A Universal Turing Machine in Conway's Game of Life

Paul Rendell
Department of Computer Science
University of the West of England
paul@rendell-attic.org

ABSTRACT

In this paper we present a Universal Turing Machine build in the Cellular Automaton Conway's Game of Life. This is an extension of the Turing Machine built previously by the author [10]. It is example of spatio-temporal collision based computation and has infinite tape provided by two stack structures which grow continuously using collision based construction. Two patterns the fanout and takeout are described which are key in solving the routing and synchronization problems. The procedure used to find a viable order of synthesis of the parts in the stack construction is described.

KEYWORDS: Universal Turing Machine, Conway's Game of Life

1. INTRODUCTION

The author constructed a Turing Machine in Conway's Game of Life [10]. This machine has a Finite State Machine of three states and uses three symbols and a short fixed length of tape. Figure 1 shows a snapshot of this machine which doubles the length of a string of symbols on its tape.

This machine was claimed to be extendable to a universal Turing machine. In this paper we present the results of that extension. It takes two parts. Firstly the Finite State Machine has been expanded to accommodate a simple universal Turing machine. Secondly the stack has been modified and the moving convoys of rakes added to construct stack cells continuously.

We will describe the old Turing machine and then go on to describe the extensions and the methods used to achieve these results.

2.1. Conway's Game of Life

The Game of Life, invented by J.H. Conway [2] is a cellular automaton, where an infinite universe is divided into cells. Each cell takes a state from a binary set and updates its state according to certain strict rules. All cells change their states simultaneously in discrete time. For Conway's Game of Life the cells have two states, usually called live and dead, and the rules are based on the number of neighbouring cells, which are alive.

2.2. Turing Machines

A Turing Machine consists of a finite state machine which interacts with an infinite date storage medium. The data storage medium takes the form of a theoretically infinite long tape on which symbols can be written and read back via a moving read/write head. The symbols which can appear on the tape must be members of a finite alphabet.

The machine performs a regular cycle, reading a symbol off the tape, overwriting that symbol with another or the same one and them moving the read/write head one symbol width either right or left along the tape. This continues until the machine reaches a conclusion in its internal workings and halts.

The finite state machine is the program which determines from the symbol read and an internal 'state' what the symbol to write should be and which way to move the read/write head. It also has the opportunity to change its internal state. There is one last thing it can do, it can halt.

The operation of the machine is completely determined by a table which gives for each combination of input symbol and internal state the quintuple:

- The symbol to write.
- The new internal state.

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.

- The direction to move the read/write head.
- Whether to halt or continue.

A Turing Machine has no other communication with the outside world than the contents of the tape. The contents of the tape before it starts represent its input and the contents on the tape when it has finished represents its output. It is possible that the machine never stops.

The Turing Machine is as a mathematical tool to probe the limits of computability. its power comes from its simplicity, speed and efficiency are of no consequence.

2.2. Universal Turing Machines

A universal Turing machine is a Turing machine which simulates the actions of another Turing machine. It takes as its input a description of a Turing machine and a description of that Turing machines tape. It leaves as part of its output a description of the simulated Turing machines tape as if the simulated machine had performed the calculation.

The existence of universal Turing machines proves that a Turing machine is capable of performing every possible calculation.

3. The GoL Turing Machine

The Turing Machine built in Conway's Game of Life in 2000 is shown in the snapshot in Fig. 1 and diagrammatically in Fig. 2.

The finite state machine part can clearly be seen as the square pattern in the bottom left of the picture. This has an addressing mechanism on the left (state value) and at the bottom (symbol value) and nine memory cells to hold the data table to describe the action for each combination of the internal state and symbol values.

The Turing Machines tape is represented by the two stack mechanises seen extending top left and bottom right. Each stack cell can trap 3 gliders using the kickback reaction. This is the reaction that turns a glider through 180 degrees when encountering another at right angles. The traps can be seen as empty rectangles between the denser patterns in the cells. These denser patterns delay the transit of the gliders from one cell to the next so that the destination cell is empty when they arrive. The control signals for the cells travel up the sides of the stack.

Between the two stacks is the logic to perform serial to parallel and parallel to serial conversion and generate the stack control signals so that one stack performs a push operation and the other performs a pop operation.

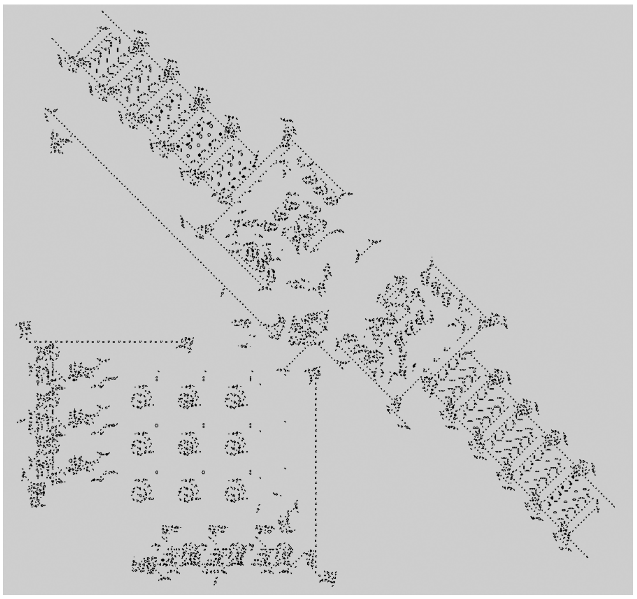


Figure 1. GoL Turing Machine

The other item visible is the delay loop for the next state which extends from the centre towards the left top corner underneath the left stack. The next state value is copied from the data read from the finite state machine and sent round this loop to address the finite state machine for the next cycle in conjunction with the symbol popped from one of the stacks.

3.1. Important Patterns

3.1.1. Queen Bee

The queen bee shuttle is little symmetrical pattern that move back and forth and lays a bee hive as it turns. It dies if the bee hive is still there when it returns. Figure 3 shows the basic pattern (a), then after 15 generations (b) and after 30 generations (c). There are a number of ways of removing the bee hive, An eater, A block another queen bee shuttle, pentadecathlon and so on.

3.1.2. Gosper Gun

This is formed from 2 queen bee shuttles back to back. The debris from the bee hive sparks creates a glider every cycle of 30 generations. The pattern is shown top left in Fig. 4.

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.

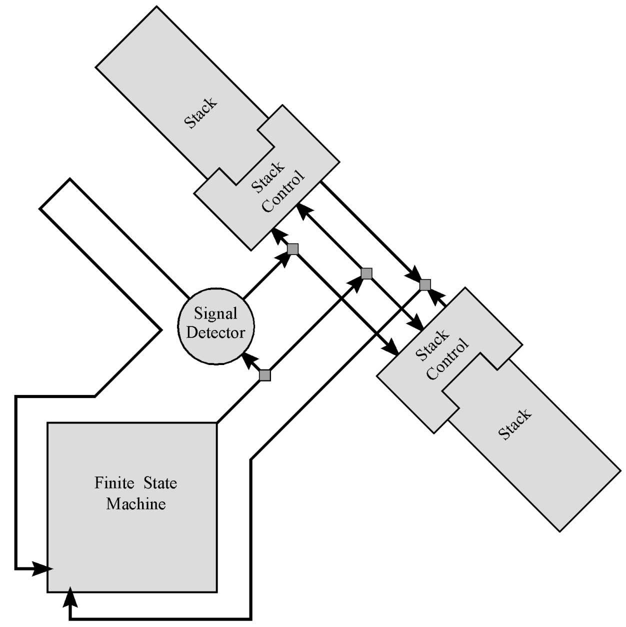
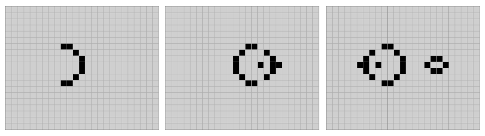


Figure 2. GoL Turing Machine Diagram



(a) (b) (c)

Figure 3. Queen Bee

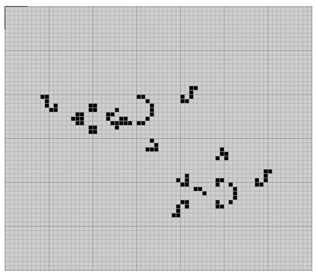
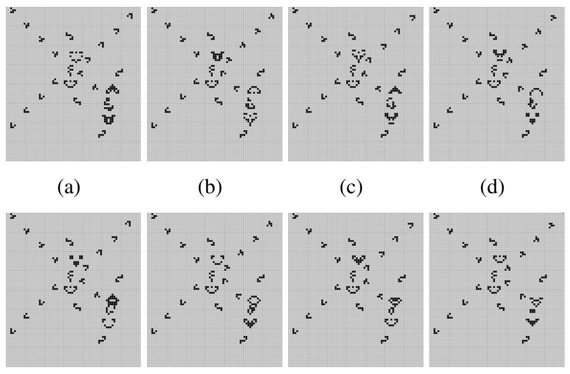


Figure 4. Gosper Gun and Buckaroo

3.1.3. Buckaroo

This is formed from a single queen bee shuttle stabilized by an eater. It is of particular interest because of the spark created as the bee hive is removed. This can reflect a glider as shown bottom left in Fig. 4.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 5. Fanout

3.1.4. Fanout

This makes use of the reaction stabilising one end of queen bee shuttle by reflecting a glider. Two such patterns are placed back to back so that both can reflect gliders. If the input side has a missing glider then its queen bee is stabilised by the other queen bee and the glider. This suppresses the reflecting action and no glider is emitted from the output that side either. A standard gosper gun supplies the stream of gliders.

One very useful attribute of this setup is that it does not work for just one configuration but continues to operate over a range as shown in fig. 5. This allows loops to be built easily. One of these is the memory cell described below .

3.1.5. Kickback Reaction

This is the reaction that turns a glider through 180 degrees when encountering another at right angles. This is used to form the walls of the stack cells.

3.1.6. Takeout

This is a 90 degree glider reflector made up of two pentadecathlons. A glider hitting the spark of a pentadecathlon just right makes a block and the spark from another pentadecathlon converts this into a glider which moves out of the way just in time. Figure 6 shows this in single generation steps.

The pentadecathlons sit on one side of the glider path and at just the right distance from the site of a kickback reaction a glider can pass by the takeout in one direction but be picked up and reflected by 90 degrees on its return from

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.

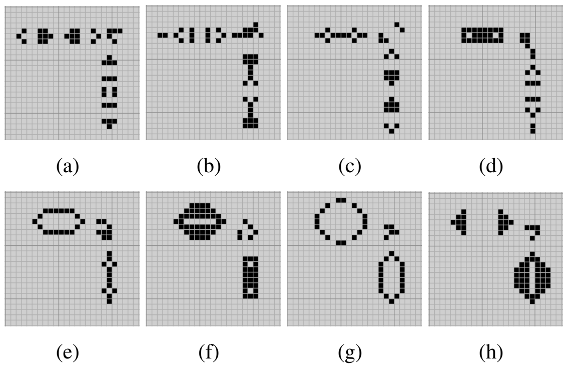


Figure 6. Take Out reflector in steps of 1 generations

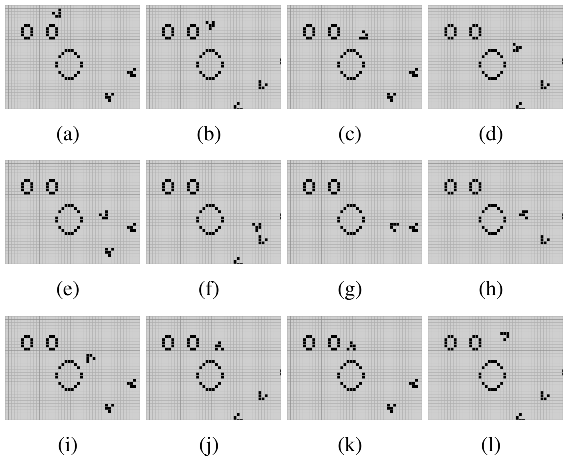


Figure 7. Take Out reaction in steps of 15 generations

the kickback reaction. Figure 7 shows this in 15 generation steps.

The addition of another pentadecathlon acting as a 180 degree glider reflector removes the limitation on the distance from the kickback reaction site and also adds the ability to adjust the timing. Changing the distance of this extra pentadecathlon by one cell changes the path length by eight generations.

3.2. The Finite State Machine

The heart of the finite state machine is an array of memory cells. Each cell holding one quintuple for a specific state symbol combination.

The memory cell is made up of a fanout with one output looped back to connect to the input using buckaroos. The smallest cell has a loop of 240 generations with places for 8 gliders as shown in Fig. 8.

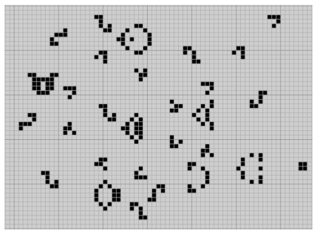


Figure 8. Memory Cell

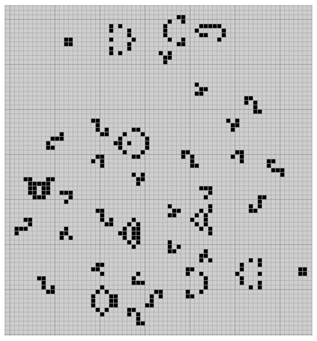


Figure 9. Gated Memory Cell

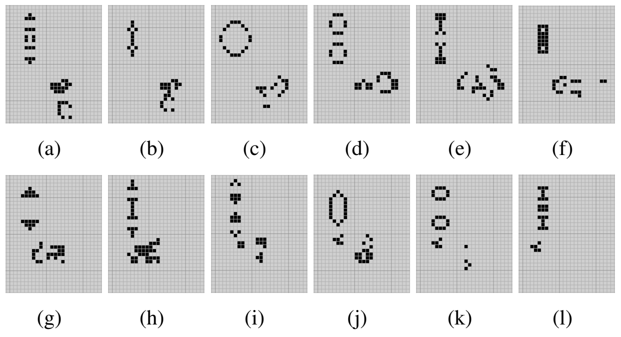


Figure 10. Matrex Addressing in steps of 4 generations

The other output can be gated with another gosper gun and a single glider supported by an eater can make an eight glider hole in this to let out the data. Figure 9 shows the cell with the gate at the top and the eater on the left.

A memory cell is addressed by spaceships travelling orthogonally through the gaps between the cells. The collision of a Light Weight Space Ship and a Medium Weight Space Ship forms a block which the spark from a pentadecathlon can transform into a glider. Figure 10 shows this with snapshots 4 generations apart.

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.

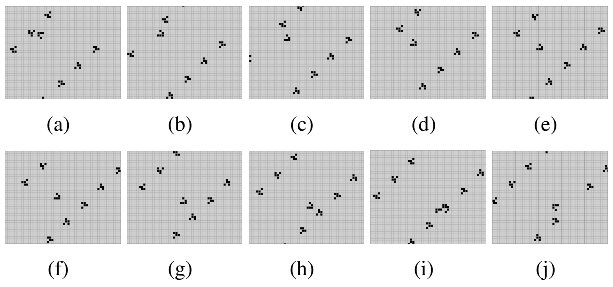


Figure 11. KickBack reaction in steps of 8 generations

3.3. The Stack

The kickback reaction is used to make stack cell walls. A glider is trapped between to streams of gliders by being kicked back from one to the other. For this to work the stack cell walls must be placed so that:

- The trapped glider loop is a multiple of the period of the gliders forming the wall (30 generations).
- The trapped glider loop is a multiple of the period of a glider (4 generations).
- The distance between the walls is an integer.

The minimum loop is thus 120 generations with walls spaced 15 cells apart. This stack cell could hold 4 gliders although only 3 are used in this design.

Control signals to open holes in the stack cell walls pass up both sides of the stack. One fanout for each stack cell copies these to make a stack cell wall.

The takeout allows gliders coming out of a stack cell to be separated from those going in. This pattern is used in the original stack to create the delay required between stack cells so that the cell is empty when gliders enter it. This is also the reason why the old stack is not at 45 degrees. The takeout plus a buckaroo provide a delay of 120 generations with an offset of just 6 Life cells. Figure 12 shows a snapshot of the stack.

4. GOL UNIVERSAL TURING MACHINE

The universal Turing machine built in Conway's Game of Life in 2010 is shown in the snapshot in Fig. 13. This is a straight forward extension of the TM described in Sec. . It has a longer tape and a larger finite state machine which contains a 13 state 8 symbol universal Turing machine. This directly simulate a Turing machine using the relative distance between transition descriptors instead of transition identities. A full description of this can be found in [11].

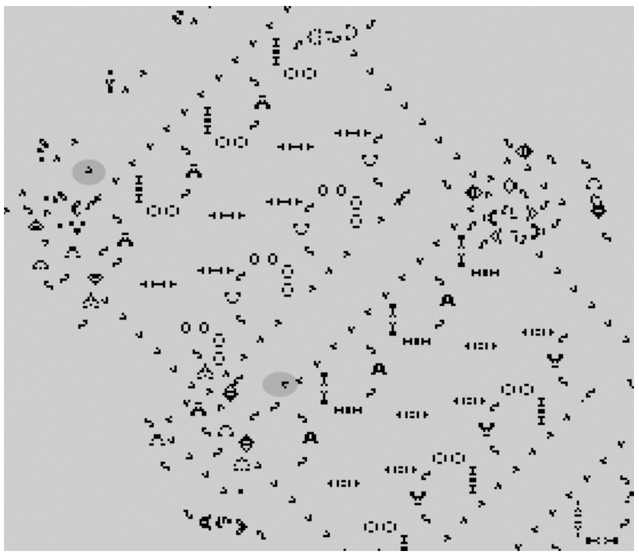


Figure 12. The stack with trapped gliders shaded

The UTM was found to be efficient for the example for which it was optimised, a unary multiplication TM multiplying [1]. It took fewer cycles than would be required to move the UTM read/write head from the centre of the TM tape description to the centre of the TM description and back again for each TM cycle.

5. THE FORTY FIVE DEGREE STACK

A forty five degree stack is required so that it can be constructed by streams of gliders generated by two convoys of rakes. Each rake generates a glider periodically and move along at a constant speed. The gliders from one convoy arriving at the construction site in the opposite direction to the gliders from the other convoy.

The new design replaces the takeout delay mechanism with a second kickback cell. Originally this meant duplicating all the control mechanisms for the walls and having 2 sets of controls on both sides. This was superceeded by the idea of using one control for both sides of the stack cell by bending it into a 'U' shape. For this to work the width of the cell had to be increased so that a hole created to allow a glider out of the far side of the 'U' did not let it out of the near side.

Doubling the cell size from the minimum loop of 120 generations to 240 generations provides 8 gliders in the cell wall to control the 3 trapped gliders. A single hole is sufficient

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.

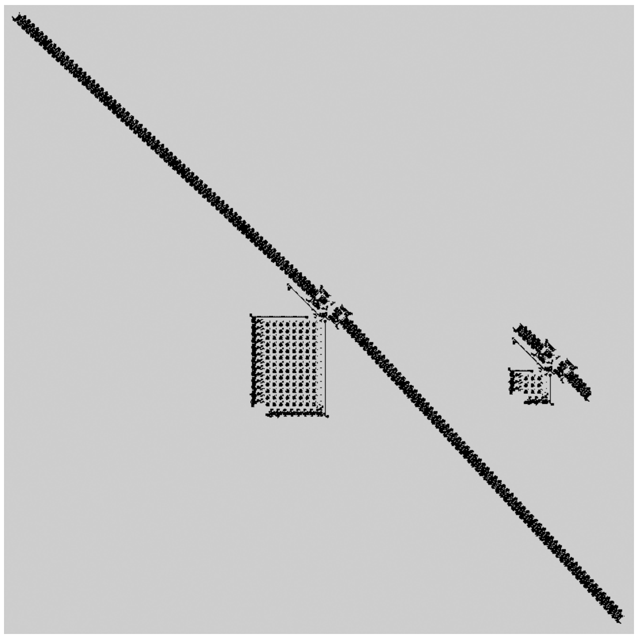


Figure 13. GoL Universal Turing Machine has the description of a Turing machine as part of the data on the tape with the TM alongside at the same scale.

to allow a glider into the cell but a double hole is required to let it out. There is no problem letting gliders out of the near size of the 'U' but to let gliders out of the far size the double hole would need to pass round the 'U' without disturbing the trapped gliders. This was solved by choosing length of the 'U' and the phase of the gliders in the trap so that the hole made by the containing kickback reaction on the nearside appears next to the hole made by the kickback reaction on the far side. Thus if this latter glider is missing from the cell wall there is always a double hole to let the glider out. Figure 14 shows a snapshot of the stack.

6. THE STACK CONSTRUCTOR

The stack cells are constructed by gliders sent in from two convoys of rakes one on each side of the stack. There are three stages to the construction, in the first stage a still life field is built up. In the second phase the still life components are activated and in the final phase a little tidying up is required so that the new stack cell can be used.

6.1. The Convoys

Convoys have built from diagonal rakes with speeds of c/12 (moving one life cell diagonally in 12 generations) and c/5 (in 5 generations). However these are very large and slow.

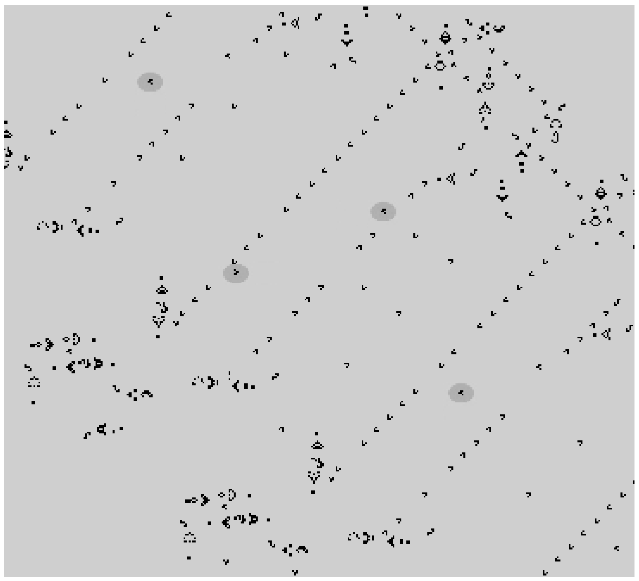


Figure 14. The 45 degree stack with trapped gliders shaded

We will describe the conveys built using the much smaller period 360 c/2 orthogonal rakes.

The period 360 rakes where easily constructed from the parts in Jason Summers collection [14]. The rakes work in pairs, a forward rake and a backward rake, to add gliders to the construction stream using the kickback reaction. Figure 15 shows two snapshots of a pair of rakes inserting a glider into the stream.

The procedure used to build the convoy of rakes is very simple and a changing the type of rake is trivial, Tab. 1 shows the full python function used. There are two points to note in this function, firstly no effort has been made to place the rakes as close together as possible and secondly that when creating the convoy for the left hand side there is an option of two different rake constructs. The first is the pair of rakes described above, the second is a slightly more complex group to place to gliders even closer by using an intermediate eater. The latter is used just once to enable the synthesis for a boat.

6.1.Construction Tools

The basic synthesis of the components was taken from Mark Niemiecs collection [8] which can now also be found at [7]. A cautious approach was taken in the synthesis to keep the density of gliders in the construction stream low.

The objects requiring synthesis are; Queen Bee, Eater, Boat,

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.

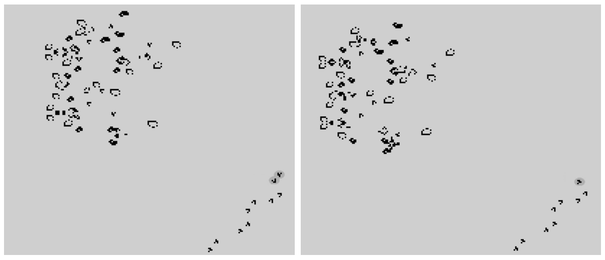

(a) The gliders produced by the rakes are shaded (b) The inserted glider is shaded

Figure 15. A pair of period 360 rakes inserting a glider

```
def getPatt(self):
  patt = pattern()
  # process list sorted by y then x
  if self.name=='left':
    for [x,y,r,Pty] in sorted(self.list,\
        key=itemgetter(1,0),reverse=True):
      if Pty=='glider':
        step=71
        patt=patt[step*4](-step,step)
        patt+=p360kickback[r](x,y,flip_x)
      else:
        step=180
        patt=patt[step*4](-step,step)
        patt+=p360boatpairA[r](x,y,flip_x)
    else:
    step=71
    for [x,y,r,Pty] in sorted(self.list,\
        key=itemgetter(0,1),reverse=True):
      patt=patt[step*4](step,-step)
      patt+=p360kickback[r](x,y,rccw)
  return(patt)
```

Table 1. Python Function `getPatt` to create a convoy of rakes from a list of coordinates.

Block, Pond and Ship.

The only item requiring activation is a queen bee and in most cases this is archived with a single glider. The single glider can transform a ship into a queen bee. The ship can be created from a pond by another glider and the pond can be created by 2 gliders in an number of ways.

The synthesizing gliders are routed to there destination by:

- Glider reflection by an eater Fig. 16 (a).
- Glider reflection by a boat Fig. 16 (b).
- Glider reflection by the kickback reaction Fig. 16 (c).
- Glider reflection by Glider Collision Fig. 16 (b).

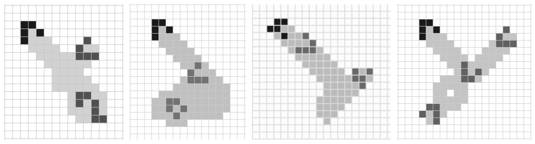

(a) Eater (b) Boat (c) Kickback (d) Tee

Figure 16. A trace by Golly script *envelope* [5] showing the path of the gliders in reflection reactions

The design of the construction was performed backwards. Starting with the completed stack cell and adding eaters and boats to route the activating gliders to the correct place with the correct timing. Once a field of still life object had been achieved the next step was to create a path all the way through the field so that gliders from both sides can work together to synthesis the objects. A python script was used for the design.

The key feature of this script is that it allows a pattern to be generated for a specific time in the construction. It shows the synthesising gliders at the correct location for that time regardless of the fact that they might not be able to get there because of other objects in the way. Table 2 shows the code for one part.

The first line `01` adds a buckaroo to the pattern `stkl` at location (2,-56) with orientation '`swap_xy`', at time 28 and with the name '`B3`'. As `stkl` already has a name '`L`' the full name of this buckaroo is '`L.B3`'. The string '`:h.s.p.90 :hb.90tttxyf`' is a list of qualifiers which give details of the orientation of the subparts, as explained below.

The line `02` specifies that the time at which the part '`L.B3.RE`' will be synthesised is `7458` generations before completion. '`RE`' is the reflecting eater of the buckaroo. A trace of this is shown in Fig. 17 with the eater labelled `B`. The other parts of the buckaroo shown in Fig. 17 are the ship `A` which will become the queen bee shuttle later in the construction and a block `C` to stabilise the other end of the shuttle.

The next line `03` specifies that the source of one of the two gliders ('`G1`' and '`G1`') which synthesis the buckaroo eater is a kickback reaction. This reaction will source the '`G1`' glider and occurs nine cells away from the site of the eater synthesis. This reaction has the full name '`L.B3.RE.LB3n1`' and the glider has the full name '`L.B3.RE.G1`'. The site of the kickback reaction is labelled `D` in Fig. 17.

Line `04` specifies the source of one of the two gliders

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.

```
01 stkl.addsubpart(ppart(’B3’,’buckaroo\
     ,’:h.s.p.90:hb.90tttxyf’)\
     ,2,-56,28,swap_xy)
02 stkl.getobj(’L.B3’).\
     changeLock(’RE’,ppart.locked,7458)
03 stkl.triggerbend(’L.B3.RE.G1’,\
     ’LB3n1’,’kickback-f’,5,9)
04 stkl.triggerbend(’L.B3.RE.LB3n1.GP1’\
     ,’LB3n2’,’teedown-’,-6,31)
05 stkl.getobj(’L.B3.H.Q’).\
     changeLock(’SH’,ppart.locked,7548)
06 stkl.getobj(’L.B3.H.Q.SH’).\
     changeLock(’P’,ppart.locked,50)
07 stkl.triggerbend(’L.B3.H.Q.SH.P.G2’\
     ,’LB3n3’,’teedown-’,-3,29)
08 stkl.getobj(’L.B3.H’).changeLock\
     (’B’,ppart.locked,lockm+3400)
09 stkl.triggerbend(’L.B3.H.B.G1’,\
     ’LB3n3’,’teedown-’,-4,31)
```

Table 2. Python code fragment for the synthesis one of the buckaroos

(’GP1’ and ’GP1’) involved in the kickback reaction. It is the product of a ‘teedown’ reaction which occurs 31 cells away from the kickback reaction. This site, labelled E in Fig. 17, is in the path through the stack so the three gliders for the ‘teedown’ reaction come directly from the convoys of rakes in the construction stream.

The line 05 specifies the synthesis time for the buckaroo ship and the line after that, line 06 specifies the synthesis time for the pond from which the ship is made. That is 50 generations before the ship. These times will set the relative positions of the gliders for these synthesises. Figure 17 shows these gliders, one labelled labelled F to make the ship from the pond and three labelled G to make the pond.

One of the qualifiers for the buckaroo in line 01 is ‘:h.s.p.90’. This specifies a non default synthesis for the pond using gliders colliding at 90 degrees. This qualifier is passed though the hierarchy of components making up the buckaroo each of which strips of a prefix so the ’h....’ is passed to the half queen bee shuttle, the ‘s....’ is passed to the ship, the ‘p...’ is passed to the pond which recognises the ‘90’.

The remaining lines specifies the source of further gliders. The other qualifier for the buckaroo :hb.90tttxyf’ refers to hb the half queen bee shuttle block. It specifies a 90 degree collision synthesis and tttxyf specifies the orientation of the synthesis. The other gliders all have a clear path from the left and require no further modification.

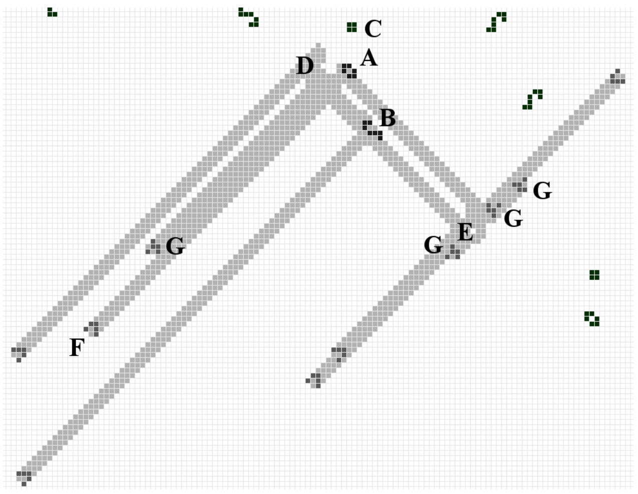


Figure 17. A trace by the Golly script *envelope* [5] showing the path of the gliders constructing the ship and the eater of the B3 Buckaroo specified in Tab. 2.

7. CONCLUSION

We have presented a fully universal Turing machine in Conway’s Game of Life which is reasonably easy to program, from which the results can clearly be read and it which runs is a convenient time in Golly. A clear demonstration of the universality of the game of life.

The continuous construction of stack cells also demonstrate the ability to built complex repetitive patterns across the space of a cellular automaton at a practical speed. This might be a useful property if a physical material can be persuaded to perform as a cellular automaton. It demonstrates that practical use can be made of a cellular automaton by injecting patterns from the edges rather than having to initialise patterns in the bulk of the material.

ACKNOWLEDGMENT

The patterns illustrating the chapter can be downloaded from the site [9] [12] and run on several freeware programs. We personally use Golly [5] which uses the very fast Hashlife algorithm [6].

The patterns are based on the authors own work and extensive public collections for example Mark D. Niemiec’s syntheses collection [8], Jason Summers collection [14] including Jasons Rake collection and Leithner D. and Rott P. Dieter and Peter’s Gun Collection [4]. Stephen Silvers Life Lexicon is also an invaluable resource [13].

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.

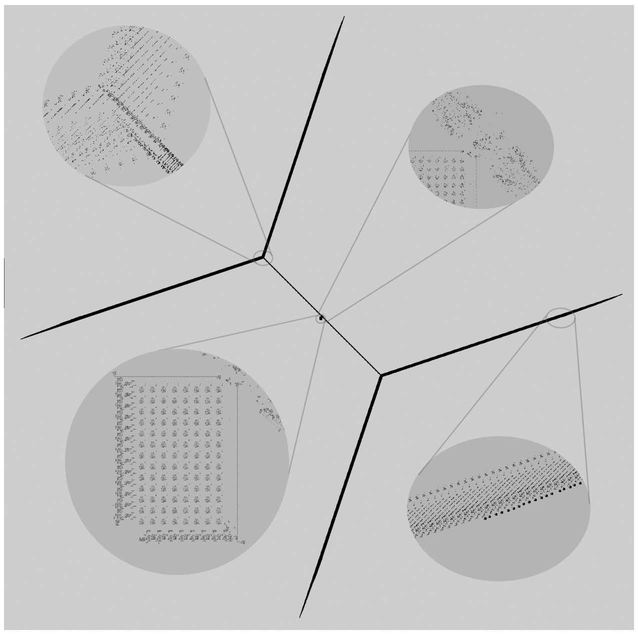


Figure 18. A snapshot of the fully universal Turing machine. This snapshot was taken after the stacks have been programmed and before running the program.

The work on Turing machines was greatly speeded up with the aid of a Turing Machine Simulator by S. Britton [3].

REFERENCES

[1] Boozer D. *Unary Multiplication Turing Machine* `http://www.its.caltech.edu/~boozer/turing/turing-machines.html`.

[2] Berlekamp E. R., Conway John Horton, Guy R.K.*Winning Ways for your Mathematical Plays* (2nd ed.). A K Peters (2001 2004).

[3] Britton S. *Java Applet Turing Machine Simulator* `http://ironphoenix.org/tril/tm/`.

[4] Leithner D. and Rott P. *Dieter and Peter's Gun Collection* `http://entropymine.com/jason/life/`.

[5] Golly *An open source, cross-platform application for exploring Conway's Game of Life and other cellular automata* `http://golly.sourceforge.net/`.

[6] Hashlife *an algorithm for computing the long-term fate of a given starting configuration in Conway's Game of Life and related cellular automata* `http://en.wikipedia.org/wiki/Hashlife`.

[7] *Game of Life Information* `http://pentadecathlon.com`.

[8] Niemiec M. D. *Mark D. Niemiec's Life Page - syntheses* `http://home.interserv.com/~mniemiec/lifepage.htm`.

[9] Rendell P. *Conway's Game of Life Turing Machine* `www.rendell-attic.org/gol`.

[10] Rendell P. *Turing universality of the game of life* (Collision-based computing Pages: 513 - 539 2001).

[11] Rendell P. *A Simple Universal Turing Machine for the Game of Life Turing Machine* (Game of Life Cellular Automata: Pages 519 - 545 2010).

[12] Rendell P. *A Fully Universal Turing Machine in Conway's Game of Life* `http://rendell-attic.org/gol/fullutm/index.htm`.

[13] Silver S. *Stephen Silver's Life Lexicon* `http://www.argentum.freeserve.co.uk/lex_home.htm`.

[14] Summers J. *Jason Summers' pattern collections* `http://entropymine.com/jason/life/`.

Authorized licensed use limited to: CALIF STATE UNIV NORTHRIDGE. Downloaded on October 23,2024 at 18:54:58 UTC from IEEE Xplore. Restrictions apply.