The N-Strikes-Out Algorithm: A Steady-State Algorithm for Coevolution

Thomas Miconi and Alastair Channon
University of Birmingham
Edgbaston B152TT
Birmingham, UK
txm@cs.bham.ac.uk

***Abstract*— We introduce the N-strikes-out algorithm, a simple steady-state genetic algorithm for competitive coevolution. The algorithm can be summarised as follows: Run competitions between randomly chosen individuals, keep track of the number of defeats for each individual, and remove any individual which has been defeated N times. Naive application of the algorithm in 2-population problems leads to severe disengagement. We find that disengagement can be eliminated (for all tasks involving real-valued continuous scores) by determining 'victories' and 'defeats' between fellow members of the same species, using competitions against a single member of the opposing species as a point of comparison. We apply our algorithm to the "box-grabbing" problem for artificial 3D creatures introduced by Sims. We compare our algorithm with Sims' original Last Elite Opponent algorithm, and describe (and explain) different results obtained with two different implementations differing mainly by the harshness of their selection regimes.**

Coevolution is a generic term describing any situation in which two or more lineages evolve in such a way that the fitness landscape of each lineage is determined (at least in part) by the evolving features of other lineages. The defining concept of coevolution is that of *coupled fitness landscape*. A consequence of such a situation is that evolutionary changes in one lineage may alter the fitness landscape of other coevolving lineages, and therefore prompt evolutionary changes in these lineages as well, which may in turn alter the fitness landscape of other lineages, etc.

It has been argued that such a process of constant mutual adaptation may lead to some form of global progress over time, through incremental accumulation of adaptive features. This idea forms the core of the concept of "evolutionary arms race" introduced by Dawkins and Krebs [1]. Rosin and Belew [2] summarise the transposition of the arms race concept to artificial evolution:

> Since the parasites are also evolving with a fitness based on a competition's outcome, the success of a host implies failure for its parasites. When the parasites evolve to overcome this failure, they create new challenges for the hosts; the continuation of this may lead to an evolutionary "arms race" (...) New parasite types should serve as a drive toward further innovation, creating ever-greater levels of complexity and performance by forcing hosts to respond to a wider range of more challenging parasite test cases.

Unfortunately, despite early successes such as those reported by Hillis [3], it soon emerged that coevolution was prone to complex dynamics which often led to unexpected results. As Ficici [4] pointed out, it did not help that many early works in coevolution relied on somewhat vague assumptions regarding evolutionary progress, often linking it with an increase in complexity[1]. While it was generally expected that coevolving entities should become "better", what exactly "better" meant was usually not made explicit precisely.

Besides this problem of ambiguous expectations, it was quickly recognised that coevolution often follows dynamics which contradict any acceptable notion of long-term progress. An early identified problem is that of *intransitivity* in the global fitness landscape (even if A defeats B and B defeats C, it is still possible that C may defeat A), which may lead to coevolutionary "cycles" or "circularities" [8], [9]. This feature also underlies other problems such as *opportunism*, in which seemingly promising, sophisticated solutions are displaced by trivial solutions which exploit one specific weakness; a symmetric problem is that of over-specialisation, in which a given solution is being overly refined with regard to its current competitive environment, only to be displaced by a trivial solution which poses a completely different problem (Watson and Pollack [10] call such phenomena "focusing on the wrong thing"). As we will see, opportunism and over-specialisation may be a major hindrance on the road to progress if the algorithm is vulnerable to it.

Another difficulty which may arise is that of disengagement, or loss of gradient [10]. This occurs mostly in n-species situations, when one competing species "out-evolves" the others, in such a way that members of this successful species can defeat all of their opponents. The result is that the opposing species lose any meaningful gradient to guide their evolution, which becomes effectively random.

I. Coevolutionary algorithms

The overwhelming majority of algorithms in use for competitive coevolution are generational. Large parts of the population are evaluated and replaced simultaneously. A common algorithm is precisely the one introduced by

[1]The widespread idea that evolution or coevolution creates a *driving force* towards increasing complexity (beyond the mere stochastic increase inherent to any random branching process with a hard lower bound) has been dismissed in no uncertain terms by most prominent specialists; let us mention Dawkins [5], Maynard-Smith and Szathmary [6] and of course Gould [7].

Sims [11], also called the "Last Elite Opponent" model: at every generation, each individual from population A is pitted against the current champion of population B. The resulting score is used as a fitness value for selection and reproduction, as well as choosing a new champion for population A (the individual from A which obtained the best score against the current champion of B). Then the same process is applied to population B, using the new champion of population A for evaluation, and the cycle starts over again.

A problem with this approach is that, since the referent (that is, the individual used for evaluation) changes at every generation, so does the direction of selection pressure. This may result in an unstable selective process, in which the direction of the selective pressure changes so fast that no overall progress can occur: even though selection mechanically improves the population from one generation to the next (with regard to the current selective environment defined by the current population), this may fail to translate into a longer-term historical progress as implied by the arms race concept.

To enhance the stability of the process, an *archive* may be used. An archive is a collection of previously encountered individuals which are kept for evaluation purposes so as to provide a more stable selective pressure. A simple option is to keep the champions of the n previous generations (which is the "basic" method used by Nolfi and Floreano [8]); however this only reduces instabilities over a scope of n generations backwards, so n should be high; furthermore, if evolution stabilises, then champions from contiguous generations will tend to be similar to each other, which may reduce the diversity of the selective pressure provided by n successive champions.

Another possibility is to maintain a "Hall of Fame" containing the champions of all previous generations, then pick n opponents at random from this population at every evaluation [2]. The consequences of such a method are interesting. Because newer individuals have to prove their worth against champions of all previous generation, the selective process will mechanically favour individuals able to defeat many (ideally all) of their predecessors, thereby encouraging a proper arms race. However, as Nolfi and Floreano point out, this also means that an ever larger portion of the selective process becomes fixed, diminishing the importance (and expected benefits) of coevolutionary dynamics: " As the process goes on, there is less and less pressure to discover strategies that are effective against the opponent of the current generation and greater and greater pressure to develop solutions (...) against opponents of previous generations" [8]. Because of this 'ossification' of the selective process, selection becomes tailored to the particular history of that run, which may hamper generality. Nolfi and Floreano report that in their more complex settings, individuals evolved through 'naked' coevolution were better at defeating individuals evolved through "Hall of Fame" coevolution than the reverse. This happened even though individuals evolved with a Hall of Fame were demonstrably better at defeating their own ancestors, indicating a more successful arms race. While progress had been more straightforward, it had also been more limited in scope[2].

In Nature, the process of replacement is quite different from these approaches. Individuals (or lineages) are usually not massively decimated and replaced at discrete instants in time. Individuals and lineages are constantly exposed to a varying selective pressure. Evaluation does not occur against a few selected champions, or against an (impossible) archive, but against the current population itself. The process of selection, elimination and replacement is gradual and asynchronous, and therefore, so is the change in the selective landscape. Furthermore, evaluation occurs continuously, often through random encounters against random opponents, until an individual (or a lineage) cannot keep up with the selective environment and is eliminated. The population maintains its own archive, but ossification is prevented through continuous removal of elements which appear unfit.

While this process has no global direction, and is apparently not oriented towards any particular overarching goal (complexity, intelligence, etc.), it nevertheless produces remarkable examples of smaller-scale mutual optimisation patterns (the "arms races" identified by Dawkins [1] as a prominent source of adaptive complexity in Nature), the local, interlaced "eddies" of the evolutionary maelstrom. Capturing these dynamics is precisely the justification of using coevolution for optimisation. How can we translate the asynchronous, steady-state pattern of natural coevolution into practical algorithms capable of producing useful results for human operators?

II. (Lack of) steady-state coevolutionary algorithms

Despite the popularity of steady-state models for traditional genetic algorithms, it is not easy to find examples of steady-state coevolutionary algorithms. One such method is Reynold's Steady-State Genetic Programming (SSGP) algorithm for the game of Tag [12]: in this system, evaluation and replacement do occur in a one-at-a-time fashion. However, evaluation still occurs in a synchronous way (the fitness of an individual is the result of its interaction with 6 other individuals randomly chosen within the population).

In Paredis' Life-Time Fitness Evaluation (LTFE) system [13], the fitness of each individual is evaluated after the result of its last 20 encounters. 20 interactions between fitness-selected individuals are performed, their fitnesses are updated, a new individual is created by recombining two

[2]These results, as well as those obtained by Ficici, hint at a necessary distinction between *local* progress (newer individuals are better than their immediate predecessors against their local, current opponents), *historical* progress (newer individuals are better than their predecessors against their ancestral opponents) and *global* progress (newer individuals are better than their predecessors against the entire search space). The first one is all that natural selection is concerned with. The second one is a definition of an arms race, and can be brought about by using an archive. The latter is the real objective of coevolutionary optimisation; unfortunately it is not necessarily brought about by the former two.

selected parents, the initial fitness of the new individual is evaluated by pitting it against 20 opponents, the new individual is inserted in the population (it is not specified whether another individual should be deleted) and the cycle starts again. Thus evaluation occurs in a one-time manner for many individuals, while superior candidates have their fitness refined through subsequent competitions. This concentrates computational power on promising solutions, although at the expense of exploration.

III. The N-Strikes Out Algorithm

We propose a simple algorithm to capture the desired properties described above. In the context of one-population, symmetric coevolution, the proposed algorithm can be described as follows:

1: Pick two individuals A and B from the population at random.
2: Pit them against each other; determine the winner and the loser of the confrontation (if any).
3: If the loser has been defeated N times over its entire history, remove it from the population and replace it with a new individual.
4: Start again.

From this description we can make several observations about the algorithm. First, it is truly a steady-state algorithm. Population change occurs in a one-at-a-time fashion, while evaluation occurs continuously. Poorly adapted candidates are rejected early, while promising solutions are constantly re-evaluated against the changing environment, until this environment proves too much for them. In effect, in the N-strikes out algorithm, the population acts as a self-maintaining archive, constantly updating and reevaluating itself.

In addition, the dynamics of the algorithm in terms of evaluations and reproductions are globally predictable. After an initial transient phase, a steady regime will settle in which, on average, *one new individual will be created for every N evaluations*. Thus N provides a convenient control of the balance between exploration and exploitation.

The algorithm shares another common feature of steady-state genetic algorithms: simplicity (both in concept and in implementation). It removes the necessity for managing generational changes among populations. The only overhead is to keep track of the number of defeats for each individual, which is usually easy.

The algorithm is flexible in that various adaptations are possible, especially with regard to the choice of parents for the new individual. One possible method is to replace the loser with a recombination of itself and the winner, or with a mutated version of itself or of the winner. This method has the advantage of being easily implemented, and is also quite conservative: it makes it likely that some of the genetic material of the loser will be preserved in the new offspring, which may be desirable in some situations but not in others. A more neutral method consists in replacing the loser with a recombination of parents chosen after some particular statistic, such as their number of victories, the ratio between victories and defeats, etc.

A consequence of the algorithm as described above is that defeats are never forgotten. This implies that even individuals with a high victories-to-defeats ratio, indicating high performance, can be eliminated when their number of defeats reaches N (which is bound to happen at some point). If one wants to counter this, a possible modification consists in specifying that the loser will only be replaced if it has suffered N defeats over its last K competitions: defeats which happened more than K contests ago are 'forgiven'. Introducing this finite memory window has several consequences, both positive and negative: it has the advantage of ensuring that only currently unadapted individuals will be removed. However it also means that there is no longer an immediate relation between the number of contests and the number of creations of new individuals (although the impact will be small if most eliminations occur before K contests, as is likely to be the case for most new individuals).

IV. The Two-Populations Case: Eliminating Loss-of-Gradient

A. The naive approach to two-populations problems

The description given in the previous section applies to one-population, symmetric problems. However, when the problem is based on competition between two or more species, the algorithm must be modified (if only because competitors cannot be recombined together, or replaced by each other, since they belong to different species). A naive transposition could be expressed as follows: take one individual from each population, pit them against each other, and if the loser of this confrontation has reach his Nth defeat, replace it with a new individual created through some suitable reproductive strategy.

Unfortunately, early experiments (based on the experimental settings described in the following section) showed that this naive method consistently induced severe cases of *disengagement*, or *loss of gradient*. At some point, one of the populations (say, population A) would find a somewhat effective method to solve the problem, while individuals from population B would still be in an early, poorly adapted stage. This adaptation would propagate throughout population A in such a way that eventually, every individual from A would be able to defeat every individual from B. At this point, every competition would lead to a defeat for the individual from B, so all individuals from B would receive the same selective information (uninterrupted losses), leading to a uniform selective pressure. No gradient would allow the algorithm to favour one individual from B over the other. This would effectively remove any selective direction and lead to a non-selective evolution based solely on random drift.

Loss of gradient, also called disengagement, is a well-known problem of coevolutionary dynamics. Several remedies have been proposed, often based on a distinction between the two populations ("hosts", which are to be op-

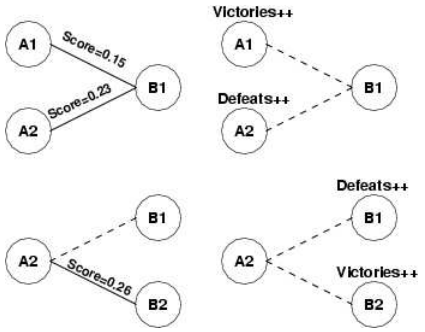

Fig. 1. 'Ricocheting' oblique evaluation: two individuals a_1 and a_2 are pitted against a common individual b_1. Their performances are compared, which provides a winner and a loser: victories and defeats counters are updated accordingly. Then a new individual b_2 is compared with b_1 based on its performance against a_2. Because the performance of b_1 against a_2 is already known, this only requires one additional competition.

timised, and "parasites" which are an instrument of this optimisation). Cartlidge [14] suggest to punish parasites which prove too effective. Ficici and Pollack [15] (followed in this by De Jong [16]) propose the more radical approach of selecting parasites directly after their capacity to discriminate between hosts. In addition to this, Bongard [17] choose to save tests which are currently too difficult for later reuse.

We noticed, however, that without using any of these various techniques, Sims' original algorithm was *not* subject to loss of gradient when applied to similar experiments. We found that one cause of this immunity was that in Sims, selective gradient was obtained through comparison of real-valued, continuous scores, instead of binary victory/defeat indicators. To use the real-valued scores directly in our algorithm would be difficult, however: we would need to compare scores obtained against different individuals, which might be unfair; also finding an equivalent to the concept of 'N-strikes out' with continuous scores might be non-trivial.

B. Oblique evaluation eliminates disengagement

There is another way. The fundamental reason why the LEO generational algorithm appears immune to disengagement and loss of gradient lies not only in the use of real-valued scores, but also in the way individuals are 'evaluated': the reproductive success of an individual is not determined by its isolated performance against a member of a competing species; rather, it is determined by comparing its performance with the performance of fellow members of the *same* species, against a common opponent. This, together with real-valued scores, effectively eliminates loss of gradient, even in the case when one species 'out-evolves' the other: even if all individuals from a species are thoroughly beaten by their competitors, the difference in the severity of their defeats will provide a usable gradient.

Following this approach, instead of directly evaluating individuals through their interactions with opponents from the other population, we decide instead to firmly limit the selective process to a comparison between fellow members of the $same$ species. This is done by comparing the results of two individuals against one single member of the opposing population. This process of *oblique* evaluation effectively ensures that a usable gradient will always be present.

Thus "victories" and "defeats" are defined as follows: two individuals a_1 and a_2 from population A are pitted in turn against the same member b_1 of population B; whichever of a_1 and a_2 obtains the best score (if any) is the winner, and the other one is the loser. We then apply the same "N-strikes-out" algorithm as described in the previous section, with this new definition of victories and defeats.

One final refinement is necessary to complete the algorithm. At first sight, oblique evaluation seems to double the amount of computation required for evaluation: while only one interaction was needed in the naive method, now two interactions are necessary to obtain an evaluation. However this is merely illusory, because some of the information gained in one evaluation can be used in the next: after pitting both a_1 and a_2 against b_1, we now pick an individual b_2 at random from B, oppose it to a_2, and compare its score against the (already known) score obtained by b_1 against the same individual a_2. The process is then repeated with a_2, b_2 and yet another individual a_3, etc. Thus evaluation is constantly "ricocheting" from one population to the other, always using some of the information found in the previous round in the next. Each new evaluation only requires one new interaction. The process is illustrated in Fig. 1[3].

C. Disengagement, oblique evaluation, and selective inspiration from Nature

Disengagement / loss-of-gradient is usually seen as a troublesome impediment to "proper" coevolution. However, it might be argued that it is actually an *expected feature* of coevolution. In Nature, a situation in which one lineage is utterly out-evolved by others to the point of being unable to defeat any opponent, corresponds essentially to an extinction event. But the basis of Van Valen's original Red Queen hypothesis [18] is precisely that such events occur continuously *because* of coevolution (the Red Queen effect imposes a constantly changing environment, and species routinely fail to adapt and go extinct with a probability which is independent of the species' age). Thus disengagement is actually the direct expression of the Red Queen effect, within the more restricted environment of artificial coevolution. Therefore, to fight disengagement, we should not try to copy Nature as faithfully as possible; rather, it is preferable try and find which factors cause it and, if feasible, eliminate them.

In essence, the process of oblique evaluation implicitly present in the LEO algorithm (and explicitly present in ours) counters the Red Queen effect by separating *inter-specific* and *intra-specific* competition; more precisely, it *removes* inter-specific competition altogether, and replaces it

[3]If a_2 has reach the maximum number of defeats and must be replaced, then the result of the opposition between b_1 and the *new* a_2 is now unknown, and cannot be used for further comparisons. In this case we keep a_1 as the current referent for the next round.

with purely intra-specific competition. Individuals are not in direct competition for survival with members of the opposing species, as they are in Nature; instead, they compete for survival solely against members of their own population. In Nature, competition occurs between all individuals, both between and within species. The consequence, through the Red Queen effect, is that species routinely go extinct. In the context of artificial evolution, however, competition between species is an unwanted feature (one might say a 'bug') of evolution: we are interested in optimising individuals, but we do not want any species to go extinct (i.e. we do not want disengagement to occur). Disengagement occurs when the algorithm does not, or cannot, produce meaningful intra-specific competition.

Our conclusion is that the use of real-valued, continuous scores, even though it always provides a readily available gradient between individuals, is not sufficient to prevent disengagement in itself. This latent gradient must be properly exploited by emphasising intra-specific competition against inter-specific competition. Algorithms which do just this (such as Sims' algorithm, or our own corrected algorithm) effectively eliminate disengagement. This may be seen as a case of analysing natural evolution, and "carving it up" to pick precisely what can be useful to us, leaving aside unwanted features.

V. Experiments and Results

A. Experimental settings

1) The task: Our experiments are based on the simulation of artificial three-dimensional (3D) creatures controlled by neural networks, similar to those introduced by [11]. Creatures are composed of rigid blocks linked by hinge joints. Their neural networks are distributed throughout their limbs. Initially creatures also have four kinds of external sensors, measuring the x and y distances of either the opponent or the box, within the frame of reference of the limb in which the sensor exists; in our later experiments we remove the y-sensors. These external sensors come in addition to internal proprioceptors which measure the current angle at the joint between a limb and its parent limb. There is one such proprioceptor for each limb, except of course for the 'root' limb which has no parent. Each limb (except the root limb) also has an actuator neuron, which specifies the current desired angular speed around the joint between this limb and its parent. The software platform being used is essentially a reimplementation of Sims' model, the main difference being our use of usual McCulloch-Pitts neurons instead of Sims' more complex neurons, with an improved genetic encoding and developmental system. A complete description and justification of our platform can be found in another paper [19]; source code and video captures can be found at `www.cs.bham.ac.uk/~txm`.

The task being considered is the "box-grabbing" contest described in [11]. Two creatures compete to gain control of a cubic box. At the beginning of each contest, the box is placed at the centre of the environment. Both competing creatures are placed on opposite sides of the box, at a certain distance from it. As in Sims, creatures are pushed behind a diagonal plane slanted by 45 degrees so they cannot gain an undue advantage by their height. Both creatures are then left to act for a given period of time. At the end of the evaluation period, the score for each creature is determined in the following way: if one creature is at distance $d1$ from the box (as defined by the distance between the centre of the creatures' closest limb to the box, and the centre of the box) and the other at distance $d2$, then the former creature's score is $\frac{d2-d1}{d1+d2}$, while the latter's is $\frac{d1-d2}{d1+d2}$.

2) Methodology: To make our comparison as meaningful as possible, we follow Sims' choice of using two species, which we arbitrarily call A and B. We maintain this separation throughout our experiments and analysis. We then proceed to compare our own implementation of Sims' original LEO algorithm (described in the introduction of this paper) with a 2-strikes-out algorithm, using oblique "ricocheting" evaluation, as described above. For both algorithms, the population is divided into 2 species containing 100 individuals each.

We performed our comparisons by pitting champions obtained with one algorithm against champions (of the opposite population) obtained with the other algorithm, at discrete points in time, in order to track the relative performance of both algorithms over time. Thus those comparisons were 'equal effort' comparisons: we only opposed champions obtained after equal numbers of simulated contests (that is, champions obtained by either algorithms with equal computational costs). To do this, we evenly sampled 25 champions in each population of every run. This was done by picking the current champion of each population after 6000, 12000, 18000, etc. contests had been simulated (our runs were allowed to proceed for 150000 simulated contests)[4].

This created two sets of 25 champions (one set for population A and one set for population B) for each run of each algorithm. We then proceeded to oppose the champions obtained by each algorithms at a given time, against champions obtained by the other algorithm at the same time. That is, for each n in the $[1..25]$ range, we pitted the nth champion of each run generated by one algorithm, against the nth champions of all runs generated by the other algorithm. Note that we only pitted individuals from A-populations against individuals from B-populations.

For every n in the $[1..25]$ range, this amounts to $12 * 12 * 2 = 288$ competitions in total: the nth A-champions of 12 LEO runs against the nth B-champions of 12 NSO runs, plus the nth A-champions of 12 NSO runs against the nth B-champions of 12 LEO runs The repartition of victories between both algorithms, for each n, gives a 'snapshot' of the relative performance of both algorithms at a given point

[4]In the LEO algorithm the champion is automatically defined by the algorithm as the current member of one population which obtained the best score against the current champion of the opposing population. In the NSO algorithm, we simply picked as a current champion the individual within the population which had accumulated the highest number of victories throughout its history.

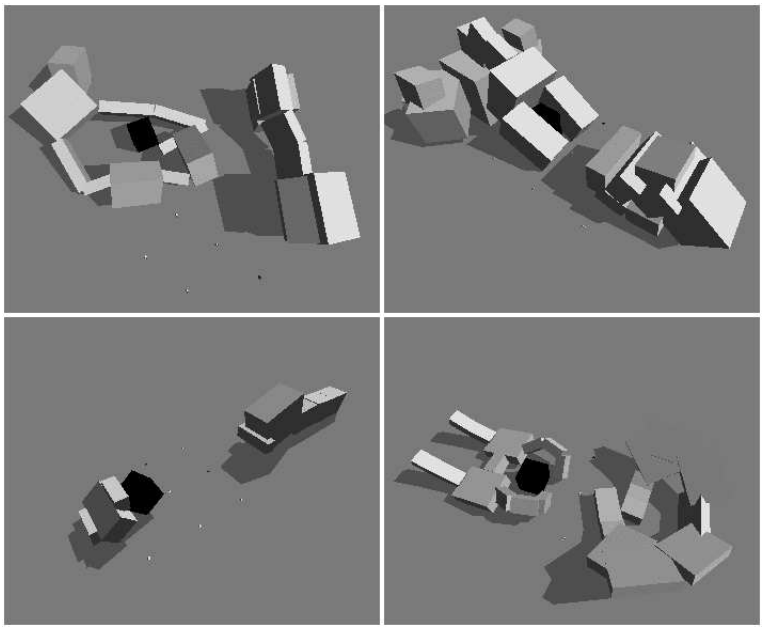

Fig. 2. Four creatures from four different runs, evolved with the 'harsher' versions of both algorithms (top row: LEO algorithm, bottom row: 2SO algorithm). In the top-left picture, one creature has managed to enclose the box between its symmetric arms. The protrusions on each arm will induce a 'locking' effect which will secure the creature's grip, allowing it to resist the attempts of its opponent. In the top-right picture a large creature has pinched the box and has begun to retreat, with the opposing 'caterpillar' creature following it. In the bottom-right picture a very simple, 3-limbed creature has used sinusoidal motion to get to the box and is using exactly the same motion to go backwards, dragging the box as a result. In the bottom-right corner one creature has managed to snatch the box between its arms and will slowly move away from its unsuccessful opponent.

in time (namely, after $n*6000$ contests have been simulated).

B. Results

Our initial experiments compared a plain, 'naked' 2-strikes-out algorithm (2SO) against an implementation of Sims' LEO algorithm. These early implementations put emphasis on maintaining existing genetic material, in an attempt at avoiding premature convergence detected in preliminary test experiments. Thus the replacement method for the 2SO consisted in replacing the eliminated individual with either a mutated copy of itself, or a recombination between itself and its latest defeater. Similarly our implementation of Sims' algorithm was genetically conservative, with a 50% survival rate and a similarly lenient replacement method in which the eliminated individual contributed genetic material to its replacer. More precisely, at each generation, the following cycle was iterated: three individuals were randomly chosen, and the individual with lowest fitness (i.e. having obtained the lowest score against the current champion of the opposing population) was replaced either with a mutated copy of itself, or with a recombination of itself and one of the two others. This was repeated until 50% of the population had been replaced. With these initial settings, the 2-strikes-out algorithm outperformed the LEO algorithm by a wide margin.

However, we decided to reimplement our version of Sims' algorithm to make it much closer to the original, which implied significantly harsher replacement policies: we lowered the survival rate to 20% and selected parents from the survivors only, with a roulette-wheel selection method based on the same normalised fitness function as used by Sims. We also brought a few changes to the platform, such as removing y-sensors (leaving only sensors for the distance in the direction of the x-axis of the limb) and changing various parameters. This led to a massive improvement in the performance of the LEO algorithm, which significantly outperformed various N-strikes-out algorithms.

This prompted us to introduce a less conservative replacement policy in our 2-strikes-out algorithm: new individuals would be generated from parents selected based on their number of victories only (eliminated individuals would no longer contribute to the genome of their replacer). On the other hand we introduced a time window beyond which ancient defeats, were 'forgiven'. We found that the resulting algorithm produced initially weaker creatures, but exhibited more reliable progress, and eventually managed to equal the LEO algorithm after enough cycles had elapsed. The long-term behaviour of these curves indicate similar performance of newer champions in the long run. Our interpretation is that under these settings, the LEO algorithm produces well-adapted designs faster than the 2SO algorithm.

Fig. 3 shows the results of our equal-effort comparisons between the LEO algorithm and the 2SO algorithm. Some of the creatures evolved in our early experiments were shown

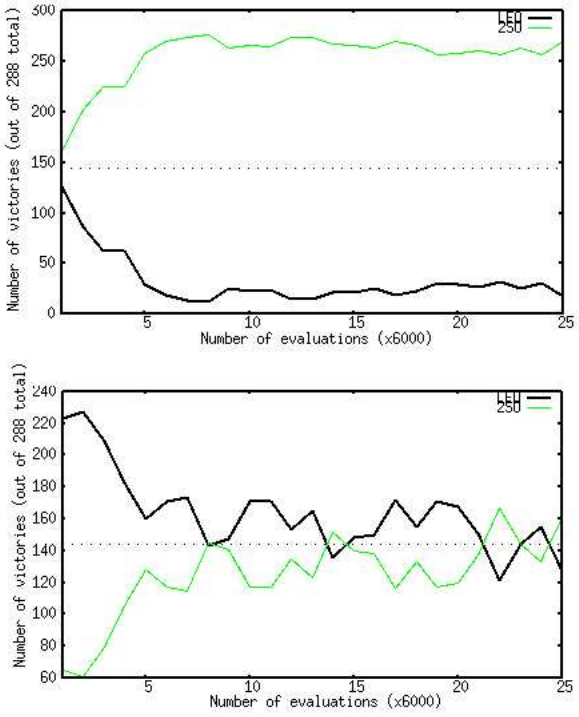

Fig. 3. "Equal effort" comparison between LEO and 2SO: repartition of victories in head-to-head competitions between current champions generated by both algorithm at discrete points in time. The top graph shows results for early, 'lax' implementations of both algorithms. The bottom graph shows results with increased selection pressure and less conservative reproduction. See text for details.

in another paper [19]. Fig. 2 shows four creatures evolved with the improved versions of our algorithms.

Through visual inspection, we determined that the superiority of the NSO method in our early experiments was in great part caused by the instability of the LEO algorithm. Typically, both populations would come to some interesting degree of adaptiveness and functionality, but then one of the populations would come up with simplistic individuals which would exploit a particular weakness of the opposing champion. Because these individuals would do rather well against the opposing champion (no matter how poorly they performed otherwise) they would be promoted to new champions of their own population, thereby upsetting the hierarchies of both populations, and forcing evolution to start again from scratch.

A common source of instability was "sensor hijacking": one creature would exhibit an effective locomotive behaviour, which would unfortunately be dependent on the position of its opponent because the input from a particular sensor would be able to dramatically influence the behaviour of the creature: the successful individual would be dependent on the presence, or absence, of the opponent within a certain distance (either through evolved dependence, or through fortuitous, spurious connections which would not have been eliminated by evolution). Therefore, all the opponent had to do to outperform this fragile behaviour was to foil this expectation - which often happened simply by staying motionless, or falling on one side. This is a specific type of opportunism, exploiting weaknesses in evolved neural controllers. This is what prompted us to simplify the set of sensors by removing y-sensors altogether.

Our reimplementation of the algorithm significantly reduced this problem. However the fundamental instability of 3D physics implies that in some circumstances, small differences (or even different random seeds) can have large consequences: for example, a creature might "only just" scrape the box away on one occasion, but fail on others, which may result in completely different outcomes for the contest. The improved LEO algorithm was still occasionally fragile to such situations, in which one creature would 'get lucky' during a particular contest by performing some action that would fail most of the time. This one-time success would allow the creature to become a champion, despite its lack of reproducibility.

While instability can be observed by visual inspection, it can also be identified in a more objective manner through simple analysis tools. We drew coarse-grained tournament matrices (CGTM) of several runs, both with Sims' algorithm and ours. A Tournament Matrix [8] is drawn by pitting all champions of one population against all champions of the other population, but is computationally expensive and the resulting graph is often difficult to analyse in practice. A CGTM consists in simply taking a limited number of champions for each population over the whole run (in this case, 25, evenly sampled exactly in the same way as for our equal-effort comparisons) and often produces more readable outputs [20]. The graphs shown in Fig. 4 indicate that 2-strikes-out runs produce a more stable evolutionary process than LEO runs. With the 2-strikes out runs, even when patches of different colours are scattered over the graph, a distinct division of the graph in two zones (one darker zone near the bottom-right and one lighter zone near the top-left) can be observed. Such a pattern is difficult to identify in the LEO runs.

In comparison, the 2SO algorithm followed a more steady approach, at the price of taking much longer to come up with really efficient results. While in the LEO algorithm the design of competing creatures would often be completely transformed over a few generations, the NSO used largely incremental modifications of working designs, with the occasional profound change (when a new individual would displace an eliminated champion to which it would not be related). While no utterly non-adaptive champions were found in the NSO algorithm, the curves indicate that appearance and tuning of successful designs took significantly longer than in the improved LEO algorithm.

VI. Conclusion and Future Work

We have introduced the N-strikes-out algorithm as a steady-state algorithm for coevolution. This seems to be the first truly steady-state coevolutionary algorithm, in that

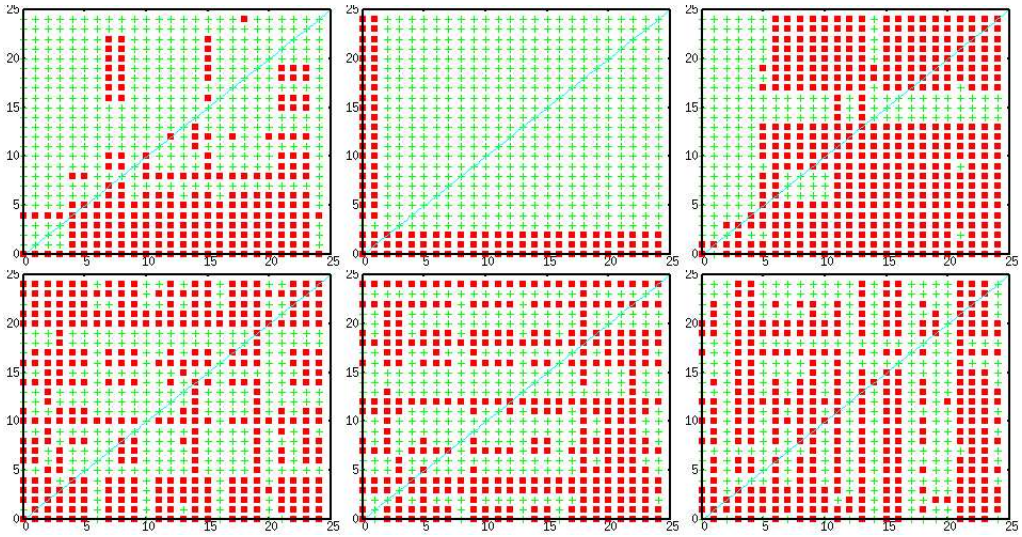

Fig. 4. Coarse-Grained Tournament Matrices (CGTM): For each run, 25 champions are evenly sampled from each population and then pitted against each other in an all-against-all manner. Point (x, y) is a dark square if champion x of population A defeated champion y of population B, a light cross otherwise. Top row graphs describe 2-strikes-out runs, bottom row graphs describe LEO runs. 2SO runs exhibit a pattern of darker/lighter separation, which is difficult to identify in the more chaotic grids generated by LEO runs. For more details on CGTM graphs see [20].

both evaluation and replacement occur asynchronously, in a steady-state fashion. We compared this algorithm against the Last Elite Opponent method introduced by Sims. With our original settings, using conservative replacement method and low selection pressure, a 2-strikes-out algorithm significantly outperformed the LEO algorithm. However, with an increased selection pressure and more explorative replacement methods, the LEO algorithm produced well-adapted designs much faster than the 2SO algorithm (though both methods obtained similar performance in the late stages of evolution).

Comparing our algorithm with Sims' original algorithm is significant for two reasons. First, it was apparently the only algorithm ever used in successful coevolutionary experiments involving artificial 3D creatures of this type. Second, it has been used (under different forms) in several studies. However the comparisons performed in this paper are still very preliminary, if only because we have used a simple, 'naked' LEO algorithm. More work, and more comparisons (including with refined versions of the LEO algorithm) is needed to determine the general properties of N-strikes-out algorithms, and how they can be adapted or improved.

REFERENCES

[1] R. Dawkins and J. R. Krebs, "Arms races between and within species," *Procs of the Royal Society of London, Series* B, vol. 205, pp. 489–511, 1979.

[2] C. D. Rosin and R. K. Belew, "New methods for competitive coevolution," *Evolutionary Computation*, vol. 5, no. 1, pp. 1–29, 1997.

[3] W. Hillis, "Co-evolving parasites improve simulated evolution as an optimization procedure," *Physica D*, vol. 42, pp. 228–234, 1990.

[4] S. G. Ficici, "Solution concepts in coevolutionary algorithms," Ph.D. dissertation, Brandeis University, May 2004.

[5] R. Dawkins, "Human chauvinism," *Evolution*, vol. 51, no. 3, 1997.

[6] J. M. Smith and E. Szathmary, *The Major Transitions in Evolution*. Oxford University Press, 1995.

[7] S. J. Gould, *Full House / Life's Grandeur - The Spread of Excellence from Plato to Darwin*. London: Jonathan Cape, 1996.

[8] S. Nolfi and D. Floreano, "Coevolving predator and prey robots: Do "arms races" arise in artificial evolution?" *Artificial Life*, vol. 4, no. 4, pp. 311–335, 1998.

[9] E. D. Jong, "Intransitivity in coevolution," in *Procs 8th Intl Conf on Parallel Problem Solving from Nature (PPSN 04)*, 2004.

[10] R. A. Watson and J. B. Pollack, "Coevolutionary dynamics in a minimal substrate," in *Procs GECCO 2001*, L. Spector, E. D. Goodman, A. Wu, and W. B. Langdon, Eds. Morgan Kaufmann, 2001.

[11] K. Sims, "Evolving 3d morphology and behavior by competition," in *Procs 4th Intl Works on Synthesis and Simulation of Living Systems (ALIFE IV)*, R. Brooks and P. Maes, Eds. MIT Press, 1994, pp. 28–39.

[12] C. W. Reynolds, "Competition, coevolution and the game of tag," in *Procs 4th Intl Works on Synthesis and Simulation of Living Systems (ALIFE IV)*, R. Brooks and P. Maes, Eds. MIT Press, 1994.

[13] J. Paredis, "Coevolutionary computation," *Artificial Life*, vol. 2, no. 4, 1995.

[14] J. Cartlidge and S. Bullock, "Learning lessons from the common cold: How reducing parasite virulence improves coevolutionary optimization." in *Congress on Evolutionary Computation (CEC 2002)*, D. Fogel, Ed. IEEE Press, 2002.

[15] S. G. Ficici and J. B. Pollack, "Pareto optimality in coevolutionary learning," in *Advances in Artificial Life: 6th European Conference (ECAL 2001)*, J. Kelemen and P. Sosik, Eds. Springer, 2001.

[16] E. D. De Jong and J. B. Pollack, "Ideal evaluation from coevolution," *Evolutionary Computation*, vol. 12, no. 2, 2004.

[17] J. Bongard and H. Lipson, "'managed challenge' alleviates disengagement in co-evolutionary system identification," in *Genetic and Evolutionary Computation Conference (GECCO 2005)*. MIT Press, 2005.

[18] L. V. Valen, "A new evolutionary law," *Evolutionary Theory*, vol. 1, pp. 1–30, 1973.

[19] T. Miconi and A. Channon, "An improved system for artificial creatures evolution," in *Procs 10th Intl Conf on Simulation and Synthesis of Living Systems (ALIFE X)*, L. Rocha, M. Bedau, D. Floreano, R. Goldstone, A. Vespignani, and L. Yaeger, Eds. MIT Press, 2006.

[20] ——, "Analysing coevolution among artificial creatures," in *Procs Evolution Artificielle 2005 (EA 05)*, E. G. Talbi, Ed. Springer-Verlag, 2005.