

Kapitel 12

Modellbasierte Systeme und qualitative Modellierung

Peter Struss

12.1 Einleitung

Modellbasierte Systeme stellen eine Klasse wissensbasierter Systeme dar, deren Anwendungspotential sich derzeit vor allem in industriellen Einsatzbereichen zunehmend entfaltet. Sie stützen sich also auf eine Wissensbasis 5, auf der ein Inferenzmechanismus 5.4 operiert, um ein bestimmtes Problem zu lösen. Worin besteht die Besonderheit modellbasierter Systeme, und warum stellen sie einen eigenen Forschungsbereich dar? Sie entwickeln und spezialisieren das allgemeine Prinzip der Trennung zwischen (deklarativem) Wissen und Inferenzmaschine, indem sie explizit zwischen dem aufgabenspezifischen Problemlösewissen und dem Wissen über den Gegenstandsbereich, also dem Modell, differenzieren.

Um dies an einem Beispiel zu illustrieren: ein traditionelles, regelbasiertes Diagnose-Expertensystem, etwa für ein Kraftfahrzeug, würde als Wissensbasis eine Menge von Assoziationen zwischen Symptomen und ihren möglichen Fehlerursachen als Regeln enthalten und darauf einen Regelinterpreter als allgemeinen Inferenzmechanismus anwenden. In Regeln wie

WENN Motor_springt_nicht_an
DANN Mögliche_Ursache_Batterie_leer
WENN Scheinwerfer_leuchten
DANN NICHT(Batterie_leer)

spiegelt sich Wissen wider, das sowohl aufgabenorientiert (nämlich diagnosebezogen) ist, als auch spezifisch für den Gegenstand der Aufgabenstellung, eine bestimmte Klasse von Fahrzeugen. Letzteres wird relevant, wenn man Fahrzeuge betrachtet, die über eine separate Batterie für das Antriebssystem verfügen. Weil bei ihnen trotz funktionierender Scheinwerfer das Fahrzeug durchaus wegen einer defekten Batterie nicht anspringen könnte, müßte die zweite Regel geändert werden. Dem liegt zugrunde, daß die Auswirkungen einzelner (Komponenten-)Fehler von Struktur und Verhalten des Gesamtsystems abhängen, was in die Regeln „hineincodiert“ ist.

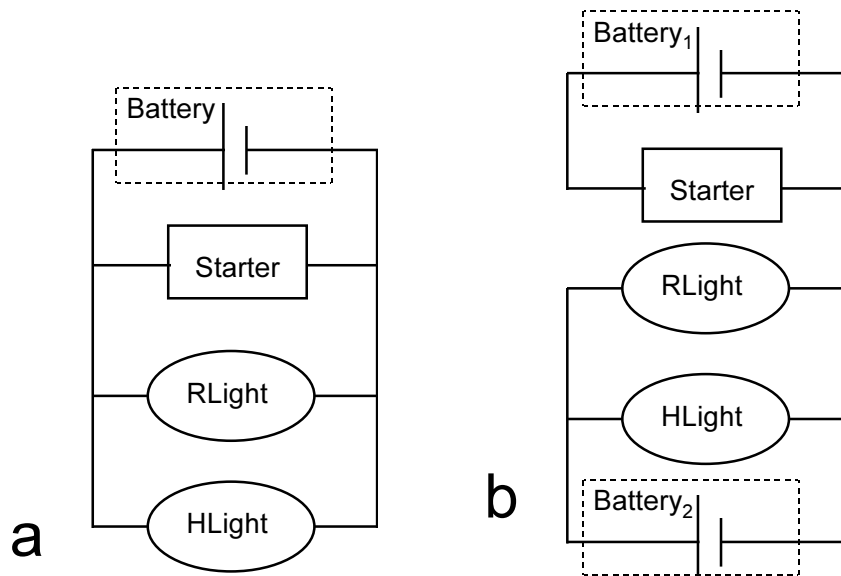


Abbildung 12.1: Anlasser und Lichtanlage mit ein und zwei Batterien

Modellbasierte Systeme repräsentieren dieses Wissen explizit und separat vom aufgabenspezifischen Wissen, eben in einem Modell. Ein Inferenzmechanismus, der „Prädiktor“, leitet aus diesem Modell Aussagen über das Systemverhalten in verschiedenen Situationen ab, etwa daß bei einer Schaltung wie in Abb. 12.1a eine betriebsbereite Batterie zusammen mit intakten Leitungen und einem funktionierenden Anlasser den Motor anspringen läßt. Neben dem allgemeinen, nicht diagnosebezogenen Prädiktor (z.B. einem Constraint-Solver) existiert ein diagnosespezifischer Inferenzmechanismus. Dieser wird z.B. aufgrund des Prädiktorergebnisses und der Beobachtung *Motor_springt_nicht_an* Batterie, Leitungen und Anlasser als mögliche Diagnosehypothesen generieren, diese ggf. durch den Prädiktor überprüfen, etc. Dabei stützt er sich evtl. auf weiteres, diagnosespezifisches Wissen, etwa bestimmte Teststrategien. Verglichen mit dem regelbasierten System werden also im modellbasierten Ansatz

- Wissen über Gegenstand und Aufgabe getrennt und
- spezifischere und aufgabenbezogene Inferenzmechanismen verwendet (welche sich allerdings oft durch allgemeinere implementieren lassen).

Als Folge wird ein so organisiertes System ein Fahrzeug mit zwei Batterien wie in Abb. 12.1b korrekt diagnostizieren, da der strukturelle Unterschied im Modell explizit gemacht und vom Prädiktor berücksichtigt wird. Da hier im dargestellten Wissen auf die grundlegenden (physikalischen) Prinzipien zurück gegangen wird, ist modellbasiertes Schließen auch als „reasoning from first principles“ charakterisiert in Abgrenzung zum empiriegestützten regelbasierten oder fallbasierten Schließen 11.

Modellbasierte Systeme müssen sich also auf eine

- *systematische, strukturierte Darstellung von Wissen über einen komplexen Gegenstand(sbereich), aus der Aussagen über das Verhalten des Gegenstands ableitbar sind,*

stützen und damit auf Lösungen zu fundamentalen Aufgabenstellungen der Künstlichen Intelligenz, nämlich Wissen über die (natürliche oder technische) physikalische Welt

zu formalisieren. Damit wird auch deutlich, daß die entsprechenden Modelle und Systeme über die gebräuchlichen (etwa numerische Simulationsmodelle) und darauf gegründete Ansätze im industriellen Einsatz oder in den Ingenieurwissenschaften hinausgehen müssen. Letztere beschränken sich zumeist auf die Nutzung eines rein mathematischen (oft geschlossenen) Verhaltensmodells im Computer, während modellbasierte KI-Systeme darauf abzielen, zusätzlich eine *konzeptuelle Sicht* auf den Gegenstand, seine physikalischen Konstituenten und seine Struktur, kausale Beziehungen, Modellierungsannahmen etc. zu repräsentieren, um darauf Verfahren des automatischen Schließens anwenden zu können. In der Regel sind diese Modelle *kompositional* und heben ausdrücklich auf die Unterstützung des Modellierungsschritts ab. Ferner handelt es sich oft um *qualitative* Modelle, die sowohl kognitiven Aspekten Rechnung tragen, als auch einen anwendungsbezogenen Abstraktionsgrad aufweisen. „Zu geringe“ Batteriespannung zum Beispiel liefert eine intuitive Erklärung für „Motor springt nicht an“ (im Gegensatz zu „ $U < 1.4 \text{ V}$ “), reflektiert aber auch, daß eine ausreichende Minimalspannung situationsabhängig und damit nicht eindeutig numerisch zu definieren ist. Diese und weitere Charakteristika werden im Abschnitt 12.3 diskutiert.

Technische Anwendungsbereiche werden in unserer Darstellung als Beispiele im Vordergrund stehen, auch weil die Technologie hierfür am ausgereiftesten ist. Gemäß der Prinzipien modellbasierter Systeme muß dieser Beitrag zum einen Grundlagen und Techniken der Modellierung und des modellbasierten Schließens als auch die Verwendung der Modelle in aufgabenspezifischen Problemlösern darstellen. Im folgenden Abschnitt wird daher zunächst eine Systematisierung und z.T. eine formale Charakterisierung von aktuellen und möglichen Aufgabenstellungen modellbasierter Systeme vorgenommen, um einen Hintergrund für die folgende Darstellung von Formalismen zur Modellierung und Verhaltensvorhersage bereitzustellen. Abschnitt 12.4 präsentiert dann Theorien und Verfahren modellbasierter Diagnose, dem Anwendungsgebiet mit der gegenwärtig größten praktischen Bedeutung. Es folgen Überblicke über weitere Anwendungsfelder, Quellen und Werkzeuge sowie Perspektiven und wichtige Forschungsthemen des Gebietes.

12.2 Anwendungsaufgaben

Ein *Modell* eines technischen oder natürlichen Systems repräsentiert das Wissen über bestimmte Aspekte des Verhaltens dieses Systems in einer formalen Sprache. Weil es immer nur einen Teil des Systemverhaltens widerspiegelt und eine Abstraktion ist, steht es genau genommen für eine *Klasse* von Systemen. Es kann als „Stellvertreter“ dieser Systemklasse zur Lösung von Problemen benutzt werden, weil und insoweit es gestattet, mit Hilfe eines Ableitungsmechanismus neue Aussagen über das tatsächliche Systemverhalten zu gewinnen.

Eine Datenbank von Produktbeschreibungen, der man lediglich die gespeicherten Parameter und Merkmale wieder entnehmen kann, betrachten wir ebensowenig als Modell wie einen Satz von Daten, die bei einem Experiment mit einem System erhoben wurden. Ein Modell ist nie allein durch eine formalsprachliche Darstellung gegeben, sondern nur in Verbindung mit einem Mechanismus, der auf dieser Darstellung operiert. Eine Gleichung, die das Ohmsche Gesetz darstellt, ist bloß eine Zeichenfolge ohne den Algorithmus, etwa einen Gleichungslöser, den wir entweder im Kopf oder auf dem Rechner anwenden.

Wenn wir im folgenden von einem Modell sprechen, schließt dies meist beides ein. Nur in Ausnahmen wird man ein Modell in eine extensionale Darstellung aller möglichen Ableitungen kompilieren können.

Modelle können bei vielfältigen Problemlöseprozessen während des Lebenszyklus technischer Produkte angewendet werden, und dies geschieht auch in den derzeitigen Arbeitsprozessen, wenn auch oft unbewußt, implizit oder informell. Es sollen kurz einige dieser Aufgaben diskutiert und die Verwendung von Modellen formal charakterisiert werden, wobei wir keine besonderen Voraussetzungen über deren Form und Inhalt machen. Wir benutzen eine logische Notation für diese Charakterisierung. Im Einzelfall kann aber (und wird meist auch) ein anderes, etwa mathematisches Ableitungssystem benutzt werden.

12.2.1 Entwurf

Der Entwurf als erster Schritt im Lebenszyklus zielt auf eine Beschreibung eines zu fertigenden Systems, das eine gewisse Spezifikation erfüllt. Diese umschließt eine Bestimmung des gewünschten Systemverhaltens, im folgenden durch „GOALS“ repräsentiert. Das Entwurfsziel läßt sich dann charakterisieren als eine Systembeschreibung, deren Modell die Spezifikation des Verhaltens erfüllt. Logisch ausgedrückt, soll das Modell, MODEL, das Zielverhalten, GOALS, ableiten:

$$\text{MODEL} \vdash \text{GOALS} .$$

Eine notwendige Voraussetzung ist, daß das Modell zumindest konsistent mit dem Zielverhalten ist:

$$\text{MODEL} \cup \text{GOALS} \not\vdash \perp .$$

Gerade in frühen Phasen des Entwurfs, wenn Komponentenauswahl, Parameter oder sogar Strukturen noch nicht vollständig festgelegt sind, ist oft gar nicht mehr als die Konsistenz, d.h. die *Möglichkeit* (aber nicht Garantie) der Erfüllung der Zielspezifikation, überprüfbar. Außerdem ist der Entwurfsprozeß zumeist kein einmaliger Wurf und auch kein Suchprozeß mit monotoner Verfeinerung sondern eine Sequenz von Entwurfshypothesen und deren Revision, d.h. er beinhaltet den Schritt von einem mit dem Ziel inkonsistenten zu einem konsistenten Modell:

$$\text{MODEL}_0 \cup \text{GOALS} \vdash \perp \quad \rightsquigarrow \quad \text{MODEL}_1 \cup \text{GOALS} \not\vdash \perp$$

Der Begriff der Revision soll dabei andeuten, daß der Übergang zum neuen Modell in der Regel kein Sprung zu einer völlig neuen Hypothese ist, sondern sich systematisch durch eine Veränderung einzelner Elemente der Ausgangshypothese unter Bewahrung von wesentlichen Teilen ergibt. Dies führt zu einer Vorgehensweise, in der nicht nur die Konsistenz des Gesamtmodells mit den Zielen zu überprüfen ist, sondern auch möglichst eng eingegrenzte Elemente identifiziert werden, die für eine Inkonsistenz (allein) verantwortlich sein können. Dies sind dann Kandidaten für eine begrenzte Revision der Entwurfshypothese. Dieser Ansatz als „Debugging von Entwürfen“ hat offenkundige Affinität zu Diagnoseaufgaben, auf die wir noch eingehen werden. Der Entwurfsprozeß erhält oft eine Struktur dadurch, daß die Ziele sich in verschiedenen Phasen in Art, Umfang und Granularität unterscheiden können. Ferner soll ausdrücklich betont werden, daß wir uns hier auf diejenigen Aspekte der Spezifikation beschränken, die Ziele im Hinblick auf das

Systemverhalten darstellen und daher auf der Basis von Verhaltensmodellen unterstützt werden können. In der Praxis treten immer Anforderungen hinzu, die jenseits dieser Methode liegen und durchaus das dominierende Kriterium für den Entwurf darstellen: Anforderungen der Herstellung und Montage, sowie des Recycling, vor allem Kosten (Material-, Fertigungs-, Wartungskosten), aber auch geometrische Beschränkungen, gesetzliche Vorschriften, u.s.w.

12.2.2 Failure-Modes-and-Effects Analysis

Ferner treten Sicherheitsaspekte hinzu. Insbesondere ist sicher zu stellen, daß das zu entwerfende System auch im Fehlerfalle bestimmten Anforderungen genügt. In verschiedenen Branchen (etwa der Auto- und Luftfahrtindustrie) wird daher die FMEA (Failure-Modes-and-Effects Analysis) betrieben, in der in systematischer Weise die Auswirkungen von Komponentenfehlern auf die Funktion des Systems untersucht werden. Offenkundig liegt diese Aufgabe im Bereich modellbasierter Methoden, wenn Modelle des Fehlverhaltens verwendet werden können. Die Aufgabe besteht dann entweder einfach darin, aus diesem Modell bestimmte Verhaltensaspekte abzuleiten,

$$\text{MODEL}_{\text{Fault}} \vdash ? ,$$

oder wiederum die Konsistenz mit explizit gegebenen Zielen zu überprüfen:

$$\text{MODEL}_{\text{Fault}} \cup \text{GOALS} \stackrel{?}{\vdash} \perp ,$$

etwa von der Art „Auch im Fehlerfalle liegt bei ausgeschalteter Zündung nie eine Spannung an Komponente C 17“.

12.2.3 Fertigung

Wir klammern Aufgabenstellungen im Hinblick auf den Fertigungsprozeß weitgehend aus, denn hier wird der Gegenstand, das Produkt, zumeist nicht unter dem Aspekt seines Verhaltens betrachtet, sondern im Hinblick auf seine Eigenschaften, die für den Einsatz der jeweiligen Produktionswerkzeuge relevant sind, also insbesondere materielle und geometrische Bedingungen. Soweit man den Fertigungsprozeß selbst als System betrachtet, taucht er als Gegenstand anderer genannter Aufgaben auf, also von Entwurf, Überwachung, Diagnose etc.

12.2.4 Testgenerierung

Um das Produktverhalten geht es am Schluß der Fertigung bei der Endprüfung. Testgenerierung zielt darauf, eine Reihe von Beeinflussungen des Systems und Messungen des Systems vorzuschlagen, die die Entscheidung gestatten, ob es gemäß Entwurf gefertigt ist oder Defekte enthält. In der Regel ist es kein gangbarer Weg, direkt zu überprüfen, ob das gefertigte Produkt sich entsprechend dem Entwurf verhält, denn dies soll für alle möglichen Situationen (z.B. als Kombination von Inputwerten) gelten, was zumindest theoretisch unendlich viele oder jedenfalls praktisch zu viele sind. Will man das System nur in einer möglichst geringen Zahl von Situationen prüfen, so wird Evidenz dafür benötigt, daß sich das System auch in den nicht behandelten Situationen korrekt verhält.

Dies ist gleichbedeutend damit, daß sich ein eventuell vorliegender Fehler mit Sicherheit in den betrachteten Situationen durch abweichendes Verhalten manifestiert hätte. Formal läßt sich diese Aufgabe fassen als Bestimmung einer Situation, SIT, deren mögliche zugehörige Beobachtungen mit dem einen Modell (z.B. dem des Korrekt gefertigten Systems) konsistent sind, mit einem anderen (des Fehlverhaltens) aber inkonsistent:

$$\begin{aligned} \text{MODEL}_{\text{Correct}} \cup \text{SIT} &\vdash \text{OBS}_0 \\ \text{MODEL}_{\text{Fault-}i} \cup \text{SIT} &\vdash \text{OBS}_1 \\ \text{OBS}_0 \cup \text{OBS}_1 &\vdash \perp \end{aligned}$$

Für eine vollständige Prüfung muß $\text{MODEL}_{\text{Fault-}i}$ dann über alle möglichen (oder wahrscheinlichen) Fehler laufen. Dabei gilt es letztlich, eine möglichst kleine (oder kostengünstige) Sequenz von Tests in möglichst günstiger Reihenfolge zu durchlaufen.

12.2.5 Überwachung und Diagnose

Für das fertige System stellt sich die Aufgabe der Überwachung seines Betriebs („monitoring“). Dies bedeutet, aufgrund der verfügbaren Beobachtungen OBS über das Systemverhalten (etwa aufgrund von Sensorsignalen) festzustellen, ob das System korrekt arbeitet. Formal bedeutet dies wieder eine Konsistenzprüfung,

$$\text{MODEL} \cup \text{OBS} \cup \text{GOALS} \stackrel{?}{\vdash} \perp ,$$

oder die Frage nach der Ableitung der Ziele:

$$\text{MODEL} \cup \text{OBS} \stackrel{?}{\vdash} \text{GOALS}$$

Setzt man voraus, daß der Entwurf diese Ziele erfolgreich umgesetzt hat, d.h.

$$\text{MODEL} \vdash \text{GOALS}$$

gilt, so reduziert sich das Problem auf die Überprüfung

$$\text{MODEL} \cup \text{OBS} \stackrel{?}{\vdash} \perp .$$

Wird eine Inkonsistenz festgestellt, d.h. liegt ein unerwünschtes Verhalten vor, ist eine Diagnose durchzuführen, um mögliche Ursachen zu ermitteln. Für uns läßt sich dies wiederum formalisieren als Suche nach einem revidierten Modell, das zumindest konsistent mit den Beobachtungen ist:

$$\text{MODEL}_0 \cup \text{OBS} \vdash \perp \quad \rightsquigarrow \quad \text{MODEL}_1 \cup \text{OBS} \not\vdash \perp ,$$

oder das die Beobachtungen erklärt, d.h. ableitet:

$$\text{MODEL}_1 \vdash \text{OBS} .$$

Im Vergleich zur Designaufgabe sind also Entwurfsziele durch Beobachtungen ersetzt, und der Raum möglicher Modellrevisionen ist gegeben durch die möglichen Störungen des Systems, also insbesondere durch Fehlverhalten der Komponenten des Systems. Wiederum geht es nicht um beliebige Revisionen, sondern, sogar stärker noch als beim Entwurf, um minimale Modifikationen des Modells (etwa gemessen an der Menge defekter Komponenten).

Oft gelingt aufgrund der anfangs vorliegenden Beobachtungen keine eindeutige Eingrenzung der Störung, und es gilt, möglichst gezielt und effizient weitere Systembeobachtungen bereitzustellen, die diesem Ziel dienen. Dies führt auf die bereits diskutierte Aufgabenstellungen der Testgenerierung, hier mit dem leicht modifiziertem Ziel, die möglichen Störungen voneinander, statt nur vom korrekten Verhalten zu unterscheiden.

12.2.6 Therapie

Natürlich wird dies meist mit dem Ziel durchgeführt, die korrekte Funktion des Systems wiederherstellen zu können, d.h. mit dem Ziel der Therapieregenerierung. Es geht darum, durch verfügbare Aktionen so auf das gestörte System einzuwirken, daß es das Zielverhalten (oder eine Modifikation davon) einhält:

$$\text{MODEL}_1 \cup \text{GOALS} \vdash \perp \quad \rightsquigarrow \quad \text{MODEL}_1 \cup \text{ACTIONS} \cup \text{GOALS} \not\vdash \perp$$

bzw.

$$\text{MODEL}_1 \cup \text{ACTIONS} \vdash \text{GOALS} .$$

Der Form nach haben wir wieder eine Entwurfsaufgabe. Der Unterschied liegt darin, daß die Modellrevisionen durch die am realisierten System möglichen Manipulationen gegeben sind statt durch die Entwurfsalternativen.

Der sehr allgemein gewählte Begriff der Therapie umfaßt (theoretisch und praktisch) sehr unterschiedliche Fälle:

- *Reparatur* im engeren Sinne bedeutet zumeist das Ersetzen defekter Komponenten. In diesem Fall bestimmt die Diagnose, die die Defekte lokalisiert, bereits die Therapie.
- *Rekonfigurierung* versucht, ein (eventuell eingeschränktes) Zielverhalten ohne Behebung der Ursachen, d.h. im Rahmen des gestörten Systems zu realisieren. Bei der Restauration eines Hochspannungsnetzes wird bei einem Ausfall von Leitungen oder Knoten die Versorgung durch Umlegen von Schaltern, d.h. Ausnutzen struktureller Redundanz, sichergestellt. Ein MITA-Kopierer wählt bei ungenügender Bildqualität aufgrund eines veränderten Systemparameters einen anderen Arbeitspunkt, der die negativen Einflüsse kompensiert. Steuergeräte auf Fahrzeugen ermöglichen bei bestimmten Fehlern einen eingeschränkten Betrieb, etwa durch Begrenzung gewisser Betriebsgrößen.
- Therapie im allgemeinen Sinn kann das System durch zusätzliche Elemente erweitern, um es in den gewünschten Zustand zu überführen. Dies betrifft häufig biologische oder chemische technische Prozesse und natürliche, etwa Ökosysteme. Maßnahmen zur Behandlung eines verseuchten Flusses sind ein Beispiel dafür.

Letzteres illustriert auch, daß die obige Formalisierung der Therapieregenerierung einen wesentlichen Aspekt unberücksichtigt läßt, nämlich die Dimension der Zeit. Die Anwendung der Therapieaktion führt ja immer bestenfalls mit einer zeitlichen Verzögerung zum Erreichen der Ziele. Analog dazu können natürlich auch die Ursachen für Systemstörungen in der Vergangenheit liegen.

12.2.7 Verallgemeinerung

Die Gründe dafür, die genannten Problemlöseschritte im Lebenszyklus auf Modelle statt das physikalische System selbst zu stützen, sind im wesentlichen:

- Das physikalische System steht nicht zur Verfügung, etwa beim Entwurf.
- Es ist zu aufwendig oder unmöglich, gewisse Größen zu messen.
- Es ist zu kostspielig oder unmöglich, es in die gewünschten Zustände zu versetzen. Dies trifft vor allem für die Realisierung verschiedener Fehler- oder Therapiemöglichkeiten zu.

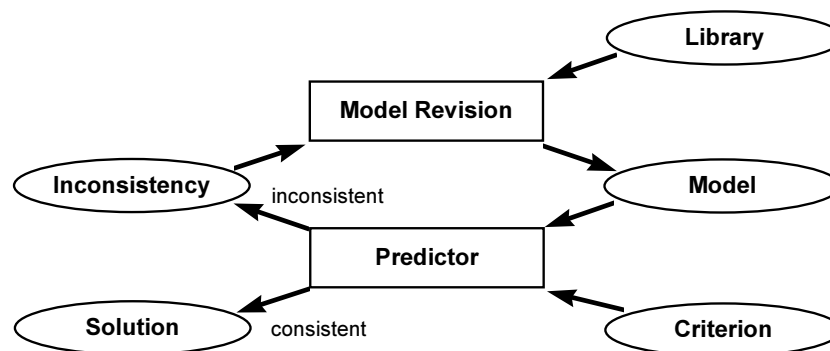


Abbildung 12.2: Modellbasierte Verhaltensvorhersage, Konsistenzprüfung und Modellrevision

Die Systematisierung der Aufgaben zeigt, daß der Kern modellbasierter Problemlösungen im dargestellten Ansatz in drei sehr allgemeinen Teilaufgaben besteht.

- Verhaltensvorhersage:

$$\text{MODEL} \vdash ?$$

- Konsistenzprüfung:

$$\text{MODEL}_1 \cup \text{CRITERION} \vdash ? \perp$$

- Modellrevision:

$$\text{MODEL}_1 \cup \text{CRITERION} \vdash \perp$$

$$\rightsquigarrow \text{MODEL}_2 \cup \text{CRITERION} \not\vdash \perp$$

Der Zusammenhang ist in Abb. 12.2 dargestellt, wobei das abstrakte Kriterium je nach Aufgabenstellung variiert und aus Beobachtungen, Entwurfs- oder Therapiezielen etc. besteht.

Während die ersten beiden Teilaufgaben, eng miteinander verbunden, natürlich das sind, was man von einem Modell erwartet, stellt die dritte zusätzliche Anforderungen, die wir im folgenden zu berücksichtigen haben. Wir haben betont, daß der Modellrevisions-schritt durch Veränderung, Hinzufügen oder Eliminieren von (möglichst wenigen) Modell-elementen geschieht, deren Natur von der jeweiligen Aufgabe abhängt: z.B. Austausch von Modellen des (korrekten oder fehlerhaften) Komponentenverhaltens bei der Diagnose, Hinzufügen von Therapieaktionen oder Austausch von Entwurfsalternativen (im einfachsten Falle Auswahl von Komponententypen oder Parameterwerten). Die Schlußfolgerungen zur Modellrevision finden auf einer *konzeptuellen* Ebene statt, was zunächst zu der Forderung führt, daß das Modell diese Schicht überhaupt explizit repräsentieren muß, wie schon eingangs betont. Die Verhaltensvorhersage aber geschieht auf der Grundlage eines *mathematischen* Modells. Diese Kluft zu überbrücken, erfordert als weitere Komponente die *Modellkomposition*, die aus den konzeptuellen Beschreibungen und Revisionen automatisch ein Verhaltensmodell erzeugt (s. Abb. 12.3).

Wenn dabei die konzeptuelle Struktur im mathematischen Verhaltensmodell erhalten bleibt, so bildet dies die Basis dafür, daß es nur lokal geändert werden muß, was wesentlich für die Effizienz der Verfahren ist. Noch grundlegender ist dies aber für die Bestimmung, welche Elemente des Modells Kandidaten für eine Revision sind: statt nur die Inkonsistenz des Gesamtmodells zu konstatieren (was alle seine Elemente „verdächtig“ macht), kann das System Inkonsistenzen in Teilmodellen entdecken und Revisionshypothesen darauf fokussieren. Ohne diese Eigenschaft wäre der jeweilige Prozeß nur eine blinde Suche.

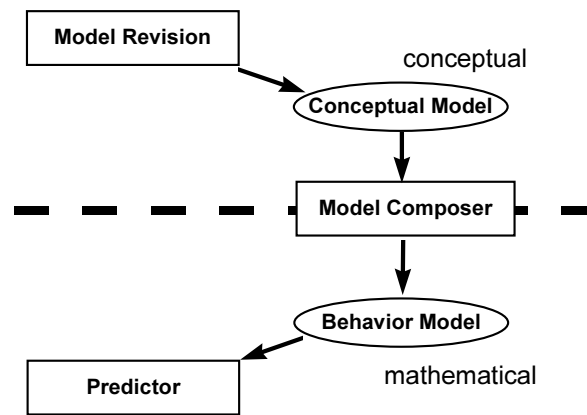


Abbildung 12.3: Modellkomposition als Brücke zwischen konzeptueller Modellrevision und mathematischer Verhaltensvorhersage

Wir bezeichnen die geschilderte Klasse modellbasierter Systeme im folgenden als *konsistenzbasierte Problemlöser*.

12.3 Modellierung und Verhaltensvorhersage

12.3.1 Grundlagen und Ziele

Aus dem Vorangegangenen ergeben sich grundlegende Ziele und Eigenschaften von Modellen und Modellierung, die in diesem Abschnitt diskutiert werden sollen. Insbesondere die Anforderung der kompositionalen Modellierung ist fundamental und prägend sowohl für die konzeptuelle Ebene als auch die der Verhaltensbeschreibung. Sie beinhaltet

- daß es möglich ist, auf *algorithmischem* Wege aus einer konzeptuellen Beschreibung eines Systems eine *Verhaltensbeschreibung zu erzeugen* (manchmal „structure-to-behavior reasoning“ genannt). Dabei kann solch eine konzeptuelle Beschreibung im einfachen Fall eine Aufzählung der Komponenten und ihrer Verbindungen (etwa in einer elektrischen Schaltung) sein, aber auch die Beschreibung eines Teichs, in dem sich gelöster Wasserstoff, tote organische Masse, bestimmte Bakterien, Blau- und Grünalgen etc. befinden und in den landwirtschaftliche Düngemittel eingespült werden.
- daß sich dabei das *Verhaltensmodell* aus einer *endlichen Menge von Elementen* eines gewissen Vorrats („Modellbibliothek“, „domain theory“) *zusammensetzen* läßt, die voneinander unabhängige physikalische Phänomene repräsentieren (reduktionistischer Ansatz). Dies können Gleichungen sein, die das Verhalten elektrischer Komponenten beschreiben, oder die Charakterisierung von im Teich aktiven Prozessen wie Zersetzung organischer Masse, Austauschprozesse, Oxidation, etc.

Im Hinblick auf den Ansatz von Problemlösen durch Modellrevision tritt die Bedingung hinzu

- daß die Einheiten, die *Gegenstand einzelner Revisionsschritte* sind, zu den *konzeptuellen Einheiten* zählen und mit ihnen ein unabhängiges Verhaltensmodellfrag-

ment assoziiert ist. Dies sind, je nach Aufgabenstellung, etwa Entwurfshypothesen, Störungen und Fehlerarten oder Therapiemaßnahmen.

Damit ist es dann nicht nur möglich, überhaupt das aus einer Revision resultierende Verhaltensmodell automatisch zu erzeugen (und damit den menschlichen Modellierer in dem Zyklus entbehrlich zu machen), sondern auch, dies durch eine *lokale* Modifikation zu realisieren. Dies verspricht effiziente Lösungen, und zwar nicht nur für die Modellgenerierung, sondern auch bei der Verhaltensvorhersage durch die Übernahme von Vorhersagen, die aus dem nicht revidierten Teil des Modells resultieren (vgl. dazu Abschnitt 12.4.2.7). Vor diesem Hintergrund stellt sich im folgenden zunächst die Frage nach geeigneten Sprachen für die konzeptuelle Schicht, nach

- „*Ontologien*“, d.h. danach, welches die geeigneten Primitiva sind, die als Konstituenten ein System auf der *konzeptuellen* Modellebene repräsentieren.

Danach werden

- Formalismen für die Elemente der *Verhaltensbeschreibung* als kombinierbare Bausteine vorgestellt. Die Verhaltensmodelle sind oft *qualitativ*, um der Natur von vorhandenem Wissen oder Information gerecht zu werden (z.B. verfügbare Beobachtungen, gewünschte Vorhersagen, Unbestimmtheit des Entwurfsmodells), oder vereinigen als hybride Modelle verschiedene Ebenen der Granularität.

Auf dieser Grundlage wird dann

- die Aufgabe der *Modellkomposition* behandelt, also die automatische Erzeugung des Verhaltensmodells eines komplexen Systems aus seiner konzeptuellen Beschreibung mit Hilfe einer Bibliothek von Modellfragmenten.

Das Ziel der Modellierung ist natürlich

- die Lösung von Aufgabenstellungen der *Verhaltensvorhersage* auf der Basis der Modelle.

Schließlich werden einige spezielle Aspekte diskutiert.

12.3.2 Konzeptuelle Modelle

Die KI hat im wesentlichen zwei Ontologien hervorgebracht, die verschiedene Sichten darauf widerspiegeln, wie Veränderung in einem System herbeigeführt wird. In vielen technischen Bereichen werden Systeme aus abgrenzbaren physikalischen Bausteinen mit wohlspezifiziertem Verhalten zusammengesetzt, um ein gewünschtes Gesamtverhalten zu produzieren. Hier wird man auch komponentenorientiert modellieren: Komponenten sind aktive Objekte, die ihre Effekte über definierte Nahtstellen austauschen. Digitale Schaltungen sind ein typisches Beispiel dafür.

Eine andere Sicht besteht darin, daß sich unter bestimmten Konstellationen von Objekten und Substanzen und weiteren Voraussetzungen eine Veränderung der beteiligten Objekte aus ihrer *Interaktion*, also nicht durch das einzelne Objekt, ergibt. Dies wird im Konzept eines aktiven *Prozesses* gefaßt. Chemische Reaktionen und biologische Prozesse illustrieren diese Sicht, aber auch Gravitation als wechselseitig bewirkte Kraft, die nicht einem einzelnen Objekt als „Komponente“ zugewiesen werden kann.

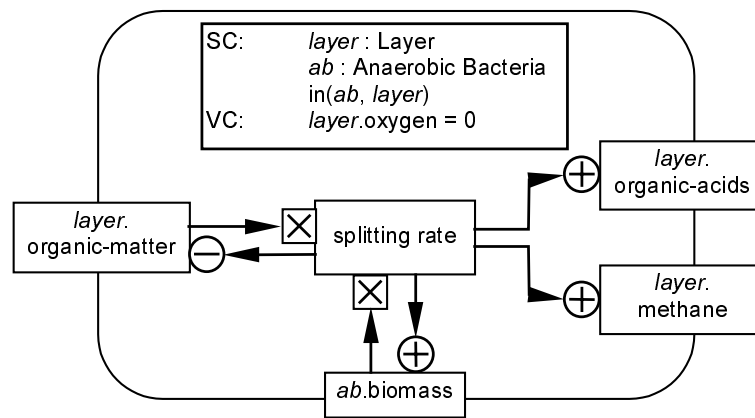


Abbildung 12.4: Der Prozeß „Anaerobe Zersetzung“.

12.3.2.1 Prozesse

Die prozeßorientierte Ontologie wurde zuerst in der Qualitativen Prozeßtheorie (Qualitative Process Theory, QPT) [Forbus, 84] mit dem Ziel entwickelt, ein Repräsentationssystem für die „Naive Physik“ (s. [Hayes, 85]) zu bieten. Prozesse lassen sich so darstellen, daß sich die komponentenorientierte Modellierung als Spezialfall ergibt. Daher charakterisieren wir sie auf allgemeiner Ebene (vgl. [Struss, Heller, 98]) und illustrieren dies am Beispiel „Anaerobe Zersetzung“ in Abb. 12.4. Es existiert eine Menge von

- *Objekttypen* („Individuals“ bei Forbus) also etwa Substanzen wie Methan oder eine Wasserschicht, und
- *Relationen* zwischen Objekten wie „enthalten in“, sowie
- *Assoziationen von Systemgrößen* mit Objekttypen und Wertebereichen, z.B. von Parametern mit Komponenten oder auch „Konzentration“ mit Substanzen in Flüssigkeiten.

Eine *Systemstruktur* wird durch instanziierte Objekte und Relationen zwischen Ihnen beschrieben. Eine Bibliothek (bzw. „Domain Theory“) beinhaltet ferner

- *Typen von Verhaltenskonstituenten* („processes“), die ein Verhaltensmodellfragment mit Vorbedingungen für dessen Gültigkeit verknüpfen, z.B. den Prozeß „Anaerobe Zersetzung“.

Die Vorbedingungen können

- *Strukturbedingungen* („preconditions“) sein, also Objekttypen und Relationen, etwa „Organische Materie enthalten in Wasserschicht“ sowie
- *Wertebedingungen* („quantity conditions“), die lokal sind, d.h. sich ausschließlich auf Systemgrößen der in den Strukturbedingungen auftretenden Objekte beziehen („Sauerstoffkonzentration gleich null“).

Ein Verhaltensmodellfragment repräsentiert den Effekt des modellierten Phänomens, der an keine weiteren Vorbedingungen als die genannten geknüpft ist und ebenfalls lokal sein muß:

$$\text{STRUCT-CONDS} \wedge \text{QUANT-CONDS} \\ \Rightarrow \text{STRUCT-EFFECTS} \wedge \text{QUANT-EFFECTS} ,$$

und zwar in Form von

- *Struktureffekten*, d.h. Objekten und Relationen, die – falls nicht schon existent – neu kreiert werden, z.B. Zersetzungsprodukte wie Methan,
- *Wertebeschränkungen* („relations“ und „influences“ bei Forbus), die sich auf Systemgrößen von Objekten beziehen, die in den Strukturbedingungen oder -effekten erscheinen oder rein lokale Größen sind, die in keinen anderen Verhaltenskonstituenten auftauchen, wie die Zersetzungsrate bei der anaeroben Zersetzung.

Diese Wertebeschränkungen können z.B. numerische Gleichungen oder auch Relationen (siehe 8) über qualitativen Wertebereichen sein, wenn sie tatsächlich uneingeschränkt, d.h. unabhängig von anderen existierenden Verhaltenskonstituenten, gelten, im Beispiel: die Bestimmung der lokalen Größe „Zersetzungsrate“ durch die Biomasse organischer Stoffe und deren Konzentration in der Wasserschicht. Die Forderung nach Kombinierbarkeit der Modellfragmente verlangt aber ggf. die *partielle Spezifikation der Effekte*. D.h. wenn sich der tatsächliche Effekt erst nach Kombination mit anderen Modellfragmenten bestimmen läßt, so ist dieser als *Einfluß* zu beschreiben. So hat die anaerobe Zersetzung einen positiven Einfluß auf die Methankonzentration, die aber durch andere Prozesse wie Aufstieg oder chemische Reaktionen insgesamt sogar sinken könnte. Die mathematische Form dieses Konzepts wird im folgenden Abschnitt definiert, die Kombination der Einflüsse wird in Abschnitt 12.3.4 behandelt.

12.3.2.2 Komponenten

Die komponentenorientierte Sicht (eingeführt z.B. in ENVISION [de Kleer, Brown, 84; Davis, 84]) betrachtet Systeme als aus *Komponenten* (Components) und *Verbindungen* (Conduits) zusammengesetzt, die über *Terminals* verbunden sind. Sie ergibt sich als Spezialisierung der oben dargestellten Konzepte zunächst durch Beschränkung der Objekttypen auf

- *Komponententypen*: physikalisch abgrenzbare Objekte von unterschiedlichem Typ (z.B. ein Widerstand, ein Ventil oder ein Zahnrad), die ein für den Typ charakteristisches und vorhersagbares Verhalten aufweisen.
- *Verbindungstypen*: passive Kanäle, durch die Materie oder Information zwischen Komponenten ausgetauscht werden kann, wie z.B. ein Draht oder ein Rohr ohne Widerstand. Grundsätzlich speichern oder verändern sie dabei nichts und können als spezielle Komponententypen aufgefaßt werden.
- *Terminaltypen*: Verbindungspunkte zwischen Komponenten und Verbindungen. Komponenten können ausschließlich über ihre Terminals mit ihrer Umgebung kommunizieren.

Komponenten mit Fehlverhalten sind dabei als eigene Komponententypen zu sehen. Alternativ kann eine lokale Variable eingeführt werden, die als symbolische Werte die verschiedenen Fehlverhalten aufweist und in Abhängigkeit davon im Verhaltensmodell auf verschiedene Verhaltensbeschreibungen verzweigt. Die grundlegende Relation ist

- *Terminal-von*, um die Zugehörigkeit von Terminals zu Komponenten und Verbindungen anzugeben.

Als Systemgrößen sind Komponententypen Parameter und Zustandsgrößen zugeordnet, während die Terminals die im Hinblick auf die Komponenten exogenen Variablen enthalten. Verhaltenskonstituenten haben

- *ausschließlich Strukturbedingungen*, und nur solche, die einen einzigen Komponententyp und eine Menge von Terminals unter der Terminal-von-Relation beinhalten:

$$\begin{aligned} \text{STRUCT-CONDS} \equiv & \exists \text{BEH-MODE}_j(\text{COMPONENT}_i) \wedge \\ & \exists \text{TERMINALS} \forall \text{TERMINAL}_k \in \text{TERMINALS} \\ & \text{TERMINAL-OF}(\text{TERMINAL}_k, \text{COMPONENT}_i). \end{aligned}$$

Dies reflektiert, daß das entsprechende Verhalten nur von der Existenz der Komponente bzw. des Verhaltensmodus abhängt. Die Effekte sind hingegen

- *ausschließlich Wertebeschränkungen*, denn in der herkömmlichen komponentenorientierten Modellierung werden keine neuen Komponenten oder Terminals erzeugt. Lokalität dieser Wertebeschränkungen heißt hier, daß sie sich ausschließlich auf Parameter und Zustandsvariablen der Komponente und die Variablen der Terminals beziehen.

Gewöhnlich wird die Sicht eingenommen, daß die Effekte einer Komponente vollständig festgelegt sind, und entsprechend enthält das Verhaltensmodell ausschließlich Relationen und keine Einflüsse. Unter der obigen verallgemeinerten Sicht läßt sich der Standardansatz der Komponentenmodellierung aber durchaus nutzbringend erweitern: z.B. lassen sich über Einflüsse Behälter mit einer unbestimmten oder erweiterbaren Anzahl von Zuflüssen realisieren.

Im allgemeinen wird die Struktur eines Gerätes als fest vorgegeben angesehen. Obwohl dies für große Klassen, wie z.B. Schaltkreise, plausibel ist, ist es doch eine wichtige Einschränkung, die insbesondere die Behandlung mechanischer Systeme erschwert, bei denen sich bewegende Teile ständig Verbindungen neu erschaffen und aufheben können. Probleme treten auch bei Diagnoseaufgaben auf, wenn die Ursache einer Fehlfunktion eine Verletzung der beabsichtigten Struktur ist (z.B. ein Brückenfehler auf Platinen). Auch dies läßt sich über die Benutzung von Einflüssen in den Komponentenmodellen bewältigen.

Mit der oben genannten Beschreibung hat ein System eine flache Struktur. Dies ist für komplexe System wie große Schaltungen unangebracht. In [Davis, 84; Hamscher, 91] werden Systeme präsentiert, die eine hierarchische und nach verschiedenen Aspekten differenzierte Strukturbeschreibung erlauben. Formal bedeutet dies als erstes nur die zusätzliche Einführung einer *Teil-von*-Relation für Komponenten. Eine Komponente und ihre Teile können wiederum gemeinsame Terminals haben. Allerdings benötigt man für die Verhaltensvorhersage auf Basis des Modells Mechanismen, um diese auf bestimmte Hierarchie-Ebenen zu fokussieren.

12.3.3 Qualitative Verhaltensbeschreibungen

In diesem Abschnitt wollen wir uns auf die Darstellung von Formalismen beschränken, wie sie in der KI für die qualitative Verhaltensmodellierung entwickelt wurden. Prinzipiell können für etliche der Aufgabenstellungen auch numerische Modelle eingesetzt werden. Allerdings müssen diese die Anforderungen erfüllen, die sich aus der Modellkomposition (vgl. Abschnitt 12.3.4) und dem Revisionsansatz (vgl. Abschnitte 12.2.7 und 12.4.2.7) ergeben, was für viele gebräuchliche numerische Modellierungssysteme nicht gewährleistet ist. Andere Gründe dafür, qualitative Modelle als Grundlage zu wählen, sind

- *Effizienz* der Verhaltensvorhersage und
- *Natürlichkeit* und eine *intuitive Darstellung* der Modelle.

Vor allem aber können sie reflektieren, daß gewisse

- *Unterscheidungen* zwischen Werten *nicht notwendig* oder
- *nicht möglich* sind.

Für die konsistenzbasierten Problemlöser ist ein Kriterium, Unterscheidungen zu vermeiden, die keinen Beitrag zur Erkennung von Inkonsistenzen leisten. All dies motiviert qualitative Methoden im folgenden Sinne:

Qualitative Modellierung zielt darauf, nur die wesentlichen (und die möglichen) Unterscheidungen in einem Verhaltensmodell widerzuspiegeln.

Man kann die – in vielen Fällen durchaus angreifbare – Sicht einnehmen, daß sich die zu modellierenden Systeme „eigentlich“ in einer Weise verhalten, die sich durch gewöhnliche Differentialgleichungen und reellwertige (etwa stetige oder differenzierbare) Funktionen als Lösungen beschreiben läßt, dies aber aus praktischen Gründen nicht gewollt oder möglich ist. Aus dieser Perspektive lassen sich qualitative Modelle als Abstraktion oder Transformation von reellwertigen (oder jedenfalls feineren) Verhaltensmodellen auffassen. Im wesentlichen kommen sie dann zustande durch das Zusammenfassen von Werten oder funktionalen Beziehungen der relevanten Systemgrößen. Es stellt sich dann die Frage, wie sich aus der reellwertigen Beschreibung die qualitative ergibt und in welchem Verhältnis die Ergebnisse der Verhaltensbeschreibung zueinander stehen.

12.3.3.1 Relationale Modelle

Es geht uns bei allen Ansätzen der Modellierung darum, die möglichen Zustände und Entwicklungen eines Systems zu charakterisieren und durch Beobachtungen und Hypothesen weiter eingrenzen zu können. Diese Charakterisierung bezieht sich auf eine ausgewählte Menge von Systemgrößen v_i (die ggf. auch Ableitungen solcher Größen enthält), die wir zu einem Vektor $\underline{v}_S = (v_1, \dots, v_n)$ zusammenfassen, mit ihren zugehörigen Wertebereichen $\text{DOM}_i(v_i)$, womit der Raum aller beschreibbaren Zustände

$$\text{DOM}(\underline{v}_S) = \text{DOM}(v_1) \times \text{DOM}(v_2) \times \dots \times \text{DOM}(v_n)$$

ist. Unabhängig von der speziellen syntaktischen Form einer Modellbeschreibung interessiert eigentlich nur, welche Zustände aus diesem Raum durch sie zugelassen bzw. ausgeschlossen werden, d.h. welche Relation

$$R_S \subset \text{DOM}(\underline{v}_S)$$

die Modellbeschreibung darstellt. Eine Zeichenfolge $i = v^*R$ ist eben nur eine Zeichenfolge und wird erst zusammengenommen mit einer Interpretation bzw. einem Lösungsalgorithmus zu einem Modell, das dann durch R_S repräsentiert werden kann. Dies scheint zunächst aber nur die Zustände zu charakterisieren; wie sind aber die möglichen *Verhalten*, d.h. Änderungen des Zustands über die Zeit, zu bestimmen? Die meisten Formalismen und Verfahren benutzen eine Menge von *allgemeinen* Regeln oder Prozeduren, die Konzepte wie Stetigkeit, Integration und Ableitung fassen und die festlegen, wie Zustände in der Zeit aufeinander folgen können. Dies gilt sowohl für numerische als auch qualitative Verfahren. „Allgemeine“ Regeln heißt hier, daß sie für *alle* Modelle gelten. Dies bedeutet aber, daß (für diese Hauptklasse von Modellierungsverfahren) die möglichen Verhalten

eines Systems bereits durch die Menge der möglichen Zustände festgelegt ist, was im folgenden Satz (vgl. [Struss, 97]) festgehalten ist:

Theorem (Äquivalenz von Zustands- und Verhaltensmengen)

Zwei Modelle model_1 , model_2 , auf die dieselben Zustandsübergangsregeln angewandt werden, haben dieselben Systemverhalten genau dann, wenn sie dieselben Mengen von Zuständen besitzen:

$$\text{STATES}(\text{model}_1) = \text{STATES}(\text{model}_2)$$

$$\Leftrightarrow \text{BEHAVIORS}(\text{model}_1) = \text{BEHAVIORS}(\text{model}_2).$$

(Es sei noch einmal darauf hingewiesen, daß hier der Zustand auch Ableitungen von Systemgrößen einschließen kann). Dieser Satz hat ziemlich weitreichende Bedeutung für konsistenzbasierte Verfahren, etwa bei der Diagnose dynamischer Systeme, da sich damit die Feststellung der Inkonsistenz zweier Modelle auf die Prüfung der Vereinbarkeit der Zustandsmengen reduzieren läßt (s. Abschnitt 12.4.2.9).

12.3.3.2 Wertebereichsabstraktion

Wenn wir nur reellwertige Verhaltensbeschreibungen als Ausgangspunkt nehmen, so gewinnen wir qualitative Werte als Zusammenfassung reeller Zahlen, die jeweils nicht voneinander unterschieden werden sollen (oder können). Dies kann durch eine Abbildung in die Potenzmenge \wp von \mathbb{R} beschrieben werden:

$$\tau_i: \mathbb{R} \rightarrow \text{DOM}_q(v_i) \subset \wp(\mathbb{R}),$$

wobei $\text{DOM}_q(v_i)$ eine Partitionierung von \mathbb{R} darstellt, d.h. die Bildmengen sind disjunkt und schöpfen \mathbb{R} voll aus. Wenn wir mit stetigen Funktionen zu tun haben, so liegt nahe, daß die zusammengefaßten Werte jeweils zwischen zwei Grenzen liegen, an denen eine wesentliche Veränderung stattfindet, sog. „Landmarks“, also etwa Gefrier- und Siedepunkt von Stoffen. Dann sind für eine Menge von Landmarks

$$L \subset \mathbb{R}_\infty := \mathbb{R} \cup \{-\infty, \infty\}$$

die qualitativen Werte als Intervalle zu interpretieren:

$$\tau_i: \mathbb{R} \rightarrow \text{DOM}_q(v_i) \subset E_L(\mathbb{R}),$$

wobei $E_L(\mathbb{R})$ die Menge der „elementaren“ Intervalle zwischen benachbarten Landmarks und der Landmarks selbst ist:

$$E_L(\mathbb{R}) := L \cup \{(l_i, l_j) \mid l_i < l_j \wedge \neg \exists l_k \ l_i < l_k < l_j\}.$$

(Dazu darf L natürlich keine Häufungspunkte in haben; aber das Ziel ist ja ohnehin, endliche Wertebereiche zu erhalten). Im Vergleich zur herkömmlichen Intervallmathematik, die etwa Intervalle als Ausdruck der Rechenungenauigkeit benutzt, liegt im Bezug auf die Landmarks als Intervallgrenzen ein wesentlicher Unterschied. Dennoch lassen sich Resultate der Intervallmathematik auf qualitative Modellierung übertragen bzw. mit ihr kombinieren.

Häufig sind Landmarks fest vorgegeben, z.B. als Schwellwerte innerhalb eines Komponentenmodells, etwa als maximale Füllhöhe eines Behälters. Sie können aber auch erweitert werden um Werte, an denen sich in einer konkreten Verhaltensanalyse Änderungen ergeben, z.B. wenn eine Größe ein Extremum erreicht (vgl. [Kuipers, 86]). Als weiterer

Unterschied zur Intervallmathematik sind die Landmarks nicht unbedingt als reelle Werte fest gegeben, sondern als Symbole mit einer totalen oder einer Halbordnung, die über den reellen Zahlen als $<$ oder \leq interpretiert wird.

Hat man für die einzelnen Systemgrößen die qualitative Abstraktion τ_i gegeben, so wird damit eine durch das reellwertige Modell gegebene Verhaltensrelation $R_0 \subset \mathbb{R}^n$ „vergrößert“ zu

$$\tau(R_0) \subset \tau_i(\mathbb{R}).$$

Wenn R_0 das tatsächliche Verhalten des Systems abdeckt (d.h. alle überhaupt jemals real möglichen Zustände liegen auch in R_0), so trifft dies auch auf $\tau(R_0)$ zu. Umgekehrt bedeutet dies:

Lemma

Ist $\tau(R_0)$ inkonsistent mit einem Kriterium, das als Relation über den Systemgrößen dargestellt werden kann, so ist auch R_0 damit inkonsistent.

Dies ist die Basis dafür, ein qualitatives Modell als Ersatz für ein präziseres in konsistenzbasierten Problemlösern einzusetzen.

Nun soll ja aber gerade vermieden werden, die reellwertige Relation R_S und ihre Abstraktion unter der Abbildung τ zu berechnen. Vielmehr geht es ja darum, aus einer Beschreibung eines reellwertigen Modells, etwa als System von Gleichungen oder Differentialgleichungen, eine Beschreibung des qualitativ abstrahierten Modells zu erzeugen und zu „lösen“. Hierzu sind die reellwertigen Operatoren, Funktionen etc. auf solche über dem qualitativen Wertebereich abzubilden. Z.B. wird aus der reellwertigen Addition, $+$, über dem Intervallwertebereich

$$\oplus: I^2(\mathbb{R}) \rightarrow I(\mathbb{R}),$$

definiert als

$$\tau(a) \oplus \tau(b) = (l_1, u_1) \oplus (l_2, u_2) := (l_1 + l_2, u_1 + u_2),$$

falls die Menge L der Landmarks abgeschlossen ist in Bezug auf die Addition, bzw.

$$\tau(a) \oplus \tau(b) = (l_1, u_1) \oplus (l_2, u_2) := (\max \{l \in L \mid l \leq l_1 + l_2\}, \min \{u \in L \mid u \geq u_1 + u_2\}),$$

wobei also zu vorhandenen Landmarks „auf/abgerundet“ wird. Analog sind andere arithmetische Operatoren oder Funktionen zu definieren. Was aber tritt an die Stelle von Gleichungen als Modellelementen? Gleichheit auch für die Intervalle als qualitative Werte zu fordern, ist schon wegen der obigen „Rundung“ nicht adäquat. Falls $\tau(a)=(1,2)$, $\tau(b)=(1,2)$, $\tau(c)=(2,3)$ jeweils Intervalle zwischen benachbarten Landmarks sind, sind diese dann eine Lösung der Abstraktion von

$$a + b = c ?$$

Gemäß obiger Definition gilt nicht

$$\tau(a) \oplus \tau(b) = \tau(c).$$

Aber die drei Intervalle decken Lösungen der reellwertigen Gleichung ab, die wir dann durch $\tau(a)$, $\tau(b)$, $\tau(c)$ als Lösungen auf der qualitativen Ebene repräsentiert sehen wollen. Damit ist die geeignete Definition für die qualitativ abstrahierte Gleichung („Confluence“ genannt)

$$\tau(a) \oplus \tau(b) \cap \tau(c) \neq \emptyset.$$

Damit verfügen wir über ein Verfahren, das reellwertige Gleichungen in Confluences überträgt, und über eine Definition der Lösungen von Confluences, und es stellt sich die Frage nach Vollständigkeit und Korrektheit dieser Lösungen. Dies ist im Hinblick auf die Relation $\tau(R_S)$ zu untersuchen, die die qualitativ abstrahierten Lösungen des entsprechenden Gleichungssystems repräsentiert. Hier gilt [Struss, 88].

Theorem

Intervallbasierte Verfahren zur Lösung qualitativ abstrahierter algebraischer Gleichungen sind vollständig, aber im allgemeinen nicht korrekt.

Das bedeutet, sie decken $\tau(R_S)$ ab, produzieren aber zusätzliche „Lösungen“, denen keine reellwertige entspricht. Dies ergibt sich schon aus Ergebnissen der Intervallarithmetik, und der tiefere Grund läßt sich am folgenden Beispiel nachvollziehen:

$$\tau(a)=(1,2), \tau(b)=(2,3)$$

ist eine Lösung von

$$\tau(a) \oplus \tau(a) \cap \tau(a) \oplus \tau(b) \neq \emptyset ,$$

obwohl sie offenkundig keine reellwertige Lösung der Gleichung

$$a + a = a + b$$

abdecken. Es läßt sich zwar in $\tau(a) \oplus \tau(a) \cap \tau(a) \oplus \tau(b) \neq \emptyset$ in jedem Intervall ein Wert wählen, so daß die addierten Werte der linken und der rechten Seite gleich sind, aber dabei müssen *verschiedene* Werte aus dem Intervall (1,2) gewählt werden. Da diese alle mit a assoziiert werden, hat a keinen eindeutigen Wert, und es existiert keine reellwertige Lösung. Zwar kann man im angegebenen Beispiel die offenkundige Ursache, nämlich das mehrfache Auftreten von a , eliminieren. Aber im allgemeinen Fall wird dies über mehrere Gleichungen verstreut auftreten, und entsprechende Umformungen des Gleichungssystems drohen die Anforderung zu verletzen, daß die Struktur des konzeptuellen Modells im mathematischen Modell erhalten bleibt.

12.3.3.3 Schließen über Größenordnung

Die bisher beschriebenen Ansätze charakterisieren Größen durch qualitative Werte bzw. symbolische Werte mit einer Ordnung. Manchmal ist jedoch zusätzliche, aber immer noch qualitative, Information über die *Größenordnungsbeziehungen zwischen Werten* verfügbar und nützlich. Schließen über Größenordnungen (Order of Magnitude Reasoning) versucht, derartiges Wissen zu formalisieren und auszunutzen.

[Raiman, 91] führt im Formalismus FOG drei Relationen ein:

- \ll : *negligible* (vernachlässigbar gegenüber)
- \cong : *close* (annähernd gleich), wobei $A \cong B$ definiert ist als $(A - B) \ll B$
- \sim : *comparable* (vergleichbar mit, selbe Größenordnung), was schwächer ist als *close*.

FOG definiert diese Relationen über eine Menge von Inferenzregeln, die Schlüsse wie

$$A \ll B \wedge B \sim C \Rightarrow A \ll C$$

erlauben, Symmetrie und Transitivität von Relationen festlegen und die Verbindung mit arithmetischen Operatoren beschreiben. FOG wurde in DEDALE verwendet, einem System für die Fehlerbehebung in analogen Schaltkreisen [Dague *et.al.*, 87; Raiman, 91], um Symptome und Fehler als Abweichungen in den Größenordnungen zu charakterisieren (vgl. Abschnitt 12.4.2.1).

Größenordnungen lassen sich aber auch über Schranken für den Quotienten oder die Differenz innerhalb des oben diskutierten intervallbasierten Ansatzes definieren, z.B. durch

$$x \cong y :\Leftrightarrow 1-\epsilon < x/y < 1+\epsilon.$$

(vgl. [Mavrovouniotis, Stophanopoulos, 87], [Dague, 93]), wobei [Dague, 93] eine vierte Relation als Negation von *close* mit der Bedeutung „unterscheidet sich von“ einführt. In [Davis, 90a] wird Schließen über Größenordnungen für die Simulation auf der Grundlage von qualitativ abstrahierten Differentialgleichungen angewandt.

12.3.3.4 Hybride Modelle

Das System Q3 [Berleant, Kuipers, 92] ist eine Erweiterung für den qualitativen Simulationsalgorithmus QSIM. Die qualitativen Landmarks (inklusive der Zeitpunkte) und auch die monotonen funktionalen Abhängigkeiten können durch ein numerisches Intervall möglicher Werte eingeschränkt werden. In [Hellerstein, 92] werden statistische Verfahren mit qualitativen Modellen kombiniert, um Vorhersagen auf der Basis von quantitativ vorliegenden historischen Daten und rein qualitativen funktionalen Abhängigkeiten zu machen. FUSIM [Shen, Leitch, 92] verwendet Fuzzy-Techniken, um unscharfe Information zu behandeln.

12.3.3.5 Funktionsabstraktion

Zunächst ist hervorzuheben, daß durch die Wertebereichsabstraktion die Abstraktionen reellwertiger Funktionen nicht notwendig qualitative Funktionen, sondern allgemein Relationen sind. Z.B. wird bei einer qualitativen Wertebereichsabstraktion, die durch die natürlichen Zahlen \mathbb{N} als Landmarks induziert wird,

$$\tau: \mathbb{R} \rightarrow I_{\mathbb{N}}(\mathbb{R}),$$

unter $y = x^2$ der qualitative Wert (1,2) auf $\{(1,2), 2, (2,3), 3, (3,4)\}$ abgebildet. In aller Regel wird ein qualitatives Modell Ambiguität aufweisen. Dies ist aber unter der Zielsetzung, partielles Wissen explizit auszudrücken, beabsichtigt und keineswegs ein Nachteil gegenüber Verfahren, die Exaktheit suggerieren, aber nur einzelne unter vielen möglichen Approximationen eines nicht exakt bekannten Systems betrachten.

Unvollständiges Wissen kann sich darin niederschlagen, daß funktionale Beziehungen zwischen Größen nicht durch eine spezifizierte reellwertige Funktion gegeben sind. Dies gilt z.B. für Beziehungen zwischen relevanten Faktoren beim Verbrennungsprozeß in einem Automotor. Was aber häufig bekannt ist, sind gewisse monotone Beziehungen, z.B. daß eine Reduktion des Sauerstoffgehalts zu einer Erhöhung der CO-Emissionen führen kann. Als Alternative dazu, solche Zusammenhänge nun durch numerische Funktionen anzunähern (und evtl. anschließend Fehlerabschätzungen durchzuführen), kann

man versuchen, aus dem tatsächlich vorhandenen qualitativen Wissen Schlüsse zu ziehen. Schließlich reicht die genannte monotone Beziehung ja aus, um Fehlerhypothesen beim Auftreten erhöhter CO-Emissionen aufzustellen.

Allgemein können beim Übergang zum qualitativen Modell Funktionsklassen und ihre Charakteristika verwendet werden. Dies geschieht fast ausschließlich für die Klasse monotoner Funktionen, z.B.

$$y = M^+(x)$$

für die Klasse der monoton wachsenden Funktionen. Eine Semantik dafür läßt sich über den reellen Zahlen durch

$$y = f(x) + c \text{ mit } df/dx > 0$$

definieren (vgl. z.B. [Kuipers, 86]). Dies reflektiert die Sicht, daß die Größe sich „wirklich“ durch eine Funktion beschreiben läßt, die reellwertig streng monoton steigt, und damit auch über dem qualitativen Wertebereich. Soll die qualitative Ebene aber z.B. verrauschten Signalen Rechnung tragen, legt dies eine andere Betrachtung nahe, nämlich daß die Abhängigkeit nur qualitativ monoton ist, während das für das reellwertige Signal aufgrund von Störungen überhaupt nicht zutrifft. Eine passende Definition für diesen Fall ist

$$\begin{aligned} \tau(x_1) > \tau(x_2) &\Rightarrow \tau(y_1) \geq \tau(y_2) \\ \exists \tau(x_1), \tau(x_2) \quad \tau(x_1) > \tau(x_2) &\wedge \tau(y_1) > \tau(y_2) \end{aligned}$$

Insgesamt eine streng qualitativ streng monotone Beziehung zu verlangen (d.h. $\tau(y_1) > \tau(y_2)$ in der ersten Implikation), wäre eine zu starke Forderung, da dann ausgerechnet an den Landmarks, d.h. beim Übergang zwischen benachbarten qualitativen Werten, auch reellwertig strenge Monotonie herrschen müßte.

Andere Funktionsklassen neben den monotonen Funktionen werden in [Heller, Struss, 96] betrachtet und im Hinblick auf Modelltransformationen ausgenutzt.

Eine weitere wesentliche Funktionsabstraktion wird durch die Forderung nach kompositionaler Modellierung und damit einer rein lokalen Spezifikation von Effekten von Modellkonstituenten, etwa in der prozeßorientierten Modellierung, erforderlich. Durch das Konzept von *Einflüssen* soll ausgedrückt werden, daß eine Verhaltenskonstituente einen Beitrag zur Veränderung einer Größe leistet, diese aber nicht festlegen kann, da in einem komponierten Modell entgegenwirkende Einflüsse existieren können. Ein positiver Zufluß in einen Behälter bewirkt nicht notwendig ein Anwachsen der Flüssigkeitsmenge, wenn gleichzeitig etwas abfließen oder verdunsten kann. Dies kann nicht durch einen Constraint ausgedrückt werden, sondern z.B. durch einen positiven Einfluß

$$dy/dt = I^+(x_i)$$

mit der Bedeutung

$$\exists f \ y = f(x_1, x_2, \dots, x_i, x_j, \dots, x_n) \wedge \partial f / \partial x_i > 0.$$

Welche anderen Variablen x_k in f auftreten und welche Form f hat, ist erst nach der vollständigen Modellkomposition festgelegt, wie im folgenden Abschnitt beschrieben.

Die relativ schwache Charakterisierung monotoner Beziehungen kann verstärkt werden, indem

- *korrespondierende Werte*, d.h. bekannte (qualitative) Wertepaare der Funktion festgehalten und ausgenutzt werden, und

- durch *Funktionsspezifikation* Beziehungen, die auf dieselben physikalischen Phänomene zurückgehen, identifiziert werden können [Forbus, 84].

Letzteres erst ermöglicht z.B. den Schluß, daß im stationären Zustand die Füllhöhen in kommunizierenden Röhren gleich sind (da die Abhängigkeit des Drucks von der Höhe zwar nur als monoton, aber als für alle gleiche Funktion spezifiziert ist).

12.3.4 Modellkomposition

12.3.4.1 Kontextfreie Modelle

Modellfragmente zu korrekten Modellen komponieren zu können, setzt voraus, daß sie rein lokal sind. Dies bedeutet nicht nur, wie in Abschnitt 12.3.2 beschrieben, sicherzustellen, daß nur lokale Größen in Constraints und Einflüsse eingehen. Die entscheidende Forderung ist, daß die Formulierung der Wertebeschränkung einer Verhaltenskonstituenten keinerlei Voraussetzungen über einen bestimmten Kontext in Form anderer Konstituenten oder die Verwendung zu einem bestimmten Zweck macht (sog. „*No-Function-in-Structure-Prinzip*“). Soll z.B. ein (korrekt funktionierendes) Auslaßventil modelliert werden, so sollte es nicht eine monotone Abhängigkeit des Flusses nach außen von der Öffnung des Ventils beinhalten. Hat sich, etwa im Fehlerfall, außen ein größerer Druck aufgebaut, so wird beim Öffnen des Ventils eine Vergrößerung des Flusses nach innen eintreten. Die Wiederverwendbarkeit, Korrektheit und Nützlichkeit eines Modellfragments wird durch die Verletzung des No-Function-in-Structure-Prinzips in Frage gestellt.

In idealer Form wäre der Anspruch, Modelle für Komponenten zu erstellen, die vollständig sämtliche möglichen Verhalten in beliebigen Kontexten beschreiben. Daß dies praktisch unmöglich ist, macht das Prinzip nicht überflüssig. Es ist für die Analyse unerwarteten oder fehlerhaften Verhaltens unumgänglich. Das Ziel ist daher, lokale Modelle zu erstellen, die für eine hinreichend große relevante Klasse von Systemen gültig sind, ohne deren Funktion vorauszusetzen, und alle den Modellen zugrunde liegenden Annahmen explizit zu machen.

12.3.4.2 Realisierung der Modellkomposition

Automatische Modellkomposition soll aus einer konzeptuellen Beschreibung des Systems ein Verhaltensmodell aus Modellfragmenten einer Bibliothek erzeugen (vgl. Abb. 12.3). Wie in Abschnitt 12.3.2 dargestellt, besteht die Bibliothek aus Konstituenten, die bestimmte Objekte (etwa Komponenten oder deren Fehlverhalten) oder Strukturen von Objekten mit einem Verhaltensmodell assoziieren, wobei evtl. zusätzliche Wertebedingungen erfüllt sein müssen, damit das Verhaltensmodellfragment gültig ist.

Der Ausgangspunkt für die Modellkomposition ist also eine Systembeschreibung bestehend aus

- einer *Menge von Objekten* bestimmten Typs und Relationen zwischen ihnen,
- eine *Beschränkung von Werten* von Systemgrößen dieser Objekte, die Parameter der Objekte einschränken oder die zu betrachtende Klasse von Situationen beschreiben.

Modellkomposition erfordert eine Suche in der Modellbibliothek danach, welche Vorbedingungen von Modellfragmenten durch diese Beschreibung erfüllt und damit aktiv sind. Der einfachste Fall ist die komponentenorientierte Modellierung: in der Systembeschreibung ist eine Liste von Komponenten(verhaltensweisen) sowie deren Verknüpfung mit Terminals und evtl. Einschränkungen auf Parametern und Variablen gegeben. Die Komponentenbibliothek stellt direkt für jede Komponenten(verhaltensweise) ein Verhaltensmodellfragment ohne weitere Vorbedingungen bereit. Diese komponentenbezogenen Fragmente zusammen mit der Identifikation von Variablen über gemeinsame Terminals bilden das Verhaltensmodell, das also in einem Schritt zu erstellen ist:

$$\bigwedge_i \text{STRUCT-CONDS}_i \Rightarrow \bigwedge_i \text{QUANT-EFFECTS}_i ,$$

wobei die Strukturbedingungen wie in 12.3.2.2 beschrieben nur den Verhaltensmodus der Komponente und die zugehörigen Terminals enthalten.

Der allgemeine Fall, gegeben durch die prozeßorientierte Modellierung, ist komplexer aus folgenden Gründen:

- Objekte und Objektrelationen können als Effekt von aktiven Verhaltensmodellen kreiert werden und damit die Vorbedingungen für neue aktive Modellfragmente schaffen.
- Modellfragmente können von der Erfüllung von Wertebedingungen abhängen, die wiederum Resultat von Vorhersagen des Verhaltensmodells sein können.
- Dieses Verhaltensmodell hängt nun davon ab, daß alle Modellfragmente kombiniert sind.

Diese zyklische Abhängigkeit, die in Abb. 12.5 illustriert ist, kann zu einem monotonen Wachstum der Menge von Objekten, Objektrelationen und Verhaltensmodellfragmenten und damit zu immer stärkeren Vorhersagen von Werten führen. Dies gilt aber nicht mehr, wenn die Verhaltensmodellfragmente Einflüsse enthalten: die Komposition des Verhaltensmodells beinhaltet dann für jede Variable die Kombination *aller vorhandenen* Einflüsse. Dies ist Voraussetzung für die Verhaltensvorhersage, die aber wiederum Prozesse aktivieren, so die Menge der Einflüsse erweitern und damit potentiell die Vorbedingung der Vorhersage selbst invalidieren kann. Die für die Einflußkombination notwendige Abschlußannahme „alle Einflüsse sind bekannt“ führt also zu einer Nicht-Monotonie (als „Closed-World-Assumption“, s. Kapitel 7), die die theoretische und praktische Behandlung der Modellkomposition zu einem Problem macht.

Eine Lösungsmöglichkeit ist, Objekte, Relationen und Verhaltenskonstituenten in zwei Schritten einzuführen:

- zunächst als potentiell existent, d.h. nur unter Berücksichtigung der Strukturbedingungen, dann
- in der Verhaltensvorhersage als tatsächlich existent und aktiv, wenn die Strukturbedingungen und Wertebedingungen erfüllt sind, wie in Abb. 12.5 dargestellt (vgl. [Forbus, 84]):

$$\begin{aligned} \text{STRUCT-CONDS}_i &\Rightarrow \text{STRUCT-EFFECTS}_i \wedge \text{POTENTIAL-BEH-CONST}_i \\ \text{POTENTIAL-BEH-CONST}_i &\wedge \text{QUANT-CONDS}_i \\ &\Rightarrow \text{ACTIVE-BEH-CONST}_i \\ \text{ACTIVE-BEH-CONST}_i &\Rightarrow \text{QUANT-EFFECTS}_i \end{aligned}$$

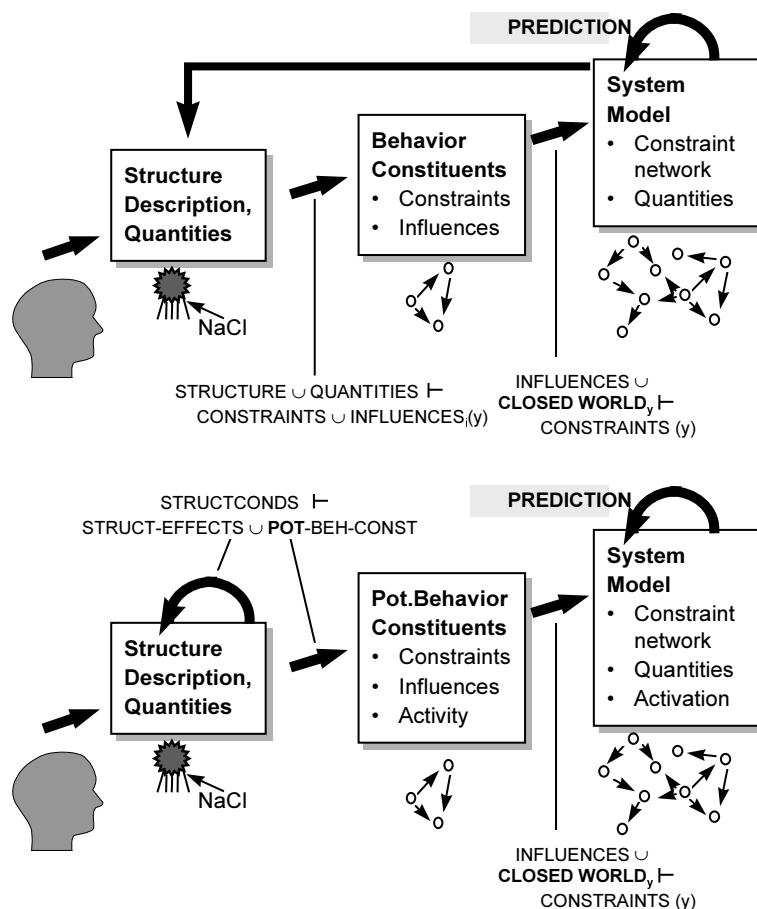


Abbildung 12.5: Nicht-monotone (oben) und monotone Modellkomposition

Im ersten Schritt wird also garantiert eine Obermenge der jemals tatsächlich existenten Objekte und Relationen, damit auch der aktiven Verhaltensmodellfragmente und Einflüsse erzeugt. Diese (potentiellen) Einflüsse können kombiniert werden, wobei nicht-aktive Einflüsse den Beitrag „0“ liefern.

Tatsächliche Existenz und Aktivität können als Variable und ihre Abhängigkeit von Vorbedingungen als Constraints in das komponierte Verhaltensmodell aufgenommen werden (vgl. [Struss, Heller, 98; Struss, Heller, 00]). Auf diese Weise erhält man ein vollständiges Verhaltensmodell, in das die Vorhersage von Existenz von Objekten, Relationen und Verhaltenskonstituenten und deren Aktivität eingebettet ist. Der Preis ist ein Constraint-Netz, das evtl. Teile enthält, die unter einer gegebenen Situationsbeschreibung nie aktiviert werden. Eben deshalb wird aber die Verhaltensvorhersage selbst nicht wesentlich berührt und hat lediglich größere Constraints für die Einflußkombination auszuwerten. In der Regel wird die Kombination der Einflüsse als Linearkombination vorgenommen:

$$dy/dt = \sum f_i(x_i)$$

mit $df_i/dx_i > 0$, falls $y = I^+(x_i)$, und $df_i/dx_i < 0$, falls $y = I^-(x_i)$.

Als Ergebnis erhalten wir eine Komponente, die die Aufgabenstellung der Modellkomposition erfüllt (vgl. Abb. 12.6).

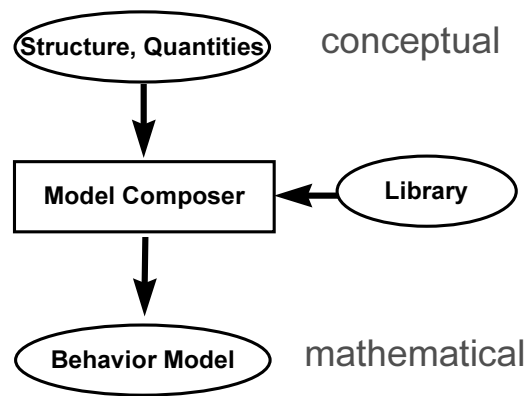


Abbildung 12.6: Modellkomposition schließt die Lücke zwischen konzeptionellen und Verhaltensmodell

12.3.5 Verhaltensvorhersage

12.3.5.1 Vervollständigung der Zustandsbeschreibung

Das Constraint-Netz, das von der Modellkomposition erzeugt wird, beschreibt die möglichen Zustände des modellierten Systems, also die Zustandsrelation R . Sind in der Situationsbeschreibung Parameter- oder Variablenwerte beschränkt, so ist die erste Aufgabe der Verhaltensvorhersage, daraus ggf. Einschränkungen für andere Systemgrößen abzuleiten. Diese Vervollständigung der Zustandsbeschreibung („*State completion*“) ist eine Aufgabe des Constraint-Solvers und nicht spezifisch für modellbasierte Systeme, weshalb wir hier auf eine nähere Darstellung verzichten und auf Kapitel 8 verweisen.

Es ist aber festzuhalten, daß ein unvollständiger Constraint-Solver natürlich, bei qualitativen Modellen zusätzlich zu den in Abschnitt 12.3.3 diskutierten Problemen, weitere Abweichungen von der Relation $\tau(R_0)$ mit sich bringt.

12.3.5.2 Qualitative Simulation

Wie für numerisch modellierte Systeme ist es möglich, die Entwicklung eines Systems ausgehend von einem qualitativen Anfangszustand zu bestimmen (s. [Kuipers, 94]). Diese Entwicklung wird aber nicht als Folge von Zuständen entlang fester Zeitschritte berechnet, da die meisten rein qualitativ operierenden Systeme keine Metrik über der Zeit haben und ausnutzen können. Verhalten eines Systems wird hier beschrieben als Folge unterschiedlicher qualitativer Zustände, was unter kognitiven Gesichtspunkten eine bessere Charakterisierung liefert als eine Folge von reellwertigen Vektoren, denn die wesentlichen Änderungen, die sich in der Verhaltensbeschreibung ausdrücken, sind Erreichen und Verlassen von Landmarks und Aktivierung bzw. Deaktivierung von Prozessen, etc. Aufgrund der Ambiguität der Verhaltensvorhersage ist das Ergebnis häufig kein lineares Verhalten, sondern ein Baum.

Für jeden erreichten Zustand werden die möglichen nachfolgenden bestimmt. Dabei müssen

- der Nachfolgezustand die Constraints des Verhaltensmodells erfüllen (ideal: in R liegen) und

$QS(f, t)$	\rightarrow	$QS(f, t_i, t_{i+1})$	$QS(f, t_i, t_{i+1})$	\rightarrow	$QS(f, t_{i+1})$
P1:	$\langle l_j, std \rangle$	$\langle l_j, std \rangle$	I1:	$\langle l_j, std \rangle$	$\langle l_j, std \rangle$
P2:	$\langle l_j, std \rangle$	$\langle (l, l_{j+1}), inc \rangle$	I2:	$\langle (l_j, l_{j+1}), inc \rangle$	$\langle l_{j+1}, std \rangle$
P3:	$\langle l_j, std \rangle$	$\langle (l_{j-1}, l_j), dec \rangle$	I3:	$\langle (l_j, l_{j+1}), inc \rangle$	$\langle l_{j+1}, inc \rangle$
P4:	$\langle l_j, inc \rangle$	$\langle (l, l_{j+1}), inc \rangle$	I4:	$\langle (l_j, l_{j+1}), inc \rangle$	$\langle (l_j, l_{j+1}), inc \rangle$
P5:	$\langle (l, l_{j+1}), inc \rangle$	$\langle (l, l_{j+1}), inc \rangle$	I5:	$\langle (l_j, l_{j+1}), dec \rangle$	$\langle l_j, std \rangle$
P6:	$\langle l_j, dec \rangle$	$\langle (l_{j-1}, l_j), dec \rangle$	I6:	$\langle (l_j, l_{j+1}), dec \rangle$	$\langle l_j, dec \rangle$
P7:	$\langle (l_j, l_{j+1}), dec \rangle$	$\langle (l, l_{j+1}), dec \rangle$	I7:	$\langle (l_j, l_{j+1}), dec \rangle$	$\langle (l_j, l_{j+1}), dec \rangle$
			I8:	$\langle (l_j, l_{j+1}), inc \rangle$	$\langle l^*, std \rangle$
			I9:	$\langle (l_j, l_{j+1}), dec \rangle$	$\langle l^*, std \rangle$

Abbildung 12.7: P-Transitionen und I-Transitionen. In den Regeln wird jeweils das Paar der qualitativen Werte der Variablen und ihrer Ableitung angegeben, für die Ableitung mit den Werten „dec“ (decreasing), „std“ (steady), „inc“ (increasing) für -, 0, +.

- der Übergang ein zulässiger sein.

Letzteres wird anhand einer Menge von Übergangs- oder *Transitionsregeln* bestimmt, die Stetigkeitsbedingungen sowie die Werte von Ableitungen berücksichtigen. In Abb. 12.7 sind die Transitionsregeln von QSIM (s. [Kuipers, 86]) aufgeführt, die zeigen, daß es nur 16 mögliche Klassen von Übergängen gibt, die die Stetigkeit der Funktionen und ihrer Ableitungen erfüllen. Dies sind sieben mögliche Übergänge von einem Zeitpunkt auf das anschließende Zeitintervall (P-Transitionen) und neun mögliche Übergänge auf Zeitpunkte (I-Transitionen). Die I-Transitionen I8 und I9 sind es, die für zusätzlich mögliche kritische Punkte neue Landmarks erzeugen.

Der QSIM-Algorithmus, der diese Analyse ausführt, arbeitet wie folgt:

Initialisiere die Menge der aktiven Zustände so, daß sie den initialen Zustand umfassen.

Für jeden Zustand in der Menge der aktiven Zustände führe folgendes aus:

1. Bestimme für jede Variable des Zustands die Menge möglicher Übergänge in andere Zustände.
2. Überprüfe für jeden Constraint die Konsistenz der Übergänge.
3. Prüfe für je zwei angrenzende Constraints die Konsistenz der Übergänge ihrer gemeinsamen Parameter (Dies geschieht mit dem Waltz-Algorithmus, verweis 8.3).
4. Generiere alle möglichen nachfolgenden Zustände für den Zustand, d.h. die Menge konsistenter Übergänge.
5. Filtere alle Zyklen, Endzustände sowie unveränderte Zustände aus, und füge die restlichen zu der Menge der aktiven Zustände hinzu.

Die Konsistenzprüfung für die möglichen Transitionen findet also rein lokal statt, d.h. nur zwischen zwei benachbarten Zuständen. Ob zwei oder mehr hintereinander ausgeführte Transitionen noch zu einem gültigen Verhalten gehören, kann durch den obigen Algorithmus nicht geprüft werden. Dies führt dazu, daß selbst unter der Voraussetzung, daß nur physikalisch mögliche Zustände betrachtet werden, Verhaltenspfade erzeugt werden können, die keine Abstraktion einer reellwertigen Lösung darstellen („spurious behaviors“, vgl. [Kuipers, 86]). Zusätzliche Filter, die globale Eigenschaften der Verhalten

berücksichtigen, sind z.B. durch zusätzliche Erhaltungsgesetze zu realisieren oder durch Betrachtungen im Phasenporträt beschränkt auf Differentialgleichungen 2. Ordnung.

12.3.5.3 Envisioning

Da die Menge der konsistenten qualitativen Zustände endlich ist, wird es auch möglich, eine gesamte Charakterisierung der möglichen Verhaltensweisen eines Systems zu erzeugen. Hierzu wird die Transitionsanalyse auf alle Zustände angewandt. Das Ergebnis dieser Analyse, das sogenannte *Envisionment* [de Kleer, Brown, 84] kann als Zustandsdiagramm repräsentiert werden, das die konsistenten Gerätezustände und alle möglichen Übergänge zwischen ihnen beinhaltet. Abb. 12.8 zeigt ein mögliches Diagramm für eine an einer Feder befestigte Masse.

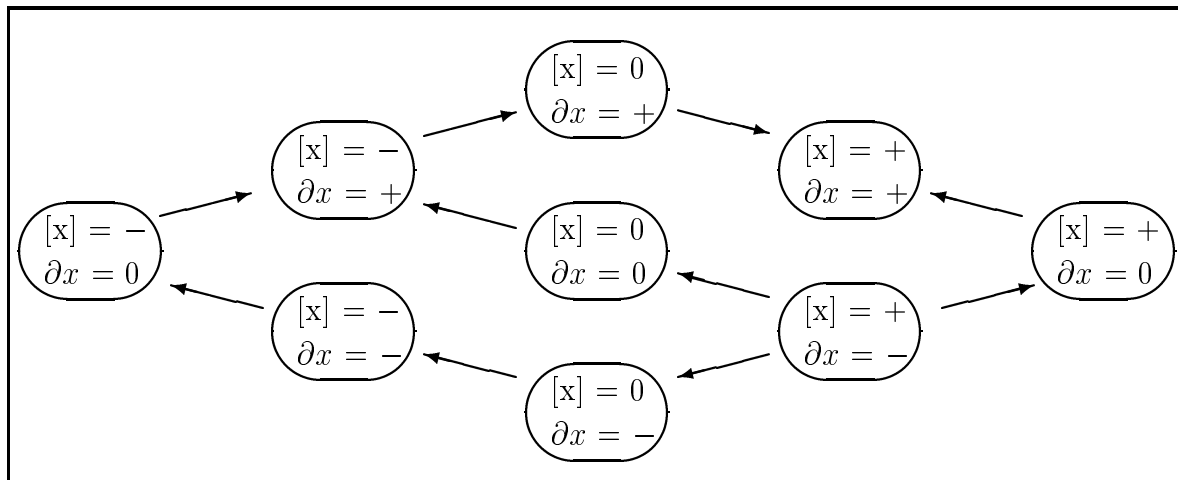


Abbildung 12.8: Zustandsdiagramm für eine Masse an einer Feder, wobei X die Entfernung der Masse von der Ruheposition der Feder und \dot{X} deren Ableitung ist.

12.3.5.4 Vergleichende Analyse

Eine weitere Aufgabe sind vergleichenden Analysen (comparative analysis, [Weld, 88; Neitzke, Neumann, 94; de Jong, 99]. Dabei wird der qualitative Effekt einer Änderung eines Ausgangsparameters oder eines Modells (z.B. aufgrund einer Störung) relativ zu der Ausgangssituation bestimmt. Das Ergebnis solcher Analysen ist oft der Art, daß etwas „wächst“, „schneller abläuft“ oder „länger dauert“. Um solche Ergebnisse liefern zu können, müssen die entsprechenden Verfahren in der Lage sein, qualitativ über die Auswirkungen von Effekten über die Zeit schließen zu können (qualitative Integration).

12.3.6 Weitere Modellierungsaspekte

12.3.6.1 Multiple Modelle und Modelltransformation

Verschiedene Aufgabenstellungen und Systeme können die Berücksichtigung unterschiedlicher Phänomene, Abstraktionsebenen und Vereinfachungen erfordern. Es gibt eine Reihe von Arbeiten zu Problemen

- der Erzeugung und Transformation unterschiedlicher Modelle [Weld, Addanki, 92; Heller, Struss, 96; Sachenbacher, Struss, 00],
- der Darstellung und Verwaltung der Beziehungen multipler Modelle [Addanki *et al.*, 91; Struss, 92] und
- der Erzeugung aufgabenspezifischer Modelle durch Auswahl und Komposition von Fragmenten [Falkenhainer, Forbus, 91; Iwasaki, 92; Rickel, Porter, 94; Nayak, 95].

Mit dem wachsenden Eindringen von modellbasierten Lösungen in industrielle Anwendungen wächst die Bedeutung der automatischen Modellierung und Verwaltung von Modellen, denn der Nutzen dieser Lösungen kann durch hohe Aufwände für die Modellierung und eingeschränkte Wiederverwendung der Modelle bedroht werden.

12.3.6.2 Räumliche und zeitliche Aspekte

Obwohl Raum und Zeit offenkundig von zentraler Bedeutung für die Modellierung sind, gehen wir an dieser Stelle nicht näher darauf ein. Wir verweisen auf Kapitel 10, fügen aber die Feststellung hinzu, daß die geschilderten Methoden und Systeme der qualitativen Verhaltensmodellierung zumeist über wenig mächtige Repräsentationen und Inferenzen über räumliche und zeitliche Beziehungen verfügen und umgekehrt die in diesen Spezialgebieten produzierten Resultate den Bezug zur Verhaltensmodellierung wenig entwickelt haben.

12.4 Modellbasierte Diagnose

In diesem Abschnitt stellen wir Theorien, Techniken und Implementierungen modellbasierter Diagnose genauer vor. Daß wir gerade Diagnose als Anwendungsgebiet herausgreifen, ist nicht zufällig. Zwar haben wir in Abschnitt 2 gezeigt, daß die der Diagnose zugrunde liegenden Kernaufgaben, Konsistenzprüfung und Modellrevision, auch die Grundlage für die Lösung anderer Aufgabenstellungen bieten. Dennoch ist modellbasierte Diagnose sowohl theoretisch als auch im Hinblick auf industrielle Anwendung am weitesten fortgeschritten. Daß auch dies wiederum kein Zufall ist, kann man verstehen, wenn man genauer charakterisiert, auf welche Art von Diagnoseaufgaben diese Aussage zutrifft: nämlich Identifikation unabhängig voneinander versagender Komponenten in Geräten mit wohldefinierter Struktur. In diesem Falle ist eben der Raum der Modellrevision durch die Menge der gegebenen Komponenten und ggf. ihrer betrachteten Fehler sehr klar und eng charakterisiert. Ferner gibt es in Form eines Minimalitätsprinzips (die Suche nach kleinsten oder plausibelsten Mengen von Fehlern) eine sehr wirksame Kontrollheuristik. Beim Entwurf hingegen, soweit es sich nicht um Routineentwurf handelt, ergibt sich durch praktisch kaum beschränkte die Struktur verändernde Revisionen ein wesentlich komplexerer und weniger stark strukturierter Suchraum. Gleichzeitig erkennt man, daß durchaus Diagnoseaufgaben mit anderer Charakteristik existieren, die wesentlich offener und z.T. in ihren Grundlagen nicht gelöst sind, worauf wir auch kurz eingehen werden. Und selbst die obige Aussage über den Reifegrad komponentenorientierter Diagnose muß man weiter diskutieren.

12.4.1 Diagnose als Arbeitsprozeß

In Abschnitt 12.4 haben wir den Kern einer Lösung der Diagnoseaufgabe (für ein korrekt entworfenes System) als Übergang von einem mit den Beobachtungen inkonsistenten Modell zu konsistenten Modellen charakterisiert:

$$\text{MODEL}_0 \cup \text{OBS} \vdash \perp \quad \rightsquigarrow \quad \text{MODEL}_1 \cup \text{OBS} \not\vdash \perp .$$

Diagnose in diesem Sinne ist also die Erzeugung von Fehlerhypothesen aus einer gegebenen Menge von Beobachtungen. In der Tat stellt dies nur einen kleinen Schritt in einer praktischen Diagnoseaufgabe dar, etwa bei der Fehlersuche und Reparatur eines defekten Fahrzeugs in einer Werkstatt. Dieser Schritt ist einer in der *gedanklichen* Tätigkeit des Diagnostikers, im Schlußfolgern. Dessen gesamte Tätigkeiten und die dabei auftretenden Probleme beziehen sich aber oft überwiegend (vor allem zeitlich betrachtet) auf andere Aufgaben:

- *Testen*: Beobachtungen sind (mit Ausnahme vorhandener Kundenbeschwerden oder initialer Alarmmeldungen) normalerweise nicht einfach gegeben, sondern müssen erhoben werden. Weil dies mit z.T. erheblichem Aufwand verbunden ist (z.B. wenn das Gerät zerlegt werden muß), muß dies möglichst planmäßig und (kosten-)günstig geschehen.
- *Therapie (Reparatur)*: Das Erzeugen von Diagnosehypothesen ist für sich genommen kein Ziel, sondern dient der (evtl. eingeschränkten) Wiederherstellung der Funktionalität des Gerätes. Erst von hieraus bestimmt sich der Ablauf der Hypothesengenerierung und deren Terminierung, insbesondere wenn keine eindeutige Hypothese erzeugt werden kann.

Eine Systematisierung des Diagnoseprozesses und der Entwurf von Werkzeugen zu seiner Unterstützung müssen ihn als einen komplexen *Arbeitsprozeß* reflektieren. Selbst wenn, wie etwa bei der On-Board-Diagnose von Fahrzeugen, kein menschlicher Agent direkt beteiligt ist, so ist Diagnose allgemein wie folgt zu charakterisieren:

Diagnose ist ein planmäßiger, kostenorientierter Prozeß des Einwirkens auf ein System und der Beobachtung seines Verhaltens mit dem Zweck, die Abweichung von einem Zielverhalten beseitigen oder minimieren zu können.

Ein wirksamer Diagnoseagent, gleichgültig ob menschlich oder ein Computerprogramm, ist also in dem in Abb. 12.9 dargestellten Zusammenhang zu sehen, und dasselbe gilt für seine Werkzeuge. Es gibt dabei nicht *die* Diagnoseaufgabe, sondern sehr unterschiedliche Ausprägungen dieser Grundstruktur. Die On-Board-Diagnose z.B. hat aufgrund der Signale einer fest vorgegebenen Menge von Sensoren eine Reaktion auszuwählen (vom Aufleuchten der Warnlampe über Notfahrprogramme bis zum Abschalten des Motors), die die Nutzung des Fahrzeugs möglichst wenig beeinträchtigt, aber Risiken vermeidet. Die Werkstattdiagnose eines defekten Fahrzeugs hingegen basiert wesentlich auf der Erhebung zusätzlicher Beobachtungen über das Fahrzeug, ggf. nach Demontageschritten, und dem Ersetzen defekter Komponenten mit dem Ziel, die Funktionsfähigkeit vollständig wiederherzustellen.

Genauer analysiert sind folgende Schritte potentielle Elemente eines konkreten Diagnoseprozesses:

- *Aktionen*:
 - *Einwirkung* durch Modifikation

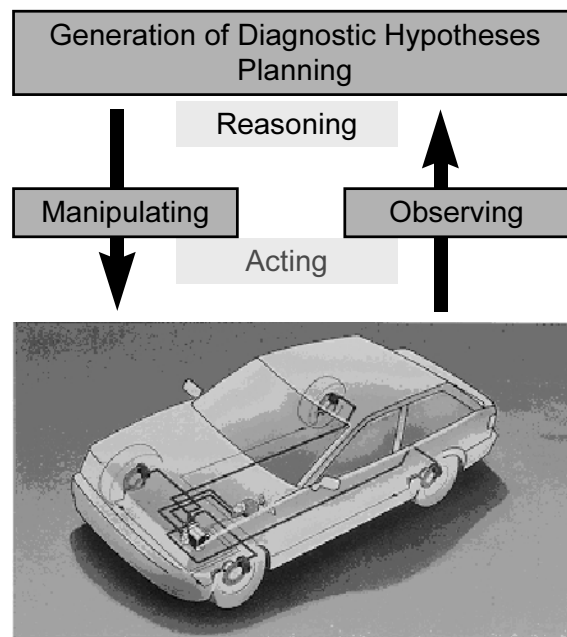


Abbildung 12.9: Diagnose als Arbeitsprozeß

- * der *Systemstruktur*: Hinzufügen und Entfernen von Komponenten, Objekten oder Verbindungen, z.B. Zerlegung oder Demontage von Teilen, Anschluß von Meßgeräten, Überbrücken, Isolieren, Ersetzen von Komponenten
- * des *Systemzustands*: Festlegen und Ändern von Systemgrößen, z.B. Eingabe von Werten, Ändern von Zuständen, etwa Schalterpositionen
- *Beobachtung* (ohne Einwirkung)
 - * der *Systemstruktur*: z.B. Sichtprüfung von Verbindungen
 - * des *Systemverhaltens*: z.B. Ablesen von Meßgeräten, visuelle und akustische Eindrücke
- *Inferenzen*
 - *Information* ableiten: über
 - * das *System*: Hypothesen bestätigen, falsifizieren oder vergleichen über gegenwärtige, vergangene oder künftige
 - *Systemstruktur*, z.B. vorhandene Objekte und Prozesse, fehlerhafte Komponenten und Verbindungen
 - *Systemverhalten*, z.B. Systemgrößen, interne Zustände, unterschiedliche Auswirkung von Fehlern
 - * den *Problemlöseprozeß*:
 - *Problemlösungszustand*, z.B. verbleibende Fehlerhypothesen, mögliche Therapien, erfüllte Ziele, vorhandene Risiken, Kosten von Plänen
 - *Ziele*, z.B. Einhaltung von Grenzwerten, Vergleich mit erwartetem Resultat oder Zeitverhalten
 - *Planung*
 - * *Ziele* modifizieren und präzisieren,
 - * *Aktionen* und Aktionssequenzen auswählen.

Die modellbasierten Methoden greifen natürlich zunächst bei der ersten Klasse von Inferenzen: Modellkomposition und -revision widmen sich der Bestimmung der Systemstruktur, Vorhersage und Konsistenzprüfung liefern Hypothesen über das Systemverhalten. Sie liefern ferner Aussagen über den Problemlösungsprozeß (etwa in Gestalt von ermittelten Diagnosehypothesen oder geeigneten Aktionen) und die Einhaltung von Zielen, soweit diese verhaltensbezogen (im Gegensatz z.B. zu kostenorientiert) sind. Letzteres benötigt im allgemeinen eine Brücke zwischen funktionaler und Verhaltensbeschreibung. Dies geschieht in einer Weise, die durch den Abstraktionsgrad des Systemmodells gegeben ist und die normalerweise die praktischen Bedingungen und Konsequenzen nicht beinhaltet.

Es wird deutlich, wie verkürzt „klassische“ Sichten der KI auf „Diagnose als Klassifikation“ (vor allem bei Expertensystemen) waren. Vor allem, wenn man berücksichtigt, daß die realen Kosten mit den *Aktionen* verbunden sind, wird deutlich, daß viele Diagnoseansätze, die sich auf die Erzeugung von Fehlerhypothesen aus vorliegenden Beobachtungen beschränken, oft keinen oder nur geringen praktischen Nutzen haben. Dieser Arbeitsschritt ist natürlich dennoch ein wesentlicher Kern der meisten Lösungsansätze, und die Arbeit an modellbasierten Systemen hat hier wesentliche Durchbrüche erzielt, die im folgenden Abschnitt 12.4.2 vorgestellt werden. Daneben gibt es fundierte Verfahren für die Gewinnung nützlicher Beobachtungen (12.4.3) und einige Ansätze für die Entwicklung von Therapievorschlügen (12.4.4), allerdings zumeist unter Vernachlässigung der praktischen Aspekte von Aktionen, ihrer Bedingungen und Kosten.

12.4.2 Konsistenzbasierte Diagnose

12.4.2.1 Detektion von Verhaltensdiskrepanzen

Der erste Schritt in einem Diagnoseprozeß ist natürlicherweise die Feststellung, daß überhaupt eine signifikante Abweichung vom Zielverhalten eines Systems aufgetreten ist, also die *Fehlerdetektion*. Ebenso offenkundig bedarf es dazu einer gewissen Menge initialer Beobachtungen, OBS, sowie einer Charakterisierung des Ziels, GOALS. Aber nur im Sonderfall, nämlich wenn sich die Zielspezifikation und die Beobachtungen auf dieselben Größen beziehen, läuft dies auf eine Konsistenzprüfung

$$\text{OBS} \cup \text{GOALS} \stackrel{?}{\vdash} \perp$$

hinaus. Im allgemeinen ist Wissen über Aufbau und Verhalten des Systems nötig, um von den Beobachtungen auf die Einhaltung oder Verletzung von Zielen zu schließen:

$$\text{MODEL} \cup \text{OBS} \cup \text{GOALS} \stackrel{?}{\vdash} \perp.$$

Z.B. läßt sich das Emissionsverhalten eines Dieselmotors während der Fahrt nicht unbedingt direkt messen, sondern muß aus erhobenen Sensor- und Steuergerätedaten über Luftmasse und -temperatur, Einspritzmenge etc. ermittelt werden (vgl. [Sachenbacher *et.al.*, 00]). Wie in diesem Beispiel ist oft das Zielverhalten nicht einfach durch die Produktion bestimmter Ausgabegrößen definiert, sondern auch darüber, *wie* diese erzielt werden. Dies ist in allen Fällen relevant (und oft schwierig), wenn automatische Steuerungen auftretende Fehler kompensieren können.

Die Konsistenzprüfung kann durch verschiedene Faktoren nicht-trivial sein:

- *Beeinflussung der Beobachtungen* durch
 - Meßungenauigkeit der Sensoren
 - nicht meßbare externe Störungen (etwa Einfluß des Straßenbelags auf das Bremsverhalten),
- *Unschärfe der Ziele* (Bei welchem Wert ist der Bremsweg länger als normal?)

Als Antwort darauf bieten sich verschiedene Lösungen an:

- Man kann die Verhaltensvorhersage „exakt“ durchführen und die *Unschärfe nur bei der Konsistenzprüfung* berücksichtigen. Das heißt, daß zwei verschiedene Werte einer Variablen nur jenseits einer Toleranzgrenze als inkonsistent betrachtet werden, z.B. gegeben als relative Abweichung,

$$\text{var}=\text{val}_1 \wedge \text{var}=\text{val}_2 \wedge |\text{val}_1 - \text{val}_2| > \Delta * \text{val}_1 \vdash \perp ,$$

absolute Abweichung,

$$\text{var}=\text{val}_1 \wedge \text{var}=\text{val}_2 \wedge |\text{val}_1 - \text{val}_2| > \Delta \vdash \perp ,$$

oder als Abweichung in der Größenordnung (s. Abschnitt 12.3.3):

$$\text{var}=\text{val}_1 \wedge \text{var}=\text{val}_2 \wedge (\text{val}_1 \sim \text{val}_2) \vdash \perp .$$

Dieses Verfahren liegt, angewendet auf die direkt gemessenen Größen, herkömmlichen Prozeßüberwachungssystemen zugrunde. Es ist in diesen wie auch in modellbasierten Anwendungen problematisch, weil in der Regel der Zusammenhang zwischen der Abweichung einer lokalen Größe und einer echten Störung des Gesamtverhaltens von Werten und Abweichungen anderer Größen anhängig und damit gar nicht lokal bestimmbar ist.

- Es ist möglich, die *Unschärfe im Wertebereich* der Systemgrößen und damit auch im Modell ausdrücken, d.h. durch Abstraktion zu *qualitativen Werten*,

$$\text{var}=\text{qval}_1 \wedge \text{var}=\text{qval}_2 \wedge \text{qval}_1 \neq \text{qval}_2 \vdash \perp ,$$

bzw. disjunkte Intervalle gegeben:

$$\text{var}=\text{int}_1 \wedge \text{var}=\text{int}_2 \wedge \text{int}_1 \cap \text{int}_2 = \emptyset \vdash \perp .$$

Dies schöpft die in Abschnitt 12.3 diskutierten Vorteile der qualitativen Verfahren wie Endlichkeit, Effizienz und Natürlichkeit aus, verlangt aber auch die Bestimmung einer Granularität, die dem System und der Diagnoseaufgabe angemessen ist.

Zusammen mit den in Abschnitt 12.3.5 vorgestellten Verfahren zur Verhaltensanalyse können wir auf diesen Grundlagen Lösungen für die Konsistenzprüfung von Modellen (mit Beobachtungen) erzeugen (Abb. 12.10).

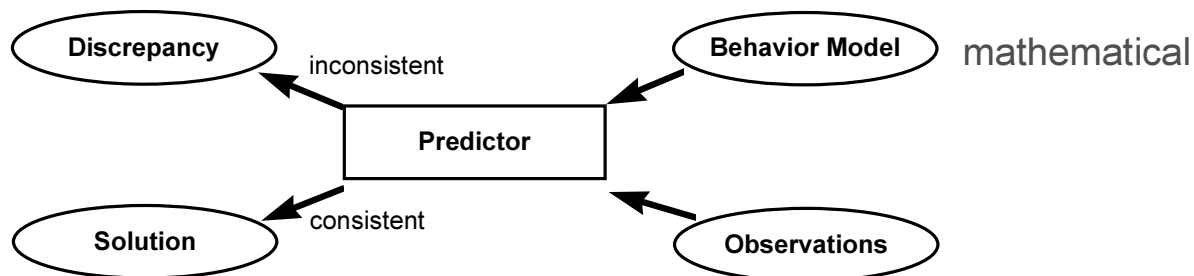


Abbildung 12.10: Modellbasierte Vorhersage und Konsistenzprüfung.

12.4.2.2 Detektion von Komponentenfehlern

Wenn wir uns auf technische Systeme beschränken, so kann man bei der Fehlerdetektion oft davon ausgehen, daß sie im Hinblick auf das Zielverhalten korrekt entworfen wurden (Dies gilt durchaus nicht immer, wie die Beispiele Software und A-Klasse zeigen. Auch Diagnose während der Inbetriebnahme von Anlagen kann sich nicht unbedingt auf diese Annahme stützen). Dies bedeutet, daß das Zielverhalten vom Modell des korrekten Verhaltens impliziert wird,

$$\text{MODEL}_{OK} \Rightarrow \text{GOALS} ,$$

womit sich die Konsistenzprüfung vereinfacht zu

$$\text{MODEL}_{OK} \cup \text{OBS} \stackrel{?}{\vdash} \perp$$

Beschränken wir uns noch weiter, nämlich auf Systeme, die aus Komponenten in einer festen Struktur aufgebaut sind, so ergibt sich das Gesamtmodell aus der Komposition der Modelle des korrekten Verhaltens aller Komponenten $C \in \text{COMPS}$:

$$\text{LIB} \cup \text{STRUCTURE} \cup \{ \text{ok}(C) \mid C \in \text{COMPS} \} \Rightarrow \text{MODEL}_{OK}$$

Fassen wir ([Reiter, 87] folgend) Verhaltensmodelle und Systemstruktur als *Systembeschreibung* SD („system description“) zusammen,

$$\text{SD} = \text{LIB} \cup \text{STRUCTURE} ,$$

so wird die Konsistenzprüfung zu

$$\text{SD} \cup \{ \text{ok}(C) \mid C \in \text{COMPS} \} \cup \text{OBS} \stackrel{?}{\vdash} \perp .$$

Für die Bestimmung geeigneter Therapieschritte benötigt man über Fehlerdetektion hinaus natürlich in der Regel eine nähere Charakterisierung des vorliegenden Fehlverhaltens.

12.4.2.3 Generierung von Komponentenfehler-Hypothesen

Wenn wir annehmen, daß die Störung des Gesamtsystems durch Fehlverhalten seiner Komponenten hervorgerufen wird, so sind Fehlerhypothesen Aussagen über Abweichungen vom korrekten Verhalten der einzelnen Komponenten. Jeder Komponente C_i (oder jedem Komponententyp) werde eine Menge möglicher Verhaltensweisen $\text{modes}(C_i)$ zugeordnet. Dann geht es in dieser Art komponentenorientierter Diagnose darum, den einzelnen Komponenten mögliche Verhaltensmodi zuzuordnen.

Definition (Moduszuweisung)

Sei $\text{COMPS}' \subseteq \text{COMPS}$:

$$\bigwedge_{C_i \in \text{COMPS}'} m_{j_i}(C_i), \text{ mit } m_{j_i} \in \text{modes}(C_i).$$

ist eine Moduszuweisung. Sie heißt vollständig, wenn $\text{COMPS}' = \text{COMPS}$ gilt.

Während $\text{ok}(C_i)$ stets in $\text{modes}(C_i)$ enthalten sein sollte, können die restlichen Charakterisierungen in der Abstraktionsweise stark variieren, wie wir im folgenden sehen werden. Haben wir in Abschnitt 2 Diagnose definiert als Suche nach einem Systemmodell, das im Einklang mit den Beobachtungen ist, so sind unter unseren Annahmen die gesuchten Modelle durch (vollständige) Moduszuweisungen charakterisiert, und wir können definieren:

Definition (Diagnose)

Eine Moduszuweisung MA ist eine (konsistenzbasierte) Diagnose für eine Systembeschreibung SD und eine Menge von Beobachtungen OBS, falls sie konsistent mit ihnen ist:

$$SD \cup \{MA\} \cup OBS \not\vdash \perp$$

Leitet MA die Beobachtungen aus dem Model ab,

$$SD \cup \{MA\} \vdash OBS$$

so heißt sie abduktive Diagnose.

Die so definierten Diagnosen werden oft als *Diagnosekandidaten* bezeichnet, da sie im Prozeß der Diagnose eventuell aufgrund weiterer Beobachtungen verworfen werden könnten. Die abduktiven Diagnosen bilden offenbar eine Teilmenge der konsistenzbasierten. Verlangt man nur für eine Teilmenge der Beobachtungen, daß sie aus dem Modell ableitbar sind (also z.B. die erhöhten Abgaswerte als „Output“, aber nicht den „Input“ der Gaspedalstellung), so erhält man ein Spektrum von Diagnosen (vgl. [Console, Torasso, 91]). Ansätze und Probleme abduktiver Diagnose werden u.a. in [Poole, 89] vorgestellt. Wir konzentrieren uns im folgenden auf konsistenzbasierte Verfahren. Die Erzeugung von Diagnosekandidaten kann unterschiedlichen Zielen dienen:

- *Fehlerlokalisierung*, d.h. Erkennung *welche Komponenten* C_i defekt sind, und
- *Fehleridentifikation*, d.h. Erkennung, *welche Fehler* $m_j(C_i) \in \text{modes}(C_i)$ vorliegen.

12.4.2.4 Lokalisierung von Komponentenfehlern

Man kann also Fehlerlokalisierung als Spezialfall $\text{modes}(C) = \{\text{ok}(C), \neg\text{ok}(C)\}$ der Identifikation auffassen. Ferner impliziert jede Fehleridentifikation eine Fehlerlokalisierung, wenn alle von „ok“ verschiedenen Verhaltensweisen einer Komponente inkorrektes Verhalten bedeuten:

$$\forall \text{mode}(C) \in \text{modes}(C) \setminus \{\text{ok}(C)\} \text{ mode}(C) \Rightarrow \neg \text{ok}(C).$$

Definition (Fehlerlokalisierung) $\text{FAULTY} \subset \text{COMPS}$ ist eine Fehlerlokalisierung für SD und OBS, wenn die Moduszuweisung

$$\bigwedge_{C \in \text{FAULTY}} \neg \text{ok}(C) \wedge \bigwedge_{C \in \text{OK}} \text{ok}(C)$$

mit $\text{OK} = \text{COMPS} \setminus \text{FAULTY}$ eine Diagnose für SD und OBS ist. Sie heißt minimal, falls keine echte Teilmenge von FAULTY eine Fehlerlokalisierung ist.

Dies ist die „klassische“ Definition, wie sie der General Diagnosis Engine ([de Kleer, Williams, 87]) und der Theorie von [Reiter, 87] und [de Kleer, *et.al.*, 92] zugrunde liegt. Wir betonen noch einmal, daß sie nur unter einer ganzen Reihe von einschränkenden Annahmen über das System, die austretenden Fehler und die Randbedingungen Sinn macht:

- *Korrektter Entwurf*: Das System wurde korrekt entworfen, d.h. der Entwurf ist im Einklang mit den Zielen.
- *Fixierte Konstituenten*: es setzt sich aus (bekannten) Komponenten zusammen.
- *Bekannte Struktur*: die Interaktion der Komponenten ist bekannt und fixiert.

- *Keine Strukturfehler*: insbesondere ändert sich die Struktur auch im Fehlerfall nicht (wie etwa durch Nebenschlüsse in einer Schaltung).
- *Nur Komponentenfehler*: alle auftretenden Fehler sind Fehlverhalten einzelner Komponenten. Die Kenntnis der defekten Komponenten ist eine hinreichende Basis für die Therapie (also z.B. Therapie durch Komponentenaustausch), oder sie ist nur Ausgangspunkt für eine anschließende Fehleridentifikation.

Diese Voraussetzungen kennzeichnen tatsächlich die am besten beherrschte Klasse von Anwendungen. Die obigen Definitionen von Diagnosen und Fehlerlokalisierung geben noch keinen Hinweis darauf, wie diese theoretisch charakterisiert und berechnet werden können. Wir haben allerdings bereits einen speziellen Zugang dazu: wir haben grundsätzlich Diagnose als einen Modellrevisionsprozeß ausgehend vom Modell des korrekten Verhaltens betrachtet, und unter den oben angenommenen Voraussetzungen lassen sich alle Modellrevisionen auf Revisionen von Moduszuweisungen zurückführen, die damit einen ganz speziellen und wohlstrukturierten Suchraum aufspannen.

Ein Weg zur Lösung wäre, alle möglichen Moduszuweisungen, evtl. geordnet nach Wahrscheinlichkeit, auf Konsistenz mit den Beobachtungen zu überprüfen. Er wurde tatsächlich in [Davis, 84] beschritten, indem jeweils das Modell einer Komponente „suspendiert“ und die Konsistenz des verbleibenden Gesamtmodells überprüft wurde („constraint suspension“). Dies verbietet sich natürlich spätestens, wenn nicht nur Einfachfehler betrachtet werden. Es gibt aber Möglichkeiten, die gewünschten Diagnosen präzise zu charakterisieren und fokussierter zu berechnen. Die Grundlage dafür läßt sich intuitiv leicht nachvollziehen. Wenn trotz Betätigung des Anlassers der Motor eines Kraftfahrzeuges nicht anspringt, so werden wir einen relativ geringen Teil der Subsysteme und Komponenten als mögliche Verursacher des Fehlers in Erwägung ziehen. Batterie, elektrische Verbindungen zum Anlasser, Kraftstoffzufuhr, Anlasser etc. erscheinen, im Gegensatz etwa zur Bremshydraulik oder der Scheibenwaschanlage, verdächtig – weil sie eben, anders als die unverdächtigen Komponenten, in intaktem Zustand gemeinsam das intendierte, korrekte Verhalten „Motor springt an“ gewährleisten würden.

Auf die modellbasierte Diagnose übertragen bedeutet dies, daß nicht einfach das Gesamtmodell des Systems (hier: des Fahrzeugs) mit den Beobachtungen inkonsistent ist, sondern schon Teilmodelle. Existieren mehrere solche inkonsistenten Teilmodelle, so werden Fehlerhypothesen sich auf deren Durchschnitt beziehen oder, falls dieser leer ist, auf Kombinationen von Fehlern aus disjunkten dieser Teilmodelle. Eine (partielle) Moduszuweisung, die einem solchen Teilmodell zugrunde liegt, beinhaltet offenbar wesentliche Informationen für eine *gezielte* Modellrevision.

Definition (Konflikt) Seien $COMPS' \subset COMPS$ und

$$MA = \bigwedge_{C_i \in COMPS'} m_{j_i}(C_i), .$$

eine Moduszuweisung so, daß

$$SD \cup \{MA\} \cup OBS \vdash \perp.$$

Dann heißt die Negation von MA,

$$\bigvee_{C_i \in COMPS'} \neg m_{j_i}(C_i)$$

Konflikt. Ein Konflikt heißt minimal, wenn es keinen anderen Konflikt gibt, der ihn impliziert. Er heißt positiv, wenn für alle $m_{ji}(C_i) = \text{ok}(C_i)$ gilt.

Positive Konflikte beinhalten also ausschließlich disjunktive Hypothesen über defekte Komponenten: mit Sicherheit ist zumindest eine der erwähnten Komponenten defekt. Bei minimalen Konflikten ist jede Einschränkung der entsprechenden Moduzuweisung konsistent mit $\text{SD} \cup \text{OBS}$. In diesem Sinne stellen minimale Konflikte die schärfsten Beschränkungen für die möglichen Diagnosen dar, und die nicht-minimalen liefern keine zusätzliche Information. Stärker noch: das folgende Theorem aus [de Kleer, *et.al.*, 92] besagt, daß sie die *gesamte* diagnostische Information verkörpern, die in der Systembeschreibung und den Beobachtungen zusammen enthalten ist und daher $\text{SD} \cup \text{OBS}$ in der Definition der Diagnosen ersetzen können:

Theorem

Sei MIN-CONFLICTS die Menge aller minimalen Konflikte für $\text{SD} \cup \text{OBS}$. Eine vollständige Moduzuweisung MA ist eine Diagnose für $\text{SD} \cup \text{OBS}$ genau dann, wenn sie mit den minimalen Konflikten konsistent ist:

$$\text{MIN-CONFLICTS} \cup \{\text{MA}\} \not\vdash \perp.$$

Für die Fehlerlokalisierung geht es nur darum, festzustellen, welche Komponenten defekt sind. Das bedeutet, daß für alle Komponenten C nur

$$\text{modes}(C) = \{\text{ok}(C), \neg\text{ok}(C)\}$$

und die entsprechenden Moduzuweisungen interessant sind. Das folgende Theorem aus [de Kleer, *et.al.*, 92] faßt die Intuition, daß die für die Fehlerlokalisierung wesentliche Information in den *positiven* Konflikten liegt, d.h. in den Disjunktionen ausschließlich über defekte Komponenten.

Theorem

FAULTY \subset COMPS ist eine minimale Fehlerlokalisierung genau dann, wenn die Moduzuweisung

$$\bigwedge_{C \in \text{FAULTY}} \neg\text{ok}(C)$$

ein Prim-Implikant der positiven minimalen Konflikte ist.

Definition (Prim-Implikant)

Eine konsistente Konjunktion von Literalen, CC, heißt Implikant einer Menge Th von propositionalen Formeln, wenn CC jede einzelne Formel ableitet:

$$CC \vdash Th$$

CC heißt Prim-Implikant von Th, wenn es keine echt in CC enthaltene Konjunktion gibt, die ebenfalls ein Implikant von Th ist:

$$\forall CC' \subset CC : CC' \not\vdash Th$$

Damit ist eine präzise logische Charakterisierung von minimalen Fehlerlokalisierungen gegeben, eine wichtige Voraussetzung dafür, Korrektheit, Vollständigkeit, Voraussetzungen und Eigenschaften von Verfahren zu ihrer Berechnung zu untersuchen. Der Bezug auf die minimalen Fehlerlokalisierungen ist gewöhnlich keine wesentliche Beschränkung: wenn

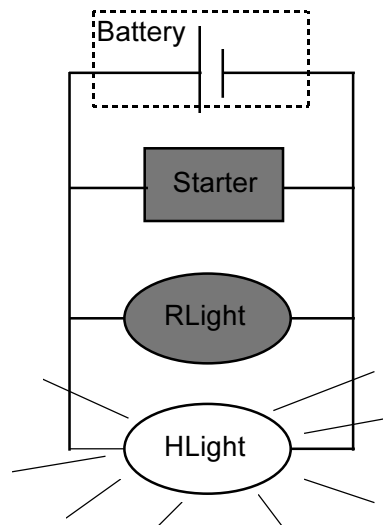


Abbildung 12.11: Eine Fahrzeugdiagnose: Anlasser und Rücklicht funktionieren nicht, die Scheinwerfer leuchten

der Ausfall einer gewissen Menge von Komponenten hinreicht, um eine Abweichung vom erwarteten Verhalten zu erklären, besteht auch intuitiv kein Zwang, zusätzliche Komponenten zu verdächtigen. (Man erinnere sich daran, daß minimale Fehlerlokalisierungen über minimale Mengen, nicht über deren Kardinalität definiert sind).

Es ist an der Zeit, ein illustratives Beispiel zu betrachten. Nehmen wir für das Fahrzeug die in Abb. 12.11 dargestellte Situation an: Der Anlasser (Starter) funktioniert nicht, das Rücklicht (RLight) leuchtet nicht, aber die Frontscheinwerfer (HLight) strahlen hell.

Die Modelle des korrekten Verhaltens der beteiligten Komponententypen sind die offenkundigen: eine Batterie liefert Spannung, Drähte sind elektrische Verbindungen, und bei Anlegen einer Spannungsdifferenz (und nur dann) leuchten Glühlampen und ist der Anlasser aktiv. Aus den Modellen für die Batterie, die Drähte $Wire_1$, $Wire_2$ und den Anlasser sagt das Modell für das korrekte Gesamtverhalten voraus, daß letzterer aktiv sein sollte, was der Beobachtung widerspricht:

$$\begin{aligned} & \text{ok}(\text{Battery}) \wedge \text{ok}(\text{Wire}_1) \wedge \text{ok}(\text{Wire}_2) \wedge \text{ok}(\text{Starter}) \Rightarrow \text{active}(\text{Starter}) \\ & \text{OBS} \Rightarrow \neg \text{active}(\text{Starter}). \end{aligned}$$

Daraus folgt

$$\text{SD} \cup \text{OBS} \cup \{ \text{ok}(\text{Battery}) \wedge \text{ok}(\text{Wire}_1) \wedge \text{ok}(\text{Wire}_2) \wedge \text{ok}(\text{Starter}) \} \vdash \perp,$$

d.h. wir erhalten einen Konflikt

$$\text{Conflict}_1 \equiv \neg \text{ok}(\text{Battery}) \vee \neg \text{ok}(\text{Wire}_1) \vee \neg \text{ok}(\text{Wire}_2) \vee \neg \text{ok}(\text{Starter}),$$

der auch minimal ist, da keine Teilmenge der Komponentenmodelle einen Widerspruch zu den Beobachtungen ableitet. In ähnlicher Weise ergibt sich

$$\begin{aligned} \text{Conflict}_2 \equiv & \neg \text{ok}(\text{Battery}) \vee \neg \text{ok}(\text{Wire}_1) \vee \neg \text{ok}(\text{Wire}_2) \vee \\ & \neg \text{ok}(\text{Wire}_3) \vee \neg \text{ok}(\text{Wire}_4) \vee \neg \text{ok}(\text{RLight}). \end{aligned}$$

Das Modell kann aber weitere Ableitungen vollziehen, nämlich daß aus dem Leuchten der Frontscheinwerfer eine Potentialdifferenz folgt, die bei korrekten Drähten $Wire_5$, $Wire_6$ auch am Rücklicht anliegt, woraus deren Leuchten zu schließen ist, ebenfalls im Widerspruch zu den Beobachtungen:

$$\text{ok(HLight)} \wedge \text{ok(Wire}_5) \wedge \text{ok(Wire}_6) \wedge \text{ok(RLight)} \Rightarrow \text{lit(RLight)}$$

$$\text{OBS} \Rightarrow \neg \text{lit(RLight)},$$

mithin erhält man

$$\text{Conflict}_3 \equiv \neg \text{ok(HLight)} \vee \neg \text{ok(Wire}_5) \vee \neg \text{ok(Wire}_6) \vee \neg \text{ok(RLight)}.$$

Ganz analog ergibt sich

$$\text{Conflict}_4 \equiv \text{ok(HLight)} \vee \neg \text{ok(Wire}_5) \vee \neg \text{ok(Wire}_6) \vee \\ \neg \text{ok(Wire}_3) \vee \neg \text{ok(Wire}_4) \vee \neg \text{ok(Starter)}.$$

Dies illustriert noch einmal, daß das Verhaltensmodell nicht (kausal vorwärts) „simuliert“, sondern alle möglichen Inferenzen durchführt. Die Moduszuweisung

$$\neg \text{ok(RLight)} \wedge \neg \text{ok(Starter)}$$

impliziert offenbar alle vier minimalen Konflikte und ist selbst minimal, also ein Prim-Implikant der minimalen Konflikte. Im Einklang mit unseren Erwartungen ist also

$$\{\text{RLight}, \text{Starter}\}$$

eine Fehlerlokalisierung.

Dieses Beispiel demonstriert darüber hinaus zwei wichtige Eigenschaften konsistenzbasierter Diagnose:

- Die Diagnose von *Mehrfachfehlern* ist möglich, was einen erheblichen Vorteil gegenüber assoziationsbasierten Diagnosesystemen darstellt, die an der Aufzählung der Symptome wegen der fehlenden empirischen Grundlage oder schlicht an der Komplexität scheitern.
- Es ist möglich, Diagnosehypothesen allein auf der Grundlage von *Modellen des korrekten Verhaltens* der Komponenten zu gewinnen. Wenn, wie im Beispiel, keine Voraussetzungen über vorliegende Fehler gemacht werden, können somit auch unbekannte Fehler lokalisiert werden, ein Fortschritt gegenüber empiriebasierten (etwa fallbasierten) Ansätzen.

Die letzte Eigenschaft gründet sich theoretisch darauf, daß die (minimalen) Diagnosen durch die (minimalen) positiven Konflikte bestimmt sind: diese haben ja die Form

$$\bigvee_{C_i \in \text{COMPS}'} \neg \text{ok}(C_i)$$

und entstehen aus Moduszuweisungen

$$\bigwedge_{C_i \in \text{COMPS}'} \text{ok}(C_i)$$

mit den Beobachtungen und dem Modell des korrekten Verhaltens. Gibt es im Modell keine Beschränkung des Fehlverhaltens, so können Fehlerannahmen $\neg \text{ok}(C)$ keine Inkonsistenzen erzeugen, und es existieren ausschließlich positive Konflikte. Dann gilt:

Theorem

Für jede Fehlerlokalisierung $\text{FAULTY} \in \text{COMPS}$ ist jede Obermenge $\text{FAULTY}' \subset \text{FAULTY}$ ebenfalls eine Fehlerlokalisierung genau dann, wenn alle Konflikte von $\text{SD} \cup \text{OBS}$ positiv sind.

In diesem Falle charakterisieren die minimalen Fehlerlokalisierungen in eindeutiger und kompakter Weise die Menge aller Fehlerlokalisierungen, nämlich als untere Schranke im Verband der Teilmengen von COMPS.

Es ist wichtig, sich vor Augen zu führen, was das Fehlen einer Beschränkung des Fehlverhaltens für das Modell bedeutet. Dies ist *nicht* gleichbedeutend damit, daß *überhaupt kein* Fehlermodell vorliegt. Einer Komponente C kann durchaus eine Menge von Fehlverhaltensmodi $\{\text{fault}_i(C)\}$ zugeordnet sein, von der aber nicht Vollständigkeit angenommen wird, d.h. es gilt *nicht*

$$\neg \text{ok}(C) \Rightarrow \bigvee_i \text{fault}_i(C).$$

Wenn weiter SD, wie es dem Prinzip der kontextfreien, kompositionalen Modellierung entspricht, keine Beschränkungen auf Kombinationen von Verhaltensmodi verschiedener Komponenten enthält (etwa in der Art „ $\text{ok}(\text{HLight}) \Rightarrow \text{ok}(\text{RLight})$ “), dann sind alle minimalen Konflikte positiv und $\neg \text{ok}(C)$ ist nicht widerlegbar. Anders gesagt: die Verwendung von Fehlermodellen ist für die Fehlerlokalisierung wirkungslos, wenn nicht zusätzliche Beschränkungen hinzutreten, etwa basierend auf Wahrscheinlichkeitskriterien oder Inferenzen. Solch eine Inferenz wird z.B. durch die Annahme über die Vollständigkeit der Fehlermodelle bereitgestellt, die in [Struss, Dressler, 89] als Prinzip der „physikalischen Negation“ eingeführt wurde. Sie erlaubt, durch

$$\bigwedge_i \neg \text{fault}_i(C) \Rightarrow \text{ok}(C)$$

auf die Korrektheit einer Komponente zu schließen, d.h. $\neg \text{ok}(C)$ auszuschließen, wenn alle Fehlermodelle inkonsistent sind.

Wir illustrieren anhand des obigen Beispiels, daß dadurch die Menge der minimalen Fehlerlokalisierungen verändert werden kann, und zwar im Einklang mit unseren Intuitionen und in diesem Fall sogar drastisch. Neben der akzeptablen Fehlerlokalisierung $\{\text{RLight}, \text{Starter}\}$ gibt es nämlich noch 21 weitere! Z.B. sind auch

$$\neg \text{ok}(\text{Battery}) \wedge \neg \text{ok}(\text{HLight})$$

und

$$\neg \text{ok}(\text{Wire}_1) \wedge \neg \text{ok}(\text{Wire}_5)$$

Prim-Implikanten der minimalen positiven Konflikte! Solche Fehlerlokalisierungen sind sicherlich unerwünscht, weil im ersten Fall die defekte Batterie zwar den Ausfall von Anlasser und Rücklicht erklären würde, aber mit dem Leuchten der Scheinwerfer unvereinbar ist, wie erst recht die Annahme, diese seien defekt. Dieser Widerspruch aber basiert auf Wissen darüber, welches Verhalten der Komponenten im Fehlerfall möglich oder unmöglich ist, etwa Leuchten der Scheinwerfer ohne anliegende Spannungsdifferenz. Ohne (vollständiges) Modell des Fehlverhaltens kann die konsistenzbasierte Diagnose gar nicht in der Lage sein, den obigen Widerspruch zu erkennen und zu nutzen. Stellt man z.B. Modelle für die Fehlverhalten „Batterie leer“, „Draht unterbrochen“, „Lampe durchgebrannt“, „Starter unterbrochen“ bereit und betrachtet diese als die einzig möglichen, so sind weitere Schlüsse möglich: „Lampe durchgebrannt“ widerspricht dann direkt den Beobachtungen, womit $\text{ok}(\text{HLight})$ abgeleitet wird und sich die Konflikte Conflict_3 und Conflict_4 verkleinern. Es ergibt sich ferner, daß z.B. Wire_5 stromdurchflossen ist, was sein Fehlermodell invalidiert und ihn ebenfalls aus den Konflikten Conflict_3 und Conflict_4 entfernt. Durch ähnliche Inferenzen verbleiben schließlich die beiden positiven minimalen Konflikte

$\neg\text{ok}(\text{RLight})$, $\neg\text{ok}(\text{Starter})$

mit einem einzigen, dem gewünschten, Prim-Implikanten. In GDE^+ ([Struss, Dressler, 89] wird dies realisiert. [Friedrich *et.al.*, 92] schlagen alternativ vor, nicht alle Fehlverhalten, sondern das unter ihnen Unmögliche („physical impossibility“) zu repräsentieren, also etwa

$\neg\text{ok}(\text{HLight}) \Rightarrow \neg\text{lit}(\text{HLight})$

Unser Beispiel illustriert auch den Fall, daß bei Verwendung vollständiger Fehlermodelle eine Obermenge einer Fehlerlokalisierung nicht unbedingt ebenfalls eine Fehlerlokalisierung ist: $\{\text{RLight}, \text{Starter}\}$ läßt sich überhaupt nicht durch Fehler anderer Komponenten erweitern. Die Voraussetzung des obigen Theorems ist ja verletzt, denn $\text{broken}(\text{HLight})$ ist ein minimaler Konflikt, aber kein positiver.

Vollständige Mengen von Fehlermodellen können also die Fehlerlokalisierung wesentlich eingrenzen, indem sie unplausible Fehlerhypothesen ausräumen. Allerdings geht dabei eventuell eine weitere grundlegende Voraussetzung ein, nämlich daß keine *strukturellen* Fehler vorliegen. Im betrachteten Beispiel könnten die Beobachtungen auch dadurch erklärt werden, daß z.B. Wire_1 offen ist und Wire_5 ebenfalls, dabei aber HLights mit dem Pluspol der Batterie kurzschließt. Dies in einem entsprechenden Fehlermodell des Drahtes zu berücksichtigen, ist nicht adäquat, da der Fehler eine Interaktion zweier Komponenten darstellt, und kann, zumindest bei Mehrfachfehlern, zu falschen Resultaten führen. Außerdem ist durch eine entsprechende Lokalisierung nicht unbedingt ein korrekter Hinweis auf die geeignete Therapie gegeben. Wir werden im Abschnitt 12.4.2.8 auf das Problem der strukturellen Fehler zurückkommen.

12.4.2.5 Identifikation von Komponentengehlern

Fehlermodelle werden natürlich nicht nur eingesetzt, um die Lokalisierung zu verbessern, sondern auch, wenn Fehleridentifikation verlangt ist, z.B. um die richtigen Therapiemaßnahmen bestimmen zu können. So hängen die Reaktionen („recovery actions“) eines Fahrzeug-On-Board-Diagnosesystems davon ab, welche *Klasse* von Fehler vorliegt, nicht einfach davon, welche Komponente defekt ist. Man ist also an Diagnosen im Sinne der Definition mit multiplen Fehlverhaltensweisen (statt nur $\neg\text{ok}(C)$) interessiert. Solche Diagnosen für Fehleridentifikation stehen offenbar in einer m:1-Beziehung zu den Fehlerlokalisierungen.

Mit der Betrachtung multipler Verhaltensmodelle wächst der Raum der Moduszuweisungen und damit der möglichen Diagnosen (im obigen, sehr überschaubaren Beispiel sind es schon 2^9). Auch hier ist man in der Regel nicht an beliebigen Diagnosen interessiert, und das Minimalitätsprinzip stellt einen natürlichen Fokus bereit. Meist interessieren nur minimale Diagnosen im folgenden Sinn:

Definition (Minimale Diagnose)

Eine Diagnose für $\text{SD} \cup \text{OBS}$ heißt minimal, wenn die zugehörige Fehlerlokalisierung minimal ist.

Meist reicht auch dieser Fokus nicht aus, denn wenn einer Komponente n Fehlermodelle zugeordnet sind, entsprechen einer Fehlerlokalisierung der Kardinalität k bis zu n^k Moduszuweisungen, die es auf Konsistenz zu prüfen gilt. In aller Regel werden aber die verschiedenen möglichen Fehler nicht als „gleichwertig“ betrachtet. Einige von ihnen sind

häufiger, kritischer oder wahrscheinlicher als andere und werden daher auch im normalen Diagnoseprozeß eher untersucht als andere. Das heißt: Es gibt eine Bewertung der Fehlverhaltensweisen, etwa in numerischer Form (z.B. aufgrund von Fehlerstatistiken). Eine schwächere Form, die „Bevorzugung“ einiger Fehler gegenüber anderen auszudrücken, ist eine (partielle) Ordnung auf den Verhaltensweisen einer Komponente. Ein Ziel ist, die unter Berücksichtigung dieser Ordnung präferierten Diagnosen im Hinblick auf Fehleridentifikation theoretisch zu charakterisieren, wie zuvor die Fehlerlokalisierungen.

Dazu definieren wir:

Definition (Modus-Präferenz)

Sei $\text{modes}(C)$ die Menge der Verhaltensmodi der Komponente C . Eine Modus-Präferenz für C ist eine partielle Ordnung „ $<$ “ auf $\text{modes}(C)$:

$$\geq \subseteq \text{modes}(C_i) \times \text{modes}(C_i)$$

wobei das korrekte Verhalten das maximale und das unbekannte Verhalten das minimale Element ist:

$$\begin{aligned} \forall m_j(C_i) \in \text{modes}(C_i) \setminus \{ok(C_i)\} & : ok(C_i) > m_j(C_i), \\ \forall m_j(C_i) \in \text{modes}(C_i) \setminus \{unknown(C_i)\} & : m_j(C_i) > unknown(C_i), \end{aligned}$$

Dabei ist „ $<$ “ definiert als

$$x > y :\Leftrightarrow x \geq y \wedge \neg(y \geq x).$$

Die Präferenz auf den Verhaltensweisen der einzelnen Komponenten induziert eine partielle Ordnung auf den Moduszuweisungen: Für zwei Zuweisungen

$$\begin{aligned} D &= \{m_{j_i}(C_i)\} \\ D' &= \{m'_{j_i}(C_i)\} \end{aligned}$$

sei

$$D \geq D' \quad :\Leftrightarrow \quad \forall i \ m_{j_i}(C_i) \geq m'_{j_i}(C_i).$$

Diese Ordnung soll nun auch bestimmen, welches die „bevorzugten“ Diagnosen sind (vgl. [Dressler, Struss, 96]):

Definition (Präferierte Diagnose)

Eine Diagnose D ist eine präferierte Diagnose, wenn es keine andere Diagnose D' gibt, die unter der Präferenzordnung echt größer ist, d.h.

$$\forall D' \ D' \geq D \Rightarrow D' = D .$$

Bemerkung

Jede präferierte Diagnose ist eine minimale Diagnose, weil für jede Komponente C $ok(C)$ das maximale Element ist.

Intuitiv besagt die Definition, daß ein bestimmter Fehlermodus $m_j(C_i)$ genau dann in einer Diagnose D auftaucht, wenn

- die Moduszuweisungen, die aus D entstehen, wenn $m_j(C_i)$ durch einen strikt präferierten Modus $m'_{j_i}(C_i) > m_j(C_i)$ ersetzt wird, keine Diagnosen, d.h. inkonsistent mit $SD \cup OBS$, sind, und natürlich
- D eine Diagnose ist, d.h. konsistent mit $SD \cup OBS$.

Dies läßt sich direkt in Default-Logik (siehe Abschn. 7.2.1) ausdrücken. Eine (normale) Default-Regel ist eine Inferenz der Form

$$a : b / b ,$$

die besagt „Falls a gilt und es konsistent ist, b anzunehmen, so gilt b“. Eine Default-Theorie enthält neben einer Menge P von klassischen Formeln eine Menge D von Default-Regeln. Mit ihnen läßt sich aus P eventuell mehr ableiten, als in der deduktiven Hülle

$$\text{Cn}(P) := \{p \mid P \vdash p\}$$

enthalten ist, allerdings in der Regel nicht in eindeutiger Weise. Dies liegt daran, daß mehrere Default-Regeln zwar auf P anwendbar sein können, aber eventuell nicht alle gemeinsam. ein Beispiel dafür wäre die Annahme eines bestimmten Verhaltensmodus einer Komponente per Default, was die Annahme eines anderen, ebenfalls konsistenten Verhaltensmodus ausschließt. Als Folge davon hat eine Default-Theorie (DEF, P) in der Regel mehrere sog. *Extensionen* (statt einer eindeutigen deduktiven Hülle). In der Tat benutzen wir Default-Regeln, um auszudrücken, daß ein Verhaltensmodus $m_j(C_i)$ dann angenommen werden soll, wenn alle seine strikt präferierten Vorgänger

$$\text{pre}_j(C_i) := \{m_k(C_i) \mid m_k(C_i) > m_j(C_i)\}$$

widerlegt wurden, und $m_j(C_i)$ selbst konsistenterweise angenommen werden kann:

$$\bigwedge_{m_k(C_i) \in \text{pre}_j(C_i)} \neg m_k(C_i) \quad : \quad m_j(C_i) / m_j(C_i)$$

Insbesondere wird als Ausgangspunkt korrektes Verhalten angenommen:

$$: \text{ok}(C) / \text{ok}(C)$$

Das folgende Theorem [Dressler, Struss, 96] faßt die Intuition, daß durch diese Präferenz-Defaults die gewünschten präferierten Diagnosen bestimmt sind:

Theorem

Sei $\text{DEF} = \{\text{def}_{ij}\}$ die Menge der Präferenz-Defaults. Eine Diagnose D ist eine präferierte Diagnose genau dann, wenn

$$\text{Cn}(\text{SD} \cup \text{OBS} \cup D)$$

eine Extension der Default-Theorie (DEF, SD \cup OBS) ist.

Wir haben somit eine logische Charakterisierung auch für (präferierte) Diagnosen für Fehleridentifikation. Als Spezialfall, nämlich für $\text{modes}(C) = \{\text{ok}(C), \neg\text{ok}(C)\}$, enthält sie auch eine Charakterisierung, wie sie in [Reiter, 87] für Fehlerlokalisierungen gegeben wurde.

12.4.2.6 Implementierung: Modellrevision als konfliktgetriebene Diagnose

In den vorangegangenen Abschnitten wurden Fehlerlokalisierungen und Diagnosen logisch charakterisiert. Nun soll kurz skizziert werden, wie sie berechnet werden können. Dies liefert uns die in Abschnitt 12.2.7 diskutierte Modellrevision als Softwarekomponente (Abb. 12.12). Dabei können an die Verfahren folgende Anforderungen (mit unterschiedlichem Gewicht in unterschiedlichen Anwendungsaufgaben und -kontexten) gestellt sein:

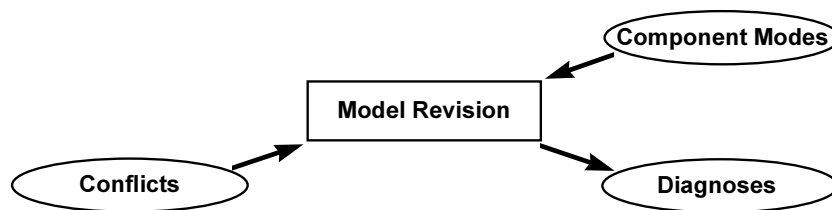


Abbildung 12.12: Modellrevision als Erzeugung von Diagnosehypothesen aus Konflikten

- Sie sollten *inkrementell* arbeiten können im Hinblick auf eine wachsende Menge von Konflikten, hervorgerufen entweder durch weitere Beobachtungen (vgl. Abschnitt 4.3) oder durch eine erweiterte Verwendung des Modells. D.h. die neue Menge von Diagnosehypothesen sollte sich beim Hinzufügen von Konflikten als Modifikation der bisherigen Hypothesen aufgrund der neuen Konflikte ergeben, statt insgesamt neu berechnet werden zu müssen.

Wenn Fehleridentifikation betrieben wird, läßt sich die Erzeugung von Diagnosehypothesen, also die Suche im Raum der Moduszuweisungen, in einem *Hypothesize-and-Test*-Zyklus verstehen. Dieser Raum wächst exponentiell mit der Zahl der Komponenten. Dies erfordert:

- Der Prozeß der Generierung von Moduszuweisungen als Diagnosehypothesen muß *fokussiert* ablaufen, d.h. insbesondere gesteuert durch das Kriterium der Minimalität der Diagnosen sowie ggf. weitere Ordnungen auf den Moduszuweisungen wie Wahrscheinlichkeiten oder die in Abschnitt 12.4.2.5 eingeführten Präferenzen.
- Insbesondere sollte dieser Fokus durch die Bedeutung der Hypothesen für die *Therapie* mitbestimmt sein.

Letzteres ist für die meisten bisher realisierten Algorithmen und Systeme nicht ins Blickfeld gerückt.

Ein verbreiteter Algorithmus für die Berechnung von Fehlerlokalisierungen (ohne Fehlermodelle) ist der *Hitting-Set*-Algorithmus (s. [Reiter, 87; Greiner *et.al.*, 89]). Er läßt sich intuitiv leicht begreifen: wenn man (minimale) Konflikte und (minimale) Fehlerlokalisierungen als Menge von Verhaltensmodi repräsentiert, so enthält jede Konfliktmenge mindestens eine defekte Komponente. Mithin sind gültige Fehlerlokalisierungen nur solche, die mindestens eine Komponente aus jedem Konflikt enthalten. Minimale enthalten *genau eine Komponente aus jedem Konflikt* und haben keine echte Teilmenge mit derselben Eigenschaft. Dies ist genau die Definition von minimalen hitting sets. So sind z.B. für die Konfliktmengen

$$\{\neg(\text{ok}(C_1)), \neg(\text{ok}(C_2))\}, \{\neg(\text{ok}(C_1)), \neg(\text{ok}(C_3))\}$$

die minimalen Fehlerlokalisierungen

$$\{\neg(\text{ok}(C_1))\}, \{\neg(\text{ok}(C_2)), \neg(\text{ok}(C_3))\}.$$

Konsistenzbasierte Fehlerlokalisierung wurde erstmals in der *General Diagnosis Engine* (GDE, [de Kleer, Williams, 87]) implementiert. In [Dressler, Struss, 96] wird beschrieben, wie man das im folgenden Abschnitt skizzierte Assumption-based Truth-Maintenance System (ATMS) für die inkrementelle Berechnung minimaler Fehlerlokalisierungen einsetzen kann.

Die Hypothesen für Fehlerlokalisierungen können ohne Fehlermodelle (die Verhaltensmodelle für Fehlermodi der Komponenten bereitstellen) nicht weiter durch das Modell, sondern nur durch zusätzliche Beobachtungen überprüft werden. Bei der Fehleridentifikation hingegen resultiert aus der Modellrevision, d.h. dem Ersetzen eines Modus (z.B. des ok-Modus) durch einen anderen (Fehlermodus), eine Moduszuweisung, die auf Konsistenz zu überprüfen ist, falls sie nicht bereits vorher auf anderem Wege erzeugt und geprüft wurde. Die Suchalgorithmen führen eine Menge von „aktiven“ Moduszuweisungen als Diagnosekandidaten. Daraus wird ein Teil zur Überprüfung ausgewählt. Tritt bei der Überprüfung einer Moduszuweisung eine Inkonsistenz auf, so sind die Diagnosekandidaten, die diesen Konflikt enthalten, zu eliminieren, und die einzelnen im zugehörigen Konflikt enthaltenen Verhaltensmodi sind Kandidaten für eine Revision. Auf diese Weise wird eine Menge von noch nicht widerlegten Nachfolgern der Kandidatenmenge hinzugefügt und der Zyklus beginnt von neuem.

In *SHERLOCK* [de Kleer, Williams, 89] ist das Auswahlkriterium für die zu prüfenden Kandidaten deren Wahrscheinlichkeit, die sich als Produkt der Wahrscheinlichkeiten der beteiligten Verhaltensmodi ergibt. Diese ist nur am Anfang die a-priori-Fehlerwahrscheinlichkeit. Sie ist nach jeder Änderung aufgrund von Beobachtungen und verworfenen Diagnosekandidaten neu zu berechnen. Das Verfahren terminiert aufgrund eines Abbruchkriterium als Funktion der Wahrscheinlichkeiten der geprüften und konsistenten Diagnosen. *SHERLOCK* enthält für jede Komponente einen unbekanntem Fehlermodus mit geringer Wahrscheinlichkeit, der kein assoziiertes Verhaltensmodell hat und daher nicht widerlegbar ist. Damit bleibt die Möglichkeit erhalten, auch unbekannte Fehler nicht auszuschließen.

GDE⁺ [Struss, Dressler, 89] benutzt Fehlermodelle und Fehleridentifikation, um die Fehlerlokalisierung zu verbessern. Dies geschieht durch die Regel der „physikalischen Negation“, die, im Gegensatz zur rein logischen Negation eines korrekten Verhaltensmodus, diesen nicht einfach ersatzlos suspendiert, sondern durch Fehlermodelle ersetzt und die Korrektheit aus der Negation aller möglichen Fehler ableitet („exoneration“). Die dabei zugrunde liegende, manchmal zweifelhafte Annahme, daß alle Fehlermodi bekannt sind, kann explizit gemacht und ggf. auf Basis des Revisionsalgorithmus selbst in Frage gestellt und zurückgezogen werden (vgl. [Dressler, Struss, 96]).

Die Berechnung der präferierten Diagnosen (Abschnitt 12.4.2.5) in der *Default-based Diagnosis Engine* (DDE, s. [Dressler, Struss, 96]) folgt für jede Komponente in einer verworfenen Diagnosehypothese bei der Bestimmung der nachfolgenden Verhaltensmodus der Präferenzrelation. Eine Moduszuweisung wird erst dann auf Konsistenz geprüft, wenn alle ihr gegenüber strikt präferierten Diagnosehypothesen geprüft und verworfen wurden.

12.4.2.7 Implementierung: abhängigkeitsbasierte Konfliktgenerierung

Wir haben nun Elemente bereitgestellt, mit denen wir ein modellbasiertes System zur Generierung von Komponentenfehlerhypothesen als Spezialisierung und Verfeinerung der Grundarchitektur aus Abschnitt 12.2.7 (Abb. 12.2) realisieren können:

- *Modellrevision*: Konfliktgetriebene Generierung von Fehlerhypothesen (Abb. 12.12) erzeugt, wie im vorangegangenen Abschnitt beschrieben, auf Konsistenz zu prüfende Moduszuweisungen.

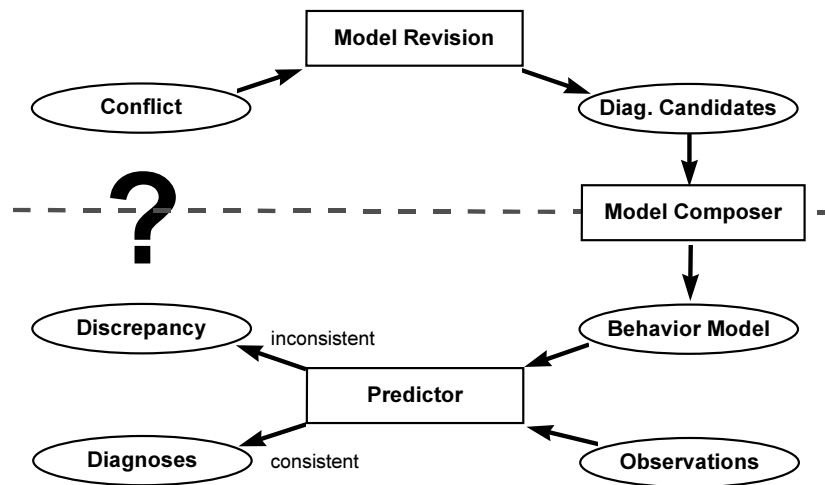


Abbildung 12.13: Die Lücke zwischen festgestellten Inkonsistenzen und Konflikten

- *Modellkomposition* (Abschnitt 12.3.4, Abb. 12.6) erzeugt (in einem Schritt) Verhaltensmodelle für diese Moduszuweisungen.
- *Konsistenzprüfung*: Modellbasierte Verhaltensvorhersage (Abschnitt 12.3.5) und Feststellung von Diskrepanzen (Abschnitt 12.4.2.1) prüfen die Konsistenz dieser Modelle mit den Beobachtungen (Abb. 12.10).

Aber ihre Kombination, dargestellt in Abb. 12.13, läßt noch eine wesentliche Lücke erkennen: die Konsistenzprüfung durch die Verhaltensvorhersage findet auf der „unteren“ mathematischen Modellebene statt. Die Modellrevidierung, die Generation von Diagnosehypothesen, hingegen benötigt Konflikte aus Moduszuweisungen, also Elemente der konzeptuellen Modellebene als Input.

Wie ist die Brücke zwischen der mathematischen und der konzeptuellen Ebene zu schaffen? Wie sind die entdeckten Diskrepanzen, also inkonsistente vorhergesagte Werte, in Konflikte, d.h. inkonsistente Moduszuweisungen umzusetzen? Die intuitive Antwort ist klar: das System muß feststellen, welche Moduszuweisung zu der Diskrepanz geführt hat. Dabei reicht es nicht festzustellen, welche *vollständige* Moduszuweisung gerade getestet wurde, denn die Präzision der Fehlerhypothesen hängt ja gerade davon ab, ob die *minimalen* Konflikte identifiziert werden.

Auf der Generierungsseite wird die Verbindung zwischen Elementen der konzeptuellen Schicht und der mathematischen durch die Modellkomposition hergestellt, und offenbar ist dies die einzige Quelle. „Welcher Komponenten-Verhaltensmodus liegt jeweils einem Verhaltensmodellfragment zugrunde?“ – ohne diese Abhängigkeit aufzuzeichnen, kann das System offenbar keine Konflikte generieren. Im weiteren müssen dann bei der Verhaltensvorhersage die Abhängigkeiten der vorhergesagten Werte und Diskrepanzen von den sie produzierenden Elementen des Verhaltensmodells und anderen Vorhersagen ermittelt werden (Abb. 12.14).

In einem solchen gerichteten Graphen wie in Abb. 12.14, der gewissermaßen die vollzogenen Inferenzen einfriert, kann die Ursache von Diskrepanzen als minimale Menge von Verhaltensmodi, d.h. ein Konflikt identifiziert werden. Für den Aufbau solcher Strukturen ist keine spezielle Lösung für modellbasierte Systeme nötig. Er kann durch allgemeine

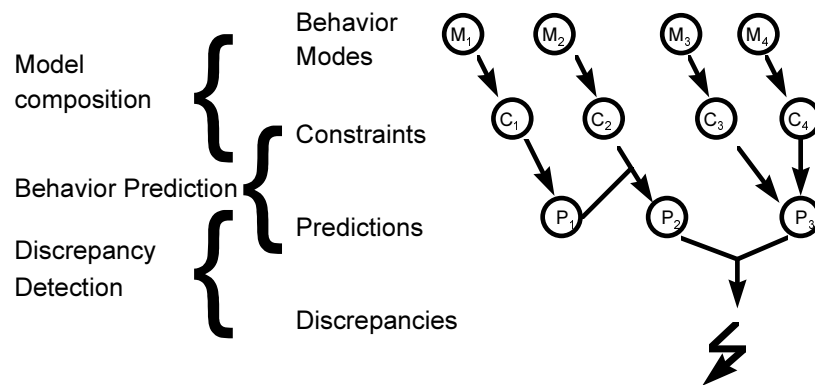


Abbildung 12.14: Abhängigkeiten von der Modellkomposition bis zur Entdeckung von Diskrepanzen.

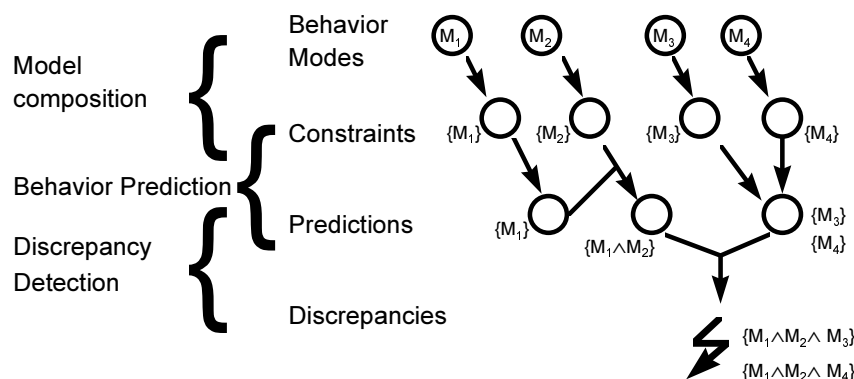


Abbildung 12.15: Vom ATMS berechnete Label und Nogoods

Verfahren implementiert werden, nämlich Truth-Maintenance-Systeme (TMS), die derartige Rechtfertigungsnetze („justification networks“) kreieren. Deren eigentliche Funktion ist es, auf Basis dieser Strukturen die Gültigkeit von Aussagen zu ermitteln und zu verwalten. Auch dies kann für die Zwecke der Konfliktgenerierung direkt ausgenutzt werden: statt nach dem Auftreten einer Diskrepanz die Struktur rückwärts bis zu den beteiligten Verhaltensmodi zu durchsuchen, können die Abhängigkeiten bei jedem Inferenzschritt kombiniert und weiterpropagiert werden und liegen dann beim Auftreten einer Diskrepanz bereits vor. Dies wird von einem Assumption-Based-Truth-Maintenance-System (ATMS, [de Kleer, 86]) geleistet, das zu jeder Aussage ein sog. *Label* erzeugt und aktualisiert, das aus allen konsistenten minimalen Mengen von Annahmen besteht, unter denen diese Aussage gültig ist. Insbesondere geschieht dies für eine abgeleitete Inkonsistenz, womit das ATMS direkt minimale Konflikte (als sog. *Nogoods*) liefert! Im Beispiel von Abb. 12.14 führt dies zu den in Abb. 12.15 gezeigten Labels und insbesondere zu den gewünschten minimalen Konflikten.

Natürlich werden durch das ATMS auch Abhängigkeiten aufgebaut, die nie zu Diskrepanzen führen und im Hinblick auf die Konfliktgenerierung und daher scheinbar für die Diagnose überflüssig sind. Allerdings bilden sie die Basis für Effekte, die vom Problemlöser und insbesondere der Verhaltensvorhersage ausgenutzt werden können:

- Vermeidung wiederholter Ableitungen für verschiedene Annahmenmengen. Statt-

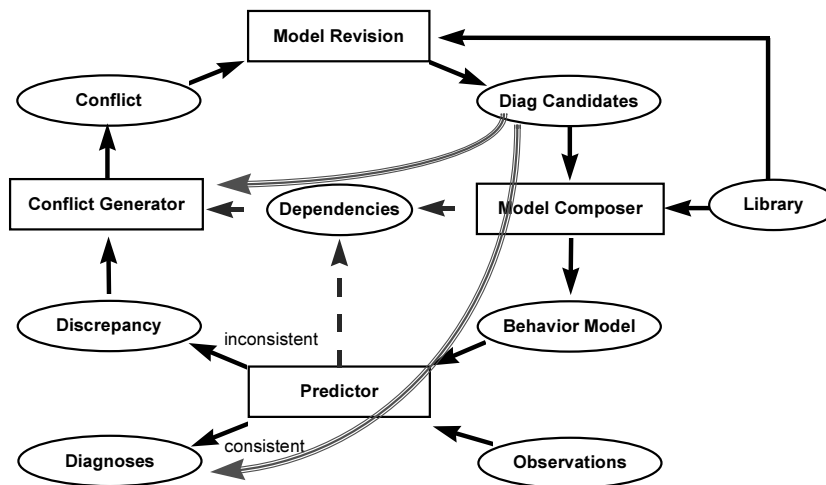


Abbildung 12.16: Das vollständige konsistenzbasierte Diagnosesystem.

dessen wird einfach ein erweitertes Label über alle nachfolgenden Rechtfertigungen propagiert.

- Vermeidung von Ableitungen, die keine konsistente Begründung (und damit leere Label) haben. Jeder einmal entdeckte Nogood eliminiert wirksam einen ganzen Teil des Suchraums.

Darüber hinaus kann

- das Label als Basis genutzt werden, um Inferenzen auf einen bestimmten Fokus des Interesses (z.B. bestimmte Moduszuweisungen) zu beschränken (s. [Dressler, Farquhar, 89]),
- das Label für die Bestimmung weiterer nützlicher Meßpunkte Hinweise liefern (s. Abschnitt 12.4.3),
- das ATMS direkt zur Berechnung von Diagnosehypothesen aus Konflikten benutzt werden (s. [Dressler, Struss, 96]).

Eine Variante des ATMS ermöglicht die Codierung von Defaults und die Berechnung von präferierten Diagnosen (Abschnitt 12.4.2.5). Durch eine Verwaltung von Abhängigkeiten, wie sie z.B. vom ATMS vorgenommen wird, ist also die notwendige Basis gegeben, um einen Konfliktgenerator zu realisieren und somit das modellbasierte Diagnosesystem zu komplettieren (Abb. 12.16).

12.4.2.8 Prozeßorientierte Diagnose

In Abschnitt 12.4.2.4 haben wir die Voraussetzungen für die anschließend dargestellten Verfahren der komponentenorientierten Fehlerlokalisierung und -identifikation aufgelistet. Diese erweisen sich damit als sehr spezialisiert und in ihrer Anwendbarkeit eingeschränkt. Um an das andere Ende des Aufgabenspektrums zu gehen, betrachten wir ein Beispiel aus [Struss, Heller, 98]. Hier geht es um die Diagnose der Ursache von schlechtem Geschmack im Trinkwasser. Dieser ist letztlich auf ein plötzlich im Wasserreservoir auftretendes Algenblütenphänomen (und mangelnde diesbezügliche Behandlung des Trinkwassers) zurückzuführen. Unter Bezugnahme auf die Liste der genannten Vorbedingungen ist hier festzuhalten:

- *Korrektur Entwurf*: Ein Teil des Systems wurde gar nicht entworfen, sondern ist ein natürliches. Es gibt damit auch kein dem System inhärentes Zielverhalten. Ein Ziel, wie z.B. schlechten Geschmack, bzw. toxische Substanzen zu vermeiden, ist ein dem System und seinem Modell äußerliches.
- *Fixierte Konstituenten*: wesentliche Konstituenten sind Prozesse, und diese, wie auch die Objekte, die sie aktivieren (eingetragene Nährstoffe), sind gerade nicht alle bekannt.
- *Bekannte Struktur*: damit ist auch die Interaktion der Verhaltenskonstituenten nicht bekannt, sondern bleibt gerade zu bestimmen.
- *Keine Strukturfehler*: die Störung besteht gerade in einer unvorhergesehenen Struktur der interagierenden Prozesse.
- *Nur Komponentenfehler*: es sind gar keine Komponenten defekt. Prozesse haben ebenfalls keinen Fehlermodus. Die Störung ist keinem der bekannten Objekte, sondern einem nicht vermuteten anzulasten. Eine Therapie kann auch nicht durch Austausch dieses Objekts geschehen, sondern durch zusätzliche Intervention im System.

Nicht nur natürliche Systeme wie in diesem Beispiel, auch z.B. Anwendungen in der Prozeßindustrie und die Notwendigkeit, in komponentenorientierten Systemen Strukturfehler zu diagnostizieren, verletzen zumindest einige der obigen Anwendungsvoraussetzungen und verlangen nach anderen Methoden. Eine Voraussetzung ist offenbar ein Modellierungsformalismus, der offener ist als der komponentenorientierte und Änderungen der Struktur zuläßt. Dies trifft auf die prozeßorientierte Modellierung zu, die kompositional ist und damit den Anforderungen der automatischen Modellrevision genügen kann, wie in Abschnitt 12.3.2 beschrieben wurde.

Gegenstand der Revision sind in diesem Modellierungsformalismus offenbar nicht Verhaltensmodi, sondern zunächst die Elemente, die in der Systembeschreibung als Strukturbedingungen (Objekte, Relationen) oder Wertebedingungen (Parameter, beobachtete Variable) auftreten und keine gesicherten Fakten sind. Dies reicht offenkundig noch nicht aus, denn auf dieser Grundlage könnten Diagnosehypothesen sich weiterhin nur auf die explizit genannten Elemente beziehen. Der Schlüssel für gezielte Hypothesen über das Vorhandensein zusätzlicher Strukturelemente liegt in der Einflußkombination: den entsprechenden Constraints liegt die Closed-World-Assumption bezogen auf die Einflüsse der jeweiligen Variablen zugrunde. Werden diese explizit gemacht, so können sie in Konflikten auftreten und somit zum Ausgangspunkt von Modellrevision werden.

Die Closed-World-Assumption zu revidieren kann und muß dabei nicht heißen, beliebige Objekte und Relationen einzuführen. Die Beschränkung ergibt sich durch die Elemente der Domain Theory, die damit (wie auch eine Komponentenbibliothek für die Fehleridentifikation) die möglichen Störungen als Objekte und die dadurch bedingten Verhaltenskonstituenten enthalten muß. Abb. 12.17 illustriert, daß die Revision der Closed-World-Assumption zielgerichtet ist, da sich durch den Typ der Variablen und den Typ des zugehörigen Objekts eine Auswahl von Verhaltenskonstituenten ergibt, die zusätzliche Einflüsse auf die Variable ausüben könnten, aber nicht aktiv sind. Dies führt zur Annahme ihrer Bedingungen, die aber wiederum innerhalb der Domain Theory eine begrenzte Menge von Ursachen haben.

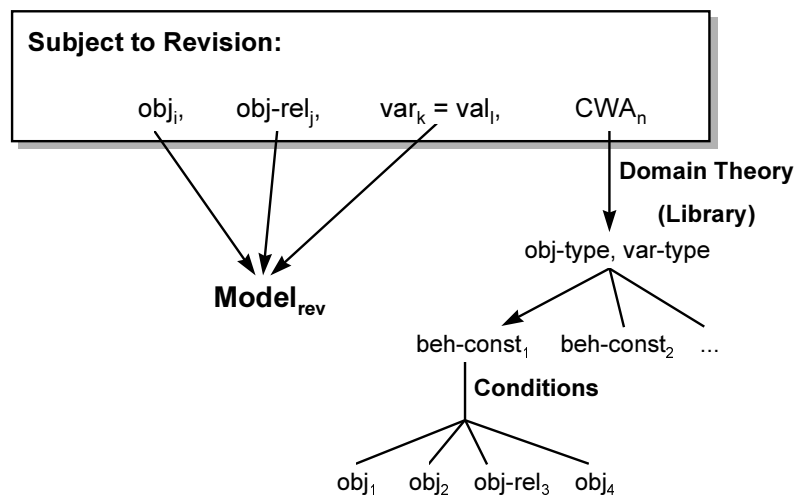


Abbildung 12.17: Die variablenbezogene Closed-World-Assumption als Ausgangspunkt für die Hypothese zusätzlicher Objekte und Prozesse.

In [Collins, 93] wurde erstmals auf Basis von QPT prozeßorientierte Diagnose mit abduktiven Verfahren realisiert. In [Struss, Heller, 98; Struss, Heller, 00] werden die Rekonstruktion prozeßorientierter Modellierung auf logischer Grundlage und G^+DE als deren Kombination mit der konsistenzbasierten Diagnose beschrieben. Tatsächlich wird in der Implementierung dieselbe Diagnosemaschine verwendet, die den Kern der komponentenorientierten Fahrzeug-On-Board-Diagnose von [Sachenbacher *et.al.*, 00] darstellt.

12.4.2.9 Komplexitäts- und Kontrollaspekte

Für die Konsistenzprüfung benötigt die konsistenzbasierte Diagnose das Verhaltensmodell und die darauf basierende Vorhersage. Die Verhaltensvorhersage kann in diesem Zusammenhang aus mehreren Gründen Komplexitätsprobleme produzieren und geeignete Kontrollheuristiken verlangen:

- Bei der Fehleridentifikation führt die Verwendung von *Fehlermodellen* dazu, daß Vorhersagen nicht mehr nur auf der Basis von einem Systemmodell gemacht werden, sondern ggf. *viele Moduzuweisungen* geprüft werden müssen. Dies macht effektive Kontrollmechanismen notwendig, die es gestatten, die Verhaltensvorhersage auf die entsprechenden Moduzuweisungen zu fokussieren. Das ATMS bietet hierfür eine Basis, da es protokolliert, in welchem Kontext, d.h. hier unter welchen Moduzuweisungen bestimmte Ableitungen gültig sind. [Dressler, Farquhar, 89] und [de Kleer, 91] stellen entsprechende Implementierungen vor.
- Systeme mit *vielen Komponenten* können zu aufwendiger Verhaltensvorhersage führen. Abhängig von den zur Verfügung stehenden Beobachtungen kann aber die Verhaltensvorhersage oft auf eine *relevante Substruktur* eingegrenzt werden, z.B. eine Umgebung eines Kurzschlusses in einem Hochspannungsnetz ([Beshta *et.al.*, 93]). Auch hier sind die ATMS-basierten Fokussierungsmechanismen eine gute Basis. Zumindest für komponentenorientierte Diagnose ist eine andere gängige Antwort auf das Problem, die Diagnose entlang einem hierarchischen Systemmodell zu organisieren [Hamscher, 91].

- *Komplexe Verhaltensmodelle* können manchmal zumindest für den Einstieg in die Diagnose abstrahiert oder vereinfacht und für eine weitere Analyse verfeinert werden. Solche Verwendungen multipler Modelle auf verschiedenen Abstraktionsgraden und deren Rechtfertigung durch vereinfachende Annahmen werden in [Struss, 92] analysiert und formalisiert.
- Systeme mit *dynamischem Verhalten* stellen eine weitere Herausforderung dar. Die Dimension der Zeit kann insbesondere das Problem der Prüfung einer Vielzahl von Fehlerhypothesen verschärfen, wenn dazu die Simulation der zugehörigen Verhaltensmodelle notwendig ist. In [Struss, 97] werden Grundlagen und Voraussetzungen dafür entwickelt, nur durch Konsistenzprüfung der Zustände dieselben Diagnoseresultate zu erzielen wie auf der ungleich aufwendigeren Basis von Simulation und Vergleich von Verhalten. Die Grundlage ist durch das in Abschnitt 12.3.3.1 angeführte Theorem über die Gleichheit von Zustands- und Verhaltensmengen gegeben. Diese *zustandsbasierte Diagnose* war eine der Grundlagen für einen erfolgreichen Prototypen der Fahrzeug-On-Board-Diagnose [Sachenbacher *et.al.*, 00]. Ferner war die qualitative Abstraktion entscheidend, um den Diagnoseprototypen realzeitfähig zu machen. Weitere Theorien und Systeme über die Diagnose von Systemen mit Dynamik oder zeitlich variierendem Input werden in [Brusoni *et.al.*, 98; Mosterman *et.al.*, 98; McIlraith *et.al.*, 99] vorgestellt.

12.4.2.10 Anwendungssysteme und -prototypen

Konsistenzbasierte Diagnosesysteme oder Derivate davon haben in den letzten Jahren zu einer Reihe von anwendungsnahen Prototypen und einsatzfähigen Systemen geführt. Die folgende Liste soll einen knappen Eindruck von der Art der Anwendungsgebiete geben:

- Fehlerlokalisierung in Hochspannungsnetzen [Beshta *et.al.*, 93].
- Überwachung und Fehlerdetektion in Ballastwassertanksystemen [Dressler, 96].
- Diagnose und Selbstrekonfigurierung für Raumsonden der NASA [Williams, Nayak, 96].
- Modellbasierte Fehlerdetektion und -identifikation im Rahmen des kommerziellen Überwachungssystems für Gasturbinen TIGER [Trave-Massuyes, Milne, 87].
- Einsatz modellbasierter Verfahren im Rahmen des Fahrzeugdiagnose-Systems SIDIS der Firma Siemens.
- Automatische Erzeugung von Prüfprogrammen für Werkstatt-Tester für Fahrzeuge der Firma GenRad.

Im Rahmen des BMBF-geförderten Projekts INDIA [Hotz *et.al.*, 00]:

- Erzeugung von Fehlerbäumen für die Gabelstaplerdiagnose [Cunis, 00].
- Diagnose von Farbe-Anlagen [Guckenbiehl, Münker, 00].
- Erzeugung von Fehlersuchanleitungen für die Werkstattdiagnose von Kraftfahrzeugen [Struss *et.al.*, 00].

Im Rahmen des EU-Projekts Vehicle Model-Based Diagnosis (VMBD, [Bidian *et.al.*, 99]):

- Erzeugung von Entscheidungsbäumen für die Fahrzeug-On-Board-Diagnose [Cascio *et.al.*, 99].
- Fahrzeug-On-Board-Diagnose von emissionsrelevanten Fehlern eines Diesel-Turboladers [Sachenbacher *et.al.*, 00].

Oft wird die Frage gestellt, ob die Diagnoseverfahren auch auf Software-Systeme angewandt werden können. Eine präzise Antwort darauf ist, daß einerseits konsistenzbasierte Verfahren durchaus hilfreich für das Debugging von Programmen sind, es sich dabei aber eigentlich nicht um ein Diagnoseproblem handelt. Wir diskutieren dies kurz in Abschnitt 12.5.2.

Die Entwicklung weiterer Anwendungssysteme wird sicherlich in den nächsten Jahren weitergehen und dabei auf Fragestellungen stoßen, die sich aus dem gesamten Diagnose-Arbeitsprozeß und seinen Randbedingungen ergeben, wie in Abschnitt 12.4.1 geschildert. Dies stellt neue Herausforderungen an das Forschungsgebiet.

12.4.3 Generierung von Tests, Meßvorschlägen und Entscheidungsbäumen

In der Regel führt die Erzeugung von Fehlerhypothesen zunächst zu keinem eindeutigen Ergebnis, selbst unter Beschränkung auf minimale Diagnosen und Berücksichtigung von zusätzlichen Ordnungen durch Präferenzen oder Wahrscheinlichkeiten. Die einzige Möglichkeit, weitere Information in Gestalt zusätzlicher Konflikte zu erhalten, besteht dann darin, die Menge der Beobachtungen zu vergrößern. Dies kann geschehen durch

- *Beobachtung zusätzlicher Größen* („*probing*“) in einer gegebenen Situation. Für verschiedene Anwendungen ist dieser Weg nicht oder nur eingeschränkt gangbar, etwa bei der Fahrzeug-On-Board-Diagnose, die einen festen Satz von Sensorsignalen zur Verfügung hat.
- Beobachtungen des Verhaltens in *zusätzlichen Situationen*. Werden diese Situation durch Beeinflussung von außen gezielt herbeigeführt, so sprechen wir von *Testen*. Auch dies ist z.B. bei einem Fahrzeug im Normalbetrieb weitgehend ausgeschlossen. In solchen Fällen kann nur passiv auf zusätzliche Information durch die Evolution des Systems gewartet werden.

Zusätzliche Beobachtungen und die Beeinflussung eines Systems zu diesem Zweck erfordert zumeist Tätigkeiten und verursacht damit Zeit und Kosten. Daher wird die Frage bedeutsam, welche dieser Aktionen ein optimales Verhältnis von Aufwand und Nutzen versprechen. Der Aufwand hängt dabei wesentlich von Faktoren ab, die außerhalb des Verhaltensmodells liegen. Der *Nutzen von Tests* aber bestimmt sich durch den Beitrag, den die zusätzlichen Beobachtungen zur Gewinnung zusätzlicher Konflikte liefern, die die Menge der aktuellen Diagnosekandidaten verringern. Dies läuft auf die Untersuchung der Frage hinaus, in welchen Zuständen sich die verschiedenen konsistenten Moduszuweisungen, bzw. deren assoziierte Verhaltensmodelle (beobachtbar) unterscheiden. Dies aber ist modellbasierten Lösungen zugänglich.

Im Abschnitt 12.2.4 haben wir das Problem formalisiert als Suche nach (herstellbaren) Situationen SIT, in denen die verschiedenen Verhaltensmodelle unterscheidbare Beobachtungen vorhersagen:

$$\begin{aligned} \text{MODEL}_1 \cup \text{SIT} &\vdash \text{OBS}_1 \\ \text{MODEL}_2 \cup \text{SIT} &\vdash \text{OBS}_2 \\ \text{OBS}_1 \cup \text{OBS}_2 &\vdash \perp. \end{aligned}$$

Repräsentiert man die Verhaltensmodelle extensional durch ihre Zustandsrelationen, R_{S1} , R_{S2} , so läßt sich die Menge der diskriminierenden Zustände als

(gnd, gnd)			
(gnd, btw)			
(gnd, bat)			
(btw, gnd)			
(btw, btw)			
(btw, bat)			
(bat, gnd)			
(bat, btw)			
(bat, bat)			
	neg	0	pos

(gnd, gnd)			
(gnd, btw)			
(gnd, bat)			
(btw, gnd)			
(btw, btw)			
(btw, bat)			
(bat, gnd)			
(bat, btw)			
(bat, bat)			
	neg	0	pos

Abbildung 12.18: Qualitativ abstrahierte Verhaltensrelationen für einen Widerstand. Links: korrekt. Rechts: offen.

$$R_D = R_{D12} \cup R_{D21} = R_{S1} \setminus R_{S2} \cup R_{S2} \setminus R_{S1}$$

bestimmen. Ein Algorithmus zum Vorschlag von Tests muß dann folgende Fragen entscheiden:

- *Einstellbarkeit*: formal heißt dies (im statischen Fall) zu prüfen, ob die Projektionen von R_D auf die *beeinflussbaren* Größen einen nicht-leeren Durchschnitt haben. Für die Situationen SIT_D , die diesem Durchschnitt entsprechen:
- *Beobachtbarkeit*: unterscheiden sich die Projektionen auf die *beobachtbaren* Größen überhaupt?
- *Eindeutigkeit*: sind diese Projektionen disjunkt? In gegenteiligen Fall ist nicht gewährleistet, daß mit Sicherheit unterschiedliche Beobachtungen erzeugt werden (Dies bezieht sich auf nicht-deterministisches Verhalten, etwa verursacht durch Rauschen).

Man sieht, daß diese Kriterien durch Mengenoperationen und Projektionen auf den Verhaltensrelationen zu beantworten sind. Allerdings stellt sich die Frage, ob diese Operationen auch praktisch berechenbar sind. Die Antwort liegt in der geeigneten Abstraktionsebene des Verhaltensmodells: durch qualitative Abstraktion kommt man zu endlich vielen endlichen Verhaltensrelationen und schafft damit eine Grundlage für die Implementierung des Verfahrens. Abbildung 12.18 illustriert dies am einfachen Beispiel eines Widerstands: Sie zeigt die Relationen der Verhaltensmodi „ok“ und „offen“ unter einer qualitativen Abstraktion, die nur die Stromrichtung (neg, 0, pos) unterscheidet und für die Spannung nur Masse (gnd) und Batteriespannung (batt) betrachtet. Dies reicht aus, um zu entscheiden, unter welchen Spannungspaaren sich die beiden Verhaltenmodi mit Sicherheit unterscheiden lassen.

In [Struss, 94] werden die theoretischen Grundlagen und Algorithmen entwickelt. Diese wurden implementiert und angewendet auf Schaltungen von Weichenschaltungen [Inderst *et.al.*, 95] und Generierung von Prüfplänen für Fahrzeugsysteme [Struss *et.al.*,

00]. Liegen Wahrscheinlichkeitsverteilungen für die Zustandsrelationen und/oder Wahrscheinlichkeiten für die Moduszuweisungen vor, so läßt sich die Nützlichkeit von Tests durch das Kriterium der Minimierung der Entropie bestimmen. Dabei verspricht der Test-Input \underline{v}_{cause_0} (im Mittel) den größten Informationsgewinn für die Diskriminierung von Verhaltensmodi $\delta_i \in \Delta$, der den Ausdruck

$$- \sum_{\underline{v}_{obs_0} \in O(\underline{v}_{cause_0})} p(\underline{v}_{obs_0}) \log p(\underline{v}_{obs_0}) + \sum_{\delta_i \in \Delta} p(\delta_i) \sum_{\underline{v}_{obs_0} \in O(\underline{v}_{cause_0})} p(\underline{v}_{obs_0} | \delta_i) \log p(\underline{v}_{obs_0} | \delta_i)$$

maximiert. Dabei ist $O(\underline{v}_{cause_0})$ die Menge der möglichen Beobachtungen bei Anlegen von \underline{v}_{cause_0} , und die Wahrscheinlichkeit der beobachtbaren Tupel ist gegeben durch Verteilungen und Wahrscheinlichkeiten der einzelnen Verhaltensmodi:

$$p(\underline{v}_{obs_0}) = \sum_{\delta_i \in \Delta} p(\underline{v}_{obs_0} | \delta_i) p(\delta_i)$$

(vgl. [Struss, 94]).

Weil diese Methode auf dem relationalen Verhaltensmodell aufsetzt, ist sie unabhängig von der Modellierungs-Ontologie und kann damit nicht nur für die Testgenerierung bei komponentenorientierter Diagnose angewandt werden. Sie ist ferner allgemein in dem Sinne, daß sie nicht nur Veränderungen des Zustands berücksichtigen kann, sondern auch strukturelle Veränderungen des Systems, die zu diskriminierenden Beobachtungen führen. Dies ist durchaus kein exotischer Fall, denn schon die Widerstandsmessung z.B. an einem Sensor im Fahrzeug erfordert die Unterbrechung der Batterieverbindung und den Anschluß eines Meßgeräts, also eine strukturelle Veränderung. Damit tritt allerdings das Problem auf, geeignete Strukturmodifikationen generieren zu können. Als Einschränkung liegt den gegenwärtig realisierten Verfahren die Voraussetzung zugrunde, daß die Einstellung der Zustände durch eine Transition geschieht und keine Sequenz von Beeinflussungen erfordert.

Eine logische Charakterisierung von Tests findet sich in [McIlraith, Reiter, 92]. Die Arbeiten zur Testgenerierung von [Lange *et al.*, 96] gehen von einer Menge vorgegebener Typen von Tests für einzelne Komponenten und deren Kombination aus. Z.B. lassen sich in Serie geschaltete Widerstände durch einen Test gleichzeitig auf „offen“ prüfen.

Geht man davon aus, daß in einer festen Situation weitere Beobachtungen gemacht werden können oder müssen, so liegt SIT in der obigen Charakterisierung der Testgenerierung fest. D.h. potentielle Beobachtungen zusätzlicher Variablen beziehen sich auf die Vorhersagen des Verhaltensmodells in dieser gegebenen Situation. Da wir hier aber davon ausgehen, daß die Frage des Meßvorschlags nach der Generierung von Diagnosehypothesen aufgrund von Verhaltensvorhersagen aufgeworfen wird, sind damit Vorhersagen für die konsistenten Moduszuweisungen bereits durchgeführt. Unter der Voraussetzung, daß diese Vorhersagen vollständig sind, ist es unnötig, auf die Verhaltensrelationen zurückzugehen wie im oben geschilderten Verfahren. Die Meßvorschlagskomponente für GDE [de Kleer, Williams, 87] und allgemein für abhängigkeitsbasierte Diagnose kann ausnutzen, daß die Abhängigkeiten der vorhergesagten Werte von Moduszuweisungen (allgemeiner: von Elementen der konzeptuellen Systembeschreibung) protokolliert sind. Damit lassen sich

Meßvorschläge erzeugen, indem für die noch nicht gemessenen aber meßbaren Größen die diskriminierende Wirkung der von verschiedenen Diagnosen vorhergesagten Werte analysiert wird. Wiederum können vorhandene Wahrscheinlichkeiten zur Auswahl des besten Meßpunktes ausgenutzt werden durch ein Kriterium, das eine Spezialisierung der oben angeführten Entropie-Minimierung ist.

Statt nur den nächsten besten Test vorzuschlagen, kann das Ziel auch darin bestehen, einen ganzen Entscheidungsbaum für die bestmögliche Diskriminierung durch Sequenzen von Beobachtungen zu generieren. Dies ist u.a. in [Cascio *et.al.*, 99] für die Fahrzeug-On-Board-Diagnose auf der Basis des ID3-Algorithmus 14.3 und in [Cunis, 00] für Gabelstapler-On-Line-Diagnose durchgeführt worden.

12.4.4 Therapiegenerierung und -planung

Die Erzeugung von Diagnosehypothesen findet ihren Zweck nur darin, die zumindest partielle Wiederherstellung der Funktion des gestörten Systems zu unterstützen. In der komponentenorientierten Diagnose reicht eine Fehlerlokalisierung aus, um den Austausch von Komponenten als Therapie festzulegen. Für die Fahrzeug-On-Board-Diagnose z.B. geht es darum, andere Klassen von Fehlern voneinander abzugrenzen. So könnten etwa Leckagen unabhängig von ihrer Lokalisierung eine Abschaltung des Motors auslösen, andere Fehler derselben Komponenten hingegen ein Notfahrprogramm. Die verfügbaren Therapiemaßnahmen haben also einen starken Einfluß auf die Zielsetzung und den Ablauf der Generierung von Diagnosehypothesen, der aber bisher kaum systematisch Eingang in die Diagnosesysteme gefunden hat. [Friedrich *et.al.*, 91] entwickelt eine Theorie der Integration von Diagnose und Therapie. Zum Beispiel ist eine genaue Lokalisierung eines Fehlers in Hochspannungs- oder anderen Netzen oft nicht nötig (und auch nicht möglich aufgrund der vorhandenen Meldungen) für die Auswahl einer geeigneten Restaurierung, d.h. Rekonfigurierung des Netzes. Im Extremfall ist es sogar möglich, den Schritt der Erzeugung von Diagnosehypothesen völlig entfallen zu lassen und ausschließlich Therapiegenerierung sowie Beobachten und Testen dafür zu betreiben. Dies entspricht in etwa einem „Kurieren der Symptome“ (d.h. der Abweichungen vom Zielverhalten) ohne Behebung der Ursachen.

Die formale Charakterisierung der Aufgabe wurde im Abschnitt 12.2.6 als Revisions-schritt

$$\text{MODEL}_1 \cup \text{GOALS} \vdash \perp \rightsquigarrow \text{MODEL}_2 \cup \text{ACTIONS} \cup \text{GOALS} \not\vdash \perp$$

geliefert. GOALS kann hier Beschränkungen von Werten von Systemgrößen (also eine Relation in $\text{DOM}(\underline{v}_S)$) aber auch Forderungen über Objekte und Relationen enthalten, etwa daß bestimmte Objekte oder Objektrelationen nicht verändert oder nicht hergestellt werden dürfen. Damit läßt sich zum Beispiel ausdrücken, daß bestimmte Substanzen nicht in gewisse Räume und Behälter gelangen dürfen, etc. Es ist festzuhalten, daß Ziele nicht immer direkt im Hinblick auf das Modell formuliert sind. Oft sind relevante Ziele globale Charakterisierungen gewünschter Funktionen, die sich z.T. schwer oder nicht eindeutig in entsprechende Beschränkungen umsetzen lassen, etwa „Vermeidung der Beeinträchtigung

von Fahrkomfort“ oder „Bewahrung von Artenvielfalt“.

Therapiegenerierung kann außer wegen eingeschränkter Diagnosemöglichkeiten, d.h. Schwierigkeiten der hinreichend guten Bestimmung von $MODEL_1$, aus einer Reihe von Gründen Schwierigkeiten bieten:

- *Großer Suchraum*: ACTIONS, die Menge geeigneter Maßnahmen, kann groß sein, insbesondere, wenn eine Kombination von Maßnahmen erforderlich ist. Der einfachere Fall liegt vor, wenn im System bereits Redundanz angelegt ist, die im Fehlerfall ausgenutzt werden kann. Dies geschieht insbesondere über das Umlegen von „Schaltern“ unterschiedlicher physikalischer Art, etwa elektrischen Schaltern (elektrische Netze) oder Ventilen (Rohrleitungssysteme). Die Revisionskandidaten sind somit Werte von Zustandsvariablen von Systemkomponenten und somit eine wohldefinierte Menge von möglichen Revisionschritten, die zudem noch völlig im Rahmen des Verhaltensmodells vorhersagbar und überprüfbar sind. Flugzeuge, Satelliten und Raumsonden sind wegen der höchst eingeschränkten Wartbarkeit in bestimmten Situationen ein weiteres Beispiel für solche redundanten Systeme. So ist bei der NASA ein System für die Selbstdiagnose und Selbstrekonfigurierung entwickelt worden und u.a. in der Deep Space 1 Mission getestet worden [Williams, Nayak, 96].
- *Konfligierende Ziele*: GOALS kann mehrere Einzelziele enthalten, die aufgrund des Defektes nicht alle gleichzeitig eingehalten werden können. Z.B. kann es bei der Restauration eines Spannungsversorgungsnetzes Fälle geben, in denen nicht alle Kunden gleichzeitig wieder ans Netz gebracht werden können. Die konsistenzbasierte Problemlösung läßt hier aber durchaus zu, auch Teile von GOALS zum Gegenstand von Revision zu machen, wobei die Revision wie üblich nach gewissen Kriterien minimiert werden kann.
- *Dynamik von Therapie und Kontext*: jede Therapie kann erst nach einer gewissen Zeit das Ziel erreichen. Mehr noch, es kann eine Menge von Therapieschritten nötig sein, die in bestimmter Reihenfolge und unter Einhaltung bestimmter Randbedingungen durchzuführen sind. Dies Umlegen der Schalter für die Restauration des Versorgungsnetzes kann nicht für alle gleichzeitig, sondern unter Einhaltung gewisser Reihenfolge geschehen. Dabei ist die Verletzung von Zielen in Zwischenzuständen, also etwa die temporäre Abtrennung von Kunden vom Netz, zu vermeiden, ebenso die Überlastung von kurzfristig auftretenden Engpässen im Netz. Es ist möglich, daß aus diesem Grunde zwischenzeitlich Schalter umgelegt werden müssen, die eigentlich gar nicht direkt vom Übergang vom Ausgangs- zum Zielzustand berührt sind. Im allgemeinen kann sogar der Fall eintreten, daß ein konsistenter Zielzustand auf keinem Pfad ohne zwischenzeitliche Verletzung von Zielen erreichbar ist. Schließlich kann an eine Therapie der Anspruch gestellt werden, robust zu sein in dem Sinne, daß sie auch beim Eintreten weiterer Störungen durchführbar bleibt. Kurz, im allgemeinen ist Therapie ein Planungsproblem 13. In [Sun, Weld, 93] sind Schritte in diese Richtung gemacht worden. Als eigentlicher Zweck der ganzen Diagnoseaktivität ist der Therapieerzeugung mehr Bedeutung im Forschungsgebiet der modellbasierten Diagnose zu geben, und es sollten Methoden für modellbasierte Planung entwickelt werden.

12.5 Weitere Anwendungen

12.5.1 Failure-Mode-and-Effects Analysis (FMEA)

Der Kern dieser Aufgabe ist, für einen gegebenen Systementwurf die potentiellen Auswirkungen von Komponentenfehlern zu ermitteln und zu bewerten. Dies ist, zumindest für große Klassen von Systemen wie elektrische Schaltungen, hydraulische Systeme etc. keine besonders schwere, aber aufgrund der Anzahl der zu analysierenden Fälle eine zeitraubende Tätigkeit, die zudem für Systemvarianten immer wiederkehrt und sich für eine Automatisierung anbietet. Dies verlangt

- *kompositionale* Modellierung, um die Systemmodelle für die Fehler automatisch zu erstellen,
- *qualitative* Modelle, da es darum geht, die *wesentlichen* Auswirkungen von Fehlerklassen und nicht von numerisch spezifizierten Einzelfehlern zu bestimmen und natürlich
- *automatische* Verhaltensvorhersage.

AutoSteve [Price, 98] ist ein Produkt der Firma FirstEarth, das von Ford weltweit für die FMEA elektrischer Schaltungen eingesetzt wird. Das System führt eine qualitative Analyse von Schaltungen mit fehlerhaften Komponenten für verschiedene Zustände durch und vergleicht diese mit dem Verhalten der korrekten Schaltung. Ferner kann *Sneak-Circuit Analysis* durchgeführt werden, d.h. die Untersuchung, ob in einer entworfenen Schaltung Zustände existieren, in denen beabsichtigte Funktionen nicht ausgeführt werden können oder unbeabsichtigte aktiviert werden. Diese Funktionen sind als Aktivität bestimmter Subsysteme charakterisiert, die letztlich auf Stromvariablen im Verhaltensmodell der Schaltung abgebildet werden kann. Schon bei relativ kleinen Schaltungen ist diese Aufgabe vom Menschen nicht immer leicht erschöpfend zu bewältigen und daher ein gutes Beispiel für eine wirksame Anwendung modellbasierter Methoden.

12.5.2 Entwurf und Konfigurierung

Anwendungsnahe Arbeiten beziehen sich vor allem auf Entwurfsaufgaben, in denen die Struktur eines Systems schon weitgehend festliegt und im wesentlichen Komponententypen ausgewählt und parametrisiert werden müssen. [Brinkopp *et.al.*, 95] stellt eine erfolgreiche Anwendung in der Konfigurierung von Rührwerken vor. Die Unterstützung innovativen Designs ist ein herausforderndes Thema für modellbasierte Methoden und speziell qualitative Modellierung (s. etwa [Williams, 92]).

Man sollte sich darüber im klaren sein, daß das Problem der „*Software-Diagnose*“ tatsächlich ein *Entwurfsproblem* ist. Wir sind hier nicht mit einer Situation konfrontiert, in der ein vormals funktionierendes System plötzlich eine Störung zeigt, sondern daß wir dabei sind, ein System zu entwerfen und zu realisieren, das eine gewisse Zielspezifikation erfüllen soll. Es kann hier also gerade nicht, wie bei Geräten, davon ausgegangen werden, daß das System korrekt entworfen wurde und nun einige Komponenten abweichendes Verhalten zeigen. Es geht darum, herauszufinden, warum das System die Zielspezifikation nicht erfüllt und was zu revidieren ist. Insbesondere kann diese Revision

gravierende strukturelle Änderungen erfordern, was große Probleme für die Modellrevision aufwirft. Je näher der Software-Entwurf an der Konfigurierung von vorgefertigten Software-Bausteinen entlang einer bestimmten Architektur liegt, desto beherrschbarer ist die Problemlösung. Arbeiten, die konsistenzbasierte Verfahren für Software-Debugging anwenden, sind z.B. in [Stumptner, Wotawa, 99] dargestellt.

12.5.3 Tutorsysteme

Konzeptuelle und qualitative Modellierung sind eine Voraussetzung für Computerunterstützung für Lernen über physikalische und technische Zusammenhänge. So war ein System zur Ausbildung über Schiffsantriebe bei der US Navy einer der Ursprünge für die Forschung über qualitatives Schließen. *CyclePad* [Forbus *et.al.*, 98] von der North-Western University in Illinois ist ein im Internet verfügbares Tutorsystem im Bereich thermodynamischer Systeme.¹ An der Universität Amsterdam wird an einem System zur Lernunterstützung von Undergraduates in Physik gearbeitet. Eine Einführung gibt [Bredeweg, 98].

12.5.4 Modellierungsunterstützung

Automatische Modellkomposition kann auch für die Erstellung numerischer Simulationsmodelle ausgenutzt werden. *SIMGEM* [Forbus, Falkenhainer, 92] nutzt prozeßorientierte Modellkomposition, um die möglichen qualitativ unterschiedlichen Verhalten zu identifizieren. Für diese wird dann unter Festlegung von numerischen Parametern ein numerisches Simulationsmodell in C++ generiert und anschließend simuliert. Wesentlich ist, daß auch für die numerische Simulation die konzeptuelle, prozeßorientierte Schicht erhalten bleibt. Auf dieser Grundlage können intuitiv nachvollziehbare Interpretationen und Erklärungen der numerischen Simulationsergebnisse in Gestalt von Aussagen über Aktivität von Prozessen, Erzeugung und Existenz von Objekten etc. erzeugt werden. Ferner verhindern die im konzeptuellen Modell explizit repräsentierten Vorbedingungen der Modellfragmente, daß die Simulation den Bereich verläßt, für den das Modell gültig ist.

Methoden der qualitativen Modellierung werden auch angewendet, um „Systemidentifikation“, richtiger: Modellidentifikation, zu unterstützen. In [Capelo *et.al.*, 98] wird damit sichergestellt, daß durch die geeignete Auswahl der Modellstruktur und Parameteridentifikation wichtige *qualitative* Eigenschaften der empirischen Daten (im Gegensatz zu numerischen Approximationskriterien) im Modell widergespiegelt werden. Das Verfahren wird auf die Modellidentifikation für visko-elastische Materialien angewandt. PRET [Bradley, Stolle, 96] ist ein generelles Werkzeug zur Modellidentifikation.

12.6 Zusammenfassung

Modellbasierte Systeme können sich auf unterschiedliche Arten von Modellen beziehen. Am erfolgreichsten sind sie bisher zum einen auf der Grundlage qualitativer Modelle und zum anderen auf dem Gebiet der Diagnose und Fehleranalyse mit dem in Abschnitt

¹<http://www.qrg.iis.nwu.edu/software/cyclepad/>

4 deutlich gemachten Fokus auf komponentenorientierte Systeme und Fehler. Das Forschungsgebiet berührt grundlegende Aufgaben und Ziele der KI im Hinblick auf die Repräsentation von Wissen und das Schließen über unsere materielle, natürliche und technische, Umwelt. Dennoch ist es dabei, den Eingang in industrielle Anwendungen zu beschleunigen, wie kaum ein anders Gebiet der KI unter Bewahrung der systematischen Herangehensweise und theoretischen Grundlagen.

Die Aktualität und das Potential modellbasierter Systeme für industrielle Anwendungen folgt direkt aus den diskutierten Prinzipien. Die Modellierungsformalismen stellen, etwa in Form von Modellbibliotheken, einen Weg zur Wissensrepräsentation für technisches Know-How bereit und können somit einen Beitrag zum *Wissensmanagement* in technologie-orientierten Unternehmen leisten. Daß diese Repräsentation *systematisch* und vor allem *kompositional* ist, erlaubt die *Wiederverwendung* von Modellelementen und damit reduzierten Aufwand bzw. Automatisierung bei der Modellierung.

Die Gebietsunabhängigkeit der problemlösenden Inferenzmechanismen wiederum erlaubt deren *Wiederverwendung* für viele Anwendungssysteme. Insgesamt reduziert sich im Idealfall – bei existierender Modellbibliothek – die Implementierung eines maßgeschneiderten Problemlösers auf die Eingabe der jeweiligen Systemstruktur sowie ggf. von Parametern und situationsspezifischen Beobachtungen oder Hypothesen.

In diesem Beitrag haben wir uns auf die Darstellung der Arbeiten beschränkt, die eine gewisse Reife, Akzeptanz und allgemeine Bedeutung erlangt haben. Viele interessante Aspekte sind dabei unberücksichtigt geblieben. Viel bleibt noch zu lösen. Der Schritt in die Anwendungswelt behindert dabei keineswegs die wissenschaftliche Arbeit in diesem Gebiet, sondern hat, im Gegenteil, wichtige Fragen und Forschungsziele aufgeworfen und die Forschung fokussiert und ihr Evaluierungskriterien für ihre Resultate geboten.

Gerade in der Konfrontation mit der Anwendungswelt stellen sich theoretische Herausforderungen und Ziele. Zu den wichtigsten, die mehr Forschungsarbeit verlangen, kann man zählen:

- Eine umfassendere Modellierung von Problemlösungs- und Arbeitsprozessen im technischen Bereich und die Erweiterung der bisherigen Diagnostiktheorien und -systeme, die Integration von Testen und Therapie als Tätigkeiten und die Entwicklung von Methoden des Planens unter Ausnutzung von Systemmodellen.
- Methoden und Techniken für die Behandlung dynamischer und komplexer Systeme unter Einschluß von natürlichen Systemen. Gerade bei letzteren sind qualitative Methoden mit ihrer Fähigkeit, partielles Wissen auszudrücken und unvollständige Information zu verarbeiten, herausgefordert. Dabei bleibt zu überprüfen, ob die vorhandenen Ontologien und mathematischen Formalismen hinreichend sind oder erweitert und modifiziert werden müssen.
- Unterstützung und Automatisierung der Modellierung. Die Erstellung und Nutzbarkeit von Modellbibliotheken ist entscheidend. Andererseits existieren in der industriellen Praxis Modelle, die nicht direkt als Modelle für die geschilderten Systeme tauglich sind, aber deren Ausgangspunkt sein könnten oder sogar müßten. Automatische oder zumindest unterstützte Akquisition und Transformation von Modellen wird eine zunehmend praktisch wichtige Aufgabe. Für Bereiche der Modellierung natürlicher, etwa ökologischer Systeme wiederum stellt sich die Aufgabe, die Ent-

wicklung, Überprüfung und Revision von Elementen allgemeinerer Bibliotheken zu unterstützen.

All dies kann letztlich nur gelingen, wenn die Methoden und Werkzeuge gemeinsam mit den jeweiligen Fachexperten entwickelt und ausgewertet werden.

12.7 Infrastruktur, Quellen und Werkzeuge

- *Literatur*: Obwohl das aktive und dynamische Forschungsgebiet zunehmend in industrielle Anwendungen und auch in die Ingenieurdisziplinen eindringt, gibt es derzeit keine umfassenden Lehrbücher. Den besten Einstieg vermitteln Sammlungen von Papieren wie [Bobrow, 84] als wichtiger Ausgangspunkt des Gebietes und [Faltings, Struss, 92], vor allem aber die Sammlungen der einflußreichsten Papiere in [Weld, de Kleer, 90] über qualitatives Schließen und [Hamscher *et.al.*, 92] über modellbasierte Diagnose. [Davis, 90] ist eine gute Darstellung über Common-Sense-Wissen und -Schließen über physikalische Systeme, [Kuipers, 94] ein Buch über qualitative Simulation. In [de Kleer, Forbus, 93] werden vor allem ATMS-basierte Problemlöser behandelt. [Price, 99] stellt anwendungsorientierte Diagnosesysteme vor. Modellbasierte Systeme und qualitative Modellierung ist auch als Gebiet der Electronic Transactions in AI (ETAI²) in Vorbereitung.
- *Organisationen*: Innerhalb der Gesellschaft für Informatik (GI) ist das hier behandelte Gebiet in der Fachgruppe „Wissensrepräsentation und Schließen“ organisiert,³ die auch die einschlägige deutsche Mailing-List verwaltet (Eintragung bei struss@in.tum.de). Als Network of Excellence der EU existiert derzeit MONET,⁴ das sich der Vermittlung zwischen Forschung und Industrie auf diesem Gebiet verschrieben hat und auch die internationale mailing list qphysics⁵ verwaltet.
- *Workshops*: Die wichtigsten internationalen (jährlichen) Workshops sind der über „Principles of Diagnosis (DX)“ und „Qualitative Reasoning (QR)“. Daneben gibt es Workshops auf den großen internationalen KI-Konferenzen (z.B. über „Engineering Applications“ bei der IJCAI).
- *Software*: Leider gibt es bisher nur wenige allgemein verfügbare Werkzeuge. Das Simulationssystem QSIM (www.xx) und die CyclePad⁶ sind Public-Domain-Software. Kommerzielle Software-Systeme für modellbasierte Diagnose sind rodon der Firma R.O.S.E. GmbH⁷ und RAZ'R von OCC'M Software GmbH (www.occm.de).

Literaturverzeichnis

[Addanki *et.al.*, 91] S. Addanki, R. Cremonini und J.S. Penberthy. Graphs of models. Artificial Intelligence, 51(1-3), Oktober 1991.

²<http://www.ida.liu.se/ext/etai/>

³<http://www.gi-ev.de/verein/>

⁴<http://monet.aber.ac.uk>

⁵http://monet.aber.ac.uk/mailling_lists/qphysics/

⁶<http://www.cs.utexas.edu/users/qr/QR-software.html>

⁷<http://www.rose.de>

- [Berleant, Kuipers, 92] D. Berleant und B. Kuipers. Combined Qualitative and Numerical Simulation with Q3. In [Faltings, Struss, 92].
- [Beschta *et al.*, 93] A. Beschta, O. Dressler, H. Freitag M. Montag, und P. Struss. A model-based approach to fault localization in power transmission networks. In: Intelligent Systems Engineering, Vol. 2, No. 1, pp. 3-14, 1993.
- [Bidian *et al.*, 99] Bidian, P., Tatar, M., Cascio, F., Theseider-Dupré, D., Sachenbacher, M., Weber, R., Carlén, C.: Powertrain Diagnostics: A Model-Based Approach, Proceedings of ERA Technology Vehicle Electronic Systems Conference '99, Coventry, UK, 1999
- [Bobrow, 84] D. Bobrow (Hg.). Artificial Intelligence (Special Volume on Qualitative Reasoning About Physical Systems), Band 24(1-3). Elsevier, 1984.
- [Bradley, Stolle, 96] E. Bradley und R. Stolle. Automatic construction of accurate models of physical systems, in Annals of Mathematics and Artificial Intelligence, 17, pp. 1-28, J.C. Baltzer AG, Science, 1996.
- [Bredeweg, 98] B. Bredeweg und R. Winkels. Qualitative models in interactive learning environments: an introduction, in Interactive Learning Environments, 5(1), pp. 1-18, Swets & Zeitlinger, 1998.
- [Brinkopp *et al.*, 95] A. Brinkopp, N. Laudwein und R. Maassen. Routine Design for Mechanical Engineering. In AI Magazine, 16(1), Spring 1995, AAAI-Press, 1995
- [Brusoni *et al.*, 98] V. Brusoni, L. Console, P. Terenziani und D. Theseider Dupré. A spectrum of definitions for temporal model-based diagnosis. Artificial Intelligence Journal, vol.102 n.1, 1998, pp 39-79.
- [Capelo *et al.*, 98] A. Capelo, L. Ironi und S. Tentoni. Automated Mathematical Modeling from Experimental Data. In IEEE Transactions on Systems, Man and Cybernetics 28(3), 1998
- [Cascio *et al.*, 99] F. Cascio, L. Console, M. Guagliumi, M. Osella, A. Panati, S. Sottano, D. Theseider-Dupré. Strategies for on-board diagnostics of dynamic automotive systems using qualitative models, AI Communications, June 1999.
- [Collins, 93] Collins, J. W., Process-based Diagnosis: An Approach to Understanding Novel Failures. Doctoral thesis, Institute for the Learning Sciences, Northwestern University, 1993.
- [Console, Torasso, 91] L. Console und P. Torasso. A Spectrum of Logical Definitions of Model-based Diagnosis. In Computational Intelligence, 7(3): 133-141, 1991. Auch in [Hamscher *et al.*, 92]
- [Cunis, 00] R. Cunis. Modellbasierte Diagnose mit DiaMon. In [Hotz *et al.*, 00]
- [Dague, 93] P. Dague. Symbolic Reasoning with Relative Orders of Magnitude. In Proceedings of the 13th International Joint Conference on Artificial Intelligence, 1993.
- [Dague *et al.*, 87] P. Dague, O. Raiman und P. Deves. Troubleshooting: When Modelling is the Trouble. In Proceedings of the 6th National Conference of the American Association for Artificial Intelligence, 1987. Auch in [Hamscher *et al.*, 92]
- [Davis, 84] R. Davis. Diagnostic Reasoning Based on Structure and Behavior. Artificial Intelligence, 1984. In: [Bobrow, 84]. Auch in [Hamscher *et al.*, 92]
- [Davis, 90a] E. Davis. Order of Magnitude Reasoning in Qualitative Differential Equations. In [Weld, de Kleer, 90]. Morgan Kaufmann, 1990.
- [Davis, 90] E. Davis, Representations of Commonsense Knowledge, Morgan Kaufmann Publishers, Inc.; San Mateo/CA 1990
- [de Jong, 99] H. de Jong: „Comparative environment construction: A technique for the comparative analysis of dynamical systems“. Artificial Intelligence, Vol. 115, Issue 2, pp. 145-214, 1999
- [de Kleer, 86] J. de Kleer. An Assumption-Based TMS. Artificial Intelligence, 28(2), 1986.
- [de Kleer, 91] J. de Kleer. Focusing the Diagnosis Engine. In Proceedings of the 9th National Conference of the American Association for Artificial Intelligence, 1991.
- [de Kleer, Brown, 84] J. de Kleer und J.S. Brown. A Qualitative Physics Based on Confluences. In: [Bobrow, 84]. Auch in [Weld, de Kleer, 90]
- [de Kleer, Forbus, 93] K. Forbus und J. de Kleer. Building Problem Solvers. MIT Press, Cambridge, Ma, 1993
- [de Kleer, *et al.*, 92] J. de Kleer, A. Mackworth und R. Reiter. Characterizing Diagnoses and Systems. Artificial Intelligence, 56, 1992
- [de Kleer, Williams, 87] J. de Kleer und B.C. Williams. Diagnosing Multiple Faults. Artificial Intelligence, 32(1), 1987. Auch in [Hamscher *et al.*, 92]
- [de Kleer, Williams, 89] J. de Kleer und B.C. Williams. Diagnosis with Behavioral Modes. In Proceedings of the 11th International Joint Conference on Artificial Intelligence, 1989. Auch in [Hamscher *et al.*, 92]

- [Dressler, 96] O. Dressler, On-line Diagnosis and Monitoring of Dynamic Systems based on Qualitative Models and Dependency-based Diagnostic Engines. In: Proceedings of the European Conference on Artificial Intelligence (ECAI-96), John Wiley & Sons, 1996.
- [Dressler, Farquhar, 89] O. Dressler und A. Farquhar. Problem Solver Control over the ATMS. In Proc. of GWAI, 1989.
- [Dressler, Struss, 96] O. Dressler und P. Struss. The Consistency-based Approach to Automated Diagnosis of Devices. In: Brewka, G. (ed.), Principles of Knowledge Representation, CSLI Publications, Stanford, pp. 267-311, 1996.
- [Falkenhainer, Forbus, 91] B. Falkenhainer und K.D. Forbus. Compositional Modeling of Physical Systems. Artificial Intelligence, 51(1-3), Oktober 1991. Auch in: [Faltings, Struss, 92].
- [Faltings, Struss, 92] B. Faltings und P. Struss. Recent Advances in Qualitative Physics. MIT Press, 1992.
- [Forbus, 84] K. D. Forbus. Qualitative Process Theory. In: [Bobrow, 84]. Auch in [Weld, de Kleer, 90]
- [Forbus *et al.*, 98] K. Forbus, J. Everett, L. Ureel, M. Brokowski, J. Baher und S. Kuehne. Distributed Coaching for an Intelligent Learning Environment. 12th International Workshop on Qualitative Reasoning, Cape Cod, Ma, 1998
- [Forbus, Falkenhainer, 92] K.D. Forbus und Falkenhainer W.B. Self-explanatory Simulations: Integrating Qualitative and Quantitative Knowledge. In [Faltings, Struss, 92].
- [Friedrich *et al.*, 91] G. Friedrich, G. Gottlob und W. Nejdl. Formalizing the Repair Process. In Working Notes of the 2nd International Workshop on Principles of Diagnosis.
- [Friedrich *et al.*, 92] G. Friedrich, G. Gottlob und W. Nejdl. Physical Impossibility Instead of Fault Models. In [Hamscher *et al.*, 92]
- [Greiner *et al.*, 89] R. Greiner, B. Smith und R. Wilkerson. A Correction to the Algorithm in Reiter's Theory of Diagnosis. Artificial Intelligence, 41(1), November 1989. Auch in [Hamscher *et al.*, 92]
- [Guckenbiehl, Münker, 00] T. Guckenbiehl und B. Münker. Überwachung und Diagnose in Färbereianlagen. In [Hotz *et al.*, 00]
- [Hamscher, 91] W. Hamscher. Modeling Digital Circuits for Troubleshooting. Artificial Intelligence, 51(1-3), 1991. Auch in [Hamscher *et al.*, 92]
- [Hamscher *et al.*, 92] W. Hamscher, J. de Kleer und L. Console (Hg.). Readings in Model-based Diagnosis: Diagnosis of Designed Artifacts Based on Descriptions of their Structure and Function. Morgan Kaufmann, San Mateo, 1992.
- [Hayes, 85] P. J. Hayes. The Second Naive Physics Manifesto. In J. Hobbs und R.C. Moore (Hg.), Formal Theories of the Commonsense World, Seite 1-36. Ablex, Norwood, NJ, 1985. Auch in [Weld, de Kleer, 90]
- [Heller, Struss, 96] U. Heller und P. Struss. Transformation of Qualitative Dynamic Models – Application in Hydroecology. In: 10th International Workshop on Qualitative Reasoning (QR-96), TR-96-01, AAAI Press, Stanford, 1996. (pp. 83-92)
- [Hellerstein, 92] J. Hellerstein. Obtaining Quantitative Predictions from Monotone Relationships. In [Faltings, Struss, 92].
- [Hotz *et al.*, 00] L. Hotz, T. Guckenbiehl, P. Struss. Intelligente Diagnose in der industriellen Anwendung. Shaker Verlag, Aachen, 2000
- [Inderst *et al.*, 95] R. Inderst, P. Struss, P. und O. Dressler. Automatische Testgenerierung für Weichenschaltungen auf der Basis qualitativer Modellierung, Tagungsband zum Treffen der GI-Fachgruppe 1.5.2 "Diagnostik und Klassifikation", Goslar, 1995
- [Iwasaki, 92] Y. Iwasaki. Reasoning with multiple abstraction models. In [Faltings, Struss, 92].
- [Kuipers, 86] B. Kuipers. Qualitative Simulation. Artificial Intelligence, 29(3), 1986. Auch in [Weld, de Kleer, 90]
- [Kuipers, 94] B. Kuipers. Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge. MIT Press, Cambridge, Massachusetts, 1994.
- [Lange *et al.*, 96] H. Lange, R. Möller und B. Neumann. Avoiding Combinatorial Explosion in Automatic Test Generation: Reasoning about Measurements is the Key. In Proc. KI'96, Dresden, September 1996.
- [Mavrouniotis, Stophanopoulos, 87] M.L Mavrouniotis und G. Stephanopoulos. Reasoning with Orders of Magnitude and Approximate Relations. In Proceedings of the 6th National Conference of the American Association for Artificial Intelligence, 1987.
- [McIlraith *et al.*, 99] S. McIlraith, G. Biswas, D. Clancey und V. Gupta. Towards Diagnosing Hybrid Systems. 13th International Workshop on Qualitative Reasoning. Loch Awe, Scotland, 1999

- [McIlraith, Reiter, 92] On Tests for Hypothetical Reasoning. In [Hamscher *et al.*, 92]
- [Mosterman *et al.*, 98] P. Mosterman, G. Biswas und E. Manders. A comprehensive framework for model-based diagnosis. 9th International Workshop on Principles of Diagnosis, Cape Cod, USA, 1998
- [Nayak, 95] P. Nayak. Automated Modeling of Physical Systems. Springer-Verlag, Berlin, 1995
- [Neitzke, Neumann, 94] Simulating Physical Systems with Relative Descriptions of Parameters. European Conference on Artificial Intelligence (ECAI-94), Amsterdam 1994
- [Poole, 89] D. Poole. Normality and Faults in Logic-based Diagnosis. In Proceedings of the 11th International Joint Conference on Artificial Intelligence, 1989. Auch in [Hamscher *et al.*, 92]
- [Price, 98] Price, C.: Function-directed Electrical Design Analysis, AI in Engineering 12(4), pp. 445-456, 1998.
- [Price, 99] C. Price. Computer-Based Diagnostic Systems. Springer-Verlag, Berlin, 1995
- [Raiman, 91] O. Raiman. Order of magnitude Reasoning. Artificial Intelligence, 51(1-3), Oktober 1991.
- [Reiter, 87] R. Reiter. A Theory of Diagnosis from First Principles. Artificial Intelligence, 32(1):57-96, 1987. Auch in [Hamscher *et al.*, 92]
- [Rickel, Porter, 94] J. Rickel und B. Porter. Automated Modeling for Answering Prediction Questions: Selecting the Time Scale and System Boundaries. In Proceedings of the 12th National Conference of the American Association for Artificial Intelligence, 1994.
- [Sachenbacher, Struss, 00] M. Sachenbacher und P. Struss. Automated Determination of Qualitative Distinctions: Theoretical Foundations and Practical Results. In 14th International Workshop on Qualitative Reasoning, Morelia, Mexico, 2000. Auch in [Hotz *et al.*, 00]
- [Shen, Leitch, 92] Q. Shen und R. Leitch. Integrating Common-Sense and Qualitative Simulation by the Use of Fuzzy Sets. In [Faltings, Struss, 92].
- [Sachenbacher *et al.*, 00] M. Sachenbacher, P. Struss und C. Carlen. A Prototype for Model-based On-board Diagnosis of Automotive Systems. In AI Communications, 2/00
- [Struss, 88] P. Struss. Mathematical Aspects of Qualitative Reasoning. International Journal for Artificial Intelligence in Engineering, 3(3), 1988. Überarbeitete Version: Problems of Interval-Based Qualitative Reasoning. In [Weld, de Kleer, 90].
- [Struss, 92] P. Struss. What's in SD? Towards a Theory of Modeling for Diagnosis. In [Hamscher *et al.*, 92]. Morgan Kaufmann, 1992.
- [Struss, 94] P. Struss. Testing for Discrimination of Diagnoses. In 5th International Workshop on Principles of Diagnosis (DX-94), New Paltz, USA, 1994.
- [Struss, 97] P. Struss. Fundamentals of Model-Based Diagnosis of Dynamic Systems. 15th International Joint Conference on Artificial Intelligence (IJCAI-97), Nagoya, Japan, pp. 480-485, 1997.
- [Struss, Dressler, 89] P. Struss und O. Dressler. 'Physical Negation' – Integrating Fault Models into the General Diagnostic Engine. In Proceedings of the 11th International Joint Conference on Artificial Intelligence, 1989. Auch in [Hamscher *et al.*, 92]
- [Struss, Heller, 98] Struss, P., Heller, U.: Process-oriented Modeling and Diagnosis – Revising and Extending the Theory of Diagnosis from First Principles. In 9th International Workshop on Principles of Diagnosis (DX-98), Cape Cod, MA, USA, pp. 110-117, 1998.
- [Struss, Heller, 00] P. Struss und U. Heller. An Approach to Consistency-based Process Diagnosis. In [Hotz *et al.*, 00]
- [Struss *et al.*, 00] P. Struss, U. Heller, A. Malik und M. Sachenbacher. Modellbasierte Werkzeuge für Diagnose und Fehleranalyse von Fahrzeugsystemen. In [Hotz *et al.*, 00]
- [Stumptner, Wotawa, 99] M. Stumptner und F. Wotawa. Debugging Functional Programs. 16th International Joint Conference on Artificial Intelligence (IJCAI-99), Stockholm, Sweden, 1999
- [Sun, Weld, 93] Y. Sun und D. Weld. A Framework for Model-Based Repair. In Proceedings of AAAI-93, 182-187, Washington, D.C., July 1993.
- [Trave-Massuyes, Milne, 87] L. Trave-Massuyes und R. Milne. Gas Turbine Condition Monitoring Using Qualitative Model Based Diagnosis. In IEEE Expert Magazine, June 1997
- [Weld, 88] D.S. Weld. Comparative Analysis. Artificial Intelligence, 36, 1988.
- [Weld, Addanki, 92] D.S. Weld und S. Addanki. Task-Driven Model Abstraction. In [Faltings, Struss, 92].
- [Weld, de Kleer, 90] D. Weld und J. de Kleer. Readings in Qualitative Reasoning about Physical Systems. Morgan Kaufmann, San Mateo, CA, 1990.
- [Williams, 92] B.C. Williams. Interaction-based Invention: Designing Devices from First Principles. In [Faltings, Struss, 92].
- [Williams, Nayak, 96] B. Williams und P. Nayak, A Model-based Approach to Reactive Self-Configuring Systems. In 7th International Workshop on Principles of Diagnosis (DX96), Montreal, 1996.