A Population-Differential Method of Moni toring Success and Failure in Coevolution

Ari Bader-Natal and Jordan B. Pollack.   DEMO Lab
Dept. of Computer Science, Brandeis University, Waltham, MA 02454

The task of monitoring success and failure in coevolution is inherently difficult, as domains need not provide any external metric that can be used to measure performance. Past work on monitoring "progress" all strive to identify and measure success, but none attempt to identify failures. We suggest that this limitation is due to the common reliance on a "best-of-generation" (BOG) memory mechanism, and propose an alternate "all-of-generation" (AOG) mechanism free of this limitation. Using AOG data, we propose a population-differential method for monitoring coevolution in arbitrary domains. With this method, we demonstrate the ability to profile and distinguish an assortment of coevolutionary successes and coevolutionary failures, including arms-race dynamics, disengagement, cycling, forgetting, and relativism.

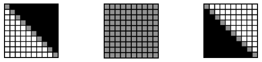
Coevolution requires no domain-specific notion of objective fitness, enabling coevolutionary algorithms to learn in domains for which no objective metric is known or for which known metrics are too expensive. But this benefit comes at the expense of accountability, as there is consequently no external metric with which to measure an algorithm's performance. Several counter-productive behaviors have been identified in the literature using simple but measurable domains such as the *numbers game*, introduced by Watson and Pollack in [10]. After reviewing existing best-of-generation (BOG) coevolutionary monitoring techniques, we present an all-of-generation (AOG) alternative. **Upon this framework we develop a new method called population-differential analysis and demonstrate that it can provide rich feedback about successes and failures over any coevolutionary domain.** This feedback can be used both to refine variations in algorithms and to automate ways of controlling parameters for coevolution in novel domains.

Best-of-Generation Techniques

Analysis based on generation tables was first introduced in coevolution by Cliff and Miller, and has been pursued further by others, including Floreano and Nolfi, and Stanley and Mikkulainen [3,6,9]. Rosin and Belew later incorporated BOG data into the selection mechanism, in Hall of Fame [7].

A generation table assigns the table rows to the first population's sequence of generations, and assigns table columns to successive generations of the second population. Internal table entries contain the results of evaluating the combination of the corresponding row and column generations. For data visualization, Cliff and Miller turn their tables into bitmap images (one pixel per table entry), and we visualize tables similarly.

This organization of data is valuable in making apparent the Red Queen effect: values drawn from evaluations along the table's diagonal. Graphs displaying this instantaneous fitness over time are excellent illustrations of the Red Queen effect. Generation table values are only comparable if either the candidate or the test is kept constant.



Best-of-Generation Criticism

As Ficici and Pollack note in [5], because of the history of single-objective fitness measurements, almost all approaches in the literature (including those cited above) concern themselves solely with the "best-of-generation" (BOG) member of each population. No other individuals are retained for analysis. This BOG approach appears particularly limitingfor two reasons: **First, results of an analysis can vary with the definition of best population member.** [3,6,9] all adopt the *Last Elite Opponent* criterion proposed by Sims in [8], but it should be noted that any number of alternate definitions may be equally plausible. Indeed, the coevolutionary algorithm under examination may itself define *best* differently. Pareto coevolution, for example, would define *best* as the subset of individuals along the Pareto front. **Second, while BOG-based analysis may give insight into algorithmic dynamics of the highly-successful individuals, it provides little about the population as a whole.** While no claims were made about these monitors regarding sensitivity to failure, it ought to be noted that these BOG techniques are not sufficient to detect the common coevolutionary failures detailed by Watson and Pollack in [10].

All-of-Generation Technique

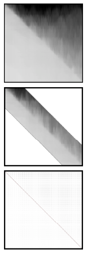
We introduce an alternative that addresses the shortcomings of BOG-based methods in monitoring failure based upon all-of-generation (AOG) data evaluation. For the sake of generality, the technique presented here is designed for simulations involving two asymmetric populations, but it is equally applicable to symmetric two-population simulations and to single-population simulations (via partitioning.) Adopting terminology from [2], we refer to these as populations of candidates and tests. In order to minimize domain-specificity, the result of a candidate-test pairing is simply one element of the ordered set R = {*candidate-failed-test* < *candidate-tied-test* < *candidate-passed-test*}, as in [2].

Population-Grained Evaluation

Where an entry in Cliff and Miller's generation table is the result of evaluating the "best" candidate against the "best" test for the specified generations, an entry in an AOG-based generation table must somehow represent the result of evaluating all candidates against all tests for the specified generations. Where individual-grained evaluation is well-defined, population-grained evaluation must be introduced. We simply average the numeric results of evaluating all candidates against all tests.

Reducing Complexity of AOG Analysis

Clearly this change from BOG- to AOG-analysis increases the computational complexity of evaluation. Assuming population sizes remain constant (at |C| candidates and |T| tests per population), a g-generation simulation can be described as follows: a simple BOG analysis requires $g^2+g|C|+g|T|$ evaluations, while a simple AOG-based analysis requires $g^2|C||T|$ evaluations. **Much of the computational burden of analysis can be alleviated by implementing a memory policy.**

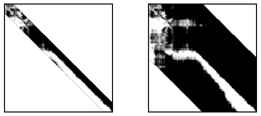


One can restrict computation to a representative subset of entries, and base analysis on only this data. While we will not address the issue of optimally specifying "representative subset" here, we do provide examples of simple example subsets. We view this as a memory-maintenance operation: At each generation, we can decide whether or not to add the newest generation to the generational "memory," and whether or not to eliminate anything from the memory. We then evaluate with respect to memories. The following were suggested in [3]:

Lossless Memory - The AOG techniques described above are "lossless" by default. All generations in a simulation are added to the generation table, and they are actively evaluated for every generation of the simulation. This memory policy requires $g^2|C||T|$ evaluations for the analysis.

Sliding-Window Memory - By implementing the memory as a fixed-size FIFO queue, the size of the memory can be easily bounded. The size of the bounds can vary behavior from that of lossless memory to that of no memory at all; between accuracy and efficiency. Given window bound b<g, $(2bg-b^2)|C||T|$ evaluations are required.

Sampled Memory - If only every j^{th} generation is added to the memory, computation is reduced by a factor of j^2, to $(g/j)^2|C||T|$ evaluations. Sampling can be used in conjunction with a sliding-window, requiring $(2b(g/j)-b^2)|C||T|$ evaluations.



Window size affects performance analysis.
AOG data for one simulation is shown with two different window sizes. Performance analysis of the two sets of data will likely drastically differ.

Population-Differential Analysis

Next we construct a perfomance measure based on the data available in the memory. Nolfi and Floreano addressed this by averaging data per generation [6], but this makes most trends less apparent (e.g. intransitive cycling.) As an alternative, **we compare the current population to the oldest population in memory as an indicator of directionality of change over time.** See Tech. Report for details.

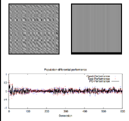
The *candidate-population performance* at generation i is defined to be the average of the population comparators between the newest (i^{th}) candidate population and the oldest candidate population currently in the memory, with respect to all test populations currently in memory. (The *test-popluation performance* measure is defined similarly.)

By restricting interest to the far reaches of the memory, a population-differential analysis does not reward or penalize for localized variability. The only change that is of interest is that between the population's ancestry and its current state. Additionally, this keeps the computational complexity of calculating performance low. (For a memory of size n, only 2n comparisons need to be calculated.) Intuitively, these performance measures reflect the directionality of change over available memory.

Experiments

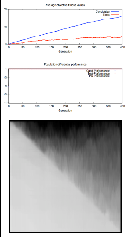
In order to demonstrate the value of the population-differential performance monitor, the first four experiments presented use *numbers game* variants as a domain. In these games, tests and candidates are each a point on a two-dimensional grid. Mutation simply moves an individual to a nearby location. Evaluation of candidate-test pairs varies by game variant, and details are included in [10]. These domains are trivially simple and, more importantly, offer an acceptable external metric that can be used to support or challenge the PC-Performance monitor. The fifth experiment uses the *Rock-Paper-Scissors* game as a domain, showing that the performance monitor can provide useful insight into games with no such external metric. We examine five known coevolutionary behaviors here, which are labeled as follows in the figures below: *arms-race dynamics, lock-in failure, variation, disengagement, and cycling.*

References
1. Anthony Bucci and Jordan B. Pollack. Focusing versus intransitivity: Geometrical aspects of coevolution. E. Cantú-Paz, et. al, editors, *GECCO-2003*, volume 2723 *LNCS*, pages 250-261. Springer, 2003.
2. Anthony Bucci and Jordan B. Pollack. A mathematical framework for the study of coevolution. In Kenneth A. De Jong, Riccardo Poli, and Jonathan E. Rowe, editors, *Foundations of Genetic Algorithms 7*, pages 221-235. Morgan Kaufmann, San Francisco, 2003.
3. D. Cliff and G. F. Miller. Tracking the red queen : Measurements of adaptive progress in co-evolutionary simulations. *Lecture Notes in Computer Science*, 929:200-218, 1995.
4 Edwin D. de Jong and Jordan B. Pollack. Learning the ideal evaluation function. E. Cantú-Paz, et. al, editors, *GECCO-2003*, volume 2723 of *LNCS*, pages 274-285, Chicago, 12-16 2003. Springer-Verlag.
5. Sevan G. Ficici and Jordan B. Pollack. A game-theoretic memory mechanism for coevolution. In E. Cantú-Paz, et. al, editors, *GECCO-2003*, volume 2723 of *LNCS*, pages 286-297, Chicago, 12-16 2003. Springer-Verlag.
6. Dario Floreano and Stefano Nolfi. God save the red queen! competition in co-evolutionary robotics. In John R. Koza, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max Garzon, Hitoshi Iba, and Rick L. Riolo, editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 398-406, Stanford University, CA, USA, 13-16 1997. Morgan Kaufmann.
7. Christopher D. Rosin and Richard K. Belew. New methods for competitive coevolution. *Evolutionary Computation*, 5(1):1-29, 1997.
8. Karl Sims. Evolving 3d morphology and behavior by competition. In Rodney A. Brooks and Pattie Maes, editors, *Proceedings of the 4th International Workshop on the Synthesis and Simulation of Living Systems*, pages 28-39, Cambridge, MA, USA, 7 1994. MIT Press.
9. Kenneth O. Stanley and Risto Miikkulainen. The dominance tournament method of monitoring progress in coevolution. In Alwyn M. Barry, editor, *GECCO 2002: Proceedings of the Bird of a Feather Workshops*, pages 242-248, New York, 8 2002. AAAI.
10. R. A. Watson and J. B. Pollack. Coevolutionary dynamics in a minimal substrate. In Lee Spector, et. al, editors, *Proceedings of the 2001 Genetic and Evolutionary Computation Conference*. Morgan Kaufmann, 2001.

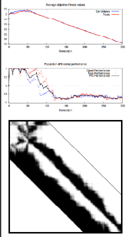


Cycling.
Proportional coevolutionary algorithm (top left and above) and Pareto hill-climbing algorithm (top right) on *rock-paper-scissors* domain.

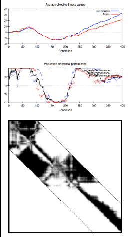
Arms-race dynamics.
Pareto hill-climbing algorithm [1] on *compare-on-one numbers game* [4] domain.



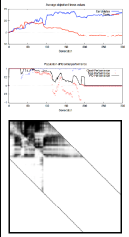
Lock-in failure.
Fitness-proportional coevolutionary algorithm on *intransitive numbers game* domain.



Variation.
Fitness-proportional coevolutionary algorithm on *intransitive numbers game* domain.



Disengagement.
Fitness-proportional coevolutionary algorithm on *intransitive numbers game* domain.


Average objective fitness values
Candidates
Tests
Generation
Population-differential performance
Cand-Performance
Test-Performance
PC-Performance