Evolutionary Computing

The Inspiration from Biology

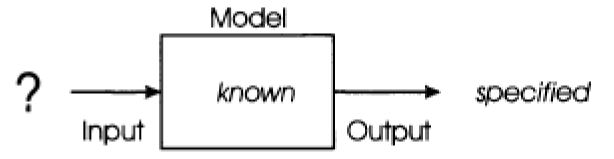
- **Darwinian Evolution**
- Given an environment that can host only a **limited number of individuals**, and the basic instinct of individuals to **reproduce, selection** becomes inevitable if the population size is not to grow exponentially.
- **Natural selection** favors those individuals that compete for the given resources most effectively, in other words, those that are adapted or fit to the environmental conditions best.
- This phenomenon is also known as **survival of the fittest**.

Evolutionary Computing: Why?

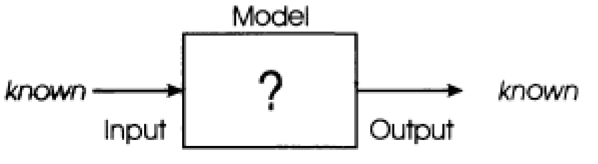
- Developing **automated problem solvers** (that is, algorithms) is one of the central themes of mathematics and computer science.
- Nature's solutions has always been a source of inspiration, copying "**natural problem solvers**"
- The most powerful natural problem solver, there are two rather straight forward candidates:
  - **The human brain**-neurocomputing
  - **The evolutionary process**-evolutionary computing

Evolutionary Computing: Why?

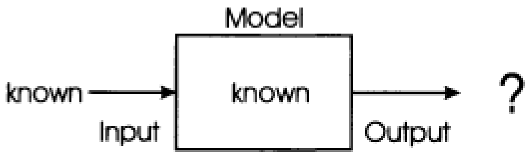
1. Optimization problems



2. Modeling or system identification problem



3. Simulation problem

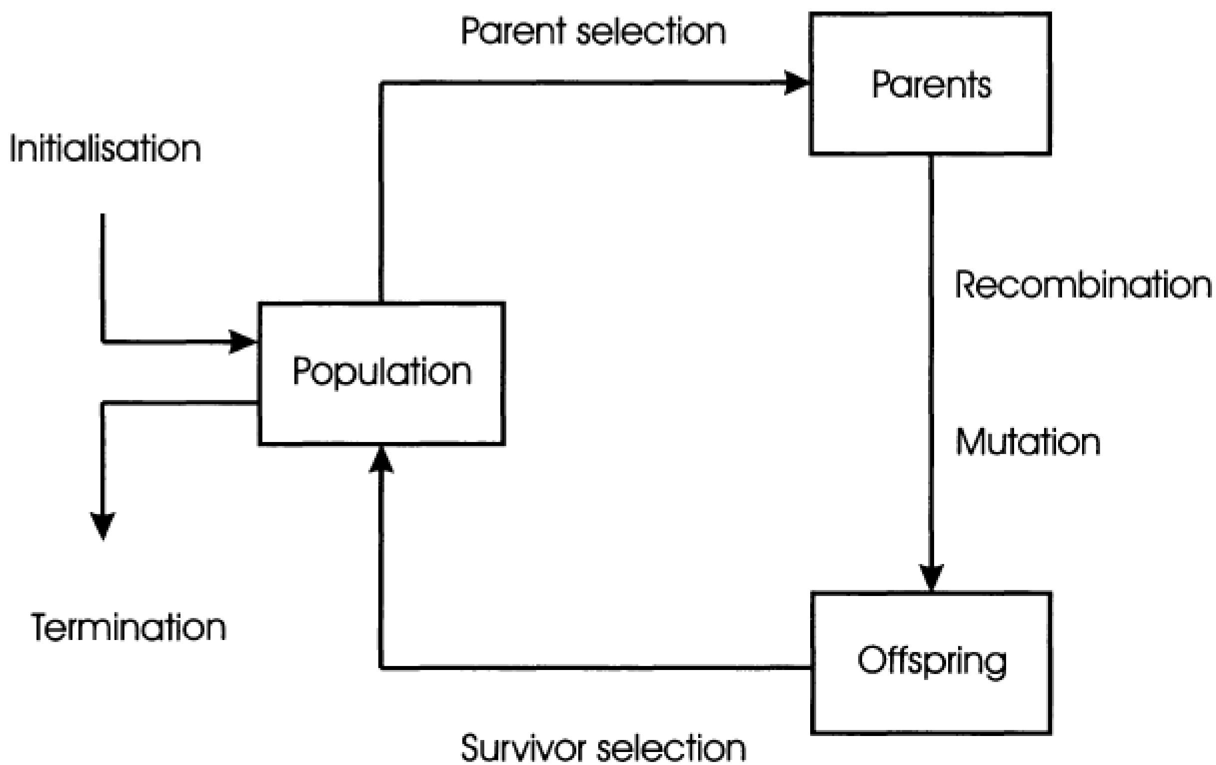


What is an Evolutionary Algorithm?

- There are many different variants of evolutionary algorithms. The common underlying idea behind all these techniques is the same:

1. Given a **population of individuals**
2. The environmental pressure causes **natural selection (survival of the fittest)**, which causes a rise in the fitness of the population.

What is an Evolutionary Algorithm?



What is an Evolutionary Algorithm?

```
BEGIN
  INITIALISE population with random candidate solutions;
  EVALUATE each candidate;
  REPEAT UNTIL ( TERMINATION CONDITION is satisfied ) DO
    1 SELECT parents;
    2 RECOMBINE pairs of parents;
    3 MUTATE the resulting offspring;
    4 EVALUATE new candidates;
    5 SELECT individuals for the next generation;
  OD
END
```

Properties of Evolutionary Algorithm

- EAs are **population based**, i.e., they process a whole collection of candidate solutions simultaneously.

- EAs mostly use **recombination** to mix information of more candidate solutions into a new one.

- EAs are **stochastic**.
  - Having a random probability distribution or pattern that may be analyzed statistically but **may not be predicted precisely**.

Components of Evolutionary Algorithms

1. **Representation** (definition of individuals)
2. **Evaluation function** (or fitness function)
3. **Population**
4. **Parent selection mechanism**
5. **Variation operators, recombination and mutation**
6. **Survivor selection mechanism** (replacement)

Representation (Definition of Individuals)

- The first step in defining an EA is to link the "real world" to the "EA world“.
  - **Phenotypes** - Objects forming possible solutions within the original problem context.
  - **Genotypes** - Objects encoding, that is, the individuals within the EA.
- **Representation**
  - Specifying a **mapping** from the phenotypes onto a set of genotypes that are said to represent these phenotypes.
  - In case of set of integers, **18** would be seen as a phenotype, and **10010** as a genotype.

Evaluation Function (Fitness Function)

- It is a **function or procedure** that assigns a **quality measure** to genotypes.

- To **maximize square(*x*)** fitness of the genotype **10010** could be defined as the square of its corresponding phenotype: **Square(18)=324**.

- Also called **objective function.**

Population

- The role of the population is to hold (the representation of) **possible solutions**.
- A population is a **multiset** of genotypes.
- Defining a population can be as simple as specifying how many individuals are in it, that is, setting the **population size**.
- **Best individual** of the given population is chosen to seed the **next generation**, or the **worst individual** of the given population is chosen to be **replaced by a new one**.
- The **diversity** of a population is a **measure of the number of different solutions** present.

Parent Selection Mechanism

- The role of **parent selection or mating selection is to distinguish among** individuals based on their quality, in particular, to allow the better individuals to become parents of the next generation.
- An individual is a **parent if it has been selected** to undergo variation in order to create offspring.
- **High-quality individuals** get a **higher chance** to become parents than those with low quality.
- Nevertheless, low-quality individuals are often given a small, but positive chance; otherwise the whole search could become too greedy and **get stuck in a local optimum**.

Variation Operators

- The role of **variation operators is to create new individuals from old ones**.
- **Mutation**
- **Recombination**

Mutation

- A **unary variation** operator is commonly called mutation.
- It is applied to **one genotype and delivers a (slightly) modified mutant**, the child or offspring of it.

Recombination

- A **binary variation** operator is called recombination or crossover.
- As the names indicate, such an **operator merges information from two parent genotypes into one** or two offspring genotypes.

Survivor Selection Mechanism (Replacement)

- The role of survivor selection or environmental selection is to **distinguish among individuals based on their quality**.
- **Survivor selection** is also often called **replacement or replacement strategy**.

Initialization

- Initialization is kept simple in most EA applications: The first population is seeded by **randomly generated** individuals.
- In principle, problem specific heuristics can be used in this step aiming at an **initial population with higher fitness**.

Termination Condition

- If the problem has a **known optimal fitness level**, probably coming from a known optimum of the given objective function, then reaching this level (perhaps only with a given precision **E > 0**) should be used as stopping condition.

- The **maximally allowed CPU time elapses**.
- The **total number of fitness evaluations reaches a given limit**.
- For a given period of time (i.e, **for a number of generations or fitness evaluations**), the fitness improvement remains under a threshold value.
- The population **diversity drops under a given threshold**.

The Eight-Queens Problem

- Our candidate solutions are complete, rather than partial, board configurations where **all eight queens are placed**.
- The quality q(p) of any phenotype can be simply quantified by the **number of checking queen pairs**.
- **q(p) = 0**, indicates a good solution.
- As for mutation we can use an operator that **selects two positions in a given chromosome randomly** and **swaps the values** standing on those positions.

The Eight-Queens Problem

- we select two parents delivering two children and the new population of size n will contain the best n of the resulting **n + 2** individuals.
- **Parent selection** will be done by choosing five individuals randomly from the population and taking the best two as parents that undergo crossover.

1. Select a random position, the crossover point, $i \in \{1, \ldots, 7\}$
2. Cut both parents in two segments after this position
3. Copy the first segment of parent 1 into child 1 and the first segment of parent 2 into child 2
4. Scan parent 2 from left to right and fill the second segment of child 1 with values from parent 2, skipping those that are already contained in it
5. Do the same for parent 1 and child 2

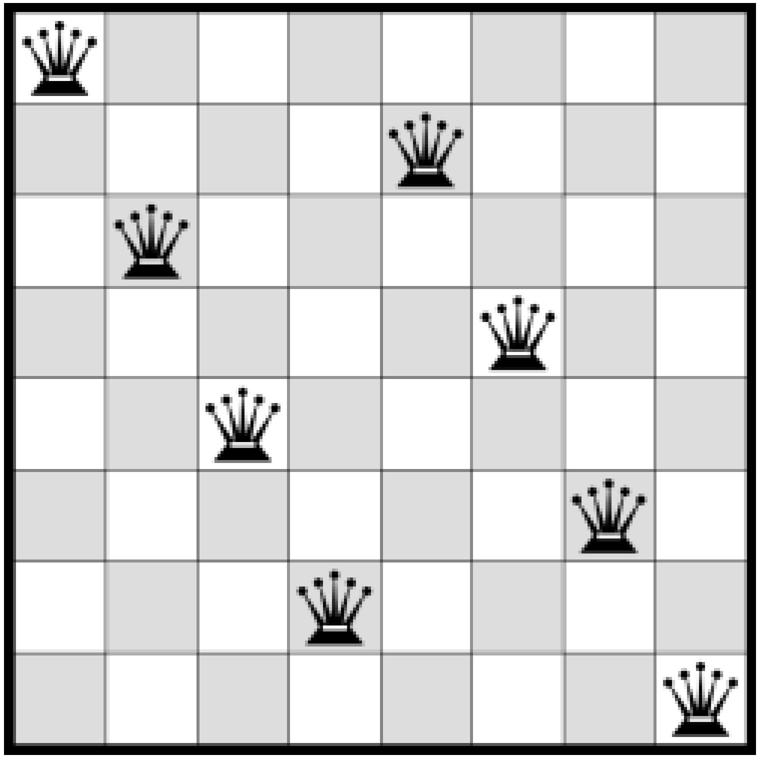
Fig. 2.3. "Cut-and-crossfill" crossover

The Eight-Queens Problem

- The strategy we will use merges the population and offspring, then ranks them according to fitness, and **deletes the worst two**.
- Terminate the search if we find a **solution** or **10,000 fitness** evaluations have elapsed.

Representation	Permutations
Recombination	"Cut-and-crossfill" crossover
Recombination probability	100%
Mutation	Swap
Mutation probability	80%
Parent selection	Best 2 out of random 5
Survival selection	Replace worst
Population size	100
Number of Offspring	2
Initialisation	Random
Termination condition	Solution or 10,000 fitness evaluation

The Eight-Queens Problem



0-1 Knapsack

Let us consider that the capacity of the knapsack **W = 60** and the list of provided items are shown in the following table.

Item	A	B	C	D
Profit (pi)	280	100	120	50
Weight(wi)	40	10	20	10
Ratio (pi/wi)	7	10	6	5

0-1 Knapsack

After sorting, the items are as shown in the following table.

Item	B	A	C	D
Profit (pi)	100	280	120	50
Weight(wi)	10	40	20	10
Ratio (pi/wi)	10	7	6	5

0-1 Knapsack

After sorting, the items are as shown in the following table.

Item	B	A	C	D
Profit (pi)	**100**	**280**	**120**	**50**
Weight(wi)	**10**	**40**	**20**	**10**
Ratio (pi/wi)	**10**	**7**	**6**	**5**

The total weight of the selected items is **10 + 40 + 10= 60**
And the total profit is **100 + 280 + 50= 380 + 50 = 430**

The Knapsack Problem

Representation	Binary strings of length n
Recombination	One point crossover
Recombination probability	70%
Mutation	Each value inverted with independent probability p_m
Mutation probability p_m	$1/n$
Parent selection	Best out of random 2
Survival selection	Generational
Population size	500
Number of offspring	500
Initialisation	Random
Termination condition	No improvement in last 25 generations

Thank you