOF)	44.86	15.30		
3	LAPCA	3	47.72	14.90	2.52	0.179
	BOG	Sp	45.20	12.36		
4	LAPCA	Sp	47.22	14.20	2.32	0.205
	BOG	Sp	44.90	13.91		
5	LAPCA	Sp	46.90	13.70	1.40	0.312
	LAPCA	1	45.50	14.84		
6	BOG	3	46.74	16.42	1.10	0.356
	BOG	Sp	45.64	13.29		
7	LAPCA	3	46.66	15.21	1.08	0.359
	BOG	3	45.58	14.68		
8	LAPCA	3	46.86	15.07	0.98	0.373
	LAPCA	1	45.88	15.25		
9	BOG	1 (HOF)	46.68	16.41	0.76	0.403
	LAPCA	1	45.92	14.56		
10	BOG	Sp	46.44	13.08	0.72	0.404
	BOG	1 (HOF)	45.72	16.24		
11	LAPCA	3	46.56	15.08	0.66	0.408
	LAPCA	Sp	45.90	13.21		
12	BOG	Sp	46.80	13.40	0.64	0.416
	LAPCA	1	46.16	16.72		
13	BOG	1 (HOF)	46.52	14.87	0.24	0.468
	LAPCA	3	46.28	15.49		
14	LAPCA	1	46.72	15.33	0.22	0.471
	BOG	3	46.50	15.10		
15	BOG	3	46.22	14.29	0.12	0.484
	BOG	1 (HOF)	46.10	15.34		

outperformed by "Top Learners and Top Testers" and also by "Random Learners and Top Testers". The conclusion is that candidate fitness is more important than candidate diversity at least for the Pong domain studied. Besides, using the Best of Generation comparison (Figure 2), there is a significant performance gap between "Top Learners and Top Testers" and "Random Learners and Random Testers" around generation 70.

In conclusion, at least in the Pong domain, it is better to introduce the fittest individuals to the archive. Using random individuals to exploit their erratic behavior does not lead to a more solid learning. For this reason, in the next experiment the candidates introduced to the LAPCA are the fittest in their species.

4.3 BOG vs. LAPCA CMs

The third experiment compares the performance of the BOG and LAPCA CMs, in three different aspects: performance of evolved players, number of evaluations, and storage size.

For each of the two memories, three different sizes of the candidate sets were evaluated: "1", "3" and "Sp". Size "1" means that after every generation only the fittest player of the population is either added to the BOG or presented to the LAPCA. Size "3" means that the candidates are the three fittest players in the generation that belong to different species. Size "Sp" means that one candidate from each species is presented to the memory, for all species in the generation (as shown in Figure 1, there are about 8 species in average in the last 40 generations). The HOF is the particular case of a BOG memory with candidate size 1 (i.e., HOF is the same as "BOG – 1") and is a good baseline to judge the performance of the LAPCA.

Table 3. Evaluations vs. Memory size trade-off between the BOG and LAPCA CMs (N=50). Whereas in a BOG memory more candidates lead to a bloated CM without the cost of additional evaluations, a LAPCA with more candidates requires more evaluations but uses less storage.

Coevolutionary Memory	Candidates Introduced		Final Memory Size		Evaluations	
	Average	Std Dev	Average	Std Dev	Average	Std Dev
BOG – 1 (HOF)	100	0.0	100	0.0	**200,000**	0
BOG – 3	300	0.0	300	0.0	**200,000**	0
BOG – Sp	630	73.1	630	73.1	**200,000**	0
Pareto – 1	100	0.0	**20**	3.77	262,478	10,691
Pareto – 3	300	0.0	**38**	6.18	503,392	39,026
Pareto – Sp	618	64.4	**45**	8.97	1,000,838	135,341

There was no statistically significant difference in the performance of the CMs, as measured by the quality of evolved players. However, there are indications that larger candidate sets might have an advantage over smaller ones. Specifically, according to the results of the Best of Run method from Table 2, "LAPCA – Sp" has better performance than "BOG – 3" ($p < 0.06$). Also, from Figure 3, "BOG – Sp" is better ($p < 0.05$) than "BOG – 1" (i.e., the HOF)" around generation 50.

In terms of number of evaluations, the LAPCA has a clear disadvantage (Table 3). The algorithm to implement any BOG memory does not require extra evaluations. Evaluations are only needed to compute the fitness of a newly created population. Since the update algorithm of the LAPCA needs to establish dominance relationships between all the candidates presented and all players in the archive, the number of extra evaluations is not only approximately proportional to the number of candidates but also to the size of the archive, which for this experiment increased monotonically over generations (Figure 1).

The main advantage of the LAPCA is that it requires less storage (Table 3). As a consequence of the update procedure filtering new candidates and eliminating subsumed strategies, the LAPCA requires less storage than the HOF and the rest of the BOG memories for all sizes of the candidate set. Since the players evolved with the six methods have similar performance (Table 2 and Figure 3) the LAPCA reduces the redundancy present in the BOG CMs without sacrificing selective pressure.

In conclusion, in the Pong domain the LAPCA is not significantly better or worse than the BOG CM in terms of performance of the evolved players, but it uses less memory and requires more evaluations. In addition, in the few cases in which there was a significant difference, more candidates per generation produced better results, suggesting that the HOF can be outperformed when the diversity present on the population is exploited.

5. DISCUSSION AND FUTURE WORK

One possible reason for the similar playing performance of the HOF and the LAPCA is that there is not enough forgetting in the Pong domain to reveal the differences. Additional experiments were performed to test this hypothesis. No measurable difference was found in the average quality of the players evolved either with a CM with only one element (the fittest individual of the generation immediately before) or with the full HOF, suggesting that there is indeed little forgetting. The experiments described in this research should be repeated in a domain with more coevolutionary forgetting. A good criterion to make sure that such domain allows meaningful comparisons is precisely the difference in the performance of players evolved remembering either *the last* or *all* members of the HOF.

In spite of the resistance to forgetting in the Pong domain, important conclusions can be drawn about the implementation of the LAPCA as a CM for the coevolution of neural networks, and about the interaction between the LAPCA and NEAT. One such conclusion is that there is a positive, measurable impact from feeding the archive with candidates that are the fittest within their species, as opposed to randomly sampling candidates from the population (Section 4.2). Along the same lines, in the few comparisons that had statistical significance, the bigger number of generational candidates corresponded to faster or better evolution (Section 4.3). Consequently, a CM can take advantage of the population diversity maintained by speciation, and the best way to do it is by introducing the fittest individuals from different species.

Another important conclusion is that the number of evaluations required by the update procedure of the LAPCA is practical for real world applications. The number of evaluations per generation needed by the archive update algorithm is roughly proportional to the number of candidates and to the size of the archive. The archive does not grow too fast, because many old players get subsumed by newer players and are immediately removed. The number of candidates per generation is arbitrarily set by the experimenter. Hence, it is possible to adjust the number of evaluations needed to update the archive and have it in the same order of magnitude as the number of evaluations needed by NEAT to compute fitness (Section 4.3).

In his presentation of the LAPCA algorithm, De Jong [5] used the archive in a different way than in this research: not to provide evaluation for the population, but to re-introduce genetic material into it. Such genotypical use of the archive could be implemented with NEAT as well. Although it might interfere with the stability of NEAT's speciation mechanism or potentially stifle complexification, re-introducing previous genomes might give them a second chance to evolve into the most successful players for the new competitive environment. It would be interesting to compare the phenotypical and genotypical uses of the archive.

A central simplifying assumption in this research is that only one population coevolves, and does so by competing against its own CM. Coevolving a single population against itself is a departure from the way coevolution has been implemented in the past. In other symmetric domains like the robot duel and compare-on-one, the two players are evolved in different populations, each with its own CM [21,4,5]. It would be interesting to measure the impact of merging the two populations, as was done in this paper.

In order to make fair comparisons between different CMs, the best of run individuals were the winners of an internal master tournament among generation champions. Nevertheless, since the LAPCA memory contains dominance information about the players, the winner should be drawn directly from archive, with no need for a master tournament. Which individual to choose and how good its performance is with respect to the winner of the master tournament are further questions for future work.

The LAPCA update procedure implemented in this research can be optimized to require fewer evaluations. First, when a new generation of candidates is presented to the current archive, the scores of all possible games between candidates and members of the archive are being computed and stored in a matrix, disregarding whether they are actually used or not. To make the process more efficient, all known matrix entries can be considered first and the rest obtained only when necessary. Second, it should be possible to determine the impact of a limited number of layers on the performance of the evolved players; if the impact is small, fewer evaluations would be required to get essentially the same players.

There are alternative methods of coevolution that should be applied to neural networks, as well. One variation is replacing the solution concept of the Pareto Front by that of Nash equilibrium, using the Nash memory mechanism proposed by Ficici and Pollack [10]. Another possibility is to apply one or both solution concepts to neural network controllers for *team* competitive games like soccer, in which the interaction between teammates introduces cooperation to coevolution.

The results obtained in this research are encouraging because they show that the principles of Pareto coevolution can be applied to practical domains. The computational expense of keeping an archive could be offset by faster evolution (fewer generations) reducing the overall number of evaluations and making coevolution practical for real world applications.

6. CONCLUSION

The main goal of this research was to construct a bridge between two branches of the study of coevolution that have not been investigated together. The first branch comes from neuroevolution and is very practical: coevolving optimal neural network controllers using the fewest number of evaluations. The computational simplicity of maintaining a HOF (which does not require additional evaluations) has made it the CM of choice for the competitive coevolution of neural networks. The second branch comes from the theoretical analysis of coevolution seen as multi-objective optimization: implementing an ideal CM. Such a memory requires many evaluations to make explicit the Pareto-dominance relationship between evolving individuals. The recently introduced LAPCA algorithm [5] decreases the number of evaluations needed to approximate a good CM. However, it has only been applied to a Numbers Game which is not a very practical domain.

This research implemented LAPCA as the CM for neural networks that control the players of the game Pong. The evolutionary algorithm chosen was NEAT [16] because its speciation and complexification features have already been found useful for continual coevolution. The question of whether coevolution can discover better solutions using a LAPCA memory than using a HOF memory remains open because there is not enough forgetting in the Pong domain. However, three interesting results were obtained. The first is a practical technique to provide generational candidates to a CM: choose the fittest individuals that belong to different species. In the Pong domain this technique brings more information to the memory than introducing the single best player of the generation, and is more effective than introducing a random sample with the same size. The second result is that the LAPCA scales up to a complex neural network domain. When applied to the evolution of neural networks, the bounded growth of the archive makes the LAPCA a feasible option in terms of total number of evaluations. Third, in all experiments performed, the LAPCA required significantly less memory than the HOF. Therefore,

LAPCA has a storage advantage in applications in which evolution lasts for many generations.

In sum, LAPCA has been demonstrated to be a practical CM for neural networks in the Pong domain. Further research in domains that are more susceptible to forgetting is necessary to determine whether it also allows the discovery of solutions that perform better. Eventually it should be possible to establish the circumstances under which each of the two algorithms provides a better CM for a given real-world application.

7. ACKNOWLEDGEMENTS

We would like to recognize Vishal Arora, for his valuable contributions to early experiments. Thanks to Ugo Vieruchi for coding JNeat, used as the base to write the simulations, and to four anonymous reviewers for important corrections and suggestions.

This research was supported in part by the National Science Foundation under CISE Research Infrastructure Grant EIA-0303609.

8. REFERENCES

[1] Bucci, A. & Pollack, J. B. (2002). A mathematical framework for the study of coevolution. *Foundations of Genetic Algorithms 7, Proceedings of FOGA VII*, 221-236.

[2] Cliff, D., & Miller, G. (1995). Tracking the red queen: Measurements of adaptive progress in coevolutionary simulations. *Proceedings of the Third European Conference on Artificial Life*, 200-218.

[3] Cohen, P. R. (1995). *Empirical methods for artificial intelligence*. MIT Press, 117-129.

[4] De Jong, E. D. (2004). The incremental Pareto-coevolution archive. *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-04*, 525-536.

[5] De Jong, E. D. (2004). Towards a bounded Pareto-coevolution archive. *Proceedings of the Congress on Evolutionary Computation CEC-04,* 2341-2348.

[6] De Jong, E. D., & Pollack, J. B. (2003). Learning the ideal evaluation function. *Proceedings of the 2003 Genetic and Evolutionary Computation Conference*, 277-288.

[7] De Jong, E. D., & Pollack, J. B. (2004). Ideal evaluation from coevolution. *Evolutionary Computation, 12*(2), 159-192.

[8] Ficici, S. G. (2004). Solution concepts in coevolutionary algorithms. *Dissertation Abstracts International, 65* (03), 1399B.

[9] Ficici, S. G., & Pollack, J. B. (2000). A game-theoretic approach to the simple coevolutionary algorithm. *Parallel Problem Solving From Nature (PPSN-VI), 1917,* 467-476.

[10] Ficici, S. G., & Pollack, J. B. (2003). A game-theoretic memory mechanism for coevolution. *Proceedings of the 2003 Genetic and Evolutionary Computation Conference*, 286-297.

[11] Floreano, D., & Nolfi, S. (1997). God save the Red Queen! Competition in co-evolutionary robotics. *Genetic Programming 1997: Proceedings of the Second Annual Conference*, 398-406.

[12] Noble, J., & Watson, R. A. (2001). Pareto coevolution: Using performance against coevolved opponents in a game as dimensions for Pareto selection. *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2001*, 493-500.

[13] Poppleton, T. (2002). *Can co-evolution play ball? Competitive co-evolution in a Pong game*. Unpublished master's thesis, University of Sussex, UK. Retrieved August 6, 2005, from http://www.informatics.susx.ac.uk/easy/Publications/Online/MSc2002/ajp25.pdf

[14] Rosin, C. D., & Belew, R. K. (1997). New methods for competitive evolution. *Evolutionary Computation, 5*, 1-29.

[15] Stanley, K., Bryant, B. D., & Miikkulainen, R. (2005). Real-time neuroevolution in the NERO video game. *IEEE Transactions on Evolutionary Computation, 9*(6), 653-668.

[16] Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolution, 10*(2), 99-127.

[17] Stanley, K. O., & Miikkulainen, R. (2002). Efficient evolution of neural network topologies. *Proceedings of the 2002 Congress on Evolutionary Computation (CEC '02), 2*, 1757-1762.

[18] Stanley, K. O., & Miikkulainen, R. (2002). Efficient reinforcement learning through evolving neural network topologies. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, 569-577.

[19] Stanley, K. O., & Miikkulainen, R. (2002). Continual coevolution through complexification. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, 113-120.

[20] Stanley, K. O., & Miikkulainen, R. (2003). Achieving high-level functionality through Complexification. *Proceedings of the AAAI-2003 Spring Symposium on Computational Synthesis*, 226-232.

[21] Stanley, K. O., & Miikkulainen, R. (2004). Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research, 21*, 63-100.

[22] Watson, R. A., & Pollack, J. B. (2000). Symbiotic combination as an alternative to sexual recombination in genetic algorithms. *Parallel Problem Solving From Nature (PPS-VI)*, *1917,* 425-436.

[23] Winter, D. (2005). *Magnavox Odyssey: First home video game console*. Retrieved August 6, 2005, from http://www.pong-story.com/odyssey.htm