Branch-and-Cut (Cutting Plane Method)

Compact Formulations

So far, we have only dealt with compact (I)LP formulations

$$\begin{aligned} \max\ & \boldsymbol{c}'\boldsymbol{x} \\ \text{s.t. } & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \text{ (and integer)} \end{aligned}$$

where

- the number of variables n, and
- the number of constraints m

are both polynomial in the input size.

Can we go beyond that?

Cutting Plane Method

Consider the LP

$$\begin{aligned} \max\ & \boldsymbol{c}'\boldsymbol{x} \\ \text{s.t. } & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{aligned}$$

and assume that the number of constraints m is very large (e.g., exponential in the input size).

Can we solve such an LP (in polynomial time)? And if so, how?

$\Rightarrow$ **Cutting Plane Method**

Cutting Plane Method – Basic Idea

- Start with a relaxation of the LP formulation that only contains a small (e.g., polynomially-sized) subset of the original LP's constraints (= inequalities).
- Repeat the following procedure:
  1. Solve the current LP.
  2. Try to find inequalities that are valid for the original LP, but violated by the current solution.
     - if none exist: current solution is optimal for the original LP $\Rightarrow$ stop
     - otherwise: add at least one such inequality to the current LP, then resolve the current LP (step 1)
- It turns out that frequently, only a few of the original LP's constraints need to be added to find an optimal solution.

First proposed by

- G. Dantzig, R. Fulkerson, and S. Johnson, *Solution of a Large-Scale Traveling-Salesman Problem*, Operations Research 2 (1954), 393-410.

Cutting Plane Algorithm

for LP $\max\{\boldsymbol{cx} : \boldsymbol{x} \in X\}$ with $X = \{\boldsymbol{x} : \boldsymbol{Ax} \leq \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\}$

Algorithm 1: Cutting Plane Algorithm

```
t = 0 and P^0 = P // P is the initial formulation
repeat
    solve P^t, let x^t be the obtained optimal solution
    if ∃ valid inequality π^t x ≤ π_0^t for X that is violated by x^t then
        P^{t+1} = P^t ∩ {x : π^t x ≤ π_0^t}
        t = t + 1
until no more violated constraints
```

Some Basic Definitions

Definition 1 (Valid Inequality)

An inequality $\boldsymbol{\pi x} \leq \pi_0$ is a *valid inequality* for $X \subset \mathbb{R}^n$ if $\boldsymbol{\pi x} \leq \pi_0$ for all $\boldsymbol{x} \in X$.

Intuitively, an inequality is valid for some set X if every element $\boldsymbol{x} \in X$ satisfies it.

Definition 2 (Separation Problem)

The *Separation Problem* associated with a set $X \subset \mathbb{R}^n$ is the problem: Given $\boldsymbol{x}^* \in \mathbb{R}^n$, is $\boldsymbol{x}^* \in \mathrm{conv}(X)$? If not, find an inequality $\boldsymbol{\pi x} \leq \pi_0$ satisfied by all points in X (i.e., valid for X), but violated by the point $\boldsymbol{x}^*$.

Such an inequality $\boldsymbol{\pi x} \leq \pi_0$ is also called a cutting plane that separates $\boldsymbol{x}$ from X.

Optimization = Separation

Theorem 3 (Grötschel, Lovasz, Schrijver; 1981)

We can optimize a linear function over a rational polyhedron P in polynomial time if and only if we can solve the separation problem associated with P in polynomial time.

Thus, we can solve an exponentially-sized LP in polynomial time if we can find a polynomial-time separation algorithm for its constraints.

We will later see some exponentially-sized families of constraints for which such efficient separation algorithms exist.

Cutting Plane Method for ILP

We can now solve exponentially-sized LPs. What about ILPs?

Consider the ILP

$$\begin{aligned} \max\ & \boldsymbol{c}'\boldsymbol{x} \\ \text{s.t. } & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \in \mathbb{Z}_+^n \end{aligned}$$

and assume that the number of constraints m is very large (e.g., exponential in the input size).

Can we solve such an ILP quickly (at least for some practically relevant instances, i.e., as well as compact ILPs)? And if so, how?

Ideal Formulations

Consider the previous ILP's feasible set

$$X = \{\boldsymbol{x} : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}, \boldsymbol{x} \in \mathbb{Z}_+^n\}$$

where $\boldsymbol{A}$ and $\boldsymbol{b}$ are rational.

Proposition 4

$\mathrm{conv}(X) = \{\boldsymbol{x} : \tilde{\boldsymbol{A}}\boldsymbol{x} \leq \tilde{\boldsymbol{b}}, \boldsymbol{x} \geq \boldsymbol{0}\}$ *is a polyhedron.*

Thus, every ILP can (in theory) be reformulated as a linear program (the perfect or ideal formulation)

$$\begin{aligned} \max\ & \boldsymbol{c}'\boldsymbol{x} \\ \text{s.t. } & \tilde{\boldsymbol{A}}\boldsymbol{x} \leq \tilde{\boldsymbol{b}} \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{aligned}$$

Note: The same is true for MILPs if defined on a rational polyhedron.

Cutting Plane Method for ILP

Observation: If $X = \{\boldsymbol{x} \in \mathbb{Z}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}$ and $\mathrm{conv}(X) = \{\boldsymbol{x} \in \mathbb{R}^n : \tilde{\boldsymbol{A}}\boldsymbol{x} \leq \tilde{\boldsymbol{b}}\}$, the constraints $\boldsymbol{a^i}\boldsymbol{x} \leq b_i$ and $\tilde{\boldsymbol{a}}^{\boldsymbol{i}}\boldsymbol{x} \leq \tilde{b}_i$ are clearly valid inequalities for X.

In theory, we can solve every ILP with a cutting plane algorithm by finding and solving its ideal formulation. However,

- such formulations are usually hard to find (both in theory and practice), and
- their associated separation problem will be NP-hard for NP-hard optimization problems ("Optimization = Separation").

But if we solve the (exponentially-sized) LP relaxation of the original ILP, we will (in general) get fractional optimal solutions.

$\Rightarrow$ **embed cutting plane method into LP-based branch-and-bound**

Branch-and-Cut

Definition 5 (Branch-and-Cut Algorithm)

A branch-and-cut algorithm is an LP-based branch-and-bound algorithm where the LP relaxation at each branch-and-bound node is solved with a cutting plane algorithm.

This means that, in general, we now have to solve multiple LP relaxations at each branch-and-bound node (one for each cutting plane iteration).

Branch-and-Cut – Motivation

State-of-the-art methods for solving hard combinatorial optimization problems to proven optimality are often based on branch-and-cut!

- Exponentially-sized formulations are often stronger than compact ones.
  - better bounds $\Rightarrow$ earlier pruning, fewer explored nodes
  - often, fewer variables
- Weaker models can be strengthened by adding cutting planes
  $\Rightarrow$ more on this later
- In practice, only relatively few constraints are necessary for finding an optimal solution.
- The alternative – finding and solving an ideal formulation – is usually not tractable for NP-hard optimization problems.

Algorithm 2: Branch-and-Cut

node list $L : \max\{\boldsymbol{cx} : \boldsymbol{x} \in X\}$ with formulation P
$\underline{z} = -\infty$, incumbent $\boldsymbol{x}^* = NULL$
while $L \neq \emptyset$ **do**
  choose and remove node i with formulation P^i for set X^i from L; $k = 1$; $P^{i,1} = P^i$
  repeat
    solve $\overline{z}^{i,k} = \max\{\boldsymbol{cx} : \boldsymbol{x} \in P^{i,k}\}$ // *LP relaxation*
    if *infeasible* **then** prune by infeasibility
    else if $\overline{z}^{i,k} > \underline{z}$ **then**
      try to cut off $\boldsymbol{x}^{i,k}$ // $\boldsymbol{x}^{i,k} \in P^{i,k}$ *(current solution)*
      if *new cuts found* **then** add cuts giving $P^{i,k+1}$, k++
  until *no new cuts found*
  if $\overline{z}^{i,k} > \underline{z}$ **then**
    if $\boldsymbol{x}^{i,k} \in X$ **then** primal bound $\underline{z} = \overline{z}^{i,k}$; incumbent $\boldsymbol{x}^* = \boldsymbol{x}^{i,k}$
    else create new problems X^i_t (formulations P^i_t); add to node list

Notation

In the following for a graph $G = (V, E)$ we use

- $E(S) = \{\{i,j\} \in E \mid i,j \in S\}$, for $S \subseteq V$
- Cutsets
  - $\delta(i) = \{e = \{i,j\} \in E\}$ for node $i \in V$
  - $\delta(S) = \{e = \{i,j\} \in E \mid i \in S, j \in V \setminus S\}$, for $S \subseteq V$
- Directed arc set $A = \{(i,j),(j,i) \mid \{i,j\} \in E\}$
- $A(S) = \{(i,j) \in A \mid i,j \in S\}$, for $S \subseteq V$
- Directed cutsets for $S \subseteq V$

$$\delta^-(S) = \{(i,j) \in A \mid i \in V \setminus S, j \in S\}$$
$$\delta^+(S) = \{(i,j) \in A \mid i \in S, j \in V \setminus S\}$$

(and equivalently $\delta^-(i)$ and $\delta^+(i)$ for $i \in V$)

Example: Acyclic Subgraph Problem

Definition

Given

- a directed graph $D = (V, A)$ and
- arc weights $w_{ij} \in \mathbb{R}^+$, $\forall (i, j) \in A$,

find a subgraph of maximum total weight that does not contain any directed cycles.

Acyclic Subgraph Problem – ILP

1. **Variables:** $x_{ij} \in \{0, 1\}$, $\forall (i, j) \in A$
   $x_{ij} = 1$ if arc (i, j) is included in the subgraph, $x_{ij} = 0$ otherwise
2. **Objective function:** maximize total weight of included arcs

$$\max \sum_{(i,j) \in A} w_{ij} x_{ij}$$

3. **Constraints:**
   - Cycle elimination constraints (CEC)

$$\sum_{(i,j) \in C} x_{ij} \leq |C| - 1 \quad \forall C \subseteq A,\ |C| \geq 2,\ C \text{ forms a cycle}$$

Acyclic Subgraph Problem – Discussion of ILP

$$
\begin{aligned}
\max \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} \\
& \sum_{(i,j)\in C} x_{ij} \leq |C| - 1 && \forall C \subseteq A,\ |C| \geq 2,\ C \text{ forms a cycle} \\
& x_{ij} \in \{0,1\} && \forall (i,j) \in A
\end{aligned}
$$

Main "problem"

- Exponentially many cycles $\Rightarrow$ exponentially many constraints
- We cannot simply apply branch-and-bound using the full model for nontrivial instances
  $\Rightarrow$ start with LP not containing CEC and apply branch-and-cut

Separation Problem for CEC

Given a solution $\bar{\boldsymbol{x}}$, we need to either

- identify a cycle C such that $\sum_{(i,j)\in C} \bar{x}_{ij} > |C| - 1$ or
- prove that no such cycle exists

Reformulating the CEC:

$$
\begin{aligned}
\sum_{(i,j)\in C} \bar{x}_{ij} &\leq |C| - 1 \\
\Leftrightarrow |C| - \sum_{(i,j)\in C} \bar{x}_{ij} &\geq 1 \\
\Leftrightarrow \sum_{(i,j)\in C} (1 - \bar{x}_{ij}) &\geq 1
\end{aligned}
$$

Objective: Find a cycle C that violates

$$
\sum_{(i,j)\in C} (1 - \bar{x}_{ij}) \geq 1
$$

Separation Problem for CEC

To find a cycle C such that

$$\sum_{(i,j)\in C} (1 - \bar{x}_{ij}) < 1$$

we

1. substitute arc weights by $\bar{w}_{ij} = 1 - \bar{x}_{ij}$ for all arcs $(i,j) \in A$ (Note: $\bar{w}_{ij} \geq 0$, $\forall (i,j) \in A$!)
2. search for a cycle C with total costs $\sum_{(i,j)\in C} \bar{w}_{ij} < 1$

$\Rightarrow$ Identify a cheapest cycle in a weighted, directed graph with nonnegative arc costs!

Separation Problem for CEC

- For each arc $(u, v) \in A$:
  1. Compute cheapest path P from v to u in D w.r.t. $\bar{w}$
  2. P together with (u, v) is a cheapest cycle that contains (u, v)

  $\Rightarrow$ If $\sum_{(i,j)\in P} \bar{w}_{ij} + \bar{w}_{uv} < 1$ we have identified a violated cycle elimination inequality!

- If we found violated cycle elimination constraints, we add at least one to the LP, reoptimize, resolve the separation problem, ...
- Otherwise, we successfully solved the associated LP relaxation

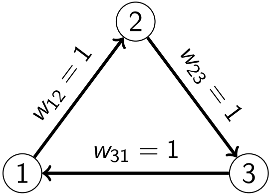
$\Rightarrow$ We can solve the LP relaxation of the acyclic subgraph problem in polynomial time although it contains an exponential number of constraints!

Acyclic Subgraph Problem – Complexity

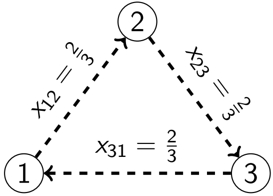
Careful – the acyclic subgraph problem is still NP-hard [Karp, 1971]

- reduction from vertex cover to feedback arc set problem

Our polynomial-time separation procedure only shows that we can solve the (exponentially-sized) LP relaxation of our formulation in polynomial time. But this optimal solution might still be fractional.



instance



optimal LP solution

To solve the acyclic subgraph problem, we still need integrality constraints on the $\mathbf{x}$ variables, and therefore branching.

Branch-and-Cut – Improvements

In our current branch-and-cut algorithm, we always solve the full node LP relaxation with the cutting plane method before

- branching, or
- checking whether the current solution is integral.

Questions:

- Is that necessary?
- Can we somehow exploit the integrality of an LP solution?

Types of Constraints

In an integer linear program, we have two kinds of linear constraints

- "model constraints"
  - required for correctness
  - forgetting about some of these can lead to solutions that are infeasible for the original optimization problem
  - problem-specific
- strengthening inequalities
  - optional
  - redundant for integral solutions, but can strengthen the LP relaxation by cutting off fractional solutions
  - can be problem-specific or general-purpose (more on this in a later lecture)

Branch-and-Cut – Integral Solutions

If a node LP solution $\bar{\boldsymbol{x}}$ is integral (in CPLEX terminology: a "candidate" solution), we must only check whether it satisfies all model constraints. Their separation problem is then often easier to solve than in the fractional case.

E.g., **cycle-elimination constraints**:

- fractional $\bar{\boldsymbol{x}}$: $\mathcal{O}(|E|)$ shortest path computations
- integral $\bar{\boldsymbol{x}}$: topological sort on $G' = (V, A')$, where $A' = \{(i,j) \in A \mid \bar{x}_{ij} = 1\}$
  1. finds directed cycles in selected subgraph or proves that none exist
  2. linear runtime!

Note: Strengthening inequalities are automatically satisfied.

Branch-and-Cut – Fractional Solutions

If a node LP solution $\bar{\boldsymbol{x}}$ is fractional (in CPLEX terminology: a "relaxation" solution), we can choose: do we

- try to find more cutting planes
  - might improve current dual bound $\Rightarrow$ better chance to prune
  - might not find any, or none that improve bound $\Rightarrow$ wasted time
- branch
  - child nodes might be pruned quickly, or be necessary anyways
  - child nodes might be superfluous (if parent would have been pruned with more cutting planes)

$\Rightarrow$ highly problem- and model-specific

- separation may be done heuristically, or not at all (for both model and strengthening constraints)
  - handling of model constraints for integral solutions guarantees that they are all eventually considered

Some (Obvious) Questions

- What kind of valid inequalities may be considered?

$\Rightarrow$ We will see some important examples in the following

- Which are the “good”, “best” or at least “useful” valid inequalities?

$\Rightarrow$ This question will be addressed in one of the next lectures

Definition 6 (Separation Problem for a Family of Inequalities $\mathcal{F}$)

The Separation Problem associated with the family of inequalities $\mathcal{F}$ is the problem: Given $\boldsymbol{x}^* \in \mathbb{R}^n$, find an inequality from $\mathcal{F}$ that is violated by $\boldsymbol{x}^*$ or prove that no such inequality exists.

Examples:

- cycle-elimination constraints
- subtour elimination constraints
- cutset constraints
- ...

Traveling Salesman Problem (TSP)

Definition

Given

- A set of n cities
- Costs c_{ij} for traveling from city i to city j

The problem is to find a tour with minimum traveling costs. Each city has to be visited exactly once and the salesman has to return to his starting city at the end.

We define a complete graph $G = (V, E)$, $V = \{1 \ldots, n\}$, where

- each node corresponds to a city
- travel costs c_{ij} are used as edge costs

and use variables $x_e \in \{0, 1\}$, $\forall e \in E$, indicating whether or not the salesman travels directly from i to j.

Note: undirected model – we'll see directed ones later

Traveling Salesman Problem (LP relaxation)

$$\begin{aligned} \min \ & \sum_{e \in E} c_e x_e \\ \text{s.t. } & x(\delta(v)) = 2 && \forall v \in V \ \textbf{(degree constraints)} \\ & 0 \leq x_e \leq 1 && \forall e \in E \end{aligned}$$

...not yet a valid formulations (subtours!)
Hence, we either add

- Subtour elimination constraints (SEC)

$$x(E(S)) \leq |S| - 1 \quad \forall S \subset V, \ S \neq \emptyset, V$$

- or undirected cutset constraints (CUT)

$$x(\delta(S)) \geq 2 \quad \forall S \subset V, \ S \neq \emptyset, V$$

(both from [Dantzig, Fulkerson and Johnson, 1954])

Separation Problem for CUT

Separation Problem for CUT

Given a point $\bar{\boldsymbol{x}} \in \mathbb{R}^{|E|}$ satisfying $0 \leq \bar{x}_e \leq 1$ for all $e \in E$, find a set $S \subset V$, $1 \leq |S| \leq n-1$, such that $\bar{x}(\delta(S)) < 2$ holds, or prove that no such $S \subset V$ exists.

$\Rightarrow$ We can solve this by finding a minimum cut (i.e., a cut $S \subseteq V$ with minimum cutset capacity $\sum_{e \in \delta(S)} \bar{x}_e$) in graph $G = (V, E)$ with edge capacities $\bar{x}_e$.

1. choose a source vertex $s \in V$ (w.l.o.g., $s = 1$)
2. find a min-cut between s and every $t \in V \setminus \{s\}$
    - Ford-Fulkerson, Push-relabel algorithm, ...
3. select s-t cut with minimum cutset capacity as overall minimum cut S

If S has a cutset capacity $\bar{x}(\delta(S)) < 2$, we found a violated inequality. Otherwise, none exists.

Separation Problem for SEC

Separation Problem for SEC

Given a point $\bar{\boldsymbol{x}} \in \mathbb{R}^{|E|}$ satisfying $0 \leq \bar{x}_e \leq 1$ for all $e \in E$, find a set $S \subset V$, $2 \leq |S| \leq n-1$, such that $\bar{x}(E(S)) > |S| - 1$ holds, or prove that no such $S \subset V$ exists.

Questions:

- How can we solve the separation problem for SEC?
- Which constraints are stronger: SEC or CUT?

Separation Problem for SEC

Theorem 7

Let P_{tspcut} and P_{tspsec} be the polyhedra corresponding to the LP relaxations of the CUT and SEC formulations, then $P_{\text{tspcut}} = P_{\text{tspsec}}$.

Note: This shows that SEC and CUT are equivalent for the TSP!

Proof:

We first observe that due to the degree constraints

$$x(\delta(v)) = 2 \Rightarrow \sum_{v \in V} x(\delta(v)) = 2n \Rightarrow \sum_{e \in E} x_e = n$$

and that

$$E = E(S) \cup \delta(S) \cup E(V \setminus S)$$

for each $S \subset V$.

Separation Problem for SEC

Proof (cont.)
If $\boldsymbol{x} \in P_{\text{tspsec}}$, then

$$\begin{aligned} n = x(E) &= x(E(S)) + x(\delta(S)) + x(E(V \setminus S)) \\ &\leq |S| - 1 + x(\delta(S)) + |V| - |S| - 1 \\ &= n - 2 + x(\delta(S)) \end{aligned}$$

Hence $x(\delta(S)) \geq 2$ and thus $\boldsymbol{x} \in P_{\text{tspcut}}$, i.e. $P_{\text{tspsec}} \subseteq P_{\text{tspcut}}$.
For each $S \subset V$, since $x(\delta(i)) = 2$, $\forall i \in V$:

$$2|S| = \sum_{i \in S} x(\delta(i)) = 2 \cdot x(E(S)) + x(\delta(S))$$

$$\Leftrightarrow x(\delta(S)) = 2|S| - 2 \cdot x(E(S))$$

If $x \in P_{\text{tspcut}}$, then $x(\delta(S)) \geq 2$ and hence

$$2 \leq 2|S| - 2 \cdot x(E(S)) \Leftrightarrow x(E(S)) \leq |S| - 1$$

$\Rightarrow x \in P_{\text{tspsec}}$ and thus $P_{\text{tspcut}} \subseteq P_{\text{tspsec}}$. $\square$

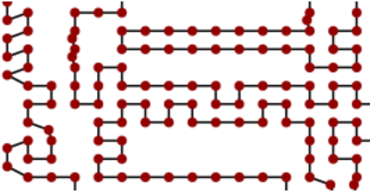
Separation Problem for SEC

- We just showed that the separation problem for SEC can also be solved by solving a min-cut problem
  - violated cutset constraint for $\delta(S) \Rightarrow$ violated subtour elimination constraint for S, $V \setminus S$
- Hence, the LP relaxation of the SEC and CUT formulation can be solved in polynomial time
- In general the associated polytope has fractional extreme points

  $\Rightarrow$ use branch-and-cut
- Several further, strengthening classes of valid inequalities for the TSP are known (e.g., comb inequalities)
  - Often the associated separation problem is NP-hard and
  - separation heuristics are used

Some Remarks for the TSP

- Recommended webpage: `http://www.math.uwaterloo.ca/tsp`
- state-of-the-art TSP solver: Concorde
  - `http://www.math.uwaterloo.ca/tsp/concorde/`
- Until recently, the largest optimally solved instance consisted of 85,900 vertices from a VLSI application.



Recent developments in TSP solving

largest instances with road driving/walking distances

- 2016: 24,727 pubs in the UK
- 2016: 49,603 landmarks in the US
- 2018: 49,687 pubs in the UK
  - 63,739,687 meters, 250 CPU years(!)
  - `http://www.math.uwaterloo.ca/tsp/uk/`

currently largest TSP ever solved: 109,339 stars

- 3D euclidean distances
- 7.5 CPU months
- current work on instance with 2,079,471 stars
  - best known solution: 28,884,456.3 parsec, gap 0.0024%

(Constrained) Spanning and Steiner Trees

We now present various formulations for the minimum spanning tree (MST) problem, because

- Spanning tree problems arise in various contexts, often as subproblems to more complex problems.
- While the MST is solvable in polynomial time, further constraints usually make the problem NP-hard.
- Formulations for the MST problem are well understood and relatively straightforward

(Constrained) Spanning and Steiner Trees

Problem Variants

- Minimum (weight) Spanning Tree Problem
- (Prize Collecting) Steiner Tree Problem ⇒ exercise
- {Delay, Resource, Hop, Diameter, ...} - Constrained Spanning / Steiner Tree Problem
- Minimum Label Spanning Tree Problem
- k-node MST ⇒ programming exercise
- ...

Minimum Spanning Tree Problem (MSTP)

Definition

Given

- An undirected graph $G = (V, E)$, $n = |V|$, $m = |E|$
- Edge weights $w_e \in \mathbb{R}^+$, $\forall e \in E$

The problem is to find a minimum weight subgraph that is a tree and connects all vertices.

Used Variables

- $x_e \in \{0, 1\}$, $\forall e \in E$, indicate if edge e is part of the solution or not
- $y_{ij} \in \{0, 1\}$, $\forall (i, j) \in A$, indicate if arc (i, j) is part of the (directed) solution or not

Exemplary Instance

In the following, we will frequently consider the following instance of the minimum spanning tree problem, with

- $|V| = n = 5$
- $|E| = m = 6$
- Edge weights w_e as given in blue

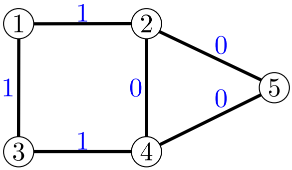


Figure 1: An exemplary instance to the MSTP

Reminder: Flow Formulations (SCF, MCF) for the MSTP

Single- and multi-commodity flows

- Single-commodity flow (SCF)
  flow variables $0 \leq f_{ij} \leq n-1$; amount of flow on arc $(i,j) \in A$ (directed).
- Multi-commodity flow (MCF)
  flow variables $0 \leq f_{ij}^k \leq 1$; amount of flow from 1 to k on arc $(i,j) \in A$ (directed).
- for both formulation we select $1 \in V$ as root node (origin of flow)

Note: SCF and MCF are *compact* formulations, $\Rightarrow$ solved by LP-based branch-and-bound.

SCF formulation for the MSTP

$$
\begin{aligned}
\min \ & \sum_{e \in E} w_e x_e && && \text{(1a)}\\
\text{s.t.} \ & \sum_{(1,j) \in \delta^+(1)} f_{1j} - \sum_{(j,1) \in \delta^-(1)} f_{j1} = n - 1 && && \text{(1b)}\\
& \sum_{(i,j) \in \delta^+(i)} f_{ij} - \sum_{(j,i) \in \delta^-(i)} f_{ji} = -1, && \forall i \in V \setminus \{1\} && \text{(1c)}\\
& f_{ij} \leq (n-1) x_e && \forall e = \{i,j\} \in E && \text{(1d)}\\
& f_{ji} \leq (n-1) x_e && \forall e = \{i,j\} \in E && \text{(1e)}\\
& \sum_{e \in E} x_e = n - 1 && && \text{(1f)}\\
& f_{ij} \geq 0 && \forall (i,j) \in A && \text{(1g)}\\
& x_e \in \{0,1\} && \forall e \in E && \text{(1h)}
\end{aligned}
$$

Note: Directed flows but “undirected formulation” due to (1d) and (1e)!

MCF formulation for the MSTP

$$
\begin{aligned}
\min \quad & \sum_{e \in E} w_e x_e && \text{(2a)} \\
\text{s.t.} \quad & \sum_{(i,j) \in \delta^+(i)} f_{ij}^k - \sum_{(j,i) \in \delta^-(i)} f_{ji}^k = \begin{cases} 1 & \text{if } i = 1 \\ -1 & \text{if } i = k \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in V \setminus \{1\}, \forall i \in V && \text{(2b)} \\
& f_{ij}^k \leq x_e \qquad \forall k \in V \setminus \{1\},\ \forall e = \{i,j\} \in E && \text{(2c)} \\
& f_{ji}^k \leq x_e \qquad \forall k \in V \setminus \{1\},\ \forall e = \{i,j\} \in E && \text{(2d)} \\
& \sum_{e \in E} x_e = n - 1 && \text{(2e)} \\
& f_{ij}^k \geq 0 \qquad \forall k \in V \setminus \{1\},\ \forall (i,j) \in A && \text{(2f)} \\
& x_e \in \{0,1\} \qquad \forall e \in E && \text{(2g)}
\end{aligned}
$$

Note: Directed flows but "undirected formulation" due to (2c) and (2d)!

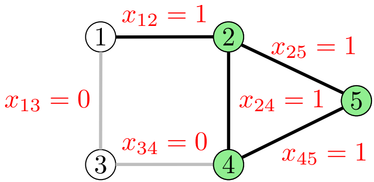
Subtour Elimination Constraint Formulation (SEC)

$$
\begin{aligned}
\min \quad & \sum_{e \in E} w_e x_e & & (3a) \\
\text{s.t.} \quad & \sum_{e \in E(S)} x_e \leq |S| - 1 & \forall S \subseteq V,\ S \neq \emptyset \qquad & (3b) \\
& \sum_{e \in E} x_e = n - 1 & & (3c) \\
& x_e \in \{0, 1\} & \forall e \in E \qquad & (3d)
\end{aligned}
$$

SEC - Formulation: Initial Model (Instance given in Figure 1)

$$z = \min \quad \sum_{e \in E} w_e x_e$$
$$\text{s.t.} \ \sum_{e \in E} x_e = 4$$
$$0 \leq x_e \leq 1$$

Solution with $z^* = 1$

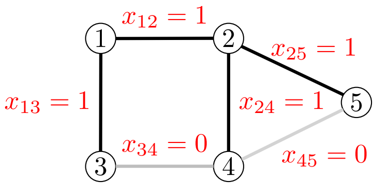


$\Rightarrow$ violated subtour elimination constraint for $S = \{2, 4, 5\}$: $x_{24} + x_{25} + x_{45} > 2$

SEC - Formulation: Current Model

$$\begin{aligned}\min \quad & \sum_{e \in E} w_e x_e \\ \text{s.t.} \quad & \sum_{e \in E} x_e = 4 \\ & x_{24} + x_{25} + x_{45} \leq 2 \\ & 0 \leq x_e \leq 1\end{aligned}$$

Solution with $z^* = 2$



$\Rightarrow$ integral solution, no further violated inequalities. Hence solution is optimal.

Cycle Elimination Constraint Formulation (CEC)

$$
\begin{aligned}
\min \quad & \sum_{e \in E} w_e x_e && \text{(6a)} \\
\text{s.t.} \quad & \sum_{e \in E} x_e = n - 1 && \text{(6b)} \\
& \sum_{e \in C} x_e \leq |C| - 1 \qquad \forall C \subseteq E,\ |C| \geq 2,\ C \text{ forms a cycle} && \text{(6c)} \\
& x_e \in \{0, 1\} && \text{(6d)}
\end{aligned}
$$

Cutset Formulation (CUT)

$$\min \quad \sum_{e \in E} w_e x_e \tag{7a}$$

$$\text{s.t.} \quad \sum_{e \in E} x_e = n - 1 \tag{7b}$$

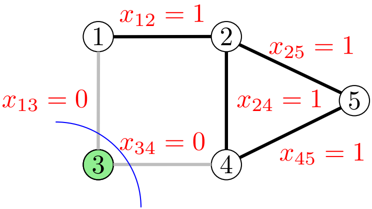
$$\sum_{e \in \delta(S)} x_e \geq 1, \ \forall S \subset V, \ S \neq \emptyset \tag{7c}$$

$$x_e \in \{0, 1\}, \ \forall e \in E \tag{7d}$$

CUT - Formulation: Initial Model (instance given in Figure 1)

$$z = \min \quad \sum_{e \in E} w_e x_e$$
$$\text{s.t.} \quad \sum_{e \in E} x_e = 4$$
$$0 \leq x_e \leq 1$$

Solution with $z^* = 1$

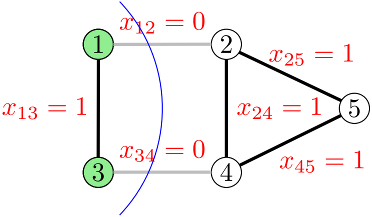


$\Rightarrow$ violated connectivity constraint for $S = \{3\}$: $x_{13} + x_{34} < 1$

CUT - Formulation: Current Model

$$\begin{aligned} \min \quad & \sum_{e \in E} w_e x_e \\ \text{s.t.} \quad & \sum_{e \in E} x_e = 4 \\ & x_{13} + x_{34} \geq 1 \\ & 0 \leq x_e \leq 1 \end{aligned}$$

Solution with $z^* = 1$

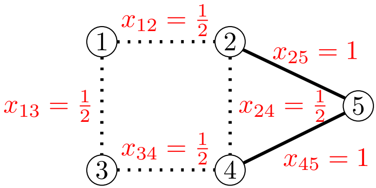


$\Rightarrow$ violated connectivity constraint for $S = \{1, 3\}$: $x_{12} + x_{34} < 1$

CUT - Formulation: Current Model

$$\begin{aligned} \min \quad & \sum_{e \in E} w_e x_e \\ \text{s.t.} \quad & \sum_{e \in E} x_e = 4 \\ & x_{13} + x_{34} \geq 1 \\ & x_{12} + x_{34} \geq 1, \ 0 \leq x_e \leq 1 \end{aligned}$$

Solution with $z^* = \frac{3}{2}$

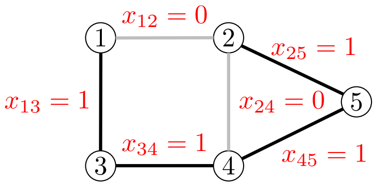


$\Rightarrow$ fractional solution, no further violated connectivity constraints; branch, e.g., on x_{12}

CUT - Formulation: 2nd Node ($x_{12} = 0$)

$$\begin{aligned}
\min \quad & \sum_{e \in E} w_e x_e \\
\text{s.t.} \quad & \sum_{e \in E} x_e = 4 \\
& x_{13} + x_{34} \geq 1 \\
& x_{12} + x_{34} \geq 1 \\
& x_{12} = 0, \; 0 \leq x_e \leq 1
\end{aligned}$$

Solution with $z^* = 2$

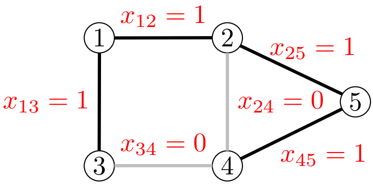


$\Rightarrow$ integral solution, no further violated inequalities; prune by optimality

CUT - Formulation: 3rd Node ($x_{12} = 1$)

$$
\begin{aligned}
\min \quad & \sum_{e \in E} w_e x_e \\
\text{s.t.} \quad & \sum_{e \in E} x_e = 4 \\
& x_{13} + x_{34} \geq 1 \\
& x_{12} + x_{34} \geq 1 \\
& x_{12} = 1, \; 0 \leq x_e \leq 1
\end{aligned}
$$

Solution with $z^* = 2$



$\Rightarrow$ integral solution, no further violated inequalities; prune by optimality

Directed Formulations

We now propose directed formulations for the MSTP by

- considering the set of arcs $A = \{(i,j),(j,i) \mid \{i,j\} \in E\}$
- arc variables $y_{ij} \in \{0,1\}$, $\forall (i,j) \in A$
- modeling feasible solutions as outgoing arborescences with root $r \in V$ (e.g. node 1)

Note:

- We could easily remove undirected edge variables $\boldsymbol{x}$ in the following formulations. They are included to ease the comparison to the previous models.
- Directed formulations are often stronger than their undirected counterparts (This is, however, not always true e.g., directed SEC for TSP).

Directed Subtour Elimination Formulation (DSEC)

$$\min \quad \sum_{e \in E} w_e x_e \tag{13a}$$

$$\text{s.t.} \quad \sum_{(i,j) \in A(S)} y_{ij} \leq |S| - 1 \qquad \forall S \subseteq V,\ S \neq \emptyset \tag{13b}$$

$$\sum_{(i,j) \in \delta^-(j)} y_{ij} = 1 \qquad \forall j \in V \setminus \{r\} \tag{13c}$$

$$\sum_{(i,j) \in A} y_{ij} = n - 1 \tag{13d}$$

$$y_{ij} + y_{ji} = x_e \qquad \forall e = \{i,j\} \in E \tag{13e}$$

$$x_e \geq 0 \qquad \forall e \in E \tag{13f}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{13g}$$

Note: Constraints (13c) imply $\sum_{j \in V \setminus \{r\}} \sum_{(i,j) \in \delta^-(j)} y_{ik} = n - 1$, which together with (13d) implies that $\sum_{(i,r) \in \delta^-(r)} y_{ir} = 0$.

Directed Cut Formulation (DCUT)

$$\min \quad \sum_{e \in E} w_e x_e \tag{14a}$$

$$\text{s.t.} \quad \sum_{(i,j) \in \delta^+(S)} y_{ij} \geq 1 \qquad \forall S \subset V,\ r \in S \tag{14b}$$

$$\sum_{(i,j) \in A} y_{ij} = n - 1 \tag{14c}$$

$$y_{ij} + y_{ji} = x_e \qquad \forall e = \{i,j\} \in E \tag{14d}$$

$$x_e \geq 0 \qquad \forall e \in E \tag{14e}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{14f}$$

Note: Constraints (14b) for $S = V \setminus \{j\}, j \neq r$, imply $\sum_{(i,j) \in \delta^-(j)} y_{ij} \geq 1$. Together with Constraints (14c), this implies $\sum_{(i,j) \in \delta^-(j)} y_{ij} = 1$

Inequalities (14b) are **directed connectivity constraints** (also called **directed cutset constraints**).

Directed Multi-Commodity Flow Formulation (DMCF)

$$\min \sum_{e \in E} w_e x_e \tag{15a}$$

$$\text{s.t.} \sum_{(i,j) \in \delta^+(i)} f_{ij}^k - \sum_{(j,i) \in \delta^-(i)} f_{ji}^k = \begin{cases} 1 & \text{if } i = r \\ -1 & \text{if } i = k \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in V \setminus \{r\}, \forall i \in V \tag{15b}$$

$$f_{ij}^k \leq y_{ij} \qquad \forall k \in V \setminus \{r\},\ \forall (i,j) \in A \tag{15c}$$

$$\sum_{(i,j) \in A} y_{ij} = n - 1 \tag{15d}$$

$$y_{ij} + y_{ji} = x_e \qquad \forall e = \{i,j\} \in E \tag{15e}$$

$$f_{ij}^k \geq 0 \qquad \forall k \in V \setminus \{1\},\ \forall (i,j) \in A \tag{15f}$$

$$x_e \geq 0 \qquad \forall e \in E \tag{15g}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{15h}$$

Final Comments and Remarks

Strategies

- Add “most violated” cut vs. add all cuts (usually more than only one cut added)
- Add cuts “globally”, or just for the current node (and its children) of the branch-and-bound tree

Remark 1: Considering the edges/nodes in a randomized order might improve the overall process.

Remark 2: For many (tree) problems, the directed connection cuts show the best performance in practice (due to their fast separation).