Dantzig-Wolfe Decomposition
– *Changing a problem ...*

Thomas Stidsen

thst@man.dtu.dk

DTU-Management
Technical University of Denmark

DTU

Outline

- General Background for Dantzig-Wolfe
- Mathematical background
- A 2-dim example
- Multi Commodity Flow problem (the last exercise)
- Block-Angular Structure

Dantzig-Wolfe

Historically:

- Dantzig-Wolfe decomposition was invented by Dantzig and Wolfe 1961.
- The method is so closely connected to column generation that they in some aspects may be considered to be identical.
- Dantzig-Wolfe and Column-Generation is one of the most used methods for practical problems.

Notice that column generation and Dantzig-Wolfe are used interchangeably ...

From Appendix A we have

Given the convex set $X = \{x | Ax \leq b\}$ can be represented by the extreme points and extreme rays of the convex set (Minkowski-Weyl's Theorem):

$$X = \{x | x = \sum_p \lambda^p \cdot \overline{x}^p + \sum_r \delta^r \cdot \overline{x}^r\}$$

where

$\sum_p \lambda^p = 1,\ \lambda^p \geq 0,\ \delta^r \geq 0$ and where

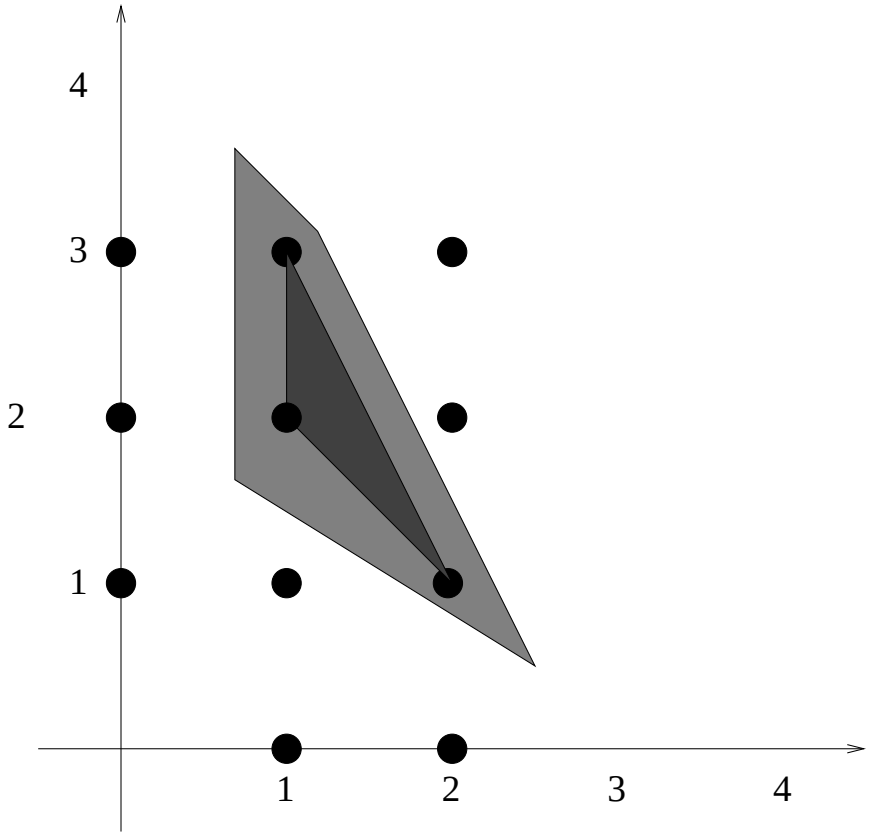
$p \in \{1, \ldots, P\},\ r \in \{1, \ldots, R\}$ See Appendix A, p.

638

Extreme rays

Good news: In theory we need extreme rays to represent arbitrary polyhedrons, but in reality we only encounter this in Dantzig-Wolfe decomposition very seldom. I have **never** seen a Dantzig-Wolfe decomposition where an extreme ray was necessary. Hence we will from here on simply assume that we can just use extreme points.

The importance of a good polytope



If we can achieve the smallest polytope in the figure, we can solve MIP problems with LP solvers !!!

Given a Linear program

Min:

$$c^T x$$

s.t.:

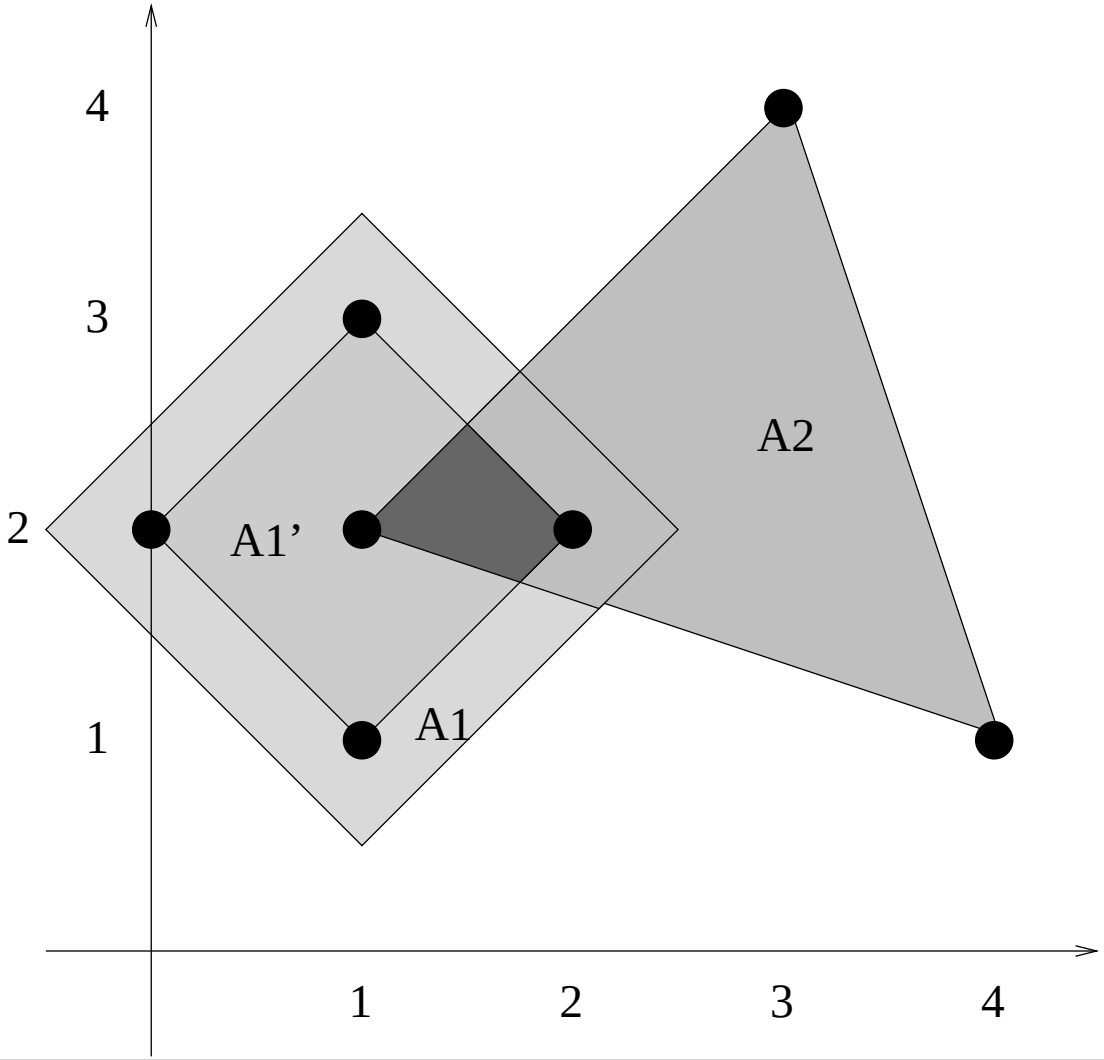
$$\begin{aligned} A_1 x &\geq b_1 \\ A_2 x &\geq b_2 \\ x &\geq 0 \end{aligned}$$

Changing representation

Dantzig-Wolfe decomposition is actually about **changing representation** from a set of constraints to a set of extreme points. The good question is now, how many constraints should be replaced by extreme points ?

- All: You tried that, in the first exercise, it does not really help ...
- None: Well that does not change the problem !!!
- Some: Yes, but why should that help ?

Visualized



The mathematical formulation

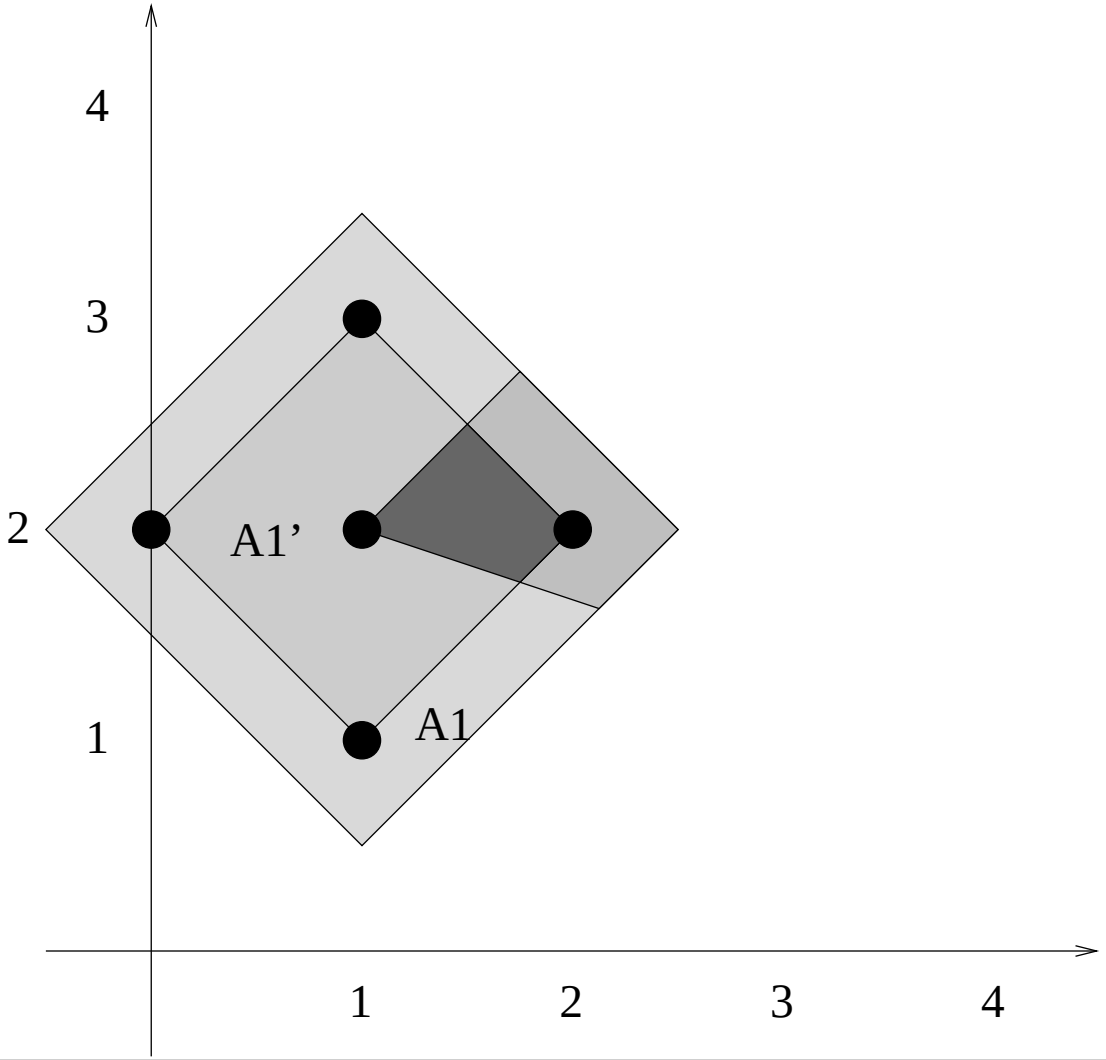
Max:

$$x + 2y$$

s.t.:

$$\begin{aligned}
-x + y &\leq \frac{5}{2} \quad (A1) \\
x + y &\leq \frac{9}{2} \quad (A1) \\
-x - y &\leq -\frac{1}{2} \quad (A1) \\
-x - y &\leq -\frac{3}{2} \quad (A1) \\
-x + y &\leq 1 \quad (A2) \\
2x + y &\leq 13 \quad (A2) \\
-x - 3y &\leq -7 \quad (A2)
\end{aligned}$$

Lets look at A1



Change representation

Now we represent *part*, the A2 part, of the constraints with the convex combination of extreme points and extreme arrays:

$X = \{x | x = \sum_p \lambda^p \cdot \overline{x}^p\}$

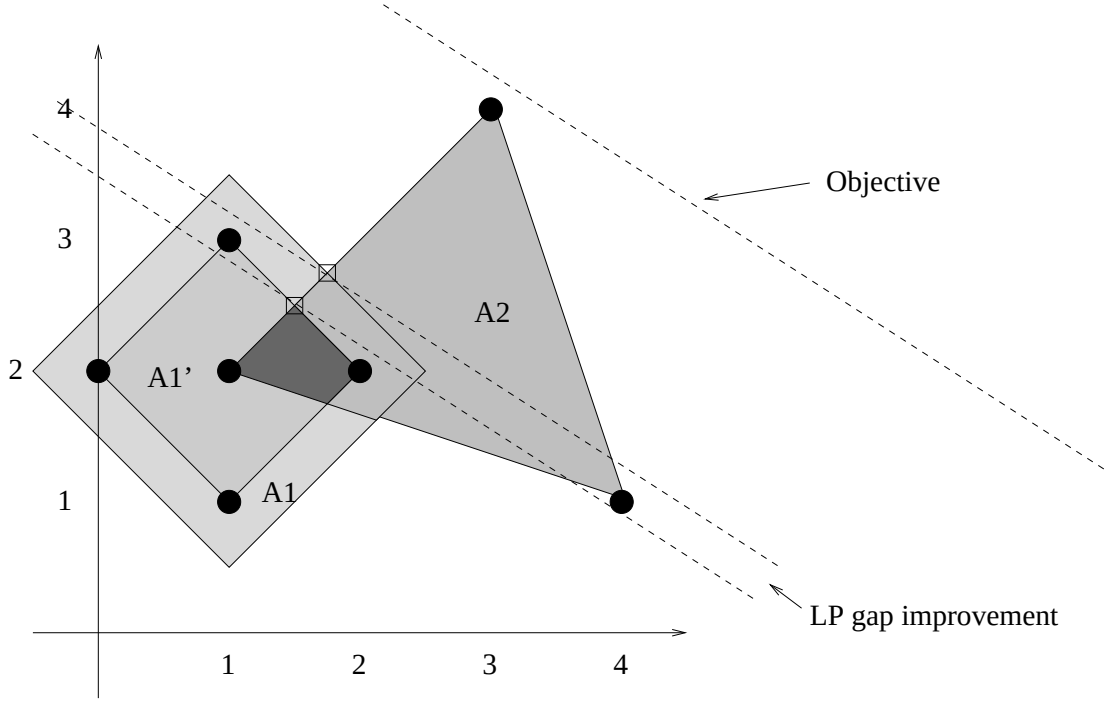
where the extreme points p define the set $X = \{x | A_2 x \geq b_2\}$, **BUT WHICH EXTREME POINTS ????**

Improving the bound of A1: A1'

$$
\begin{aligned}
-x + y &\leq \frac{5}{2} \quad (A1) \\
x + y &\leq \frac{9}{2} \quad (A1) \\
-x - y &\leq -\frac{1}{2} \quad (A1) \\
-x - y &\leq -\frac{3}{2} \quad (A1)
\end{aligned}
$$

$$
x, y \in R^{+}(A1) \quad \textit{OR} \quad x, y \in Z^{+}(A1')
$$

The LP improvement !



Insertion

Min:

$$c^T(\sum_p \lambda^p \cdot \overline{x}^p + \sum_r \delta^r \cdot \overline{x}^r)$$

s.t.:

$$\begin{aligned} A_2(\sum_p \lambda^p \cdot \overline{x}^p + \sum_r \delta^r \cdot \overline{x}^r) &\geq b_2 \\ \sum_p \lambda^p &= 1 \\ \lambda^p, \delta^r &\geq 0 \end{aligned}$$

Notice: x^p and x^r are now **constants** and λ^p and δ^r are now the variables

Insertion without extreme rays

Min:

$$c^T(\sum_p \lambda^p \cdot \overline{x}^p)$$

s.t.:

$$\begin{aligned} A_2(\sum_p \lambda^p \cdot \overline{x}^p) &\geq b_2 \\ \sum_p \lambda^p &= 1 \\ \lambda^p &\geq 0 \end{aligned}$$

Insertion without extreme rays, matrix version

Assume that:

- The A_2 matrix is a m_2 by n matrix, i.e. m_2 rows and n x variables.
- We replace the values of the x variables with the column vector of variables λ^p.
- We have a new matrix $\overline{X}$ with n rows and p columns

The transformed problem

Min:

$$c^T \lambda^p$$

s.t.:

$$\begin{aligned} A_2 \overline{X}\ \lambda^p &\geq b_2 \\ 1\ \lambda^p &= 1 \\ \lambda^p &\geq 0 \end{aligned}$$

Where

- Generally: $x = \overline{X}\lambda^p$.
- The matrix $AX = A_2\overline{X}$ is a matrix with m_2 rows and p columns (corresponding to the new variables.
- Each matrix element in the new matrix can be found as $ax_{m_2,p} = A_{1(m_1)}\overline{X}_{(p)}$
  - $A_{(m_2)}$: The m_2'th row of the A_2 matrix
  - $\overline{X}_{(p)}$: The p'th column of the $\overline{X}$ matrix

This means that we can calculate for each added column the new resulting column in the master problem

How can we find the extreme points of A_1 ?

We need two things:

- Satisfy the constraints of A_1 or A_1' (otherwise it will not be an extreme points)
- Calculate the reduced costs of the extreme point in the A_1 or A_1' constraints, based on the original costs *and* the dual variables from the A_2 part.

The sub-problem

Min:

$$(c - \pi \cdot A_2)x - \alpha$$

s.t.:

$$\begin{array}{ccc} A_1 \cdot x & \geq & b_2 \\ x, y \in R^+(A_1) & \textit{OR} & x, y \in Z^+(A_1') \end{array}$$

The sub-problem

Stopping criteria, if we cannot generate an extreme point:

$(c - \pi \cdot A_2)x - \alpha < 0$

Stopping criteria, if we cannot generate an extreme ray:

$(c - \pi \cdot A_2)x < 0$

If we cannot generate either an extreme point or an extreme ray then we have the optimal solution.

Of course, for maximization problems, the **reduced profits** for extreme points and extreme rays is required to be positive.

The column generation algorithm

Ensure feasibility of the master problem

repeat

SOLVE $min\{c^T x | A_2 X\ \lambda^p \geq b, 1\ \lambda^p = 1, \lambda^p \geq 0\}$

get π

SOLVE

$$min\{c_{red} = (c - \pi \cdot A_1)x - \alpha | A_1 \cdot x \geq b_1, \\ x, y \in R^+(A_1) \textbf{OR} x, y \in Z^+(A_1')\}$$

calculate new column data:

$$c_p = c^T x$$

$$ax_{m_2,p} = A_{2(m_2)}\overline{X}_{(p)}$$

until $c_{red} \geq 0$

Guideline when decomposing

The guideline when performing Dantzig-Wolfe decomposition:

- Find a problem which *has* Non-Integral property
- Find a way to solve the sub-problem fast.

Examples of the combinatorial sub-problems: The knapsack problem and the constrained shortest path problem.

Much more about this in the next lecture !

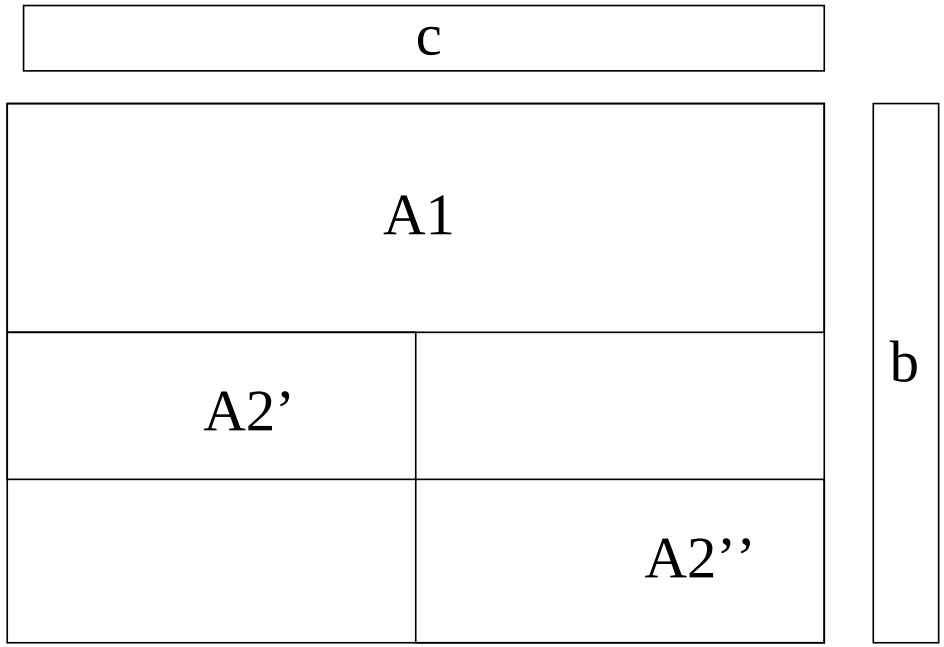
The Block-Angular Structure

We can use the structure in the following way:

- The variables are linked together by the A_1 matrix.
- If we seperate the matrixes and are able to solve the problems seperatly, we can solve the problem by solving the subproblem for the matrix'es A'_1 and A''_2

The Block-Angular Structure

Suppose the optimisation problem has the following structure Given two convex sets, in two dimensions:



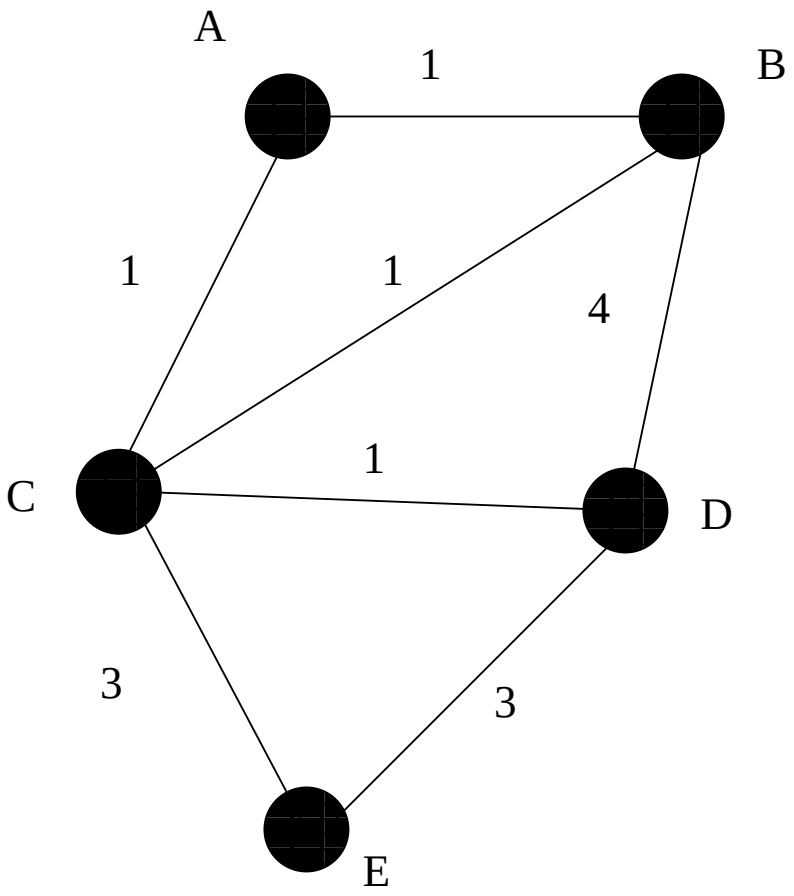
The Block-Angular Structure

- Notice that the two sub-problems **share** the prices π ...
- Sometimes it is beneficial to simply use a standard MIP solver.
- You have already experienced the division into several sub-problems in the multi-commodity flow exercise

Multi-Commodity flow

- Maximize the flow AE and BE through a network with limited capacity.
- An example could be: A producer of natural gas, production at A and B. Supply the customer, who does not care where the gas comes from.
- Actually this problem *can* be solved by the max flow algorithm (how ?). This formulation was chosen in order to make the problem simple.
- We will only consider *simple* paths.
- Orientation is tricky !

Multi-Commodity flow: The network



Demand1: A −> E

Demand2: B −> E

AE matrix

A_{AE}=

AB	1	1	1	1			
AC					1	1	1
BC	1		1				1
BD		1		1			1
CD			1	1		1	
CE	1			1	1		
DE		1	1			1	1

BE matrix

A_{BE}=

AB	1	1				
AC	1					
BC			1	1		
BD					1	1
CD		1		1		1
CE	1		1			1
DE		1		1	1	

c and b

$c_p = 1$

$b =$

AB	1
AC	1
BC	1
BD	4
CD	1
CE	3
DE	3

The link-path problem

Max:

$$c^T(x_p^{AE} + x_p^{BE})$$

s.t.:

$$\begin{aligned} A_{AE}x_p^{AE} + A_{BE}x_p^{BE} &\leq b \\ x_p^{AE}, x_p^{BE} &\geq 0 \end{aligned}$$

How did we get here ?

The arc-flow formulation

Max:

$$c^T(y^{AE} + y^{BE})$$

s.t.:

$$\sum_j x^{AE}_{(ij)} - \sum_j x^{AE}_{(ji)} = \begin{cases} y^{AE} & i = A \\ -y^{AE} & i = E \\ 0 & \text{otherwise} \end{cases} \quad \forall i$$

$$\sum_j x^{BE}_{(ij)} - \sum_j x^{BE}_{(ji)} = \begin{cases} y^{BE} & i = B \\ -y^{BE} & i = E \\ 0 & \text{otherwise} \end{cases} \quad \forall i$$

$$x^{AE}_{(ij)} + x^{AE}_{(ji)} + x^{BE}_{(ij)} + x^{BE}_{(ji)} \leq b \quad \forall \{ij\}$$

$$y^{AE}, y^{BE}, x^{AE}_{(ij)}, x^{BE}_{(ij)} \geq 0$$

Comments

- The arc-flow constraints correspond to a polytope ($A_1 \cdot x = b_1$) which is replaced with the path variables
- Notice that each *path variable* corresponds to a number of *arc flow variables*
- We have two separate subproblems ...
- We can easily calculate an arc-flow solution based on our paths ...

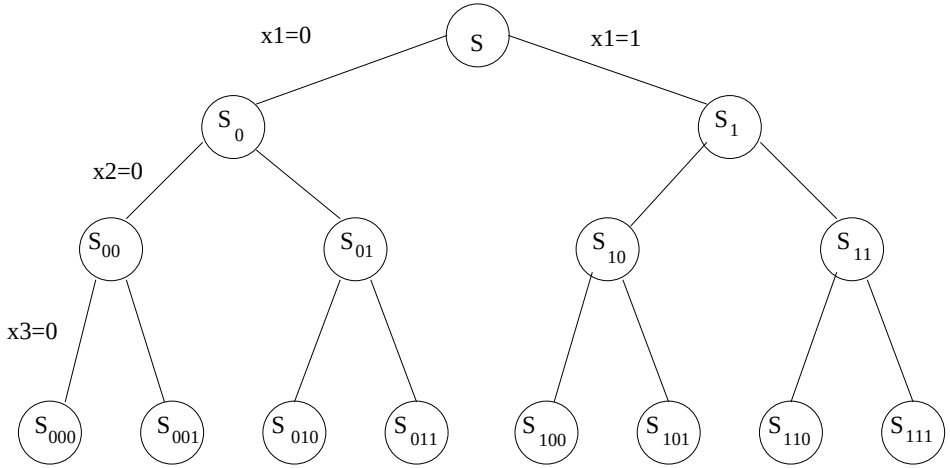
Integer Solutions

As I mentioned the last time, there are several methods:

- Rounding up, not always possible ....
- Solve the LP problem with column generation, and after the column generation algorithm has finished, the MIP model is solved using a standard solver.
- Use standard branching *in the original space*
- Branch and price, hard and quite time consuming ...

Branch and Price: Branching Problem

If we branch on the *transformed* variables, we have big problems in "down branch", because that variable is generated again.



Branch and Price: Branching in the *original var*

We *can* perform standard branching, but we have to do it in another variable space ! This I will talk more about in the next lecture.

Branching

So called Ryan-Foster branching can be applied (for set-partitioning problems)
Min:

$$\sum_j x_j$$

s.t.:

$$\sum_j a_{i,j} \cdot x_j = 1 \qquad \forall i$$

$$x_j \in \{0,1\}$$

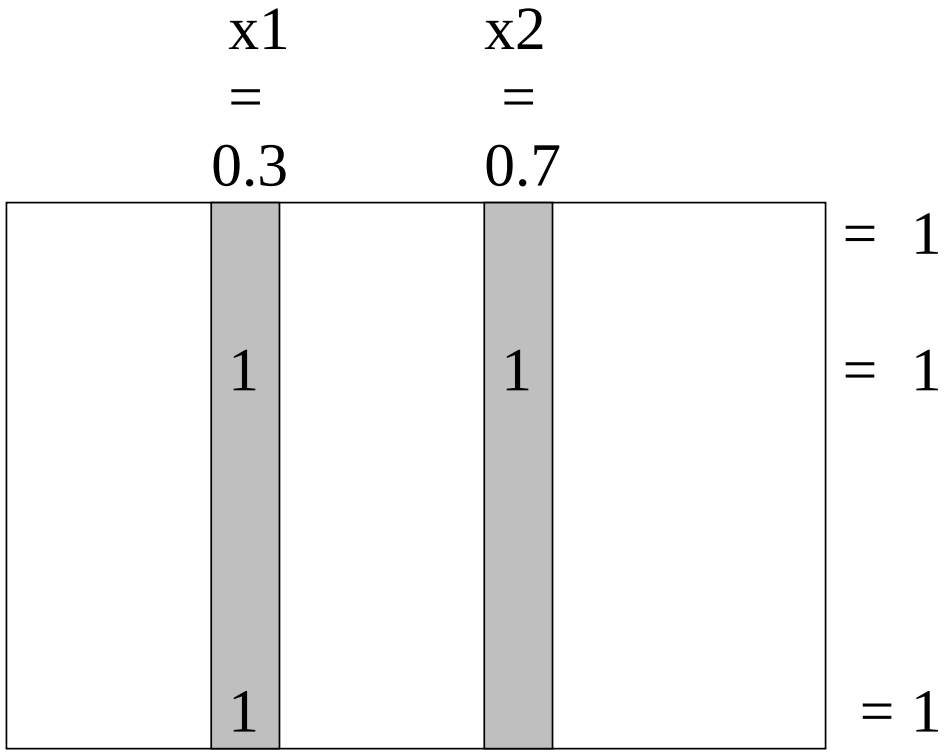
Further, notice that $a_{i,j} \in \{0,1\}$

Ryan-Foster Branching

The “end rule”: Each row will (in the optimal integer solution) be “covered” by **EXACTLY** one column (variable). When we have a fractional solution, this end rule is broken: At least two rows will be covered by two variables. So called Ryan-Foster branching can be applied (for set-partitioning problems)

Ryan-Foster Branching

Finding the branching cuts, find two variables which are fractional and which share at least one “cover”.



The branching rules

Given two rows i_1 and i_2 in one branch require $a_{i1,j'} = a_{i2,j'}$ for all variables, i.e. rows i_1 and i_2 are covered by the same variables, and in the other branch: $a_{i1,j'} \neq a_{i2,j'}$, i.e. rows i_1 and i_2 are covered by different variables.