Integer vs. constraint programming

Practical Problem Solving

- Model building: Language
- Model solving: Algorithms

IP vs. CP: Language

<table>
<thead>
<tr>
<th></th>
<th>IP</th>
<th>CP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
<td>(mostly) 0-1</td>
<td>Finite domain</td>
</tr>
<tr>
<td>Constraints</td>
<td>Linear equations and inequalities</td>
<td>Arithmetic constraints</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbolic/global constraints</td>
</tr>
</tbody>
</table>

Example

- Variables: $x_1, \ldots, x_n \in \{0, \ldots, m - 1\}$
- Constraint: Pairwise different values

Example (2)

- Integer programming: Only linear equations and inequalities
  \[
  x_i \neq x_j \iff x_i < x_j \lor x_i > x_j \iff x_i \leq x_j - 1 \lor x_i \geq x_j + 1
  \]
- Eliminating disjunction
  \[
  x_i - x_j + 1 \leq my_1, \quad x_j - x_i + 1 \leq my_2, \quad y_1 + y_2 = 1,
  \]
  \[
  y_1, y_2 \in \{0, 1\}, \quad 0 \leq x_i, x_j \leq m - 1,
  \]
- New variables: $z_{ik} = 1$ iff $x_i = k, i = 1, \ldots, n, k = 0, \ldots, m - 1$
  \[
  z_{i0} + \cdots + z_{im-1} = 1, \quad z_{1k} + \cdots + z_{mk} \leq 1,
  \]
- Constraint programming $\rightarrow$ symbolic constraint

  \[
  \text{alldifferent}(x_1, \ldots, x_n)
  \]

Symbolic/global constraints

- alldifferent([x_1, \ldots, x_n])
- cumulative([s_1, \ldots, s_n], [d_1, \ldots, d_n], [r_1, \ldots, r_n], c, e).
  - $n$ tasks: starting time $s_i$, duration $d_i$, resource demand $r_i$
  - resource capacity $c$, completion time $e$
Diffn Constraint

Beldiceanu/Contejean’94

- Nonoverlapping of \( n \)-dimensional rectangles \([O_1, \ldots, O_n, L_1, \ldots, L_n]\), where \( O_i \) (resp. \( L_i \)) denotes the origin (resp. length) in dimension \( i \)

- \( \text{diffn}([O_{11}, \ldots, O_{1n}, L_{11}, \ldots, L_{1n}], \ldots, [O_{m1}, \ldots, O_{mn}, L_{m1}, \ldots, L_{mn}]) \)

- **General form:** \( \text{diffn}(\text{Rectangles}, \text{Min}\_\text{Vol}, \text{Max}\_\text{Vol}, \text{End}, \text{Distances}, \text{Regions}) \)

**IP vs. CP: Algorithms**

<table>
<thead>
<tr>
<th></th>
<th>IP</th>
<th>CP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Inference</strong></td>
<td>Linear programming</td>
<td>Domain filtering</td>
</tr>
<tr>
<td></td>
<td>Cutting planes</td>
<td>Constraint propagation</td>
</tr>
<tr>
<td><strong>Search</strong></td>
<td>Branch-and-relax</td>
<td>Branch-and-bound</td>
</tr>
<tr>
<td></td>
<td>Branch-and-cut</td>
<td></td>
</tr>
<tr>
<td><strong>Bounds on the objective function</strong></td>
<td>Two-sided</td>
<td>One-sided</td>
</tr>
</tbody>
</table>

**Local vs. global reasoning**

Linear arithmetic constraints

\[
3x + y \leq 7,
3y + x \leq 7,
x + y = z,
x, y \in \{0, \ldots, 3\}
\]
Global reasoning in CP ? $\Rightarrow$ global constraints!

Global reasoning in CP

Example

- $x_1, x_2, x_3 \in \{0,1\}$
- pairwise different values
- **Local** consistency: 3 disequalities: $x_1 \neq x_2, x_1 \neq x_3, x_2 \neq x_3$
  $\Rightarrow$ $x_1, x_2, x_3 \in \{0,1\}$, i.e., no domain reduction is possible
- **Global** constraint: alldifferent($x_1, x_2, x_3$)
  $\Rightarrow$ detects infeasibility (uses bipartite matching)

Global reasoning in CP: inside global constraints

Summary

<table>
<thead>
<tr>
<th></th>
<th>ILP</th>
<th>CP(FD)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language</strong></td>
<td>Linear arithmetic</td>
<td>Arithmetic constraints</td>
</tr>
<tr>
<td></td>
<td>—</td>
<td>Symbolic constraints</td>
</tr>
<tr>
<td><strong>Algorithms</strong></td>
<td>Global consistency (LP)</td>
<td>Local consistency</td>
</tr>
<tr>
<td></td>
<td>Cutting planes</td>
<td>Domain reduction</td>
</tr>
<tr>
<td></td>
<td>Branch-and-bound</td>
<td>User-defined enumeration</td>
</tr>
<tr>
<td></td>
<td>Branch-and-cut</td>
<td></td>
</tr>
</tbody>
</table>

- Symbolic constraints $\Rightarrow$ more expressivity + more efficiency
- Unifying framework for CP and IP: *Branch-and-infer*  
  (Bockmayr/Kasper 98), . . . , SCIP

Discrete Tomography

- Binary matrix with $m$ rows and $n$ columns
  - Horizontal projection numbers ($h_1, \ldots, h_m$)
  - Vertical projection numbers ($v_1, \ldots, v_n$)
Properties

- Horizontal convexity (h)
- Vertical convexity (v)
- Connectivity (polyomino) (p)

Complexity (Woeginger’01)

- Polynomial: ( ), (p,v,h)
- NP-complete: (p,v), (p,h), (v,h), (v), (h), (p)

IP Model

Variables \( x_{ij} \) =

- 0 cell\((i,j)\) is labeled white
- 1 cell\((i,j)\) is labeled black

Constraints I: Projections

\[ \sum_{j=1}^{n} x_{ij} = h_i, \quad \sum_{i=1}^{m} x_{ij} = v_j \]

Constraints II: Convexity

\[ h_i x_{ik} + \sum_{l=k+1}^{n} x_{il} \leq h_i, \quad v_j x_{kj} + \sum_{l=k+1}^{m} x_{lj} \leq v_j, \]

IP Model (contd)

Constraints III: Connectivity

\[ \sum_{k=j}^{j+h_i-1} x_{ik} - \sum_{k=j}^{j+h_i-1} x_{i+1,k} \leq h_i - 1 \]

Various linear arithmetic models possible, e.g. convexity

Enormous differences in size and running time, e.g. 1 day vs. < 1 sec

Large number of constraints (\(~3mn\) in the above model)

Finite Domain Model

Variables

- \( x_i \) start of horizontal convex block in row \( i \), for \( 1 \leq i \leq m \)
- \( y_j \) start of vertical convex block in column \( j \), for \( 1 \leq j \leq n \)
• **Domain**
  - $x_i \in [1, \ldots, n - h_i + 1]$, for $1 \leq i \leq m$
  - $y_j \in [1, \ldots, m - v_j + 1]$, for $1 \leq j \leq n$

**Conditional Propagation**

• Projection/Convexity modelled by FD variables

• **Compatibility of $x_i$ and $y_j$**

$$x_i \leq j < x_i + h_i \iff y_j \leq i < y_j + v_j$$

for $1 \leq i \leq m$ and $1 \leq j \leq n$

• **Conditional propagation**

$$\text{if } x_i \leq j \text{ then (if } j < x_i + h_i \text{ then } (y_j \leq i, i < y_j + v_j))$$

**Finite Domain Model (contd)**

• **Connectivity**

• **Block $i$ must start before the end of block $i + 1$**

$$x_i \leq x_{i+1} + h_{i+1} - 1, \text{ for } 1 \leq i \leq m - 1$$

• **Block $i+1$ must start before the end of block $i$**

$$x_{i+1} \leq x_i + h_i - 1, \text{ for } 1 \leq i \leq m - 1$$

**Cumulative**
Propositional satisfiability

- $x_1, \ldots, x_n \in \{0,1\}$ 0-1 variables (or atomic formulae in propositional logic)
- A literal $L$ is a 0-1 variable $x$ or its negation $\overline{x} = 1 - x$.
- A clause is set of literals $C = \{L_1, \ldots, L_k\}$ corresponding to the logical disjunction $L_1 \vee \cdots \vee L_k$ or the clausal inequality $L_1 + \cdots + L_k \geq 1$.
- A clause set is a set of clauses $S = \{C_1, \ldots, C_m\}$ corresponding to the logical conjunction $C_1 \wedge \cdots \wedge C_m$.
- A clause set $S$ is satisfiable if there exists an assignment $I: \{x_1, \ldots, x_n\} \rightarrow \{0,1\}$ making the logical formula true (equivalently, if the system of clausal inequalities has a 0-1 solution).

Examples

1. Clause set
   \[ S = \{ \{x_1, x_2\}, \{\overline{x_1}, \overline{x_2}\} \} \]
   Corresponding logical formula: $(x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2})$
   Corresponding system of clausal inequalities: $x_1 + x_2 \geq 1, -x_1 - x_2 \geq -1$
   Satisfying assignments: $I(x_1) = 1, I(x_2) = 0$ and $I'(x_1) = 0, I'(x_2) = 1$.

2. The clause set
   \[ S = \{ \{x_1, x_2\}, \{\overline{x_1}, \overline{x_2}\}, \{x_1, \overline{x_2}\}, \{\overline{x_1}, x_2\} \} \]
   is unsatisfiable.
### SAT problem

- **SAT problem**: Given a set of clauses $S$, is $S$ satisfiable?

- **Theorem (Cook’71)**: SAT is NP-complete.

- There exist highly efficient SAT solvers.

- Enormous progress has been made during the last 10-15 years, see e.g. [http://www.satlive.org/](http://www.satlive.org/) → SAT competitions

- SAT is a third general approach for solving constraint satisfaction/optimization problems (in addition to IP and CP)

### Davis-Putnam procedure

```plaintext
function Satisfiable(S) return boolean
/* unit resolution */
repeat
    for each clause $\{L\}$ in $S$ of length 1 do
        delete from $S$ every clause containing $L$
        delete $\overline{I}$ in every clause of $S$ containing $\overline{I}$
    od
    if $S = \emptyset$ then true
    else if $S$ contains the empty clause $C = \emptyset$ then false
    fi
until no further changes /* branching */
choose a literal $L$ occurring in $S$
if Satisfiable($S \cup \{L\}$) then true
else if Satisfiable($S \cup \{\overline{L}\}$) then true
else false
fi
end function
```

### Example

Let $S$ be the clause set

\[
\begin{align*}
\{ & x_1, x_2, x_3, x_4, x_5 \}, \\
\{ & x_1, x_2, x_3, \overline{x_4} \}, \\
\{ & \overline{x_1}, x_2, \overline{x_3}, x_5 \}, \\
\{ & \overline{x_2}, x_3, \overline{x_5} \}, \\
\{ & \overline{x_1}, x_2, \overline{x_5} \}, \\
\{ & x_1, \overline{x_2}, \overline{x_5} \}, \\
\{ & \overline{x_5} \}.
\end{align*}
\]

Satisfying assignment: $I(x_1) = 0$, $I(x_2) = I(x_3) = 1$, $I(x_5) = 0$. 