The UK's European university

University of Kent

# CO656 / COMPUTATIONAL INTELLIGENCE IN BUSINESS, ECONOMICS AND FINANCE

**Fernando Otero**
febo@kent.ac.uk
cs.kent.ac.uk/people/staff/febo

# housekeeping

- sample solutions to class exercises will be available on Moodle a week after the class
  - try the tasks first!
- quizzes are available on Moodle for you to check your understanding
  - there are no marks associated with them
  - automatic feedback

# outline:

1. a little bit more on GAs

2. another GA example

# why GA works?

- many random choices
- …but still we see improvements (evolution)

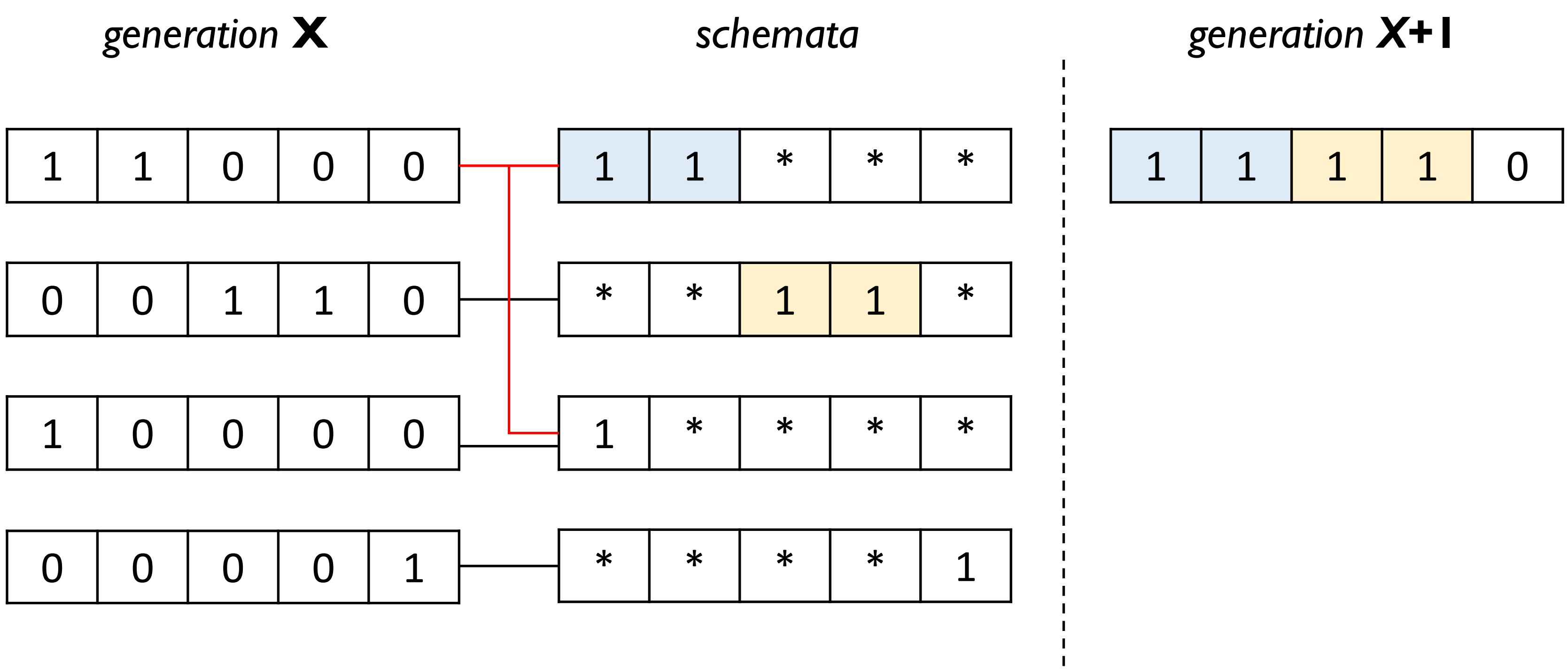
- **stochastic** but different than random search
  - selection is based on fitness
  - … therefore search is biased towards good regions of the search space
- need for multiple runs to estimate the "true" performance of a GA

# why GA works?

- evolution preserves good building blocks
  - blocks of genes in a chromosome
- building blocks are propagated in the population by the fitness-based selection
  - crossover of 2 good individuals with different building block will likely result in an even better offspring
- there are theories that try to characterize the evolution of a GA
  - schema theorem and building block hypothesis
  - …but there is not a universally accepted theory

# building block hypothesis:

- better solutions are created from the best **partial** solutions of past generations



# observations:

- you might have noticed that the best fitness of the population can **fluctuate** – e.g., be lower than the best fitness of the previous generation
- roulette wheel selection can be **affected** if you have extreme fitness values – e.g., one individual having a fitness value much higher than the rest of the population
- more options of genetic operators?

# elitism:

- **best** individual of a population is guaranteed to be carried over to the next
- solution quality will **not** decrease from one generation to the next

# tournament selection:

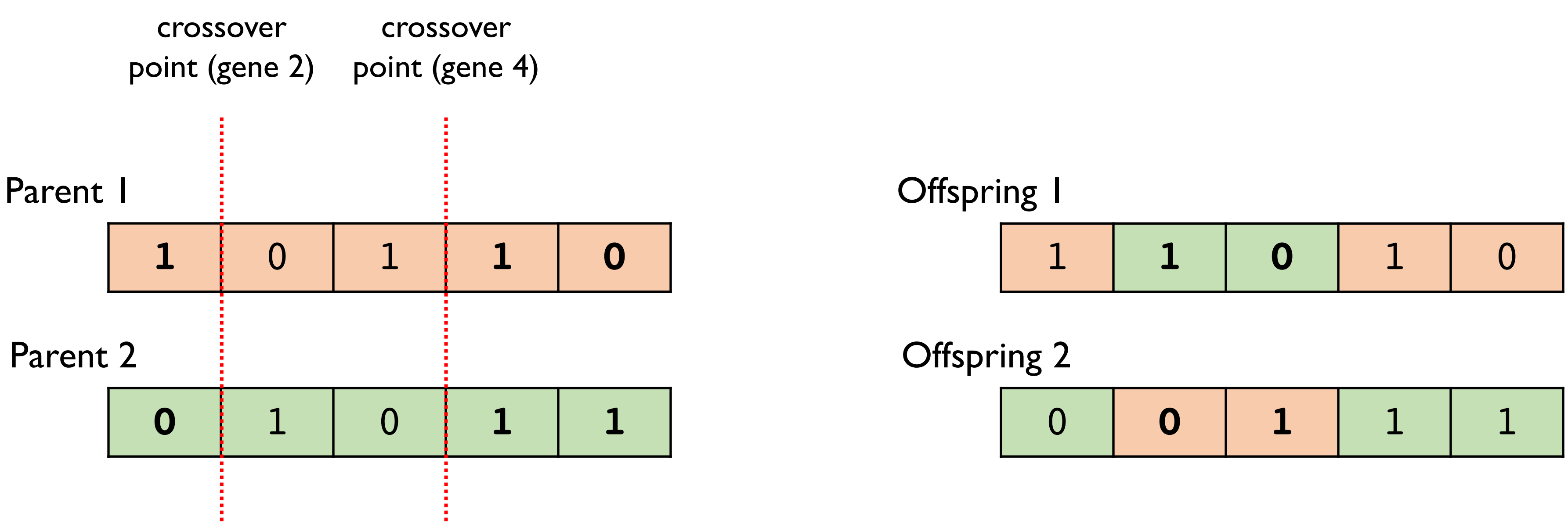
- small **subset** of *k* individuals is chosen at random
- best individual in this set is selected (tournament winner)
- *k* = tournament size (user-specified parameter)
- easy to control the selection pressure – the higher the value of *k*, the higher is the selective pressure

**what happens if *k* = 1?**

**what happens if *k* = population size?**

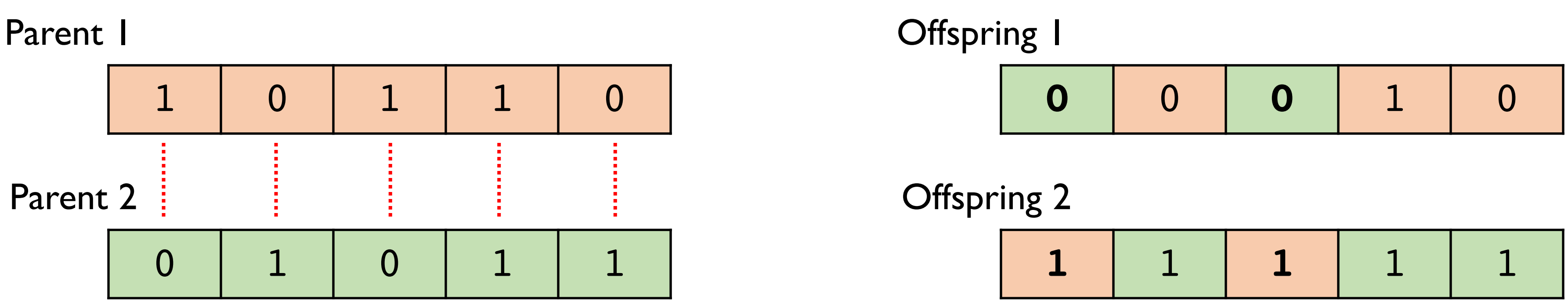
# two-point crossover:

- same principle as the one-point variation, but uses 2 crossover points



# uniform crossover:

- combines genes sampled uniformly from the 2 parents
  - each gene is subject to crossover subject to a probability
  - avoids positional bias



# bit string mutation:

- each gene has a probability of $\frac{1}{length}$ of being mutated

Parent 1

| | | | | |
|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 |

Offspring 1

| | | | | |
|---|---|---|---|---|
| 1 | **1** | **0** | 1 | **1** |

# GA examples:

- Goldberg's book (1989)
    - real-value encoding

# GA example:

- **toy** problem: finding the maximum value of the function $\mathbf{x^2}$ in the interval [0..31]
- individual encoding: five bits representing x in [0..31]
- fitness function: $x^2$ (the larger the fitness, the better the individual)
    - decode 5-bits (genotype) and then compute $x^2$ (phenotype)

# GA example:

- roulette wheel selection
- genetic operators:
  - one-point crossover
  - point mutation
- termination criteria:
  - optimal solution found
  - maximum number of generations

# GA example:

- initial population (randomly generated):

| | | | | | |
|---|---|---|---|---|---|
| 1: | 0 | 1 | 1 | 0 | 1 |
| 2: | 1 | 1 | 0 | 0 | 0 |
| 3: | 0 | 1 | 0 | 0 | 0 |
| 4: | 1 | 0 | 0 | 1 | 1 |

# GA example:

- measuring the fitness of each individual in the population:

| | | | | | | $x$ | $x^2$ (fitness) | % of total fitness |
|---|---|---|---|---|---|---|---|---|
| 1: | 0 | 1 | 1 | 0 | 1 | 13 | 169 | 14.4 |
| 2: | 1 | 1 | 0 | 0 | 0 | 24 | 576 | 49.2 |
| 3: | 0 | 1 | 0 | 0 | 0 | 8 | 64 | 5.5 |
| 4: | 1 | 0 | 0 | 1 | 1 | 19 | 361 | 30.9 |

**decoding individual 1:** $0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = \mathbf{13}$

# GA example:

- suppose the selected individuals are:
  - one copy of individual 1
  - one copy of individual 4
  - two copies of individual 2

  (individual 3 was not selected)
- selected individuals undergo crossover
  - user-defined probability: about 90% - 95%
- they can also undergo mutation
  - user-defined probability: about 1% - 5%

  (lower mutation rate since in nature most mutations are harmful)

# GA example:

- one-point crossover

crossover of individuals 1 and 2

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1: | 0 | 1 | 1 | 0 | \| | 1 | | 0 | 1 | 1 | 0 | \| | 0 |
| 2: | 1 | 1 | 0 | 0 | \| | 0 | | 1 | 1 | 0 | 0 | \| | 1 |

crossover of individuals 2 and 4

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1: | 1 | 1 | \| | 0 | 0 | 0 | | 1 | 1 | \| | 0 | 1 | 1 |
| 2: | 1 | 0 | \| | 0 | 1 | 1 | | 1 | 0 | \| | 0 | 0 | 0 |

# GA example:

population at
generation 0

| | | | | | | x | $x^2$ (fitness) |
|---|---|---|---|---|---|---|---|
| 1: | 0 | 1 | 1 | 0 | 1 | 13 | 169 |
| 2: | 1 | 1 | 0 | 0 | 0 | 24 | 576 |
| 3: | 0 | 1 | 0 | 0 | 0 | 8 | 64 |
| 4: | 1 | 0 | 0 | 1 | 1 | 19 | 361 |
| | | | | | | Average: | 293 |
| | | | | | | Maximum: | 576 |

population at
generation 1

| | | | | | | x | $x^2$ (fitness) |
|---|---|---|---|---|---|---|---|
| 1: | 0 | 1 | 1 | 0 | 0 | 12 | 144 |
| 2: | 1 | 1 | 0 | 0 | 1 | 25 | 625 |
| 3: | 1 | 1 | 0 | 1 | 1 | 27 | 729 |
| 4: | 1 | 0 | 0 | 0 | 0 | 16 | 256 |
| | | | | | | Average: | 439 |
| | | | | | | Maximum: | 729 |

**note:** generation 1 has better individuals than generation 0 – the population evolves

# another example:

- **task:** find the weights (real numbers) of the polynomial

$\boldsymbol{a}x^3 + \boldsymbol{b}x^2 + \boldsymbol{c}x + \boldsymbol{d}$

- how would you encode your individual?

# another example:

- each weight is represented by a gene (string of real-valued numbers)

| *a* | *b* | *c* | *d* |
|---|---|---|---|
| 0.5 | 0.7 | 0.2 | 0.3 |

- crossover works the same
- mutation generates a new real-valued number

# another example:

- fitness calculation will be based on **fitness cases**
    - values of **x** for which the value of the polynomial is known
    - fitness is a notion of how far from the desired value the individual is

| *a* | *b* | *c* | *d* |
|---|---|---|---|
| 0.5 | 0.7 | 0.2 | 0.3 |

$= 0.5 \cdot x^3 + 0.7 \cdot x^2 + 0.2 \cdot x + 0.3$

| x | individual | correct value | fitness (error) |
|---|---|---|---|
| 2 | 7.5 | 8.8 | 1.3 |
| 3 | 20.7 | 27.3 | 6.6 |
| | | **total:** | **7.9** |

the lower the error, the better the fitness

# advantages:

- perform a **global** search in the search space
  - work with a population of individuals, rather than a single individual (candidate solution)
  - broader exploration of the search space, less likely to get trapped in a local maxima
- candidate solution is represented in a declarative way, independent of the search method
- easy to implement

# considerations:

- do not offer any guarantee of finding the optimal solution, nor any lower bound on the quality
- generally computationally expensive
  - although can be easily parallelised
- several parameters need to be set
  - population size
  - number of generations
  - mutation / crossover probabilities
  - tournament size
  - … among others

# finishing off:

Practical class **this week:**

- more on **GA**
- finish the implementation of the GA before moving on to this week's exercise