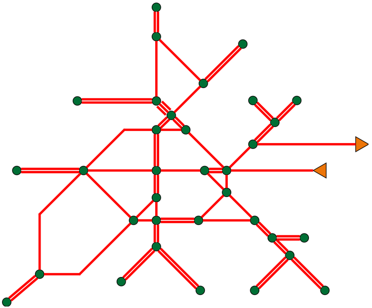
Traffic Optimization:

Optimal Tours in Graphs

Niels Lindner



Lecture 8

December 2, 2019

Chapter 4

Linear and Integer Programming

§4.1 Linear Programming (Recall)

Polyhedra

A (convex) **polyhedron** in $\mathbb{R}^n$ is the set

$$\{x \in \mathbb{R}^n \mid Ax \leq b\}$$

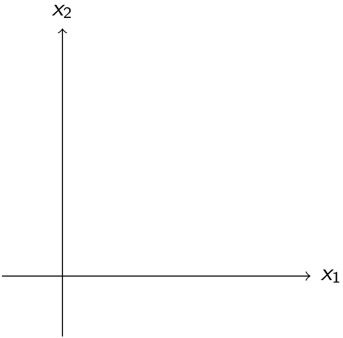
for some matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$. A bounded polyhedron is called a **polytope**.

Polyhedra

A (convex) **polyhedron** in $\mathbb{R}^n$ is the set

$$\{x \in \mathbb{R}^n \mid Ax \leq b\}$$

for some matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$. A bounded polyhedron is called a **polytope**.



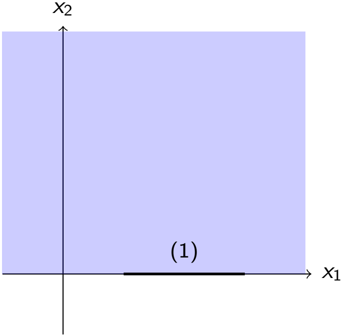
$$\begin{aligned} -x_2 &\leq 0 && (1) \\ -x_1 - x_2 &\leq -1 && (2) \\ -x_1 + x_2 &\leq 3 && (3) \\ x_1 &\leq 3 && (4) \\ x_1 + 2x_2 &\leq 9 && (5) \end{aligned}$$

Polyhedra

A (convex) **polyhedron** in $\mathbb{R}^n$ is the set

$$\{x \in \mathbb{R}^n \mid Ax \leq b\}$$

for some matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$. A bounded polyhedron is called a **polytope**.



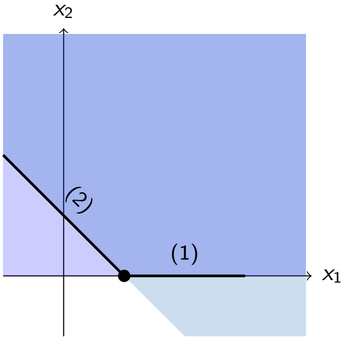
$$\begin{aligned} -x_2 &\leq 0 && (1)\\ -x_1 - x_2 &\leq -1 && (2)\\ -x_1 + x_2 &\leq 3 && (3)\\ x_1 &\leq 3 && (4)\\ x_1 + 2x_2 &\leq 9 && (5) \end{aligned}$$

Polyhedra

A (convex) **polyhedron** in $\mathbb{R}^n$ is the set

$$\{x \in \mathbb{R}^n \mid Ax \le b\}$$

for some matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$. A bounded polyhedron is called a **polytope**.



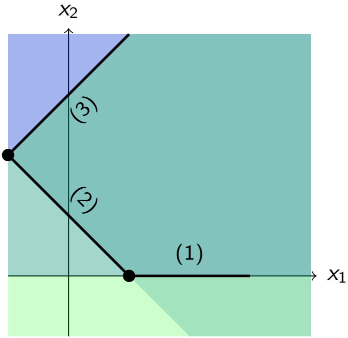
$$
\begin{aligned}
-x_2 &\le 0 && (1)\\
-x_1 - x_2 &\le -1 && (2)\\
-x_1 + x_2 &\le 3 && (3)\\
x_1 &\le 3 && (4)\\
x_1 + 2x_2 &\le 9 && (5)
\end{aligned}
$$

Polyhedra

A (convex) **polyhedron** in $\mathbb{R}^n$ is the set

$$\{x \in \mathbb{R}^n \mid Ax \leq b\}$$

for some matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$. A bounded polyhedron is called a **polytope**.



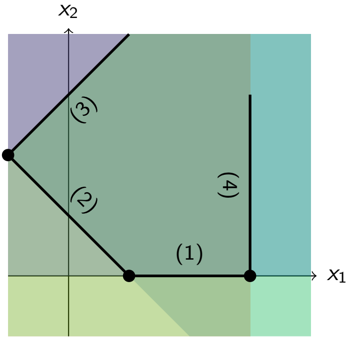
$$
\begin{aligned}
-x_2 &\leq 0 && (1)\\
-x_1 - x_2 &\leq -1 && (2)\\
-x_1 + x_2 &\leq 3 && (3)\\
x_1 &\leq 3 && (4)\\
x_1 + 2x_2 &\leq 9 && (5)
\end{aligned}
$$

Polyhedra

A (convex) **polyhedron** in $\mathbb{R}^n$ is the set

$$\{x \in \mathbb{R}^n \mid Ax \leq b\}$$

for some matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$. A bounded polyhedron is called a **polytope**.



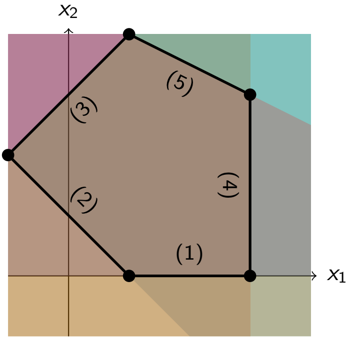
$$
\begin{aligned}
-x_2 &\leq 0 && (1)\\
-x_1 - x_2 &\leq -1 && (2)\\
-x_1 + x_2 &\leq 3 && (3)\\
x_1 &\leq 3 && (4)\\
x_1 + 2x_2 &\leq 9 && (5)
\end{aligned}
$$

Polyhedra

A (convex) **polyhedron** in $\mathbb{R}^n$ is the set

$$\{x \in \mathbb{R}^n \mid Ax \leq b\}$$

for some matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$. A bounded polyhedron is called a **polytope**.



$$
\begin{aligned}
-x_2 &\leq 0 && (1)\\
-x_1 - x_2 &\leq -1 && (2)\\
-x_1 + x_2 &\leq 3 && (3)\\
x_1 &\leq 3 && (4)\\
x_1 + 2x_2 &\leq 9 && (5)
\end{aligned}
$$

Linear programming

Linear programming is the optimization of a linear functional on a polyhedron.

Given a matrix $A \in \mathbb{R}^{m \times n}$, a right-hand side vector $b \in \mathbb{R}^m$ and a cost vector $c \in \mathbb{R}^n$, the task

$$\text{Minimize } c^t x \text{ subject to } Ax \leq b \text{ and } x \in \mathbb{R}^n$$

is called a **linear program (LP)**.

Any point x with $Ax \leq b$ is called a **feasible solution** to the above LP.

Note that since

$$\min\{c^t x \mid Ax \leq b\} = -\max\{-c^t x \mid Ax \leq b\} = \min\{c^t x \mid -Ax \geq -b\},$$

this formulation of linear programs covers maximization, "$\geq$"-inequalities, and equalities as well.

Feasibility and Duality

For a matrix $A \in \mathbb{R}^{m \times n}$ and a right-hand side $b \in \mathbb{R}^m$ as before, the set $\{x \in \mathbb{R}^n \mid Ax \le b\}$ is either

(1) empty $\to$ *infeasible* LP
e.g., $\{x \in \mathbb{R}^n \mid x \le 0, -x \le -1\}$

(2) unbounded $\to$ for certain cost vectors c, $\inf\{c^t x \mid Ax \le b\} = -\infty$

(3) non-empty & bounded $\to$ $\{x \in \mathbb{R}^n \mid Ax \le b\}$ is a non-empty polytope.

Theorem (Linear programming duality)

With A, b, c as before,

$$\min\underbrace{\{c^t x \mid Ax \le b\}}_{\textit{primal LP}} = \max\underbrace{\{b^t y \mid A^t y = c, y \le 0\}}_{\textit{dual LP}}$$

if both LPs are feasible. In particular, if x is a feasible solution to the primal LP and y is feasible for the dual LP, then $c^t x \ge b^t y$. Moreover, the primal LP is infeasible if and only if the dual LP is unbounded.

Vertices of polyhedra

Observe that a subset $X \subseteq \mathbb{R}^n$ is a single point if and only if X is the intersection of n linearly independent affine hyperplanes.

A **vertex** of a polyhedron $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ is a point $x \in P$ such that there are n linearly independent rows $a_{i_1}, \dots, a_{i_n}$ of A with $a_{i_j} x = b_{i_j}$, $j = 1, \dots, n$.

Theorem

Let $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ be a non-empty polytope. Then P has finitely many vertices $x_1, \dots, x_k$, and P is the convex hull of its vertices, i.e.,

$$P = \left\{ \sum_{i=1}^{k} \lambda_i x_i \;\middle|\; \lambda_1 \geq 0, \dots, \lambda_k \geq 0, \lambda_1 + \dots + \lambda_k = 1 \right\}$$

A polytope can hence be described by finitely many affine halfspaces (*H-description*) or by its finitely many vertices (*V-description*). In linear programming, polytopes are always given by their H-description.

Linear programming and vertices

Theorem

Let $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ be a non-empty polytope, and let $c \in \mathbb{R}^n$. Then there is a vertex x^* of P such that

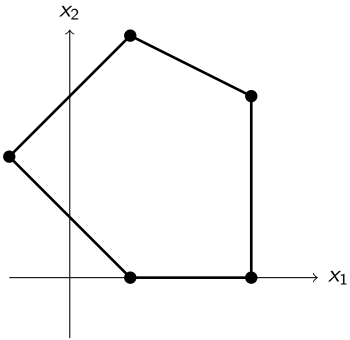
$$c^t x^* = \min\{c^t x \mid x \in P\}.$$

Linear programming and vertices

Theorem

Let $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ be a non-empty polytope, and let $c \in \mathbb{R}^n$. Then there is a vertex x^* of P such that

$$c^t x^* = \min\{c^t x \mid x \in P\}.$$



Minimize $x_1 + 3x_2$ s.t.

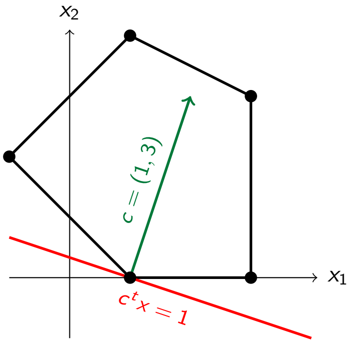
$$\begin{aligned} -x_2 &\leq 0 \\ -x_1 - x_2 &\leq -1 \\ -x_1 + x_2 &\leq 3 \\ x_1 &\leq 3 \\ x_1 + 2x_2 &\leq 9 \end{aligned}$$

Linear programming and vertices

Theorem

Let $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ be a non-empty polytope, and let $c \in \mathbb{R}^n$. Then there is a vertex x^* of P such that

$$c^t x^* = \min\{c^t x \mid x \in P\}.$$



Minimize $x_1 + 3x_2$ s.t.

$$\begin{aligned} -x_2 &\leq 0 \\ -x_1 - x_2 &\leq -1 \\ -x_1 + x_2 &\leq 3 \\ x_1 &\leq 3 \\ x_1 + 2x_2 &\leq 9 \end{aligned}$$

Simplex algorithm

Basic primal simplex algorithm (Dantzig, 1947)

Input: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$ s.t. $P := \{x \mid Ax \leq b\}$ is a polytope
Output: a vertex x^* of P s.t. $c^t x^* = \min\{c^t x \mid x \in P\}$ or "infeasible"

(1) Find any vertex $x \in P$ ($\rightarrow$ (2)) or decide that $P = \emptyset$ ($\rightarrow$ "infeasible").

(2) Let I be the set of indices of n linearly independent rows such that $A_I x = b_I$. Let y be a solution to $A^t y = c$ with $y_i = 0$ for $i \notin I$.

(3) If $y \leq 0$, then

$$c^t x = (A^t y)^t x = y^t A x = y^t b \leq \max\{b^t y \mid A^t y = c, y \leq 0\}$$

$$\stackrel{\text{duality}}{=} \min\{c^t x \mid Ax \leq b\},$$

so x is optimal $\rightarrow$ return $x^* := x$.

(4) If $y \not\leq 0$, then find indices $i \in I$ and $j \notin I$ such that $A_{I'}$ with $I' = I \cup \{j\} \setminus \{i\}$ has full rank, and $c^t x' < c^t x$ for the unique solution x' to $A_{I'} x' = b_{I'}$. Set $x := x' \rightarrow$ go to (2).

Simplex algorithm: Remarks

- Intuitively, the simplex algorithm moves from a vertex of the polytope to an adjacent vertex, strictly improving the objective value.
- Finding an initial vertex resp. detecting infeasibility can be done for an LP of the form $\min\{x \mid Ax \leq b, x \geq 0\}$ with $b \geq 0$ by considering first

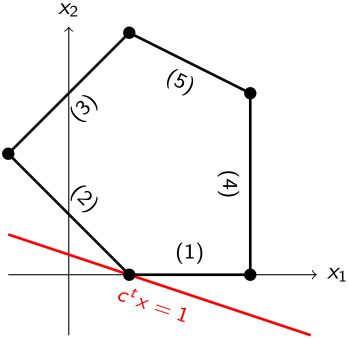
$$\min\{1^t z \mid Ax + z \leq b, x \geq 0, z \geq 0\},$$

  using $(0, b)$ as initial vertex. If the minimum value is 0, then the simplex finds an initial vertex, otherwise the LP is infeasible. A similar strategy works for arbitrary LPs.
- In step (4), there is no need to enumerate all adjacent vertices: There are many clever pivoting rules concerning the selection of the next vertex (e.g., Bland, Dantzig).

Simplex algorithm: Complexity

- On rational polytopes, the worst-case running time of the simplex algorithm is exponential for the most common pivot rules ($\rightarrow$ *Klee-Minty cube*).
- It is an open research question if there is a pivoting rule with polynomial running time ($\rightarrow$ *polynomial Hirsch conjecture*).
- There are polynomial-time algorithms for rational LPs on n variables and b-bit input numbers:
  - *ellipsoid method* (Khachiyan, 1979)
    $\mathcal{O}(n^6 \cdot b)$: not of practical interest
  - *interior point/barrier method* (Karmarkar, 1984)
    $\mathcal{O}(n^{3.5} \cdot b)$: can be fast on large LPs
- In practice, the (dual) simplex is the fastest method to solve LPs.

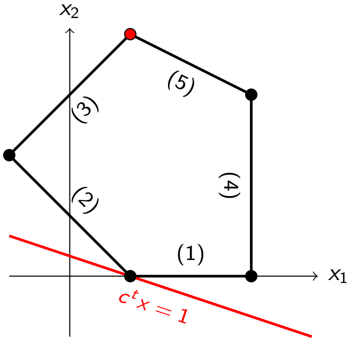
Simple simplex example



Minimize $x_1 + 3x_2$ s.t.

$$
\begin{aligned}
-x_2 &\leq 0 & (1)\\
-x_1 - x_2 &\leq -1 & (2)\\
-x_1 + x_2 &\leq 3 & (3)\\
x_1 &\leq 3 & (4)\\
x_1 + 2x_2 &\leq 9 & (5)
\end{aligned}
$$

Simple simplex example



Minimize $x_1 + 3x_2$ s.t.

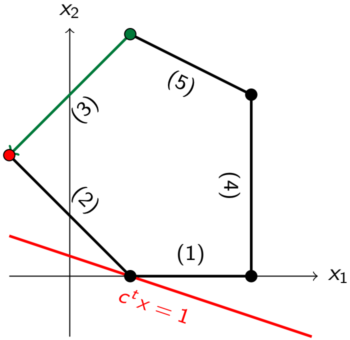
$$\begin{aligned} -x_2 &\leq 0 && (1)\\ -x_1 - x_2 &\leq -1 && (2)\\ -x_1 + x_2 &\leq 3 && (3)\\ x_1 &\leq 3 && (4)\\ x_1 + 2x_2 &\leq 9 && (5) \end{aligned}$$

Initial vertex $(1,4)$:

Rows: $-x_1 + x_2 = 3$ and $x_1 + 2x_2 = 9 \rightarrow \begin{pmatrix} -1 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 4 \end{pmatrix} = \begin{pmatrix} 3 \\ 9 \end{pmatrix}$

Solve $\begin{pmatrix} -1 & 1 \\ 1 & 2 \end{pmatrix} y = \begin{pmatrix} 1 \\ 3 \end{pmatrix} \rightarrow y = \begin{pmatrix} 1/3 \\ 4/3 \end{pmatrix} > 0 \rightarrow$ not optimal.

Simple simplex example



Minimize $x_1 + 3x_2$ s.t.

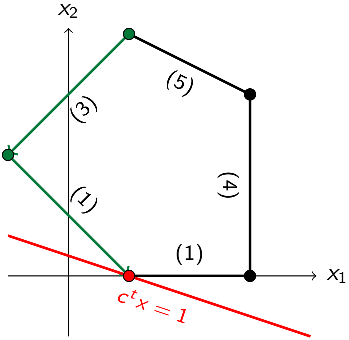
$$
\begin{aligned}
-x_2 &\leq 0 && (1)\\
-x_1 - x_2 &\leq -1 && (2)\\
-x_1 + x_2 &\leq 3 && (3)\\
x_1 &\leq 3 && (4)\\
x_1 + 2x_2 &\leq 9 && (5)
\end{aligned}
$$

Vertex $(-1, 2)$:

Rows: $-x_1 + x_2 = 3$ and $-x_1 - x_2 \leq -1 \rightarrow \begin{pmatrix} -1 & 1 \\ -1 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \end{pmatrix} = \begin{pmatrix} 3 \\ -1 \end{pmatrix}$

Solve $\begin{pmatrix} -1 & -1 \\ 1 & -1 \end{pmatrix} y = \begin{pmatrix} 1 \\ 3 \end{pmatrix} \rightarrow y = \begin{pmatrix} 1 \\ -2 \end{pmatrix} \nleq 0 \rightarrow$ not optimal.

Simple simplex example



Minimize $x_1 + 3x_2$ s.t.

$$-x_2 \le 0 \qquad (1)$$
$$-x_1 - x_2 \le -1 \qquad (2)$$
$$-x_1 + x_2 \le 3 \qquad (3)$$
$$x_1 \le 3 \qquad (4)$$
$$x_1 + 2x_2 \le 9 \qquad (5)$$

Vertex $(1, 0)$:

Rows: $-x_2 \le 0$ and $-x_1 - x_2 \le -1 \rightarrow \begin{pmatrix} 0 & -1 \\ -1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix}$

Solve $\begin{pmatrix} 0 & -1 \\ -1 & -1 \end{pmatrix} y = \begin{pmatrix} 1 \\ 3 \end{pmatrix} \rightarrow y = \begin{pmatrix} -2 \\ -1 \end{pmatrix} \le 0 \rightarrow$ **optimal!**

Chapter 4

Linear and Integer Programming

§4.2 Integer Programming (Recall)

Integer programming

Given a matrix $A \in \mathbb{R}^{m \times n}$, a right-hand side vector $b \in \mathbb{R}^m$ and a cost vector $c \in \mathbb{R}^m$, the task

$$\text{Minimize } c^t x \text{ subject to } Ax \leq b \text{ and } x \in \mathbb{Z}^n$$

is called an **integer program (IP)** or **integer linear program (ILP)**.

If not all entries of x are required to be integer, one speaks of a **mixed integer program (MIP)** or **mixed integer linear program (MILP)**.

Each integer program has a **natural LP relaxation** by replacing $x \in \mathbb{Z}^n$ with $x \in \mathbb{R}^n$.

Feasibility and Duality

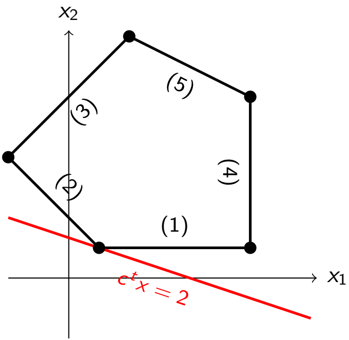
Again, IPs can be infeasible or unbounded. However, only *weak duality* is known:

Theorem (Weak duality)

Let A, b, c as before. Then, if all programs are feasible,

$$\begin{aligned}\underbrace{\min\{c^t x \mid Ax \le b, x \in \mathbb{Z}^n\}}_{\textit{primal IP}} &\ge \underbrace{\min\{c^t x \mid Ax \le b, x \in \mathbb{R}^n\}}_{\textit{primal LP relaxation}} \\ &= \underbrace{\max\{b^t y \mid A^t y = c, y \le 0, y \in \mathbb{R}^n\}}_{\textit{dual LP relaxation}} \\ &\ge \underbrace{\max\{b^t y \mid A^t y = c, y \le 0, y \in \mathbb{Z}^n\}}_{\textit{"dual" IP}}.\end{aligned}$$

IP example

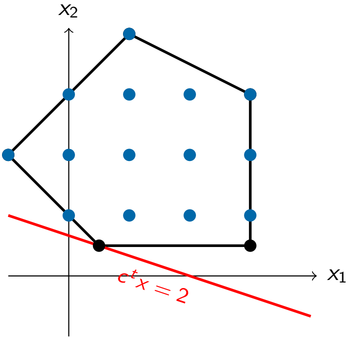


Minimize $x_1 + 3x_2$ s.t.

$$
\begin{aligned}
-2x_2 &\le -1 && (1)\\
-x_1 - x_2 &\le -1 && (2)\\
-x_1 + x_2 &\le 3 && (3)\\
x_1 &\le 3 && (4)\\
x_1 + 2x_2 &\le 9 && (5)
\end{aligned}
$$

LP relaxation: optimal solution $(0.5, 0.5)$ with objective value 2

IP example



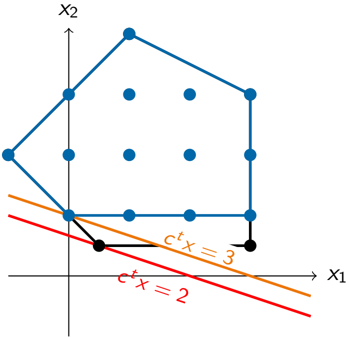
Minimize $x_1 + 3x_2$ s.t.

$$-2x_2 \leq -1 \qquad (1)$$
$$-x_1 - x_2 \leq -1 \qquad (2)$$
$$-x_1 + x_2 \leq 3 \qquad (3)$$
$$x_1 \leq 3 \qquad (4)$$
$$x_1 + 2x_2 \leq 9 \qquad (5)$$

and $x_1, x_2 \in \mathbb{Z}$**!**

LP relaxation: optimal solution $(0.5, 0.5)$ with objective value 2

IP example



Minimize $x_1 + 3x_2$ s.t.

$$-2x_2 \leq -1 \qquad (1)$$

$$-x_1 - x_2 \leq -1 \qquad (2)$$

$$-x_1 + x_2 \leq 3 \qquad (3)$$

$$x_1 \leq 3 \qquad (4)$$

$$x_1 + 2x_2 \leq 9 \qquad (5)$$

and $x_1, x_2 \in \mathbb{Z}$**!**

LP relaxation: optimal solution $(0.5, 0.5)$ with objective value 2
Integer program: optimal solution $(0, 1)$ with objective value 3

IP, polytopes, and *NP*-completeness

Lemma

Let $P \subseteq \mathbb{R}^n$ be a polytope and let Q be the convex hull of $P \cap \mathbb{Z}^n$. Then

(1) *Q is a polytope contained in P.*

(2) *$P \cap \mathbb{Z}^n$ is empty if and only if Q is empty.*

(3) *For any $c \in \mathbb{R}^n$, $\min\{c^t x \mid x \in P \cap \mathbb{Z}^n\} = \min\{c^t x \mid x \in Q\}$.*

In particular, any integer program is a linear program. However, computing an H-description of Q from an H-description of P is not doable in polynomial-time unless $P = NP$:

Theorem (NP-completeness of integer programming, Karp 1972)

Given a rational polytope P, deciding if $P \cap \mathbb{Z}^n \neq \emptyset$ is NP-complete.

Corollary

Any optimization problem (whose decision version is) in NP has a formulation as integer program with polynomially many variables and constraints.

Integrality gaps

Let $P \subseteq \mathbb{R}^n$ be a polytope and let $c \in \mathbb{R}^n$ be a cost vector. The quotient

$$\frac{\min\{c^t x \mid x \in P \cap \mathbb{Z}^n\}}{\min\{c^t x \mid x \in P\}} \geq 1$$

is called the **integrality gap** of the LP relaxation.

When the integrality gap is bounded by k for a set of polytopes, solving the LP relaxation sometimes produces a k-factor approximation algorithm:

- shortest s-t-path in a directed graph: $k = 1$
  (*total unimodularity* of the incidence matrix, Hoffman-Kruskal thm.)
- minimum vertex cover in a graph: $k = 2$
  (LP relaxation has *half-integral* vertices)
- MaxSAT: $k = 1 - 1/e$
  (*derandomization* of *randomized rounding*)

Unfortunately, integrality gaps are often unbounded.

Chapter 4

Linear and Integer Programming

§4.3 Cutting Planes

Solving IPs

There is no polynomial-time algorithm to solve integer programs unless $P = NP$. What are good alternatives to enumeration of all integer points? How can we exploit that linear programs are comparably easy to solve?

First approach: **Cutting plane algorithms**.

Definition

Let P be a polytope in $\mathbb{R}^n$.

(1) A **valid inequality** for P is a linear inequality $\alpha^t x \leq \beta$ for some $\alpha \in \mathbb{R}^n$ and $\beta \in \mathbb{R}$ such that $P \subseteq \{x \in \mathbb{R}^n \mid \alpha^t x \leq \beta\}$.

(2) If $x^* \notin P$, a **cutting plane** or **cut** separating x^* from P is a valid inequality $\alpha^t x \leq \beta$ for P such that $\alpha^t x^* > \beta$.

(3) Finding a cutting plane as in (2) is called the **separation problem**.

Note that "cut" in the sense of "cutting plane" is different to "cut" in the context of network flows.

Since P is convex and closed, cutting planes always exist.

Cutting plane algorithm

Basic cutting plane algorithm (Gomory, 1958)

Input: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$ s.t. $P := \{x \mid Ax \leq b\}$ is a polytope
Output: a vertex x^* of the convex hull Q of $P \cap \mathbb{Z}^n$ s.t.
$c^t x^* = \min\{c^t x \mid x \in P \cap \mathbb{Z}^n\}$ or "infeasible"

(1) Solve the LP relaxation $\min\{c^t x \mid Ax \leq b\}$. If infeasible, then return "infeasible", otherwise let x^* be an optimal vertex.

(2) If x^* is integral, return x^*.

(3) Find a cutting plane $\alpha^t x \leq \beta$ separating x^* from Q and append it to $Ax \leq b$. Go to (1).

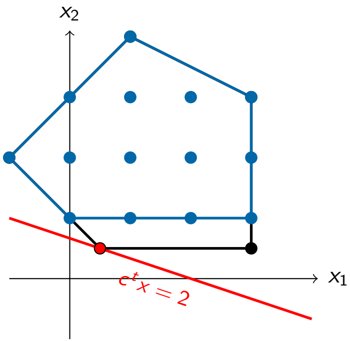
Lemma

If the basic cutting plane algorithm terminates, it is correct.

Proof.

Idea: If P' is any polytope considered in step (1), then $Q \subseteq P' \subseteq P$ and $Q \cap \mathbb{Z}^n = P' \cap \mathbb{Z}^n = P \cap \mathbb{Z}^n$. □

Simple cutting plane algorithm example

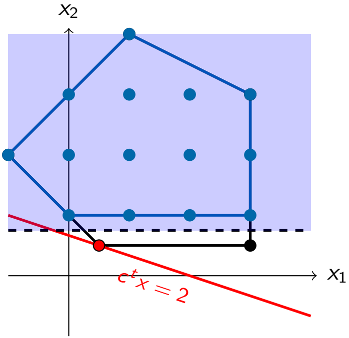


Minimize $x_1 + 3x_2$ s.t.

$$
\begin{aligned}
-2x_2 &\le -1 \\
-x_1 - x_2 &\le -1 \\
-x_1 + x_2 &\le 3 \\
x_1 &\le 3 \\
x_1 + 2x_2 &\le 9 \\
x_1, x_2 &\in \mathbb{Z}
\end{aligned}
$$

LP relaxation #1: $x^* = (0.5, 0.5), c^t x^* = 2$

Simple cutting plane algorithm example

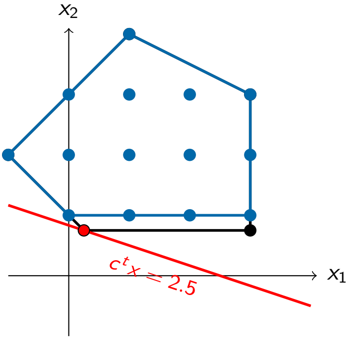


Minimize $x_1 + 3x_2$ s.t.

$$\begin{aligned} -2x_2 &\le -1 \\ -x_1 - x_2 &\le -1 \\ -x_1 + x_2 &\le 3 \\ x_1 &\le 3 \\ x_1 + 2x_2 &\le 9 \\ x_1, x_2 &\in \mathbb{Z} \end{aligned}$$

LP relaxation #1: $x^* = (0.5, 0.5), c^t x^* = 2$, cutting plane: $x_2 \geq 0.75$

Simple cutting plane algorithm example

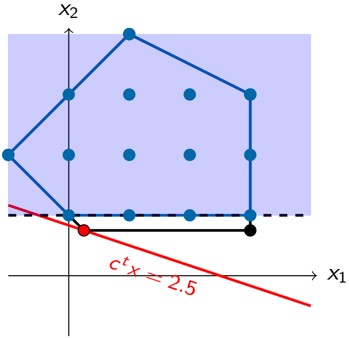


Minimize $x_1 + 3x_2$ s.t.

$$\begin{aligned} -2x_2 &\leq -1 \\ -x_1 - x_2 &\leq -1 \\ -x_1 + x_2 &\leq 3 \\ x_1 &\leq 3 \\ x_1 + 2x_2 &\leq 9 \\ x_1, x_2 &\in \mathbb{Z} \end{aligned}$$

LP relaxation #1: $x^* = (0.5, 0.5), c^t x^* = 2$, cutting plane: $x_2 \geq 0.75$
LP relaxation #2: $x^* = (0.25, 0.75), c^t x^* = 2.5$

Simple cutting plane algorithm example

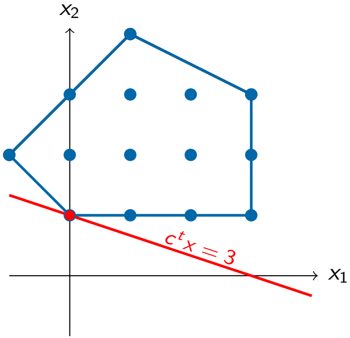


Minimize $x_1 + 3x_2$ s.t.

$$\begin{aligned} -2x_2 &\le -1 \\ -x_1 - x_2 &\le -1 \\ -x_1 + x_2 &\le 3 \\ x_1 &\le 3 \\ x_1 + 2x_2 &\le 9 \\ x_1, x_2 &\in \mathbb{Z} \end{aligned}$$

LP relaxation #1: $x^* = (0.5, 0.5), c^t x^* = 2$, cutting plane: $x_2 \ge 0.75$
LP relaxation #2: $x^* = (0.25, 0.75), c^t x^* = 2.5$, cutting plane: $x_2 \ge 1$

Simple cutting plane algorithm example



Minimize $x_1 + 3x_2$ s.t.

$$\begin{aligned} -2x_2 &\le -1 \\ -x_1 - x_2 &\le -1 \\ -x_1 + x_2 &\le 3 \\ x_1 &\le 3 \\ x_1 + 2x_2 &\le 9 \\ x_1, x_2 &\in \mathbb{Z} \end{aligned}$$

LP relaxation #1: $x^* = (0.5, 0.5), c^t x^* = 2$, cutting plane: $x_2 \ge 0.75$
LP relaxation #2: $x^* = (0.25, 0.75), c^t x^* = 2.5$, cutting plane: $x_2 \ge 1$
LP relaxation #3: $x^* = (0, 1), c^t x^* = 3 \rightarrow$ integral $\rightarrow$ optimal.

Termination: Gomory-Chvátal truncations

Definition

For a rational polyhedron $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$, define its **Gomory-Chvátal truncation** P' as

$$P' := \{x \in \mathbb{R}^n \mid y^t Ax \leq \lfloor y^t b \rfloor \text{ for all } y \geq 0 \text{ with } y^t A \text{ integral}\}.$$

Lemma

$P \cap \mathbb{Z}^n \subseteq P' \subseteq P$.

Theorem (Gomory, Giles, Pulleyblank, Schrijver)

The Gomory-Chvátal truncation of a rational polyhedron P is a polyhedron whose H-description can be determined in exponential time from an H-description of P (e.g., by Gomory cuts).

Theorem (Chvátal, 1973, Schrijver, 1980)

For any rational polyhedron, there is an $r \in \mathbb{N}_0$ (Chvátal rank) s.t. the convex hull of $P \cap \mathbb{Z}^n$ equals the r-th Gomory-Chvátal truncation $P^{\underbrace{\prime\prime\cdots\prime}_{r}}$.

Termination and Remarks

Corollary

For every rational polytope P, cutting planes can be chosen in such a way that the cutting plane algorithm terminates after a finite number of steps.

Remarks

- It is often much more reasonable to use other problem-specific cuts than the generic Gomory cuts.
- The best cutting planes are *facet-defining inequalities*, i.e., the inequalities defining the inclusion-maximal faces of $\mathrm{conv}(P \cap \mathbb{Z}^n)$.
- Today, pure cutting plane approaches are rarely used to solve IPs, due to running time and numerical issues.
- Each intermediate LP value $c^t x^*$ is a lower bound on the minimum value of the IP, and this lower bound does not decrease during the course of the algorithm.
- It is advantageous to use the dual simplex, because adding an extra primal constraint (dual variable) preserves dual feasibility (*warmstart*).

Separation vs. Optimization

Theorem (Grötschel, Lovasz, Schrijver, 1981, 1988)

Let $P \in \mathbb{R}^n$ be a well-described rational polyhedron. Then the following problems are polynomially equivalent:

(1) *The optimization problem*
"Given $c \in \mathbb{Q}^n$, find $x^* \in P$ s.t. $c^t x^* = \min\{c^t x \mid x \in P\}$."

(2) *The separation problem*
"Given $x^* \in \mathbb{Q}^n$, decide if $x^* \in P$ or find $\alpha \in \mathbb{Q}^n$ s.t. $\alpha^t x < \alpha^t x^*$ for all $x \in P$."

This means that there is a polynomial-time optimization algorithm if and only if there is a polynomial-time algorithm for cutting planes. In particular, unless $P = NP$, there is no polynomial-time algorithm solving the separation problem for the convex hull of the integer points of an arbitrary polytope.

Back to TSP

Cutting planes were first developed in the context of TSP by Dantzig, Fulkerson and Johnson (1954).

Question: How can TSP be formulated as an integer program?

Let (K_n, c) be a TSP instance. Encode a Hamiltonian circuit C as follows: Introduce a binary variable $x_e \in \{0,1\}$ for each edge $e \in E(K_n)$ with the interpretation that

$$x_e = \begin{cases} 1 & \text{iff } e \in E(C), \\ 0 & \text{iff } e \notin E(C). \end{cases}$$

I.e., $x \in \{0,1\}^{E(K_n)}$ is the *incidence vector* of C.
Clearly, each vertex has to be incident to exactly two edges of C:

$$\sum_{e \in \delta(v)} x_e = 2, \quad v \in V(K_n),$$

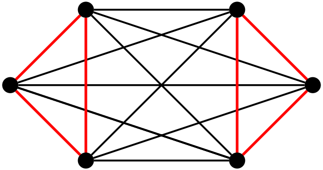
where $\delta(v)$ is the set of edges incident to v.

TSP IP formulation – first idea

$$
\begin{aligned}
\text{Minimize} \quad & \sum_{e \in E(K_n)} c_e x_e \\
\text{s.t.} \quad & \sum_{e \in \delta(v)} x_e = 2, && v \in V(K_n), \\
& x_e \in \{0, 1\}, && e \in E(K_n).
\end{aligned}
$$

Problem

Not every feasible solution to the IP is an incidence vector of a Hamiltonian circuit – there might be *subtours*.



Subtour elimination constraints

How can subtours be excluded from the IP?

Lemma

A vector $x \in \{0,1\}^{E(K_n)}$ *is an incidence vector of a Hamiltonian circuit in* K_n *if and only if*

$$\sum_{e \in \delta(v)} x_e = 2, \qquad v \in V(K_n), \tag{1}$$

$$\sum_{e \in E(K_n[X])} x_e \leq |X| - 1, \qquad \emptyset \subsetneq X \subsetneq V(K_n), \tag{2}$$

where $E(K_n[X]) := \{\{v,w\} \in E(K_n) \mid v \in X, w \in X\}$.

Proof.

Let x be a feasible solution to the IP. By (1), x is the incidence vector of a union of vertex-disjoint circuits covering all vertices of K_n. If X is the vertex set of such a circuit C, then C has $|X|$ edges. But by (2), the circuit has less than $|X| - 1$ edges unless $X = V(K_n)$, i.e., C is Hamiltonian.

Subtour elimination constraints

Proof (cont.)

Conversely, let x be an incidence vector of a Hamiltonian circuit C. Then (1) is clearly satisfied. Let X be a proper subset of $V(K_n)$. Then the restriction of C to the subgraph $(X, E(K_n[X]))$ is a union of vertex-disjoint paths, and hence has at most $|X| - 1$ edges, so (2) is fulfilled. □

Remarks

- We obtain an IP formulation for TSP with $n(n-1)/2$ binary variables and $n + 2^n - 2$ constraints.
- In fact, since every circuit in a subtour has ≥ 3 vertices, it suffices to consider $X \subseteq V(K_n)$ with $3 \leq |X| \leq n - 3$.
- For large n, it is hard to write down all constraints explicitly.
- Idea: Integrate the subtour elimination constraints into the cutting plane approach!

TSP IP formulation – working

The following is a valid IP formulation for the TSP:

$$
\begin{array}{llll}
\text{(TSP-SEC)} & \text{Minimize} & \displaystyle\sum_{e \in E(K_n)} c_e x_e & \\
 & \text{s.t.} & \displaystyle\sum_{e \in \delta(v)} x_e = 2, & v \in V(K_n), \\
 & & \displaystyle\sum_{e \in E(K_n[X])} x_e \leq |X| - 1, & \emptyset \subsetneq X \subsetneq V(K_n), \\
 & & x_e \in \{0, 1\}, & e \in E(K_n).
\end{array}
$$

TSP cutting plane algorithm

TSP subtour cutting plane algorithm

Input: TSP instance (K_n, c)

Output: Incidence vector x^* of a minimum cost Hamiltonian circuit

(1) Let $P = \{x \in [0,1]^{E(K_n)} \mid \sum_{e \in \delta(v)} x_e = 2\}$ be the LP relaxation of (TSP-SEC) without subtour elimination constraints.

(2) Let x^* be the optimal solution to $\min\{c^t x \mid x \in P\}$.

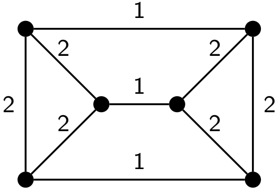
(3) Find a proper subset X of $V(K_n)$ such that the subtour inequality for X and x^* is violated, and add this inequality to P. If such an X was found, go to (2).

(4) If x^* is integral, return x^*.

(5) Find a Gomory cut separating x^* from the convex hull of $P \cap \mathbb{Z}^n$. Add this cut to P, and go to (2).

Integrality of the subtour polytope

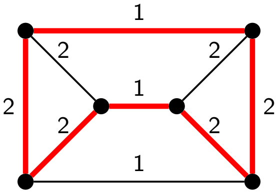
Unfortunately, the LP relaxation of (TSP-SEC) contains non-integral points. I.e., one might have to add Gomory cuts in the end in order to achieve integrality:



All missing edges have very large cost.

Integrality of the subtour polytope

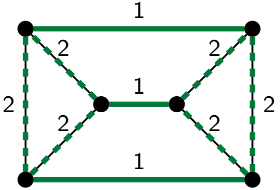
Unfortunately, the LP relaxation of (TSP-SEC) contains non-integral points. I.e., one might have to add Gomory cuts in the end in order to achieve integrality:



An optimal TSP tour has cost 10.

Integrality of the subtour polytope

Unfortunately, the LP relaxation of (TSP-SEC) contains non-integral points. I.e., one might have to add Gomory cuts in the end in order to achieve integrality:



An optimal TSP tour has cost 10.
An optimal fractional TSP tour (dashed: $x_e = 0.5$) has cost 9 satisfying all subtour constraints.

Separation of subtour inequalities

However, there are also good news:

Lemma

Given a rational point $x \in [0,1]^{E(K_n)}$ with $\sum_{e \in \delta(v)} x_e = 2$ for all $v \in K_n$, there is a polynomial-time algorithm that computes a proper subset X of $V(K_n)$ such that the subtour inequality for X and x is violated, or decides that no such X exists.

Proof.

Let x be as in the lemma. Let X be a proper subset of $V(K_n)$ and let $\delta(X)$ be the set of edges connecting X with $V(K_n) \setminus X$. Then

$$\sum_{e \in \delta(X)} x_e + 2 \sum_{e \in E(K_n[X])} x_e = \sum_{v \in X} \sum_{e \in \delta(v)} x_e = 2|X|,$$

so the subtour inequality is equivalent to

$$\sum_{e \in \delta(X)} x_e \geq 2.$$

Separation of subtour inequalities

Proof (cont.)

Hence the subtour inequality is violated for some X if and only if the minimum cut w.r.t. x has capacity less than 2. Finding a minimum cut can be done in polynomial time ($\rightarrow$ Optimization I). □

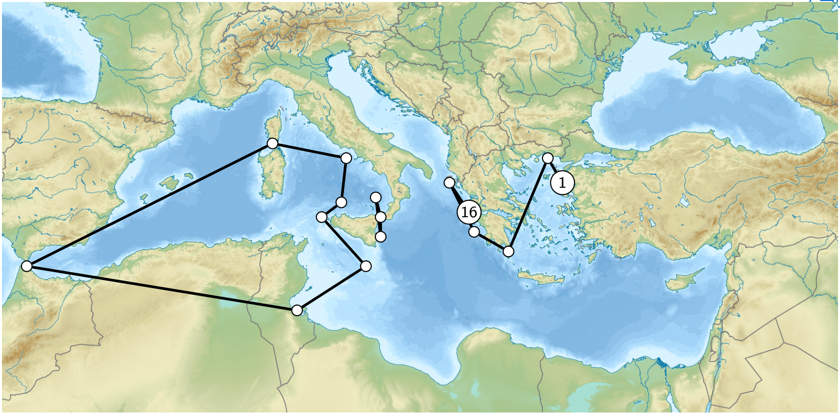
Remarks

- The subtour inequalities are facet-defining (Grötschel, Padberg, 1979) and can be separated efficiently $\rightarrow$ probably a good source for cuts.
- Because of the separation-optimization equivalence, optimization over the *subtour polytope* – i.e., the polytope described by the LP relaxation of (TSP-SEC) – can hence be done in polynomial time, although it has exponentially many constraints.
- Note that if x is integral – i.e., the incidence vector of a union of vertex-disjoint circuits – then a violated subtour elimination constraint can be detected by a simple traversal of one of the circuits.

Odysseus' cutting planes

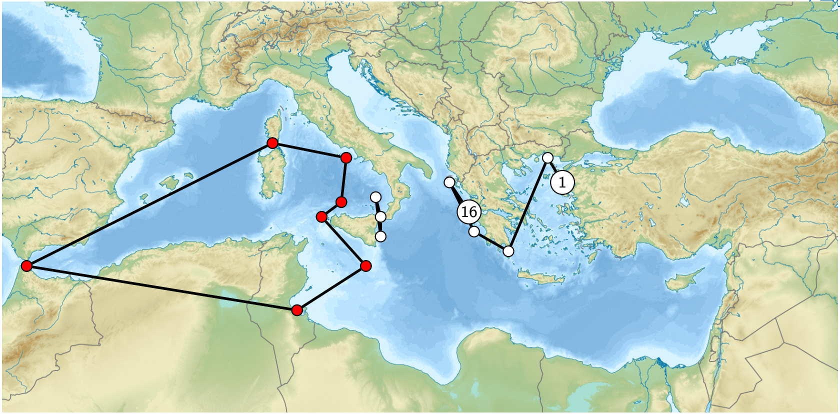
Remember: Odysseus wants to travel from Troy (1) to Ithaca (16) on a Hamiltonian path of minimum length visiting all 16 places exactly once. This problem can be transformed to a standard TSP (Problem Set 6). Let's solve this TSP with a subtour elimination cutting plane approach.

Odysseus' cutting planes



LP #1: 0 subtour constraints, 0 fractional variables, length = 5850

Odysseus' cutting planes

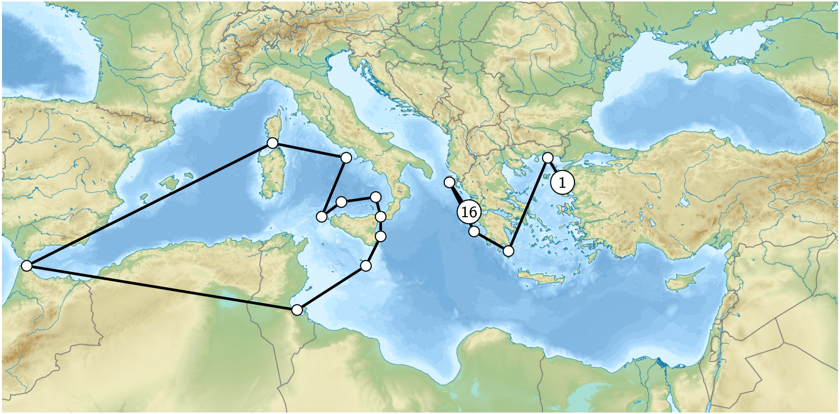


LP #1: 0 subtour constraints, 0 fractional variables, length = 5850

add subtour constraint for $X = \{4, 5, 6, 9, 10, 11, 14\}$:

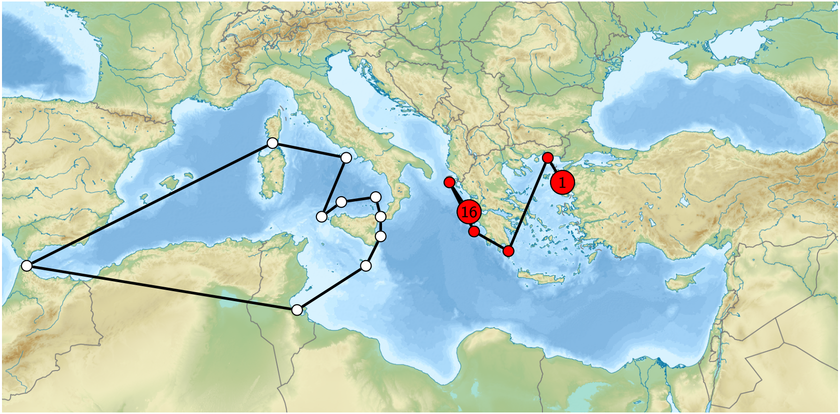
$$x_{4,5} + x_{4,6} + x_{4,9} + x_{4,10} + x_{4,11} + x_{4,14} + x_{5,6} + x_{5,9} + x_{5,10} + x_{5,11} + x_{5,14}$$
$$+ x_{6,9} + x_{6,10} + x_{6,11} + x_{6,14} + x_{9,10} + x_{9,11} + x_{9,14} + x_{10,11} + x_{10,14} + x_{11,14} \leq 6$$

Odysseus' cutting planes



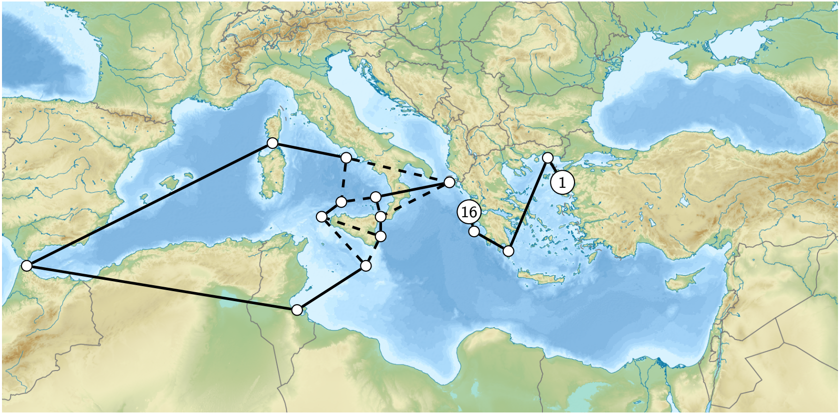
LP #2: 1 subtour constraint, 0 fractional variables, length = 5942

Odysseus' cutting planes



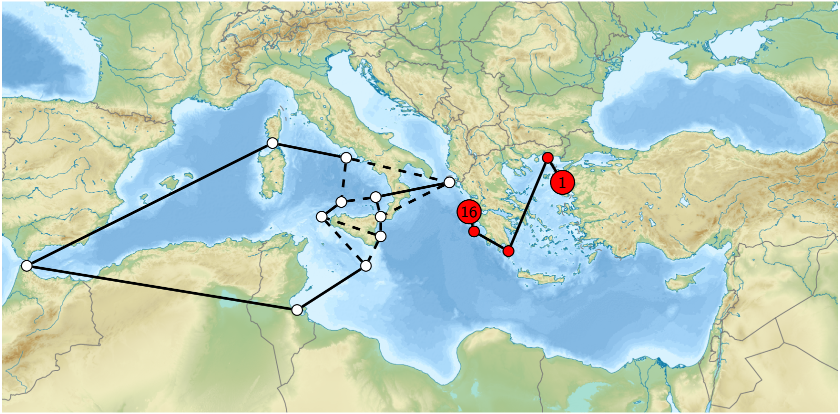
LP #2: 1 subtour constraint, 0 fractional variables, length = 5942
add subtour constraint for $X = \{1, 2, 3, 8, 15, 16\}$

Odysseus' cutting planes



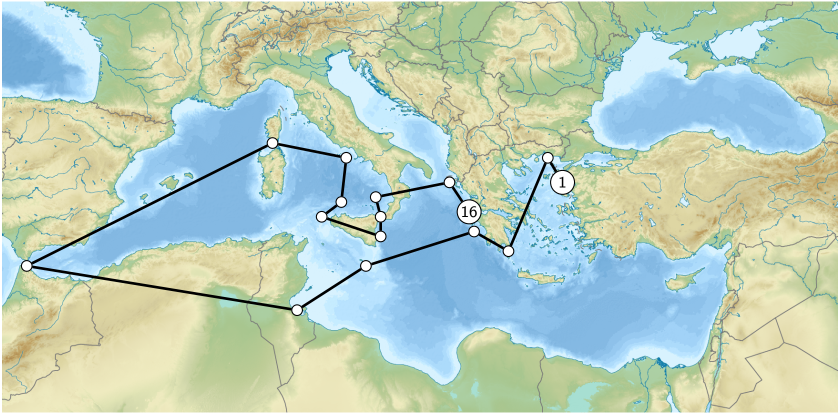
LP #3: 2 subtour constraints, 8 fractional variables, length = 6379.5

Odysseus' cutting planes



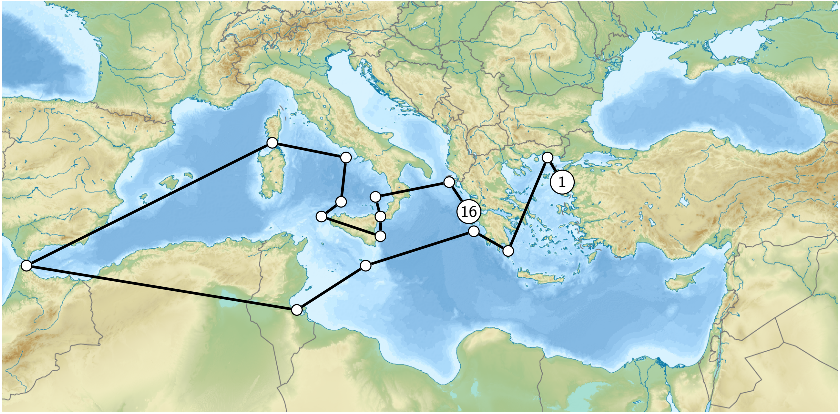
LP #3: 2 subtour constraints, 8 fractional variables, length $= 6379.5$
add subtour constraint for $X = \{1, 2, 3, 8, 16\}$

Odysseus' cutting planes



LP #4: 3 subtour constraints, 0 fractional variables, length = 6507

Odysseus' cutting planes



LP #4: 3 subtour constraints, 0 fractional variables, length = 6507

no violated subtour constraint

optimal solution found!

Odysseus' cutting planes

Remarks

- Odysseus' problem could be solved in only 4 iterations of the TSP subtour cutting plane algorithm: We solved 4 linear programs and added in total 3 subtour elimination constraints.
- A full formulation of (TSP-SEC) for Odysseus' TSP instance would have required $2^{17} - 2 = 131\,070$ subtour elimination constraints.
- ZIB's LP solver SoPlex (`soplex.zib.de`) solves each LP in less than 10 ms.
- We achieved integrality without adding further non-subtour cuts.

Chapter 4

Linear and Integer Programming

§4.4 Branch-and-Bound

Branch-and-Bound

Branch-and-Bound (Land, Doig, 1960) is a method that applies to various combinatorial optimization problems, and in particular to integer programming. It is an alternative to cutting plane methods, avoiding the generation of cuts by enumeration of feasible solutions. It comprises two main subroutines:

Branch: Given a subset of feasible solutions, find a partition into at least two non-empty subsets.

Bound: Given a subset of feasible solutions, compute a lower bound (for minimization problems) on the objective value of any element.

The running time of a branch-and-bound algorithm depends on the precise realization of these two subroutines.

Branch-and-Bound

Basic branch-and-bound algorithm

Input: a minimization problem instance with cost function c and (an implicitly given) non-empty finite set $\mathcal{X}$ of feasible solutions
Output: $x^* \in \mathcal{X}$ s.t. $c(x^*) = \min\{c(x) \mid x \in \mathcal{X}\}$

(1) Let T be a tree with precisely one vertex $\mathcal{X}$. Mark $\mathcal{X}$ as active. Set $U := \infty$.

(2) If there is no active vertex of T, return x^*.

(3) Node Selection: Let X be an active vertex of T, mark X non-active.

(4) Branch: Find a partition $X = X_1 \dot\cup \ldots \dot\cup X_k$.

(5) Bound: For each $i = 1, \ldots, k$:
    Find a lower bound L_{X_i} on any solution in X_i.
    If $X_i = \{x\}$ and $L_{X_i} < U$: Set $U := c(x)$ and $x^* := x$.
    If $|X_i| > 1$ and $L_{X_i} < U$, add the vertex X_i and an edge $\{X, X_i\}$ to T and mark X_i as active.

    Go to (2).

Remarks

- During the course of the algorithm, U is always an upper bound on the optimal value. The algorithm *prunes* subsets $X_i \subseteq X$ for which

$$\min\{c(x) \mid x \in X_i\} \geq L_{X_i} \geq U \geq c(x^*) \geq \min\{c(x) \mid x \in \mathcal{X}\},$$

  and all other feasible solution are enumerated.
- The number of iterations is at most the number of vertices in the tree T. The vertices of T are distinct subsets of $\mathcal{X}$, so that the algorithm terminates within $O(2^{|\mathcal{X}|})$ steps.
- In the worst case – when the bounds L_{X_i} are weak – all elements of $\mathcal{X}$ get enumerated.
- Calling heuristics can improve the bound U.
- The better the bounds L_{X_i} and U, the smaller the tree, the faster the algorithm.
- In the context of TSP, branch-and-bound has been applied first by Little et al., 1963, and they in fact invented the name.

Branch-and-bound for 0-1-IPs

0-1-IP branch-and-bound algorithm

Input: an IP of the form $\min\{c^t x \mid Ax \leq b, x \in \{0,1\}^n\}$ over a polytope
Output: an optimal solution x^* or "infeasible"

(1) Let T be a tree with precisely one vertex labeled with $\emptyset$. Mark the vertex as active. Set $U := \infty$ and $x^* :=$"infeasible".

(2) If there is no active vertex of T, return x^*.

(3) Node Selection: Let X be an active vertex of T, mark X non-active.

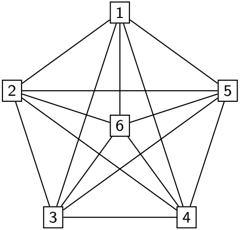
(4) Bound: Let x be an optimal solution to the LP relaxation with the additional constraints X (go to (2) if infeasible). Set $L_X := c(x)$.
If x is not integral, $|X| < n$ and $L_X < U$: Go to (5).
If x is integral and $L_X < U$: Set $U := c(x)$ and $x^* := x$.
Go to (2).

(5) Branch: Select a variable x_i not listed in X, and connect X to two new active tree vertices $X \cup \{x_i = 0\}$ and $X \cup \{x_i = 1\}$. Go to (3).

Branch-and-bound for 0-1-IPs

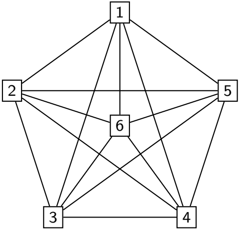
- This is a specialization of the basic branch-and-bound algorithm to integer programming with binary variables. LP relaxations are natural candidates for lower bounds.
- It is reasonable to solve the LP at the root node $\emptyset$ as well.
- When an LP solution is integral, it is not reasonable to branch further, because the objective value cannot decrease in deeper tree levels, as more and more constraints are added.
- When all n variables appear in X, all variables are fixed, so that the LP is trivial to solve.
- Again, heuristics can improve U, so that more tree nodes are pruned.
- As a branching rule, one typically selects variables with relaxation values close to 0.5.
- At any time, there is a *global* lower bound L, and hence there is information on the quality of the current solution by means of the *optimality gap* defined as $(U - L)/U \geq 0$ in the case $U > 0$.
- One may stop if the optimality gap is below a certain threshold.

TSP branch-and-bound example



c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

TSP branch-and-bound example

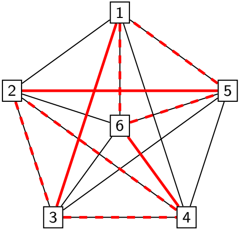


c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = \infty$ $\qquad$ $L = -\infty$

root

TSP branch-and-bound example



c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

LP solution at $X = \emptyset$ (root):

$x_{1,3} = 1$ $\quad x_{2,5} = 1$

$x_{1,5} = 0.5$ $\quad x_{3,4} = 0.5$

$x_{1,6} = 0.5$ $\quad x_{4,6} = 1$

$x_{2,3} = 0.5$ $\quad x_{5,6} = 0.5$

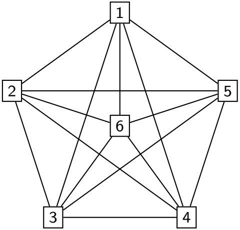
$x_{2,4} = 0.5$

fractional, objective value: 7

$U = \infty$ $\qquad L = 7$

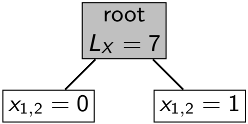
root
$L_X = 7$

TSP branch-and-bound example

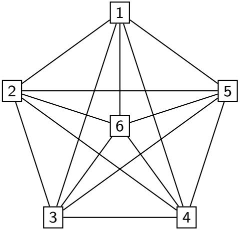


c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = \infty$ $\qquad$ $L = 7$

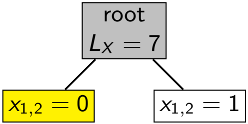


TSP branch-and-bound example

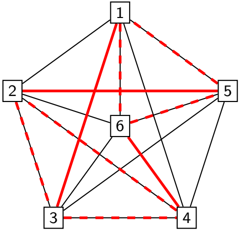


c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = \infty$ $\quad$ $L = 7$

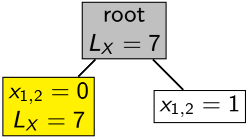


TSP branch-and-bound example



c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = \infty$ $\quad$ $L = 7$



LP solution at $X = \{x_{1,2} = 0\}$:

$x_{1,3} = 1$ $\quad$ $x_{2,5} = 1$

$x_{1,5} = 0.5$ $\quad$ $x_{3,4} = 0.5$

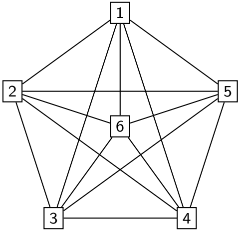
$x_{1,6} = 0.5$ $\quad$ $x_{4,6} = 1$

$x_{2,3} = 0.5$ $\quad$ $x_{5,6} = 0.5$

$x_{2,4} = 0.5$

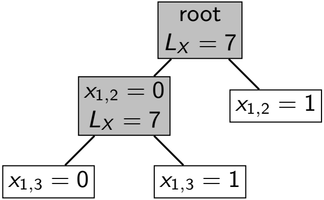
fractional, objective value: 7

TSP branch-and-bound example

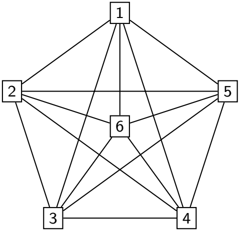


c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = \infty$ $L = 7$

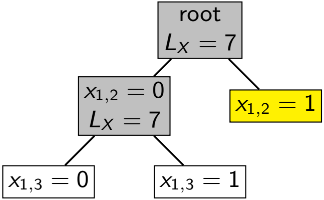


TSP branch-and-bound example

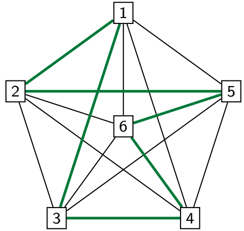


c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = \infty$ $\qquad$ $L = 7$

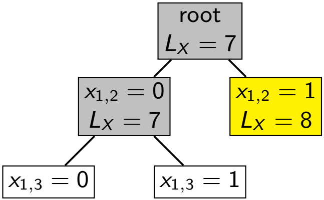


TSP branch-and-bound example



c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = 8$ $\qquad$ $L = 7$

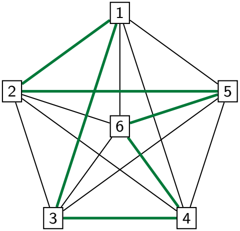


LP solution at $X = \{x_{1,2} = 1\}$:

$$x_{1,2} = 1 \qquad x_{3,4} = 1$$
$$x_{1,3} = 1 \qquad x_{4,6} = 1$$
$$x_{2,5} = 1 \qquad x_{5,6} = 1$$

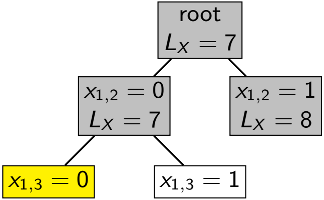
integral, objective value: 8

TSP branch-and-bound example

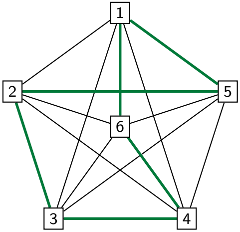


c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = 8$ $L = 7$

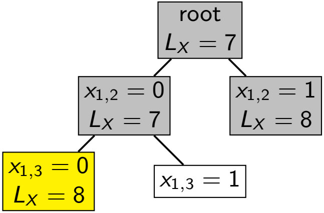


TSP branch-and-bound example



c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = 8$ $\qquad$ $L = 7$



LP solution at $X = \{x_{1,2} = 1, x_{1,3} = 0\}$:

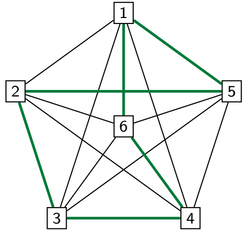
$x_{1,5} = 1$ $\qquad$ $x_{2,5} = 1$

$x_{1,6} = 1$ $\qquad$ $x_{3,4} = 1$

$x_{2,3} = 1$ $\qquad$ $x_{4,6} = 1$

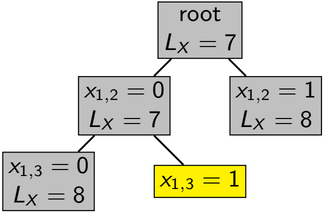
integral, objective value: 8

TSP branch-and-bound example

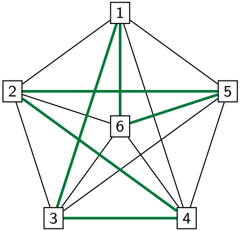


c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = 8$ $\qquad$ $L = 7$

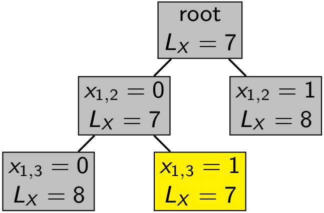


TSP branch-and-bound example



c	1	2	3	4	5	6
1	0	3	1	3	2	2
2	3	0	1	1	1	3
3	1	1	0	1	4	4
4	3	1	1	0	3	1
5	2	1	4	3	0	1
6	2	3	4	1	1	0

$U = 7$ $\qquad L = 7$



LP solution at $X = \{x_{1,2} = 1, x_{1,3} = 1\}$:

$x_{1,3} = 1 \qquad x_{2,5} = 1$

$x_{1,6} = 1 \qquad x_{3,4} = 1$

$x_{2,4} = 1 \qquad x_{5,6} = 1$

integral, objective value: 7

Branch-and-cut

Cutting planes and branch-and-bound can be combined to **branch-and-cut**:

At every node of the branch-and-bound tree, look for cutting planes that are easy to find, e.g., subtour inequalities for TSP. Solve the LP relaxation again until no more cutting planes are found, the solution is integral or some iteration limit is reached. Then start branching.

It is also possible to add *local cuts*, i.e., cutting planes that are only valid for the subproblem at the current b&b tree node.

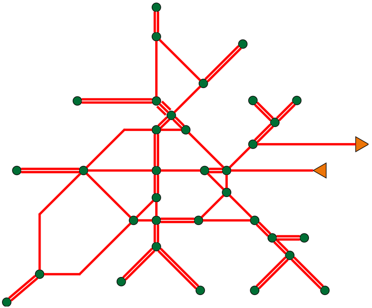
Branch-and-cut produces very small branch-and-bound trees and is the method of choice for hard IPs. It is also used by the TSP solver `concorde`, which holds the world record for the largest TSP instance (85 900 vertices) solved to proven optimality.

If you want to try out branch-and-cut software, have a look at *SCIP*, the open-source branch-and-cut framework developed at ZIB (`scip.zib.de`).

Traffic Optimization:

Optimal Tours in Graphs

Niels Lindner



Lecture 8

December 2, 2019