Genetic Algorithms and Genetic Programming
Lecture 4

Gillian Hayes

3rd October 2006

School of informatics

Evolving Strategies Using the Canonical GA

- Recap of the Canonical GA
- Variety of applications
- A Function Optimisation Example
- The Prisoner's Dilemma
- Encoding a Strategy
- Experiment 1: static evolution
- Evolution and coevolution
- Experiment 2: changing evaluation
- Summary and conclusions

The Canonical GA

- Representation: fixed-length binary chromosome of length l
- Population: n chromosomes, initially randomly generated
- Fitness: an evaluation function which maps each possible chromosome to a numerical fitness value
- Selection: fitness-proportionate (roulette wheel) selection into an intermediate population
- Breeding: one-point crossover and reproduction
- Mutation: flip bits with low probability
- Stop when fitness of best solution is good enough

Variety of Applications

- Numerical and combinatorial optimisation
- Automatic programming
- Machine learning
- Immune systems
- Population biology
- Financial systems, economics
- Evolutionary robotics
- Artificial life

A Function Optimisation Example

Minimise Rastrigin's Function:

$f(x) = 10 + x^2 - 10\cos(2\pi x),\ -5.12 \leq x \leq 5.12$

- Representation: binary strings

  $x = x_{min} + b(x_{max} - x_{min})/(2^m - 1)$

- So for 8-bit strings

  $x = -5.12 + b(5.12 - -5.12)/(2^8 - 1)$

- If $b = 10011001$ then this represents the integer 153, so

  $x = -5.12 + (153 \times 10.24/255) = 1.024$

- Generate lots of binary strings
- Calculate fitness of each (low value = fit – minimisation)
- Selection, crossover, mutation, reproduction
- Till no improvement possible. $x = 0, f(x) = 0$

The Prisoner's Dilemma (1)

- Two players, no communication
- Each decides to cooperate (**C**) or defect (**D**) (cooperate with each other = not testify against the other person, defect = testify against the other person):
  - **CD** gives a payoff of [0,5]
  - **DC** gives a payoff of [5,0]
  - **CC** gives a payoff of [3,3]
  - **DD** gives a payoff of [1,1]
- Payoff of 5 means don't go to jail, payoff of 0 means go to jail for a long time
- Played repeatedly (iterated prisoner's dilemma)
- Aim: each player must maximise payoff

The Prisoner's Dilemma (2)

- Tournaments organised by Axelrod
- Memory of the last 3 games:
  CC CC CD
- Contestants submitted computer programs
- Best strategy was TIT FOR TAT:
  - Cooperate on the first turn
  - Then do at time $t+1$ whatever the opponent did at time t
  - **CC CC CD** $\rightarrow$ **D**

Encoding a Strategy

- Can a GA evolve strategies for this game?
- Need to know what to do given the last 3 games e.g. **CC CC CC** $\rightarrow$ **D**
- There are 64 possibilities for the last 3 games:
  CC CC CC (case 1)
  CC CC CD (case 2) . . .
  DD DD DD (case 64)
- Encode strategy as a 64-bit string: 101110011. . .
  the bit at position i encodes the response for case i: 1 means cooperate, 0 means defect
- Add 6 bits to encode a hypothetical set of 3 previous games to give a 70-bit string (giving a search space of 1.18×10^{21})

Experiment 1: static evaluation

- Population of 20 random chromosomes
- Play each against 8 typical strategies selected from the Prisoner's Dilemma tournaments (did not include TIT FOR TAT)
- Static environment – search for a specialist individual
- 40 runs of 50 generations each
- Found strategies which scored substantially higher than TIT FOR TAT

An Example

- Original aim: to use biological evolution
- In our systems, static fitness function
- In biology, competition with each other
- Coevolution: different species develop
- Arms races break out between prey/predator, parasite/host, etc.
- No single solution to the problem

Experiment 2: changing evaluation

- Population: as before
- Fitness: play each strategy against the other 19
- Fitness landscape formed by other individuals
- Results in population dynamics
  - The initial strategies play randomly
  - Then uncooperative strategies dominate
  - Then cooperative strategies such as TIT FOR TAT appear that can do well with one another
- The resultant strategies are generalists
- Nature of good solutions depends on opponents: in a population of mostly "Always Defect" and a few "Tit for Tat" the latter will win – do very well against each other and not too badly against Always Defect.

Summary and Conclusions

- Canonical GA is very simple
- Application involves:
  - Deciding on the representation
  - Setting the length of the binary chromosome
  - Implementing a fitness function $f(c_i)$
  - Setting the size of the population
- An amazing variety of applications
- Can use GA to evolve strategies
- Static evaluation can find the best strategy for playing against a fixed set of strategies in PD
- Coevolution can produce generalist strategies which can play well against any other strategy