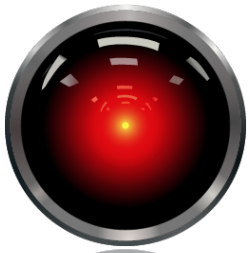


INTELLIGENT COMPUTING

(COMPUTATIONAL INTELLIGENCE)
Finite State Automata & Artificial Life (Cellular Automata)

S. Sheridan

FINITE STATE AUTOMATA (FSA)

- A finite-state automaton (FSA) (plural: automata) or finite-state machine, or simply a state machine, is a mathematical model of computation used to design both computer programs and sequential logic circuits.
- An FSA is an abstract machine that can be in one of a finite number of states. The machine is in only one state at a time; the state it is in at any given time is called the current state.
- A change from one state to another can be initiated by a triggering event or condition; this is called a transition. A particular FSA is defined by a list of its states, and the triggering condition for each transition.

FINITE STATE AUTOMATA (FSA)

- The behaviour of state machines can be observed in many devices in modern society which perform a predetermined sequence of actions depending on a sequence of events with which they are presented.
- Simple examples are vending machines which dispense products when the proper combination of coins is deposited, elevators which drop riders off at upper floors before going down, traffic lights which change sequence when cars are waiting, and combination locks which require the input of combination numbers in the proper order.
- Finite-state automata can model a large number of problems, among which are electronic design automation, communication protocol design, language parsing and other engineering applications. FSAs have also been used extensively in computer games to control non player characters.

FINITE STATE AUTOMATA (FSA)

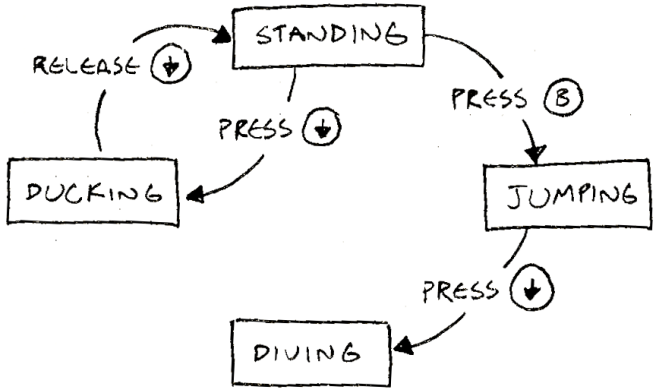
- Finite state automaton consist of 4 main elements:
  1. A set I called the input alphabet
  2. A set S of states that automation can be in
  3. A designated state S0, the initial state
  4. A next state function N: $S_t \rightarrow S_{t+1}$, that assigns a next state to each ordered pair consisting of a current state and a current input
- A finite state automaton must have an initial state which provides a starting point, and a current state which remembers the product of the last state transition. The best way to visualise a FSA is to think of it as a flow chart or a directed graph of states.

FINITE STATE AUTOMATA EXAMPLE

- Consider working on a side-scrolling platform game. Your job as a programmer is to implement the player avatar. This means making the avatar respond to user input. For example, push the **B** button and the avatar should jump.
- The avatar might have the following states **STANDING**, **DUCKING**, **JUMPING**, and **DIVING**.
- Sounds simple doesn't it? If you have ever attempted to code such logic you will know that it can escalate in complexity very quickly.
- For example, the avatar should not be able to jump while already in the air, it should not be able to duck while jumping but should be able to execute an air attack by pressing **down** in the middle of a jump.

FINITE STATE AUTOMATA EXAMPLE

- There are two approaches one might consider here. Dive in and start writing conditional blocks of code and try to make sure you have handled all eventualities. Or stand back and draw a diagram to model the avatar behaviour.



FINITE STATE AUTOMATA EXAMPLE

- The directed graph shown on the previous slide is really a FSA diagram that can be interpreted in order to define the possible states and transitions that are required in order to code our avatar.

Current State	Input	Next State	Output
Standing	Press **B**	Standing	Jump
Standing	Press ⬇	Standing	Duck
Ducking	Release ⬇	Standing	Stand
Jumping	Press ⬇	Standing	Dive

EMERGENT BEHAVIOUR

- While the advantages of using FSAs to model complex logic can be easily appreciated, they are hardly groundbreaking systems by themselves. What is interesting to us is what happens when many FSAs interact with their environment and neighbours. It seems that complex behaviour can emerge from systems that have very simple rules.
- The basic ingredients for emergence seem to be:

1. A simple memory (current state)
2. A simple set of interaction rules (neighbours/environment)
3. A population of embodied individuals (FSAs)

EMERGENT BEHAVIOUR

- Here are some opinions on emergent behaviour expressed by AI researchers;
  - Emergence is "the appearance of novel properties in whole systems" (Moravec 1988)
  - "Global functionality emerges from the parallel interaction of local behaviours" (Steels 1990)
  - "Intelligence emerges from the interaction of the components of the system" (Brooks 1991)
  - "Emergent functionality arises by virtue of interaction between components not themselves designed with the particular function in mind" (McFarland & Bosser 1993)
  - "They are a consequence underlying the complexity of the world in which the robotic agent resides and the additional complexity of perceiving that world" (Arkin 1998)
  - "....the arising of novel and coherent structures, patterns and properties during the process of self-organisation in complex systems"(Goldstein 1999)
  - ".... where the agent appears to do something fairly complex, but is really just the result of interaction between simple modules" (Murphy 2000)
  - "Behaviours serve as the basic building blocks for robotic actions, and the overall behaviour of the robot is emergent" (Murphy 2000)
  - "Emergence is ubiquitous" (de Haan 2007)

EMERGENT BEHAVIOUR

- Some advantages of emergent behaviour when compared to directly programmed behaviour;
  - No additional structure is needed inside an agent to get additional capabilities. Therefore, we do not need any special explanations on how the behaviour may come about.
  - Emergent behaviour tends to be more robust because it is less dependent on accurate sensing or action and because it makes less environmental assumptions.
  - Emergent behaviour is parallel and distributed by its very nature. This makes it very attractive as an approach to solving complex problem because solutions can be realised on parallel/distributed architectures.

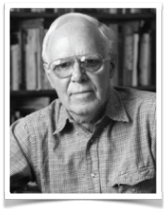
EMERGENT BEHAVIOUR

http://curiosity.discovery.com/question/emergent-behavior

ARTIFICIAL LIFE (CELLULAR AUTOMATA)

- An example of *artificial life* can be found in games such as the Game of Life, which was originally created by the mathematician John Horton Conway and introduced to the larger community by Martin Gardener in Scientific America (1970,1971).


John Horton Conway (Born 1937)


Martin Gardener (1914 - 2010)

- In this game, the birth, survival, or death of individuals is a function of their own state and that of their near neighbours. Typically, a small number of rules, usually 3 or 4 are sufficient to define the game. In spite of this simplicity, experiments with the game of life have shown it to be capable of evolving structures of extraordinary complexity.

ARTIFICIAL LIFE (CELLULAR AUTOMATA)

- Work in artificial life or a-life, has simulated the conditions of biological evolution through the interactions of finite-state automata, complete with sets of states and transition rules.

- These automata are able to accept information from outside themselves, in particular, from their closest neighbours. Their transition rules include instructions for *birth*, *continuing life*, and *dying*.

- When a population of such automata is set loose in a domain and allowed to act as parallel asynchronous cooperating agents, we sometimes witness the evolution of seemingly independent '**life forms**'.

ARTIFICIAL LIFE (CELLULAR AUTOMATA)

- The output of a CA is a function of their present state and the states of their near neighbouring CAs. Thus the state of a CA at time (t + 1) is a function of its state and the state of its neighbours at time (t).

- It is through these interactions with their neighbours that collections of cellular automata can achieve much richer behaviours than simple individual automata would achieve.

- This emergence of complexity is one of the most fascinating properties of populations of CAs. Because the output of a state is a function of its neighbouring states, we describe the evolution of a set of neighbouring CAs as society-based adaptation.

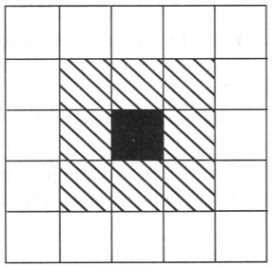
ARTIFICIAL LIFE (CELLULAR AUTOMATA)

- There need not even be an explicit evaluation of fitness of individual members. Fitness results from interactions in the population, interactions that may lead to the "death" of individual automata.

- Fitness is implicit in the survival of individuals from generation to generation. Learning among CAs is typically unsupervised; as occurs in natural evolution, adaptation is shaped by the actions of other, co-evolving members of the population.

THE "GAME OF LIFE"

- Consider the simple two-dimensional grid shown below. Here we have one square occupied, in black, with its eight nearest neighbours indicated by grey shading. The board is transformed over time periods, where the state of each square at time t + 1 is a function of its own state and the state of these indicated nearest neighbours.



THE "GAME OF LIFE"

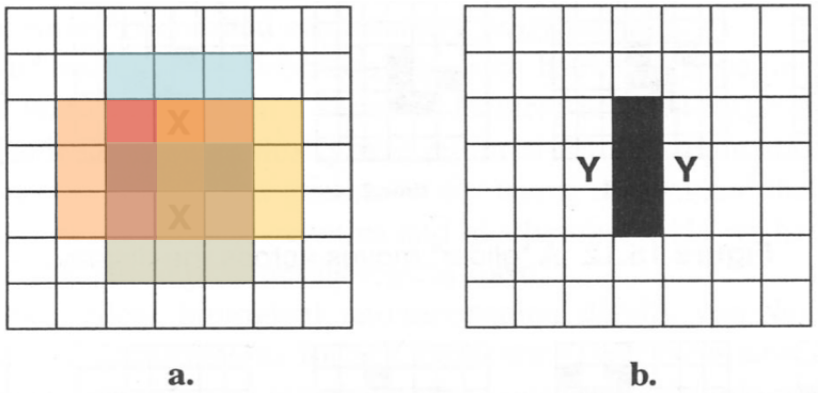
- These simple rules can drive evolution in the game:

1. Any live cell with fewer than two live neighbours dies, as if caused by **under-population**.
2. Any live cell with two or three live neighbours lives on to the next generation, **survival**.
3. Any live cell with more than three live neighbours dies, as if by **overcrowding**.
4. Any dead cell with exactly three live neighbours becomes a live cell, as if by **reproduction**.

THE "GAME OF LIFE"

- One interpretation of these rules is that, for each generation or time period, life at any location, that is, whether or not the square is occupied and has a state value 1, is a result of its own as well as its neighbours life during the previous generation.
- Specifically, too dense a population of surrounding neighbours (more than three) or too sparse a neighbouring population (less than two) at any time period will not allow life for the next generation.
- Consider, for example, the state of life for the diagram a) on the next slide. Here exactly two squares, indicated by an X, have have exactly three occupied neighbours. At the next life cycle diagram b) will be produced.

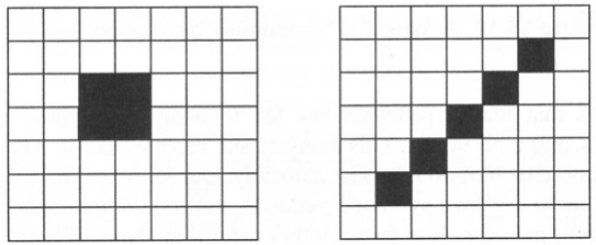
THE "GAME OF LIFE"



a. b.

THE "GAME OF LIFE"

- What will happen to these patterns at the next time cycle?

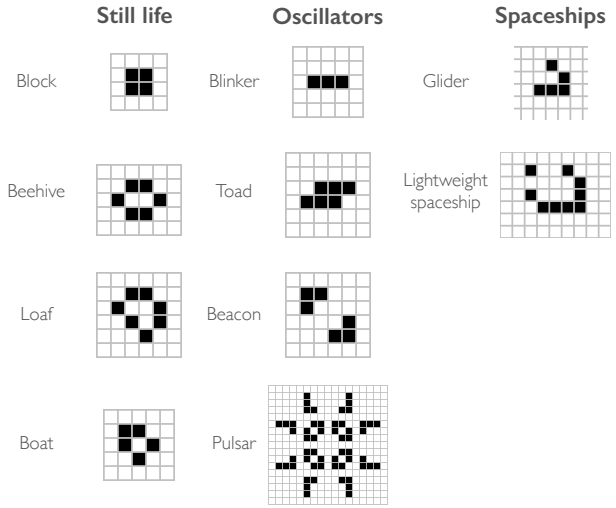


a. b.

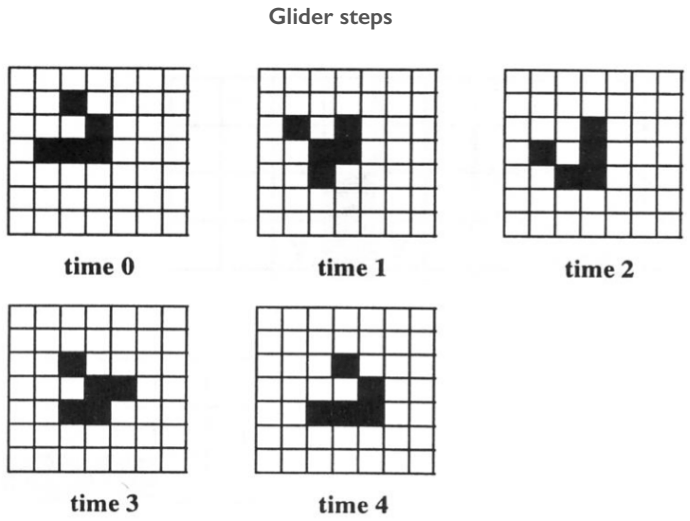
THE "GAME OF LIFE"

- Poundstone (1985) describes the extraordinary variety and richness of the structures that can emerge in the game of life, such as gliders, patterns of cells that move across the world through repeated cycles of shape changes.
- Because of their ability to produce rich collective behaviours through interactions of simple cells, cellular automata have proven a powerful tool for studying the mathematics of the emergence of life from simple, inanimate components.
- Artificial life is defined as life made by human effort rather than by nature. As can be seen in the previous example, artificial life has a strong bottom-up flavour.

THE "GAME OF LIFE"



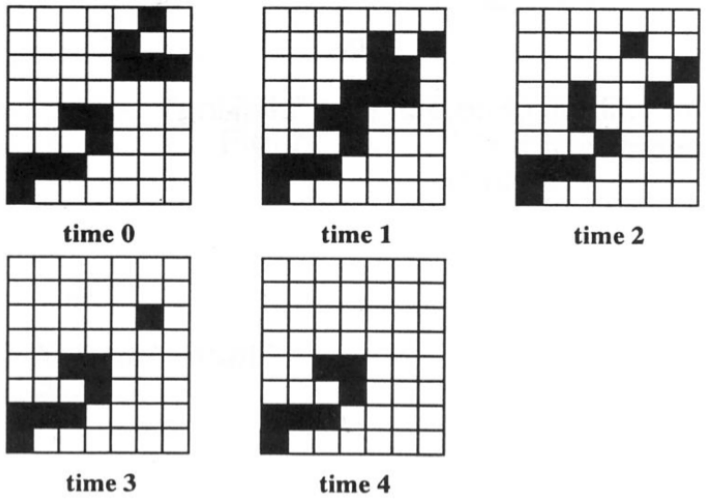
THE "GAME OF LIFE"



THE "GAME OF LIFE"

- An interesting aspect of the game of life is that entities such as the glider persist until they interact with other members of their society. What happens then can be difficult to understand and predict.
- For example, the time series shown on the next slide shows a situation where two gliders emerge and engage. After four time periods, the glider moving down and to the left is "consumed" by the other entity.

THE "GAME OF LIFE"



THE "GAME OF LIFE"

- It is interesting to note that our ontological descriptions, that is, our use of terms such as "entity", "blinking light", "glider", "consumed", reflects our own biases on viewing life forms and interactions, whether artificial or not.
- It is very human of us to give names to regularities as they emerge within our social structures.
- For more information see http://www.bitstorm.org/gameoflife/