Abstract—Generalization is a perennial issue in evolutionary computation. The ability of evolution to find excellent special-purpose solutions to a problem means that, in some cases, evolutionary techniques generalize poorly. In this study we demonstrate a system that generalizes apoptotic cellular automata rules from a small evaluation arena to a larger one. The generalization preserves many of the features of the cellular automata while increasing the size of the automata’s time-history. The fidelity of the appearance of the generalized rules to their progenitors is high but varies for different progenitors. The generalization is attained by use of single parent techniques. These techniques employ a set of one or more immortal progenitors that are available for crossover but do not otherwise participate in the population. The form of single parent technique used here is novel and the study includes parameter tuning for its use.

I. INTRODUCTION

Cellular automata are a type of discrete model of computation. A cellular automata has four parts,
1) A set of cells.
2) A set of states that cells can have.
3) For each cell, a subset of the set of cells that form the neighborhood of a cell.
4) A rule that maps the set of possible cell states of a neighborhood to a new state for the cell with which the neighborhood is associated.

Cellular automata (CA) are a form of discrete dynamical systems that often exhibit self-organizing behavior. A cell population will evolve according to local transitions rules. The updating according to the transition rules may be synchronous or asynchronous. Many real world systems are dynamic in nature, and can be modeled as cellular automata. CA can be used as models for complex natural systems that contain large numbers of identical components experiencing local interactions[20], [15].

CA have been applied to the study of a diverse range of topics, such as structure formation[8], heat conduction[9], language recognition[14], traffic dynamics[12] and cryptography[2], to name a few. CA have also been used for more aesthetic purposes, such as image and sound generation. Serquera and Miranda of the Interdisciplinary Centre for Computer Music Research, UK, have published many works on the use of CA for sound synthesis [16], [1]. Much of their work consists of mapping the histogram sequence of a CA evolution onto a sound spectrogram, which produces spectral structures evolving in time. It is claimed that the mapping produces a “natural” behaviour, and can replicate acoustic instruments[17].

CA have also been applied to the visual arts. CA have been used to produce artistic images[3], [13], and their use has been extended to the fields of architecture and urban design[18], [10]. An interesting application has been the use of CA in simulating the emergence of the complex architectural features found in ancient Indonesian structures, such as the Borobudur Temple[19]. Ashlock and Tsang[3] produced evolved art using 1-dimensional CA rules. CA rules were evolved using a string representation. The CA either underwent slow persistent growth, or planned senescence. The resulting fitness landscapes were conjectured to be rugged with many local optima. These systems produced aesthetically pleasing images. This study revisits these fitness functions with a more general space of cellular automata rules, focusing on understanding the fitness landscape.

In this study we examine the ability of a unique form of evolutionary computation that uses single parent techniques [5], [4] to generalize cellular automata rules evolved under one fitness function to satisfy a related but distinct fitness function. The rest of this study is structured as follows. In Section II we give the design of experiments, including a detailed description of the single parent technique. In Section III the results are presented and discussed. In Section IV we draw conclusions and discuss potential next steps.

II. DESIGN OF EXPERIMENTS

We begin with an overview. A collection of runs of an evolutionary algorithm are used to secure an initial population of cellular automata rules. Four of the rules, with unique appearances, are harvested and used in single parent experiments. The first of the harvested rules is used in a series of six experiments that tune the single parent rate parameter of the algorithm. The other three harvested rules are used with the parameters selected in the first six experiments. All single-parent experiments consist of 30 replicates using independent random number seeds. An additional set of 30 runs is done to create baseline fitness values to compare to the single parent rules.

A. The Cellular Automata Representation

The cell states for the automata used in this study are the numbers \{0, 1, 2, 3, 4, 5, 6, 7\}. The automata are one-dimensional arrays of 401 cells that wrap at the ends. The neighborhood of each cell consists of the cell itself and two cells to the left and right of it, yielding a neighborhood consisting of five adjacent cells. We will refer to neighborhoods as windows. The rules for neighborhood updating are the target of evolution. They are represented as arrays of 36 cell
Definition 1: **Apoptotic fitness** uses an evaluation width and an evaluation length for a time history of a cellular automata. The time history is run. If any living cells remain at the time equal to the evaluation length then fitness is zero, otherwise it is the number of living cells in the time history. We call a chromosome **apoptotic** if it has a nonzero apoptotic fitness.

**B. Single Parent Crossover**

Single parent techniques are inspired by a science fiction novel [7] about an all-male society in which children had only a father, and the female contribution came from artificial womb technology and ovarian tissue cultures donated when the society was founded. This situation means that the male gene compliment evolves normally but the female gene compliment is frozen. When the single parent variation operator is used, children have only one parent, and crossover is done with a fixed set of unchanging ancestors. These ancestors are analogous to the ovarian tissue cultures. Since our goal is to generalize a specific cellular automata rule, we limit the ancestor set to that rule. Earlier studies that used single parent techniques used it for a parse tree representation [5], [4] where it has a number of benefits including preserving critical operations and terminals and automatically limiting tree size. These studies tested ancestor sets with single and with multiple members.

When single parent crossover is used on a cellular automata rule, a population member undergoes uniform crossover with the ancestor rule. Each loci is independently chosen to either be in the ancestor or left unmodified with 50% probability. This means that the probability of immediately reproducing the ancestor rule is quite small (one in $2^{35} \approx 34,359,738,368$) but, since the ancestor may not use all its loci, the probability of eventually cloning the ancestor is fairly high. For this reason the rate at which single parent crossover (as opposed to normal two-point crossover with another population member) is used forms a critical parameter.

**C. Evolutionary Algorithm Design**

Other than using single parent techniques in the generalization experiments, the evolutionary algorithm in this study is a standard one. The cellular automata updating rules are stored as strings of 36 integers with values in the range 0-7, corresponding to the cell states. This method of encoding updating rules lumps large numbers of neighborhood patterns together because only the sum of a neighborhood is considered. It also forces the resulting automata to be symmetric, if their initial conditions are.

The variation operators used are those detailed earlier. Selection and replacement are accomplished with generational size-four tournament selection. The population is shuffled into groups of four CA-rules. The two more fit rules are copied over the two less fit with ties broken uniformly at random. The copies are subjected to both crossover and mutation. The algorithm uses a population of 100 rules and
Fig. 2. The upper panel shows the traces, over evolution, of the highest fitness in the first ten runs of the experiment with the blue ancestor. The lower panel shows 95% confidence intervals on the mean fitness, together with the maximum fitness, for the first run.

Ancestor Rules

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Blue</td>
<td>0342030503252222410340737577746214513</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Green</td>
<td>036013060355332130261227070253511</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Red</td>
<td>06040204201514021605444525543570256</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yellow</td>
<td>007120150253015070555354720406532363</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 3. Rules for the four cellular automata used as ancestors in this study.

continues for 1000 generations. A set of 390 replicates of the algorithm, without single parent crossover, were run during a survey of high-fitness apoptotic rules that is not the focus of this study. These runs all used an evaluation length and width of 401. This survey generated the standard apoptotic data set (SADS). Thirty examples of the type of automata that were found appear in Figure 4. The four automata shown in Figure 1 were then selected from the SADS as the ancestor for single parent experiments. The rules for these automata are given in Figure 3.

The following experiments were performed.

- Three single parent runs with the yellow rule, identical to those above, except that the evaluation length was increased to 800. These runs used the rules for the blue, green, and red automata shown in Figure 1.

- A set of baseline runs, using the same code as the single parent experiments and with evaluation length 800. These runs are done with the probability of single parent crossover set to zero. They permit quantification of the impact of the single parent techniques on fitness.

The term generalization in this study means creating new automata rules with time histories that have a similar appearance to the rule used as the ancestor, but fill more space. Increasing the evaluation length from 401 to 800 gives the potential for much higher apoptotic fitness because more time is granted before the automata must die.

III. RESULTS AND DISCUSSION

The results of the parameter setting study are given in Table I. These results are partly qualitative, in that the authors computed the figures reported by inspection of the automata. The results are broken down by the evaluation length and rate of single parent crossover. The following were assessed: what number of the results had lower (worse) fitness than the ancestor, how many of the results were simply duplicates of the ancestor, how many distinct results were there. In this case “distinct” is defined as having a different time history. Changing unused loci in the cellular automata rule can yield distinct chromosomes that generate the time history. We chose to identify such neutral variants. Finally a qualitative assessment of the number of rules that incorporated substantial features of the ancestor is tabulated in the Non-yellow column. Figure 9 gives the 20%, evaluation length 800 results which the reader may use to check the evaluation of yellow versus non-yellow results.

The results of the parameter setting study show that 50% application is too high because it yields relatively low novelty among the solutions. Both number of clones for evaluation length 401 and number of distinct automata speak to this point. On the other hand, 5% application permits the algorithm too ignore the ancestral genetic contribution to easily, as shown by both the number of results with fitness inferior to the ancestor and the number of non-yellow results. The 20% application rate was chosen as the happy medium with relatively good novelty, fitness, and generalization that preserves ancestral characteristics.

It is worth noting that, when added evaluation length was used, the system always managed to increase fitness. The
rules passing the non-yellow test were all true generalizations in the following sense: they did not die fast enough to satisfy the apoptotic fitness function for an evaluation length of 401. This means that appearance is preserved in a distinct cellular automata rule, not via some form of simple duplication. Figure 5 shows that using single parent techniques enhanced final fitness with the degree of enhancement strongly influenced by the fitness of the ancestor. The blue ancestor was the most fit. Presumably a very low fitness ancestor might retard fitness but it seems likely that such an ancestor would simple be lost, possibly after jump-starting evolution.

Figure 2 shows how fitness changes over evolution. Large jumps in maximum fitness are typical of evolution attempting to satisfy the apoptotic fitness function. The most fit results of the evaluation length 800 experiments with 20% single parent crossover for the blue, green, red, and yellow ancestors are shown in Figures 6-9. While appearance varies considerably these figures make it clear that generalizations of the ancestral rule, rather than rules with novel appearances, are the most common result. The blue rule seemed, subjectively, to have the highest rate of generalization while the red rule had the lowest rate of generalization. It may be that the red rule, which blends two colors to achieve its appearance, simply has more chromatic variation among its generalizations.

IV. CONCLUSIONS AND NEXT STEPS

This study demonstrates that single parent techniques are able to generalize specific apoptotic cellular automata rules. Since fitness evaluation is directly proportional to the size of the arena in which the cellular automata is evaluated, the techniques here permit an artist or designer to invent his
appearances at a relatively small size and then use single parent techniques to generalize to larger sizes on demand. The original automata rules could be supplied in a catalog or generated by the designer. Since the coloring rules used in this study were chosen to maximize contrast, a design tool that uses generalization should also include rendering tools. An artist could re-render automata in a catalog until a desired appearance was achieved and then use the generalizer to produce larger examples of automata with the desired appearance.

The demonstration of generalization in this study using single parent techniques is clear, visual, and limited to apoptotic cellular automata. Other techniques for incorporating stored genetic information, such as a Hall of Fame [11], have demonstrated a similar ability to aid generalization. The earlier work on single parent techniques [5] demonstrated very simple generalization from $n-1$ variable parity to $n$ variable parity. A clear area for future work is to check the reach, across problems and representations, of single parent techniques for generalization or improvement of performance.

A. CA Rules as Hypercompressed Pictures

Examine the rules for the ancestors given in Figure 3. If we ignore the obligatory initial zero then these rules are $35 \times 3 = 105$ bits or 14 bytes long. They unpack, via the rendering algorithm, into complex pictures. Their apoptotic character means that these pictures are self-delimiting. The updating algorithm draws the picture and stops at a boundary determined when the rule is selected. Such pictures are potentially useful for applications like backgrounds in games or phone apps, especially as thousands of color pallets could be used to render the automata. While the techniques developed here are demonstrated on one-dimensional automata there is no reason that they could not be used for higher dimensional automata. If particular sorts of shapes are required, then some states of the automata can be assigned to the background color.

The results in this study show that it is easy to modify cellular automata rules via single parent techniques. This was attempted with one the rules evolved in [3] and it failed. The current rules use eight states, instead of six, and a window.
size of five instead of three. Both these changes increase the space of rules combinatorially; apparently they also make the automata far more robust to mutation and crossover. Increasing the number of cell states or window size farther will increase the rule space, it is a topic for future research to determine if they will again increase the effectiveness of single parent techniques in generalizing rules. If it does it will increase the flexibility of the system to find extremely lean-data representations for particular sorts of images.

B. The Single Parent Fitness Landscape

The results showing that cloning is common in the experiments with evaluation length 401 and 50% application of single parent crossover demonstrate that the use of single parent techniques has the ability to strongly limit search to a small part of the fitness landscape. Leaving the definition of distance vague, this restricted search space is “near” the ancestor. Smaller spaces can be searched faster and, in the past, single parent techniques have been shown to enhance performance (rapid location of high fitness individuals). The point at which this becomes interesting is that the single parent techniques can limit search to a small portion of a space not previously searched. This is an alternate point of view on the meaning of generalization. The much longer automata located when the evaluation length was increased share the appearance of the ancestor and so, in some sense, are “near” it. In another sense they are “far” from the ancestor: they would all have zero fitness if evaluated with the same fitness function used to produce the ancestor.

This study, because of space limitations, examined only single parent experiments using a single ancestor. Multiple ancestor sets could easily be used to broaden the search and, potentially, to hybridize the ancestors. This leads to another interesting viewpoint. In his book *Fractals Everywhere* [6] the author introduces a class of fractals generated by contraction maps. One of the simplest contraction maps available is to average toward a point. An example of such a fractal, constructed by the authors, appears in Figure 10. The uniform crossover that is the basis of single parent crossover is a map that stochastically averages a chromosome toward the ancestor’s chromosome in the Hamming metric on the genes. This means that multiple-ancestor single parent systems are Barnsley-style contraction map fractals in the space of automata-rule chromosomes.

The generalization demonstrated in this study is from a $401 \times 401$ drawing arena to a $401 \times 800$ arena. Figure 11 demonstrates that additional generalization is possible: an
example of generalization of the blue rule to a 401 × 1200 drawing arena. The single parent techniques, in addition to restricting the search space, substantially reduce the time to find a high fitness automata. This pushes back the limit on how large such apoptotic automata can be. While we lack the space to show the generalization, the technique has no problem adapting rules to wider drawing arenas as well.

C. Forced Hybridization

On direction for further research is to restrict the search space even more in a manner that forces novel results. We propose the following mechanism for doing so. Select two cellular automata rules to be blended. Use an evolutionary algorithm with a binary gene the same length as the cellular automata rules. Where a rule has a zero, use the loci from the first selected rule, where it has a one, use the loci from the second selected rule. The fitness of such rules are computed normally except that if the gene, of length \( n \) has \( k < C \) zeros or \( m < C \) ones then the fitness is multiplied by \( \frac{q}{k} \) where \( q \) is the minimum of \( m \) or \( k \). This, in effect, envelopes the fitness away from the two selected rules. The results are thus forced to incorporate structures from both parents if \( C \) is large enough. Having \( C \) near half the rule length is a surprisingly small restriction - the number of rules granted full fitness if \( C = 16 \) for the length 35 rules used in this study is \( 2 \cdot \left( \begin{array}{c} 35 \\ 17 \end{array} \right) = 9,075,135,300 \). This algorithm can be generalized to search the “convex hull” of any number of rules by replacing the binary gene with a \( z \)-ary gene where \( z \) is the number of automata rules from which loci could be selected.

REFERENCES

Fig. 8. Single parent descendants of the Red rule

Fig. 9. Single parent descendants of the Yellow rule