Department of Computer Science
George Mason University Technical Reports

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/ 703-993-1530

Large Scale Empirical Analysis of Cooperative Coevolution

Sean Luke, Keith Sullivan, and Faisal Abidi
sean@cs.gmu.edu, ksulliv1@cs.gmu.edu, fabidi@gmu.edu

Technical Report GMU-CS-TR-2011-2

Abstract

We present a study of cooperative coevolution applied to moderately complex optimization problems in large-population environments. The study asks three questions. First: what collaboration methods perform best, and when? Second: how many subpopulations are desirable? Third: is it worthwhile to do more than one trial per fitness evaluation? We discovered that parallel methods tended to work better than sequential ones, that "shuffling" (a collaboration method) predominated in performance in more complex problems, that more subpopulations generally did better, and that more trials performed marginally better.

1 Introduction

Cooperative coevolution [14] is an evolutionary algorithm framework which breaks a population into several subpopulations, each optimizing a sub-part of the complete solution. An individual in a subpopulation is tested by grouping it with one individual from each of the other subpopulations (known collectively as its *collaborators*) to form a complete candidate solution, which is then assessed. We call this process a *trial*. The fitness of the individual is then *evaluated* from a single trial or from the best or average result of a series of trials.

The promise of cooperative coevolution is that it can take advantage of problems which are *partially decomposable*: breakable into separate subproblems which can be somewhat, though not entirely, optimized independently due to low linkage between variables across subproblem boundaries. In such cases, cooperative coevolution projects the joint optimization space into much smaller independent subspaces.

There are many such problems. For example, consider a soccer team: we are evolving attacker, midfielder, defender, and goalie behaviors. The strongest linkages exist within individual behaviors, and those behaviors may to some degree be optimized independently. However, they benefit from being optimized jointly, such as goalies optimized with defenders.

This promise comes with a downside: by projecting the joint space, we are throwing considerable information away. As a result, cooperative coevolution may fall victim to pathologies such as *relative overgeneralization* [17], causing it to gravitate towards broad suboptimal peaks surrounding Nash Equilibria in the joint space. Cooperative coevolution may also be subject to *miscoordination* [8]. As we set the fitness to the maximum of more and more trials, cooperative coevolution in theory loses these pathologies. Unfortunately, trials are the currency of stochastic optimization, and so large numbers of trials per fitness assessment typically means fewer fitness assessments, either via smaller populations or fewer generations.

Cooperative coevolution entails many parameters beyond those common to evolutionary computation: one must also decide on how collaborators are chosen, how fitness is determined, how many subpopulations to use, how many trials to perform, and so on. Empirical studies of these parameters have often made simplifying assumptions, such as easy problems or small genomes, small population sizes, binary representations, and often two subpopulations. We wished to examine cooperative coevolution in problems and scenarios more closely matching its promise.

In this paper we test certain coevolutionary parameters against several problems of varying complexity, all with a moderately large genome size (100), with large numbers of trials (one million per run), and with the large populations made possible by current computer hardware and multithreading. We asked the following:

- Which collaboration schemes work best?
- Assuming the problem constraints permit them, are more subpopulations helpful?
- Are more trials helpful?

This report is an expansion on a paper of the same name at the GECCO 2011 conference [7].

Problem	*Description*	*Bounds*
Sum	$\sum_{i=1}^{n} x_i$	$-5.12 \leq x_i \leq 5.12$
Median	$\text{median}(x_1, ..., x_n)$	$-5.12 \leq x_i \leq 5.12$
Min	$\min(x_1, ..., x_n)$	$-5.12 \leq x_i \leq 5.12$
Rastrigin*	$10n \sum_{i=1}^{n} x_i^2 - 10\cos(2\pi x_i)$	$-5.12 \leq x_i \leq 5.12$
Schwefel*	$\sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	$-512.03 \leq x_i \leq 511.97$
Rosenbrock*	$\sum_{i=1}^{n-1} (1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2$	$-2.048 \leq x_i \leq 2.048$
Rotated Rastrigin*	$\text{Rastrigin}(\text{rotate}(x_1, ..., x_n))$	$-5.12 \leq x_i \leq 5.12$
Rotated Schwefel*	$\text{Schwefel}(\text{rotate}(x_1, ..., x_n))$	$-512.03 \leq x_i \leq 511.97$

Table 1: Test problems used in the study. * indicates a minimization problem: the study negates all minimization problems to make them maximization problems for consistency.

The remainder of this paper discusses related work, then lays out empirical experiments which examine these issues.

2 Related Work

As originally formulated [13, 14] cooperative coevolution was *sequential*: each generation only one subpopulation (selected round-robin) would be evaluated, then breed and update itself. The obvious alternative is to perform *parallel* coevolution, where all subpopulations would be evaluated and bred each generation. There have been only a few comparisons of the two [6, 12]. Somewhat in-between have been various "punctuated" approaches [9, 11, 15]. For example, the population might be evaluated jointly, then broken into subpopulations, with each subpopulation evaluated and breed for some number of generations, always using the best collaborators from the earlier parallel evaluation. Eventually the populations would be joined up again and the cycle would continue anew.

Another aspect of cooperative coevolution is how collaborators are chosen to test an individual. Nearly all literature either uses the best-performing collaborators from the previous generation (commonly known as *single-best*), or selects collaborators at *random* with replacement. Here again empirical analysis has been sparse (notably [19, 18]).

Though seminal work [13] employed many subpopulations, much later cooperative coevolution analysis has assumed two subpopulations. However as cooperative coevolution has been applied to increasingly complex problems, a recent trend has seen three [14, 18], ten [15, 20], 30 [21], and even 1000 [3] subpopulations, sometimes dynamically reorganized to identify linkages. Not much empirical work has directly examined the effect of many subpopulations.

Finally, how many trials should be applied before determining the fitness of an individual? The original cooperative coevolution papers compared two methods: CCEA-1 (a single trial using single-best) and CCEA-2 (two trials: one with the single-best and one with a random collaborator). In certain basic test problems it has been shown that the convergence properties improve with more trials [8], leading to various approaches to maximize the impact of trials by forming archives of collaborators [2, 4, 8].

Once trials are completed, how should an individual's fitness be assessed? Some literature has examined basing the fitness on the worst or mean of the trials (for example [21]), but it's since been established that the maximum over trials best overcomes known pathologies [8].

For a survey of cooperative coevolution and current theoretical and empirical issues, see [10].

3 Experimental Parameters

Evolutionary computation entails many parameters, and coevolution only increases that number. We cannot examine all their combinations. As such we held the following parameters fixed. All problems had $n = 100$ variables, and were run for one million trials, distributed in various ways. Breeding used tournament selection (of size 2), then one-point crossover, then gaussian mutation ($\sigma = 0.01$), reselecting mutation until the new gene was within valid gene boundaries.

We ran each experiment for 100 independent runs. Statistical significance tests were performed using one-way ANOVAs each set to $p = 0.00025$ in order to retain slightly better than $p = 0.05$ for the paper as a whole. Experiments were performed using ECJ [1]. We then varied the following parameters:

Coevolved Subpopulation Sizes Given the complexity of the problems and the wide availability of multiple cores, we chose to use a large population: $p = 1000$ individuals. The population was then nominally divided into subpopulations to perform coevolution. This size also allowed us to examine larger numbers of subpopulations. We examined $s = 1, 2, 4, 10, 50$, and 100 subpopulations (with p/s individuals per subpopulation,

Problem	*Number of Subpopulations*				
	2	*4*	*10*	*50*	*100*
Sum	SB PB	SB PB	PB SB	SB PB PP PG	PB SB PP PG
Median	PB SB PP	SB PB	SB PB	SB PB PP PT PR PG	SB PB PP PT PR PG
Min	PP PG	PG PP	PG	PG	PG
Rosenbrock	PP	PP	PP	PP	PP
Rastrigin	PP	PP	PP	PP	PP
Schwefel	PP	PP	PP	PP	PP
Rotated Rastrigin	PP	PP	PP	PP	PP
Rotated Schwefel	PT PR PG*	PG	PG	PG	PG

Legend: SB: Sequential Single Best PB: Parallel Single Best PP: Parallel Shuffled Pops PG: Parallel Shuffled Gens PT: Parallel Shuffled Trials PR: Parallel Random

Table 2: Summary of Results for First Experiment. Each entry shows the best-performing collaboration schemes with no statistically significant differences among them, in decreasing order of performance. Note * where PR and SB are also not significantly different; and PG, SB, PP, and PB are not significantly different.

except in one case described next). Individuals in each subpopulation had genome lengths of n/s respectively.

Collaboration Schemes and Update Timing We tested using both *parallel* and *sequential* update timing. Parallel update timing runs lasted for $g = 1000$ generations, except with one exception discussed next. All sequential update timing runs lasted for $g \times s$ generations. Collaborators were chosen using one of eight collaboration schemes based on the following criteria:

- Individuals were tested with the *single best* collaborators of the prior generation from other subpopulations.
- Individuals were tested with collaborators chosen at *random*, with replacement, from the current individuals of other subpopulations. As a result while all individuals would have at least one trial, some individuals may also have been selected as collaborators in several additional trials. Individuals' fitnesses are set to the maximum over all trials in which they appeared.
- In a method of our own devising, individuals in each subpopulation were randomly *shuffled*, then individuals were paired off: for each i, individuals indexed i from each subpopulation were tested together. If the update timing was parallel, a single shuffling would result in p/s total trials: the surplus in trials could be redistributed using *larger subpopulation sizes* (p individuals each), *more generations* ($g \times s$ generations), or *more trials* (s per individual, with new shufflings per trial). Sequential updates on the other hand do not result in redistribution options.

In summary, we had eight combinations of update timing and collaboration schemes in total: Sequential Single Best, Sequential Random, Sequential Shuffled, Parallel Single Best, Parallel Random, Parallel Shuffled Pops, Parallel Shuffled Gens, and Parallel Shuffled Trials. In the third experiment, we then varied certain of these schemes to allow for more trials per fitness evaluation.

Test Problems Table 1 shows the problems used in the study, chosen for their different traits:

- *Sum* is simply the sum of the variables. It is linearly separable.
- *Min* and *Median* are somewhat more difficult than *Sum*: in *Median*, improving a variable will not improve fitness unless the amount changes enough that the variable crosses the median. In *Min*, improving a variable will not improve fitness unless it is the minimum-value variable. However in either case improving a variable will never *worsen* fitness.
- *Rastrigin*, *Schwefel*, and *Rosenbrock* are the standard functions. Rastrigin is additively separable.
- *Rotated Rastrigin* and *Rotated Schwefel* are randomly rotated versions of these functions, in order to increase linkage among variables. A single fixed $m \times m$ rotation matrix M was used for all problems. The rotation matrix was generated in a fashion similar to that described in [5]: first, all entries in M were produced using random standard-normal gaussian noise. Then for i from 1 to m, each row M_i was decreased by $\sum_{j=i}^{m} \langle M_i \, M_j \rangle M_j$, then renormalized.

4 First Experiment: Collaboration Scheme Choice

We began by comparing the eight collaboration schemes in the context of all combinations of problem type and

number of subpopulations. We asked: which collaboration schemes worked best and in which situations?

We started with a one-population baseline GA with a population size of 1000, run for 1000 generations. This came to 1,000,000 trials per run. When performing coevolution, these trials may be divvied up in different ways depending on the collaboration scheme. Our eight schemes result in different distributions of generations, subpopulation sizes, and minimum number of trials per individual to compute fitness. For example, for 2-subpopulation coevolution, the distribution was:

Collaboration Scheme	*Gens*	*Subpop-Size*	*Subpops*	*Trials*
Parallel Single Best	1000	500	2	1
Parallel Random	1000	500	2	1
Parallel Shuffled Pops	1000	1000	2	1
Parallel Shuffled Gens	2000	500	2	1
Parallel Shuffled Trials	1000	500	2	2
Sequential Single Best	2000	500	2	1
Sequential Random	2000	500	2	1
Sequential Shuffled	2000	500	2	1

At the extreme other end, the 100-subpopulation coevolution distribution was:

Collaboration Scheme	*Gens*	*Subpop-Size*	*Subpops*	*Trials*
Parallel Single Best	1000	10	100	1
Parallel Random	1000	10	100	1
Parallel Shuffled Pops	1000	1000	100	1
Parallel Shuffled Gens	100000	10	100	1
Parallel Shuffled Trials	1000	10	100	100
Sequential Single Best	100000	10	100	1
Sequential Random	100000	10	100	1
Sequential Shuffled	100000	10	100	1

Observation Abridged results are shown in Figures 1 and 2 and summarized in Table 2. We had hypothesized that single-best strategies would work well for simple, linearly separable problems, but less well for others. In fact this is what occurred: Parallel Single Best and Sequential Single Best both yielded the best results for the two easiest problems (Sum and Median) but failed to impress anywhere else.

What was surprising was the near dominance of the remaining problems by Parallel Shuffled Pops and, to a lesser extent, Parallel Shuffled Gens. Parallel Shuffled Pops did best on Rosenbrock, Rastrigin, Schwefel, and Rotated Rastrigin, and performed well in Min, Median, and Sum. In fact the *only problem* where Parallel Shuffled Pops did not perform well was in Rotated Schwefel. Parallel Shuffled Gens did well in Min and in Rotated Schwefel.

The sole exception was Schwefel with 2 subpopulations, where Parallel Shuffled Trials and Parallel Random performed best. In nearly all problems, the baseline (1-population) GA was middling. Sequential Shuffled and Sequential Random were poor performers There was no result in which a sequential scheme statistically significantly outperformed parallel schemes, and many cases where the opposite was true.

Problem	*Collaboration Scheme*	*Subpops*
Sum	Sequential Single Best	2 4 10 50 100
Median	Sequential Single Best	2 4 10 $\overline{50\ 100}$
Min	Parallel Shuffled Gens	2 4 $\overline{100\ 50\ 10}$
Rosenbrock	Parallel Shuffled Pops	2 4 $\overline{10\ 50\ 100}$
Rastrigin	Parallel Shuffled Pops	2 4 10 $\overline{50\ 100}$
Schwefel	Parallel Shuffled Pops	2 4 $\overline{10\ 50}$ 100
Rotated Rastrigin	Parallel Shuffled Pops	2 4 10 $\overline{50\ 100}$
Rotated Schwefel	Parallel Shuffled Gens	2 4 10 $\overline{100\ 50}$

Table 3: Summary of Results for the Second Experiment. Subpopulations are ordered in increasing performance. Overbars indicate statistically insignificant differences.

5 Second Experiment: Number of Subpopulations

As the number of subpopulations grows, cooperative coevolution begins to look similar (in an important sense) to univariate estimation of distribution algorithms (EDAs), with which it shares much in common [16]. As they assume low linkage, univariate EDAs perform well on separable problems; and so we hypothesized the same would occur with large numbers of subpopulations. Correspondingly we assumed that high-linkage problems (rotated Rastrigin and rotated Schwefel) would prove difficult for large numbers of subpopulations.

Observation The results were very surprising. Generally speaking, as the number of subpopulations increased, the performance order of the collaboration schemes stayed roughly the same, but the variance among results increased dramatically. This can be seen in Figures 1 and 2. The poorer-performing collaboration schemes generally performed *much worse*, but more importantly, the highest-performing collaboration schemes consistently performed *even better*. We do not have an explanation for this phenomenon, but it is encouraging: *if* you pick a good method, subdividing into more populations will help.

We performed statistical significance tests across numbers of subpopulations when solely using the best collaboration scheme for a given problem. These results are summarized in Table 3. As can be seen, at least up through 50 subpopulations, high-performing collaboration schemes get better and better, even on the high-linkage problems tested.

6 Third Experiment: Extra Trials

The prevailing wisdom regarding cooperative coevolution is that it may be worthwhile to perform multiple trials to compute a fitness evaluation, because individuals are evaluated in the context of other individuals who may or may not prove to be worthwhile collaborators

Problem	*Chosen Collaboration Scheme*	*Number of Subpopulations*				
		2	*4*	*10*	*50*	*100*
Sum	Parallel Single Best (PB)	PB	PB	PB	PB	PB
Median	Parallel Single Best (PB)	PB	PB	PB	PB	PB 2G
Min	Parallel Shuffled Gens (PG)	PG	2P PG	2P PG	2G	2G
Rosenbrock	Parallel Shuffled Pops (PP)	PP 2G	2G PP	2G	2G	2G
Rastrigin	Parallel Shuffled Pops (PP)	PP 2G	2G PP	2G	2G	2G
Schwefel	Parallel Shuffled Pops (PP)	PP	PP	PP	PP 2G	PP 2G
Rotated Rastrigin	Parallel Shuffled Pops (PP)	2G PP	2G	2G	2G	2G
Rotated Schwefel	Parallel Shuffled Gens (PG)	2G 2P*	2P PG	2P PG	PG 2P 2G	2P 2G PG

Legend: PB: Parallel Single Best PP: Parallel Shuffled Pops PG: Parallel Shuffled Gens
2P: 2-evaluation, half-subpopulation size version of PB (single best + 1 random), PP, or PG
2G: 2-evaluation, half-generations version of PB (single best + 1 random), PP, or PG

Table 4: Summary of Results for the Third Experiment. For each problem we compared the best performing collaboration scheme (PB, PG, PP, as shown in the *Scheme* column) against two kinds of two-trial variations of that scheme. Each entry shows the best-performing collaboration schemes with no statistically significant differences among them, in decreasing order of performance. See text for details on 2G and 2P variations. Note * where 2P is not significantly different from PG (but 2G is).

[8]. However increasing the number of trials per fitness evaluation is expensive.

We wondered if this prevailing wisdom was true. To test this we selected the best collaboration method for each problem. These methods originally had used one trial each. We then constructed two-trial variations of these methods and compared them to the originals.

For the Sum and Median problems, we chose Parallel Single Best as the collaboration method to test, and compared it against a variation where an individual is tested against the single best collaborators and also against randomly-chosen collaborators (so-called "CCEA-2" from [13]). For the Min and Rotated Schwefel problems, we chose Parallel Shuffled Gens as the collaboration method to test, and compared it against a variation where two randomly shuffled trials were performed. For the remaining problems we chose Parallel Shuffled Pops as the collaboration method to test, again compared against a variation where two randomly shuffled trials were performed.

In all cases, fitness was assessed as the maximum over the trials. In order to retain the same number of total trials but perform two trials per evaluation, something must be halved. For each variation, we tried halving the subpopulation size and halving the total number of generations. In Table 4 these are referred to as the "2P" and "2G" subvariations. Note that for 2 subpopulations, PG/2G and PP/2P are equivalent to PT.

Observation Table 4 shows the results. We expected that for separable problems (such as Sum, and to a lesser extent Median) more trials would provide no benefit: and this was in fact the case. Interestingly, in Schwefel and Rotated Schwefel more trials weren't useful either.

In the remaining problems the clear pattern was that, with more subpopulations, it was useful to have more trials. This made sense, as more subpopulations meant more collaborators to add contextual noise to the fitness evaluation. We note that it was universally as good or better to halve the generations rather than to halve the population size.

Even so, the improvements, while statistically significant, were rarely large. Figure 3 shows situations selected from those where two trials were preferred over one trial. As can be seen, the differences were not as dramatic as in the other experiments.

We note that for Rotated Schwefel, 2 subpopulation, PG/2G (that is, PT) was statistically significantly better than PG, where as in the first experiment, it was not.

7 Conclusions and Future Work

This paper presented a study of parameter settings for cooperative coevolution for moderately large problems, and using large populations. In three experiments we examined various collaboration methods (including *shuffling*, an approach of our own devising), subpopulation numbers ranging from 1 to 100, and the addition of more trials per fitness evaluation. A summary of our findings:

- Shuffling methods allow redistribution of trials for other purposes, such as longer generations or larger population sizes: and this paid off. While "single-best" methods performed well in simple, separable problems, shuffling methods (notably Parallel Shuffled Pops) predominated in more complex methods.
- Sequential update timing was generally equaled or outperformed by its parallel counterpart. This is particularly good news, as parallel update timing

can better take advantage of parallel architectures. This corroborates earlier studies, such as [12].

- Even for strongly linked problems, more subpopulations was better.
- In the more complex problems, doubling the trials per Parallel Shuffled Pops fitness evaluation, and halving the generations, seemed worthwhile. The improvement, however, was not large, unlike in [8]. This may cast some doubt on the need for complex archive methods (as in [2, 4, 8]).

There are many further parameters to explore as future work: for example, to reexamine these parameters for smaller population sizes or methods other than the genetic algorithm (like differential evolution). Some recent work has taken to more exotic variations [3, 15, 20], and it is not yet known if these result hold in those cases.

8 Acknowledgment

This research was supported by NSF Grant 0916870.

References

[1] ECJ 20: An evolutionary computation library in Java. Available at http://cs.gmu.edu/∼eclab/projects/ecj/, 2011.

[2] Anthony Bucci and Jordan B. Pollack. On identifying global optima in cooperative coevolution. In *Proc. Genetic and Evolutionary Computation Conference*, 2005.

[3] Wenxiang Chen, Thomas Weise, Zhenyu Yang, and Ke Tang. Large-scale global optimization using cooperative coevolution with variable interaction learning. In *Proc. Parallel Problem Solving from Nature*, 2011.

[4] Edwin D. de Jong. Towards a bounded pareto-coevolution archive. In *Proc. IEEE Congress on Evolutionary Computation*, 2004.

[5] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[6] Thomas Jansen and R. Paul Wiegand. Sequential versus parallel in cooperative coevolutionary (1+1) EAs. In *Proc. IEEE Congress on Evolutionary Computation*, pages 30–37, 2003.

[7] Sean Luke, Keith Sullivan, and Faisal Abidi. Large scale empirical analysis of cooperative coevolution. In *Proc. Genetic and Evolutionary Computation Conference*, 2011.

[8] Liviu Panait. *The Analysis and Design of Concurrent Learning Algorithms for Cooperative Multiagent Systems*. PhD thesis, George Mason University, Fairfax, Virginia, 2006.

[9] Gary B. Parker and H. Joseph Blumenthal. Comparison of sampling sizes for the co-evolution of cooperative agents. In *Proc. IEEE Congress on Evolutionary Computation*, pages 536–543, 2003.

[10] Elena Popovici, Anthony Bucci, R. Paul Wiegand, and Edwin D. de Jon. Coevolutionary principles. In G. Rozenberg, T. Baeck, and J.N. Kok, editors, *The Handbook of Natural Computing*. Springer, 2010.

[11] Elena Popovici and Kenneth De Jong. The effects of interaction frequency on the optimization performance of cooperative coevolution. In *Proc. Genetic and Evolutionary Computation Conference*, pages 353–360, 2006.

[12] Elena Popovici and Kenneth De Jong. Sequential versus parallel cooperative coevolutionary algorithms for optimization. In *Proc. IEEE Congress on Evolutionary Computation*, pages 1610–1617, 2006.

[13] Mitchell A. Potter. *The Design and Analysis of a Computational Model of Cooperative Coevolution*. PhD thesis, George Mason University, Fairfax, VA, 1997.

[14] Mitchell A. Potter, Lisa A. Meeden, and Alan C. Schultz. Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In *Proc. International Conference on Artificial Intelligence*, 2001.

[15] Tapabrata Ray and Xin Yao. A cooperative coevolutionary algorithm with correlation based adaptive variable partitioning. In *Proc. IEEE Congress on Evolutionary Computation*, pages 983–989, 2009.

[16] Christopher Vo, Liviu Panait, and Sean Luke. Cooperative coevolution and univariate estimation of distribution algorithms. In *Proc. Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms*, pages 141–150. ACM, 2009.

[17] R. Paul Wiegand. *An Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, George Mason University, 2004.

[18] R. Paul Wiegand, William C. Liles, and Kenneth A. De Jong. The effects of representational bias on collaboration methods in cooperative coevolution. In *Proc. Parallel Problem Solving from Nature*, 2002.

[19] R. Paul Wiegand, Willian C. Liles, and Kenneth A. De Jong. An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. In *Proc. Genetic and Evolutionary Computation Conference*, 2001.

[20] Zhenyu Yang, Jingqiao Zhang, Ke Tang, Xin Yao, and Arthur C. Sanderson. An adaptive coevolutionary differential evolution algorithm for large-scale optimization. In *Proc. IEEE Congress on Evolutionary Computation*, pages 102–109, 2009.

[21] Chern Han Yong and Risto Miikkulainen. Cooperative coevolution of multi-agent systems. Technical Report AI07-338, Department of Computer Sciences, The University of Texas at Austin, 2001.

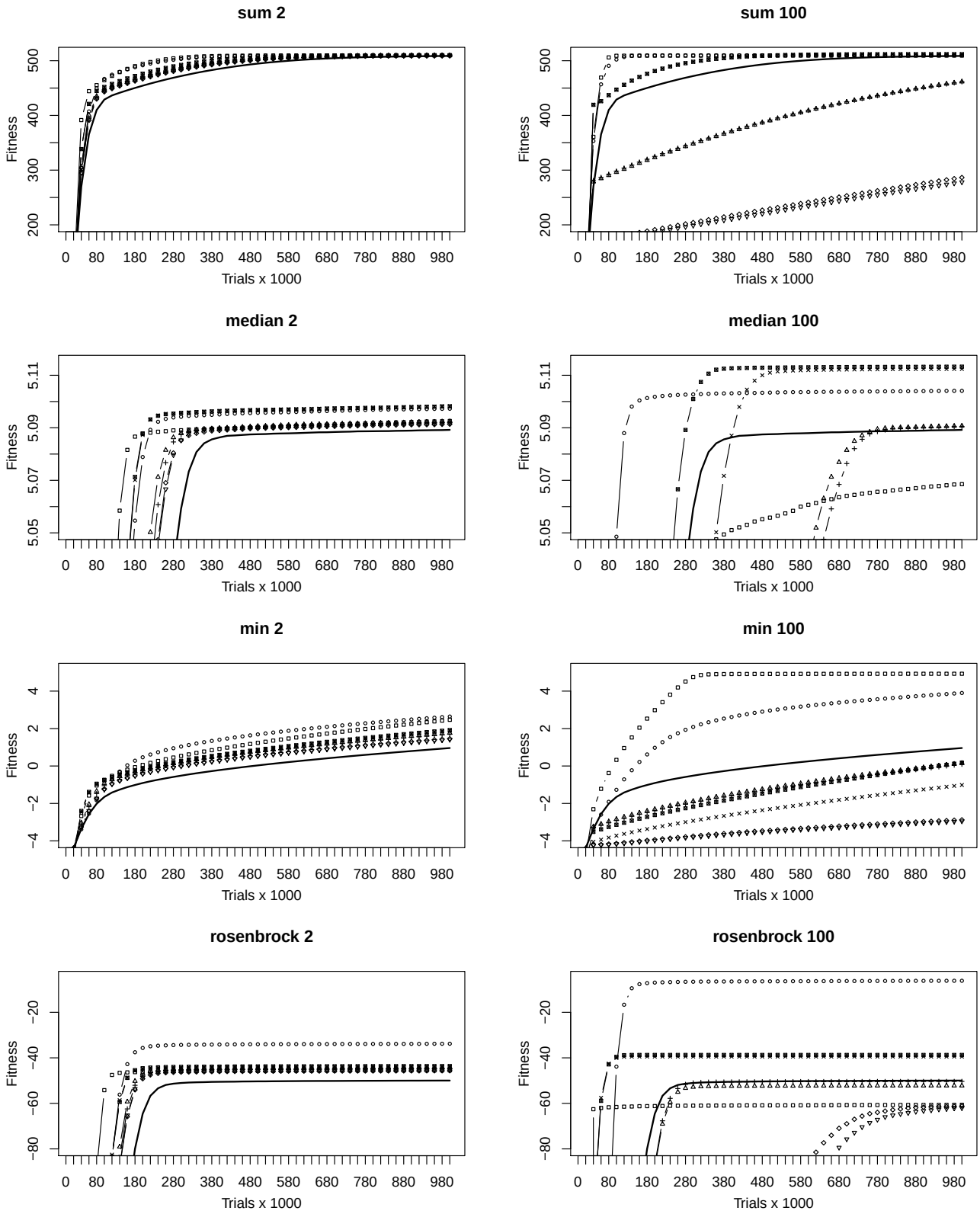

Figure 1: Experiment 1 mean best fitness results for Sum, Median, Min, and Rosenbrock. Subpopulations of 2 and 100 shown.

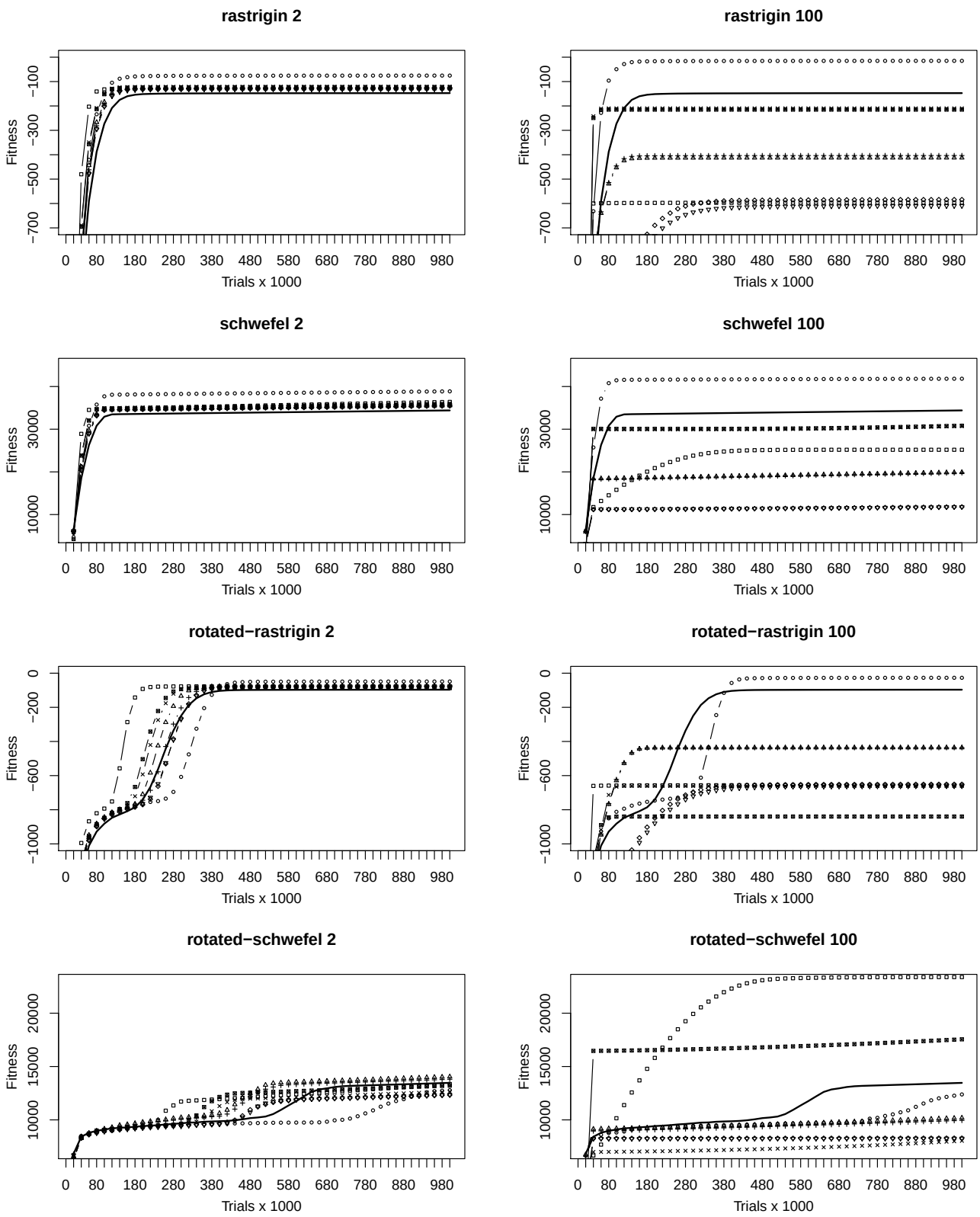

Figure 2: Experiment 1 mean best fitness results for Rastrigin, Schwefel, and rotated versions. Subpopulations of 2 and 100 shown.

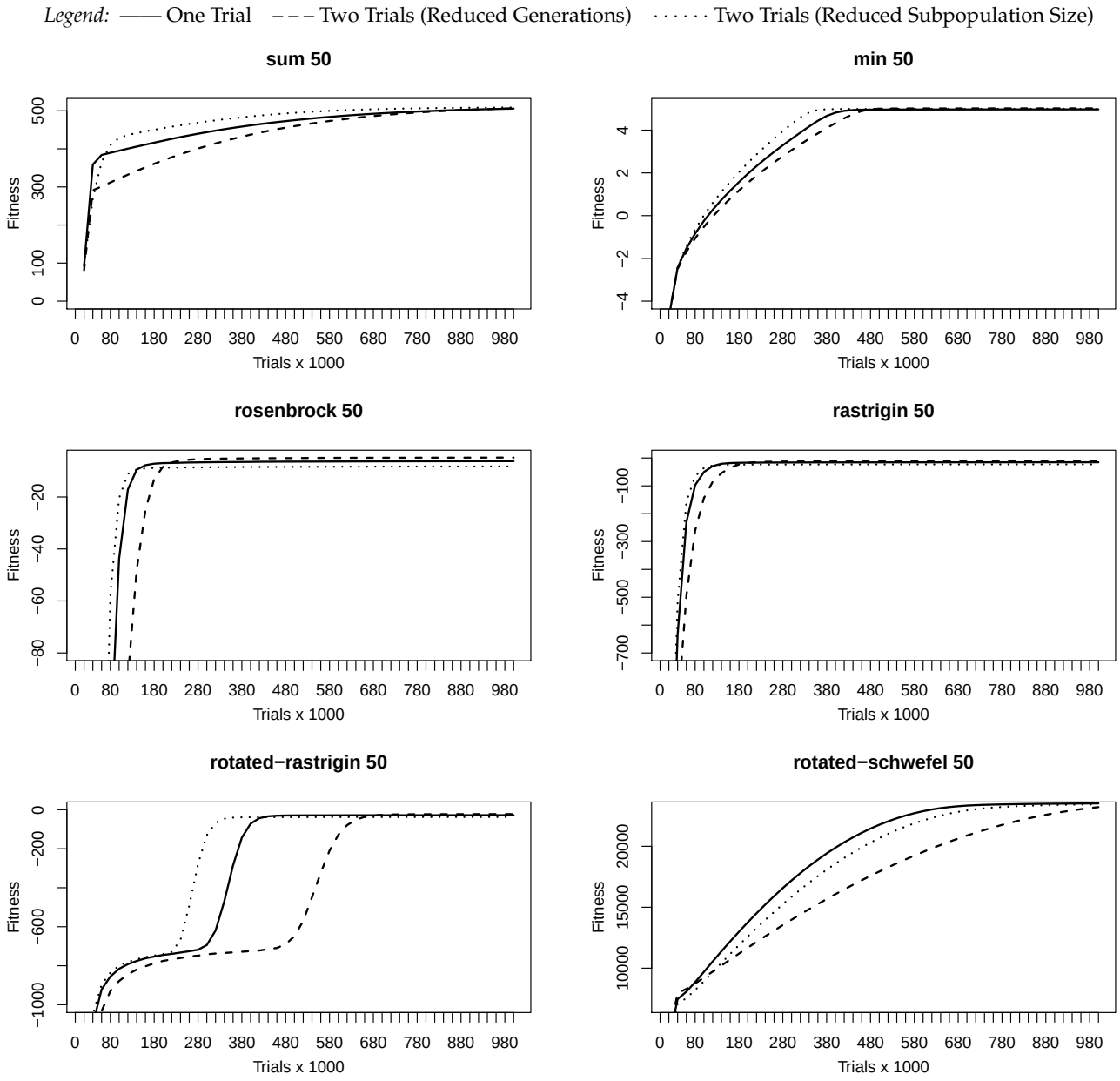

Figure 3: Experiment 4 selected mean best fitness results showing typical situations where two trials outperformed one trial.