Advanced Artificial Intelligence

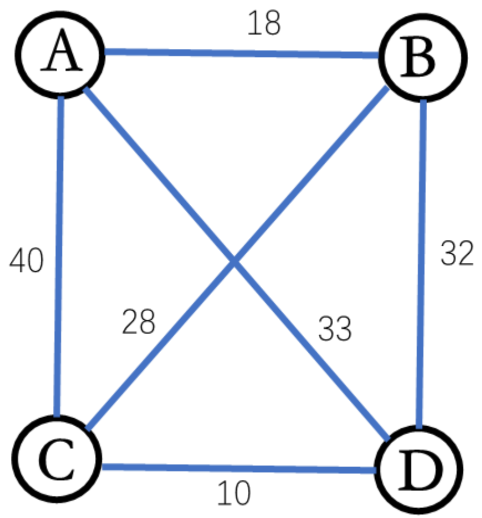
Lab 04

Outline

- Travelling Salesman Problem (TSP)
- Using Genetic Algorithms (GAs) for the Travelling Salesman Problem (TSP)
- The ordinal representation
- The path representation
- Exercise
- References

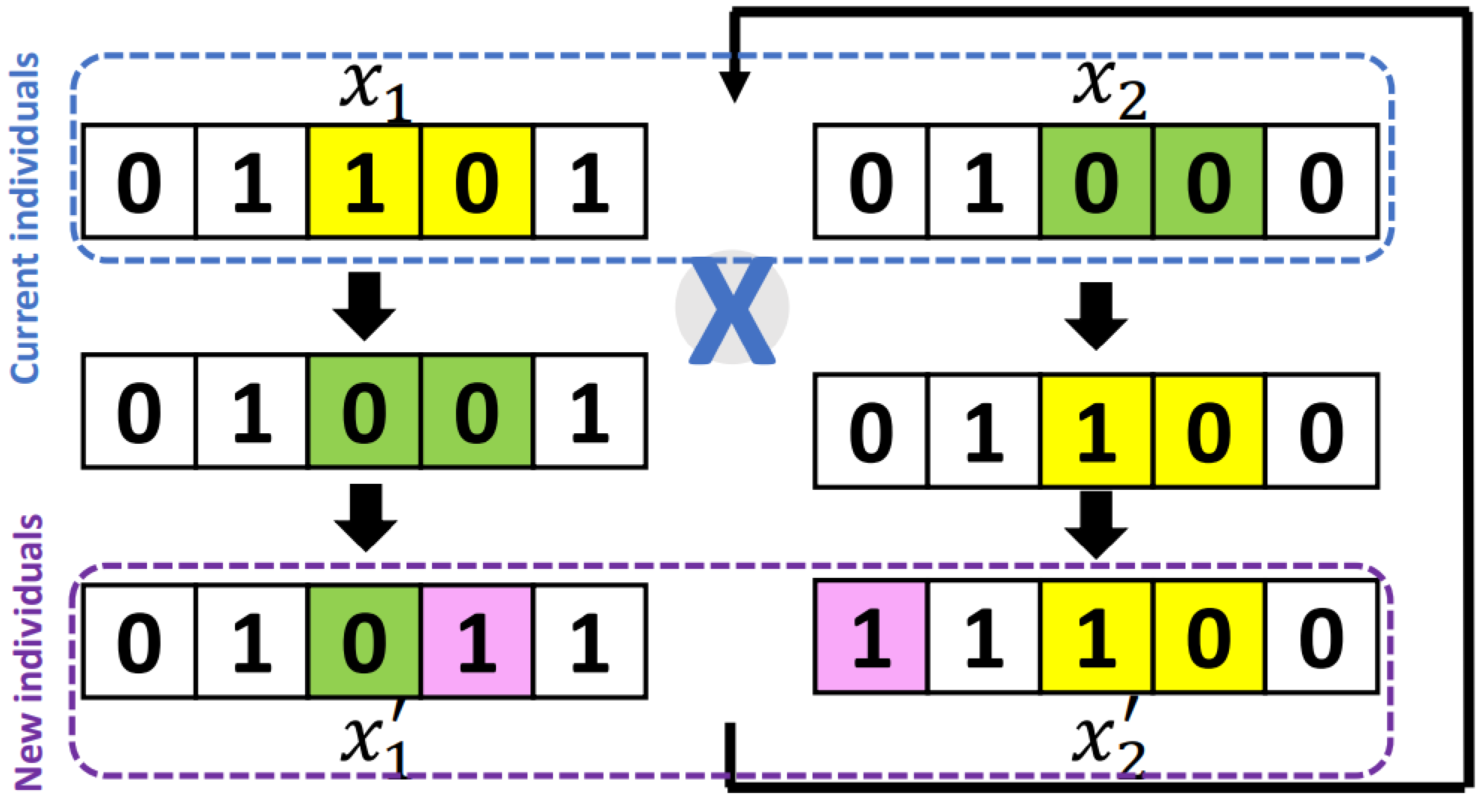
- Travelling Salesman Problem (TSP)

Given a set of cites and the distances between each pair of cities, we aim at finding the shortest route that visits each city once and only once and returns to the origin city. The following figure shows an example solution to a TSP problem, the solution is A→B→C→D→A, which leads to the shortest distance 89.



- Genetic Algorithms (GAs)
  - Binary strings have been used extensively as individuals (chromosomes).
  - Simulate Darwinian evolution.
  - Search operators are only applied to the genotypic representation (chromosome) of individuals.
  - Emphasize the role of recombination (crossover). Mutation is only used as a background operator.
  - Often use roulette-wheel selection.

- Genetic Algorithms (GAs)

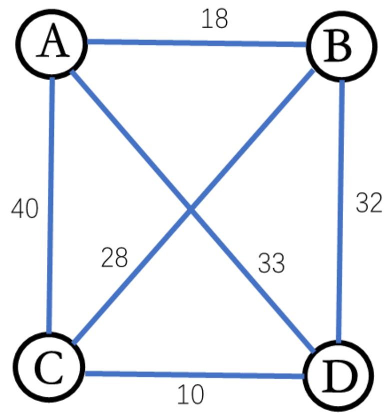


In order to better understand how genetic algorithms can be applied to combinatorial optimization problems, the following equivalence will be useful.

Combinatorial optimization	Genetic algorithm
Encoded solution	Chromosome
Solution	Decoded chromosome
Set of solutions	Population
Objective function	Fitness function

- In the TSP context, each chromosome encodes a solution to the problem (i.e., a tour).
- The fitness of the chromosome is related to the tour length, which in turn depends on the ordering of the cities.
- Since the TSP is a minimization problem, the tour lengths must be transformed so that high fitness values are associated with short tours, and conversely.
- A well-known approach is to subtract each tour length to the maximum tour length found in the current population.

- The genetic algorithm searches the space of solutions by combining the best features of two good tours into a single one.
- Since the fitness is related to the length of the edges included in the tour, it is clear that the edges represent the basic information to be transferred to the offspring.



The preferred research avenue for the TSP:

- to design representational frameworks that are more sophisticated than the bit string.
- to develop specialized operators to manipulate these representations and create feasible sequences.

- The chromosomes shown in the following figure are based on the "path" representation of a tour (on cities 1 to 8).
- The mutation operator and the one-point crossover are still likely to generate infeasible tours when they are applied to this integer representation.

tour	(12564387)	:	**1**	**2**	∣	**5**	**6**	**4**	**3**	**8**	**7**
tour	(14236578)	:	1	4	∣	2	3	6	5	7	8
offspring 1		:	**1**	**2**		2	3	6	5	7	8
offspring 2		:	1	4		**5**	**6**	**4**	**3**	**8**	**7**

- For example, applying the crossover operator at position 2 creates two infeasible offspring.
- None of the two offspring is a permutation of the cities. The TSP, as opposed to most problems tackled by genetic algorithms, is a pure ordering problem.

tour	(12564387)	:	**1**	**2**	|	**5**	**6**	**4**	**3**	**8**	**7**
tour	(14236578)	:	1	4	|	2	3	6	5	7	8
offspring 1		:	**1**	**2**		2	3	6	5	7	8
offspring 2		:	1	4		**5**	**6**	**4**	**3**	**8**	**7**

- All chromosomes carry exactly the same values and differ only in the ordering of these values.
- Specialized permutation operators must be developed for this problem.

tour	(12564387)	:	**1**	**2**	|	**5**	**6**	**4**	**3**	**8**	**7**
tour	(14236578)	:	1	4	|	2	3	6	5	7	8
offspring 1		:	**1**	**2**		2	3	6	5	7	8
offspring 2		:	1	4		**5**	**6**	**4**	**3**	**8**	**7**

The ordinal representation

- In ‘Genetic algorithms for the traveling salesman problem’, the authors developed an ingenious coding scheme for the classical one-point crossover.
- With this coding scheme, the one-point crossover always generates feasible offspring.
- The sequencing information in the two parent chromosomes is not well transferred to the offspring, and the resulting search is close to a random search.

The ordinal representation

The encoding is based on a reference or canonic tour.

- For N = 8 cities, assume that this canonic tour is 12345678.
- Then, the tour to be encoded is processed city by city.
- The position of the current city in the canonic tour is stored at the corresponding position in the resulting chromosome.
- The canonic tour is updated by deleting that city, and the procedure is repeated with the next city in the tour to be encoded.

The ordinal representation

- An example

Current tour	Canonic tour	Ordinal representation
$\underline{1}$ 2 5 6 4 3 8 7	$\underline{1}$ 2 3 4 5 6 7 8	1
1 $\underline{2}$ 5 6 4 3 8 7	$\underline{2}$ 3 4 5 6 7 8	1 1
1 2 $\underline{5}$ 6 4 3 8 7	3 4 $\underline{5}$ 6 7 8	1 1 3
1 2 5 $\underline{6}$ 4 3 8 7	3 4 $\underline{6}$ 7 8	1 1 3 3
1 2 5 6 $\underline{4}$ 3 8 7	3 $\underline{4}$ 7 8	1 1 3 3 2
1 2 5 6 4 $\underline{3}$ 8 7	$\underline{3}$ 7 8	1 1 3 3 2 1
1 2 5 6 4 3 $\underline{8}$ 7	7 $\underline{8}$	1 1 3 3 2 1 2
1 2 5 6 4 3 8 $\underline{7}$	$\underline{7}$	1 1 3 3 2 1 2 1

The ordinal representation

- The resulting chromosome 11332121 can be easily decoded into the original tour 12564387 by the inverse process.
- Two parent chromosomes encoded in this way always generate a feasible offspring when they are processed by the one-point crossover.
- In fact, each value in the ordinal representation corresponds to a particular position in the canonic tour.

parent 1 (12564387)	:	1	1	|	3	3	2	1	2	1
parent 2 (14236578)	:	1	3	|	1	1	2	1	1	1
offspring (12346578)	:	1	1		1	1	2	1	1	1

The ordinal representation

- Exchanging values between two parent chromosomes simply modified the order of selection of the cities in the canonic tour. Consequently, a permutation is always generated.
- For example, the two parent chromosomes in figure 7 encode the tours 12564387 and 14236578, respectively.
- After a cut at position 2, a feasible offspring is created.

parent 1 (12564387)	:	1	1	3	3	2	1	2	1
parent 2 (14236578)	:	1	3	1	1	2	1	1	1
offspring (12346578)	:	1	1	1	1	2	1	1	1

The path representation

- As opposed to the ordinal representation, the path representation is a natural way to encode TSP tours.
- However, a single tour can be represented in 2N distinct ways.
- The factor N can be removed by fixing a particular city at position 1 in the chromosome.
- The crossover operators based on this representation typically generate offspring that inherit either the relative order or the absolute position of the cities from the parent chromosomes.

The path representation

- Crossover operators preserving the absolute position
  - Partially-mapped crossover (PMX) *(Goldberg and Lingle [1])*
    - This operator first randomly selects two cut points on both parents.
    - In order to create an offspring, the substring between the two cut points in the first parent replaces the corresponding substring in the second parent.
    - Then, the inverse replacement is applied outside of the cut points, in order to eliminate duplicates and recover all cities.

The path representation

- Crossover operators preserving the absolute position
  - Partially-mapped crossover (PMX) *(Goldberg and Lingle [1])*

    In the following figure, the offspring is created by first replacing the substring 236 in parent 2 by the substring 564.

    - Hence, city 5 replaces city 2, city 6 replaces city 3, and city 4 replaces city 6 (step 1).
    - Since cities 4 and 5 are now duplicated in the offspring, the inverse replacement is applied outside of the cut points. Namely, city 2 replaces city 5, and city 3 replaces city 4 (step 2).

parent 1 : 1 2 | 5 6 4 | 3 8 7
parent 2 : 1 4 | 2 3 6 | 5 7 8

offspring
(step 1) : 1 4 5 6 4 5 7 8
(step 2) : 1 3 5 6 4 2 7 8

The path representation

- Crossover operators preserving the absolute position
  - Partially-mapped crossover (PMX) *(Goldberg and Lingle [1])*

    In the following figure, the offspring is created by first replacing the substring 236 in parent 2 by the substring 564.

    - In the latter case, city 6 first replaces city 4, but since city 6 is already found in the offspring at position 4, city 3 finally replaces city 6.
    - Multiple replacements at a given position occur when a city is located between the cut points on both parents, like city 6 in this example.

parent 1 : **1 2 | 5 6 4 | 3 8 7**
parent 2 : 1 4 | 2 3 6 | 5 7 8

---

offspring
(step 1) : 1 4 **5 6 4** 5 7 8
(step 2) : 1 3 **5 6 4** 2 7 8

The path representation

- Crossover operators preserving the absolute position
  - Partially-mapped crossover (PMX) *(Goldberg and Lingle [1])*
    - PMX tries to preserve the absolute position of the cities when they are copied from the parents to the offspring.
    - The number of cities that do not inherit their positions from one of the two parents is at most equal to the length of the string between the two cut points.
    - In the example, only cities 2 and 3 do not inherit their absolute position from one of the two parents.

parent 1	:	**1**	**2**	**5**	**6**	**4**	**3**	**8**	**7**
parent 2	:	1	4	2	3	6	5	7	8
offspring									
(step 1)	:	1	4	**5**	**6**	**4**	5	7	8
(step 2)	:	1	3	**5**	**6**	**4**	2	7	8

Exercise

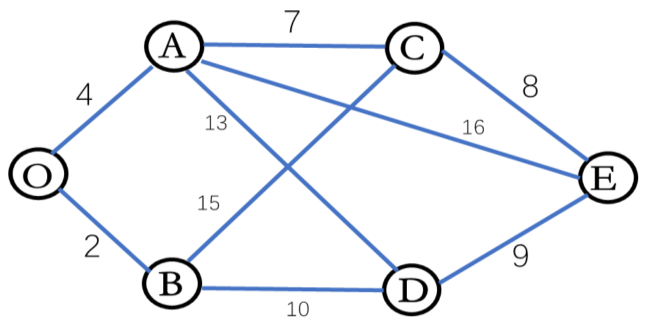
Consider a TSP problem illustrated in the following figure.

- Exercise 1

  Design an appropriate crossover operator and justify your design.

- Exercise 2

  Design an appropriate mutation operator and justify your design.



References

- [1] D.E. Goldberg and R. Lingle, Alleles, loci and the traveling salesman problem, in: Proc. 1st Int. Conf. on Genetic Algorithms (ICGA '85), Carnegie-Mellon University, Pittsburg, PA (1985) pp. 154-159.
- [2] I.M. Oliver, D.J. Smith and J.R.C. Holland, A study of permutation crossover operators on the traveling salesman problem, in: Proc. 2nd Int. Conf. on Genetic Algorithms (ICGA '87), Massachusetts Institute of Technology, Cambridge, MA (1987) pp. 224-230.
- [3] L. Davis, Applying adaptive algorithms to epistactic domains, in: Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI '85), Los Angeles, CA (1985) pp. 162-164.
- [4] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning (AddisonWesley, 1989).
- [5] G. Syswerda, Schedule optimization using genetic algorithms, in: Handbook of Genetic Algorithms, ed. L. Davis (Van Nostrand Reinhold, 1990) pp. 332-349.