Serendipitous Scaffolding to improve a Genetic Algorithm's Speed and Quality

Heather J. Goldsby
Department of Computer Science and Engineering, Michigan State University, BEACON Center for the Study of Evolution in Action
hjg@msu.edu

Rebecca L. Young
Department of Integrative Biology, Center for Computational Biology and Bioinformatics, The University of Texas at Austin, BEACON Center for the Study of Evolution in Action
youngrl@utexas.edu

Jory Schossau
Department of Integrative Biology, Michigan State University, BEACON Center for the Study of Evolution in Action
jory@msu.edu

Hans A. Hofmann
Department of Integrative Biology, Center for Computational Biology and Bioinformatics, Institute for Cellular and Molecular Biology, Institute for Neuroscience, The University of Texas at Austin, BEACON Center for the Study of Evolution in Action
hans@utexas.edu

Arend Hintze
Department of Integrative Biology, Department of Computer Science and Engineering, Michigan State University, BEACON Center for the Study of Evolution in Action
hintze@msu.edu

ABSTRACT

A central challenge to evolutionary computation is enabling techniques to evolve increasingly complex target end products. Frequently, direct approaches that reward only the target end product itself are not successful because the path between the starting conditions and the target end product traverses through a complex fitness landscape, where the directly accessible intermediary states may be require deleterious or even simply neutral mutations. As such, a host of techniques have sprung up to support evolutionary computation techniques taking these paths. One technique is scaffolding where intermediary targets are used to provide a path from the starting state to the end state. While scaffolding can be successful within well-understood domains it also poses the challenge of identifying useful intermediaries. Within this paper we first identify some shortcomings of scaffolding approaches — namely, that poorly selected intermediaries may in fact hurt the evolutionary computation's chance of producing the desired target end product. We then describe a light-weight approach to selecting intermediate scaffolding states that improve the efficacy of the evolutionary computation.

CCS CONCEPTS

• **Computing methodologies → Genetic algorithms**; *Artificial life*;

KEYWORDS

Evolution, Genetic Algorithms, Scaffolding, Optimization

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5618-3/18/07…$15.00
https://doi.org/10.1145/3205455.3205617

ACM Reference Format:
Heather J. Goldsby, Rebecca L. Young, Jory Schossau, Hans A. Hofmann, and Arend Hintze. 2018. Serendipitous Scaffolding to improve a Genetic Algorithm's Speed and Quality. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205617

1 INTRODUCTION

Scaffolding is an evolutionary computation approach to enabling techniques to evolve more complex end products than are possible with direct evolution and one fitness function. In particular, scaffolding tries to subdivide the complex path of evolutionary adaptation into smaller steps of intermediary states. Each state is supposed to be much simpler to evolve than the ultimate state, and the solutions to these simple states build on one another to advance the population to the final complex end state, which was previously unobtainable. Bongard [2, 3] has applied scaffolding for both morphology and behavior and demonstrated its efficacy. Grabowski *et al.* have made use of scaffolding for evolving complex robot behavior, but reportd only a marginal improvement [8]. While scaffolding can be implemented as to mirror the structure of phylogenies (and thus have clear parallels to Darwinian evolution), one problem is identifying useful intermediate states where each successive intermediate state easily helps evolution move the population toward the final desired state. It seems intuitive that these intermediary states should fall between the initial condition and the desired objective. Imagine the evolving substrate (system) should eventually integrate information from four input sources. A fitness function that only measures success if all sources are integrated will not reward any intermediary steps integrating fewer than four sources. In this example the scaffolding steps appear to be obvious: reward the integration of sources in a graduated fashion. However, even in this simple case, it is unclear what order the sources should be rewarded in. Moreover, for other fitness functions, the scaffolding steps of reward may be less obvious due to the complexity of the environment, or in cases of multi-objective optimization [4] individual components could be antagonistic. In the previously tested

cases [2, 3, 8] intermediary steps were hand-designed to fit intuitive graduation of the final problem, and while improving adaptation, finding an optimal intermediary is an unsolved problem. Here we ask what one could do if it is unclear how to hand-design the intermediary steps: Is it possible to discover serendipitous scaffolds for arbitrary problems, where either the intermediary steps are counter intuitive to humans or discovered intermediaries outperform those identified by humans?

In this paper, we propose an approach to selecting scaffolding states for fitness functions. We compared the speed of adaptation on a scaffolded evolutionary path to that on a non-scaffolded evolutionary path. For this study, the intermediary environments used for scaffolding are either hand-designed or randomly generated. However, the speed of a genetic algorithm does not only depend on the fitness function, but also the substrate subject to evolution (the system), the specific selection regime, and the kinds of mutations (changes) the evolved system experiences. If the system does not contain meaningful interactions (epistasis or pleiotropy) [15], or does not require valley crossing mutations, we would not expect intermediary steps to make a difference — the fitness landscape would essentially be smooth. On the other hand, if the system has too much epistasis, mutations could interact in unpredictable ways and the fitness landscape would be rugged. Therefore, we will take epistasis into account, by using a simple computational model system with low epistasis and a complex model system with a much higher degree of epistasis. We ignore specific treatments for pleiotropy because higher levels of pleiotropy in computational evolution often include epistasis as a result. Any system we choose must allow for randomly generated and hand-designed intermediary patterns. Lastly, we assume not all combinations of intermediary patterns when serially forced upon the populations will accelerate evolutionary adaptation, so we need an automated method of differentiating patterns by their effect on evolution.

To satisfy these conditions we used a pattern formation task where a grid of cells is sequentially populated by colored cells. This kind of task comes from evolutionary developmental biology [7] (sometimes referred to as the French Flag model where similarity to a striped pattern is used as the fitness function) and investigates how developmental systems evolve. The initial pattern was often either blank or random and the target and intermediary patterns were easily generated. Patterns that humans would intuitively hand-design as intermediary steps between the initial and final patterns are usually averages (or weighted averages) between those two patterns, with biases toward one or the other to generate finer gradations. The two computational systems we used start with an empty pattern and are then allowed to perform modifications by using commands like "add cell", "move cell to the left", or "divide cell". Aside from their complexity, the main difference is in the way these commands are encoded. In the simple system the linearly applied commands are specified by a list susceptible to mutation so the pattern formation is affected by those mutations. In the complex system each cell is controlled by a Markov Brain [9] which have been used before as a substrate for neuroevolution in various contexts [5, 12, 14] as well as in pattern formation [7].

Here we will show: randomly generated intermediary scaffolds can be used to improve evolution (either by accelerating adaptation or by achieving a superior final fitness) with no knowledge of problem domain; that even long sequences of random intermediate patterns can evolve better than evolving straight to a final pattern; and we will compare evolution on serendipitous paths with intuitively-designed paths.

2 BACKGROUND & METHODS

Here we provide a brief overview of both the simple and more complex models of developmental processes. Both models capture relevant details of biological developmental processes however, they represent different trade-offs between level of abstraction (in particular epistatis) and compute time. In particular, the simplified model of development is more abstract (and contains less epistasis) but runs at a fraction of the speed of the complex model of development. As such, it is both suitable for many questions within developmental biology and also provides an excellent test bed for exploring pattern space more thoroughly. Thus, we use this model with many more patterns (100) and replicates to ensure the generality of our approach. The more complex model includes a higher degree of epistasis. The increased detail results in additional run time and thus we use the complex model for more targeted testing with fewer patterns and replicates. Together the models provide complementary analyses and insights.

2.1 Simplified Model: Pattern Matching

In the simple model of development we abstracted away much of the developmental process into expression of 100 serial commands. The expression of these commands created the phenotype pattern that was then matched to a randomly generated target pattern. Each target pattern was constructed as a 6x6 grid of colored cells with each cell representing 1 of 5 possible colors, fundamentally represented as integers 0 through 4. Each organism's fitness was assessed based on its ability to produce a phenotype pattern matching the target pattern according to fitness $w = 2^{\text{matches}}$ with the number of matches as exactly the Hamming distance between the phenotype pattern and target pattern (a value between 0 and 36). Each organism's genome is expressed into a phenotype pattern by serial translation of 100 position-color commands comprised of 3 numbers, making 300 genomic sites in total. There were 5 such possible commands: increment color with modulo at location, and 4 commands for copy color to location from adjacent (from Left, Right, Up, or Down) which does nothing if there is no valid adjacent cell in that direction. For example, an inefficient algorithm could have a genome whose expression means the first 6 numbers detail 2 color increments at position (0,0) to achieve the 3$^{\text{rd}}$ color, and the next 12 numbers describing the same operations in the next 2 adjacent locations. An evolved more efficient algorithm might better use the copy commands for the next 2$^{\text{nd}}$ and 3$^{\text{rd}}$ locations, thereby reducing the encoding by 6 genomic sites. A population of 100 organisms were evolved from random initial genomes through entire-population replacement at each t to $t+1$ with Roulette Wheel selection. Mutations were applied at replication time with a per-site probability of $\mu = 0.001$ to generate a random number within the valid range for that site: $0-5$ for position x and position y within the 6x6 grid, and $0-4$ for color commands.

Target patterns were generated with 4 hand-designed and 96 randomly designed. The hand-designed patterns were selected for

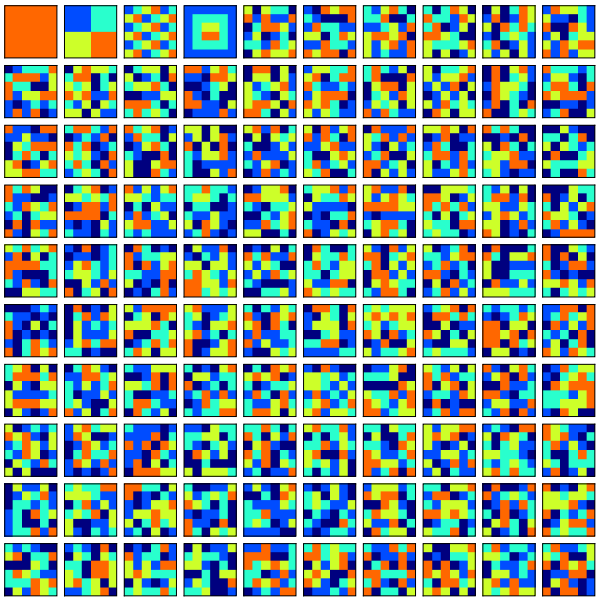

Figure 1: 100 patterns, 4 experimenter-designed, the other 96 random. The initial pattern organisms start their developmental process from is made from empty (value is 0) cells.

ease of testing early in experimental design, and thus allow easy visual inspection of pattern similarity. See Figure 1.

Evolution could create a genome that would express non-constrained development with all color increment commands to directly set the colors of each position in the grid. Evolution could construct such a genome for patterns comprised of only the first few colors in the series, as achieving such a target pattern would require less than 100 increment commands. However, colors later in the sequence require multiple increment commands and thus become rather costly from a genomic perspective if using only increment commands. In these cases evolution should favor the use of the more developmentally-constrained 4 color direction copy commands.

2.2 Complex model: Digital Tissue

This model was originally developed to enable the evolution of multicellular organisms (called *digital tissues*), where each digital tissue starts as a single cell that develops into a 2D tissue of differentiated cell (similar to [6]). These tissues can be used to address evolutionary-development questions within biological studies. The behavior of the cells are controlled by evolving Markov Brains, which are networks of deterministic and probabilistic logic gates encoded in an evolvable fashion [5, 12]. These 2D tissues of differentiated cells evolve in response to selection for a particular target pattern or body plan. Figure 2 provides an illustration of the 15 different target patterns used for this study. Each large square represents a body plan or pattern. Each smaller square represents a cell, where cell fate is indicated by color.

Each tissue starts its life as an isolated cell and over development time can replicate to produce a digital tissue and, in conjunction with its neighboring cells, select a cell fate to produce a pattern.

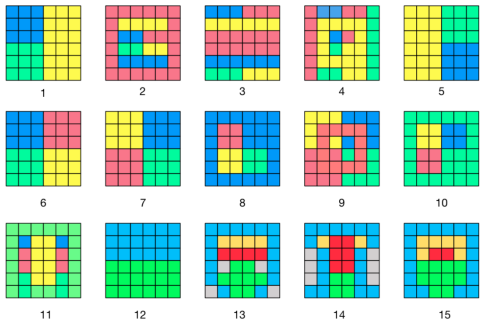

Figure 2: Patterns used for the digital tissues experiments

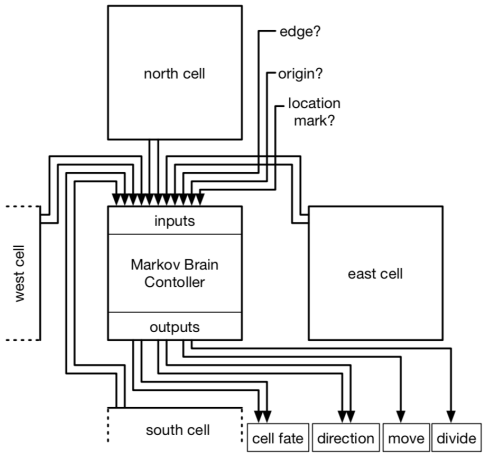

Figure 3: Illustration of the inputs and outputs for each cell. Each cell, except those on the edge of the digital tissue, have four cardinal neighboring cells. Each cell receives inputs about the cell fate each neighboring cell expresses, as well as information about being on the edge of the tissue, and if it is the origin cell or not. Cells can decide to migrate or divide, whereas migration happens before division. In addition cells have to identify a direction they would like to migrate or divide into. Each cell can identify which current cell fate it has, signaling its fate to its own four neighbors.

As a result of evolution, these patterns correspond to – but are not always exactly the same as – the target patterns. Each cell within the tissue has several capabilities described in Figure 3 and following. In general, we provided the cells with a wide-variety of capabilities to enable evolution to discover the most useful for the desired task. The capabilities can be used by the cell to determine and then express its own cell fate (depicted as color), which is the only aspect of the cell that affects the body pattern of the tissue and thus its fitness. The evolutionary success (or failure) of the digital tissue is determined by the degree to which its cells express a target pattern rewarded exponentially so that more matching cells give an exponential increase in fitness:

$$W = 1.5^{(number\ of\ cells\ correct)}, \quad (1)$$

where a cell is correct if its color matches the color of the corresponding cell in the target pattern.

The behavior of each cell, including its replication, migration, communication, and expression of cell fate, is controlled by a Markov Brain [5, 12]. Markov Brains are networks of probabilistic and deterministic logic gates. A string of numbers (i.e, the genome) encodes the function and connectivity of gates. Mutating the numbers of the genome thus can change the function of a Markov Brain. These Markov Brains have been used to study a wide-variety of topics within evolutionary computation [16], artificial life [10], and biology [14]. Here we select Markov Brains using different techniques based on the efficacy with which they encode the development of a digital tissue into a desired target pattern. The evolutionary setup is straightforward: a population of Markov Brains experience selection on the basis of a pattern. Moran selection, where part of the population is replaced each generation is used. When the fitness of a Markov Brain is evaluated it is placed within a single-cell within an environment for a digital tissue. We allow the digital cell (controlled by the Markov Brain) to unfold for 150 time steps, where each time step the outputs of the brain reflect the behavior of the cell (replicate, migrate, establish cell fate). At the end of this time period, we compare the tissue pattern to the desired pattern and calculate fitness based on similarity.

When an individual Markov Brain from the population is selected for replication then point mutations, duplications, and deletions are applied to a duplicate genome. This new genome is then translated into a Markov Brain. For the translation step, the genome is read sequentially. Once a particular number combination (42 and then 213) is found, the next region of the genome is translated into a gate. This process is similar to how the genes of natural organisms are transcribed and then translated into proteins. Each gate can either connect to other gates by writing their outputs into hidden nodes that other gates can read from, or by connecting to input nodes and output nodes. This way, the resulting Markov Brain can obtain information about its environment by using input nodes, integrate information by using hidden nodes, and relay actions to the environment using output nodes.

Sensor Inputs: We provide the Markov Brains (and thus the digital cells they control) with the following types of sensory information (Figure 3): (1) if the cell is at the origin (the top left cell); (2) if the cell is on the edge of the space provided for the digital tissue body; (3) any marking (a number) in the cell's current location (a form of stigmergic communication [1] designed to be analogous to secreting a chemical at a location); (4) the cell fate (color) of the cell's cardinal neighbors; (5) any messages emitted by neighbor.

Actuator Outputs: We provide the Markov Brains (and thus the digital cells they control) with the following capabilities to interact with neighboring cells and specify their fate (Figure 3): (1) a cell fate (depicted as the color of their cell); (2) if the cell is going to migrate; (3) if the cell is going to reproduce; (4) the direction that the cell would like to move and reproduce. (5) a mark for its location within the digital tissue (analogous to a chemical); (6) a message that will be shared with its cardinal neighbors. To avoid conflicts, a cell first moves and then reproduces ensuring it does not attempt to move over the offspring cell it has just created.

For all experiments using the complex system we used replicate populations of evolving Markov Brains ($N = 100$). Each genome began as length 10,000 and 10 initial gates. We used a Moran Process, where 5% of the population was replaced every update. In particular, parents were selected using Roulette Wheel selection and survivors were selected randomly. The population replicated asexually. Mutations were applied at the time of replication. Each genome had a 5% chance of receiving one insertion and one deletion mutation. Each site within the genome had a 5% chance of receiving a copy mutation.

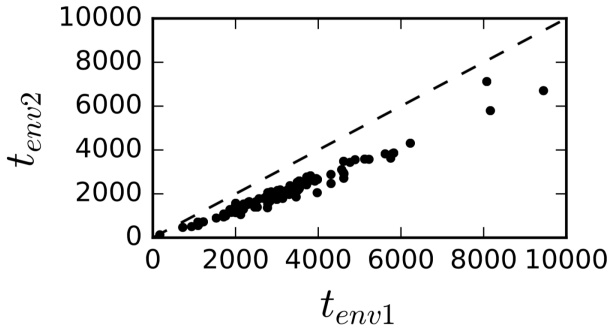

Figure 4: Time to pattern A vs. time to pattern B, dashed line is the point equivalency where time to pattern A is as fast as time to pattern B.

3 RESULTS: THE SIMPLIFIED MODEL

We use the simplified model to perform a suite of experiments studying the effects of serendipitous scaffolding. In particular, we performed serial transfer between targets: Evolution was allowed to continue until the population achieved 32 matches of the total 36 for their one pattern, then after all populations achieved 32 they were transferred into those 99 environments they had not yet experienced. Replication was $N = 100$, which were sorted and the 5% longest-running results were removed as they were likely extremely long and unrepresentative outliers [11, 13]. We then compared the elapsed time for a population to achieve 32 matches on the first pattern (from native to pattern A) with the time to get 32 matches on the second pattern (see Figure 4). As it turns out, the second pattern is generally achieved faster than the first.

The skew in Figure 4 suggests that regardless of which two patterns were chosen, the patterns share common elements general to the problem domain allowing populations to partially preadapt to the novel second pattern. In addition, the wide spread of adaptation times suggests that scaffolding is possible: Maybe an intermediary pattern X can be found such that the sequence A to X to B might take less time than A to B directly?

To estimate if these paths with an intermediary step exist, the ratio between the time it takes to go from A to X to B and the time it takes to go from A to B directly is computed:

$$r = \frac{t(A, X, B)}{t(A, B)} \tag{2}$$

This ratio is $r = 1$ if the detour over the intermediary X takes as long as evolving from pattern A to B. A naïve expectation is that each transition takes the same amount of time and that a path with $2x$ more transitions results in a $2x$ increase in the r ratio. We find

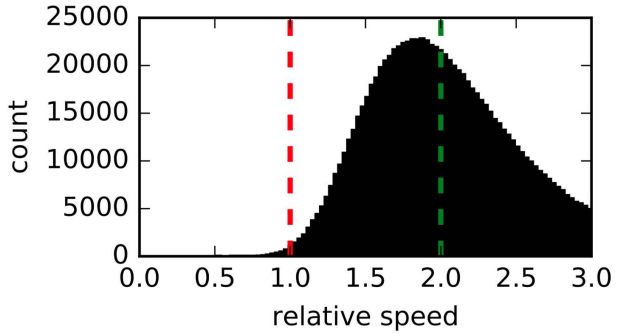

Figure 5: Distribution of time ratios (see 2 for all possible paths A to X to B, the vertical green line indicates the naïve assumption that a path with two transitions takes twice the time of a path that contains only one. Ratios smaller than $r = 1.0$ (red line) indicate paths that contain three steps and evolve faster than the original path.

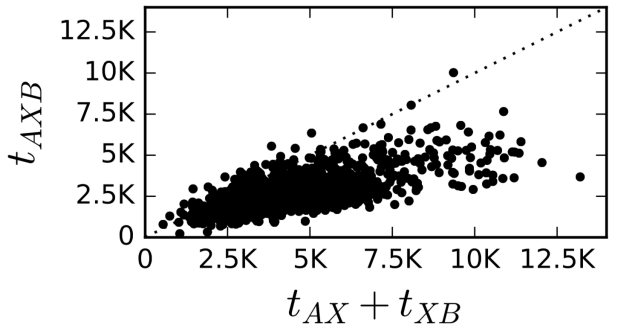

Figure 6: Correlation between the predicted time for evolution (x-axis $t_{AX}+t_{XB}$) and the actual time it takes to evolve (y-axis t_{AXB}). The dashed line represents a ration of 1.0 where the speed of prediction and measurement are identical. 100 experimental replicates per condition, changing the pattern when 32 out of 36 cells matched between the patterns.

that indeed most paths containing 2 transitions would take about twice as long as those with 1 transition (See Figure 5).

While we found that adding an intermediary step generally increased the time to evolve to the final pattern (the mean is 2.1287 with a standard deviation of 0.6476 and a variance of 0.4194), there were also a small number of length three paths that evolved significantly faster than the original (see the Figure 5; ratios left of the red line).

This increase in efficiency begs the question if our estimates have any power to predict the time to evolve from A to B by going through an intermediary environment rewarding similarity to pattern X. As validation, we picked 1,800 triplets of patterns (A to B to X) in such a way that each ratio (see 2) was represented equally often. When we then evolved these populations over the triplet paths from A to X to B and compared that with the predicted time, we found a correlation between prediction and empirical measurement (Pearson correlation coefficient of 0.6679 with a p-value of $6.3975x10^{-}233$, see Figure 6). In fact, populations evolved faster than expected in most cases.

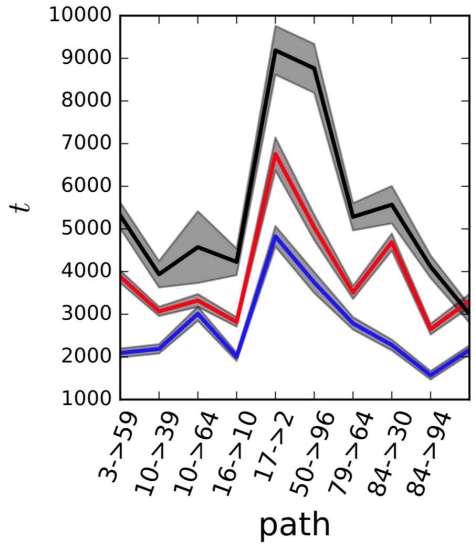

Figure 7: Comparison of evolutionary time elapsed for populations to reach 32 matches. Evolution from pattern A to B in black (actual numbers shown as labels on the x-axis), Evolution with three extra intermediate steps X, Y, and Z in red, and evolution with intuitively-generated intermediate steps based on Hamming distance in blue. The gray backgrounds for each line represent the standard errors.

This optimization strategy may hold for even longer intermediary path lengths that yield shorter evolution times, but there may reasonably be an upper bound on path length beyond which no improvements are observed. In other words, it might be that our prediction method is only applicable to pathways of single intermediary patterns, such as A to X to B. To see if predictive capability holds for longer pathways we performed a similar analysis with 3 intermediary patterns between A and B: A, X, Y, Z, B. When searching through the space of our initial measurement, we find 10 pathways where 3 intermediary patterns produces a shorter evolutionary run than 0 intermediary patterns. For each predicted sequence we tried $N = 100$ empirical replicates and compared the elapsed time to the prediction. As a comparison, we also generated 3 intermediary patterns X', Y', and Z' for all 10 pathways. These intermediary patterns differ in a graduated fashion, and represent possible transition stages one would intuitively create in order to speed up evolution through naïve interpolation.

We find that the evolutionary path through a specific extra 3 intermediary patterns takes less time than the original path with 0 intermediary patterns in 9 of 10 observations (See Figure 7 red). At the same time the purposefully designed intermediate patterns all evolve even faster, as expected (See Figure 7 blue).

It is noteworthy that the faster paths discovered by this search technique do not fit preconceived notions typically attributed to intermediary steps that "scaffold" to the end target, since they are very different from the hand-designed intermediary patterns (see Figure 8). This confirms the original idea that even serendipitous intermediary fitness functions or objectives can accelerate evolutionary adaptation.

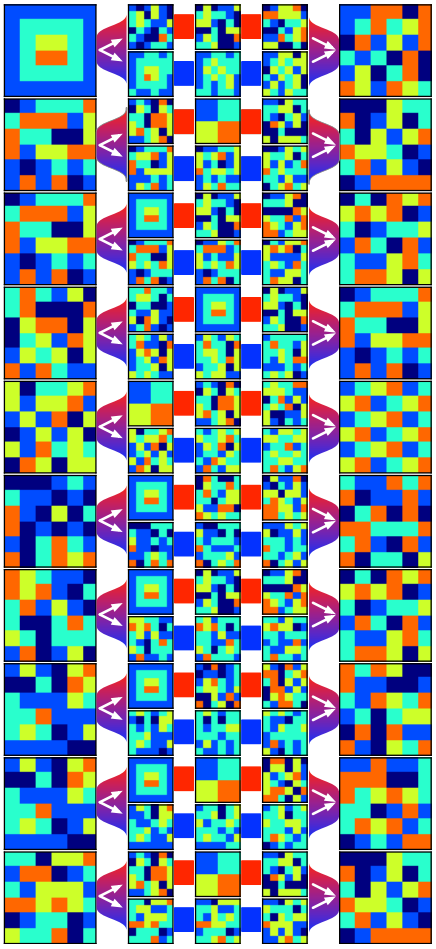

Figure 8: Pattern comparison. The left column represents the 10 start patterns A, and the right column presents the target patterns B, the intermediary patterns in the middle are the extra introduced steps X, Y, and Z, which shorten the time to evolve from pattern A to B. For each start and target pair (A to B) there are two possible paths, the upper (highlighted in red) is the serendipitous path generated from the 100 initial patterns (see Figure 1, the lower ones (highlighted in blue) are the generated intermediary patterns (the red and blue color reflect the measurements from Figure 7. The gray backgrounds for each line represent the standard errors.

There could be various explanations for why this method works. While it is possible that intermediary patterns provide a smoother path through the fitness landscape detouring potential local maxima, it is also possible that evolvability itself is improved. There is no computationally tractable way to map a highly dimensional fitness landscape, but we can assess evolvability. For the 100 initial patterns we can create a sequence of patterns with an increasing transition time. We first identified the pair with the longest transition time, moved that pair to a list, and kept searching for the next longest time adding that to the list and so forth until we had a list of unique patterns. This list as a sequence predicts an ever-increasing transition time (slope of 22.49, r-value 0.62 see Figure 9). We performed 100 replicate experiments evolving from one to the next pattern until we reached the final pattern following the predicted path, transitioning again when 32 out of 36 colors match. We found that the time it took for each pattern to evolve to the next did not increase in time, but instead became shorter the more transitions had been made (see Figure 9, slope -1.66, r-value -0.09). While this experiment did not exclude the idea that additional scaffolding patterns flatten the fitness landscape, it shows that evolvability improved.

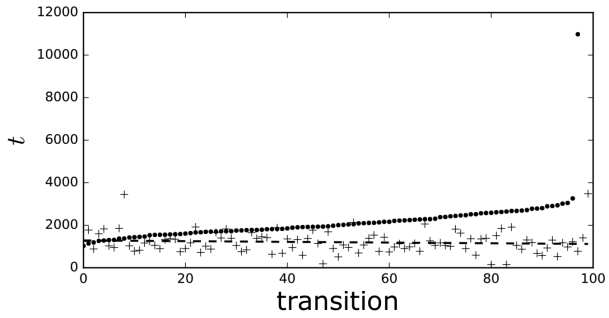

Figure 9: Evolution of evolvability. Black dots show the time to adapt to each pattern sequentially chosen from a sequence for which we predicted an steady increase in adaptation time (slope 22.49, fit not shown). Black + shows the average time it takes to evolve from pattern to pattern over 100 replicate experiments. The slope for the experiment was fitted and shown as a dashed line (slope -1.66).

4 RESULTS: DIGITAL TISSUE

For these experiments, we use Markov Brains to evolve the patterns depicted in Figure 2. For the simplified developmental model portion of the work, we quantified the desirability of the approach based on the time it took to reach $88.\overline{8}\%$ pattern similarity (32 of 36 possible cell matches). For this portion, there are some patterns where the overall fitness does not reach 32/36, the benchmark for the simplified model. Thus, rather than using time, all measurements are of total fitness achieved over a given period of time.

To start, we observe how transferring between environments affects fitness. Here we compare the total fitness achieved when evolving directly within an environment B and when adding an earlier environment A producing the path AB. Figure 10 depicts these results. In general, adding an earlier pattern A can either improve or disrupt fitness. This suggests that even within the complex developmental model, which exhibits a higher degree of epistasis, there are cases where intermediary fitness functions will improve overall results and showing scaffolding can aid evolution.

Furthermore, we can also demonstrate that the paths through patterns selected in this fitness landscape are not always intuitive. Adding intermediary scaffolding patterns has the potential to either improve or decrease final fitness values as expected. See Figure 11 for further details. The pathways that improve the final fitness values for the end state are not always intuitive to users.

To explore the role of intermediary states, we first identified paths AB through the fitness landscape that had poor performance

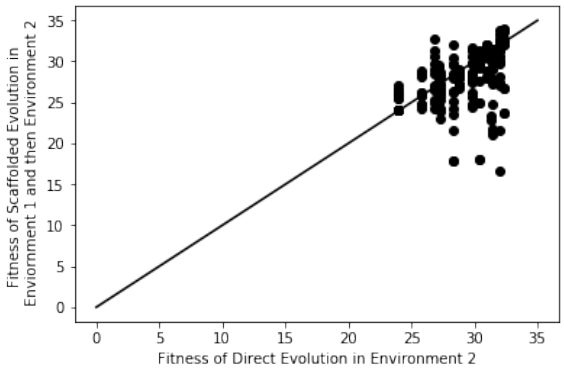

Figure 10: Transition fitness values. The x-axis represents the fitness of replicates evolved directly to a particular pattern. The y-axis represents the fitness of replicates evolved first to one pattern and later to the target pattern. Pre-evolving in a different environment in this way produces negative and positive fitness benefits depending on the environment.

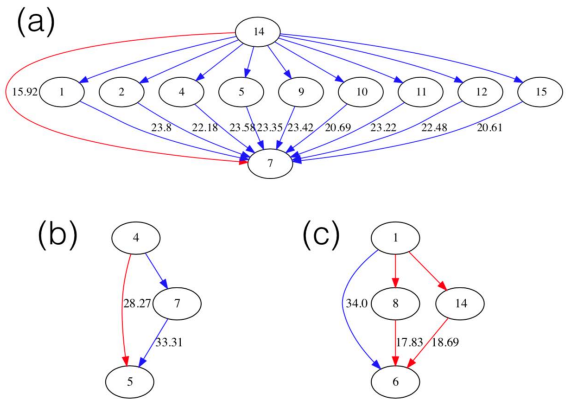

Figure 11: Examples of scaffolding positively and negatively affecting final fitness scores. Here each node represents a pattern. Arrows denote the direction of evolution. Arrow values denote the fitness in the final state. (a) The transition from pattern 14 to pattern 7 is quite challenging, achieving a fitness of 15.92 on pattern 7. However, by adding an intermediary (patterns 1, 2, 4, 5, 9, 10, 11, 12, or 15) fitness is substantially improved. (b) Similar to part (a), fitness is improved by adding scaffolding. However, as part (c) demonstrates adding intermediaries does not always improve final fitness.

and achieved low fitness values over 40,000 updates. We then examined alternative pathways AX and XB, where X could be any of the other patterns, and focused on those pathways where AX and XB both achieved fitness scores greater than $AB + delta$, where $delta$ is a fitness offset used to make sure the transition truly had a greater chance of outperforming the more direct approach. Note that similarly to our experiments in the simplified model, these

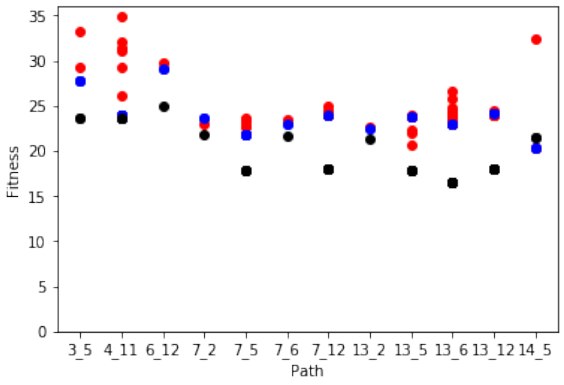

Figure 12: A comparison of the performance of direct paths AB with indirect paths AXB for the complex model of development. In general, adding an intermediary step in over 75% of cases improved fitness. Along the x-axis are paths that were identified as being able to be improved (AB). Along the y-axis is fitness. Black dots represent the predicted fitness values for the path based on original test results. Blue dots represent a direct path between the two states (AB). Red dots represent a scaffolded path (AXB). In the same amount of time, the majority of the scaffolded paths were more effective at achieving higher fitness.

predictions are based on the independent evolution of AX and XB, not the combined result.

To assess our approach for selecting scaffolding steps, we analyzed 12 different paths AB, where each path had one or more potential AXB improvements (49 potential improvement paths overall). We performed replicated $N = 5$ for each path AB and each potential improvement AXB. We then compared the fitness of the direct route (AX) with the fitness of the indirect route (AXB). Both approaches were constrained to the same amount of evolutionary time. In short, if the direct route used 60,000 updates to go from pattern A to B (30,000 updates in each environment), then the indirect route from pattern A to pattern X to pattern B also used 60,000 updates total (20,000 updates in each environment). Of the 49 potential improvement paths, over 75% (37 paths) resulted in an improvement, where the fitness of the indirect scaffolded (AXB) path was greater than the fitness of the direct path (AX). Figure 12 shows our results.

We extended this approach by examining paths with two intermediary steps ($AXYB$). We used the same method as previously for identifying potential intermediary patterns. In particular, we identified partial paths where the fitness of AX, XY, and YB all had a fitness greater than $AB + delta$. We then evolved these direct (AB) and indirect routes ($AXYB$) for the same amount of time, i.e., 80,000 updates. The scaffolded approach had 20,000 updates per pattern, while the direct approach had 40,000 updates per pattern. We explored 169 potential indirect paths ($AXYB$) for improving fitness results over 6 direct paths (AB). Of these, over 65% improved final fitness compared with the direct approach. Moreover, 100% improved fitness compared with the shorter runs used to identify paths with low fitness values. This suggests scaffolding steps do

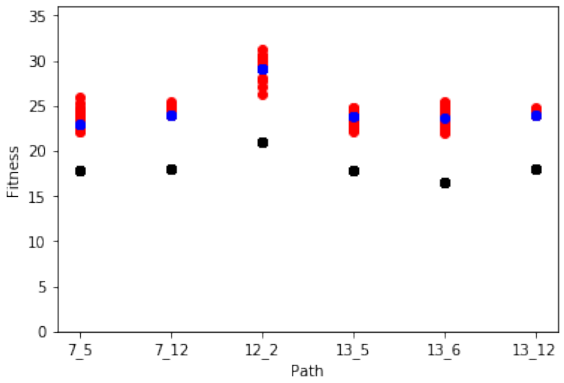

Figure 13: Adding two scaffolded steps in over 65% of cases improved fitness. Along the x-axis are paths that were identified as being able to be improved. Along the y-axis is fitness. Black dots represent the predicted fitness values for the path based on original test results. Blue dots represent a direct path between the two states. Red dots represent a scaffolded path. In the same amount of time, the majority of the scaffolded paths were more effective at achieving higher fitness.

help improve fitness. Unsurprisingly, simply allowing more evolution of the population in the desired target environment also improves fitness. Figure 13 depicts our results. These results also suggest that epistatic interactions may make the identification of desirable paths more complicated than in the simplified model.

5 CONCLUSIONS

One of the most intuitive ideas for improving GA performance is scaffolding. When evolution is unable to produce satisfactory results the improvement is often to decompose the problem to allow evolution smaller, and presumably easier, intermediary goals that build on each other to more quickly reach the final goal.

We showed that the intermediary goals humans would design can indeed accelerate evolutionary adaptation, but that even randomly generated ones can serve the same purpose. However, not any set of random patterns suffices, these patterns need to be carefully selected. In this case, we measured the elapsed time to evolve from one pattern to the other, and constructed paths based on these predictions. This approach is computationally expensive, so for future applications we suggest to fit this equation [17]:

$$\bar{W} = (\beta t + 1)^{\alpha} \tag{3}$$

to the fitness data from short evolutionary experiments in order to estimate how mean performance ($\bar{W}$) would have behaved with more run-time (t).

We also showed that this method not only works in a simple model with little epistasis and smooth fitness landscape, but also in a complex model with high epistasis. While we can distinguish what phenomenon explains this acceleration due to serendipitous intermediary patterns, we found evidence that evolvability of the system improves over multiple transitions. The model we used lends itself to this kind of experiment, since new intermediary patterns are easily created. Obviously, one can not apply this method to a new application without some adjustments. We think this method is particularly useful in cases where the fitness function is not well-understood and the intermediary steps are counter intuitive. In these cases the experimenter can discover and utilize such serendipitous paths to either accelerate adaptation or, like shown for the complex model, allow for an increase of performance or complexity of the final solution. For example, the Handwritten Digit problem may benefit from this approach. Some constructed lines and curves could be scaffolds, but it is not intuitive which ones to use and in what order to use them.

ACKNOWLEDGMENTS

This material is based in part upon work supported by the National Science Foundation under Cooperative Agreement No. DBI-0939454 and grant DEB-1655715. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Eric Bonabeau. 1999. Editor's introduction: stigmergy. *Artificial Life* 5, 2 (1999), 95–96.
[2] Josh Bongard. 2011. Morphological change in machines accelerates the evolution of robust behavior. *Proceedings of the National Academy of Sciences* 108, 4 (2011), 1234–1239.
[3] Josh C Bongard. 2011. Morphological and environmental scaffolding synergize when evolving robot controllers: artificial life/robotics/evolvable hardware. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 179–186.
[4] Kalyanmoy Deb. 2014. Multi-objective optimization. In *Search methodologies*. Springer, 403–449.
[5] Jeffrey A Edlund, Nicolas Chaumont, Arend Hintze, Christof Koch, Giulio Tononi, and Christoph Adami. 2011. Integrated information increases with fitness in the evolution of animats. *PLoS Comput Biol* 7, 10 (2011), e1002236.
[6] Diego Federici and Keith Downing. 2006. Evolution and development of a multicellular organism: scalability, resilience, and neutral complexification. *Artificial life* 12, 3 (2006), 381–409.
[7] Heather J Goldsby, Rebecca L Young, Hans A Hofmann, and Arend Hintze. 2017. Increasing the complexity of solutions produced by an evolutionary developmental system. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 57–58.
[8] Laura M Grabowski and Maganã Javier M. 2013. Building on Simplicity: Multistage Evolution of Digital Organisms. *Artificial Life* 14 (2013).
[9] Arend Hintze, Jeffrey A Edlund, Randal S Olson, David B Knoester, Jory Schossau, Larissa Albantakis, Ali Tehrani-Saleh, Peter Kvam, Leigh Sheneman, Heather Goldsby, et al. 2017. Markov brains: A technical introduction. *arXiv preprint arXiv:1709.05601* (2017).
[10] Arend Hintze and Masoud Miromeni. 2014. Evolution of autonomous hierarchy formation and maintenance. In *Proceedings of the 14th International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14), New York, NY, USA*, Vol. 30. Citeseer, 366–367.
[11] John R Koza. 1992. *Genetic programming: on the programming of computers by means of natural selection*. Vol. 1. MIT Press.
[12] Lars Marstaller, Arend Hintze, and Christoph Adami. 2013. The evolution of representation in simple cognitive networks. *Neural computation* 25, 8 (2013), 2079–2107.
[13] Jens Niehaus and Wolfgang Banzhaf. 2003. More on computational effort statistics for genetic programming. In *EuroGP*. Springer, 164–172.
[14] Randal S Olson, Arend Hintze, Fred C Dyer, David B Knoester, and Christoph Adami. 2013. Predator confusion is sufficient to evolve swarming behaviour. *Journal of The Royal Society Interface* 10, 85 (2013), 20130305.
[15] Bjørn Østman, Arend Hintze, and Christoph Adami. 2011. Impact of epistasis and pleiotropy on evolutionary adaptation. *Proceedings of the Royal Society of London B: Biological Sciences* (2011), rspb20110870.
[16] Jory Schossau, Christoph Adami, and Arend Hintze. 2015. Information-theoretic neuro-correlates boost evolution of cognitive systems. *Entropy* 18, 1 (2015), 6.
[17] Michael J Wiser, Noah Ribeck, and Richard E Lenski. 2013. Long-term dynamics of adaptation in asexual populations. *Science* (2013), 1243357.