

Introduction to Integer Linear Programming

Leo Liberti, Ruslan Sadykov

LIX, École Polytechnique

`liberti@lix.polytechnique.fr`

`sadykov@lix.polytechnique.fr`

Contents

- IP formulations and examples
- Total unimodularity
- The Cutting Planes method
- The Branch-and-Bound method
- The Branch-and-Cut method

Definitions

- Mathematical programming formulation:

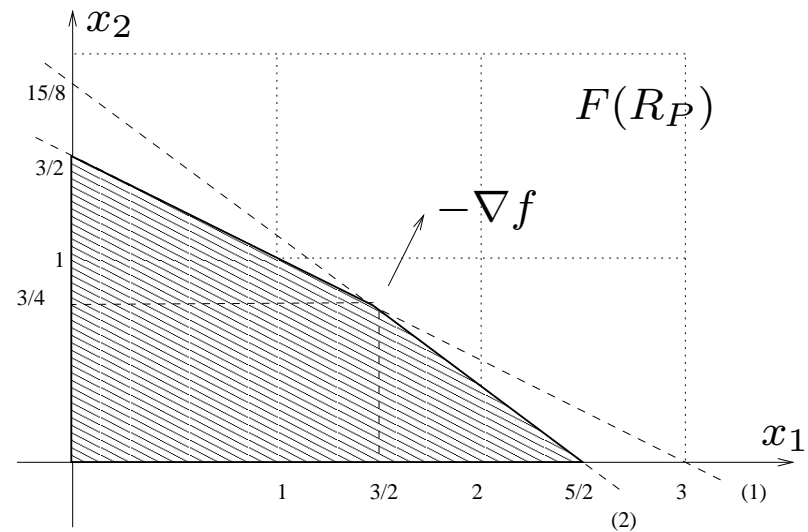
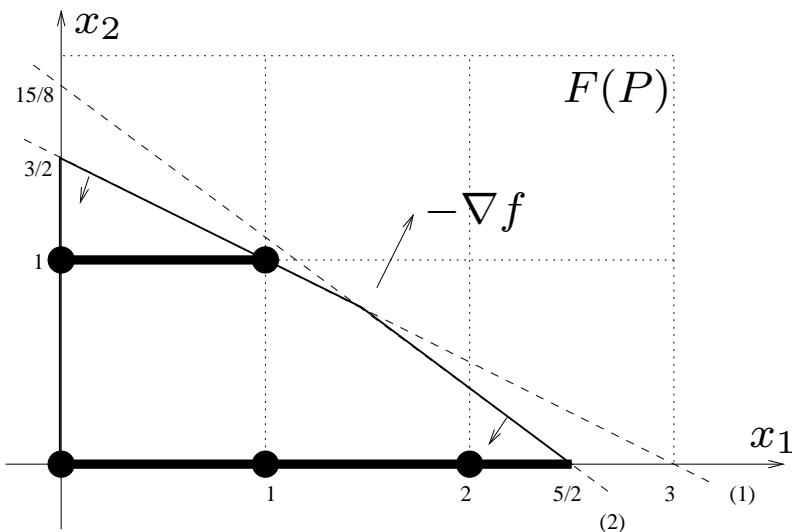
$$\left. \begin{array}{l} \min_x \quad c^\top x + d^\top y \\ \text{s.t.} \quad Ax + By \leq b \\ \quad \quad x \geq 0, y \geq 0, \\ \quad \quad x \in \mathbb{Z}^n \end{array} \right\} [P] \quad (1)$$

- The *linear (or continuous) relaxation* R_P of P is obtained by P relaxing (i.e. removing) the integrality constraints
- Let $F(P)$ be the feasible region of P : we have $F(P) \subseteq F(R_P)$
- Let (x^*, y^*) be the solution of P and (\bar{x}, \bar{y}) be the solution of R_P ; then $c^\top \bar{x} + d^\top \bar{y} \leq c^\top x^* + d^\top y^*$: R_P is a *lower bounding problem* w.r.t. P

Simple example

Consider example:

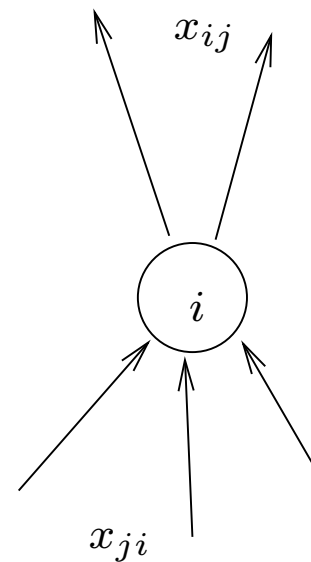
$$\left. \begin{array}{l} \min -2x_1 - 3x_2 \\ x_1 + 2x_2 \leq 3 \\ 6x_1 + 8x_2 \leq 15 \\ x_1 \in \mathbb{R}_+, x_2 \in \mathbb{Z}_+ \end{array} \right\}$$



Maximum flow problem

Given a network on a directed graph $G = (V, A)$ with a source node s , a destination node t , and integer capacities u_{ij} on each arc (i, j) . We have to determine the maximum amount of integral material flow that can circulate on the network from s to t . The variables $x_{ij} \in \mathbb{Z}$, defined for each arc (i, j) in the graph, denote the number of flow units.

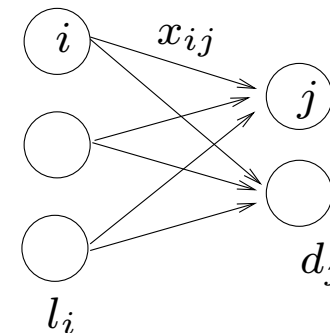
$$\begin{array}{l}
 \max_x \quad \sum_{(s,i) \in A} x_{si} \\
 \forall i \leq V, \quad \begin{array}{l} i \neq s \\ i \neq t \end{array} \quad \sum_{(i,j) \in A} x_{ij} = \sum_{(j,i) \in A} x_{ji} \\
 \forall (i,j) \in A \quad 0 \leq x_{ij} \leq u_{ij} \\
 \forall (i,j) \in A \quad x_{ij} \in \mathbb{Z}
 \end{array}$$



Transportation problem

Let x_{ij} be the (discrete) number of product units transported from plant $i \leq m$ to customer $j \leq n$ with respective unit transportation cost c_{ij} from plant i to customer j . We model the problem of determining x minimizing the total cost, subject to production limits l_i at plant i and demand d_j at customer j , as follows:

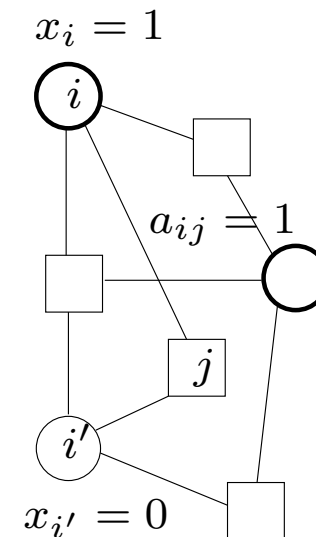
$$\begin{array}{l}
 \min_x \quad \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij} \\
 \forall i \leq m \quad \sum_{j=1}^n x_{ij} \leq l_i \\
 \forall j \leq n \quad \sum_{i=1}^m x_{ij} \geq d_j \\
 \forall i, j \quad x_{ij} \in \mathbb{Z}_+
 \end{array}$$



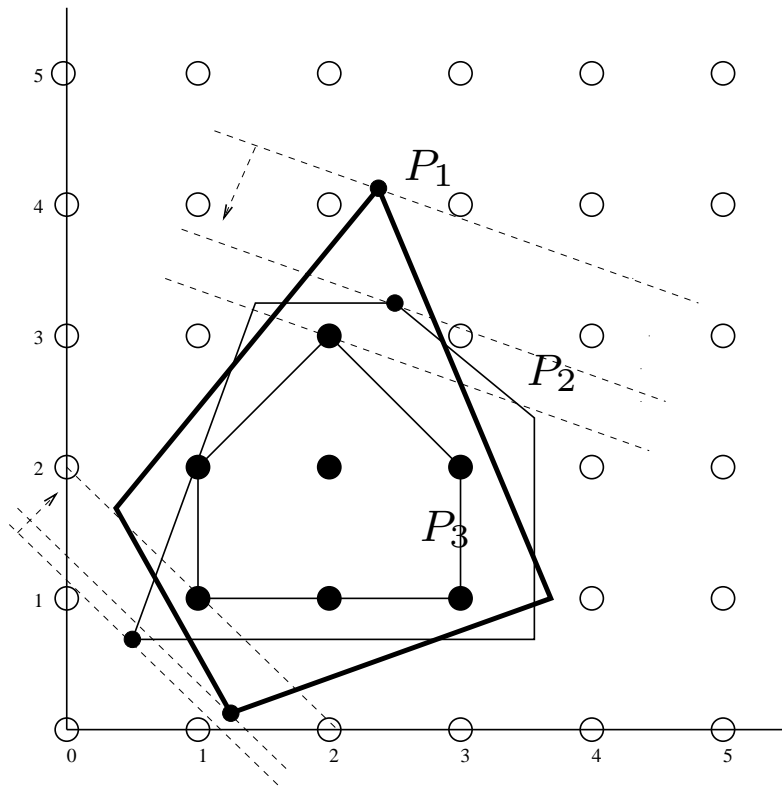
Set Covering problem

Let $x_i = 1$ if a servicing facility will be built on geographical region $i \leq m$ and 0 otherwise. The cost of building a facility on region i is f_i , and $a_{ij} = 1$ if a facility on region i can serve town $j \leq n$, and 0 otherwise. We need to determine $x \in \{0, 1\}^m$ so that each town is serviced by at least one facility and the total cost is minimum.

$$\left. \begin{array}{l} \min_x \sum_{i=1}^m f_i x_i \\ \forall j \leq n \quad \sum_{i=1}^m a_{ij} x_i \geq 1 \\ \forall i \leq m \quad x_i \in \mathbb{Z}_+ \end{array} \right\}$$



Good and ideal formulations



The smaller is $F(R_P)$, the bigger (better) is the lower bound produced by R_P . As $F(R_{P_3}) \subset F(R_{P_2})$ and $F(R_{P_1}) \subset F(R_{P_2})$, the formulation P_3 is better than P_1 if and P_2 .

Here P_3 is the best possible (ideal) formulation.

Formally, R_{P_3} defines the *convex hull* of P .

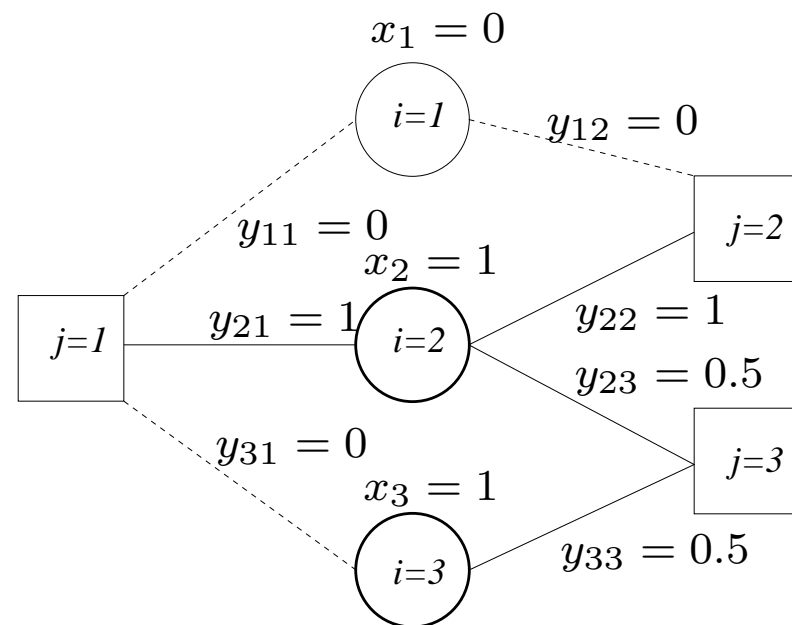
$P = \{x^1, \dots, x^t\}$, then $\text{conv}(P) = \{x :$

$$x = \sum_{i=1}^t \lambda_i x^i, \sum_{i=1}^t \lambda_i = 1, \lambda_i \geq 0, \forall i = 1, \dots, t\}.$$

Uncapacitated Facility Location problem

Similar to the Set Covering Problem, except for the addition of the variable transportation costs c_{ij} , which arise if the demand of town j is fully served by facility i . Let y_{ij} be the fraction of demand of town j served by facility i .

$$\begin{aligned}
 & \min_{x,y} \sum_{i=1}^m f_i x_i + \sum_{i=1}^m \sum_{j=1}^n c_{ij} y_{ij} \\
 & \forall j \leq n, \quad \sum_{i=1}^m y_{ij} = 1 \\
 & \forall i \leq m, \quad \sum_{j=1}^n y_{ij} \leq n x_i \\
 & \forall i \leq m, \forall j \leq n, \quad y_{ij} \geq 0, \\
 & \forall i \leq m, \quad x_i \in \{0, 1\}.
 \end{aligned}$$



UFL problem II

We can change constraints

$$\forall i \leq m, \quad \sum_{j=1}^m y_{ij} \leq nx_i \quad [R_1]$$

to constraints

$$\forall i \leq m, \forall j \leq n, \quad y_{ij} \leq x_i. \quad [R_2]$$

Formulation R_2 is better than R_1 as $F(R_2) \subset F(R_1)$. We can verify it by showing $F(R_2) \subseteq F(R_1)$ and finding a point $(x, y) \in F(R_1) \setminus F(R_2)$.

Rounding heuristic

There is a strong relation between an integer program and its linear relaxation.

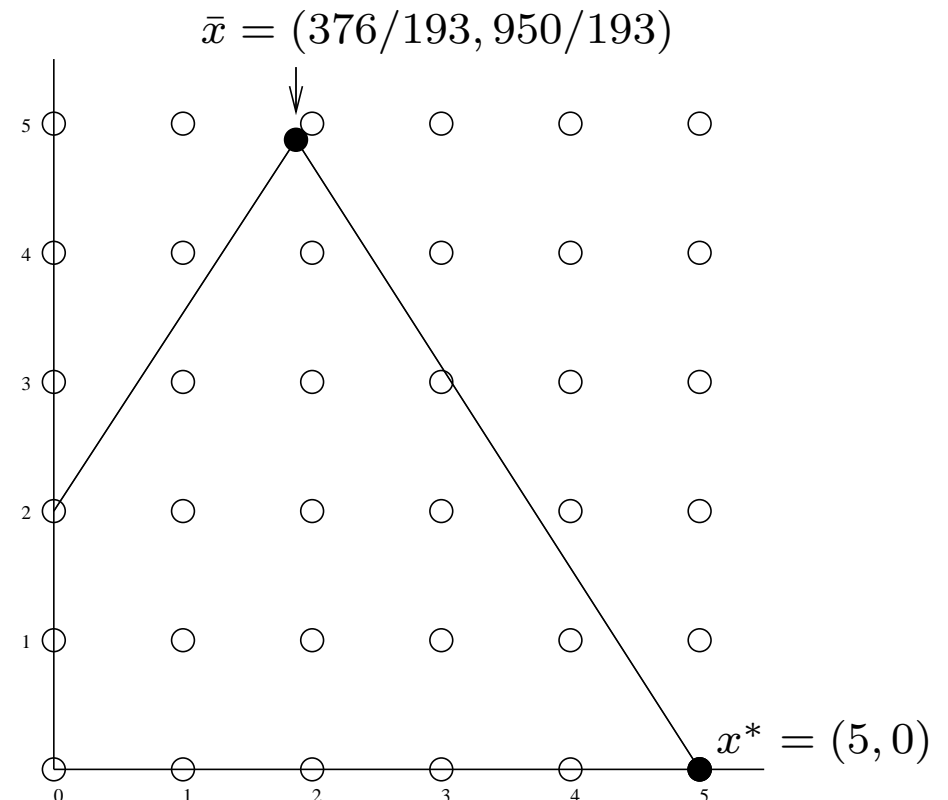
But just rounding the solution \bar{x} of the LP relaxation does not always produce good results. Consider the integer program:

$$\max 1.00x_1 + 0.64x_2$$

$$50x_1 + 31x_2 \leq 250$$

$$3x_1 - 2x_2 \geq -4$$

$$x_1, x_2 \in \mathbb{Z}_+$$



Main algorithmic ideas

- If we can say a priori that $\bar{x} \in \mathbb{Z}^n$ then can solve P by simply solving R_P (*total unimodularity property*).
- Add constraints to get P' such that $\bar{x}' \in \mathbb{Z}^n$ (*cutting planes algorithm*).
- Solve by “smart” enumeration of all solutions (*Branch-and-Bound algorithm*).
- Combine adding constraints and enumeration (*Branch-and-Cut algorithm*).
- Modern Integer Programming solvers (like *Cplex*) use the Branch-and-Cut algorithm.

Total unimodularity I

- Consider system $Bx = b$ where $B = (b_{ij})$ is invertible $n \times n$ s.t. $b_{ij} \in \mathbb{Z}$ for all i, j
- Solve for x , get $B^{-1}b$
- From inverse matrix formula, infer $B^{-1} = \frac{1}{|B|}C$ with C integral
- If $|B| \in \{1, -1\}$ then $x = B^{-1}b = \pm Cb \in \mathbb{Z}^n$
- A square invertible matrix B s.t. $|B| = \pm 1$ is *unimodular*
- An $m \times n$ matrix A s.t. every square submatrix has determinant in $\{-1, 0, 1\}$ is *totally unimodular* (TUM)
- Theorem: if A is TUM, then for all $b \in \mathbb{R}^n$, every vertex of the polyhedron $\{x \in \mathbb{R}^n \mid Ax \leq b\}$ is integral. *Intuitively, every vertex can be written as $B^{-1}b$ for B square submatrix of A*

Total unimodularity II

- If A is TUM, A^T and $(A|I)$ are TUM
- *TUM Sufficient conditions.* An $m \times n$ matrix A is TUM if:
 1. for all $i \leq m, j \leq n$ we have $a_{ij} \in \{0, 1, -1\}$;
 2. each column of A contains at most 2 nonzero coefficients;
 3. there is a partition R_1, R_2 of the set of rows such that for each column j , $\sum_{i \in R_1} a_{ij} - \sum_{i \in R_2} a_{ij} = 0$.
- Example: take $R_1 = \{1, 3, 4\}$, $R_2 = \{2\}$

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & -1 & 1 \\ -1 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & -1 & 0 \end{pmatrix}$$

Total unimodularity III

- Consider digraph $G = (V, A)$ and a nonnegative flow $x_{ij} \in \mathbb{R}_+$ on each arc; the flow conservation equations

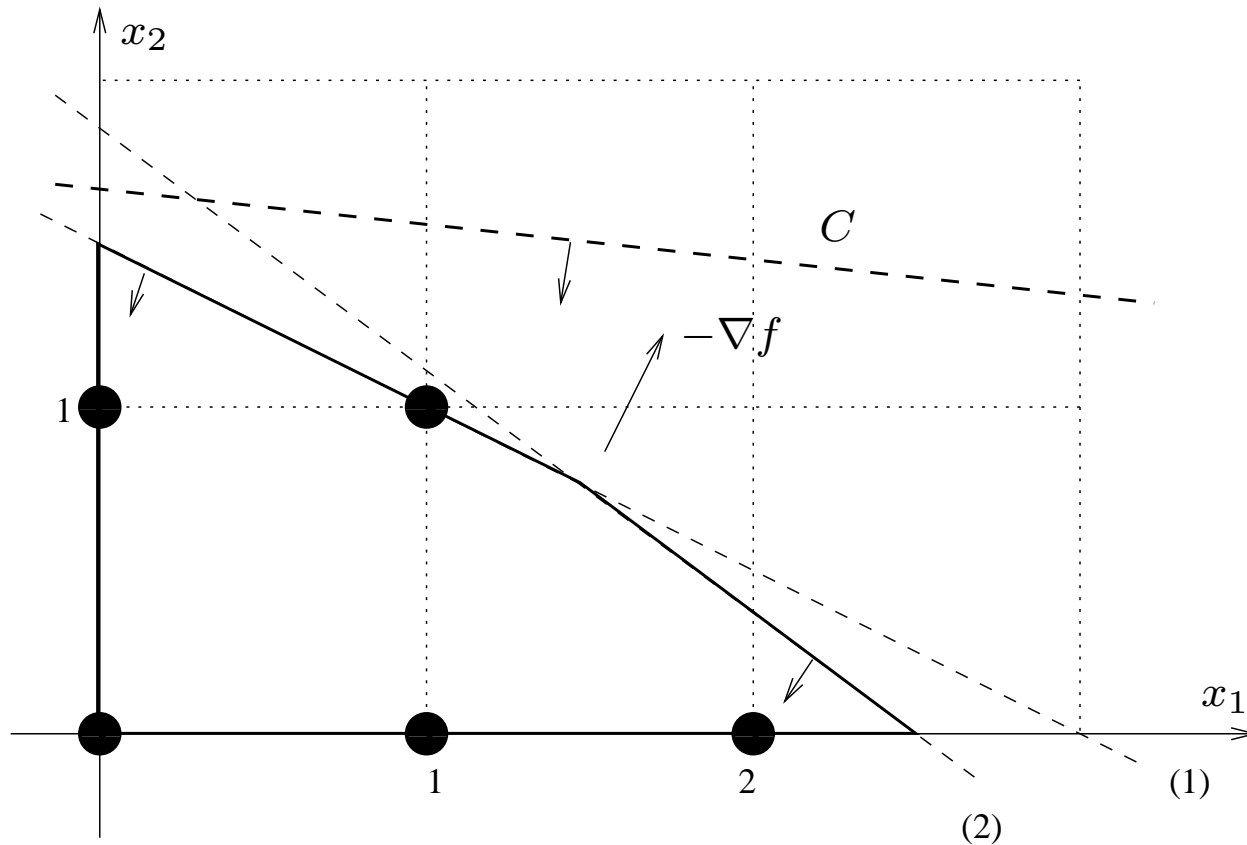
$$\boxed{\forall i \in V \quad \sum_{(i,j) \in A} x_{ij} - \sum_{(j,i) \in A} x_{ji} = 0} \text{ yield a TUM matrix}$$

(partition: $R_1 = \text{all rows}$, $R_2 = \emptyset$).

- Maximum flow problem can be solved to integrality by simply solving the continuous relaxation with the simplex algorithm
- Constraints of the transportation problem also form a TUM matrix. Partition: $R_1 = \{\sum_{j=1}^n x_{ij} \leq l_i\}_{i \leq m}$,
 $R_2 = \{\sum_{i=1}^m x_{ij} \geq d_j\}_{j \leq n}$.
- Constraints of the set covering problem do not form a TUM. To prove this, you just need to find a counterexample.

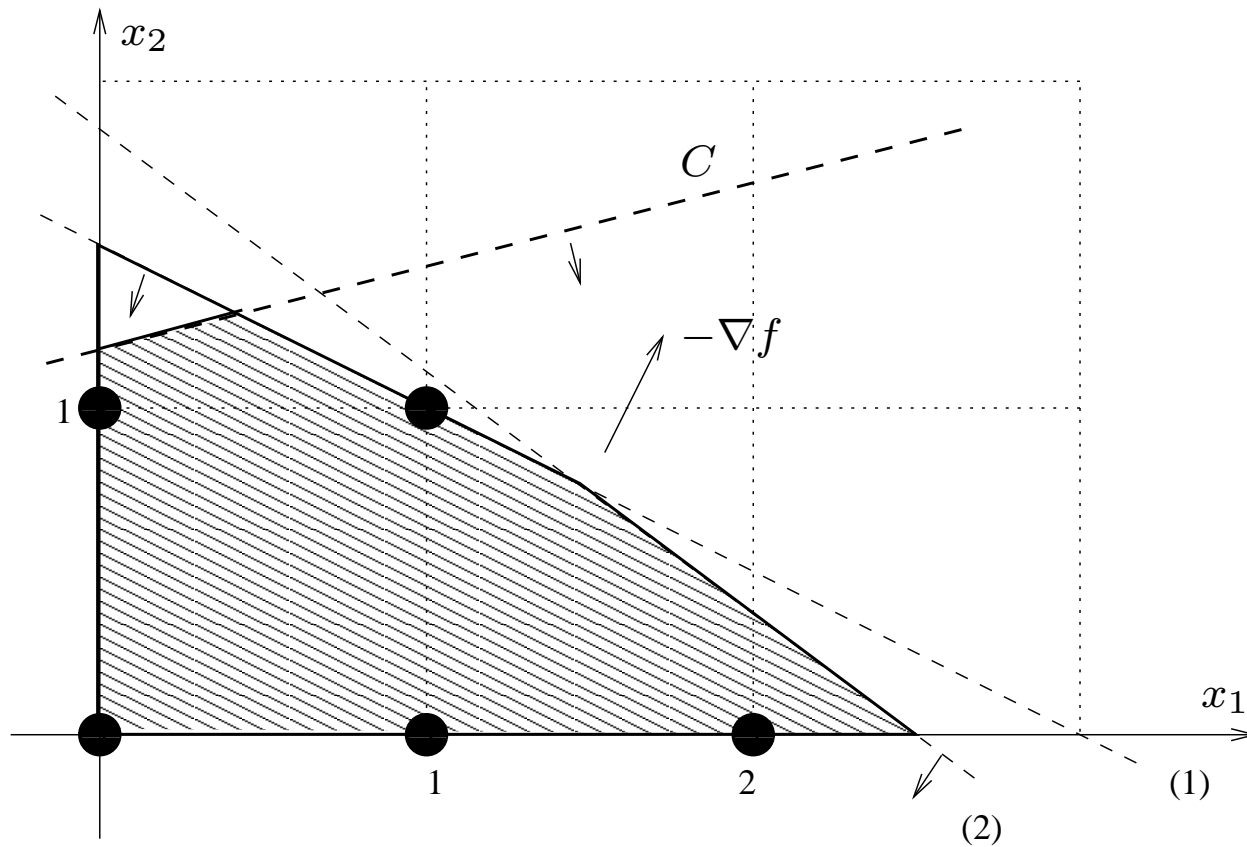
Cutting planes: definitions I

A constraint $C \equiv \pi^\top x \leq \pi_0$ is *valid* for P if
 $\forall x' \in F(P) \quad (\pi^\top x' \leq \pi_0)$



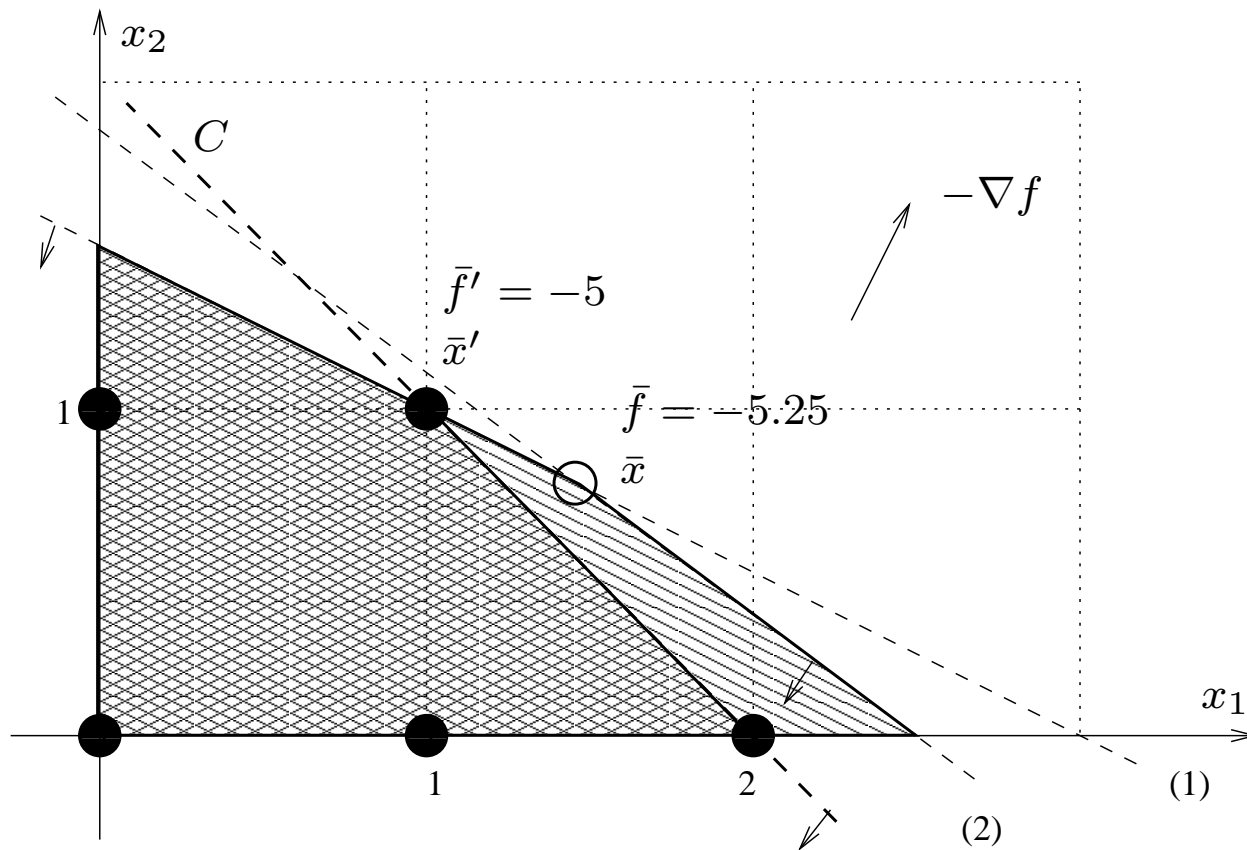
Cutting planes: definitions II

Let P' be problem P with the added valid constraint C . C is a *cutting plane* for P if $F(R_{P'}) \subseteq F(R_P)$



Cutting planes: definitions III

Let f be the optimal objective function value for R_P and f' for $R_{P'}$. A cutting plane $C : \pi^T x \leq \pi_0$ is a *valid cut* for $\bar{x} \in R_P$ if $\pi^T \bar{x} > \pi_0$.

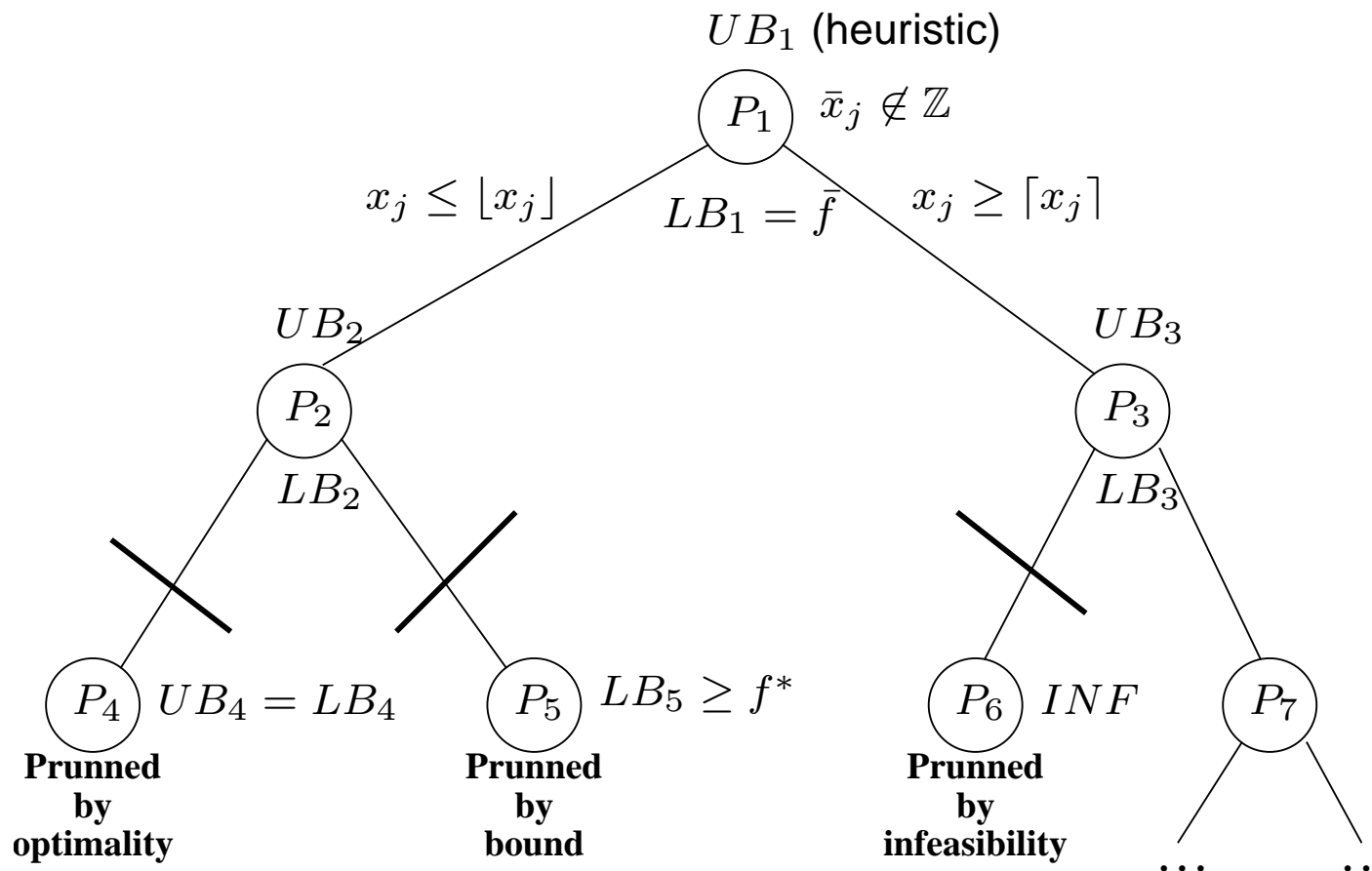


Cutting Plane Algorithm

- Overall strategy:
 1. Solve R_P , get relaxed solution \bar{x}
 2. If $\bar{x} \in \mathbb{Z}^n$ problem is solved, exit
 3. Use solution \bar{x} of R_P to construct a valid cut C for P
 4. Add the constraint C to the formulation of P
 5. Go back to 1
- The most important step of the algorithm: step 3 (*separation problem*).
- Cutting Plane algorithms may depend on the particular problem structure or be completely general.
- Independent of problem structure: *Gomory cutting planes*.
- Problem structure: *Row generation for the TSP*.

Branch-and-Bound I

Here we use the “divide and conquer” approach. If we cannot solve a problem, we break it into easier subproblems. We do it using an enumeration tree.



Branch-and-Bound II

1. Initialize list problem $L = \{P\}$, best objective function value $f^* = \infty$, $x^* = \text{"infeasible"}$
2. If $L = \emptyset$, terminate with solution x^*
3. Select a subproblem Q from L and remove it from L
4. (*Bound*) Solve R_Q to find solution \bar{x} with objective value \bar{f}
5. If R_Q is infeasible, back to 2 (prune by infeasibility)
6. If $\bar{f} \geq f^*$, Q cannot contain optimal solution, back to 2 (prune by bound)
7. If \bar{x} is integral and $\bar{f} < f^*$: update $x^* = \bar{x}$, $f^* = \bar{f}$, back to 2 (prune by optimality)
8. (*Branch*) Select a fractional component \bar{x}_j , generate two subproblems from Q with added constraints $x_j \leq \lfloor \bar{x}_j \rfloor$ and $x_j \geq \lceil \bar{x}_j \rceil$ respectively, add them to L , then back to 2

Branch-and-Bound III

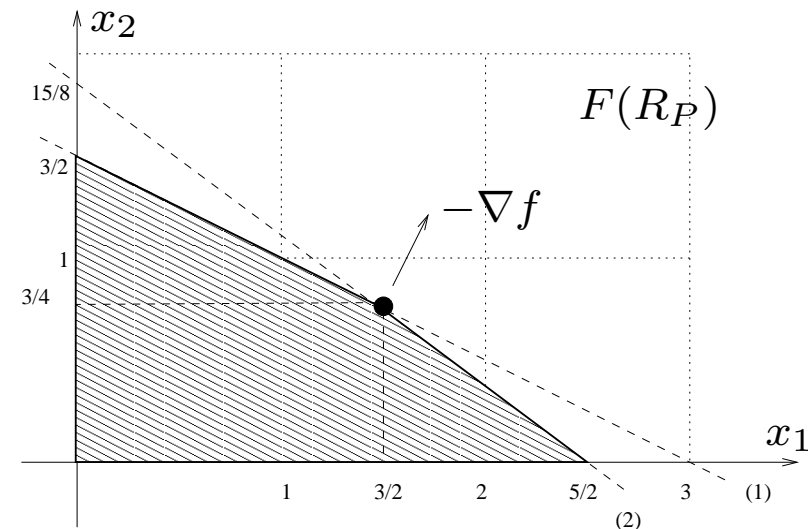
- How do we choose a subproblem Q from L (step 3)?
- How do we select a fractionary component \bar{x}_j from \bar{x} (step 8)?
- No “best answer”, depends on problem structure.
- Choice of subproblem: associate $LB = \bar{f}$ to each generated problem, then choose subproblem with minimum LB .
- Choice of fractionary component: choose the component with fractionary value closest to 0.5.

BB example I

Consider simple example:

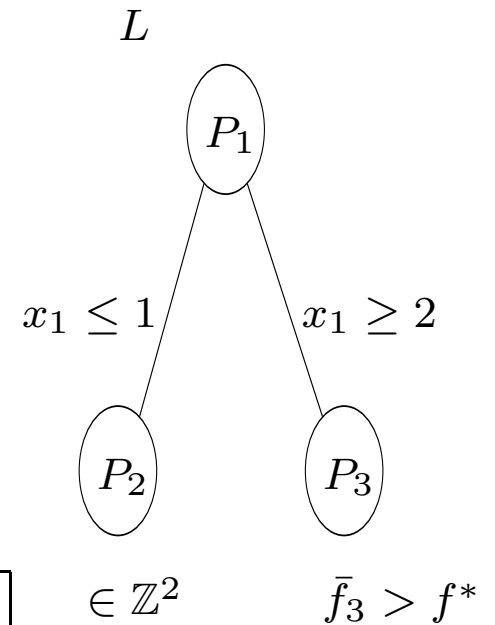
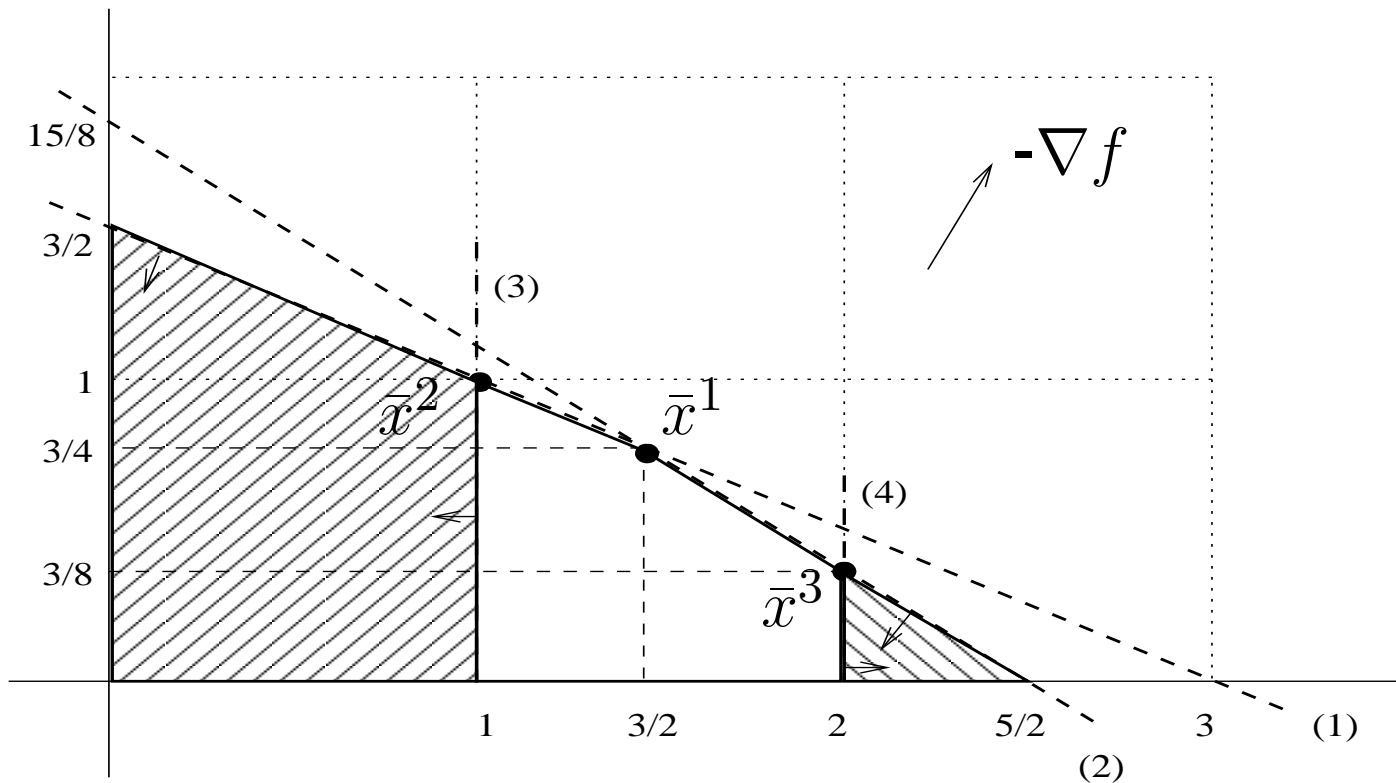
$$\left. \begin{array}{l} \min \quad -2x_1 - 3x_2 \\ \quad \quad x_1 + 2x_2 \leq 3 \\ \quad \quad 6x_1 + 8x_2 \leq 15 \\ \quad \quad x_1, x_2 \in \mathbb{Z}_+ \end{array} \right\}$$

Solution of R_P is at
 $\bar{x} = (3/2, 3/4)$ with
 $\bar{f} = -21/4$



BB example II

$\bar{x}^i =$ solution of R_{P_i} , $\bar{f}_i =$ optimal objective value of R_{P_i} , $\forall i$



P_2	P_1	P_3
$\bar{x}^2 = (1, 1)$	$\bar{x}^1 = (1.5, 0.75)$	$\bar{x}^3 = (2, 0.375)$
$\bar{f}_2 = -5$	$\bar{f}_1 = -5.25$	$\bar{f}_3 = -5.125$

Branch-and-Cut

- In the Branch-and-Bound algorithm, before branching, we generate valid cuts for the current fractional solution \bar{x} .
- The cuts are generated until there is no much progress on the value \bar{f} of the objective function.
- Cuts can be general or problem specific.
- Solvers, like *Cplex* generate cuts by default.
- Most used classes of general cuts: *Gomory cuts*, *(flow) cover cuts*.

Gomory inequalities

- 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, \dots, a_n) , and $u \in \mathbb{R}_+^m$.
- $\sum_{j=1}^n ua_j x_j \leq ub$ is valid for P as $u \geq 0$;
- $\sum_{j=1}^n \lfloor ua_j \rfloor x_j \leq ub$ is valid for P , as $x \geq 0$;
- $\sum_{j=1}^n \lfloor ua_j \rfloor x_j \leq \lfloor ub \rfloor$ is valid for X , as x is integer.
- Using this procedure, we can generate **all** valid inequalities for an integer program.

Cover inequalities

- Let $X = \left\{ x \in \{0, 1\}^n : \sum_{j=1}^n a_j x_j \leq b \right\}$,
 $a_j \geq 0, \forall j \leq n, b \geq 0, N = \{1, 2, \dots, n\}$.
- Set $C \subseteq N$ is a *cover* if $\sum_{j \in C} a_j > b$.
- If $C \subseteq N$ is a cover, then the cover inequality

$$\sum_{j \in C} x_j \leq |C| - 1$$

is valid for X .

Course material

- C. Papadimitriou, K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Dover, New York, 1998
- L. Wolsey, *Integer Programming*, John Wiley & Sons, Inc, New York, 1998.