Wissensbasierte Systeme

Sebastian Iwanowski
FH Wedel

Kap. 5: Modellbasierte Diagnose
Teil 3: Konfliktgenerierung und Wertpropagierung

Konfliktgenerierung

Die Kandidatengenerierung löst folgende Aufgabe:

- **Gegeben eine Menge von Konflikten: Finde die wahrscheinlichsten präferierten Diagnosen zu diesen Konflikten.**
- **Damit reduziert sich das Diagnoseproblem auf folgende Aufgabe: Finde die Menge der Konflikte !**

Was ist ein Konflikt ?

- **Zuweisung von genau einem Verhaltensmodus an einige Komponenten des Systems**
- **Ein Konflikt entspricht logisch einer Disjunktion von negativen Literalen**
- Zum Vergleich: Eine Diagnose entspricht einer Konjunktion von positiven Literalen

Wie entsteht ein Konflikt ?

- **durch Werte, die einander widersprechen**
- **Die sich widersprechenden Werte werden durch verschiedene Annahmen unterstützt.**
- **Dann muss eine dieser Annahmen falsch sein.**

TMS: Truth Maintenance System

Behauptungsknoten

Justifications

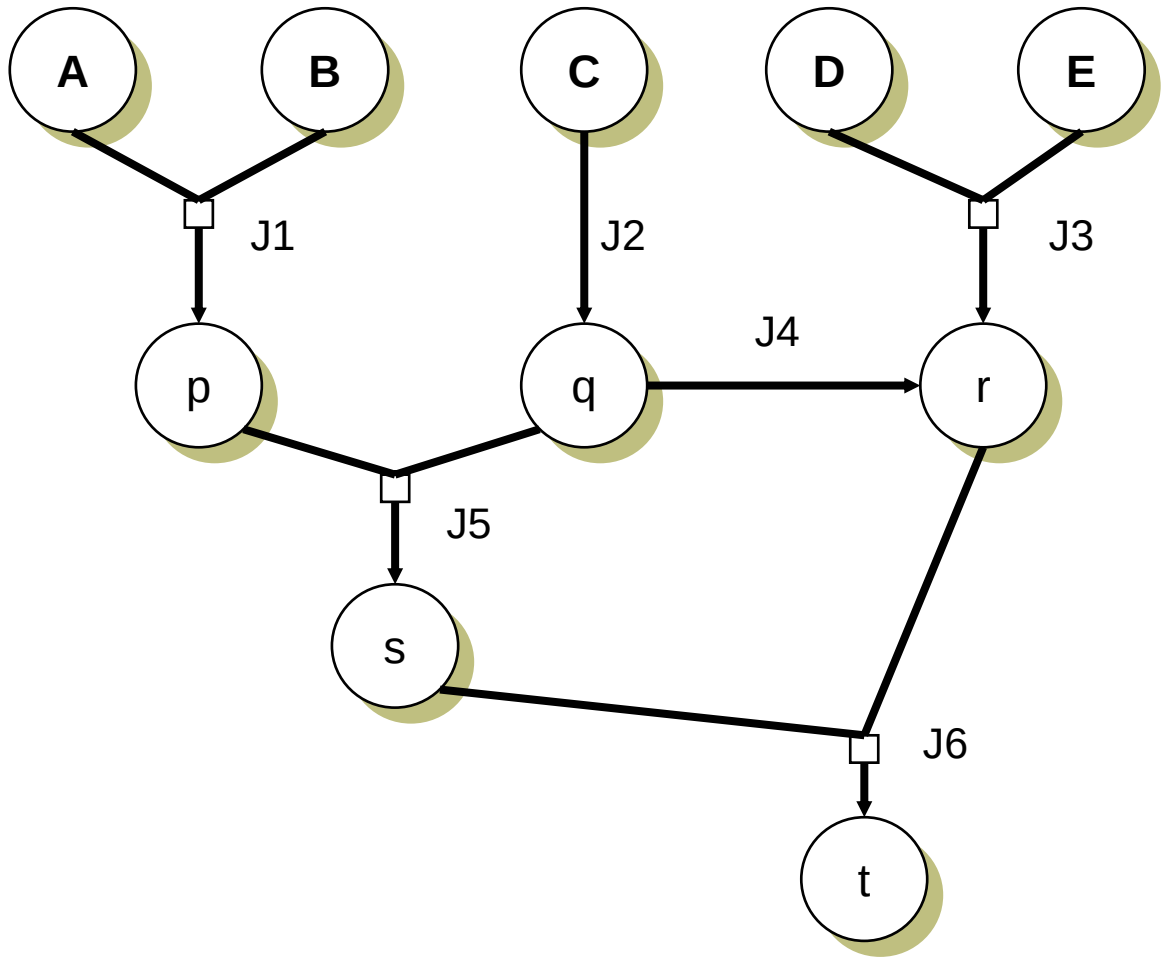
Behauptungsknoten

Justification

Behauptungsknoten

Justification

Behauptungsknoten



Aus der Kombination von Behauptungen entsteht durch eine Justification eine neue Behauptung

TMS: Truth Maintenance System

Begriffswelt TMS:

Behauptungsknoten (propositional node):

steht für eine beliebige Aussage (kann wahr oder falsch sein)

Justification:

$A_1 \wedge A_2 \wedge ... \wedge A_n \Rightarrow C$ wobei $A_1, A_2, ..., A_n, C$ Behauptungsknoten sind

$A_1, A_2, ..., A_n$ heißen Antecedents (Vorgänger) der Justification

C heißt Conclusion (Folgerung) der Justification

Widerspruchsknoten ($\perp$):

Steht für eine Behauptung, die auf keinen Fall gilt

ATMS: Assumption-based Truth Maintenance System

Funktionalität eines allgemeinen TMS:

1) **Bestimmte Behauptungsknoten werden als wahr angenommen (beliefs).**
2) **Das TMS stellt durch Propagation dieser Annahmen über die Justifications fest, welche anderen Behauptungen dann auch gelten müssen.**
3) **Insbesondere wird durch Benutzung des Widerspruchsknotens festgestellt, ob die Kombination von bestimmten Annahmen in sich widersprüchlich ist.**

Zusätzliche Funktionalität eines ATMS:

- **Es wird gleichzeitig mit mehreren Kontexten gearbeitet: Ein Kontext ist die Menge verschiedener Annahmen, die gleichzeitig gelten sollen.**

1) **Die Behauptungen werden mit den Annahmekontexten versehen, unter denen sie gelten müssen.**
2) **Das ATMS stellt durch Propagation dieser Annahmekontexte über die Justifications fest, welche anderen Behauptungen dann auch gelten müssen.**
3) **Insbesondere wird durch Benutzung des Widerspruchsknotens festgestellt, welche Annahmekontexte in sich widersprüchlich sind.**

ATMS: Assumption-based Truth Maintenance System

Begriffswelt ATMS:

Behauptungsknoten (propositional node):

Behauptungen werden unterschieden in normale Behauptungen und Annahmen (assumption node)

Environment:

Annahmekontext: *Konjunktion* von Annahmen, unter denen eine Behauptung gilt (wenn die Annahmen richtig sind)

Label:

Menge aller Annahmekontexte für einen Behauptungsknoten. Verschiedene Annahmekontexte müssen nicht gegeneinander konsistent sein (es gilt die *Disjunktion* der Environments).

Konflikt (nogood):

Environment des Labels des Widerspruchsknotens

ATMS: Assumption-based Truth Maintenance System

Anwendung eines ATMS für die modellbasierte Diagnose:

Behauptungsknoten (propositional nodes):

1) „Normale“ Knoten: Zuweisung eines konkreten Wertes an einer bestimmten Stelle (Variable) des Systems
2) Annahmeknoten: Zuweisung eines Verhaltensmodus an eine Komponente

Justification:

Anwendung einer Verhaltensregel auf konkrete Werte

Environment:

Gleichzeitige (Konjunktion) Zuweisung von Verhaltensmodi an Komponenten, unter der eine Behauptung gelten würde (Zuweisung muss nicht vollständig sein)

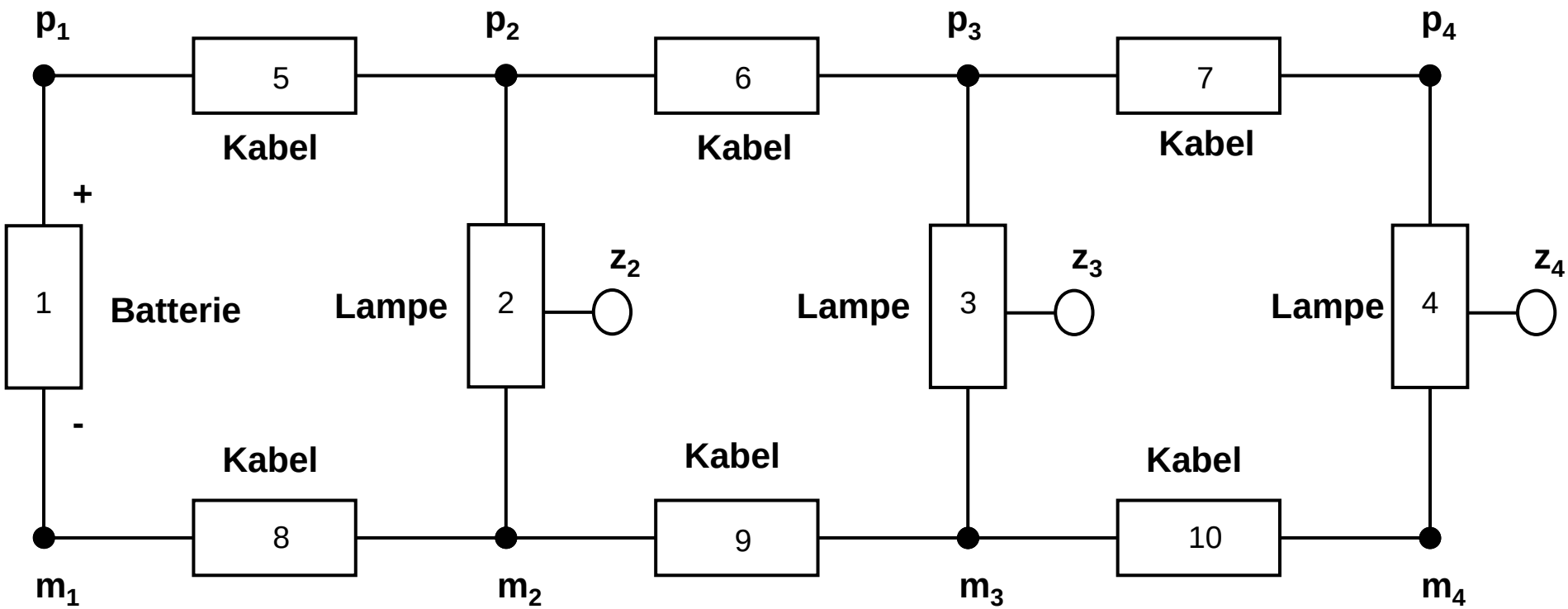
Bei Annahmeknoten: Zuweisung von einem Verhaltensmodus an genau eine Komponente

Konflikt (nogood):

Environment des Labels des Widerspruchsknotens: Zuweisung von Verhaltensmodi an Komponenten, deren Konjunktion nicht gelten kann (mindestens eine muss falsch sein)

Damit können Konflikte in der Notation und Bedeutung verwendet werden wie bei der Begriffswelt der GDE eingeführt (Teil 5.1).

Beispiel für die Benutzung eines ATMS



Verhaltensmodi:

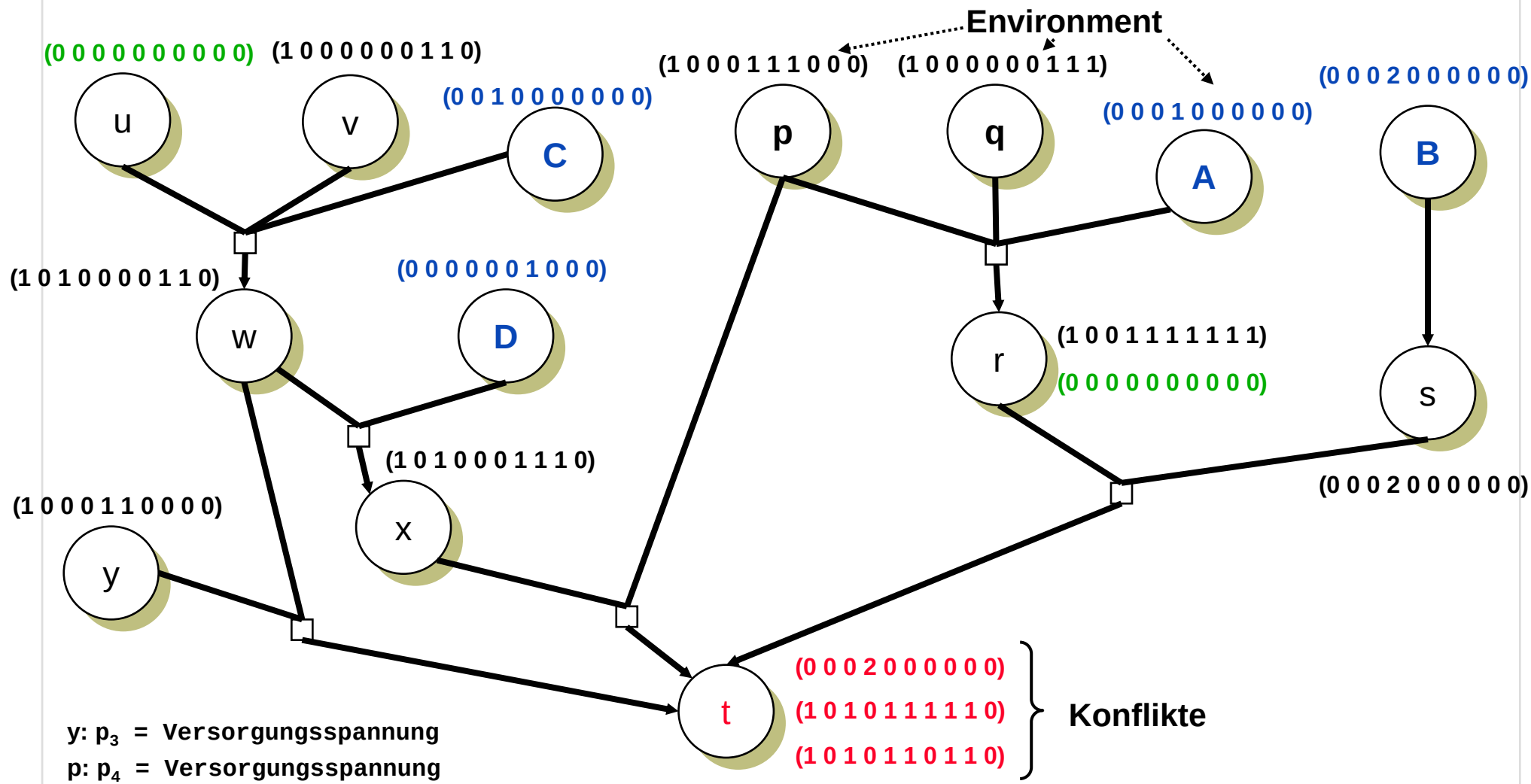
Modus 1 für alle Komponententypen: Normalverhalten

Modus 2 für alle Komponententypen: einziger Fehlermodus

Batterie: Modus 2 ⇒ (minus = `Masse`)

Lampe: Modus 2 ⇒ (z = `dunkel`)

Beispiel für die Benutzung eines ATMS



y: p_3 = Versorgungsspannung

p: p_4 = Versorgungsspannung

q: m_4 = Masse

A: Komponente 4 ist ok

r: z_4 = hell

B: Komponente 4 ist defekt

s: z_4 = dunkel

t: ⊥ (Widerspruch)

u: z_3 = dunkel

v: m_3 = Masse

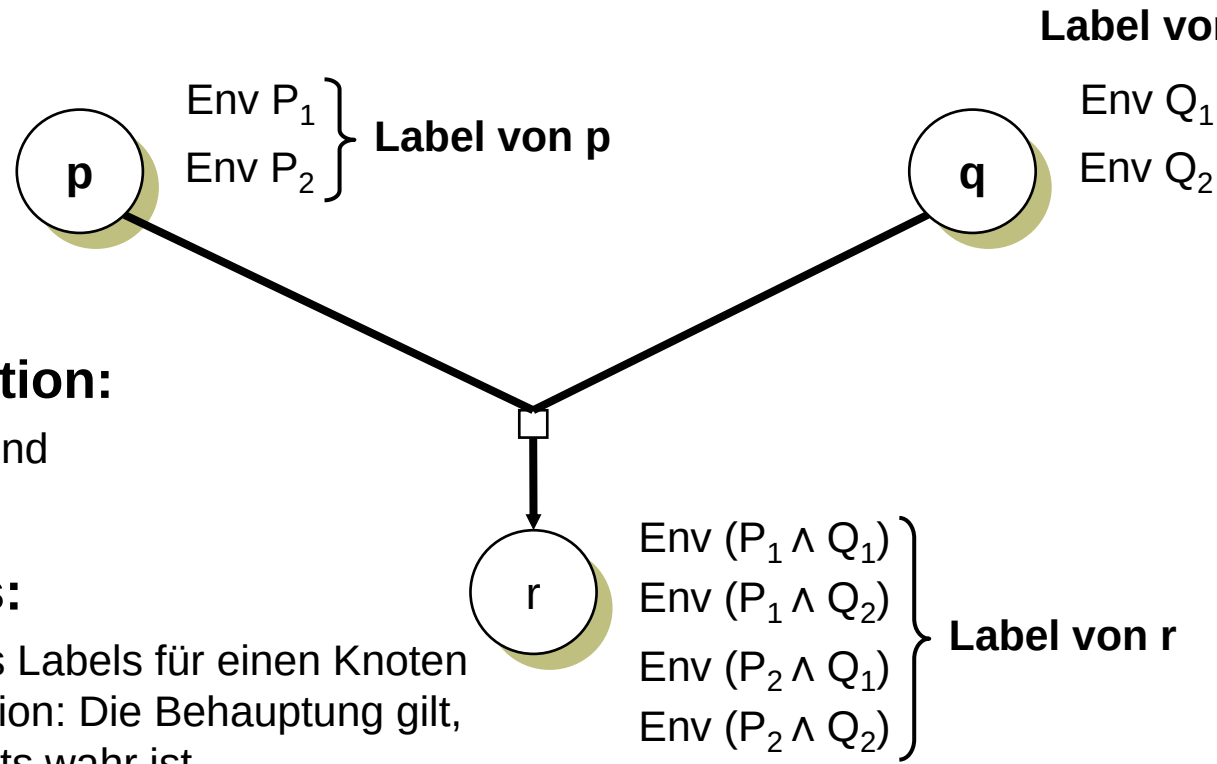
C: Komponente 3 ist ok

w: p_3 = Masse

D: Komponente 7 ist ok

x: p_4 = Masse

Labelaktualisierung in einem ATMS



Bedeutung der Justification:

- r gilt, wenn p und q wahr sind (Konjunktion)

Bedeutung eines Labels:

- Mehrere Environments des Labels für einen Knoten entsprechen einer Disjunktion: Die Behauptung gilt, wenn eine der Environments wahr ist

Eliminierung überflüssiger Environments:

- Widersprüchliche Environments können weggelassen werden.
- Damit können auch alle Environments weggelassen werden, die Konflikte enthalten.
- Environments, die aus anderen Environments desselben Labels folgen, können weggelassen werden.

Anwendung eines ATMS durch den Problemlöser

Eingabe vom Problemlöser:

- Annahmeknoten
- “Normale” Knoten
- Justifications zwischen den Knoten

Ausgabe an den Problemlöser:

- Menge der minimalen Konflikte

Das ATMS macht automatisch:

Das sind sehr viele Operationen !

- Erzeugung der Labels für die Annahmeknoten
- Aktualisierung der Labels aller Conclusions, von denen sich der Label eines Antecedents geändert hat
- Eliminierung aller überflüssigen Environments

Anwendung eines ATMS durch den Problemlöser

Eingabe vom Problemlöser:

- Annahmeknoten
- “Normale” Knoten
- Justifications zwischen den Knoten

Ausgabe an den Problemlöser:

- Menge der minimalen Konflikte

Zusammenspiel mit dem Kandidatengenerierer:

- Bilde alle Annahmeknoten für die Fokusdiagnosen
- Berechne alle Werte, die sich aus den Annahmen der Fokusdiagnosen ergeben, und bilde die entsprechenden Behauptungsknoten und Justifications
- Entnimm dem ATMS die neuen Konflikte

Optimierung 1: Focusing ATMS

Eigenschaft, die verbessert werden soll:

- Das ATMS berechnet ***alle*** minimalen Konflikte, die zu einer beliebigen Kombination von Annahmen gelten.
- Die meisten davon sind für den Problemlöser gar nicht von Interesse.

Daher zusätzliche Eingabe vom Problemlöser:

- Diejenigen Environments, die von Interesse sind (Fokusenvironments)

Ausgabe:

- Menge der minimalen Konflikte, die in einem Fokusenvironment enthalten sind

Geänderte Funktionsweise des ATMS:

- Environments werden nur gebildet, wenn sie Teilmengen eines Fokusenvironments sind
- Bei Eingabe eines neuen Fokusenvironments werden automatisch alle Labels aktualisiert

Optimierung 2: Lazy ATMS

Eigenschaft, die verbessert werden soll:

- Das ATMS muss seine Labels häufig aktualisieren
- Viele Aktualisierungen werden nach außen gar nicht sichtbar, bevor sie von neuem überschrieben werden.

Geänderte Funktionsweise des ATMS:

- Labels werden erst berechnet, wenn nach ihnen explizit gefragt wird (in der Regel wird vor allem nach dem Label des Widerspruchsknotens gefragt)

In MDS (DaimlerChrysler) realisierte Variante:

- ATMS, das gleichzeitig fokussiert und lazy arbeitet

Beschreibung in der Dissertation von Mugur Tatar

Wertpropagierung und ATMS

Was versteht man unter Propagierung im MDS-Kontext ?

- Propagierung ist die Weiterleitung von Informationen über ein Netzwerk aus Kanten und Knoten.

Trennung von Wertpropagierung und ATMS:

- Das **ATMS** ist verantwortlich für die Propagierung der Environments in einem gegebenen Netzwerk von Wertabhängigkeiten.
- Das Netzwerk von Wertabhängigkeiten wird in einem Rule Propagator (**RP**) hergestellt, der die Justifications aus den Regeln für die Verhaltensmodi der Komponenten zusammensetzt.
- Der RP ist genauso faul wie sein ATMS:
  - Werte werden nur weiterpropagiert, wenn sie von Environments unterstützt werden, die im gegenwärtigen Fokus stehen.
  - Das ATMS teilt dem RP unaufgefordert mit, welche Werte neu von Fokusenvironments unterstützt werden und initiiert die **(Propagation) Tasks**

Wertpropagierung und ATMS

Was bringt die Trennung von Wertpropagierung und ATMS ?

1. Antwort: Bessere Softwarearchitektur durch Modularisierung

- **Werte** entstehen meistens aus Beobachtungen (Messungen) und gezielten Eingaben. Diese sind spärlich, daher gibt es **nicht viele** resultierende Werte.

- **Environments** entstehen aus Annahmen über Verhaltensmodi. Von diesen gibt es **sehr viele** (selbst bei Einfachfehlern mindestens so viele wie Komponenten).

- Daher werden Fokusenvironments häufiger revidiert, als neue Werte berechnet werden. Diese Revision kann dann als ein ATMS-internes Problem behandelt werden.

Anm.: Die Aufteilung in ein RP- und ATMS-Modul fördert enge Modulbindung und lose Modulkopplung

Wertpropagierung und ATMS

Was bringt die Trennung von Wertpropagierung und ATMS ?

2. Antwort: Einsatz des ATMS für erweiterte Aufgaben

- Es können auch andere Annahmen als Verhaltensmodi für Komponenten untersucht werden:

  Beispiele:

  - Annahmen über Werteingaben (control inputs)

    (für die Berechnung sinnvoller Testsituationen)

  - Annahmen über Komponentenzustände (bei dynamischen Komponenten)

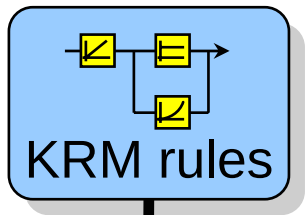
    (für dynamische Komponenten, deren Zustand unbekannt ist)

  - Annahmen über beliebige andere Werte

    (könnte für Beobachtungspunkte interessant sein)

Wertpropagierung und ATMS

Wissensbasis
(Komponentenmodellierung
plus Systemzusammenhang)



KRM:
Knowledge **R**epresentation **M**anager