Constraint-Programmierung

Grundlagen und Anwendungen

Thom Frühwirth
Institut für Informatik
Programmierung und Softwaretechnik
Ludwig-Maximilians-Universität München
www.pst.informatik.uni-muenchen.de/~fruehwir/

April 2000

Inhalt

Constraint-Programmierung

Constraint Handling Rules (CHR)

- Syntax und Semantik
- Programmanalyse

Anwendungen und Projekte

Deduktive Datenbanken, Typtheorie,

Temporale Logik, zeitliches und räumliches Schließen,

Programmtransformation und Partielle Evaluation,

Interpreter- und Übersetzerbau,

Simulation und Verifikation Hybrider Systeme.

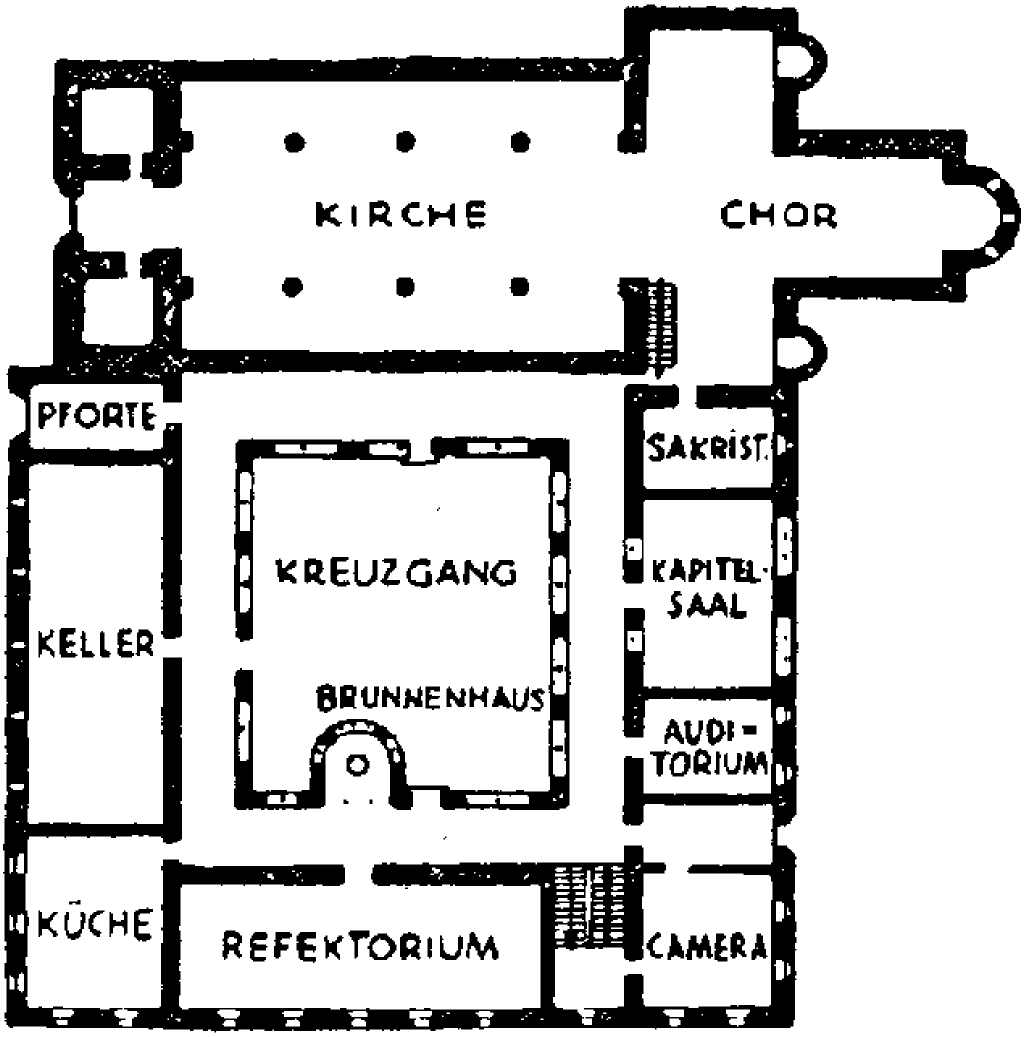
Planung von drahtlosen Kommunikationssystemen

Die Herausforderung

ECRC Forschungszentrum (Siemens, ICL, Bull)

- Lokale digitale Telekommunikation
- Braucht kleine Sendestationen
- Optimale Plazierung der Sender:
  - Vollständige Abdeckung
  - Minimale Senderanzahl

Beispiel Kloster



Constraint-Programmierung

Der Ansatz

- Beispiel Zahlenschloß:

  0 1 2 3 4 5 6 7 8 9

  Größer oder gleich 5.

  Primzahl.
- Deklarative Problemrepräsentation durch Variablen und Constraints:

  $x \in \{0, 1, \ldots, 9\} \wedge x \geq 5 \wedge primzahl(x)$
- Constraint Propagierung und Vereinfachung reduzieren den Suchraum:

  $x \in \{0, 1, \ldots, 9\} \wedge x \geq 5 \rightarrow x \in \{5, 6, 7, 8, 9\}$

[Frühwirth, Abdennadher, *Constraint-Programmierung*, Lehrbuch, Springer Verlag, 1997]

Constraint-Programmierung

Vorteile

- Deklarative Modellierung durch Constraints:
  Beschreibung der Eigenschaften und Beziehungen von tlw. unbekannten Objekten.
  $x \in \{0, 1, \ldots, 9\} \wedge x \geq 5 \wedge primzahl(x)$
- Automatisches Schließen, Constraintlösen:
  Propagierung der Effekte neuer Informationen.
  Vereinfachung macht implizite Inform. explizit.
  $x \in \{0, 1, \ldots, 9\} \wedge x \geq 5 \rightarrow x \in \{5, 6, 7, 8, 9\}$
- Kombinatorische Probleme gut lösen:
  Gute Kombinierbarkeit von Constraintlösen mit Such- und Optimierungsverfahren.

Constraint-Programmierung

Kommerzielle Anwendungen

- Lufthansa: Personalplanung nach Störungen.
- Renault: Kurzfristige Fertigungsplanung.
- Airbus: Layout der Pilotenkabine.
- Siemens: Verifizierung von Schaltkreisen.
- Caisse d'epargne: Portfolio Management.

In Entscheidungsunterstützungssystemen zum Planen, Konfigurieren, für Design, Analyse.

Das Problem

Unterschiedliche Klassen von Anwendungen haben unterschiedliche Constraints.

Constraintlöser ist eine Blackbox.
Kein Einblick in Implementierung.

- Anpassung ineffizient und unflexibel.
- Kombination von Constraintlösern schwierig.
- Fehlersuche und Programmanalyse schwierig.

Mindert Vorteile der Constraint-Programmierung.

Lösungsvorschläge

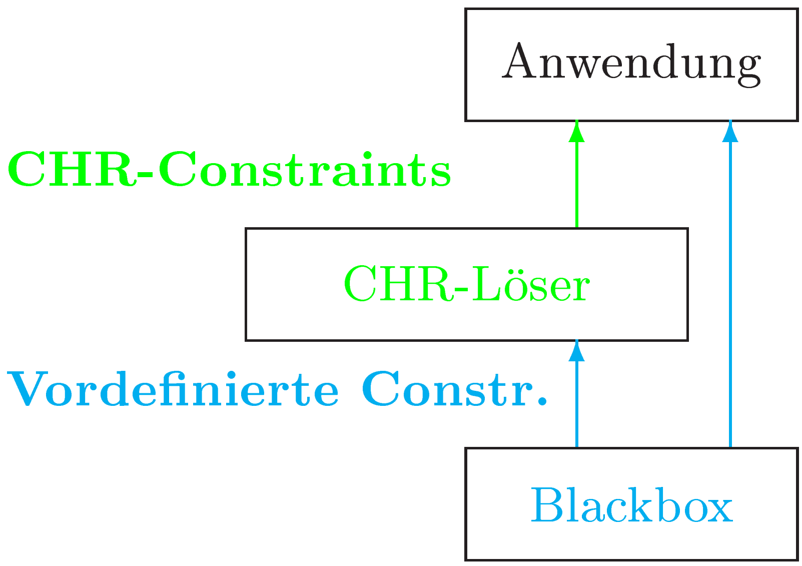
- Constr.-Konstruktoren, v. Hentenryck, 1991
- Meta-Constraints, Older, 1992
- Attributierte Variablen, Holzbaur, 1992
- Index-Constraints, Codognet, 1994

Begrenzte Erweiterbarkeit bzw. hohe Kompliziertheit.

Constraint Handling Rules (CHR)

Höhere, deklarative Sprache zur Spezifikation und Implementierung von Constraintlösern.

CHR-Constraintlöser sind einsehbar, anpaßbar, komponierbar, debugbar, analysierbar.



Basissprache (Prolog, Lisp, Java,...)

[Frühwirth, *Journal of Logic Programming*, Vol 37(1-3), Elsevier, 1998]

Beispiel Ordnungsrelation $\leq$

`x≤x ⇔ true`	Reflexivitaet
`x≤y ∧ y≤x ⇔ x=y`	Antisymmetrie
`x≤y ∧ y≤z ⇒ x≤z`	Transitivitaet

`a≤b ∧ b≤c ∧ c≤a`	*Anfrage*
	Transitivitaet
`a≤c`	
	Antisymmetrie
`a=c`	Blackbox-Loeser
	Antisymmetrie
`a=b`	
`a=b ∧ a=c`	*Antwort*

Deklarative Semantik

Kopf	K	CHR-Constraints
Wächter	W	vordefinierte Constraints
Rumpf	R	vordefinierte u. CHR-Constraints

Simplifikationsregel

$$K \Leftrightarrow W \mid R \qquad \forall\, (W \rightarrow (K \leftrightarrow \exists \bar{x}\; R))$$

Propagierungsregel

$$K \Rightarrow W \mid R \qquad \forall\, (W \rightarrow (K \rightarrow \exists \bar{x}\; R))$$

Deklarative Semantik eines CHR-Programms P

$\mathcal{P}$: Logische Leseweise der CHR-Regeln von P

$\cup$ $\mathcal{T}$: Theorie für vordefinierte Constraints

Operationale Semantik

Zustandsübergangssystem

Zustand: Konjunktion von Constraints.

$$
\begin{array}{ll}
\textit{Simplifizieren} & A \wedge C \;\longmapsto\; R\sigma \wedge C \\
& \text{wenn } (K \Leftrightarrow W \mid R) \in P \text{ und} \\
& (A = K\sigma) \text{ und } \mathcal{T} \models \forall\, (C_V \rightarrow W\sigma) \\
\textit{Propagieren} & A \wedge C \;\longmapsto\; R\sigma \wedge A \wedge C \\
& \text{wenn } (K \Rightarrow W \mid R) \in P \text{ und} \\
& (A = K\sigma) \text{ und } \mathcal{T} \models \forall\, (C_V \rightarrow W\sigma)
\end{array}
$$

C_V steht für die vordefinierten Constraints von C.

Übereinstimmung der Semantiken

Lemma (Äquivalenz von Zuständen) Wenn der Zustand B in einer Ableitung von A vorkommt, dann gilt

$$\mathcal{P}, \mathcal{T} \models \forall\ (A \leftrightarrow \exists \bar{x} B).$$

Satz (Korrektheit) Wenn C eine Antwort von A ist, dann gilt

$$\mathcal{P}, \mathcal{T} \models \forall\ (A \leftrightarrow \exists \bar{x} C).$$

Satz (Vollständigkeit) A habe mindestens eine endliche Ableitung. Wenn $\mathcal{P}, \mathcal{T} \models \forall\ (A \leftrightarrow \exists \bar{x} C)$, dann hat A eine Antwort C', sodaß

$$\mathcal{P}, \mathcal{T} \models \forall\ (\exists \bar{x} C \leftrightarrow \exists \bar{y} C').$$

[Abdennadher, Frühwirth, Meuss, *Constraint Journal*, Vol 4(2), Kluwer, 1999]

Programmanalyse

Terminierung

CHR-Constraintlöser terminieren meist mit einfachen Termordnungen.

[Frühwirth, Kapitel in Springer LNAI, 2000]

Konfluenz

Die Antwort auf eine Anfrage ist unabhängig von der Reihenfolge der Regelanwendung.

[Abdennadher, Frühwirth, Meuss, *Constraint Journal*, Vol 4(2), Kluwer, 1999]

Operationale Äquivalenz

Constraints, die in unterschiedlichen Programmen definiert sind, haben das gleiche Verhalten.

[Abdennadher, Frühwirth, *CP'99*, Springer LNCS, 1999]

Konfluenz

- Impliziert Konsistenz.
- Verbessert Übereinstimmung der Semantiken.
- Enthüllt Programmierfehler.

$$\texttt{x}\leq\texttt{y} \;\wedge\; \texttt{y}\leq\texttt{x} \;\Leftrightarrow\; \texttt{x}=\texttt{y}$$
$$\texttt{x}\leq\texttt{y} \;\wedge\; \texttt{y}\leq\texttt{z} \;\Rightarrow\; \texttt{x}\leq\texttt{z}$$

$$\texttt{x}\leq\texttt{y} \;\wedge\; \texttt{y}\leq\texttt{x}$$

$$\texttt{x}\leq\texttt{y} \;\wedge\; \texttt{y}\leq\texttt{x} \;\wedge\; \texttt{x}\leq\texttt{x} \qquad\qquad \texttt{x}=\texttt{y}$$

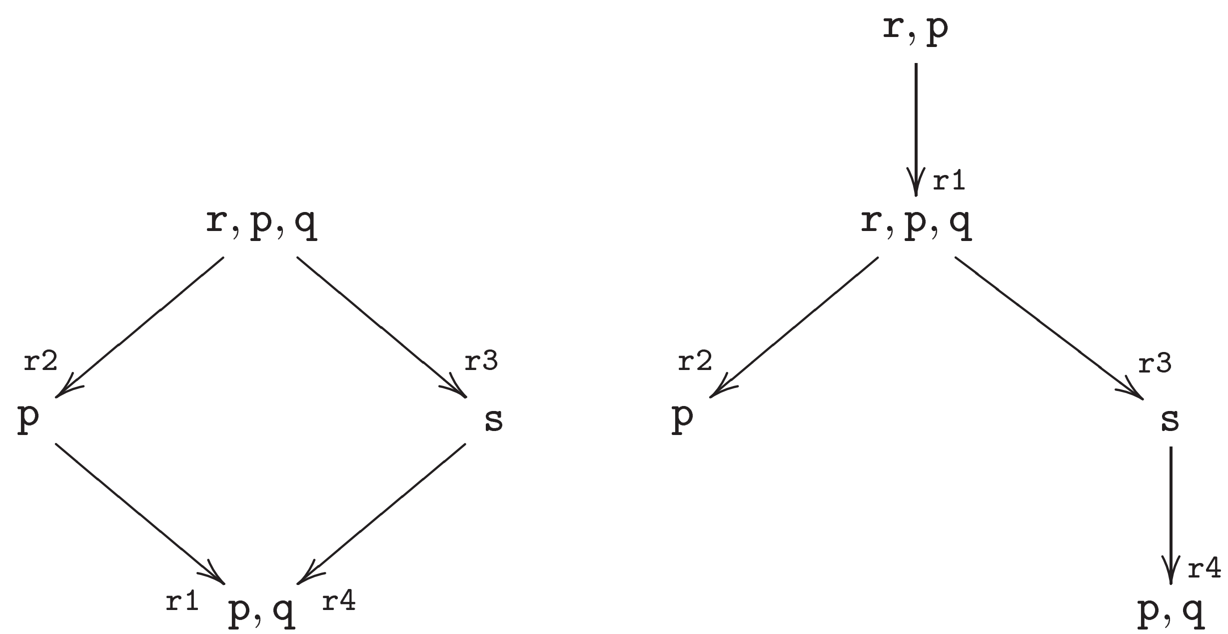
$$\texttt{x}=\texttt{y} \;\wedge\; \texttt{x}\leq\texttt{x} \qquad\qquad \texttt{x}=\texttt{y}$$

Satz (Konfluenz) Ein terminierendes CHR-Programm ist konfluent gdw. seine *kritischen Paare zusammenführbar* sind.

[Abdennadher, Frühwirth, Meuss, *Constraint Journal*, Vol 4(2), Kluwer, 1999]

Konfluenz bei Propagierung

`p ⇒ q.`	r1
`r,q ⇔ true.`	r2
`r,p,q ⇔ s.`	r3
`s ⇔ p,q.`	r4



Operationale Äquivalenz

- Korrektheit von Programmtransformationen.
- Kombination von Constraintlösern.
- Bibliotheken mit ähnlicher Funktionalität.

```
max(x,y,z) ⇔ x≤y | z=y.    max(x,y,z) ⇔ x<y | z=y.
max(x,y,z) ⇔ x>y | z=x.    max(x,y,z) ⇔ x≥y | z=x.
```

```
max(x,y,z) ∧ x≤y

z=y ∧ x≤y
```

Satz (Äquivalenz) Zwei terminierende und konfluente CHR-Programme P_1 und P_2 sind operational äquivalent gdw. ihre *kritischen Zustände* P_1, P_2*-zusammenführbar* sind.

[Abdennadher, Frühwirth, *CP'99*, Springer LNCS, 1999]

Optimale Sender-Plazierung

Der Constraint-Ansatz

Simulation der Funkwellen

- Gebäude-Modell: Material der Wände und Decken
- Funkwellenausbreitung durch Pfadverfolgung

Plazierung der Sender

- Suche Wahlweises Gleichsetzen von Sendern
- Constraintlösen Schneiden der Funkzellen
- Optimierung Branch and Bound

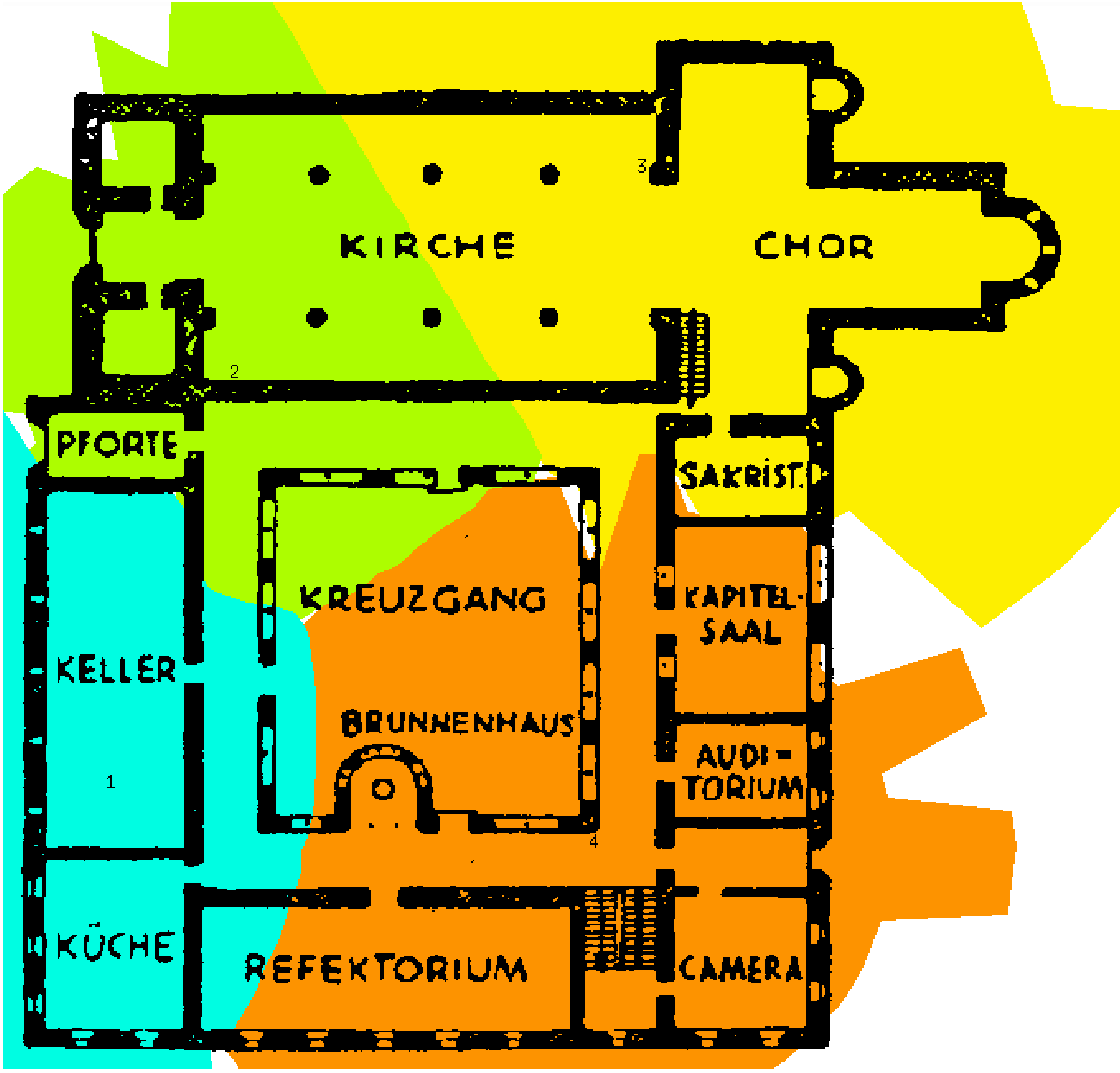
[Frühwirth, Brisset, IEEE Intelligent Systems Magazine, IEEE Press, Vol 15(1), 2000]

CHR Constraint-Programm

```
% Constraintloeser; Sender, Funkzellen

S in F ==> nicht_leer(F).
S in F1, S in F2 <=>
    durchschnitt(F1, F2, F3), S in F3.

% Suche; Liste

waehle(X,cons(Y,L)) <=> (X=Y ; waehle(X,L)).

sender_gleichsetzen(nil) <=> true.
sender_gleichsetzen(cons(S,L)) <=>
    (waehle(S,L) ; true),
    sender_gleichsetzen(L).

% Optimierung; Constraints, Max. Anzahl, Loesung

branch_and_bound(C,M,L) <=>
    constraints(C,S), max_sender_anzahl(S,M),
    sender_gleichsetzen(S), sender_anzahl(S,A),
    (bb(C,A,L) ; L=S).
```

Beispiel Kloster

IEEE Expert Magazine Telecom-Innovation 1996.
Telecom Application Award auf der CP 1998.



Zusammenfassung

Constraint-Programmierung

- Modellieren mit unvollständiger Information
- Lösen kombinatorischer (Optimierungs-)Probleme

in Entscheidungsunterstützungssystemen.

Constraint Handling Rules (CHR)

Höhere Sprache für Constraintlöser.

- Optimale Anpassung an Anwendungsklassen
- Deklarative Semantik
- Einfacher Kalkül
- Theoretische Fundierung
- Gute theoretische Eigenschaften
- Programmanalyse lohnenswert
- Effiziente Implementierungen
- Konkrete Anwendungen
- Weltweit im Einsatz...

CHR Anwender (Auswahl)

7 Implementierungen

Eclipse, Sicstus Prolog (CHR Online), Java...

M.T. Escrig and F. Toledo, Univ. Castellon

Qualitative Spatial Reasoning, IOS Press, 1998.

B. Bauer, Siemens München

Planen von verteilten Software-Agenten

K. Apt, CWI Amsterdam

Automatische Generierung von Constraintlösern

A. G. Cohn, Univ. Leeds

Topologisches räumliches Schließen mit RCC

E. Sandewall, Univ. Linköping

PMON Zeit-Aktions-Logik

A. Frisch, Univ. York

Vergleich von Resolutionsverfahren

Ausführbare Z-Spezifikationen, Abstrakte Interpretation für CLP(R), Automatische Generierung von Testdaten,...

[Constraint Handling Rules, *Sondernummer Journal of Applied Artificial Intelligence*, Taylor & Francis, 2000]

Meine Anwendungen und Projekte

- Planung drahtloser Kommunikationssysteme
  Telecom Application Award auf der CP 1998
- Münchener Mietspiegel Online
  Beste Anwendung auf der JFPLC 1996
- Stunden- und Raumplanung Online
  Seit WS 97/98 im Einsatz am Institut

Leitung laufender Drittmittel-Projekte

- FLPC II - Funkt.-log. Prog. mit Constraints
  Visualisierung und Diagrammatisches Schließen für UML mit Monash Uni Melbourne; DFG
- JACK - Java Constraint Kit
  mit Siemens, FAST, Uni Tandil; BMBF/SCyT
- DExVal - Simulation u. Verif. Hybrider Systeme
  mit BMW, Siemens, PUC Rio; BMBF/CNPQ
- ZEITRAUM - Zeitlich-räumliches Schließen
  mit Univ. Pisa; VIGONI DAAD

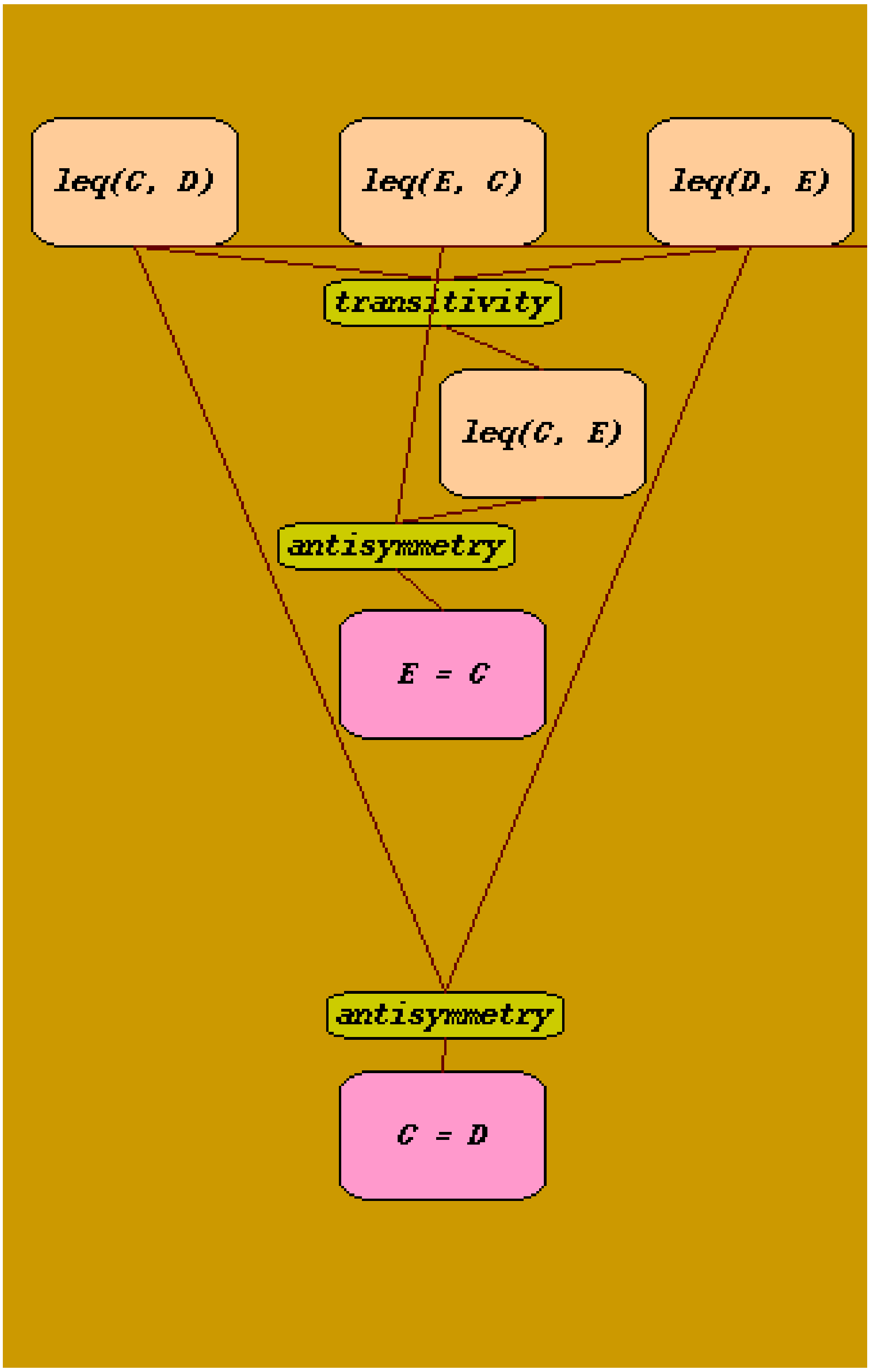
[LMU, *Constraint-Arbeitsgruppe Faltblatt*, 1998]

Ausblick

Programmiersprachen

- Programmanalyse von CHR
  Terminierung, Äquivalenz, Komplexität,...
- Einbettung von Constraintlöser-Komponenten
  in große, verteilte Softwaresysteme (JACK)
- JACK Compiler und Entwicklungsumgebung
  (Debugger, Visualisierung)
- Testfallgenerierung mit Constraints (DexVal)
- Diagrammatisches Schließen für UML (FLPC II)

Visualisierung Beispiel $\leq$



Programmiersprachen-Vergleich

CHR	Prolog	TRS
Spez. Sprache	Allg. Sprache	Allg. Sprache
Relationen	Relationen	Funktionen
Constraints	Prädikate	Terme
Simplifikationsreg.	Klauseln	Ersetzungsreg.
Propagierungsreg.	-	-
Mehrköpfige Reg.	-	(AC)
Committed-choice	Backtracking	Committed-ch.
Matching	Unifikation	Matching
Wächter	-	Bedingungen

Constraint-Programmierung

Zinseszinsrechnung

`D` Darlehen, derzeitige Schuld

`M` Dauer der Rückzahlung in Monaten

`Z` Zinssatz pro Monat

`R` Monatliche Rückzahlungsrate

`S` Rest-Schuld nach M Monaten

```
darlehen(D, M, Z, R, S):-
      M = 0,
      D = S.
darlehen(D, M, Z, R, S):-
      M  > 0,
      M1 = M - 1,
      D1 = D + D*Z - R,
      darlehen(D1, M1, Z, R, S).
```