Nature-Inspired Optimisation
Nature-Inspired Optimisation (Extended)

Lecture 5: Evolutionary Algorithms

Shan He

School for Computational Science
University of Birmingham

January 28, 2020

What we have learned and what we will learned

- What we have learned:
  - Randomised algorithms
  - Optimisation and local search algorithms
  - Stochastic Local Search algorithms – Simulated Annealing
- What will we learn this week:
  - Evolutionary Algorithms for optimisation
  - Binary and real-coded Genetic Algorithms

Assign worse solutions with a probability

Outline of Topics

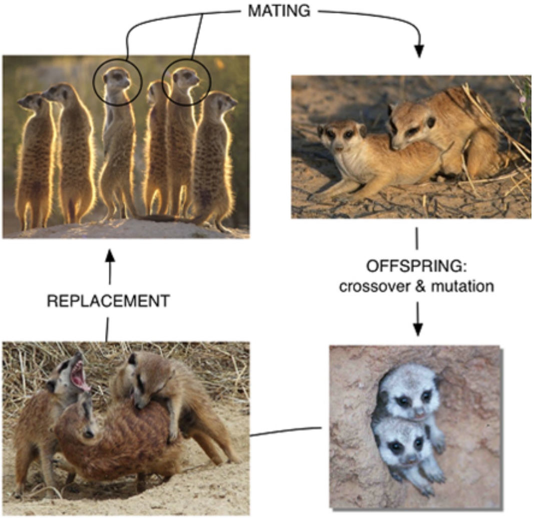
Evolution is amazing

http://www.environmentalgraffiti.com/featured/amazing-insect-camouflage/14128

How evolution works in nature?

- Evolution: **change** in the **inherited characteristics** of biological populations over successive generations.
    - **Heritable characteristics** or heritable traits
        - Example: the colour of your eyes are passed from one generation to the next via DNA
        - DNA: Deoxyribonucleic acid, a molecule that encodes genetic information
    - **Change** or genetic variation comes from:
        - Mutations: changes in the DNA sequence,
        - Crossover: reshuffling of genes through sexual reproduction and migration between populations
- **Driving force** of evolution: natural selection - survival of the fittest
    - Genetic variations that enhance survival and reproduction become and remain more common in successive generations of a population.

Evolution in Nature



What evolution taught us

- Living species are reproductions of earlier species with genetic variations
- Evolution is driven by natural selection - survival of the fittest
    - Living species are in some sense successful optimising individuals
- **Question**: How you map the relationship between evolution to optimisation?
    - Fitness $\rightarrow$ _ _ ? _ _ objective function value in optimisation
    - Individuals of a specie $\rightarrow$ _ _ ? _ _ solution in optimisation

What are Evolutionary Algorithms?

- Evolutionary Algorithms (EAs): a subset of metaheuristic algorithm inspired by biological evolution, which include Genetic Algorithm, Evolutionary Programming, Evolution Strategies, Differential Evolution.
- Essentially a kind of stochastic local search optimisation algorithm
- Evolutionary Algorithms $\in$ Metaheuristics $\in$ Heuristic algorithms $\in$ Stochastic Local Search algorithm $\in$ Search and enumeration algorithms
- Distinct characteristic of EAs:
    - **Population based**: generate, maintain and optimise a population of candidate solutions

Generic Evolutionary Algorithm

Generic Evolutionary Algorithm

$\mathbf{X}_0$:= generate initial population of solutions
terminationflag := false
t := 0
Evaluate the fitness of each individual in $\mathbf{X}_0$.
while (terminationflag != true)
1) **Selection**: Select parents from $\mathbf{X}_t$ based on their fitness.
2) **Variation**: Breed new individuals by applying variation operators to parents
3) **Fitness calculation**: Evaluate the fitness of new individuals.
4) **Reproduction**: Generate population $\mathbf{X}_{t+1}$ by replacing least-fit individuals
t := t + 1
If a termination criterion is met: terminationflag := true
Output x_{best}

Building blocks of Evolutionary Algorithms

- An Evolutionary Algorithms consists of:
  - **representation**: each solution is called an individual
  - **fitness** (objective) function: to evaluate solutions
  - **variation operators**: mutation and crossover
  - **selection and reproduction** : survival of the fittest
- Optimisation: finding global optimum is about the balance of exploration and exploitation
- **Question**: How EAs achieve the balance of exploration and exploitation?

  space ↑
  - Variation operators → exploration or exploitation?
  - Selection and reproduction → exploration or exploitation ?

  focus by selecting better solutions

Building blocks of Evolutionary Algorithms

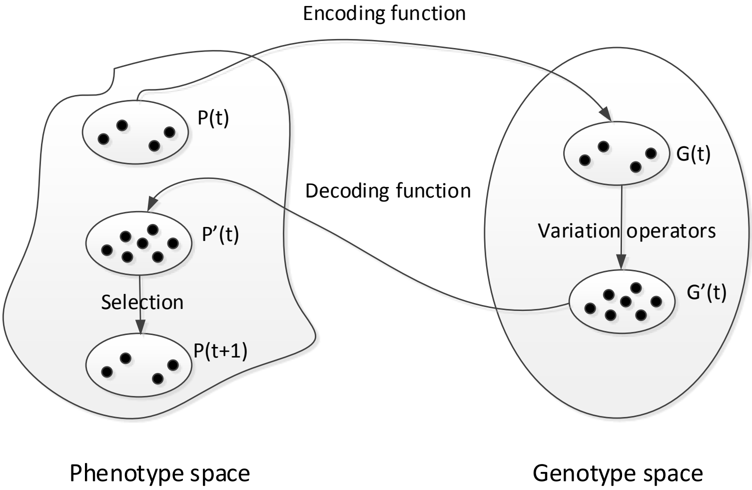
as a first try to solving problems

- In order to apply an EA to a problem, you need:
    - A suitable **representation** of solutions to the problem
    - A way to evaluate solutions: **fitness (objective) function**
    - A way to explore the space of solutions: **variation operators**
    - A way to guide the algorithm to find better solutions (exploitation): **selection and reproduction**

Representation

- Representation: a way to represent (encode) solutions
- Suppose we have a optimisation problem, of which the fitness (objective) function is $f(x)$, where x is the solutions. In EAs:
  - Solutions x is called **phenotypes**
  - We encode the solutions using some form of **representation**, e.g., binary strings (explain later)
  - The representation of solutions is called **genotypes**
  - Variation operators act on genotypes
  - Genotypes are decoded into phenotypes (solution) x
  - Phenotypes are evaluated using fitness function $f(x)$
  - Decoding and encoding functions map phenotypes and genotypes x2
  - Search space of solutions is the set of genotypes and phenotypes

Representation: process



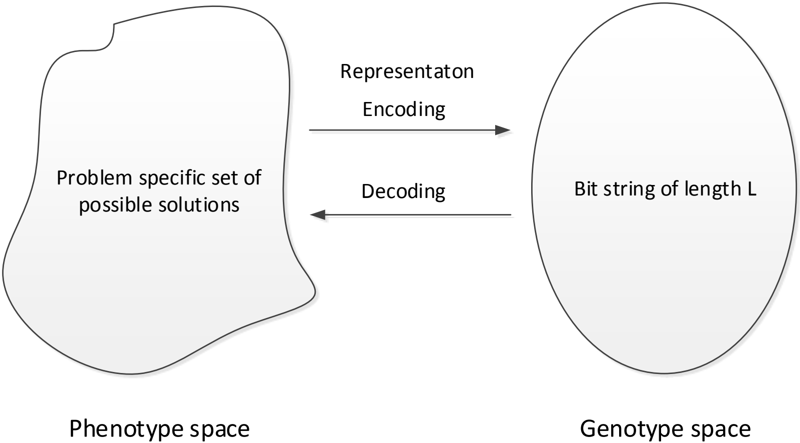
Representation

- The selection of representation depends on the problem
- We have the following choices:
  - **Binary representation**
  - **Real number representation**
  - **Random key representation**
  - Permutation representation: suitable for TSP
  - Other problem specific representations

Binary Representation

- Traditionally was the most popular representation used in Genetic Algorithms
- Represent an individual (solution) as a bit string of length L: $\vec{a} \in \{0,1\}^L$ – Genotypes
- Map phenotypes (solutions) to bit strings genotypes $\{0,1\}^L$ by encoding function
- Map genotypes (bit strings $\{0,1\}^L$) to phenotypes (solutions) by decoding function

Binary Representation



Decoding function

- Using a bit string to represent a binary or an integer solution is trivial
- **Question**: given an optimisation problem with n **continuous** variables, e.g., $\mathbf{x} \in \mathbb{R}^n$, how to represent them using a bit string of length L, e.g., $\vec{a} \in \{0, 1\}^L$
- Note: usually each continuous variable have a interval bound, e.g., $x_i \in [u_i, v_i]$

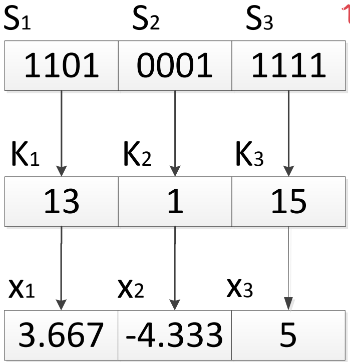
Decoding function

- Divide $\vec{a} \in \{0,1\}^L$ into n segments of equal length $\vec{s_i} \in \{0,1\}^{\frac{L}{n}}$, $i = 1, \cdots, n$
- Decode each segment into an integer K_i, $i = 1, \cdots, n$, and $K_i = \sum_{j=0}^{\frac{L}{n}} s_{i_j} \cdot 2^j$
- Apply decoding function $h(K_i)$, i.e., map the integer linearly into the interval bound $x_i \in [u_i, v_i]$:

$$h(K_i) = u_i + K_i \cdot \frac{v_i - u_i}{2^{\frac{L}{n}} - 1}$$

Decoding function: Example

- Assume $\mathbf{x} = \{x_1, x_2, x_3\}$ and $\mathbf{x} \in [-5, 5]$
- Use a bit string of $L = 12$, therefore $\frac{L}{3} = 4$ bits segment $\vec{s}$



$$u_i + K_i \cdot \frac{v_i - u_i}{2^{\frac{L}{n}} - 1}$$

$$= -5 + 15 \times \frac{5-(-5)}{15}$$

$$= -5 + 10$$

$$= 5$$

Building blocks of Evolutionary Algorithms

- In order to apply an EC to a problem, you need:
  - A suitable representation of solutions to the problem ✓
  - A way to evaluate solutions: fitness (objective) function ✓
  - A way to explore the space of solutions: **variation operators**
  - A way to guide the algorithm to find better solutions (exploitation): selection and reproduction

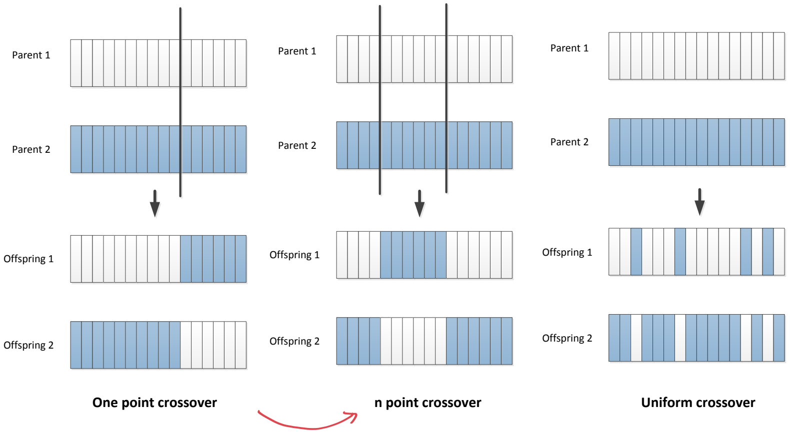
Mutation

- **Mutation**: flip each bit with a probability p_m, called mutation rate
- The standard mutation rate is $p_m = \frac{1}{L}$ but can be $p_m \in [\frac{1}{L}, \frac{1}{2}]$
- Example:
  - Parent: 00101011
  - After Mutation: 0$\underline{\mathbf{1}}$1010$\underline{\mathbf{0}}$1
- If mutation rate is small, mutation can be seen as creating a small random perturbation on the parent genotype
  - The mutated offspring is largely similar to its parent, so will stay near (in terms of Hamming Distance) the parent in the genotype search space
  - Together with selection what mutation actually does is stochastic local search: it **exploit** current good solutions by randomly **explore** the search space around them

Crossover

- Randomly select two parents with probability $p_c \in [0, 1]$ for crossover
- 1-point crossover: select a single crossover point on two strings, swap the data beyond that point in both strings.
- n-point crossover:
  - Select multiple crossover points on two strings,
  - Split strings into parts using those points
  - Alternating between the two parents and then glue parts
- Uniform crossover:
  - For each $i \in \{1, \cdots, L\}$: toss a coin
  - If 'head': copy bit i from parent 1 to offspring 1, parent 2 to offspring 2
  - If 'tail': copy bit i from parent 1 to offspring 2, parent 2 to offspring 1

Crossover illustration



Conclusion

- Evolutionary Algorithms (EAs): metaheuristics optimisation algorithm or stochastic local search algorithm
- One feature of EAs: population-based
- Genetic Algorithm traditionally used binary string as representation: might not be the best problem representation
- Two essential mechanisms:
  - Variation operators: introduce randomness to **explore** the space of solutions
  - Selection and reproduction: **exploit** current good solutions to find better solutions (Explain on Thursday)