Diploma Thesis: Efficient data structures for finite set and multiset constraint variables

Patrick Pekczynski

Supervisor: Guido Tack
Responsible Professor: Prof. Gert Smolka
Timeframe: February 2006 - January 2007

Programming Systems Lab
Department of Computer Science
Saarland University, Saarbrücken

23.03.2006

Mainstream

Constraint Programming

- largely restricted to finite domain variables (FDVar)

Example

- variables:
  - $x \in [1..6]$
  - $y \in \{1, 3, 6, 8, 12\}$
  - $z \in \{10\}$
- constraints as relations between variables
  - $x + y = z$
  - $x \neq y$

Beyond finite domains

When do we use sets?

- constraints are domain specific
- interested in collection of elements
- symmetries among elements have to be avoided
  - students in tutorial groups
  - players in a team
  - workers at a shift
- use finite set variables (FSVar)

Beyond finite domains

Example

- variables:
  - $g \in \{\{1,3\},\{2,7,12\},\{11,\ldots,14\}\}$
  - $h \in \{\{1,3,5,6\},\{7,9,13\},\{1,\ldots,20\}\}$
  - $u \in \{\emptyset,\ldots,\{1,\ldots,20\}\}$
- constraints:
  - $g \subset h$
  - $|h| = 4$
  - $u = g \cup h$

Complete but naive

Naive representation

- keep track of every possible value s can take

Problem

- $D = \mathcal{P}(\{1, \ldots, 400\})$, $|D| = 2^{400}$
- exponential size
- representation impracticable

Predominant and approximate

Bounds representation[Ger95]

- bounded lattice $\langle \mathcal{P}(\mathcal{U}), \subseteq \rangle$, $\forall a \in \mathcal{P}(\mathcal{U}) : \emptyset \subseteq a \subseteq \mathcal{U}$
- FSVar $s \in D \subset \mathcal{P}(\mathcal{U})$
- approximate domain D by convex hull

$$conv(D) = [\inf(D)..\sup(D)] = [\lfloor D \rfloor..\lceil D \rceil] = \left[\bigcap_{a \in D} a..\bigcup_{b \in D} b\right]$$

- properties:

P1 extension $D \subseteq conv(D)$

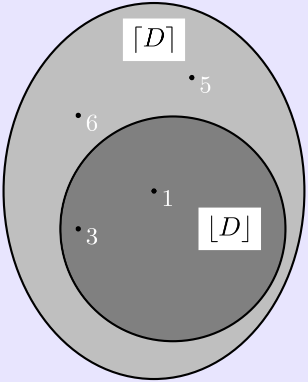
- let $d \in D$ be the set finally assigned to s
- $a \in \lfloor D \rfloor \Leftrightarrow a \in d$
- $a \notin \lceil D \rceil \Leftrightarrow a \notin d$

P2 idempotency $conv(D) \subseteq conv(conv(D))$

P3 monotonicity $D \subseteq E \Rightarrow conv(D) \subseteq conv(E)$

Graphical view

Venn diagram



$$
\begin{aligned}
D &= [\{1,3\}..\{1,3,5,6\}] \\
&= \{\{1,3\},\{1,3,5\}, \\
&\quad \{1,3,6\},\{1,3,5,6\}\} \\
\lfloor D \rfloor &= \{1,3\} \\
\lceil D \rceil &= \{1,3,5,6\}
\end{aligned}
$$

Predominant and approximate

Modeling set constraints

- $s_1 \subseteq s_2 \Rightarrow s_1 \subseteq \lceil s_2 \rceil \wedge \lfloor s_1 \rfloor \subseteq s_2$
- $s_1 \cup s_2 = s_3 \Rightarrow \lfloor s_1 \rfloor \cup \lfloor s_2 \rfloor \subseteq s_3 \subseteq \lceil s_1 \rceil \cup \lceil s_2 \rceil$
- $s_1 \cap s_2 = s_3 \Rightarrow \lfloor s_1 \rfloor \cap \lfloor s_2 \rfloor \subseteq s_3 \subseteq \lceil s_1 \rceil \cap \lceil s_2 \rceil$

Predominant and approximate

Conclusion

- store only two sets instead of exponentially many
- state-of-the-art implementation in most constraint solvers (Gecode[The06a], Mozart[The06b],ILOG[ILO00], Choco[Lab00], ECLiPSe[WNS97])

Drawback

- $\lfloor D \rfloor$ is represented twice, since $\lfloor D \rfloor \subseteq \lceil D \rceil$

Complete domain representation

ROBDD representation[HLS05]

- finite set D represented by its canonical function

$$\chi_D : \mathbb{Z} \mapsto \mathbb{B} : \chi_D(i) = \begin{cases} 1 & \text{if } i \in D \\ 0 & \text{otherwise} \end{cases}$$

- analogy set domains and boolean functions
- ROBDD canonical function representation up to reordering
- domain D for FSVar $s \in D$ as single ROBDD $D(s)$

Graphical view

From domain to ROBDD

- FSVar $s \in D = \{\{1,3\},\{1,3,5\},\{1,3,6\},\{1,3,5,6\}\} \subseteq \mathcal{P}(\{1,3,5,6\})$
- associate boolean variables $\{s_1, s_3, s_5, s_6\}$ with s

Graphical view

From domain to ROBDD

- FSVar $s \in D = \{\{1,3\},\{1,3,5\},\{1,3,6\},\{1,3,5,6\}\} \subseteq \mathcal{P}(\{1,3,5,6\})$
- associate boolean variables $\{s_1, s_3, s_5, s_6\}$ with s

$$
\begin{array}{rcl}
f & = & (s_1 \wedge s_3 \wedge \neg s_5 \wedge \neg s_6) \\
 & \vee & (s_1 \wedge s_3 \wedge s_5 \wedge \neg s_6) \\
 & \vee & (s_1 \wedge s_3 \wedge \neg s_5 \wedge s_6) \\
 & \vee & (s_1 \wedge s_3 \wedge s_5 \wedge s_6)
\end{array}
$$

Graphical view

From domain to ROBDD

- FSVar $s \in D = \{\{1,3\},\{1,3,5\},\{1,3,6\},\{1,3,5,6\}\} \subseteq \mathcal{P}(\{1,3,5,6\})$
- associate boolean variables $\{s_1, s_3, s_5, s_6\}$ with s

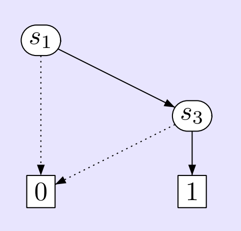
$$
\begin{aligned}
f \quad = \quad & (s_1 \land s_3 \land \neg s_5 \land \neg s_6) \\
\lor \quad & (s_1 \land s_3 \land s_5 \land \neg s_6) \\
\lor \quad & (s_1 \land s_3 \land \neg s_5 \land s_6) \\
\lor \quad & (s_1 \land s_3 \land s_5 \land s_6)
\end{aligned}
\quad \Rightarrow
$$

Graphical view

From domain to ROBDD

- FSVar $s \in D = \{\{1,3\},\{1,3,5\},\{1,3,6\},\{1,3,5,6\}\} \subseteq \mathcal{P}(\{1,3,5,6\})$
- associate boolean variables $\{s_1, s_3, s_5, s_6\}$ with s

$$
\begin{array}{lll}
f & = & (s_1 \wedge s_3 \wedge \neg s_5 \wedge \neg s_6) \\
 & \vee & (s_1 \wedge s_3 \wedge s_5 \wedge \neg s_6) \\
 & \vee & (s_1 \wedge s_3 \wedge \neg s_5 \wedge s_6) \\
 & \vee & (s_1 \wedge s_3 \wedge s_5 \wedge s_6)
\end{array}
\Rightarrow
$$



Complete domain representation

Modeling set constraints ($v \subseteq w$)

- $v \in E, w \in F, E, F \subset \mathcal{P}(\{1, 2, 3\})$
- create boolean variables $\{v_1, v_2, v_3\}, \{w_1, w_2, w_3\}$

Complete domain representation

Modeling set constraints ($v \subseteq w$)

- $v \in E, w \in F, E, F \subset \mathcal{P}(\{1, 2, 3\})$
- create boolean variables $\{v_1, v_2, v_3\}$, $\{w_1, w_2, w_3\}$

$$
\begin{aligned}
f \quad = &\quad (v_1 \Rightarrow w_1) \\
\wedge &\quad (v_2 \Rightarrow w_2) \\
\wedge &\quad (v_3 \Rightarrow w_3)
\end{aligned}
$$

Complete domain representation

Modeling set constraints ($v \subseteq w$)

- $v \in E, w \in F, E, F \subset \mathcal{P}(\{1,2,3\})$
- create boolean variables $\{v_1, v_2, v_3\}, \{w_1, w_2, w_3\}$

$$
\begin{aligned}
f \quad = \quad & (v_1 \Rightarrow w_1) \\
\wedge \quad & (v_2 \Rightarrow w_2) \\
\wedge \quad & (v_3 \Rightarrow w_3)
\end{aligned}
$$

$$\Rightarrow$$

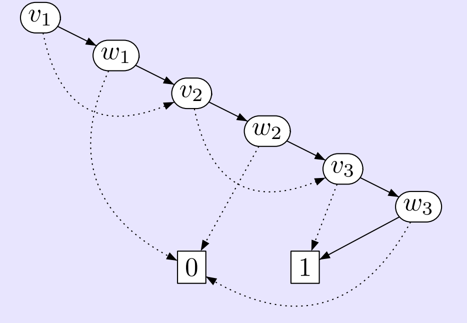
Complete domain representation

Modeling set constraints ($v \subseteq w$)

- $v \in E, w \in F, E, F \subset \mathcal{P}(\{1,2,3\})$
- create boolean variables $\{v_1, v_2, v_3\}, \{w_1, w_2, w_3\}$

$$f = (v_1 \Rightarrow w_1) \wedge (v_2 \Rightarrow w_2) \wedge (v_3 \Rightarrow w_3)$$

$\Rightarrow$



Complete domain representation

Conclusion

- data structure with efficient operations[HLS05]:
  - $R_1 \circ R_2 \in \mathcal{O}(|R_1| \cdot |R_2|), \circ \in \{\vee, \wedge, \Leftrightarrow\}$
  - test for identical ROBDDs in $\mathcal{O}(1)$
- complete representation

Drawback

- still exponential size possible

Overview

Diploma thesis

- empirical analysis of representations
- implementation
  - efficient implementation of bounds representation
  - using efficient BDD libraries to integrate complete ROBDD representation
- evaluation of the implemented data structures
- generalization to finite multiset variables

Outlook

- propagators working on both representations
- can choose between bounds and domain representation
- posted through general description language

Framework

Gecode Constraint Library

- **ge**neric
- **co**nstraint
- **d**evelopment
- **e**nvironment

Gecode[The06a], a C++ library for constraint programming.
Version 1.0.1 available from http://www.gecode.org

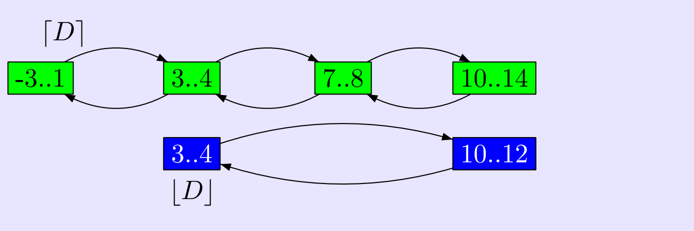
Developers

- **Dr. Christian Schulte** (head, KTH, Sweden)
- **Guido Tack** (PS Lab, Saarbrücken, Germany)

Sets in Gecode

Representation of finite integer sets

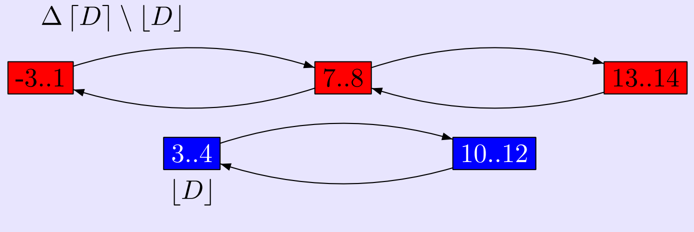
- bounds representation of domain D by $[\lfloor D \rfloor .. \lceil D \rceil]$
- each bound represented by a range list
- $D = [\{3, 4, 10, 11, 12\}..\{-3, -2, -1, 0, 1, 3, 4, 7, 8, 10, 11, 12, 13, 14\}]$



Sets in Gecode

Representation of finite integer sets

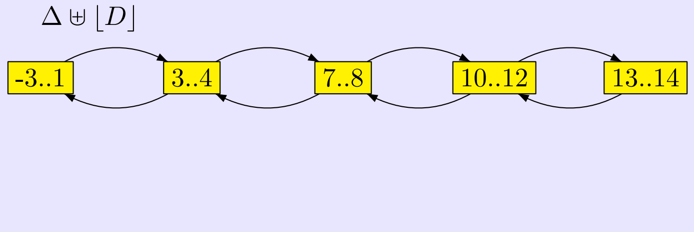
- remove $\lfloor D \rfloor$ from $\lceil D \rceil$
- obtain $\Delta = \lceil D \rceil \setminus \lfloor D \rfloor$



Sets in Gecode

Minimal bounds representation [AB00]

- minimal bounds rep
- store disjoint union of Δ and $\lfloor D \rfloor$ such that $\Delta \uplus \lfloor D \rfloor = \lceil D \rceil$



References I

[AB00] Francisco Azevedo and Pedro Barahona.
Applications of an extended set constraint solver.
2000.

[Ger95] Carmen Gervet.
Set Intervals in Constraint Logic Programming.
PhD thesis, L'Université de Franche-Comté, 1995.

[HLS05] P.J. Hawkins, V. Lagoon, and P.J. Stuckey.
Solving set constraint satisfaction problems using ROBDDs.
J. Artif. Intell. Res. (JAIR), 24:109–156, 2005.

[ILO00] ILOG Inc., Mountain View, CA, USA.
ILOG Solver 5.0 reference Manual, 2000.

References II

[Lab00] F. Laburthe.
Choco: Implementing a CP kernel.
In *TRICS*, pages 71–85, September 2000.

[The06a] The Gecode team.
Generic constraint development environment.
Available from `http://www.gecode.org`, 2006.

[The06b] The Mozart Consortium.
The Mozart programming system.
`http://www.mozart-oz.org`, 2006.

[WNS97] M. Wallace, S. Novello, and J. Schimpf.
Eclipse: A platform for constraint logic programming.
Technical report, IC Parc, Imperial College, London, 1997.

Orders

Total Order

Partial Order

- tuple $\langle X, \prec \rangle$ such that $\prec$ is:
  1. reflexive $\forall a \in X : a \prec a$
  2. antisymmetric
     $\forall a, b \in X : a \prec b \wedge b \prec a \Rightarrow a = b$
  3. transitive
     $\forall a, b, c \in X : a \prec b \wedge b \prec c \Rightarrow a \prec c$

Additional axiom

comparability $\forall a, b \in X : a \prec b \vee b \prec a$.