

Grundlagen der Künstlichen Intelligenz

Sebastian Iwanowski
FH Wedel

Kap. 5:
Technische Diagnose

5.2: MDS: Funktionsweise und SW-Aufbau (Überblick)

Modellbasierte Diagnose

Begriffswelt GDE:

Komponente:

Einheit, deren Verhalten diagnostiziert werden soll
üblicherweise nummeriert von 1 bis n

Komponententyp:

fasst Komponenten gleichartigen Verhaltens zusammen

Verhaltensmodus:

Einem Komponententyp zugeordnete Verhaltensbeschreibung
üblicherweise nummeriert von 1 bis k:

1 steht für ok

2 bis k sind die Fehlermodi (geordnet nach Wahrscheinlichkeit)

(Diagnose-)Kandidat:

Zuweisung von genau einem Verhaltensmodus an jede Komponente des Systems

Modellbasierte Diagnose

Begriffswelt GDE:

Kandidat:

(2 1 3 1 1 2 1) bedeutet: Komponente Nr. 1 ist in Verhaltensmodus 2
Komponente Nr. 2 ist in Verhaltensmodus 1
Komponente Nr. 3 ist in Verhaltensmodus 3
Komponente Nr. 4 ist in Verhaltensmodus 1
Komponente Nr. 5 ist in Verhaltensmodus 1
Komponente Nr. 6 ist in Verhaltensmodus 2
Komponente Nr. 7 ist in Verhaltensmodus 1

Konflikt:

Zuweisung von genau einem Verhaltensmodus an einige Komponenten des Systems

(0 1 0 0 0 2 0) bedeutet: Komponente Nr. 2 ist in Verhaltensmodus 1
Komponente Nr. 6 ist in Verhaltensmodus 2
über die anderen Komponenten wird keine Aussage gemacht

Interpretation: Es ist unvereinbar, dass sich Komponente 2 in Verhaltensmodus 1 und Komponente Nr. 6 in Verhaltensmodus 2 befindet.

Modellbasierte Diagnose

Begriffswelt GDE:

Diagnose (= konsistenter Kandidat):

Kandidat, der keinen Konflikt enthält

Beispiele: $(2\ 1\ 3\ 1\ 1\ 2\ 1)$ enthält den Konflikt $(0\ 1\ 0\ 0\ 0\ 2\ 0)$, ist also keine Diagnose

Wenn $(0\ 1\ 0\ 0\ 0\ 2\ 0)$ der einzige Konflikt ist, ist $(1\ 1\ 1\ 1\ 1\ 1\ 1)$ eine Diagnose

Wenn $(0\ 1\ 0\ 0\ 0\ 2\ 0)$ und $(1\ 1\ 0\ 0\ 0\ 0\ 0)$ die Konflikte sind, ist $(1\ 2\ 1\ 1\ 1\ 1\ 1)$ eine Diagnose

Präferenz zwischen Kandidaten:

Ein Kandidat A ist einem anderen Kandidaten B präferiert, wenn A für jede Komponente maximal den Verhaltensmodus von B zuweist.

Beispiel: $(1\ 1\ 1\ 1\ 1\ 1\ 1)$ ist präferiert zu $(1\ 2\ 1\ 1\ 1\ 1\ 1)$

Präferierte Diagnose:

Eine Diagnose ist präferiert, wenn alle ihr präferierten Kandidaten Konflikte enthalten, sie also bezüglich der Präferenz maximal ist.

Beispiel: Wenn $(0\ 1\ 0\ 0\ 0\ 2\ 0)$ und $(1\ 1\ 0\ 0\ 0\ 0\ 0)$ die Konflikte sind, sind $(1\ 2\ 1\ 1\ 1\ 1\ 1)$ und $(2\ 1\ 1\ 1\ 1\ 1\ 1)$ die beiden einzigen präferierten Diagnosen.

Modellbasierte Diagnose

Ziel von MDS (Daimler-Weiterentwicklung der GDE):

Basisfunktionalität Diagnosefindung:

- 1) Finde die wahrscheinlichsten präferierten Diagnosen !**
(aus Komplexitätsgründen wird die Stückzahl stark begrenzt)

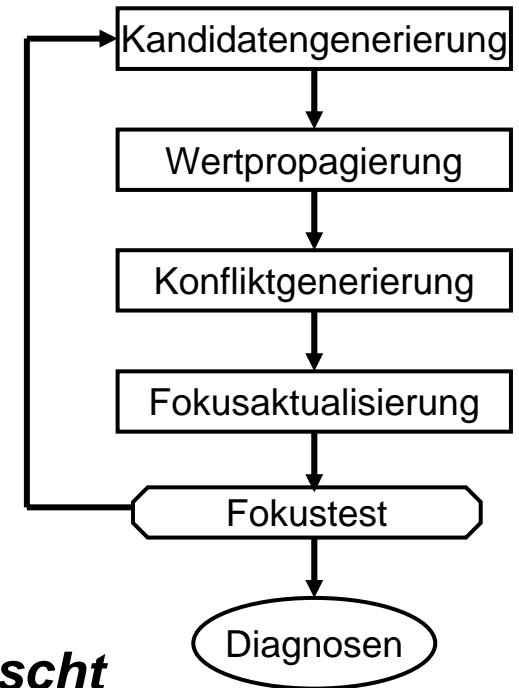
Erweiterte Funktionalität Reparaturanleitung:

- 2) Schlage Aktionen und Tests vor, um die möglichen Diagnosen weiter einzuschränken !**

Modellbasierte Diagnose

Algorithmus zum Finden der wahrscheinlichsten präferierten Diagnosen (**Aufgabenstellung 1**):

1. Nimm Kandidaten in den Fokus auf.
2. Generiere und propagiere alle Werte, die sich aus den Verhaltensmodi der Kandidaten im Fokus ergeben.
3. Finde die minimalen Konflikte aus den propagierten Werten.
4. Schließe die Kandidaten aus, die Konflikte enthalten.
5. Falls Fokus noch genügend groß, dann Ziel erreicht, anderenfalls weiter bei 1.



In der Realisierung werden die Schritte 1 bis 4 vermischt
(erreicht durch ereignisorientierte Programmierung)

Im Folgenden werden die Verfahren für die **Kandidatengenerierung** und **Konfliktgenerierung** getrennt beschrieben.

MDS: Kandidatengenerierung

INPUT:

- **Alte Konflikte und die für diese Konflikte präferierten und konsistenten Kandidaten**
- **Neue Konflikte**

OUTPUT:

- **Menge der präferierten Kandidaten, die auch für die neuen Konflikte konsistent sind**

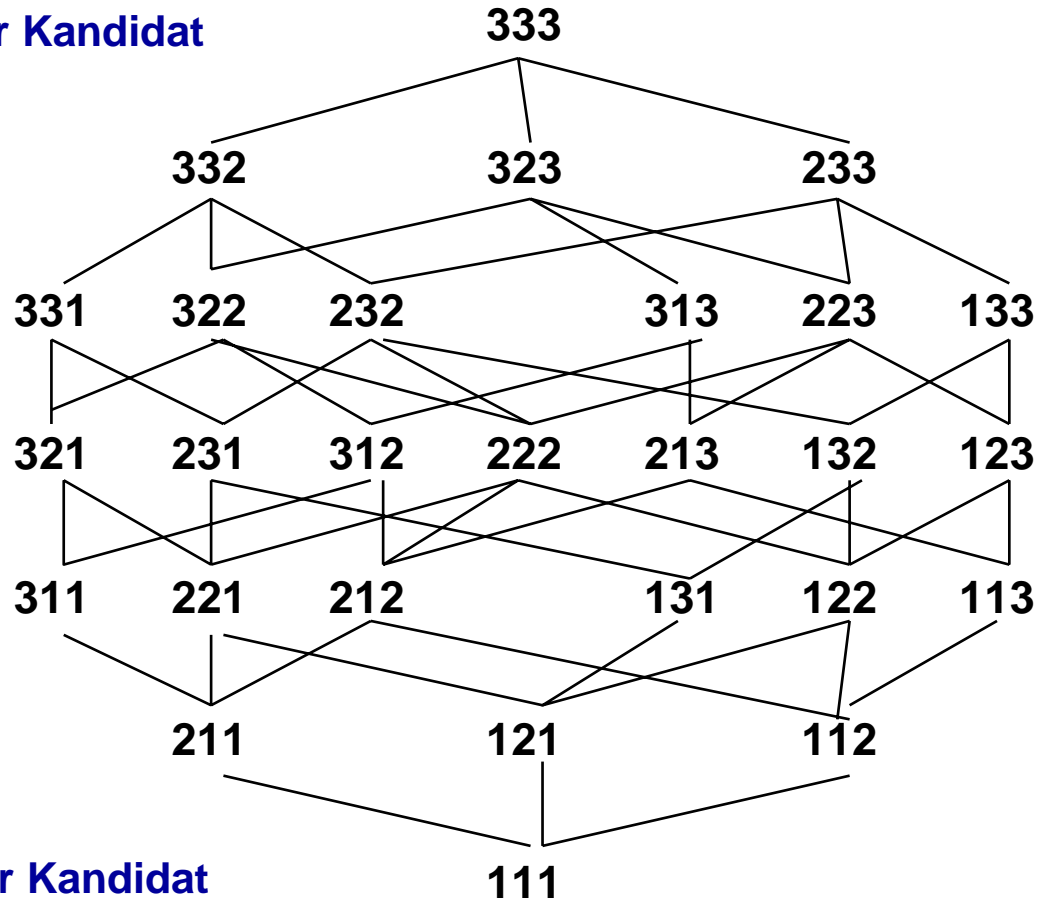
Einbettung der Kandidatengenerierung in den Diagnoseprozess:

- Output der Kandidatengenerierung wird in den Fokus genommen
- Durch Wertepropagierung werden neue Konflikte gefunden
- Diese Konflikte werden als Input für eine neue Runde der Kandidatengenerierung genommen
- Wenn keine neuen Konflikte gefunden werden, ist der Diagnoseprozess beendet.

MDS: Die Kandidaten im Präferenznetz

Beispiel: 3 Komponenten
Für jede Komponente 3 Verhaltensmodi

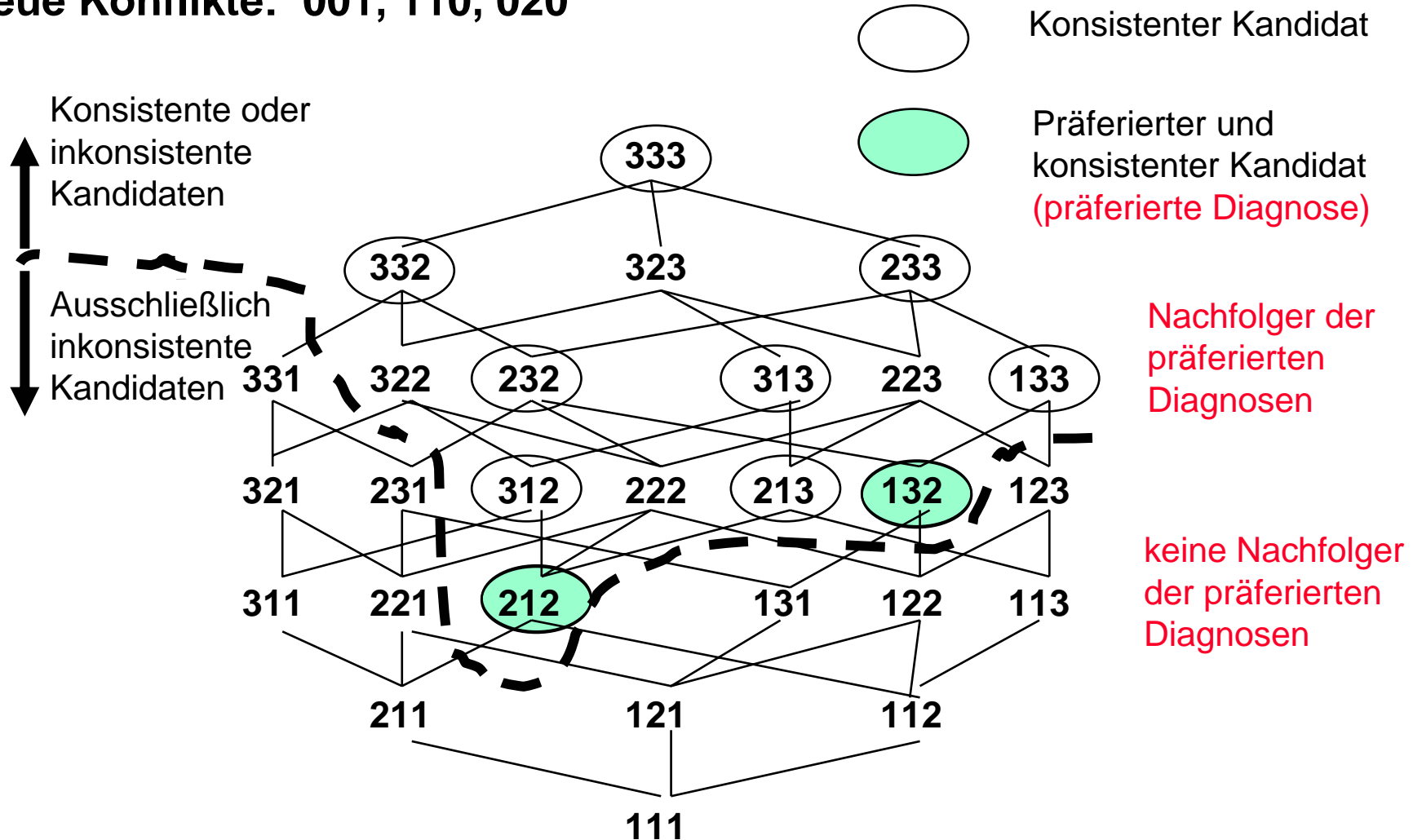
minimal präferierter Kandidat



maximal präferierter Kandidat

MDS: Die Kandidaten im Präferenznetz



Neue Konflikte: 001, 110, 020

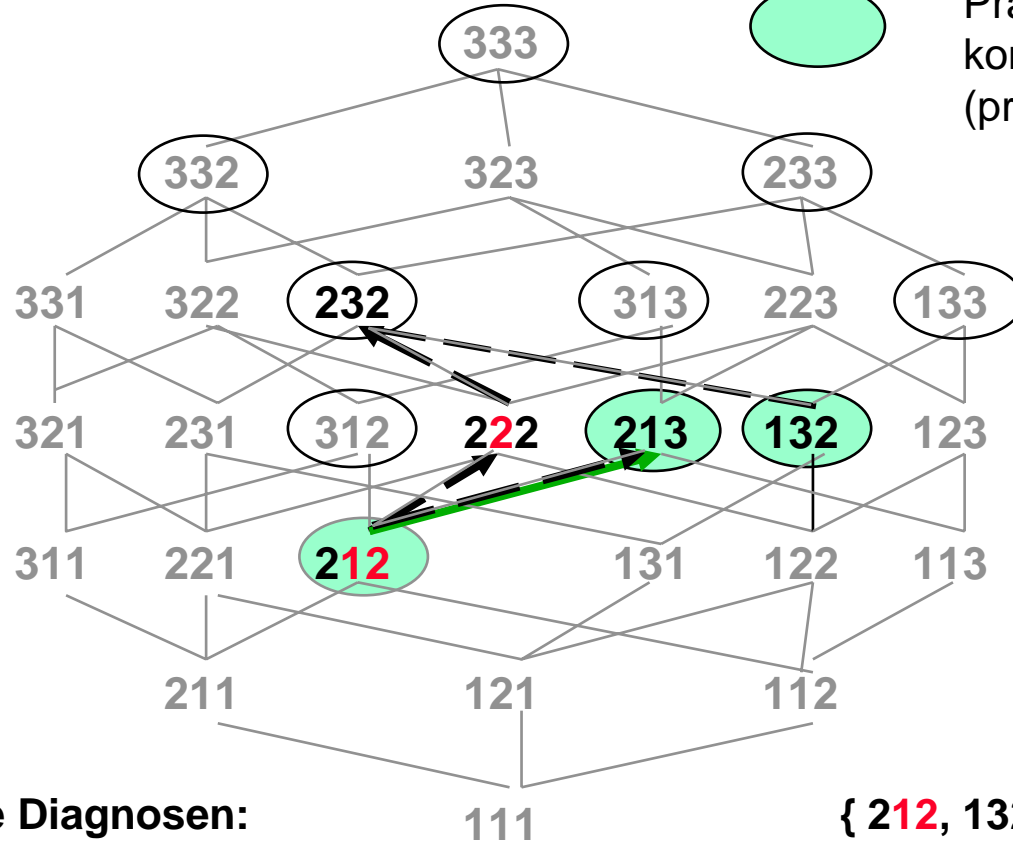


MDS: Kandidatenaktualisierung

Alte Konflikte: 001, 110, 020

Neuer Konflikt: 012

-  Konsistenter Kandidat
-  Präferierter und konsistenter Kandidat (präferierte Diagnose)



Bisherige präferierte Diagnosen:

{ 212, 132 }

Neue präferierte Diagnosen (Stufe 1):

{ 222, 213, 132 }

Neue präferierte Diagnosen (Stufe 2):

{ 213, 132 }

MDS: Kandidatenaktualisierung

Aktionen bei Entdeckung eines neuen Konflikts:

- 1) Konsistenzcheck aller präferierten Diagnosen
- 2) Entfernen aller jetzt als inkonsistent erkannten Kandidaten
- 3) Bilden der Präferenznachfolger jedes eben entfernten Kandidaten
- 4) Aufnehmen der Präferenznachfolger, die folgende Bedingung erfüllen:
 - Der Nachfolger ist nicht von einer anderen noch als konsistent erkannten Diagnose präferiert.
 - Der Nachfolger ist selbst konsistent.

MDS: Kandidatenaktualisierung

Aktionen bei Entdeckung eines neuen Konflikts:

3) Bilden der Präferenznachfolger jedes eben entfernten Kandidaten:

- Wenn C der Konflikt ist, der in der alten Diagnose enthalten ist, dann Bilden nur der Nachfolger, die den Verhaltensmodus **genau einer Komponente** ändern, **die in C enthalten** ist

(Bildung nur von direkten Nachfolgern, Nachfolgebildung bzgl. Konflikt C)

Anmerkung: Auf diese Weise geht keine Diagnose verloren:

Satz: Jede Nachfolgediagnose, die C nicht enthält, ist Nachfolgediagnose eines direkten Nachfolgers, der C nicht enthält.

- Wenn einer dieser direkten Nachfolger einen Konflikt C' enthält, dann Bildung weiterer direkter Nachfolger bzgl. C'

MDS: 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
- Werte entstehen durch Systembeobachtungen und deren Weiterleitung im Netz unter Berücksichtigung von Verhaltensmodusannahmen.
- Annahmen, die widersprüchliche Werte vorhersagen, sind Konflikte

Modulare SW-Architektur für Propagierung

Was versteht man unter Propagierung im MDS-Kontext ?

- Propagierung ist die Weiterleitung von Informationen über ein Netzwerk aus Kanten und Knoten.
- Weitergeleitete Informationen sind Werte sowie die sie unterstützenden Verhaltensannahmen.

Modulare SW-Architektur:

Getrennte Propagierung von Werten und Verhaltensannahmen:

- Das **ATMS** ist verantwortlich für die Propagierung der Verhaltensannahmen in einem gegebenen Netzwerk von Wertabhängigkeiten.
- Das Netzwerk von Wertabhängigkeiten wird in einem Rule Propagator (**RP**) hergestellt, der die Abhängigkeiten konkreter Werte aus den Regeln für die Verhaltensmodi der Komponenten zusammensetzt.

Modulare SW-Architektur für Propagierung

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.
- Es gibt **sehr viele** Verhaltensannahmen (selbst bei Einfachfehlern mindestens so viele wie Komponenten).
- Verhaltensannahmen werden viel 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

Modulare SW-Architektur für Propagierung

Was bringt die Trennung von Wertpropagierung und ATMS ?

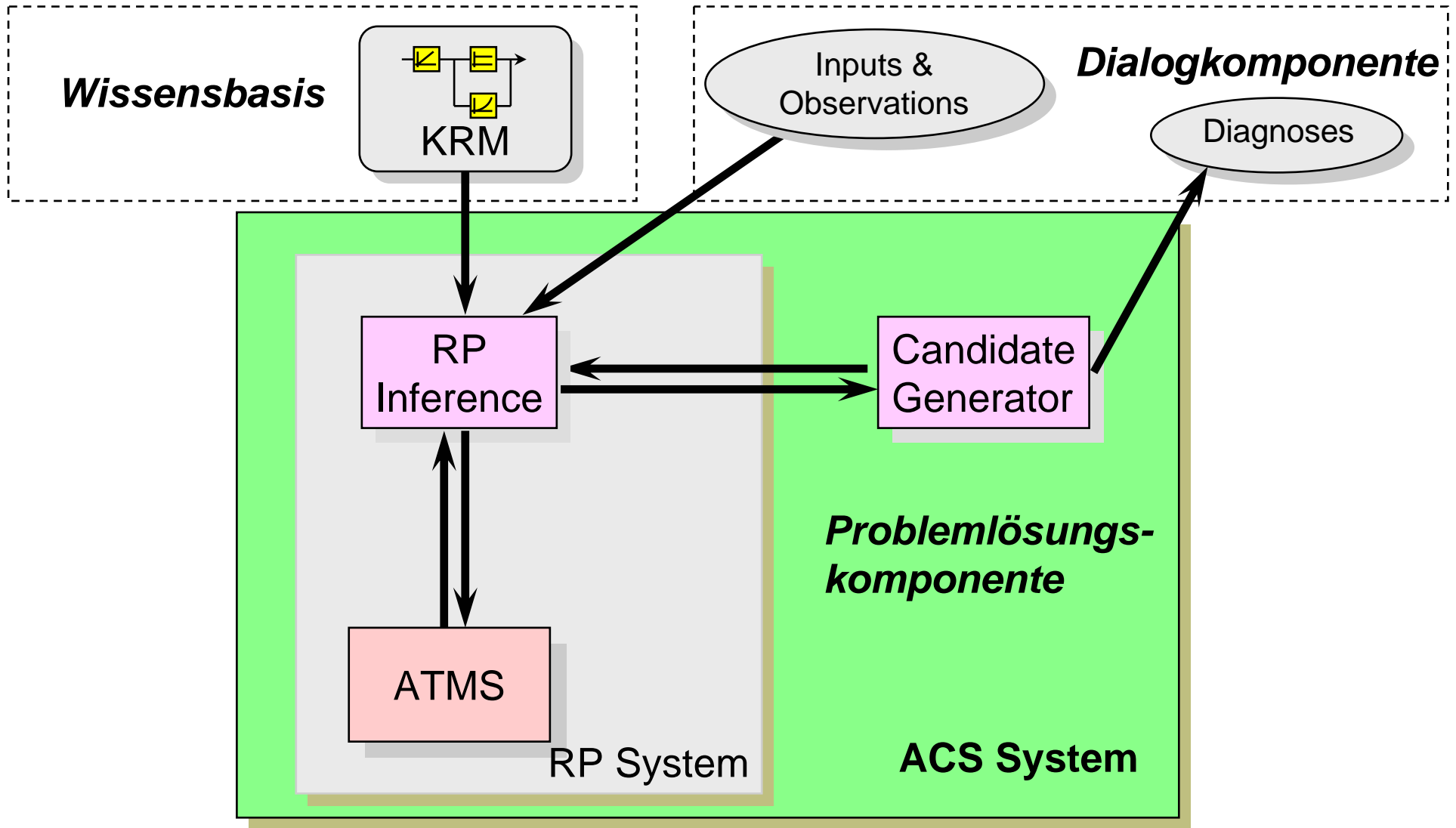
2. Antwort: Einsatz des ATMS für erweiterte Aufgaben

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

Beispiele:

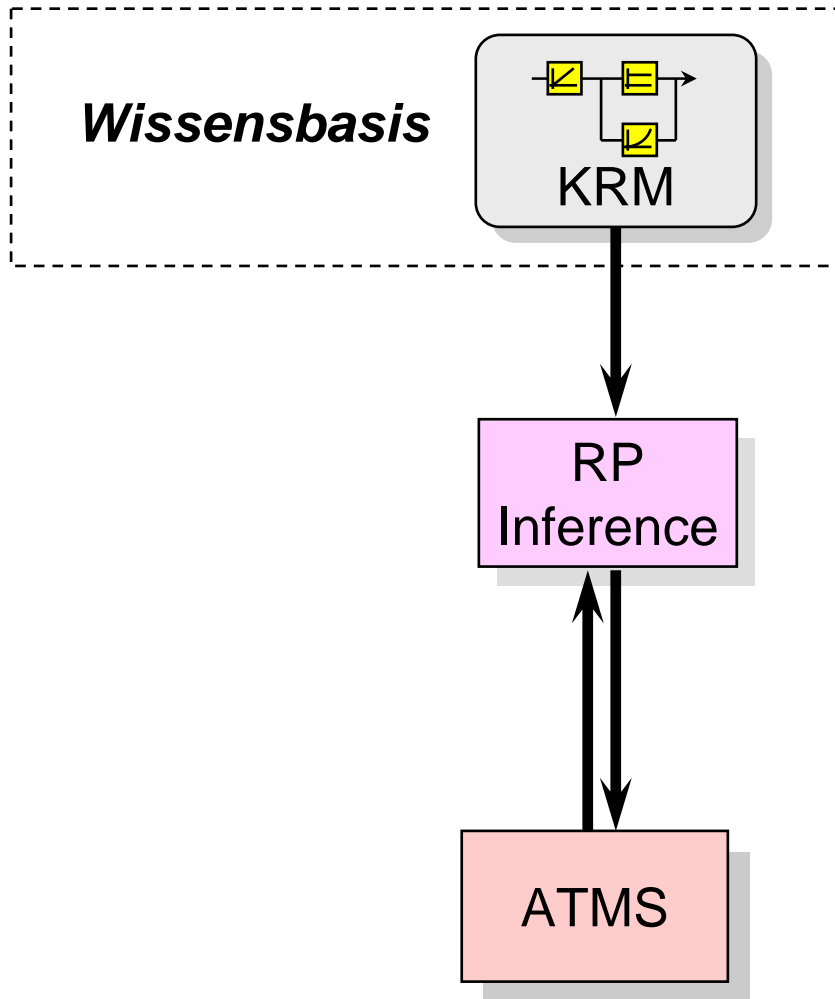
- Annahmen über Werteingaben (control inputs)
(für die Berechnung sinnvoller Testsituationen)
- Annahmen über Komponentenzustände (bei dynamischen Komponenten)
(für dynamische Komponenten, deren Zustand unbekannt ist)
- Annahmen über beliebige andere Werte
(könnte für Beobachtungspunkte interessant sein)

MDS (Basisfunktionalität): SW-Architektur



ACS: Assumption-based Constraint Solver

Anforderung an die Wissensbasis



Was muss die Wissensbasis an die Inferenzkomponente liefern ?

- Regeln für die Wertzusammenhänge in den einzelnen Verhaltensmodi (*Komponentenmodellierung*)
- Kenntnis über die Wertdomänen: Wann gelten zwei Werte als widersprüchlich ?

MDS löst diese Anforderungen durch das Anbieten einer Constraint-Sprache für die Komponentenmodellierung

Vom ACS zur kompletten Diagnosesoftware

Was kann das ACS für den Anwender leisten?

Eingabe:

- Einstellung bestimmter Werte im System
- Beobachtung davon abhängiger Werte im System

Ausgabe:

- Mehrere Diagnosen folgender Art:
 - Jede Diagnose weist jeder Komponente einen Verhaltensmodus zu: entweder ok oder ein definierter Fehlermodus
 - Die Regeln aller zugewiesenen Verhaltensmodi sind konsistent (mit allen eingestellten und beobachteten Werten)

Was braucht der Anwender ?

Eingabe: s.o.

Ausgabe:

- Eine eindeutige Anweisung, welche Komponenten wie repariert werden sollen

Vom ACS zur kompletten Diagnosesoftware

Was fehlt also noch ?

1) Vorschlag von Testeinstellungen (control inputs)

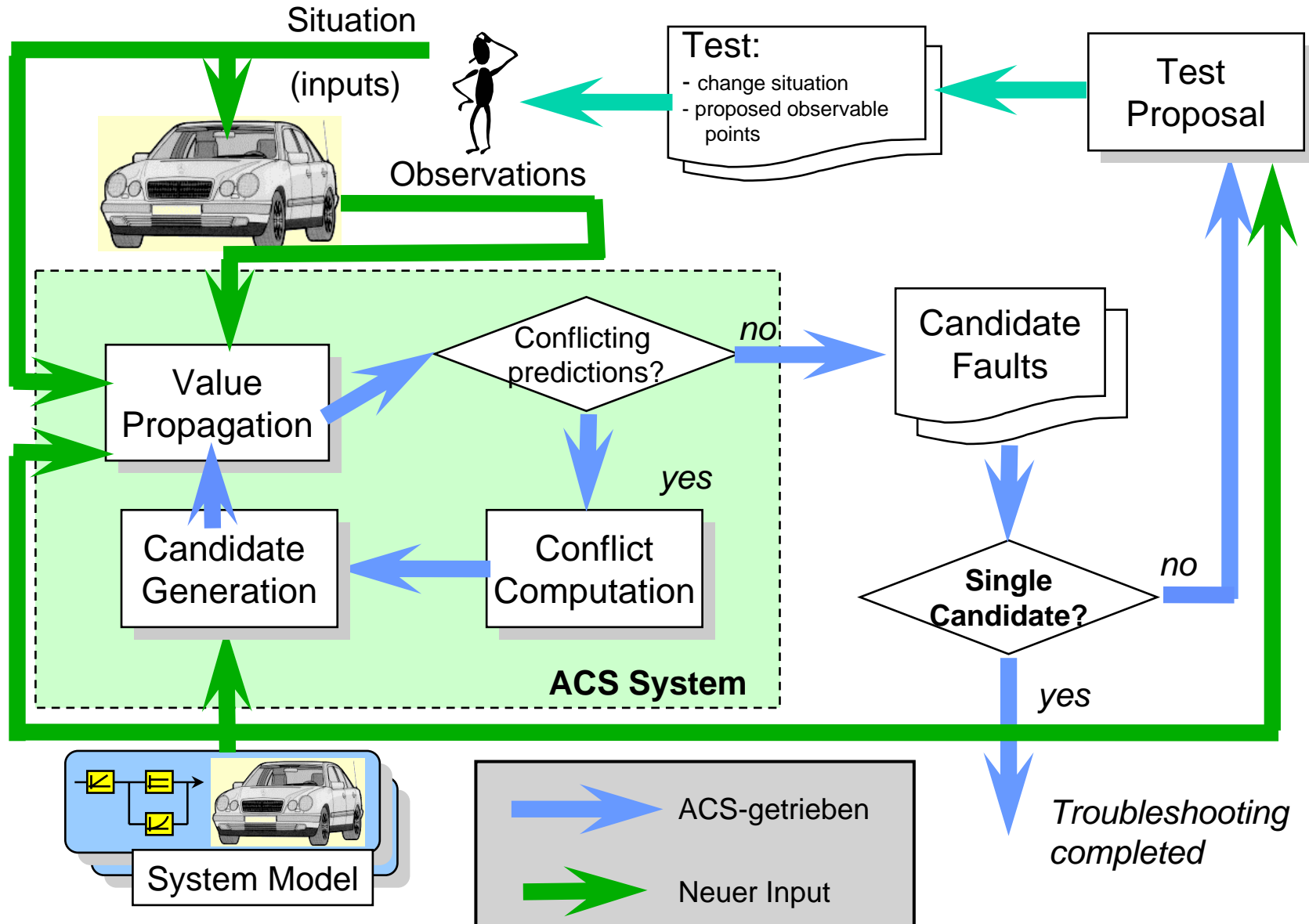
- Einstellung bestimmter Werte an bestimmten Stellen im System
(derart, dass die zu erwartenden Beobachtungen die bisher gültigen Diagnosen bestmöglich unterscheiden)

2) Vorschlag von Beobachtungspunkten

- Auswahl von Messstellen im System
(derart, dass die zu erwartenden Beobachtungen die bisher gültigen Diagnosen bestmöglich unterscheiden)

Test

Vom ACS zur kompletten Diagnosesoftware



Vom ACS zur kompletten Diagnosesoftware

Zusammenfassung: Diagnose mit MDS

- MDS schlägt dem Benutzer systematisch Einstellungen im technischen System vor und gibt vor, wo er Werte messen soll.
- Dieser Vorgang wird solange wiederholt, bis eine eindeutige Diagnose mit hinreichend genauer Wahrscheinlichkeit gegeben werden kann.
- Die Diagnose gibt dem Benutzer explizit Hinweise, welche Komponenten defekt sind und wie er den Fehler beheben kann (wegen der Angabe des konkreten Fehlermodus)
- In der Wissensbasis kann für jeden Fehlermodus eine konkrete Abhilfemaßnahme hinterlegt werden.
 - ➔ Die Ausgabe von MDS ist eine konkrete Reparaturanleitung

Model based troubleshooting