

Zur Analyse fallbasierter Problemlöse- und Lernmethoden in Abhängigkeit von Charakteristika gegebener Aufgabenstellungen und Anwendungsdomänen

Klaus-Dieter Althoff, Agnar Aamodt

Abstract

Fallbasiertes Schließen (case-based reasoning; CBR) hat gegenüber anderen Methoden den Vorteil der inhärenten Kombination von Problemlösen und kontinuierlichem Lernen aus Erfahrung. Diese Kombination ist ein außerordentlich interessantes Thema, zu dem genügend Erfahrung vorliegt, um es systematisch zu untersuchen. Hierzu haben wir begonnen, ein Instrumentarium (framework) zur Analyse von CBR-Methoden zu entwickeln. Es beinhaltet eine explizite Ontologie grundlegender CBR-Aufgabentypen, Domänencharakteristika sowie Problemlöse- und Lern-Methodentypen. Die mit unserem Ansatz verbundene Methodologie kombiniert eine top-down Knowledge-Level-Analyse mit einer bottom-up „fallgetriebenen“ Analyse. Wir beschreiben neben der zugrunde liegenden Sichtweise unseres analytischen Instrumentariums seine Hauptkomponenten, die eingebettete Methodologie sowie Beispielstudien und ihre Beziehung zu unserem Ansatz.

1. Einführung

In den letzten Jahren ist im Bereich des Fallbasierten Schließens (case-based reasoning; CBR) ein substantieller Fortschritt erzielt worden. Probleme sind klarer identifiziert und Ergebnisse intensiver Forschungsarbeiten haben zu besseren Methoden des Retrievals, der Adaption und des Lernens geführt [21]. Als eine besondere Stärke fallbasierter Ansätze im Vergleich zu anderen Methoden erwies sich ihre inhärente Kombination aus Problemlösen und kontinuierlichem Lernen aus Erfahrung. Diese Kombination ist somit ein interessanter Untersuchungsgegenstand sowie Ansatzpunkt für eine detailliertere Analyse. Eine derartige Untersuchung von Problemlösen und Lernen erfordert ein analytisches Instrumentarium (analytic framework) zum Beschreiben, Analysieren und Vergleichen verschiedener Typen und Aspekte fallbasierter Methoden.

Der Integration von Lernen und Problemlösen können unterschiedliche Zielsetzungen zugrunde liegen und sie kann aus verschiedenen Sichtweisen heraus betrachtet werden. Ein Beispiel für eine Sichtweise ist „Begriffsbildung“ als Ziel und das Bilden und Verwenden von Operationalisierungskriterien in Relation zur jeweiligen Aufgabenstellung als Perspektive. Eine weitere Sichtweise ist „Performanzverbesserung“ (als Ziel) und das Verbessern der gesamten „Problemlösegeschwindigkeit“ für Computer und Mensch zusammen (als Perspektive). Ein drittes Beispiel ist „kontinuierliches Lernen durch Problemlöseerfahrungen“ (sustained learning) als Ziel und der Einfluß der jeweiligen Aufgabenstellung (application task) auf die Lernmethode als Perspektive. Weitere Beispiele ließen sich aufzählen und für jedes dieser Beispiele ein bestimmter „Überlappungsbereich“ zwischen Methoden des maschinellen Lernens (ML) einerseits und Problemlösemethoden (PS) andererseits identifizieren. Innerhalb dieses Überlappungsbereiches ist es möglich, Abhängigkeiten und sonstige Relationen zwischen ML- und PS-Methoden zu beschreiben und zu analysieren,

vorausgesetzt es steht ein geeignetes Instrumentarium zur Strukturierung dieses Bereiches zur Verfügung.

Wir haben damit begonnen, ein derartiges Hilfsmittel zu entwickeln, einschließlich einer Methodologie zur Definition und Verwendung einer geeigneten Struktur. Da wir fallbasierte Methoden untersuchen, ist unser Interesse durch die dritte oben erwähnte Zielsetzung bestimmt: kontinuierliches Lernen aus jeder Problemlöseerfahrung als natürlicher Bestandteil eines Problemlösers. Innerhalb einer breiteren Sicht auf die Integration von Lernen und Problemlösen ist es ebenso natürlich, eine Untersuchung von Lernen so "nahe" zur "Quelle des Lernens" dem "möglichen der konkreten Erfahrung" zu beginnen wie möglich. Unser Ansatz orientiert sich an einigen früheren Arbeiten wie z.B. der ähnlichkeitsorientierten Sichtweise von Richter und Wess [19], Althoffs Analyse technischer Diagnosesysteme [6], Aamodts Vergleich wissensintensiver fallbasierter Methoden [1], Armengols und Plasas Analyse fallbasierter Systemarchitekturen [12] sowie dem INRECA-Ansatz zur Systemevaluation [11]. Das von uns derzeit entwickelte analytische Instrumentarium geht in vielerlei Hinsicht über diese früheren Vorschläge hinaus. Es beinhaltet eine explizite Ontologie grundlegender fallbasierter Aufgabenstellungen, Anwendungscharakteristika und Methodentypen für Problemlösen und Lernen. Darin eingebettet ist eine Methodologie zur Kombination einer Knowledge-Level Top-Down-Analyse mit einer "fallgetriebenen" Bottom-Up-Analyse. In diesem Beitrag werden wir die grundlegenden Sichtweisen sowie unsere Hauptansatzpunkte darlegen, die Hauptbestandteile des Instrumentariums und der begleitenden Methodologie beschreiben sowie einige Beispielstudien und ihren Bezug zu unserem Instrumentarium vorstellen.

2. Ansatz

2.1. Knowledge-Level-Analyse

Problemlöse- und Lernverhalten kann sinnvoll beschrieben werden durch die *Ziele*, die erreicht, die *Aufgaben*, die gelöst, die *Methoden*, die verwendet werden sollen, sowie das *Wissen über die Anwendbarkeit*, das die Methoden benötigen.

Eine derartige Systembeschreibung wird häufig als Knowledge-Level-Beschreibung bezeichnet und neuere Arbeiten im Bereich der Wissensakquisition (z.B. [20;25]) haben eindeutig die Nützlichkeit dieses Ansatzes unterstrichen. Im Laufe der Jahre hat die ursprüngliche Idee des Knowledge-Levels einige Änderungen erfahren, ausgehend von Newells sehr intentionalen, zweckorientierten Art der Systembeschreibung [17], hin zu einer strukturierteren und damit nützlicheren Art der Beschreibung.

Das Verwenden einer Knowledge-Level-Sichtweise zur Analyse integrierter PS-ML-Systeme ermöglicht die Beschreibung von Methoden und Systemen sowohl von einem generellen (intensionalen) Standpunkt aus als auch von einem fallspezifischen (extensionalen). Eine generelle Beschreibung assoziiert eine Methode mit beschreibenden Begriffen und Relationen innerhalb einer Ontologie von Aufgabentypen, Domänencharakteristika und Methodentypen. Methoden können "fallbasiert" dadurch verstanden werden, daß sie zu bereits beschriebenen/implementierten Systemen in Beziehung gesetzt werden ("CBR wird mit regel- und modellbasiertem Schließen kombiniert wie in CREEK" [2]; "Ein Entscheidungsbaum wird generiert wie in INRECA" [10;23]; "Das Ähnlichkeitsmaß wird angepasst wie in PATDEX" [22]; "Determinationsregeln werden generiert und verwendet wie in MOLTKE" [6] etc.). Sind eine Reihe von Systemen entwickelt/beschrieben, so können wir eine Systembeschreibung auf dem Knowledge-Level auswählen/intanziiieren durch die kombinierte Verwendung genereller und fallbasierter Deskriptoren. Unser Ziel ist es, durch die Integration dieser beiden Sichtweisen Top-Down- und Bottom-Up-Analyse sowie -Modellierungsmethoden effektiv miteinander zu kombinieren.

Aus der Knowledge-Engineering-Sicht ermöglicht dies \mathcal{D} ausgehend von dem Verständnis einer realen Aufgabenstellung und der jeweiligen Domänencharakteristika \mathcal{D} eine Symbol-Level-Systembeschreibung¹ auszuwählen bzw. eine ausgewählte zu instanziiieren. Natürlich ist eine Knowledge-Level-Beschreibung nicht detailliert genug, um ein System auf dem Symbol-Level zu beschreiben bzw. zu spezifizieren. Wir verwenden daher einen zusätzlichen Fokus und ein analytisches Hilfsmittel zur weiteren Detaillierung der Knowledge-Level-Beschreibung.

2.2. Ähnlichkeit als Fokus

Die hier zugrunde gelegte Sichtweise betrachtet alle CBR-Methoden vom „Standpunkt der Ähnlichkeit“. Probleme \mathcal{P} können verstanden werden als der Prozeß der initialen Ähnlichkeitsbewertung (Fallretrieval) gefolgt von einer detaillierteren Ähnlichkeitsbewertung (Falladaptation). Lernen (Fallextraktion, Fallindizierung, Aktualisierung generellen Wissens) kann beschrieben werden, indem es in Beziehung gesetzt wird zu einer späteren Ähnlichkeitsbewertung, d.h. zu einem pragmatischen Lernziel. In Anlehnung an [19] betrachten wir Ähnlichkeit als ein a posteriori Kriterium, so daß jegliche Ähnlichkeitseinschätzung eines Falles so lange hypothetisch ist, bis dieser sich als „ähnlich“ erwiesen hat. Natürlich sollen die verwendeten Retrievalmethoden versuchen, die Differenz zwischen hypothetischer und realer Ähnlichkeitsbewertung zu minimieren, sobald versucht wurde, den Fall wiederzuverwenden. Generelles Domänenwissen (general domain knowledge) kann nun als ein Hilfsmittel verstanden werden, die Unsicherheit dieser initialen Ähnlichkeitsbewertung zu reduzieren mit Blick auf die abschließende Ähnlichkeitsbewertung, die nach erfolgreicher Wiederverwendung eines (möglichlicherweise geänderten) Falles durchgeführt werden kann.

2.3. Charakteristika von Aufgabenstellung und Anwendungsdomäne

Wir betrachten ein breites Spektrum von Anwendungsdomänen, ausgehend von Anwendungen mit wohldefinierten Begriffen und Relationen \mathcal{D} z.B. Diagnose rein technischer Systeme \mathcal{D} bis hin zu „offenen“ Anwendungsbereichen mit weniger klaren Begriffen und unsicheren Relationen wie z.B. vielen medizinischen Diagnoseanwendungen. Dies ist ein wesentlicher Aspekt unseres Ansatzes, da wir die Charakteristika der realen Aufgabenstellung sowie des zugehörigen Wissens (Was ist zu tun?) in Beziehung setzen wollen zu den Methoden (Wie ist es zu tun?), die das Lösen des Problems auf Basis des gegebenen Wissens ermöglichen sollen. Ausgangspunkt ist jeweils die reale Umgebung, in der das System eingesetzt werden soll. So ist z.B. in der medizinischen Diagnose die reale Aufgabenstellung nicht nur eine Klassifikationsaufgabe [9]. Sollen die anfallenden Hauptaufgaben praktisch bewältigt werden, müssen ebenso Planungsaufgaben (wie das Erstellen und kontinuierliche Anpassen von Untersuchungsprotokollen) sowie Vorhersageaufgaben (wie das Einschätzen der Konsequenzen möglicher Behandlungen) berücksichtigt werden.

Im nächsten Abschnitt werden wir die Kernbestandteile unseres Ansatzes vorstellen mit einem Schwerpunkt auf der Knowledge-Level-Beschreibung und -Analyse sowie der kombinierten top-down-/bottom-up-orientierten Methodologie. Das Einbeziehen des „Ähnlichkeitsfokus“ ist Teil laufender Forschungsarbeiten.

3. Analytisches Instrumentarium und Methodologie

3.1. Hauptbestandteile

Aus einer abstrakten Sicht heraus kann ein allgemeiner CBR-Zyklus durch vier Aufgaben beschrieben werden [5]: Bereitstellen des ähnlichsten Falles (*Retrieve*), Wiederverwenden des enthaltenen

¹ Eine Symbol-Level-Beschreibung korrespondiert zu einer operationalen Beschreibung

Wissens zur Problemlösung (*Reuse*), Anpassen der vorgeschlagenen Lösung (*Revise*) sowie Bewahren für zukünftige Probleme wichtiger Erfahrungen (*Retain*) (s. Abb. 1).

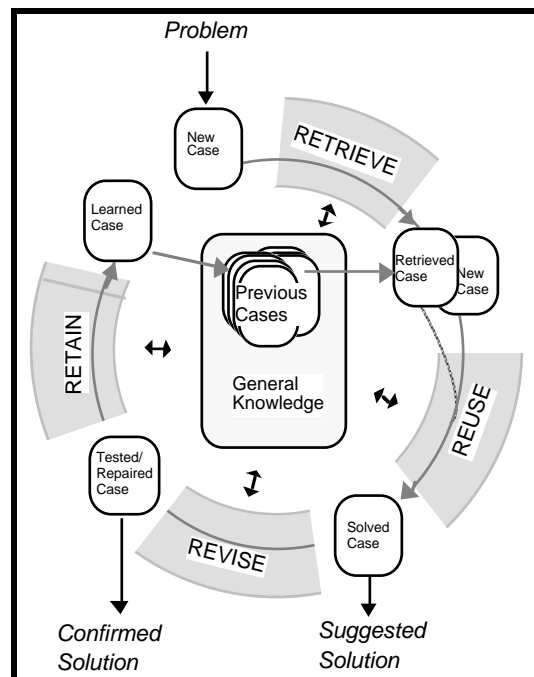


Abb. 1 Ⓝ CBR-Zyklus

Wir werden im folgenden auf Aufgaben im Sinne internen Schlußfolgerungsaufgaben näher eingehen. Diese sind von Anwendungsaufgaben wie Diagnose oder Planung zu unterscheiden. Die vier CBR-Aufgaben bringen eine Reihe von spezifischen Unteraufgaben mit sich. Eine initiale Problembeschreibung (oben in Abb. 1) definiert einen neuen Fall. Dieser wird in der *Retrieve*-Aufgabe dazu verwendet, unter den bereits bekannten Fällen einen ähnlichen Fall zu finden. In der *Reuse*-Aufgabe wird der gefundene Fall mit dem neuen Fall kombiniert und ergibt einen gelösten Fall, d.h. einen Lösungsvorschlag für das initiale Problem. Die *Revise*-Aufgabe überprüft diese Lösung durch Testen in der realen Umgebung bzw. durch einen Lehrer, der die Lösung gegebenenfalls anpaßt. Das hier einbezogene Feedback ist wichtig für die Lernphase, die hauptsächlich innerhalb der *Retain*-Aufgabe stattfindet. Durch Anpassen von Fallbasis und generellem Domänenwissen werden hier nützliche Erfahrungen bewahrt. Generelles Wissen ist üblicherweise in alle CBR-Prozesse einbezogen, wobei das jeweilige Ausmaß vom Typ der verwendeten CBR-Methode abhängt. Mit generellem Wissen meinen wir hier generelles, domänenabhängiges Wissen im Gegensatz zu spezifischem Domänenwissen innerhalb von Fällen. Soll z.B. ein Patient durch Wiederverwenden des Falles eines früheren Patienten diagnostiziert werden, dann könnte ein anatomisches Modell zusammen mit kausalen Beziehungen zwischen pathologischen Zuständen als generelles Wissen innerhalb eines CBR-Systems funktionieren, ebenso eine Menge von Regeln.

Knowledge-Level-Analyse ist ein allgemeiner Ansatz zur Systemanalyse. Er ist daher auch anwendbar für die Analyse realer Aufgabenstellungen und Domänen wie z.B. in den oben erwähnten Wissensakquisitionsmethodologien sowie interner Aufgaben der Problemlöse- und Lernkomponente. Unser analytisches Instrumentarium umfaßt daher einen „Aufgabe-Methode-Domänenwissen“-Ansatz zur Analyse sowohl von realen Anwendungsaufgaben als auch CBR-Schlußfolgerungsaufgaben. Es gibt folgende Beziehungen: Methoden aus der Anwendungsanalyse (Wie ist ein technisches Diagnoseproblem zu lösen? Wie ist der nächste Test zu bestimmen?) dekomponieren entweder die Anwendungsaufgabe in Unteraufgaben oder lösen sie direkt. In beiden Fällen haben diese Methoden

Aufgaben auf der Schlußfolgerungsebene zur Folge (z.B. Problemlösen und Lernen aus Erfahrung). Wir werden uns nun näher mit Schlußfolgerungsaufgaben beschäftigen.

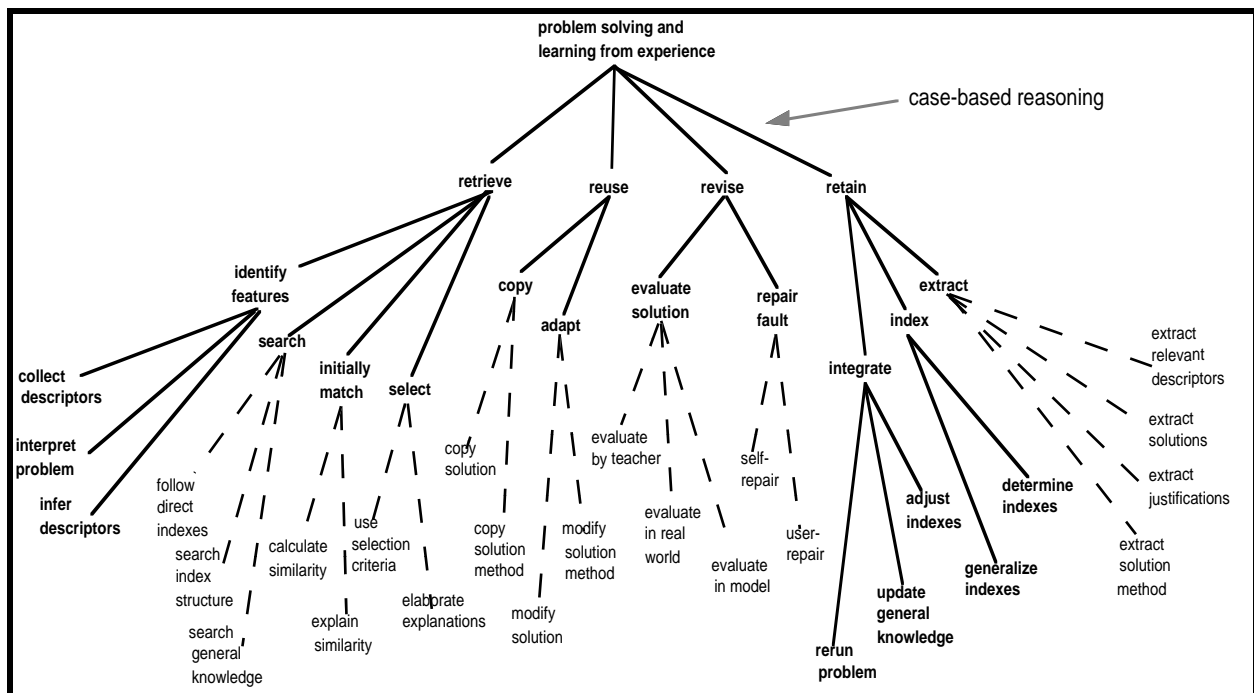


Abb. 2 D Dekomposition von CBR-Aufgaben und -Methoden

Die Aufgaben aus Abb. 1 sind in Abb. 2 weiter dekomponiert. Aufgaben sind fett gedruckt, zur Unterscheidung von den normal gedruckten Methoden. Die Kanten zwischen Aufgabenknoten (fett gedruckt) bedeuten Aufgabendeckompositionen, d.h. Part-of-Relationen. Die Kanten zwischen Aufgaben und Methoden (gestrichelt gedruckt) stehen für alternative Methoden zur Lösung der Aufgabe. Die Relationen sind von oben nach unten gerichtet. Die oberste Aufgabe ist *problem solving and learning from experience* und die Methode zur Lösung ist eine CBR-Methode. Dadurch wird die oberste Aufgabe in die vier CBR-Hauptaufgaben aus Abb. 1 zerlegt. Alle vier Teilaufgaben sind zu behandeln, um die oberste Aufgabe zu erfüllen. Die *Retrieve*-Aufgabe wiederum ist auf die gleiche Weise zerlegt in die Aufgaben *identify* (bestimme relevante Deskriptoren), *search* (um eine Menge fröhlicher Fälle zu finden), *initially match* (vergleiche relevante Deskriptoren mit fröhlichen Fällen) und *select* (wähle den ähnlichsten Fall). Alle Aufgabenzerlegungen in Abb. 2 sind vollständig, d.h. auf diesem Detaillierungsgrad reicht die Menge der Unteraufgaben aus, um die jeweilige Aufgabe zu lösen. Abb. 2 zeigt keinerlei Kontrollinformationen hinsichtlich der Unteraufgaben. Dies ist Teil der Problemlösungsmethode. Eine Methode spezifiziert den Algorithmus zur Identifikation und Kontrolle von Unteraufgaben bzw. zur Verwendung von erforderlichem Wissen und benötigter Information. Die angegebenen Methoden bezeichnen abstrakte Methodenklassen, aus denen eine oder mehrere Methoden ausgewählt werden können. Die dargestellte Methodenmenge ist unvollständig, d.h. eine Aufgabe kann mit einer Methode gelöst werden, mit einer Kombination mehrerer Methoden oder mit zusätzlichen Methoden (Abb. 4).

Die obige Dekompositionsstruktur für CBR-Aufgaben und -Methoden bildet die Basis für unseren analytischen Ansatz. Sie muß weiter detailliert werden, Charakterisierungen von Domänenwissenstypen müssen ergründet sowie Abhängigkeiten zwischen den verschiedenen Wissenstypen identifiziert werden.

3.2 Methodologie

Wie bereits erwähnt, unser grundsätzlicher methodologischer Ansatz ist die Kombination einer top-down-orientierten Analyse von Anwendungsaufgaben, Domänenwissensbeschreibungen und Methoden mit einer bottom-up-orientierten fallbasierten Analyse existierender Systeme. Ziel ist die Entwicklung eines kohärenten Instrumentariums und einer Beschreibungssprache, die es erlauben, abstrakte Analysen zu detaillieren bzw. von Beispielsystemen zu generalisieren.

Die Grundzüge des Ansatzes sind wie folgt. Wir *beschreiben* sowohl die Möglichkeiten und Grenzen von CBR-Systemen mit Hilfe technischer und ergonomischer Kriterien als auch die Charakteristika von Domänen und Anwendungsaufgaben auf der Basis von Domänen- und Aufgabenkriterien [9;11]. Beispiele für Domänen- und Aufgabenkriterien sind *size*¹, *theory strength*², *openness*³, *change over time*⁴ und *complexity*⁵, während *case and knowledge representation*, *similarity assessment*, *validation* und *data acquisition and maintenance* wichtige technische und ergonomische Kriterien sind. Wir analysieren die zugrunde liegenden Methoden und Domänen- bzw. Aufgabencharakteristika, indem wir die technischen/ergonomischen Kriterien mit den Domänen-/Aufgabenkriterien in Beziehung setzen. Abb. 3 gibt ein Beispiel wie CBR-Systeme mit Domänenkriterien assoziiert werden können. Die Kombination einer derartigen Beschreibung mit einer allgemeinen Analyse basierend auf der Struktur aus Abb. 2 erwies sich für die Abschlussequaluation des INRECA-Systems als nützlich [7;11].

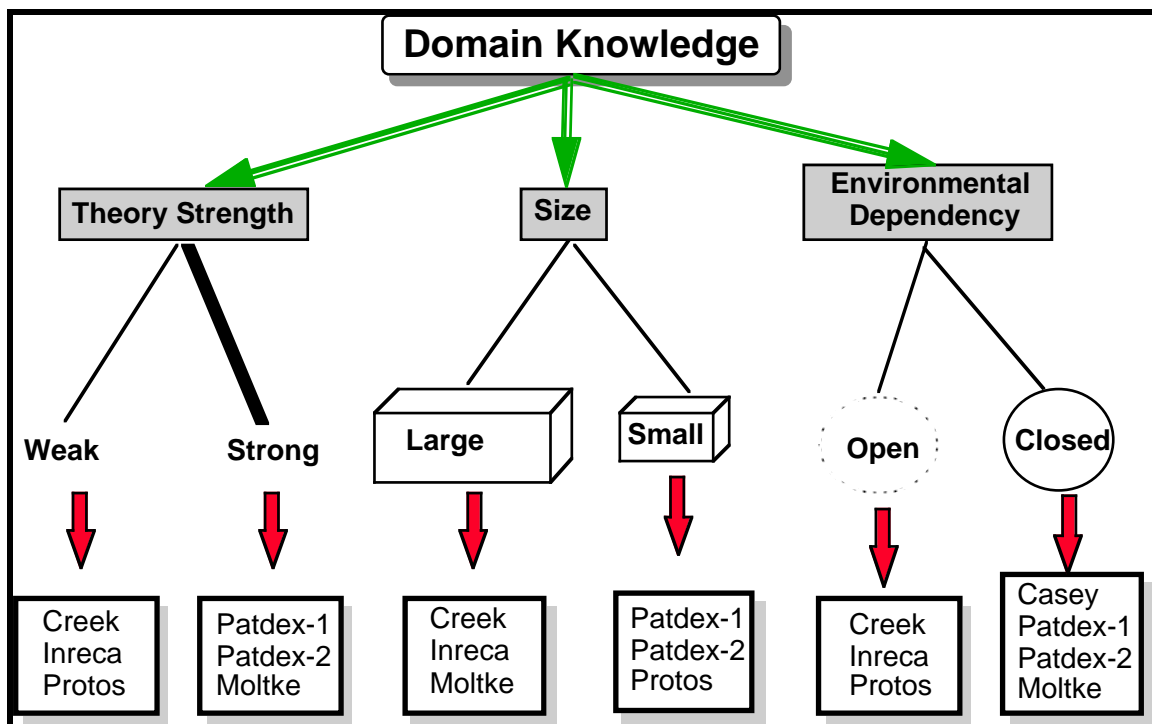


Abb. 3 Domänenkriterien in Relation zu existierenden Systemen

Aus Sicht des Software und Knowledge Engineering versuchen wir, nichtfunktionale Systemeigenschaften zur Entwicklung von (CBR-)Systemen einzusetzen. Da wir uns auf CBR-Systeme konzen-

1 *Size* bedeutet den Umfang einer Domäne gemessen in expliziten Wissensentitäten
 2 *Theory strength* bedeutet den Grad der Sicherheit der vorkommenden Domänenrelationen
 3 *Openness* bezeichnet den Grad der Abhängigkeit von der Umgebung
 4 *Change over time* bezeichnet die Veränderlichkeit einer Domäne über die Zeit
 5 *Complexity* bedeutet den Umfang unterschiedlicher taxonomischer Relationen einer Domäne

trieren, können wir präzisere Kriterien formulieren als für Softwaresysteme im allgemeinen. Zudem kombinieren wir diese systemorientierten eher technischen Kriterien mit einer Analyse von Anwendungsdomänen, mit denen wir Erfahrung haben. Einerseits kann Feedback aus den Anwendungssystematisch in eine Bewertung basierend auf Domänen- und Aufgabenkriterien umgesetzt werden. Andererseits können aus CBR-Systemen extrahierte Methoden mit den Ergebnissen dieser Kriterien assoziiert werden (Abb. 3). Das Ziel ist wiederum die Lücke zu schließen zwischen abstrakten Beschreibungen einerseits und konkreten Systemen andererseits. Allgemeine Knowledge-Level-Analyse und fallbasierte Analyse werden integriert, um auf der Basis einer gemeinsamen Sprache und eines einheitlichen Modells anwendungsbezogene Instrumentarien für bestimmte Systemtypen zu entwickeln. Hierzu werden technische und ergonomische Kriterien mit Domänen- und Aufgabenkriterien kombiniert.

Die intendierte Verwendung unseres Ansatzes zur Analyse und Entwicklung integrierter Problemlöse- und Lernsysteme kann zusammenfassend betrachtet werden als ein Beantworten der nachfolgend aufgeführten (leicht modifizierten) Fragen aus [14]:

- ¥ Inwiefern ist die zu bewertende CBR-Methode eine Verbesserung, eine Alternative oder eine Ergänzung gegenüber anderen Methoden? Deckt sie mehr Situationen ab, liefert sie eine größere Vielfalt gewünschter Verhaltensweisen, ist sie effizienter hinsichtlich Zeitbedarf/Platzverbrauch oder modelliert sie menschliches Verhalten nützlicher bzw. detaillierter?
- ¥ Wie sehen die zugrunde liegenden Voraussetzungen aus? Sind alle Designentscheidungen gerechtfertigt? Basiert die Methode auf anderen integrierten Methoden? Umfasst sie andere Methoden?
- ¥ Worauf kann die Methode angewendet werden? Wie ausbaufähig ist sie? Kann sie auf komplexere Probleme angewendet werden?
- ¥ Warum funktioniert die Methode? Unter welchen Umständen funktioniert sie nicht?
- ¥ Wie gestaltet sich die Beziehung der Methode zur zugrunde liegenden Klasse der aktuellen Aufgabe? Kann diese Beziehung so klar beschrieben werden, daß die Methode für die gesamte Aufgabenklasse verwendet werden kann?

4. Beispielstudien

Wir illustrieren unseren (bottom-up) Ansatz am Beispiel zweier Untersuchungen von CBR-Methoden. Abb. 4 zeigt die für die *Retain*-Aufgabe eine Beschreibung der Lernmethode, die von einer Reihe von analysierten Systemen (geeignet) abstrahiert wurde [11]. Das zweite Beispiel bezieht sich auf die oberste Aufgabe in Abb. 2 (Problemlösen und Lernen aus Erfahrung) sowie Methoden zur Integration von CBR und Schließen auf Basis generellen Wissens.

Quellen des Algorithmus: CREEK, MOLTKE und PROTOS.

```

IF no_similar_past_case(current_case)
    THEN construct_new_case;
    ELSE lazy_generalise(old_case);
IF current_case_successful
    THEN integrate_into_successful_cases;
    ELSE integrate_into_total_problem_cases;
DO adaptation UNTIL system_behaves_as_wanted

```

Abb. 4 Die abstrahierte Methode für die *Retain*-Aufgabe

In Abb. 4 führen die THEN- und ELSE-Zweige jeweils zu Dekompositionsmethoden für die *Retain*-Aufgabe, d.h. zu einer geeigneten Auswahl bzw. Verknüpfung der *Extract*-, *Index*- und *Integrate*-Aufgaben (Abb.2).

Tabelle 1 faßt eine Menge von Methoden aus einer Reihe von Forschungssystemen hinsichtlich des Grades ihrer Integration von CBR und Schließen auf Basis generellen Wissens zusammen.

	BOLERO	CASEY	CcC+	CREEK	INRECA	MOLTKE	PROTOS	S ³⁺ /INRECA
Toolbox	●	●	●		●	●	●	●
Kooperation	●	●	●	●	●	●		●
Werkbank	●			●	●	●		●
Volle Integration				●	●			●

Tabelle 1. Aufgabe: Problemlösen und Lernen aus Erfahrung; Methode: Kombination fallspezifischen und generellen Wissens

Die unterschiedlichen Integrationsstufen sind wie folgt beschrieben:

- ¥ *Toolbox*: Dies ist die einfachste Integrationsstufe, wo die Integration beschränkt ist auf die gemeinsame Nutzung von Teilen der Wissensbasis. Es wird eine Methode ausgewählt und als einzelne Methode ausgeführt. Der Austausch von Zwischenergebnissen ist nicht möglich.
- ¥ *Kooperation*: Es können verschiedene Methoden verwendet werden. Zwischenergebnisse werden über einen gemeinsamen Repräsentationsformalismus ausgetauscht.
- ¥ *Werkbank*: Neben dem Umschalten zwischen verfügbaren Methoden können diese zu einer kombinierten (einzelnen) Methode aggregiert werden. So könnte z.B. ein Diagnosesystem regelbasiert klassifizieren und fallbasiert den jeweils nächsten Test auswählen.
- ¥ *Volle Integration*: Dies ist die höchste Integrationsstufe, wo unterschiedliche Methoden in einem neuen Algorithmus „aufgehen“ und als nicht mehr separierbar betrachtet werden. Der Wechsel von Methoden während des Schlußfolgerungsprozesses bleibt somit vor dem Benutzer verborgen.

Während CASEY modellbasiertes Schließen mit CBR beschleunigt [16], kombiniert BOLERO fallbasierte Testauswahl mit regelbasierter Diagnose [18]. CREEK integriert CBR mit erklärungs-basiertem, wissensintensivem Schließen innerhalb einer Wissensstruktur [2]. MOLTKE ergänzt regelbasierte Diagnose mit CBR zur Behandlung von Ausnahmen [6]. INRECA kombiniert CBR mit integrierten induktiven k-d-Bäumen zum Filtern von Fällen und Lernen von Präferenzen [10;23]. Darüber hinaus wurde innerhalb von S³⁺/INRECA ein auf generellem Wissen basierendes Service-Support-System integriert [11]. CcC+ ist ein fallbasiertes Klassifikationssystem, das mit den anderen Teilsystemen der D3-Diagnose-Toolbox kooperieren kann [15]. Bei PROTOS handelt es sich um ein alleinstehendes CBR-System für Klassifikation, Wissensakquisition und Lernen [13].

5. Ausblick

Die langfristige Zielsetzung unserer Forschungsarbeit ist zweifach. Das erste Ziel bezieht sich auf die KI als eine experimentelle Wissenschaft, d.h. ein analytisches Instrumentarium aufzubauen, das es erlaubt, ein verbessertes Verständnis für das PS-ML-Integrationsproblem zu entwickeln, ausgehend von einem Fokus auf CBR-Methoden. Ein verbessertes Verstehen des „CBR-Raumes“ bildet die Basis für Erweiterungen wie z.B. durch Lernmethoden für „Eager-Generalisation“ (als Ergänzung zur „Lazy-Generalisation“ in Abb. 4). Das Instrumentarium soll als „Sprache“ und als eine Menge von Prinzipien für analytische und vergleichende Studien verschiedener Systeme in unterschiedlichen Forschungsgruppen dienen. Das zweite Ziel bezieht sich auf das Knowledge Engineering wissensbasierter Systeme, nämlich die ML-PS-Integration als eine wichtige Aufgabe zu

formulieren, die vom Knowledge Engineer bewältigt werden muß, und eine Methode für ihre Handhabung bereitzustellen.

Bislang ist nur eine vereinfachte Version unseres Ansatzes teilweise getestet worden; nichtsdestotrotz glauben wir, Schritte in die richtige Richtung unternommen zu haben. Das Kernstück unseres Ansatzes ist die Kombination einer Top-Down-Analyse basierend auf einer Aufgaben-Methoden-Dekomposition sowie eine Bottom-Up-Analyse existierender Systeme. Es ist geplant, die aus unserer Analyse resultierenden Systembeschreibungen in ein fallbasiertes Informationssystem einzubringen [8;9], das Hinweise auf der Basis von Fällen über CBR-Anwendungen und -systeme gibt, eingebettet in eine allgemeine Methodologie zur CBR-System-Dekomposition, -Analyse und -Entwicklung. Eine weitere Verwendung unseres Ansatzes geschieht als Modellierungshilfsmittel zur prinzipiellen Integration von wissensbasierten Systemen in Informations- und Datenbanksysteme, basierend auf der jeweiligen „Rolle“, die diese (Teil-)Systeme innerhalb eines entscheidungsunterstützenden (Gesamt-)Systems spielen [4]. Eine wichtige Erweiterung unseres Ansatzes wird die Einbeziehung und Detaillierung des in Abschnitt 2.2 nur angedeuteten „Ähnlichkeitsfokus“ sein.

6. Danksagung

Teile der hier beschriebenen Forschungsarbeit wurden im Rahmen des INRECA-Projektes (Esprit Nr. 6322) durchgeführt und profitierten insbesondere von Diskussionen zwischen den Projektpartnern AcknoSoft (Paris), IMS (Dublin), tecInno (Kaiserslautern) und der Universität Kaiserslautern. Wir bedanken uns ebenso bei Brigitte Bartsch-Spörl, Enric Plaza und Ralph Traphöner für ihre Beiträge während einer Vielzahl von Diskussionen zu unterschiedlichen Aspekten dieses Artikels sowie bei Pinar Ozturk für hilfreiche Kommentare zu früheren Versionen dieses Beitrages.

7. Literatur

- [1] Aamodt, A. (1991). *A knowledge-intensive approach to problem solving and sustained learning*. Ph.D. Dissertation, University of Trondheim, Norwegian Institute of Technology
- [2] Aamodt, A. (1994). Explanation-Driven CBR. In: [24], 274-288
- [3] Aamodt, A. (1995). Knowledge Acquisition and learning by experience: The role of case-specific knowledge. In: Y. Kodratoff, & G. Tecuci (eds.), *Integration of Knowledge Acquisition and Machine Learning*, Kluwer Academic Publishers, forthcoming, 197-245
- [4] Aamodt, A. and Nygård, M. (1995). Different roles and mutual dependencies of data, information, and knowledge - an AI perspective on their integration. *Accepted for publication in Data and Knowledge Engineering*
- [5] Aamodt, A. and Plaza, E. (1994). CBR: Foundational Issues, Methodological Variations and System Approaches. *AI Communications Vol. 7, No. 1*, 39-59
- [6] Althoff, K.-D. (1992). *Eine fallbasierte Lernkomponente als integrierter Bestandteil der MOLTKE-Werkbank zur Diagnose technischer Systeme*. Dissertation, Universität Kaiserslautern; auch: Sankt Augustin: infix Verlag
- [7] Althoff, K.-D., Auriol, E., Barletta, R. & Manago, M. (1995). *A Review of Industrial CBR Tools*. AI Intelligence, Oxford, UK
- [8] Althoff, K.-D. & Bartsch-Spörl, B. (1995). Call for Participation: Decision Support for Case-Based Applications. *Initiative for systematically collecting information about existing CBR systems and applications*. BSR Consulting (München) & Universität Kaiserslautern

- [9] Althoff, K.-D. & Bartsch-Spörl, B. (1996). Decision Support for Case-Based Applications. *Wirtschaftsinformatik 1/96, Sonderheft Fallbasierte Entscheidungsunterstützung* (Hrsg.: D. Ehrenberg)
- [10] Althoff, K.-D., Bergmann, R., Globig, C., Wess, S., Auriol, E. & Manago, M. (1995). *The Seamless Integration of Induction and CBR in INRECA*. Bericht, Universität Kaiserslautern
- [11] Althoff, K.-D., Wess, S., Weis, K.-H. jun., Auriol, E., Bergmann, R., Holz, H., Johnston, R., Manago, M., Meissonnier, A., Priebisch, C., Traphöner, R. & Wilke, W. (1995). *An Evaluation of the Final Integrated System*. INRECA (Esprit project 6322), Deliverable D6
- [12] Armengol, E. & Plaza, E. (1994). A Knowledge Level Model of CBR. In: [24], 53-64
- [13] Bareiss, R. & Porter, B. (1987). PROTOS - an exemplar-based learning apprentice. *Proc. 4th International Workshop on ML*, Irvine, 12-23
- [14] Cohen, P. R. (1989). Evaluation and CBR. In: K. Hammond (ed.), *Proc. of the 2nd Darpa Workshop on CBR*, Morgan Kaufmann, 168-172
- [15] Goos, K. (1995). *Fallbasiertes Klassifizieren & Methoden, Integration und Evaluation*. Dissertation, Universität Würzburg
- [16] Koton, P. (1989). *Using experience in learning and problem solving*. MIT, Laboratory of Computer Science Ph.D. Dissertation, October 1988), MIT/LCS/TR-441
- [17] Newell, A. (1982). The knowledge level. *Artificial Intelligence*, Vol. 18, pp 87-127
- [18] López, B. and Plaza, E. (1991). Case-based Learning of Strategic Knowledge. In: Y. Kodratoff (ed.), *Proc. EWSL-91*, Springer Verlag, 398-411
- [19] Richter, M. M. & Wess, S. (1991). Similarity, Uncertainty and CBR in Patdex. In: R. S. Boyer (ed.), *Automated Reasoning*, Kluwer Academic Publishers, 249-266
- [20] Steels, L. (1990). Components of Expertise. *AI Magazine*, 11(2), Summer 1990, 29-49
- [21] Veloso, M. & Aamodt, A. (eds.) (1995). *CBR Research and Development*. Springer Verlag
- [22] Wess, S. (1993). PATDEX - ein Ansatz zur wissensbasierten und inkrementellen Verbesserung von Ähnlichkeitsbewertungen in der fallbasierten Diagnostik. In: F. Puppe and A. Günter (eds.) (1993). *Expertensysteme 93*. Springer Verlag, 42-55
- [23] Wess, S. (1995). *Fallbasiertes Schließen in wissensbasierten Systemen zur Entscheidungsunterstützung und Diagnostik*. Dissertation, Universität Kaiserslautern
- [24] Wess, S., Althoff, K.-D. & Richter, M. M. (eds.) (1994). *Topics in CBR*, Springer Verlag
- [25] Wielinga, B. J., Van de Velde, W., Schreiber, G. & Akkermans, H. (1993). Towards a unification of knowledge modeling approaches. In: J.-M. David, J.-P. Krivine & R. Simmons (eds.), *Second Generation Expert Systems*. Springer Verlag, 299-335