Constraint Programming with CHR

Thom Frühwirth

Faculty of Computer Science
University of Ulm, Germany

Uni Dortmund
November 2005

Images are subject to copyright of the respective owners

Citations may be incomplete for space reasons
Part I

Constraint Programming

1. Constraint Reasoning
2. Constraint Programming
**The Holy Grail**

*Constraint Programming* represents one of the closest approaches computer science has yet made to the *Holy Grail* of programming: the user states the problem, the computer solves it.

The Idea

- **Combination Lock Example**
  
  0 1 2 3 4 5 6 7 8 9
  
  Greater or equal 5.
  
  Prime number.

- **Declarative problem** representation by variables and constraints:
  
  \[ x \in \{0, 1, \ldots, 9\} \land x \geq 5 \land \text{prime}(x) \]

- **Constraint propagation and simplification** reduce search space:
  
  \[ x \in \{0, 1, \ldots, 9\} \land x \geq 5 \rightarrow x \in \{5, 6, 7, 8, 9\} \]
The Idea

- **Combination Lock Example**

  0 1 2 3 4 5 6 7 8 9

  Greater or equal 5.

  Prime number.

- **Declarative problem** representation by variables and constraints:

  \[ x \in \{0, 1, \ldots, 9\} \land x \geq 5 \land \text{prime}(x) \]

- **Constraint propagation and simplification** reduce search space:

  \[ x \in \{0, 1, \ldots, 9\} \land x \geq 5 \rightarrow x \in \{5, 6, 7, 8, 9\} \]
The Idea

- **Combination Lock Example**

  0 1 2 3 4 5 6 7 8 9
  Greater or equal 5.
  Prime number.

- **Declarative problem** representation by
  variables and constraints:

  \[ x \in \{0, 1, \ldots, 9\} \land x \geq 5 \land \text{prime}(x) \]

- **Constraint propagation and simplification**
  reduce search space:

  \[ x \in \{0, 1, \ldots, 9\} \land x \geq 5 \to x \in \{5, 6, 7, 8, 9\} \]
Constraint Reasoning Everywhere

Combination
Simplification
Contradiction
Redundancy
Sudoku

gendomain(Matrix,1..9),
Matrix=[A1,A2,...,A9,B1,...,I9],
alldiff(A1,A2,A3,A4,A5,A6,A7,A8,A9),
..., 
alldiff(A1,B1,C1,D1,E1,F1,G1,H1,I1),
..., 
alldiff(A1,B1,C1,A2,B2,C2,A3,B3,C3),
..., 
labeling(Matrix).

sudoku(Matrix) <=>
c wikipedia.com
Sudoku

\[
\text{sudoku}(\text{Matrix}) \iff \\
g\text{domain}(\text{Matrix}, 1..9), \\
\text{Matrix} = [A1, A2, \ldots, A9, B1, \ldots, I9], \\
\text{all}\text{diff}(A1, A2, A3, A4, A5, A6, A7, A8, A9), \\
\ldots, \\
\text{all}\text{diff}(A1, B1, C1, D1, E1, F1, G1, H1, I1), \\
\ldots, \\
\text{all}\text{diff}(A1, B1, C1, A2, B2, C2, A3, B3, C3), \\
\ldots, \\
\text{labeling}(\text{Matrix}).
\]
Mortgage

D: Amount of Loan, Debt, Principal
T: Duration of loan in months
I: Interest rate per month
R: Rate of payments per month
S: Balance of debt after T months

\[ \text{mortgage}(D, T, I, R, S) \Leftrightarrow \]
\[ \begin{align*}
T & = 0, \\
D & = S \\
\text{;} & \\
T & > 0, \\
T1 & = T - 1, \\
D1 & = D + D \times I - R, \\
\text{mortgage}(D1, T1, I, R, S). \\
\end{align*} \]
Mortgage II

\[
\text{mortgage}(D, T, I, R, S) \iff \\
\begin{align*}
T &= 0, \quad D = S \\
; \\
T &> 0, \quad T1 = T - 1, \quad D1 = D + D*I - R, \\
\text{mortgage}(D1, T1, I, R, S).
\end{align*}
\]

- \(\text{mortgage}(100000, 360, 0.01, 1025, S)\) yields \(S=12625.90\).
- \(\text{mortgage}(D, 360, 0.01, 1025, 0)\) yields \(D=99648.79\).
- \(\text{mortgage}(100000, T, 0.01, 1025, S), \ S<0\) yields \(T=374, \ S=-807.96\).
- \(\text{mortgage}(D, 360, 0.01, R, 0)\) yields \(R=0.0102861198*D\).
Advantages of Constraint Logic Programming

**Theoretical**
Logical Foundation – First-Order Logic

**Conceptual**
Sound Modeling

**Practical**
Efficient Algorithms/Implementations
Combination of different Solvers
Early Commercial Applications (in the 90s)

- **Lufthansa**: Short-term staff planning.
- **Hongkong Container Harbor**: Resource planning.
- **Renault**: Short-term production planning.
- **Nokia**: Software configuration for mobile phones.
- **Airbus**: Cabin layout.
- **Siemens**: Circuit verification.
- **Caisse d’epargne**: Portfolio management.

In *Decision Support Systems for Planning and Configuration*, for *Design and Analysis*.
Part II

CHR Constraint Handling Rules

3 Constraint Handling Rules (CHR)

4 Constraint Solvers

5 Program Analysis
Example Partial Order Constraint

\[ X \leq X \iff \text{true} \quad \text{(reflexivity)} \]
\[ X \leq Y \land Y \leq X \iff X = Y \quad \text{(antisymmetry)} \]
\[ X \leq Y \land Y \leq Z \Rightarrow X \leq Z \quad \text{(transitivity)} \]

\[ A \leq B \land B \leq C \land C \leq A \]
\[ \quad \downarrow \]
\[ A \leq B \land B \leq C \land C \leq A \land A \leq C \quad \text{(antisymmetry)} \]
\[ \quad \downarrow \]
\[ A \leq B \land B \leq C \land A = C \quad \text{(built-in solver)} \]
\[ \quad \downarrow \]
\[ A \leq B \land B \leq A \land A = C \quad \text{(antisymmetry)} \]
\[ \quad \downarrow \]
\[ A = B \land A = C \]
Example Partial Order Constraint

\[
X \leq X \iff \text{true} \quad \text{(reflexivity)}
\]
\[
X \leq Y \land Y \leq X \iff X = Y \quad \text{(antisymmetry)}
\]
\[
X \leq Y \land Y \leq Z \implies X \leq Z \quad \text{(transitivity)}
\]

\[
\overbrace{A \leq B} \land \overbrace{B \leq C} \land \overbrace{C \leq A}
\]
\[
A \leq B \land B \leq C \land C \leq A \land A \leq C \quad \text{(transitivity)}
\]
\[
A \leq B \land B \leq C \land C \leq A \land A \leq C \land A = C \quad \text{(antisymmetry)}
\]
\[
A \leq B \land B \leq A \land A = C \quad \text{(built-in solver)}
\]
\[
A = B \land A = C \quad \text{(antisymmetry)}
\]
Example Partial Order Constraint

\[ X \leq X \iff \text{true} \quad \text{(reflexivity)} \]
\[ X \leq Y \land Y \leq X \iff X = Y \quad \text{(antisymmetry)} \]
\[ X \leq Y \land Y \leq Z \implies X \leq Z \quad \text{(transitivity)} \]

\[
\frac{A \leq B \land B \leq C \land C \leq A}{A \leq B \land B \leq C \land C \leq A \land A \leq C} \quad \text{(transitivity)}
\]

\[
\frac{A \leq B \land B \leq C \land C \leq A \land A \leq C}{A \leq B \land B \leq A \land A = C} \quad \text{(antisymmetry)}
\]

\[
\frac{A \leq B \land B \leq A \land A = C}{A = B \land A = C} \quad \text{(antisymmetry)}
\]

\[
\text{Thom Frühwirth} \quad \text{Constraint Programming with CHR}
\]
Example Partial Order Constraint

\[X \leq X \iff \text{true} \quad \text{(reflexivity)}\]
\[X \leq Y \land Y \leq X \iff X = Y \quad \text{(antisymmetry)}\]
\[X \leq Y \land Y \leq Z \implies X \leq Z \quad \text{(transitivity)}\]

\[
\begin{align*}
A \leq B & \land B \leq C \land C \leq A \\
\Downarrow & \\
A \leq B & \land B \leq C \land C \leq A \land A \leq C \\
\Downarrow & \\
A \leq B & \land B \leq C \land A = C \\
\Downarrow & \\
A \leq B & \land B \leq A \land A = C \\
\Downarrow & \\
A = B & \land A = C
\end{align*}
\]

(antisymmetry)

(antisymmetry)

(built-in solver)
Example Partial Order Constraint

\[ X \leq X \ \Leftrightarrow \ true \quad \text{(reflexivity)} \]
\[ X \leq Y \land Y \leq X \ \Leftrightarrow \ X = Y \quad \text{(antisymmetry)} \]
\[ X \leq Y \land Y \leq Z \ \Rightarrow \ X \leq Z \quad \text{(transitivity)} \]

\[
\begin{align*}
A \leq B \land B \leq C \land C \leq A &
\quad \downarrow \quad \text{(transitivity)} \\
A \leq B \land B \leq C \land C \leq A \land A \leq C &
\quad \downarrow \quad \text{(antisymmetry)} \\
A \leq B \land B \leq C \land A = C &
\quad \downarrow \quad \text{(built-in solver)} \\
A \leq B \land B \leq A \land A = C & \quad \downarrow \quad \text{(antisymmetry)} \\
A = B \land A = C &
\end{align*}
\]
Example Partial Order Constraint

\[ X \leq X \iff true \quad \text{(reflexivity)} \]
\[ X \leq Y \land Y \leq X \iff X = Y \quad \text{(antisymmetry)} \]
\[ X \leq Y \land Y \leq Z \Rightarrow X \leq Z \quad \text{(transitivity)} \]

\[
\begin{align*}
A \leq B & \land B \leq C & \land C \leq A \\
\downarrow & & \\
A \leq B & \land B \leq C & \land C \leq A & \land A \leq C \\
\downarrow & & \\
A \leq B & \land B \leq C & A = C \\
\downarrow & & \\
A \leq B & \land B \leq A & A = C \\
\downarrow & & \\
A = B & \land A = C
\end{align*}
\]

(antisymmetry)  
(built-in solver)  
(antisymmetry)
Operational Semantics

Apply rules until exhaustion in any order (fixpoint computation).

**Simplify**

If \((H \Leftrightarrow C \mid B)\) rule with renamed fresh variables \(\bar{x}\)
and \(CT \models G_{\text{builtin}} \rightarrow \exists \bar{x}(H=H' \land C)\)
then \(H' \land G \leftrightarrow G \land H=H' \land B\)

**Propagate**

If \((H \Rightarrow C \mid B)\) rule with renamed fresh variables \(\bar{x}\)
and \(CT \models G_{\text{builtin}} \rightarrow \exists \bar{x}(H=H' \land C)\)
then \(H' \land G \leftrightarrow H' \land G \land H=H' \land B\)
Operational Semantics

Apply rules until exhaustion in any order (fixpoint computation).

**Simplify**

If \((H \iff C \mid B)\) rule with renamed fresh variables \(\bar{x}\)
and \(CT \models G_{builtin} \rightarrow \exists \bar{x}(H=H' \land C)\)
then \(H' \land G \iff G \land H=H' \land B\)

**Propagate**

If \((H \Rightarrow C \mid B)\) rule with renamed fresh variables \(\bar{x}\)
and \(CT \models G_{builtin} \rightarrow \exists \bar{x}(H=H' \land C)\)
then \(H' \land G \iff H' \land G \land H=H' \land B\)
Operational Semantics

Apply rules until exhaustion in any order (fixpoint computation).

**Simplify**

If \((H \iff C | B)\) rule with renamed fresh variables \(\bar{x}\)
and \(CT \models G_{builtin} \rightarrow \exists \bar{x}(H=H' \land C)\)
then \(H' \land G \iff G \land H=H' \land B\)

**Propagate**

If \((H \Rightarrow C | B)\) rule with renamed fresh variables \(\bar{x}\)
and \(CT \models G_{builtin} \rightarrow \exists \bar{x}(H=H' \land C)\)
then \(H' \land G \iff H' \land G \land H=H' \land B\)
Anytime Algorithm

Computation can be interrupted and restarted at any time. Intermediate results approximate final result.

\[ A \leq B \land B \leq C \land C \leq A \]
\[ A \leq B \land B \leq C \land C \leq A \land A \leq C \] (transitivity)
\[ A \leq B \land B \leq C \land A = C \] (antisymmetry)
\[ A \leq B \land B \leq A \land A = C \] (built-in solver)
\[ A = B \land A = C \] (antisymmetry)
Anytime Algorithm

Computation can be interrupted and restarted at any time. Intermediate results approximate final result.

\[
\begin{align*}
A \leq B & \land B \leq C & \land C \leq A \\
A \leq B & \land B \leq C & \land C \leq A & \land A \leq C \\
A \leq B & \land B \leq C & \land A = C \\
A \leq B & \land B \leq A & \land A = C \\
A = B & \land A = C
\end{align*}
\]

(transitivity) \hspace{1cm} (antisymmetry) \hspace{1cm} (built-in solver) \hspace{1cm} (antisymmetry)
Anytime Algorithm

Computation can be interrupted and restarted at any time. Intermediate results approximate final result.

\[
\begin{align*}
A \leq B \land B \leq C \land C \leq A \\
\quad \quad \downarrow \\
A \leq B \land B \leq C \land C \leq A \land A \leq C \\
\quad \quad \downarrow \\
A \leq B \land B \leq C \land A = C \\
\quad \quad \downarrow \\
A \leq B \land B \leq A \land A = C \\
\quad \quad \downarrow \\
A = B \land A = C
\end{align*}
\]

(transitivity)

(antisymmetry)

(built-in solver)

(antisymmetry)
Online Algorithm

The complete input is initially unknown. The input data arrives incrementally during computation. No recomputation from scratch necessary.

**Monotonicity and Incrementality**

If \( G \implies G' \)

then \( G \land C \implies G' \land C \)

\[
\begin{align*}
A \leq B \land B \leq C \land C \leq A & \quad \text{(transitivity)} \\
A \leq B \land B \leq C \land A \leq C \land C \leq A & \quad \text{(antisymmetry)}
\end{align*}
\]

\[
A \leq B \land B \leq C \land A = C
\]

\[
\vdots
\]
Online Algorithm

The complete input is initially unknown. The input data arrives incrementally during computation. No recomputation from scratch necessary.

**Monotonicity and Incrementality**

If \( G \mapsto G' \)
then \( G \land C \mapsto G' \land C \)

\[
\begin{align*}
\text{transitivity:} & \quad A \leq B \land B \leq C \land C \leq A \\
\text{antisymmetry:} & \quad A \leq B \land B \leq C \land A = C
\end{align*}
\]
Online Algorithm

The complete input is initially unknown. The input data arrives incrementally during computation. No recomputation from scratch necessary.

Monotonicity and Incrementality
If \( G \mapsto G' \)
then \( G \land C \mapsto G' \land C \)

\[
A \leq B \land B \leq C \land C \leq A \quad \text{(transitivity)}
\]

\[
A \leq B \land B \leq C \land A \leq C \land C \leq A \quad \text{(antisymmetry)}
\]

\[
A \leq B \land B \leq C \land A = C
\]

\[
\ldots
\]
Online Algorithm

The complete input is initially unknown.
The input data arrives incrementally during computation.
No recomputation from scratch necessary.

Monotonicity and Incrementality

If \( G \longrightarrow G' \)
then \( G \land C \longrightarrow G' \land C \)

\[
A \leq B \land B \leq C \land C \leq A \\
\quad \downarrow \\
A \leq B \land B \leq C \land A \leq C \land C \leq A \\
\quad \downarrow \\
A \leq B \land B \leq C \land A = C \\
\quad \downarrow \\
\ldots
\]

(transitivity)

(antisymmetry)
Concurrency

Rules can be applied in parallel to different parts of the problem.

If \( A \rightarrow B \) and \( C \rightarrow D \) then \( A \land C \rightarrow B \land D \)

\[
\begin{align*}
A \leq B & \land B \leq C \\
\downarrow \\
A \leq B & \land B \leq C \land A \leq C \\
\downarrow \\
\ldots & A = C \ldots
\end{align*}
\]

\[
\begin{align*}
& \land \\
C \leq D & \land D \leq A \\
\downarrow \\
& \land \\
C \leq D & \land D \leq A \land C \leq A \\
\downarrow \\
\ldots & A = C \ldots
\end{align*}
\]
Concurrency

Rules can be applied in parallel to different parts of the problem.

If \( A \rightarrow B \) and \( C \rightarrow D \) then \( A \land C \rightarrow B \land D \)

\[
\begin{align*}
A \leq B & \land B \leq C \\
\Rightarrow & \\
A \leq B & \land B \leq C \land A \leq C \\
\land & \\
C \leq D & \land D \leq A \\
\Rightarrow & \\
C \leq D & \land D \leq A \land C \leq A \\
\land & \\
\cdots & \land A = C \cdots
\end{align*}
\]
Concurrency

Rules can be applied in parallel to different parts of the problem.

If \( A \rightarrow B \)
and \( C \rightarrow D \)
then \( A \land C \rightarrow B \land D \)

\[
\begin{align*}
A \leq B \land B \leq C & \quad \land \quad C \leq D \land D \leq A \\
\downarrow & \\
A \leq B \land B \leq C \land A \leq C & \quad \land \quad C \leq D \land D \leq A \land C \leq A \\
\downarrow & \\
\ldots A = C \ldots
\end{align*}
\]
Concurrency

Rules can be applied in parallel to different parts of the problem.

If \( A \rightarrow B \) and \( C \rightarrow D \) then \( A \land C \rightarrow B \land D \)

\[
\begin{align*}
A &\leq B \land B \leq C \land A \leq C \\
&\downarrow \\
A \leq B \land B \leq C \land A \leq C \\
&\downarrow \\
&\vdots \\
A &\leq C \quad \vdots
\end{align*}
\]
Equations of the form $a_1x_1 + \ldots + a_nx_n + b = 0$.

**Solved form:** leftmost variable occurs only once.

Reach solved normal form by variable elimination.

$$A_1\times + P_1 = 0 \land XP = 0 \iff$$

$$\text{find}(A_2\times, XP, P_2) \land$$
$$\text{compute}(P_2 - (P_1/A_1)\times A_2, P_3) \land$$
$$A_1\times + P_1 = 0 \land P_3 = 0.$$

$$B = 0 \iff \text{number}(B) \land \text{zero}(B).$$

$$1\times + 3\times Y + 5 = 0 \land 3\times + 2\times Y + 8 = 0$$
$$\text{compute}((2\times Y + 8) - ((3\times Y + 5)/1)\times 3, P_3) \% P_3 = -7\times Y + -7$$
$$1\times + 3\times Y + 5 = 0 \land -7\times Y + -7 = 0 \% Y = -1$$
$$\text{compute}((1\times Y + 5) - ((-7)/-7)\times 3, P_3') \% P_3' = 1\times + 2$$
$$1\times + 2 = 0 \land -7\times Y + -7 = 0 \% X = -2$$
Linear Polynomial Equations

Equations of the form $a_1x_1 + \ldots + a_nx_n + b = 0$.

Solved form: leftmost variable occurs only once.
Reach solved normal form by variable elimination.

\[
\begin{align*}
A1*X+P1=0 \land XP=0 & \iff \\
& \text{find}(A2*X,XP,P2) \land \\
& \text{compute}(P2-(P1/A1)*A2,P3) \land \\
& A1*X+P1=0 \land P3=0.
\end{align*}
\]

\[B=0 \iff \text{number}(B) \land \text{zero}(B).\]

\[
\begin{align*}
1*X+3*Y+5=0 & \land 3*X+2*Y+8=0 \\
& \text{compute}\left((2*Y+8) - ((3*Y+5)/1)*3,P3\right) \% P3=-7*Y+ -7 \\
1*X+3*Y+5=0 & \land -7*Y+ -7=0 \% Y=-1 \\
& \text{compute}\left((1*X+5) - ((-7)/-7)*3,P3’\right) \% P3’=1*X+2 \\
1*X+2=0 & \land -7*Y+ -7=0 \% X=-2
\end{align*}
\]
Linear Polynomial Equations

Equations of the form $a_1x_1 + \ldots + a_nx_n + b = 0$.

Solved form: leftmost variable occurs only once.
Reach solved normal form by variable elimination.

$A1\cdot X + P1 = 0 \land XP = 0 \iff$

$\textit{find}(A2\cdot X,XP,P2) \mid$
$\textit{compute}(P2 - (P1/A1)\cdot A2,P3) \land$
$A1\cdot X + P1 = 0 \land P3 = 0.$

$B = 0 \iff \textit{number}(B) \mid \textit{zero}(B).$

$1\cdot X + 3\cdot Y + 5 = 0 \land 3\cdot X + 2\cdot Y + 8 = 0$
$\textit{compute}((2\cdot Y + 8) - ((3\cdot Y + 5)/1)*3,P3) \ % \ P3 = -7\cdot Y + -7$
$1\cdot X + 3\cdot Y + 5 = 0 \land -7\cdot Y + -7 = 0 \ % \ Y = -1$
$\textit{compute}((1\cdot X + 5) - ((-7)/-7)*3,P3') \ % \ P3' = 1\cdot X + 2$
$1\cdot X + 2 = 0 \land -7\cdot Y + -7 = 0 \ % \ X = -2$
Linear Polynomial Equations

Equations of the form $a_1x_1 + \ldots + a_nx_n + b = 0$.

Solved form: leftmost variable occurs only once.
Reach solved normal form by variable elimination.

\[
\begin{align*}
A1\cdot x + P1 &= 0 \land XP = 0 \iff \\
\text{find}(A2\cdot x, XP, P2) \land \\
\text{compute}(P2 - (P1/A1)\cdot A2, P3) \land \\
A1\cdot x + P1 &= 0 \land P3 = 0.
\end{align*}
\]

\[
B = 0 \iff \text{number}(B) \land \text{zero}(B).
\]

\[
\begin{align*}
1\cdot x + 3\cdot y + 5 &= 0 \land 3\cdot x + 2\cdot y + 8 &= 0 \\
\text{compute}((2\cdot y + 8) - ((3\cdot y + 5)/1)\cdot 3, P3) &\% P3 = -7\cdot y + -7 \\
1\cdot x + 3\cdot y + 5 &= 0 \land -7\cdot y + -7 &= 0 &\% Y = -1 \\
\text{compute}((1\cdot x + 5) - ((-7)/-7)\cdot 3, P3') &\% P3' = 1\cdot x + 2 \\
1\cdot x + 2 &= 0 \land -7\cdot y + -7 &= 0 &\% X = -2
\end{align*}
\]
Confluence

Given a goal, every computation leads to the same result no matter what rules are applied.
A decidable, sufficient and necessary condition for confluence of terminating CHR programs through joinability of critical pairs.

\[
X \leq X \iff \text{true} \quad \text{(reflexivity)}
\]

\[
X \leq Y \land Y \leq X \iff X = Y \quad \text{(antisymmetry)}
\]

Start from overlapping minimal states
Confluence

Given a goal, every computation leads to the same result no matter what rules are applied.
A decidable, sufficient and necessary condition for confluence of terminating CHR programs through joinability of critical pairs.

\[ X \leq X \iff true \] (reflexivity)

\[ X \leq Y \land Y \leq X \iff X = Y \] (antisymmetry)

Start from overlapping minimal states
Confluence

Given a goal, every computation leads to the same result no matter what rules are applied.
A decidable, sufficient and necessary condition for confluence of terminating CHR programs through joinability of critical pairs.

\[ X \leq X \iff \text{true} \quad \text{(reflexivity)} \]
\[ X \leq Y \land Y \leq X \iff X = Y \quad \text{(antisymmetry)} \]

Start from overlapping minimal states
Analogous to RAM and Turing machines. CHR is turing-complete. Every algorithm can be implemented in CHR with best known time and space complexity. [Sneyers, Schrijvers, Demoen, CHR’05]
Part III

Applications

6 Classical Applications

7 Trends in Applications

8 Application Projects
T. Frühwirth, P. Brisset

Voted Among Most Innovative Telecom Applications of the Year by IEEE Expert Magazine, Winner of CP98 Telecom Application Award.
T. Frühwirth, S. Abdennadher


Most Popular Constraint-Based Internet Application.
S. Abdennadher, M. Saft, S. Will

Operational at University of Munich. Room-Allocation for 1000 Lectures a Week.
Spatio-Temporal Reasoning

M. T. Escrig, F. Toledo,
Universidad Jaume I, Castellon, Spain.
Qualitative Spatial Reasoning: Theory and Practice,
Qualitative Spatial Reasoning on 3D Orientation Point Objects, QR2002.

Integrates orientation, distance, cardinal directions over points as well as extended objects.

- RCC Reasoning - B. Bennet, A.G. Cohn, Leeds UK.
- PMON logic for dynamical temporal systems - E. Sandewall, Linkoeping Univ.
Testing and Verification

Model Based Testing for Real: The Inhouse Card Case Study,
A. Pretschner, O. Slotosch, E. Aiglstorfer, S. Kriebel,
TU Munich,

- Automatic Generation of Test Data - J. Harm, University Rostock, Germany.
- Executable Z-Specifications - P. Stuckey, Ph. Dart, University Melbourne.
Agents and Actions


Specification and Verification of Agent Interaction...

- Multi Agent Systems Using Constrains Handling Rules, IC-AI 2002 - B. Bauer, M. Berger, Siemens Munich, Germany - S. Hainzer, Uni Linz, Austria.
- PMON logic for dynamical temporal systems with actions and change - M. Bjgareland, E. Sandewall, Linkoeping University, Sweden.
Semantic Web

COIN Context Interchange Project,
Stuart E. Madnick, MIT Cambridge.
Reasoning About Temporal Context Using
Ontology and Abductive CLP,
PPSWR 2004 LNCS 3208.

Semantic Web Reasoning for Ontology-Based Integration of Resources,
Liviu Badea, Doina Tiliuva and Anca Hotaran, PPSWR 2004 LNCS 3208.

Context Knowledge Representation and Reasoning in the Context Interchange
System, Applied Intelligence, Vol 13:2, 2000;
Context Interchange...for the intelligent integration of information, ACM
Java Memory Machine

JMM by Vijay Saraswat, IBM TJ Watson Research and Penn State Univ. Implementation JMMSolve by Tom Schrijvers, K.U. Leuven, Belgium

Conditional Read

\[ X_r = (\text{Cond})?X_w1:X_i \]

\[
\begin{align*}
\text{ite}(\text{true}, X_r, X_w1, X_i) & \iff X_r = X_w1. \\
\text{ite}(\text{false}, X_r, X_w1, X_i) & \iff X_r = X_i. \\
\text{ite}(\text{Cond}, X_r, X, X) & \iff X_r = X.
\end{align*}
\]
Lung Cancer Diagnosis

Veronica Dahl, Simon Fraser University, Vancouver, Canada.
Lung cancer is leading cause of cancer death, very low survival rate.
Use bio-markers indicating gene mutations to diagnose lung cancer.

Concept Formation Rules (CFR) in CHR.
Retractable constraints.

age(X,A), history(X,smoker),
serum_data(X,marker_type) \iff
marker(X,marker_type,P,B),
probability(P,X,B) \iff
possible_lung_cancer(yes,X).

Lung Cancer and Metastasis
Multimedia Transformation Engine for Web Presentations

Joost Geurts, University of Amsterdam.
Automatic generation of interactive, time-based and media centric WWW presentations from semi-structured multimedia databases.

Genre painting and Johannes Vermeer

Genre paintings, drawings or prints depict people in their everyday surroundings: at home, in a café or at work. They appear to be painted from life, but in reality were usually thought up in the artist’s studio. Sometimes (but not always!) they contain a moral lesson. In some works the message is clear, in other cases the viewer has to make an effort to interpret the picture. Often, however, these household scenes are simply decorative paintings designed to entertain and amuse.

The Kitchen Maid (c. 1658)
Business Rules for Optimization

MANIFICO - Francois Fages, Claude Kirchner, Hassan Ait-Kaci,…France

Business Rule: defines or constrains behavior or structure of business.
“A car must be available to be assigned to a rental agreement”.

DERBY EU Car Rent Case in CHR, O. Bouissou.

reservation(Renter, Group, From, To),
available(car(Id, Group, …), From) <=>
rentagreement(Renter, Id, From, To).
References

Google “constraint handling rules”

Essentials of Constraint Programming
Thom Frühwirth,
Slim Abdennadher

Constraint-Programmierung
Lehrbuch
Thom Frühwirth,
Slim Abdennadher