Busy Beaver Genome Project

Vinnie Monaco
Dr. Benjamin CS385
December 16, 2008

Abstract: The busy beaver problem is traditionally attacked with brute-force algorithms, searching that have been pruned through mathematical proofs but still remain enormous. This project takes an artificial intelligence approach to the problem, with an intelligent agent that uses a genetic algorithm to search for Turing Machines which may be busy beaver candidates. Part I is an overview of the project and some of my objectives, part II describes the program, part III mentions some of the limitations I anticipate for the agent, part IV contains some of the results, and part V is a conclusion with some ideas for further work.

Part I: Overview and Objectives

The busy beaver problem was first introduced by Radó in his paper on non-computable functions [Rad62] . It consists of finding a *k*-state Turing Machine ™, with a binary alphabet {0, 1}, such that the function ∑(k) is the maximum number of 1's produced by any halting TM with *k* states. A similar problem that is very closely related, and often used to find ∑(k), is S(k) = the maximum number of shifts performed by any halting TM with k states. The TM must be searched for because both of these are non-computable and grow faster than any computable function [Rad62] . They grow so fast that running a 5 or 6 state busy beaver until completion would be impossible. This creates an additional problem of efficiency, because every TM in consideration must be executed in order to observe its behavior and output. There is no way to compute a TM's tape output by the actions it is defined by.

The traditional approach to the busy beaver problem is by brute force, to test every possible *k*-state machine. This task becomes impossible for 4 and 5 state machines, so tree normalization techniques are used to test only a subset of the machines [MaS90]. There are exactly $(4k + 1)^{2k}$ distinct binary alphabet machines with k states, and $(4k + 1)^{2k} - (4k)^{2k}$ machines when excluding those without halt states. The number grows smaller with each additional requirement, such as excluding machines whose output from the initial state is not 1 and so on. This reduces the search tree considerably, but is not effective for larger values of k. With only a few states, the behavior of a TM can become incredibly complex, and it is impossible to predict how a machine will act by looking at its actions alone.

With all of this in mind, I am attempting to take a different approach to the problem, which may or may not be successful. Looking at it from an artificial intelligence point of view is a completely different angle than that of computability theory, and can help give a new understanding to the complexity of this problem. I will use a genetic algorithm to create, grow, and reproduce populations of Turing Machines, in hope that this approach may yield any results at all. I have not studied computability theory, so I am doing this with only a limited knowledge of the problem and the best way to go about it, but have come up with a rough set of objectives. First, I want to test the theory that some parts of a TM can be considered good attributes of a busy beaver candidate. A well defined fitness test might be able to identify the machines that contain these “good parts” and assign them with a high fitness score, making them more likely for reproduction. I think that an effective fitness test for this kind of problem will determine the entire outcome of the project, and be the most difficult part of the program to get right. The reason for this is that in theory there is no “right,” because there are no set of guidelines or proofs for what makes a TM a busy beaver. Some of the good attributes of a busy beaver may only be observed after running the machine and observing the output, so this is what my fitness test will do. I will compare the results of the genetic search to known busy beavers and determine how effective the search was. I'll look at similarities and differences in the two machines as well at run-time behavior. My second objective is to determine if this is an effective approach to the problem, and whether or not it would be able to compete with the traditional brute force and tree-normalization techniques used. I may use some of the common tree-normalization techniques as a foundation for my fitness test, and this may lead to a similar search path.

My approach to the busy beaver problem I though was unique, but discovered that a genetic algorithm has been used to search for busy beavers in the past, with (what I though were) some impressive results [MPC99]. My genetic algorithm differs though, in machine representation and priorities of the fitness test.

The agent has been designed to accommodate the environment it will be working it. The search space can become very large, but the environment is fully observable and deterministic because and aspect of a machine can be observed, and only the agent's actions affect the environment.

Part II: The program

The genetic algorithm and utilities are all written in Lisp. Lisp seems to be one of the best languages for prototyping and abstraction, so that is why I chose to use it. There are three major parts to the program, as well as a set of general utilities and instantiation functions. The Turing-machine section of code deals explicitly with running the Turing Machines to completion or until a predefined shift limit has been reached. The tape is seemingly infinite in two directions, limited only by the capabilities of the hardware. The representation of each machine is similar to [LiR65], but I do not use the binary number system. Each machine is represented as a number, which is easily converted to a list of digits for easy access to a particular digit. The position of each digit determines its importance and meaning to the machine. The whole serial number can be broken down into blocks of 6, each block containing information for a particular state. The first half of the block contains the actions for reading a 0 and the second half for reading a 1. The first digit in each block is the symbol to write to the tape, the second digit is the direction to move (0 for left, 1 for right), and the third digit is the next state. For example, a 3 state machine might look like this:

```
        State 1             State 2             State 3
  (1 1 2   1 0 3     1 0 1   1 1 2     1 0 2   1 1 0)

Read 0    Read 1    Read 0    Read 1    Read 0    Read 1
Write 1   Write 1   Write 1   Write 1   Write 1   Write 1
Move R    Move L    Move L    Move R    Move L    Move R
Go to 2   Go to 3   Go to 1   Go to 2   Go to 2   Halt
```

With this type of machine representation, the block can easily be interchanged and crossed over with other machines without the possibility of creating a TM that will halt on error. The current state and read symbols are interpreted as the location of the other directions in the machine. As a single number, the machine would just be 112103101112102110, but is more easily manipulated as a list of digits. In [LiR65], each three-digit block is thought of as being on a single card, and these six blocks on each card would make up the action table for the machine. Treating each individual card as a "gene" of the machine allows for easy reproduction between two Turing Machines. With an effective fitness test, the TMs with good genes will be chosen for reproduction and pass the good parts to children populations.

The second major part of the program is the genetic algorithm itself. All of this information is contained in evolution.lisp. It works like most genetic algorithms do: a population is randomly generated, a fitness test is applied to every member of the population, and then machines are chosen based on the probability they have obtained from the fitness test. A random number is used and then compared to a random member of the population. That machine is chosen for reproduction if it scored high enough such that its probability is greater than the random number (which acts as a cutoff point). This way machines with the highest scores are likely to be chosen because their probability will be close to or equal to 1. A mutation is then applied to preferably a small set of machines, although the mutation rate can be adjusted thorough a global parameter. I have defined a mutation as a random change in the third digit of each block of the machine. This way, the state changes of the machine in runtime are mutated, leaving the output and tape head direction digits unchanged. Other mutations can easily be introduced with different mutation functions, but this one seemed sensible enough. There was no other reason why I chose to perform mutations in this way.

The genetic algorithm is recursively performed on the new population once the size has been reached by reinserting the reproduced and mutated machines. This process is continued until the algorithm has completed a defined number of generations. At this point, the machine with the high fitness score is selected as the result of the search.

The genetic algorithm is written so that any fitness test can be passed as an argument and applied to the members of the population. For the most part, the genetic algorithm is not Turing Machine specific, although some parts of it have been modified for this purpose to increase efficiency or the simplicity of the code. I have kept the fitness tests in a separate section of code though, for they are most likely to change. The fitness test is what will either make the entire program either successful or unsuccessful. The test itself can be found in fitness.lisp, along with several helper and formatting functions. It consists of a number of weighted sub-tests that analyze properties of the machine itself and the machine's behavior after runtime. This is where I have used my own judgment on defining a machine as being “closer” to a busy beaver, and I may be wrong. Not all of the tests I have used will be agreed on as good attributes of a busy beaver. Another issue I have encountered is trying to decide which set of properties will be more important for the fitness test: properties of the machine itself (by looking at the machine's number or action table), or properties of the machine's execution. I have not separated the tests, but each one could be put into one of those categories. For example, the number of 1's left on a tape after the execution is complete would be a property of the run time, while counting the number of goto states used in the action table is a property of the machine itself. Assigning weights to each of these tests is another issue that is not very straightforward, and for this reason I have made all of the weights global parameter in the main.lisp file that can be easily changed before or during the search.

I decided that the number of ones left on the tape is an important characteristic for the fitness test to check for, but not more important than others. I have not included any functions or tests to prove non-halting machines, so machines that print 1's in a loop until the shift limit has been reached will be given a higher score than less productive, halting machines. Right now the this test is weighted relatively low, but the weight could be changed to the result of another test, one that checked for the non-halting property. This has been done in [MaS90], in which the authors have classified the infinite loops of 3 and 4-state machines. (Actually there are 6 classes of non-halting 3 and 4-state machines including: simple loop, back-track, Christmas tree, shadow Christmas, counter, and the holdouts which had to be examined by hand. Each class of machines displays unique runtime behaviors when observing the tape output.) Proving non-halting machines might be a good idea reason to write another agent altogether, to perform just this task.

The fitness test also checks for the number of shifts performed (similar problems as measuring the number of 1's output), the length of tape traversed, the maximum length of a block of 1's left on the tape, the property of a successful halt (reaching state 0), the distance of the tape head from the initial position on the tape, and the average length of continuous 1's left on the tape. Properties of the machine I have tested for include the number of states used in the action table (goto states), and whether is contained a halt state or not. These are all weighted, and likely to change as I test the genetic algorithm and usefulness of the fitness function.

Part III: Limitations

This agent is designed only to search for busy beaver candidates. Parts of the code seem quite complex, but I have tried to take an overall design of simplicity while still being as complete as possible. Good Lisp programming is hard (see the section 2.2, “Good Lisp programming is Hard,” in Grabriel's worse is better paper [Gab94]. This is important to keep in mind, especially since this is the first program I have written in Lisp. I am new to the language and its features, but just beginning to realize how powerful it can be in the hands of a good programmer. I've developed a deeper understanding of the language through this project, and it seemed to be a good starting point.

The code is not efficient and probably would not compare to the performance of a C program with the same construct the way it stands now. But, writing this in C would have been tedious and more time consuming than in Lisp (for me). The algorithm, however, will always be limited by its exponential growth, which is language independent. Right now, I do not see a way around running each machine to test it for certain fitness properties, and this must be done for every generation. The size of the population and number of generations evolved do change the scope of the search though. A larger population will lead to a much wider search, while more generations will increase the depth of the search. It is not clear right now which trade-off will benefit my objectives yet, but it is something to keep in mind. Space does not seem to be an issue for the program right now, because it only keeps a fixed number of machines in memory at any time during the search. Running each machine is probably the most time consuming part of the test, especially for running 4 or 5 state machines to completion. With a relatively low shift limit, some of the characteristics of runtime should be able to be obtained while conserving time, but there is no guarantee that a busy beaver in population will be scored highly.

I am not overly optimistic about the performance or results of the program, although it will be interesting to see what is found.

Part IV: Results of the agent

The sub-tests for the fitness test used include:

sub-test-a : number of shifts performed
sub-test-b : number of 1's left on the tape
sub-test-c : length of tape traversed
sub-test-d : maximum length of continuous 1's left on the tape
sub-test-e : 1 for halting successfully, 0 otherwise
sub-test-f : distance of tape head from starting position
sub-test-g : average length of non-blank symbols
sub-test-h : number of states used
sub-test-i : 1 for containing a halt state, 0 otherwise

I have performed several searches with different parameters and the following pages contain some of the results I have obtained and think are most interesting. The condensed tape represents a list of the size of each block of 1's. For example a TM which leaves the tape in this state:

```
(0 0 1 0 1 1 1 1 0 1 1 0 0 0 1 0 0 1 1)
```

would have a condensed tape that looks like:

```
(1 4 2 1 2).
```

Fitness test progress of a of the successive generations is included in one of the smaller searches to see how the population progressed through natural selection and whether or not the fitness test was effective. The results are plotted in a scatter plot and

```
******** Search completed in 2.787 secs ********

Population size  : 200
Generations      : 20
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/50
  Weight b   : 1/20
  Weight c   : 1/50
  Weight d   : 1/30
  Weight e   : 1200
  Weight f   : 1/50
  Weight g   : 1/30
  Weight h   : 1
  Weight i   : 700

**************************************************


****************** best fit ********************
Machine #: (1 0 2  0 1 3   1 0 3  1 0 1   1 0 0  1 0 3)

States: 3

State   Read    Write   Move    Next-State
1       0       1        0       2
1       1       0        1       3
2       0       1        0       3
2       1       1        0       1
3       0       1        0       0
3       1       1        0       3

Fitness Score       : 1904.39
Halted              : T
Sigma               : 3
Shifts              : 3
Last State          : 0
Last Position       : -3
Length Traversed    : 3
Avg Block Size      : 3.0
Max Block Size      : 3
Condensed Tape: (3)
```

After adjusting the weights to what seemed appropriate, I started coming up with results for the 3-state machines similar to this one. This is a simple machine that uses all three states, but with no feedback (changes made to the tape do not cause any perturbations or affect the behavior of the machine. It could have printed three 0's or one 0 and two ones, and still halt in 3 shifts after reaching state 3). It moves only in one direction, successively reading a 0 and printing a 1. It met the naïve lower bound for a 3-state busy-beaver: a number of 1's printed equal to the number of states of the machine. Any *k* state machine like this can be built to leave exactly *k* ones on the tape before halting successfully. This is one of the simplest lower bounds that can be put on a busy beaver candidate.

```
******** Search completed in 2.95 secs ********

Population size  : 200
Generations      : 20
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/50
  Weight b   : 1/20
  Weight c   : 1/50
  Weight d   : 1/30
  Weight e   : 1200
  Weight f   : 1/50
  Weight g   : 1/30
  Weight h   : 1
  Weight i   : 700

**************************************************


******************* best fit ********************
Machine #: (1 0 2  1 1 2   1 1 1  1 0 3   0 1 2  1 0 0)

States: 3

State   Read    Write   Move    Next-State
1       0       1       0       2
1       1       1       1       2
2       0       1       1       1
2       1       1       0       3
3       0       0       1       2
3       1       1       0       0

Fitness Score       : 1904.6466
Halted              : T
Sigma               : 4
Shifts              : 7
Last State          : 0
Last Position       : -1
Length Traversed    : 4
Avg Block Size      : 4.0
Max Block Size      : 4
Condensed Tape: (4)
```

This is an improvement over the last search, with all of the same search parameter. The search is focused on breadth as opposed to depth with a population of 200 and 20 generations. The machine found here exhibits much more complex behavior though, halting after 7 steps and leaving a block of 4 ones on the tape. This is quite productive for a 3-state machine, and comes close to the known 3-state busy beaver, which prints 6 ones and halts in 14 steps. For halting machines, once the number of steps performed has exceeded the number of states of that machine, it is obvious that some previous change to the tape has affected the machines future behavior. It is not clear just from looking at the specifications of this machine whether it enters the third state or not.

```
******** Search completed in 35.945 secs ********

Population size  : 500
Generations      : 100
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 500
  Weight f   : 1/5
  Weight g   : 1/3

**************************************************

****************** best fit ********************
Machine #: (1 1 2  1 0 2   1 0 1  1 0 0   1 1 2  1 1 0)

States: 3

State   Read    Write   Move    Next-State
1       0       1       1       2
1       1       1       0       2
2       0       1       0       1
2       1       1       0       0
3       0       1       1       2
3       1       1       1       0

Fitness Score       : 1306.0667
Halted              : T
Sigma               : 4
Shifts              : 6
Last State          : 0
Last Position       : -2
Length Traversed    : 4
Avg Block Size      : 4.0
Max Block Size      : 4
Condensed Tape: (4)
```

The machine produced in this search has been another productive 3-state machine so far, halting in 6 steps and leaving a block of four 1's on the tape. It moves in two directions, which is important for productivity. It contains the same first block (1 1 2) as the known busy beaver:

```
(1 1 2  1 0 3   1 0 1  1 1 2   1 0 2  1 1 0)
```

as well as the same third and sixth blocks. This machine is interesting because it only uses 2 of the available 3 states. Since this machine does not use the third state, that part of the action table can be eliminated, which results in the 2-state busy beaver:

```
(1 1 2  1 0 2   1 0 1  1 0 0)
```

This is significant because machines that exhibit productive behavior but do not enter every state can be mapped to lesser-state machines. The productivity remains the same, while the number of states is decreased, yielding a machine with comes closer to satisfying $\sum(k)$. Note the different weights used.

```
******** Search completed in 76.374 secs ********

Population size  : 200
Generations      : 500
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 500
  Weight f   : 1/5
  Weight g   : 1/3
  Weight h   : 100
  Weight i   : 200

**************************************************


****************** best fit ********************
Machine #: (1 0 2  1 0 1   1 0 3  1 0 0   1 0 0  1 0 0)

States: 3

State   Read    Write   Move    Next-State
1       0       1        0       2
1       1       1        0       1
2       0       1        0       3
2       1       1        0       0
3       0       1        0       0
3       1       1        0       0

Fitness Score       : 1103.9
Halted              : T
Sigma               : 3
Shifts              : 3
Last State          : 0
Last Position       : -3
Length Traversed    : 3
Avg Block Size      : 3.0
Max Block Size      : 3
Condensed Tape: (3)
```

This search concentrated on depth, rather than breadth, and resulted in a simple TM, halting successfully after 3 shifts. There is no feedback, as the machine only moves in one direction. The weights are different than some of the earlier searches, with more concentration on the tape output. (see list of weights at the beginning of this section). I consider a halting machine like this more successful, or "closer," to a busy beaver than an infinitely looping machine, but I may be wrong.

```
******** Search completed in 37.673 secs ********

Population size  : 500
Generations      : 100
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 500
  Weight f   : 1/5
  Weight g   : 1/3
  Weight h   : 100
  Weight i   : 200

**************************************************


****************** best fit ********************
Machine #: (0 1 2  0 1 0   1 1 3  0 1 1   1 1 0  1 1 1)

States: 3

State   Read    Write   Move    Next-State
1       0       0       1       2
1       1       0       1       0
2       0       1       1       3
2       1       0       1       1
3       0       1       1       0
3       1       1       1       1

Fitness Score       : 1104.1333
Halted              : T
Sigma               : 2
Shifts              : 3
Last State          : 0
Last Position       : 3
Length Traversed    : 4
Avg Block Size      : 2.0
Max Block Size      : 2
Condensed Tape: (2)
```

This search used the same parameters as the last search, and resulted in a machine very closely related to the last one found. It halts in three shifts and moves in only one direction, but prints a 0 at one of the states instead of a 1. Since the initial population is randomly generated, the outcome of identical searches will differ slightly, but remain consistent for the most part, which is good.

```
******** Search completed in 100.513 secs ********

Population size  : 1500
Generations      : 100
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/20
  Weight b   : 1/20
  Weight c   : 1/15
  Weight d   : 1/10
  Weight e   : 550
  Weight f   : 1/5
  Weight g   : 1/10
  Weight h   : 100
  Weight i   : 200

**************************************************


****************** best fit ********************
Machine #: (1 0 2  1 1 2   1 1 1  1 1 0   1 1 1  1 1 0)

States: 3

State   Read    Write   Move    Next-State
1       0       1        0       2
1       1       1        1       2
2       0       1        1       1
2       1       1        1       0
3       0       1        1       1
3       1       1        1       0

Fitness Score       : 2191.7666
Halted              : T
Sigma               : 4
Shifts              : 6
Last State          : 0
Last Position       : 2
Length Traversed    : 4
Avg Block Size      : 4.0
Max Block Size      : 4
Condensed Tape: (4)
```

This was a very broad search and was able to discover another 3-state machine with the known 2-state busy beaver embedded within. It does not use the third state, and halts in 6 shifts.

```
******** Search completed in 33.367 secs ********

Population size  : 500
Generations      : 100
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 500
  Weight f   : 1/5
  Weight g   : 1/3
  Weight h   : 100
  Weight i   : 200

**************************************************


******************* best fit *********************
Machine #: (1 0 2  1 1 2   1 1 1  1 0 0   1 0 0  1 0 1)

States: 3

State   Read    Write   Move    Next-State
1       0       1        0       2
1       1       1        1       2
2       0       1        1       1
2       1       1        0       0
3       0       1        0       0
3       1       1        0       1

Fitness Score       : 1006.4667
Halted              : T
Sigma               : 4
Shifts              : 6
Last State          : 0
Last Position       : 0
Length Traversed    : 4
Avg Block Size      : 4.0
Max Block Size      : 4
Condensed Tape: (4)
```

Narrower than the last search, but still manages to find another, different 3-state machine with a 2-state busy beaver embedded within it. The first two states of this machine, representing a known 2-state busy beaver, behave differently than the last one found, but still halt in 6 shifts and print 4 ones on the tape.

```
******** Search completed in 33.714 secs ********

Population size  : 500
Generations      : 100
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/15
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 100
  Weight i   : 200

**************************************************


******************* best fit ********************
Machine #: (1 1 2  1 0 2   1 0 1  1 1 0   0 0 2  1 1 1)

States: 3

State   Read    Write   Move    Next-State
1       0       1       1       2
1       1       1       0       2
2       0       1       0       1
2       1       1       1       0
3       0       0       0       2
3       1       1       1       1

Fitness Score       : 2001.6333
Halted              : T
Sigma               : 4
Shifts              : 6
Last State          : 0
Last Position       : 0
Length Traversed    : 4
Avg Block Size      : 4.0
Max Block Size      : 4
Condensed Tape: (4)
```

With the same parameters as the last search, another different 2-state busy beaver is found within this machine. The third state is not used, and may have been filtered out of the population through each generation. The search may have reached a local maxima by finding and stopping at this high-scoring 3-state machine, of which the first two states alone are a busy beaver.

```
******** Search completed in 386.117 secs ********

Population size  : 500
Generations      : 1000
Mutation Rate    : 1/5
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 500
  Weight f   : 1/5
  Weight g   : 1/3
  Weight h   : 100
  Weight i   : 200

**************************************************


******************* best fit ********************
Machine #: (1 0 3  1 0 3   1 0 0  1 0 1   1 0 2  1 0 3)

States: 3

State   Read    Write   Move    Next-State
1       0       1        0       3
1       1       1        0       3
2       0       1        0       0
2       1       1        0       1
3       0       1        0       2
3       1       1        0       3

Fitness Score       : 1103.9
Halted              : T
Sigma               : 3
Shifts              : 3
Last State          : 0
Last Position       : -3
Length Traversed    : 3
Avg Block Size      : 3.0
Max Block Size      : 3
Condensed Tape: (3)
```

Another broad and very deep search, which resulted just in a simple machine. Depending on the initial population, a local maxima may have been reached, with not enough mutation to break free from it. This resulted in population with probably very similar genes (after 1000 generations of the same 500 machines) and only a simple.

```
******** Search completed in 3.104 secs ********

Population size  : 200
Generations      : 20
Mutation Rate    : 1/5
Machine States   : 2
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/50
  Weight b   : 1/20
  Weight c   : 1/50
  Weight d   : 1/30
  Weight e   : 1200
  Weight f   : 1/50
  Weight g   : 1/30
  Weight h   : 1
  Weight i   : 700

**************************************************


******************* best fit ********************
Machine #: (1 0 2  1 0 1   1 0 0  0 1 0)

States: 2

State   Read    Write   Move    Next-State
1       0       1        0       2
1       1       1        0       1
2       0       1        0       0
2       1       0        1       0

Fitness Score       : 1903.2533
Halted              : T
Sigma               : 2
Shifts              : 2
Last State          : 0
Last Position       : -2
Length Traversed    : 2
Avg Block Size      : 2.0
Max Block Size      : 2
Condensed Tape: (2)
```

This was a search for a 2-state busy beaver, but failed to find one. The fitness test may not be effective enough for a machine with only 2 states, even though the agent was able to fine several 3-state machines with 2-state busy beavers embedded within. This is a simple machine with no feedback, halting in 2 shifts and printing 2 ones on the tape, moving only to the left. The same first block, (1 0 2) is seen in more productive machines found though.

```
******** Search completed in 2.736 secs ********

Population size  : 200
Generations      : 20
Mutation Rate    : 1/5
Machine States   : 2
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/50
  Weight b   : 1/20
  Weight c   : 1/50
  Weight d   : 1/30
  Weight e   : 1200
  Weight f   : 1/50
  Weight g   : 1/30
  Weight h   : 1
  Weight i   : 700

**************************************************


******************* best fit *********************
Machine #: (1 1 2  0 0 2   1 0 1  1 1 0)

States: 2

State   Read    Write   Move    Next-State
1       0       1       1       2
1       1       0       0       2
2       0       1       0       1
2       1       1       1       0

Fitness Score       : 1903.4467
Halted              : T
Sigma               : 3
Shifts              : 6
Last State          : 0
Last Position       : 0
Length Traversed    : 4
Avg Block Size      : 1.5
Max Block Size      : 2
Condensed Tape: (2 1)
```

This is another search for a 2-state machine, but still fails to find a 2-state busy beaver. The machine found here could have been a busy beaver, because it exhibits complex behavior, but fails to print a continuous block of 4 ones on the tape. The action that would have made this machine output 4 ones may not have been available in the population, or stuck in a low-scoring machine not likely to be chosen for reproduction.

```
******** Search completed in 64.325 secs ********

Population size  : 200
Generations      : 100
Mutation Rate    : 1/5
Machine States   : 4
Shift Limit      : 500
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/12
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 10
  Weight i   : 200

**************************************************


****************** best fit ********************
Machine #: (1 1 2  1 0 2   1 0 1  1 1 0   1 0 4  1 1 3   1 0 1  1 0 4)

States: 4

State   Read    Write   Move    Next-State
1       0       1       1       2
1       1       1       0       2
2       0       1       0       1
2       1       1       1       0
3       0       1       0       4
3       1       1       1       3
4       0       1       0       1
4       1       1       0       4

Fitness Score       : 751.7
Halted              : T
Sigma               : 4
Shifts              : 6
Last State          : 0
Last Position       : 0
Length Traversed    : 4
Avg Block Size      : 4.0
Max Block Size      : 4
Condensed Tape: (4)
```

This was a search for a 4-state machine, but was able to find another 2-state busy beaver within the first two states of the machine found. States 3 and 4 are not used in this machine.

```
******** Search completed in 137.584 secs ********

Population size  : 800
Generations      : 50
Mutation Rate    : 1/5
Machine States   : 4
Shift Limit      : 500
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/12
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 10
  Weight i   : 200

*************************************************


****************** best fit ********************
Machine #: (1 1 4  1 1 3   0 1 0  0 1 0   0 0 3  1 1 0   1 1 2  0 0 1)

States: 4

State   Read    Write   Move    Next-State
1       0       1       1       4
1       1       1       1       3
2       0       0       1       0
2       1       0       1       0
3       0       0       0       3
3       1       1       1       0
4       0       1       1       2
4       1       0       0       1

Fitness Score       : 751.3
Halted              : T
Sigma               : 2
Shifts              : 3
Last State          : 0
Last Position       : 3
Length Traversed    : 4
Avg Block Size      : 2.0
Max Block Size      : 2
Condensed Tape: (2)
```

This was a broader search for a 4-state busy beaver, but was less successful than the previous search with a smaller population and more generations. Only a simple, successfully halting machine was found, which left 2 ones on the tape and did not exhibit any complex behavior.

```
******** Search completed in 30.051 secs ********

Population size  : 200
Generations      : 50
Mutation Rate    : 1/10
Machine States   : 5
Shift Limit      : 500
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/12
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 10
  Weight i   : 200

**************************************************


****************** best fit ********************
Machine #:
(1 0 3  1 0 4   1 1 0  1 0 2   1 1 3  1 1 1   1 0 0  1 0 4   1 1 5  1 1 4)

States: 5

State   Read    Write   Move    Next-State
1       0       1        0       3
1       1       1        0       4
2       0       1        1       0
2       1       1        0       2
3       0       1        1       3
3       1       1        1       1
4       0       1        0       0
4       1       1        0       4
5       0       1        1       5
5       1       1        1       4

Fitness Score       : 761.7
Halted              : T
Sigma               : 4
Shifts              : 9
Last State          : 0
Last Position       : -3
Length Traversed    : 5
Avg Block Size      : 4.0
Max Block Size      : 4
Condensed Tape: (4)
```

This was a search with an average sized population of 200, but only 50 generations. Decreasing the depth of the search may or may not have helped it produce the most complex machine so far. It is clear that the machine only used 4 of the 5 states available, and successfully halts at state 4. The changes made to the tape affect the behavior of the machine, as is does not halt until 9 shifts have been completed. It's score is consistent, yet slightly higher, than other searches for 4 state machines. This machine number could easily be truncated to 4 states, yielding a good candidate (compared to the results so far) for a 4-state busy beaver.

```
******** Search completed in 55.73 secs ********

Population size  : 100
Generations      : 200
Mutation Rate    : 1/10
Machine States   : 5
Shift Limit      : 500
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/12
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 10
  Weight i   : 200

**************************************************


****************** best fit ********************
Machine #:
(1 0 2  1 1 5   1 1 4  1 0 1   1 0 5  1 0 0   1 1 3  1 1 2   1 0 5  1 1 0)

States: 5

State   Read    Write   Move    Next-State
1       0       1        0       2
1       1       1        1       5
2       0       1        1       4
2       1       1        0       1
3       0       1        0       5
3       1       1        0       0
4       0       1        1       3
4       1       1        1       2
5       0       1        0       5
5       1       1        1       0

Fitness Score       : 762.4167
Halted              : T
Sigma               : 5
Shifts              : 7
Last State          : 0
Last Position       : 3
Length Traversed    : 5
Avg Block Size      : 5.0
Max Block Size      : 5
Condensed Tape: (5)
```

With the same parameters as the last search, another productive and complex machine has been found. It is not clear from looking at the action table whether all of the states are used, but it does leave the highest number of ones on the tape so far. One property that seems to be consistent with complex machines like this is an balance of moves to the left and right in the action table. I am not sure whether this machine halts after state 5 is reached or state 4, both containing halt states. The fitness score of this machine is also consistent with other high scoring and productive 4-5 state machines.

```
******** Search completed in 59.241 secs ********

Population size  : 200
Generations      : 50
Mutation Rate    : 1/10
Machine States   : 6
Shift Limit      : 800
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/12
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 10
  Weight i   : 200

**************************************************


******************* best fit ********************
Machine #:
(0 1 3  1 1 6   0 0 5  0 0 2   0 0 4  0 1 1   1 1 2  0 0 5   0 0 5  0 0 0   0 0 3 0
0 2)

States: 6

State   Read    Write   Move    Next-State
1       0       0       1       3
1       1       1       1       6
2       0       0       0       5
2       1       0       0       2
3       0       0       0       4
3       1       0       1       1
4       0       1       1       2
4       1       0       0       5
5       0       0       0       5
5       1       0       0       0
6       0       0       0       3
6       1       0       0       2

Fitness Score       : 770.43335
Halted              : T
Sigma               : 0
Shifts              : 5
Last State          : 0
Last Position       : -1
Length Traversed    : 3
Avg Block Size      : 0.0
Max Block Size      : 0
Condensed Tape: (0)
```

This 6-state machine found leaves the tape unchanged after successfully halting, and may have a complex behavior. It is not clear which states are reached or if it prints any 1's and then changes them back to 0's. Only 5 of the 6 states could have been reached because it halts in 5 shifts.

```
******** Search completed in 118.519 secs ********

Population size  : 75
Generations      : 300
Mutation Rate    : 1/10
Machine States   : 6
Shift Limit      : 800
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/12
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 10
  Weight i   : 200

**************************************************


******************* best fit ********************
Machine #:
(1 1 4  1 1 6   1 1 2  1 1 0   1 1 0  1 1 4   1 1 3  1 1 1   1 1 4  1 1 4   1 1 5 1
1 5)

States: 6

State   Read    Write   Move    Next-State
1       0       1       1       4
1       1       1       1       6
2       0       1       1       2
2       1       1       1       0
3       0       1       1       0
3       1       1       1       4
4       0       1       1       3
4       1       1       1       1
5       0       1       1       4
5       1       1       1       4
6       0       1       1       5
6       1       1       1       5

Fitness Score       : 771.55
Halted              : T
Sigma               : 3
Shifts              : 3
Last State          : 0
Last Position       : 3
Length Traversed    : 4
Avg Block Size      : 3.0
Max Block Size      : 3
Condensed Tape: (3)
```

With a small population and a longer search, another simple machine has been found. It moves in only one direction, so is able only to read 0's from the tape and possible print 1's. This machine halts successfully after 3 shifts though, not reaching every state. The action (1 1 4) appears 4 times in the machine number.

```
******** Search completed in 4.361 secs ********

Population size  : 75
Generations      : 10
Mutation Rate    : 1/10
Machine States   : 6
Shift Limit      : 800
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/12
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 10
  Weight i   : 200

**************************************************


******************* best fit ********************
Machine #:
(0 1 6  0 1 2   1 0 4  1 1 5   0 1 4  0 1 5   1 1 5  0 1 1   0 0 0  1 1 4   1 1 3 0
0 4)

States: 6

State   Read    Write   Move    Next-State
1       0       0       1       6
1       1       0       1       2
2       0       1       0       4
2       1       1       1       5
3       0       0       1       4
3       1       0       1       5
4       0       1       1       5
4       1       0       1       1
5       0       0       0       0
5       1       1       1       4
6       0       1       1       3
6       1       0       0       4

Fitness Score       : 771.36664
Halted              : T
Sigma               : 2
Shifts              : 5
Last State          : 0
Last Position       : 3
Length Traversed    : 5
Avg Block Size      : 1.0
Max Block Size      : 1
Condensed Tape: (1 1)
```

Similar behavior to other 6-state machine found. The 6-state machines the agent finds seem to be less productive and complex than 4 and 5 state machines. This one does leave two 1's on the tape though, but probably does not have any type of complex behavior (because the length traversed is 5 and it halts in 5 shifts).

```
******** Search completed in 14.644 secs ********

Population size  : 200
Generations      : 20
Mutation Rate    : 1/10
Machine States   : 10
Shift Limit      : 500
Fitness weights:
  Weight a   : 1/10
  Weight b   : 1/12
  Weight c   : 1/10
  Weight d   : 1/5
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 10
  Weight i   : 200

**************************************************

****************** best fit ********************
Machine #:
(1 1 4 0 0 2 0 1 10 1 0 1 0 1 0 0 1 5 0 0 3 1 1 5 0 0 6 0 0 8 1 1 3 0 0 3 0 1 3
 1 0 6 0 1 3 0 0 7 0 0 9 1 1 6 0 0 0 0 1 1)

States: 10

State   Read    Write   Move    Next-State
1       0       1       1       4
1       1       0       0       2
2       0       0       1       10
2       1       1       0       1
3       0       0       1       0
3       1       0       1       5
4       0       0       0       3
4       1       1       1       5
5       0       0       0       6
5       1       0       0       8
6       0       1       1       3
6       1       0       0       3
7       0       0       1       3
7       1       1       0       6
8       0       0       1       3
8       1       0       0       7
9       0       0       0       9
9       1       1       1       6
10      0       0       0       0
10      1       0       1       1

Fitness Score       : 811.38336
Halted              : T
Sigma               : 1
Shifts              : 6
Last State          : 0
Last Position       : 2
Length Traversed    : 3
Avg Block Size      : 1.0
Max Block Size      : 1
Condensed Tape: (1)
```

This search for a 10-state busy beaver resulted in what looked like only a simple machine.

```
******** Search completed in 6.303 secs ********

Population size  : 150
Generations      : 5
Mutation Rate    : 1/5
Machine States   : 10
Shift Limit      : 1000
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 1000
  Weight f   : 1/5
  Weight g   : 1/3
  Weight h   : 100
  Weight i   : 800
**************************************************
****************** best fit ********************
Machine #:
(1 1 8 0 0 1 0 1 7 1 1 8 0 1 8 1 0 6 1 0 9 0 0 1 1 1 8 0 0 5 1 0 6 1 0 9 0 0 2
 0 1 7 1 1 5 0 0 5 0 0 0 0 1 7 1 0 3 0 0 4)

States: 10

State   Read    Write   Move    Next-State
1       0       1       1       8
1       1       0       0       1
2       0       0       1       7
2       1       1       1       8
3       0       0       1       8
3       1       1       0       6
4       0       1       0       9
4       1       0       0       1
5       0       1       1       8
5       1       0       0       5
6       0       1       0       6
6       1       1       0       9
7       0       0       0       2
7       1       0       1       7
8       0       1       1       5
8       1       0       0       5
9       0       0       0       0
9       1       0       1       7
10      0       1       0       3
10      1       0       0       4

Fitness Score       : 3566.6667
Halted              : NIL
Sigma               : 1000
Shifts              : 1000
Last State          : 5
Last Position       : 1000
Length Traversed    : 1001
Avg Block Size      : 1000.0
Max Block Size      : 1000
Condensed Tape: (1000)
```

The machine found in this search is an infinite loop (alternating between states), but was found in only 5 generations. The fitness test may be less effective in finding successfully halting machines the way it is weighted right now.

```
******** Search completed in 6.359 secs ********

Population size  : 150
Generations      : 5
Mutation Rate    : 1/5
Machine States   : 20
Shift Limit      : 1000
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 1200
  Weight f   : 1/5
  Weight g   : 1/3
  Weight h   : 100
  Weight i   : 1000

**************************************************


******************* best fit ********************
Machine #:
(0 0 13 1 0 12 0 0 4 1 1 17 0 1 12 0 0 9 0 1 1 0 1 12 0 0 19 1 0 10 1 0 6 1 0 0
 0 0 4 0 0 12 1 1 8 1 1 12 1 1 4 1 0 5 0 0 1 0 0 20 0 1 9 1 0 10 0 1 8 0 1 14 0
 0 3 1 1 11 1 1 16 0 1 17 0 0 16 1 1 3 0 0 2 0 0 6 1 0 0 0 0 0 0 0 11 1 1 9 1 0
 8 1 0 18 0 0 8 0 1 15)

States: 20

(action table omitted for space)

Fitness Score       : 4760.8
Halted              : NIL
Sigma               : 996
Shifts              : 1000
Last State          : 8
Last Position       : 996
Length Traversed    : 999
Avg Block Size      : 996.0
Max Block Size      : 996
Condensed Tape: (996)
```

Another search for a 20-state machine in 5 generations resulted in a non-halting machine. It is an infinitely looping machine. Changing the weights of the fitness test or the shift limit may be able to avoid the reproduction of machines like this. A high shift limit lets machines in an infinite loop score higher in the fitness test because it tests for the number of shifts and 1's left on the tape. My objective is only to test the initial behavior of the machine for busy beaver-like characteristics.

```
******** Search completed in 7.604 secs ********

Population size  : 150
Generations      : 5
Mutation Rate    : 1/5
Machine States   : 50
Shift Limit      : 1000
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 1200
  Weight f   : 1/5
  Weight g   : 1/3
  Weight h   : 100
  Weight i   : 1000

**************************************************


******************* best fit *********************
Machine #:
(0 0 24 1 1 45 0 0 35 1 1 2 0 0 46 1 0 22 1 1 35 0 1 48 0 1 17 1 0 35 0 1 22 0
 1 19 0 1 27 1 0 20 0 1 13 1 0 0 1 1 19 1 0 18 0 0 21 0 0 35 1 1 6 1 0 35 1 1
 36 0 0 22 0 0 19 0 0 46 0 0 9 0 1 11 1 1 13 1 0 36 1 1 7 0 0 1 1 1 40 1 0 47 1
 1 17 1 0 2 1 0 40 0 1 29 0 1 39 1 0 41 0 1 27 1 0 27 1 1 31 1 1 19 0 1 47 1 1
 7 0 0 31 0 0 12 0 1 34 1 1 15 0 0 41 0 0 44 1 0 43 1 0 17 0 0 17 1 0 30 1 0 11
 0 0 12 0 1 5 1 0 36 1 1 31 0 0 15 1 1 41 0 1 23 0 0 5 1 0 39 1 0 16 0 0 38 0 0
 34 0 0 14 0 0 13 0 0 19 1 1 34 1 0 24 1 1 45 0 0 43 0 0 32 1 0 8 1 0 37 0 0 28
 1 0 7 1 0 18 0 0 49 0 1 44 1 0 25 1 1 9 1 1 0 1 1 19 0 1 34 1 1 3 1 1 46 0 0 8
 0 1 45 0 0 34 1 0 37 0 0 31 1 1 42 0 0 31 1 1 22 0 0 46)

States: 50

(action table omitted for space)

Fitness Score       : 7363.1333
Halted              : NIL
Sigma               : 998
Shifts              : 1000
Last State          : 31
Last Position       : 996
Length Traversed    : 999
Avg Block Size      : 998.0
Max Block Size      : 998
Condensed Tape: (998)
```

This test for a 50-state machine is similar to the results for 10 and 20-state machines. The agent gets stuck in a local maxima of infinitely looping machines, which are highly productive, but (I think) not close to being a successfully halting busy beaver.

```
******** Search completed in 14.704 secs ********

Population size  : 150
Generations      : 10
Mutation Rate    : 1/5
Machine States   : 100
Shift Limit      : 1000
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/2
  Weight c   : 1/5
  Weight d   : 1/3
  Weight e   : 1200
  Weight f   : 1/5
  Weight g   : 1/3
  Weight h   : 100
  Weight i   : 1000

**************************************************
****************** best fit ********************
Machine #:
(1 0 36 1 0 31 1 0 8 0 0 41 1 0 60 0 1 31 1 1 28 1 1 29 0 1 98 1 1 31 1 1 99 0
 0 69 0 1 72 1 1 69 1 0 8 1 1 58 0 0 57 1 1 43 0 0 84 0 1 63 0 0 48 0 1 30 1 1
 19 0 0 95 1 0 4 0 1 70 1 1 14 0 1 19 0 0 43 0 1 95 1 0 24 0 0 89 1 0 30 1 0 53
 1 1 98 0 0 97 0 0 5 1 1 55 1 1 50 1 1 26 0 0 55 0 0 44 0 0 42 1 0 18 1 1 64 1
 0 38 0 0 22 1 1 47 0 0 4 1 0 22 1 0 65 1 0 31 0 1 23 0 0 58 0 1 97 1 0 59 0 0
 18 0 1 65 1 1 50 0 0 54 1 0 72 0 1 75 1 0 97 1 1 80 1 1 55 1 1 48 1 0 4 0 1 54
 0 0 2 1 1 3 0 1 37 1 0 46 1 1 33 0 1 85 1 0 4 1 0 37 1 0 67 1 0 65 0 0 98 1 0
 60 0 1 11 1 1 56 1 1 79 0 1 94 1 1 78 1 1 70 0 0 53 0 1 74 1 1 85 1 0 73 1 1
 50 0 0 47 1 1 14 0 0 63 0 1 73 0 0 27 0 1 46 1 1 82 0 0 92 1 0 96 0 1 96 1 1
 80 0 1 97 0 0 0 0 1 70 1 0 40 0 0 3 1 0 60 1 0 7 1 1 22 0 0 17 0 0 3 1 1 62 1
 1 75 1 1 59 0 0 86 0 0 44 0 0 46 1 0 34 1 1 89 1 0 96 0 0 81 1 1 34 0 0 35 1 0
 18 1 0 74 1 0 33 0 1 49 0 0 87 0 1 33 1 0 68 0 0 39 0 1 65 1 1 54 0 0 35 1 0
 18 0 1 88 1 0 51 1 1 49 1 1 24 1 1 18 1 0 74 0 1 76 0 0 87 1 1 13 0 1 38 0 0
 73 1 0 36 1 1 60 1 1 11 1 1 69 0 0 21 1 1 92 0 0 69 1 0 85 1 0 27 1 1 61 0 1
 48 1 0 23 1 1 83 0 1 74 1 0 55 1 0 21 1 0 97 1 0 22 0 0 63 1 0 3 0 1 8 1 1 24
 1 1 1 0 1 63 1 1 12 1 1 57 0 0 67 0 1 0 1 1 26 1 0 25 1 1 10 0 0 93 1 1 23 0 0
 98 0 1 95 0 0 36 0 1 69 1 1 97 0 0 68 1 0 39 0 1 6 1 1 6 0 1 22 0 0 0 0 1 57 0
 0 90 1 1 40 1 1 51 0 1 14 1 0 15 1 0 24 1 0 22 0 0 53)

States: 100

(action table omitted for space)

Fitness Score       : 11590.467
Halted              : NIL
Sigma               : 993
Shifts              : 1000
Last State          : 14
Last Position       : 994
Length Traversed    : 996
Avg Block Size      : 496.5
Max Block Size      : 992
Condensed Tape: (1 992)
```

This search for a hundred state machine, with a high shift limit, results in non-halting infinitely looping machines, a local maxima that the agent gets stuck in. A deeper search with high mutation rate, or differently weight fitness test may be able to avoid this for high-state machines. Any candidate or top contender found with this many states would never be able to be run to completion, so I would not be able to know whether I had a busy beaver or not.

```
******** Search completed in 171.183 secs ********

Population size  : 200
Generations      : 20
Mutation Rate    : 1/5
Machine States   : 1000
Shift Limit      : 1000
Fitness weights:
  Weight a   : 1/50
  Weight b   : 1/20
  Weight c   : 1/50
  Weight d   : 1/30
  Weight e   : 1200
  Weight f   : 1/50
  Weight g   : 1/30
  Weight h   : 1
  Weight i   : 700

**************************************************


******************* best fit *********************
Machine #: (omitted for space)

States: 1000

(action table omitted for space)

Fitness Score       : 2793.2383
Halted              : T
Sigma               : 27
Shifts              : 921
Last State          : 0
Last Position       : -37
Length Traversed    : 46
Avg Block Size      : 2.25
Max Block Size      : 7
Condensed Tape: (1 1 2 1 1 2 1 3 1 1 6 7)
```

With the shift limit extremely low compared to the size of the 1000-state machines, it looks like a semi-productive successfully halting machine has been found. It is not very impressive considering the upper bound of ∑(1000), a number so large it is not even known, but the fact that the agent found a halting and productive machine is satisfying. It looks like this machine has a complex behavior, because it does not halt until 921 shifts, but traverses only over 46 cells of the tape. This proves that not every one of the 1000 states is reached, but it is not clear how many states this machine uses. It would be possible to map this machine to a machine of lesser states, and build a better busy beaver candidate, using only the states used in this machine. Searching for very large-state busy beavers and reducing the results to smaller state machines may be another effective and unexplored attempt at better understanding this problem.

```
******** Search completed in 170.766 secs ********

Population size  : 200
Generations      : 20
Mutation Rate    : 1/5
Machine States   : 1000
Shift Limit      : 1000
Fitness weights:
  Weight a   : 1/50
  Weight b   : 1/20
  Weight c   : 1/50
  Weight d   : 1/30
  Weight e   : 1200
  Weight f   : 1/50
  Weight g   : 1/30
  Weight h   : 1
  Weight i   : 700

***************************************************


****************** best fit ********************
Machine #: (omitted for space)

States: 1000

(action table omitted for space)

Fitness Score       : 2798.2644
Halted              : T
Sigma               : 30
Shifts              : 841
Last State          : 0
Last Position       : -13
Length Traversed    : 44
Avg Block Size      : 3.3333333
Max Block Size      : 7
Condensed Tape: (5 2 3 5 2 2 2 7 2)
```

With the same parameters, the agent finds another highly-productive, successfully halting machine. This has been the most productive machine found so far, and halts in 841 shifts. Obviously, all of the states are not access, and it is not a simple machine because only 44 cells of the tape are traversed, leaving 30 non-blank symbols on the tape. A large difference between the shifts performed and length traversed might be a characteristic of a busy beaver candidate, because it implies complex behavior. When traversing over a small subset of the infinite tape, changes made to the tape will determine the behavior of the machine. This machine also could be mapped to a TM with less states, using only the states reached here.

```
******** Search completed in 0.368 secs ********

Population size  : 20
Generations      : 5
Mutation Rate    : 1/10
Machine States   : 3
Shift Limit      : 500
Fitness weights:
  Weight a   : 1/5
  Weight b   : 1/5
  Weight c   : 1/10
  Weight d   : 1/5
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 100
  Weight i   : 200

**************************************************

******************* best fit *********************
Machine #: (1 0 2  0 0 3   1 1 1  1 0 0   0 0 0  0 1 2)
States: 3

State   Read    Write   Move    Next-State
1       0       1       0       2
1       1       0       0       3
2       0       1       1       1
2       1       1       0       0
3       0       0       0       0
3       1       0       1       2

Fitness Score       : 1102.5
Halted              : T
Sigma               : 2
Shifts              : 7
Last State          : 0
Last Position       : -1
Length Traversed    : 3
Avg Block Size      : 2.0
Max Block Size      : 2
Condensed Tape: (2)
```

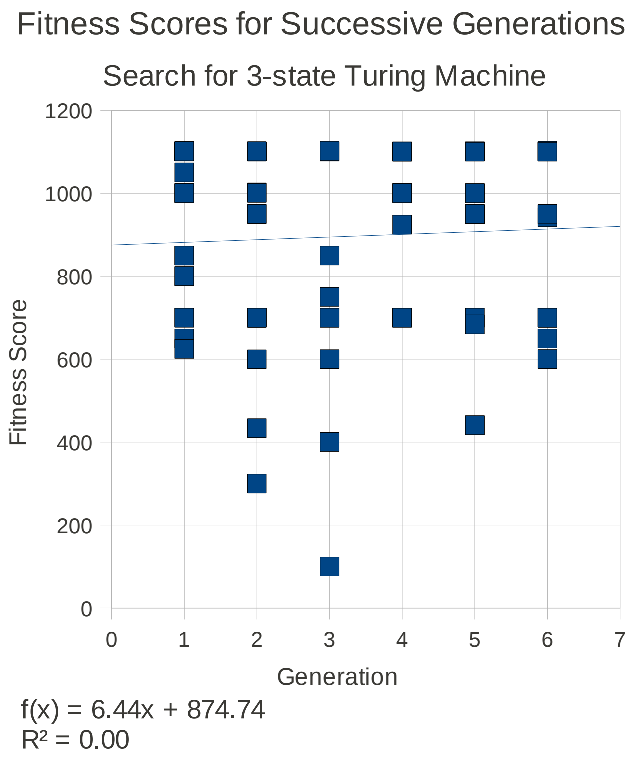
The result of a small search of 20 machines over 5 generations is seen here with the fitness scores of the machines in each population. The machine found actually does have a complex behavior, even though it leaves only 2 ones on the tape. It traverses over 3 cells in 7 shifts and reaches all three states.

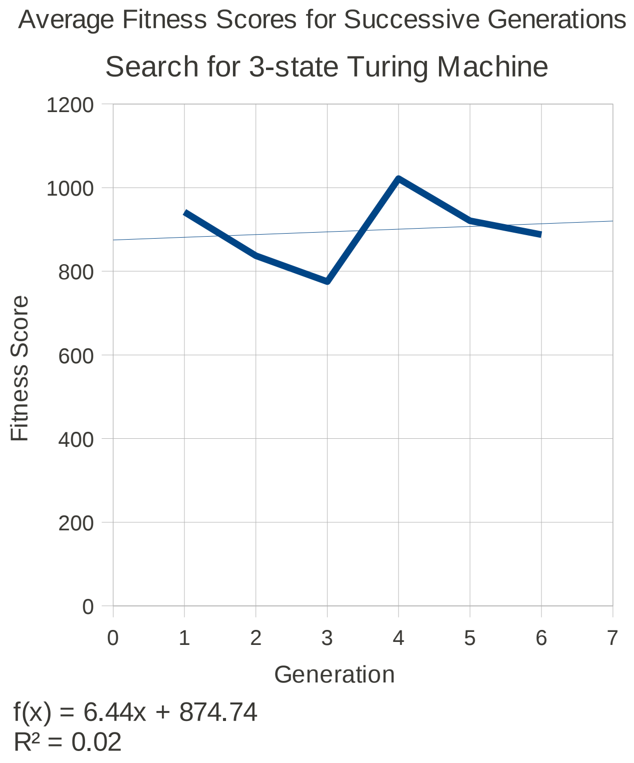
The fitness scores below are for each machine in each generation. The fitness ratio is the ratio of the machine's score to the highest fitness score in the population.

```
                                                Fitness Ratio         Fitness Score
Initial population
(0 0 1 0 1 3 1 0 0 0 0 3 1 1 2 1 0 1)   2333/3673              699.9
(1 1 0 1 1 0 1 1 1 0 0 2 1 0 0 1 0 1)   10009/11019           1000.9
(1 1 3 1 1 1 0 1 2 1 0 3 0 0 0 1 0 0)   3670/3673              1101.0
(1 0 1 0 0 3 1 0 3 1 0 3 0 1 0 0 0 1)   2833/3673              849.9
(0 1 1 1 0 1 0 0 0 0 0 2 1 1 2 0 1 2)   7000/11019             700.0
(0 0 0 1 1 1 1 0 3 0 0 3 0 1 3 0 0 0)   10001/11019           1000.1
(1 1 3 0 0 2 0 0 2 0 0 1 0 0 0 0 1 0)   3670/3673              1101.0
(1 0 3 1 1 3 1 1 2 1 0 3 1 0 3 1 1 1)   6499/11019             649.9
(1 1 0 1 0 3 0 0 1 1 1 2 0 1 2 0 0 2)   11009/11019           1100.9
(1 0 0 1 0 0 0 1 1 0 1 0 1 1 3 1 0 2)   11006/11019           1100.6
(1 1 3 0 0 2 1 1 2 0 1 2 0 1 2 0 1 2)   12487/22038            624.35
(1 1 1 0 0 3 0 0 2 1 0 3 1 0 0 0 1 0)   3500/3673              1050.0
(1 1 0 0 1 2 1 0 1 0 0 1 1 0 3 0 1 3)   11009/11019           1100.9
(1 1 2 1 1 2 1 0 1 1 0 0 1 1 0 1 1 3)   1                      1101.9
(1 0 1 1 0 0 0 0 2 0 1 2 0 0 0 0 1 0)   2833/3673              849.9
(0 1 1 0 0 2 0 0 3 1 0 0 0 1 1 0 0 1)   8000/11019             800.0
(1 1 3 1 1 0 0 1 3 1 1 3 0 0 1 1 0 2)   3671/3673              1101.3
(0 1 0 0 1 1 0 1 2 0 0 2 0 0 0 1 1 0)   10004/11019           1000.4
(0 1 1 1 0 0 1 0 1 0 1 2 1 0 3 0 1 1)   8000/11019             800.0
(0 1 2 0 0 0 0 1 0 0 1 0 1 0 1 1 0 3)   11008/11019           1100.8
```

```
Generation : 1
(0 0 1 0 1 3 1 0 0 0 0 3 1 1 2 1 0 1)   2333/3670              699.9
(0 0 1 0 1 3 1 0 0 0 0 3 1 1 2 1 0 1)   2333/3670              699.9
(0 0 1 0 1 3 1 0 0 0 0 3 1 0 0 1 0 1)   5999/11010             599.9
(0 0 1 0 1 3 1 0 0 0 0 3 1 1 2 1 0 1)   2333/3670              699.9
(0 0 3 1 1 2 1 0 1 1 1 0 1 1 0 1 1 1)   1                      1101.0
(1 1 0 0 1 3 1 0 0 0 0 3 1 1 2 1 0 1)   11009/11010           1100.9
(1 0 1 0 0 1 0 1 3 1 0 0 0 0 3 1 1 2)   9499/11010             949.9
(1 0 0 1 1 3 1 1 1 0 1 2 1 0 3 0 0 0)   5503/5505              1100.6
(0 0 1 0 1 3 1 0 0 0 0 3 1 1 2 1 0 0)   2333/3670              699.9
(1 1 2 1 0 1 1 1 0 1 1 0 1 1 1 0 0 2)   5009/5505              1001.8
(1 1 3 1 1 1 0 1 2 1 0 3 0 0 0 1 0 0)   1                      1101.0
(1 1 2 1 0 1 0 0 2 0 1 1 1 0 1 0 0 3)   723/1835               433.8
(1 0 1 1 1 3 1 1 1 0 1 2 1 0 3 0 0 0)   9499/11010             949.9
(0 0 1 0 1 3 1 1 1 0 0 2 1 0 0 1 0 1)   2333/3670              699.9
(1 0 0 0 0 3 1 1 2 1 0 1 1 1 0 1 1 0)   5503/5505              1100.6
(0 0 1 0 1 3 1 0 0 0 0 3 1 1 2 1 0 1)   2333/3670              699.9
(0 0 1 0 1 3 1 0 0 0 0 2 1 0 0 1 0 1)   2333/3670              699.9
(1 0 2 0 0 2 1 1 2 1 0 3 0 0 3 0 1 3)   3001/11010             300.1
(1 1 0 1 1 1 0 0 2 1 0 0 1 0 1 0 0 1)   10009/11010           1000.9
(1 1 3 1 1 1 0 1 2 1 0 3 0 0 0 1 0 0)   1                      1101.0
.
.
.
.other generations omitted
.
.
.
******** Final Population ********
(0 1 3 1 0 2 0 0 0 1 1 3 1 0 1 0 0 2)   1573/1575             1101.1
(0 0 1 0 1 0 1 0 3 0 0 3 1 1 3 1 0 2)   2333/3675              699.9
(0 0 0 1 1 3 1 0 1 0 0 2 0 0 1 0 1 0)   3667/3675             1100.1
(0 0 1 0 1 0 1 0 3 0 0 3 1 1 3 1 0 2)   2333/3675              699.9
```

```
(0 0 1 0 1 0 1 0 3 0 0 3 1 1 3 0 0 2)    2333/3675          699.9
(1 0 2 0 0 3 1 1 1 1 0 3 0 0 3 0 0 0)    11008/11025        1100.8
(1 0 2 0 1 3 1 0 2 0 0 0 1 1 3 1 0 1)    1357/1575          949.9
(0 1 3 1 0 2 0 0 0 1 1 3 1 0 1 0 0 2)    1573/1575         1101.1
(0 1 3 1 0 2 0 0 0 1 1 3 1 0 1 0 0 2)    1573/1575         1101.1
(1 0 2 0 0 3 1 1 1 1 0 0 0 0 0 0 1 2)    1                  1102.5
(1 0 3 0 0 3 1 1 0 1 0 3 0 0 3 0 1 2)    6004/11025         600.4
(0 0 1 0 1 0 1 0 3 0 0 3 1 1 3 1 0 2)    2333/3675          699.9
(0 0 1 0 1 0 1 0 3 0 0 3 1 1 3 1 0 2)    2333/3675          699.9
(0 1 3 1 0 2 1 0 3 0 0 3 1 1 3 1 0 2)    433/735            649.5
(1 1 3 1 0 2 0 0 1 0 1 0 1 0 3 0 0 3)    88/105             924.0
(0 0 1 0 1 0 1 0 3 0 0 3 1 1 3 1 0 2)    2333/3675          699.9
(0 1 3 1 0 2 0 0 0 1 1 3 1 0 1 0 0 2)    1573/1575         1101.1
(0 1 3 0 1 0 1 0 3 0 0 3 1 1 3 1 0 2)    211/245            949.5
(0 0 0 1 1 3 1 0 1 0 0 2 0 0 1 0 1 0)    3667/3675          1100.1
(0 1 3 1 0 2 1 0 3 0 0 3 1 1 3 1 0 2)    433/735            649.5
******************* end of file ********************
```

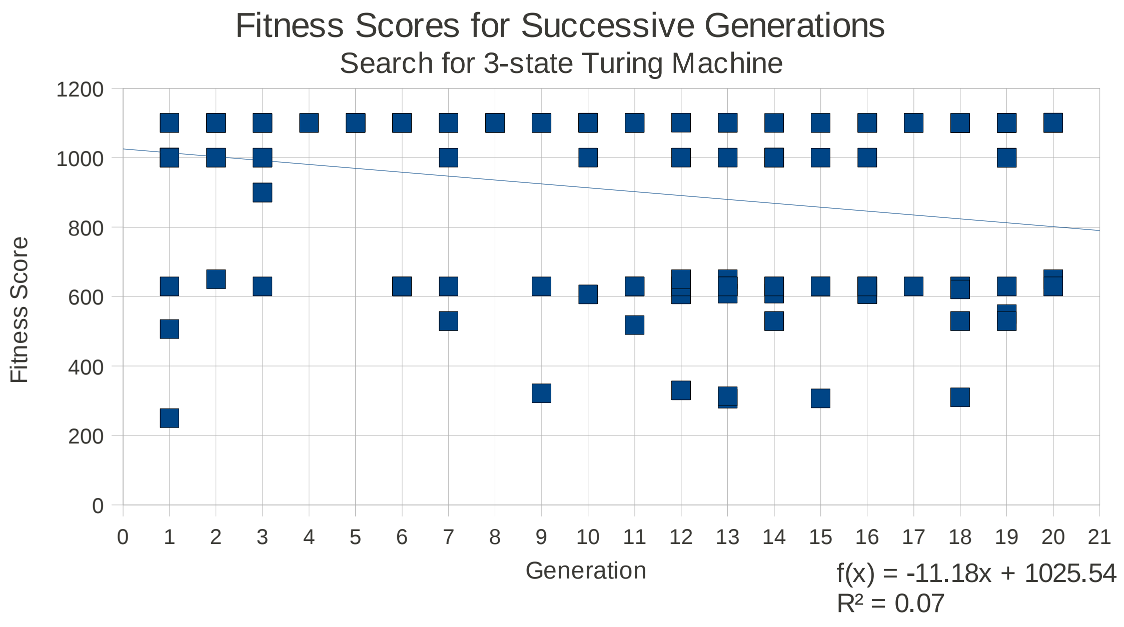




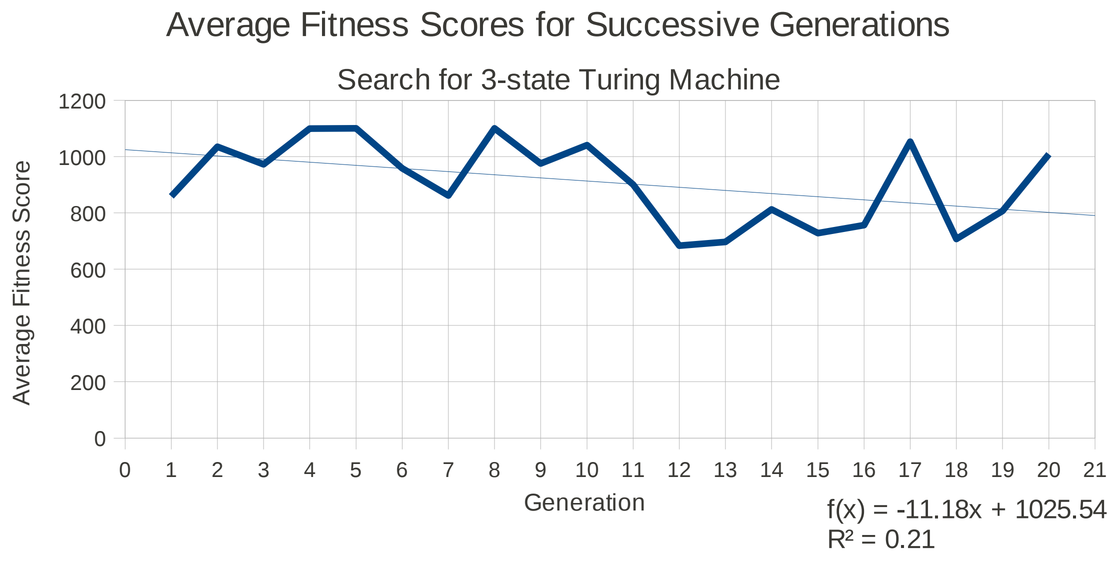
The fitness scores for this short search look like they might share a direct relationship with each generation but the correlation coefficient implies no relation at all. The average score fluctuates between every two generations.

```
******** Search completed in 0.115 secs ********

Population size  : 10
Generations      : 20
Mutation Rate    : 1/10
Machine States   : 3
Shift Limit      : 100
Fitness weights:
  Weight a   : 1/15
  Weight b   : 1/15
  Weight c   : 1/10
  Weight d   : 1/15
  Weight e   : 500
  Weight f   : 1/10
  Weight g   : 1/10
  Weight h   : 100
  Weight i   : 200

**************************************************


******************* best fit *********************
Machine #: (1 0 2  1 0 3   1 1 2  1 1 1   0 0 0  1 0 1)
States: 3

State   Read    Write   Move    Next-State
1       0       1       0       2
1       1       1       0       3
2       0       1       1       2
2       1       1       1       1
3       0       0       0       0
3       1       1       0       1

Fitness Score       : 1101.4
Halted              : T
Sigma               : 3
Shifts              : 9
Last State          : 0
Last Position       : -3
Length Traversed    : 5
Avg Block Size      : 3.0
Max Block Size      : 3
Condensed Tape: (3)
```

This search resulted in a complex 3-state machine, which halted after 9 steps. This close to the number of steps performed by the known 3-state busy beaver, which halts in 14 steps, and about half of S(3), which is 21. This search was small so that the data for the population fitness scores could be analyzed (see next page).



This chart represents a scatter plot and linear regression for the fitness scores of the population at each generation. The correlation coefficient suggests no relationship between the fitness scores of each population and the generation. The regression line actually steadily declines, but there does not seem to be any type of order or overall improvement in each generation. There looks like 2 large gaps in the data though, between 700-900 and 350-450, splitting all of the machines into three classes. The fitness test may be too discrete and only able to place the machine into one of the 3 classes seen in the chart. There also looks like a ceiling at about 1100, which many machines in the upper class score close to, but not higher than. The 3-state machine found that displays complex behavior was in this class.



The average scores for each generation, with the same regression line, suggest that at any particular generation, most of the machines are in one of the 3 classes. The average score never increases or decreases for more than 2 generations, fluctuating between the three classes (although it never looks like the average score enters the bottom class of machines). The trend is downward, but probably not enough to imply a relationship between the average score and each generation.

Part V: Conclusion

The agent was able to find a number of simple and complex machines, but was not consistent enough to be very reliable. The fitness test was able to pick out successfully halting, productive machines, but the reproduction process may not have been able to keep the good parts of the machines intact. The fitness scores of each generation seemed too random and variable to imply a successful evolution. One interesting result when searching for k-state machines was finding busy beavers with less than *k* states. The 2-state busy beaver appeared in several searches for 3-state machines, embedded within 3-state machines without using the third state. An algorithm to measure the state changes and simplify the machine to a fewer number of states may be helpful in better measuring and understanding properties of complex Turing Machines. This is something I did not anticipate, but may implement.

Many improvements can be made to the program after gaining a better understanding of the problem. The fitness score should be able rate the machines on a fuller scale, and this may be achieved by changing the weights or the sub-tests used. Efficiency is also an issue, although the algorithm turned out to perform better than I had expected after a few optimizations. I sacrificed simplicity in a few cases to achieve this, to avoid processing the same data twice or performing the same operation in different parts of the program.

Overall, the agent failed to find any promising results, which is not surprising. Even simple system which display complex behavior are very difficult to evaluate accurately with mathematical equations alone. Certain properties of the output and machine itself can be looked at to increase the probability of finding a true busy beaver, but it is not clear which properties will be the most valuable to use for searching. A learning mechanism may be introduced to the program to adjust the weights accordingly, but that might not yield any better results. Using the results of some of the sub-tests to weight the other sub-tests is also an option, and can help distinguish non-halting machines from the halting ones.

Even though the agent was unable to find any true busy beavers, it was successful in discovering some known candidates for 3-state machines and some other interesting machines. The heuristic was able to pick out halting and somewhat productive machines, but it is not clear if the evolutionary process helped produce these machines. There were high scoring machines in every population, including the initial, randomly generated population. These machines may have passed on parts to high scoring machines in later generations, but I do not know for sure because I did not track the genes or build a family tree. If an evolutionary tree was built for several generations of small populations, it would be clear if the parts of high-scoring machines went to high or low scoring machines in the next generation.

Lisp was a good language to use for this project, which involved many unknowns. It is the first program I have written in Lisp, so there are many improvements that can be made to the code. With some the ones mentioned, the agent may be able to produce more consistent results. It is unlikely that it will be able to compete with traditional brute-force method of finding busy beavers, but may give some insight to the nature of the busy beaver problem.

Works Cited

[Rad62] Tibor Radó. On non-computable functions, *Bell Systems Tech. J*. 41, 3 (May 1962).

[MaS90] Rona Machlin and Quentin F. Stout. The Complex Behavior of Simple Machines. *Physica D*, 42, pages 85-98, 1990.

[MPC99] Penousal Machado, Francisco B. Pereira, Amílcar Cardoso, Ernesto Costa. Busy Beaver – The Influence of Representation. In *Genetic Programming*, LNCS 1598, pages 29-38, Springer-Verlag, Berlin, 1999.

[LiR65] S. Lin and T. Rado. Computer Studies of Turing Machine Problems. Journal *of the Association for Computer Machinery,* 12(2), pages 196-212, April 1965.

[Gab94] Gabriel, Richard. *Lisp: Good News, Bad News, How to Win Big*. Technical report, 1994.