Complex Systems and Network Science:

Adaptation and Genetic Algorithms

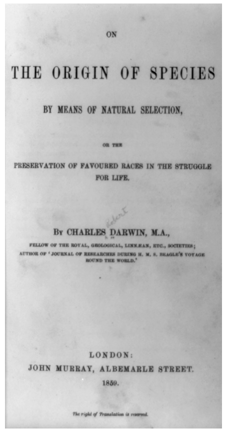
Ozalp Babaoglu
Dipartimento di Informatica — Scienza e Ingegneria
Università di Bologna
www.cs.unibo.it/babaoglu/

Evolution in the real world

- Each cell of a living thing contains *chromosomes* — strings of DNA
- Each chromosome contains a set of *genes* — blocks of DNA
- Each gene largely determines some physical aspect of the organism (like eye color)
- A collection of genes is called the *genotype*
- A collection of aspects is called the *phenotype*

Evolution in the real world

- Reproduction involves recombination of genes from parents and then small amounts of mutation (errors) in copying
- *Evolutionary biology*: organism's genotype largely determines its phenotype and hence its *fitness* — ability to produce new offsprings and propagate its genotype
- Basis for "Darwinian Evolution"
- *Evolutionary game theory*: the fitness of an organism determined by its interactions with other organisms in a population



ON

THE ORIGIN OF SPECIES

BY MEANS OF NATURAL SELECTION,

OR THE

PRESERVATION OF FAVOURED RACES IN THE STRUGGLE FOR LIFE.

Robert

BY CHARLES DARWIN, M.A.,

FELLOW OF THE ROYAL, GEOLOGICAL, LINNÆAN, ETC., SOCIETIES;
AUTHOR OF 'JOURNAL OF RESEARCHES DURING H. M. S. BEAGLE'S VOYAGE ROUND THE WORLD.'

LONDON:
JOHN MURRAY, ALBEMARLE STREET.
1859.

The right of Translation is reserved.

Genetic algorithms: History

- Pioneered by John Holland in the 1970's
- Became popular in the late 1980's
- Based on ideas from "Evolution by Natural Selection"
- Can be used to solve a variety of problems that are not easy to solve using traditional techniques

Genetic algorithms

- Suppose you have a problem
- You don't know how to solve it "algorithmically"
- Can we use a computer to somehow "find" a solution?

A dumb solution

```
Repeat
  Generate a random feasible solution
  Test the solution to evaluate its "goodness"
Until (solution is good enough)
```

Can we use this dumb idea?

- Sometimes — yes:
  - if there are only a few possible solutions
  - and you have enough time
  - then such a method could be used
- For most problems — no:
  - many possible solutions
  - with no time to try them all
  - so this method can not be used

A "less-dumb" idea (GA)

```
Generate a set of random solutions
Repeat
  Test each solution in the set and rank them
  Remove some bad solutions from set
  Duplicate some good solutions
  Make small changes to some of them
Until ("best" solution is good enough)
```

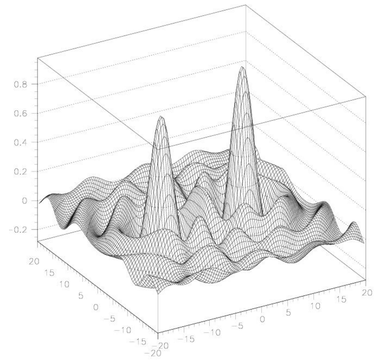
How do you encode a solution?

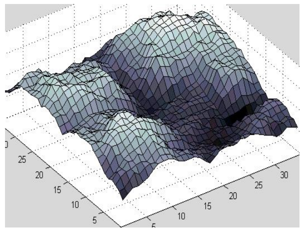
- Obviously this depends on the problem!
- GA's often encode solutions as fixed length "bit strings"
- Each bit represents some aspect of the proposed solution to the problem
- For GA's to work, we need to be able to "test" any string and get a "score" indicating how "good" that solution is

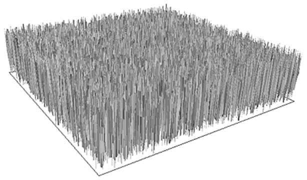
Fitness function, Search space

- Generate a score for each solution based on a function that measures "how good" it is — this is called a *fitness function*
- For a simple fitness function $f(x)$ the search space is one dimensional
- But by encoding several values into the chromosome, many dimensions can be searched e.g. two dimensions $f(x,y)$
- Search space can be visualized as a surface or a *fitness landscape* in which fitness dictates height
- Each possible genotype is a point in the space
- A GA tries to move the points to better places (higher fitness) in the the space

Fitness landscapes







Search space

- The nature of the search space dictates how a GA will perform
- A "random" search space leads to poor performance of GA
- Also GA's can get stuck in local maxima
- Generally, spaces that are "smooth" where small movements get closer to the global optimum are good

Back to the (GA) algorithm

```
Generate a set of random solutions
Repeat
   Test each solution in the set (rank them)
   Remove some bad solutions from set
   Duplicate some good solutions
   Make small changes to some of them
Until (best solution is "good enough")
```

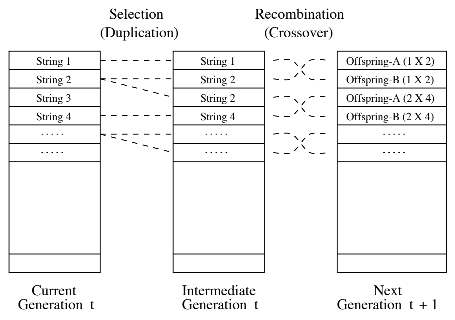
Adding sex — Crossover

- Although it may work for simple search spaces, our algorithm is still very naive
- It relies on random mutations and "asexual reproduction" to find a good solution
- By introducing "sexual reproduction" into the algorithm, we can achieve better results
- This is done by selecting two parents during reproduction and combining their genes to produce offspring

Adding sex — Crossover

- Two high scoring "parent" bit strings (chromosomes) are selected and are *combined* with some probability (crossover rate)
- Producing two new offsprings (bit strings)
- Each offspring may then be changed randomly (mutation)

Outline of a Genetic Algorithm



Selecting parents

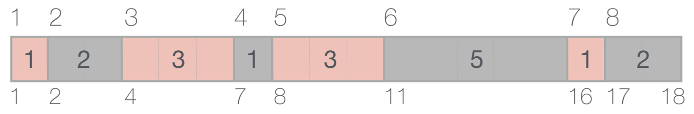
- Many schemes are possible for guaranteeing that individuals with *better scoring* chromosomes are more likely to be selected, thus increasing their likelihood of passing their genes to the next generation
- Score is equated with fitness
- Not *politically correct*!
- "Roulette Wheel" selection is often used

Example population

No.	*Chromosome*	*Score*
1	1010011010	1
2	1111100001	2
3	1011001100	3
4	1010000000	1
5	1000000000	3
6	1001011111	5
7	1010101010	1
8	1011100111	2

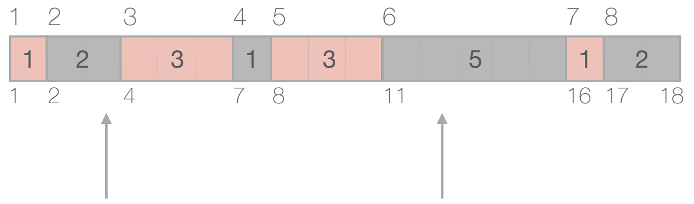
"Roulette Wheel" selection

- Construct a "wheel" with number of wedges equal to the number of chromosomes
- The width of each wedge is proportional to the score of the corresponding chromosome



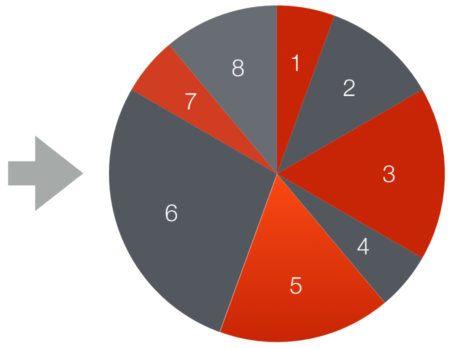
- Mark the summation of the wedge widths starting from 1
- Generate a uniform random number between 1 and the maximum summation value
- Select the chromosome that "covers" the random value

"Roulette Wheel" selection

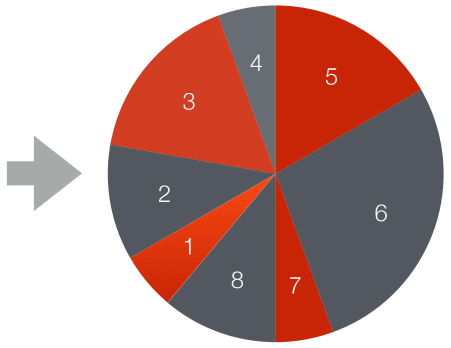


- *rand*(1,18) = 12
- Parent 1 has chromosome 6
- *rand*(1,18) = 3
- Parent 2 has chromosome 2

"Roulette Wheel" selection



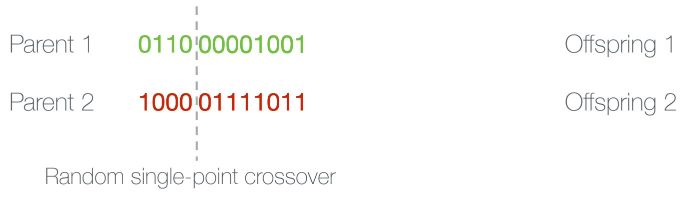
"Roulette Wheel" selection



- The result is the same as spinning a roulette wheel twice and reading off the names of the corresponding chromosomes

Crossover — Recombination

- With some high probability (crossover rate) apply crossover to the parents (typical values are 0.8 to 0.95)



Mutation

- With some small probability called the *mutation rate* (typical values between 0.1 and 0.001) flip some bit in the offspring

	Before mutation	After mutation
Offspring 1	0110 01111011	0110 01110011
Offspring 2	1000 00001001	1001 00001001

Back to the (GA) Algorithm

```
Generate a population of random chromosomes
Repeat (generation)
   Calculate fitness of each chromosome
   Repeat
      Select pairs of parents with roulette selection
      Generate offsprings with crossover and mutation
   Until (a new population has been produced)
Until (best solution is good enough)
```

Many variants of GA

- Different kinds of selection (not roulette)
  - Tournament
  - Elitism, etc.
- Different recombination
  - Multi-point crossover
  - 3 way crossover etc.
- Different kinds of encoding other than bit strings
  - Integer values
  - Ordered set of symbols
- Different kinds of mutation

Many parameters to set

- Any GA implementation needs to decide on a number of parameters: Population size (N) , mutation rate (m), crossover rate (c)
- Often these have to be “tuned” based on results obtained — no general theory to deduce good values
- Some common values: $N = 50$, $m = 0.05$, $c = 0.9$

Evolving a GA solution to the Iterated Prisoner’s Dilemma

- “Iterated PD” (IPD) — the same two players play PD for many rounds instead of just a single round
- Players have complete knowledge of their own and their opponent’s previous moves (and can use this knowledge to select their next strategy)
- Iterated games are interesting because they allow for “revenge” or “retribution” which was not possible in one-shot games

Evolving a GA solution to the Iterated Prisoner's Dilemma

- In 1980, Robert Axelrod organized a 200-round IPD tournament and solicited entries (as programs) from scientists in different disciplines from all over the world
- Axelrod received 14 submissions to his tournament
- Payoff matrix used by Axelrod

		B	
		Deny	*Confess*
A	*Deny*	3, 3	0, 5
	Confess	5, 0	1, 1

Evolving a GA solution to the Iterated Prisoner's Dilemma

- Some simple strategies plausible for IPD:
  - *Always confess* (ALL-C): Cannot be exploited and impossible to win against
  - *Always deny* (ALL-D): Does well against "nice" strategies but can be exploited by ALL-C
  - *Random* (RAND): Randomly select "confess" or "deny"

		B			
		ALL-D	*RAND*	*ALL-C*	*Average*
A	*ALL-D*	3.0	1.5	0.0	1.5
	RAND	4.0	2.25	0.5	2.416
	ALL-C	5.0	3.0	1.0	3.0

Expected one round score for *A* after *A* versus *B*

Evolving a GA solution to the Iterated Prisoner's Dilemma

- ALL-C does the best in terms of average score
- If everyone adopted ALL-C, the population as a whole would achieve a low score (1.0) when in fact a higher global score is possible (3.0) by everyone adopting ALL-D
- ALL-C optimizes local fitness (at the expense of global fitness) while ALL-D optimizes global fitness (at the expense of local fitness)
- What is missing in all these simple strategies is *memory*
- Of the 14 submissions, the shortest (and simplest) one performed the best — TFT (*Tit-for-Tat*)
- TFT: be "nice" at first and then do exactly what you opponent did in the previous round

Evolving a GA solution to the Iterated Prisoner's Dilemma

- Axelrod repeated the tournament after publishing that TFT was the winner of the first edition
- 63 submissions, some of them extremely complicated, were received for the second edition
- TFT again resulted the winner
- Can GA be used to "evolve" good strategies for playing IPD?
- How to encode a strategy as a string

Evolving a GA solution to the Iterated Prisoner's Dilemma

- Suppose each player remembers only one previous game
- There are four possibilities for the previous game:
  - *CC*
  - *CD*
  - *DC*
  - *DD*
- A *strategy* can then be formulated as a rule specifying an action for each one of these four cases
- For example, TFT could be formulated as
  - if *CC* then ***C***
  - if *CD* then ***D***
  - if *DC* then ***C***
  - if *DD* then ***D***

Evolving a GA solution to the Iterated Prisoner's Dilemma

- If the cases are ordered in this canonical way, a strategy can be encoded as a string (chromosome) of 4 letters: ***CDCD***
- Axelrod considered strategies that remembered *three* previous games, resulting in 64 possibilities:
  - *CC CC CC*
  - *CC CC CD*
  - *CC CC DC*
  - ⋮
  - *DD DD DC*
  - *DD DD DD*

Evolving a GA solution to the Iterated Prisoner's Dilemma

- A strategy can then be encoded as a 64-letter chromosome
- Axelrod added 6 extra letters (resulting in 70-letter chromosomes) to encode 3 hypothetical previous games to be used for deciding the action of first game
- Initial population of 20 such strategies
- Axelrod selected 8 representative strategies among the submissions (not including TFT) to define an "environment" for computing fitnesses
- Each individual in the population played iterated games with each of the 8 fixed strategies and the fitness was computed as the average score over all the played games

Evolving a GA solution to the Iterated Prisoner's Dilemma

- Axelrod performed 40 different runs of 50 generations each (resulting in only 20×50=1000 strategies to be tested for each run rather than the 2^{70} possible strategies)
- Most of the evolved strategies were similar to TFT — punish selfishness and reciprocate cooperation
- Some of the evolved strategies performed *better* than TFT
- Does it mean that they were better than human-designed strategies?
- The performance of a strategy depends very much on its environment — against the 8 fixed strategies they were better
- But TFT is a "generalist" and performs well even when we do not have any knowledge of the environment

Evolutionary biology

- Evolutionary biology based on the idea that an organism's genes determine its physical characteristics (phenotype), hence its fitness
- Organisms that are more fit will tend to reproduce more offsprings, causing genes that provide better fitness to increase in the population
- Fitter genes tend to win over time because they provide higher rates of reproduction

Evolutionary game theory

- Basic idea of *evolutionary game theory* is that the fitness of an individual cannot be measured in isolation but has to be evaluated in the context of the full population in which it lives
- An organism's characteristics and behaviors (determined by its genes) are like its strategy in a game
- Its *fitness* is its payoff and this payoff depends on the strategies (characteristics, behaviors) of the organisms with which it interacts

Evolutionary game theory — Example

- Consider a species of beetle
- Beetle's fitness in a given environment determined by its ability to *find* food and *use* the nutrients from food effectively
- Suppose a *mutation* causes some beetles to grow much larger bodies
- Population now has two types of beetles — *small* and *large*
- Large beetles are better at claiming a bigger share of the food but their larger bodies require diverting more nutrients from the food they eat

Evolutionary game theory — Example

- When two beetles compete for food, the following outcomes are possible
  - Two beetles of the same size share the food equally
  - Large beetle against a small beetle gets the majority of food
  - Large beetles experience smaller benefit from a given quantity of food
- The *body-size game*

		Beetle 2	
		Small	*Large*
Beetle 1	*Small*	5, 5	1, 8
	Large	8, 1	3, 3

- Note that the beetles are not choosing a size — each is "hard-wired" to play one of these strategies

Evolutionarily stable strategies

- Notion of *Nash Equilibrium* will be replaced with that of an *evolutionarily stable strategy* — a (genetically-determined) strategy that tends to persist once it becomes prevalent
- *Fitness of an organism in a population* is the expected payoff it receives from an interaction with a random member
- Say *strategy T invades a strategy S at level x* for some small positive number x if a fraction x of the underlying population uses T and a fraction $1-x$ of the underlying population uses S
- Say *strategy S is evolutionarily stable* if there is a small positive number y such that when any other strategy T invades S at any level $x<y$, the fitness of an organism playing S is strictly greater than the fitness of an organism playing T

Evolutionarily stable strategies

- In the body-size game, is the strategy *Small* evolutionarily stable?
- Suppose "*Large* invades *Small*" — for some small positive number x, fraction x of the population uses *Large* and a fraction $1-x$ of the population uses *Small*
- Expected payoff to a small beetle in a random interaction is $\phi(S) = 5(1-x) + 1 \cdot x = 5-4x$
- Expected payoff to a large beetle in a random interaction is $\phi(L) = 8(1-x) + 3 \cdot x = 8-5x$
- For small x, the fitness of small is *not* strictly greater than the fitness of large ($\phi(L)>\phi(S)$), thus strategy *Small* is not evolutionarily stable

Evolutionarily stable strategies

- In the body-size game, is the strategy *Large* evolutionarily stable?
- Suppose "*Small* invades *Large*" — for some small positive number x, fraction x of the population uses *Small* and a fraction $1-x$ of the population uses *Large*
- Expected payoff to a large beetle in a random interaction is $\phi(L) = 3(1-x) + 8 \cdot x = 3+5x$
- Expected payoff to a small beetle in a random interaction is $\phi(S) = (1-x) + 5 \cdot x = 1+4x$
- In this case, for small x, we can verify $\phi(L)>\phi(S)$, thus concluding that strategy *Large* is evolutionarily stable (large beetles can drive out the small ones)

Evolutionarily stable strategies

- If we know that the large-body-size mutation is possible, we should expect to see populations of large beetles in the wild, rather than populations of small ones
- The notion of *evolutionary stability* has predicted a strategy for the *population*
- Similar to predicting outcomes for games among rational *players* based on the notion of *Nash Equilibrium*
- Note that the fitness of each organism in a population of small beetles is 5 which is larger than that among large beetles (3)
- Analogous to the athletes doping (PD) game — here the beetles are engaged in an arms race (competing for food)

Evolutionarily stable strategies

- Whereas to use drugs was a dominant strategy in PD, here it's evolutionary forces (over multiple generations) that achieve a similar effect — large beetles benefit at the expense of the small ones
- Note that in the beetles example, evolution by natural selection is causing the fitness of the organisms to *decrease* over time
- Appears to contradict our belief that natural selection is fitness-increasing
- Natural selection increases the fitness of individual organisms in a *fixed* environment
- If the environment changes to become more hostile to organisms (increased fraction of large beetles), then their fitness could *decrease*