AIS --- AIS --- AIS --- AIS

6. Kolloquium der Arbeitsgruppe Informatik-Systeme

Michael Sonnenschein, Ulrike Lichtblau
(Hrsg.)

Bericht Nr. AIS-12 - November 1993

Arbeitsgruppe Informatik-Systeme

FB Informatik - Universität Oldenburg

Herausgeber der Berichtsreihe:

Dr. Ulrike Lichtblau
Prof. Dr. Michael Sonnenschein

Anschrift der Autoren:

Fachbereich Informatik
Carl von Ossietzky Universität Oldenburg
Postfach 2503
26111 Oldenburg

Vorwort

Der vorliegende Bericht stellt die Ergebnisse dar, die beim sechsten Kolloquium der Arbeitsgruppe Informatik-Systeme am 9.7.1993 präsentiert wurden.

Die Arbeitsgruppe Informatik-Systeme wurde mit dem Ziel gegründet, verschiedene Arbeitsbereiche des Fachbereichs Informatik stärker miteinander in Verbindung zu bringen. Dies soll unter anderem dazu dienen, methodische Grundlagenarbeit zur Vorbereitung des Oldenburger Forschungs- und Entwicklungsinstituts für Informatik-Werkzeuge und -Systeme (OFFIS) zu leisten.

Aus Mitteln der Stiftung Volkswagenwerk (Az. 210-70631/9-13-14/89) stehen der Arbeitsgruppe vom 1.7.1990 bis zum 30.6.1994 Mittel zur Finanzierung von 7,5 wissenschaftlichen Mitarbeiterstellen, Hilfskräften, Sachmitteln und Investitionen zur Verfügung.

Michael Sonnenschein
Ulrike Lichtblau

erfordern die Verwendung einer Vielzahl unterschiedlicher Kontrollobjekte und schmälern damit gerade die leichte Erlernbarkeit.

- *Standard-Dialogkomponenten* :
  An "Verzweigungen" des Präsentationsflusses werden automatisch Standard-Dialog-Objekte eingeführt, die aber noch an die Anwendung angepaßt werden können. In diesen wird der weitere Ablauf der Präsentation - ggf. zusammen mit dem Anwender - ermittelt.
- *Leichte Erweiterbarkeit um neue Klassen multimedialer Daten* :
  Gerade multimediale Systeme müssen häufig an neue Darstellungsverfahren oder Ablageformate angepaßt werden. Solche Erweiterungen sind in FMAD durch die konsequente objektorientierte Implementierung garantiert.
- *Problemspezifische Interaktionsformen:*
  Über ein spezielles Modul können neu definierte, auf die Anwendung abgestimmte Interaktionsformen geladen und in das System eingebunden werden (z.B. Knopfleisten, Maus-sensitive Bereiche in Bildern, Schlüsselworteingaben u.v.m.).

FMAD ist in UNIX-Umgebungen verwendbar. Als Realisierungsplattform dient das Toolkit XFantasy [Götz92a], [Götz92b], daß in [Voßb93] um multimediale Mechanismen erweitert wurde. Der aktuelle Entwicklungsprototyp läuft auf Workstations von Sun Microsystems Comp. Corp. und integriert die Medien Text, Graphik, Rasterbild, Audio und Video in verschiedenen Formaten.

Literatur

[EiGö92] H. Eirund, R. Götze, *Modeling Interactive Multimedia Applications*, Proc. 3rd Eurographics Workshop in Object-Oriented Graphics, Preprints Univ. Geneve, 1992

[EiHo93] H. Eirund, M. Hofmann, *Designing Multimedia Presentations*, Proc. GI-Fachtagung für Hypermedia, Zürich, Reihe: Informatik aktuell, Springer Verlag, 1993

[GöEC93] R. Götze, H. Eirund, R. Claaßen, *Object-Oriented Dialog Control for Multimedia User Interfaces*, Proc. Human Computer Interaction, Wien, Sept. 1993

[EiGö93] H. Eirund, R. Götze, *Realisierung eines Entwicklungswerkzeugs für multimediale Anwendungen*, Proc. GI-Fachtagung Softwaretechnik 93, Dortmund, Nov. 1993

[Götz91a] Götze, R.: *Objektorientierte Dialogspezifikation.* – Tagungsband GI-Jahrestagung 1991, IFB 293, Springer-Verlag, pp. 519-528, 1991

[Götz91b] Götze, R.: *ODISS - eine objektorientierte Dialogspezifikationssprache*, in: Berichte der Arbeitsgruppe Informatik-Systeme AIS-3, Univ. Oldenburg, 1993

[Götz92a] Götze, R. et al : *Zwischenbericht der Projektgruppe XFantasy.* – Interner Bericht IS11, Fachbereich Informatik, Universität Oldenburg, FB Informatik, 1992

[Götz92b] Götze, R. et al: *Endbericht der Projektgruppe XFantasy.* – Interner Bericht IS13, Fachbereich Informatik, Universität Oldenburg, FB Informatik, 1992

[Voßb93] Voßberg, L.:*MediaManager – Eine Erweiterung von XFantasy um Multimedia-Klassen und Synchronisationsmechanismen*, Diplomarbeit, Univ. Oldenburg, FB Informatik, 1993

"Zielidentifikation und Planen in Petri-Help" [1]

Knut Pitschke, Olaf Schröder, Claus Möbus
Universität Oldenburg
Fachbereich Informatik
Abteilung Lehr- Lernsysteme
Postfach 2503
D 26111 Oldenburg

e-mail: <Vorname>.<Nachname>@arbi.informatik.uni-oldenburg.de

Zusammenfassung:

Um in Intelligenten Tutorsystemen dem Benutzer adäquate Hilfen zur Verfügung stellen zu können, ist ein Benutzermodell unerläßlich. Dieses soll das hypothetische Wissen des Benutzers bezogen auf das dem System über die gelehrte Domäne verfügbare Expertenwissen darstellen.
Hier wird ein Ansatz für Tutorsysteme vorgestellt, die über kein explizites Domänenwissen sondern stattdessen über implizites Wissen in Form eines Orakels verfügen. Unter Orakel verstehen wir ähnlich wie [Valiant 84] eine Prozedur oder auch einen menschlichen Experten, der Lösungen des Benutzers als korrekt oder fehlerhaft klassifiziert. Dieses Orakel ist natürlich abhängig von der gelehrten Domäne. Eine Lernkomponente aquiriert Wissen über das Verhalten möglichst vieler Benutzer. Ein neuer Benutzer wird innerhalb der so aufgebauten Wissensstrukturen identifiziert und sein Verhalten und seine Ziele vorhergesagt. Am Beispiel unseres Hilfesystems Petri-Help wollen wir verdeutlichen, welche Form von Wissen vom System gelernt werden kann und wie dies zur Verhaltensvorhersage eingesetzt werden kann und wie daraus Hilfen abgeleitet werden können.

[1] Dieser Bericht stellt eine modifizierte Version von [Pitschke 93] dar.

Zielidentifikation und Planen in Intelligenten Tutoriellen Systemen

"Intelligente Tutorielle Lernsysteme sind wissensbasierte Computerprogramme und -systeme, die den Lernenden mit Hilfen, mit Empfehlungen, mit Instruktionsmaterialien, mit Übungsaufgaben, mit Erklärungen usw. kompetent und behutsam unterstützen. Dazu müssen sie in der Lage sein, sich auf individuelle Lernverläufe einzustellen und so das Lernen sowohl interessant und abwechslungsreich als auch effektiv zu gestalten." [Möbus 93].
Speziell zur Bereitstellung von adäquaten Hilfen sind detaillierte Kenntnisse über den momentanen Wissensstand des Benutzers nötig.
Im Bereich der Planerkennung beschäftigt man sich damit, aus gegebenen Handlungssequenzen mögliche Ziele zu erschließen. Pläne werden hierbei verstanden als Aktionsfolgen, die einen gegebenen Zustand in einen Zielzustand überführen [Allen et al. 1991].
Die Übernahme der Ansätze aus der Planerkennung im Bereich der ITS ist naheliegend [Bauer et al. 1991], sie setzt jedoch voraus, daß in der gelehrten Domäne eine explizite Designtheorie existiert und daß diese dem System in Form von Expertenwissen zur Verfügung steht.
Hier soll ein Ansatz vorgestellt werden, der es gestattet, in Domänen, in denen keine explizite Designtheorie bekannt oder verfügbar ist, die Ziele des Benutzers zu identifizieren und Voraussagen über sein Verhalten zu machen. Die Idee wird exemplarisch an unserem System Petri-Help [Möbus et al. 1992] erläutert, läßt sich aber auf ähnliche Bereiche übertragen. Voraussetzung ist jedoch, daß ein Orakel zur Verfügung steht, das Entwürfe in der gelehrten Domäne nach deren Erzeugung bewerten kann; im Gegensatz zu einer Theorie, die es gestatten würde, Lösungen formal abzuleiten.

Petri-Help

Petri-Help ist ein Hilfesystem, das Benutzer bei der Einarbeitung in das Modellieren mit B/E-Petri-Netzen (Bedingungs-Ereignis-Netzen) unterstützt. Dazu werden Aufgaben präsentiert, für die der Anwender eine Lösung erarbeiten soll. Der Benutzer hat zu jeder Zeit die Möglichkeit, seinen Entwurf begutachten und bewerten zu lassen und Hilfen in Form von Ergänzungsvorschlägen anzufordern (eine ausführlichere Beschreibung des Systems findet sich in [Pitschke, Möbus 93]).
Eine Bewertung von Benutzerentwürfen setzt einerseits eine (formale) Spezifikation der zu lösenden Aufgabe, andererseits ein Orakel, das die Lösung bezüglich der Spezifikation bewertet voraus. In der Domäne der Petri-Netze ist es unüblich, Aufgaben formal zu spezifizieren, auch in Lehrbüchern (wie z.B. [Reisig 1986]) werden Netze meist als Lösungen präsentiert. In der Regel sind diese selbsterklärend, so daß sich die Fragen nach der zugehörigen Aufgabenstellung meist erübrigen. Eine Ausnahme bildet [Olderog 1991], wo Petri-Netze (neben Automaten) als formale Semantik von Prozeßtermen benutzt werden und diese formal aus einer Spezifikation in Trace-Logik abgeleitet werden können.
Wir können natürlich aus Gründen der Überprüfbarkeit der Lösung nicht auf eine formale Spezifikation verzichten. Die Aufgabenbeschreibung erfolgt in Petri-Help durch eine Menge temporallogischer Formeln, die das intendierte Verhalten des Petri-Netzes beschreiben. Dabei findet eine einfache Aussagenlogik Verwendung, die um die zeitlichen Prädikate "always", "eventually" und "nexttime" erweitert wurde. Unsere Logik erlaubt Branching-time und benutzt Step-Semantik. Diese Form der Aufgabenspezifikation gestattet es, (Teil-) Lösungen mittels Model-Checking (unserem Orakel) zu verifizieren [Damm et al. 1990, Josko 1990]. Dabei wird überprüft, für welche Formeln der Aufgabenspezifikation das vorgeschlagene Netz ein Modell darstellt, welcher Teil der Spezifikation erfüllt wurde. Dank der Beschränkung auf B/E-Netze sind die dabei auftretenden Modelle immer endlich und häufig zyklisch.
Leider ist die Folge von Teilnetzen, die während der Entwicklung einer (korrekten) Lösung auftreten, nicht-monoton bezogen auf die von den jeweiligen Netzen erfüllten

Formeln. Das heißt, eine Formel, die in einem früheren Stadium als korrekt verifiziert wurde, kann schon nach dem nächsten Editierschritt wieder falsch sein, obgleich dieser Schritt notwendig zu einer korrekten Lösung führt. Dieser Umstand macht es auch unmöglich, Teilen von korrekten Lösungen einfach die durch sie erfüllten Formeln zuzuweisen und dann diese Teile in anderem Kontext zu benutzen, ohne die Formeln neu beweisen zu müssen. Eine solche wünschenswerte aber unmögliche Anwendung wäre die automatische Ergänzung von Teilnetzen. Dieses Manko läßt sich teilweise kompensieren. Trotz der nicht-monotonen Eigenschaft der Netzentwicklung sind im Anfangszustand (dem leeren Netz) keine Formeln, im Finalzustand (der fertigen Lösung) jedoch alle Formeln der Spezifikation erfüllt. Es muß also Zwischenzustände (Netzfragmente) auf dem Weg zu einer Lösung geben, in denen zusätzlich Formeln erfüllt wurden. Wir klassifizieren diese Zustände als "sicher" (diese Sicherheitseigenschaft darf nicht verwechselt werden mit der Sicherheit in Petri-Netzen, siehe z.B. [Reisig 86]) nach folgender induktiven Definition:
Der Anfangszustand gilt als "sicher". Jeder Zustand (Netz), in dem eine echte Obermenge der im letzten "sicheren" Zustand erfüllten Formeln gilt, wird als "sicher" bezeichnet.
Aufgrund dieser Klassifikation von Benutzerlösungswegen werden vom System Designregeln gelernt, die als Basis für Ergänzungsvorschläge dienen.
Versuche mit Studenten haben gezeigt, daß die Möglichkeit, Netze auf Korrektheit überprüfen zu lassen, gerne und ausgiebig benutzt wird. Auch die Ergänzungsvorschläge des Systems, die auf Grund der gelernten Designregeln erzeugt wurden, wurden häufig in Anspruch genommen. Sinnvolle Ergänzungsvorschläge des Systems wurden jedoch erwartungsgemäß erst dann generiert, wenn Petri-Help adäquates Wissen aus früheren Problemlösesitzungen gelernt hatte. Kritisiert wurde, daß die angebotenen Ergänzungsvorschläge unzureichend auf den jeweiligen Benutzer abgestimmt waren. Oft wurde zu viel Information auf einmal präsentiert, oder aber, besonders in frühen Stadien der Problemlösung, wurde der Ratsuchende in eine Richtung gelenkt, die er nicht intendierte.
Dieses Manko soll der hier vorgestellte Ansatz beseitigen.

Planen

Planerkennung heißt vereinfacht, vorgegebene oder bekannte Minimalpläne in beobachteten Aktionsfolgen zu identifizieren. Planen bedeutet, eine Folge von Operationen zu finden, die eine gegebene Situation in einen Zielzustand überführt.
Im Falle von Petri-Help bedeutet dies, in einer Situation, in der der Benutzer um Hilfe nachfragt, ihm Informationen zur Verfügung zu stellen, die es ihm gestatten, sein momentan vorhandenes Netz zu einer korrekten Lösung bezogen auf die Aufgabenspezifikation weiterzuentwickeln. Dies geschieht zwar schon mit Hilfe der gelernten Designregeln, oftmals wird jedoch zuviel Information präsentiert.

Ansatz

Aufgrund des fehlenden Designwissens steht a priori kein Such- bzw. Planungsraum zur Verfügung. Es besteht jedoch die Möglichkeit, durch Beobachtung und Befragung der Benutzer sukzessive Information über die jeweiligen (Sub-) Ziele zu lernen, die auf dem Weg zu einer korrekten Lösung durchlaufen werden. Auf diese Weise kann ein Suchraum etabliert werden, der Ziele und Aktionen verknüpft.
Die Identifikation eines Benutzers geschieht durch Einordnung seines gezeigten Verhaltens in die Menge aller bislang beobachteten Verhalten. Das Benutzermodell besteht nicht aus einer Ansammlung von Annahmen über den Anwender, sondern aus der Relation seines Verhaltens zu den akquirierten Verhaltensweisen vormaliger Benutzer. Einen ebenfalls wahrscheinlichkeitsorientierten Ansatz beschreibt [Becker 93] für die Akquisition von Benutzerstereotypen.

Zielidentifikation

In Petri-Help existiert die Möglichkeit, Hypothesen über ein bislang erzeugtes Netz zu formulieren. Dies geschieht durch die Auswahl einer Untermenge der temporallogischen Formeln der Aufgabenspezifikation. Das Netz wird dann bezüglich dieser Formeln verifiziert, als Ergebnis werden die erfüllten und die unerfüllten Formeln zurückgegeben.
Diese Hypothese, also die gewählte Formelmenge, wird als das Ziel angesehen, das der Benutzer mit seiner Lösung erreichen wollte. Aus diesen, von unterschiedlichen Benutzern postulierten (Sub-) Zielen wird ein DAG (gerichteter azyklischer Graph) aufgebaut, dessen einziges Blatt dem Ziel entspricht, die spezifizierte Aufgabe komplett zu lösen. Die verschiedenen Wege durch den DAG entsprechen den verschiedenen Folgen von Subzielen.
Um die spezifizierten (Sub-) Ziele zu erreichen, muß der Benutzer Petri-Netze erzeugen, beschriften, ergänzen, Teile löschen etc. . Die dabei entstehende Folge von Netzen, die durchlaufen wird um bestimmte Ziele zu erreichen, ergibt wiederum einen gerichteten Graphen (den Aktionsgraphen), der beobachtete Benutzerverhaltensweisen widerspiegelt. Werden neue Netze erkannt, wird der Graph entsprechend erweitert. Zeigt ein Benutzer dagegen ein Verhalten, das schon früher auftrat, wird dies im Graphen identifiziert und entsprechend vermerkt. Es ist dann bei jeder Entscheidung, d.h. bei jeder Verzweigung in diesem Aktionsgraphen bekannt, wie oft sie gewählt wurde. Diese Statistik über beobachtetes Benutzerverhalten wird Grundlage der Verhaltensvorhersage sein.

Verhaltensvorhersage

Eine Vorhersage des Verhaltens eines Benutzers ist nur möglich, wenn das System schon einen Zielgraphen und einen Aktionsgraphen (siehe: Zielidentifikation) aus früheren Sitzungen mit anderen Benutzern aufbauen konnte. Ferner wird die Vorhersage genauer, je weiter der Benutzer in der Bearbeitung der Aufgabe fortgeschritten ist, d.h. je mehr Information über sein bisheriges Verhalten schon bekannt ist.
Ist das nächste Subziel des Anwenders unbekannt, muß zunächst eine Vorhersage auf der Zielebene erfolgen. Dazu dient die Identifikation der bislang erkannten Subziele im Zielgraphen. Die relativen Häufigkeiten der getroffenen Entscheidungen bei möglichen Verzweigungen ist bekannt. Diese Folge von Häufigkeiten kann nun nach verschiedenen Methoden fortgesetzt werden und liefert eine Wahrscheinlichkeitsaussage über die Entscheidung über das nächste angestrebte Ziel. Ist das Ziel bekannt, kann ein ähnlicher Prozeß auf der Aktionsebene stattfinden. Dazu wird nur der Teil des Aktionsgraphen betrachtet, der zum bisherigen Weg des Benutzers durch den Zielgraphen korrespondiert, der Rest des Aktionsgraphen wird ausgeblendet. Die wahrscheinlichste nächste Benutzeraktion wird nach dem gleichen Verfahren ermittelt, wie die Zielvorhersage im Zielgraphen. Nach welchen Kriterien die induktive Fortsetzung der Folge von bekannten Entscheidungen erfolgen soll, wird zur Zeit im Rahmen einer Studienarbeit untersucht.

Anwendung der Verhaltensvorhersage

Ein Manko in Petri-Help bestand darin, daß die angebotene Hilfsinformation zwar korrekt, aber unzureichend auf den jeweiligen Benutzer abgestimmt war. Mit den beiden oben geschilderten Methoden ist es möglich, Hilfen anzubieten, die zum wahrscheinlichsten nächsten Ziel hinführen, bzw. die prognostizierten künftigen Aktionen motivieren. Auch die "Körnigkeit" der Information kann besser auf den Anwender abgestimmt werden, da Information über die bisherigen Schrittweiten beim Problemlöseprozeß vorliegen.

Literatur:

Allen, J., Kautz, H., Pelavin, R., Tenenberg, J.,
Reasoning about Plans, Morgan Kaufmann Publishers, San Mateo, California, 1991

Bauer, M., Biundo, S., Dengler, D., Hecking, M., Koehler, J., Merziger, G.,
Integrated Plan Generation and Recognition - A Logic Based Approach DFKI Research Report RR-91-26, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Kaiserslautern/ Saarbrücken,1991

Becker, A. Automatisch akquirierte Wissensebenen als Basis für Benutzermodellierung, in: Alfred Kobsa und Wolfgang Pohl (Hrsg.): Arbeitspapiere des Workshops 'Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen' auf der KI93, Berlin,13.-15.9.93. WIS-Memo 7, September 1993, AG Wissensbasierte Informationssysteme, Informationswissenschaft, Universität Konstanz.

Damm, W., Döhmen, G., Gerstner, V., Josko, B.,
Modular Verification of Petri Nets. The Temporal Logic Approach. In: de Bakker, de Roever, Rozenberg (eds), Proceedings REX-Workshop on Stepwise Refinement of Distributed Systems: Models, Formalisms, Correctness. Springer: LNCS 430, 1990

Josko, B., Verifying the Correctness of AADL Modules using Model Checking, in: de Bakker, de Roever, Rozenberg (eds), Proceedings REX-Workshop on Stepwise Refinement of Distributed Systems: Models, Formalisms, Correctness, Springer LNCS 430, 1990

Möbus, C. , Workshop der Fachgruppe "Intelligente Tutorielle Systeme" der GI, 8./9. Juni 1993, Oldenburg , Interner Bericht des Fachbereichs Informatik, Universität Oldenburg 1993

Möbus, C., Pitschke, K., Schröder, O.,
Towards the Theory-Guided Design of Help Systems for Programming and Modelling Tasks, in C. Frasson, G. Gauthier, G.I. McCalla (eds), Intelligent Tutoring Systems, Proceedings of the Second International Conference ITS 92, Montreal, Berlin: Springer LNCS 608, 1992

Olderog, E.R., Nets, Terms, and Formulas, New York: Cambridge (Cambridge Tracts in Theoretical Computer Science 23), 1991

Pitschke, K. Zielidentifikation und Planen in Intelligenten Tutoriellen Systemen; In: Alfred Kobsa und Wolfgang Pohl (Hrsg.): Arbeitspapiere des Workshops 'Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen' auf der KI93, Berlin,13.-15.9.93. WIS-Memo 7, September 1993, AG Wissensbasierte Informationssysteme, Informationswissenschaft, Universität Konstanz.

Pitschke, K., Möbus, C.,
Improvement of Help Information by Accumulative Learning, in: [Möbus 93]

Reisig, W., Petrinetze- eine Einführung, Springer Berlin Heidelberg, 1986

Valiant, L. A Theory of the Learnable. Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing 1984, 436-445