(RE)USING DUAL INFORMATION IN MILP

ICS 2015

SAS
THE POWER TO KNOW.

Philipp M. Christophel
Menal Güzelsoy
Imre Pólik

AGENDA

1. LP Duality
2. Reusing Dual Information
   - Pruning
   - Bound tightening
   - Variable selection
   - IP bound
   - Dual branching
3. Computational Results

WEAK-STRONG DUALITY

- Consider the following LP (bounds are finite):

$$
\begin{aligned}
z^* = \min\ & c^T x \\
& L \le Ax \le U \\
& \ell \le x \le u
\end{aligned}
$$

- Assume z^* is finite. Any *dual solution* y satisfies:

$$
\begin{aligned}
z^* &\ge \sum_{i:y_i>0} L_i y_i + \sum_{i:y_i<0} U_i y_i \quad + \sum_{i:c_i - A_i^T y>0} \ell_i (c_i - A_i^T y) + \sum_{i:c_i - A_i^T y<0} u_i (c_i - A_i^T y) \\
&= z_y(L, U, \ell, u)
\end{aligned}
$$

- There exists an optimal dual solution y^* that satisfies $z^* = z_{y^*}(L, U, \ell, u)$

The following are equivalent:

- The following system is infeasible:

$$L \leq Ax \leq U$$
$$\ell \leq x \leq u$$

- There is a dual ray r to prove it:

$$z_r(L, U, \ell, u) =$$
$$\sum_{i:r_i>0} L_i r_i + \sum_{i:r_i<0} U_i r_i + \sum_{i:-A_i^T r>0} \ell_i(-A_i^T r) + \sum_{i:-A_i^T r<0} u_i(-A_i^T r) > 0$$

All bounds are finite, so r is unconstrained.

DUAL INFORMATION IN BRANCH AND BOUND

Standard usage

- Reduced cost fixing (local and root)
- Conflict analysis

New ways

- Keep the dual rays and solutions
- Prune nodes
- Reduced cost fixing globally
- Dual ray fixing (no incumbent or bound needed)
- Variable selection
- Get tree bounds

NODE PRUNING

Using dual solution y:

- y is globally valid and gives a lower bound (weak duality) at any other node
- Compute dual objective value $z_y(L, U, \ell, u) =$

$$\sum_{i:y_i>0} L_i y_i + \sum_{i:y_i<0} U_i y_i + \sum_{i:c_i - A_i^T y>0} \ell_i (c_i - A_i^T y) + \sum_{i:c_i - A_i^T y<0} u_i (c_i - A_i^T y)$$

- Let $\bar{z}$ be upper bound/incumbent objective value
- Prune node if $z_y(L, U, \ell, u) \geq \bar{z}$

NODE PRUNING

Using dual ray r:

- r can be used to detect infeasible nodes
- Compute ray objective value $z_r(L, U, \ell, u) =$

$$\sum_{i:r_i>0} L_i r_i + \sum_{i:r_i<0} U_i r_i + \sum_{i:-A_i^T r>0} \ell_i(-A_i^T r) + \sum_{i:-A_i^T r<0} u_i(-A_i^T r)$$

- Prune node if $z_r(L, U, \ell, u) > 0$
- What if $z_y < \bar{z}$ and $z_r \leq 0$?

REUSING DUAL INFORMATION

COLUMN BOUND TIGHTENING

- Using dual solution y: Reduced cost fixing
  $z_y(L, U, \ell, u) =$

$$\sum_{i:y_i>0} L_i y_i + \sum_{i:y_i<0} U_i y_i + \sum_{i:c_i - A_i^T y>0} \ell_i(c_i - A_i^T y) + \sum_{i:c_i - A_i^T y<0} u_i(c_i - A_i^T y)$$

- If $c_i - A_i^T y > 0$, then increasing ℓ_i also increases z_y
- If there is $\ell_i < \ell_i' < u_i$ such that

$$z_y(L, U, \ell, u) + (\ell_i' - \ell_i)(c_i - A_i^T y) \geq \overline{z},$$

  then we can tighten u_i to ℓ_i' where

$$\ell_i' = \ell_i + \frac{\overline{z} - z(L, U, \ell, u)}{c_i - A_i^T y}$$

  knowing that there are no better feasible solutions where $x_i > \ell_i'$

REUSING DUAL INFORMATION

COLUMN BOUND TIGHTENING

- Using dual ray r: Dual ray fixing
  $z_r(L, U, \ell, u) =$
  $$\sum_{i:r_i>0} L_i r_i + \sum_{i:r_i<0} U_i r_i + \sum_{i:-A_i^T r>0} \ell_i(-A_i^T r) + \sum_{i:-A_i^T r<0} u_i(-A_i^T r)$$
- If $-A_i^T r > 0$, then increasing ℓ_i also increases z_r
- If there is $\ell_i < \ell_i' < u_i$ such that
  $$z_r(L, U, \ell, u) + (\ell_i' - \ell_i)(-A_i^T r) > 0,$$
  then we can tighten u_i to ℓ_i' where
  $$\ell_i' = \ell_i + \frac{-z_r(L, U, \ell, u)}{-A_i^T r}$$
  knowing that there are no feasible solutions where $x_i > \ell_i'$
- Note that we don't need an upper bound on the objective

REUSING DUAL INFORMATION

ROW BOUND TIGHTENING

- Same as reduced cost fixing/dual ray fixing applied to row bounds
- For $y_i > 0$, U_i to L'_i where

$$L'_i = L_i + \frac{\bar{z} - z_y(L, U, \ell, u)}{y_i}$$

- For $r_i > 0$, U_i to L'_i where

$$L'_i = L_i + \frac{-z_r(L, U, \ell, u)}{r_i}$$

DUAL SOLUTION CACHE VS CONFLICT ANALYSIS

Conflict analysis

- Extract clauses from dual rays and solutions
- Discard the dual solution
- Hard to implement
- Faster to apply
- Extraction takes time
- No local cuts

Dual cache

- Keep the dual information
- Use it later to tighten bounds
- Easy to implement
- More general
- Usage takes more time
- Works with local cuts

REUSING DUAL INFORMATION | VARIABLE SELECTION

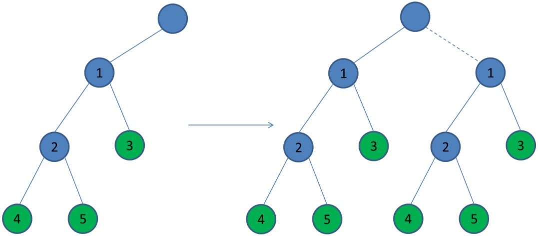
- Estimate child node objective values using dual solution y
- Assume we branch on x_i that takes a fractional value in current node LP: $x_i = f$
- Let Z_L $([\ell_i, \lfloor f \rfloor])$ and Z_R $([\lceil f \rceil, u_i])$ be the LP objective values of left and right child nodes
- If $c_i - A_i^T y < 0$:
  $Z_L \geq \sigma_L^i = z_y(L, U, \ell, u) + (\lfloor f \rfloor - u_i)(c_i - A_i^T y)$
- If $c_i - A_i^T y > 0$:
  $Z_R \geq \sigma_R^i = z_y(L, U, \ell, u) + (\lceil f \rceil - \ell_i)(c_i - A_i^T y)$
- Choose the variable that maximizes σ_L^i or σ_R^i

VARIABLE SELECTION

- Estimate how close the child nodes are to be infeasible using dual ray r
- If $-A_i^T r < 0$:
  $$\delta_L^i = z_r(L, U, \ell, u) + (\lfloor f \rfloor - u_i)(-A_i^T r)$$
- If $-A_i^T r > 0$:
  $$\delta_R^i = z_r(L, U, \ell, u) + (\lceil f \rceil - \ell_i)(-A_i^T r)$$
- Choose the variable that maximizes δ_L^i or δ_R^i

REUSING DUAL INFORMATION | SUBTREES - IP BOUND

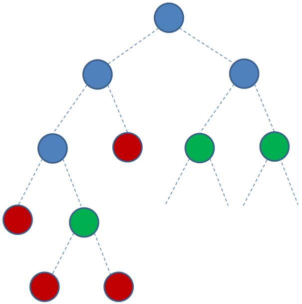
- Branch on x_i. Left child $[\ell_i, \lfloor f \rfloor]$, right child $[\lceil f \rceil, u_i]$



- Dual solutions: $y^1, \ldots, y^5$
- LP bound: $Z_R \geq \max_j \{ z_{y^j}(L, U, \ell, u) + (\lceil f \rceil - \ell_i)(c_i - A_i^T y^j) \}$
- IP bound:
  - z_j: the dual objective value of node j with variable bounds of x_i replaced with $[\lceil f \rceil, u_i]$
  - $P_R \geq \max\{z_1, \min\{z_3, \max\{z_2, \min\{z_4, z_5\}\}\}\}$

DUAL BRANCHING

- Simulate branching with a pool of dual solutions and dual rays



- Select branching variable using dual information
- Evaluate child nodes
  - Prune if proven infeasible by a dual ray
  - Prune by bound by a dual solution
- Otherwise, select a child node and branch

COMPUTATIONAL RESULTS

SAS/OR

(Yes, SAS has a full featured optimization suite.)

- 500 benchmark instances (internal + public)
- 1 hour time limit
- Variable selection: 15% speedup + 15 instances
- Node pruning + bound tightening: 7% speedup + 8 instances
- Helps a lot for feasibility instances
- Conflict analysis has become obsolete

BLOG | OUR NEW BLOG

http://blogs.sas.com/content/operations

http://support.sas.com/or

(Re)using dual information in MILP