Competitive Coevolution for Defense and Security: Elo-Based Similar-Strength Opponent Sampling

Sean N. Harris
BONSAI Lab
Auburn University
Auburn, Alabama, U.S.A.
snh0037@auburn.edu

Daniel R. Tauritz
BONSAI Lab
Auburn University
Auburn, Alabama, U.S.A.
dtauritz@acm.org

Abstract

Competitive coevolution is an important technique for fields such as defense and security which inherently involve adversarial games. One advantage that the field of computer security has in particular is that games in this space are often naturally able to be simulated at a high fidelity by interacting with the involved software or hardware directly. However, such high-fidelity evaluations are typically slow, so it is especially important in these cases to get as much useful information out of as few evaluations as possible. This paper proposes a new competitive coevolutionary evaluation method of Similar-Strength Opponent Sampling, which selects opponent pairings of similar skill levels so that evaluations can more efficiently distinguish the performances of similar individuals. This is enabled through the use of Elo ratings as a surrogate fitness function that prevents bias against individuals assigned stronger opponents. Care is taken to ensure that this technique is applicable to complex games where there is no explicit winner or loser, allowing ratings to be based on relative fitness. Mixed results are presented, showing that significant benefits are gained from pairing similar-strength opponents, but finding that the use of Elo rating instead of raw fitness harms evolution for intransitive games.

CCS Concepts

• **Computing methodologies → Genetic algorithms**; **Adversarial learning**; • **Theory of computation → Evolutionary algorithms**.

Keywords

Evolutionary Algorithms, Competitive Coevolution, Competitive Co-evolution, Elo Rating System

ACM Reference Format:
Sean N. Harris and Daniel R. Tauritz. 2021. Competitive Coevolution for Defense and Security: Elo-Based Similar-Strength Opponent Sampling. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3449726.3463193

1 Introduction

Competitive coevolution is a natural technique to apply to the adversarial games commonly found in defense and security, in which real-world attackers and defenders are constantly improving their technical and strategic capabilities in order to gain supremacy over their opponents. Much existing research has found great success by employing competitive coevolutionary algorithms on these scenarios [11, 20, 25]. When applying this to cyber-security games, practitioners additionally have the extremely powerful option of evaluating their evolved strategies directly against a high-fidelity representation of their real-world use case, by the nature of such problems typically consisting of software or hardware which evolutionary algorithms can interface with. When using these methods for evolution, evolved strategies can be expected to be much more applicable to the real world than those evolved for a hand-designed simulation which might be missing subtle but important details that affect performance. Existing work has explored the effectiveness of this methodology [9, 14, 28], but a typical downside noted is the greatly increased wall time cost of running evaluations in real hardware or software, since unlike in a simulation such an environment can not easily be sped up past real-time [27]. It is therefore particularly important to develop ways to speed up competitive coevolution for this category of games in order to enable better use of such high-fidelity evaluation tools.

One of the primary difficulties encountered in competitive coevolutionary algorithms is determining solution quality. Since there is generally no exact fitness function to score an individual, fitness must be measured using the relative performance of opponents competing in the same environment. While the hypothetical ideal is to evaluate each individual against all possible opponents, such an approach is rarely feasible. Instead, individuals are evaluated against a small subset of theoretical opponents through competitive coevolution. Even then, a full round-robin comparison of an individual to the entire population of opponents is extremely costly, meaning that individuals are usually only tested against a very small number of opponents. Each opponent that an individual is evaluated against further improves the fidelity of the calculated fitness, but the expense in computation increases as well. Methods of increasing fidelity in fitness while minimizing the impact to computation time are particularly valuable. A variety of different methods have been developed to improve the running time of competitive coevolution, such as surrogate fitness functions or fitness estimation techniques that take advantage of expected transitivity in competitive outcomes, as well as the use of more sophisticated methods of evaluation than random opponent sampling.

This paper introduces a novel method of sampling opponents by pairing solutions with opponents found to have similar performance in past evaluations (Similar-Strength Opponent Sampling, or SSOS). This technique is intended to reduce the number of needed evaluations per generation of evolution by using more intelligent pairing of opponents to more efficiently distinguish individuals of similar skill level. We hypothesize that by preferentially comparing individuals that are similar in ability, more useful information about their quality can be obtained per evaluation than would be obtained by random sampling, which often pairs individuals against opponents for whom the outcome of their competition is a foregone conclusion based on previous evaluations. However, similarity in capability can not be assumed from similarity in fitness, because the fact that strong individuals are being tested against strong opponents means that their fitnesses will be artificially low, and the fitnesses of weak individuals against weak opponents artificially high. To remedy this we propose the application of the Elo rating system [6], a method which rates the skill of competing players such that the difference in rating between two players predicts the expected score achieved in competition between the two, and uses observed deviation from these expected scores to update its ratings. The Elo rating becomes a surrogate fitness function for evolution, resulting in a combined method of Elo-Based Similar-Strength Opponent Sampling (EBSSOS).

Many of the design decisions made as part of this research are motivated in particular by the game-theoretic nature of security and defense scenarios: such games are frequently asymmetric, non-zero-sum, and multiobjective, and most existing work is incapable of simultaneously handling these complications. For example, defenders typically have priorities beyond thwarting potential attackers, such as maintaining usability of their systems or minimizing cost; a heavy-handed defense that disables all functionality might successfully thwart attacks, but is generally not an acceptable option. Since many attacker missions are not aimed at degrading those objectives, this results in an asymmetric, non-zero-sum, multiobjective game that presents complications for algorithm design and means that many existing methods of speeding up competitive coevolutionary algorithms can not be directly applied to security and defense games. The specific formulation of the Similar-Strength Opponent Sampling algorithm presented in this paper concerns zero-sum, single objective games, but is designed to be extensible to these non-zero-sum, multiobjective environments [10].

2 Background

Here we describe the mathematics of the Elo rating system and the problem domains used to evaluate EBSSOS in this paper.

2.1 Elo Rating System

The Elo rating system was originally developed by Arpad Elo [6] for rating the skill level of chess players based on the outcome of their games against other players. The premise of the Elo rating system is that, given the skill ratings of two players, one can calculate the expected score of a game between the two. When the actual result of a game deviates from the expected score, it is then possible to update those players' ratings to account for this. Elo ratings most commonly use the base $\sqrt{10}$ logistic function to model the expected score $P(D)$ given a rating difference of D, defined by the following function[1], where expected score is normalized to $[0, 1]$:

$$P(D) = \frac{1}{1 + 10^{-D/(2\cdot 200)}} \tag{1}$$

In games like chess which lack score as a gameplay element, the score values of 1 for a win, $\frac{1}{2}$ for a draw, and 0 for a loss are conventionally assigned. Through this, larger differences in rating are expected to produce larger differences in score. To update ratings, Elo gives the equation:

$$R_n = R_o + K(W - W_e) \tag{2}$$

where R_o and R_n are the old and new Elo ratings, W and W_e are the actual and expected game scores, and K is a learning rate parameter called the K-factor, representing how much each update should affect the ratings (and the maximum possible update). Reasonable K-factors vary depending on application, but values of 10 to 32 are common in chess. Often, different categories of players are given different K-factors, depending on how quickly their ratings or actual skills are expected to change.

Elo additionally provides an algorithm to calculate ratings for a group of unrated players, which he refers to as "the method of successive approximations". This algorithm can work to statically calculate a rating given a series of past games. The premise of this method is that, in the same way that the logistic function can be used to estimate an expected score given a rating difference, the inverse logistic function (sometimes called the logit function) can be used to estimate an expected rating difference given a score. The inverse of the above logistic function is defined by the function:

$$D(P) = 200 \cdot \log_{\sqrt{10}}\left(\frac{P}{1 - P}\right) \tag{3}$$

Given a previous rating estimate for the population, each individual's rating estimate can be updated by the following:

$$R_p = R_c + D(P) \tag{4}$$

where R_p is the new rating estimate, R_c is the average of the rating estimates of all that individual's opponents, and P is the average of the normalized scores that individual has achieved. Notably, this method eliminates the sensitive K-factor parameter, since player skill is assumed to be static. During experiments we found that these successive approximations will diverge given the low number of games provided per generation, so we modify Equation 4 to:

$$R_p = \gamma \cdot R_c + D(P) \tag{5}$$

where γ is a decay value close to 1. These iterative updates are repeated until the average change in ratings between iterations is below a given convergence threshold. Through parameter tuning we found that a γ of 0.9 and a convergence threshold of 0.1 produced stable ratings after relatively few iterations. An initial rating of 0 is used so that calculated ratings will be centered around zero. Algorithm 1 gives a more detailed overview of this process.

2.2 Problem Domains

To evaluate the performance of EBSSOS, this work uses three problem domains with differing complexity, including two problems commonly used in prior research on methods of sampling opponents. Our problem domains are a Predator-Prey game with a

[1] The value 200 here is the scale of the curve, originally chosen based on pre-Elo chess tradition separating different "classes" of players into 200-point rating bands. It is typically re-used in other non-chess applications of the Elo rating system.

Algorithm 1: Method of successive approximations

```
Input: $population$, $opponents[]$, $scores[][]$,
       $convergence\ threshold$, $\gamma$, $D_{perfect}$
Output: $R_p[]$
Set all initial ratings $R_p[i] \leftarrow 0$;
repeat
    forall $i$ in population do
        $R_c[i] \leftarrow avg(R_p[j])$ for $j$ in $opponents[i]$;
        $P \leftarrow avg(scores[i][j])$ for $j$ in $opponents[i]$;
        if $P = 0$ then
            $D \leftarrow -D_{perfect}$;
        else if $P = 1$ then
            $D \leftarrow D_{perfect}$;
        else
            $R_{next}[i] \leftarrow \gamma \cdot R_c + 200 \cdot \log_{\sqrt{10}}(\frac{P}{1-P})$;
        end
    end
    $R_{last} \leftarrow R_p$;
    $R_p \leftarrow R_{next}$;
    if first round of evaluations then
        return $R_p$;
    end
until $Avg(|R_p[i] - R_{last}[i]|) <$ convergence threshold for i
  in population;
return $R_p$;
```

continuous action space and genetic programming-based agents, the simpler game of Nim with a small, discrete action space and binary genotypes, and the even simpler transitive "game" described by the Internal Rosenbrock domain, in which opponents are simply competing minimizations to the Rosenbrock function.

2.2.1 Predator-Prey We examine a two-agent Predator-Prey environment, in which a predator agent is tasked with maneuvering to intercept a prey agent, who's goal is to evade the predator. Predator-Prey scenarios (also referred to as pursuit-evasion games) are common domains for testing competitive coevolutionary algorithms, due to the complexity of available behaviors despite their simplicity. One of the first applications of these games to study competitive coevolution was by Reynolds [22], who framed it as a game of tag where agents switched roles upon capture. This environment was used as a testbed for exploring the competitive coevolution of agent behavior. Cliff and Miller [3] similarly used neuroevolution in a two-population pursuit environment to explore difficulties inherent to competitive coevolution. Pursuit-evasion games have been more broadly studied in the field of differential games, with foundational work by Rufus Isaacs [12].

Two differing formats of Predator-Prey environments are common in past research. Especially associated with the "pursuit-evasion" descriptor are unbounded environments, in which the predator and prey can move indefinitely in any direction. Generally in these scenarios the predator is faster than the prey, so that fleeing in a straight line is not a dominant prey strategy. Alternatively, bounded environments in which the predator and prey are restricted to a finite area are common. Here, the predator need not

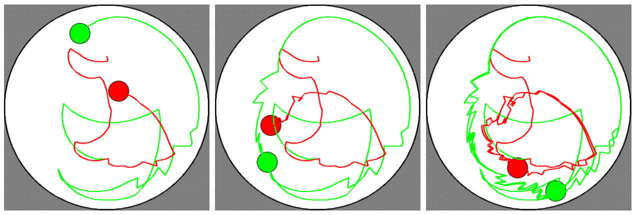
Figure 1: A game in the Predator-Prey domain demonstrating common strategies, with the predator in red and the prey in green. The prey escapes from the predator by moving along the outside edge of the world, while the predator tries to cut across the center to catch it. The prey zig-zags repeatedly in a behavior evolved to confuse the predator's predictive movements, causing the predator to move erratically and avoiding capture.

be faster than the prey, because prey agents are forced to circle back when they reach the edge of the environment. Even a predator who is significantly slower than the prey can still utilize a strong strategy to intercept a prey agent by using a shorter route.

The environment used in the following experiments takes the second format, using a circular area to ensure symmetry and lack of corners. Predator and prey agents move simultaneously at fixed speeds, and are controlled by selecting an angle to move for each timestep. The agents' bodies are circles with a given radius; if the predator and prey circles overlap each other, the scenario ends. The scenario also ends if the prey evades capture past a set time limit. If an agent attempts to make a move outside the edge, its destination is projected towards the center of the world to the nearest valid location. Agents start at opposite sides of the world, half-way to the edge. In particular, the predator/prey speed ratio of 0.06 to 0.10 was tuned in order to produce the best coevolutionary results. With our implementation, a predator speed of 0.05 typically resulted in the eventual emergence of extremely dominant prey, and a predator speed of 0.07 typically prevented the prey from gaining a foothold at all. We use a world radius of 1.0, an agent radius of 0.1, and a time limit of 200 steps. Fitness is zero-sum, given by the number of time steps survived for the prey, or the number of time steps remaining after the prey is captured for the predator (zero if the predator fails entirely). Figure 1 shows the results of an example Predator-Prey game.

2.2.2 Nim Rosin and Belew [24] provide a simple genetic algorithm implementation for playing the game of Nim, in which players are given one or more piles of stones, and must take turns removing one, two, or three stones from a pile until none remain, where the player who takes the last stone is the winner. Their work and most others using this implementation select pile sizes of [3, 4, 5, 4]. A strategy for playing this game is encoded as a list of binary values each associated with a specific game state, representing whether that state is desirable or not. In a given state, players will enumerate valid moves, and take the first move that leads to a desired state, or the first in the list if there are no such moves. For an initial state of [3, 4, 5, 4], the size of the genome is 599, for the 599 possible states that can be reached from that initial state (excluding the initial state itself). Usage of this game and implementation to compare competitive coevolutionary algorithms is widespread [13, 15, 21].

2.2.3 Internal Rosenbrock Panait and Luke [21] use a competitive "game" based on the Rosenbrock function, commonly used to evaluate evolutionary algorithms. The two players in the Internal Rosenbrock game are real-valued vectors attempting to minimize the 100-dimensional Rosenbrock function, and fitness is simply based on the difference between the output values for the two players, scaled to [-1, 1]. This game has two main advantages in evaluating the behavior of a competitive coevolutionary algorithm: it has a known "true" fitness for every individual to compare against, which is uncommon in competitive problems, and it is clearly a transitive game, which is a best-case scenario for many methods of evaluating competitive populations. In particular, Elo rating assumes that player skills are transitive, and though it is frequently used for games with some amount of intransitivity such as chess, it is expected to perform better on a transitive game. This problem was also used by Jaśkowski et al. [13] to evaluate their fitnessless coevolution. As in the original description, we use a 100-dimensional function with parameters in the domain [-5.12, 5.12].

3 Related Work

The Elo rating system is well-known as a result of its adoption by chess and subsequently many other game communities, so it is frequently adopted throughout the field of artificial intelligence to measure the performance of agents against each other. Samothrakis et al. [26] examine the behavior of two derivatives of the Elo rating system, BayesElo [5] and Glicko [7] and the ways these can be used to effectively evaluate competing AI agents, finding that Glicko ratings are particularly quick to reach accurate predictions with only a fraction of possible round-robin pairings available, though noting possible representation issues with intransitive strategies. The difficulties of intransitivity to competitive coevolution and rating systems is studied further by Richter [23], who introduces a variety of numerical measurements of intransitivity and its effects.

Less frequently, Elo rating has been applied directly to competitive coevolutionary algorithms as a fitness metric. Wijngaarden and Jong [29] compare several different evaluation methods including Elo and Glicko ratings as fitness functions for competitive coevolutionary algorithms under different algorithmic configurations. They do not find any consistent benefit from using these ratings directly as fitness measures, but discuss their relationships with intransitivity in competitive coevolution, and how intransitivity can harm coevolutionary growth, recommending the use of crowding-based phenotypic diversity preservation to prevent such intransitivity from causing coevolutionary failure. Cotton et al. [4] obtained strong results from using Elo rating as a fitness measure for a hybrid of competitive coevolutionary algorithms and gradient descent-based deep reinforcement learning, finding that the use of Elo rating as a surrogate fitness measure outperformed the direct use of training rewards. These works only use Elo rating as fitness, rather than using it to manipulate opponent sampling as in EBSSOS.

Liskowski and Jaśkowski [17] give an alternative surrogate fitness method for competitive coevolution with their Matrix Factorization-based Interaction Scheme. Opponents in this scheme are selected through K-random sampling, but the results of these evaluations are then used to estimate the relationships between other pairs of individuals through factorization of the interaction matrix. This method is commonly used in machine learning to fill gaps in sparse matrices for which only a small fraction of the elements are known. This method produced comparable results to a round-robin tournament in generating position evaluating agents for the game of Othello, with significantly less runtime. While this work only uses K-random sampling, its methods could plausibly be used as an alternative to Elo rating for SSOS.

While random sampling of opponents for competitive coevolutionary algorithms is the most common approach, more sophisticated methods of opponent sampling and evaluation have also been studied. Rosin and Belew [24] introduce the technique of shared sampling, which aims to provide a strong, diverse teaching set that overall is challenging to the whole population. This is done by assigning bounties to each individual which are shared between all the opponents who succeeded against that individual, thus favoring opponents that provided more unique counters to otherwise undefeated strategies. The authors find sharply better performance with this shared sampling method over simpler methods of opponent sampling. Panait and Luke [21] perform a variety of experiments comparing the use of single-elimination tournaments (previously introduced by Angeline and Pollack [1]) to random opponent sampling. They find that single-elimination tournaments perform advantageously under many circumstances, and are particularly effective at certain games such as one particular formulation of Nim, but are susceptible to noise in fitness evaluations. Tournaments also have the disadvantages of strongly relying on the assumption of transitivity in the tournament results, even more so than rating systems like Elo. Kim et al. [15] propose a more complex tournament structure called Entry Fee Exchange Tournament Competition. Here, individuals are assigned a uniform initial fitness, and then random pairs of individuals are selected to compete, where the winner steals a percentage of their opponent's fitness (the "entry fee"), and the loser is eliminated from the tournament, becoming no longer eligible to compete. This technique was found to produce similar-quality strategies to shared sampling on several games, but with better runtime and speed of evolution. Despite the success of these methods, they rely on the assumption that interactions between competitive agents result in a well-defined winner and loser, which is not true in general. As a result, there is a need for the development of opponent sampling techniques which are more applicable to these domains.

4 Methodology

The overall algorithm of EBSSOS consists of the following steps:

(1) For each population, construct a list of the current individuals from that population plus that population's hall of fame.
(2) Randomly pair each individual with an opponent[2], avoiding repeated pairings. Skip to Step 5 for initial evaluations, as no Elo ratings have been calculated yet.
(3) Randomly swap the opponents of two individuals, reverting this if the squared difference in Elo ratings within the two new opponent pairings is greater than the squared difference in Elo ratings within the two old opponent pairings.

[2]For single-population competitive coevolution, these pairings are against opponents within the same population. For more than two populations, these "pairings" are instead tuples of opponents.

(4) Repeat the previous step until no improvements have occurred for 200 (see Section 4.1) attempted swaps.
(5) Run evaluations for all pairings and record the resulting evaluation fitnesses.
(6) Calculate new Elo ratings for the population using Algorithm 1.
(7) Return to Step 2 and repeat until the desired number of evaluations per individual have been performed.
(8) Record the Elo rating for each individual as its final fitness.

Evaluations are performed in rounds, where every individual is assigned one opponent (or as few as possible such that every individual has at least one opponent). After a round of evaluations has been completed, the recorded evaluation fitnesses are normalized to $[0, 1]$ on the range of minimum and maximum values yet observed[3]. From these values, their Elo ratings can be calculated statically through the method of successive approximations (Algorithm 1). The static Elo rating is recalculated from scratch after each round of evaluations using this method.

4.1 Opponent Selection

In order to make the best use of evaluations, opponent pairs are chosen such that individuals are evaluated against opponents which are close in skill, maximizing the information gain from evaluation. The goal is to minimize the sum of squared distances in Elo ratings between paired individuals. As every individual needs an opponent, this becomes an instance of the Assignment Problem, the problem of selecting disjoint pairs in a weighted bipartite graph to minimize total cost. Existing exact algorithms for solving this problem, such as the Hungarian Method [16], were found to be too costly for large population sizes. An inexpensive hill-climber was used instead and exhibited comparable effectiveness in minimizing distance. A hill-climber for this problem starts by randomly pairing individuals with opponents who they have not yet been evaluated against while minimizing pairings per individual. Hill-climbing then repeatedly selects two individuals, exchanges their opponents, and keeps this new pairing if the new opponents do not increase the sum of squared Elo distances. Termination occurs after a set number of iterations pass without decreasing the sum of squared distances. Higher values for this iteration limit produce more accurate results, but we found that for our population sizes a value of 200 iterations produced the best trade-off of quality to speed.

This process is repeated over several rounds, evaluating agents against a new opponent each round who is close to them in Elo rating. Their ratings are adjusted based on the result of these evaluations, providing information not only on their performance against their opponent as in a standard competitive coevolutionary algorithm, but also information on their theoretical performance against other agents, due to the transitive relationships encoded in their opponent's Elo rating from past evaluations. We hypothesize that this will decrease the amount of evaluations necessary to achieve healthy coevolution.

[3]We do not recommend normalizing to theoretical maximum and minimum values, because if these values do not occur in earlier generations, it might result in one population's ratings being extremely high or low compared to their opponents, greatly limiting the feasibility of pairing individuals with opponents of similar ratings.

Number of parents (μ)	50
Number of children (λ)	150
Generations	50
Mutation rate	50%
GP: Tree initialization	Ramped half-and-half
GP: Initial tree size	3-7
Nim: Gene mutation chance	10%
Rosenbrock: Gene mutation chance	33%
Rosenbrock: Mutation amount	Gaussian, $\sigma = 0.1024$

Table 1: Evolutionary parameters

4.2 Hall of Fame

Cycling was found to be a problem in experiments in the Predator-Prey domain, as predators and prey would commonly over-fit to contemporary opponents, rather than providing agents that are strong in general. Fortunately, EBSSOS very easily allows for the addition of a hall of fame. The hall of fame is an evolutionary archive which stores strong individuals from previous generations, first described by Rosin and Belew [24]. The purpose of the hall of fame is to ensure that new individuals maintain the ability to counter old strategies no longer used by extant populations. In these experiments, a population's highest-rated individual is added to its hall of fame after each generation. This hall of fame is then appended to the current population during evaluation. Members of the hall of fame which are found to beat current strategies will have their ratings increase, ensuring that high-ranking individuals will need to defend their rating against them. In contrast, an otherwise unremarkable individual which is able to counter a strategy in the hall of fame that few others can will be rewarded.

5 Experiment Design

Agents are evolved using the parameters listed in Table 1. The remainder of this section discusses the design of genetic programming agents for the Predator-Prey domain, and the methodology we use to compare the performance of two competitive coevolutionary algorithms. For the Nim and Internal Rosenbrock domains, we use the agent design given by Rosin and Belew [24] and Panait and Luke [21] respectively.

5.1 Predator-Prey Agents

Predator and prey agents are structured as genetic programming parse trees returning an angle to move at each timestep, using strongly-typed genetic programming [18]. The primitive set for these trees uses angle types and distance types. Table 2 provides a list of genetic programming primitives used in this experiment. The set of genetic programming primitives used here were chosen to enable agents to respond to their absolute location in the world, and to their location relative to their opponent. Angle data types are eventually combined to produce the final angle (frequently using *average_angles*), with distance data types being secondary, eventually used for either scalar multiplication of an angle (using *multiply_angle*) or to enable conditional behavior when near the opponent (using *if_greater_than*). Of particular note is the presence of the *last_move* and *opponent_last_move* nodes, which allow agents to respond to their own behavior, or the opponent's behavior. Because *opponent_last_move* allows predators to move to predict the prey's future location, prey agents often evolve zigzag strategies which thwart prediction.

5.2 Comparison Methodology

To compare the performance of two configurations of a competitive coevolutionary algorithm, a statistical test is needed to determine if the solutions produced by a run of one configuration are, on average, "better" than those produced by the other. Given only one run of each configuration, we can determine (in a non-statistically-rigorous manner) which has the highest-quality solutions by running a round-robin tournament of the top k individuals produced by each population of each run, to calculate average fitness scores relative to their rivals from the other run. These two runs are then each scored by the fraction of their own individuals in the top 50% for each population. We use $k = 10$, as k should represent a broad set of individual strategies, but shouldn't be so high as to make a round-robin tournament prohibitive or so low as to include low-quality mutants from the final generation. Performing a statistical test needs a large number of runs as samples to compare, in order to infer the behavior of an average run. We perform 100 runs of each algorithm. Then, each run from the first configuration can be compared against each run in the second configuration using the above methodology, and their resulting scores summed per-run. The resulting value is that run's average share of top solutions relative to runs from the other configuration. A two-tailed F-test and t-test are then be used to measure whether one configuration produces a significantly higher average share of top solutions in comparisons against the other configuration.

6 Analysis of Elo Rating Convergence

A major drawback of the Elo rating system is that its ratings will be inaccurate until a sufficient number of games have been played for them to stabilize. In order for evolution based on these ratings as fitness values to perform well, the amount of noise that they contain should be limited. Therefore, it is necessary to determine how many rounds of evaluation are necessary to get useful ratings. To measure this, we increase the number of pairings per individual to 20, well above the number of evaluations for which the Elo rating is expected to converge. The Elo ratings of each individual's fitness scores are recorded after every round of evaluation, to show how they update over time. Since the baseline Elo rating was chosen as zero, these curves are divided by their mean values to produce comparable curves which can be averaged together. Additionally, for each point in time, the coefficient of variation (the ratio of the standard deviation to the mean) can be calculated for the set of Elo ratings occuring after that time to measure the stability of the rating after that many evaluations. These are made positive by taking the absolute value (in case of negative means), and are also averaged together. These averaged Elo rating and coefficient of variation curves can then be used to infer the overall behavior of Elo rating convergence across several evaluations.

6.1 Results

The convergence properties of 105,294 individuals in the predator/prey domain were analyzed and are displayed in Figure 2. The resulting ratings and coefficients of variation show that the Elo rating converges fairly quickly, with average variation of 2.16 after

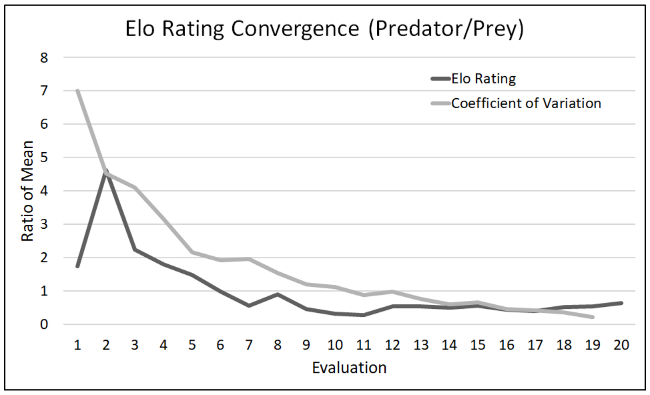

Figure 2: Normalized Elo rating at evaluation round n and coefficient of variation from evaluation round n onward, averaged across 105,294 individuals.

5 evaluations, 1.12 after 10 evaluations, and 0.65 after 15 evaluations. The average coefficient of variation for the entire 20-evaluation life of an individual is 7.01, so even after five evaluations, most of the instability in Elo rating has stopped. This is promising, as it indicates that stable Elo ratings can be achieved with very few evaluations as the rating algorithm quickly infers the relationships between individuals which have not directly interacted.

7 Analysis of Evolution Quality

In order to evaluate the overall performance of EBSSOS, a series of experiments are performed comparing EBSSOS with K rounds of evaluations, K-random opponent sampling, and K-random opponent sampling with Elo rating as fitness. The presence of this third intermediate method allows the effects of pairing similar-performing opponents to be separated from the effects of using Elo rating instead of raw fitness. Examination of $K = 5$, 10, and 20 allows characterization of whether too few rounds of evaluation produce a detrimental effect on the usability of the calculated Elo ratings. Performing the same experiment in the Predator-Prey, Nim, and Internal Rosenbrock domains allow the evaluation of EBSSOS and its characteristics in different environments, in order to ensure that any results in the one domain are not solely due to characteristics of the design of that game. The Internal Rosenbrock domain notably has an objective fitness measure, so this can be further used to analyze whether the surrogate values generated through the Elo rating system are reflective of true agent skill on this problem.

7.1 Results

As shown in Table 3, EBSSOS ("Paired Elo") was found to underperform random selection of opponent pairs with standard fitness ("No Elo"). While this result is disappointing, the remaining results demonstrate that the reasons for this are more complicated than a simple lack of effectiveness. Compared to runs in which Elo rating was used as fitness, but pairings were random ("Unpaired Elo"), "Paired Elo" produced a significant improvement, demonstrating that the choice to select opponents with similar strength is a sound one. Comparing the "Unpaired Elo" scenario to "No Elo" illustrates the true problem: using Elo rating as a surrogate fitness function noticeably harms evolution, presumably because Elo rating is less

Primitive Name	Output	Inputs	Description
distance_to_center/edge/opponent	Dist.	None	Distance to the center, edge, or opponent
angle_to_center/opponent	Angle	None	Angle to the center or opponent
last_move/opponent_last_move	Angle	None	Angle of the agent's or opponent's last move
distance_literal	Dist.	None	A randomly-initialized distance between 0 and 2
angle_literal	Angle	None	A randomly-initialized angle
flip_angle	Angle	Angle	Adds 180 degrees to the angle
add/subtract_angles	Angle	Angle, Angle	Adds or subtracts two angles
average_angles	Angle	Angle, Angle	Circular average of two angles
multiply_angle	Angle	Angle, Dist.	Multiplies an angle by a distance
add/subtract/multiply/divide_distances	Dist.	Dist., Dist.	Performs arithmetic on two distances
if_greater_than	Angle	Dist., Dist., Angle, Angle	If the first distance is greater than the second, return the first angle, otherwise return the second angle

Table 2: Genetic programming primitives for both predators and prey

effective as a fitness measure than raw fitness values. When comparing "Paired Elo" against "No Elo", these two effects occur simultaneously, with the harm done by imperfect surrogate fitness canceling out the gains made from intelligent sampling of opponents. This distinction is encouraging: Elo rating is not inherently tied to the methodology of SSOS. Rather, when biasing opponent sampling based on strength, some unbiased fitness measure is needed to prevent individuals from being punished for facing stronger opponents. The substitution of another measure instead of Elo rating might serve as a more accurate surrogate to true fitness, and result in the positive effects of SSOS overpowering the remaining negatives of the surrogate fitness measure. The results for comparisons with 5, 10, and 20 opponents sampled per individual were almost identical, strongly supporting the conclusions of Section 6.1 that Elo rating converges quickly enough that only five opponents are needed to get accurate results.

Nim resulted in similar conclusions to the Predator-Prey domain: "Paired Elo" significantly outperformed "Unpaired Elo", indicating that SSOS was effective, but "No Elo" outperformed "Unpaired Elo", indicating that the use of Elo rating was harmful. This harm again was found to cancel out the benefits of pairing similar opponents, and no significant overall benefit was found from using EBSSOS. The corroboration of these results with those from the Predator-Prey domain suggests that the effects observed are not unique to that domain, but are a more general property of the algorithm. The differences in performance of resulting strategies between the different configurations for Nim were much smaller than those for Predator-Prey. The reasons for this are unclear, but it could be that the decreased complexity of Nim resulted in smaller differences in high-performing individuals between different configurations.

The Internal Rosenbrock domain produced very different results to the other two problem domains. Here, any Elo-based fitness, whether used for opponent sampling or not, produced a significant improvement. This is in sharp contrast to the deleterious effects of Elo rating as a surrogate fitness function observed in the other two games. It is possible that this is related to the fact that Internal Rosenbrock is a completely transitive game (as the resulting score is just the difference in two absolute measures of the competing individuals). Mathematically, the Elo rating system assumes that player skill is transitive, despite this not even being true of chess, which

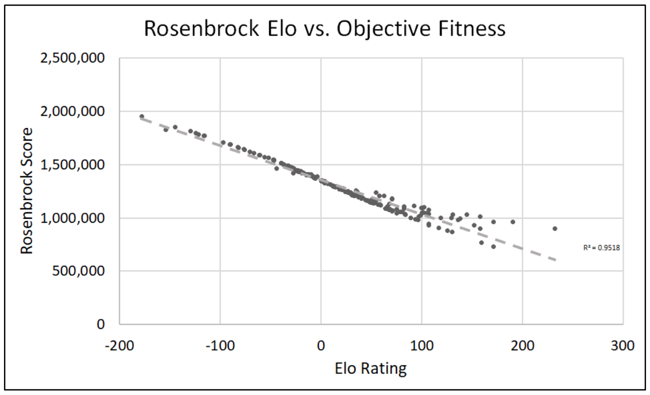

Figure 3: Comparison of fitness scores on the Rosenbrock problem with competitive Elo rating for 200 agents given 20 opponents each under EBSSOS. The results can be very cleanly fit with a line.

it was designed for. So while Elo rating may still be effective in intransitive games, it might inherently perform much better in the truly transitive Internal Rosenbrock game, allowing for a stronger estimate of agent skill level than simply averaging fitnesses between a handful of games. Figure 3 provides a comparison between the Elo ratings assigned to individuals in the Internal Rosenbrock domain and their actual objective fitness values. 200 randomly generated solutions were used, and had their Elo ratings evaluated over 20 games paired using EBSSOS. On the Internal Rosenbrock domain, then, it is clear that Elo rating is very closely correlated with actual agent performance. However, this is likely more true for this domain than it would be for the Predator-Prey domain or Nim, due to the perfect transitivity of Internal Rosenbrock.

8 Conclusions

Our experiments as a whole found that Elo-Based Similar-Strength Opponent Sampling, in its current state, does not provide a significant benefit to competitive coevolution. However, analysis of the individual contributing effects to this algorithm's performance demonstrates that the principle of Similar-Strength Opponent Sampling is promising, but held back by the use of Elo rating as a surrogate fitness. While Elo rating, when calculated through the

		Predator-Prey	Nim	Internal Rosenbrock
Paired Elo vs. Unpaired Elo	Means	78.1% vs. 21.9%	53.7% vs. 46.3%	51.5% vs. 48.5%
5 Opponents per Individual	Best	**Paired Elo**	**Paired Elo**	(Inconclusive)
Paired Elo vs. No Elo	Means	20.4% vs. 79.6%	49.4% vs. 50.6%	72.0% vs. 28.0%
5 Opponents per Individual	Best	**No Elo**	(Inconclusive)	**Paired Elo**
Unpaired Elo vs. No Elo	Means	45.2% vs. 54.8%	47.1% vs. 52.9%	72.1% vs. 27.9%
5 Opponents per Individual	Best	**No Elo**	**No Elo**	**Unpaired Elo**
Paired Elo vs. Unpaired Elo	Means	64.9% vs. 35.1%	52.8% vs. 47.2%	56.1% vs. 43.9%
10 Opponents per Individual	Best	**Paired Elo**	**Paired Elo**	**Paired Elo**
Paired Elo vs. No Elo	Means	45.1% vs. 54.9%	48.4% vs. 51.6%	78.3% vs. 21.7%
10 Opponents per Individual	Best	**No Elo**	(Inconclusive)	**Paired Elo**
Unpaired Elo vs. No Elo	Means	36.2% vs. 63.8%	46.2% vs. 53.8%	72.2% vs. 27.8%
10 Opponents per Individual	Best	**No Elo**	**No Elo**	**Unpaired Elo**
Paired Elo vs. Unpaired Elo	Means	64.8% vs. 35.2%	53.5% vs. 46.5%	49.0% vs. 51.0%
20 Opponents per Individual	Best	**Paired Elo**	**Paired Elo**	(Inconclusive)
Paired Elo vs. No Elo	Means	42.0% vs. 58.0%	44.5% vs. 55.5%	78.9% vs. 21.1%
20 Opponents per Individual	Best	**No Elo**	**No Elo**	**Paired Elo**
Unpaired Elo vs. No Elo	Means	34.6% vs. 65.4%	42.4% vs. 57.6%	78.8% vs. 21.2%
20 Opponents per Individual	Best	**No Elo**	**No Elo**	**Unpaired Elo**

Table 3: Experimental results for the experiments described in Section 7, displaying the average fraction of dominant strategies in pairwise comparisons. Statistically significant results are shown in bold ($p < 0.05$).

method of successive approximations, converges to stable values relatively quickly, it appears that the assumed transitive structure of the Elo rating system is not sufficiently representative of the actual relationships between relative agent performances, as only on the intransitive Internal Rosenbrock game was the use of Elo rating not harmful. Importantly, SSOS isn't reliant upon Elo rating specifically, it only needs a method of scoring individuals based upon the outcome of games against a non-representative sample of the opponent population. As a result, other representations besides Elo rating might result in a stronger positive effect by reducing the negative side effects associated with the use of Elo rating. On the Internal Rosenbrock domain, which matches the assumptions inherent in Elo rating better than the other two games, the use of Elo rating alone provided a strong improvement to coevolutionary performance, so it's possible that using a rating system or other method which more effectively models the relationships between different strategies in complex games might result in a direct improvement in coevolution, rather than harming it.

9 Future Work

While Elo rating is a well-known and easily implemented method of estimating individual skill, it has a number of shortcomings impacting its application in competitive coevolutionary algorithms. Primarily, Elo ratings take time to converge, and they don't model intransitive structures in the strategy space. A number of different rating methodologies have been proposed since the introduction of Elo, many of which are intended to outperform Elo in addressing these issues. Further investigation, then, should focus on these alternative rating methodologies as potential ways to maintain the benefits of SSOS while reducing the negative effects of using Elo rating as a surrogate fitness. Our experiments found that Elo ratings tended to converge to stable values within five to ten rounds of evaluations, which is not prohibitively expensive to compute, but is more than would be ideal. Additionally, until that point, the interim Elo ratings can vary wildly and be extremely inaccurate, which limits the effectiveness of SSOS, since Elo is not accurately reporting which agents have similar strengths. BayesElo [5] is an extension to Elo ratings which uses Bayesian statistics to limit the extreme variation in ratings for agents who have few evaluations to work off of. Similarly, the Glicko [7] rating system functions similarly to Elo ratings, but includes a secondary "ratings deviation" rating representing the expected accuracy of their rating, and an additional rating volatility measure added in Glicko-2 [8]. Samothrakis et al. [26] suggest that these algorithms, Glicko in particular, have desirable properties of quick convergence to accurate predictions.

Another issue with not only Elo, but most rating systems, is that they treat skill levels as transitive. If agent A usually beats agent B, and agent B usually beats agent C, then A will be expected to beat C. This is not always the case, however, as strategies often dominate each other cyclically, in a "rock-paper-scissors" relationship. Fortunately, some works have proposed modifications to Elo and similar rating systems that would allow them to represent intransitive relationships. Balduzzi et al. [2] introduce a rating system called Multidimensional Elo, in which the Elo rating is augmented by a multidimensional vector updated with an approximation of cyclic properties of the agent's interactions. A much more complex methodology, α-Rank [19], developed by Omidshafiei et al., gives a novel ranking system based on Markov chains and game-theoretic evolutionary dynamics which is designed specifically for the ranking of AI agents, applicable to highly intransitive and asymmetric games. While more complex than other methods, the authors suggest that this ranking system is computationally simple enough that it can be used directly in the training of agents, which makes it a good candidate for this application.

References

[1] Peter J. Angeline and Jordan B. Pollack. 1993. Competitive Environments Evolve Better Solutions for Complex Tasks. In *Proceedings of the 5th International Conference on Genetic Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 264–270.

[2] David Balduzzi, Karl Tuyls, Julien Perolat, and Thore Graepel. 2018. Re-Evaluating Evaluation *(NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 3272–3283.

[3] Dave Cliff and Geoffrey P. Miller. 1995. Co-evolution of pursuit and evasion II: Simulation Methods and results.

[4] D. Cotton, J. Traish, and Z. Chaczko. 2020. Coevolutionary Deep Reinforcement Learning. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2600–2607. https://doi.org/10.1109/SSCI47803.2020.9308290

[5] Rémi Coulom. 2007. Computing "elo ratings" of move patterns in the game of go. *ICGA journal* 30, 4 (2007), 198–208.

[6] Arpad E. Elo. 1978. *The rating of chessplayers, past and present.* Arco Publishing.

[7] Mark E. Glickman. 1999. Parameter Estimation in Large Dynamic Paired Comparison Experiments. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 48, 3 (1999), 377–394. https://doi.org/10.1111/1467-9876.00159 arXiv:https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/1467-9876.00159

[8] Mark E. Glickman. 2001. Dynamic paired comparison models with stochastic variances. *Journal of Applied Statistics* 28, 6 (2001), 673–689.

[9] Sean N. Harris, Eric Michalak, Kevin Schoonover, Adam Gausmann, Hannah Reinbolt, Joshua Herman, Daniel Tauritz, Chris Rawlings, and Aaron Scott Pope. 2018. Evolution of Network Enumeration Strategies in Emulated Computer Networks *(GECCO '18)*. Association for Computing Machinery, New York, NY, USA, 1640–1647. https://doi.org/10.1145/3205651.3208270

[10] Sean N. Harris and Daniel R. Tauritz. 2021. Elo-based Opponent Sampling for Multiobjective Competitive Coevolution. In *Proceedings of the 23rd Annual Conference on Genetic and Evolutionary Computation (GECCO '21)*. https://doi.org/10.1145/3449726.3459559

[11] Philip Hingston and Mike Preuss. 2011. Red teaming with coevolution. In *2011 IEEE Congress of Evolutionary Computation (CEC)*. IEEE, 1155–1163.

[12] Rufus Isaacs. 1965. *Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization.* John Wiley & Sons, Inc.

[13] Wojciech Jaśkowski, Krzysztof Krawiec, and Bartosz Wieloch. 2008. Fitnessless coevolution. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation (GECCO '08)*. Association for Computing Machinery, New York, NY, USA, 355–362. https://doi.org/10.1145/1389095.1389161

[14] Jonathan Kelly, Michael DeLaus, Erik Hemberg, and Una-May O'Reilly. 2019. Adversarially adapting deceptive views and reconnaissance scans on a software defined network. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 49–54.

[15] Yeo Keun Kim, Jae Yun Kim, and Yeongho Kim. 2004. A Tournament-Based Competitive Coevolutionary Algorithm. *Applied Intelligence* 20, 3 (May 2004), 267–281. https://doi.org/10.1023/B:APIN.0000021418.72362.fb

[16] H. W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2, 1-2 (1955), 83–97. https://doi.org/10.1002/nav.3800020109

[17] Paweł Liskowski and Wojciech Jaśkowski. 2017. Accelerating Coevolution with Adaptive Matrix Factorization *(GECCO '17)*. Association for Computing Machinery, New York, NY, USA, 457–464. https://doi.org/10.1145/3071178.3071320

[18] David J. Montana. 1995. Strongly Typed Genetic Programming. *Evol. Comput.* 3, 2 (June 1995), 199–230. https://doi.org/10.1162/evco.1995.3.2.199

[19] Shayegan Omidshafiei, Christos Papadimitriou, Georgios Piliouras, Karl Tuyls, Mark Rowland, Jean-Baptiste Lespiau, Wojciech M Czarnecki, Marc Lanctot, Julien Perolat, and Remi Munos. 2019. α-rank: Multi-agent evaluation by evolution. *Scientific reports* 9, 1 (2019), 1–29.

[20] Una-May O'Reilly, Jamal Toutouh, Marcos Pertierra, Daniel Prado Sanchez, Dennis Garcia, Anthony Erb Luogo, Jonathan Kelly, and Erik Hemberg. 2020. Adversarial genetic programming for cyber security: A rising application domain where GP matters. *Genetic Programming and Evolvable Machines* 21, 1 (2020), 219–250.

[21] Liviu Panait and Sean Luke. 2002. A comparison of two competitive fitness functions. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation (GECCO'02)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 503–511.

[22] Craig W. Reynolds. 1994. Competition, coevolution and the game of tag. In *Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*. 59–69.

[23] Hendrik Richter. 2015. Coevolutionary intransitivity in games: A landscape analysis. *CoRR* abs/1501.04010 (2015). arXiv:1501.04010 http://arxiv.org/abs/1501.04010

[24] Christopher D. Rosin and Richard K. Belew. 1997. New methods for competitive coevolution. *Evolutionary Computation* 5, 1 (March 1997), 1–29. https://doi.org/10.1162/evco.1997.5.1.1

[25] George Rush, Daniel R Tauritz, and Alexander D Kent. 2015. Coevolutionary agent-based network defense lightweight event system (CANDLES). In *Proceedings of the 17th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '15)*. 859–866.

[26] S. Samothrakis, D. Perez, S. M. Lucas, and P. Rohlfshagen. 2016. Predicting Dominance Rankings for Score-Based Games. *IEEE Transactions on Computational Intelligence and AI in Games* 8, 1 (March 2016), 1–12. https://doi.org/10.1109/TCIAIG.2014.2346242

[27] Kevin Schoonover, Eric Michalak, Sean Harris, Adam Gausmann, Hannah Reinbolt, Daniel R Tauritz, Chris Rawlings, and Aaron Scott Pope. 2018. Galaxy: a network emulation framework for cybersecurity. In *11th USENIX Workshop on Cyber Security Experimentation and Test (CSET 18)*.

[28] Sevil Sen, Emre Aydogan, and Ahmet I Aysan. 2018. Coevolution of mobile malware and anti-malware. *IEEE Transactions on Information Forensics and Security* 13, 10 (2018), 2563–2574.

[29] Rients P. Wijngaarden and Edwin D. Jong. 2008. Evaluation and Diversity in Co-Evolution. In *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature — PPSN X - Volume 5199*. Springer-Verlag, Berlin, Heidelberg, 631–640.