Evolutionary Algorithms

Lecture 2 - 28.04.2024

Lecturer and examinar: Baran Çürüklü
Robotics Group
Mälardalen University
721 23 Västerås, Sweden
Homepage: http://www.es.mdh.se/staff/15-Baran r kl
Mob: +46 (0)73-9607453
Email: baran.curuklu@mdu.se

2024-03-23 1

1

Evolutionary Algorithms

- What are Evolutionary algorithms (EA)
- Related to other techniques
  - Single-state versus population-based techniques
- Why EA?
- General structure
- Four different types

2024-03-23 2

2

What is EA?

- These algorithm are used in search and optimization problems.
  - Probabilistic (random) behaviour
  - Population based

Types of Evolutionary Algorithms

Evolutionary Strategies

Genetic Algorithms

Evolutionary Programming

Genetic Programming

Related to other techniques

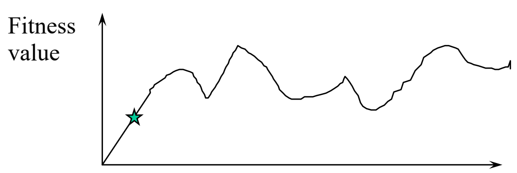
- **Single State** search or optimization techniques
  - Hill Climbing (Search)
  - Simulated Annealing (Search)
  - Artificial neural networks (ANN)
  - STRIPS (Planning)
- **Population Based** search or optimization techniques
  - Breadth and Depth first (Search)
  - A* (Search)
  - ***Evolutionary Algorithms***

2024-03-23 5

5

Single-state search

- Only **one** state in the search space
  - Faster to operate one solution than a population
  - Less memory needed
  - Cannot cover the search space as good as a population



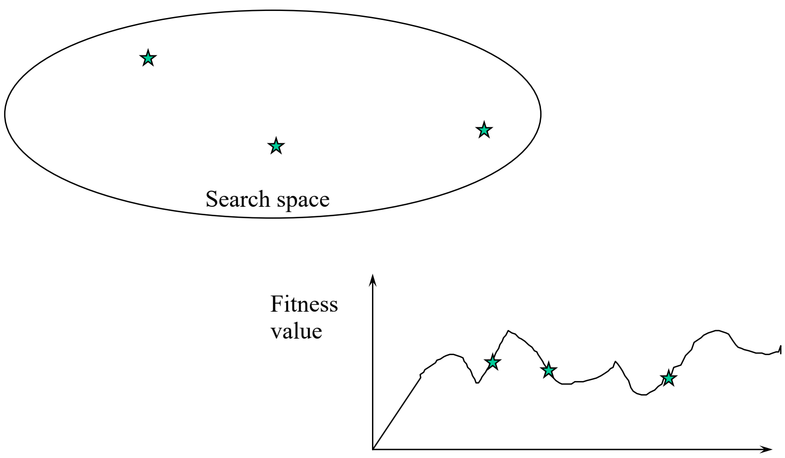
2024-03-23 6

6

Population based search

- Several states in the search space
  - No formal definition of the problem needed
  - Efficient
  - Solves in theory all types of problems
    - Not specific
    - Often high memory requirements
      - Not fast, if you are unlucky

Population based search



Why EA?

- Many problems have huge search space
- Analytical (mathematically precise) solutions don't exist
  - Sometimes no alternatives
- There is a reasonable evaluation measure that tells that a good solutions is
  - Translated into the fitness function
- **Relatively easy to use**
  - **and without domain knowledge**

2024-03-23 9

9

General structure of EA (1/4)

- Every individual in the population has
  - its own DNA (represents the solutions)
  - and its fitness value (tells how good the solutions is)
- DNA consists of genes for representing the solution

2024-03-23 10

10

General structure of EA (2/4)

function evalution-program
 $t \leftarrow 0$;
 init *Pop(t)*; /*init the first generation of individuals (solutions).*/
 eval *Pop(t)*; /* compute fitness-values of them. */
 while (**not** termination-condition) **do**
  $t \leftarrow t+1$; /* generation counter. */
  select *Pop(t)* from *Pop(t-1)*; /* parents to next generation */
  alter *Pop(t)*; /* generate new individuals. */
  eval *Pop(t)*;
end

2024-03-23 11

11

General structure of EA (3/4)

- $t \leftarrow 0$
  - The **generation** clock.
- init *Pop(t)*
  - *Pop(t)* is a set of individuals (**potential solutions**).
  - Initiate the population **randomly**.
- eval *Pop(t)*
  - Evaluate the **fitness-values** of the individuals in generation *t*.

2024-03-23 12

12

General structure of EA (4/4)

- select *Pop*(t) from *Pop*(t-1)
  - **Selection** of the parents to the next generation.
- alter *Pop(t)*: Randomly modification of ind.
  - **crossover:** <par1, par2> --> <kid1, kid2>
    - <01100111, 11010101> --> <01000101, 11110111>
  - **mutation:** Randomly modification of an individual (after crossover)
    - < 01000101> --> <01001100>

2024-03-23 13

13

Selection

- The goal is to find individuals that will be first combined through crossover, and later mutated.
  - Only mutation in Evolutionary strategies, thus no crossover. It still works!

2024-03-23 14

14

Selection

- The problem: One super individual that dominates the populations.
  - Solution: Decrease its probability to mate.
    - Rank-based selection is a good solution (see next slides)
    - Steady-state in combination with placing individuals in a 2D-space and letting them to mate with others in close surroundings
      - instead of replacing the whole population

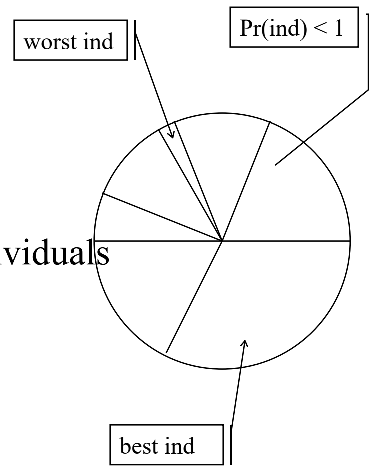
2024-03-23 15

15

Selection - Roulette-wheel

- $\Pr(ind_j) = \dfrac{fitness(ind_j)}{\sum_{i=1}^{|Pop|} fitness(ind_i)}$

- Select $|Pop|$ number of individuals
  - randomly



2024-03-23 16

16

Crossover - Arithmetic

- 60% probability that a gene pair will X-over.
- (Par1; Par2) => (Offspring1; Offspring2)
  - $[x_{11}, y_{11}, x_{12}, y_{12}, x_{13}, y_{13}]$
  - $[x_{21}, y_{21}, x_{22}, y_{22}, x_{23}, y_{23}]$

$$x_{11}^{'} = r \cdot x_{11} + (1-r) \cdot x_{21}$$
$$x_{21}^{'} = (1-r) \cdot x_{11} + r \cdot x_{21}$$

$$0 \le r \le 1$$

2024-03-23 17

17

Mutation - Uniform

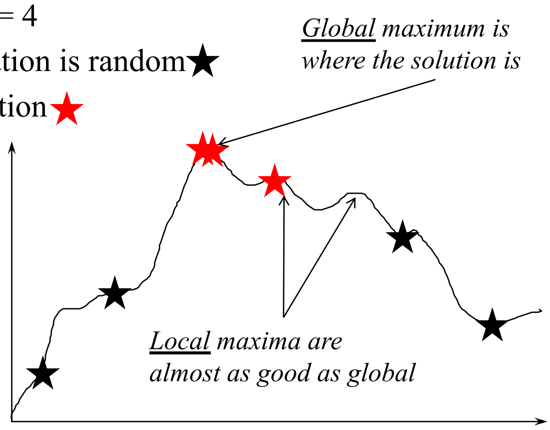
- 3% probability that a gene will be mutated.
- Offspring1 = $[x_{11}^{'}, y_{11}, x_{12}^{'}, y_{12}^{'}, x_{13}, y_{13}]$
- $y_{12}^{'} \xrightarrow{mutate} y_{12}^{''},\ \min(y_{i2}) \le y_{12}^{''} \le \max(y_{i2})$
- Mutated Offspring1 = $[x_{11}^{'}, y_{11}, x_{12}^{'}, y_{12}^{''}, x_{13}, y_{13}]$

2024-03-23 18

18

About convergence

- Population = 4
- First generation is random★
- Last generation★



2024-03-23 19

19

Genetic Algorithms (GA)

- Most well-known of all 4 techniques
- First version used binary coding
  - You can use integer and floating point as well
- Mutation and crossover

2024-03-23 20

20

Genetic Programming (GP)

- Similar to GA
- Solutions are computer programs or fractions of computer programs
- Mutation and crossover
- Make sure that the syntax is correct after mutation and crossover
  - This can be time consuming

2024-03-23 21

21

Genetic Programming (GP)

- Individuals and crossover may look like:

Parent 1: sin(x + cos(1.354 * x)) + 2.23 * y

Parent 2: 1.06 + x*y

Offspring 1: sin(x + cos(1.354 * x)) + x*y

Offspring 2: 1.06 + 2.23 *y

2024-03-23 22

22

Genetic Programming (GP)

- **Main problem:** Make sure that the offspring follow the grammar
  - Time consuming to check and correct syntax/semantic errors
    - Sometimes very hard, thus most crossover operations make no sense

2024-03-23 23

23

Evolutionary Programming (EP)

- Similar to Genetic programming but the syntax, or the computer program, is static
  - No need to check the grammar as in GP

2024-03-23 24

24

Evolutionary Strategies (ES)

- No crossover
  - Example: Select 20% of the population and mutate each 5 times to generate new individuals
  - For some problems crossover is hard to define