CIVIL-557
Decision-aid methodologies in transportation
Lecture II
Solution methods

Fabian Torres

Transport and Mobility Laboratory TRANSP-OR
École Polytechnique Fédérale de Lausanne EPFL

EPFL

Outline

How do we solve MILPs

$$\text{Minimize} \sum_{(i,j)\in A} \sum_{k\in K} c_{ij} x_{ijk}$$

$$\text{s.t.} \sum_{k\in K} y_{ik} = 1 \qquad \forall i \in N,$$

$$\sum_{j\in\delta^+(i)} x_{ijk} - \sum_{j\in\delta^-(i)} x_{ijk} = 0 \qquad \forall i \in N, k \in K,$$

$$\sum_{j\in\delta^+(o)} x_{ojk} - \sum_{j\in\delta^-(o)} x_{ojk} = 1 \qquad \forall k \in K,$$

$$y_{ik} = \sum_{j\in\delta^-(i)} x_{ijk} \qquad \forall i \in N \cup \{o\}, k \in K,$$

$$y_{dk} = \sum_{i\in\delta^-(d)} x_{idk} \qquad \forall k \in K,$$

$$u_{ik} - u_{jk} + Q x_{ijk} \leq Q - q_j \qquad \forall (i,j) \in A, k \in K,$$

$$q_i \leq u_{ik} \leq Q \qquad \forall i \in V, k \in K,$$

$$x_{ijk} \in \{0,1\}, y_{ik} \in \{0,1\} \qquad \forall (i,j) \in A, k \in K.$$

EPFL

Outline

Branch-and-Bound

Consider the problem:

$$z = min\{c^T x : x \in X\}$$

Definition: Feasible Solution

A feasible solution is a feasible point x^* that satisfies all constraints described by the set X.

Definition: Optimal Solution

An optimal solution is a feasible solution $x^* \in X$ such that $c^T x^* \leq c^T x$ for all $x \in X$.

Definition: Non Feasible Solution

A non feasible solution is a point x^* that violates some constraint of the problem ($x^* \notin X$). The violated constraints could be the integrality constraints if x^* is fractional.

Branch-and-Bound

Consider the problem:

$$z = min\{cx : x \in X\}$$

Let $X = X_1 \cup X_2 \cdots \cup X_k$ be separable into different sets from 1 to k. And the sets $X_1 \cap X_2 \cdots \cap X_k = \emptyset$ are disjoints sets.

And let $z_1 = min\{cx : x \in X_1\}, \ldots, z_k = min\{cx : x \in X_k\}$ be the value of the objective value for each subset

Then $z = min\{z_1, z_2, \ldots, z_k\}$ the smallest value of the objective function for all subsets is also the solution over the entire set X.

EPFL

Disjoint sets

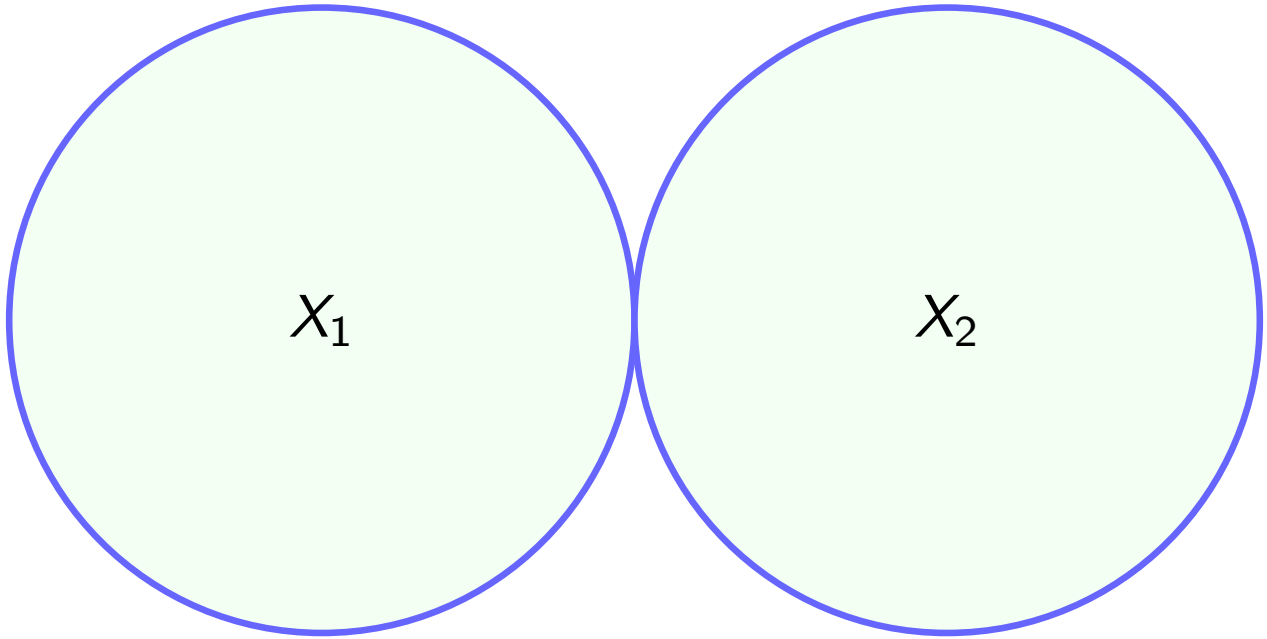


Figure: Disjoint sets

Small example

$$
\begin{aligned}
\textit{Min} \quad & x_1 + x_2 + x_3 \\
s.t. \quad & \\
& x \in X \\
& x \in \{0,1\}^3
\end{aligned}
$$

Idea

If we enumerate all possible solutions we can solve the problem.

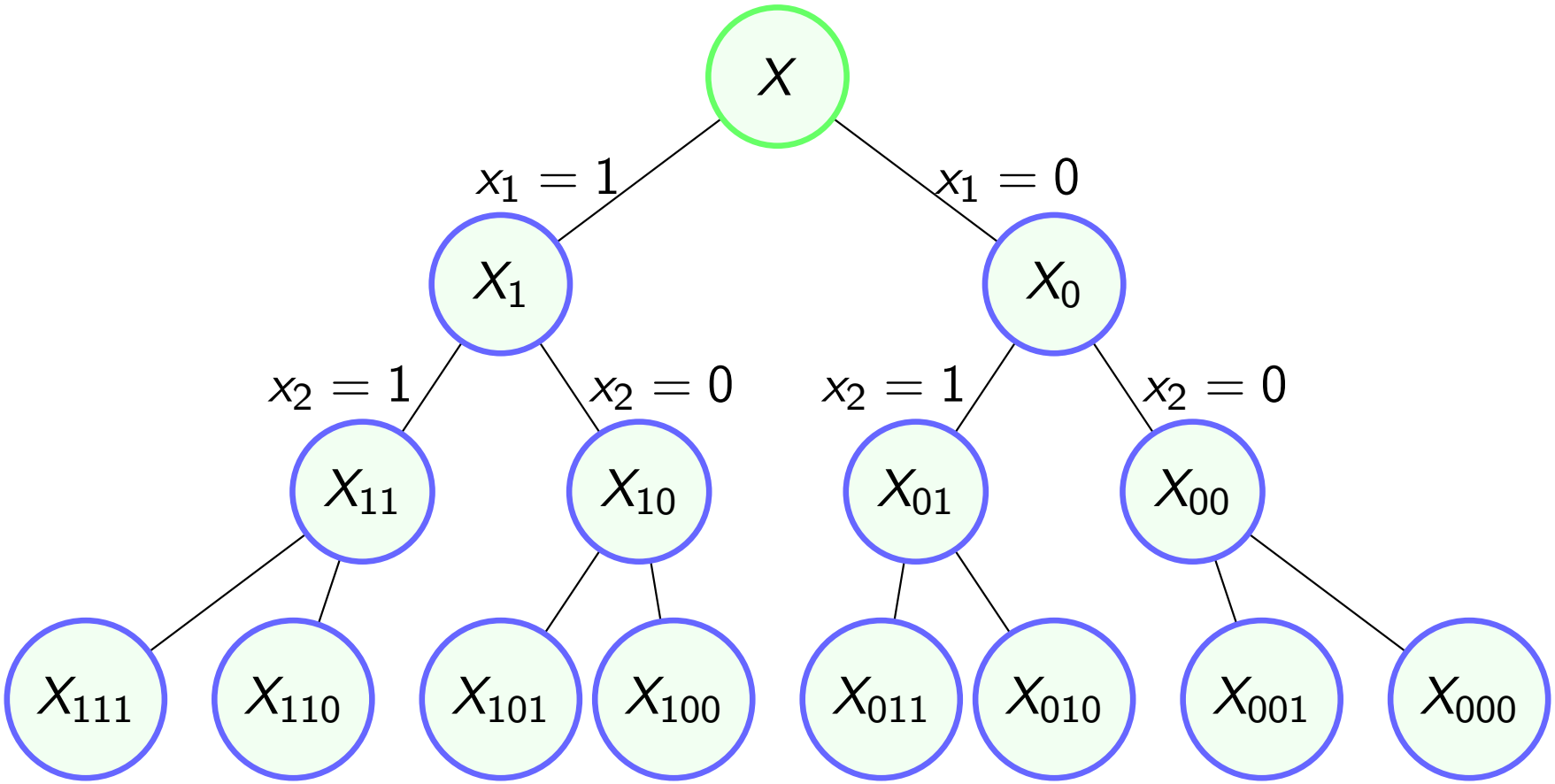


Figure: Binary enumeration tree

EPFL

If we use this method on larger problems the total number of problems in the binary enumeration tree will be:

$$2^N$$

Where N is the number of variables. It will take too long if we have a large number of variables.

Idea

What if we could use some method to eliminate large portions of the enumeration tree before we branch.

Let look at some concepts first.

Upper bound

Feasible solution value $\geq$ Optimal solution value

- Feasible solutions are solutions that meet all the constraints of the problem, however, they may or may not be optimal.
- The objective value of a feasible solution gives an upper bound on the problem. Since we have a feasible solution, the optimal solution cannot be worse!
- A feasible solution can be difficult to find in some cases, but we can find them as we explore branch-and-bound tree.

EPFL

Linear relaxation

Linear Relaxation

If we eliminate the binary constraints and we allow the variables to take fractional solutions we obtain the linear relaxation. Notice that feasible solutions are not eliminated in the relaxations since we are only removing constraints, i.e., the variables can also take integer values.

EPFL

Lower Bound

Relaxation solution value $\leq$ Optimal solution value

- The value of the objective function of the linear relaxation is always less than or equal to the objective function of the original problem.
- Hence, whatever the optimal solution is, it cannot be better than the value of the relaxation.

EPFL

Pruning

At each node of the enumeration tree we can evaluate the bounds. The upper bound (feasible solution)is global, and the lower bound(solution of the relaxation) is local.

1. Prune by infeasibility (feasible region $= \emptyset$);
2. Prune by bound (LB is more than the UB);
3. Prune by optimality (the solution is feasible, e.g., binary variables are binary).

EPFL

Branching

If we cannot prune the problem, we select a fractional variable and we branch on it.

$$X_1 = \{x : x_f \leq \left\lfloor x_f^* \right\rfloor\}$$

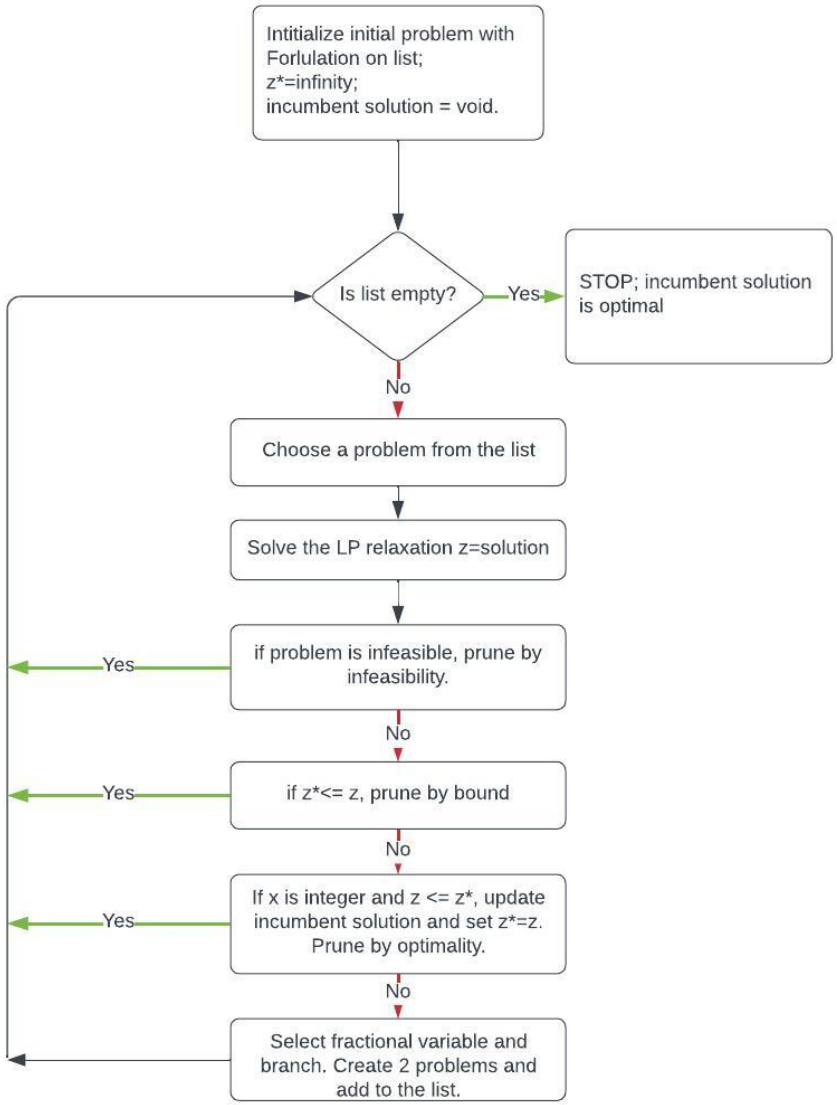
$$X_2 = \{x : x_f \geq \left\lceil x_f^* \right\rceil\}$$

No feasible solution is eliminated $X = X_1 \cup X_2$

Two disjoint sets $\emptyset = X_1 \cap X_2$

EPFL

B&B


Intitialize initial problem with Forlulation on list;
z*=infinity;
incumbent solution = void.
Is list empty?
Yes
STOP; incumbent solution is optimal
No
Choose a problem from the list
Solve the LP relaxation z=solution
if problem is infeasible, prune by infeasibility.
Yes
No
if z*<= z, prune by bound
Yes
No
If x is integer and z <= z*, update incumbent solution and set z*=z. Prune by optimality.
Yes
No
Select fractional variable and branch. Create 2 problems and add to the list.

Example

$$
\begin{aligned}
z = min \; & -4x_1 + x_2 \\
& 7x_1 - 2x_2 \leq 14 \\
& x_2 \leq 3 \\
& 2x_1 - 2x_2 \leq 3 \\
& x \in \mathbb{N}^2
\end{aligned}
$$

Example

Linear relaxation: we add the linear relaxation to the list of active problems.

$$
\begin{aligned}
z = min \; & -4x_1 + x_2 \\
& 7x_1 - 2x_2 \leq 14 \\
& x_2 \leq 3 \\
& 2x_1 - 2x_2 \leq 3 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

Ex. iteration 1

Select node and Optimize:
We relax the integer constraints and solve using the simplex algorithm (i.e., Gurobi)

$$z = -\frac{59}{7}$$
$$x_1 = \frac{20}{7}$$
$$x_2 = 3$$

Since no feasible solution is found (upper bound), we set $z^* = \infty$

EPFL

Ex. iteration 1

Branching: We can see that the value of x_1 is not integer and hence it violates the integrality constraints. Therefore, we choose the variable x_1 to branch.

$$X_1 = \{x : x_1 \leq \left\lfloor \frac{20}{7} \right\rfloor = 2\}$$

$$X_2 = \{x : x_1 \geq \left\lceil \frac{20}{7} \right\rceil = 3\}$$

No feasible solution is eliminated $X = X_1 \cup X_2$

Two disjoint sets $\emptyset = X_1 \cap X_2$

EPFL

Ex. iteration 1

Add problems to the list of active problems

- We have now created 2 new problems (X_1 and X_2),
- We call these problems active problems since they have not been explored yet,
- We keep a list of all active problems that need to be explored,
- Next iteration we must select an active problem to explore, e.g., X_1.

.

Ex. iteration 2

Select node and Optimize:
We select problem X_1 and solve it using the simplex algorithm (i.e., Gurobi)

$$z = -\frac{15}{2}$$
$$x_1 = 2$$
$$x_2 = \frac{1}{2}$$

EPFL

Ex. iteration 2

Branching:
We can see that the value of x_2 is not integer and hence it violates the integrality constraints. Therefore, we choose the variable x_2 to branch.

$$X_{11} = \{x : x_2 \leq \left\lfloor \frac{1}{2} \right\rfloor = 0\}$$

$$X_{12} = \{x : x_2 \geq \left\lceil \frac{1}{2} \right\rceil = 1\}$$

No feasible solution is eliminated $X_1 = X_{11} \cup X_{12}$

Two disjoint sets $\emptyset = X_{11} \cap X_{12}$

Ex. iteration 3

Select node and Optimize:
Now the list of active problems is (X_2, X_{11} and X_{12}).
We arbitrarily select X_2 and reoptimize using the simplex algorithm.

Prune by infeasibility:
The problem is infeasible. The constraint $x_1 \geq 3$ is not feasible in the model. Therefore, we prune the problem, i.e., we eliminate it from the active problem list and do not have to look at it any further.

The active list now contains only 2 problems (X_{11} and X_{12})

EPFL

Ex. iteration 4

Select node and Optimize:
We select arbitrarily an active problem X_{12}.

$$z^{12} = -7$$
$$x_1 = 2$$
$$x_2 = 1$$

The solution is integer! **Update incumbent solution:** At the beginning we did not have a feasible solution and we set $z^* = \infty$, we now found an feasible solution. $z^* > -7$ We up date the incumbent solution $z^* = -7$ and $x^*(2, 1)$. **Prune by integrality:** There is no need to branch.

Ex. iteration 5

Select node and Optimize:
There is only 1 active problem in the list X_{11}.

$$z^{11} = -6$$
$$x_1 = \frac{3}{2}$$
$$x_2 = 0$$

Check bound: $z^* = -7 < -6 = z^{11}$ any solution obtained from branching will be higher than -6, therefore we prune by bound.
Prune by bound:

Ex. iteration 5

Select node and Optimize:
There are no active problems and we stop the algorithm.
The incumbent solution is the optimal solution.

$$z^* = -7$$
$$x_1^* = 2$$
$$x_2^* = 1$$

EPFL

Node selection strategies

During the previous example of B&B we were selecting the problem from the list of active problems arbitrarily, however, there are more intelligent ways of selecting an active problem from the list. Following are some strategies:

Best node first strategy

Select the node that has the best bound first.

Depth first search strategy

Descend the enumeration tree as quickly as possible to find a feasible solution. Select the node that has been branch the most.

Best node first strategy

Advantage

This strategy is guaranteed to lead to the smallest B&B tree.

Disadvantage

It can take a long time to find a feasible solution that can be used to prune the tree.

Depth first search strategy

Advantage

This strategy finds a feasible integer solution quickly.

Disadvantage

It leads to a bigger tree with more problems to solve.

Why not use both?

- In practice, we **do** use both.
- Start with depth first search to find a feasible solution.
- Once a feasible solution is found select the best node first to improve the lower bound.
- We can even go back and forth between the two strategies.

EPFL

Branching strategies

During the example we chose any fractional variable to branch on.

Most fractional variable

Select the variable that has the fractional value closest to 0.5.

Strong branching

We branch up and down on a subset of fractional variables, then we solve the linear relaxation for each up and down branch. The variable that has the largest increase on the lower bound is chosen.

Most fractional variable

Let C be a set of all fractional variables, and f_j the value of variable $j \in C$

$$j^* = \arg\ \max_{j \in C} \min[f_j, 1 - f_j]$$

Advantage

Quick rule to select a variable.

Disadvantage

It could lead to small increases of the lower bound.

EPFL

Strong branching

$$j^* = \arg\max_{j \in C} \min[z_j^{up}, z_j^{down}]$$

Advantage

It leads to a larger increase of the lower bound.

Disadvantage

It takes the solution of two LR for each variable.

Outline

Ideas

Adding constraints

If we add a constraints to an MILP formulation, we run the risk of eliminating feasible solutions. **Valid inequalities are inequalities that do not eliminate feasible solutions, therefore they can be added**.

Removing constraints

If we remove a constraint of an MILP formulation, we will not eliminate feasible solutions, however, we will allow some solutions that are not feasible, i.e., solutions that violate the removed constraint.

EPFL

How do we solve MILPs

$$\text{Minimize} \sum_{(i,j)\in A} c_{ij} x_{ij}$$

$$\text{s.t.} \sum_{j\in\delta^+(i)} x_{ij} = 1 \qquad \forall i \in N,$$

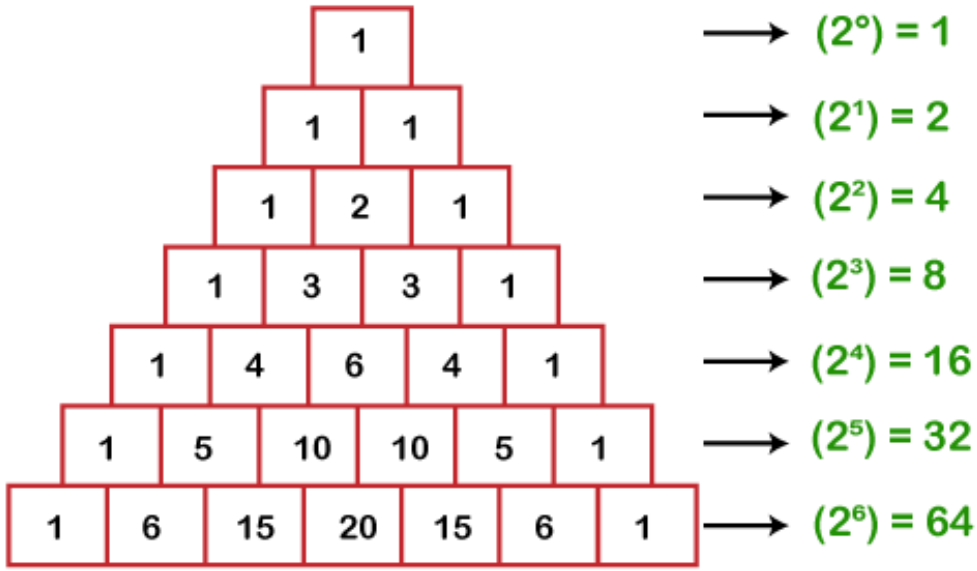
$$\sum_{i\in\delta^-(j)} x_{ij} = 1 \qquad \forall j \in N,$$

$$\sum_{(i,j)\in\delta^+(S)} x_{ij} \geq \left\lceil \frac{\sum_{i\in S} q_i}{Q} \right\rceil \qquad \forall S \subseteq N, S \neq \emptyset,$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A.$$

Exponential constraints

Pascal's triangle



Subtours

Some subtours are more expensive than others!

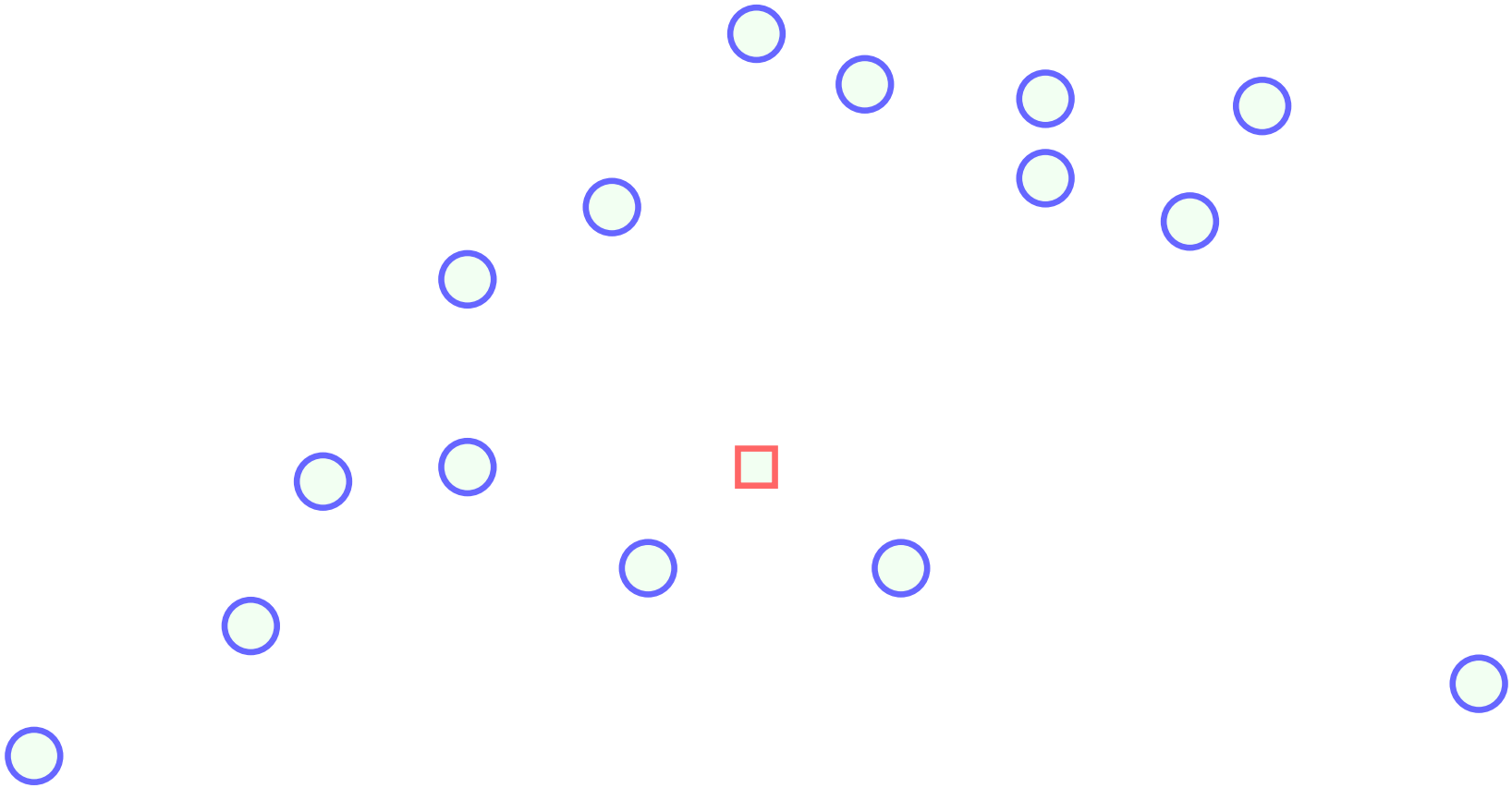


Figure: Subtours ?

Subtours

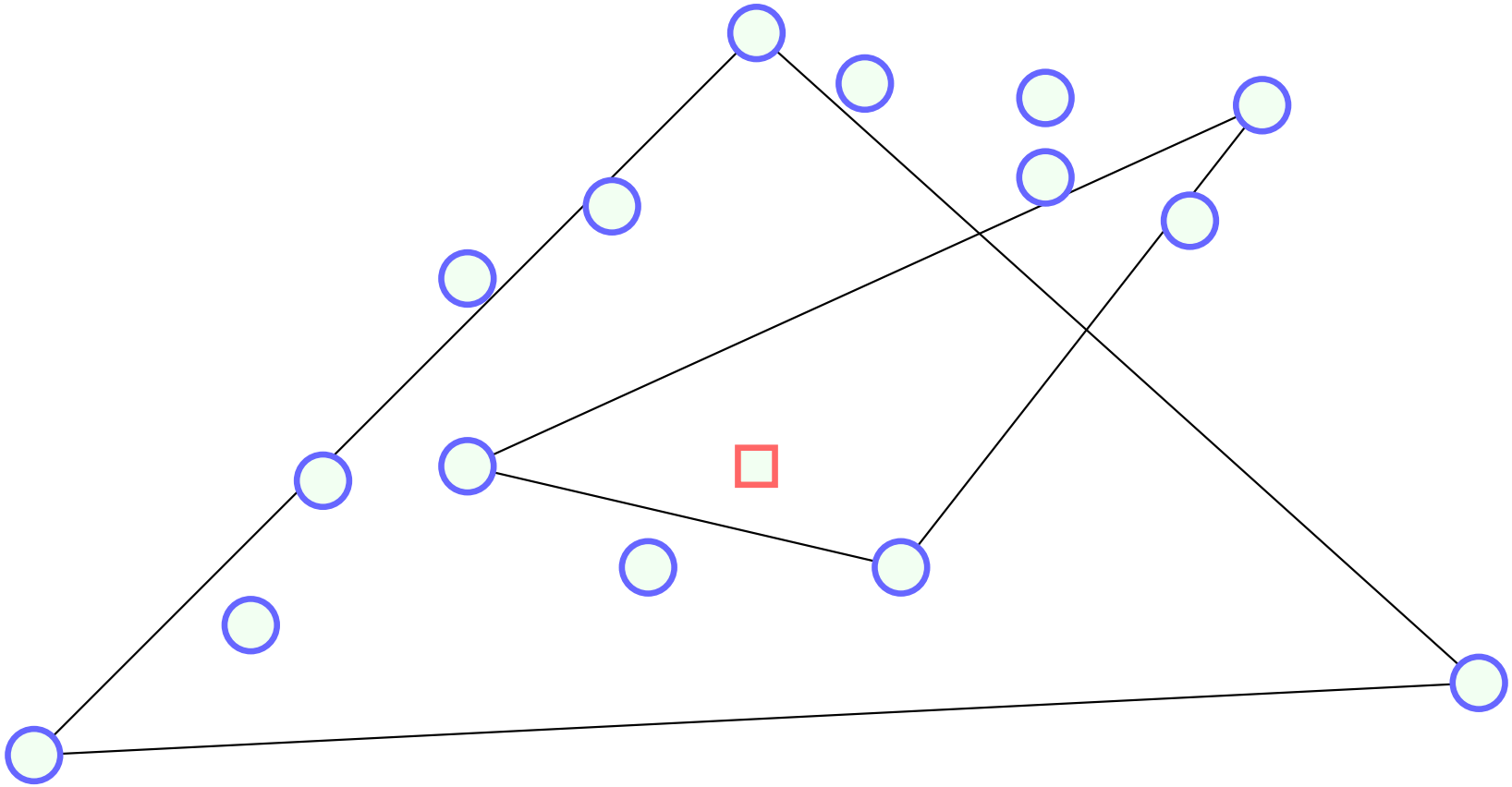


Figure: Unlikely subtours

Subtours

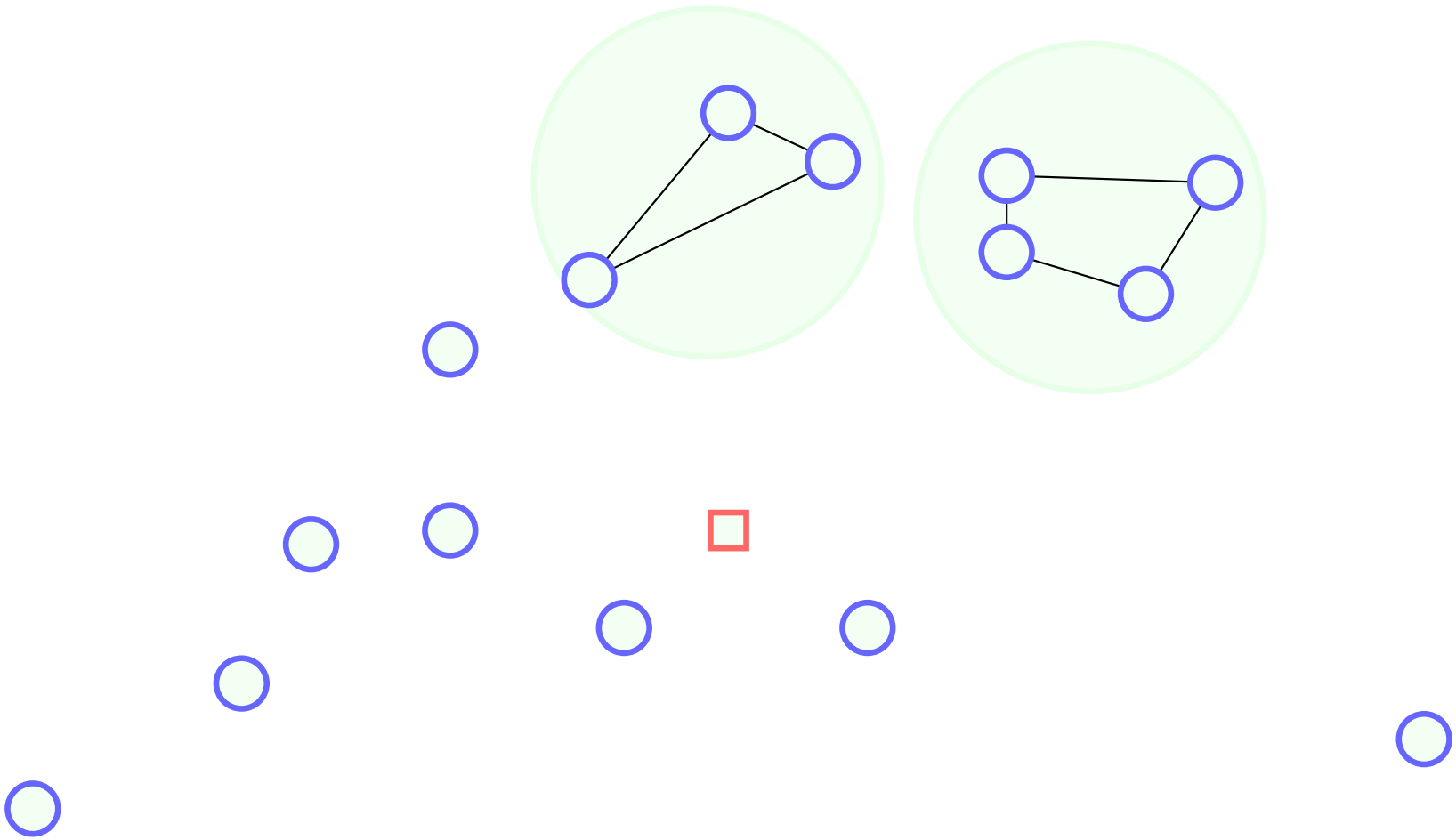
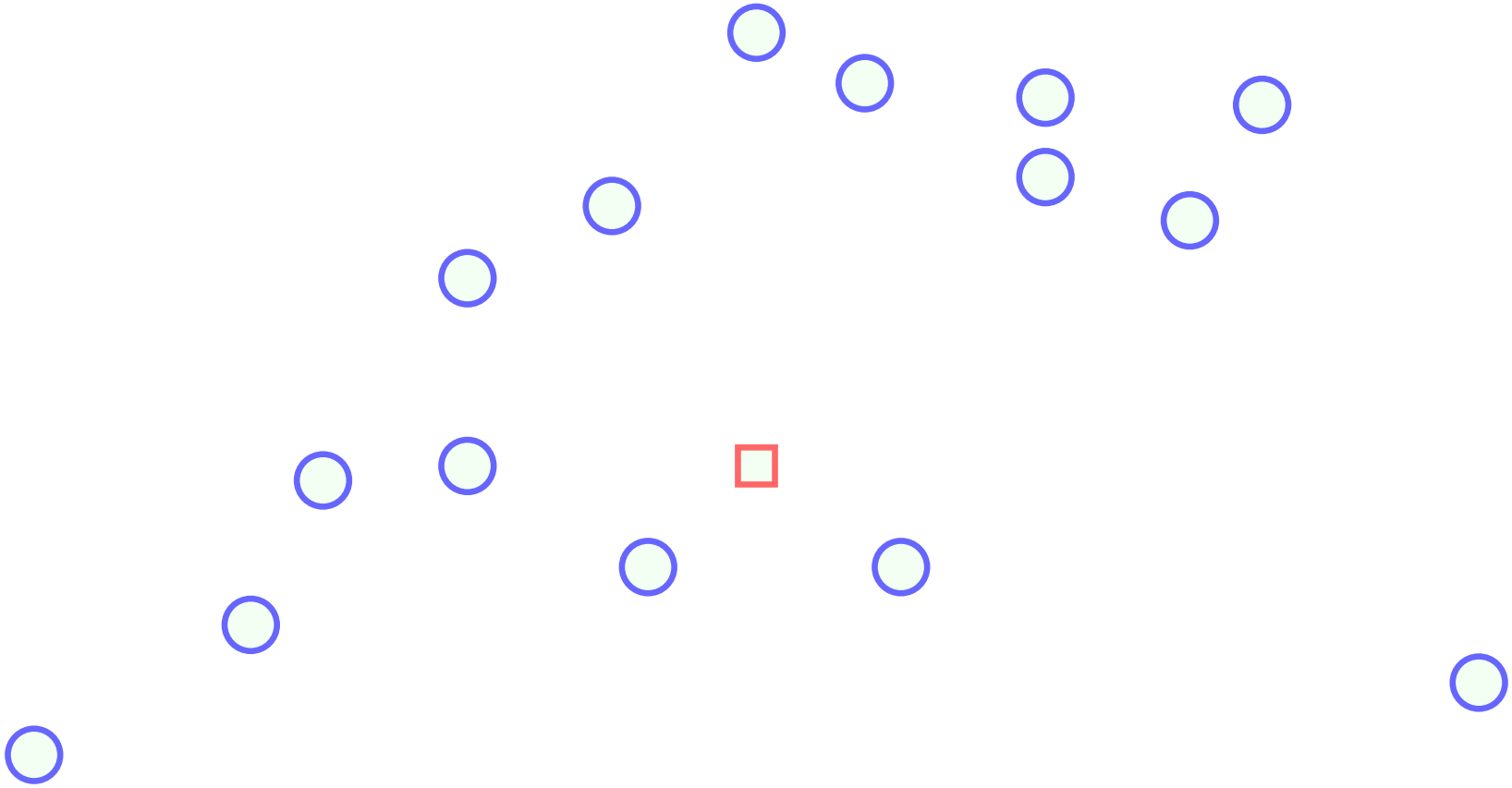


Figure: Likely subtours

Subtours

Can we know what subsets of customers will form "Good" subtours beforehand and add the corresponding constraints to the model?



Subtours

No, there is no simple way to find all SECs that will be needed!

The way to find the subsets of customers that will form subtours is to solve the relaxation and see in the solution of the relaxation what SECs are violated, add them to the model and continue to find more iteratively, until the optimal solution is found.

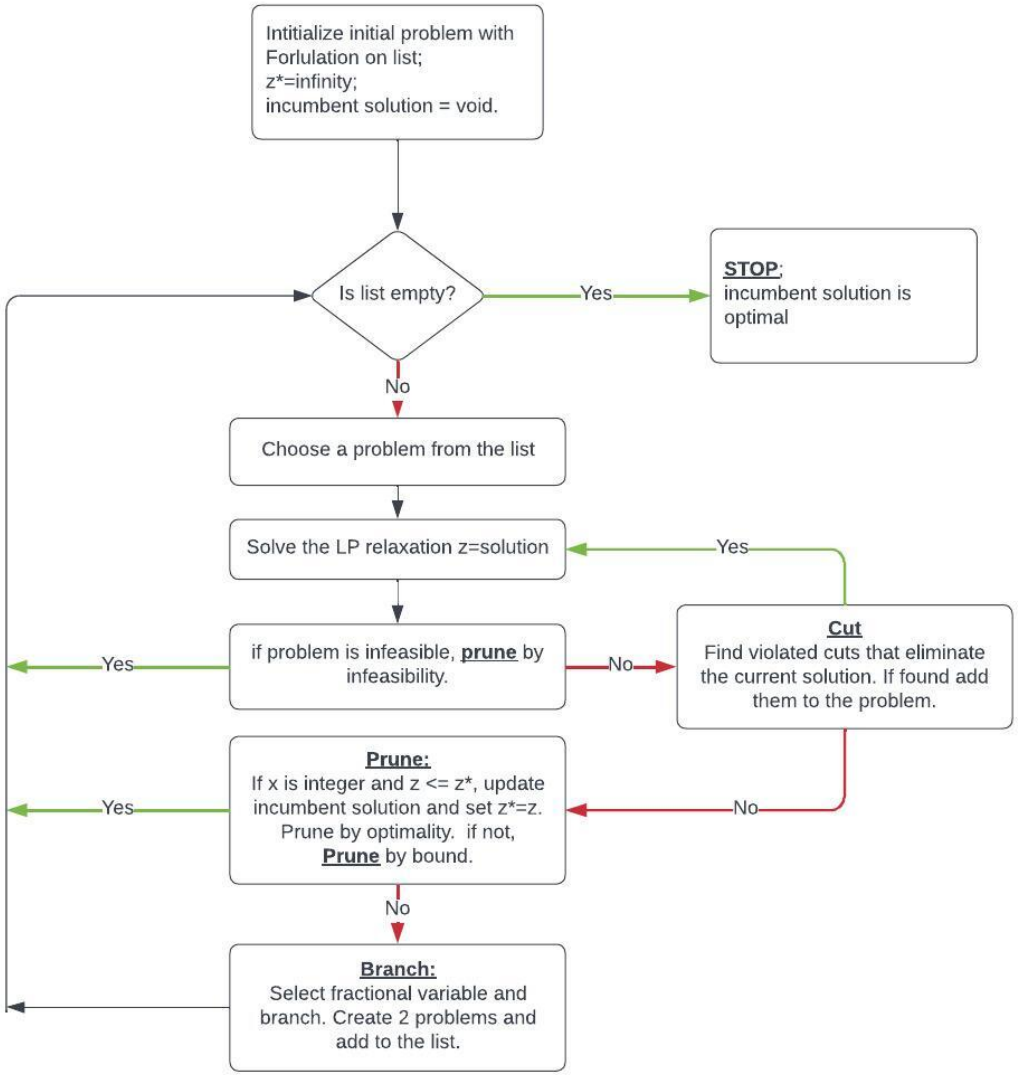
EPFL

Linear Relaxation (LR)

$$
\begin{aligned}
\textit{Minimize} \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} & \\
s.t. \quad & \sum_{j\in\delta^+(i)} x_{ij} = 1 & \forall i \in N, \\
& \sum_{i\in\delta^-(j)} x_{ij} = 1 & \forall j \in N. \\
& x_{ij} \geq 0 & \forall (i,j) \in A
\end{aligned}
$$

We eliminate the Rounded Capacity Inequalities (RCIs) and binary constraints.

Branch-and-cut



Branch-and-cut

Finding violated RCIs can be difficult (not impossible) if the solution in fractional. It is easy if the solution is integer.

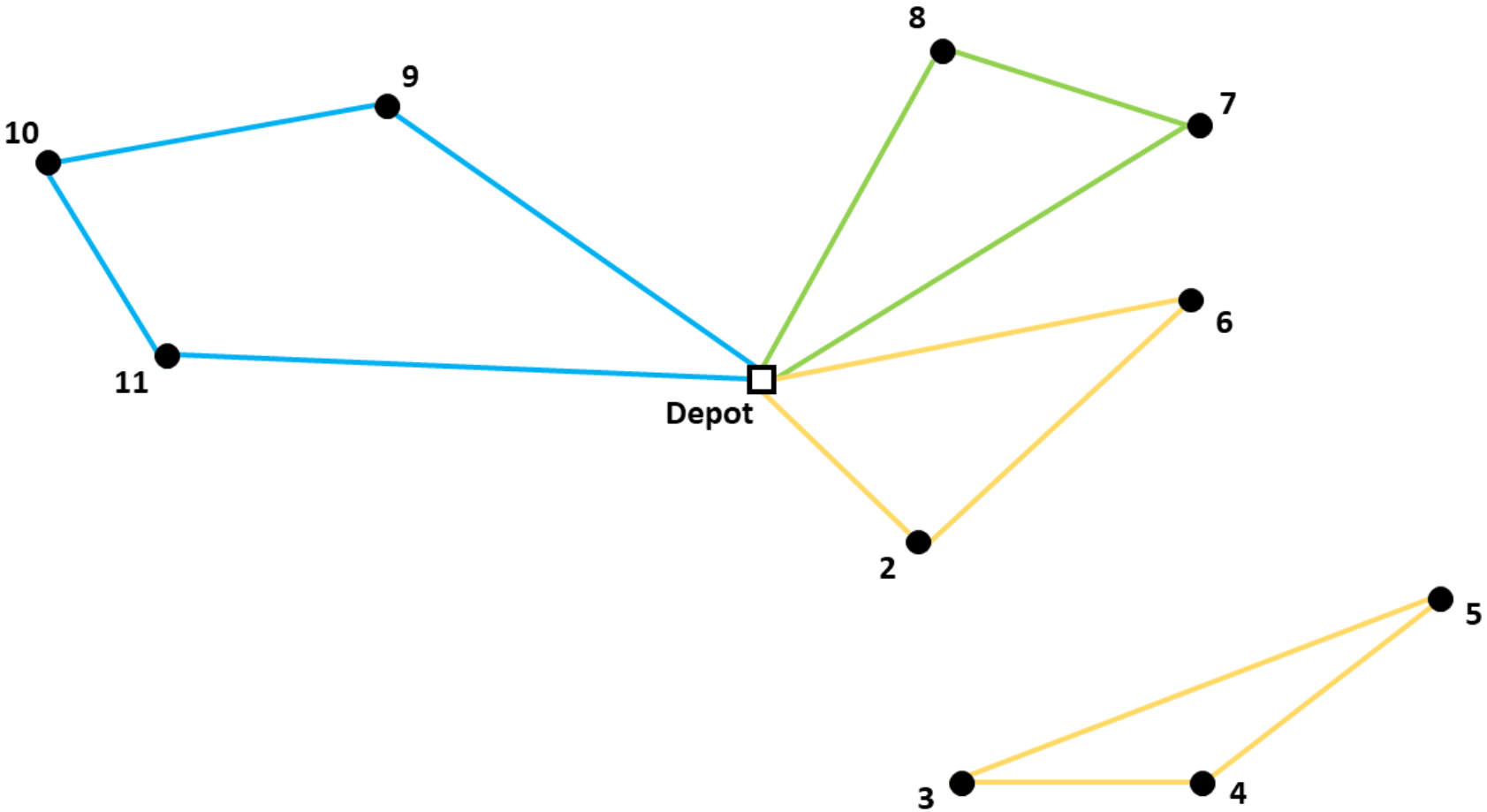
Separation of cuts

An algorithm is needed to take an integer solution of the relaxation of the problem and find all violated RCIs.

- When the current solution is fractional there is no guarantee that all violated cuts are identified.
- If the solution is integer, we can find easily all violated RCIs by looking at the solution and identifying connected subsets that violate the RCIs.

EPFL

Branch-and-cut



Branch-and-cut

Branching is expensive!

Every time we branch in a B&B algorithm, we create 2 new problems. These problems can grow until there are too many and we must stop the algorithm

Branch-and-Cut spend more time in each node

The main idea is to spend more time at each solution of the LR so as to increase the lower bound. If the lower bound is high enough we can prune by bound. Therefore we avoid branching.

Chvátal-Gomory cuts

- Not only can we add SECs but we can add any valid inequality in a Branch-and-cut algorithm.
- Branching creates an exponential number of problems (doubling each time), cuts do not.
- The idea is to use more effort eliminating subproblems before we branch, in the hopes that we can prune by bound.
- When we add cuts to a LR the lower bound increases.

Recall

Valid inequalities are inequalities that do not eliminate feasible integer solutions.

EPFL

Chvátal-Gomory procedure

Let $X = P \cap \mathbb{Z}^n$, where $P = \{x \in \mathbb{R}^n_+ : Ax \leq b\}$, A is an $m \times n$ matrix with columns $\{a_1, a_2, \ldots, a_n\}$ and $v \in \mathbb{R}^m_+$
The inequality is valid:

$$\sum_{j \in N} v a_j x_j \leq vb \qquad \text{This is true since v is positive}$$

Next:

$$\sum_{j \in N} \lfloor v a_j \rfloor x_j \leq vb \qquad \text{Left hand side is less than or equal to } vb$$

$$\sum_{j \in N} \lfloor v a_j \rfloor x_j \leq \lfloor vb \rfloor \qquad \text{The LHS is now integer so the RHS is integer}$$

Outline

VRP Set-partitioning formulation

Master problem (MP)

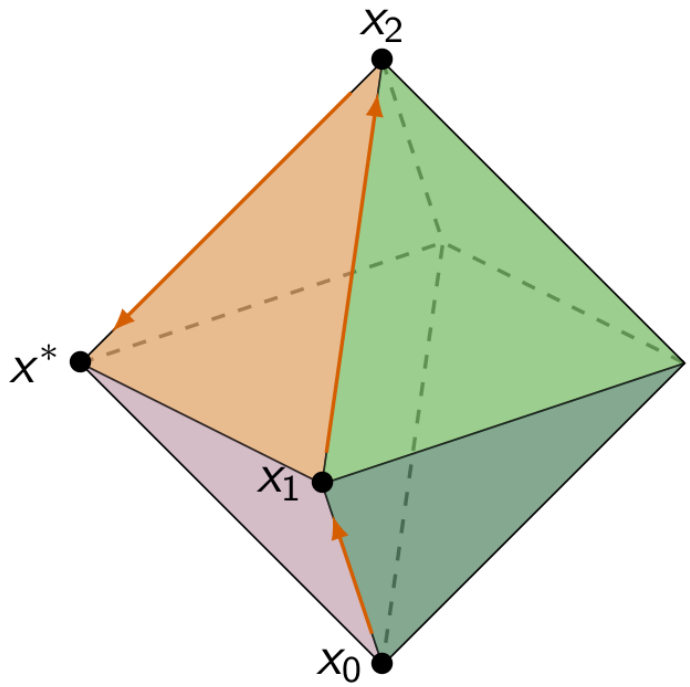
$$\text{Minimize} \sum_{r \in \Omega} c_r \lambda_r \tag{1}$$

$$\text{s.t.} \sum_{r \in \Omega} a_{ir} \lambda_r = 1 \qquad \forall i \in N, \tag{2}$$

$$\lambda_r \in \{0, 1\} \qquad \forall r \in \Omega.$$

EPFL

Simplex Algorithm



Simplex Algorithm

Main ideas

- The current vertex is defined by active constraints.
- The corresponding non basic variables have been set to zero.
- Select one of them, and increase its value.
- It therefore enters the basis.
- Is it worth it? Yes, if the basic direction is descending.
- That is, if the reduced cost is negative.
- If so, follow the basic direction as far as possible. How far?
- Until a constraint is hit, is activated.
- The corresponding variable is set to zero.
- It leaves the basis.

EPFL

Standard form

$$\min_{x \in \mathbb{R}^n} c^T x$$

subject to

$$Ax = b,$$
$$x \geq 0,$$

where

- $A \in \mathbb{R}^{m \times n}$,
- $b \in \mathbb{R}^m$,
- $c \in \mathbb{R}^n$.

Simplex tableau

Definition

$B^{-1}A$	$B^{-1}b$
$c^T - c_B^T B^{-1} A$	$-c_B^T B^{-1} b$

Basic feasible solution $\widetilde{x}$

$B^{-1}A_1$	$\cdots$	$B^{-1}A_n$	$\widetilde{x}_{j_1}$ $\vdots$ $\widetilde{x}_{j_m}$
$\bar{c}_1$	$\cdots$	$\bar{c}_n$	$-c^T\widetilde{x}$

Linear Relaxation Master Problem

$$
\begin{aligned}
\textit{Minimize} \sum_{r \in \Omega} c_r \lambda_r & \\
s.t. \sum_{r \in \Omega} a_{ir} \lambda_r = 1 & \qquad \forall i \in N, \\
\longrightarrow 1 \geq \lambda_r \geq 0 & \qquad \forall r \in \Omega.
\end{aligned}
$$

Linear Relaxation Master Problem

- At each iteration of the Simplex, we have a set of basic variables and a set of non-basic variables.
- Basic variables are $\lambda_{\mathbf{B}} = \{\lambda_{i_1}, \lambda_{i_2}, \ldots, \lambda_{i_{|N|}}\}$. A total of $|N|$, i.e., the number of customers.
- Non-basic variables are $\lambda_{\mathbf{NB}} = \{\lambda_{i_{|N|+1}}, \lambda_{i_{|N|+2}}, \ldots, \lambda_{i_{|\Omega|}}\}$. In total, there are $|\Omega| - |N|$.
- Non-basic variables are always equal to zero at each iteration of the Simplex, i.e., $\lambda_{\mathbf{NB}} = \mathbf{0}$.
- Therefore, to calculate the solution value at each simplex iteration we just need the basic variables, i.e., $\lambda_{\mathbf{B}}$.

EPFL

Restricted Master Problem

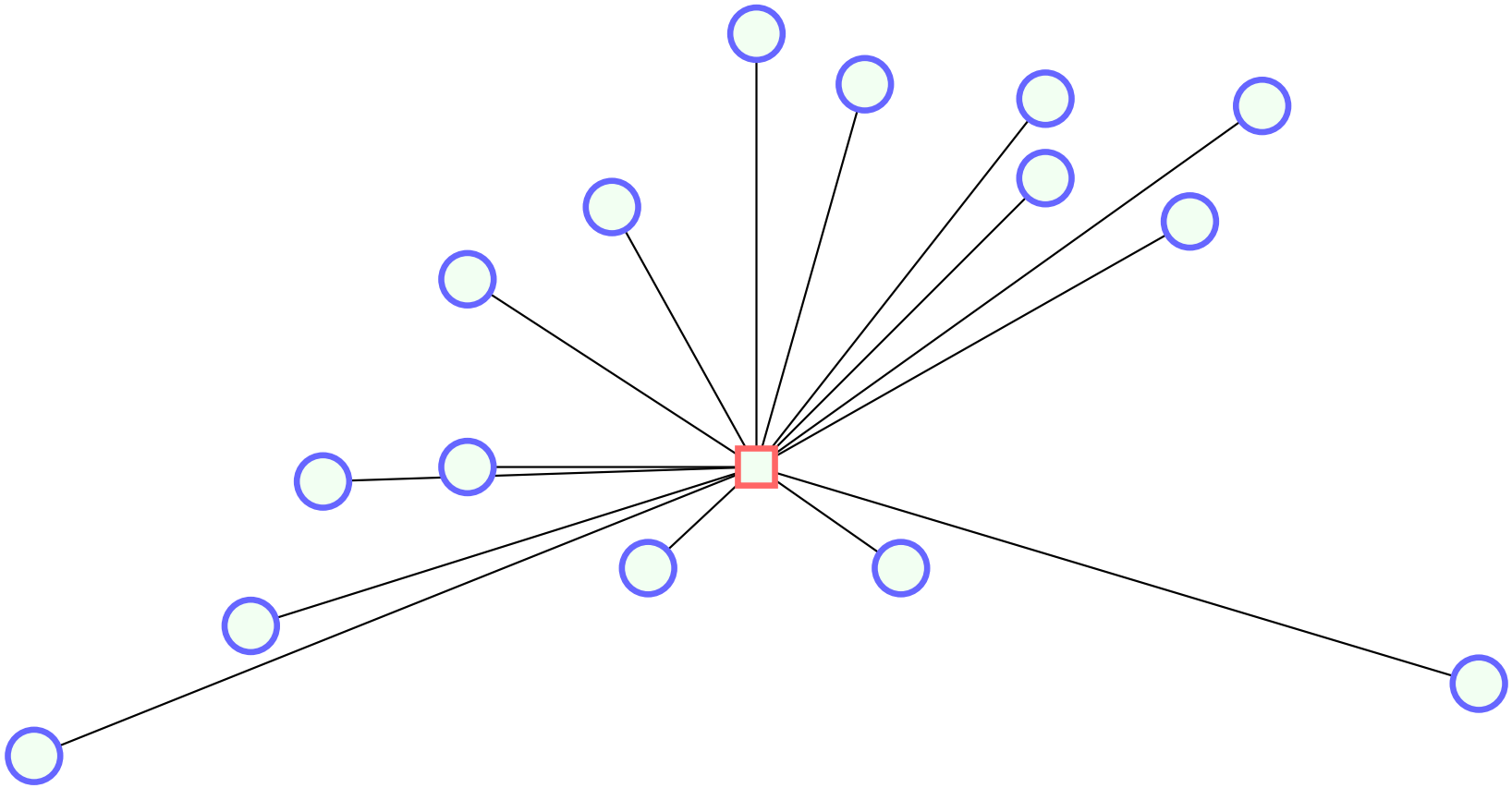
What is the **RMP**?
optimal if reduced cost are non negative

- The set Ω is too big ($\approx (|N|-1)!$).
- Lets consider a smaller set of variables, e.g., $P \subset \Omega$
- Where $|P|$ is a small number, i.e., not exponential.
- The following model is the Restricted Master Problem (RMP):

$$
\begin{aligned}
& & Min \sum_{r \in P} c_r \lambda_r & & \\
\text{(RMP)} & & s.t. \sum_{r \in P} a_{ir} \lambda_r = 1 & & \forall i \in N, \\
& & 1 \geq \lambda_r \geq 0 & & \forall r \in P.
\end{aligned}
$$

What variables should we add to P

Feasible solutions can be difficult to find.



Optimal solution of the MP

When does the **MP = RMP** ?

Optimal if the reduced costs of NB varibles is non negative

- Recall that a basic solution is optimal if there are no non-basic variables (λ_{NB}) with a negative reduced cost.
- We must find non-basic variables (i.e., routes) that have a negative reduced cost to add to the set P and reoptimize.
- If we can proof that all non-basic variables have a non-negative reduced cost, the current basic solution of the RMP is also optimal for the MP and we do not have to search for more variables.

EPFL

Reduced cost

The reduced cost formula is the following:

$$\hat{c}_r = c_r - c_B^T B^{-1} A_r$$

Let $\pi_i = (c_B^T B^{-1})_i$ be the dual variables corresponding to each constraint of the RMP, and let R_r be a list of all customers visited in route $r \in \Omega$. Then the reduced cost can be calculated for each route with the following equation:

$$\hat{c}_r = c_r - \sum_{h \in R_r} \pi_h \qquad \forall r \in \Omega$$

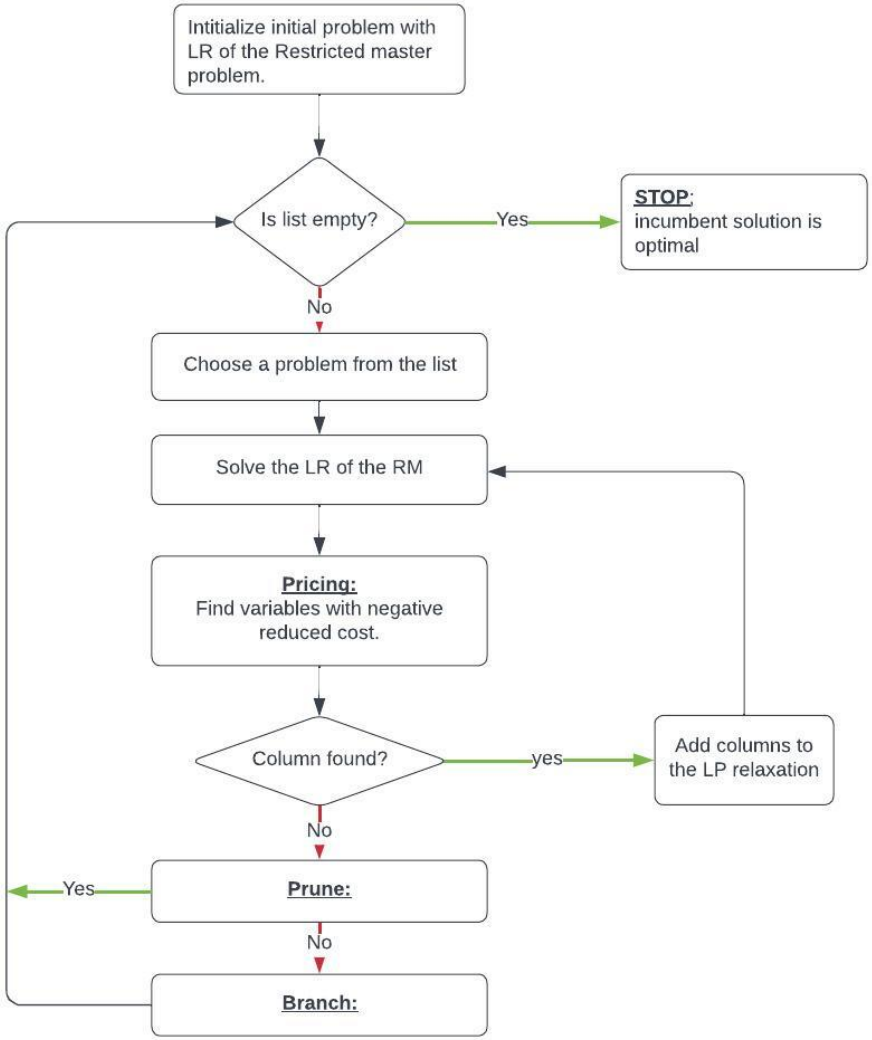
$$\hat{c}_{ij} = c_{ij} - \pi_j \qquad \forall i, j \in V$$

EPFL

Minimum reduced cost route

Elemetary Shortest Path Problem with Resource Constraints (ESPPRC)

$$
\begin{aligned}
&Min \sum_{i \in V} \sum_{j \in V} \hat{c}_{ij} x_{ij} \\
&s.t. \sum_{i \in N} q_i \sum_{j \in V} x_{ij} \leq Q, \\
&\sum_{j \in N} x_{0j} = 1, \\
&\sum_{i \in V} x_{i0} = 1, \\
&\sum_{i \in N} x_{ih} - \sum_{j \in N} x_{hj} = 0 && \forall h \in N, \\
&T_i + t_{ij} - M_{ij}(1 - x_{ij}) \leq T_j && \forall i, j \in V, \\
&a_i \leq T_i \leq b_i && \forall i \in N, \\
&x_{ij} \in \{0, 1\} && \forall i, j \in N.
\end{aligned}
$$


Intitialize initial problem with LR of the Restricted master problem.
Is list empty?
Yes
STOP: incumbent solution is optimal
No
Choose a problem from the list
Solve the LR of the RM
Pricing: Find variables with negative reduced cost.
Column found?
yes
Add columns to the LP relaxation
No
Yes
Prune:
No
Branch:

Feasible solutions

How do we know a node is infeasible if we do not have all variables? Maybe one variable will satisfy the violated constraints!

- In the B&B algorithm, it is possible to encounter nodes in the enumeration tree that are not feasible.
- In B&P we don't have the complete set of variables available.
- How do we know that there does not exist a variable that satisfies the constraints?
- How do we prove that the current node is not feasible if we do not have all variables?

EPFL

Dummy variable

We can add a dummy variable λ_D with a large cost, i.e., $c_D = M$. The dummy variable satisfies all constraints.

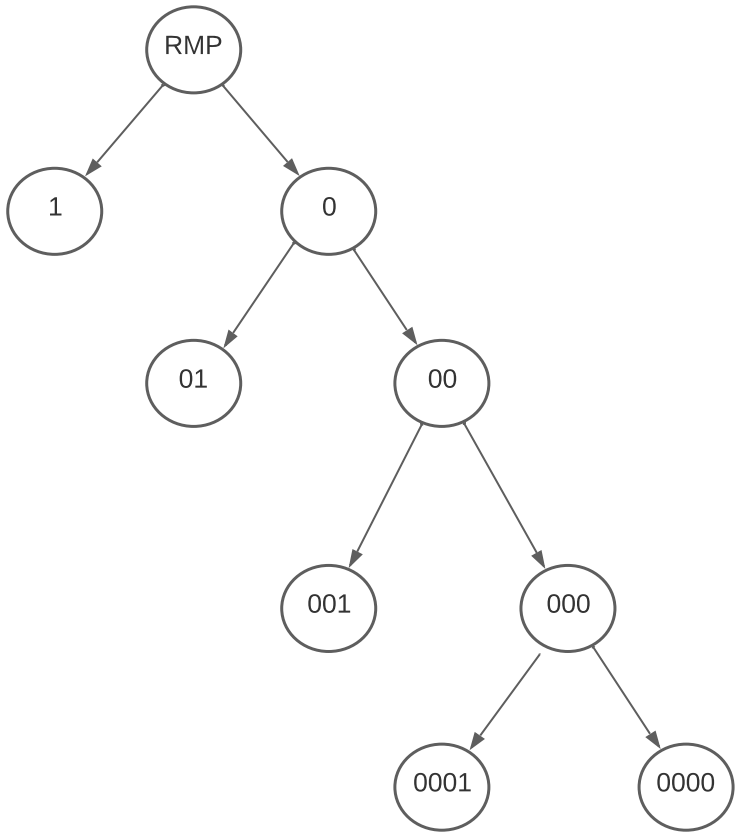
$$\begin{aligned} \min\ & M\lambda_D \\ s.t.\ & \lambda_D = 1 && \forall i \in N, \\ & 1 \geq \lambda_D \geq 0 \end{aligned}$$

- To eliminate the dummy variable we have to solve the pricing problem (i.e., ESPPRC). If a feasible solution exist, routes will be created with a negative reduced cost and added to the RMP.
- If we cannot eliminate the dummy variable, the current problem is infeasible. Thus, we prune the problem and continue with B&P.

EPFL

Branching

Unbalanced binary tree: Branching on variables makes the binary tree unbalanced since most paths are not in the optimal solution. Setting $\lambda = 0$ is not significant.



Pricing problem

ESPPRC and Forbidden Paths.

$$Min \sum_{i \in V} \sum_{j \in V} \hat{c}_{ij} x_{ij}$$

$$s.t. \sum_{i \in N} q_i \sum_{j \in V} x_{ij} \leq Q,$$

$$\sum_{j \in N} x_{0j} = 1,$$

$$\sum_{i \in V} x_{i0} = 1,$$

$$\sum_{i \in N} x_{ih} - \sum_{j \in N} x_{hj} = 0 \qquad \forall h \in N,$$

$$\sum_{(i,j) \in \mathcal{P}} x_{i,j} \leq |\mathcal{P}| - 1 \leftarrow Forbiddenpath \qquad \forall \mathcal{P} \in \mathcal{B},$$

$$T_i + t_{ij} - M_{ij}(1 - x_{ij}) \leq T_j \qquad \forall i, j \in V,$$

$$a_i \leq T_i \leq b_i \qquad \forall i \in N,$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in N.$$

EPFL

Branching

Option

Branch: On x_{ij} variables instead.

- At each node of the branch-and-bound tree set the corresponding variables to 1 or 0, and solve the pricing problem.
- The pricing problem will not produce routes that contain the variables that are set to 0.

Summary

- Branch-and-Bound is a powerful method to solve MILPs. We can reduce the size of the enumeration tree by pruning the tree:
  - Prune by optimality;
  - Prune by infeasibility;
  - Prune by bound.
- Branch-and-Cut algorithms
  - B&C can be used solve problems with a large number of constraints by adding them as needed;
  - Separation procedures are necessary to find violated cuts;
  - Adding valid inequalities in B&B reduces the number of problems.

EPFL

Summary

- Branch-and-Price is an advanced method to solve MILPs with a large number of variables by column generation;
  - The Restricted Master Problem (RMP) has only a few variables;
  - Find variables with a negative reduced by solving a pricing problem and add them to the RMP;
  - If no variable with negative reduced cost is found in the pricing problem, then, the current solution of the RMP is also optimal for the master problem (MP);
  - Once a solution of the MP has been found we continue with B&B, i.e, we prune by bound, feasibility or optimality, or we branch on some fractional variable.

EPFL

Summary

General idea in optimization

When we have a complicating property in a model, create a relaxation without the complicating property and consider it incrementally as needed in the problem (if possible).

EPFL