Nature-inspired Algorithms

Lecture 6

Algorithm Engineering Group
Hasso Plattner Institute, University of Potsdam

19 June 2017

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

Evolutionary algorithms

Main idea: evolve a population of solutions

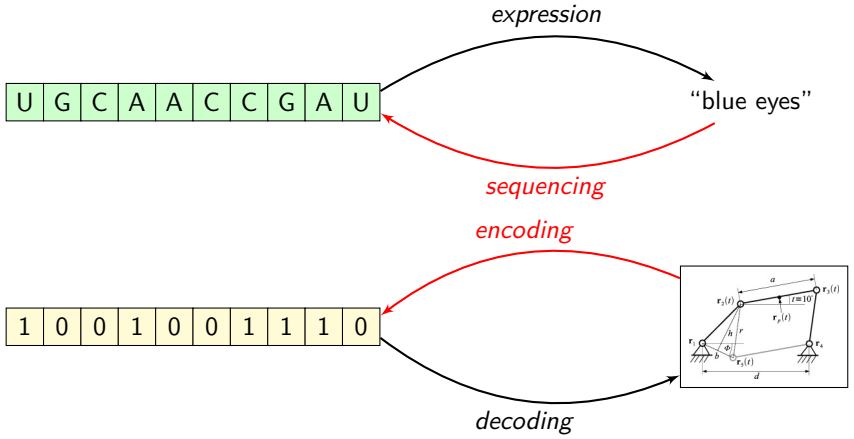
Design considerations

1. Solution *representation*
2. Operators
3. Fitness (selection, diversity, etc)

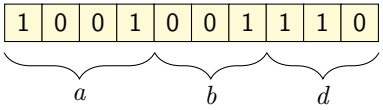
Implementing an EA on a computer

1. define a representation (encoding-decoding)
2. define a fitness function f
   - possibly incorporate constraints
3. define the genetic operators
   - initialization
   - selection
   - crossover
   - mutation
4. define survival and breeding mechanisms

Representation (encoding/decoding)



Encoding



$$1001 \Rightarrow 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 9$$

Representation

So far, we've only looked at bit strings

- easier to analyze from a theoretical perspective
- often the most efficient representation
- natural choice if optimization domain is (more or less) isomorphic to $\{0,1\}^n$
  - true/false assignments to n variables
  - bipartition of vertices
  - elements can be numbered from 1 to 2^n
  - subsets of a set of size n (e.g., sets of vertices/edges on a graph)

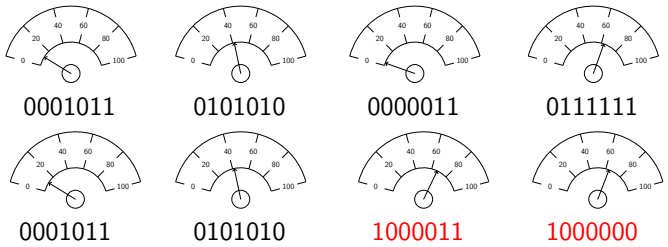
But doesn't a computer store everything as a bitstring anyway?

Problems with Bitstring Representations

Real-coding: code values with bitstring representation.

Consider a configuration $(11, 42, 3, 63)$

Encode as: 00010110101010000000110111111

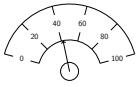


Combinatorial solution: Gray codes

- Idea from physical mechanical switches
- (off,on,on) → (off,off,on) → (on,off,on) → (on,off,off)

Decimal	Binary	Gray
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

0	0	00	00	000
1	1	01	01	001
	1	11	11	011
	0	10	10	010
			10	110
			11	111
			01	101
			00	100



Representation

Representation should be *efficient*

We also need to consider the design of *search operators*

Other representations

How to represent TSP with a bitstring?

How would search operators look?

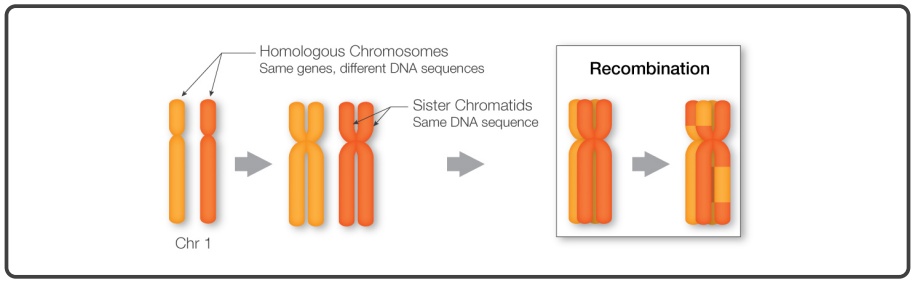
A better representation

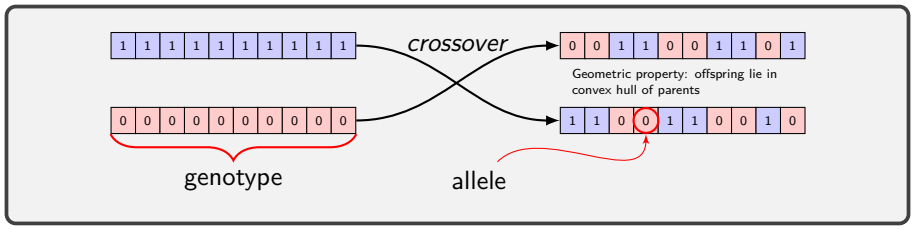
- Label the vertices from 1 to n
- A bijective mapping $\pi : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$. (a **permutation**)
- $\pi(i)$ is the label of the i-th vertex in a tour
- Let S_n be the set of all such permutations
- $f : S_n \to \mathbb{R}$ fitness function

What kinds of operators can you think of for S_n?

What other kinds of problems would work well with S_n representation?

Crossover

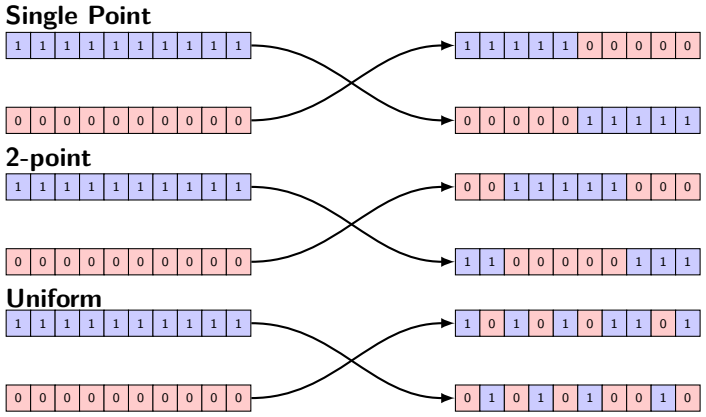




In both biology and EC: recombination introduces *variability* to the population

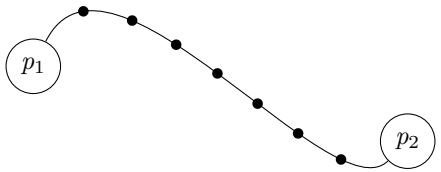
Crossover operators

Produce new offspring from two parent strings by copying selected bits from each parent



Path relinking

Given two parents $p_1, p_2 \in \{0,1\}^n$:



Create a *sequence* of children $(x_1, x_2, \ldots, x_m)$:

- $p_1 = x_1$
- $p_2 = x_m$
- Unit Hamming distance for every pair x_i, x_{i+1}

$$
\begin{array}{rccccccc}
p_1 = & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\
 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
p_2 = & 0 & 1 & 1 & 0 & 0 & 1 & 1
\end{array}
$$

Selection

Types of selection:

Parent (or sexual) selection

Fitter individuals in a population get a *better chance* to **reproduce**

Survival selection

Fitter individuals in a population get a *better chance* to **survive** to the next generation

Nature doesn't just kill off unfit genes – might become recessive for a long period of time and eventually mutate to something useful

Trade-off: better individuals and diversity

Selection operators

Truncation selection: keep the best μ

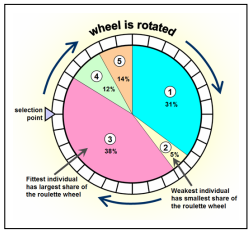
Fitness proportional: select $x \in P$ with probabilty

$$\frac{f(x)}{\sum_{y \in P} f(y)}$$

Rank-based: sort by fitness, select the k-th fittest individual with probability

$$\frac{1}{k}\left(\sum_{i=1}^{|P|}\frac{1}{i}\right)^{-1}$$

Roulette wheel:



Tournament selection: Choose k individuals uniformly at random Choose the i-th fittest individual with probability $p(1-p)^{i-1}$

Crowding

Crowding occurs when the individuals that are most fit quickly reproduce so that a large percentage of the entire population looks very similar.

Diversity

Genotypic diversity

A *diverse* set of genotypes

Phenotypic diversity

A *diverse* set of phenotypes (image under f)

Monomorphic	Low genotypic diversity	High genotypic diversity
1110010	1110010	1110010
1110010	1110011	0001101
1110010	1110010	1011010
1110010	1110110	0100101
1110010	1110110	1010110

Diversity

Why is diversity important?

- exploration
- crossover
- human-in-the-loop
- secondary objective

How do we measure diversity?

Genotypic diversity

- Count of duplicates/unique individuals
- Average genotypic distance

Phenotypic diversity

- Fitness variance, measures of spread

How does an algorithm maintain/promote diversity?

Implicitly: selection mechanism, mutation mechanism

Explicitly: diversity preserving mechanisms

Diversity-preserving mechanisms

Duplicate avoidance: never accept duplicates

Duplicate elimination: When breaking ties for removing individuals, kick out duplicates first (e.g., (μ+1) EA)

Duplicate minimization: As above, but always remove individual with the highest number of duplicates

Fitness sharing: penalize the fitness of similar individuals

$$f(x, P) = \frac{f(x)}{\sum_{y \in P} \mathrm{sh}(x, y)}, \qquad \mathrm{sh}(x, y) = \max\{0, 1 - (d(x, y)/\sigma)^{\alpha}$$

- σ : sharing distance
- $\alpha > 0$: shape