Solving Symmetric Integer Programs

IMA New Directions Short Course on Mathematical Optimization

Jeff Linderoth

Department of Industrial and Systems Engineering
Wisconsin Institutes of Discovery
University of Wisconsin-Madison

August 11, 2016
Solving Symmetric Integer Programs

JEFF LINDEROTH
Dept. of Industrial and Systems Engineering
Univ. of Wisconsin-Madison
linderoth@wisc.edu

JIM OSTROWSKI
University of Tennessee

FRANÇOIS MARGOT
Carnegie Mellon University
Italian Arnolds

Fabrizio Rossi
Stefano Smriglio
Università di L’Aquila
Binary Integer Program

\[
\min_{x \in \{0,1\}^n} \{ c^T x \mid Ax \geq b \} \quad \text{(BIP)}
\]

- For simplicity, we will assume that \( A \in \{0,1\}^{m \times n} \) and

\[
\begin{align*}
\text{SCP} & \quad \min_{x \in \{0,1\}^n} \{ 1^T x \mid Ax \geq 1 \} \quad \text{or} \quad \text{SPP} & \quad \max_{x \in \{0,1\}^n} \{ 1^T x \mid Ax \leq 1 \}
\end{align*}
\]
Binary Integer Program

\[
\min_{x \in \{0,1\}^n} \{ c^T x \mid Ax \geq b \} \quad \text{(BIP)}
\]

- For simplicity, we will assume that \( A \in \{0,1\}^{m \times n} \) and

\[
\begin{align*}
\text{SCP} & \quad \min_{x \in \{0,1\}^n} \{ 1^T x \mid Ax \geq 1 \} \quad \text{or} \quad \\
\text{SPP} & \quad \max_{x \in \{0,1\}^n} \{ 1^T x \mid Ax \leq 1 \}
\end{align*}
\]

Outline

1. What is Symmetry in IP?
2. Motivation - The Football Pool Problem
3. Orbital Branching
4. Extensions (Very brief)
5. Computational Results
Warning! This All Happened Some Time Ago...

“This is really embarrassing. I just forgot our state governor’s name, but I know that you will help me recall him.”

—Arnold, speaking to a taxpayer advocacy group
Warning! This All Happened Some Time Ago...

“This is really embarrassing. I just forgot our state governor’s name, but I know that you will help me recall him.”

—Arnold, speaking to a taxpayer advocacy group

I hope I recall enough to give an informative lecture
Football Pool Problem

- We motivate symmetry in IP via gambling

The Football Pool Problem

What is the minimum number of tickets you must buy to assure yourself a win?
Football Pool Problem

- We motivate symmetry in IP via gambling
- Predict the outcome of $v$ soccer matches
- Giant Prize if all $v$ are correct
- Also win if $v - 1$ are correct: you mis-predict at most 1 game
Football Pool Problem

- We motivate symmetry in IP via gambling
- Predict the outcome of $v$ soccer matches
- Giant Prize if all $v$ are correct
- Also win if $v - 1$ are correct: you mis-predict at most 1 game

The Football Pool Problem

What is the minimum number of tickets you must buy to assure yourself a win?
Football IP

- Let $N$ be the set of possible outcomes (also the set of tickets) ($|N| = 3^v$)
- Binary variables: $x_j = 1$ if I purchase ticket $j \in N$
- Let $A \in \{0, 1\}^{|N| \times |N|}$ with $a_{ij} = 1$ iff ticket $j \in N$ is a winner for outcome $i \in N$
Football IP

- Let $N$ be the set of possible outcomes (also the set of tickets) ($|N| = 3^v$)
- Binary variables: $x_j = 1$ if I purchase ticket $j \in N$
- Let $A \in \{0, 1\}^{|N| \times |N|}$ with $a_{ij} = 1$ iff ticket $j \in N$ is a winner for outcome $i \in N$

**IP Formulation**

$$\begin{align*}
\text{min} & \quad 1^T x \\
\text{s.t.} & \quad Ax \geq 1 \\
& \quad x \in \{0, 1\}^{|N|}
\end{align*}$$
Football Matrix, $\nu = 3$

Playing Football
Chose columns to cover all rows
It’s Pretty! The Football Matrix, $v = 6$
\( v = 3, \) Solution \#1

<table>
<thead>
<tr>
<th>M1</th>
<th>M2</th>
<th>M3</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>W</td>
</tr>
<tr>
<td>L</td>
<td>W</td>
<td>L</td>
</tr>
<tr>
<td>W</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>

Answer \#1
\( v = 3 \), Solution \#2

<table>
<thead>
<tr>
<th>M1</th>
<th>M2</th>
<th>M3</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>W</td>
<td>W</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>W</td>
</tr>
<tr>
<td>L</td>
<td>W</td>
<td>L</td>
</tr>
<tr>
<td>D</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>W</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>
Solutions for $v = 3$

- These solutions are isomorphic.
  - Swap $W \leftrightarrow D$ in the first match
Solutions for $v = 3$

- These solutions are isomorphic.
  - Swap $W \leftrightarrow D$ in the first match
- There are LOTS of isomorphic solutions:
  1. “Rename” $W, L, D$ for any subset of the matches: $(3!)^v$
  2. Reorder the matches: $v!$
- There are $(3!)^3(3!) = 1296$ ways to reorder solutions for $v = 3$
- There are $(3!)^6(6!) = 33,592,320$ ways to reorder solutions for $v = 6$
How Many Must I Buy?

Known Optimal Values

<table>
<thead>
<tr>
<th>( v )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>C_v^*</td>
<td>)</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

Despite significant effort on this problem for > 40 years, it is only known that \( \leq C_v^* \leq 73 \).

Hooray For Us!
We were able to improve the lower bound to 71.
It only took 140 CPU years!

Jeff Linderoth (UW-Madison)
How Many Must I Buy?

### Known Optimal Values

<table>
<thead>
<tr>
<th>$v$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>C_v^*</td>
<td>\mid$</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

### The Football Pool Problem

What is $|C_6^*|$?

Despite significant effort on this problem for > 40 years, it is only known that $|C_6^*| \leq 73$.

Hooray For Us!

We were able to improve the lower bound to $71$.

It only took 140 CPU years!
How Many Must I Buy?

<table>
<thead>
<tr>
<th>Known Optimal Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>( v )</td>
</tr>
<tr>
<td>(</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>The Football Pool Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is (</td>
</tr>
</tbody>
</table>

- Despite significant effort on this problem for > 40 years, it is only known that

\[
65 \leq C^*_6 \leq 73
\]
How Many Must I Buy?

Known Optimal Values

<table>
<thead>
<tr>
<th>$v$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>C_v^{*}</td>
<td>$</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

The Football Pool Problem

What is $|C_6^{*}|$?

- Despite significant effort on this problem for $>40$ years, it is only known that
  
  $$71 \leq C_6^{*} \leq 73$$

Hooray For Us!

- We were able to improve the lower bound to 71
- It only took 140 CPU years!
CPLEX Can’t Solve Every IP

- Roughly $10^8$ universe lifetimes in order to establish that $|C^*_6| > 72$
A Review of Symmetry and Algebra
Symmetry

- Let $\Pi^n$ be collection of permutations of $\{1, 2, \ldots, n\}$
- Given $\lambda \in \mathbb{R}^n$, $\pi \in \Pi^n$ acts on $\lambda$ by permuting its coordinates: $\pi(\lambda) = (\lambda_{\pi_1}, \lambda_{\pi_2}, \ldots, \lambda_{\pi_n})$. 

The set of symmetries of IP (with composition of permutations) forms the symmetry group of IP $G(IP) = \{\pi \in \Pi^n | \pi(\lambda) \in F, c^T x = c^T \pi(\lambda) \forall x \in F\}$, where $F = \{x \in \{0, 1\}^n | Ax \geq b\}$ is the set of feasible solutions.
Symmetry

- Let $\Pi^n$ be collection of permutations of $\{1, 2, \ldots, n\}$
- Given $\lambda \in \mathbb{R}^n$, $\pi \in \Pi^n$ acts on $\lambda$ by permuting its coordinates:
  $$\pi(\lambda) = (\lambda_{\pi_1}, \lambda_{\pi_2}, \ldots \lambda_{\pi_n}).$$

$\pi \in \Pi^n$ is a symmetry of IP if...

1. $x$ feasible $\iff \pi(x)$ feasible
2. $c^T x = c^T \pi(x)$
Symmetry

- Let $\Pi^n$ be collection of permutations of $\{1, 2, \ldots, n\}$
- Given $\lambda \in \mathbb{R}^n$, $\pi \in \Pi^n$ acts on $\lambda$ by permuting its coordinates:
  $\pi(\lambda) = (\lambda_{\pi_1}, \lambda_{\pi_2}, \ldots \lambda_{\pi_n})$.

- $\pi \in \Pi^n$ is a **symmetry** of IP if...
  1. $x$ feasible $\iff \pi(x)$ feasible
  2. $c^T x = c^T \pi(x)$

- The set of symmetries of IP (with composition of permutations)
  forms the **symmetry group** of IP

  $\mathcal{G}(\text{IP}) = \{ \pi \in \Pi^n \mid \pi(x) \in \mathcal{F}, c^T x = c^T \pi(x) \quad \forall x \in \mathcal{F} \}$,

  where $\mathcal{F} = \{ x \in \{0, 1\}^n \mid Ax \geq b \}$ is the set of feasible solutions
Symmetry and Branching

Symmetry “Erases” Branching Decision

In the presence of symmetry, branching does not effectively change the solution to the LP relaxation.

Let \( \hat{x} = (\frac{1}{2}, 0, \frac{1}{4}, 1, \ldots) \) be a solution to the LP relaxation. If \( \pi_1 = (1, 2) \) \( \in G \), \( \pi_2 = (1, 4) \) \( \in G \), then \( z^-i = z^+i = z_{LP} \). You are guaranteed to have a bad branch.
Symmetry and Branching

**Symmetry “Erases” Branching Decision**

- In the presence of symmetry, branching does not effectively change the solution to the LP relaxation.

Let \( \hat{x} = (1/2, 0, 1/4, 1, \ldots)^T \) be a solution to the LP relaxation.

If \( \pi_1 = (1, 2) \in \mathcal{G}, \pi_2 = (1, 4) \in \mathcal{G} \) then \( z_i^- = z_i^+ = z_{LP} \).

You are guaranteed to have a bad branch.

\[
\hat{x}_1 = \frac{1}{2} \\
\]

- Apply \( \pi_2 \)
- Apply \( \pi_1 \)
Searching with Symmetry

“Big Mistake.”

Jack Slater, Last Action Hero
Searching with Symmetry

"Big Mistake."

Jack Slater, Last Action Hero

Suppose the permutation \( (1, 2) \in \mathcal{G} \)

You evaluate many completely equivalent (isomorphic) subtrees
About Symmetry Groups

- $\mathcal{G}(IP)$ is a property of the feasible region: $\mathcal{F} = \emptyset \Rightarrow \mathcal{G}(IP) = \Pi^n$
- For our methods, we can work with any subgroup $\Gamma \subset \mathcal{G}(IP)$
About Symmetry Groups

- $\mathcal{G}(\text{IP})$ is a property of the feasible region: $\mathcal{F} = \emptyset \Rightarrow \mathcal{G}(\text{IP}) = \Pi^n$
- For our methods, we can work with any subgroup $\Gamma \subset \mathcal{G}(\text{IP})$
- If $c = 1$, $b = 1$, we can use the symmetry group of the matrix $A$: 
  
  $$ \mathcal{G}(A) \overset{\text{def}}{=} \{ \pi \in \Pi^n \mid \exists \sigma \in \Pi^m \text{ such that } P_\sigma A P_\pi = A \} $$

- Given $A$ of “reasonable” size, there exist software packages (\texttt{nauty}, \texttt{saucy}) that can compute (generators of) $\mathcal{G}(A)$ “effectively”
  - Actual algorithm is exponential, but in general works quickly
Orbits

For a point \( z \in \mathcal{Z} \), the orbit of \( z \) under \( \mathcal{G} \) is the set of all elements of \( \mathcal{Z} \) to which \( z \) can be sent by permutations in \( \mathcal{G} \):

\[
\text{orb}(\mathcal{G}, z) \overset{\text{def}}{=} \{ \pi(z) \mid \pi \in \mathcal{G} \}.
\]
Orbits

- For a point \( z \in \mathbb{Z} \), the orbit of \( z \) under \( \mathcal{G} \) is the set of all elements of \( \mathbb{Z} \) to which \( z \) can be sent by permutations in \( \mathcal{G} \):

\[
\text{orb}(\mathcal{G}, z) \overset{\text{def}}{=} \{ \pi(z) \mid \pi \in \mathcal{G} \}.
\]

- Consider the orbits of each of coordinate axes: \( e_j, j \in \mathbb{N} \)
- By definition, if \( e_j \in \text{orb}(\mathcal{G}, e_k) \) then \( e_k \in \text{orb}(\mathcal{G}, e_j) \), i.e. the variables \( x_j \) and \( x_k \) share the same orbit. Therefore, the union of the orbits

\[
\mathcal{O}(\mathcal{G}) \overset{\text{def}}{=} \bigcup_{j=1}^{n} \text{orb}(\mathcal{G}, e_j)
\]

forms a partition of \( N = \{1, 2, \ldots, n\} \), which we refer to as the orbits of \( \mathcal{G} \).
Orbits

- For a point \( z \in \mathcal{Z} \), the orbit of \( z \) under \( \mathcal{G} \) is the set of all elements of \( \mathcal{Z} \) to which \( z \) can be sent by permutations in \( \mathcal{G} \):

\[
\text{orb}(\mathcal{G}, z) \overset{\text{def}}{=} \{ \pi(z) \mid \pi \in \mathcal{G} \}.
\]

- Consider the orbits of each of coordinate axes: \( e_j, j \in \mathcal{N} \)
- By definition, if \( e_j \in \text{orb}(\mathcal{G}, e_k) \) then \( e_k \in \text{orb}(\mathcal{G}, e_j) \), i.e. the variables \( x_j \) and \( x_k \) share the same orbit. Therefore, the union of the orbits

\[
\mathcal{O}(\mathcal{G}) \overset{\text{def}}{=} \bigcup_{j=1}^{n} \text{orb}(\mathcal{G}, e_j)
\]

forms a partition of \( \mathcal{N} = \{1, 2, \ldots, n\} \), which we refer to as the orbits of \( \mathcal{G} \).
- The orbits encode which variables are “equivalent” (symmetric) with respect to the symmetry \( \mathcal{G} \).
Ugh... More Notation

- Branch-and-bound node $a = (F_1^a, F_0^a)$,
  - $F_1^a$: Set of variables fixed to one
  - $F_0^a$: Set of variables fixed to zero
- $\mathcal{F}(a)$: The set of feasible solutions to the IP at node $a$
Ugh... More Notation

- Branch-and-bound node $a = (F_1^a, F_0^a)$,
  - $F_1^a$: Set of variables fixed to one
  - $F_0^a$: Set of variables fixed to zero
- $\mathcal{F}(a)$: The set of feasible solutions to the IP at node $a$

The stabilizer of a set $S$ in $\mathcal{G}$ is the set of permutations in $\mathcal{G}$ that send $S$ to itself: $\text{stab}(S, \mathcal{G}) = \{ \pi \in \mathcal{G} \mid \pi(S) = S \}$.
- $\text{stab}(S, \mathcal{G})$ is a subgroup of $\mathcal{G}$
Ugh... More Notation

- Branch-and-bound node $a = (F^a_1, F^a_0)$,
  - $F^a_1$: Set of variables fixed to one
  - $F^a_0$: Set of variables fixed to zero
- $\mathcal{F}(a)$: The set of feasible solutions to the IP at node $a$

---

- The stabilizer of a set $S$ in $\mathcal{G}$ is the set of permutations in $\mathcal{G}$ that send $S$ to itself: $\text{stab}(S, \mathcal{G}) = \{ \pi \in \mathcal{G} \mid \pi(S) = S \}$.
- $\text{stab}(S, \mathcal{G})$ is a subgroup of $\mathcal{G}$

---

The Upshot

- As we fix variables (to 1) at node $a$, the symmetry “remaining” in the problem becomes $\text{stab}(\chi^a_{F_1}, \mathcal{G})$
Orbital Branching: A Simple Idea

Schwarzenegger

Go ahead, you tell him you didn't do your homework.

Kindergarten COP

As an undercover cop...he's in a class by himself.
Orbital Branching

- A way to exploit symmetry in your branching decision
- Let \( O \in \mathcal{O}(\mathcal{G}(IP)) \) be an orbit of the symmetry group of the IP.
- Surely we can branch as

\[
\sum_{i \in O} x_i \geq 1 \quad \text{or} \quad \sum_{i \in O} x_i \leq 0.
\]
Orbital Branching

- A way to exploit symmetry in your branching decision
- Let $O \in \mathcal{O}(G(IP))$ be an orbit of the symmetry group of the IP.
- Surely we can branch as

\[
\sum_{i \in O} x_i \geq 1 \quad \text{or} \quad \sum_{i \in O} x_i \leq 0.
\]

- If at least one variable $i \in O$ is going to be one, and they are all “equivalent”, then you may as well pick ($i^*$) one arbitrarily.

\[
x_{i^*} = 1 \quad \text{or} \quad \sum_{i \in O} x_i = 0
\]
Orbital Branching

- A way to exploit symmetry in your branching decision
- Let $O \in \mathcal{O}(\mathcal{G}(IP))$ be an orbit of the symmetry group of the IP.
- Surely we can branch as

$$\sum_{i \in O} x_i \geq 1 \quad \text{or} \quad \sum_{i \in O} x_i \leq 0.$$  

- If at least one variable $i \in O$ is going to be one, and they are all “equivalent”, then you may as well pick ($i^*$) one arbitrarily.

$$x_{i^*}^* = 1 \quad \text{or} \quad \sum_{i \in O} x_i = 0$$

- No, really. That’s it. :-(
An Alternative View of Orbital Branching

- Suppose that you have found that the variables $x_e, x_f, x_g$ and $x_h$ share an orbit at node $a$, $O = \{e, f, g, h\}$.
- Then you can surely branch as:

$$x_e = 1 \quad x_f = 1 \quad x_g = 1 \quad x_h = 1 \quad \sum_{j \in O} x_j = 0$$
An Alternative View of Orbital Branching

- Suppose that you have found that the variables $x_e, x_f, x_g$ and $x_h$ share an orbit at node $a$, $O = \{e, f, g, h\}$.

- Then you can surely branch as:

  $$\sum_{j \in O} x_j = 0$$

- But the best solution you can find from nodes $f, g$, and $h$ will be the same as the best solution you can find from node $e$.

- In fact, solutions will be isomorphic.

  $\Rightarrow$ Prune nodes $f$, $g$, and $h$. 

Orbital Branching Theorems

**Theorem: OB is Valid**

All optimal solutions are not eliminated.

---

But Can We Do Better!?

Can we branch and prune such that \( x \in F(b) \) and \( y \in F(c) \) are not equivalent (isomorphic) with respect to the original symmetry group \( G \)?
**Orbital Branching Theorems**

**Theorem: OB is Valid**
All optimal solutions are not eliminated.

**Theorem: OB Reduces Symmetry**
Let $b$ and $c$ be any two subproblems in the enumeration tree. Let $a$ be the first common ancestor of $b$ and $c$. If $x \in F(b)$ and $y \in F(c)$, then $\not\exists \pi \in G(A(F_0^a, F_1^a))$ with $\pi(x) = y$. 

But Can We Do Better!? 
Can we branch and prune such that $x \in F(b)$ and $y \in F(c)$ are not equivalent (isomorphic) with respect to the original symmetry group $G$?
**Orbital Branching Theorems**

**Theorem: OB is Valid**
All optimal solutions are not eliminated.

**Theorem: OB Reduces Symmetry**
Let $b$ and $c$ be any two subproblems in the enumeration tree. Let $a$ be the first common ancestor of $b$ and $c$. If $x \in F(b)$ and $y \in F(c)$, then
$$\not\exists \pi \in G(A(F_0^a, F_1^a)) \text{ with } \pi(x) = y.$$

**But Can We Do Better!?**
Can we branch and prune such that $x \in F(b)$ and $y \in F(c)$ are not equivalent (isomorphic) with respect to the original symmetry group $G$?
Isomorphism Pruning

Can we "terminate" the search without exploring equivalent solutions?
Isomorphism Pruning

The admitedly very bad Joke

Can we “terminate” the search without exploring equivalent solutions?
Search the Fundamental Domain!

**It's Fundamental**

- The (minimal) *Fundamental Domain* of a feasible region $\mathcal{F}$ with respect to a (permutation) group $\mathcal{G}$ is the *smallest subset* $F \subseteq \mathcal{F}$, such that if $x \in \mathcal{F}$, then $x = \pi(y)$ for some $\pi \in \mathcal{G}$, $y \in F$.
Search the Fundamental Domain!

**It’s Fundamental**
- The (minimal) Fundamental Domain of a feasible region $\mathcal{F}$ with respect to a (permutation) group $\mathcal{G}$ is the *smallest subset* $F \subseteq \mathcal{F}$, such that if $x \in \mathcal{F}$, then $x = \pi(y)$ for some $\pi \in \mathcal{G}$, $y \in F$.

**Key Idea: Exploit Symmetry**
- Restrict search to $F$, not $\mathcal{F}$!
- Put another way: for any feasible solution, $x$, we only need to consider one element in $\text{orb}(\mathcal{G}, x)$.
- By definition, any method that restricts itself to a fundamental domain $\mathcal{F}$ will not encounter isomorphic solutions.
Some History

- Francois Margot (in 2002) wrote an influential paper where we demonstrated how to prune search nodes that were not in a fundamental domain: **Isomorphism Pruning**
- His idea (more or less) required that you branch on the variables in a fixed, lexicographic, order
Some History

- Francois Margot (in 2002) wrote an influential paper where we demonstrated how to prune search nodes that were not in a fundamental domain: Isomorphism Pruning
- His idea (more or less) required that you branch on the variables in a fixed, lexicographic, order

---

**Isomorphism Pruning Theorem:**

For node $a$ and symmetry group $\mathcal{G}$, let $F_1^a$ be the set of variables fixed to one (by branching decision) at node $a$. If $F_1^a$ is not lexicographically minimal with respect to $\text{orb}(\mathcal{G}, \chi_{F_1^a})$, node $a$ can be pruned.
Isomorphism Pruning Tree

\[ x_1 = 1 \quad x_2 = 1 \quad x_3 = 1 \]

\[ x_1 = 0 \quad x_2 = 0 \quad x_3 = 0 \]

No Flexibility!
You must branch on variable, regardless of impact on bound.
Even if \( \hat{x}_d \) is not fractional at level \( d \)
This is typically a very bad idea for branch and bound

Jeff Linderoth (UW-Madison)
Isomorphism Pruning Tree

No Flexibility!

- You must branch on variable, regardless of impact on bound.
- Even if $\hat{x}_d$ is not fractional at level $d$
Isomorphism Pruning

Isomorphism Pruning Tree

No Flexibility!
- You must branch on variable, regardless of impact on bound.
- Even if \( \hat{x}_d \) is not fractional at level \( d \)
- This is typically a very bad idea for branch and bound
I am the Greatest Thesis Advisor Ever!

"Why can’t we define a 'local ordering' of the variables to define lexicographic min?"
I am the Greatest Thesis Advisor Ever!

“Why can’t we define a ’local ordering’ of the variables to define lexicographic min?”

“Because if we could, then I am sure that François would have thought of it”
Jim’s Reaction
Jim’s Reaction

- Thankfully, Jim rarely listens to me...
Flexible Isomorphism Pruning
François Does Not Think of Everything!

- Each node $a$ has rank vector $R^a$.
- $R^a[i] = j$ implies $x_i$ was branched on at the ancestor node of $a$ at depth $j$. Variables not fixed by a branching at node $a$ are assigned the rank $(-1)$. 

Flexible Isomorphism Theorem: For node $a$ and symmetry group $G$, let $F_a$ be the set of variables fixed to one (by branching decision) at node $a$. If $R^a(F_a)$ is not lexicographically minimal with respect to $R^a(\text{orb}(G, \chi F_a))$, node $a$ can be pruned.

The Good Part!
This is true for any branching decisions. No additional (costly) computations are needed compared to regular isomorphism pruning.
François Does Not Think of Everything!

- Each node $a$ has rank vector $R^a$.
- $R^a[i] = j$ implies $x_i$ was branched on at the ancestor node of $a$ at depth $j$. Variables not fixed by a branching at node $a$ are assigned the rank $(-1)$.

**Flexible Isomorphism Theorem:**

For node $a$ and symmetry group $G$, let $F_1^a$ be the set of variables fixed to one (by branching decision) at node $a$. If $R^a(F_1^a)$ is not lexicographically minimal with respect to $R^a(\text{orb}(G, \chi F_1^a))$, node $a$ can be pruned.
François Does Not Think of Everything!

- Each node $a$ has rank vector $R^a$.
- $R^a[i] = j$ implies $x_i$ was branched on at the ancestor node of $a$ at depth $j$. Variables not fixed by a branching at node $a$ are assigned the rank $(-1)$.

**Flexible Isomorphism Theorem:**

For node $a$ and symmetry group $G$, let $F_1^a$ be the set of variables fixed to one (by branching decision) at node $a$. If $R^a(F_1^a)$ is not lexicographically minimal with respect to $R^a(orb(G, \chi F_1^a))$, node $a$ can be pruned.

**The Good Part!**

- This is true for any branching decisions.
- No additional (costly) computations are needed compared to regular isomorphism pruning.
Most Flexible Isomorphism Pruning Tree

\[ x_4 = 1 \]
\[ x_4 = 0 \]
\[ x_1 = 1 \]
\[ x_1 = 0 \]
\[ x_2 = 1 \]
\[ x_2 = 0 \]
\[ x_5 = 1 \]
\[ x_5 = 0 \]

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R^A(i) )</td>
<td>2</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R^B(i) )</td>
<td>3</td>
<td>2</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
</tr>
</tbody>
</table>
Branching: What To Do?
Branching: What To Do?

Combine “Strength” of Isomorphism Pruning
Branching: What To Do?

Combine “Strength” of Isomorphism Pruning

With “Flexibility” of Branching on Any Variable/Orbit
Which Branchable Orbit to Choose?

- Given $\hat{x}$ and (branchable) orbits $O_1, O_2, \ldots, O_p$ at node $a$, which orbit should be choose?
- We investigated (so far) lots of different branching rules. I will only talk of 3
Which Branchable Orbit to Choose?

- Given \( \hat{x} \) and (branchable) orbits \( O_1, O_2, \ldots O_p \) at node \( a \), which orbit should be choose?
- We investigated (so far) lots of different branching rules. I will only talk of 3

1. **Branch Min Index**: Branch on orbit that contains the smallest unfixed variable index. “Close” to Margot’s original branching
2. **Branch Largest LP Solution**: Branch on Orbit with most LP solution
3. **Strong Branching**: For each orbit, create orbital branching dichotomy. Evaluate the two resulting children, and choose “the best”
Instance Families

- **(Binary) Error Correcting Codes** ($\text{cod}(n,d)$): Find maximum number of $(0,1)$ $n$-vectors such that Hamming distance between each pair is $\geq d$
- **Covering Design** ($\text{cov}(v,k,t)$): $v > k > t$: Find minimum number of $k$-sets of $\{1, \ldots, v\}$ to “cover” all $t$-sets of $\{1, \ldots, v\}$.
- **Covering Code** ($\text{codbt}(b,t)$): Find minimum number of “codewords” such that every word is at most a (Hamming) distance 1 from a codeword.
- **Steiner Triple System**: ($\text{sts}(n)$): Find the “incidence width” of a Steiner Triple System of order $n$
## Number of Nodes

<table>
<thead>
<tr>
<th>Instance</th>
<th>CPLEX v11 w/Sym</th>
<th>Largest LP</th>
<th>Min Index</th>
<th>Str. Branching</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>O.B.</td>
<td>IsoP.</td>
<td>O.B.</td>
</tr>
<tr>
<td>cod83</td>
<td>9338</td>
<td>35</td>
<td>23</td>
<td>23</td>
</tr>
<tr>
<td>cod93</td>
<td>287998*</td>
<td>3531</td>
<td>711</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2933</td>
<td>269</td>
<td>127</td>
</tr>
<tr>
<td>cov954</td>
<td>1226</td>
<td>113</td>
<td>701</td>
<td>39</td>
</tr>
<tr>
<td>cov1053</td>
<td>262628</td>
<td>3535</td>
<td>893</td>
<td>569</td>
</tr>
<tr>
<td>cov1054</td>
<td>94949*</td>
<td>52509</td>
<td>567</td>
<td>426</td>
</tr>
<tr>
<td>cov1075</td>
<td>21076</td>
<td>107</td>
<td>471</td>
<td>71</td>
</tr>
<tr>
<td></td>
<td></td>
<td>105</td>
<td>367</td>
<td>63</td>
</tr>
<tr>
<td>codbt42</td>
<td>-</td>
<td>73</td>
<td>1059</td>
<td>37</td>
</tr>
<tr>
<td>codbt05</td>
<td>107816</td>
<td>303</td>
<td>1521</td>
<td>103</td>
</tr>
<tr>
<td></td>
<td></td>
<td>285</td>
<td>1245</td>
<td>103</td>
</tr>
<tr>
<td>sts45</td>
<td>19931</td>
<td>9373</td>
<td>6037</td>
<td>1861</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4469</td>
<td>1553</td>
<td>1203</td>
</tr>
<tr>
<td>sts63</td>
<td>4805781</td>
<td>37243</td>
<td>12365</td>
<td>3221</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6327</td>
<td>4303</td>
<td>2309</td>
</tr>
<tr>
<td>sts81</td>
<td>13361288*</td>
<td>2361</td>
<td>2995</td>
<td>991</td>
</tr>
<tr>
<td></td>
<td></td>
<td>527</td>
<td>585</td>
<td>509</td>
</tr>
</tbody>
</table>
## Tale of the Tape—Gurobi v3.0

<table>
<thead>
<tr>
<th>Instance</th>
<th>Symmetry = 0</th>
<th>Symmetry = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td>Gap%</td>
</tr>
<tr>
<td>cod105</td>
<td>7200</td>
<td>50.0</td>
</tr>
<tr>
<td>cod83</td>
<td>7200</td>
<td>15.0</td>
</tr>
<tr>
<td>cod93</td>
<td>7200</td>
<td>20.0</td>
</tr>
<tr>
<td>codbt05</td>
<td>7200</td>
<td>7.4</td>
</tr>
<tr>
<td>codbt33</td>
<td>8</td>
<td>0.0</td>
</tr>
<tr>
<td>codbt42</td>
<td>159</td>
<td>0.0</td>
</tr>
<tr>
<td>codbt61</td>
<td>10</td>
<td>0.0</td>
</tr>
<tr>
<td>cov1053</td>
<td>7200</td>
<td>5.9</td>
</tr>
<tr>
<td>cov1054</td>
<td>7200</td>
<td>2.0</td>
</tr>
<tr>
<td>cov1075</td>
<td>7200</td>
<td>5.0</td>
</tr>
<tr>
<td>cov954</td>
<td>58</td>
<td>0.0</td>
</tr>
<tr>
<td>sts27</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>sts45</td>
<td>18</td>
<td>0.0</td>
</tr>
<tr>
<td>sts63</td>
<td>7200</td>
<td>4.4</td>
</tr>
<tr>
<td>sts81</td>
<td>7200</td>
<td>16.4</td>
</tr>
</tbody>
</table>

"I'm the party pooper."
# Tale of the Tape—Gurobi v3.0

<table>
<thead>
<tr>
<th>Instance</th>
<th>Symmetry = 0</th>
<th></th>
<th></th>
<th>Symmetry = 2</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td>Gap%</td>
<td>Nodes</td>
<td>Time</td>
<td>Gap%</td>
<td>Nodes</td>
</tr>
<tr>
<td>cod105</td>
<td>7200</td>
<td>50.0</td>
<td>150</td>
<td>173</td>
<td>0.0</td>
<td>7</td>
</tr>
<tr>
<td>cod83</td>
<td>7200</td>
<td>15.0</td>
<td>724601</td>
<td>6</td>
<td>0.0</td>
<td>372</td>
</tr>
<tr>
<td>cod93</td>
<td>7200</td>
<td>20.0</td>
<td>108572</td>
<td>905</td>
<td>0.0</td>
<td>54650</td>
</tr>
<tr>
<td>codbt05</td>
<td>7200</td>
<td>7.4</td>
<td>352025</td>
<td>7200</td>
<td>3.7</td>
<td>359268</td>
</tr>
<tr>
<td>codbt33</td>
<td>8</td>
<td>0.0</td>
<td>604</td>
<td>6</td>
<td>0.0</td>
<td>401</td>
</tr>
<tr>
<td>codbt42</td>
<td>159</td>
<td>0.0</td>
<td>75569</td>
<td>111</td>
<td>0.0</td>
<td>45912</td>
</tr>
<tr>
<td>codbt61</td>
<td>10</td>
<td>0.0</td>
<td>1485</td>
<td>7</td>
<td>0.0</td>
<td>950</td>
</tr>
<tr>
<td>cov1053</td>
<td>7200</td>
<td>5.9</td>
<td>919836</td>
<td>77</td>
<td>0.0</td>
<td>10958</td>
</tr>
<tr>
<td>cov1054</td>
<td>7200</td>
<td>2.0</td>
<td>189645</td>
<td>2330</td>
<td>0.0</td>
<td>103657</td>
</tr>
<tr>
<td>cov1075</td>
<td>7200</td>
<td>5.0</td>
<td>549355</td>
<td>17</td>
<td>0.0</td>
<td>665</td>
</tr>
<tr>
<td>cov954</td>
<td>58</td>
<td>0.0</td>
<td>31950</td>
<td>1</td>
<td>0.0</td>
<td>166</td>
</tr>
<tr>
<td>sts27</td>
<td>1</td>
<td>0.0</td>
<td>4044</td>
<td>0</td>
<td>0.0</td>
<td>78</td>
</tr>
<tr>
<td>sts45</td>
<td>18</td>
<td>0.0</td>
<td>61194</td>
<td>23</td>
<td>0.0</td>
<td>34839</td>
</tr>
<tr>
<td>sts63</td>
<td>7200</td>
<td>4.4</td>
<td>8698168</td>
<td>85</td>
<td>0.0</td>
<td>43135</td>
</tr>
<tr>
<td>sts81</td>
<td>7200</td>
<td>16.4</td>
<td>3252747</td>
<td>70</td>
<td>0.0</td>
<td>6317</td>
</tr>
</tbody>
</table>

"I’m the party pooper."

"Hasta la vista, baby"
### Tale of the Tape—CPLEX v12.1

<table>
<thead>
<tr>
<th>Instance</th>
<th>Symmetry = 0</th>
<th>Symmetry = 5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td>Gap%</td>
</tr>
<tr>
<td>cod105</td>
<td>7200</td>
<td>52.4</td>
</tr>
<tr>
<td>cod83</td>
<td>7200</td>
<td>14.3</td>
</tr>
<tr>
<td>cod93</td>
<td>7200</td>
<td>18.9</td>
</tr>
<tr>
<td>codbt05</td>
<td>7200</td>
<td>5.6</td>
</tr>
<tr>
<td>codbt33</td>
<td>8</td>
<td>0.0</td>
</tr>
<tr>
<td>codbt42</td>
<td>89</td>
<td>0.0</td>
</tr>
<tr>
<td>codbt61</td>
<td>8</td>
<td>0.0</td>
</tr>
<tr>
<td>cov1053</td>
<td>7200</td>
<td>5.9</td>
</tr>
<tr>
<td>cov1054</td>
<td>7200</td>
<td>2.0</td>
</tr>
<tr>
<td>cov1075</td>
<td>7200</td>
<td>6.4</td>
</tr>
<tr>
<td>cov954</td>
<td>64</td>
<td>0.0</td>
</tr>
<tr>
<td>sts27</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>sts45</td>
<td>10</td>
<td>0.0</td>
</tr>
<tr>
<td>sts63</td>
<td>1585</td>
<td>0.0</td>
</tr>
<tr>
<td>sts81</td>
<td>7200</td>
<td>13.1</td>
</tr>
</tbody>
</table>
Football Fail!

- Orbital Branching and Isomorphism Pruning solve codbt05 super fast
- The football pool problem is codbt06
Football Fail!

- Orbital Branching and Isomorphism Pruning solve codbt05 super fast
- The football pool problem is codbt06
- These methods (by themself) fail to make progress on the football pool problem
Football Fail!

- Orbital Branching and Isomorphism Pruning solve codbt05 super fast
- The football pool problem is codbt06
- These methods (by themself) fail to make progress on the football pool problem

Key Idea!

- Enumerate “necessary conditions” for there to exist an optimal solution (code) of value/cardinality $M$
- If for each “necessary” condition, no such code of value $M$ exists...
- The smallest code must be of cardinality at least $M + 1$
Necessary Conditions by Subcode Enumeration

- Partition 729 outcomes/tickets $W$ by the outcome of the first match
- $W = W_0 \cup W_1 \cup W_2$
- $w \in W_0$ covers 11 outcomes in $W_0$
- Ticket $w \in W_1$ covers 1 outcome in $W_0$
- Ticket $w \in W_2$ covers 1 outcome in $W_0$
- An optimal “code” $C^*$ (solution to the problem) has
  - $C^*_0 \subset W_0$, $|C^*_0| \overset{\text{def}}{=} y_0$
  - $C^*_1 \subset W_1$, $|C^*_1| \overset{\text{def}}{=} y_1$
  - $C^*_2 \subset W_2$, $|C^*_2| \overset{\text{def}}{=} y_2$
Necessary Conditions by Subcode Enumeration

- Partition 729 outcomes/tickets $W$ by the outcome of the first match
  - $W = W_0 \cup W_1 \cup W_2$
- Ticket $w \in W_0$ covers 11 outcomes in $W_0$
- Ticket $w \in W_1$ covers 1 outcome in $W_0$
- Ticket $w \in W_2$ covers 1 outcome in $W_0$
- An optimal "code" $C^*$ (solution to the problem) has
  - $C^*_0 \subset W_0, \quad |C^*_0| \overset{\text{def}}{=} y_0$
  - $C^*_1 \subset W_1, \quad |C^*_1| \overset{\text{def}}{=} y_1$
  - $C^*_2 \subset W_2, \quad |C^*_2| \overset{\text{def}}{=} y_2$

- So if a code of size $|C^*| = M$ exists, then it must satisfy

\[
\begin{align*}
11y_0 + y_1 + y_2 & \geq 243 \\
y_0 + 11y_1 + y_2 & \geq 243 \\
y_0 + y_1 + 11y_2 & \geq 243 \\
y_0 + y_1 + y_2 & = M
\end{align*}
\]
**Sequence IP** \((M, y_0, y_1, y_2)\)

- Enumerate *all* (non-isomorphic) integer solutions \((y_0, y_1, y_2)\) to the covering system
- Then solve...

\[
\min 1^T x
\]

\[
s.t. \quad Ax \geq 1
\]
\[
\sum_{i \in W_0} x_i = y_0
\]
\[
\sum_{i \in W_1} x_i = y_1
\]
\[
\sum_{i \in W_2} x_i = y_2
\]
\[
1^T x \leq M
\]
\[
x \in \{0, 1\}^{|W|}
\]
Sequence IP \((M, y_0, y_1, y_2)\)

- Enumerate all (non-isomorphic) integer solutions \((y_0, y_1, y_2)\) to the covering system.
- Then solve...

\[
\min 1^T x \\
\text{s.t. } A x \geq 1 \\
\sum_{i \in W_0} x_i = y_0 \\
\sum_{i \in W_1} x_i = y_1 \\
\sum_{i \in W_2} x_i = y_2 \\
1^T x \leq M \\
x \in \{0, 1\}^{|W|}
\]

**Improving the Lower Bound**

- Solve for every (enumerated) sequence: \((y_0, y_1, y_2)\).
- If you find no solution, then \(M + 1\) is a valid lower bound.
Results of Preprocessing/Enumeration

- In the end, after even more tricks, we are left the following number of difficult, symmetric, integer programs to solve:

<table>
<thead>
<tr>
<th>M</th>
<th>#sequences Modified</th>
<th>#Sequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>66</td>
<td>797</td>
<td>7</td>
</tr>
<tr>
<td>67</td>
<td>1,723</td>
<td>13</td>
</tr>
<tr>
<td>68</td>
<td>3,640</td>
<td>45</td>
</tr>
<tr>
<td>69</td>
<td>7,527</td>
<td>102</td>
</tr>
<tr>
<td>70</td>
<td>13,600</td>
<td>176</td>
</tr>
<tr>
<td>71</td>
<td>24,023</td>
<td>264</td>
</tr>
<tr>
<td>72</td>
<td>40,431</td>
<td>393</td>
</tr>
</tbody>
</table>

Solving $M = 66, 67, 68$ IPs takes less than a week on a single CPU with isomorphism pruning. Other instances are (quite) difficult ⇒ we need a BIG computer.
Results of Preprocessing/Enumeration

In the end, after even more tricks, we are left the following number of difficult, symmetric, integer programs to solve:

<table>
<thead>
<tr>
<th>$M$</th>
<th>#sequences</th>
<th>Modified</th>
<th>#Sequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>66</td>
<td>797</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>67</td>
<td>1,723</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>68</td>
<td>3,640</td>
<td>0</td>
<td>45</td>
</tr>
<tr>
<td>69</td>
<td>7,527</td>
<td>0</td>
<td>102</td>
</tr>
<tr>
<td>70</td>
<td>13,600</td>
<td>0</td>
<td>176</td>
</tr>
<tr>
<td>71</td>
<td>24,023</td>
<td>0</td>
<td>264</td>
</tr>
<tr>
<td>72</td>
<td>40,431</td>
<td>0</td>
<td>393</td>
</tr>
</tbody>
</table>

1000
Results of Preprocessing/Enumeration

- In the end, after even more tricks, we are left the following number of difficult, symmetric, integer programs to solve:

<table>
<thead>
<tr>
<th>$M$</th>
<th>#sequences</th>
<th>Modified</th>
<th>#Sequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>66</td>
<td>797</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>1,723</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>3,640</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>7,527</td>
<td>102</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>13,600</td>
<td>176</td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>24,023</td>
<td>264</td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>40,431</td>
<td>393</td>
<td></td>
</tr>
</tbody>
</table>

Solving $M = 66, 67, 68$ IPs takes less than a week on a single CPU with isomorphism pruning.

Other instances are (quite) difficult $\Rightarrow$ we need a **BIG** computer.
## Computational Results

### Computational Grid

<table>
<thead>
<tr>
<th>Site</th>
<th>Access Method</th>
<th>Arch/OS</th>
<th>Machines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wisconsin - CS</td>
<td>Flocking</td>
<td>x86.32/Linux</td>
<td>975</td>
</tr>
<tr>
<td>Wisconsin - CS</td>
<td>Flocking</td>
<td>Windows</td>
<td>126</td>
</tr>
<tr>
<td>Wisconsin - CAE</td>
<td>Remote submit</td>
<td>x86.32/Linux</td>
<td>89</td>
</tr>
<tr>
<td>Wisconsin - CAE</td>
<td>Remote submit</td>
<td>Windows</td>
<td>936</td>
</tr>
<tr>
<td>Lehigh - COR@L Lab</td>
<td>Flocking</td>
<td>x86.32/Linux</td>
<td>57</td>
</tr>
<tr>
<td>Lehigh - Campus desktops</td>
<td>Remote Submit</td>
<td>Windows</td>
<td>803</td>
</tr>
<tr>
<td>Lehigh - Beowulf</td>
<td>ssh + Remote Submit</td>
<td>x86.32</td>
<td>184</td>
</tr>
<tr>
<td>Lehigh - Beowulf</td>
<td>ssh + Remote Submit</td>
<td>x86.64</td>
<td>120</td>
</tr>
<tr>
<td>OSG - Wisconsin</td>
<td>Schedd-on-side</td>
<td>x86.32/Linux</td>
<td>1000</td>
</tr>
<tr>
<td>OSG - Nebraska</td>
<td>Schedd-on-side</td>
<td>x86.32/Linux</td>
<td>200</td>
</tr>
<tr>
<td>OSG - Caltech</td>
<td>Schedd-on-side</td>
<td>x86.32/Linux</td>
<td>500</td>
</tr>
<tr>
<td>OSG - Arkansas</td>
<td>Schedd-on-side</td>
<td>x86.32/Linux</td>
<td>8</td>
</tr>
<tr>
<td>OSG - BNL</td>
<td>Schedd-on-side</td>
<td>x86.32/Linux</td>
<td>250</td>
</tr>
<tr>
<td>OSG - MIT</td>
<td>Schedd-on-side</td>
<td>x86.32/Linux</td>
<td>200</td>
</tr>
<tr>
<td>OSG - Purdue</td>
<td>Schedd-on-side</td>
<td>x86.32/Linux</td>
<td>500</td>
</tr>
<tr>
<td>OSG - Florida</td>
<td>Schedd-on-side</td>
<td>x86.32/Linux</td>
<td>100</td>
</tr>
</tbody>
</table>
Computational Grid, cont.

<table>
<thead>
<tr>
<th>Site</th>
<th>Access Method</th>
<th>Arch/OS</th>
<th>Machines</th>
</tr>
</thead>
<tbody>
<tr>
<td>TG - NCSA</td>
<td>Flocking</td>
<td>x86_32/Linux</td>
<td>494</td>
</tr>
<tr>
<td>TG - NCSA</td>
<td>Flocking</td>
<td>x86_64/Linux</td>
<td>406</td>
</tr>
<tr>
<td>TG - NCSA</td>
<td>Hobble-in</td>
<td>ia64-linux</td>
<td>1732</td>
</tr>
<tr>
<td>TG - ANL/UC</td>
<td>Hobble-in</td>
<td>ia-32/Linux</td>
<td>192</td>
</tr>
<tr>
<td>TG - ANL/UC</td>
<td>Hobble-in</td>
<td>ia-64/Linux</td>
<td>128</td>
</tr>
<tr>
<td>TG - TACC</td>
<td>Hobble-in</td>
<td>x86_64/Linux</td>
<td>5100</td>
</tr>
<tr>
<td>TG - SDSC</td>
<td>Hobble-in</td>
<td>ia-64/Linux</td>
<td>524</td>
</tr>
<tr>
<td>TG - Purdue</td>
<td>Remote Submit</td>
<td>x86_32/Linux</td>
<td>1099</td>
</tr>
<tr>
<td>TG - Purdue</td>
<td>Remote Submit</td>
<td>x86_64/Linux</td>
<td>1529</td>
</tr>
<tr>
<td>TG - Purdue</td>
<td>Remote Submit</td>
<td>Windows</td>
<td>1460</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>19,012</strong></td>
</tr>
</tbody>
</table>

Grid 2.0

Grid technology has really “taken off,” since now I need two slides to list all of my resources.
Jealous?

- 19,012 processors sounds great, but sadly they aren’t all mine.
- I can only use them when other, more important people aren’t using them.
- This is the whole notion behind a concept called the computational grid.
Jealous?

- 19,012 processors sounds great, but sadly they aren’t all mine.
- I can only use them when other, more important people aren’t using them.
- This is the whole notion behind a concept called the computational grid

- HTCondor provides infrastructure for doing this type of computing.
- But still need to control the branch and bound algorithm.
- Computations must be flexible—fault tolerant and dynamic.
- Master-Worker: Isomorphism-Pruning enhanced branch and bound code was parallelized using software using MW.
Large Scale Computation

- We solved the (symmetric) IPs on this collection of machines over a period of a few months
Large Scale Computation

- We solved the (symmetric) IPs on this collection of machines over a period of a few months

<table>
<thead>
<tr>
<th></th>
<th>$M = 69$</th>
<th>$M = 70$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. Workers</td>
<td>555.8</td>
<td>562.4</td>
</tr>
<tr>
<td>Max Workers</td>
<td>2038</td>
<td>1775</td>
</tr>
<tr>
<td>Worker Time (years)</td>
<td>110.1</td>
<td>30.3</td>
</tr>
<tr>
<td>Wall Time (days)</td>
<td>72.3</td>
<td>19.7</td>
</tr>
<tr>
<td>Worker Util.</td>
<td>90%</td>
<td>71%</td>
</tr>
<tr>
<td>Nodes</td>
<td>$2.85 \times 10^9$</td>
<td>$1.89 \times 10^8$</td>
</tr>
<tr>
<td>LP Pivots</td>
<td>$2.65 \times 10^{12}$</td>
<td>$1.82 \times 10^{11}$</td>
</tr>
</tbody>
</table>
Simultaneous Workers, $M = 71$ attempt
Why Did I Stop, You Ask?

Global Warming Is All My Fault

200 CPU Years = 1.752M CPU Hours.
≈ 500 W per CPU hour
⇒ 876 MWH for the calculation.
⇒ Roughly 1.1388 million pounds (569 tons) of CO2 produced

Car Travel
Prius produces around one ton of CO2 for 5988 miles
⇒ I could drive my Prius about 3.4 million miles.

"Don't worry about that."
—Arnold, on the environment
Why Did I Stop, You Ask?

Global Warming Is All My Fault

- 200 CPU Years $= 1.752 \text{M CPU Hours}$. $\approx 500 \text{ W per CPU hour}$ $\Rightarrow$ 876 MWH for the calculation.

- $\Rightarrow$ Roughly 1.1388 million pounds (569 tons) of CO2 produced
Why Did I Stop, You Ask?

Global Warming Is All My Fault

- 200 CPU Years = 1.752M CPU Hours. ≈ 500 W per CPU hour ⇒ 876 MWH for the calculation.
- ⇒ Roughly 1.1388 million pounds (569 tons) of CO2 produced

Car Travel

- Prius produces around one ton of CO2 for 5988 miles
- ⇒ I could dive my Prius about 3.4 million miles.
Why Did I Stop, You Ask?

Global Warming Is All My Fault

- 200 CPU Years = 1.752M CPU Hours. ≈ 500 W per CPU hour ⇒ 876 MWH for the calculation.
- ⇒ Roughly 1.1388 million pounds (569 tons) of CO2 produced

Car Travel

- Prius produces around one ton of CO2 for 5988 miles
- ⇒ I could dive my Prius about 3.4 million miles.

“Don’t worry about that.”

—Arnold, on the environment
Conclusions

- Don’t blame Jeff for the cold.
- Some Symmetric IPs are still very difficult
- One may branch on any variable and still do isomorphism pruning.
- Still more work to determine how to best exploit this flexibility
Conclusions

- Don’t blame Jeff for the cold.
- Some Symmetric IPs are still very difficult
- One may branch on any variable and still do isomorphism pruning.
- Still more work to determine how to best exploit this flexibility

Thank you!

Any Questions?
Plans for Tonight

“Milk is for babies. When you grow up you have to drink beer.”

— Arnold Schwarzenegger, *Pumping Iron*
Plans for Tonight

“Milk is for babies. When you grow up you have to drink beer.”

— Arnold Schwarzenegger, *Pumping Iron*

- We plan to go to the Surly Brewing Company, 520 Malcolm Ave SE. We will leave East Bank rail station at 6:00. So we should be there by 6:20(?) 6:30 if we decide to walk...
Publications


