성신여자대학교

Algorithm

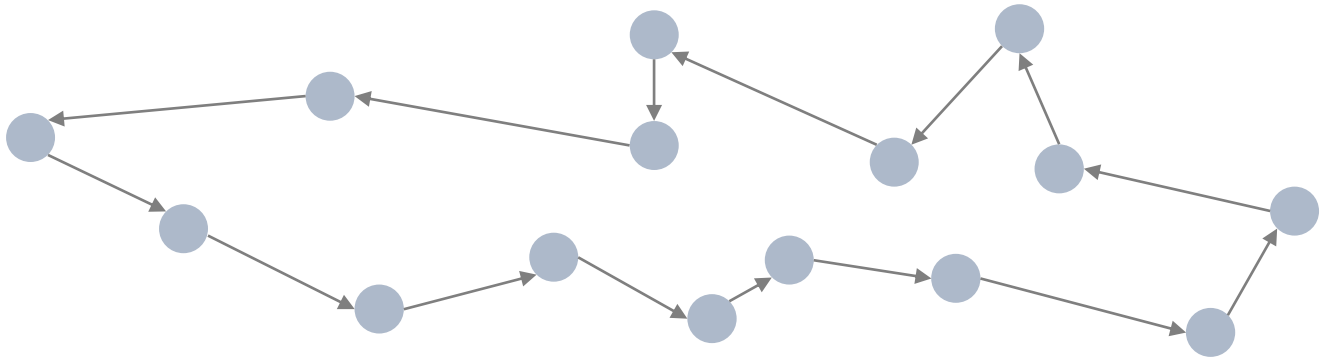
Yangjun Ahn
yangjunahn@sungshin.ac.kr
https://sites.google.com/sungshin.ac.kr/mhail

Textbook

- **Chapter 9**
- **Chapter 10**

Travelling Salesperson Problem (TSP)

- **Find the shortest possible path that visits a set of cities and returns to the starting city, visiting each city once.**



- **Q. If a computer can evaluate 1.5 million cases per second, how much time does it take to see all cases of visiting and returning to all 27 locations?**

Local Optimum

Local optimum = attractor

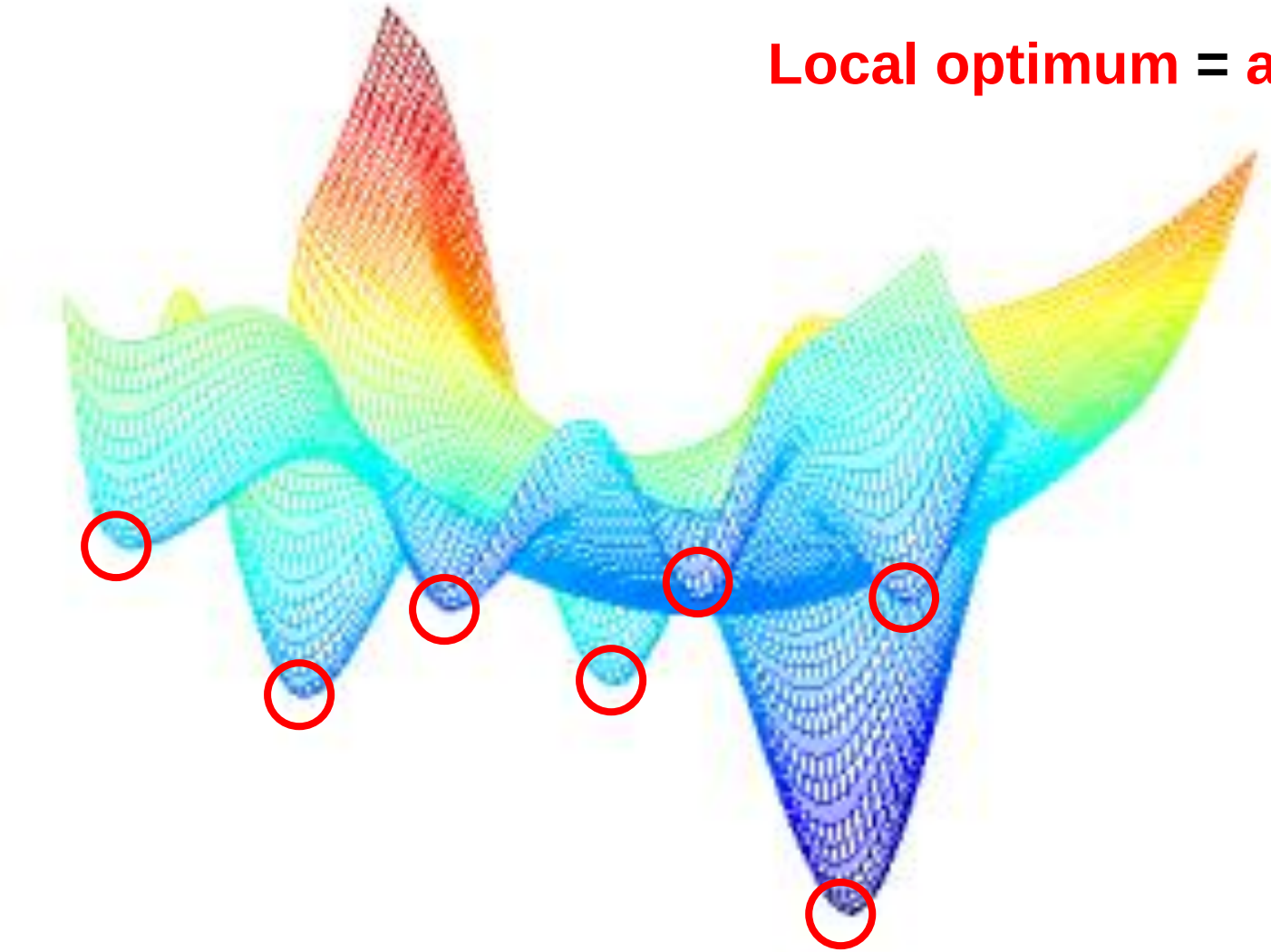
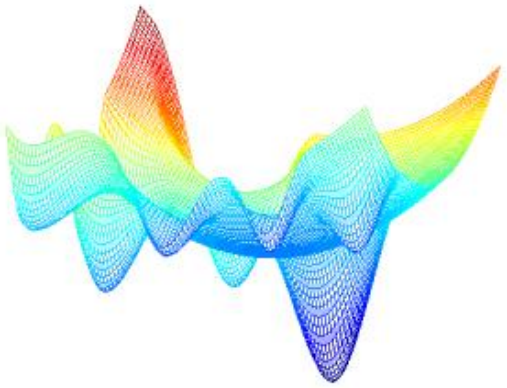


Fig: http://www.globaloptimization.org/

Number of TSP Attractors

Number of Nodes	Number of Attractors (Avg.)
10	4
20	170
100	3.4×10^{16}



- **Modern computer science deals with a problem with more than 8,000 nodes.**
- **Optimization algorithms are a way of transportation in such a vast space.**

Fig: http://www.globaloptimization.org/

Attractor

- **Getting out of an attractor takes a considerable amount of energy.**
  - Correcting a rigid swing form
  - Changing stereotypes
  - Reforming government systems
- **Revolution = strong perturbation + local optimization**
- **The attractor is both a goal and an obstacle for spatial exploration.**

Genetic Algorithm (GA) Template

```
Generate initial solutions (say, p solutions);
do {
        for (i = 0; i <k; ++i) {
                Select two parent solutions;
                Crossover;
                Mutation;
        }
        Replace;
} while (not stopping condition);
Report the best solution;
```

Vocabulary

Chromosome

Population

Gene

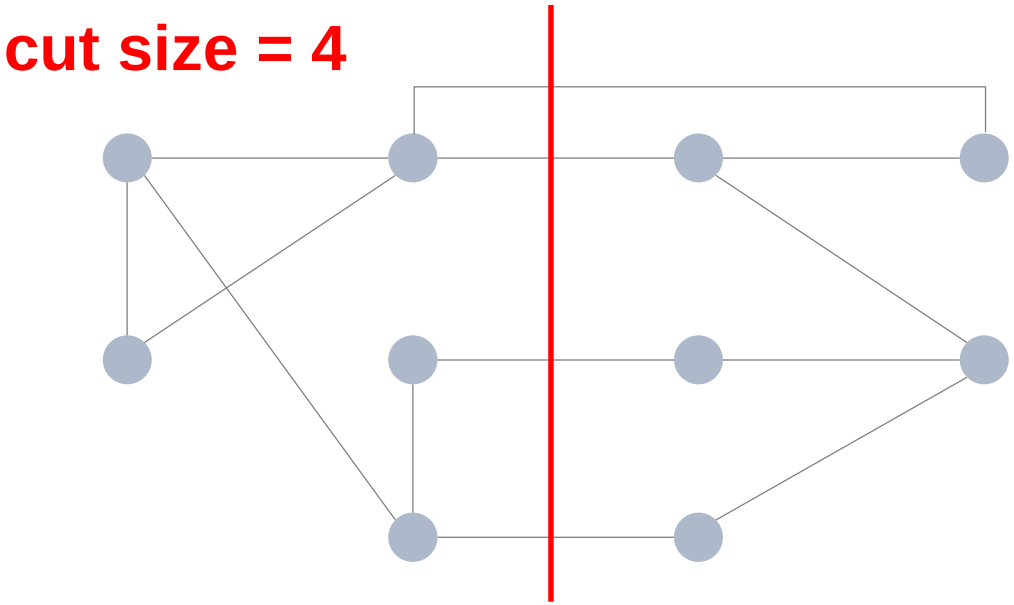
Genotype, Phenotype

Chromosome (Solution) Expression

```
Generate initial solutions (say, p solutions);
do {
        for (i = 0; i <k; ++i) {
                Select two parent solutions;
                Crossover;
                Mutation;
        }
        Replace;
} while (not stopping condition);
Report the best solution;
```

Chromosome (Solution) Expression

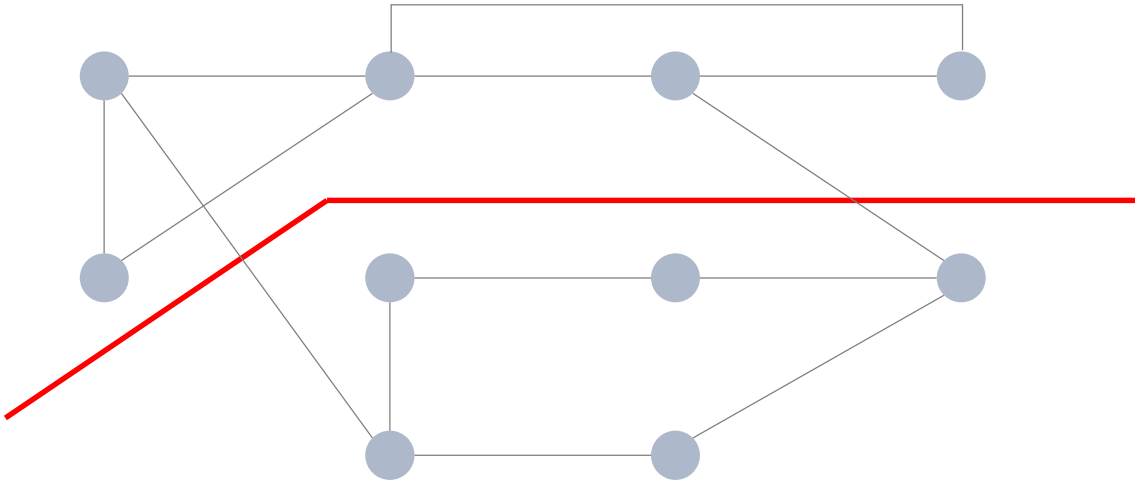
- Bisection : dividing a graph into two subgraphs of the same size
- Cut size : the number of cross edges of the bisection



Chromosome (Solution) Expression

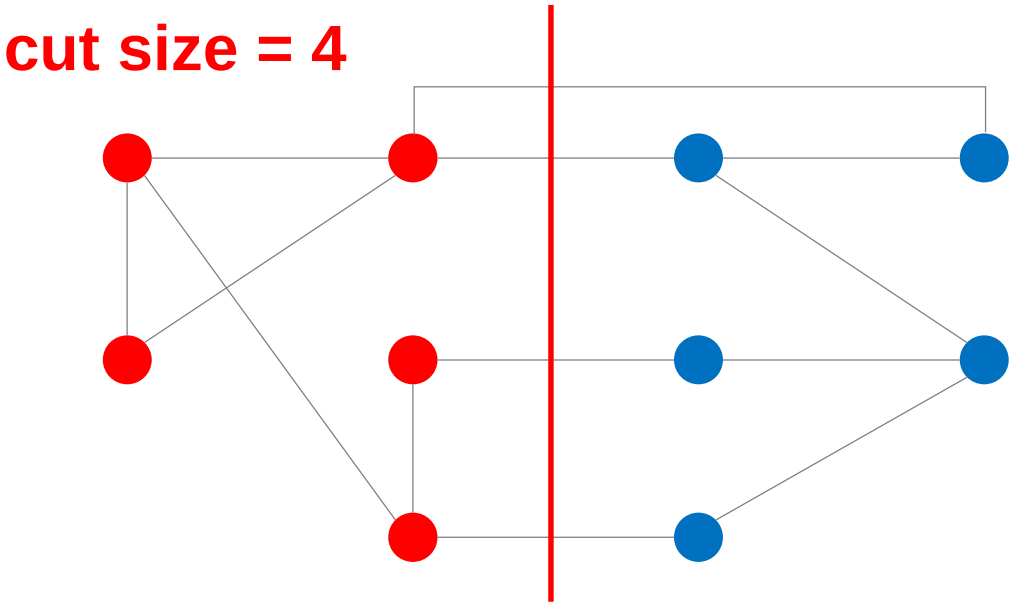
- Bisection : dividing a graph into two subgraphs of the same size
- Cut size : the number of cross edges of the bisection

cut size = 2



Binary Encoding for Graph Bisection

- Bisection : dividing a graph into two subgraphs of the same size
- Cut size : the number of cross edges of the bisection



0 0 1 1 0 0 1 1 0 1

Selection

```
Generate initial solutions (say, p solutions);
do {
        for (i = 0; i <k; ++i) {
                Select two parent solutions;
                Crossover;
                Mutation;
        }
        Replace;
} while (not stopping condition);
Report the best solution;
```

Selection

- **Each solution has a selection chance proportional to the fitness value f_i.**

$$f_i = (C_w - C_i) + (C_w - C_b) / (k - 1),\ k > 1$$

where
- C_w : cost of the worst chromosome
- C_b : cost of the best chromosome
- C_i : cost of the ith chromosome
- k : selection pressure (normally 3~4)

- **Even people who fail in society can give birth to good children. However, the probability is low.**

Roulette Wheel Selection

Roulette Wheel Selection:

```
point ←random(0, SumOfFitness);
sum ←0;
for i ← 0 to P-1{
        sum ← sum + f_i;
        if (point < sum) then return i;
}
Report error; // shouldn't be here
```

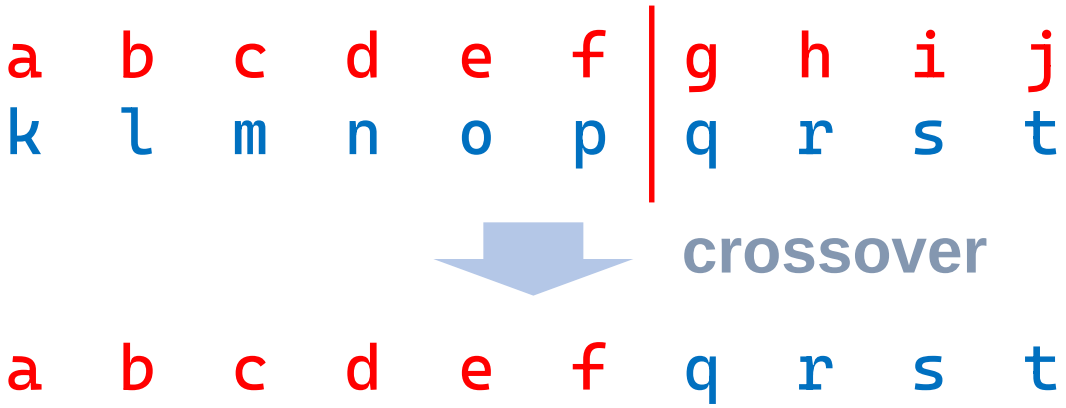
- random(0, SumOfFitness) : a random value in the [0, SumOfFitnesses) interval

Crossover

```
Generate initial solutions (say, p solutions);
do {
        for (i = 0; i <k; ++i) {
                Select two parent solutions;
                Crossover;
                Mutation;
        }
        Replace;
} while (not stopping condition);
Report the best solution;
```

Crossover

- **1-point crossover**



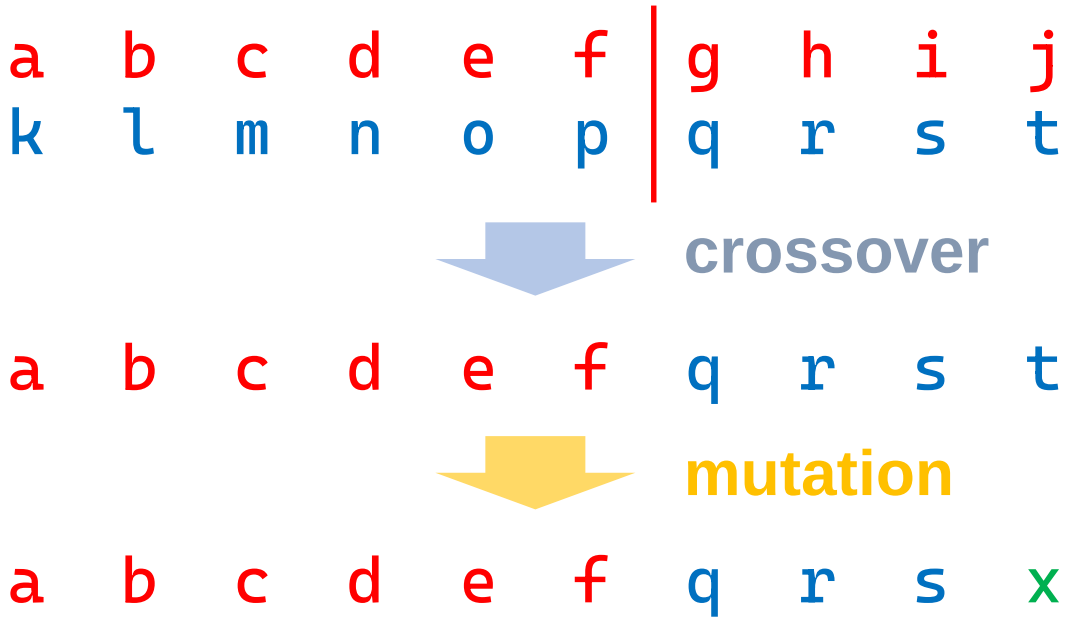
- **Many variations exist.**

Mutation

```
Generate initial solutions (say, p solutions);
do {
        for (i = 0; i <k; ++i) {
                Select two parent solutions;
                Crossover;
                Mutation;
        }
        Replace;
} while (not stopping condition);
Report the best solution;
```

Mutation

- μ : mutation strength (typically 0.015 ~ 0.01)



Replace

```
Generate initial solutions (say, p solutions);
do {
        for (i = 0; i <k; ++i) {
                Select two parent solutions;
                Crossover;
                Mutation;
        }
        Replace;
} while (not stopping condition);
Report the best solution;
```

k/p: generation gap

- if k/p →1, generational GA
- if k/p → 1/p, steady-state GA
- * k is not the selection pressure!

Replace

- ✓ **Easy in generational GAs (usually)**
- ✓ **There are some choices in steady-state GA**
- **Genitor-style**
  - Replace the worst solution in the population
- **Preselection**
  - Replace one of the parents
- **Crowding**
  - Randomly choose *k* solutions at ransom in the population
  - Replace the most similar one to the offspring
- **Combinations**

Basic Vocabulary

- **chromosome = solution**

  1 0 1 1 0 0 0 1 1 1 0 1 0 1

- **schema : a pattern inside chromosome**

  *** * * * 0 0 * 1 1 1 * * * ***

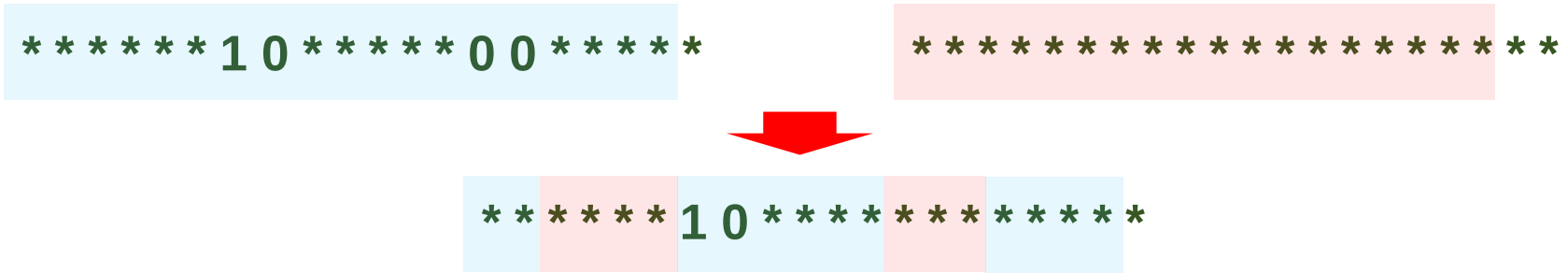
  A chromosome of length n has 2^n schemas.

  defining length = 5
  order = 5

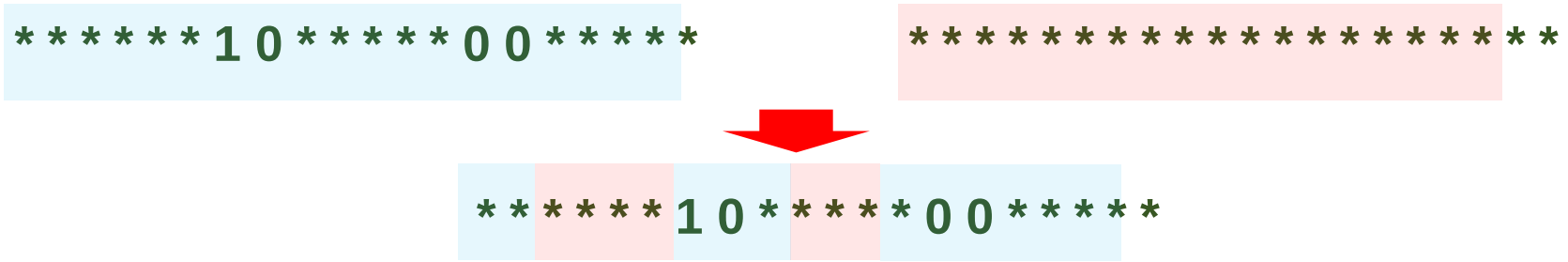
  - *** : don't care symbol**
  - **0, 1 : specific symbols**
  - **order : # of specific symbols**

Schema Destruction & Preservation

- **Destruction**



- **Preservation**



- **It will be better if a schema with good quality survives well with high probability.**

질문

- 상위 구조에서 하위 구조에서 갖지 못한 특성이 나타날 수 있나요?

Building-Block Hypothesis

- GA seeks near optimal performance through the juxtaposition of short, low-order, high-quality schemata

Four Classes of Systems*

- Class I Converges to a single state
- Class II Periodic
- Class III Chaotic
- Class IV Complex local patterns

*Chris Langton, Life at the edge of chaos

Emergence

- A higher structure is formed by combining lower structures.
- A concept that does not exist in a lower structure occurs in a higher structure.
- It is a typical characteristic of complex systems.
- The operating mechanism of GA itself is a complex system.

Typical Genetic Algorithm Summary

Create n initial chromosomes;
repeat {
    for $i = 1$ to k {
        Select two parents;
        offspring_i = crossover(p1, p2);
        offspring_i = mutation(offspring_i);
    }
    Replace k chromosomes with offspring_1, …, offspring_k;
} until (stopping condition);
return the best chromosome;

Homework

- 18th May

  Max-cut with pure GA

- 9th June

  Max-cut with heuristic GA and DP

  Eq. Optimization with GA and DP (optional)