6. Mixed Integer Linear Programming

Javier Larrosa

Albert Oliveras

Enric Rodríguez-Carbonell

Problem Solving and Constraint Programming (RPAR)

Mixed Integer Linear Programming

- A mixed integer linear program (MILP, MIP) is of the form

$$
\begin{aligned}
&\min\ c^T x \\
&Ax = b \\
&x \geq 0 \\
&x_i \in \mathbb{Z}\ \ \forall i \in \mathcal{I}
\end{aligned}
$$

- If all variables need to be integer, it is called a (pure) integer linear program (ILP, IP)
- If all variables need to be 0 or 1 (binary, boolean), it is called a $0-1$ linear program

Applications of MIP

- Used in contexts where, e.g.:
  - it only makes sense to take integral quantities of certain goods or resources, e.g.:
    - men (human resources planning)
    - power stations (facility location)
  - binary decisions need to be taken
    - producing a product (production planning)
    - assigning a task to a worker (assignment problems)
    - assigning a slot to a course (timetabling)
- And many many more...

Computational Complexity: LP vs. IP

- Including integer variables increases enourmously the modeling power, at the expense of more complexity
- LP's can be solved in polynomial time with interior-point methods (ellipsoid method, Karmarkar's algorithm)
- Integer Programming is an NP-complete problem. So:
  - There is no known polynomial-time algorithm
  - There are little chances that one will ever be found
  - Even small problems may be hard to solve
- What follows is one of the many approaches (and one of the most successful) for attacking IP's

LP Relaxation of a MIP

- Given a MIP

$$
(IP) \quad \begin{aligned} & \min \; c^T x \\ & Ax = b \\ & x \geq 0 \\ & x_i \in \mathbb{Z} \;\; \forall i \in \mathcal{I} \end{aligned}
$$

  its linear relaxation consists in the LP obtained by dropping the integrality constraints:

$$
(LP) \quad \begin{aligned} & \min \; c^T x \\ & Ax = b \\ & x \geq 0 \end{aligned}
$$

- Can we solve IP by solving LP? By rounding?

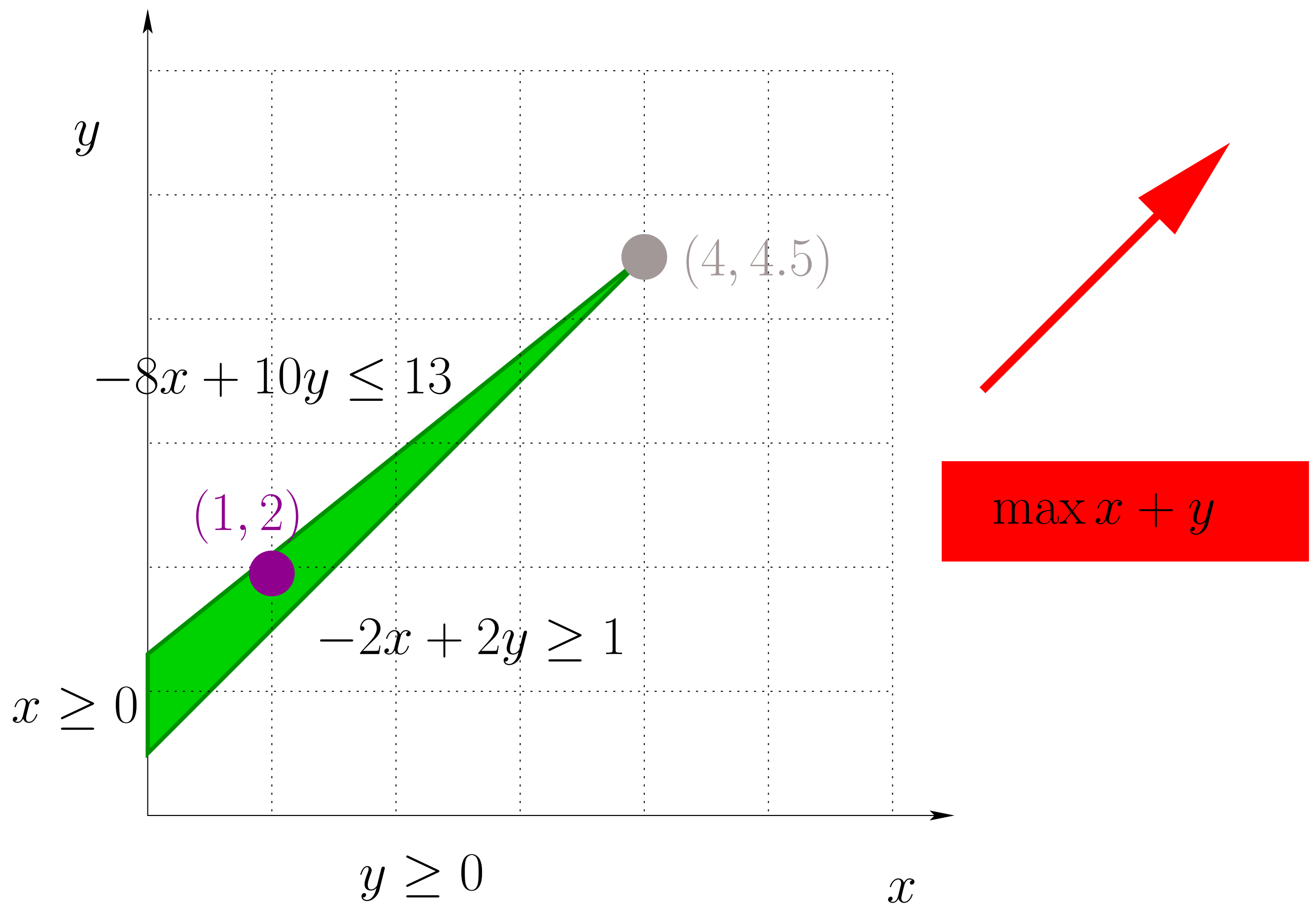
Branch & Bound (1)

- The optimal solution of

$$\begin{aligned} \max\ x + y & \\ -2x + 2y &\geq 1 \\ -8x + 10y &\leq 13 \\ x, y \in \mathbb{Z} & \end{aligned}$$

  is $(x, y) = (1, 2)$ with objective 3
- The optimal solution of its LP relaxation is $(x, y) = (4, 4.5)$ with objective 9.5
- No direct way of getting from $(x, y) = (4, 4.5)$ to $(x, y) = (1, 2)$ by rounding!
- Something more elaborate is needed: branch & bound

Branch & Bound (2)



Branch & Bound (3)

- Assume integer variables have lower and upper bounds
- Let P_0 initial problem, $\mathrm{LP}(P_0)$ LP relaxation of P_0
- If in optimal solution of $\mathrm{LP}(P_0)$ all integer variables take integer values then it is also an optimal solution to P_0
- Else
  - Rounding the solution of $\mathrm{LP}(P_0)$ may yield to non-optimal or non-feasible solutions for P_0!
  - Let x_j be integer variable whose value β_j at optimal solution of $\mathrm{LP}(P_0)$ satisfies $\beta_j \notin \mathbb{Z}$. Define

$$P_1 := P_0 \wedge x_j \leq \lfloor \beta_j \rfloor$$
$$P_2 := P_0 \wedge x_j \geq \lceil \beta_j \rceil$$

  - Feasible solutions to P_0 = feasible solutions to P_1 or P_2

Branch & Bound (4)

- Let x_j be integer variable whose value β_j at optimal solution of $\mathrm{LP}(P_0)$ satisfies $\beta_j \notin \mathbb{Z}$.

  $$P_1 := P_0 \wedge x_j \leq \lfloor \beta_j \rfloor \qquad P_2 := P_0 \wedge x_j \geq \lceil \beta_j \rceil$$

  P_i can be solved recursively
- We can build a binary tree of subproblems whose leaves correspond to pending problems still to be solved
- Terminates as integer vars have finite bounds and, at each split, range of one var becomes strictly smaller
- If $\mathrm{LP}(P_i)$ has optimal solution where integer variables take integer values then solution is stored
- If $\mathrm{LP}(P_i)$ is infeasible then P_i can be discarded (pruned, fathomed)

Example (1)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
End
==================================================================

Status:     OPTIMAL
Objective:  obj = 8.5 (MAXimum)

   No. Column name  St   Activity     Lower bound   Upper bound
------ ------------ -- ------------- ------------- -------------
     1 x            B              4             0
     2 y            B            4.5             0
```

Example (2)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
y >= 5
End
================================================================
GLPSOL: GLPK LP/MIP Solver 4.38
...
PROBLEM HAS NO FEASIBLE SOLUTION
```

Example (3)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
y <= 4
End
====================================================================
Status:      OPTIMAL
Objective:   obj = 7.5 (MAXimum)

   No. Column name  St   Activity     Lower bound   Upper bound
------ ------------ -- ------------- ------------- -------------
     1 x            B            3.5             0
     2 y            NU             4             0             4
```

Example (4)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x >= 4
y <= 4
End
====================================================================
GLPSOL: GLPK LP/MIP Solver 4.38
...
PROBLEM HAS NO PRIMAL FEASIBLE SOLUTION
```

Example (5)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x <= 3
y <= 4
End
====================================================================

Status:     OPTIMAL
Objective:  obj = 6.7 (MAXimum)

   No. Column name  St   Activity     Lower bound   Upper bound
------ ------------ -- ------------- ------------- -------------
     1 x            NU             3             0             3
     2 y            B            3.7             0             4
```

Example (6)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x <= 3
y = 4
End
====================================================================
GLPSOL: GLPK LP/MIP Solver 4.38
...
PROBLEM HAS NO PRIMAL FEASIBLE SOLUTION
```

Example (7)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x <= 3
y <= 3
End
====================================================================

Status:     OPTIMAL
Objective:  obj = 5.5 (MAXimum)

   No. Column name  St   Activity     Lower bound   Upper bound
------ ------------ -- ------------- ------------- -------------
     1 x            B            2.5             0             3
     2 y            NU             3             0             3
```

Example (8)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x = 3
y <= 3
End
====================================================================
GLPSOL: GLPK LP/MIP Solver 4.38
...
PROBLEM HAS NO PRIMAL FEASIBLE SOLUTION
```

Example (9)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x <= 2
y <= 3
End
====================================================================

Status:     OPTIMAL
Objective:  obj = 4.9 (MAXimum)

   No. Column name  St   Activity     Lower bound   Upper bound
------ ------------ -- ------------- ------------- -------------
     1 x            NU             2             0             2
     2 y            B            2.9             0             3
```

Example (10)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x <= 2
y = 3
End
==================================================================
GLPSOL: GLPK LP/MIP Solver 4.38
...
PROBLEM HAS NO PRIMAL FEASIBLE SOLUTION
```

Example (11)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x <= 2
y <= 2
End
====================================================================

Status:     OPTIMAL
Objective:  obj = 3.5 (MAXimum)

   No. Column name  St   Activity     Lower bound   Upper bound
------ ------------ -- ------------- ------------- -------------
     1 x            B            1.5             0             2
     2 y            NU             2             0             2
```

Example (12)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x = 2
y <= 2
End
==================================================================
GLPSOL: GLPK LP/MIP Solver 4.38
...
PROBLEM HAS NO PRIMAL FEASIBLE SOLUTION
```

Example (13)

```
Max obj: x + y
Subject To
c1: -2 x +  2 y >= 1
c2: -8 x + 10 y <= 13
Bounds
x <= 1
y <= 2
End
====================================================================

Status:     OPTIMAL
Objective:  obj = 3 (MAXimum)

   No. Column name  St   Activity     Lower bound   Upper bound
------ ------------ -- ------------- ------------- -------------
     1 x            NU             1             0             1
     2 y            NU             2             0             2
```

Pruning in Branch & Bound

- We have already seen that if relaxation is infeasible, the problem can be pruned
- Now assume an (integral) solution has been found
- If solution has cost Z then any pending problem P_j whose relaxation has optimal value $> Z$ can be ignored

$$\text{cost}(P_j) \geq \text{cost}(\mathrm{LP}(P_j)) > Z$$

  The optimum will not be in any descendant of P_j!
- This pruning of the search tree has a huge impact on the efficiency of Branch & Bound

Unboundedness in Branch & Bound

- We assumed integer variables are bounded
- In mixed problems,
  we allow non-integer variables to be unbounded
- Assume $\mathrm{LP}(P_i)$ is unbounded. Then:
  - If in basic solution integer variables take integer values then the problem is unbounded
    (assuming that problem data are rational numbers)
  - Else we proceed recursively
    as if an optimal solution to $\mathrm{LP}(P_i)$ had been found.

    What's different wrt $\mathrm{LP}(P_i)$ having optimal solution?

Unboundedness in Branch & Bound

- We assumed integer variables are bounded
- In mixed problems,
  we allow non-integer variables to be unbounded
- Assume $\mathrm{LP}(P_i)$ is unbounded. Then:
  - If in basic solution integer variables take integer values then the problem is unbounded
    (assuming that problem data are rational numbers)
  - Else we proceed recursively
    as if an optimal solution to $\mathrm{LP}(P_i)$ had been found.

    What's different wrt $\mathrm{LP}(P_i)$ having optimal solution?
    If $\mathrm{LP}(P_i)$ is unbounded then P_i cannot be pruned:
    no need to check this!

Branch & Bound: Algorithmic Description

$S := \{P_0\}$ /* set of pending problems */

$Z := +\infty$ /* best cost found so far */

while $S \neq \emptyset$ do

  remove P from S; solve $\mathrm{LP}(P)$

  if $\mathrm{LP}(P)$ is feasible then

    Let β be basic solution obtained after solving $\mathrm{LP}(P)$

    if β satisfies integrality constraints then

      if β is optimal for $\mathrm{LP}(P)$ then

        if cost$(\beta) < Z$ then store β; update Z

      else return UNBOUNDED

    else

      if β is optimal for $\mathrm{LP}(P) \wedge P$ can be pruned then continue

      Let x_j be integer variable such that $\beta_j \notin \mathbb{Z}$

      $S := S \quad \cup \quad \{P \wedge x_j \leq \lfloor \beta_j \rfloor, \quad P \wedge x_j \geq \lceil \beta_j \rceil\}$

return Z

Heuristics in Branch & Bound

- Possible choices in Branch & Bound
  - Choosing a pending problem
    - Depth-first search
    - Breadth-first search
    - Best-first search (select node with best cost value)
  - Choosing a branching variable
    - That closest to halfway two integer values
    - That with least cost coefficient
    - That which is important in the model (0-1 variable)
    - That which is biggest in a variable ordering
- No known strategy is best for all problems!

Remarks on Branch & Bound

- If integer variables not bounded, B&B may not terminate

$$\min \; 0$$
$$1 \leq 3x - 3y \leq 2$$
$$x, y \in \mathbb{Z}$$

  is infeasible but B&B loops forever looking for solutions!

- New problems need not be solved from scratch but starting from optimal solution of parent problem
- Dual Simplex Method can be used:
  dual feasibility preserved if change bounds of basic vars
- Often Dual Simplex needs few iterations to obtain an optimal solution to new problem (reoptimization)

Lower Bounding Procedures

- Pruning at a node is achieved here by solving the LP relaxation with dual simplex
- But there exist other procedures for giving lower bounds on best objective value: Lagrangian relaxation

$$(MIP) \begin{array}{l} \min\ c^T x \\ Ax \le b \\ \ell \le x \le u \\ x_i \in \mathbb{Z}\ \ \forall i \in \mathcal{I} \end{array} \Rightarrow (LG) \begin{array}{l} \min\ c^T x + \mu(Ax - b) \\ \ell \le x \le u \\ x_i \in \mathbb{Z}\ \ \forall i \in \mathcal{I} \end{array} \quad \text{with } \mu \ge 0$$

  - LG is a relaxation: gives lower bound on MIP
  - LG can be trivially solved
  - For good μ, LG is as good as LP relax. but cheaper
- Concrete problems have ad-hoc lower bounding procs

Cutting Planes (1)

- Let us consider a MIP of the form

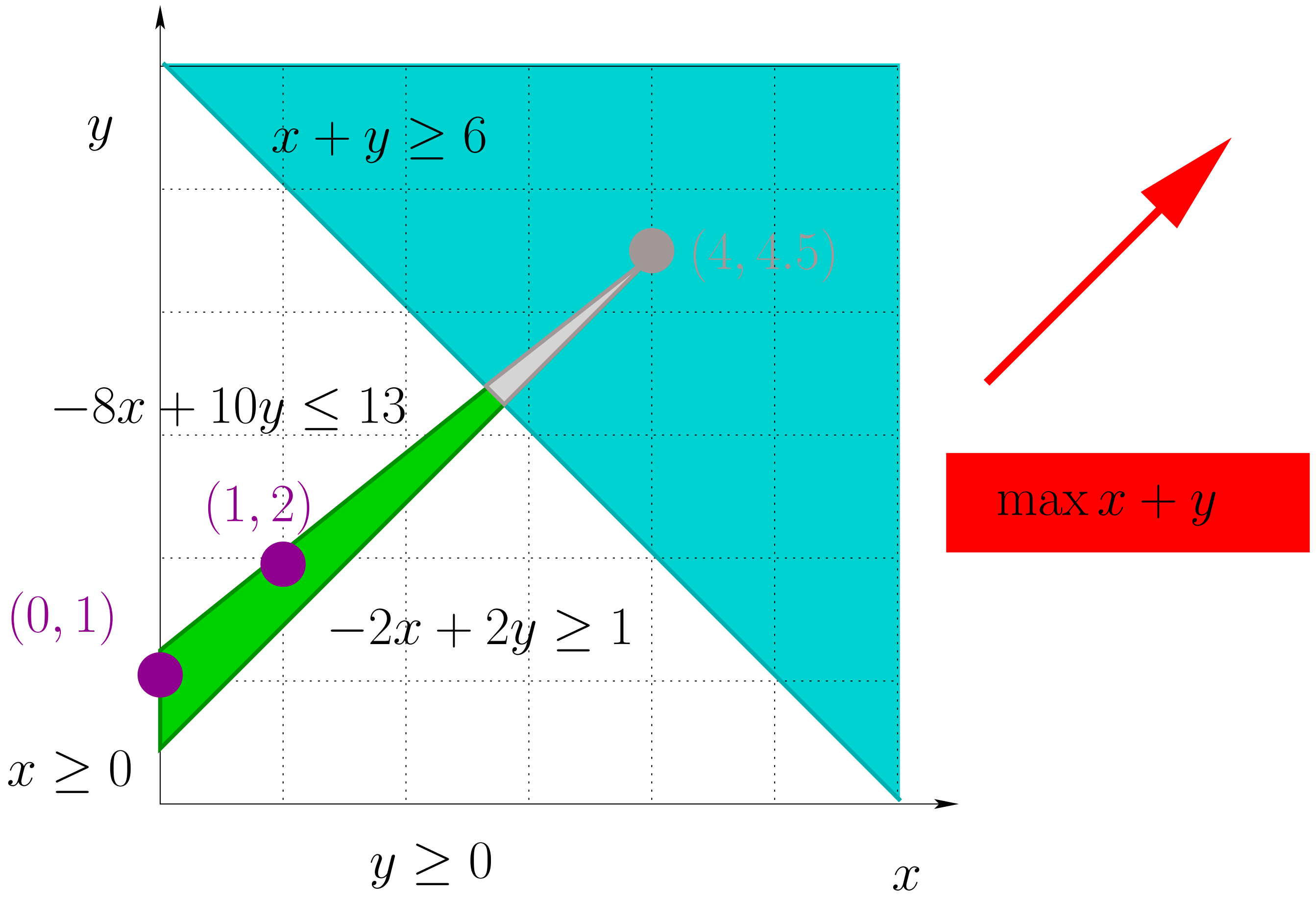
$$\begin{array}{l}\min \; c^T x \\ x \in S\end{array} \quad \text{where } S = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{l} Ax = b \\ \ell \le x \le u \\ x_i \in \mathbb{Z} \;\; \forall i \in \mathcal{I} \end{array} \right\}$$

  and its linear relaxation

$$\begin{array}{l}\min \; c^T x \\ x \in P\end{array} \quad \text{where } P = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{l} Ax = b \\ \ell \le x \le u \end{array} \right\}$$

- Let β be such that $\beta \in P$ but $\beta \notin S$.
  A cut for β is a linear inequality $\hat{a}^T x \le \hat{b}$ such that $\hat{a}^T \beta > \hat{b}$ and $\gamma \in S$ implies $\hat{a}^T \gamma \le \hat{b}$

Cutting Planes (2)



Using Cuts for Solving MIP

- Let $\hat{a}^T x \leq \hat{b}$ be a cut. Then the MIP

$$\begin{array}{l} \min \ c^T x \\ x \in S' \end{array} \quad \text{where } S' = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{l} Ax = b \\ \hat{a}^T x \leq \hat{b} \\ \ell \leq x \leq u \\ x_i \in \mathbb{Z} \ \ \forall i \in \mathcal{I} \end{array} \right\}$$

has the same set of feasible solutions S but its LP relaxation is strictly more constrained

- Rather than splitting into subproblems as in Branch & Bound, one can add the cut and solve the relaxation
- Used together with Branch & Bound: Branch & Cut
If after adding cuts no solution is found, then branch

Gomory Cuts (1)

- There are several techniques for deriving cuts
- Some are problem-specific (e.g, travelling salesman)
- Here we will see a generic technique: Gomory cuts
- Let us consider a tableau with a row of the form

$$x_i = \omega_i + \sum_{j \in \mathcal{R}} a_{ij} x_j \qquad (i \in \mathcal{B})$$

  Let β be an associated basic solution such that

  1. $i \in \mathcal{I}$
  2. $\beta(x_i) \notin \mathbb{Z}$
  3. For all $j \in \mathcal{R}$ we have $\beta(x_j) = \ell_j$ or $\beta(x_j) = u_j$

- Can think that it is the optimal tableau of the relaxation

Gomory Cuts (2)

- Let $\delta = \beta(x_i) - \lfloor \beta(x_i) \rfloor$. Then $0 < \delta < 1$ (assumption 2)
- By assumption 3, no non-basic variable is free
- Let $\mathcal{R}' = \mathcal{R} \cap \{j \mid \ell_j < u_j\}$ set of non-basic non-fixed vars
- Let $\mathcal{L} = \{j \in \mathcal{I} \cap \mathcal{R}' \mid \beta(x_j) = \ell_j\}$
- Let $\mathcal{U} = \{j \in \mathcal{I} \cap \mathcal{R}' \mid \beta(x_j) = u_j\}$
- Let $x \in S$. Then $x_i \in \mathbb{Z}$ and

$$x_i = \omega_i + \sum_{j \in \mathcal{R}} a_{ij} x_j$$

- Since β is basic solution

$$\beta(x_i) = \omega_i + \sum_{j \in \mathcal{R}} a_{ij} \beta(x_j)$$

Gomory Cuts (3)

$$x_i = \omega_i + \sum_{j \in \mathcal{R}} a_{ij} x_j$$
$$\beta(x_i) = \omega_i + \sum_{j \in \mathcal{R}} a_{ij} \beta(x_j)$$

- Subtracting

$$\begin{aligned} x_i - \beta(x_i) &= \sum_{j \in \mathcal{R}} a_{ij}(x_j - \beta(x_j)) \\ &= \sum_{j \in \mathcal{L}} a_{ij}(x_j - \ell_j) - \sum_{j \in \mathcal{U}} a_{ij}(u_j - x_j) \end{aligned}$$

- Finally

$$x_i - \lfloor \beta(x_i) \rfloor = \delta + \sum_{j \in \mathcal{L}} a_{ij}(x_j - \ell_j) - \sum_{j \in \mathcal{U}} a_{ij}(u_j - x_j)$$

- Let us define

$$\mathcal{L}^+ = \{j \in \mathcal{L} \mid a_{ij} \geq 0\} \qquad \mathcal{L}^- = \{j \in \mathcal{L} \mid a_{ij} < 0\}$$
$$\mathcal{U}^+ = \{j \in \mathcal{U} \mid a_{ij} \geq 0\} \qquad \mathcal{U}^- = \{j \in \mathcal{L} \mid a_{ij} < 0\}$$

Gomory Cuts (4)

$$x_i - \lfloor \beta(x_i) \rfloor = \delta + \sum_{j\in\mathcal{L}} a_{ij}(x_j - \ell_j) - \sum_{j\in\mathcal{U}} a_{ij}(u_j - x_j)$$

- Assume $\sum_{j\in\mathcal{L}} a_{ij}(x_j - \ell_j) - \sum_{j\in\mathcal{U}} a_{ij}(u_j - x_j) \geq 0$. Then

$$\delta + \sum_{j\in\mathcal{L}} a_{ij}(x_j - \ell_j) - \sum_{j\in\mathcal{U}} a_{ij}(u_j - x_j) \geq 1$$

$$\sum_{j\in\mathcal{L}^+} a_{ij}(x_j - \ell_j) - \sum_{j\in\mathcal{U}^-} a_{ij}(u_j - x_j) \geq 1 - \delta$$

$$\sum_{j\in\mathcal{L}^+} \frac{a_{ij}}{1-\delta}(x_j - \ell_j) + \sum_{j\in\mathcal{U}^-} \left(\frac{-a_{ij}}{1-\delta}\right)(u_j - x_j) \geq 1$$

Moreover $\sum_{j\in\mathcal{L}^-} \left(\frac{-a_{ij}}{\delta}\right)(x_j - \ell_j) + \sum_{j\in\mathcal{U}^+} \frac{a_{ij}}{\delta}(u_j - x_j) \geq 0$

Gomory Cuts (5)

$$x_i - \lfloor \beta(x_i) \rfloor = \delta + \sum_{j\in\mathcal{L}} a_{ij}(x_j - \ell_j) - \sum_{j\in\mathcal{U}} a_{ij}(u_j - x_j)$$

- Assume $\sum_{j\in\mathcal{L}} a_{ij}(x_j - \ell_j) - \sum_{j\in\mathcal{U}} a_{ij}(u_j - x_j) < 0$. Then

$$\delta + \sum_{j\in\mathcal{L}} a_{ij}(x_j - \ell_j) - \sum_{j\in\mathcal{U}} a_{ij}(u_j - x_j) \le 0$$

$$- \sum_{j\in\mathcal{L}^-} a_{ij}(x_j - \ell_j) + \sum_{j\in\mathcal{U}^+} a_{ij}(u_j - x_j) \ge \delta$$

$$\sum_{j\in\mathcal{L}^-} \left(\frac{-a_{ij}}{\delta}\right)(x_j - \ell_j) + \sum_{j\in\mathcal{U}^+} \frac{a_{ij}}{\delta}(u_j - x_j) \ge 1$$

Moreover $\sum_{j\in\mathcal{L}^+} \frac{a_{ij}}{1-\delta}(x_j - \ell_j) + \sum_{j\in\mathcal{U}^-} \left(\frac{-a_{ij}}{1-\delta}\right)(u_j - x_j) \ge 0$

Gomory Cuts (6)

In any case

$$\sum_{j\in\mathcal{L}^-}\left(\frac{-a_{ij}}{\delta}\right)(x_j-\ell_j)+$$
$$\sum_{j\in\mathcal{U}^+}\frac{a_{ij}}{\delta}(u_j-x_j)+$$
$$\sum_{j\in\mathcal{L}^+}\frac{a_{ij}}{1-\delta}(x_j-\ell_j)+$$
$$\sum_{j\in\mathcal{U}^-}\left(\frac{-a_{ij}}{1-\delta}\right)(u_j-x_j)\geq 1$$

for any $x\in S$.
However, β does not satisfy this inequality (set $x_j=\ell_j$ for $j\in\mathcal{L}$, and $x_j=u_j$ $j\in\mathcal{U}$)

Ensuring All Vertices Are Integer (1)

- Let us assume A, b have coefficients in $\mathbb{Z}$
- Sometimes it is possible to ensure for an IP that all vertices of the relaxation are integer
- For instance, when the matrix A is totally unimodular: the determinant of every square submatrix is 0 or ± 1
- Sufficient condition: property K
  - Each element of A is 0 or ± 1
  - No more than two non-zeros appear in each columm
  - Rows can be partitioned in two subsets R_1 and R_2 s.t.
    - If a column contains two non-zeros of the same sign, one element is in each of the subsets
    - If a column contains two non-zeros of different signs, both elements belong to the same subset

Assignment Problem

- $m = \#$ of workers $= \#$ of tasks
- Each worker must be assigned to exactly one task
- Each task is to be performed by exactly one worker
- $c_{ij} =$ cost when worker i performs task j

Assignment Problem

- $m = \#$ of workers $= \#$ of tasks
- Each worker must be assigned to exactly one task
- Each task is to be performed by exactly one worker
- $c_{ij} =$ cost when worker i performs task j

$$x_{ij} = \begin{cases} 1 & \text{if worker } i \text{ performs task } j \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{array}{ll} \min \ \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} & \\ \sum_{i=1}^{n} x_{ij} = 1 & \forall j \in \{1, \ldots, m\} \\ \sum_{j=1}^{n} x_{ij} = 1 & \forall i \in \{1, \ldots, m\} \\ x_{ij} \in \{0, 1\} & \forall i, j \in \{1, \ldots, m\} \end{array}$$

- This problem satisfies property K

Ensuring All Vertices Are Integer (2)

- Several kinds of IP's satisfy property K:
  - Assignment
  - Transportation
  - Maximum flow
  - Shortest path
  - ...
- Usually specialized network algorithms are more efficient for these problems than simplex techniques
- But simplex techniques are general and can be used if no implementation of network algorithms is available