Genetic Algorithms and Evolutionary Computing

Lecture 4

Travelling salesman problem

Combinatorial Optimization

Route Planning

- Traveling salesman problem (TSP)
- Vehicle routing problem (VRP)

- **TSP**: very powerful problem-specific methods and effective branch-and-bound methods exist
  → global optimal solution known for many test cases
  → well suited as benchmark problem
  for other solution methods as e.g. GAs.
- **VRP** : closer to practical situations in transport logistics,
  no powerful problem-specific methods,
  requires handling of constraints !

TSP

- See book chapter 4:
  4.5 Permutation representation (p. 67-74)
- See slides corresponding to book:
  ch04-....2014.pdf (p. 36-52)
- **First:**
  - Introduction
  - Other representation & other genetic operators (adjacency representation)

Definition TSP

From "Genetic Algorithms and Genetic Programming. Modern Concepts and Practical Applications", by M. Affenzeller et al.

Graph context:

- a finite complete graph with (integer) weights assigned to the edges
- the objective is to find a Hamiltonian cycle ('tour'), i.e., a cycle passing through all the vertices, of minimum total weight

General asymmetric TSP

$$f : \mathcal{S} \rightarrow \mathbb{R}$$

The aim is to find the optimal tour $s^* \in \mathcal{S}$ such that $f(s^*) \leq f(s_k), \forall s_k \in \mathcal{S}$. In order to state the objective function f we have to introduce a distance matrix $[d_{ij}], d_{ij} \in \mathbb{R}^+$ whose entries represent the distance from a city i to a city j. In that kind of representation the cities are considered as the nodes of the underlying graph. If there is no edge between two nodes, the distance is set to infinity.

Using the notation ~~given in [PS82],~~ that $\pi_k(i)$ represents the city visited next after city i in a certain tour s_k, the objective function is defined as

$$f(s_k) = \sum_{i=1}^{n} d_{i\pi_k(i)} \qquad (8.1)$$

Symmetric TSP: symmetric distance matrix

Euclidean TSP

- Triangle inequality: elements of distance matrix

  $$d_{ij} \leq (d_{ik} + d_{kj}), \forall (i, j, k \in 1, \ldots, n)$$

  not necessarily satisfied when d_{ij} is 'cost'
- Euclidean TSP
  symmetric & triangle inequality satisfied &

  $$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

- Many Euclidean TSP benchmark test cases
- GA do not use Euclidean structure → broadly applicable

Review of Optimal Algorithms

- Total enumeration

TSP has a worst case complexity of $\mathcal{O}(n!)$, total enumeration is only applicable to very small problem instances. For example, even for a rather small and simple 30-city symmetric TSP one would have to consider $\frac{(n-1)!}{2} = \frac{(29)!}{2}$ possible solutions which would require a computational time of about $1.4 * 10^{12}$ years assuming the use of a very powerful computer which can evaluate 100,000 million routes per second.

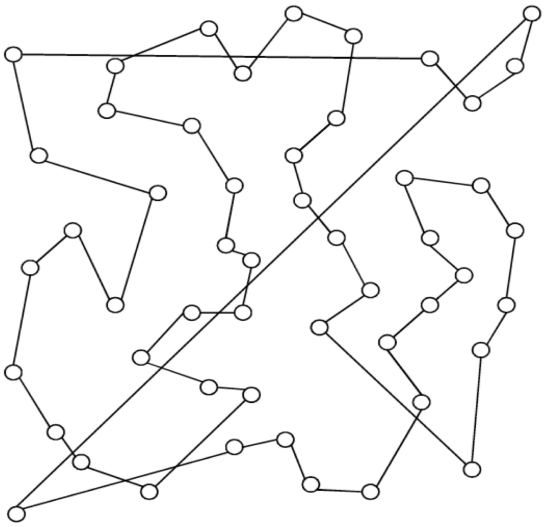
- Integer programming

Conclusion:

Unfortunately, the methods for constructing suitable cutting-planes are far away from working in an automated way and require a well versed user. Therefore, the main area of application of integer programming for the TSP is the exact solution of some large benchmark problems in order to get reference problems for testing certain heuristics.

Approximation Algorithms & Heuristics

- Nearest Neighbor Heuristics



At the beginning this strategy works well, but 'long' stretches have to be inserted when only a few cities are left.

Approximation Algorithms & Heuristics

- Local search

Typical representatives of route improving heuristics are the so-called k-change methods that examine a k-tuple of edges of a given tour and test whether or not a replacement of the tour segments effects an improvement of the actual solution quality.

A lot of established improvement methods are based upon local search strategies also causing the nomenclature "neighborhood search." The basic idea is to search through the surroundings of a certain solution s_i in order to replace s_i by an eventually detected "better" neighbor s_j.

Choosing a neighborhood of larger size can cause problems concerning computational time whereas a rather small neighborhood increases the probability of getting stuck in a local optimum [PS82]. The search process for a better solution in the neighborhood is performed successively until no better solution can be detected. Such a point is commonly referred to as a local minimum

2-opt method

The 2-Opt Method

The most popular local edge-recombination heuristic is the 2-change replacement of two edges. In this context the neighborhood is defined in the following way:

> A tour s_i is adjacent (neighboring) to a tour s_j *if and only if* s_j can be derived from s_i by replacing two of s_i's edges.

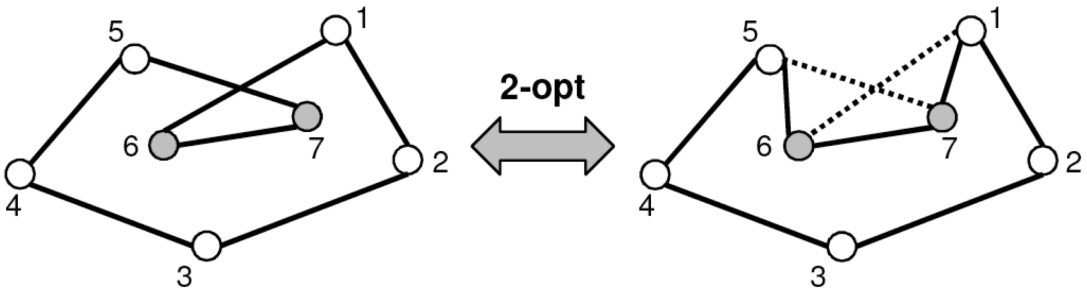


FIGURE 8.2: Example of a 2-change for a TSP instance with 7 cities.

$$(c_1 \ldots c_i c_{i+1} \ldots c_j c_{j+1} \ldots c_n) \longleftrightarrow (c_1 \ldots c_i c_j \ldots c_{i+1} c_{j+1} \ldots c_n)$$

Generalization

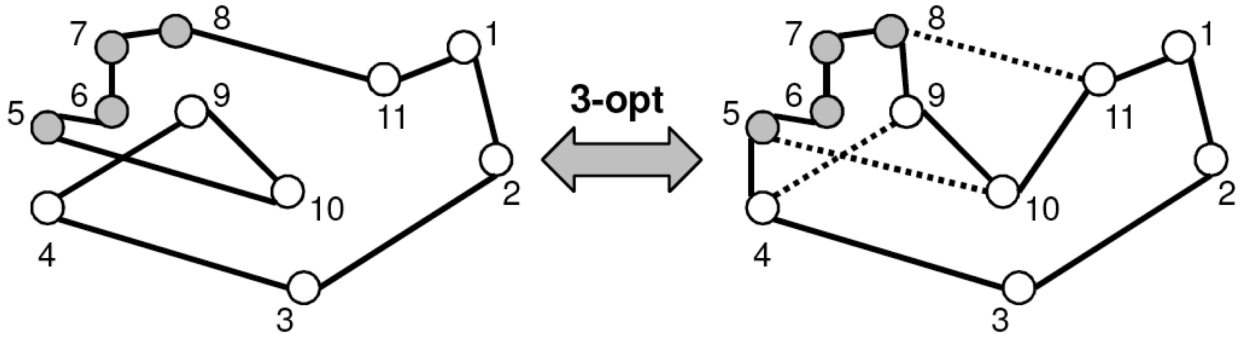


FIGURE 8.3: Example of a 3-change for a TSP instance with 11 cities.

$$(c_1 \dots c_i c_{i+1} \dots c_j c_{j+1} \dots c_k c_{k+1} \dots c_n) \longleftrightarrow (c_1 \dots c_i c_{j+1} \dots c_k c_{i+1} \dots c_j c_{k+1} \dots c_n)$$

GA: Problem Representations

- Adjacency representation
- Ordinal representation
- Path representation

+ suitable genetic operators

Adjacency representation

In the adjacency representation [GGRG85] a tour is represented as a list of n cities where city j is listed in position i if and only if the tour leads from city i to city j. Thus, the list

$$(7 \quad 6 \quad 8 \quad 5 \quad 3 \quad 4 \quad 2 \quad 1)$$

represents the tour

$$3-8-1-7-2-6-4-5.$$

In the adjacency representation any tour has its unique adjacency list representation. An adjacency list may represent an illegal tour. For example,

$$(3 \quad 5 \quad 7 \quad 6 \quad 2 \quad 4 \quad 1 \quad 8)$$

represents the following collection of cycles:

$$1-3-7, \quad 2-5, \quad 4-6, \quad 8$$

Obviously, the classical crossover operator(s) (single or n-point crossover) are very likely to return illegal tours for the adjacency representation. Therefore, the use of a repair operator becomes necessary.

Adjacency: crossover operators

- Alternating Edge Crossover
- Subtour Chunks Crossover
- Heuristic Crossover

- - : Low performance
- + : Allows schemata analysis
  (see chapter 16. Theory)

Alternating edge crossover

The alternating edge crossover chooses an edge from the first parent at random. Then, the partial tour created in this way is extended with the appropriate edge of the second parent, then by the appropriate edge of the first parent, etc.

→ partial tour is extended by choosing edges from alternating parents.

If an edge is chosen which would produce a cycle into the partial tour, an edge is selected from the edges which do not produce a cycle.

For example, the result of an alternating edge crossover of the parents

$$(2\ 3\ 8\ 7\ 9\ 1\ 4\ 5\ 6) \qquad (7\ 5\ 1\ 6\ 9\ 2\ 8\ 4\ 3)$$

Bad example: not legal tour: cycle 7-4

could for example be

$$(2\ 5\ 8\ 7\ 9\ 1\ 6\ 4\ 3)$$

Good subtours are often disrupted !!

Subtour Chunks Crossover

- cf. alternating edge crossover, but using subtours instead of edges
- A random subtour of the first parent is chosen, and this partial tour is extended by choosing a suitable subtour of random length from the second parent. Then, the partial tour is extended by taking subtours from alternating parents. If a subtour would lead to an illegal tour, then it is not added; instead, an edge is added which is randomly chosen from the edges that do not produce a cycle.

Heuristic Crossover

- Starts with randomly selecting a city
- The edges starting from this city (in the parents) are compared and the shorter of these two edges is chosen. The city on the other side of the chosen edge is selected etc.
- If at some stage a new edge introduces a cycle into the partial tour, then the tour is extended with an edge chosen at random from the remaining edges which do not introduce cycles
- The heuristic crossover operator performs far better than the other two operators; still, the performance of the heuristic operator is not remarkable either

Ordinal Representation

The easiest way to explain the ordinal representation is probably by giving an example. Assume, for instance, that the ordered list L is given as

$$L = (1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7).$$

Now the tour

$$1 - 2 - 7 - 5 - 6 - 3 - 4$$

in ordinal representation is given as

$$T = (1 \quad 1 \quad 5 \quad 3 \quad 3 \quad 1 \quad 1).$$

The first number in *T* is 1, so take the 1st city in *L* as the first city of the tour, i.e. 1; *remove it from L*; the resulting partial tour is 1. The second number in *T* is also a 1, so take the 1st city in the *current L*, i.e. 2; *remove it from L*; partial tour 1 → 2. The next item in *T* is 5, take the 5th city in the *current L,* i.e. 7; *remove it from L*; partial tour 1 → 2 → 7. Continue until all elements of *T* have been accounted for.

Path representation

- Most natural representation (??)

$$1 - 2 - 7 - 5 - 6 - 3 - 4$$

is simply represented by

$$(1 \quad 2 \quad 7 \quad 5 \quad 6 \quad 3 \quad 4).$$

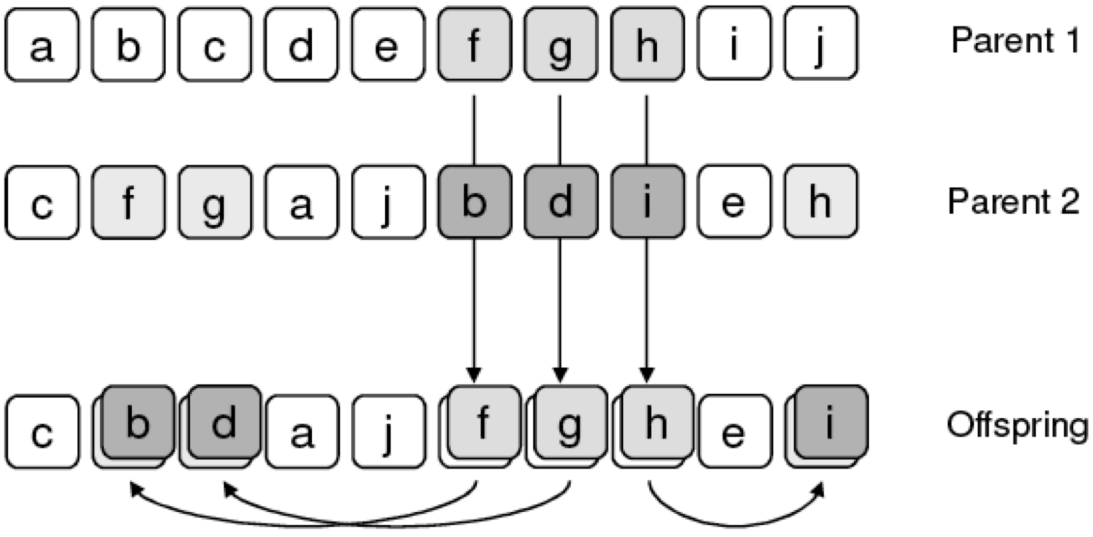
one phenotype → *n* genotypes !!!

Crossover operators:

- Partially Matched Crossover (PMX)
- Order Crossover (OX)
- Cyclic Crossover (CX)
- Edge Recombination Crossover (ERX)

See slides ch04-....2014.pdf (p. 36-52)

Partially Matched Crossover (PMX)



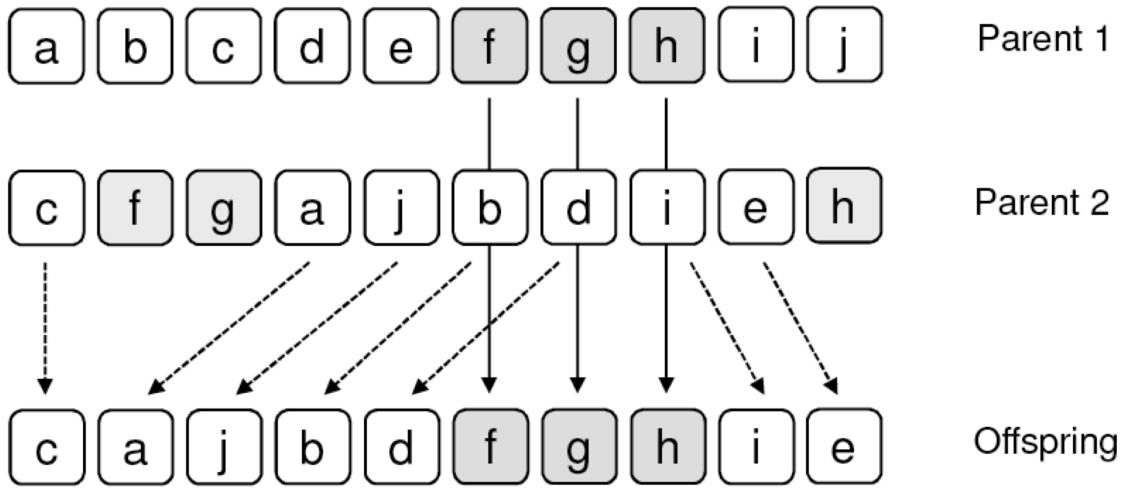
mapping may have to be applied several times!

The PMX operator tries to keep the positions of the cities in the path representation; these are rather irrelevant …

Partially Matched Crossover (PMX)

The performance of this operator for the TSP is rather poor, but this operator could perform well for other combinatorial optimization problems

Order crossover (OX)



Since a much higher number of edges is maintained, the results are much better compared to the results achieved using the PMX operator.

Cycle crossover (CX)

From theoretical and empirical results:

CX operator: slightly better results than the PMX operator.

Results of both position preserving operators CX and PMX are worse than those obtained with OX

(TSP: more important to keep sequences rather than positions)

Edge recombination operator (ERX)

Construct “edge lists”: gives edges for each parent that start or finish in it.

Choose initial city from one of the two parents → “current city”

While remaining unvisited cities **do**

remove all occurrences of “current city” from the edge lists

if “current city” has entities in its edge list **then**

determine which of the cities in the edge-list of “current city” has the fewest entities in its own edge list; city with the fewest entities → “current city” (ties broken at random)

else

randomly choose an unvisited city → “current city”

endif

endwhile

Edge recombination operator (ERX)

Parents (1 2 3 4 5 6 7 8 9) (4 1 2 8 7 6 9 3 5)

city	connected cities
1	9, 2, 4
2	1, 3, 8
3	2, 4, 9, 5
4	3, 5, 1
5	4, 6, 3
6	5, 9, 7
7	6, 8
8	7, 9, 2
9	8, 1, 6, 3

9: 4 ‘links’; 2 & 4: 3 ‘links’

Offspring: (1 4 5 6 7 8 2 3 9)

“enhanced edge recombination crossover (EERX)”:
gives priority to those edges starting from the current city which are present in both parents.

Sequential Constructive crossover

Zakir H. Ahmed, Genetic Algorithm for the Traveling Salesman Problem using Sequential Constructive Crossover Operator, International Journal of Biometrics & Bioinformatics (IJBB) Vol. 3(6), pp. 96 – 105, 2010

- The sequential constructive crossover (SCX) operator constructs an offspring from a pair of parents using better edges on the basis of their values that may be present in the parents' structure maintaining the sequence of nodes in the parent chromosomes.

Sequential Constructive crossover

1. Current node p =1
2. Sequentially search both of the parent chromosomes and consider the first ‘legitimate node' (the node that is not yet visited) appearing after 'node p’ in each parent. If no 'legitimate node' after 'node p’ is present in any of the parent, search sequentially the nodes {2, 3, ..., n} and consider the first 'legitimate' node, and go to Step 3.
3. Suppose 'node α' and 'node β' are found in 1st and 2nd parent respectively, then go to Step 4.
4. If $c_{p\alpha} < c_{p\beta}$, then select 'node α', otherwise 'node β' as the next node and concatenate it to the partially constructed offspring chromosome. If the offspring is a complete chromosome, then stop, otherwise, rename the present node as 'node p' and go to Step 2.

Sequential Constructive crossover

Instance	n	Opt. Sol.	ERX			GNX			SCX		
			Best (%)	Avg (%)	Avg Time	Best (%)	Avg (%)	Avg Time	Best (%)	Avg (%)	Avg Time
br17	17	39	0.00	0.00	2.10	0.00	0.00	0.26	0.00	0.00	0.11
ftv33	34	1286	1.94	4.98	70.22	15.47	19.49	7.64	0.00	3.58	2.25
ftv35	36	1473	3.19	5.20	76.39	17.58	18.56	1.48	0.00	0.59	9.69
ftv38	39	1530	4.38	5.45	160.87	6.99	13.18	4.03	0.24	0.46	6.89
p43	43	5620	1.44	1.89	213.99	2.46	2.58	22.33	0.05	0.10	22.98
ftv44	45	1613	5.46	6.39	157.23	14.01	15.71	18.46	0.62	0.93	19.22
ftv47	48	1776	5.97	8.48	200.70	20.10	20.38	42.67	0.51	1.73	25.99
ry48p	48	14422	2.04	2.31	185.59	15.18	18.13	39.33	0.59	0.60	25.73
ft53	53	6905	18.03	19.51	122.75	18.29	23.33	29.35	1.03	1.77	36.73
ftv55	56	1608	13.18	14.41	328.59	22.51	24.71	23.74	0.62	1.45	35.11
ftv64	65	1839	25.24	27.48	326.95	25.23	29.87	91.39	0.49	1.54	76.56
ft70	70	38673	10.40	10.56	561.14	6.53	7.81	90.13	0.70	0.86	74.19
ftv70	71	1950	30.41	34.56	432.31	20.72	23.04	135.97	2.15	2.75	58.69
kro124p	100	36230	30.96	37.15	542.57	25.72	28.58	178.64	4.24	4.93	142.02
ftv170	171	2755	62.45	66.15	526.46	51.00	60.69	483.21	6.13	8.93	259.60

TABLE 3: Summary of the results by the crossover operators for asymmetric TSPLIB instances.

%: relative ‘error’ in comparison with optimal solution

Mutation

- 2–opt and 3–opt
- Exchange Mutation (swap mutation, point mutation, reciprocal exchange, order-based mutation)

  selects two cities of the tour randomly & exchange
- Insertion Mutation:

  randomly chooses a city, removes it & inserts it at a randomly selected place
- Simple Inversion Mutation:

  randomly selects two cut points and reverses the string
- Inversion Mutation:

  randomly selects a subtour, removes it, and inserts it in reverse order at a randomly chosen position.