Some Incomplete Constraint Solvers
Pages: 178 – 196
Andreas Anderson
2004/02/26

Objectives

- Introduce specialized techniques for some specific domains and constraints for which no efficient solving methods are known to exist
- Incomplete constraint solvers
  - Equality and disequality constraints
  - Boolean constraints
  - Linear constraints on integer intervals
    - Domain reduction rules for inequality constraints

Incomplete constraint solvers

- Equivalence preserving reduction rules
- If the constraints are defined using a specified language, the general rules of Chapter 5 cannot be used
- the general framework is a customized to a specific language in which the constraints are defined and to specific domains that are used

Lemma 6.1 (Hyper-arc Consistency)

- Consider a hyper-arc consistent CSP $P$. Then $P$ is closed under the application of every domain reduction rule which is
  - equivalence preserving, and
  - has only one constraint in its premise
- If a constraint solver imposes hyper-arc consistency, then it achieves the optimal domain reduction
Equality and Disequality Constraints

• =, ≠
• Arbitrary domains

Equality and Disequality Rules (1/2)

\[
\begin{align*}
\text{EQUALLY 1} & \quad \langle x = x : x \in D \rangle \\
\quad & \quad \langle ; x \in D \rangle \\
\text{EQUALLY 2} & \quad \langle x = y : x \in D_x, y \in D_y \rangle \\
\quad & \quad \langle x = y : x \in D_x \cap D_y, y \in D_x \cap D_y \rangle 
\end{align*}
\]

Equality and Disequality Rules (2/2)

\[
\begin{align*}
\text{DISEQUALITY 1} & \quad \langle x \neq x : x \in D \rangle \\
\quad & \quad \langle ; x \in \emptyset \rangle \\
\text{DISEQUALITY 2} & \quad \langle x \neq y : x \in D_x, y \in D_y \rangle \\
\quad & \quad \langle ; x \in D_x, y \in D_y \rangle \\
\text{where } D_x \cap D_y = \emptyset; \\
\text{DISEQUALITY 3} & \quad \langle x \neq y : x \in D, y = a \rangle \\
\quad & \quad \langle ; x \in D - \{a\}, y = a \rangle \\
\text{DISEQUALITY 4} & \quad \langle x \neq y : x = a, y \in D \rangle \\
\quad & \quad \langle ; x = a, y \in D - \{a\} \rangle \\
\text{where } a \in D
\end{align*}
\]

Proof System EQU

• **Theorem 6.2 (EQU)** A CSP with only equality and disequality constraints is hyper-arc consistent iff it is closed under the applications of the proof rules of the proof system EQU
• A constraint solver determined by the EQU rules is incomplete
• However are very simple operations on the variable domain and thus easy to implement
**Boolean Constraints (1/2)**

- **Boolean variables**: range over \{0, 1\}
- **Boolean domain expression**: \(x \in D\) where \(D \subseteq \{0, 1\}\)
- **Boolean expression**: built out of Boolean variables using the connectives: \(\neg, \land, \lor\)
- **Boolean constraint**: \(s = t\), where \(s\) and \(t\) are Boolean expressions
- **Boolean CSP**: a CSP with *Boolean domain expressions* and *Boolean constraints*
- The predefined variables \(x_T\) and \(x_F\) represent the Boolean Constants *true* and *false*

**Transformation Rules (1/2)**

\[
\frac{\langle -s = t; D \rangle}{\langle -x = t, s = x; D, x \in \{0, 1\} \rangle}
\]

where \(s\) is not a variable and \(x\) is a new variable

\[
\frac{\langle -s = t; D \rangle}{\langle -s = y, t = y; D, y \in \{0, 1\} \rangle}
\]

where \(t\) is not a variable and \(y\) is a new variable

\[
\frac{\langle s \quad op \quad t = u; D \rangle}{\langle s \quad op \quad t = z, u = z; D, x \in \{0, 1\} \rangle}
\]

where \(u\) is not a variable or a variable identical to \(s\) or \(t\), \(x\) is a new variable and \(op\) is \(\land\) or \(\lor\)

**Boolean Constraints (2/2)**

- **Simple** Boolean constraints:
  - **Equity constraint**: \(x = y\)
  - **NOT constraint**: \(\neg x = y\)
  - **AND constraint**: \(x \land y = z\)
  - **OR constraint**: \(x \lor y = z\)
- **Compound** Boolean constraints can be reduced to simple Boolean constraints

**Transformation Rules (2/2)**

\[
\frac{\langle s \quad op \quad t = u; D \rangle}{\langle x \quad op \quad t = u, s = x; D, x \in \{0, 1\} \rangle}
\]

where \(s\) is not a variable or a variable identical to \(t\) or \(u\), \(x\) is a new variable and \(op\) is \(\land\) or \(\lor\)

\[
\frac{\langle s \quad op \quad t = u; D \rangle}{\langle s \quad op \quad y = u, t = y; D, y \in \{0, 1\} \rangle}
\]

where \(t\) is not a variable or a variable identical to \(s\) or \(u\), \(y\) is a new variable and \(op\) is \(\land\) or \(\lor\)
Domain Reduction Rules

- For simple Boolean constraints

\[
\langle x \land y = z; x = 1, y \in D_y, z \in D_z \rangle \\
\rightarrow \langle y = z; x = 1, y \in D_y, z \in D_z \rangle
\]

Written as

\[x \land y = z, x = 1 \rightarrow y = z\]

Proof System BOOL (1/2)

- Theorem 6.3 (BOOL) A non-failed Boolean CSP is hyper-arc consistent iff it is closed under the application of the rules of the proof system BOOL.

Example: Full Adder Circuit (1/2)

\[
(i_1 \oplus i_2) \oplus i_3 = o_1 \\
(i_1 \land i_2) \lor (i_3 \land (i_1 \oplus i_2)) = o_2
\]

Proof rules for XOR

- XOR 1 \[x \oplus y = z, x = 1, y = 1 \rightarrow z = 0\]
- XOR 2 \[x \oplus y = z, x = 0, y = 0 \rightarrow z = 0\]
Example: Full Adder Circuit (2/2)

• To show that \( i_1 = 1, i_2 = 1, o_1 = 0 \) follows from the assumption that \( i_3 = 0, o_2 = 1 \)

\[
\begin{align*}
&\langle i_1 \oplus i_2 = x_1, i_1 \land i_2 = y_1, x_1 \oplus i_1 = o_1, i_1 \land x_1 = y_2, y_1 \lor y_2 = o_2, i_3 = 0, o_2 = 1, y_2 = 0 \rangle \\
\text{AND 4} \\
&\langle i_1 \oplus i_2 = x_1, i_1 \land i_2 = y_1, x_1 \oplus i_1 = o_1, i_1 \land x_1 = y_2, y_1 \lor y_2 = o_2, i_3 = 0, o_2 = 1, y_2 = 0, y_1 = 1 \rangle \\
\text{OR 4} \\
&\langle i_1 \oplus i_2 = x_1, i_1 \land i_2 = y_1, x_1 \oplus i_1 = o_1, i_1 \land x_1 = y_2, y_1 \lor y_2 = o_2, i_3 = 0, o_2 = 1, y_2 = 0, y_1 = 1 \rangle \\
\text{AND 6} \\
&\langle i_1 \oplus i_2 = x_1, i_1 \land i_2 = y_1, x_1 \oplus i_1 = o_1, i_1 \land x_1 = y_2, y_1 \lor y_2 = o_2, i_3 = 0, o_2 = 1, y_2 = 0, y_1 = 1, i_1 = 1, i_2 = 1 \rangle \\
\text{XOR 1} \\
&\langle x_1 \oplus i_1 = o_1, i_3 = 0, o_2 = 1, y_2 = 0, y_1 = 1, i_1 = 1, i_2 = 1, x_1 = 0 \rangle \\
\text{XOR 2} \\
&\langle i_3 = 0, o_2 = 1, y_2 = 0, y_1 = 1, i_1 = 1, i_2 = 1, x_1 = 0, o_1 = 0 \rangle
\end{align*}
\]

Linear Constraints on Integer Intervals

• **Linear expression**: a term in the alphabet that contains:
  - Two constants, 0 and 1
  - The unary function symbol ‘-’
  - Binary function symbols ‘+’ and ‘-’
  - Abbreviate: and

\[
\begin{align*}
\text{\underline{Linear constraint}}: \quad s \ op \ t
\end{align*}
\]

- For example: \( 3x + 4y - 5z \leq 7 \)

- **Integer interval**: \([a..b]\)

Domain Reduction Rules for Inequality Constraints: Example

- \( 3x + 4y - 5z \leq 7 \) with \( x \in [l_x..h_x], y \in [l_y..h_y], z \in [l_z..h_z] \)

- Rewrite as: \( x \leq \frac{7 - 4y + 5z}{3} \)

any value of \( x \) that satisfies it also satisfies:

\[
x \leq \left[ \frac{7 - 4l_y + 5h_z}{3} \right]
\]

- So \([l_x..h_x]\) can be reduced to:

\[
[l_x..\min\left(\left[\frac{7 - 4l_y + 5h_z}{3}\right], h_x\right)]
\]

Domain Reduction Rules for Inequality Constraints

**LINEAR INEQUALITY 1**

\[
\begin{align*}
\langle \text{\Sigma}_{i \in \text{POS}} a_i x_i - \text{\Sigma}_{i \in \text{NEG}} a_i x_i \leq b; x_i \in [l_i..h_i], ..., x_n \in [l_n..h_n] \rangle \\
\langle \text{\Sigma}_{i \in \text{POS}} a_i x_i - \text{\Sigma}_{i \in \text{NEG}} a_i x_i \leq b; x_i \in [l'_i..h'_i], ..., x_n \in [l'_n..h'_n] \rangle
\end{align*}
\]

where for \( j \in \text{POS} \quad l_j' := l_j, h_j' := \min(h_j, \lfloor \alpha_j \rfloor) \)

\[
\alpha_j := \frac{b - \text{\Sigma}_{i \in \text{POS} \setminus \{j\}} a_i l_i + \text{\Sigma}_{i \in \text{NEG}} a_i h_i}{a_j}
\]

and for \( j \in \text{NEG} \quad l_j' := \max(l_j, \lceil \beta_j \rceil), h_j' := h_j \)

\[
\beta_j := \frac{-b + \text{\Sigma}_{i \in \text{POS}} a_i l_i - \text{\Sigma}_{i \in \text{NEG} \setminus \{j\}} a_i h_i}{a_j}
\]
Objectives

- Introduce some incomplete constraint solvers for
  - Equality and disequality constraints
  - Boolean constraints
  - Linear constraints on integer intervals
    - Domain reduction rules for inequality constraints