ALGORITHM "CHEAPEST INSERTION"

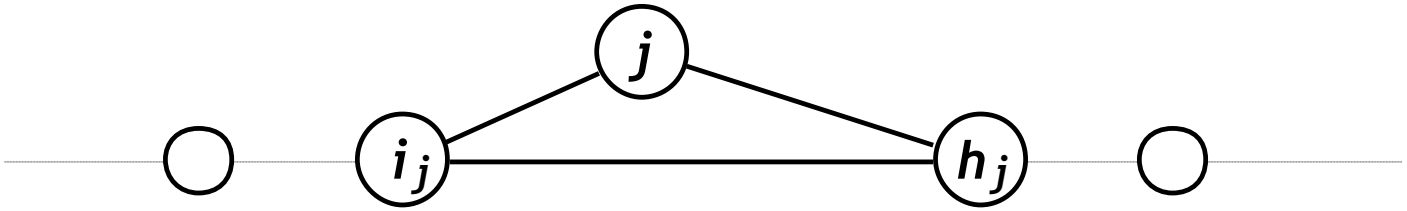
Version for STSP

1. Choose the two "furthest vertices" i and k as "initial subtour"

   ($c_{ik} = \max \{c_{hj} : (h, j) \in A\}$);

   set V':= V \ {i} \ {k} (set of the "unvisited" vertices).

2. For each unvisited vertex j ($j \in V'$) determine the "best insertion" of j in the current subtour (i. e., determine the arc (i_j, h_j) of the current subtour such that $e_j := c_{i_j,j} + c_{j,h_j} - c_{i_j,h_j}$ is minimum):



3. Determine the unvisited vertex k ($k \in V'$) such that e_k is minimum

   ($e_k = \min \{e_j : j \in V'\}$); insert vertex k between vertices i_k and h_k;

   V':= V' \ {k}. If V' $\neq \varnothing$ then return to STEP 2. Else STOP.

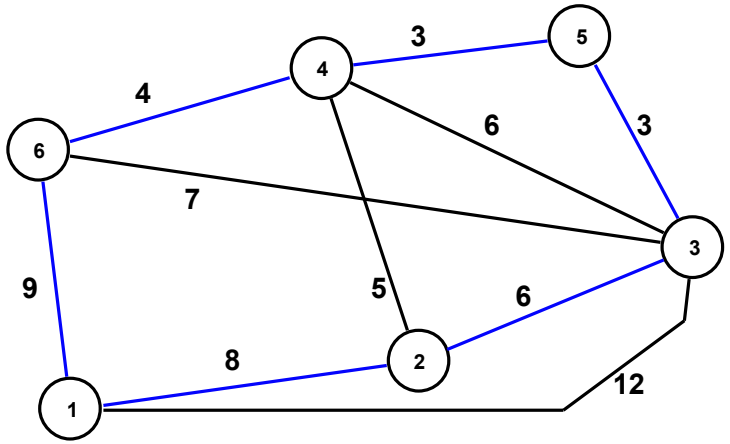
ALGORITHM "CHEAPEST INSERTION" (2)

- Time complexity: $O(n^3)$.

- Good results if the triangle inequality holds.

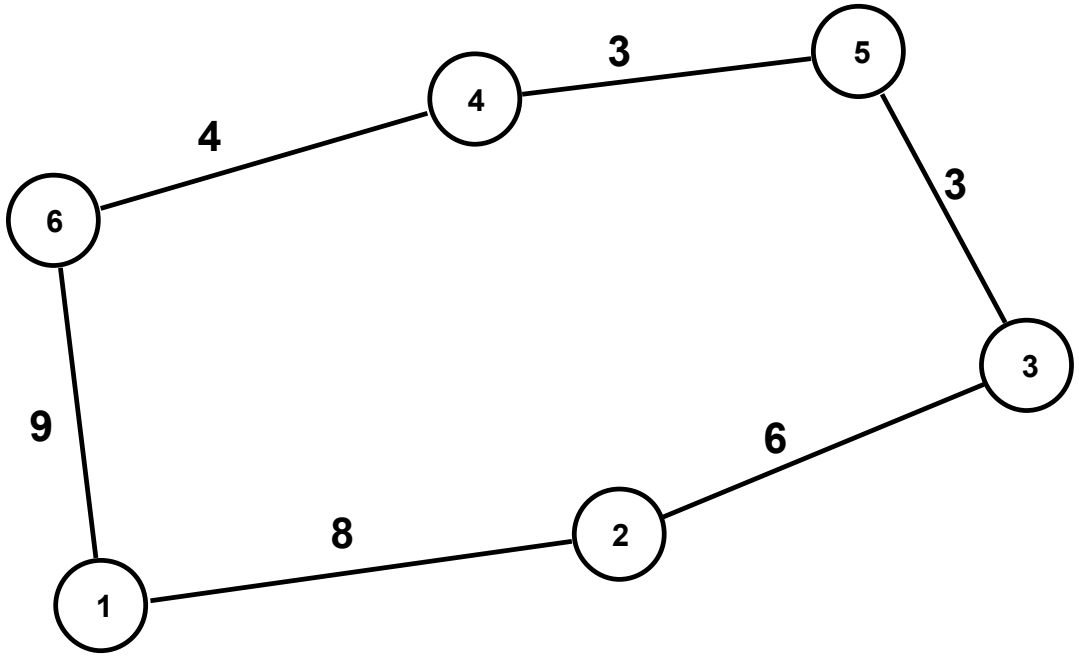
  * Version for ATSP?
  * Is "Cheapest Insertion" a greedy algorithm?

Example 2



Opt = 33

Algorithm Cheapest Insertion



(1, 3)

(1, 2, 3)

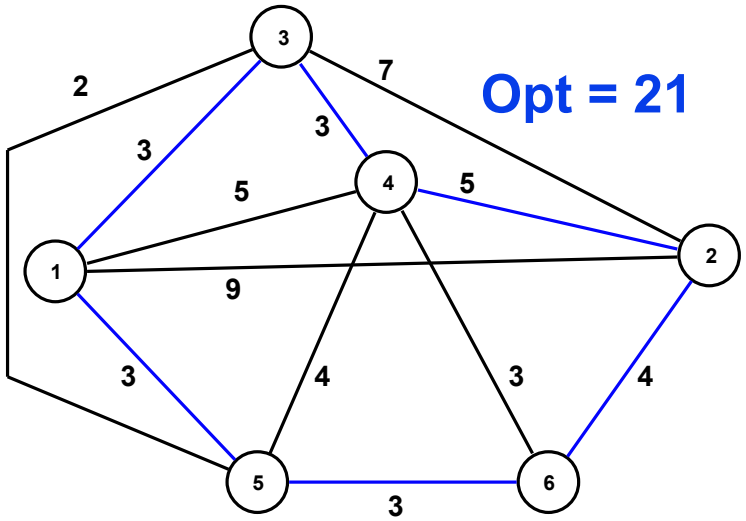
(1, 6, 3, 2)

(1, 6, 4, 3, 2)

(1, 6, 4, 5, 3, 2)

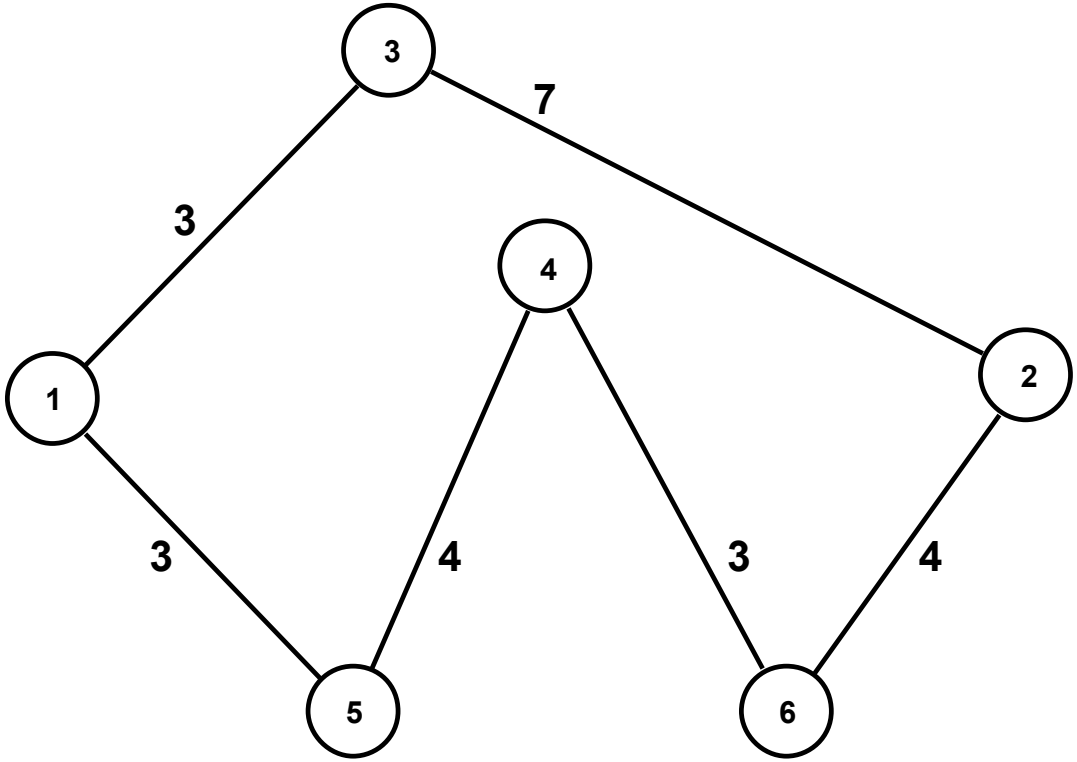
Solution cost: 33 (optimal)

Example 3



Opt = 21

Algorithm Cheapest Insertion



(1, 2)

(1, 4, 2)

(1, 4, 2, 3)

(1, 5, 4, 2, 3)

(1, 5, 4, 6, 2, 3)

Solution cost: 24

GREEDY ALGORITHM "MULTI-PATH" (Bentley, 1990)

Version for STSP

1. Order the arcs of the graph according to non-decreasing values of the associated costs (c_{ij}).

2. If the insertion of the next arc (i, j) into the current solution (given by a family of paths) is feasible, i.e. the current degrees of vertices i and j are ≤ 1 and arc (i, j) forms no subtour (i. e. there is no path from j to i whose number of arcs is less than n - 1), then "select" arc (i, j).

3. If the number of the selected arcs is less than n then return to STEP 2. Else STOP.

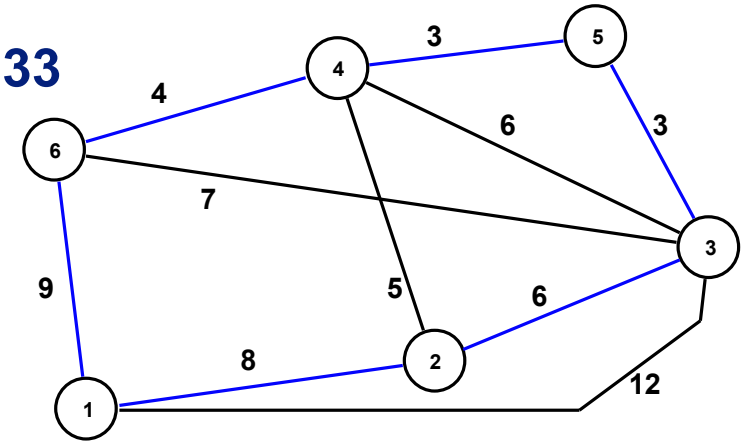
❖ "RANDOMIZED" VARIANT

At STEP 2, consider the next 2 "feasible" arcs:

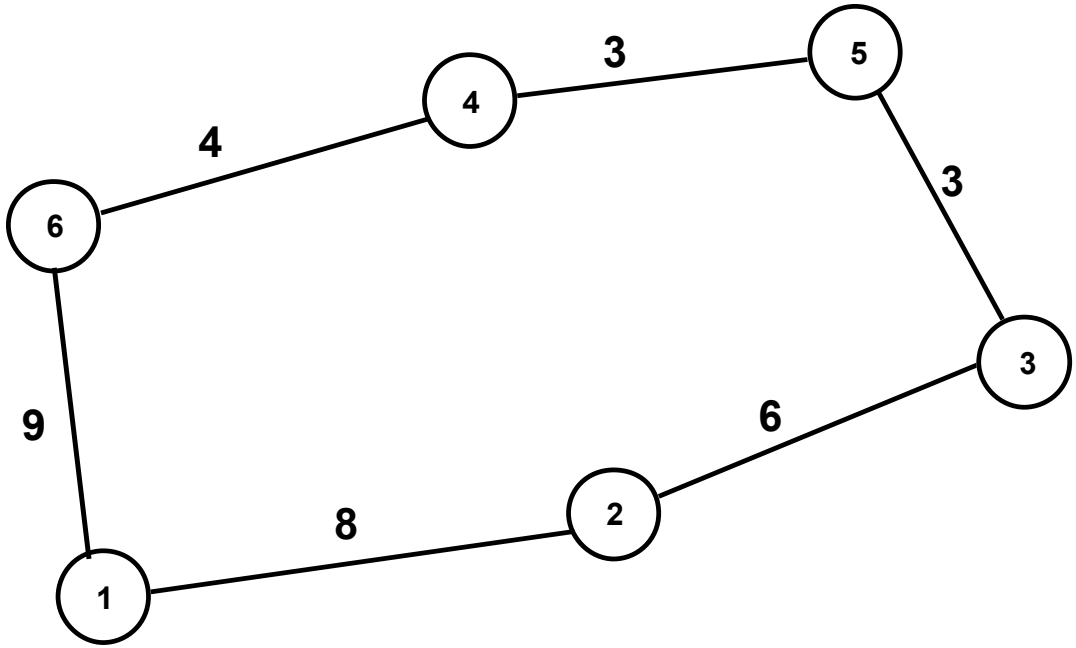
- select the next arc with probability 2/3,
- select the following arc with probability 1/3.

Example 2

Opt = 33



Algorithm Multi-Path

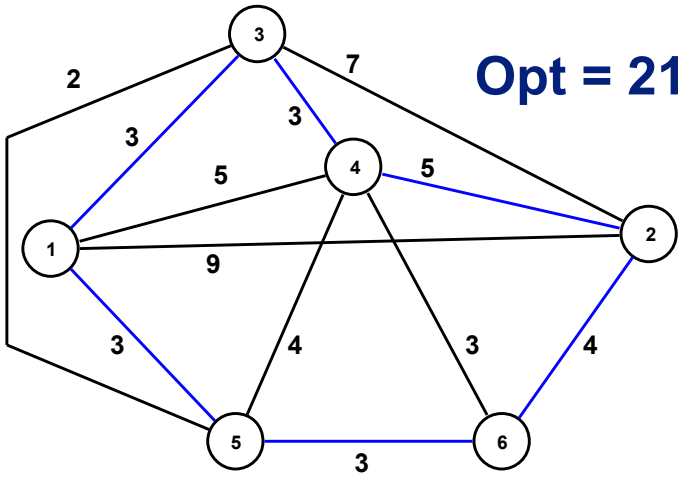


Ordered arcs:

(4, 5)	OK
(3, 5)	OK
(4, 6)	OK
~~(2, 4)~~	NO
(2, 3)	OK
~~(3, 4)~~	NO
~~(3, 6)~~	NO
(1, 2)	OK
(1, 6)	OK STOP
(1, 3)	

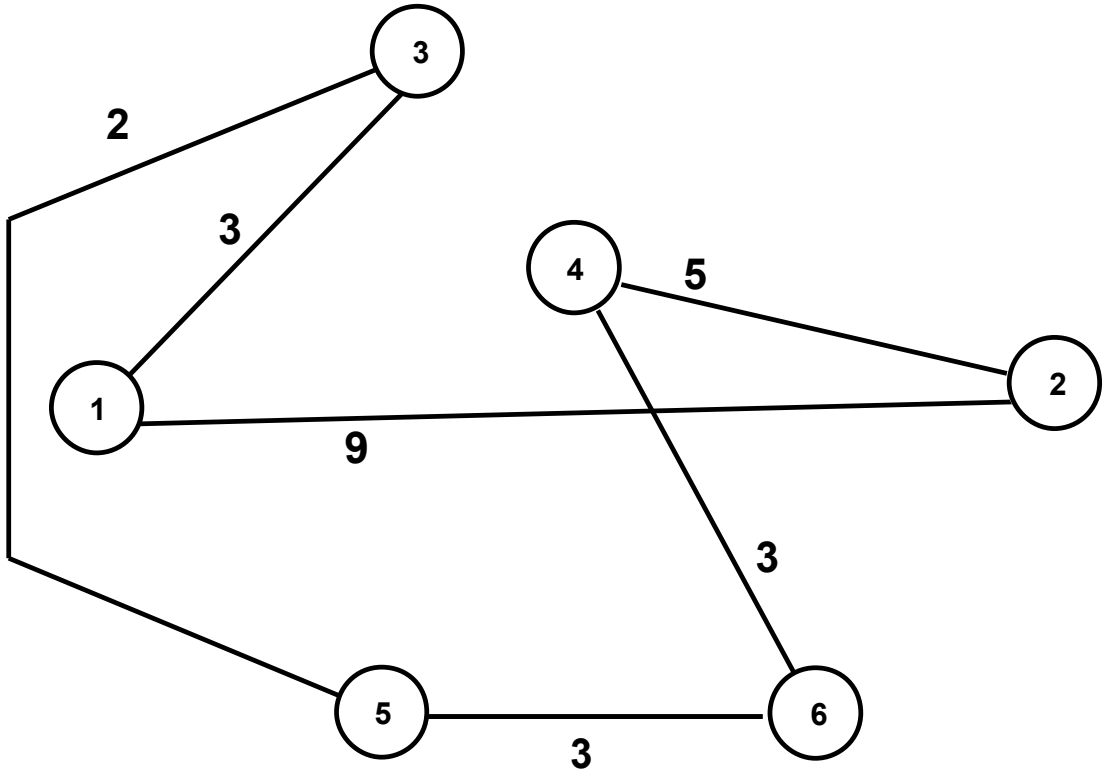
Solution cost: 33 (optimal)

Example 3



Opt = 21

Algorithm Multi-Path



Solution cost: 25

Ordered arcs:

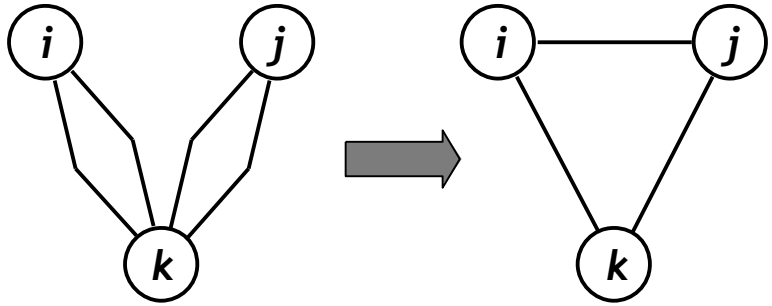
(3, 5)	OK
(1, 3)	OK
~~(1, 5)~~	NO
~~(3, 4)~~	NO
(4, 6)	OK
(5, 6)	OK
~~(2, 6)~~	NO
~~(4, 5)~~	NO
~~(1, 4)~~	NO
(2, 4)	OK
~~(2, 3)~~	NO
(1, 2)	OK STOP

ALGORITHM "SAVINGS" (Clark-Wright, 1964; VRP)

(version for STSP, good results if the triangle inequality holds)

- Choose any vertex *k* as the "depot". Set V' := V \ {*k*}.
- For each arc (*i*, *j*) ∈ A such that *i* ∈ V' and *j* ∈ V' compute the score ("saving"):

$$s_{ij} := c_{ik} + c_{kj} - c_{ij}$$



saving corresponding to the direct connection of vertex *i* with vertex *j* "bypassing" the depot *k*.

$$s_{ij} := (c_{ki} + c_{ik} + c_{kj} + c_{jk}) - (c_{ki} + c_{ij} + c_{jk})$$

ALGORITHM "SAVINGS"

1. For each vertex $i \in V'$ define the subtour (k, i, k);
   order the arcs $(i, j) \in A$ (with $i \in V'$, $j \in V'$) according to non-increasing values of the corresponding savings s_{ij}.

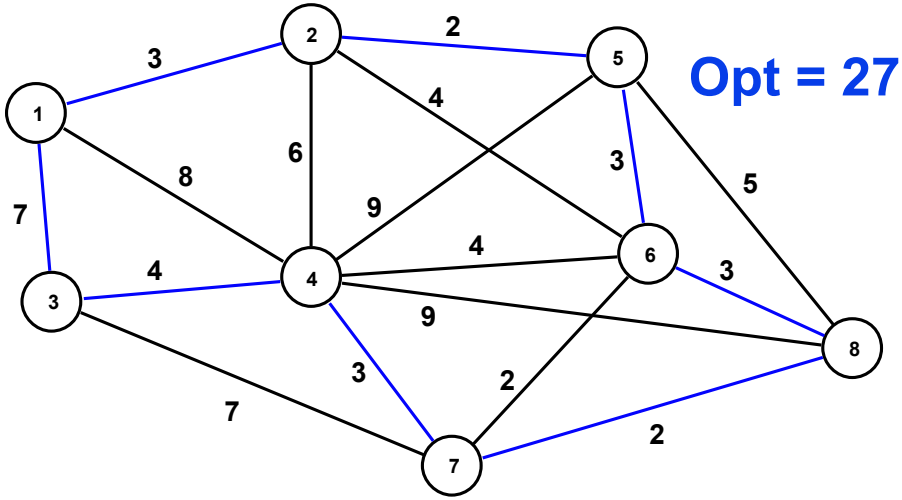
2. If the insertion of the next arc (i, j) into the current solution (given by a family of subtours passing through k) is feasible, i.e. the current degrees (w.r.t. vertices belonging to V') of vertices i and j are ≤ 1 and arc (i, j) forms no subtour containing only vertices belonging to V',

   then remove arcs (i, k) and (k, j) and insert arc (i, j).

1. If the number of the "inserted" arcs is less than n - 2 then return to STEP 2. Else STOP.

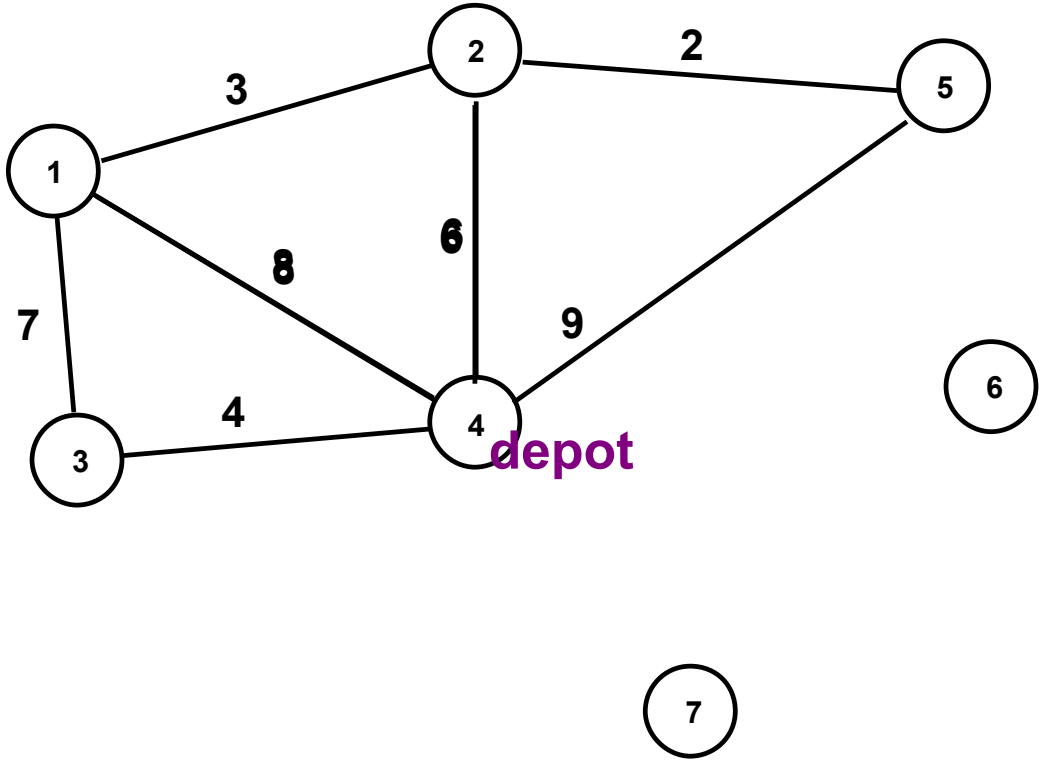
- Time complexity: $O(n^2 \log n)$
- Version for ATSP?

Example 1



Opt = 27

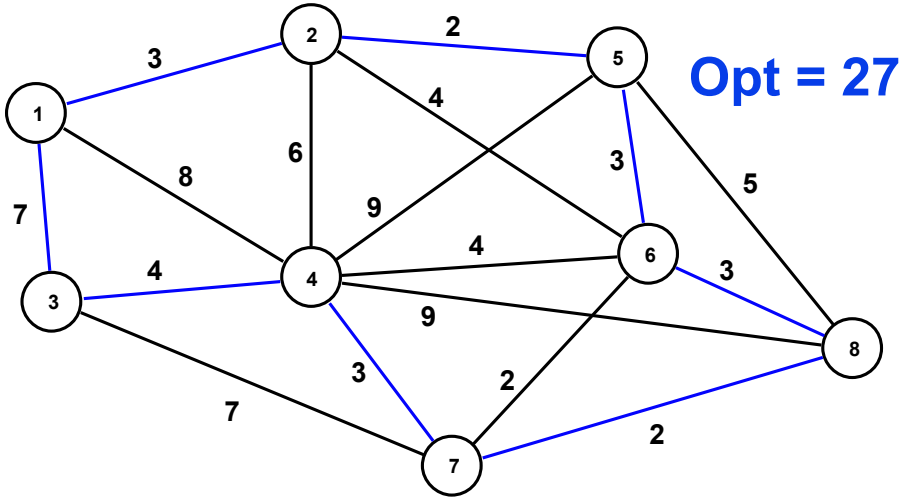
Algorithm "Savings"



k = 4. Savings: Ordered arcs

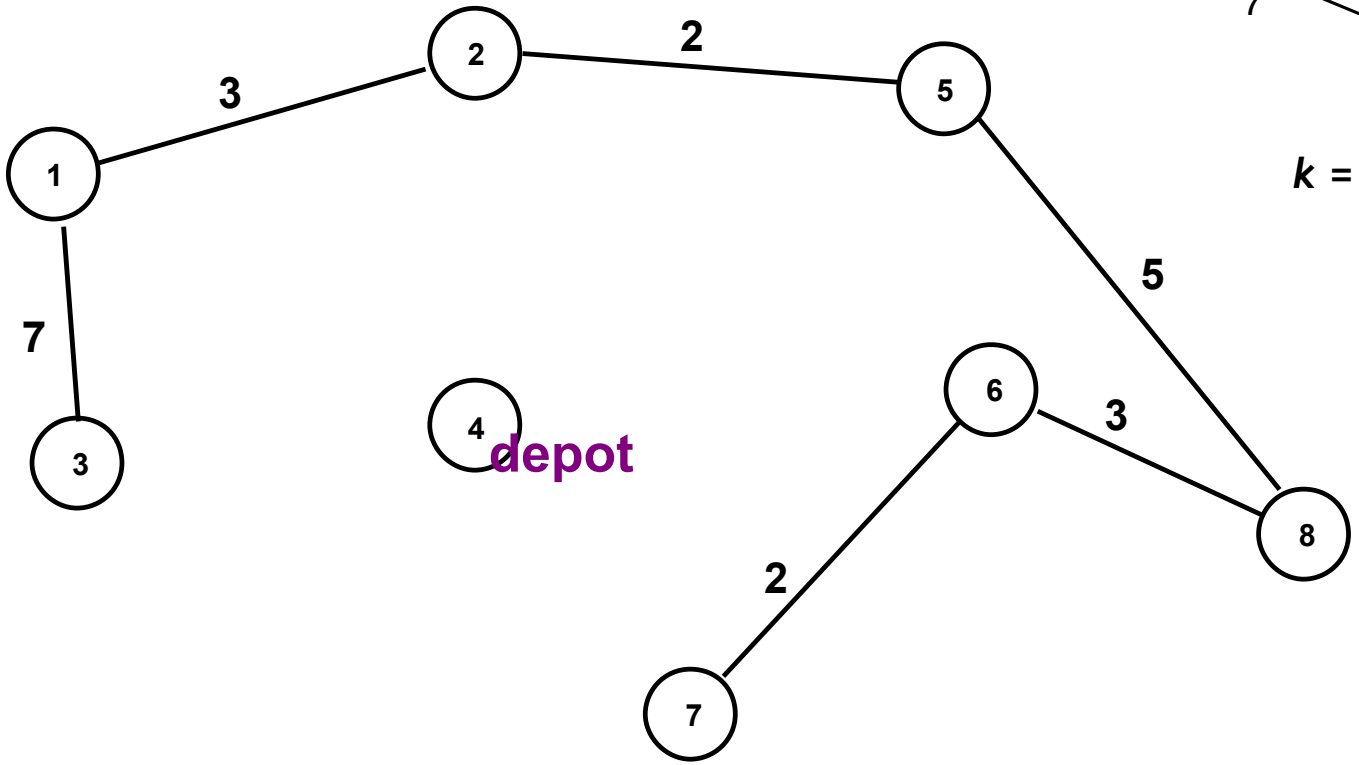
$S_{1,2} = 11$	(2, 5)
$S_{1,3} = 5$	(5, 8)
$S_{2,5} = 13$	(1, 2)
$S_{2,6} = 6$	(5, 6)
$S_{3,7} = 0$	(6, 8)
$S_{5,6} = 10$	(7, 8)
$S_{5,8} = 13$	(2, 6)
$S_{6,7} = 5$	(1, 3)
$S_{6,8} = 10$	(6, 7)
$S_{7,8} = 10$	(3, 7)

Example 1



Opt = 27

Algorithm "Savings" (2)

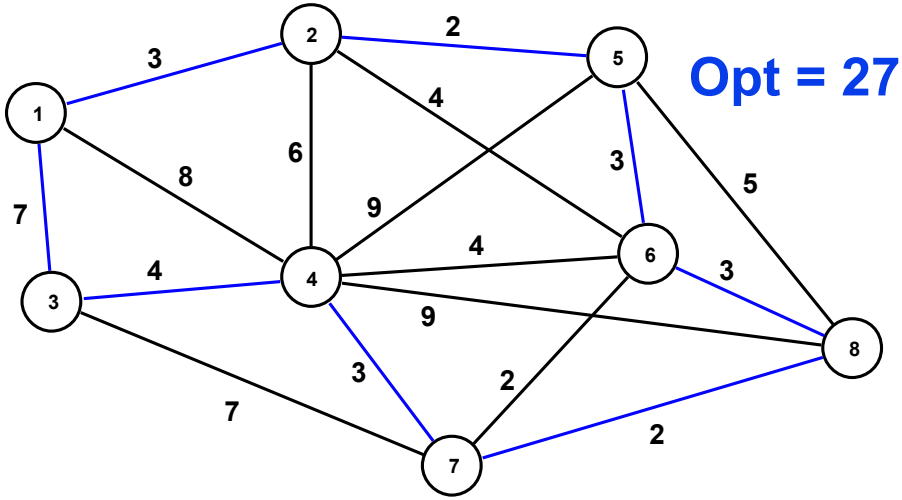


k = 4. **Ordered arcs:**

(2, 5)	OK
(5, 8)	OK
(1, 2)	OK
~~(5, 6)~~	NO
(6, 8)	OK
~~(7, 8)~~	NO
~~(2, 6)~~	NO
(1, 3)	OK
(6, 7)	OK
(3, 7)	

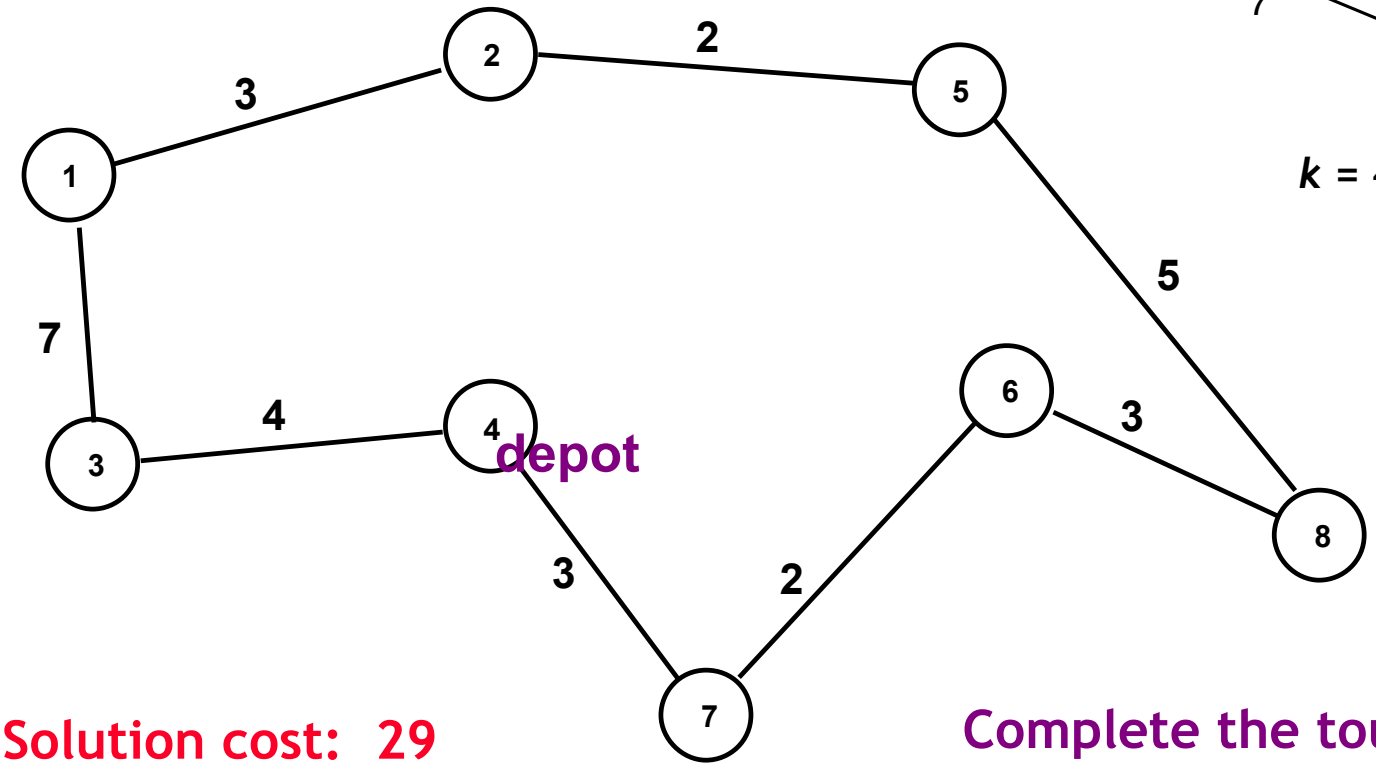
STOP

Example 1



Opt = 27

Algorithm "Savings" (3)



k = 4. **Selected arcs:**

(2, 5)
(5, 8)
(1, 2)
(6, 8)
(1, 3)
(6, 7)

Solution cost: 29

Complete the tour with arcs

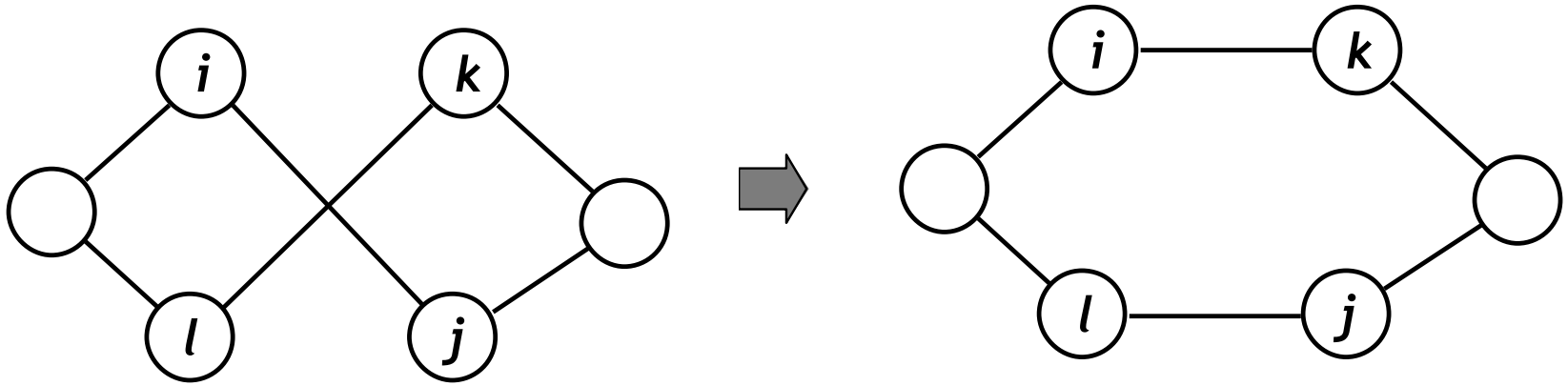
(4, 3) and (7, 4)

LOCAL SEARCH ALGORITHMS

- Sequence of "feasible exchanges" of *r* arcs belonging to the current tour with *r* arcs not belonging to the current tour (an exchange is feasible iff it leads to a new tour).
- If a feasible exchange reduces the global cost of the tour then the exchange is performed.
- Two versions:
  1. update the current tour as soon as an improving exchange is found;
  2. determine the "best" exchange (corresponding to the maximum improvement) and perform it.
- "r-optimal" algorithms (Lin, 1965).

STSP

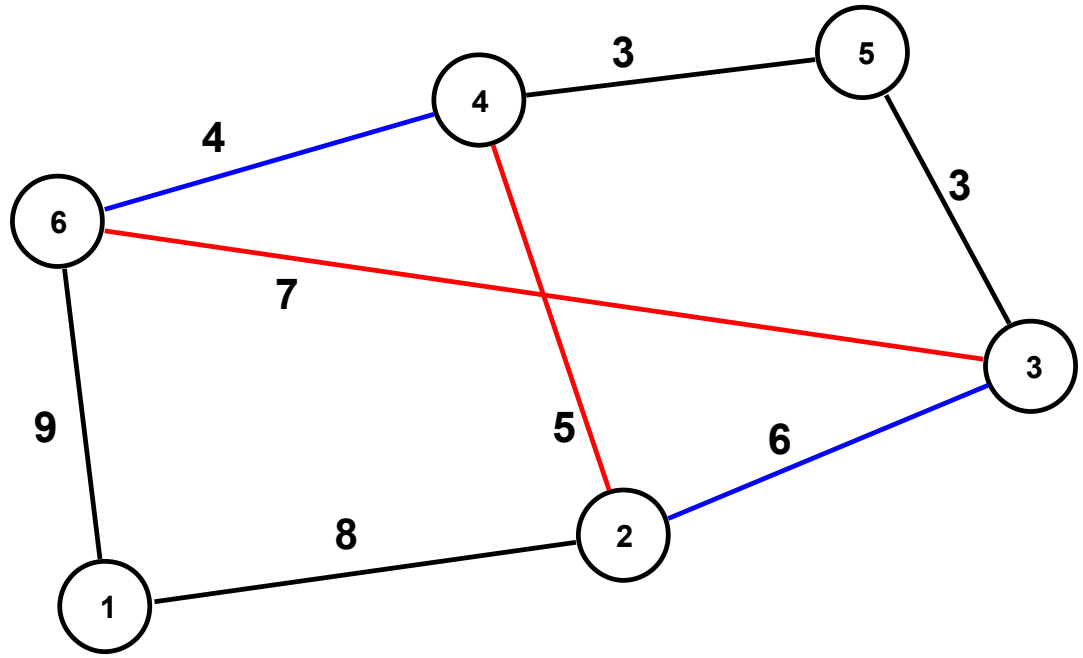
- r = 2 (2-optimal exchange)



- Time complexity $O(n^2)$ (for each iteration).
- Cost variation = $(c_{ij} + c_{kl}) - (c_{ik} + c_{jl})$
- For each exchange: time complexity O(k)

Example 2 (2-optimal exchange)

- Initial solution found by algorithm "Nearest Neighbour" with Initial Vertex = 1



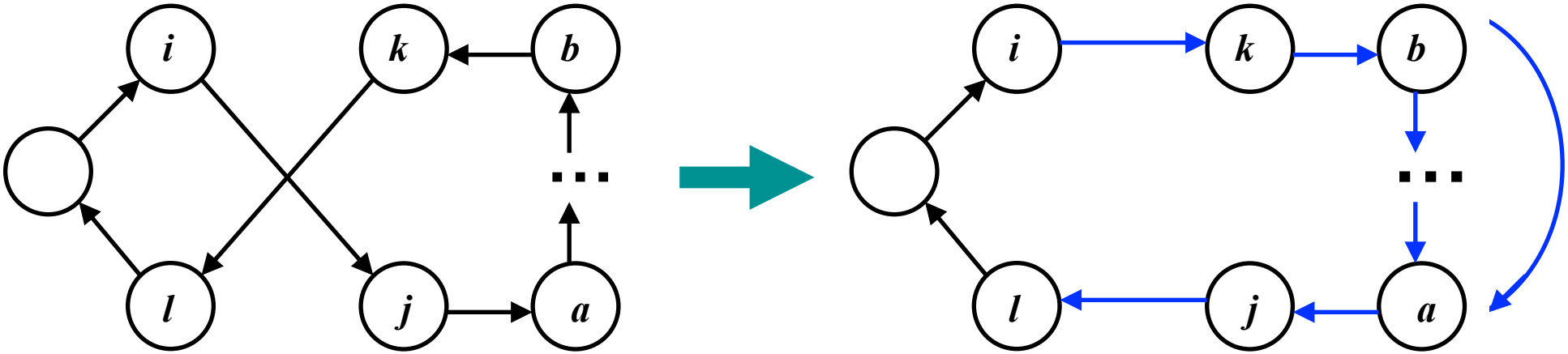
Initial cost = 35

Final cost = 33

- Exchange arcs (6, 3) and (4, 2) (cost = 12)
  with arcs (6, 4) and (3, 2) (cost = 10)

ATSP

- $r = 2$ (2-optimal exchange)



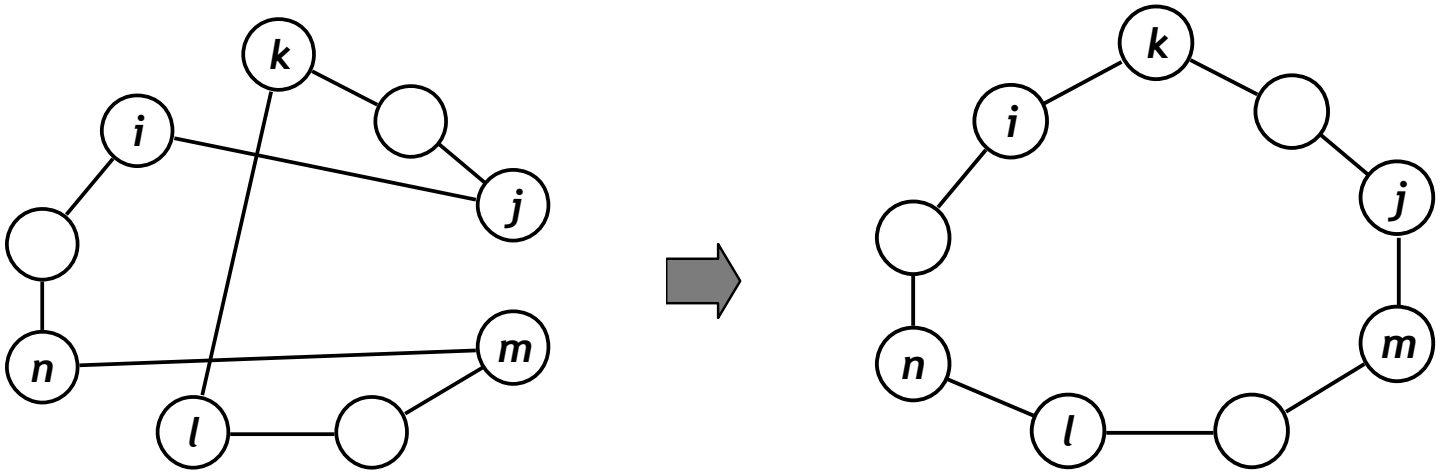
- Cost variation:

  $(c_{ij} + c_{ja} + \ldots + c_{bk} + c_{kl}) - (c_{ik} + c_{kb} + \ldots + c_{aj} + c_{jl})$

- Both directions can be considered: clockwise, anti clockwise.

STSP

- r = 3 (3-optimal exchange)



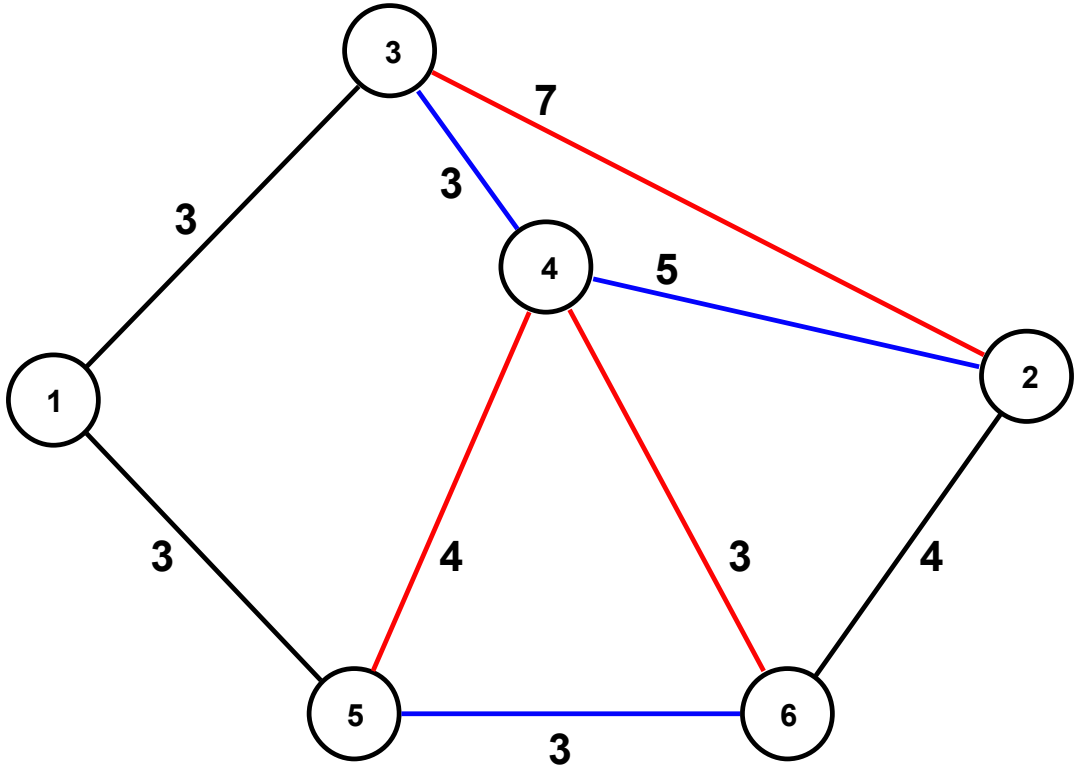
- Time complexity $O(n^3)$ (for each iteration).

- Cost variation = $(c_{ij} + c_{kl} + c_{mn}) - (c_{ik} + c_{jm} + c_{ln})$

  time complexity O(k)

Example 3 (3-optimal exchange)

- Initial solution found by algorithm “Cheapest Insertion”



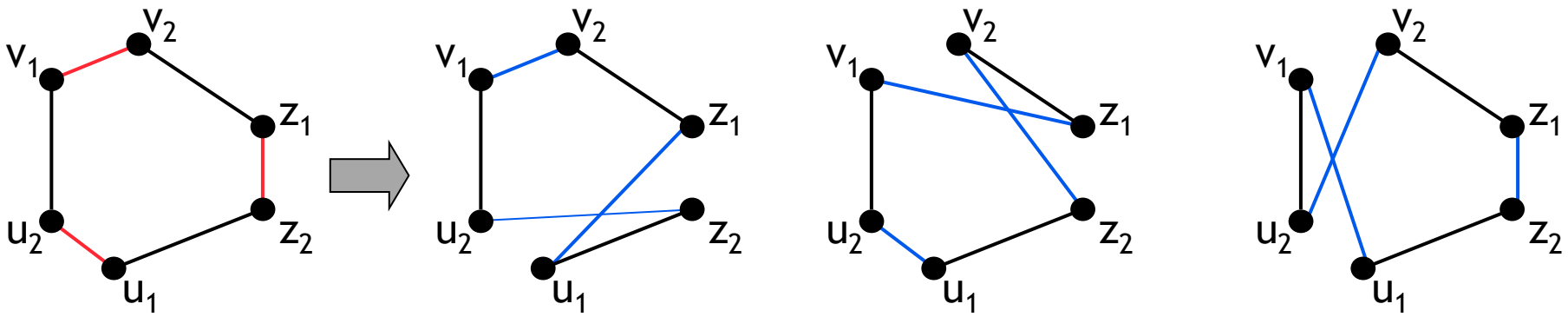
Initial cost = 24

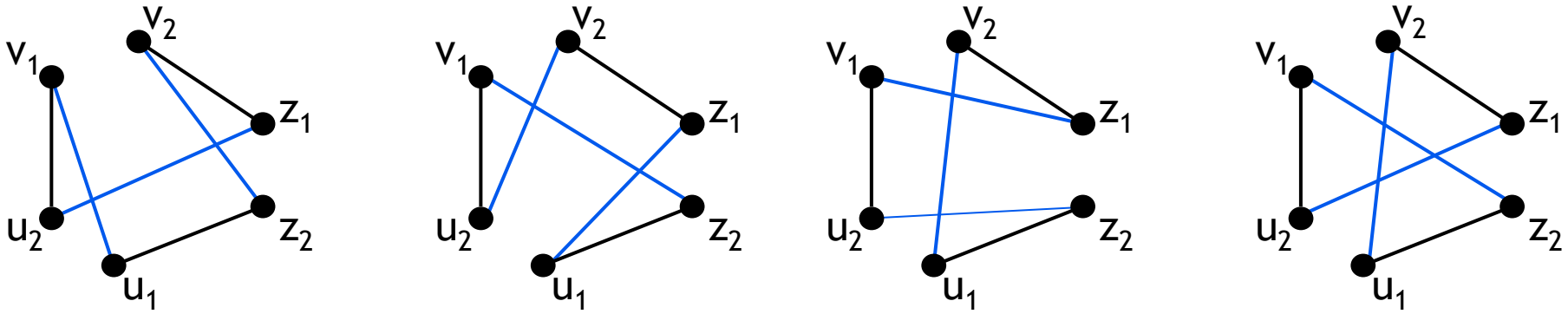
Final cost = 21

- Exchange arcs (3, 2), (6, 4) and (4, 5) (cost = 14)
  with arcs (3, 4), (4, 2) and (6, 5) (cost = 11)

STSP

- *r* = 3 (3-optimal exchange) : by removing 3 arcs from the current tour, 7 different feasible tours can be obtained:

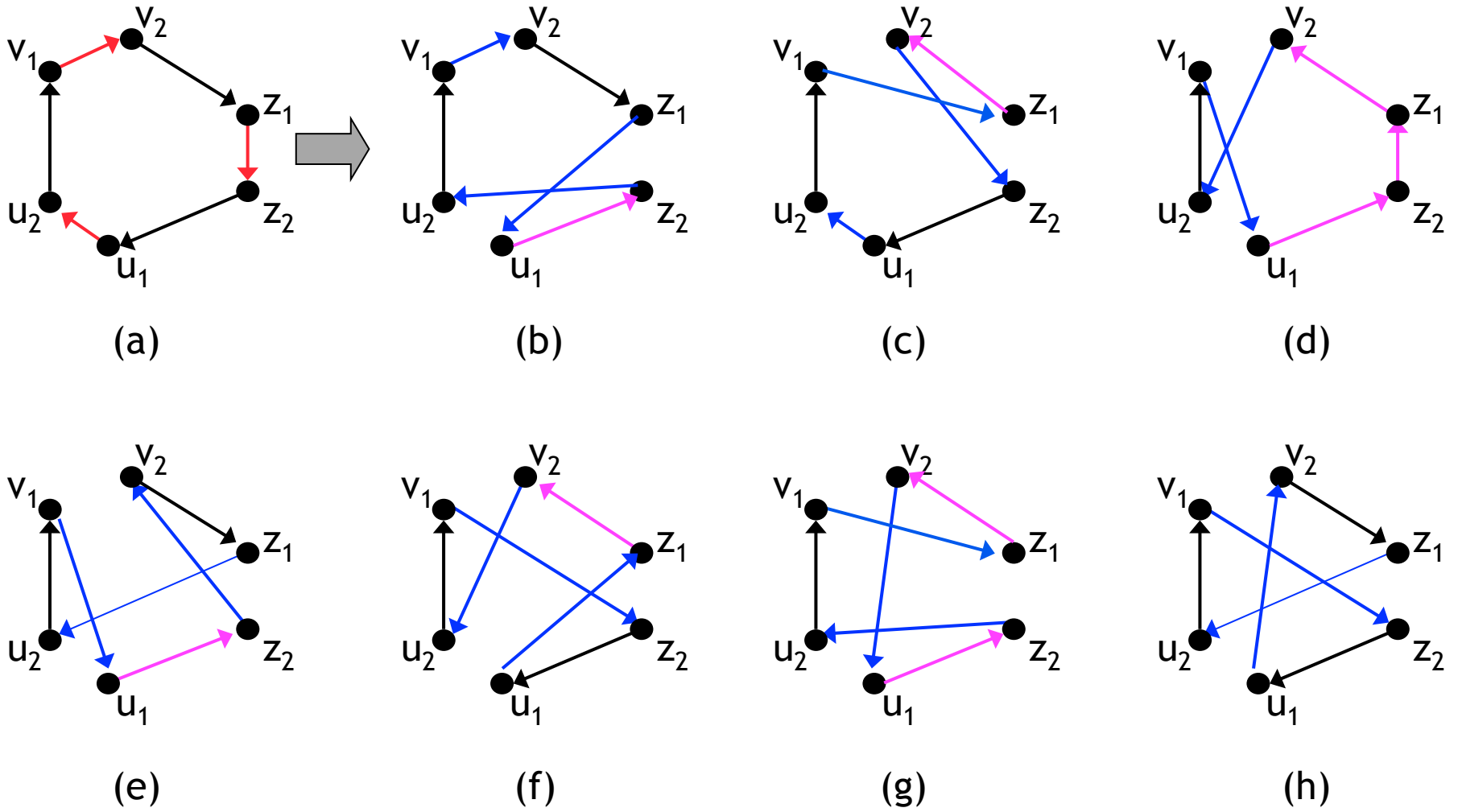




ATSP

- *r* = 3 (3-optimal exchange) : 7 different feasible tours

  (both directions can be considered)



COMPUTATIONAL RESULTS

Symmetric TSP (undirected graphs).

Two classes of randomly generated instances (n is given):

1) Random Euclidean instances

- For each vertex i (i = 1, …, n):
  generate a random point (x_i, y_i) in a square 100 x 100:
  - x_i := uniform random value in (0, 100),
  - y_i := uniform random value in (0, 100).
    - For each vertex pair (i, j) (i = 1, …, n; j = 1, …, n):

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

(symmetric cost matrix, the triangle inequality holds)

COMPUTATIONAL RESULTS FOR STSP

2) Random Distance instances

- For each vertex pair (i, j) (i = 1, …, n-1; j = i+1, …, n)
  - c_{ij} := uniform random number in (1, 100);
  - $c_{ji} := c_{ij}$
  - (symmetric cost matrix, the triangle inequality does not hold)

- n = 100, 1000, 10000, 100000.
- 10 instances for each class and for each value of n.
- Percentage "gap" between the solution cost found by the heuristic algorithm and the Lower Bound proposed by Held-Karp 1970 (1-SST Relaxation with subgradient optimization procedure).
- CPU times in seconds of a 150 MHz SGI Challenge.

STSP - RANDOM EUCLIDEAN INSTANCES

Algorithm	$n = 10^2$	$n = 10^3$	$n = 10^4$	$n = 10^5$
N. Neighbour	25.6 (0.00)	26.0 (0.03)	24.3 (0.3)	23.6 (6)
Multi-Path	19.5 (0.00)	17.0 (0.08)	16.6 (1.1)	14.9 (23)
Savings	9.2 (0.00)	11.3 (0.11)	11.9 (1.4)	12.1 (31)
2-Opt	4.5 (0.03)	4.8 (1.10)	5.0 (4.9)	4.9 (131)
3-Opt	2.5 (0.06)	3.1 (2.14)	3.0 (7.6)	3.0 (243)

- % gap (CPU times in seconds)
- procedures 2-opt and 3-opt with initial solution found by the Randomized Multi-Path algorithm

STSP – RANDOM DISTANCE INSTANCES

Algorithm	$n = 10^2$	$n = 10^3$	$n = 10^4$
N. Neighbour	130 (0.01)	240 (0.69)	360 (73)
Multi-Path	100 (0.02)	170 (0.98)	250 (107)
Savings	270 (0.03)	980 (2.23)	3200 (236)
2-Opt	34 (0.04)	70 (1.71)	125 (215)
3-Opt	10 (0.07)	33 (2.52)	63 (334)

- % gap (CPU times in seconds)
- procedures 2-opt and 3-opt with initial solution found by the Randomized Multi-Path algorithm

HEURISTIC ALGORITHMS FOR THE ATSP

(in addition to the modifications of the previously described algorithms)

ALGORITHM "PATCH" (Karp-Steele, 1985)

1. Solve the Assignment Problem (AP) corresponding to the cost matrix (c_{ij}).

2. If the current AP solution is a tour then STOP.

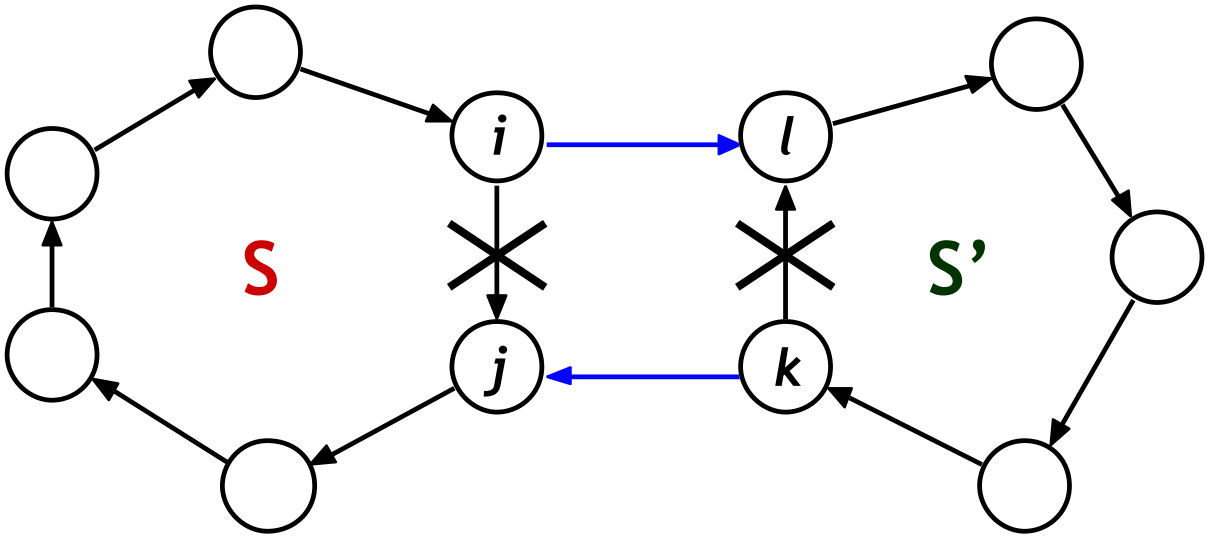
3. Consider the subtour S having the maximum number of vertices.

   "Expand" S by combining ("patching") it, through a 2 arc exchange, with a different subtour S' so as to minimize the variation of the global cost of the two subtours S and S'.

   Return to STEP 2.

Example:

- Given subtour S and a different subtour S' (for all subtours S')

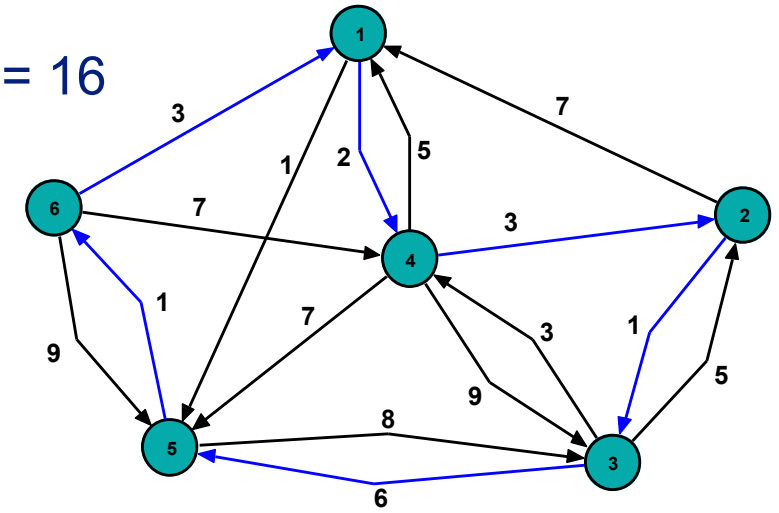


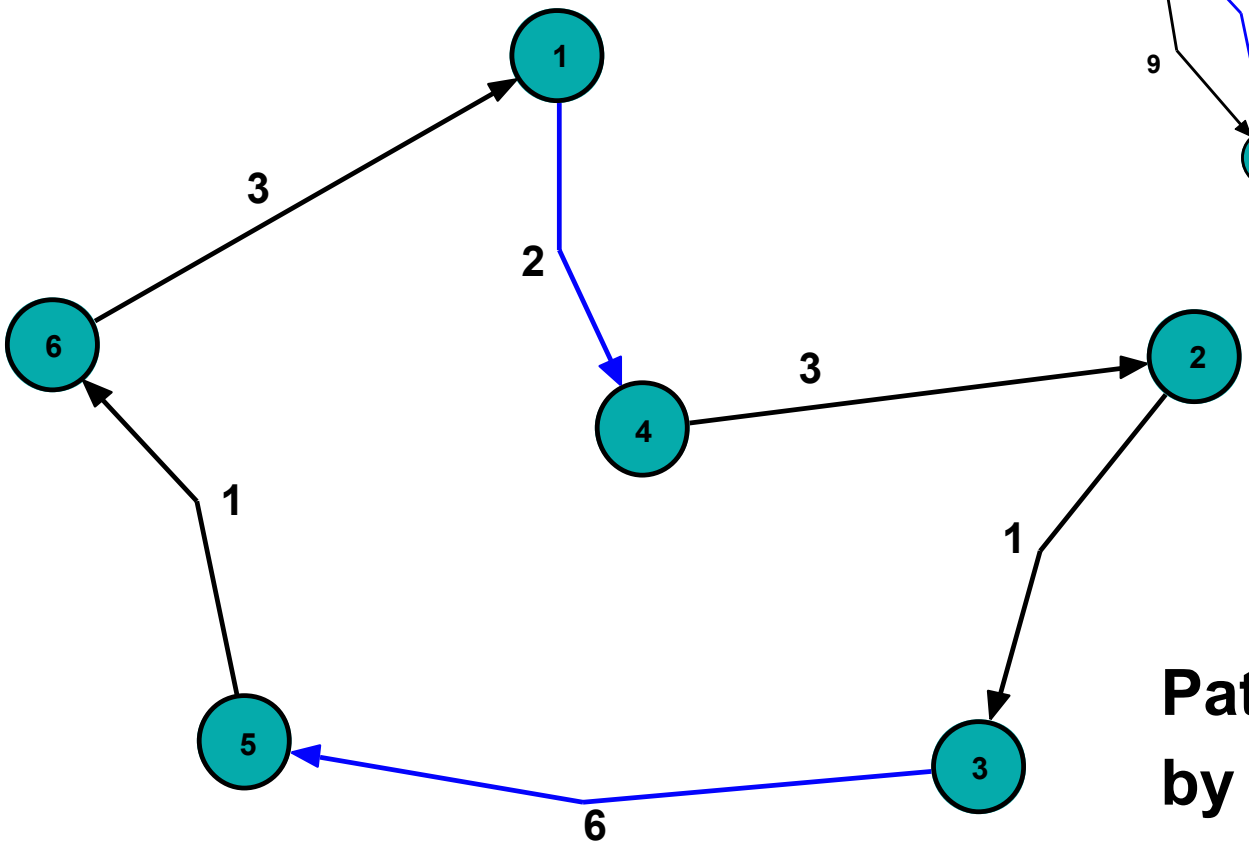
exchange arcs (i, j) and (k, l) with arcs (i, l) and (k, j)

so as to minimize: $(c_{il} + c_{kj}) - (c_{ij} + c_{kl})$ with respect to all the possible 2 arc exchanges between subtours S and S'

Example A

Opt = 16





Final cost = 16 (optimal solution)

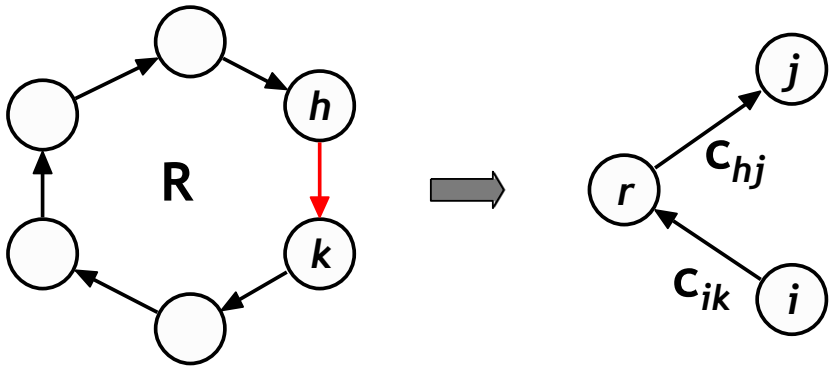
Patch the two subtours by exchanging arcs (1, 5) and (3, 4) with arcs (1, 4) and (3, 5)

ALGORITHM "CONTRACT OR PATCH"
(Glover-Gutin-Yeo-Zverovich, 2001)

- **Variant of Algorithm "Patch"**
- After STEP 2 ("if the current AP solution is a tour then STOP") add the steps:

2a. if the current solution contains at least a subtour R with less than *t* vertices then:

- remove the maximum cost arc (*h*, *k*) of R and "contract" the resulting "path" (from *k* to *h*) into a single "supervertex" *r*.



- solve the AP corresponding to the current graph, and return to STEP 2a.

2b. "re-expand", in a recursive way, the supervertices so as to obtain an AP solution corresponding to the original graph.

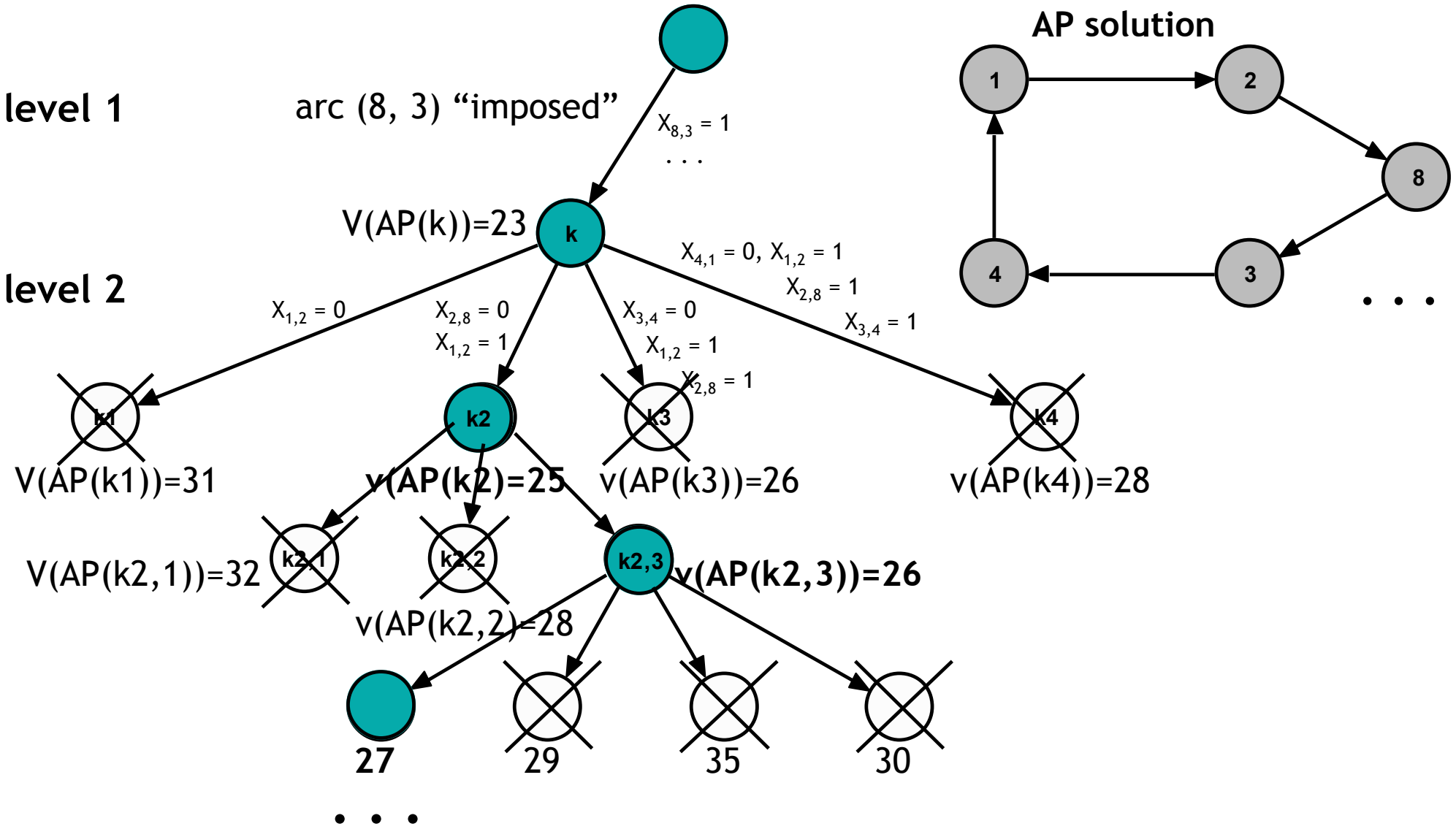
ALGORITHM "TRUNCATED BRANCH-AND-BOUND"

- Modification of the branch-and-bound algorithm based on the "AP relaxation" and the "subtour elimination branching scheme".

- At each level of the branching tree:

  consider only the "descendent node" corresponding to the minimum value of the associated Lower Bound (cost of the corresponding AP relaxation), by removing all the other descendent nodes.

ALGORITHM "TRUNCATED BRANCH-AND-BOUND" (2)



COMPUTATIONAL RESULTS FOR ATSP

- 9 classes of randomly generated instances.
- 4 different values of n: 100, 316, 1000, 3162
- 10 instances for n = 100, 316 (3 for n = 1000, 1 for n = 3162)
- HK = Held-Karp lower bound corresponding to the 1-Shortest Spanning Arborescence Relaxation with Subgradient Optimization procedure.
- 3-OPT with starting solution found by the Nearest Neighbour Algorithm
- CPU times in seconds of a 150 MHz SGI Challenge

MULTI - PATH

	Percent above HK				Time in Seconds			
Class	100	316	1000	3162	100	316	1000	3162
tmat	31.23	29.04	26.53	26.25	.03	.26	1.7	20
amat	243.09	362.86	418.56	695.29	.04	.27	1.9	21
shop	49.34	56.07	61.55	66.29	.03	.26	2.1	40
disk	188.82	307.14	625.76	1171.62	.03	.28	2.7	23
super	6.03	5.40	5.16	5.79	.03	.22	1.5	18
crane	41.86	44.09	39.70	41.60	.03	.27	1.9	21
coin	48.73	46.76	42.33	35.94	.04	.24	1.7	20
stilt	106.25	143.89	178.34	215.84	.04	.28	1.9	23
rtilt	350.12	705.56	1290.63	2350.38	.03	.28	2.0	23

NEAREST NEIGHBOUR

	Percent above HK				Time in Seconds			
Class	100	316	1000	3162	100	316	1000	3162
tmat	38.20	37.10	37.55	36.66	.03	.24	1.7	20
amat	195.23	253.97	318.79	384.90	.03	.26	1.9	21
shop	16.97	14.65	13.29	11.87	.03	.23	2.5	20
disk	96.24	102.54	115.51	161.99	.04	.27	1.9	23
super	8.57	8.98	9.75	10.62	.03	.21	1.5	18
crane	40.72	41.66	43.88	43.18	.03	.26	1.9	21
coin	26.08	26.71	26.80	25.60	.03	.23	1.7	20
stilt	30.31	30.56	27.62	24.79	.03	.30	1.9	22
rtilt	28.47	28.28	27.52	24.60	.04	.26	1.9	22

3 - OPT

	Percent above HK				Time in Seconds			
Class	100	316	1000	3162	100	316	1000	3162
tmat	6.44	9.59	12.66	16.20	.19	1.71	5.5	20
amat	39.23	58.57	83.77	112.08	.19	1.75	5.8	21
shop	3.02	7.25	10.22	10.88	.23	1.78	5.6	21
disk	12.11	16.96	20.85	25.64	.19	1.82	6.1	23
super	3.12	4.30	5.90	7.94	.15	1.43	4.8	18
crane	9.48	9.41	10.65	10.64	.19	1.76	7.3	22
coin	8.06	9.39	9.86	9.92	.18	1.62	5.3	20
stilt	11.39	12.65	12.62	12.27	.19	1.80	8.2	22
rtilt	10.04	13.09	18.00	19.83	.19	2.05	6.6	23

PATCH

	Percent above HK				Time in Seconds			
Class	100	316	1000	3162	100	316	1000	3162
tmat	.84	.64	.17	.00	.03	.22	1.8	29
amat	10.95	6.50	2.66	1.88	.03	.22	1.9	18
shop	1.15	.59	.39	.24	.04	.48	8.4	260
disk	9.40	2.35	.88	.30	.03	.26	2.9	75
super	1.86	2.84	3.99	6.22	.02	.19	1.7	29
crane	9.40	10.18	9.45	8.24	.03	.21	1.5	23
coin	16.48	16.97	17.45	18.20	.02	.18	1.4	17
stilt	23.33	22.79	23.18	24.41	.03	.24	2.2	29
rtilt	17.03	18.91	18.38	19.39	.03	.28	2.9	54

CONTRACT OR PATCH

	Percent above HK				Time in Seconds			
Class	100	316	1000	3162	100	316	1000	3162
tmat	.57	.36	.16	.00	.01	.12	.7	15
amat	9.31	3.15	2.66	1.01	.01	.15	.6	26
shop	.68	.36	.19	.10	.08	1.41	29.1	1152
disk	6.00	1.13	.51	.15	.03	.31	8.7	297
super	1.01	1.20	1.22	2.06	.03	.24	4.6	243
crane	10.32	9.08	7.28	6.21	.04	.44	3.5	53
coin	16.44	17.68	16.23	16.06	.02	.10	1.2	22
stilt	22.48	23.31	22.80	22.90	.07	.94	8.1	105
rtilt	19.62	22.86	20.95	20.37	.05	.33	5.6	117

TRUNCATED BRANCH-AND-BOUND

	Percent above HK				Time in Seconds			
Class	100	316	1000	3162	100	316	1000	3162
tmat	.06	.01	.00	.00	.03	.27	2.5	30
amat	.97	.16	.04	.04	.04	.47	7.6	296
shop	.20	.08	.03	.01	.06	1.02	19.6	460
disk	1.51	.27	.02	.01	.05	.56	6.4	105
super	.27	.17	.21	.43	.04	.61	20.4	995
crane	4.36	4.29	4.05	4.10	.07	1.96	66.7	3176
coin	8.20	11.03	11.14	11.42	.10	3.82	168.4	9610
stilt	10.75	13.99	12.66	12.86	.11	4.11	163.7	4184
rtilt	9.82	12.20	11.81	11.45	.13	4.37	178.0	9594