6. Evolution, Co-evolution (and Artificial Life) Part 1

Modelling Social Interaction in Information Systems

http://davidhales.com/msiis

David Hales, University of Szeged

dave@davidhales.com

Summary

- How can agents decide what to do (given some goals, knowledge and possible actions):
  - Rationality:
    - Individual utility maximisation (game theory, econ.)
    - Collective reasoning (Hobbs, Rawls, social contract)
  - Bounded rationality:
    - heuristics that attempt to achieve a goal (Schelling's segregation, Axelrod's tournament strategies)
  - Evolutionary / Adaptive
    - Individual learning (reinforcement, neural nets)
    - **Social / evolutionary learning (genetic algorithms, genetic programming, co-evolutionary systems, cultural evolution)**

What is evolution?

- It is a very old idea that predates modern science
- It is a theory of change
- Originally applied to human societies and ideas – because people could see these changed over time
- It isn't until recently (fossils etc) that it was realised biological life forms changed over time

Biological evolution

- Evolution in everyday language has come to mean biological evolution
- Darwin did his famous empirical work observing biological organisms
- Biological evolution draws on empirical facts and theoretical models (often mathematical)
- Here we will focus on “abstract evolution” simulated in computer programs

Abstract evolution

- Evolution can be viewed as an algorithmic abstraction that can be used to understand / implement a process of change given:
  - Things that replicate / get copied (**units** of selection)
  - **Variation** in replicators (mutation)
  - Differential **selection** of replicators
- "fitness" means how good a replicator is at replicating (how many copies are made)
- In this context "survival of the fittest" is a tautology

Book: Daniel Dennett (1995) Darwin's Dangerous Idea. Simon & Schuster

Abstract evolution (GA's)

- Genetic algorithms (which I think you know)
- Are an optimisation technique
- Define a space of solutions to a problem
- Code different candidate solutions in an "artificial chromosome" (often a bitstring but not always)
- Use an evolutionary algorithm to adapt solutions towards better (hopefully optimal or good enough) solutions
- Do this through some form of selection, recombination (crossover), mutation and reproduction
- John Holland early 70's, Alan Turing 50's, other earlier thinkers...

Book: Holland, John (1975). Adaptation in Natural and Artificial Systems. Cambridge, MA: MIT Press

Genetic Algorithms

- Initialise a population of (N) random chrom.
- Loop for some (G) number of generations
  - Loop for each chrom.
    - Test chrom. against an objective function f() – award a fitness score
  - End loop solutions
  - Reproduce chrom probabilistically proportionally into the next generation based on fitness score
  - Apply some genetic operator (such as crossover)
  - Mutate reproduced chrom. with small prob. (m)
- End loop generations

Reproduction / Selection

- Many ways to simulate reproduction:
  - Roulette Wheel Selection
  - Tournament Selection
  - Other kinds…
- In general you want an easy to implement and fast method
- That will allow for fitter solutions to tend to increase in the population over time

Roulette Wheel Selection

- Suppose you have a population of chromosomes and each has been allocated a fitness based on f()
- Add up the fitness's of all chromosomes = tf
- Repeat until next generation is full:
  - Generate a random number R in that range 0..tf
  - Select the first chromosome in the population that when all previous fitness's are added - gives you at least the value R
  - Reproduce the selected chrom. Into the next generation
- Hence it is like a roulette wheel where each spot on the wheel (representing a chromosome) is the size of the fitness of the associated chromosome

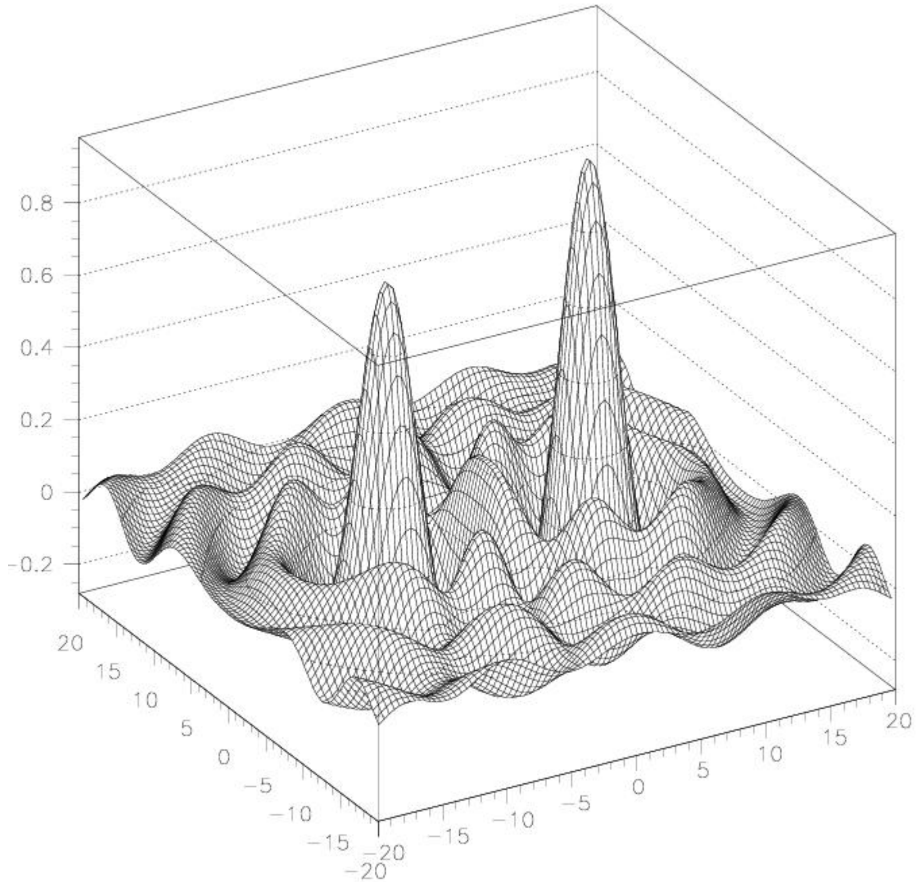
Tournament selection

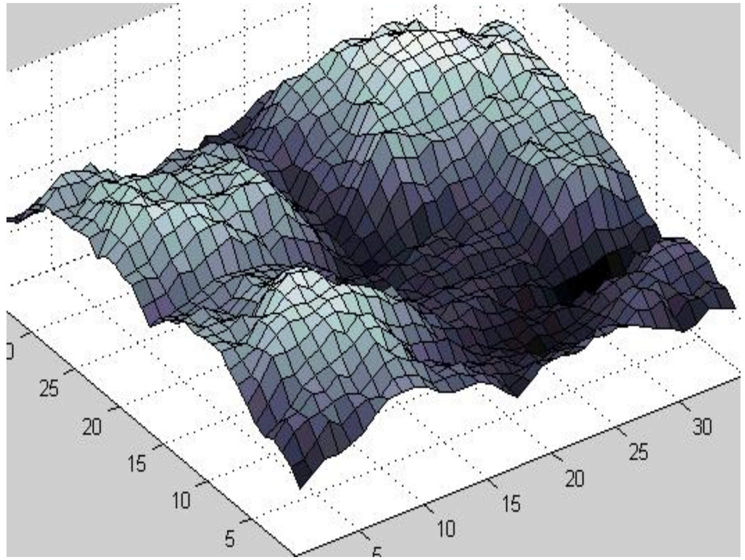
- Many variants but a very simple form is:
- Repeat until next generation is full
  - Select pairs of chromosomes randomly
  - Reproduce the one with the highest fitness
  - Or a random one if they have same fitness

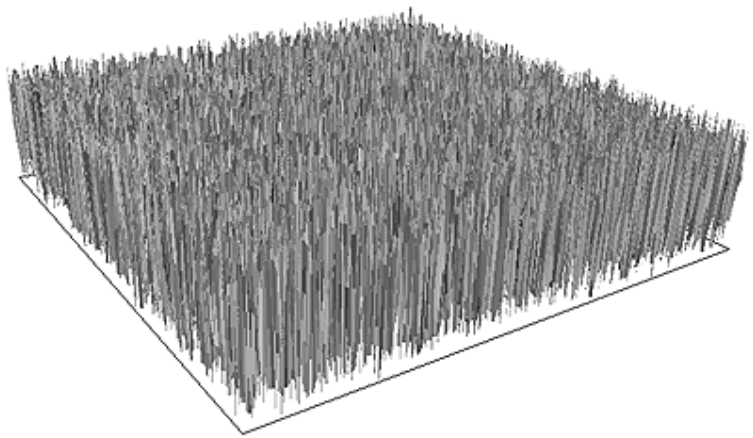
Genetic Algorithms (crossover)

- You have to choose, G, N, m and f()
- Often GA's use **crossover** (recombination) of reproduced chromosomes as well as mutation
- This involves splicing together parts of chromosomes
- Can be compared to sexual reproduction
- 1-point crossover: take two chrom., select a random cut-point and spice together the chrom. of the two parents
- Holland developed "Schema Theory" to understand how various genetic operators (such as crossover) work
- Using GA's for optimisation very much an "art"
- There is no "free lunch" for search problems!

Fitness landscapes







Complex / implicit fitness function

- Can we still use evolutionary algorithms without simple explicit fitness functions?
- Yes, simple way, let a person look at solutions and select some they like better
- Dawkins “bimorphs” (NetLogo model library: biology/evolution/sunflower biomorphs)
- OR somehow let the “world” supply the fitness function – or a simulation of the world
- Evolving robots with “real physics”

Co-evolution fitness functions

- Suppose our solutions are “agents” that must interact socially with each other in a simulated environment to gain fitness
- The fitness of an agent depends on how the other agents behave
- Remember Axelrod’s tournaments?
- To get a score (or fitness) for each algorithm he had to play them off in simulated tournaments
- Since the fitness of any agent is dependent on the other agents in the population
- This is called **co-evolution** because each agent evolves relative to the others rather than optimising an exogenous fixed fitness f()
- In this sense f() takes as inputs all the other agents
- When the agents are strategies in a simple game with known payoffs this relates to **evolutionary game theory**

An evolutionary PD game

- Suppose:
  - agents as 1 bit strategy in the the PD game where 1 = coop and 0 = defect
  - population of N agents initialised at random (0 or 1)
  - Apply an evolutionary algorithm where each generation each agent is randomly paired with some other agent in the population and plays a game of PD
  - Reproduction (roulette wheel) using average payoff from the games as the fitness of each agent
  - Apply some small (m = 0.01) mutation to each reproduced agent that causes it to flip its strategy

Evolving PD strategies

- Initialise population N to random strategies
- Loop some number of generations
  - Loop for each agent (a) in the population
    - Select another agent (b) at random from the population
    - Play PD between (a) and (b) based on their strategies accumulate payoffs in agents
  - End loop for each agent
  - Reproduce a new population of size N probabilistically in proportion to average payoff and apply mutation with probability m
- End loop for number of generations

Note: Random pairing of strategies is sometimes called "mean field" interaction or "homogenous mixing". Reproduction without cross-over is called "asexual reproduction".

Evolving PD strategies

- In this case with simple (pure) PD strategies and mean field mixing…
- Evolution will quickly lead to all defect dominating the population and stay there
- This is called an **Evolutionary Stable Strategy** (or ESS)
- A strategy is an ESS if a population all using it can resist “invasion” by a small number of any other strategy
- All ESS are Nash Equlibria (NE) but not all NE are ESS.
- Hence a link is found between game theory and evolutionary theory which biologists discovered and applied

Book: John Maynard Smith (1982) Evolution and the theory of games. Oxford University Press

Sociobiology

- More generally the application of biological evolutionary approaches to understand social interactions is called Sociobiology
- When it is applied to human social systems it is can be highly controversial
- Critics worry it starts to look like “Social Darwinism” and overlooks the role of culture as the determinant of human social systems and behaviour
- We will not discuss this controversy here but it is worthwhile to be aware of it

Book: E. O. Wilson (1975) Sociobiology: The New Synthesis.

Evolving PD strategies

- More complex strategies can be evolved in this way and analysed to see if they are ESS
- Axelrod noted in his book that tit-for-tat was "collectively stable" (almost an ESS)
- The relationship between ESS, Nash and, say, Pareto efficiency is subtle and complex even in mean field models
- However analysis (not just simulation) can be applied to simple games with a limited number of strategies

Paper: Nowak, Sigmund, Esam (1995) Automata, repeated games and noise. J. Math. Biol. 33: 703-722

Evolution of strategies

- Even if we can calculate some ESS for given a given system this does not necessarily tell us the dynamics (trajectories) that evolution will take from any given starting point
- In simple systems “replicator dynamics” equations can be used to prove things (assuming no mutation!)
- In general, simulation experiments are used to see what happens when it gets complicated

Non-random interactions

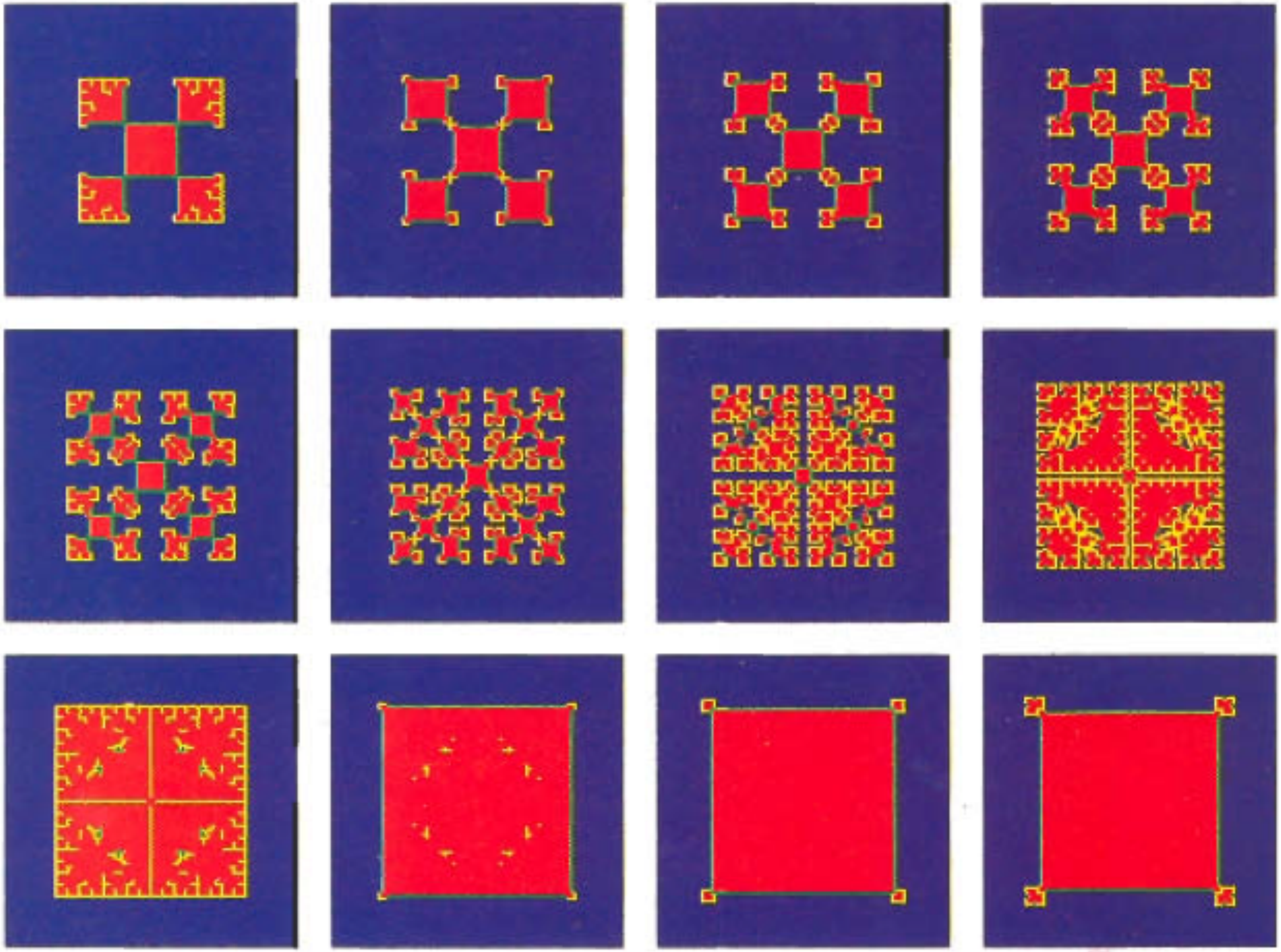
- Many forms of interaction in the “real world” are non-random
- Some work has explored this using a cellular automata (CA) where:
  - Each cell is either coop or defect state
  - Plays PD with each of it’s neighbours (and possibly itself)
  - Copies the the strategy of fittest neighbour (or stays same if it is fittest)
  - Sometimes mutation is used sometimes not

See NetLogo model library/biology/evolution/altruism

Evolving PD on a CA

- In general it has been found that over a broad range of parameters:
  - Cooperation can be sustained
  - Dynamic patterns emerge over time
  - Groups of cooperators and defectors because they are spatially clustered create these interesting dynamics
  - Pretty patterns can be produced
- The argument is that many biological and social phenomena interact in space and this can be a major factor in sustaining the evolution of cooperation

Paper: Nowak, May (1993) The Spatial Dilemmas of Evolution. Int. J. of Bifurcation and Chaos, Vol. 3, No. 1. 35-78



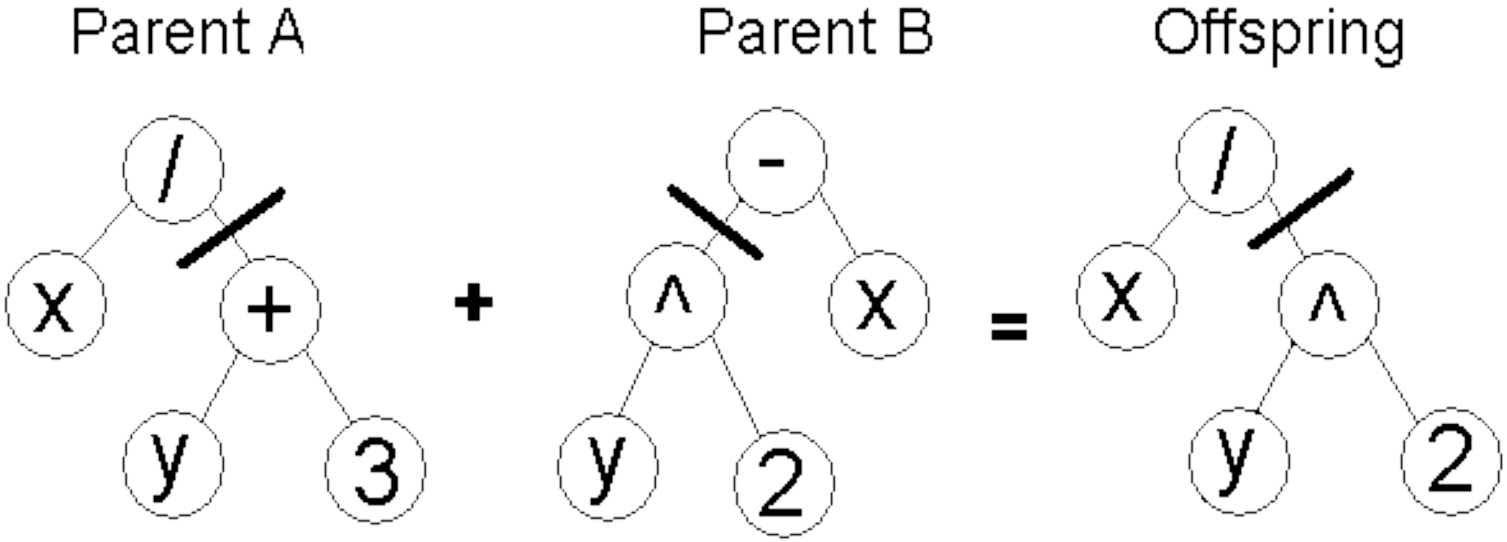
Taken from Nowak and May (1992)

Unknown coding of solution?

- Suppose we don't have simple space of solutions (or strategies) that each "chromosome" can code
- Can we use evolutionary algorithms without a simple coding of the solution space?
- Yes, evolve a computer program directly (or an artificial neural network)
- **Genetic Programming** (GP) uses simple (functional) languages and tree-like crossovers
- Such languages have to be "robust" to mutation and crossover – i.e. not "brittle" (unlike most computer languages where if you change one thing it breaks)
- In general GP are used to evolve small programs (or functions) for optimisation purposes

Book: Koza, J.R. (1992). Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press.

Genetic Programming



It is possible to evolve whole programs like this but only small ones.
Large programs with complex functions present big problems

Readings and Questions

- Readings
  - Flake (1998) Chapter 5 – Adaptation
  - Gilbert et al (2005) Chapter 10 – Learning and Evo. models
- Questions
  - Some claim TFT in the PD is an ESS others say it's not strictly an ESS. Can you think of a simple strategy that could invade a population all paying TFT?
  - What would happen if we allowed agents to move on the grid based, in some way, on game payoff?
  - "All evolution is co-evolution!" Is this true?
  - Why do you think GP struggles to evolve large programs?