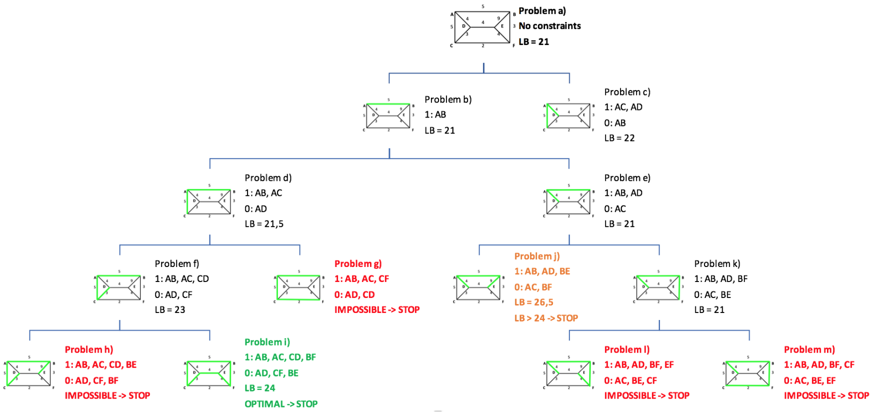
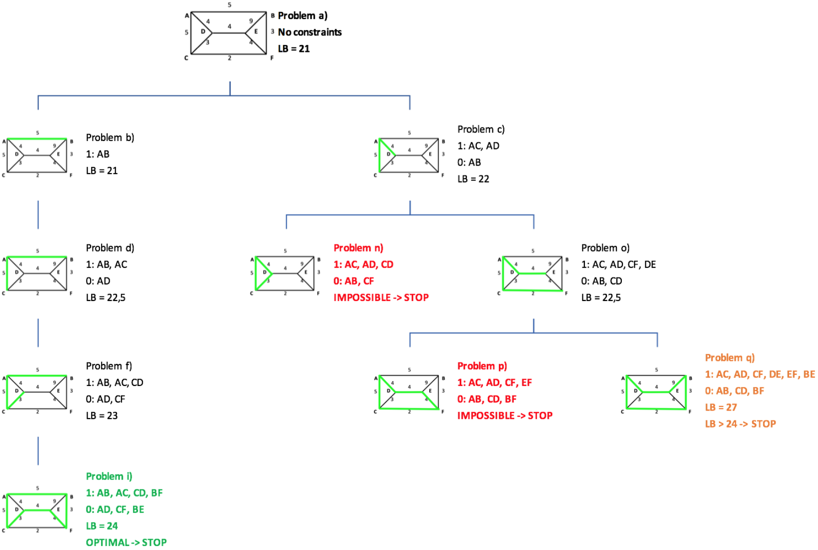
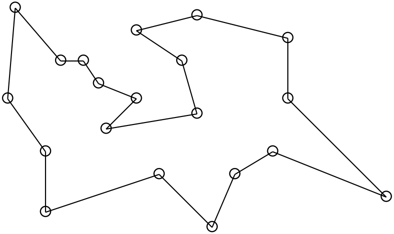
A tree search (Branch-and-Bound)




Problem a)
No constraints
LB = 21
Problem b)
1: AB
LB = 21
Problem c)
1: AC, AD
0: AB
LB = 22
Problem d)
1: AB, AC
0: AD
LB = 22,5
Problem n)
1: AC, AD, CD
0: AB, CF
IMPOSSIBLE -> STOP
Problem o)
1: AC, AD, CF, DE
0: AB, CD
LB = 22,5
Problem f)
1: AB, AC, CD
0: AD, CF
LB = 23
Problem p)
1: AC, AD, CF, EF
0: AB, CD, BF
IMPOSSIBLE -> STOP
Problem q)
1: AC, AD, CF, DE, EF, BE
0: AB, CD, BF
LB = 27
LB > 24 -> STOP
Problem i)
1: AB, AC, CD, BF
0: AD, CF, BE
LB = 24
OPTIMAL -> STOP

ILP formulation of TSP (undirected graphs)



- Let V be the set of vertices and E the set of edges.
- We introduce a binary variable x_e for every edge $e \in E$: $x = 1$ iff the edge e is included in the tour.
- The goal is to minimize the total cost of edges selected, i.e. traveled by the salesman.

- For every vertex v, precisely two of the edges incident with v are selected.
- For any vertex $v \in V$, let $\delta(v)$ be the set of edges incident with v
- We obtain the following initial formulation:

$$
\begin{aligned}
&min && \sum_{e \in E} c_e x_e && (1)\\
&\text{subject to} && \sum_{e \in \delta(v)} x_e = 2,\ \forall v \in V && (2)\\
& && x_e \in \{0,1\}. && (3)
\end{aligned}
$$

Constraints (2) are called *assignment constraints* and ensure that every vertex has a degree of 2...

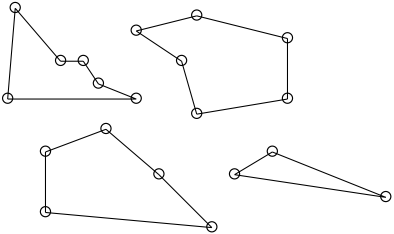
- For every vertex v, precisely two of the edges incident with v are selected.
- For any vertex $v \in V$, let $\delta(v)$ be the set of edges incident with v
- We obtain the following initial formulation:

$$\begin{aligned} & min & & \sum_{e \in E} c_e x_e & (1) \\ & \text{subject to} & & \sum_{e \in \delta(v)} x_e = 2, \ \forall v \in V & (2) \\ & & & x_e \in \{0,1\}. & (3) \end{aligned}$$

Constraints (2) are called *assignment constraints* and ensure that every vertex has a degree of 2...

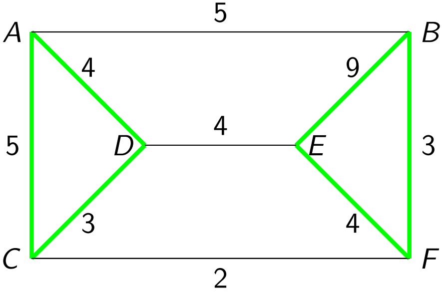
...but it's not enough!

Indeed, by using such a formulation, a solution like the following would be feasible:



But we do not want our salesman to "teleport" or "jump" from one city to another → Our formulation is hence incomplete.

In the previous example, we would have obtained the following solution:



Anyway, this solution provides valuable information: it is a lower bound for the problem.

Solving the example with AMPL

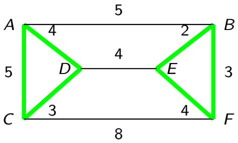
Consider the previous example but change the costs of (B,E) from 9 to 2 and (C,F) from 2 to 8:

Model:

```
set V ordered;
set E within {i in V, j in V: ord(i) < ord(j)};
param cost{E};
var X{E} binary;
minimize TourCost: sum {(i,j) in E} cost[i,j] * X[i,j];
subject to VisitAllVertices {i in V}:
sum {(i,j) in E} X[i,j] + sum {(j,i) in E} X[j,i] = 2;
```

Data:

```
set V := A B C D E F;
set E := (A,B) (A,C) (A,D)
         (B,E) (B,F) (C,D)
         (C,F) (D,E) (E,F);
param cost :=
[A,B] 5
[A,C] 5
[A,D] 4
[B,E] 2
[B,F] 3
[C,D] 3
[C,F] 8
[D,E] 4
[E,F] 4;
```



The AMPL solver finds an optimal solution with value 21, by selecting (A,C), (A,D), (B,E), (B,F), (C,D), (E,F). But it is not what we want...

To formulate the correct model for TSP, we need to add more constraints.

Subtour Elimination Constraints (I)

- TSP asks for a **Hamiltonian circuit** (i.e., a circuit passing through each vertex exactly once, without subtours).
- Given $S \subsetneq V$, $S \neq \emptyset$, the cut with shores S and V/S is the set of those edges with one endpoint in S and the other in V/S; we denote it by $\delta(S)$.
- We must impose that the solution is **connected**.

Subtour Elimination Constraints (I)

- TSP asks for a **Hamiltonian circuit** (i.e., a circuit passing through each vertex exactly once, without subtours).
- Given $S \subsetneq V$, $S \neq \emptyset$, the cut with shores S and V/S is the set of those edges with one endpoint in S and the other in V/S; we denote it by $\delta(S)$.
- We must impose that the solution is **connected**.
- For every proper subset of vertices, at least two edges in the solution have one endpoint in S and the other outside S:

$$\sum_{e \in \delta(S)} x_e \geq 2, \forall S \subsetneq V, S \neq \emptyset$$

Subtour Elimination Constraints (II)

- Actually, we can write:

$$\sum_{e \in \delta(S)} x_e \geq 2, \forall S \subset V : 2 \leq |S| \leq |V| - 2$$

  Why?

- Violated constraints will have the following form:

$$\sum_{e \in \delta(S)} x_e^* < 2.$$

- These constraints are known as *Subtour Elimination Constraints* (SEC).

The SEC problem

- By introducing this family of constraints, we are adding an **exponential** number of constraints, since we are considering the power set of V (except the empty set).
- Thus, this approach is not polynomial for a double reason:
  - we adopted an ILP model;
  - its formulation has an exponential number of variables and constraints.

A correct ILP formulation of TSP (undirected graphs)

- Putting together the objective function (1) and constraints (2), (3) and (4), we get to the correct formulation:

$$\min \sum_{e \in E} c_e x_e \tag{1}$$

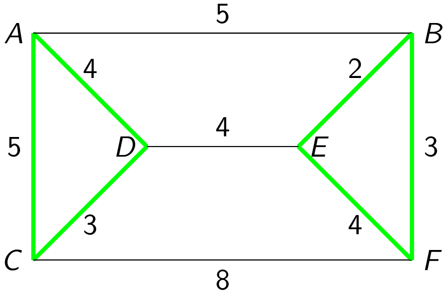
$$\text{subject to} \quad \sum_{e \in \delta(v)} x_e = 2, \ \forall v \in V \tag{2}$$

$$\sum_{e \in \delta(S)} x_e \geq 2, \forall S \subset V : 2 \leq |S| \leq |V| - 2 \tag{3}$$

$$x_e \in \{0, 1\}. \tag{4}$$

Another approach considering flow constraints

- How to avoid adding an exponential number of constraints?
- Consider the solution obtained before including subtours :



- Suppose we want to send some quantity of flow from vertex A to another vertex:
  - since A is connected to C and D, there is no problem sending it to them;
  - instead, we cannot send the flow to F.

Adding flow constraints

- For every edge $e = (u, v) \in E$, we introduce two variables $y_{(u,v)}$ and $y_{(v,u)}$ and we allow the flow to pass only through edges selected in the cycle:
  - $y_{(u,v)} \leq x_{(u,v)}$
  - $y_{(v,u)} \leq x_{(u,v)}$.
- We consider the vertices A and F as source and destination respectively; we want that the flow outgoing from A is equal to 1, whereas the incoming flow is 0:
  - $y(\delta^+(A)) = 1$
  - $y(\delta^-(A)) = 0$.
- We want all other vertices except F to conserve the flow:
  - $y(\delta^-(u)) = y(\delta^+(u)), \forall u \neq A, F$.

- In this way, we are forcing the flow to arrive in the only vertex left, i.e. F, thus connecting A and F.
- This could be repeated for every source vertex and for every destination vertex.
- Since the number of vertices is n, we are adding a polynomial number of constraints and variables.

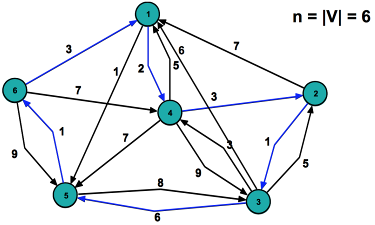
Conclusions

- We defined the TSP, by using one simple formulation and one more complete and beautiful but, unfortunately, with an exponential number of constraints.
- We saw several methods to tackle an ILP problem (heuristic, approximation and exact algorithms).
- We solved an easy instance of TSP by using Branch-and-Bound.

References

- William J. Cook, *In Pursuit of the Traveling Salesman – Mathematics at the Limits of Computation*, https://press.princeton.edu/books/paperback/9780691163529/in-pursuit-of-the-traveling-salesman
- Gilbert Laporte, *A Short History of the Traveling Salesman Problem*, http://neumann.hec.ca/chairedistributique/common/laporte-short.pdf
- Renata Mansini, *Algoritmi di Ottimizzazione*, University of Brescia, https://www.unibs.it/ugov/degreecourse/65037
- Operations Research Group, *Asymmetric TSP*, University of Bologna, http://www.or.deis.unibo.it/algottm/files/8_ATSP.pdf
- Springer, *Encyclopedia of Optimization*, http://www.springer.com/it/book/9780387747583
- University of Waterloo, *TSP*, http://www.math.uwaterloo.ca/tsp/
- Wikipedia, *Traveling Salesman Problem*, https://en.wikipedia.org/wiki/Travelling_salesman_problem

Appendix A – The asymmetric TSP (ATSP)



- Let V be the set of vertices.
- Sometimes going from vertex i to vertex j does not cost the same than going from vertex j to vertex i, or maybe it is not possible to go through both directions.
- Then, instead of the set of edges E, we need to use the set of arcs A: this variant is called **asymmetric** TSP.
- We introduce a binary variable $x_{i,j}$ for every arc $(i, j) \in A$: if $x_{i,j} = 1$, then arc (i, j) appears on the tour.

ILP formulation of ATSP (directed graphs)

- Keeping in mind what we said about symmetric TSP, we can formulate ATSP as follows:

$$
\begin{aligned}
& min \sum_{(i,j)\in A} c_{i,j}x_{i,j} && (1)\\
\text{subject to} \quad & \sum_{j\neq i} x_{i,j} = 1,\ \forall i \in V && (2)\\
& \sum_{i\neq j} x_{i,j} = 1,\ \forall j \in V && (3)\\
& \sum_{i\in S, j\in S} x_{i,j} \leq |S|-1, \forall S \subset V, S \neq \emptyset && (4)\\
& x_{i,j} \in \{0,1\}. && (5)
\end{aligned}
$$

- With constraints (2) and (3), we are distinguishing arcs entering in a certain vertex and arcs exiting from it.

Appendix B – William J. Cook, *The TSP: postcards from the edge of impossibility* (EURO 2019)

https://www.youtube.com/watch?v=5VjphFYQKj8