EXACT ALGORITHMS FOR THE ATSP

- **Branch-and-Bound Algorithms**:
- Little-Murty-Sweeney-Karel (*Operations Research*, 1963);
- Bellmore-Malone (Operations Research, 1971);
- Garfinkel (Operations Research, 1973);
- Smith-Srinivasan-Thompson (Annals Discrete Math., 1977);
- Carpaneto-T. (Management Science, 1980);
- Balas-Christofides (Mathematical Programming, 1981);
- Pekny-Miller (Oper. Res. Letters, 1989);
- Fischetti-T. (Mathematical Programming, 1992);
- Pekny-Miller (Mathematical Programming, 1992);
- Carpaneto-Dell'Amico-T. (ACM Trans. Math. Software, 1995);
- ...

EXACT ALGORITHMS FOR THE ATSP

- Branch-and-Cut Algorithms:
- Fischetti-T. (Management Science, 1997);
- Fischetti-Lodi-T. (LNCS Springer, 2003);
- …

- Surveys:
- Balas-T. (The Traveling Salesman Problem, Lawler et al. eds, Wiley, 1985);
- Fischetti-Lodi-T.(The TSP and its Variations, Gutin-Punnen eds, Kluwer, 2002);
- Roberti-T. (EURO Journal Transp. and Logistics, 2011);
- …

CLASSICAL LOWER BOUNDS

ASSIGNMENT PROBLEM (AP) RELAXATION ($O(n^3)$ time)

$$\min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}$$

s.t.

$$\sum_{j \in V} x_{ij} = 1 \qquad i \in V$$

$$\sum_{i \in V} x_{ij} = 1 \qquad j \in V$$

$$\cancel{\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1 \qquad S \subset V,\ |S| \geq 2}$$

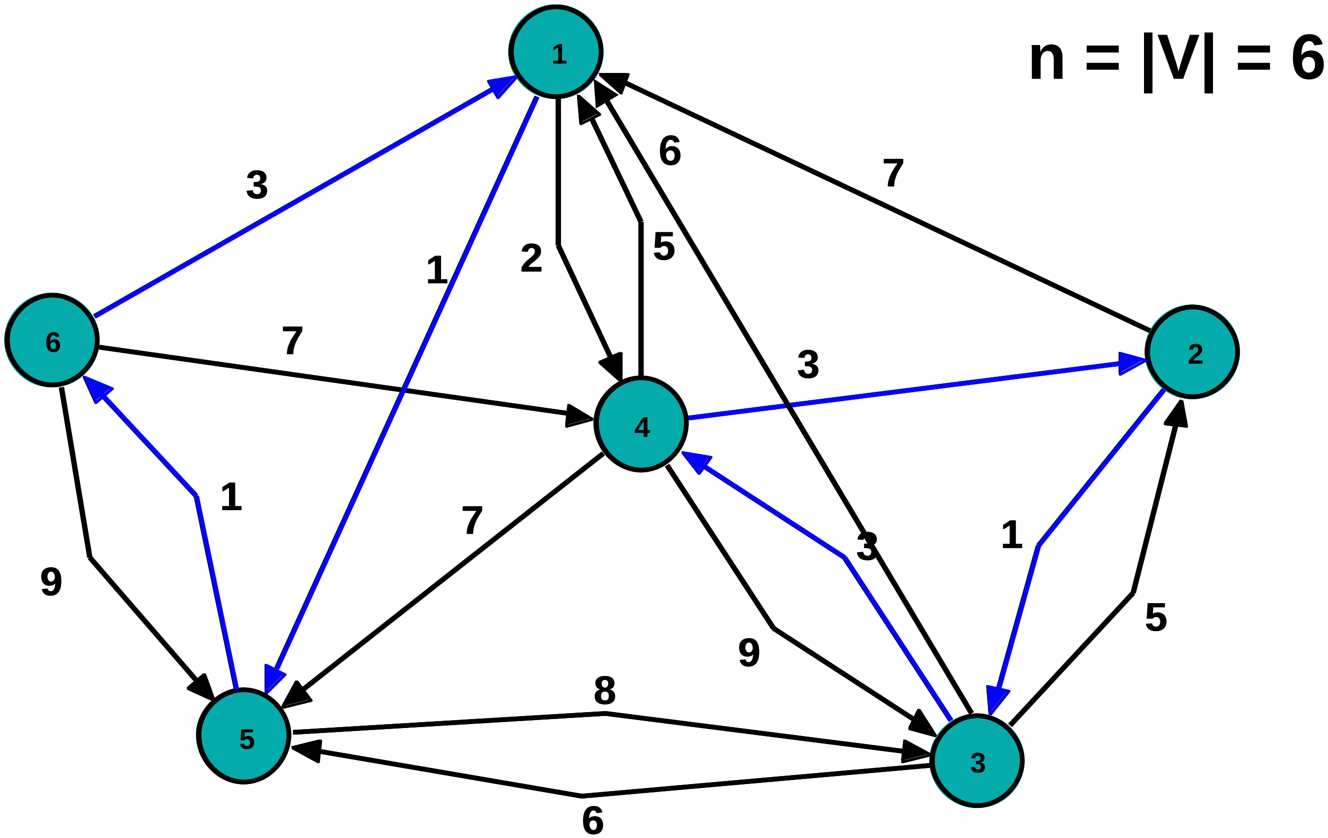
Connectivity Constraints
(Subtour Elimination Constraints)

$$\cancel{x_{ij} \in \{0, 1\} \qquad i \in V, j \in V}$$

$$\boxed{x_{ij} \geq 0 \quad \text{(LP Relaxation)} \quad i \in V, j \in V}$$

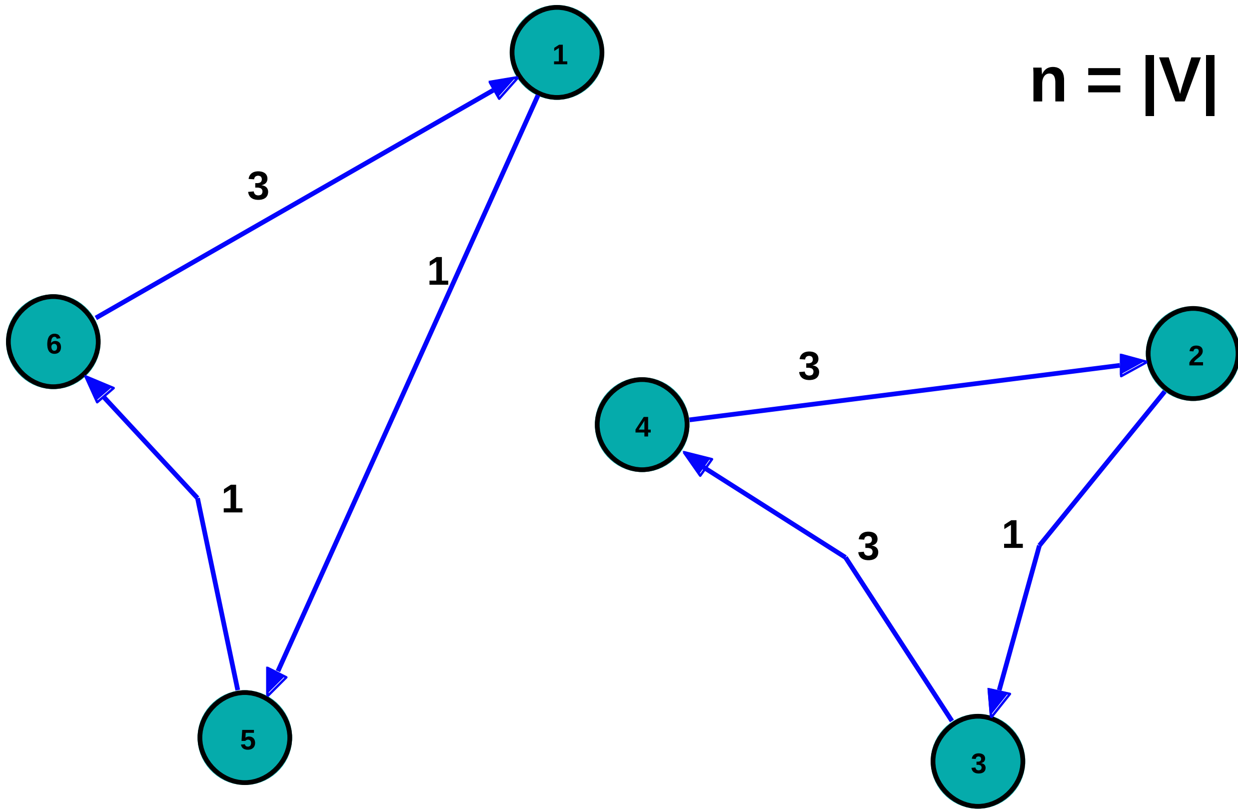
* The AP solution is given by a family of "*subtours*" (partial circuits)

Example: AP relaxation of ATSP



Optimal assignment

Example: AP relaxation of ATSP



n = |V| = 6

Optimal assignment

Lower bound = v(AP) = (1 + 1 + 3) + (3 + 1 + 3) = 12

Optimal solution Cost = 16

r - SHORTEST SPANNING ARBORESCENCE RELAXATION

$$\min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}$$

s.t.

$$\cancel{\sum_{j \in V} x_{ij} = 1 \qquad i \in V}$$ out-degree constraints

$$\sum_{i \in V} x_{ij} = 1 \qquad j \in V$$ in-degree constraints

$$\sum_{i \in S} \sum_{j \in V \setminus S} x_{ij} \geq 1 \qquad S \subset V, \; r \in S \text{ (for a fixed } r \in V)$$

$$x_{ij} \in \{0, 1\} \qquad i \in V, j \in V$$

r - SHORTEST SPANNING ARBORESCENCE RELAXATION

Partition the Arc Set A into two subsets: $(i,j) \quad i \in V, \; j \in V \setminus r$

$(i,r) \quad i \in V$

$$\min \sum_{i \in V} \sum_{j \in V \setminus r} c_{ij} x_{ij} + \sum_{i \in V} c_{ir} x_{ir}$$

s.t.

$$\sum_{i \in V} x_{ij} = 1 \qquad j \in V \setminus r$$

in-degree constraints

$$\sum_{i \in V} x_{ir} = 1$$

$$\sum_{i \in S} \sum_{j \in V \setminus S} x_{ij} \geq 1 \qquad S \subset V, \; r \in S \qquad (j \in V \setminus r)$$

$$x_{ij} \in \{0, 1\} \qquad i \in V, j \in V \setminus r$$

$$x_{ir} \in \{0, 1\} \qquad i \in V$$

The Relaxed Problem can be split into two independent problems

r - SHORTEST SPANNING ARBORESCENCE RELAXATION

Problem concerning the arc set: $(i,j)\quad i \in V,\ j \in V \setminus r$

$$\min \sum_{i \in V} \sum_{j \in V \setminus r} c_{ij}\, x_{ij}$$

s.t.

$$\sum_{i \in V} x_{ij} = 1 \qquad j \in V \setminus r \qquad \text{in-degree constraints}$$

$$\sum_{i \in S} \sum_{j \in V \setminus S} x_{ij} \geq 1 \qquad S \subset V,\ r \in S \qquad \text{connectivity constraints from } r$$

$$\cancel{x_{ij} \in \{0, 1\} \qquad i \in V, j \in V \setminus r}$$

$$x_{ij} \geq 0 \ \text{(LP Relaxation)} \qquad i \in V, j \in V \setminus r$$

The problem calls for an r-SSA: ($O(n^2)$ time)

Shortest (Minimum Cost) Spanning Arborescence rooted at vertex r (involving n - 1 arcs)

r - SHORTEST SPANNING ARBORESCENCE RELAXATION

Problem concerning the arc set: (i,r) $i \in V$

$$\min \sum_{i \in V} c_{ir}\, x_{ir}$$

s.t.

$$\sum_{i \in V} x_{ir} = 1$$ in-degree constraint for vertex r

$$\cancel{x_{ir} \in \{0, 1\} \qquad i \in V}$$

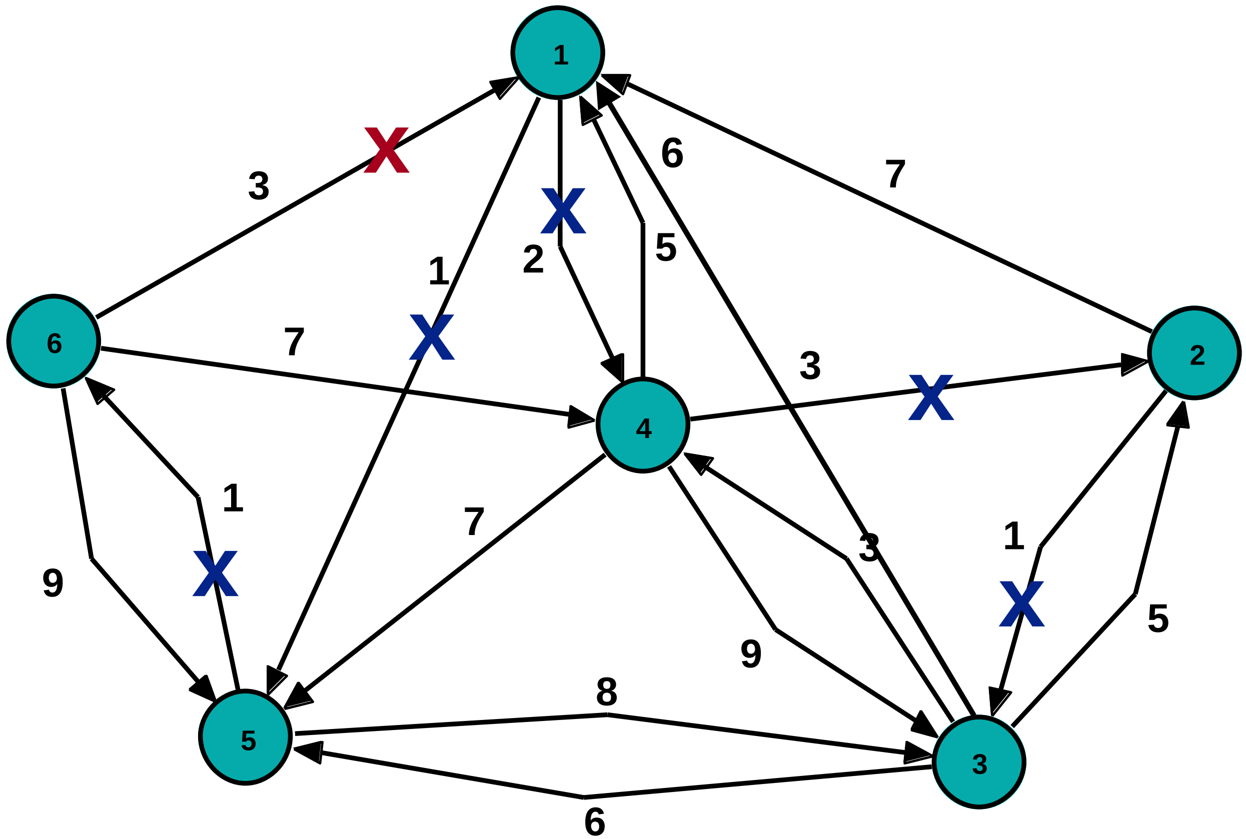
$$x_{ir} \geq 0 \quad \text{(LP Relaxation)} \qquad i \in V$$

The problem calls for an r-MCA:

Minimum Cost Arc entering vertex r (O(n) time)

Lower Bound = v(r-SSAR) = v(r-SSA) + v(r-MCA)

Example: r-SSA relaxation of ATSP (r = 1)



Optimal 1-SSA, v(1-SSA) = 1 + 1 + 2 + 3 + 1 = 8

Optimal 1-MCA, v(1-MCA) = 3,

Lower bound = 8 + 3 = 11, Optimal solution Cost = 16

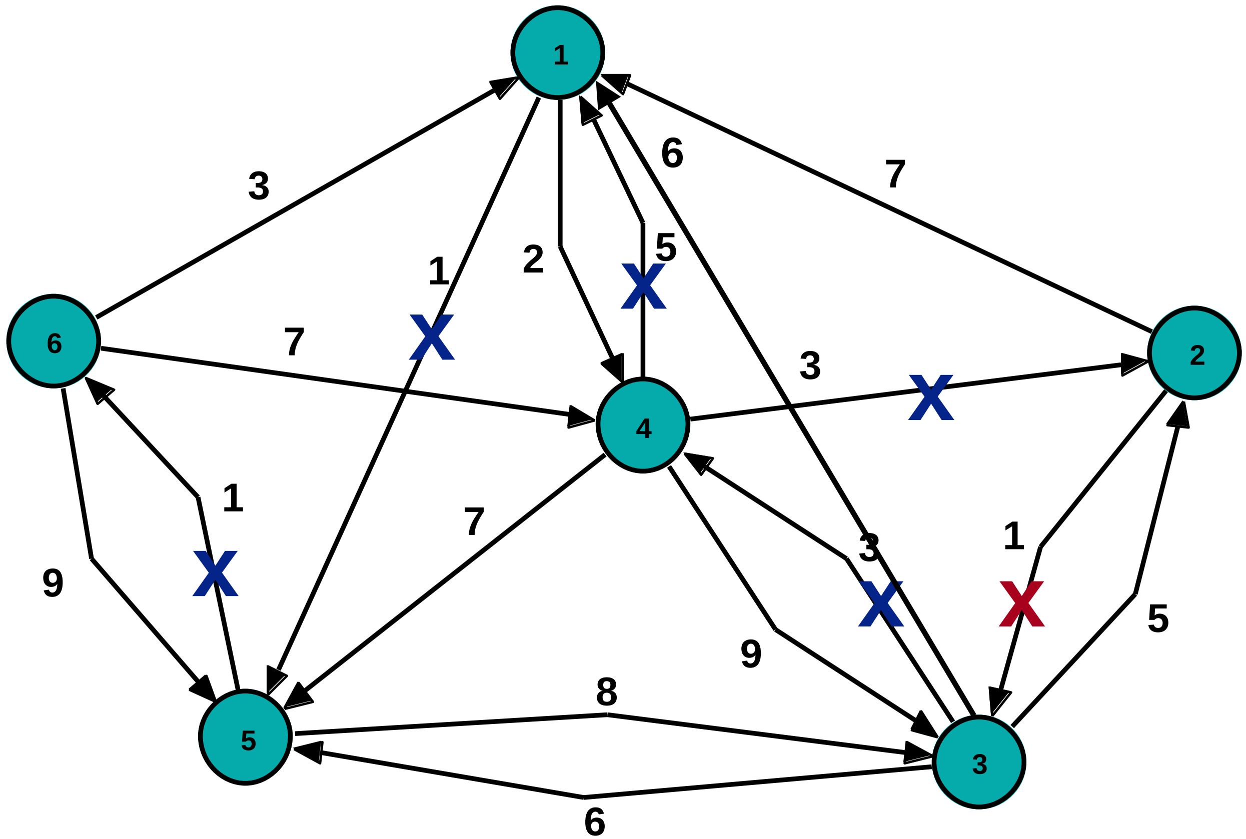
r - SHORTEST SPANNING ARBORESCENCE RELAXATION

- Different choices of the root vertex r generally produce different values of the corresponding lower bounds.

- The r - Shortest Spanning Anti - Arborescence Relaxation can be used as well (connectivity toward r).

- The lower bounds can be strengthened through Lagrangian Relaxation of the out-degree constraints, and Subgradient Optimization Procedures (to determine “good” Lagrangian multipliers).

Example: r-SSA relaxation of ATSP ($r = 3$)

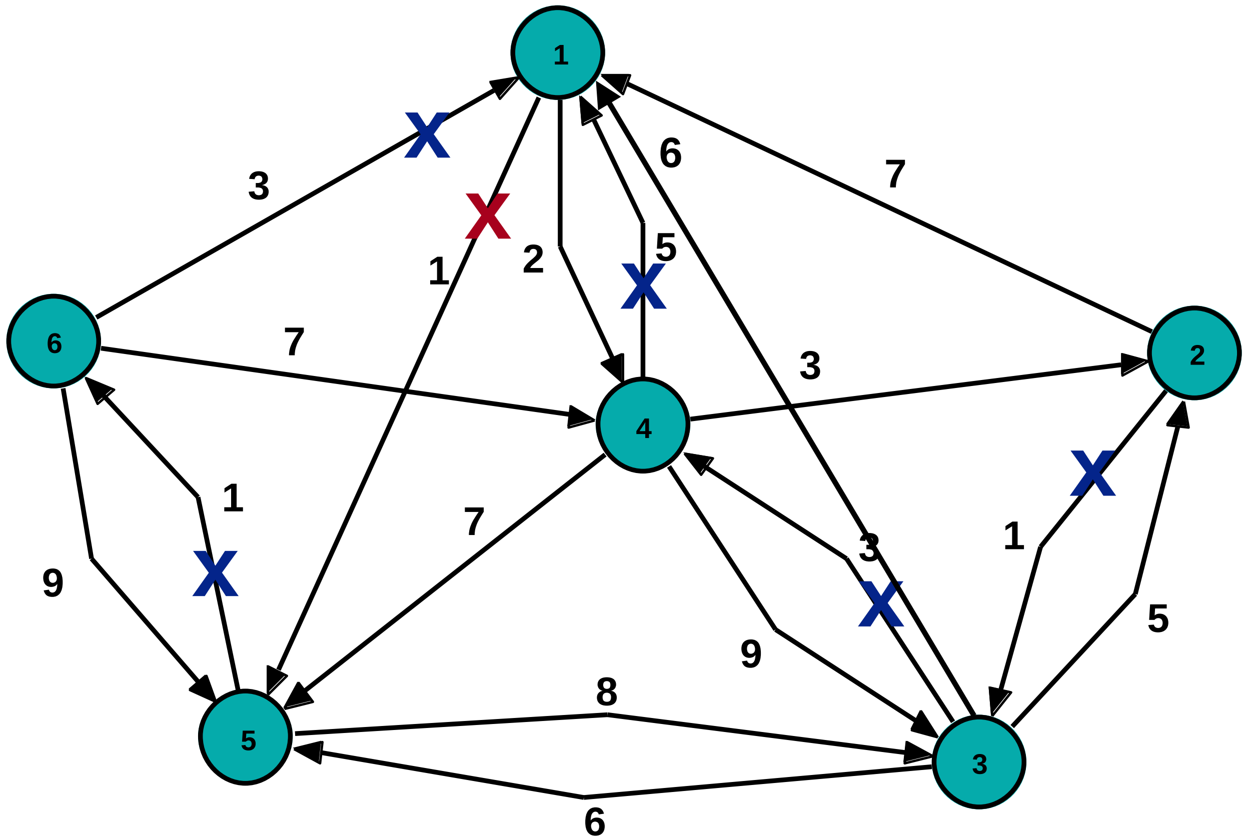


Optimal 3-SSA, v(3-SSA) = 3 + 3 + 5 + 1 + 1 = 13

Optimal 3-MCA, v(3-MCA) = 1,

Lower bound = 13 + 1 = 14, Optimal solution Cost = 16

Example: r-Anti-SSA relaxation of ATSP (r = 1)



Optimal 1-Anti-SSA, v(1-Anti-SSA) = 3 + 1 + 5 + 3 + 1 = 13

Optimal 1-Anti-MCA, v(1-Anti-MCA) = 1,

Lower bound = 13 + 1 = 14, Optimal solution Cost = 16

LAGRANGIAN RELAXATION OF THE r - SHORTEST SPANNING ARBORESCENCE RELAXATION

(u_i: "Lagrangian multiplier" associated with the i-th out-degree constraint)

$$\min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}$$

s.t.

u_i $\quad \boxed{\sum_{j \in V} x_{ij} = 1 \qquad i \in V}$ $\quad$ out-degree constraints

$$\sum_{i \in V} x_{ij} = 1 \qquad j \in V$$ in-degree constraints

$$\sum_{i \in S} \sum_{j \in V \setminus S} x_{ij} \geq 1 \qquad S \subset V,\ r \in S \text{ (for a fixed } r \in V)$$

$$x_{ij} \in \{0, 1\} \qquad i \in V, j \in V$$

LAGRANGIAN RELAXATION OF THE r - SHORTEST SPANNING ARBORESCENCE RELAXATION

(u_i: "Lagrangian multiplier" associated with the i-th out-degree constraint)

$$z(u) = \min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} + \sum_{i \in V} u_i \left(\sum_{j \in V} x_{ij} - 1 \right)$$

s.t.

u_i $\quad \cancel{\boxed{\sum_{j \in V} x_{ij} = 1 \qquad i \in V}}$ $\quad$ out-degree constraints

$$\sum_{i \in V} x_{ij} = 1 \qquad j \in V \qquad \text{in-degree constraints}$$

$$\sum_{i \in S} \sum_{j \in V \setminus S} x_{ij} \geq 1 \qquad S \subset V,\ r \in S \ (\text{for a fixed } r \in V)$$

$$x_{ij} \in \{0, 1\} \qquad i \in V, j \in V$$

LAGRANGIAN RELAXATION OF THE r - SHORTEST SPANNING ARBORESCENCE RELAXATION

$$z(u) = \min \sum_{i\in V}\sum_{j\in V} c_{ij}\, x_{ij} + \sum_{i\in V} u_i \left(\sum_{j\in V} x_{ij} - 1 \right)$$

$$= -\sum_{i\in V} u_i + \min \sum_{i\in V}\sum_{j\in V} \left(c_{ij} + u_i\right) x_{ij}$$

$$= -\sum_{i\in V} u_i + \min \sum_{i\in V}\sum_{j\in V} c'_{ij}\, x_{ij} \quad (\text{where } c'_{ij} = c_{ij} + u_i)$$

c'_{ij} is the Lagrangian cost of arc (i,j) (with $(i,j) \in A$)

LAGRANGIAN RELAXATION OF THE r - SHORTEST SPANNING ARBORESCENCE RELAXATION

$$z(u) = \min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} + \sum_{i \in V} u_i \left(\sum_{j \in V} x_{ij} - 1 \right)$$

$$= - \sum_{i \in V} u_i + \min \sum_{i \in V} \sum_{j \in V} (c_{ij} + u_i) x_{ij}$$

$$= - \sum_{i \in V} u_i + \min \sum_{i \in V} \sum_{j \in V} c'_{ij} x_{ij} \quad (\text{where } c'_{ij} = c_{ij} + u_i)$$

c'_{ij} is the Lagrangian cost of arc (i,j) (with $(i,j) \in A$)

Globally ($O(n^2)$ time)

LAGRANGIAN RELAXATION OF THE
r - SHORTEST SPANNING ARBORESCENCE RELAXATION

$$z(u) = \min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} + \sum_{i \in V} u_i \left(\sum_{j \in V} x_{ij} - 1 \right)$$

$$= - \sum_{i \in V} u_i + \min \sum_{i \in V} \sum_{j \in V} \left(c_{ij} + u_i\right) x_{ij}$$

$$= - \sum_{i \in V} u_i + \min \sum_{i \in V} \sum_{j \in V} c'_{ij} x_{ij} \quad (\text{where } c'_{ij} = c_{ij} + u_i)$$

c'_{ij} is the Lagrangian cost of arc (i,j) (with $(i,j) \in A$)

Globally ($O(n^2)$ time)

LAGRANGIAN DUAL PROBLEM

Find the optimal Lagrangian multiplier vector u^* such that:

$$z(u^*) = \max_{u} \{z(u)\}$$

Difficult problem (heuristic solution through Subgradient Optimization Procedures)

SUBGRADIENT OPTIMIZATION PROCEDURE

(Held-Karp, Operations Research 1971, for STSP)

Lagrangian cost: $c'_{ij} = c_{ij} + u_i \quad (i,j) \in A$

* Initially: $u_i := 0 \quad i \in V$; $LB := 0 \quad (c_{ij} \geq 0)$

At each iteration:

* Solve the r-SSA Relaxation with respect to the current Lagrangian costs c'_{ij} : (x_{ij}) is the corresponding optimal solution

* $LB := \max (LB, v(x_{ij}))$

* Subgradient vector (s_i): $s_i := \sum_{j \in V} x_{ij} - 1 \qquad i \in V$

* If $s_i = 0$ for all $i \in V$ then STOP ((x_{ij}) is the opt. sol. of ATSP)
*

$$u_i := u_i + a\, s_i \qquad i \in V$$

where $a := b\,(UB - LB) / \sum_{i \in V} s_i^2$ (with $0 < b \leq 2$) $(a > 0)$

Stopping criteria: max number of iterations, slow increase of LB,…

SUBGRADIENT OPTIMIZATION PROCEDURE:

UPDATING OF THE LAGRANGIAN MULTIPLIERS

* $c'_{ij} = c_{ij} + u_i$ (with $(i,j) \in A$)

* $u_i := u_i + a\, s_i \quad i \in V$ (with a positive)

* $s_i := \sum_{j \in V} x_{ij} - 1 \quad i \in V$

* d_i = outdegree of vertex i ($i \in V$)

  ($d_i = 0$ if $s_i = -1$; $d_i = 1$ if $s_i = 0$; $d_i \geq 2$ if $s_i \geq 1$)

* for a given vertex i, the Lagrangian cost c'_{ij} of arc (i, j) ($j = 1, \ldots, n$) is

  decreased if $d_i = 0$.
  unchanged if $d_i = 1$,
  increased if $d_i \geq 2$.

Note that: $a \sum_{i \in V} s_i = a \left(\sum_{i \in V} \sum_{j \in V} x_{ij} - n \right) = a\,(n - n) = 0$

$\sum_{i \in V} u_i = 0$ (if initially $u_i = 0 \quad i \in V$)

GREEDY ALGORITHM "NEAREST NEIGHBOR" FOR ATSP (Upper Bound computation)

1. Choose any vertex h as "initial vertex" of the current "path".
   Set $i := h$, $V' := V \setminus \{i\}$ (set of the "unvisited" vertices).

2. Determine the "unvisited" vertex k "nearest" to vertex i
   (k : $c_{ik} = \min \{c_{ij} : j \in V'\}$).

3. Insert vertex k just after vertex i in the current path ($V' := V' \setminus \{k\}$);
   set $i := k$;
   If $V' \neq \emptyset$ (at least one vertex is unvisited) return to STEP 2.

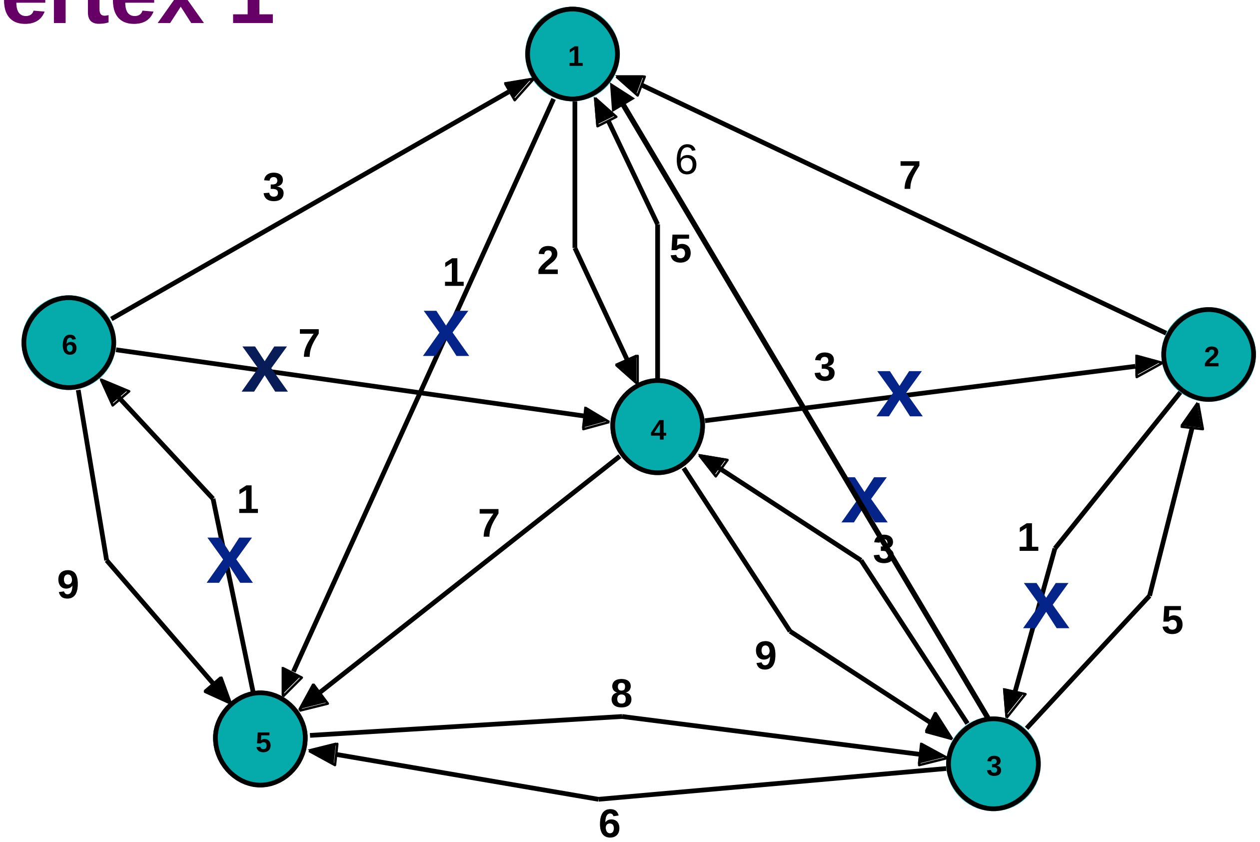
4. Complete the Hamiltonian circuit with arc (i, h);
   STOP.

- Time complexity: $O(n^2)$.

- Different choices of the "initial vertex" can lead to different solutions.

Example: Nearest Neighbor Heuristic Algorithm
(Upper Bound computation)

Initial vertex 1



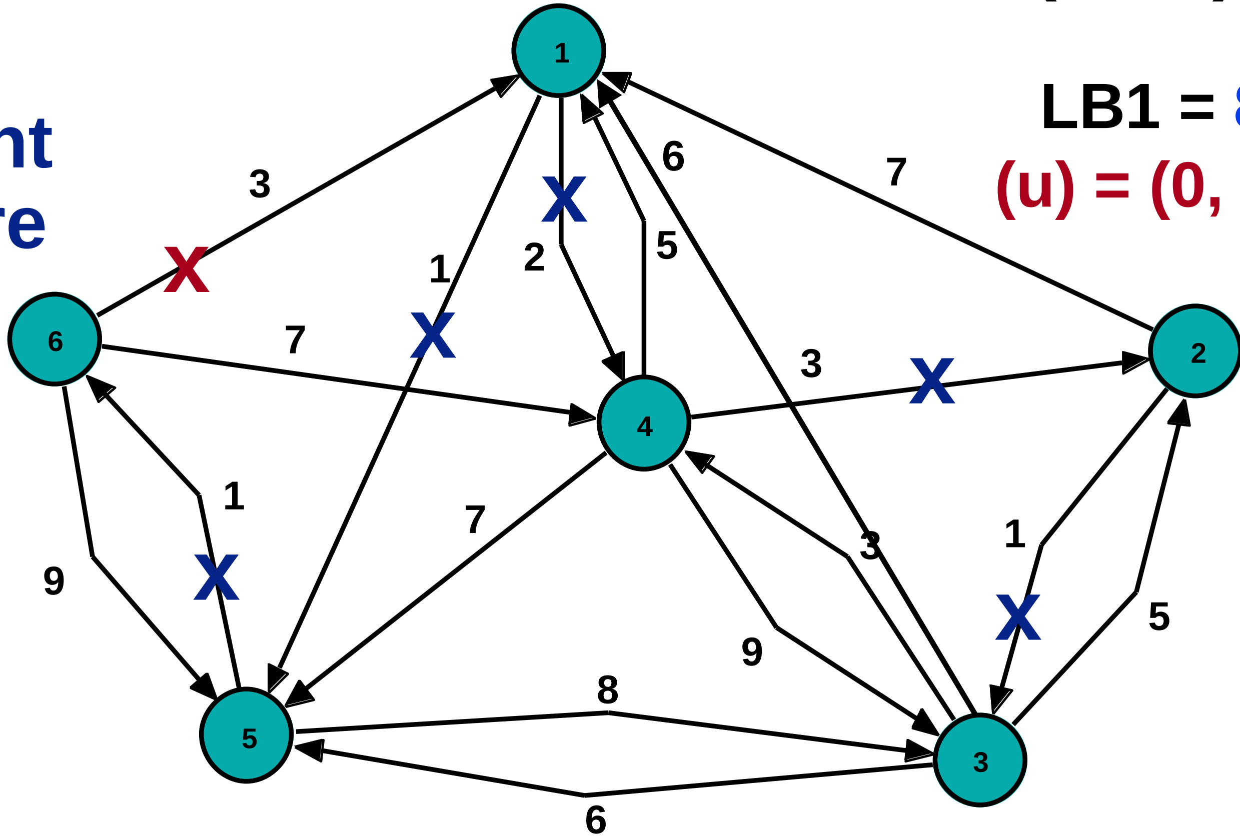
$$UB = 1 + 1 + 7 + 3 + 1 + 6 = 19$$

Example: r-SSA relaxation of ATSP (r = 1)

Subgradient Procedure

(b = 0.25,

UB = 19)



LB1 = 8 + 3 = 11

(u) = (0, 0, 0, 0, 0, 0)

(d) = (2, 1, 0, 1, 1, 1), **(s) = (1, 0, -1, 0, 0, 0)**

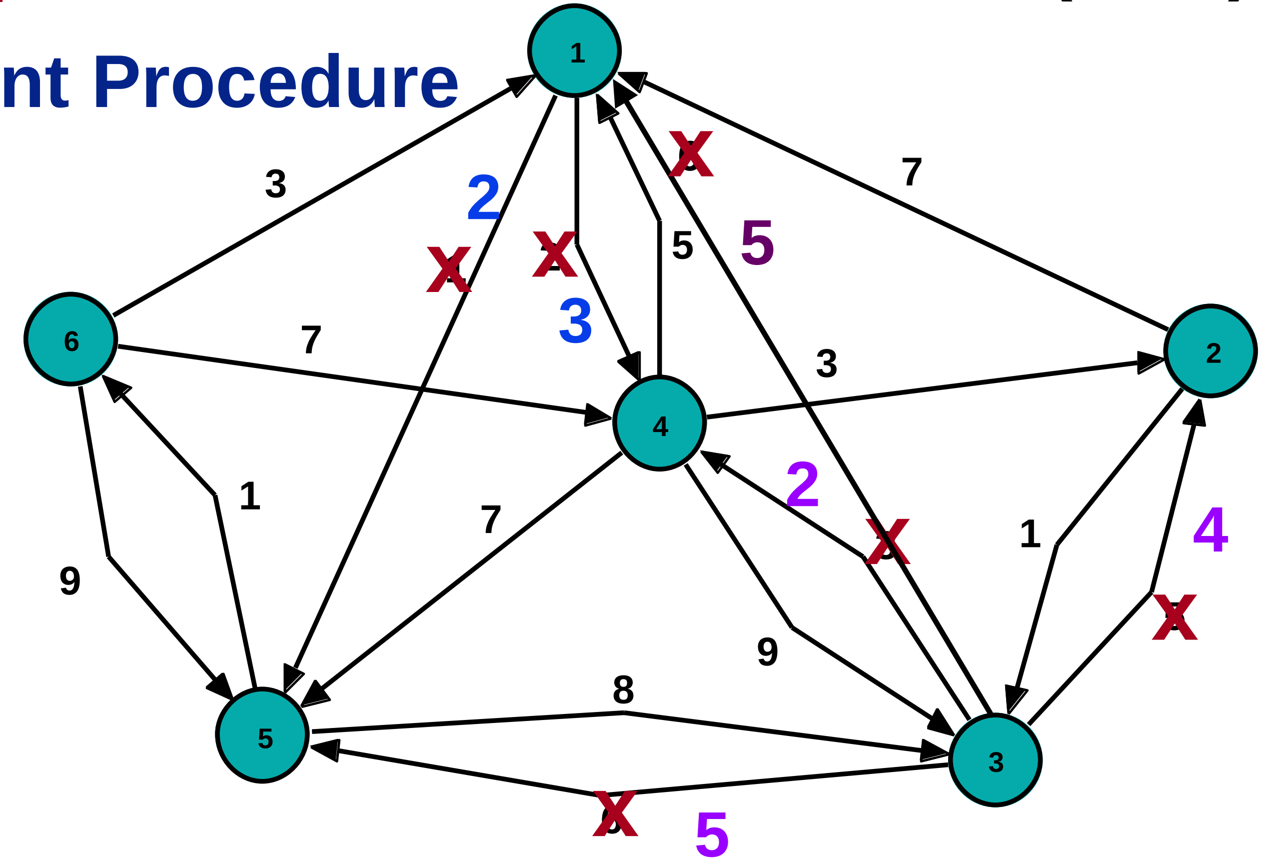
$$a := b\,(UB - LB1) / \sum_{i \in V} s_i^2 = 0.25\,(19 - 11) / 2 = 1$$

(u) = (u) + a (s) **(u) = (1, 0, -1, 0, 0, 0)**

<u>Example:</u> r-SSA relaxation of ATSP (r = 1)

Subgradient Procedure

(a = 1)



(c') = (c) + (u)

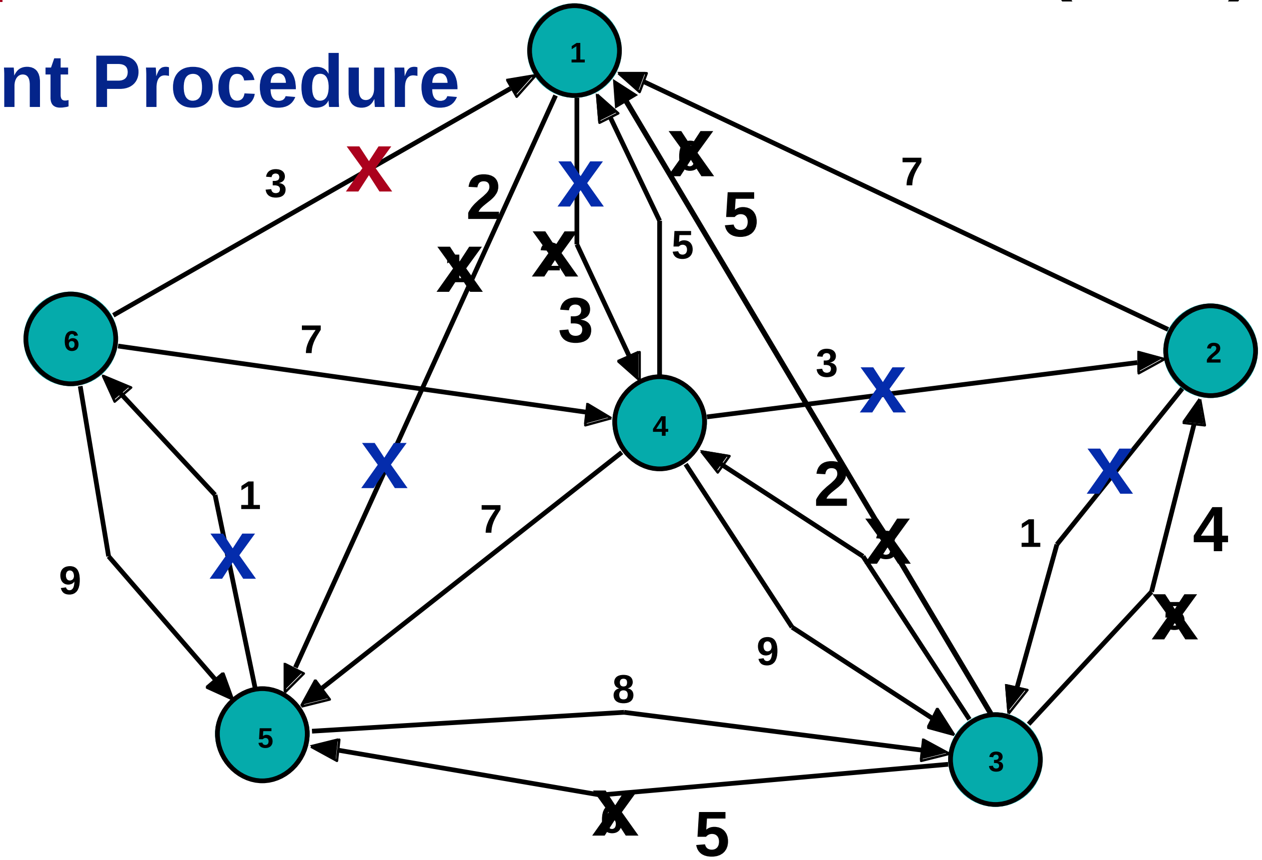
(u) = (1, 0, -1, 0, 0, 0)

Example: r-SSA relaxation of ATSP (r = 1)

Subgradient Procedure

(a = 1,

LB1 = 11)



LB2 = 3 + 3 + 1 + 2 + 1 + 3 = 13

(d) = (2, 1, 0, 1, 1, 1)

…

- The AP Relaxation can be strengthened by combining the AP substructure with the SSA substructures:

- Restricted Lagrangian Relaxation

  (Balas-Christofides, 1981):

  1) Lagrangian Relaxation of a subset of Subtour Elimination Constraints (SECs);

  2) Solve the corresponding AP problem (w.r.t. the current Lagrangian costs);

  3) Determine good Lagrangian multipliers (through a subgradient optimization procedure) and additional SECs, to be inserted into the current Lagrangian relaxation

  and iterate on Steps 2) and 3).

Additive Bounding Procedure

(Fischetti-T., Math.Progr.1992)

1) Solve the AP Relaxation through a "primal-dual" algorithm ($O(n^3)$ time)

LB := v(AP), c'_{ij} := "reduced cost" of arc (i,j);

2) Solve the r-SSA Relaxation w.r.t. costs c'_{ij} ($O(n^2)$ time):

LB := LB + v(r-SSAR), c'_{ij} := new "reduced cost" of arc (i,j);

3) Solve the r-Anti-SSA Relaxation w.r.t. costs c'_{ij} ($O(n^2)$ time):

LB := LB + v(r-Anti-SSAR), c'_{ij} := new "reduced cost" of arc (i,j);

Iterate Steps 2) and 3) with different root nodes r.

(globally $O(n^3)$ time)

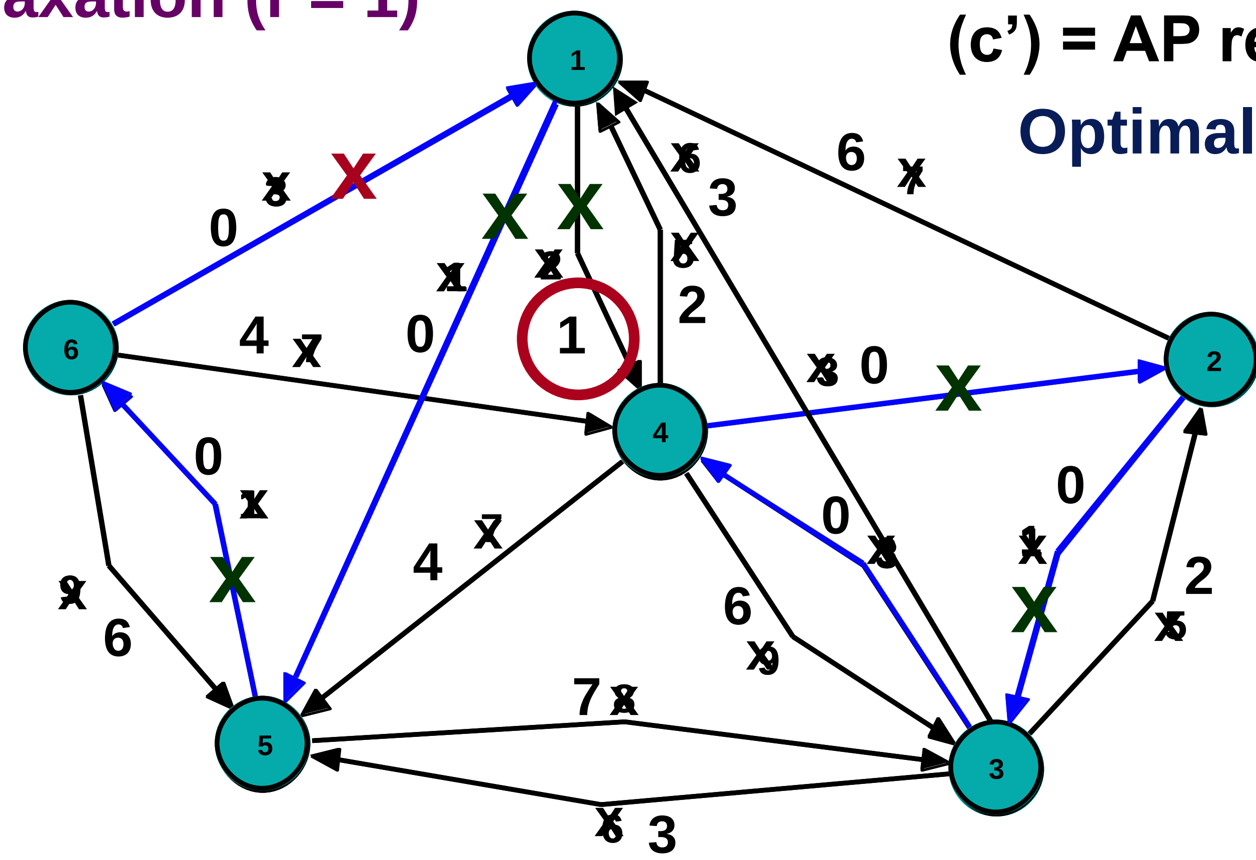
<u>Example:</u> Additive Bounding Procedure

r-SSA Relaxation (r = 1)

LB1 = 12

(c') = AP reduced costs

Optimal assignment



v(1-SSAR) = 0 + 0 + 1 + 0 + 0 + 0 = 1

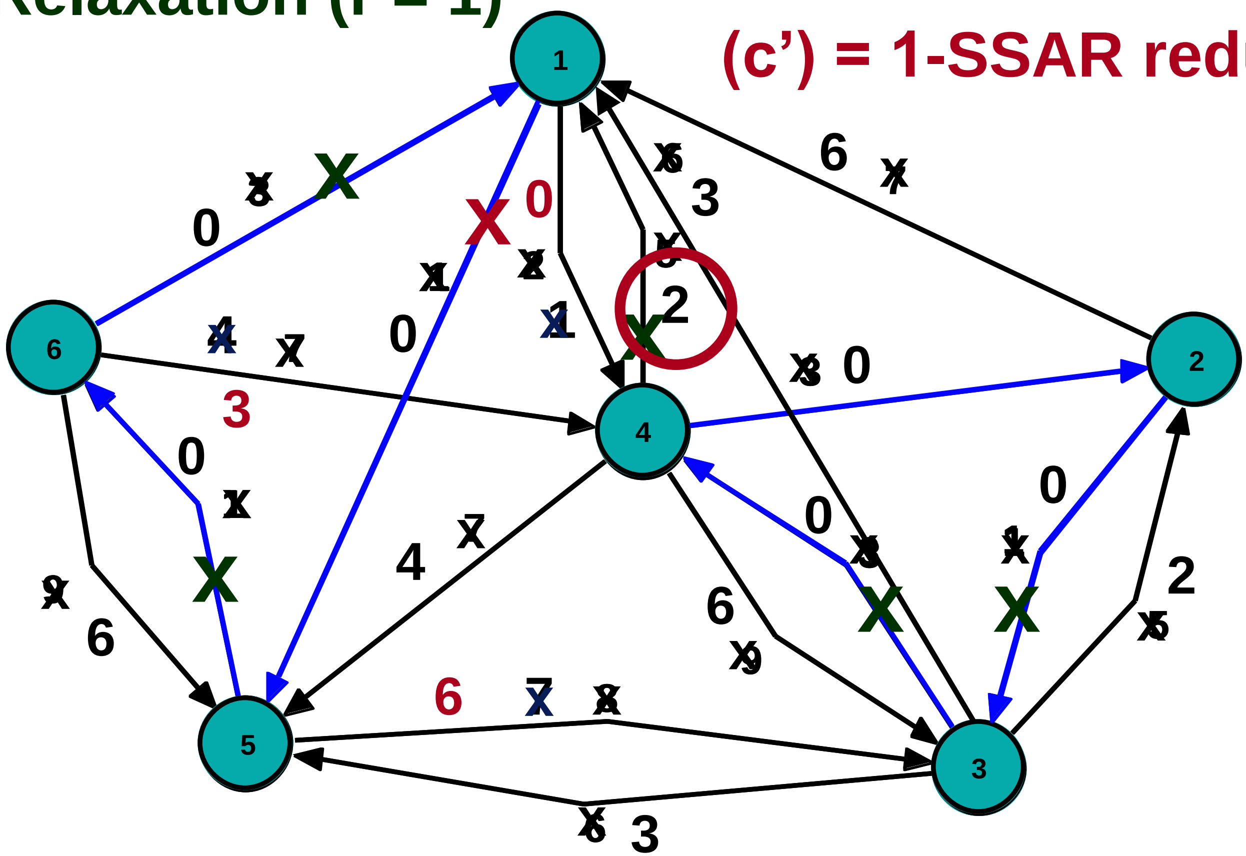
LB2 = LB1 + v(1-SSAR) = 12 + 1 = 13

Example: Additive Bounding Procedure

LB1 = 13

(c') = 1-SSAR reduced costs

r-Anti-SSA Relaxation (r = 1)



v(1-Anti-SSAR) = 0 + 0 + 2 + 0 + 0 + 0 = 2

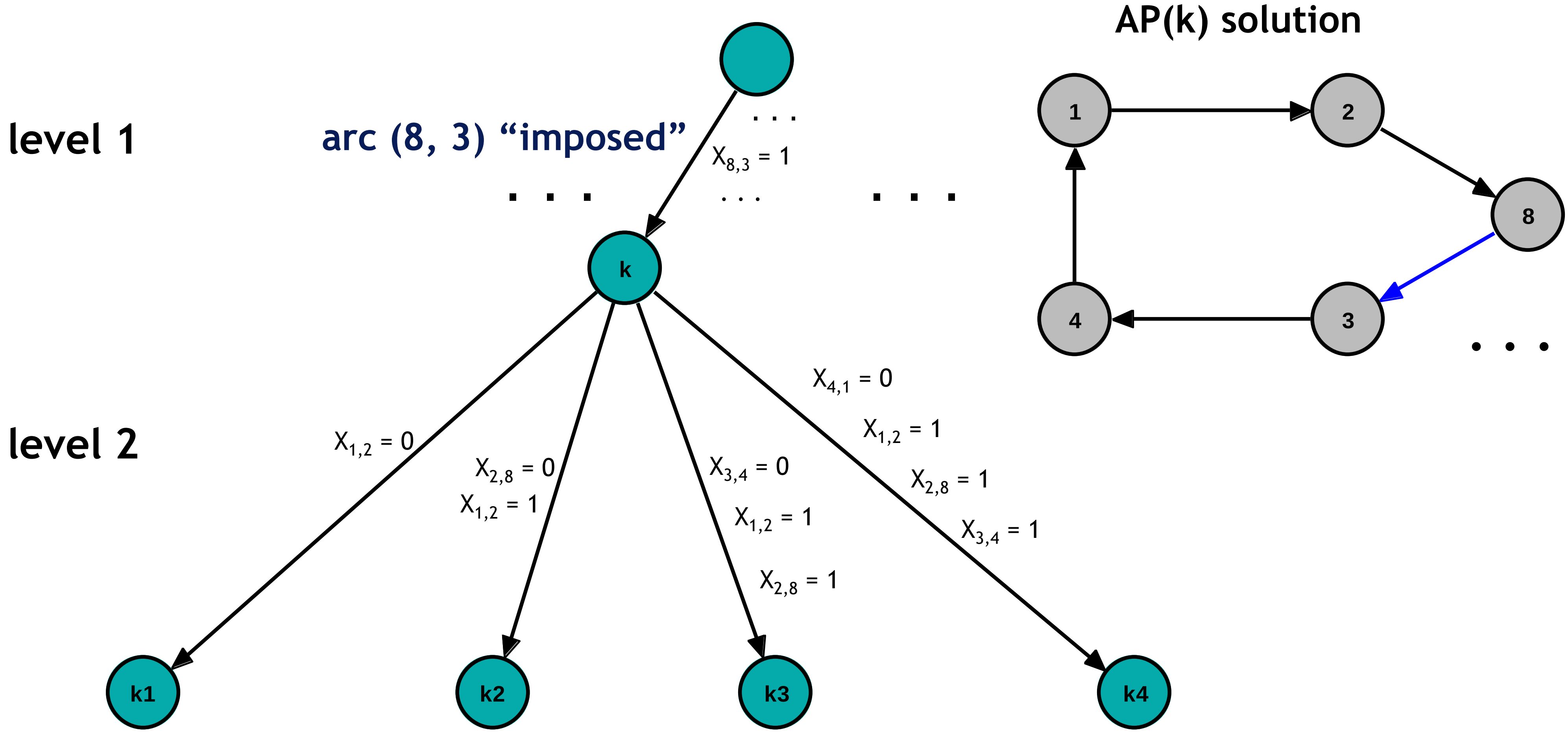
LB2 = LB1 + v(1-Anti-SSAR) = 13 + 2 = 15

BRANCH-AND-BOUND ALGORITHM FOR ATSP BASED ON THE AP RELAXATION

(Carpaneto-T., Man. Sc. 1980)

- At each node of the decision tree solve the AP-Relaxation of the corresponding subproblem (LB= v(AP)).
- If LB $\geq Z^*$ fathom the node (Z^* = cost of the best solution).
- If the AP solution contains no subtour (feasible solution), update Z^* (and the best solution) and "fathom" the node.
- Otherwise: SUBTOUR-ELIMINATION BRANCHING SCHEME:
  - Select the subtour S with the minimum number h of not imposed arcs.
  - Generate h descendent nodes so as to forbid subtour S for each of them (by "imposing" and "excluding" proper arc subsets).

BRANCHING TREE FOR ATSP



BRANCH-AND-BOUND ALGORITHM
by Carpaneto-Dell'Amico-T. (ACM TOMS, 1995)

- At the root node of the decision tree:

  - solve the AP Relaxation;

  - apply the Patch Heuristic Algorithm (Karp-Steele, 1985) to determine an initial tour of cost Z^*;

  - apply a Reduction Procedure (based on the AP "reduced costs" c'_{ij}) to fix to zero as many variables as possible (transformation of the complete graph into a sparse one): if $v(AP) + c'_{ij} \geq Z^*$ set $x_{ij} = 0$ (i.e. remove arc (i,j) from A).

- At each node, the corresponding AP Relaxation is computed (by using effective parametric techniques) in $O(n^2)$ time.

BRANCH-AND-CUT ALGORITHMS (Padberg - Rinaldi for the STSP, 1987-1991)

- At each node, a Lower Bound is obtained by solving an LP Relaxation of the corresponding subproblem, containing only a subset of the constraints (degree constraints, some connectivity constraints, ...).

- The LP Relaxation is iteratively tightened by adding valid inequalities that are violated by the current optimal LP solution (x^*_{ij}).
- These inequalities are identified by solving the Separation Problem:

- given a solution (x^*_{ij}), find a member $a\,x \leq b$ of a given family F of valid inequalities for ATSP, such that

$a\,x^* > b$ holds (maximum of $d = a\,x^* - b$, with $d > 0$).

* Exact or heuristic Separation Procedures can be used.

SEPARATION PROBLEM FOR THE CONNECTIVITY CONSTRAINTS

- Given an optimal LP solution (x^*_{ij}) such that:

$$\sum_{i\in V} x^*_{ih} = \sum_{i\in V} x^*_{hj} = 1 \qquad\qquad h \in V$$

does exist a vertex subset S ($S \subset V$, $r \in S$) such that:

$$\sum_{i\in S} \sum_{j\in V\setminus S} x^*_{ij} < 1 \quad ?$$

EXACT SEPARATION PROCEDURE FOR THE CONNECTIVITY CONSTRAINTS

- For each vertex $t \subset V \setminus \{r\}$, define the NETWORK

$G^* = (V,A)$, source = r, sink = t

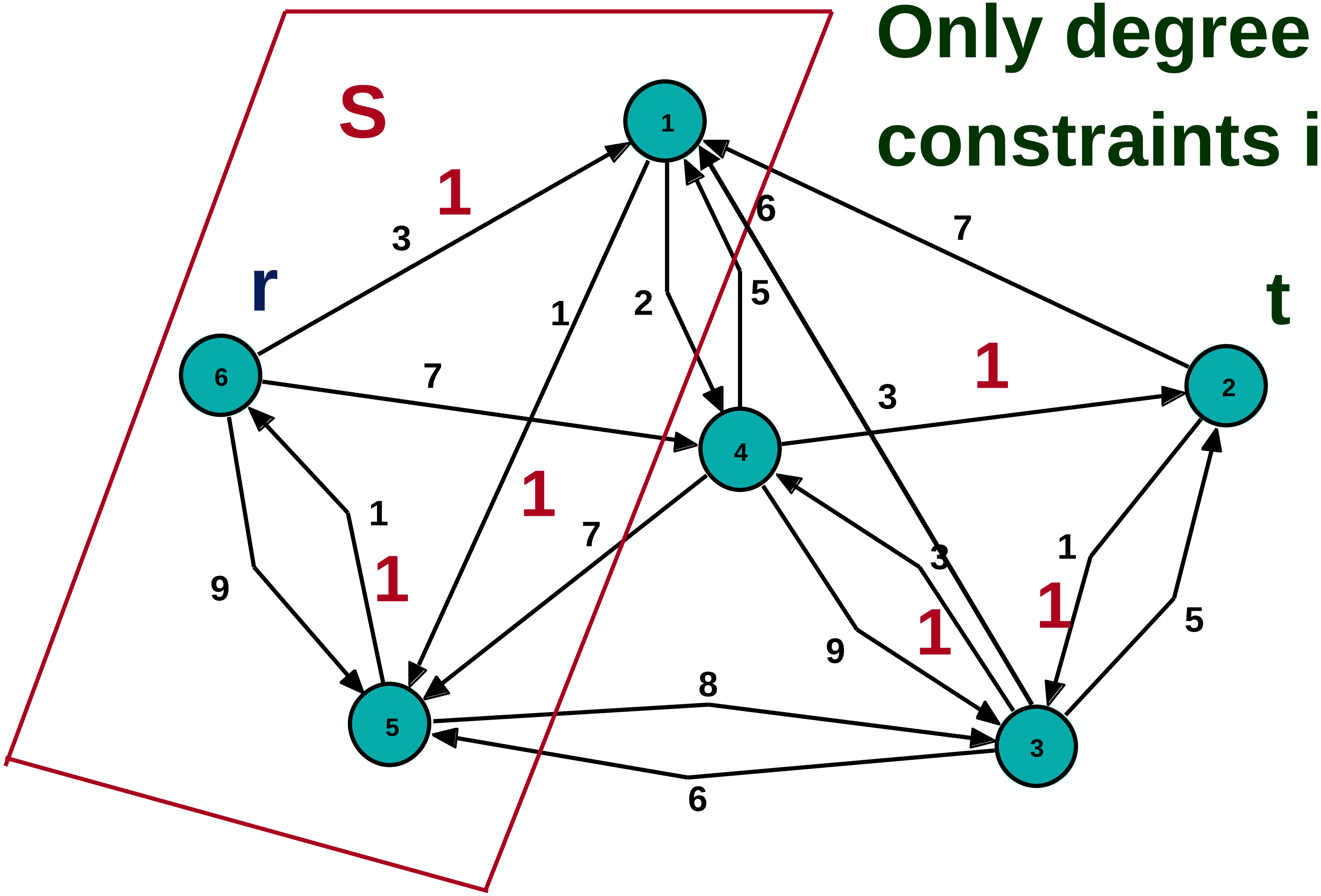
capacity of arc $(i,j) = x^*_{ij}$

- CAPACITY of “cut” $(S, V \setminus S)$ with $r \in S$ and $t \in V \setminus S$ =

$$\sum_{i \in S} \sum_{j \in V \setminus S} x^*_{ij}$$

Example NETWORK $G^* = (V,A)$ capacity (x^*)

Only degree constraints imposed



CAPACITY of “cut” (S, V \ S) = 0

EXACT SEPARATION PROCEDURE FOR THE CONNECTIVITY CONSTRAINTS

- Minimum Capacity Cut from r to t = Maximum Flow from r to t
- Determine the MAXIMUM FLOW from r to t: capacity of cut $(S^*, V \setminus S^*)$

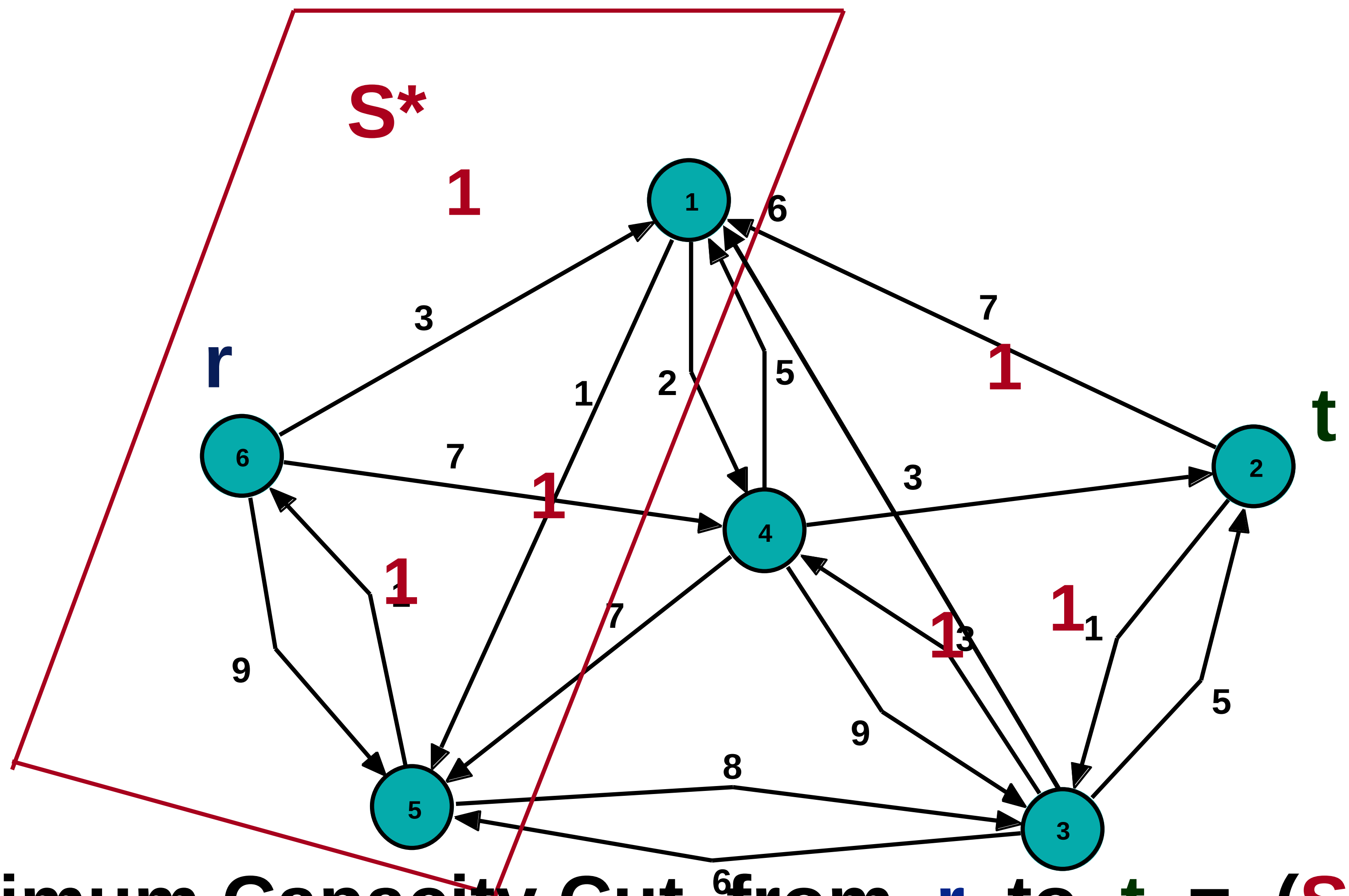
* If Maximum Flow < 1 then constraint

$$\sum_{i \in S^*} \sum_{j \in V \setminus S^*} x_{ij} \geq 1 \quad \text{is violated}$$

(most violated connectivity constraint with $r \in S$ and $t \in V \setminus S$)

$O(n^3)$ time for each vertex t (global time $O(n^4)$)

Example **NETWORK** $G^* = (V,A)$ **capacity** (x^*)



Minimum Capacity Cut from r to t = (S*, V\S*)

Maximum Flow from r to t = 0: $\sum_{i \in S^*} \sum_{j \in V \setminus S^*} x_{ij} = 0$

BRANCH-AND-CUT ALGORITHM
(Fischetti-T., Man. SC. 1997)

- **Separation procedures** for the identification of the following valid inequalities:

  - Connectivity constraints (exact alg.);

  - Comb inequalities (heur. alg.);

  - D_k^+ and D_k^- inequalities (Groetschel-Padberg, 1985) (exact and heur. alg.);

  - ODD CAT inequalities (Balas, 1989) (heur. alg.)

PRICING PROCEDURE

A pricing procedure is applied to decrease the number of variables considered in the LP Relaxation:

1) Select a subset T of variables (other variables set to zero);
2) Solve the LP Relaxation by considering only the variables in T;
3) Compute the reduced cost of the variables not in T;
4) Add to T (a subset of) the variables with negative reduced cost and return to Step 2);

   if no negative reduced cost is found then STOP (the current solution is the optimal solution of the original LP Relaxation).

- An effective Pricing Scheme, based on the optimal solution of an associated Assignment Problem, is used to decrease the number of variables added to T in Step 4).

BRANCHING SCHEME (Fischetti-Lodi-T., 2003)

- Select a fractional variable x^*_{ij} close to 0.5 and having been "persistently" fractional in the "last" optimal LP solutions.

- Generate 2 descendent nodes by imposing:

  $x_{ij} = 1$ and $x_{ij} = 0$, respectively.

TRANSFORMATION OF ATSP INTO STSP

Any ATSP instance with n vertices can be transformed into an equivalent STSP instance with 2n nodes (Jonker-Volgenant, 1983; Junger-Reinelt-Rinaldi, 1995; Kumar-Li, 2007).

Very effective Branch-and-Cut algorithms for STSP have been proposed:

- **Padberg-Rinaldi (Oper. Res. Letters 1987, SIAM Review 1991)**
- **Groetschel-Holland (Math. Progr. 1991)**
- **Applegate-Bixby-Chvatal-Cook (Doc. Math., J. DMV 1998, Concorde Code 1999, Princeton Univ. Press 2006).**

* Concorde Code

Most effective code for STSP

"POLYNOMIAL" MILP FORMULATIONS

- Miller-Tucker-Zemlin (*J. ACM*, 1960);
- Gavish-Graves (*MIT Tech. Report* 1978)
- Fox-Gavish-Graves (*Operations Research* 1980);
- Wong (*IEEE Conference …*, 1980);
- Claus (*SIAM J. on Algebraic Discrete Methods*, 1984);
- Langevin-Soumis-Desrosiers (*Operations Research Letters*, 1990)
- Desrochers-Laporte (*Operations Research Letters*, 1990);
- Padberg-Sung (*Mathematical Programming*, 1991);
- Gouveia (*European Journal of Operational Research*, 1995)
- Gouveia-Voss (*European Journal of Operational Research*, 1995)
- Gouveia-Pires (*European Journal of Operational Research*, 1999);
- Gouveia-Pires (*Discrete Applied Mathematics*, 2001);
- Sherali-Driscoll (*Operations Research* 2002);
- Sarin-Sherali-Bhootra (*Operations Research Letters*, 2005);
- Sherali-Sarin-Tsai (*Discrete Optimization*, 2006);
- Godinho-Gouveia-Pesneau (*Discrete Applied Mathematics*, 2011).

"POLYNOMIAL" MILP FORMULATIONS

Classification and Comparisons:

- Gouveia-Pesneau (*Networks*, 2006);
- Oncan-Altinel-Laporte (Review, *Computers & Operations Research*, 2009);
- Godinho-Gouveia-Pesneau (*Progress in Combinatorial Optimization*, J. Wiley, 2011);
- Roberti-T. (*EURO Journal on Transportation and Logistics*, 2013);
- Bektas-Gouveia (*European Journal of Operational Research*, 2014).

ALTERNATIVE SUBTOUR ELIMINATION CONSTRAINTS

- MTZ: Miller-Tucker-Zemlin (*Journal of ACM* 1960)
  - $(n - 1)$ additional continuous variables:

    u_i = order of vertex i in the tour ($i \in V \setminus \{1\}$)

    $(1 \le u_i \le n - 1)$
  - $O(n^2)$ additional constraints:

    $u_i - u_j + (n - 1)\, x_{ij} \le n - 2 \quad i \in V \setminus \{1\},\ j \in V \setminus \{1\}$

- if $x_{ij} = 0$: the constraint is always satisfied,

- if $x_{ij} = 1$: $u_i - u_j \le -1$: $\quad u_i \le u_j - 1$

ALTERNATIVE SUBTOUR ELIMINATION CONSTRAINTS (2)

- MTZ: Miller-Tucker-Zemlin (*Journal of ACM* 1960)
  - $(n - 1)$ additional continuous variables:

    u_i = order of vertex i in the tour ($i \in V \setminus \{1\}$)

    $$(1 \leq x_{ij} \leq n - 1)$$
  - $O(n^2)$ additional constraints:

    $$u_i - u_j + (n - 1)\, x_{ij} \leq n - 2 \quad i \in V\setminus\{1\},\ j \in V\setminus\{1\}$$

- DL: Desrochers-Laporte (*Oper. Res. Letters* 1990)

  "lifted" MTZ constraints, $2\,n$ additional constraints

- GG: Gavish-Graves (*MIT Tech. Report* 1978)

  Single Commodity Flow Formulation
  - n^2 additional continuous variables
  - $n + n^2$ constraints

ALTERNATIVE SUBTOUR ELIMINATION CONSTRAINTS 3

- **EC-MCF** (Godinho-Gouveia-Pesneau, *Progress in Combinatorial Optimization*, J. Wiley, 2011).

  $n\,(n^3 - 4\,n^2)$ binary variables:

  $2\,n^3$ constraints

Very large core memory requirements

ALTERNATIVE ATSP FORMULATIONS

Which is the BEST formulation for ATSP?

- Number of variables and constraints?
- Value of the Lower Bound of the corresponding Linear Programming Relaxation?
- Value of the Lower Bound computed through "Structured" Relaxations?
- CPU time for computing the Lower Bound?
- Global CPU time for finding the optimal integer solution (MILP Solver, "ad hoc" algorithm)?

ALTERNATIVE MILP FORMULATIONS

- **Lower bound value comparison:**

- LB(AP) $\leq$ LB(Additive Procedure) $\leq$ LB(DFJ)
- LB(AP) $\leq$ LB(MTZ)
- LB(MTZ) $\leq$ LB(GG) $\leq$ LB(DFJ)
- LB(MTZ) $\leq$ LB(DL) $\leq$ LB(EC-MCF)
- LB(DFJ) $\leq$ LB(EC-MCF)

ALTERNATIVE MILP FORMULATIONS

- Polynomial formulations:

  * involve a polynomial number of constraints,

  * can be given directly on input to a *MILP Solver*

but

- their Linear Programming Relaxations generally produce Lower Bounds weaker (and more time consuming) than those corresponding to the Dantzig-Fulkerson-Johnson formulation, without or with (for formulation LB(EC-MCF)) the addition of valid inequalities.

* Formulations MTZ, GG and DL are the best formulations to be directly used within CPLEX.

COMPUTATIONAL RESULTS

- **Test instances (34 to 444 vertices):**

18 ATSP benchmark instances from TSPLIB (Reinelt, *ORSA Journal on Computing*, 1991),

4 additional real-world instances from Balas (2000).

COMPUTATIONAL RESULTS

*** DIGITAL ALPHA 533 MHz (times in seconds).

- CDT : Carpaneto-Dell'Amico-T. (AP Relax.) (*ACM Trans*. *Math*. *Soft*. 1995);
- FT : Fischetti-T. (Additive Bounding Procedure) (*Math*. *Program*.1992);

time limit for each instance: 1,000 seconds

• LP Solver: CPLEX 6.5.3

- FLT : Fischetti-Lodi-T. (B. & C.) (*Man*. *Sc*. 1997, *LNCS Springer* 2003).
- CONCORDE (transformation from ATSP to STSP): Applegate, Bixby, Chvatal, Cook 1999, 2003.

time limit for each instance: 10,000 seconds

COMPUTATIONAL RESULTS

*** Intel Core 2 Duo 2.26 GHz (at least 10 times faster than DIGITAL ALPHA) ("scaled" times).

- LP and MILP Solver: CPLEX 11.2

- MTZ (Miller-Tucker-Zemlin),
- DL (Desrochers-Laporte),
- GG (Gavish-Graves)

"scaled" time limit for each instance: 18,000 seconds

	CDT		FT		FLT		Concorde		MTZ		DL		GG	
INST	LB	Time	LB	Time	LB	Time	LB	Time	LB	Time	LB	Time	LB	Time
Ftv33	7.85	0.0	3.73	0.1	0.00	0.0	0.00	0.3	7.64	49.1	5.35	34.0	7.03	55.1
Ftv35	6.25	0.0	3.53	0.2	0.85	0.4	0.68	9.0	6.12	62.7	4.04	52.3	5.59	119.3
Ftv38	6.01	0.0	3.01	0.3	0.88	0.6	0.52	14.5	5.87	90.2	3.45	57.1	5.44	214.6
Ftv44	5.70	0.0	4.46	0.1	0.37	0.5	0.12	9.1	5.55	164.0	2.43	31.7	5.18	180.2
Ftv47	6.98	0.1	3.49	0.4	1.01	0.5	0.62	23.4	6.77	360.4	2.83	120.0	6.54	582.7
Ftv55	10.76	1.1	6.59	1.5	0.81	1.4	0.44	9.0	10.55	437.4	6.05	194.8	10.31	911.6
Ftv64	6.42	0.8	4.89	1.3	1.36	2.6	0.33	20.8	6.31	1145.0	4.24	325.3	5.84	3225.6
Ftv70	9.44	3.3	6.92	3.7	0.92	1.1	0.26	17.8	9.26	1687.0	4.69	524.0	8.75	11256
Ft70	1.80	3.3	0.57	0.3	0.02	0.2	0.01	3.2	1.77	tl	0.88	652.2	1.10	3260.7
Ft53	14.11	tl	1.56	0.2	0.00	0.1	0.00	0.6	14.04	3123.8	12.93	tl	12.45	379.5
Br17	100.00	3.6	0.00	0.0	0.00	0.0	0.00	0.2	94.23	10.9	43.59	34.6	68.91	8.2
Balas84	14.07	tl	5.53	986.6	1.01	15.7	1.01	78.0	13.98	tl	9.67	tl	12.34	tl
Balas108	25.00	tl	9.87	tl	**1.97**	89.0	2.63	1416.0	24.80	tl	19.50	tl	18.14	tl
Balas160	19.40	tl	11.34	tl	1.26	671.1	1.26	7848.0	19.18	tl	18.83	tl	16.16	tl
Balas200	15.63	tl	8.68	tl	1.24	1712.8	0.74	2294.2	15.55	tl	15.31	tl	12.86	tl
Rbg323	0.00	0.1	0.00	0.3	0.00	0.4	0.00	23.9	0.00	tl	0.00	848.1	0.00	tl
Rbg358	0.00	0.1	0.00	0.5	0.00	0.5	0.00	29.3	0.00	tl	0.00	tl	0.00	tl
Rbg403	0.00	0.1	0.00	1.1	0.00	1.3	0.00	49.3	0.00	tl	0.00	tl	0.00	tl
Rbg443	0.00	0.1	0.00	1.2	0.00	1.4	0.00	34.5	0.00	tl	0.00	tl	0.00	tl
P43	97.37	tl	**0.37**	tl	0.16	9.3	0.16	22.7	97.34	tl	96.16	tl	85.23	tl
Ry48p	13.21	tl	2.94	20.3	0.53	0.8	0.35	22.9	12.88	1316.0	4.25	3417.3	11.17	tl
Kro124p	6.22	tl	2.73	135.7	0.04	1.0	0.00	9.9	6.13	2296.6	3.46	2316.8	5.47	tl
AVG	16.65	364	3.65	234	0.57	**114**	0.42	543	16.27	8670	11.71	7755	13.57	9918
		(14)		(18)	(21)	(22)	(19)	(22)	(7)	(12)	(11)	(13)	(8)	(11)

Comparison of the exact algorithms

- FLT and Concorde are the only algorithms able to solve all the 22 instances to optimality within the given time limit (10,000 seconds).
- At the root node, Concorde generally obtains better Lower Bounds than those obtained by FLT, but the global CPU time of FLT is always smaller than that of Concorde.

- The branch-and-bound algorithms CDT and FT dominate (w.r.t. the CPU times and the number of instances solved to optimality) the polynomial formulations MTZ, GG and DL.
- By considering the Lower Bounds at the root node:

  - LB(CDT) (=LB(AP)) is only slightly worse than LB(MTZ) (average gaps: 16.65 and 16.27, respectively);

  - LB(FT) (Additive Procedure) always better than LB(MTZ) and LB(GG), and globally better than LB(DL).

Randomly Generated Instances: c_{ij} integer uniformly random in [1,1000], n = 500, 1000

INST	CDT LB	CDT Time	FT LB	FT Time	FLT LB	FLT Time	Concorde LB	Concorde Time
Ran500.0	0.00	0.1	0.00	0.3	0.00	0.8	0.00	29.2
Ran500.1	0.06	0.1	0.06	1.1	0.00	1.8	0.00	20.0
Ran500.2	0.15	0.2	0.15	3.1	0.00	31.4	0.00	52.7
Ran500.3	0.06	0.1	0.06	6.6	0.02	55.0	0.04	232.4
Ran500.4	0.07	0.1	0.07	4.6	0.00	3.4	0.02	53.8
Ran1000.0	0.00	0.7	0.00	94.2	0.00	3.4	0.00	219.2
Ran1000.1	0.05	0.7	0.05	9.1	0.00	23.5	0.00	191.7
Ran1000.2	0.09	3.9	0.09	90.0	0.00	150.7	0.00	900.4
Ran1000.3	0.05	1.1	0.05	42.9	0.01	62.2	0.01	3977.2
Ran1000.4	0.07	1.5	0.07	48.5	0.01	148.7	0.01	3122.2
AVG	0.06	0.8	0.06	30.0	0.00	48.1	0.01	899.9