Column Basis Reduction,
Decomposable Knapsack
and Cascade Problems

Gábor Pataki

Dept. of Statistics and Operations Research
UNC, Chapel Hill

joint work with Bala Krishnamoorthy

Dept. of Mathematics, Washington State University
What is basis reduction?

Given integral matrix $A$, basis reduction (BR) computes a unimodular $U (\iff \det U = \pm 1)$ st. the columns of $AU$ are “short” and “nearly” orthogonal.

Example

$$A = \begin{pmatrix} 289 & 18 \\ 466 & 29 \\ 273 & 17 \end{pmatrix}, \quad U = \begin{pmatrix} 1 & -15 \\ -16 & 241 \\ 1 & 2 \end{pmatrix}, \quad AU = \begin{pmatrix} 1 & 3 \\ 2 & -1 \end{pmatrix}.$$  

Computing $AU \iff$ doing elementary column operations on $A$:

- adding an integer multiple of a column to another; multiplying a column by $-1$; swapping columns.
Reformulating equality constrained IP feasibility problems

Aardal, Hurkens, Lenstra (1998); Aardal, Bixby, Hurkens, Lenstra, Smeltink (1999); Aardal, Lenstra (2004); Louvaux, Wolsey (2003).

\[ x \in \mathbb{Z}^n \]
\[ Ax = d \]
\[ \ell \leq x \leq u \]
\[ \downarrow \]

\[ \lambda \in \mathbb{Z}^{n-m} \]
\[ \ell \leq B\lambda + x_d \leq u \]
Here

\[ \{ x \in \mathbb{Z}^n \mid Ax = d \} = \{ x_d + B\lambda \mid \lambda \in \mathbb{Z}^{n-m} \} \]

- \([B, x_d]\) is
  - integral, columns are short and nearly orthogonal.
  - found by doing basis reduction on an enlarged matrix using two large constants \(N_1, N_2\).

- The reformulated problem of finding

  \[ \lambda \in \mathbb{Z}^{n-m}, \ell \leq B\lambda + x_d \leq b \]

proved experimentally much easier to solve for some problems, e.g. the Cornuejols-Dawande instances.
Questions

1. Why only equality constrained problems?
2. Why does it work?
Rest of talk

1. Column BR: simplified reformulation for arbitrary IPs. 2 variants: in range space and null space.

2. Computational study.

3. Analysis for a general problem class, called decomposable knapsack problems.
Rangespace reformulation

\[ P = \{ x \mid \ell \leq Ax \leq b \} \]
\[ \tilde{P} = \{ y \mid \ell \leq (AU)y \leq b \} \]

where \( U \) is unimodular.

There is 1-1 correspondence between

\[ P \cap \mathbb{Z}^n \text{ and } \tilde{P} \cap \mathbb{Z}^n \]

given by

\[ Uy = x \]

We choose \( U \) so columns of \( AU \) are reduced. We can do the same if some of the “\( \leq \)” are actually “\( = \)”.
Nullspace reformulation

If

\[ A_1 x = b_1 \]

is a subset of the inequalities in \( \ell \leq Ax \leq b \), then

\[ \{ x \in \mathbb{Z}^n \mid A_1 x = b_1 \} = \{ x_d + B_1 \lambda \mid \lambda \in \mathbb{Z}^{n-m} \} \]

\([B_1, x_d]\) is found by a Hermite Normal Form (HNF) computation; columns are not in general short and orthogonal.

Substitute \( B_1 \lambda + x_d \) for \( x \), and do the rangespace reformulation.

If all constraints are equalities, then essentially equivalent to the Aardal et al. reformulation.
Such a simple reformulation actually works for essentially all hard IPs used to test “nontraditional” IP algorithms!

- We need a problem class on which we can *analyze* its action.
Branching on a constraint

Given polyhedron $P$, integral vector $c$,

- $\text{width}(c, P) = \max \{ cx \mid x \in P \} - \min \{ cx \mid x \in P \}$.
- **branching on** $cx$ means creating the branches $cx = \lfloor \text{min} \rfloor$, $cx = \lfloor \text{min} \rfloor + 1$, $\ldots$, $cx = \lfloor \text{max} \rfloor$. Say $\text{min} = 10.3$, $\text{max} = 15.1$, then $cx$ can be $11, 12, 13, 14, \text{ or } 15$.

- If the interval $[\text{min}, \text{max}]$ contains no integer, then $P$ contains no integral point.
Example: \[ 106 \leq 21x_1 + 19x_2 \leq 113 \]
\[ x_1, x_2 \in [0, 6] \cap \mathbb{Z} \]

Hard for branching on \( x_i \)s.

Easy for branching on \( x_1 + x_2 \): max = 5.94, min = 5.04.
After reformulation: branching on $y_2$ proves infeasibility.
The example is an instance of

\[(KP_2) \quad \beta' \leq a x \leq \beta, \quad 0 \leq x \leq u, \quad x \in \mathbb{Z}^n,\]

where

- \(a = pM + r, \) with \(p \in \mathbb{Z}^+_+, \) \(r \in \mathbb{Z}^n; \) \(M\) large;
- \(\beta, \beta'\) chosen, so \(KP_2\) is LP-feasible, IP-infeasibility proven by branching on \(px\) (but only \(a\) is given, not \(p!)\)
- In the example, \(\underbrace{(21, 19)}_{a} = \underbrace{(1, 1)}_p \times \underbrace{20}_M + \underbrace{(1, -1)}_r.\)
What does the reformulation do on these?

Recall general reformulation:

\[ P = \{ x \mid \ell \leq Ax \leq b \} \iff \tilde{P} = \{ y \mid \ell \leq (AU)y \leq b \} \]
Basis reduction in range space

We choose $U$ unimodular, s.t.

\[
\begin{pmatrix}
  pM + r \\
  I
\end{pmatrix} U \text{ is reduced.}
\]

**Theorem:** $M$ suff. large $\Rightarrow$

\[
pU = (0 \ldots 0 \alpha)
\]

for some $\alpha \in \mathbb{Z} \setminus \{0\}$.

**Corollary:**

\[
Uy = x \Rightarrow pUy = px \Rightarrow \alpha y_n = px
\]

$\Rightarrow$ branching on $y_n$ proves infeasibility.
“Sufficiently large” means:

- If LLL (Lenstra, Lenstra, Lovasz) reduction is used,
  \[ M > 2^{n+1} \| p \| \| r \|^2. \]

- If KZ (Korkhine-Zolotarev) reduction is used,
  \[ M > \sqrt{n} \| p \| \| r \|^2. \]
Basis reduction in null space

Can be used if $\beta = \beta' \rightarrow$ reformulation has $n - 1$ variables.

We can similarly prove: $M$ suff. large $\Rightarrow$ branching on $y_{n-1}$ in reformulation $\equiv$ branching on $px$ in original problem.
A classic example of a decomposable knapsack problem: Jeroslow’s problem

\[ 2(x_1 + \ldots + x_n) = n \]
\[ x_i \in \{0, 1\}^n \]

where \( n \) is odd. In B&B branching on the \( x_i \) no node is pruned above level \( n/2 \). If we branch on \( x_1 + \ldots x_n \), we solve it at the root. Here \( p = e, r = 0, M = 2 \).
Other examples:

(1) $p = e$, $r = (2^0, \ldots, 2^{n-1})$, $u = e$, $M = 2^n + \ell + 1$: Todd’s problem from Chvátal “Hard knapsack problems” (1983).

(2) $p = e$, $r = (1, \ldots, n)$, $u = e$, $M = n(n + 1)$: Avis’ problem from same paper.


(4) $p \geq 0$, $r$ arbitrary, $u = +\infty$, $\beta = \beta'$: Aardal-Lenstra Frobenius problems.

Out of these: (1), (2), (3) take an exponential # of nodes for ordinary B&B; (3) even if knapsack cuts are applied too. In (4) has a $\beta = \text{const} \cdot M^2$ for which problem is infeasible.
Recall the example, with $x_1 + x_2$ a “thin” direction.
Algorithms that find thin directions to branch on


When thinner \( \neq \) better

\[
5660 \leq 520x_1 + 725x_2 + 1156x_3 + 1574x_4 + 1794x_5 + 1829x_6 \\
+ 2023x_7 + 2221x_8 + 2267x_9 + 2465x_{10} + 2496x_{11} \leq 5661
\]

\[
x_i \in \{0, 1\} \ (i = 1, \ldots, 11).
\]  

(1)

- IP-infeasible, and ‘reasonably’ hard for B&B.
- If \( Q = \) LP relaxation, then \( \min_{c \text{ integral}} \text{width}(c, Q) = 1 - 0 \), attained at \( e_i \).
- \( \exists p_1 \) integral: \( \text{width}(p_1, Q) = 25.34 - 24.30 \Rightarrow \) constraint \( p_1x = 25 \) can be added to LP.
- If \( Q' = \) new LP relaxation, then \( \exists p_2 \) integral: \( \text{width}(p_2, Q') = 14.93 - 14.02 \Rightarrow \) proves IP-infeasibility.
• So, a direction with width $= 1.04$ beats all directions with width 1!

• Such problems are called *cascade* problems: branching on a good direction has a “cascade” effect.

• There are more extreme examples, with width in good direction $\approx 1.5$.

• This phenomenon shows up in real problems as well.
$t + 1$-level decomposable knapsack problems

- For $a = p_1 M_1 + p_2 M_2 + \ldots + p_t M_t + r$, with $M_1 > M_2 > \ldots > M_t$ and suitable $\beta, \beta'$

$$(KP_{t+1}) \quad \beta' \leq a x \leq \beta, \quad 0 \leq x \leq u, \quad x \in \mathbb{Z}^n$$

Problem is

- easy, if branching on $p_1 x, p_2 x, \ldots, p_t x$.

- hard, if branching on $x_j$ variables, if parameters suitably chosen.

- cascade problems can be constructed this way.
When using the rangespace reformulation: compute $U$ so that
\[
\left( \sum_{i=1}^{t} p_i M_i + r \right) U \quad \text{is reduced.}
\]

**Theorem:** If separation between $M_1 > M_2 > \ldots > M_t$ is suitably large, then
\[
\begin{pmatrix}
p_1 \\
p_2 \\
\vdots \\
p_t
\end{pmatrix} U =
\begin{pmatrix}
0 & 0 & \ldots & 0 & 0 & 0 & * \\
0 & 0 & \ldots & 0 & 0 & * & * \\
\vdots \\
0 & 0 & \ldots & * & \ldots & * & *
\end{pmatrix}
\]

**Remark:** When computing $U$, we do not know the decomposition!!
**Corollary:** Branching on $y_n, y_{n-1}, \ldots, y_{n-t}$ in reformulation
\[ \Leftrightarrow \text{branching on } p_1x, p_2x, \ldots, p_t x \text{ in original problem.} \]

Analogous result for nullspace reformulation.

- That is, column BR
  - takes the *unknown* “dominant” branching combinations;
  - transforms them into individual variables;
  - lines them up in reverse order of significance!
  - In spirit, similar to the decomposition approach of Cornuéjols, Urbaniak, Weismantel, Wolsey (1998).
Computational results

- BR: KZ reduction by NTL library of Victor Shoup; IP solver: CPLEX 9.0; Machine: 3.2 GHz Linux PC.
- We report: time and bb nodes taken by CPLEX 9.0 after reformulation.
- We do not report: time taken without reformulation (even in the simplest case, it is a few hundred thousand B&B nodes; usually it is $+\infty$).
To solve

\[
\begin{align*}
\text{max} & \quad cx \\
\text{st.} & \quad Ax \leq b \\
& \quad x \in \mathbb{Z}^n
\end{align*}
\]

we replace \( A \) with \( AU \), \( c \) with \( cU \), where \( U \) makes

\[
\begin{pmatrix}
c \\ A
\end{pmatrix}
\]

reduced.
Maximization versions of integer subset sum

\[
\begin{align*}
\text{max} & \quad ax \\
\text{st.} & \quad ax \leq \beta \\
& \quad x \in \mathbb{Z}^n_+.
\end{align*}
\] (2)


\[(12228, 36679, 36682, 48908, 61139, 73365); 89716837\]

Number of B&B nodes after column BR: 5, 0, 9, 0, 10.
Feasibility versions of same instances

For \((a, \beta)\), \(\beta_a := \) optimal value. Then check the feasibility of

\[
ax = \beta_a
\]

\[
x \in \mathbb{Z}^n_+,
\]

using 1) rangespace reformulation, 2) nullspace reformulation.

Number of B&B nodes is between 0 and 10 for all 5 instances, for both choices.

Same happens, if rhs is chosen as \(\beta_a + \gcd(a)\).
Marketshare problems (Cornuéjols, Dawande)

We need to find

\[ x \in \{0, 1\}^n, \quad Ax = d, \]

where \( m = 6 \) or \( m = 7 \), \( n = 10(m - 1) \). \( A, d \) are generated to make the problem difficult.

<table>
<thead>
<tr>
<th></th>
<th>range space</th>
<th>null space</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># BB</td>
<td>CPU</td>
</tr>
<tr>
<td>ms1</td>
<td>288597</td>
<td>175.30</td>
</tr>
<tr>
<td>ms2</td>
<td>220803</td>
<td>165.40</td>
</tr>
</tbody>
</table>
Relaxed marketshare problems

Same data, but we want to find

\[ x \in \{0, 1\}^n, \quad d - 1 \leq Ax \leq d. \]

After column BR

- *markshare1*: 85,466 nodes, 53 seconds; *markshare2*: 250,368 nodes, 211 seconds.
Cascade2

The “big brother” of the 11-variable instance.

- $n = 100$ variables, $a_j \leq 14,000$, $\beta$, $\beta' \leq 100,000$.
- Original problem does not solve by CPLEX after enumerating 2 billion B&B nodes.
- Easy, if we branch on $p_1 x$, then $p_2 x$.
- Reformulation solves at rootnode.
Caveats

- There are hard IPs for which the reformulation does not work :-(

- The reformulation uncovers the hidden “dominant” directions in the polyhedron - but in some hard problems, these may not exist, if the problem is symmetric.
Conclusions and further work

- A general, and very simple reformulation technique for arbitrary IPs.
- A fairly general class of IPs that are provably hard for ordinary B&B.
- Analysis: the provably hard problems turn into provably easy ones: the reformulation “uncovers” the hidden, dominant directions.
- The cascade problems: thinner ≠ better!
- Works well in on most small, hard IPs from the literature.