Efficient Propagators for Global Constraints

Claude-Guy Quimper
Supervisor: Alejandro López-Ortiz

University of Waterloo
Outline

- My first contact with constraint programming
- The all-different constraint
- The global cardinality constraint
- The inter-distance constraint
- Post-doctoral work
My first contact with constraint programming

- I took Peter’s course in Constraint Programming
- The field requires efficient algorithms that are executed gazillions of times.
- Project: To implement Thiel and Mehlhorn’s alldiff propagator.

Peter van Beek
The All-Different Constraint

\[ \text{All-Different}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

- Scheduling: We want execution times to be all different.
- Encoding permutations.
- Sometimes, one simply wants things to be different!
The All-Different Constraint

\textsc{All-Different}(X_1, \ldots, X_n) \iff X_i \neq X_j
The All-Different Constraint

\[ \text{All-Different}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

| Régis '94 | Domain | \( O(n^{1.5}d) \) |
The All-Different Constraint

\[
X_1 \in \{2, 4, 5\} \\
X_2 \in \{3, 5\} \\
X_3 \in \{1, 3\} \\
X_4 \in \{2, 3\}
\]

Domain Consistency (GAC)
The All-Different Constraint

All-Different \((X_1, \ldots, X_n)\) \(\iff\) \(X_i \neq X_j\)

Régin '94
Domain \(O(n^{1.5}d)\)

Domain Consistency (GAC)

\[
X_1 \in \{\ 2 \ 4 \ 5 \}\ \\
X_2 \in \{\ 3 \ 5 \}\ \\
X_3 \in \{1 \ 3 \}\ \\
X_4 \in \{\ 2 \ 3 \}\ \\
\]

Remove all inconsistent values
The All-Different Constraint

All-Different\(\left(X_1, \ldots, X_n\right)\) \(\iff\) \(X_i \neq X_j\)

Régin '94 Domain \(O(n^{1.5d})\)

Domain Consistency (GAC)

\[
\begin{align*}
X_1 & \in \{2, 4\} \\
X_2 & \in \{5\} \\
X_3 & \in \{1, 3\} \\
X_4 & \in \{2\}
\end{align*}
\]

Remove all inconsistent values
The All-Different Constraint

\[ \text{ALL-DIFFERENT}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

Régis '94

Domain

\[ O(n^{1.5}d) \]
The All-Different Constraint

\[ \text{ALL-DIFFERENT}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

<table>
<thead>
<tr>
<th>Author</th>
<th>Type</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Régis '94</td>
<td>Domain</td>
<td>(O(n^{1.5}d))</td>
</tr>
<tr>
<td>Leconte '96</td>
<td>Range</td>
<td>(O(n^2))</td>
</tr>
<tr>
<td>Puget '98</td>
<td>Bounds</td>
<td>(O(n \log n))</td>
</tr>
<tr>
<td>Mehlhorn &amp; Thiel</td>
<td>Bounds</td>
<td>(O(n))</td>
</tr>
<tr>
<td>López-Ortiz, Quimper, Tromp, &amp; van Beek</td>
<td>Bounds</td>
<td>(O(n))</td>
</tr>
</tbody>
</table>
The All-Different Constraint

All-Different \((X_1, \ldots, X_n)\) ⇐⇒ \(X_i \neq X_j\)

Régin '94 Domain \(O(n^{1.5}d)\)
Leconte '96 Range \(O(n^2)\)
Puget '98 Bounds \(O(n \log n)\)
Mehlhorn & Thiel Bounds \(O(n)\)
López-Ortiz, Quimper, Tromp, & van Beek Bounds \(O(n)\)

Range Consistency
1) Make domains intervals
2) Remove all inconsistent values

\[
\begin{align*}
X_1 &\in \{2, 4, 5\} \\
X_2 &\in \{3, 5\} \\
X_3 &\in \{1, 3\} \\
X_4 &\in \{2, 3\}
\end{align*}
\]
The All-Different Constraint

All-Different\((X_1, \ldots, X_n)\) $\iff X_i \neq X_j$

Régin '94 Domain $O(n^{1.5}d)$
Leconte '96 Range $O(n^2)$
Puget '98 Bounds $O(n \log n)$
Mehlhorn & Thiel Bounds $O(n)$
López-Ortiz, Quimper, Tromp, & van Beek Bounds $O(n)$

Range Consistency

1) Make domains intervals
2) Remove all inconsistent values

$X_1 \in \{2, 3, 4, 5\}$
$X_2 \in \{3, 4, 5\}$
$X_3 \in \{1, 2, 3\}$
$X_4 \in \{2, 3\}$
The All-Different Constraint

Range Consistency

1) Make domains intervals
2) Remove all inconsistent values

\[
\begin{align*}
X_1 & \in \{ 2, 3, 4 \} \\
X_2 & \in \{ 3, 5 \} \\
X_3 & \in \{ 1 \} \\
X_4 & \in \{ 2, 3 \}
\end{align*}
\]
The All-Different Constraint

\[ \text{ALL-DIFFERENT}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Régin '94</td>
<td>Domain</td>
<td>( O(n^{1.5}d) )</td>
</tr>
<tr>
<td>Leconte '96</td>
<td>Range</td>
<td>( O(n^2) )</td>
</tr>
<tr>
<td>Puget '98</td>
<td>Bounds</td>
<td>( O(n \log n) )</td>
</tr>
</tbody>
</table>
The All-Different Constraint

The All-Different Constraint

\[ (X_1, \ldots, X_n) \text{ is All-Different} \iff X_i \neq X_j \quad (i \neq j) \]

Bounds Consistency

1) Make domains intervals
2) Shrink intervals

\[ X_1 \in \{2, 4, 5\} \]
\[ X_2 \in \{3, 5\} \]
\[ X_3 \in \{1, 3\} \]
\[ X_4 \in \{2, 3\} \]
The All-Different Constraint

All-Different \((X_1, \ldots, X_n)\) \iff \(X_i \neq X_j\)

Bounds Consistency

1) Make domains intervals
2) Shrink intervals
The All-Different Constraint

Bounds Consistency

\[ X_1 \in \{2, 3\} \]
\[ X_2 \in \{4, 5\} \]
\[ X_3 \in \{1, 2\} \]
\[ X_4 \in \{2, 3\} \]

1) Make domains intervals
2) Shrink intervals
The All-Different Constraint

\[ \text{All-Different}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

<table>
<thead>
<tr>
<th>Régis '94</th>
<th>Domain</th>
<th>(O(n^{1.5}d))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leconte '96</td>
<td>Range</td>
<td>(O(n^2))</td>
</tr>
<tr>
<td>Puget '98</td>
<td>Bounds</td>
<td>(O(n \log n))</td>
</tr>
<tr>
<td>Mehlhorn &amp; Thiel</td>
<td>Bounds</td>
<td>(O(n))</td>
</tr>
</tbody>
</table>
The All-Different Constraint

\[ \text{ALL-DIFFERENT}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Région '94</td>
<td>Domain</td>
<td>( O(n^{1.5d}) )</td>
</tr>
<tr>
<td>Leconte '96</td>
<td>Range</td>
<td>( O(n^2) )</td>
</tr>
<tr>
<td>Puget '98</td>
<td>Bounds</td>
<td>( O(n \log n) )</td>
</tr>
<tr>
<td>Mehlhorn &amp; Thiel</td>
<td>Bounds</td>
<td>( O(n) )</td>
</tr>
<tr>
<td>López-Ortiz, Quimper, Tromp, &amp; van Beek</td>
<td>Bounds</td>
<td>( O(n) )</td>
</tr>
</tbody>
</table>
Hall’s Marriage Theorem

\[ \text{dom}(X_1) = [3, 4] \]
\[ \text{dom}(X_2) = [3, 4] \]
\[ \text{dom}(X_3) = [1, 3] \]
Hall’s Marriage Theorem

\[ \text{dom}(X_1) = [3, 4] \]
\[ \text{dom}(X_2) = [3, 4] \]
\[ \text{dom}(X_3) = [1, 3] \]

\[ \text{A Hall interval is an interval of } k \text{ values that contains the domains of } k \text{ variables.} \]
Hall’s Marriage Theorem

\[
\begin{align*}
\text{dom}(X_1) &= [3, 4] \\
\text{dom}(X_2) &= [3, 4] \\
\text{dom}(X_3) &= [1, 3]
\end{align*}
\]

\[
\{\text{Hall interval}\}
\]

- A Hall interval is an interval of \(k\) values that contains the domains of \(k\) variables.
Hall’s Marriage Theorem

\[
\begin{align*}
\text{dom}(X_1) &= [3, 4] \\
\text{dom}(X_2) &= [3, 4] \\
\text{dom}(X_3) &= [1, 2]
\end{align*}
\]

\{ \text{Hall interval} \}

□ A Hall interval is an interval of \( k \) values that contains the domains of \( k \) variables.
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [3, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[ \text{dom}(X_1) = \{2, 3\} \]
\[ \text{dom}(X_2) = \{2, 3\} \]
\[ \text{dom}(X_3) = \{3, 4\} \]
\[ \text{dom}(X_4) = \{2, 6\} \]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [3, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{dom}(X_1) & = [2, 3] \\
\text{dom}(X_2) & = [2, 3] \\
\text{dom}(X_3) & = [3, 4] \\
\text{dom}(X_4) & = [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [3, 4] \\
\text{dom}(X_4) &= [2, 6] \\
\end{align*}
\]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [3, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [3, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [3, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [3, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
# A Propagator for the Bounds Consistency

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Hall

\[
\begin{align*}
\text{dom}(X_1) & = [2, 3] \\
\text{dom}(X_2) & = [2, 3] \\
\text{dom}(X_3) & = [3, 4] \\
\text{dom}(X_4) & = [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

Hall

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [4, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[ \text{dom}(X_1) = [2, 3] \]
\[ \text{dom}(X_2) = [2, 3] \]
\[ \text{dom}(X_3) = [4, 4] \]
\[ \text{dom}(X_4) = [2, 6] \]
A Propagator for the Bounds Consistency

\[ \text{dom}(X_1) = [2, 3] \]
\[ \text{dom}(X_2) = [2, 3] \]
\[ \text{dom}(X_3) = [4, 4] \]
\[ \text{dom}(X_4) = [2, 6] \]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [4, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [4, 4] \\
\text{dom}(X_4) &= [2, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [4, 4] \\
\text{dom}(X_4) &= [5, 6]
\end{align*}
\]
A Propagator for the Bounds Consistency

\[
\begin{align*}
\text{dom}(X_1) &= [2, 3] \\
\text{dom}(X_2) &= [2, 3] \\
\text{dom}(X_3) &= [4, 4] \\
\text{dom}(X_4) &= [5, 6]
\end{align*}
\]
## Analysis of the Algorithm

<table>
<thead>
<tr>
<th>Version</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>First version</td>
<td>$O(n^2 + m)$</td>
<td>6 lines of C code!</td>
</tr>
</tbody>
</table>
## Analysis of the Algorithm

<table>
<thead>
<tr>
<th>Version</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>First version</td>
<td>$O(n^2 + m)$</td>
<td>6 lines of C code!</td>
</tr>
<tr>
<td>Union-find data structure</td>
<td>$O(n \log n)$</td>
<td>The fastest in practice</td>
</tr>
</tbody>
</table>
Analysis of the Algorithm

<table>
<thead>
<tr>
<th>Version</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>First version</td>
<td>$O(n^2 + m)$</td>
<td>6 lines of C code!</td>
</tr>
<tr>
<td>Union-find data structure</td>
<td>$O(n \log n)$</td>
<td>The fastest in practice</td>
</tr>
<tr>
<td>Balanced union-find data structure</td>
<td>$O(n \alpha(n))$</td>
<td>Slightly slower than the previous version</td>
</tr>
</tbody>
</table>

*Note: $\alpha(n)$ is the inverse Ackermann function.*
## Analysis of the Algorithm

<table>
<thead>
<tr>
<th>Version</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>First version</td>
<td>$O(n^2 + m)$</td>
<td>6 lines of C code!</td>
</tr>
<tr>
<td>Union-find data structure</td>
<td>$O(n \log n)$</td>
<td>The fastest in practice</td>
</tr>
<tr>
<td>Balanced union-find data structure</td>
<td>$O(n \alpha(n))$</td>
<td>Slightly slower than the previous version</td>
</tr>
<tr>
<td>Gabow and Tarjan’s data structure</td>
<td>$O(n)$</td>
<td>Slower and pages of code</td>
</tr>
</tbody>
</table>
The Global Cardinality Constraint

$$\text{GCC}([X_1, \ldots, X_n], l, u) \iff \forall v \ l_v \leq |\{i \mid X_i = v\}| \leq u_v$$

- A value $v$ must be taken at least $l_v$ times and at most $u_v$ times.
The Global Cardinality Constraint

\[ \text{GCC}([X_1, \ldots, X_n], l, u) \iff \forall v \ l_v \leq |\{i \mid X_i = v\}| \leq u_v \]

- A value \( v \) must be taken at least \( l_v \) times and at most \( u_v \) times.
- Scheduling: No more than 2 tasks can be executed at a given time.
The Global Cardinality Constraint

\[ \text{GCC}([X_1, \ldots, X_n], l, u) \iff \forall v \ l_v \leq |\{i \mid X_i = v\}| \leq u_v \]

- A value \( v \) must be taken at least \( l_v \) times and at most \( u_v \) times.
- Scheduling: No more than 2 tasks can be executed at a given time.
- Sequencing: We want to restrict the number of occurrences of an event in a sequence.
The Global Cardinality Constraint

\[ \text{GCC}([X_1, \ldots, X_n], l, u) \iff \forall v \ l_v \leq |\{i \mid X_i = v\}| \leq u_v \]

[Régin '96] gives a propagator achieving domain consistency.
The Global Cardinality Constraint

\[ \text{GCC}([X_1, \ldots, X_n], l, u) \iff \forall v \ l_v \leq |\{i \mid X_i = v\}| \leq u_v \]

- [Régisn '96] gives a propagator achieving domain consistency.
- There were no propagators for bounds consistency.
Decomposing the GCC
Decomposing the GCC

The upper bound constraint (ubc)

Each value is assigned to at most 2 variables.
Decomposing the GCC

The upper bound constraint (ubc)

Each value is assigned to at most 2 variables.

The lower bound constraint (lbc)

Each value is assigned to at least 1 variable.
Decomposing the GCC

The upper bound constraint (ubc)
Each value is assigned to at most 2 variables.

The lower bound constraint (lbc)
Each value is assigned to at least 1 variable.

GCC
The Upper Bound Constraint

- All values must be assigned to at most 2 variables.

\[ X_1 : \{1\ 2\} \]
\[ X_2 : \{1\} \]
\[ X_3 : \{1\ 2\} \]
\[ X_4 : \{2\} \]
\[ X_5 : \{1\ 2\ 3\} \]
The Upper Bound Constraint

- All values must be assigned to at most 2 variables.

\[
\begin{align*}
X_1 &: \{1 \ 2\} \\
X_2 &: \{1\} \\
X_3 &: \{1 \ 2\} \\
X_4 &: \{2\} \\
X_5 &: \{1 \ 2 \ 3\}
\end{align*}
\]

\[S = \{1, 2\}\]
The Upper Bound Constraint

- All values must be assigned to at most 2 variables.

\[
X_1 : \{1 \quad 2\} \\
X_2 : \{1\} \\
X_3 : \{1 \quad 2\} \\
X_4 : \{2\} \\
X_5 : \{1 \quad 2 \quad 3\}
\]

\[
S = \{1, 2\}
\]

Upper capacity: \(\lceil S \rceil = 2 + 2 = 4\)
The Upper Bound Constraint

- All values must be assigned to at most 2 variables.

\[
S = \{1, 2\}
\]

Upper capacity: \( |S| = 2 + 2 = 4 \)
The Upper Bound Constraint

- All values must be assigned to at most 2 variables.

\[ X_1 : \{1 \ 2\} \subseteq S \]
\[ X_2 : \{1\} \subseteq S \]
\[ X_3 : \{1 \ 2\} \subseteq S \]
\[ X_4 : \{2\} \subseteq S \]
\[ X_5 : \{3\} \]

\[ S = \{1, 2\} \]

Upper capacity: \[ |S| = 2 + 2 = 4 \]
The Upper Bound Constraint

- All values must be assigned to at most 2 variables.

\[
\begin{align*}
X_1 : & \{1 \ 2\} \subseteq S \\
X_2 : & \{1\} \subseteq S \\
X_3 : & \{1 \ 2\} \subseteq S \\
X_4 : & \{2\} \subseteq S \\
X_5 : & \{3\} \\
\end{align*}
\]

\[ S = \{1, 2\} \]

Upper capacity: \[ |S| = 2 + 2 = 4 \]
A Propagator for the UBC

Similar to the one for the all-different Constraint.
A Propagator for the UBC

- Similar to the one for the all-different constraint.
- Values can have more than one bucket.
A Propagator for the UBC

- Similar to the one for the all-different constraint.

- Values can have more than one bucket.

1 2 3 4 5 6

- ✓ ✓ ✓ ✓ ✓ -
The Lower Bound Constraint

☐ All values must be assigned to at least 1 variable.

\[
\begin{align*}
X_1 & : \{ 1 \} \\
X_2 & : \{ 4 \} \\
X_3 & : \{ 1 4 \} \\
X_4 & : \{ 1 2 3 \} \\
X_5 & : \{ 2 3 4 \}
\end{align*}
\]
The Lower Bound Constraint

☐ All values must be assigned to at least 1 variable.

\[ \mathbf{X}_1 : \{1 \} \]
\[ \mathbf{X}_2 : \{4\} \]
\[ \mathbf{X}_3 : \{1, 4\} \]
\[ \mathbf{X}_4 : \{1, 2, 3\} \]
\[ \mathbf{X}_5 : \{2, 3, 4\} \]

\[ S = \{2, 3\} \]
The Lower Bound Constraint

All values must be assigned to at least 1 variable.

\[ X_1 : \{1 \} \]
\[ X_2 : \{4\} \]
\[ X_3 : \{1, 4\} \]
\[ X_4 : \{1, 2, 3\} \]
\[ X_5 : \{2, 3, 4\} \]

\[ S = \{2, 3\} \]

Lower capacity: \[ |S| = 1 + 1 = 2 \]
The Lower Bound Constraint

- All values must be assigned to at least 1 variable.

\[ X_1 : \{1 \} \]
\[ X_2 : \{4\} \]
\[ X_3 : \{1, 4\} \]
\[ X_4 : \{1, 2, 3\} \cap S \neq \emptyset \]
\[ X_5 : \{2, 3\} \cap S \neq \emptyset \]

\[ S = \{2, 3\} \]

Lower capacity: \[ [S] = 1 + 1 = 2 \]
The Lower Bound Constraint

☐ All values must be assigned to at least 1 variable.

\[ S = \{2, 3\} \]

Lower capacity: \( [S] = 1 + 1 = 2 \)
The Lower Bound Constraint

□ All values must be assigned to at least 1 variable.

\[ S = \{2, 3\} \]

Lower capacity: \[ [S] = 1 + 1 = 2 \]
A Propagator for the LBC

- We adapted the algorithm for the All-different constraint
- Detects unstable sets rather than Hall intervals.
- Time complexity: $O(n)$
The Global Cardinality Constraint

UBC

LBC

GCC
The Global Cardinality Constraint

\[ X = 4 \]
The Global Cardinality Constraint
The Global Cardinality Constraint

**Theorem:**
A value has a support in the GCC iff it has a support in the UBC and the LBC.

**Proof:**
Based on the relationship between Hall sets and unstable sets.

**Note:**
Holds for domain, range, and bounds consistency.
A Propagator for the GCC

Filter the UBC

Filter the LBC

Is the UBC Still Consistent?

No

Yes
A Propagator for the GCC

Theorem:
This algorithm never loops!

Proof:
Based on the relationship between Hall sets and unstable sets.

Note:
Holds for domain, range, and bounds consistency
Extended GCC

- $\text{EGCC}([X_1, \ldots, X_n], [C_1, \ldots, C_m])$ is satisfied when $v$ is taken $c_v$ times.
Extended GCC

- $\text{EGCC}([X_1, \ldots, X_n], [C_1, \ldots, C_m])$ is satisfied when $v$ is taken $C_v$ times.

**Theorem**

When domains are sets, testing the satisfiability of EGCC is NP-Hard.
Extended GCC

- $\text{EGCC}([X_1, \ldots, X_n], [C_1, \ldots, C_m])$ is satisfied when $v$ is taken $C_v$ times.

Theorem
When domains are sets, testing the satisfiability of EGCC is NP-Hard.

Theorem
When domains are intervals, filtering EGCC takes linear time.

Katriel & Thiel
Beyond Integer Domains

\textbf{All-Different}(X_1, \ldots, X_n) \iff X_i \neq X_j
Beyond Integer Domains

**All-Different**($X_1, \ldots, X_n$) $\iff$ $X_i \neq X_j$

- Variables could be sets, multi-sets, or tuples.
Beyond Integer Domains

\textbf{All-Different}(X_1, \ldots, X_n) \iff X_i \neq X_j

- variables could be sets, multi-sets, or tuples.
- Sets, multi-set, and tuple variables often have large domains.
Beyond Integer Domains

\[ \text{All-Different}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

- Variables could be sets, multi-sets, or tuples.
- Sets, multi-set, and tuple variables often have large domains.

\[
\{\} \subseteq X \subseteq \{1, \ldots, u\} \Rightarrow |X| = 2^u
\]
Beyond Integer Domains

\[ \text{All-Different}(X_1, \ldots, X_n) \iff X_i \neq X_j \]

- Variables could be sets, multi-sets, or tuples.
- Sets, multi-set, and tuple variables often have large domains.
- \( \emptyset \subseteq X \subseteq \{1, \ldots, u\} \Rightarrow |X| = 2^u \)
- We adapted the propagator to obtain a polynomial complexity: \( O(n^{2.5} + n^2 u) \)
The Inter-Distance Constraint

\[ \text{INTER-DISTANCE}([X_1, \ldots, X_n], p) \iff |X_i - X_j| \geq p \]

- There must be a gap of \( p \) between each variable.
The Inter-Distance Constraint

\[ \text{Inter-Distance}([X_1, \ldots, X_n], p) \iff |X_i - X_j| \geq p \]

- There must be a gap of \( p \) between each variable.
- When \( p = 1 \), we obtain the All-Different Constraint.
The Inter-Distance Constraint

\[ \text{INTER-DISTANCE}([X_1, \ldots, X_n], p) \iff |X_i - X_j| \geq p \]

- There must be a gap of \( p \) between each variable.
- When \( p = 1 \), we obtain the All-Different Constraint.
- Scheduling: Execution times must be \( p \) units of time apart.
The Inter-Distance Constraint

\[ \text{Inter-Distance}([X_1, \ldots, X_n], p) \iff |X_i - X_j| \geq p \]

- There must be a gap of \( p \) between each variable.
- When \( p = 1 \), we obtain the All-Different Constraint.
- Scheduling: Execution times must be \( p \) units of time apart.
- Radio frequency allocation problem.
The Inter-Distance Constraint

- [Régis ‘97] introduces the global minimum distance constraint.
The Inter-Distance Constraint

- [Régisn '97] introduces the global minimum distance constraint.
- [Artiouchine & Baptiste '05]
The Inter-Distance Constraint

- [Régis '97] introduces the global minimum distance constraint.
- [Artiouchine & Baptiste '05] prove the constraint is NP-Hard when variables are sets.
The Inter-Distance Constraint

- [Régin '97] introduces the global minimum distance constraint.
- [Artiouchine & Baptiste '05]
  - prove the constraint is NP-Hard when variables are sets.
  - achieve bounds consistency in cubic time.
Block Placement

- Place two blocks of size 4 on the axis without overlapping them.
Block Placement

- Place two blocks of size 4 on the axis without overlapping them.

- No block can have its left end inside a red zone.
Internal Adjustment Intervals

Artiouchine & Baptiste ‘05

1 2 3 4 5 6 7 8 9 10 11 12 13 14
Internal Adjustment Intervals

Artiouchine & Baptiste ‘05
Internal Adjustment Intervals

Artiouchine & Baptiste ‘05
Internal Adjustment Intervals

Artiouchine & Baptiste ‘05

1 2 3 4 5 6 7 8 9 10 11 12

13 14
Internal Adjustment Intervals

Artiouchine & Baptiste ‘05
Internal Adjustment Intervals

Artiouchine & Baptiste ‘05
Internal Adjustment Intervals

Artiouchnine & Baptiste ‘05
Block Placement
Block Placement

Place the 3 blocks on the axis such that the blue blocks are in the box.
Block Placement

Place the 3 blocks on the axis such that the blue blocks are in the box.
Block Placement

- Place the 3 blocks on the axis such that the blue blocks are in the box.
Block Placement

- Place the 3 blocks on the axis such that the blue blocks are in the box.

- The green box cannot have its left end inside a red zone.
If you place $n$ blue blocks of size one inside a box of size $n$, you obtain a red zone of $n$ elements.

This is a Hall interval!
Block Placement

- Place the 3 blocks on the axis such that the blue blocks are in the box.

- The green box cannot have its left end inside a red zone.
External Adjustment Intervals

Artiouchine & Baptiste ‘05
External Adjustment Intervals

Artiouchine & Baptiste ‘05
External Adjustment Intervals

Artiouchine & Baptiste ‘05
External Adjustment Intervals

Artiouchine & Baptiste ‘05
External Adjustment Intervals

Artiouchine & Baptiste ‘05
External Adjustment Intervals

Artiouchine & Baptiste ‘05
External Adjustment Intervals

Artiouchine & Baptiste ‘05
External Adjustment Intervals

Artiouchine & Baptiste ‘05
Number of Adjustment Intervals

\[ O(n^2) \times O(n) = O(n^3) \]
Number of Adjustment Intervals

\[ O(n^2) \times O(n) = O(n^3) \]

Number of intervals \([l, u]\)
Number of Adjustment Intervals

Number of red zones produced per interval

$O(n^2) \times O(n) = O(n^3)$

Number of intervals $[l, u]$
Number of Adjustment Intervals

Number of red zones produced per interval

\[ O(n^2) \times O(n) = O(n^3) \]

Number of intervals \([l, u]\)

Total number of red zones
Number of Adjustment Intervals

Number of red zones produced per interval

$O(n^2) \times O(n) = O(n^3)$

Total number of red zones

Complexity of Artiouchine & Baptiste’s propagator

Number of intervals $[l, u]$
Dominance

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\
\hline
l & & & & & & & & & & & \\
u & & & & & & & & & & & \\
\end{array}
\]
Dominance

\[\begin{array}{c|cccccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\
l &   &   &   &   &   &   &   &   &   &   &   &   \\
\end{array}\]
Dominance
Dominance

\[
\begin{array}{cccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\
\end{array}
\]
Dominance

\[
\begin{array}{c|cccccc|cccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\
l & & & & & & & & & & & \\
u & & & & & & & & & & & \\
\end{array}
\]
Dominance

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>l</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>u</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Theorem**

Only $O(n^2)$ red zones needs to be computed to achieve bounds consistency.
Propagator

- uses a special data structure to store the adjustment intervals
Propagator

- Uses a special data structure to store the adjustment intervals
- Time complexity: $O(n^2)$
Summary

- Bounds consistency for the All-Different Constraint.
Summary

- Bounds consistency for the All-Different Constraint.
- Generalization of Hall's marriage theorem for the GCC.
Summary

- Bounds consistency for the All-Different Constraint.
- Generalization of Hall’s marriage theorem for the GCC.
- Extension to non-integer domains
Summary

- Bounds consistency for the All-Different Constraint.
- Generalization of Hall’s marriage theorem for the GCC.
- Extension to non-integer domains
- Quadratic propagator for the Inter-Distance.
Life after the PhD

NICTA
Life after the PhD
Life after the PhD

NICTA

Microsoft Research

Omega Optimisation
Life after the PhD

NICTA

Microsoft Research

Omega Optimisation

École Polytechnique Montréal
Special Thanks to a Special Supervisor

Alex López-Ortiz
Special Thanks to Special Collaborators

Peter van Beek

Toby Walsh
Thanks to my Thesis Committee
Thanks to the ACP