بسم اللّه الرحمن الرحیم

☜ تذکر۱: لطفاً یادداشت‌برداری کنید. آنچه در کلاس مطرح می‌شود ممکن است فراتر از محتوای اسلایدها باشد.
☜ تذکر ۲: لطفاً در اسرع وقت ایمیل خود را در سامانهٔ درس (lms.iut.ac.ir) وارد کنید.
☜ تذکر ۳: اگر احیاناً در ترجمه یا فهم اسلایدها مشکل داشتید، لطفاً در ساعات رفع اشکال به دفتر بنده مراجعه کنید. (ساعات رفع اشکال به‌زودی تعیین خواهند شد انشاءالله.)
☜ تذکر ۴: اسلایدها و دیگر محتواهای مرتبط با درس روی سامانهٔ درس (lms.iut.ac.ir) آپلود می‌شوند.

آنچه گذشت:

Local search provides no guarantee that the solution is optimal or even lies within any given distance from the optimum.

The motivation for local search is that a neighborhood is more easily searched than the entire solution space. By moving from neighborhood to neighborhood, the search may happen upon a good solution. Well-designed local search methods can, in fact, deliver remarkably good solutions within a reasonable time, although tuning them to work efficiently is more an art than a science. Local search has become indispensable for attacking many practical problems that are too large to solve by exact methods.

آنچه گذشت:

A metaheuristic is a general kind of solution method that orchestrates the interaction between local improvement procedures and higher level strategies to create a process that is capable of escaping from local optima and performing a robust search of a feasible region.

قبل از اینکه به‌سراغ بحث الگوریتم‌های GA برویم، یک نکته را باید یادآوری کنم.

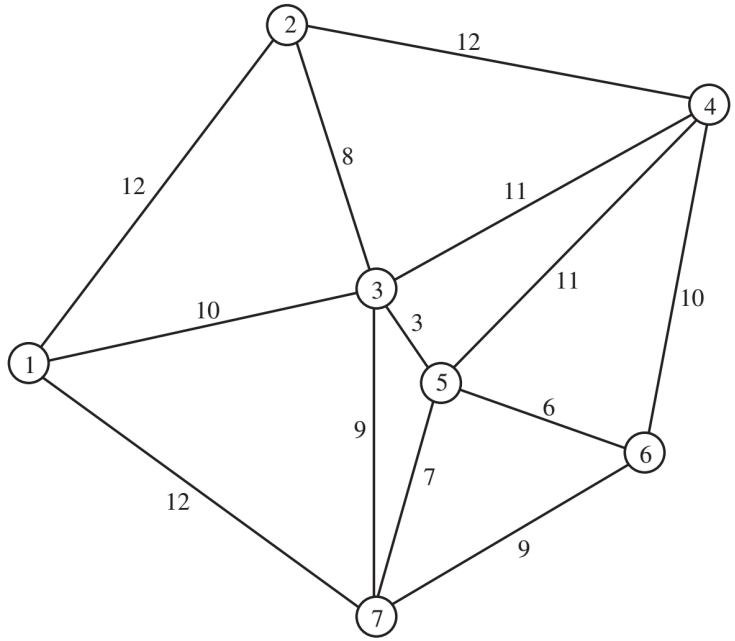
ساختار همسایگی را شما باید معین کنید:

مثلاً برای مسئلهٔ TSP، عرض کردیم که هر جواب شدنی را می‌توان توسط یک جایگشت معین کرد. حال ساختار همسایگی می‌تواند مثلاً بر اساس عملگر 2-swap باشد، یا عملگر sub-tour reversal که در زیر تعریف شده است:

A sub-tour reversal adjusts the sequence of cities visited in the current trial solution by selecting a subsequence of the cities and simply reversing the order in which that subsequence of cities is visited. (The subsequence being reversed can consist of as few as two cities, but also can have more.)

ساختار همسایگی (neighborhood structure) نقشی اساسی در نحوهٔ رفتار الگوریتم شما دارد.

Example (Sub-tour Reversal Operator)



$1, 2, 3, 4, 5, 6, 7$ ☞ **Tour Length** $= 69$

Reverse 2-3: $1, 3, 2, 4, 5, 6, 7$ ☞ **Tour Length** $= 68$

Reverse 3-4: $1, 2, 4, 3, 5, 6, 7$ ☞ **Tour Length** $= 65$

Reverse 4-5: $1, 2, 3, 5, 4, 6, 7$ ☞ **Tour Length** $= 65$

Reverse 5-6: $1, 2, 3, 4, 6, 5, 7$ ☞ **Tour Length** $= 66$

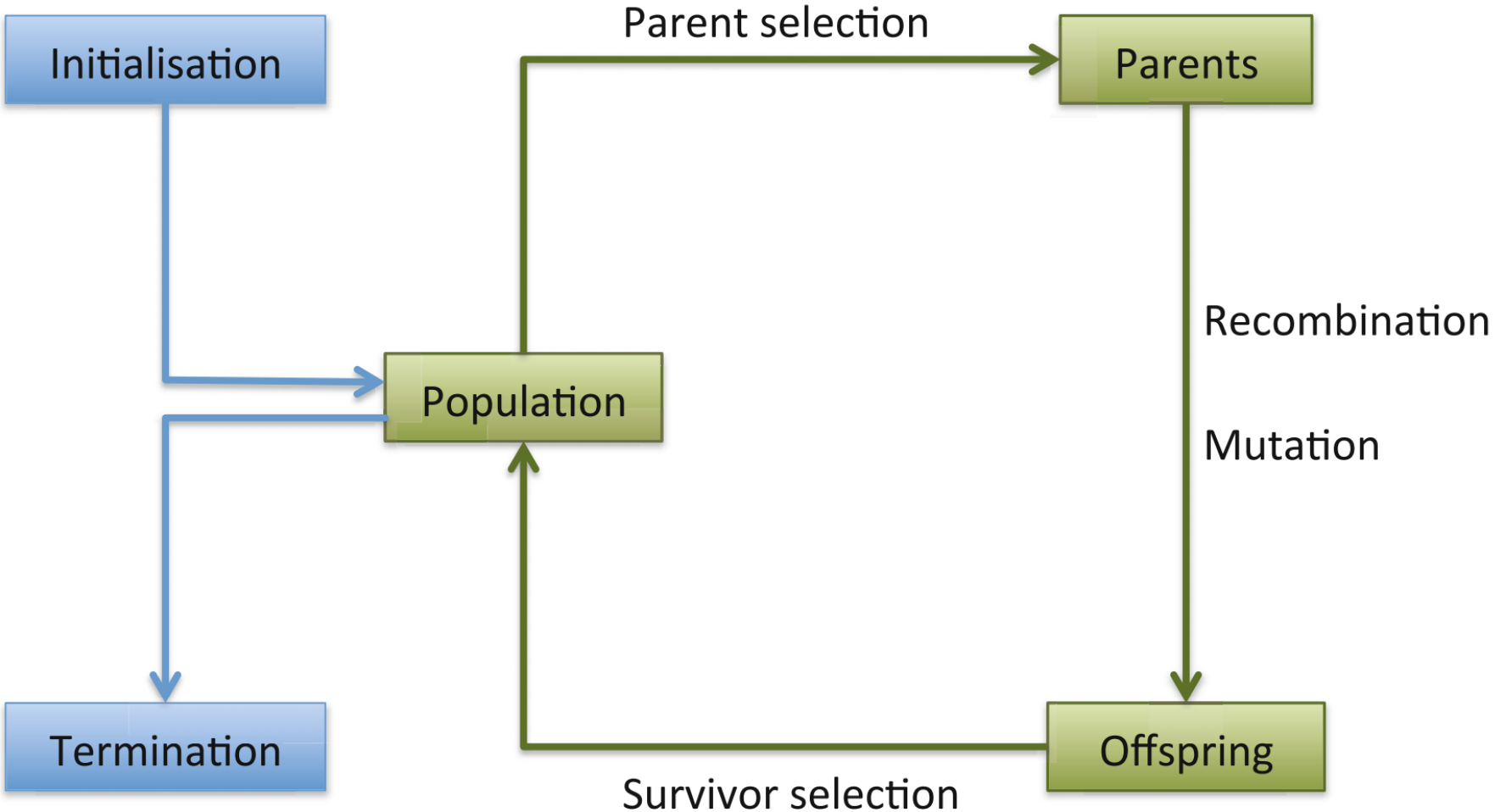
Reverse 3-4-5-6: $1, 2, 6, 5, 4, 3, 7$ ☞ **Tour Length** $= +\infty$

Genetic algorithms (GAs)

Components of Evolutionary Algorithms:

* representation (definition of individuals)
* evaluation function (or fitness function)
* population
* parent selection mechanism
* variation operators, recombination and mutation
* survivor selection mechanism (replacement)

The general scheme of an evolutionary algorithm as a flowchart



The general scheme of an evolutionary algorithm in pseudocode

```
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
```

GAs begin with a set of k randomly generated states, called the population. Each state is rated by the objective function, or (in GA terminology) the fitness function. A fitness function should return higher values for better states.

Evolution		Problem solving
Environment	$\longleftrightarrow$	Problem
Individual	$\longleftrightarrow$	Candidate solution
Fitness	$\longleftrightarrow$	Quality

Variation Operators (Mutation and Recombination)

The role of variation operators is to create new individuals from old ones.

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.

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. Like mutation, recombination is a stochastic operator: the choices of what parts of each parent are combined, and how this is done, depend on random drawings.

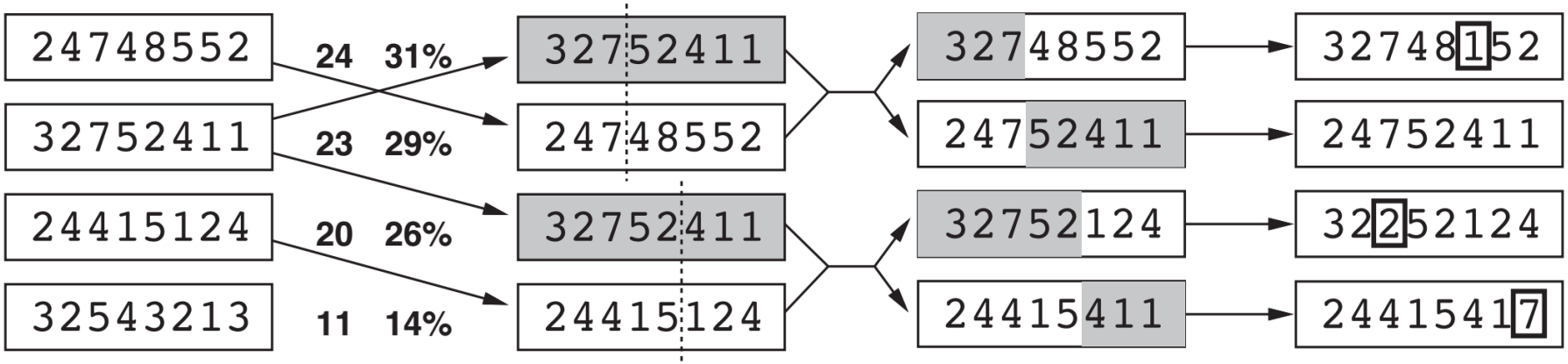
Parent selection mechanism

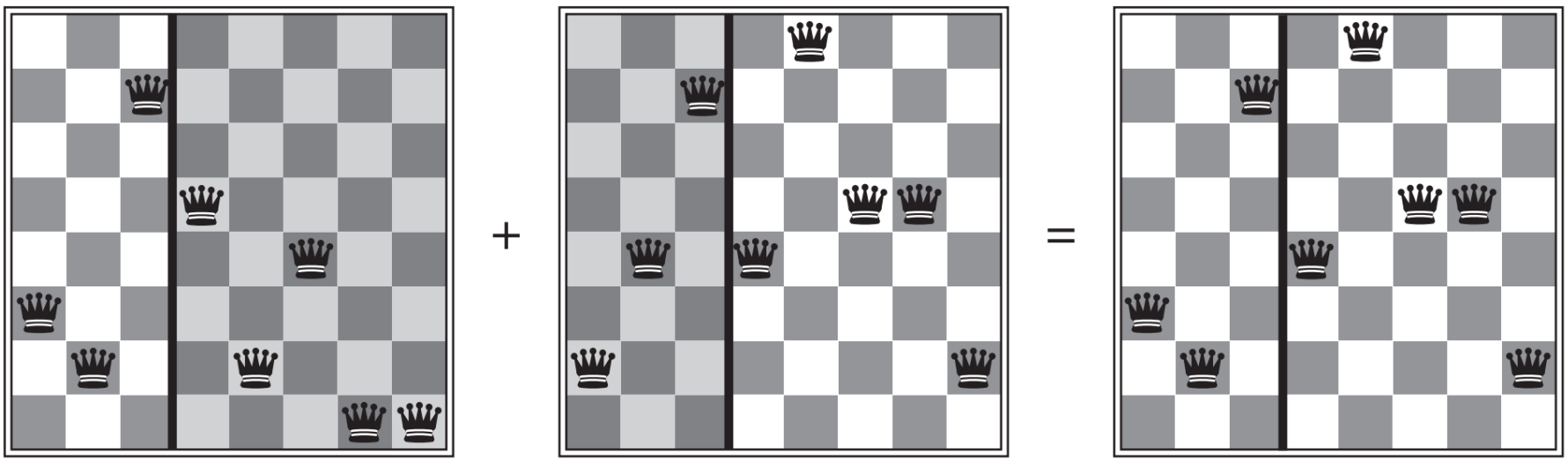
In evolutionary computing, parent selection is typically probabilistic. Thus, high-quality individuals have more chance of becoming parents than those with low quality. Nevertheless, low-quality individuals are often given a small, but positive chance; otherwise the whole search could become too greedy and the population could get stuck in a local optimum.

Survivor selection mechanism

In evolutionary computing the population size is almost always constant. This requires a choice to be made about which individuals will be allowed in to the next generation. This decision is often based on their fitness values, favouring those with higher quality, although the concept of age is also frequently used. In contrast to parent selection, which is typically stochastic, survivor selection is often deterministic. Thus, for example, two common methods are the fitness-based method of ranking the unified multiset of parents and offspring and selecting the top segment, or the age-biased approach of selecting only from the offspring. Survivor selection is also often called the replacement strategy.

مؤلفه‌های اصلی الگوریتم ژنتیک معرفی‌شده در کتاب برای مسئلهٔ ۸-وزیر





A genetic algorithm

function GENETIC-ALGORITHM(*population*, FITNESS-FN) **returns** an individual
  inputs: *population*, a set of individuals
    FITNESS-FN, a function that measures the fitness of an individual

  repeat
    new_population ← empty set
    for $i = 1$ **to** SIZE(*population*) **do**
      $x \leftarrow$ RANDOM-SELECTION(*population*, FITNESS-FN)
      $y \leftarrow$ RANDOM-SELECTION(*population*, FITNESS-FN)
      child ← REPRODUCE(x, y)
      if (small random probability) **then** *child* ← MUTATE(*child*)
      add *child* to *new_population*
    population ← *new_population*
  until some individual is fit enough, or enough time has elapsed
  return the best individual in *population*, according to FITNESS-FN

---

function REPRODUCE(x, y) **returns** an individual
  inputs: x, y, parent individuals

  $n \leftarrow$ LENGTH(x); $c \leftarrow$ random number from 1 to n
  return APPEND(SUBSTRING($x, 1, c$), SUBSTRING($y, c + 1, n$))

(Each mating of two parents produces only one offspring, not two.)

A more clever representation

A genotype, or chromosome, is a permutation of the numbers $1, 2, \ldots, 8$, and a given $g = \langle i_1, i_2, \ldots, i_8 \rangle$ denotes the (unique) board configuration, where the nth column contains exactly one queen placed on the i_nth row. For instance, the permutation $g = \langle 1, 2, \ldots, 8 \rangle$ represents a board where the queens are placed along the main diagonal.

It is easy to see that by using such chromosomes we restrict the search to board configurations where horizontal constraint violations (two queens on the same row) and vertical constraint violations (two queens on the same column) do not occur. In other words, the representation guarantees half of the requirements of a solution—what remains to be minimised is the number of diagonal constraint violations.

Mutation & crossover?

For mutation we can use an operator that randomly selects two positions in a given chromosome, and swaps the values found in those positions. A good crossover for permutations is less obvious, but the mechanism outlined below will create two child permutations from two parents ("Cut-and-crossfill" crossover).

1. Select a random position, the crossover point, $i \in \{1, \ldots, 7\}$
2. Cut both parents into two segments at 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 it already contains
5. Do the same for parent 1 and child 2