DM545

Linear and Integer Programming

Lecture 13
Branch and Bound

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

1. Branch and Bound

2. Preprocessing

Exam

- Tilladt Håndscanner/digital pen og ordbogsprogrammet fra ordbogen.com
- Ikke tilladt at anvende digitalt kamera eller webcam o. lign. metoder for at digitalisere sin besvarelse
- Du afleverer efter fristen og kun en gang
- Exam Monitor er et lille program, som logger, hvilke programmer du afvikler på din computer under eksamen, samtidig med at din skærm optages. https://em.sdu.dk/
- Internet

  > *Internet er ikke tilladt ved eksamener på NAT, men undtagelsesvis til denne eksamen er det tilladt, at benytte følgende webside http://www.imada.sdu.dk/~marco/DM545/ og siderne linket derfra. Det er ikke tilladt at benytte andre sider*

- Vejledning og templates snart tilgænglig fra kurset web siden ved afsnittet Assessment
- Kom vel forberedet, bring noget at drikke og spise

To come

- Two weeks left
- This week: two lectures + joint training class on Wednesday
- Next week: two exercise classes + one lecture.
- Question time? Thursday 31st at 9:00?

Outline

1. Branch and Bound

2. Preprocessing

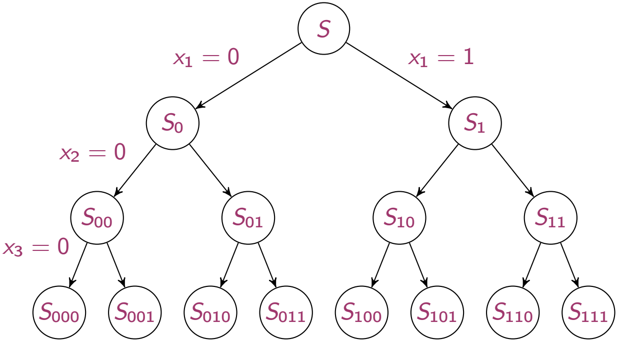
Branch and Bound

- Consider the problem $z = \max\{c^T x : x \in S\}$
- Divide and conquer: let $S = S_1 \cup \ldots \cup S_k$ be a decomposition of S into smaller sets, and let $z^k = \max\{c^T x : x \in S_k\}$ for $k = 1, \ldots, K$. Then $z = \max_k z^k$

Branch and Bound

- Consider the problem $z = \max\{c^T x : x \in S\}$
- Divide and conquer: let $S = S_1 \cup \ldots \cup S_k$ be a decomposition of S into smaller sets, and let $z^k = \max\{c^T x : x \in S_k\}$ for $k = 1, \ldots, K$. Then $z = \max_k z^k$

For instance if $S \subseteq \{0,1\}^3$ the enumeration tree is:



Bounding

Let's consider a maximization problem (gurobi's default is minimization)

- Let $\overline{z}^k$ be an upper bound on z^k (dual bound)
- Let $\underline{z}^k$ be a lower bound on z^k (primal bound)
- ($\underline{z}^k \leq z^k \leq \overline{z}^k$)

Bounding

Let's consider a maximization problem (gurobi's default is minimization)

- Let $\overline{z}^k$ be an upper bound on z^k (dual bound)
- Let $\underline{z}^k$ be a lower bound on z^k (primal bound)
- $(\underline{z}^k \leq z^k \leq \overline{z}^k)$
- $\underline{z} =$

Bounding

Let's consider a maximization problem (gurobi's default is minimization)

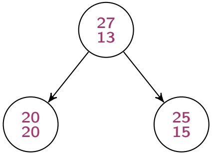
- Let $\overline{z}^k$ be an upper bound on z^k (dual bound)
- Let $\underline{z}^k$ be a lower bound on z^k (primal bound)
- $(\underline{z}^k \leq z^k \leq \overline{z}^k)$
- $\underline{z} = \max_k \underline{z}^k$ is a lower bound on z
- $\overline{z} =$

Bounding

Let's consider a maximization problem (gurobi's default is minimization)

- Let $\overline{z}^k$ be an upper bound on z^k (dual bound)
- Let $\underline{z}^k$ be a lower bound on z^k (primal bound)
- $(\underline{z}^k \leq z^k \leq \overline{z}^k)$
- $\underline{z} = \max_k \underline{z}^k$ is a lower bound on z
- $\overline{z} = \max_k \overline{z}^k$ is an upper bound on z

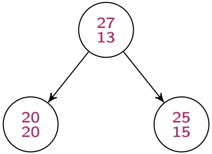
Pruning



$\overline{z} =$

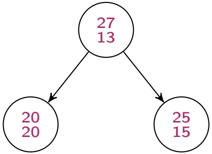
$\underline{z} =$

Pruning



$\overline{z} = 25$
$\underline{z} = 20$
pruned by optimality

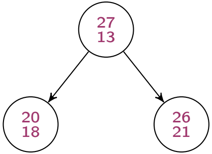
Pruning



$\overline{z} = 25$

$\underline{z} = 20$

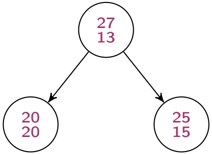
pruned by optimality



$\overline{z} =$

$\underline{z} =$

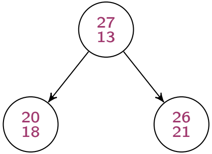
Pruning



$\overline{z} = 25$

$\underline{z} = 20$

pruned by optimality

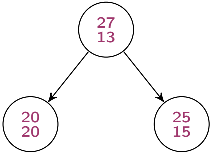


$\overline{z} = 26$

$\underline{z} = 21$

pruned by bounding

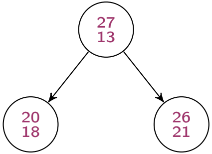
Pruning



$\overline{z} = 25$

$\underline{z} = 20$

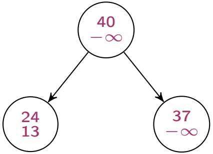
pruned by optimality



$\overline{z} = 26$

$\underline{z} = 21$

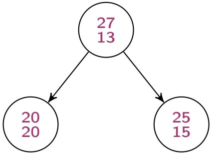
pruned by bounding



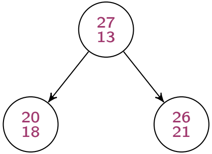
$\overline{z} =$

$\underline{z} =$

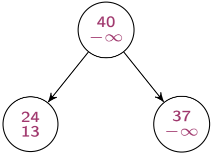
Pruning



$\overline{z} = 25$
$\underline{z} = 20$
pruned by optimality

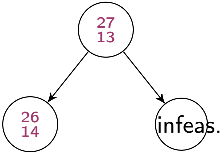


$\overline{z} = 26$
$\underline{z} = 21$
pruned by bounding



$\overline{z} = 37$
$\underline{z} = 13$
nothing to prune

Pruning



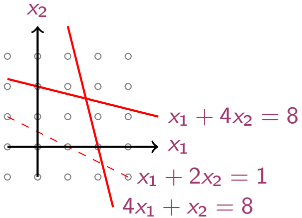
$\overline{z} = 26$

$\underline{z} = 14$

pruned by infeasibility

Example

$$
\begin{aligned}
\max \quad & x_1 + 2x_2 \\
& x_1 + 4x_2 \leq 8 \\
& 4x_1 + x_2 \leq 8 \\
& x_1, x_2 \geq 0, \text{integer}
\end{aligned}
$$

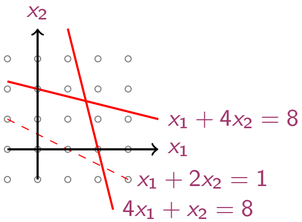


- Solve LP

```
|   | x1 | x2 | x3 | x4 | -z | b |
|---+----+----+----+----+----+---|
|   |  1 |  4 |  1 |  0 |  0 | 8 |
|   |  4 |  1 |  0 |  1 |  0 | 8 |
|---+----+----+----+----+----+---|
|   |  1 |  2 |  0 |  0 |  1 | 0 |
```

Example

$$
\begin{aligned}
\max \quad & x_1 + 2x_2 \\
& x_1 + 4x_2 \leq 8 \\
& 4x_1 + x_2 \leq 8 \\
& x_1, x_2 \geq 0, \text{integer}
\end{aligned}
$$



- Solve LP

	x1	x2	x3	x4	-z	b
	1	4	1	0	0	8
	4	1	0	1	0	8
	1	2	0	0	1	0

	x1	x2	x3	x4	-z	b
I'=I-II'	0	15/4	1	-1/4	0	6
II'=1/4II	1	1/4	0	1/4	0	2
III'=III-II'	0	7/4	0	-1/4	0	-2

- continuing

```
|                 | x1 | x2 | x3    | x4    | -z | b        |
|-----------------+----+----+-------+-------+----+---------|
| I'=4/15I        |  0 |  1 | 4/15  | -1/15 |  0 | 24/15    |
| II'=II-1/4I'    |  1 |  0 | -1/15 | 4/15  |  0 | 24/15    |
|-----------------+----+----+-------+-------+----+---------|
| III'=III-7/4I'  |  0 |  0 | -7/15 | -3/5  |  1 | -2-14/5 |
```

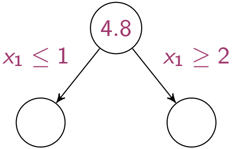
$x_2 = 1 + 3/5 = 1.6$
$x_1 = 8/5$
The optimal solution will not be more than $2 + 14/5 = 4.8$

- continuing

```
|                 | x1 | x2 | x3     | x4     | -z | b        |
|-----------------+----+----+--------+--------+----+----------|
| I'=4/15I        |  0 |  1 | 4/15   | -1/15  |  0 | 24/15    |
| II'=II-1/4I'    |  1 |  0 | -1/15  | 4/15   |  0 | 24/15    |
|-----------------+----+----+--------+--------+----+----------|
| III'=III-7/4I'  |  0 |  0 | -7/15  | -3/5   |  1 | -2-14/5  |
```

$x_2 = 1 + 3/5 = 1.6$
$x_1 = 8/5$
The optimal solution will not be more than $2 + 14/5 = 4.8$

- Both variables are fractional, we pick one of the two:

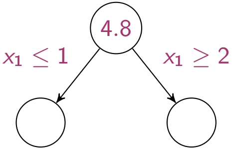


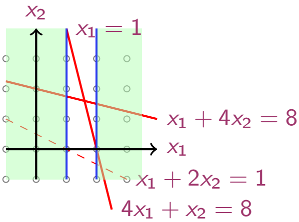
- continuing

	x1	x2	x3	x4	-z	b
I'=4/15I	0	1	4/15	-1/15	0	24/15
II'=II-1/4I'	1	0	-1/15	4/15	0	24/15
III'=III-7/4I'	0	0	-7/15	-3/5	1	-2-14/5

$x_2 = 1 + 3/5 = 1.6$
$x_1 = 8/5$
The optimal solution will not be more than $2 + 14/5 = 4.8$

- Both variables are fractional, we pick one of the two:





- Let's consider first the left branch:

- Let's consider first the left branch:

	x1	x2	x3	x4	x5	-z	b
	1	0	0	0	1	0	1
	0	1	4/15	-1/15	0	0	24/15
	1	0	-1/15	4/15	0	0	24/15
	0	0	-7/15	-3/5	0	1	-24/5

- Let's consider first the left branch:

	x1	x2	x3	x4	x5	-z	b
	1	0	0	0	1	0	1
	0	1	4/15	-1/15	0	0	24/15
	1	0	-1/15	4/15	0	0	24/15
	0	0	-7/15	-3/5	0	1	-24/5

	x1	x2	x3	x4	x5	b	-z
I'=I-III	0	0	1/15	-4/15	1	0	-9/15
	0	1	4/15	-1/15	0	0	24/15
	1	0	-1/15	4/15	0	0	24/15
	0	0	-7/15	-3/5	0	1	-24/5

- Let's consider first the left branch:

	x1	x2	x3	x4	x5	-z	b
	1	0	0	0	1	0	1
	0	1	4/15	-1/15	0	0	24/15
	1	0	-1/15	4/15	0	0	24/15
	0	0	-7/15	-3/5	0	1	-24/5

	x1	x2	x3	x4	x5	b	-z
I'=I-III	0	0	1/15	-4/15	1	0	-9/15
	0	1	4/15	-1/15	0	0	24/15
	1	0	-1/15	4/15	0	0	24/15
	0	0	-7/15	-3/5	0	1	-24/5

	x1	x2	x3	x4	x5	b	-z
I'=-15/4I	0	0	-1/4	1	-15/4	0	9/4
II'=II-1/4I	0	1	15/60	0	-1/4	0	7/4
III'=III+I	1	0	0	0	1	0	1
	0	0	-37/60	0	-9/4	1	-90/20

always a b term negative after branching:

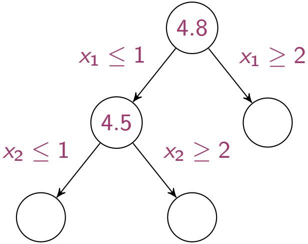
$b_1 = \lfloor \bar{b}_3 \rfloor$

$\bar{b}_1 = \lfloor \bar{b}_3 \rfloor - b_3 < 0$

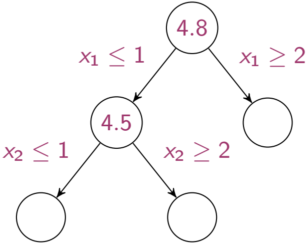
Dual simplex:

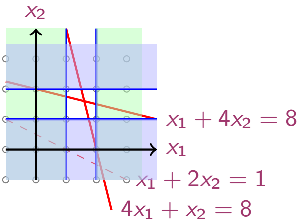
$\min_j \{ | \frac{c_j}{a_{ij}} | \ : \ a_{ij} < 0 \}$

- Let's branch again

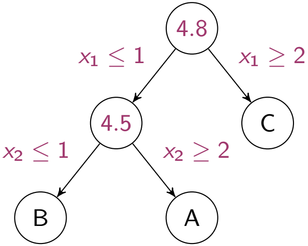


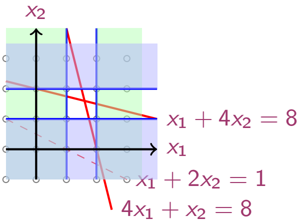
- Let's branch again





- Let's branch again





We have three open problems. Which one we choose next?
Let's take A.

	x1	x2	x3	x4	x5	x6	b	-z
	0	-1	0	0	0	1	0	-2
	0	0	-1/4	1	-15/4		0	9/4
	0	1	15/60	0	-1/4		0	7/4
	1	0	0	0	1		0	1
	0	0	-37/60	0	-9/4		1	-9/2

	x1	x2	x3	x4	x5	x6	b	-z
	0	-1	0	0	0	1	0	-2
	0	0	-1/4	1	-15/4		0	9/4
	0	1	15/60	0	-1/4		0	7/4
	1	0	0	0	1		0	1
	0	0	-37/60	0	-9/4		1	-9/2

	x1	x2	x3	x4	x5	x6	b	-z
III+I	0	0	1/4	0	-1/4	1	0	-1/4
	0	0	-1/4	1	-15/4		0	9/4
	0	1	15/60	0	-1/4		0	7/4
	1	0	0	0	1		0	1
	0	0	-37/60	0	-9/4		1	-9/2

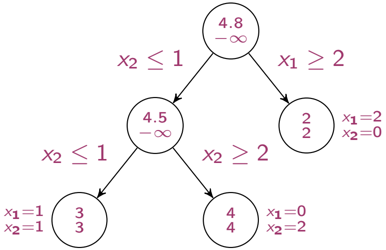
continuing we find:

$x_1 = 0$

$x_2 = 2$

$OPT = 4$

The final tree:



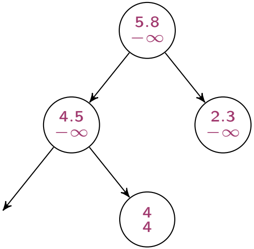
The optimal solution is 4.

Pruning

Pruning:

1. by optimality: $z^k = \max\{c^T x : x \in S^k\}$

2. by bound $\overline{z}^k \leq \underline{z}$
   Example:



3. by infeasibility $S^k = \emptyset$

B&B Components

Bounding:

1. LP relaxation
2. Lagrangian relaxation
3. Combinatorial relaxation
4. Duality

B&B Components

Bounding:

1. LP relaxation
2. Lagrangian relaxation
3. Combinatorial relaxation
4. Duality

Branching:

$$S_1 = S \cap \{x : x_j \leq \lfloor \bar{x}_j \rfloor\}$$
$$S_2 = S \cap \{x : x_j \geq \lceil \bar{x}_j \rceil\}$$

thus the current optimum is not feasible either in S_1 or in S_2.

B&B Components

Bounding:

1. LP relaxation
2. Lagrangian relaxation
3. Combinatorial relaxation
4. Duality

Branching:

$$S_1 = S \cap \{x : x_j \leq \lfloor \bar{x}_j \rfloor\}$$
$$S_2 = S \cap \{x : x_j \geq \lceil \bar{x}_j \rceil\}$$

thus the current optimum is not feasible either in S_1 or in S_2.
Which variable to choose?
Eg: Most fractional variable $\arg\max_{j \in C} \min\{f_j, 1 - f_j\}$

B&B Components

Bounding:

1. LP relaxation
2. Lagrangian relaxation
3. Combinatorial relaxation
4. Duality

Branching:

$$S_1 = S \cap \{x : x_j \leq \lfloor \bar{x}_j \rfloor\}$$
$$S_2 = S \cap \{x : x_j \geq \lceil \bar{x}_j \rceil\}$$

thus the current optimum is not feasible either in S_1 or in S_2.
Which variable to choose?
Eg: Most fractional variable $\arg\max_{j \in C} \min\{f_j, 1 - f_j\}$

Choosing Node for Examination from the list of active (or open):

- Depth First Search (a good primal sol. is good for pruning + easier to reoptimize by just adding a new constraint)
- Best Bound First: (eg. largest upper: $\overline{z}^s = \max_k \overline{z}^k$ or largest lower - to die fast)
- Mixed strategies

Reoptimizing: dual simplex

Updating the Incumbent: when new best feasible solution is found:

$$\underline{z} = \max\{\underline{z}, 4\}$$

Store the active nodes: bounds + optimal basis (remember the revised simplex!)

Enhancements

- Preprocessor: constraint/problem/structure specific
  tightening bounds
  redundant constraints
  variable fixing: eg: $\max\{\mathbf{c}^T\mathbf{x} : A\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$
  fix $x_j = l_j$ if $c_j < 0$ and $a_{ij} > 0$ for all i
  fix $x_j = u_j$ if $c_j > 0$ and $a_{ij} < 0$ for all i

Enhancements

- Preprocessor: constraint/problem/structure specific
  tightening bounds
  redundant constraints
  variable fixing: eg: $\max\{\mathbf{c}^T\mathbf{x} : A\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$
  fix $x_j = l_j$ if $c_j < 0$ and $a_{ij} > 0$ for all i
  fix $x_j = u_j$ if $c_j > 0$ and $a_{ij} < 0$ for all i
- Priorities: establish the next variable to branch

Enhancements

- Preprocessor: constraint/problem/structure specific
  tightening bounds
  redundant constraints
  variable fixing: eg: $\max\{\mathbf{c}^T\mathbf{x} : A\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$
  fix $x_j = l_j$ if $c_j < 0$ and $a_{ij} > 0$ for all i
  fix $x_j = u_j$ if $c_j > 0$ and $a_{ij} < 0$ for all i
- Priorities: establish the next variable to branch
- Special ordered sets SOS (or generalized upper bound GUB)

$$\sum_{j=1}^{k} x_j = 1 \qquad x_j \in \{0, 1\}$$

instead of: $S_0 = S \cap \{\mathbf{x} : x_j = 0\}$ and $S_1 = S \cap \{\mathbf{x} : x_j = 1\}$
$\{\mathbf{x} : x_j = 0\}$ leaves $k - 1$ possibilities
$\{\mathbf{x} : x_j = 1\}$ leaves only 1 possibility
hence tree unbalanced

here: $S_1 = S \cap \{\mathbf{x} : x_{j_i} = 0, i = 1..r\}$ and $S_2 = S \cap \{x : x_{j_i} = 0, i = r + 1, .., k\}$,
$r = \min\{t : \sum_{i=1}^{t} x^*_{j_i} \geq \frac{1}{2}\}$

- Cutoff value: a user-defined primal bound to pass to the system.
- Simplex strategies: simplex is good for reoptimizing but for large models interior points methods may work best.

- Cutoff value: a user-defined primal bound to pass to the system.
- Simplex strategies: simplex is good for reoptimizing but for large models interior points methods may work best.
- Strong branching: extra work to decide more accurately on which variable to branch:
  1. choose a set C of fractional variables
  2. reoptimize for each of them (in case for limited iterations)
  3. $\overline{z}_j^{\downarrow}, \overline{z}_j^{\uparrow}$ (dual bound of down and up branch)

$$j^* = \arg\min_{j \in C} \max\{\overline{z}_j^{\downarrow}, \overline{z}_j^{\uparrow}\}$$

  ie, choose variable with largest decrease of dual bound, eg UB for max

There are four common reasons because integer programs can require a significant amount of solution time:

1. There is lack of node throughput due to troublesome linear programming node solves.
2. There is lack of progress in the best integer solution, i.e., the upper bound.
3. There is lack of progress in the best lower bound.
4. There is insufficient node throughput due to numerical instability in the problem data or excessive memory usage.

There are four common reasons because integer programs can require a significant amount of solution time:

1. There is lack of node throughput due to troublesome linear programming node solves.
2. There is lack of progress in the best integer solution, i.e., the upper bound.
3. There is lack of progress in the best lower bound.
4. There is insufficient node throughput due to numerical instability in the problem data or excessive memory usage.

For 2) or 3) the gap best feasible-dual bound is large:

$$\text{gap} = \frac{|\text{Primal bound} - \text{Dual bound}|}{\text{Primal bound} + \epsilon} \cdot 100$$

- heuristics for finding feasible solutions (generally NP-complete problem)
- find better lower bounds if they are weak: addition of cuts, stronger formulation, branch and cut
- Branch and cut: a B&B algorithm with cut generation at all nodes of the tree. (instead of reoptimizing, do as much work as possible to tighten)

  Cut pool: stores all cuts centrally
  Store for active node: bounds, basis, pointers to constraints in the cut pool that apply at the node

Relative Optimality Gap

In CPLEX:

$$\text{gap} = \frac{|\text{best dual bound} - \text{best integer}|}{|\text{best integer} + 10^{-11}|}$$

In SCIP and MIPLIB standard:

$$\text{gap} = \frac{pb - db}{\inf\{|z|, z \in [db, pb]\}} \cdot 100 \qquad \text{for a minimization problem}$$

(if $pb \geq 0$ and $db \geq 0$ then $\frac{pb-db}{db}$)
if $db = pb = 0$ then $\text{gap} = 0$
if no feasible sol found or $db \leq 0 \leq pb$ then the gap is not computed.

Last standard avoids problem of non decreasing gap if we go through zero

```
   3186  2520     -666.6217  4096      956.6330     -667.2010  1313338  169.74%
   3226  2560     -666.6205  4097      956.6330     -667.2010  1323797  169.74%
   3266  2600     -666.6201  4095      956.6330     -667.2010  1335602  169.74%
Elapsed real time = 2801.61 sec. (tree size = 77.54 MB, solutions = 2)
*  3324+ 2656                         -125.5775     -667.2010  1363079  431.31%
   3334  2668     -666.5811  4052     -125.5775     -667.2010  1370748  431.31%
   3380  2714     -666.5799  4017     -125.5775     -667.2010  1388391  431.31%
   3422  2756     -666.5791  4011     -125.5775     -667.2010  1403440  431.31%
```

Advanced Techniques

We did not treat:

- LP: Dantzig Wolfe decomposition
- LP: Column generation
- LP: Delayed column generation
- IP: Branch and Price
- LP: Benders decompositions
- LP: Lagrangian relaxation

Outline

Preprocessing rules

Consider $S = \{\mathbf{x} : a_0 x_0 + \sum_{j=1}^{n} a_j x_j \leq b, l_j \leq x_j \leq u_j, j = 0..n\}$

- Bounds on variables.
  If $a_0 > 0$ then:

$$x_0 \leq \left(b - \sum_{j:a_j>0} a_j l_j - \sum_{j:a_j<0} a_j u_j \right) / a_0$$

  and if $a_0 < 0$ then

$$x_0 \geq \left(b - \sum_{j:a_j>0} a_j l_j - \sum_{j:a_j<0} a_j u_j \right) / a_0$$

Preprocessing rules

Consider $S = \{\mathbf{x} : a_0 x_0 + \sum_{j=1}^{n} a_j x_j \leq b, l_j \leq x_j \leq u_j, j = 0..n\}$

- Bounds on variables.
  If $a_0 > 0$ then:

$$x_0 \leq \left(b - \sum_{j:a_j>0} a_j l_j - \sum_{j:a_j<0} a_j u_j \right) / a_0$$

  and if $a_0 < 0$ then

$$x_0 \geq \left(b - \sum_{j:a_j>0} a_j l_j - \sum_{j:a_j<0} a_j u_j \right) / a_0$$

- Redundancy. The constraint $\sum_{j=0}^{n} a_j x_j \leq b$ is redundant if

$$\sum_{j:a_j>0} a_j u_j + \sum_{j:a_j<0} a_j l_j \leq b$$

- Infeasibility: $S = \emptyset$ if (swapping lower and upper bounds from previous case)

$$\sum_{j:a_j>0} a_j l_j + \sum_{j:a_j<0} a_j u_j > b$$

- Infeasibility: $S = \emptyset$ if (swapping lower and upper bounds from previous case)

$$\sum_{j:a_j>0} a_j l_j + \sum_{j:a_j<0} a_j u_j > b$$

- Variable fixing. For a max problem in the form

$$\max\{\mathbf{c}^T\mathbf{x} : A\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$$

if $\forall i = 1..m : a_{ij} \geq 0, c_j < 0$ then fix $x_j = l_j$
if $\forall i = 1..m : a_{ij} < 0, c_j > 0$ then fix $x_j = u_j$

- Infeasibility: $S = \emptyset$ if (swapping lower and upper bounds from previous case)

$$\sum_{j:a_j>0} a_j l_j + \sum_{j:a_j<0} a_j u_j > b$$

- Variable fixing. For a max problem in the form

$$\max\{\mathbf{c}^T\mathbf{x} : A\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$$

  if $\forall i = 1..m : a_{ij} \geq 0, c_j < 0$ then fix $x_j = l_j$
  if $\forall i = 1..m : a_{ij} < 0, c_j > 0$ then fix $x_j = u_j$

- Integer variables:

$$\lceil l_j \rceil \leq x_j \leq \lfloor u_j \rfloor$$

- Infeasibility: $S = \emptyset$ if (swapping lower and upper bounds from previous case)

$$\sum_{j:a_j>0} a_j l_j + \sum_{j:a_j<0} a_j u_j > b$$

- Variable fixing. For a max problem in the form

$$\max\{\mathbf{c}^T\mathbf{x} : A\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$$

if $\forall i = 1..m : a_{ij} \geq 0, c_j < 0$ then fix $x_j = l_j$
if $\forall i = 1..m : a_{ij} < 0, c_j > 0$ then fix $x_j = u_j$

- Integer variables:

$$\lceil l_j \rceil \leq x_j \leq \lfloor u_j \rfloor$$

- Binary variables. Probing: add a constraint, eg, $x_2 = 0$ and check what happens

Example

$$
\begin{aligned}
\max\ & 2x_1 + x_2 - x_3 \\
\texttt{R1}: & 5x_1 - 2x_2 + 8x_3 \le 15 \\
\texttt{R2}: & 8x_1 + 3x_2 - x_3 \ge 9 \\
\texttt{R3}: & x_1 + x_2 + x_3 \le 6 \\
& 0 \le x_1 \le 3 \\
& 0 \le x_2 \le 1 \\
& x_3 \ge 1
\end{aligned}
$$

Example

$$
\begin{aligned}
\max\ & 2x_1 + x_2 - x_3 \\
\mathtt{R1}:\ & 5x_1 - 2x_2 + 8x_3 \leq 15 \\
\mathtt{R2}:\ & 8x_1 + 3x_2 - x_3 \geq 9 \\
\mathtt{R3}:\ & x_1 + x_2 + x_3 \leq 6 \\
& 0 \leq x_1 \leq 3 \\
& 0 \leq x_2 \leq 1 \\
& x_3 \geq 1
\end{aligned}
$$

$$\mathtt{R1}: 5x_1 \leq 15 + 2x_2 - 8x_3 \leq 15 + 2 \cdot \overbrace{1}^{u_2} - 8 \cdot \overbrace{1}^{l_3} = 9 \quad \rightsquigarrow x_1 \leq 9/5$$

$$8x_3 \leq 15 + 2x_2 - 5x_1 \leq 15 + 2 \cdot 1 - 5 \cdot 0 = 17 \quad \rightsquigarrow x_3 \leq 17/8$$

$$2x_2 \geq 5x_1 + 8x_3 - 15 \geq 5 \cdot 0 + 8 \cdot 1 = -7 \quad \rightsquigarrow x_2 \geq -7/2, x_2 \geq 0$$

$$\mathtt{R2}: 8x_1 \geq 9 - 3x_2 + x_3 \geq 9 - 3 + 1 = 7 \quad \rightsquigarrow x_1 \geq 7/8$$

$$\mathtt{R1}: 8x_3 \geq 15 + 2x_2 - 5x_1 \leq 15 + 2 - 5 \cdot 7/8 = 101/8 \quad \rightsquigarrow x_3 \leq 101/64$$

$\mathtt{R3}: x_1 + x_2 + x_3 \leq 9/5 + 1 + 101/64 < 6$ Hence $\mathtt{R3}$ is redundant

Example

$$
\begin{aligned}
\max\ & 2x_1 + x_2 - x_3 \\
\mathtt{R1}:\ & 5x_1 - 2x_2 + 8x_3 \leq 15 \\
\mathtt{R2}:\ & 8x_1 + 3x_2 - x_3 \geq 9 \\
\mathtt{R3}:\ & x_1 + x_2 + x_3 \leq 6 \\
& 0 \leq x_1 \leq 3 \\
& 0 \leq x_2 \leq 1 \\
& x_3 \geq 1
\end{aligned}
$$

$$
\begin{aligned}
\mathtt{R1}: 5x_1 &\leq 15 + 2x_2 - 8x_3 \leq 15 + 2 \cdot \overbrace{1}^{u_2} - 8 \cdot \overbrace{1}^{l_3} = 9 && \rightsquigarrow x_1 \leq 9/5 \\
8x_3 &\leq 15 + 2x_2 - 5x_1 \leq 15 + 2 \cdot 1 - 5 \cdot 0 = 17 && \rightsquigarrow x_3 \leq 17/8 \\
2x_2 &\geq 5x_1 + 8x_3 - 15 \geq 5 \cdot 0 + 8 \cdot 1 = -7 && \rightsquigarrow x_2 \geq -7/2, x_2 \geq 0
\end{aligned}
$$

Example

$$
\begin{aligned}
\max\ & 2x_1 + x_2 - x_3 \\
\mathtt{R1}:\ & 5x_1 - 2x_2 + 8x_3 \leq 15 \\
\mathtt{R2}:\ & 8x_1 + 3x_2 - x_3 \geq 9 \\
\mathtt{R3}:\ & x_1 + x_2 + x_3 \leq 6 \\
& 0 \leq x_1 \leq 3 \\
& 0 \leq x_2 \leq 1 \\
& x_3 \geq 1
\end{aligned}
$$

$$\mathtt{R1}: 5x_1 \leq 15 + 2x_2 - 8x_3 \leq 15 + 2 \cdot \overbrace{1}^{u_2} - 8 \cdot \overbrace{1}^{l_3} = 9 \quad \rightsquigarrow x_1 \leq 9/5$$

$$8x_3 \leq 15 + 2x_2 - 5x_1 \leq 15 + 2 \cdot 1 - 5 \cdot 0 = 17 \quad \rightsquigarrow x_3 \leq 17/8$$

$$2x_2 \geq 5x_1 + 8x_3 - 15 \geq 5 \cdot 0 + 8 \cdot 1 = -7 \quad \rightsquigarrow x_2 \geq -7/2, x_2 \geq 0$$

$$\mathtt{R2}: 8x_1 \geq 9 - 3x_2 + x_3 \geq 9 - 3 + 1 = 7 \quad \rightsquigarrow x_1 \geq 7/8$$

$$\mathtt{R1}: 8x_3 \geq 15 + 2x_2 - 5x_1 \leq 15 + 2 - 5 \cdot 7/8 = 101/8 \quad \rightsquigarrow x_3 \leq 101/64$$

Example

$$
\begin{aligned}
\max\ & 2x_1 + x_2 - x_3 \\
\mathtt{R1}: & 5x_1 - 2x_2 + 8x_3 \leq 15 \\
\mathtt{R2}: & 8x_1 + 3x_2 - x_3 \geq 9 \\
\mathtt{R3}: & x_1 + x_2 + x_3 \leq 6 \\
& 0 \leq x_1 \leq 3 \\
& 0 \leq x_2 \leq 1 \\
& x_3 \geq 1
\end{aligned}
$$

$$\mathtt{R1}: 5x_1 \leq 15 + 2x_2 - 8x_3 \leq 15 + 2 \cdot \overbrace{1}^{u_2} - 8 \cdot \overbrace{1}^{l_3} = 9 \qquad \rightsquigarrow x_1 \leq 9/5$$

$$8x_3 \leq 15 + 2x_2 - 5x_1 \leq 15 + 2 \cdot 1 - 5 \cdot 0 = 17 \qquad \rightsquigarrow x_3 \leq 17/8$$

$$2x_2 \geq 5x_1 + 8x_3 - 15 \geq 5 \cdot 0 + 8 \cdot 1 = -7 \qquad \rightsquigarrow x_2 \geq -7/2, x_2 \geq 0$$

$$\mathtt{R2}: 8x_1 \geq 9 - 3x_2 + x_3 \geq 9 - 3 + 1 = 7 \qquad \rightsquigarrow x_1 \geq 7/8$$

$$\mathtt{R1}: 8x_3 \geq 15 + 2x_2 - 5x_1 \leq 15 + 2 - 5 \cdot 7/8 = 101/8 \qquad \rightsquigarrow x_3 \leq 101/64$$

$\mathtt{R3}: x_1 + x_2 + x_3 \leq 9/5 + 1 + 101/64 < 6$ Hence R3 is redundant

Example

$$
\begin{aligned}
\max\ & 2x_1 + x_2 - x_3 \\
\mathtt{R1}: & 5x_1 - 2x_2 + 8x_3 \leq 15 \\
\mathtt{R2}: & 8x_1 + 3x_2 - x_3 \geq 9 \\
& 7/8 \leq x_1 \leq 9/5 \\
& 0 \leq x_2 \leq 1 \\
& 1 \leq x_3 \leq 101/64
\end{aligned}
$$

Example

$$
\begin{aligned}
\max\ & 2x_1 + x_2 - x_3 \\
\mathtt{R1} :\ & 5x_1 - 2x_2 + 8x_3 \leq 15 \\
\mathtt{R2} :\ & 8x_1 + 3x_2 - x_3 \geq 9 \\
& 7/8 \leq x_1 \leq 9/5 \\
& 0 \leq x_2 \leq 1 \\
& 1 \leq x_3 \leq 101/64
\end{aligned}
$$

Increasing x_2 makes constraints satisfied $\rightsquigarrow$ $x_2 = 1$
Decreasing x_3 makes constraints satisfied $\rightsquigarrow$ $x_3 = 1$

Example

$$
\begin{aligned}
\max\ & 2x_1 + x_2 - x_3 \\
\mathtt{R1} :\ & 5x_1 - 2x_2 + 8x_3 \leq 15 \\
\mathtt{R2} :\ & 8x_1 + 3x_2 - x_3 \geq 9 \\
& 7/8 \leq x_1 \leq 9/5 \\
& 0 \leq x_2 \leq 1 \\
& 1 \leq x_3 \leq 101/64
\end{aligned}
$$

Increasing x_2 makes constraints satisfied $\rightsquigarrow$ $x_2 = 1$
Decreasing x_3 makes constraints satisfied $\rightsquigarrow$ $x_3 = 1$

We are left with:

$$\max\{2x_1 : 7/8 \leq x_1 \leq 9/5\}$$

Preprocessing for Set Covering/Partitioning

1. if $e_i^T A = 0$ then the ith row can never be satisfied

$$\begin{bmatrix} 0 & 0 & \dots & 1 & \dots & 0 \end{bmatrix} \begin{bmatrix} & & \\ \hdashline 0 \dots 0 & \dots & 0 \\ \hdashline & & \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

Preprocessing for Set Covering/Partitioning

1. if $e_i^T A = 0$ then the ith row can never be satisfied

$$[0\;0 \dots 1 \dots 0] \begin{bmatrix} & & \\ \hdashline 0 \dots 0 & \dots & 0 \\ \hdashline & & \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

2. if $e_i^T A = e_k$ then $x_k = 1$ in every feasible solution

$$[0\;0 \dots 1 \dots 0] \left[\begin{array}{c:c:c} & & \\ \hdashline 0 \quad \dots & 1 & \dots \quad 0 \\ \hdashline & & \end{array}\right] = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

Preprocessing for Set Covering/Partitioning

1. if $e_i^T A = 0$ then the ith row can never be satisfied

$$\begin{bmatrix} 0 & 0 & \dots & 1 & \dots & 0 \end{bmatrix} \begin{bmatrix} & & & \\ \hdashline 0 \dots 0 & & \dots & 0 \\ \hdashline & & & \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

2. if $e_i^T A = e_k$ then $x_k = 1$ in every feasible solution

$$\begin{bmatrix} 0 & 0 & \dots & 1 & \dots & 0 \end{bmatrix} \begin{bmatrix} & & : & & \\ \hdashline 0 & \dots & : 1 : & \dots & 0 \\ \hdashline & & : & & \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

In SPP can remove all rows t with $a_{tk} = 1$ and set $x_j = 0$ (ie, remove cols) for all cols that cover t

3. if $e_t^T A \geq e_p^T A$ then we can remove row t, row p dominates row t (by covering p we cover t)

$$
\begin{array}{c}
\\ t \\ \\ p \\ \\
\end{array}
\begin{bmatrix}
 & & & & & \\
 & 1 & 1 & & 1 & \\
 & & & & & \\
 & 1 & & & 1 & \\
 & & & & &
\end{bmatrix}
$$

3. if $e_t^T A \geq e_p^T A$ then we can remove row t, row p dominates row t (by covering p we cover t)

$$
\begin{array}{c}
t \\
\\
p \\
\end{array}
\left[
\begin{array}{cccccc}
 & 1 & 1 & & 1 & \\
 & & & & & \\
 & 1 & & & 1 & \\
\end{array}
\right]
$$

In SPP we can remove all cols j: $a_{tj} = 1, a_{pj} = 0$

4. if $\sum_{j \in S} Ae_j = Ae_k$ and $\sum_{j \in S} c_j \leq c_k$ then we can cover the rows by Ae_k more cheaply with S and set $x_k = 0$
(Note, we cannot remove S if $\sum_{j \in S} c_j \geq c_k$)

$$
\left[
\begin{array}{c|ccc|c|c|c}
 & & 1 & & & 1 & \\
 & 1 & & & & 1 & \\
 & & & 1 & & 1 & \\
 & 0 & 0 & 0 & & 0 & \\
 & 1 & & & & 1 & \\
 & 0 & 0 & 0 & & 0 & \\
\end{array}
\right]
$$

Summary

1. Branch and Bound

2. Preprocessing