Ein Constraint-Solver Framework für funktionale Constraints zur wissensbasierten Konfiguration in technischen Domänen

DA-1 Wolfgang Runte

Fachbereich Mathematik / Informatik
Universität Bremen

29.01.2003

Übersicht

- Motivation
- Konfiguration in EngCon
- Problemstellung
- Ziele der Arbeit
- Zusammenfassung

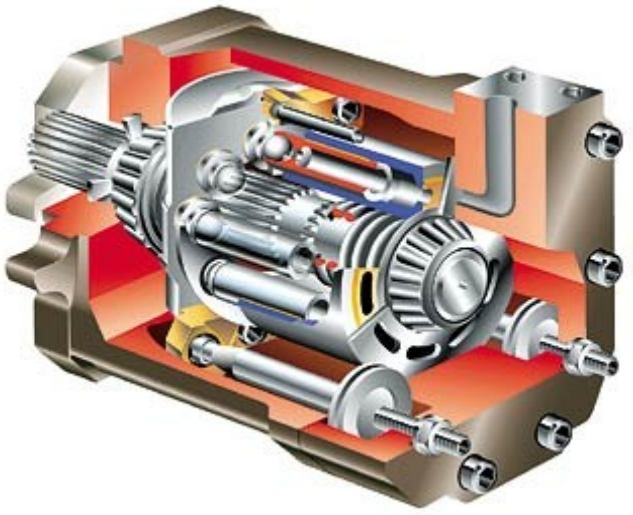
Motivation

Konfiguration (1)

- Beherrschung komplexer Systeme
- Fehlerminimierung durch konsistente Lösungen
- Beschleunigung der Angebotserstellung
- höhere Qualität der Lösungen

Beispiele:

- Antriebssysteme
- Gebäude
- Küchen
- PCs
- ...



Motivation

Konfiguration (2)

Konfigurieren ist das Zusammenfügen von Einzelkomponenten in einer Sequenz von einzelnen Konfigurationsschritten zu einer Gesamtlösung (*Konfiguration*).

Gekennzeichnet durch:

- großer Lösungsraum bei variantenreichen Produkten
- Rücknahme von Entscheidungen
- Behandlung von Abhängigkeiten

Motivation

Methoden

- Regelbasiertes Konfigurieren
- Strukturbasiertes Konfigurieren
- Constraintbasiertes Konfigurieren
- Ressourcenbasiertes Konfigurieren
- Fallbasiertes Konfigurieren
- Verhaltensbasiertes Konfigurieren

Verfahren sind selten in „Reinform“ anzutreffen – meistens Kombination mehrer Methoden

Übersicht

- Motivation
- **Konfiguration EngCon**
- Problemstellung
- Ziele der Arbeit
- Zusammenfassung

Konfiguration in EngCon

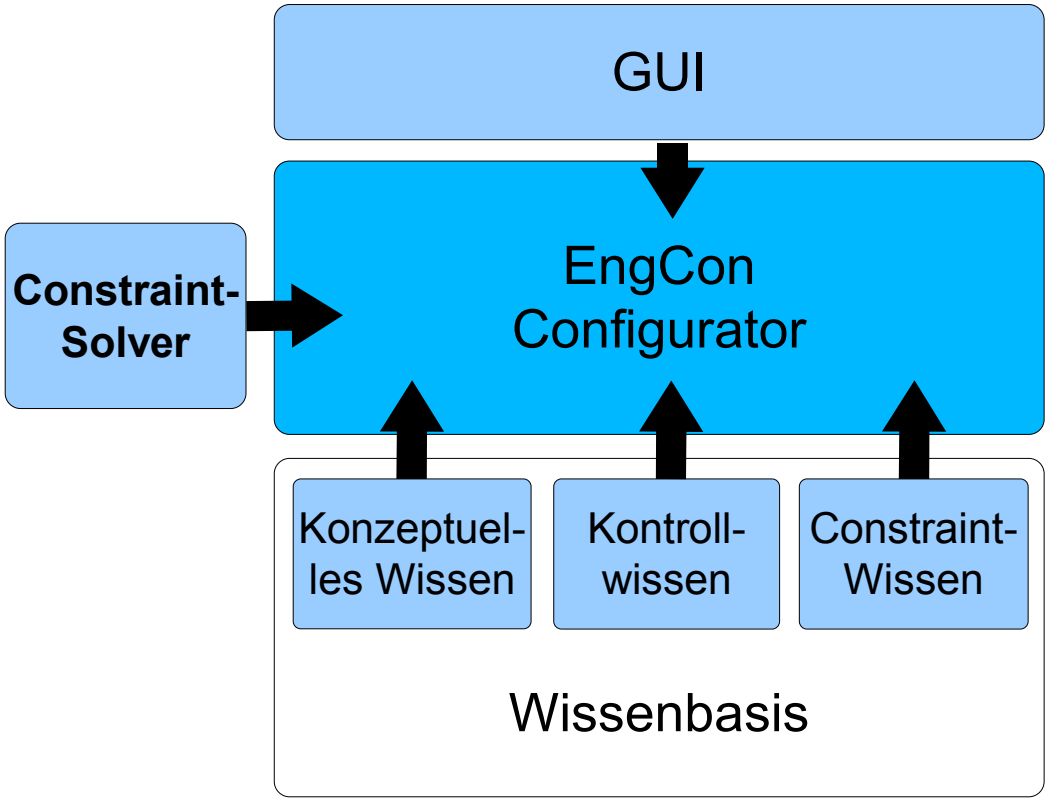
Historie

- Vorgänger: *Plakon*, *Konwerk*, entwickelt (u.a.) an der Universität Hamburg mit Unterstützung des BMBF
- prototypische Umsetzung auf moderne *JAVA*-Umgebung durch das TZI
- Überführung in ein Produkt durch die encoway GmbH: „Drive Solution Designer“ (DSD)
- Auszeichnung des DSD mit dem „Innovative Applications of Artificial Intelligence (IAAI) Award 2002“ der *American Association for Artificial Intelligence* (AAAI)
- Erfolgsmodell „Technologietransfer“

Konfiguration in EngCon

Architektur

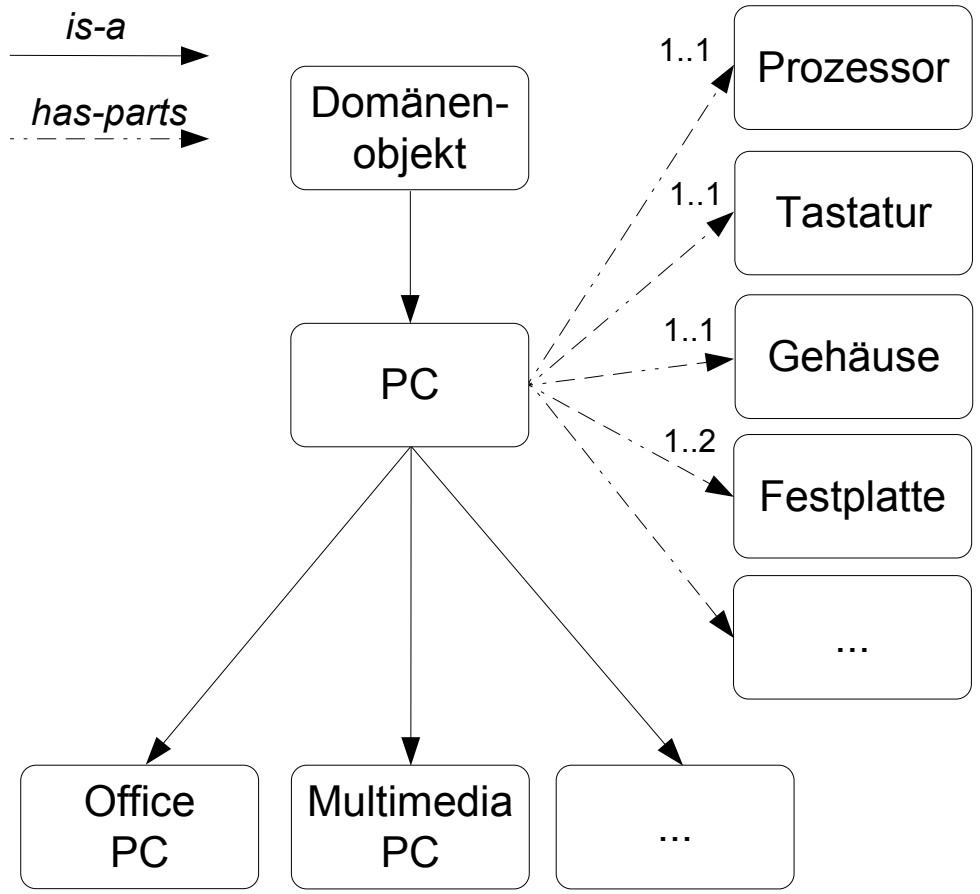
- domänenunabhängiges, *strukturbasiertes* Konfigurierungswerkzeug
- Bildung von Inferenzen aufgrund der wissens-basierten (hybriden) Architektur des Systems
- inkrementeller, interaktiver Konfigurationsverlauf, der zu *einer* Lösung führt (Tiefensuche)



Konfiguration in EngCon

Konzeptuelles Wissen

- Ontologie für die abstrakte Repräsentation der Struktur aller Lösungen des Konfigurationsproblems
- *Closed-World-Assumption*
- Konzepte stehen über *is-a* und *has-parts* Beziehungen in Relation
- Spezialisierungshierarchie / Taxonomie
- Zerlegungshierarchie / Partonomie



Konfiguration in EngCon

Kontrollwissen

- Kontrollmechanismus „interpretiert“ die Begriffshierarchie (*Begriffshierarchie-orientierte Kontrolle*)
- agendabasierte Steuerung der Suche im Lösungsraum
- Unterteilung in Phasen mittels „Strategien“
- Kontrollzyklus:
  1. Analyse der aktuellen Teilkonfiguration
  2. Konfigurationsschritt auswählen
  3. Bearbeitungsverfahren anwenden — spezialisieren, zerlegen, parametrieren
  4. Propagation des Constraint-Netzes

Konfiguration in EngCon

Constraint-Wissen

- repräsentiert Konfigurierungsrestriktionen zwischen Konzepten und Parametern der Begriffshierarchie
- Sicherstellung der Konsistenz der Konfiguration
- Propagation von Änderungen in einem Constraint-Netz
- 3-stufiges Constraint-Modell:
  - Constraint-Relationen
  - Konzeptuelle Constraints
  - Constraint-Netz

Konfiguration in EngCon

Constraint-Relationen

- **Funktions-Constraints**
  als (Un-) Gleichung formalisierte komplexe, funktionale Zusammenhänge

```
A = 150 * B
```

- **Extensionale Constraints / Tupel-Constraints**
  Aufzählung von Tupel aller möglichen Wertebereiche (relationale Abhängigkeit)

M_FSB	P_FSB	S_FSB
66	66	66
66	66	100
66	66	133
100	100	100
100	100	133
133	133	133

- **Java-Constraints**
  als *JAVA*-Methode implementierte „prozedurale Constraints“ für „beliebige“ Berechnungen

```
public static Vector setEqual(Vector a){
  if(a != null){
    if(a.size() != oldVector.size()){
      if(a.size() < oldVector.size())
        return a;
    }
  }
  return oldVector;
}
```

Konfiguration in EngCon

Konzeptuelle Constraints

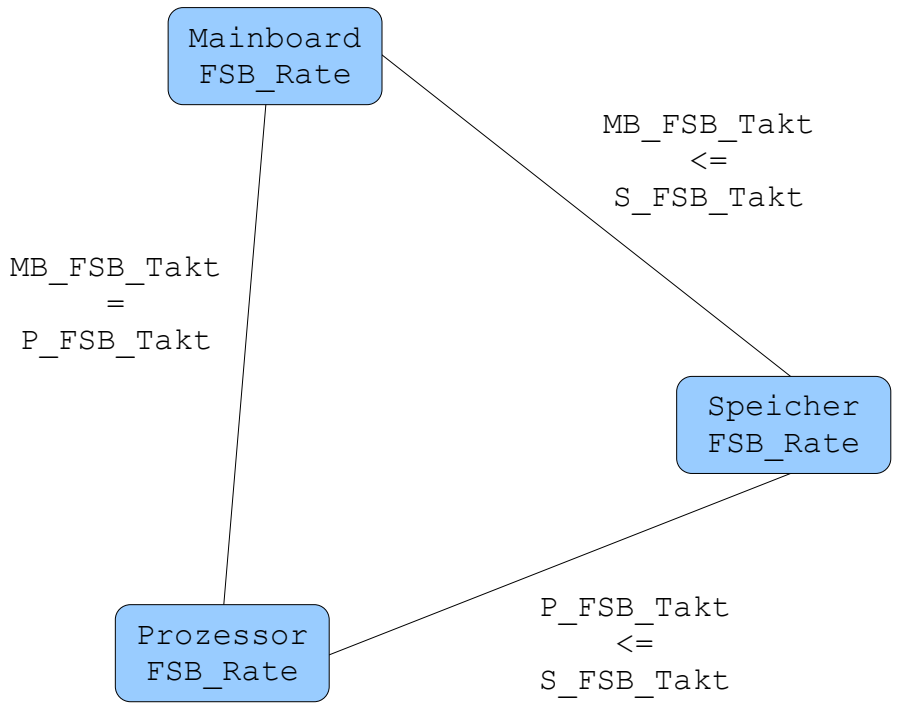
- Zuordnung der Constraint-Relationen zu den Instanzen des Konfigurationsprozesses
- Instantiierungsregeln / Bindungsmuster in Form von *Variablen-Pattern-Paaren*
- *Ein Pattern-Matcher* überprüft für neue Konzept-Instanzen, ob ein Variablen-Pattern-Paar erfüllt wird und instantiiert ggf. die entsprechenden Constraint-Relationen.

```
(def-konzeptuelles-constraint
    :name                       conc_AGP_Mainboard
    :variablen-pattern-paare ((?v :name VGA_Card
                                  :parameter((Bus 'agp)))
                              (?m :name Mainboard))
    :constraint-aufrufe      ((func_AGP_Mainboard (?m AGP_Slot))))
```

Konfiguration in EngCon

Constraint-Netz

- Inkrementeller Aufbau durch sukzessives instantiieren der Constraint-Relationen durch den *Pattern-Matcher.*
- Propagation der Wertebereiche der Constraint-Variablen bei jedem Konfigurationszyklus.
- Sicherstellung der Konsistenz, d.h. die Domänen der Constraint-Variablen dürfen nur gültige Wertebereiche bzgl. der sie enthaltenen Constraints aufweisen.



Übersicht

- Motivation
- Konfiguration EngCon
- **Problemstellung**
- Ziele der Arbeit
- Zusammenfassung

Problemstellung

Problem

- **Funktions-Constraints** werden in EngCon über einen externen Constraint-Solver propagiert (*local tolerance propagation*).
- Eine Eigenlösung hat aufgrund der Unabhängigkeit von einem Fremdhersteller den Vorteil der besseren Erweiterbarkeit, z.B.:
  - Constraint-Hierarchien
  - Constraint-Relaxion
  - ...
- Problem: Es gibt kein universelles mathematisches Verfahren, um alle Arten von Gleichungen aufzulösen ...

Problemstellung

Anforderungen an die Problemlösung (1)

Notwendige Anforderungen an einen Constraint-Solver für Funktions-Constraints in EngCon:

- Berechnung arithmetischer Funktionen auf finiten und infiniten Domänen
- Unterstützung von Intervallarithmetik mit hohem Präzisionsgrad
- Propagation eines inkrementell wachsenden Constraint-Netzes

- Unterstützung für die Rücknahme von Konfigurationsentscheidungen
- Beibehaltung der Trennung von Kontrolle und Constraint-System (*black box*)
- Berücksichtigung bestehender Schnittstellen

Problemstellung

Anforderungen an die Problemlösung (2)

Weitergehende Anforderungen:

- flexible Anpassbarkeit an unterschiedliche Anwendungsdomänen
- unterschiedliche Constraint-Lösungsmechanismen
- modularer Aufbau – zur flexiblen Verwendung von Lösungsmechanismen und für künftige Erweiterungen

Problemstellung

CSP – Definition

Ein **Constraint Satisfaction Problem** (CSP) ist ein Tripel

CSP(V,C,D)

V = $\{v_1, \ldots, v_n\}$ *endliche Menge* ***Variablen***

D = $\{D_1, \ldots, D_n\}$ *assoziierte* ***Wertebereiche*** $\{v_1 : D_1, \ldots, v_n : D_n\}$

C endliche Menge ***Constraints*** $c_i(V_i)$,

Teilmenge $V_i = \{v_{i_1}, \ldots, v_{i_l}\} \subseteq V$

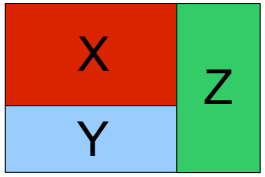
Lösungsraum für $c_i(V_i)$: $\{D_{i_1} \times \ldots \times D_{i_l}\}$

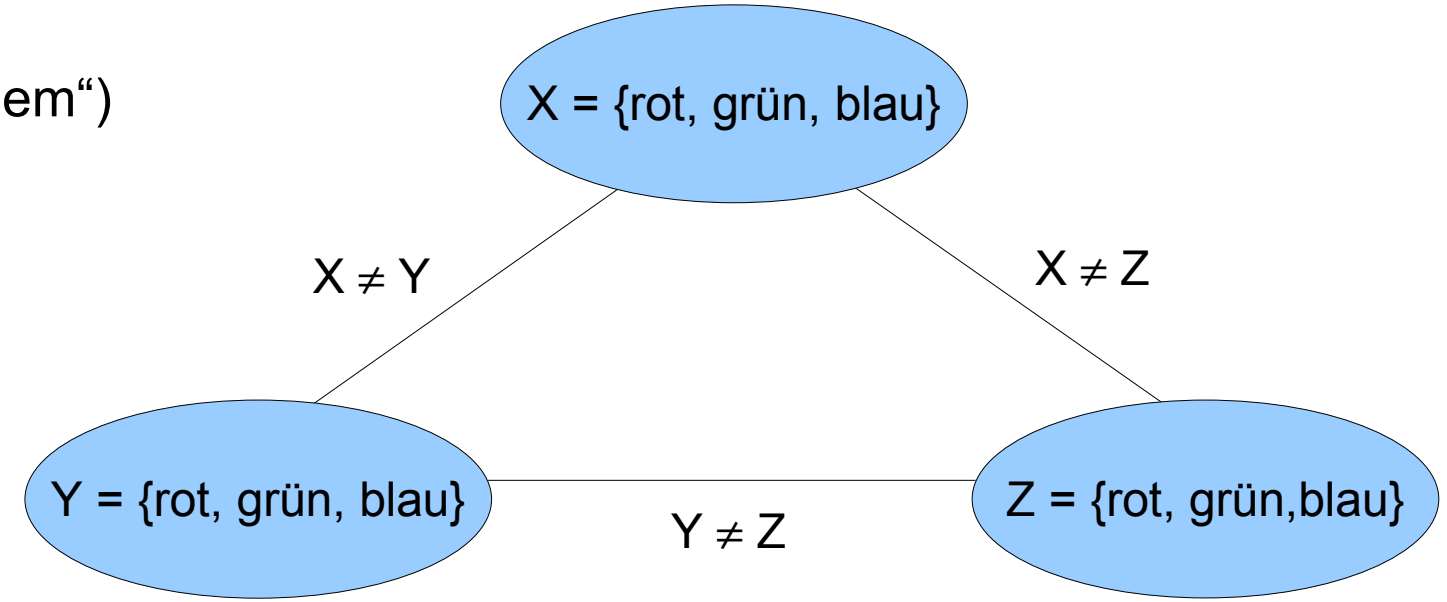
Problemstellung

CSP – Beispiel

Beispiel für einen binären Constraint-Graphen:

(„Kartenfärbeproblem“)





Problemstellung

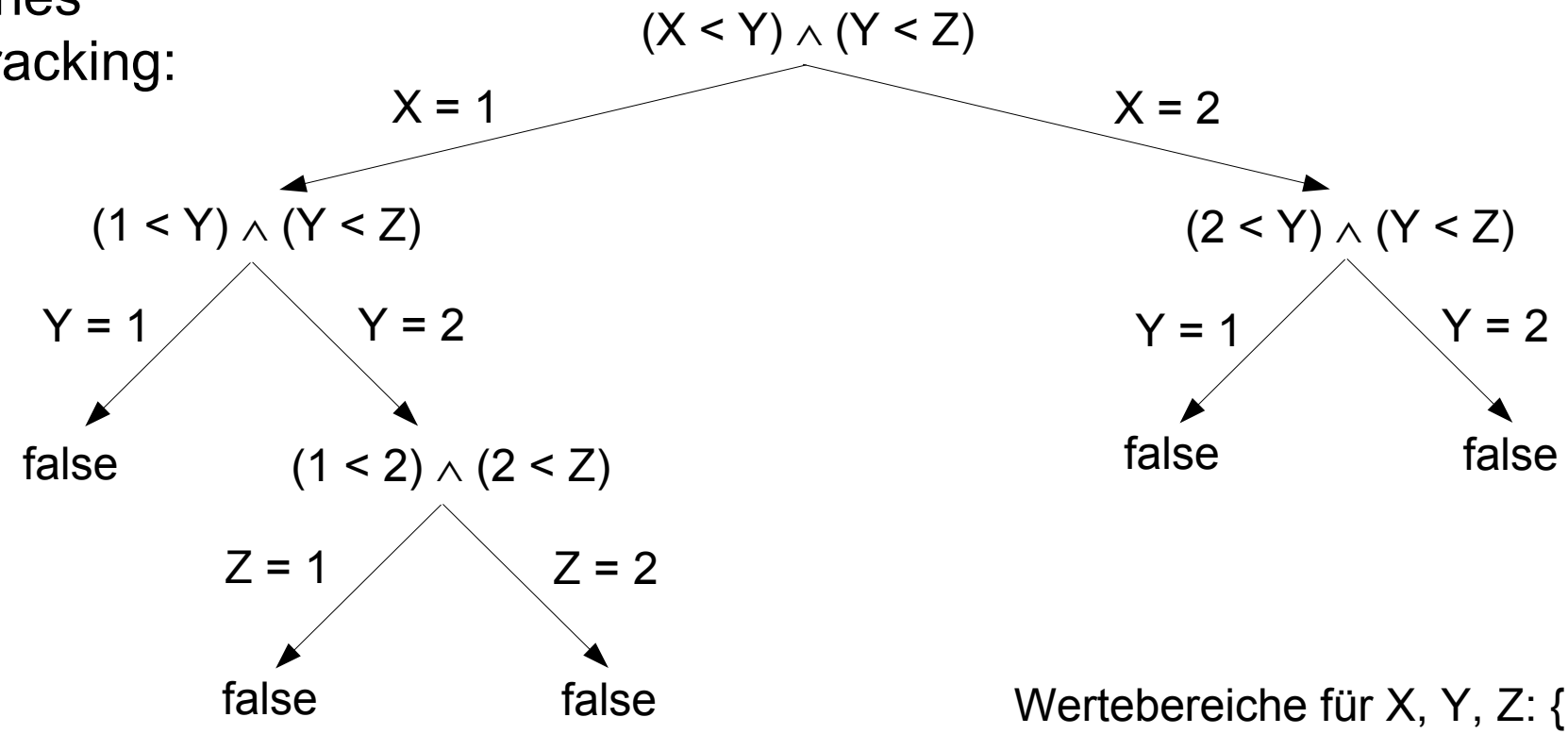
CSP – Backtracking (1)

- Ein klassisches CSP kann als **Suchproblem** gesehen werden (finite, diskrete Domänen).
- Zur Auflösung kann (chronologisches) **Backtracking** (CBT) zum Einsatz kommen
- Nachteil: ineffizient, expotentieller Aufwand
- Effizientere Verfahren zur systematischen Suche:
  - Look Back
    - Backjumping (BJ)
    - Backchecking (BC)
    - Backmarking (BM)
  - Look Ahead
    - Forward Checking (FC)
    - Partial Look Ahead (PLA)
    - (Full) Look Ahead (LA) / Maintaining Arc Consistency (MAC)

Problemstellung

CSP – Backtracking (2)

Beispiel-Suchbaum für einfaches Backtracking:



Wertebereiche für X, Y, Z: {1, 2}

Problemstellung

CSP – Konsistenztechniken (1)

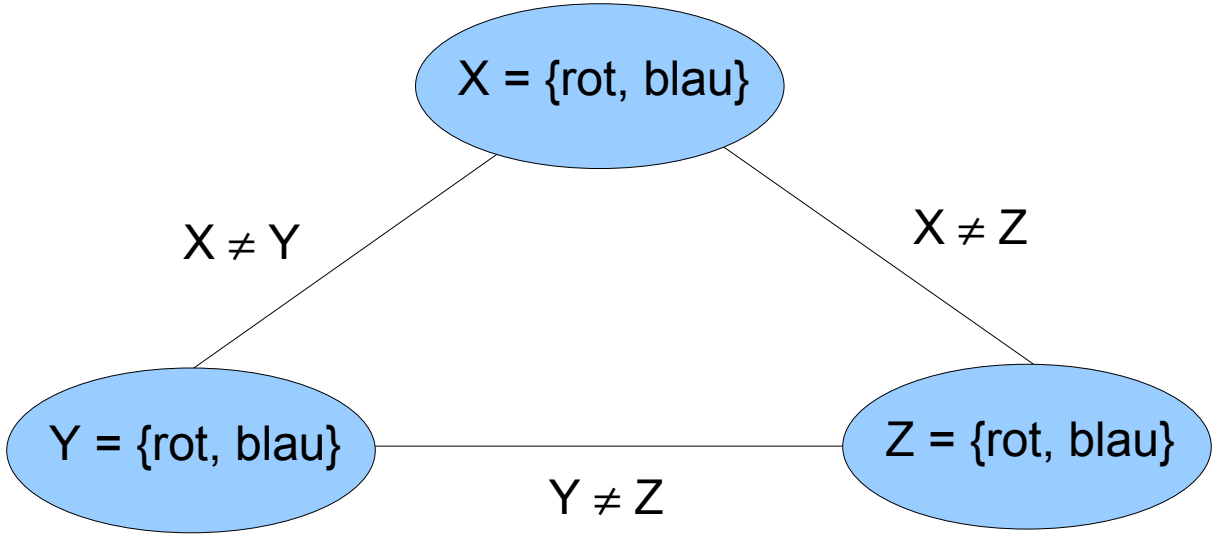
- Problemreduktion eines CSP mittels **Konsistenztechniken**
  (Ursprung: Walz 75, Mackworth 77)
- inkonsistente Werte aus den Domänen der Variablen entfernen
- erreichen im Regelfall lediglich *lokale Konsistenz*
- Konsistenzgrade:
  - **Knotenkonsistenz** (*node consistency*, NC)
  - **Kantenkonsistenz** (*arc consistency*, AC)
  - **Pfadkonsistenz** (*path consistency*, PC)
  - **k-Konsistenz** (*k-consistency*)

Problemstellung

CSP – Konsistenztechniken (2)

Beispiel für einen kantenkonsistenten Graphen („Kartenfärbeproblem“):

Keine globale Lösung vorhanden!



Problemstellung

Intervall CSP

- numerische CSP (NCSP) bzw. Intervall CSP (ICSP) besitzen reellwertige Intervall-Domänen $\rightarrow$ infinit & kontinuierlich
- Kombination von mathematischen Verfahren der Intervall-Arithmetik und Konsistenztechniken
- Verfahren:
  - label inference (Davis 87)
  - tolerance propagation (Hyvönen 92, Yang 97 / 00)
  - 2-B, 3-B, k-B-consistency (Lhomme 93)
  - box consistency (Benhamou et al. 94)
  - 2^k-tree consistency (Sam-Haroud 95, Faltings et al. 96)

Übersicht

- Motivation
- Konfiguration EngCon
- Problemstellung
- **Ziele der Arbeit**
- Zusammenfassung

Ziele der Arbeit

Lösungsansatz (1)

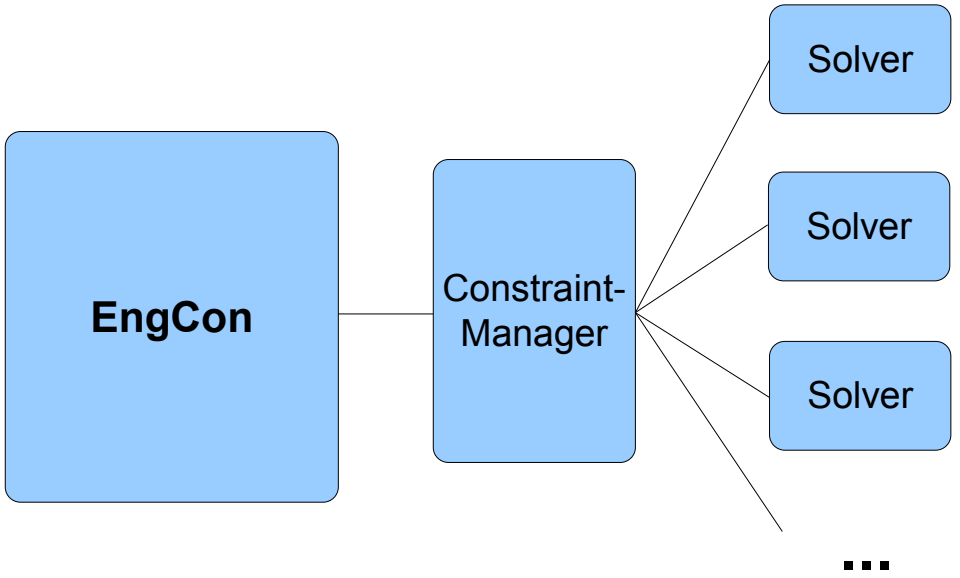
Entwicklung eines „Frameworks“ zur flexiblen Anbindung unterschiedlicher Constraint-Solver resp. Lösungverfahren:

- Berücksichtigung von Constraints über finite und infinite Domänen.
- Vor dem Hintergrund der Konfiguration ist jeweils abzuwägen, welcher Solver für welche Constraints zum Einsatz kommt.
- Evtl. „Vermischung“ der Domänen / Überlappung der Constraint-Netze erlauben (mittels geeigneter Heuristiken).

Ziele der Arbeit

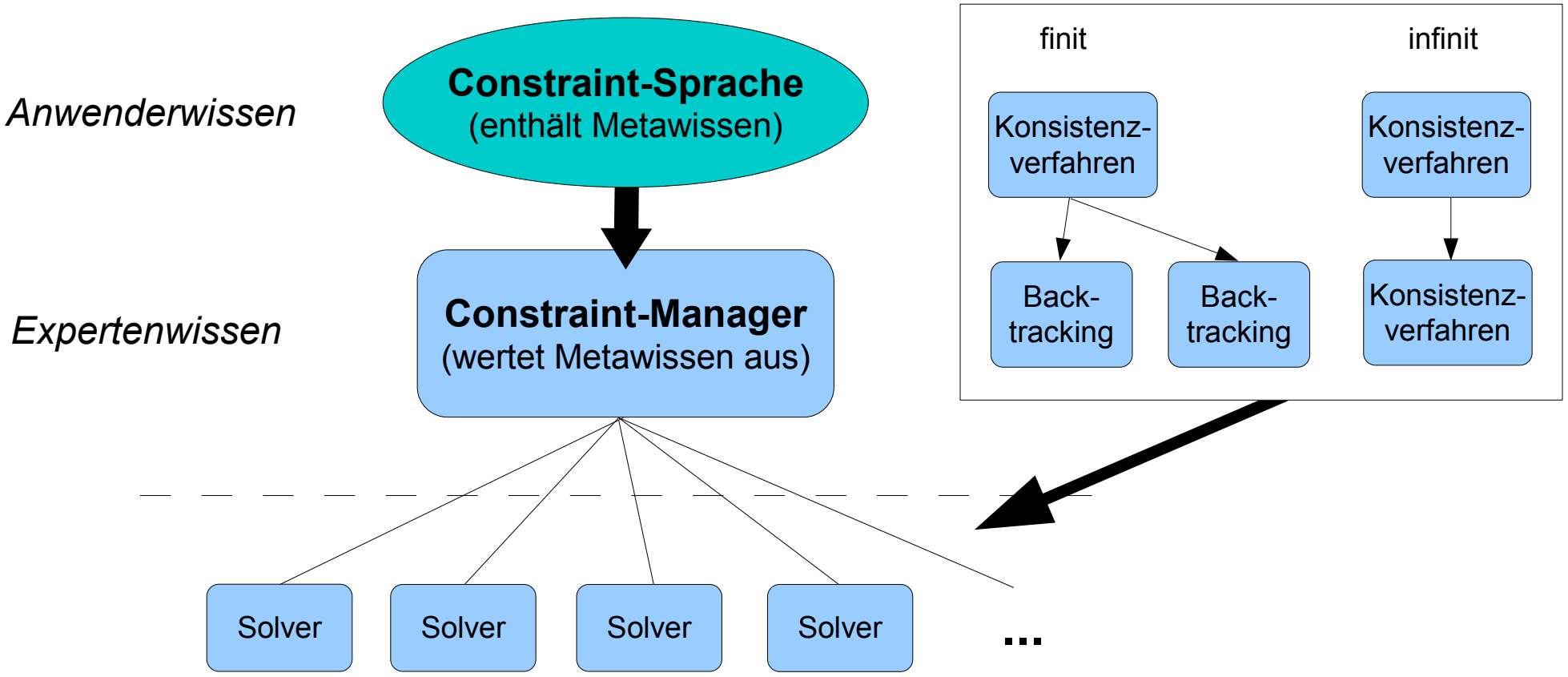
Lösungsansatz (2)

- *Constraint-Manager* verwaltet und kontrolliert, welche Constraints von welchem Solver verarbeitet werden sollen bzw. können.
- Erweiterung der „Constraint-Sprache“ um Eigenschaften (*Strategien*), z.B.:
  - linear / nicht-linear
  - finit / infinit
  - zu verwendender Solver



Ziele der Arbeit

Lösungsansatz (3)



Übersicht

- Motivation
- Konfiguration EngCon
- Problemstellung
- Ziele der Arbeit
- **Zusammenfassung**

Zusammenfassung

- Ersetzung des externen Constraint-Solvers für Funktions-Constraints für das strukturbasierte Konfigurierungswerkzeug EngCon
- intelligente Anbindung mehrerer hybrider Constraint-Solver und Propagationsmechanismen in einer modularen und flexibel erweiterbaren Architektur
- Unterstützung mehrerer Arten arithmetischer Constraints:
  - *diskrete, finite Domänen*
  - *reellwertige Intervall-Domänen*

Weiteres Vorgehen

- vor Abschluss: Constraint-Systeme evaluieren
- Verfahren bzw. Algorithmen vergleichen
  - funktionale Eigenschaften (finite / infinite Domänen, ...)
  - Leistungsfähigkeit (lineare / nicht-linear Gleichungen, ...)
  - Implementierungsaufwand
- Implementierung von geeigneten Verfahren

Literatur (1)

- Benhamou, Frédéric: Interval Constraint Logic Programming. In: Podelski, Andreas (Herausgeber): Constraint Programming: Basics and Trends, Band 910 der Reihe Lecture Notes in Computer Science, Seiten 1-21. Springer-Verlag, 1995.
- Benhamou, Frédéric, Frédéric Goualard, Laurent Granvilliers und Jean-Francois Puget: Revising Hull and Box Consistency. In: Proceedings of the 16th International Conference on Logic Programming (ICLP'99), Seiten 230-244, Las Cruces, USA, 1999. MIT Press.
- Benhamou, Frédéric, David McAllester und Pascal van Hentenryck: CLP(Intervals) Revisited. In: Proceedings of the International Symposium on Logic Programming, Seiten 124-138, Ithaca, NY, USA, 1994. MIT Press.
- Cleary, John G.: Logical Arithmetic. Future Computing Systems, 2(2):125-149, 1987.
- Davis, Ernest: Constraint Propagation with Interval Labels. Artificial Intelligence, 32(3):281-331, Juli 1987.
- Granvilliers, Laurent, Frédéric Goualard und Frédéric Benhamou: Box Consistency through Weak Box Consistency. In: Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'99), Seiten 373-380, Chicago, USA, 1999. IEEE Computer Society.

Literatur (2)

- Sam-Haroud, Jamila: Constraint Consistency Techniques for Continuous Constraints. PhD. Thesis No. 1423, Swiss Federal Institute of Technology (EPFL), Lausanne (Switzerland), 1995.
- Sam-Haroud, Djamila und Boi V. Faltings: Consistency Techniques for Continuous Constraints. CONSTRAINTS: An International Journal, 1(1/2):85-118, September 1996.
- Hyvönen, Eero: Constraint Reasoning Based on Interval Arithmetic: The Tolerance Propagation Approach. Artificial Intelligence, 58(1-3):71-112, Dezember 1992.
- Lhomme, Olivier: Consistency techniques for numeric CSPs. In: Bajcsy, Ruzena (Herausgeber): Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI'93), Seiten 232-238, Chambéry, France, 1993. IEEE Computer Society Press.
- Mackworth, Alan K.: Consistency in Networks of Relations. Artificial Intelligence, 8(1):99-118, 1977.

Literatur (3)

- Montanari, Ugo: Networks of Constraints: Fundamental Properties and Applications to Picture Processing. Information Science, 7(2):95-132, 1974.
- Puget, Jean-Francois und Pascal van Hentenryck: A constraint satisafaction approach to a circuit design Problem. Journal of Global Optimization, 13:75-93, 1998.
- Walz, David L.: Understanding line drawings of scenes with shadows. In: Winston, P.W. (Herausgeber): The Psychology of Computer Vision, Seiten 19-91. McGraw-Hill, New York, 1975.

Diskussion

Konfiguration

Systeme

- R1/XCON
- SICONFEX
- MMC-CON
- ALL-RISE

- Plakon
- Konwerk
- EngCon

- CAS-Konfigurator
- COSMOS
- ET-EPOS
- SCE
- SECON

- ConBaCon
- CAWICOMS

Problemstellung

Mathematische Lösungsverfahren

Konventionelle mathematische Verfahren (reellwertige Domänen)

klassische Methoden (lineare Constraints)

- Danzigs Simplex Algorithmus — lineare Programmierung
- Gaußsche Eliminierung — lineare Algebra

algebraische Algorithmen (nicht-lineare Constraints)

- Gröbner Basen — equalities
- PCAD (*Partial Cylindrical Algebraic Decomposition*) — inequalities

iterative nummerische Algorithmen (nicht-lineare Constraints)

- Newton
- Broyden
- Gauß-Jacobi
- Gauß-Seidel

Problemstellung

Constraint Satisfaction

„Klassische“ CSP (diskrete Domänen)

Systematische Suche

- Generate & Test
- Backtracking

Konsistenztechniken (Ursprung: Walz 75)

- Knotenkonsistenz (*node consistency*, NC)
- Kantenkonsistenz (*arc consistency*, AC)
  - directional arc consistency
- Pfadkonsistenz (*path consistency*, PC)
  - directional path consistency (DPC)
  - restricted path consistency (RPC)
- k-Konsistenz (*k-consistency*)

Propagationsverfahren

- Look Back
  - Backjumping (BJ)
  - Backchecking (BC)
  - Backmarking (BM)
- Look Ahead
  - Forward Checking (FC)
  - Partial Look Ahead (PLA)
  - Full Look Ahead (LA) / Maintaining Arc Consistency (MAC)

Problemstellung

Constraint Satisfaction

Constraint Optimization

- Constraint Satisfaction Optimization Problems (CSOP)
  - Branch & Bound

Over-Constrained Problems

- Partial Constraint Satisfaction Problems (PCSP)
  - standard Algorithmen ... (s.o.)
- Hierarchical Constraint Satisfaction Problems (HCSP) soft constraints
  - Refining Methods
  - Local Propagation

Problemstellung

Constraint-Systeme

Freie Systeme:

- Constraint Handling Rules (CHR)
- Java Constraint Kit (JACK)
- CLIP / CLP(Intervals)
- Java Constraint Library (JCL)
- IASover
- Cassowary
- Oz (object-oriented CLP)
- GNU Prolog
- DeLIC

Kommerzielle Systeme:

- JSolver
- ILOG Solver / JSolver
- ECLiPSe (CLP)
- UniCalc
- ICE InC++ Library
- CHIP (Cosytech)

Ziele der Arbeit

Lösungsverfahren

- „Vermischung“ von finiten und infiniten Wertebereichen
- Überschneidung von Constraint-Netzen mit Variablen unterschiedlicher Wertebereiche
- Heuristiken:
  - Intervall-Variable in finitem Constraint → Wertebereich diskretisieren (als Integer-Menge)
  - finite Variable in Intervall-Constraint → als Intervall [(1,1)(2,2)(3,3)]