

COLUMN GENERATION FOR SET ~~COVERING~~ PARTITIONING

Eg: Crew scheduling.

Use this approach when we have a very large number of sets (eg, millions).  
(hundreds of rows.)

min ~~min~~  $\sum_{j \in J} c_j x_j$

s.t.  $\sum_{j: i \in S_j} x_j = 1 \quad \forall i$  use ~~at least~~ <sup>exactly</sup> one set that contains item  $i$   
(<sup>exactly</sup> "at least one pairing for" each flight leg.)

$x_j$  binary

( $S_j$  corresponds to a column.  
Eg:  $S_j$  is a pairing, then the  $i \in S_j$  are the legs covered by that pairing)

LP relaxation: min  $\sum_{j=1}^n c_j x_j$

s.t.  $\sum_{j: i \in S_j} x_j = 1$

~~$x_j \geq 0$~~   
 $x_j \geq 0$

Dual problem: max  $\sum_{i=1}^m y_i$   ~~$\sum_{i=1}^m y_i$~~

s.t.  $\sum_{i \in S_j} y_i \leq c_j \quad \forall \text{ sets } S_j$

~~$x_j \geq 0$~~   ~~$x_j \geq 0$~~

Column generation:

Only work with a subset of the sets.

So  $x_j = 0$  (banned) for many sets.

Thus, when we've got an integer solution to this ~~and~~ restricted problem, need to check the ~~to~~ omitted sets.

Requires checking dual feasibility. Primal & (Checking reduced costs)

~~Some  $x_j = 0$  for these extra sets, we have  $x_j = 0$  by complementary slackness~~

Thus, we test:

$$|s \quad \sum_{i \in S_j} y_i \leq c_j \quad \forall \text{ omitted sets } S_j$$

Usually need some heuristic for this, and usually try to find

~~the~~ the most violated constraint. ~~For~~ For United Airlines,

this takes the largest part of ~~the~~ the computational effort.

Add these sets into the primal formulation, and repeat.

Branching:

We may obtain a fractional  $x$ .

Could branch on  $x_j = 1$  vs  $x_j = 0$

But this is far more restrictive than this, so two branches differ greatly in their solvability:  $x_j = 1$  branch "easy" (eliminate many items)  $x_j = 0$  branch "hard" (very similar to parent ~~problem~~ problem).

So look for a set of columns ~~splitting~~ and two rows  $i_1, i_2$  satisfying:

$$0 < \sum_{\substack{j: i_1 \in S_j \\ \text{and } i_2 \in S_j}} x_j < 1.$$

Such a set must exist (Prop. 11.3, Wolsey.)

Constraint matrix:  
 $i_1 [111111100000]$  etc  
 $i_2 [0000111111]$   
 add these components of  $x$ .

follows from being a bfs.

Then branch on:

items  $i_1$  and  $i_2$  in the same set

vs.

items  $i_1$  and  $i_2$  in different sets.

Not minimal, eg:  
 $\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$   
 $x = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$

So throw out all columns containing exactly one of  $i_1, i_2$  in pricing subproblem, add constraint  $y_{i_1} = y_{i_2}$  (ie, only consider appropriate  $S_j$ ).

So throw out all columns containing both of  $i_1, i_2$  In pricing subproblem, add constraint  $y_{i_1} + y_{i_2} \leq 1$  (ie, only consider appropriate  $S_j$ ).

This is a bfs. Last two rows don't work, but first two do. Start with  $x_i = \frac{1}{2}$  giving last two rows.

Also all these are not the  $y$  from the pricing step, they are binary variables indicating whether an object is in a set.

Note: on both branches, keep columns which contain neither  $i_1$  nor  $i_2$ .

This fails, but then this leads to rows 1 & 2, because they don't have  $x_1$  and  $x_2$  in common

Leads to better balance in the subproblems.

Rule due to Ryan & Foster, 1981.