BERICHT
aus dem
PSYCHOLOGISCHEN INSTITUT
DER UNIVERSITÄT HEIDELBERG

Johannes Unnewehr

Benutzerhandbuch Prozeduren zur Wissensdiagnose

Dezember 1992

Bericht Nr. 74

6900 HEIDELBERG 1, HAUPTSTR. 47-51, TEL. 06221/54 73 46

Die Implementierung und Dokumentation der in diesem Handbuch beschriebenen Prozeduren zur Wissensdiagnose wurde von der Deutschen Forschungsgemeinschaft unter Aktenzeichen Lu 385/2–1 gefördert.

Die Prozeduren können bei Angabe des Diskettenformats über folgende Adresse bezogen werden: Johannes Unnewehr, Psychologisches Institut der Universität Heidelberg, Hauptstraße 47–51, 6900 Heidelberg.

Inhalt

1 Einführung

Ein Lehrer prüft einen Schüler, indem er ihm Fragen stellt. Obwohl der Schüler gelegentlich die richtige Antwort rät oder Flüchtigkeitsfehler macht, hat sich der Lehrer nach einer Anzahl von Fragen ein Bild vom Wissen des Schülers gemacht.

Als "step toward practical implementation of computerized knowledge assessment procedures in an educational context" beschreiben Falmagne & Doignon (1988a) eine Klasse von Markov-Prozeduren für die Diagnose von Wissen. Der Ansatz basiert auf der Verwendung von Wissensstrukturen und berücksichtigt die Möglichkeit von Glückstreffern und Flüchtigkeitsfehlern.

Wissensstrukturen wurden von Doignon & Falmagne (1985) als Verallgemeinerung einer Guttman-Skala eingeführt. Ein Wissensgebiet ist dabei als eine Menge von Fragen konzeptualisiert, die von einer Population von Personen gelöst werden. Doignon & Falmagne definieren auf diesem Wissensgebiet wie folgt eine Wissensstruktur:

> **Definition 1:** Es sei Q eine endliche, nicht leere Menge von Fragen und κ eine Menge von Teilmengen von Q. Das Paar (Q, κ) heißt *Wissensstruktur*. Jedes Element K in κ repräsentiert eine Menge von Fragen, die eine Person lösen kann und heißt *Wissenszustand*. Wir schreiben ψ für eine Teilmenge aus κ.

Der Vorteil der Wissensstrukturen liegt darin, daß sie eine ökonomische Wissensdiagnose ermöglichen. Wie Falmagne & Doignon (1988a) erläutern, kommen aufgrund der kognitiven Struktur des Wissens manche Elemente der Potenzmenge von Q in κ nicht vor. Mit anderen Worten: Manche Wissenszustände können bei keiner Person gefunden werden und müssen deshalb bei der Suche nach dem Zustand einer Person a priori nicht berücksichtigt werden. Dies ist eine Reduktion des Diagnoseproblems und kann dazu führen, daß nicht alle Fragen gestellt werden müssen, um zu einem Diagnoseergebnis zu gelangen.

Für die Erstellung und Verifikation von Wissensstrukturen wurden unterschiedliche Techniken etabliert (siehe Albert, Schrepp & Held, 1992; Koppen & Doignon, 1990; van Leeuwe, 1974; sowie Dowling, im Druck-a). Die Existenz einer validierten Wissensstruktur wird für die Arbeit mit den im Folgenden beschriebenen Prozeduren vorausgesetzt.

Eine **Diagnoseprozedur**, die mit einer Wissensstruktur arbeitet, kann man wie in Abbildung 1 dargestellt realisieren.

Ziel der Prozedur ist es, Wissenszustände als mögliche Ergebnisse zu markieren. Die Menge der markierten Zustände wird im Weiteren auch kurz als

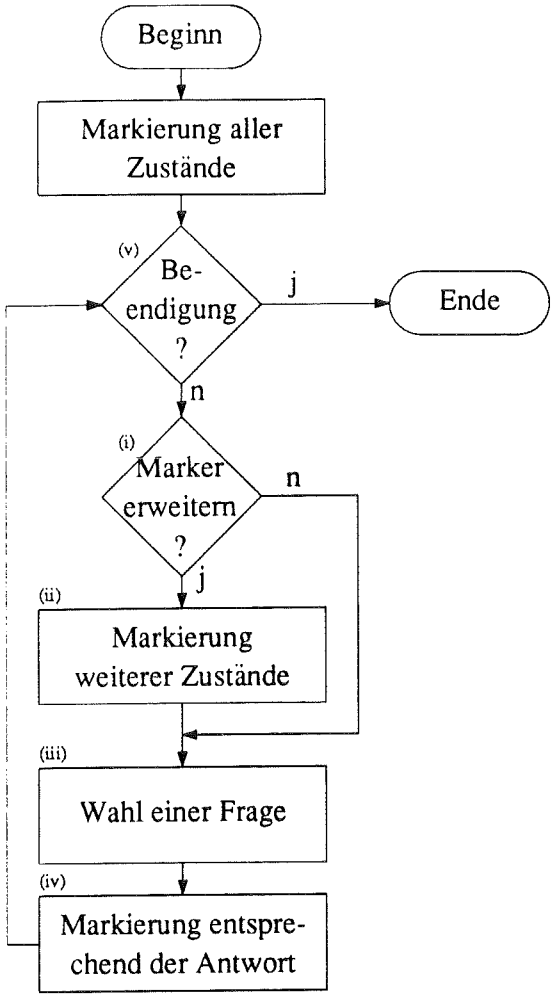


Abbildung 1: Allgemeine Diagnoseprozedur

Marker bezeichnet. Zunächst werden alle Wissenszustände einer Wissenstruktur markiert.

Sobald ein errechnetes Diagnoseergebnis ein Beendigungskriterium (v), etwa ein Gütekriterium, erfüllt, wird es ausgegeben und der Prozeß beendet. Ansonsten wird der Marker unter bestimmten Bedingungen (i) erweitert, um beispielsweise das bisher ermittelte Ergebnis zu Überprüfen und zu korrigieren (ii). Nun wird eine Frage ausgewählt und gestellt (iii). Entsprechend der Antwort wird die Menge der markierten Wissenszustände geändert (iv). Schließlich wird geprüft, ob der so entstandene neue Marker dem Beendigungskriterium genügt.

Wie Abb. 1 grob skizziert, soll die "Qualität" des Diagnoseergebnisses in einer Rückkopplungsschleife gesteigert werden, wobei unter anderem noch festgelegt werden muß, was unter "Qualität" verstanden werden soll, welche weiteren Wissenszustände zu deren Steigerung markiert werden, wie eine Frage ausgewählt wird, wann der Prozeß enden soll und was als Diagnoseergebnis anzusehen ist, ein einzelner Wissenszustand etwa oder eine (Wahrscheinlich-

keits-)Verteilung auf der Menge der Wissenszustände. Hier gibt es verschiedene Möglichkeiten der genaueren Spezifikation. Zwei davon wurden realisiert; sie werden in Kapitel 1.1 und 1.2 beschrieben.

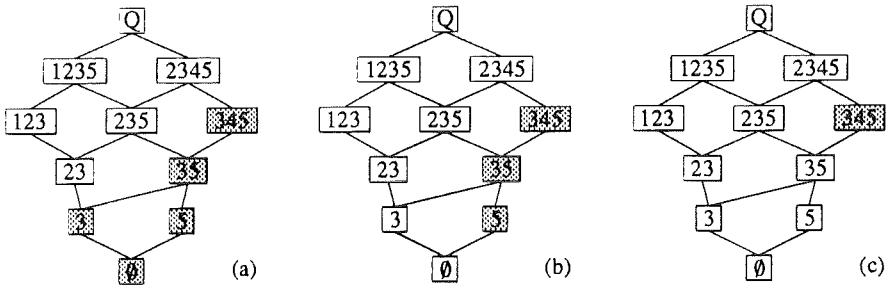


Abbildung 2: Veränderung des Markers während der ersten Schritte einer Diagnose

Ein einfaches Beispiel soll den möglichen Ablauf zu Beginn einer Diagnose verdeutlichen. Entsprechend Definition 1 sei Q = {1, 2, 3, 4, 5} eine Menge von Fragen und κ = {∅, 3, 5, 23, 35, 123, 235, 345, 1235, 2345, Q} eine Menge von Wissenszuständen (die Mengenklammern und Kommata wurden bei den Wissenszuständen zur Vereinfachung weggelassen). Die Wissensstruktur kann wie in Abb. 2 (a – c) als Hassediagramm dargestellt werden.

Zu Beginn der Diagnose werden alle Wissenszustände markiert. Abb. 2 (a) zeigt den Marker (grau), nachdem eine Person Frage 2 nicht lösen konnte. Etwa die Hälfte der Wissenszustände, nämlich alle, die Frage 2 enthalten, wurde aus dem Marker ausgeschlossen. Als nächstes konnte die Person Frage 5 lösen. Wieder wurde der Marker entsprechend verkleinert (Abb. 2 (b)). Nachdem auch Frage 4 richtig beantwortet wurde, blieb nur noch ein Wissenszustand markiert. Die Fragen 1 und 3 mußten nicht gestellt werden, um zu diesem ersten Ergebnis zu kommen. Dieser Vorteil ergibt sich aus der verwendeten Wissensstruktur.

Da es vorkommen kann, daß eine Person richtige Antworten rät oder Flüchtigkeitsfehler macht, stimmt das ermittelte Diagnoseergebnis möglicherweise nicht vollständig mit dem Wissenszustand der Person überein, sondern weicht von diesem in einer Anzahl Fragen ab. Diese Anzahl wird definiert als die Distanz zweier Zustände:

Definition 2: Die Anzahl der Elemente der symmetrischen Mengendifferenz

$$d(K,\ K') = \left|K \triangle K'\right| \tag{1}$$

heißt *Distanz* zweier Wissenszustände.

Die Distanz zwischen dem Zustand einer Person und dem diagnostizierten Zustand kann auch als Diagnosefehler angesehen werden. Um sie zu verringern, können im Verlauf der Diagnose Fragen auch mehrmals gestellt werden.

1.1 Die unitäre Diagnoseprozedur von Falmagne & Doignon (1988a)

Falmagne & Doignon (1988a) definieren zur Beschreibung ihrer Diagnoseprozedur vier Zufallsvariable:

$\mathrm{Q} \in Q$: die gestellte Frage,
$\mathrm{K} \in \kappa$: der Wissenszustand der Person,
$\mathrm{R} \in \{0, 1\}$: die Richtigkeit der Antwort und
$\mathrm{M} \subset \kappa$: die Menge markierter Wissenszustände, der "Marker".

Im Verlauf einer Diagnose entsteht eine Sequenz von Vektoren

$$(\mathrm{Q}_1, \mathrm{K}_1, \mathrm{R}_1, \mathrm{M}_1), \ldots, (\mathrm{Q}_n, \mathrm{K}_n, \mathrm{R}_n, \mathrm{M}_n) \tag{2}$$

Für den Übergang[1] von Schritt n zu Schritt n+1 werden verschiedene Regelungen getroffen, die die Komponenten der vorgestellten allgemeinen Diagnoseprozedur (siehe Abbildung 1) spezifizieren:

(i) Der Marker wird vergrößert, wenn er nur ein Element enthält.

(ii) Zusätzlich zu den markierten Zuständen werden die Wissenszustände der Nachbarschaft berücksichtigt, die wie folgt definiert ist:

Definition 3: Die Menge der Wissenszustände, die durch

$$N(\psi, \varepsilon) = \left\{ K' \in \kappa \mid d(K, K') \leq \varepsilon,\ K \in \psi \right\} \tag{3}$$

definiert ist, heißt ε-*Nachbarschaft* einer Menge von Wissenszuständen ψ.

Diese Nachbarschaft enthält alle Zustände, die sich um höchstens ε Fragen von einem beliebigen markierten Zustand unterscheiden. Im Falle der hier beschriebenen unitären Prozedur wird nur die direkte (1-)Nachbarschaft berücksichtigt.

(iii) Eine Frage wird so ausgewählt, daß der Marker möglichst halbiert werden kann. Falls mehrere Fragen diesem Kriterium genügen, wird eine von ihnen zufällig gezogen. Falmagne & Doignon (1988a) nennen diese Frageregel *half-split*. Formal geht es darum, eine Frage q zu finden, die den Wert

$$|2 * |\mathrm{M}_{n,\,q}| - |\mathrm{M}_n|| \tag{4}$$

[1] Dieser Übergang entspricht einem Schleifendurchlauf in Abbildung 1.

minimiert, wobei unter $M_{n,q}$ die markierten Wissenszustände im Schritt n, die die Frage q enthalten, verstanden werden. Die Auswahl der Fragen in dem Beispiel in Kap. 1 (Abb. 2) entspricht genau dieser Vorgehensweise.

(iv) Alle mit der gestellten Frage und der beobachteten Antwort nicht übereinstimmenden Wissenszustände werden aus der Menge der markierten Zustände gestrichen. Falmagne & Doignon (1988a) nennen diese Markierungsregel *selektiv*.

(v) Ein Beendigungskriterium wurde von Falmagne & Doignon (1988a) nicht spezifiziert.

Grundgedanke der Prozedur ist, daß der Marker durch Glückstreffer und Flüchtigkeitsfehler geringfügig fehlgeleitet worden sein kann, und das Diagnoseergebnis deshalb auch in der Nachbarschaft des Markers gesucht werden muß. Weiterhin könnte eine Person während der Diagnose eine Aufgabe lernen, wodurch ihr Wissenszustand ebenfalls in der Nachbarschaft gesucht werden müßte.

Die unitäre Prozedur berücksichtigt in Punkt (iv) jeweils nur den vorhergehenden Marker und die aktuelle Antwort einer Person, um den Lern- bzw. Änderungsschritten gerecht zu werden.

Durch die markerhalbierende Frageregel werden zunächst Fragen mittleren Schweregrades gestellt. Abhängig von den Antworten einer Person wird zu Fragen leichteren oder schwereren Niveaus übergegangen. Dies entspricht der üblichen Vorgehensweise beim adaptiven Testen (siehe z. B. Weiss, 1983).

Für die Anwendbarkeit der Prozedur besteht eine wichtige Einschränkung: Die verwendete Wissensstruktur muß wohlgraduiert sein.

Definition 4: Eine Wissensstruktur ist *wohlgraduiert*, wenn es für zwei beliebige Zustände K und $K' \in \kappa$ eine Kette von Zuständen gibt, so daß gilt:

$$K = K_1 \subset K_2 \subset \ldots \subset K_n = K'; \quad |K_{i+1}| = |K_i| + 1 \qquad (5)$$

Diese Einschränkung stellt sicher, daß ein falsches Diagnoseergebnis jederzeit durch Berücksichtigung der Nachbarschaft verbessert werden kann.

1.2 Die likelihoodbasierte Diagnoseprozedur

Die likelihoodbasierte Diagnoseprozedur berücksichtigt im Gegensatz zur unitären Prozedur bei der Markierung von Zuständen auch sämtliche vorher gegebenen Antworten einer Person. Im Falle stabiler Wissenszustände soll dadurch ein besseres Diagnoseergebnis erreicht werden.

In Kap. 1.1 wurde der Ablauf einer Diagnose als Sequenz von Vektoren $(Q_1, K_1, R_1, M_1), \ldots, (Q_n, K_n, R_n, M_n)$ beschrieben. Hier wird nun ange-

nommen, daß der Wissenszustand einer Person während der Diagnose konstant ist. Aus der Sequenz $(Q_1, R_1), \ldots, (Q_n, R_n)$ der Fragen und Antworten kann für jeden Wissenszustand $K \in \kappa$ ein Vektor k berechnet werden mit

$$k_i = \begin{cases} 0 & | \; Q_i \notin K \\ 1 & | \; Q_i \in K \end{cases} \tag{6}$$

Nach Doignon und Falmagne (1985a) unterläuft einer Person bei der Bearbeitung einer Aufgabe Q_i mit einer bestimmten Wahrscheinlichkeit β_{Q_i} ein Flüchtigkeitsfehler oder sie rät mit der Wahrscheinlichkeit γ_{Q_i} die Lösung:

$$\begin{aligned} \beta_{Q_i} &= P(R_i = 0 \mid k_i = 1) \\ \gamma_{Q_i} &= P(R_i = 1 \mid k_i = 0) \end{aligned} \tag{7}$$

Die Flüchtigkeitsfehler- bzw. Ratewahrscheinlichkeit ist abhängig von der Aufgabe.

Gegeben Q, K, β und γ, ist die Likelihood eines Datenvektors R wie folgt definiert:

$$L(R|\, Q, K, \beta, \gamma) = \prod_{i=1}^{n} (1 - \beta_{Q_i})^{R_i k_i} \beta_{Q_i}^{(1-R_i)k_i} (1 - \gamma_{Q_i})^{(1-R_i)(1-k_i)} \gamma_{Q_i}^{(R_i)(1-k_i)} \tag{8}$$

Die Wahrscheinlichkeiten β und γ sind in der Regel unbekannt. Wie Lukas (persönliche Mitteilung) zeigt, hat die Likelihoodfunktion unter den Voraussetzungen

(1) $\beta_{Q_1} = \beta_{Q_2} = \ldots = \beta_{Q_n} = \gamma_{Q_1} = \ldots = \gamma_{Q_n}$

(2) $0 < \beta_{Q_i}, \; \gamma_{Q_i} < 0.5$

das gleiche Maximum wie die Funktion

$$L'(R|\, Q, K) = \sum_{i=1}^{n} R_i k_i + (1 - R_i)(1 - k_i) \tag{9}$$

zu deren Berechnung die Wahrscheinlichkeiten β und γ nicht bekannt sein müssen. Diese Funktion summiert für einen Wissenszustand K auf, wie häufig in Verlauf der Diagnose eine Person eine korrekte Antwort (R_i=1) gab, wenn die gestellte Frage Q_i Element des Wissenszustands war (k_i=1), und wie häufig sie eine falsche Antwort gab, wenn die Frage nicht Element des Wissenszustands war. Kurz gesagt wird unter Vernachlässigung von β und γ ausgezählt, wie häufig eine Antwort einem Wissenszustand entsprach.

Für die Diagnoseprozedur schlägt sich die Änderung in den Punkten (ii) und (iv) nieder (vgl. Abb. 1, sowie Punkt (ii) und (vi) in Kap. 1.1):

(i) Der Marker wird vergrößert, wenn er nur ein Element enthält.

(ii) Bei einer Vergrößerung wird der Marker um die Wissenszustände mit der zweitgrößten Likelihood L' erweitert. Das sind alle Zustände K, für die gilt:

$$L'(\mathrm{R} \mid \mathrm{Q}, K) = \max \left\{ L'(\mathrm{R} \mid \mathrm{Q}, K') \mid K' \in \kappa \setminus \mathrm{M}_n \right\} \qquad (10)$$

(iii) Die Frageregel ist *half-split*.

(iv) Die Wissenszustände mit der maximalen Likelihood L' werden markiert.

(v) Ein Beendigungskriterium wurde auch für diese Prozedur noch nicht formuliert. Der Benutzer muß die Diagnose deshalb "von Hand" beenden.

Grundgedanke der Prozedur ist, die Likelihood möglicher Ergebnisse zu maximieren, indem sie durch Fragen von konkurrierenden Zuständen mit ebenfalls hoher Likelihood differenziert werden.

Die likelihoodbasierte Prozedur berücksichtigt in Punkt (iv) jeweils die aktuelle und alle vorhergehenden Antworten einer Person, aber keinen der vorhergehenden Marker.

Wie bei der unitären Prozedur wird mit Fragen mittleren Schweregrades begonnen. Abhängig von den Antworten einer Person wird mit Fragen leichteren oder schwereren Niveaus fortgefahren. Ein gewollter Effekt ist dabei, daß mit höherer Priorität Fragen ausgewählt werden, die noch "unentschieden" sind, d.h. die zuvor genau so häufig gelöst wie nicht gelöst wurden. Dies ist bei der unitären Prozedur nicht der Fall.

Bezüglich der Anwendbarkeit ergeben sich keine Einschränkungen. Die verwendete Wissensstruktur muß nicht wohlgraduiert sein. Die likelihoodbasierte Prozedur benutzt keine zusätzliche Information, sie schöpft die gleiche Information anders aus, die auch die unitäre Prozedur benutzt.

1.3 Aufgabenstellung

Die implementierten Diagnoseprozeduren arbeiten in der in Kap. 1.1 und 1.2 beschriebenen Weise. Sie unterscheiden sich nicht bezüglich ihrer Schnittstellen.

Die Fragen und Antworten (Q und R) einer Diagnose werden in einer Datei protokolliert. Mit Hilfe dieses Protokolls kann eine abgebrochene Diagnose wieder aufgenommen werden.

Mit den Prozeduren kann das Antworverhalten einer Person und der gesamte Diagnoseprozeß simuliert werden (siehe Kap. 1.5 und Kap. 3.3.3).

Die Prozeduren können mit einem Diagnosemonitor gehandhabt werden, der Prozeßinformationen anzeigt und die manuelle Beendigung der Diagnosen ermöglicht.

1.4 Abgrenzung gegenüber anderen Aufgaben

Wissensstrukturen werden von den Prozeduren nicht erstellt. Dazu ist gesonderte Software notwendig (siehe z.B. Koppen & Doignon (1990), van Leeuwe (1974), Dowling (im Druck-b), sowie Lukas (1990) und Anh. B).

Die zu lösenden Aufgaben werden nicht vom Diagnoseprozeß präsentiert, sondern von einem Präsentationsprozeß (siehe Kap. 1.5, Anh. B, sowie Held (1991)).

Das Diagnoseprotokoll kann für statistische Auswertungen verwendet werden, der Diagnoseprozeß nimmt diese aber nicht selbst vor.

1.5 Einbettung in das System

Die Diagnose wurde als Prozeß implementiert, der mit einem weiteren Prozeß kommuniziert, der die zu lösenden Aufgaben präsentiert. Für die Erstellung eines Präsentationsprozesses hat sich das Hypertext-System KMS ("Knowledge Management System") bewährt (siehe Held 1991). Ein Beispielpräsentationsprozeß wird in Anh. B beschrieben.

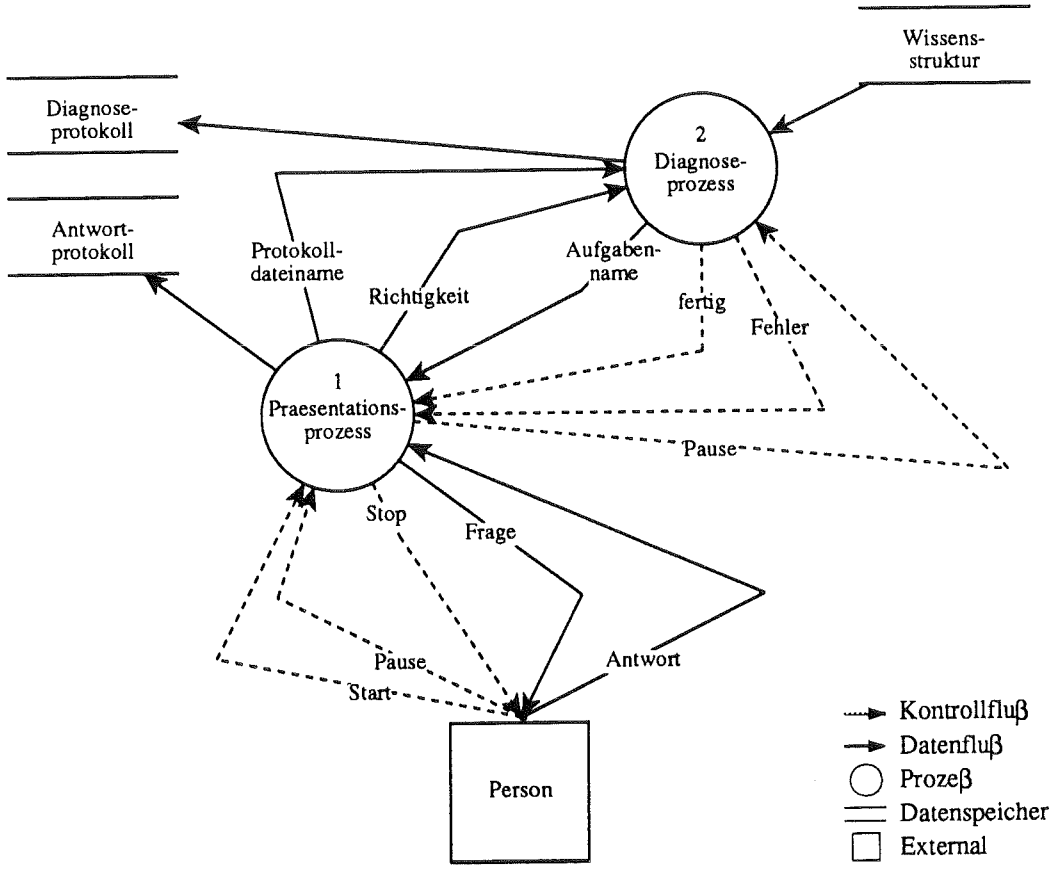


Abbildung 3: Einbettung in das System

Der Präsentationsprozeß ist zwischen Diagnoseprozeß und geprüfter Person geschaltet. Der Diagnoseprozeß steuert die Reihenfolge der Aufgaben entsprechend den in den Kapiteln 1.1 und 1.2 beschriebenen Algorithmen. Der Präsentationsprozeß bietet die Aufgaben dar und stellt fest, ob sie gelöst wurden. Präsentations- und Diagnoseprozeß schreiben jeweils getrennte Protokolldateien.

1.6 Arbeitsweise des Diagnoseprozesses

Entsprechend dem Ablauf der Diagnosen nimmt der Diagnoseprozeß verschiedene Zustände ein.

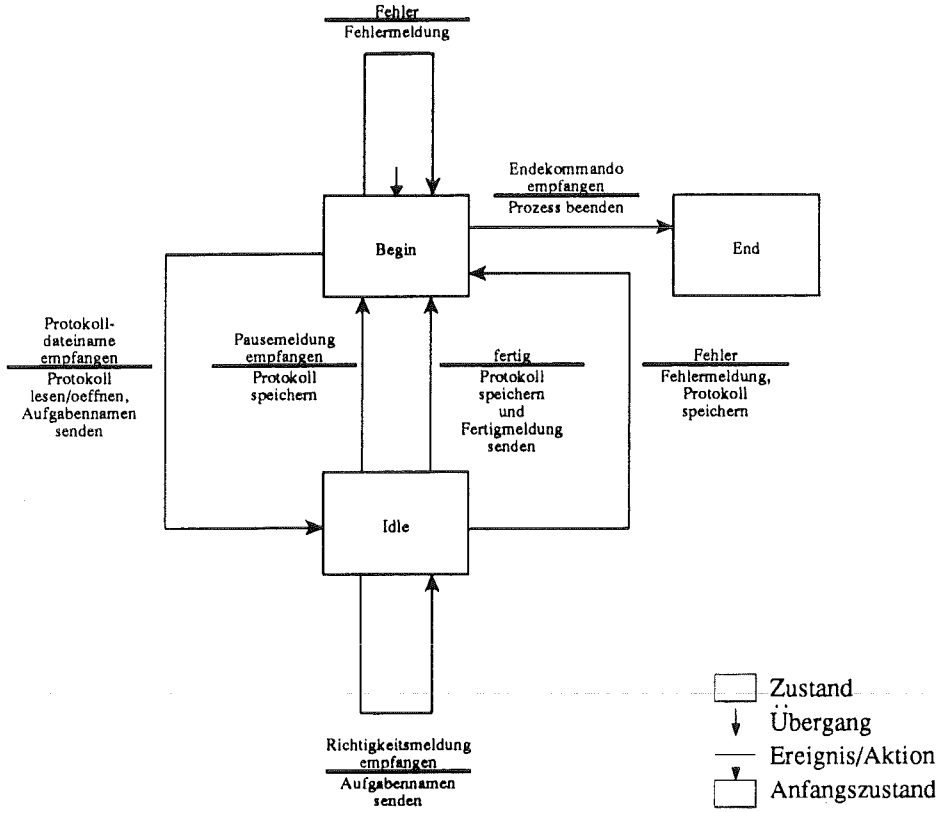


Abbildung 4: Zustandsübergangsdiagramm

Nach dem Starten des Programms (Kap. 3.1, Benutzerschnittstelle) befindet sich der Prozeß im Zustand "Begin". Vom Präsentationsprozeß wird nun der Name der Protokolldatei erwartet (Kap. 3.2, Präsentationschnittstelle). Aufgrund verschiedener Namenskonventionen (Kap. 3.3, Dateischnittstelle) ist der Prozeß damit in der Lage, die Wissensstruktur und das Diagnoseprotokoll zu laden, den ersten Aufgabennamen an den Präsentationsprozeß zu senden und in den Zustand "Idle" überzugehen.

Im Zustand "Idle" erwartet der Diagnoseprozeß vom Präsentationsprozeß Information, ob die Aufgabe gelöst wurde, und wählt danach die nächste Aufgabe aus, indem er entsprechend Abbildung 1 einen Schleifendurchlauf vollführt. Wenn die Diagnose fertig ist, meldet der Diagnoseprozeß dies dem Präsentationsprozeß und geht wieder in den Zustand "Begin" über. Die Diagnose kann auch vom Präsentationsprozeß unterbrochen werden, indem statt der Richtigkeit der Lösung ein Pausezeichen gesandt wird.

Statt des Namens der Protokolldatei kann der Präsentationsprozeß ein Endezeichen senden, woraufhin sich der Diagnoseprozeß beendet.

Im Falle eines Fehlers (Kap. 4, Fehlerbehandlung) wechselt der Diagnoseprozeß immer in den Zustand "Begin". Von dort kann die Diagnose wieder aufgenommen werden, wenn der Fehler behoben wurde (siehe Anh. A, Checkliste für Fehler).

1.7 Aussagen über Entwicklungsstufen

1.7.1 Vorversionen Zur Erprobung und Durchführung erster Versuche wurde eine Vorversion (< Version 1.0) erstellt. Diese Vorversion ist mit den nachfolgenden Versionen nicht kompatibel und wird nicht mehr unterstützt. Mit Version 1.2 wurden folgende Änderungen eingeführt:

- Einführung komprimierter Wissensstrukturen (Kap. 3.3.1).
- Kontrolle des belegten Hauptspeichers (Fehler 13 in Kap. 4).

Mit Version 1.3 wurden folgende Änderungen eingeführt:

- Kürzung des Diagnoseprotokolls (Kap. 3.3.2)
- Einführung von CR-Listen (Kap. 3.3.1)
- Simulationsoption (Kap. 3.3.3)
- Diagnosemonitor (Kap. 3.1.3)

1.7.2 Aktuelle Version Alle Beschreibungen dieser Dokumentation beziehen sich auf Version 2.0. Version 2.0 ist mit Versionen ≥ 1.0 kompatibel. Gegenüber Version 1.3 wurden folgende Änderungen vorgenommen:

- Option zur Angabe des Internetports (Kap. 5.4)
- Änderung des Nachrichtenkatalogs (Kap. 3.2.1)

1.8 Dokumentationsübersicht

Für die Bedienung des Diagnoseprozesses und für die Versorgung der Schnittstellen sind allgemeine UNIX-Kenntnisse zusammen mit diesem Benutzerhandbuch ausreichend. Damit Diagnosen durchgeführt werden können, werden aber noch eine Wissensstruktur und ein Prozeß zur Aufgabenpräsentation benötigt.

Die Erstellung von Wissensstrukturen wird in Albert, Schrepp & Held (1992), Koppen & Doignon (1990), van Leeuwe (1974) und Dowling (im Druck-a) beschrieben. Spezielle Software zur Erstellung einer Wissensstruktur aus der Basis ist durch Dowling (im Druck-b), sowie Lukas (1990) und Anh. B dokumentiert.

Ein mit dem Hypertext-System KMS erstellter Präsentationsprozeß wird von Held (1991) dokumentiert. Ein Beispiel-Präsentationsprozeß, der dem Benutzer als Vorlage für eigene Entwicklungen dienen kann, wird in Anh. B beschrieben.

Zur Änderung der Diagnoseprozeduren empfiehlt sich die Softwaredokumentation (Unnewehr, 1991).

Eine Komplexitätsanalyse (Unnewehr, 1990a) und Simulationsstudien (Unnewehr, 1990b) stellen verschiedene Leistungsmerkmale der Software vor.

Einen vollständigen Einblick in die mathematisch-psychologische Theorie, auf der die Diagnoseprozeduren basieren, geben Doignon & Falmagne (1985), Falmagne & Doignon (1988a) sowie Falmagne & Doignon (1988b).

2 Installation

Sie benötigen einen Sun-3-Computer mit Betriebssystem SunOS 4.0. Für den Diagnosemonitor und den Beispiel-Präsentationsprozeß benötigen Sie zusätzlich Sunview. Kopieren Sie den Inhalt der Diskette oder des Magnetbandes in das Verzeichnis, in dem Sie die Wissensdiagnose installieren wollen. Installieren Sie die Diagnose mit dem Kommando

```
setup
```

Die Diagnosedateien und -Verzeichnisse werden nun entkomprimiert und dearchiviert. Nach der Installation sollten Sie folgende Verzeichnisse und Dateien vorfinden:

1. Dokumentation

```
/ass/docs/
 handbook.ps   - dieses Benutzerhandbuch im Postscript-Format
 dist.l        - manual page fuer Hilfsprogramm dist
 int2bool.l    - manual page fuer Hilfsprogramm int2bool
 presproc.l    - manual page fuer Hilfsprogramm presproc
 r2set.l       - manual page fuer Hilfsprogramm r2set
 tel.l         - manual page fuer Hilfsprogramm tel
```

2. Prozeduren

```
/ass/procs/
 Makefile      - Makefile zum Erstellen des Codes
 asslike       - likelihoodbasierte Diagnoseprozedur
 assmon        - Diagnosemonitor
 assmon.pic    - Logo fuer den Diagnosemonitor
 assmon.par    - Parameterdatei fuer den Diagnosemonitor
 assunit       - unitaere Diagnoseprozedur
```

3. Sourcecode

```
/ass/procs/sources/assprocs/
 ass.h         - Typen und Header
 begin.c       - Prozesszustand BEGIN
 end.c         - Prozesszustand END
 error.c       - Fehlerbehandlung
 file.c        - Dateischnittstelle
 idle.c        - Prozesszustand IDLE
 internet.c    - Internetschnittstelle
 jobib.c       - allgemeine Hilfsprogramme
 like.c        - likelihoodbasierte Markierungsregel
 likepar.c     - Parameter fuer die likelihoodbasierte Prozedur
```

```
matrix.c      - Hilfsprogramme fuer Matritzenoperationen
neighbor.c    - Nachbarschaft
process.c     - Prozessablauf
rest.c        - Stellen der restlichen Fragen
setops.c      - Mengenoperationen
simul.c       - Simulationsoption
stop.c        - Beendigungskriterium (noch nicht implementiert)
subject.c     - Simulation des Antwortverhaltens
unitary.c     - unitaere Markierungsregel
unitpar.c     - Parameter fuer die unitaere Prozedur
visualiz.c    - (interne) Monitorschnittstelle

/ass/procs/sources/assmon/
assmon.c      - Quellcode des Diagnosemonitors
canvas.c      - "canvas"-Prozeduren fuer den Diagnosemonitor
close.icn     - Closeicon fuer den Diagnosemonitor
logo.pic      - Logo fuer den Diagnosemonitor
windef.c      - Fensterdefinitionen fuer den Diagnosemonitor
```

4. Wissensstrukturen

```
/ass/structs/Chess/
Chess.pic     - Graphik der Schachstruktur von Albert, Schrepp
                & Held (1992)
Chess.set     - boolsche Mengendarstellung der Schachstruktur
Chess.sur     - Surmisedarstellung der Schachstruktur
Chess.dia     - Protokollverzeichnis der Schachstruktur

/ass/structs/jmp/
jmp.pic       - Graphik der Beispielstruktur von Falmagne
                & Doignon (1988a)
jmp.set       - komprimierte Mengendarstellung der Beispielstruktur
jmp.sur       - Surmisedarstellung der Beispielstruktur
jmp.dia       - Protokollverzeichnis der Beispielstruktur
```

5. Tools

```
/ass/tools/
Makefile      - Makefile zum Erstellen des Codes
dist          - Statistikprogramm
int2bool      - Programm zum Umwandeln der Integer- in die boolsche
                Darstellung
presproc      - Beispiel fuer einen Praesentationsprozess
r2set         - Programm zum Umwandeln der Relationendarstellung in
```

```
                    die Mengendarstellung
tel             - Programm zum Senden und Empfangen ueber die
                    Internetschnittstelle

/ass/tools/sources/tel
 tel.c          - Quellcode zu tel

/ass/tools/sources/presproc
 close.icn      - Closeicon fuer den Praesentationsprozess
 presproc.c     - Quellcode des Praesentationsprozesses
 windef.c       - Fensterdefinitionen fuer den Praesentationsprozess
```

2.1 Vom Quellcode zum ausführbaren Programm

Wechseln Sie in das Verzeichnis `.../ass/procs` und geben Sie eins der folgenden Kommandos ein:

`make assmon` - compilieren und linken des Diagnosemonitors

`make assunit` - compilieren und linken der unitären Diagnoseprozedur

`make asslike` - compilieren und linken der likelihoodbasierten Diagnoseprozedur

Die Übersetzung erfolgt automatisch und wird am Bildschirm protokolliert.

2.2 Erste Schritte

In diesem Kapitel können Sie sich einen ersten Eindruck von der Arbeitsweise der Diagnoseprozeduren verschaffen.

Wechseln Sie dazu zunächst in das Verzeichnis `.../ass/procs` und starten Sie den Diagnosemonitor mit dem Kommando `assmon &`. Der Diagnosemonitor belegt nun den ganzen Bildschirm, allerdings sind die meisten Fenster des Monitors noch leer.

Starten Sie eine Diagnoseprozedur, indem Sie mit dem Mauszeiger zum Beispiel den Knopf `LIKELI` im Fenster `CONTROL` betätigen. Im Fenster `STATUS` wird daraufhin angezeigt, daß die likelihoodbasierte Diagnoseprozedur `asslike` als Hintergrundprozeß läuft und sich im Zustand `BEGIN` befindet.

Die Zustände der Diagnoseprozeduren sind in Kap. 1.6 näher beschrieben. Die Diagnoseprozedur erwartet im Zustand `BEGIN` die Übermittlung eines Protokolldateinamens von einem Präsentationsprozeß.

Ein Präsentationsprozeß dient dazu, Aufgaben auf dem Bildschirm anzuzeigen, und steht mit dem Diagnoseprozeß über Internet in Verbindung (siehe auch Kap. 1.5, 3.2 und 5.4). Da Sie möglicherweise noch keinen eigenen

Präsentationsprozeß besitzen, wird mit den Diagnoseprozeduren ein Beispiel-Präsentationsprozeß mitgeliefert (siehe Anh. B).

Starten Sie den Beispiel-Präsentationsprozeß am besten auf einem zweiten Rechner, der über Internet angeschlossen ist. Für den Fall, daß Ihnen kein zweiter Rechner zur Verfügung steht, enthält der Diagnosemonitor links unten ein Fenster `cmdtool`, in das Sie Kommandos eingeben können. Wechseln Sie in das Verzeichnis `.../ass/tools` und geben Sie das Kommando `presproc &` ein. Es erscheint daraufhin das Fenster des Beispiel-Präsentationsprozesses.

Im Fenster des Beispiel-Präsentationsprozesses tragen Sie im Feld `Destination Host` den Namen des Rechners ein, auf dem der Diagnoseprozeß läuft (falls Sie den Beispielprozeß auf dem selben Rechner gestartet haben, ist keine Änderung notwendig). Im Feld `Internet Port` belassen Sie es bei der Standardeinstellung `1025`. Im Feld `Protocol File` tragen Sie den Namen einer Protokolldatei ein. Dabei müssen die in Kapitel 3.3.2 beschriebenen Namenskonventionen beachtet werden. Löschen Sie beispielsweise `tools/presproc` und setzen Sie dafür `structs/Chess/Chess.dia/Chess.Test.dia` ein. Wenn sie nun den Knopf `BEGIN` betätigen, wird der Name der Protokolldatei an den Diagnoseprozeß übermittelt.

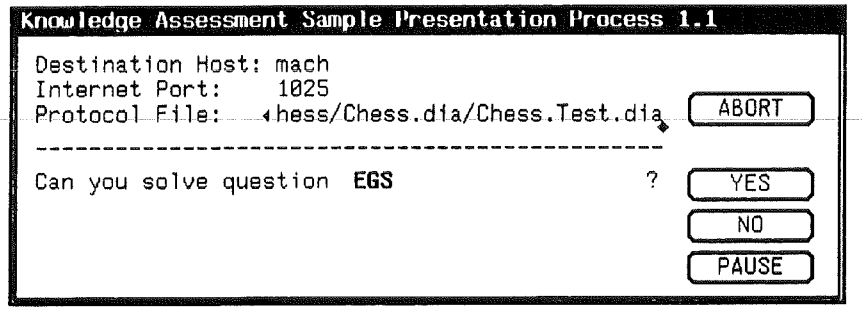


Abbildung 5: Beispiel für einen Präsentationsprozeß

Der Diagnoseprozeß lädt daraufhin zunächst die Schach-Wissensstruktur `Chess` und zeigt im Fenster `SURMISE` (oben, links) eine Graphik dieser Struktur an. Im Fenster `STRUCTURE` (rechts) finden Sie die Schachstruktur in Mengendarstellung. Im Fenster `QUESTIONS` (in der Mitte) finden Sie die Namen der Schachaufgaben aufgelistet. Die Inhalte der Fenster sind genauer in Kaptel 3.1.3 beschrieben.

Die Wissensstruktur `Chess` stammt aus einem Schachexperiment von Albert, Schrepp & Held. Welche konkreten Schachaufgaben sich hinter den Namenskürzeln (z.B. `EGS`) verbergen, ist in Albert, Schrepp & Held (1992) beschrieben.

Der Diagnoseprozeß hat inzwischen berechnet, mit welcher Aufgabe die Diagnose am besten begonnen werden kann. Entsprechend den ökonomischen

Kriterien aus Kap. 1 handelt es sich dabei immer um eine Aufgabe mittleren Schwierigkeitsgrades. Im Fall der Wissensstruktur `Chess` ist am Anfang immer Aufgabe `EGS` am günstigsten.

Der Name der Aufgabe, `EGS`, wurde inzwischen an den Beispiel-Präsentationsprozeß übermittelt, dessen Fenster nun so aussieht, wie in Abbildung 5 dargestellt. Wie man das Hypertext-System KMS als Präsentationsprozeß verwenden kann, um Schachaufgaben auf den Bildschirm zu bringen und von einer Versuchsperson spielen zu lassen, beschreibt Held (1991). Der Beispiel-Präsentationsprozeß beschränkt sich auf die Frage, ob sie die Aufgabe lösen können.

Nachdem Sie mit der Maus den Knopf `YES` betätigt haben, wird dies dem Diagnoseprozeß übermittelt, der daraufhin wie in Kap. 1 beschrieben die nächste Aufgabe auswählt. Ihr Diagnosebildschirm sieht nun wie in Abbildung 6 dargestellt aus.

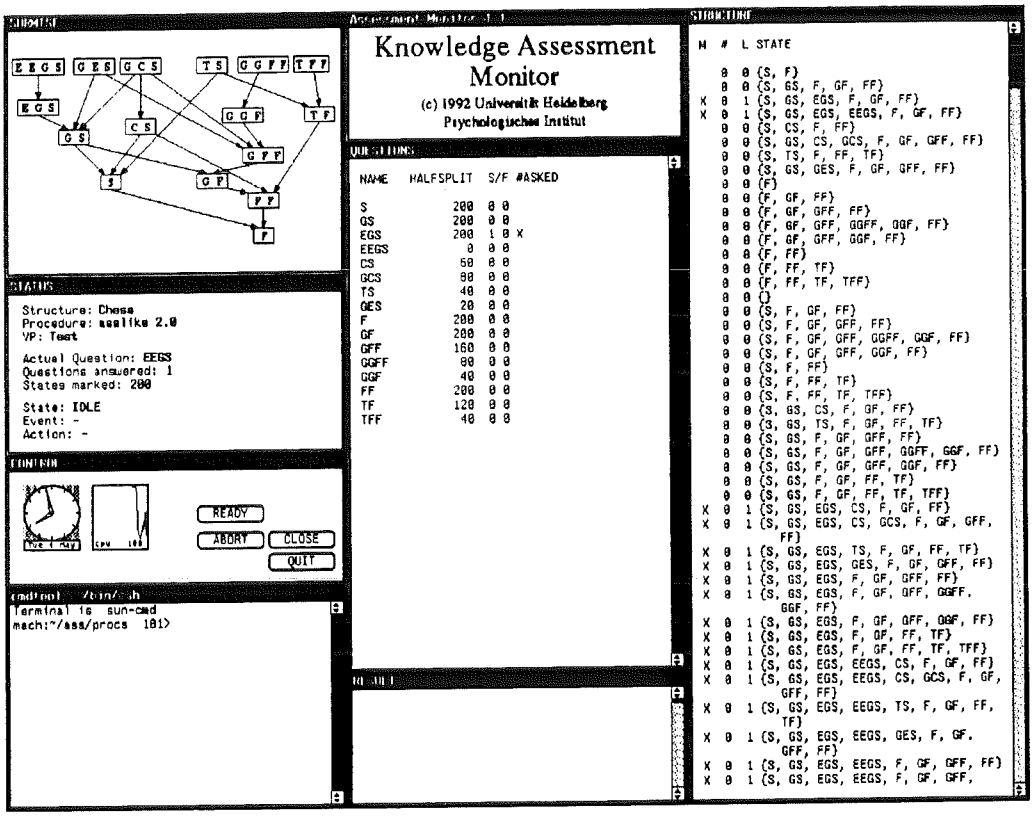


Abbildung 6: Der Diagnosemonitor

Alternativ zum Beispiel-Präsentationsprozeß können Sie auch das mitgelieferte Kommunikationsprogramm `tel` (siehe Anh. B) verwenden, um mit dem Diagnoseprozeß über Internet in Verbindung zu treten. Statt wie beschrieben den Knopf `YES` zu betätigen wechseln Sie im Fenster `cmdtool` in das Verzeichnis `.../ass/tools` und geben `tel SOLVED` ein, um zu dem gleichen

Ergebnis zu gelangen. Die möglichen Nachrichten sind im Nachrichtenkatalog in Kap. 3.2 beschrieben.

Nach ungefähr sieben Aufgaben wird im Fenster RESULT des Diagnosemonitors das erste Diagnoseergebnis angezeigt. Es handelt sich dabei um eine Menge von Aufgaben, die eine Versuchsperson (in diesem Fall Sie) vermutlich lösen kann. Nicht alle Aufgaben davon wurden präsentiert und bearbeitet, vielmehr wurde aus dem Lösen einiger Aufgaben auf die Fähigkeit geschlossen, bestimmte andere Aufgaben ebenfalls lösen zu können. Da Sie vielleicht während der Diagnose einen Flüchtigkeitsfehler gemacht oder eine Aufgabe gelernt haben, kann das Ergebnis durch weitere Fragen überprüft werden.

3 Schnittstellen

Für die Durchführung von Diagnosen sind die Benutzerschnittstelle und die Dateischnittstelle (Wissensstruktur und Protokoll) von Bedeutung.

Wenn Aufgaben und ein Präsentationsprozeß (siehe z.B. Held, 1991, sowie Anh. B) neu erstellt werden, ist zusätzlich die Kenntnis der Präsentationschnittstelle notwendig.

Für Softwareänderungen und Portierungen kann es nötig sein, die Internetschnittstelle und die Schnittstelle zum Betriebssystem zu berücksichtigen. Beide Schnittstellen beschreibt die Softwaredokumentation (Unnewehr, 1991).

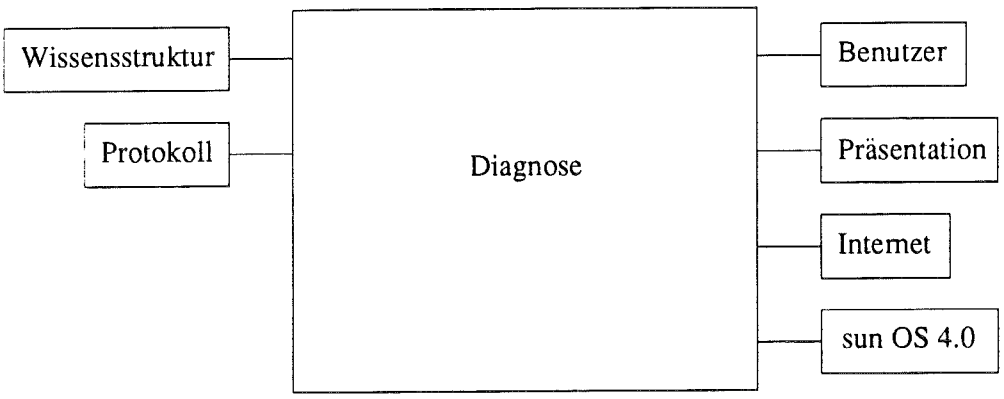


Abbildung 7: Schnittstellen

3.1 Benutzerschnittstelle

3.1.1 Kommandos Die Diagnoseprozeduren können direkt oder mit dem Diagnosemonitor gestartet werden. Zum direkten Start wechseln Sie in das Verzeichnis `.../ass/procs` und geben Sie eins der Kommandos

```
assunit [-simul file][-p port] &
asslike [-simul file][-p port] &
```

ein. Es erfolgt keine Rückmeldung. Die entsprechende Prozedur läuft nun als Hintergrundprozeß.

Wenn Sie das entsprechende Kommando ohne die Simulationsoption `-simul file` eingegeben haben, erwartet der Diagnoseprozeß nun über Internet die Nachrichten des Präsentationsprozesses (siehe z.B. Kap. 3.2.2, Kommunikationsbeispiel). Falls Sie noch nicht über eine Menge von Aufgaben und über einen Präsentationsprozeß verfügen, können Sie beispielsweise die Nachrichten des Kommunikationsbeispiels mit dem in Anhang B beschriebenen Hilfsprogramm `tel` senden und erhalten die entsprechenden Rückmeldungen. Alternativ können

Sie das ebenfalls in Anhang B beschriebene Beispielprogramm `presproc` als Präsentationsprozeß verwenden (siehe auch Kap. 2.2).

Wenn Sie mehrere Diagnosen gleichzeitig arbeiten lassen wollen, können Sie ihnen mit der Option `-p` verschiedene Internetports zuweisen (siehe auch Kap. 5.4). Standardeinstellung ist Port 1025.

Falls Sie die Simulationsoption verwendet haben, muß sich in der Datei *`file`* ein ausgefülltes Formular wie in Kap. 3.3.3 beschrieben befinden. Der Diagnoseprozeß berechnet dann die in dem Formular beschriebenen Simulationen. Das kann, je nach Fall, sehr lange dauern (siehe Beispiel in Kap. 3.3.3).

Das Arbeiten mit dem Monitor empfiehlt sich, wenn zwei Sun-Rechner zur Verfügung stehen. Die Diagnose kann dann über den Monitor beobachtet werden, während die zu lösenden Aufgaben auf einem zweiten Rechner dargeboten werden. Zum Starten mit dem Monitor wechseln Sie ebenfalls in das Verzeichnis `.../ass/procs` und geben Sie das Kommando

```
assmon &
```

ein. Der Diagnosemonitor wird nun aufgebaut. Drücken Sie mit dem Mauszeiger einen Prozedurstartknopf wie in Kap. 3.1.3 beschrieben und starten Sie damit eine Diagnoseprozedur.

3.1.2 Fehlermeldungen Alle Meldungen über auftretende Fehler der Diagnoseprozeduren (Kap. 4, Fehlerbehandlung) werden auf den Diagnosemonitor oder auf den Bildschirm (stderr) ausgegeben und in die Protokolldatei geschrieben.

3.1.3 Der Diagnosemonitor Zur Beobachtung und Beendigung der Diagnosen sowie zur Analyse der Diagnoseprozeduren wurde ein Diagnosemonitor angelegt, der verschiedene Elemente einer Diagnose wie in Kap. 1 eingeführt veranschaulicht (siehe Abb. 6).

Während eine Person an einem Rechner Aufgaben löst, kann an einem zweiten genau mitverfolgt werden, nach welchen Kriterien die Diagnoseprozedur Fragen auswählt, welche Wissenszustände markiert sind, und welche Diagnose aktuell gestellt würde. Weiterhin werden Häufigkeits- oder Likelihoodverteilungen angezeigt, die während der Diagnose auf der Menge der Wissenszustände errechnet werden.

Der Diagnosemonitor wird wie in Kap. 3.1.1 beschrieben aufgerufen. Im Bereich `CONTROL` kann dann durch Betätigen des Knopfes `UNITARY` oder des Knopfes `LIKELI` die unitäre oder die likelihoodbasierte Diagnoseprozedur gestartet werden.

Der Diagnoseprozeß erwartet nun über Internet die Nachrichten des Präsentationsprozesses (siehe z.B. Kap. 3.2.2, Kommunikationsbeispiel). Falls Sie noch nicht über eine Menge von Aufgaben und über einen Präsentationsprozeß

verfügen, können Sie die Nachrichten des Kommunikationsbeispiels mit dem in Anhang B beschriebenen Hilfsprogramm `tel` senden und erhalten die entsprechenden Rückmeldungen. Alternativ können Sie das ebenfalls in Anhang B beschriebene Beispielprogramm `presproc` als Präsentationsprozeß verwenden (siehe auch Kap. 2.2).

Die über den Monitor dargebotene Information gliedert sich wie folgt in die einzelnen Bereiche:

3.1.3.1 CONTROL Im Bereich `CONTROL` sind alle Möglichkeiten zur Steuerung des Diagnosemonitors und der Diagnoseprozesse zusammengefaßt. Die Steuerung erfolgt durch das Betätigen verschiedener Knöpfe:

1. `PARAM`: Parameterdatei für den Monitor ändern (s. u.)
2. `UNITARY`: unitäre Diagnoseprozedur als Prozeß starten
3. `LIKELI`: likelihoodbasierte Diagnoseprozedur als Prozeß starten
4. `SIMULATE`: Simulationsoption (geplant)
5. `READY`: Beenden einer Diagnose von Hand (ersetzt das noch nicht spezifizierte Beendigungskriterium in Kap. 1.1 und 1.2)
6. `ABORT`: Diagnoseprozeß abbrechen (Gefahr des Datenverlusts)
7. `END`: Diagnoseprozeß normal beenden
8. `CLOSE`: Diagnosemonitor schließen
9. `QUIT`: Diagnosemonitor beenden (falls noch ein Diagnoseprozeß arbeitet: `ABORT`)

```
logo file: ./assmon.pic

button name for proc 1: UNITARY
proc 1 file: ./assunit
internet port: 1025

button name for proc 2: LIKELI
proc 2 file: ./asslike
internet port: 1025

button name for proc 3: -
proc 3 file: -
internet port: -

button name for proc 4: -
proc 4 file: -
internet port: -
```

Abbildung 8: Beispiel einer Parameterdatei

zu 1): Die Parameterdatei `assmon.par` befindet sich im Verzeichnis des Diagnosemonitors. Sie enthält ein Formular, in das die folgenden Konfigurationsinformationen eingetragen sind: 1) Verzeichnis und Name der Datei, die das Logo des Diagnosemonitors enthält und 2) Verzeichnis und Dateiname der Diagnoseprozeduren, die Beschriftung des entsprechenden Knopfes im Bereich `CONTROL` sowie den Internetport, über den die betreffende Prozedur kommuniziert. Maximal sind vier Einträge möglich. Falls diese Einträge fehlen, können die entsprechenden Prozeduren nicht vom Monitor aus gestartet werden. Als Beispiel dient hier die Konfiguration in der mitgelieferten Parameterdatei (Abb. 8).

3.1.3.2 STATUS Im Bereich `STATUS` werden Parameter und Zustände des laufenden Diagnoseprozesses angezeigt. Genauer handelt es sich dabei um:

1. `Structure`: Name der geladenen Wisensstruktur
2. `Procedure`: Name und Version der gestarteten Diagnoseprozedur
3. `VP`: Name der Versuchsperson
4. `Actual Question`: Name der gerade gestellten Aufgabe Q_n
5. `Questions answered`: Anzahl der bereits beantworteten Fragen
6. `States marked`: Anzahl der markierten Wissenszustände im Marker M_n (entspricht der Anzahl der Markierungen im Bereich `STRUCTURE`)
7. `State`: Prozeßzustand entspr. Kap. 1.6
8. `Event`: Ereignis entspr. Kap. 1.6
9. `Action`: Aktion entspr. Kap. 1.6

3.1.3.3 SURMISE Im Bereich `SURMISE` wird die Wissensstruktur als Surmise-Relation auf der Aufgabenmenge dargestellt. Diese Darstellung ist äquivalent zu der Mengendarstellung im Bereich `STRUCTURE` (dazu siehe Doignon & Falmagne, 1985).

Damit die Surmise-Darstellung vom Monitor geladen wird, muß sie als Sun-Picture-Graphik unter dem Namen der Wissensstruktur mit der Extention `.pic` im gleichen Verzeichnis gespeichert sein wie die Wissensstruktur (Beispiel aus Kap. 2: `.../ass/structs/Chess/Chess.pic`).

3.1.3.4 STRUCTURE Im Bereich `STRUCTURE` wird die Wissensstruktur als Menge von Wissenszuständen dargestellt. Jeder Wissenszustand ist wiederum eine Teilmenge der Aufgabenmenge Q (siehe Def. 1 in Kap. 1). Diese Darstellung ist äquivalent zu der Darstellung als Surmise-Relation im Bereich `SURMISE`.

Über jeden Wissenszustand wird dabei die Information angezeigt, ob er gerade markiert ist (X oder kein X), wie häufig er als vorläufiges Ergebnis ermittelt wurde und wie groß die Likelihood L' nach Formel 9 in Kap. 1.2 ist.

3.1.3.5 QUESTIONS Im Bereich QUESTIONS werden die Namen aller Aufgaben der geladenen Wissensstruktur aufgelistet.

NAME	HALFSPLIT	S/F	#ASKED
S	60	0 0	
GS	2	0 0	
EGS	82	0 1	X
EEGS	82	0 0	
CS	2	0 0	
GCS	62	0 0	
TS	26	0 0	
GES	52	0 0	
F	80	0 0	
GF	50	0 0	
GFF	4	0 0	
GGFF	82	0 0	
GGF	82	0 1	X
FF	76	0 0	
TF	42	0 0	
TFF	20	0 0	

In der Spalte Halfsplit wird mit Formel 4 aus Kap. 1.1 berechnet, wie die entsprechende Aufgabe den Marker teilen würde, wenn sie als nächste Frage gestellt würde. Außerdem ist angeführt, wie oft eine Aufgabe von einer Person gelöst (S), nicht gelöst (F) und bearbeitet (#ASKED) wurde. Die Anzahl der Bearbeitungen ist in einem Histogramm dargestellt (für jede Bearbeitung ein 'X').

In dem hier angeführten Beispiel wurden die Aufgaben EGS und GGF nicht gelöst. Bezüglich des Halfsplit verhalten sich die Aufgaben GS und CS am günstigsten. Eine von beiden Aufgaben wird per Zufall ausgewählt und als Frage gestellt. Unabhängig davon, ob sie gelöst oder nicht gelöst wird, kann der Marker in etwa halbiert werden (± 2 Wissenszustände).

3.1.3.6 RESULT Im Bereich RESULT werden alle bisher ermittelten vorläufigen Ergebnisse aufgelistet. Die Darstellung der Wissenszustände entspricht dabei der im Bereich STRUCTURE.

3.1.3.7 cmdtool Da der Diagnosemonitor den ganzen Bildschirm ausfüllt, wurde ein cmdtool integriert, von dem aus Betriebssystemkommandos gegeben werden können. Beispielsweise ist es möglich, dort einen Präsentationsprozeß zu starten oder über Internet mit dem Diagnoseprozeß zu kommunizieren (siehe auch Kap. 2.2).

3.2 Präsentationschnittstelle

Die Kommunikation des Diagnoseprozesses mit dem Präsentationsprozeß erfolgt über Internet. Für die Präsentationsseite kann dazu das Hilfsprogramm `tel` verwendet werden (s. Anh. B).

Die Kommunikation erfolgt so, daß Präsentations- und Diagnoseprozeß abwechselnd senden und empfangen (Handshake-Verfahren). Dabei tauschen sie folgende Nachrichten aus:

3.2.1 Nachrichtenkatalog

1. Präsentation —> Diagnose
   1. Name der Protokolldatei entsprechend Kap. 3.3.2:

      ```
      [Pfad]/[Versuchsname]/[Versuchsname].dia
        /[Versuchsname][Vp-Name].dia
      ```

   2. Richtigkeit der Aufgabenlösung:

      `SOLVED` oder `FAILED`

   3. Pause-Zeichen zur Unterbrechung der Diagnose:

      `PAUSE`

   4. Beendigung des Diagnoseprozesses:

      `END`

2. Diagnose —> Präsentation
   1. Aufgabenname
   2. Fertig-Zeichen, wenn die Diagnose zu Ende ist:

      `READY`

   3. Fehler-Zeichen, wenn ein Fehlerzustand aufgetreten ist (die Diagnose befindet sich dann immer im Zustand "Begin"):

      `ERROR`

   4. Pause-Zeichen als Empfangsbestätigung eines Pause-Zeichens:

      `PAUSE`

   5. Ende-Zeichen als Empfangsbestätigung eines Ende-Zeichens:

      `END`

3.2.2 Kommunikationsbeispiel Der Arbeitsweise des Diagnoseprozesses (Kap. 1.6, Arbeitsweise) entsprechend kann es zu dem in Abb. 9 dargestellten Kommunikationsbeispiel kommen.

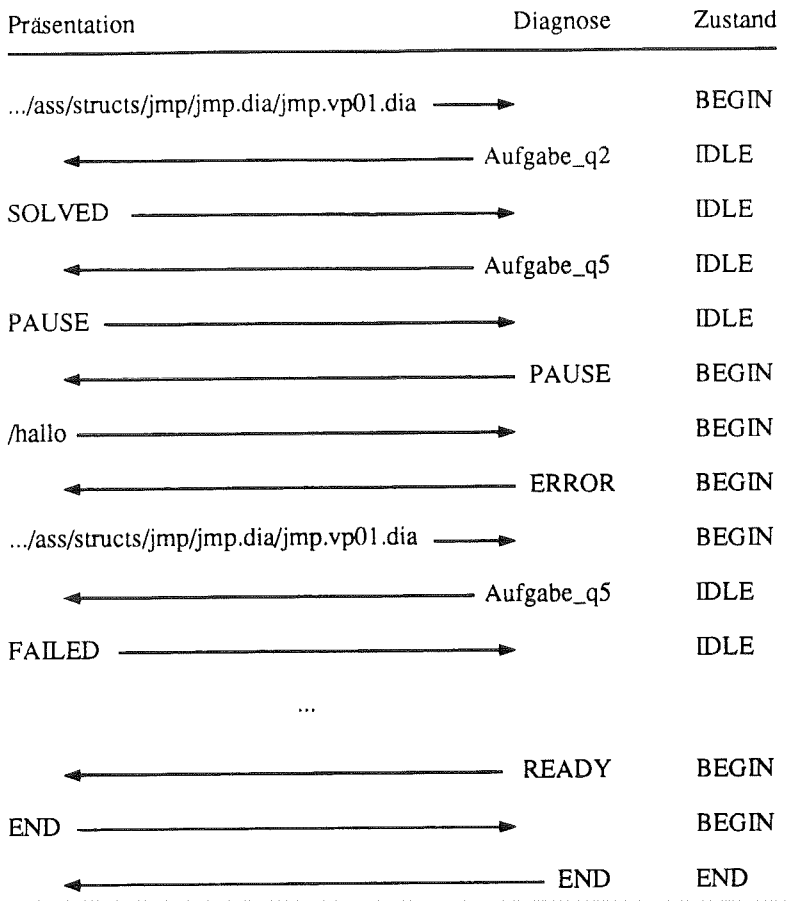


Abbildung 9: Kommunikationsbeispiel

Der Präsentationsprozeß sendet den korrekten Namen einer Protokolldatei entsprechend Kap. 3.3, Dateischnittstelle. Der Diagnoseprozeß liest die Datei und die dazu gehörende Wissensstruktur ein, geht in den Zustand IDLE über, berechnet die erste Aufgabe und sendet den Aufgabennamen. Der Präsentationsprozeß stellt die Aufgabe und sendet die Richtigkeit der Lösung. Der Diagnoseprozeß sendet den nächsten Aufgabennamen. Der Präsentationsprozeß sendet ein Pausezeichen. Der Diagnoseprozeß geht in den Zustand BEGIN, bestätigt das Pausezeichen und erwartet den Namen der nächsten Protokolldatei. Der Präsentationsprozeß sendet einen falschen Namen. Der Diagnoseprozeß antwortet mit einem Fehlerzeichen. Der Präsentationsprozeß sendet einen richtigen Namen. Die erste Diagnose wird wieder aufgenommen und zu Ende geführt. Der Diagnoseprozeß sendet ein Fertigzeichen. Der Präsentationsprozeß antwortet mit einem Endezeichen. Der Diagnoseprozeß beendet sich und müßte nun für eine weitere Diagnose neu gestartet werden (siehe Kap. 3.1.1, Kommandos).

3.3 Dateischnittstelle

Alle Dateien, die die Diagnoseprozedur für eine Diagnose benötigt, sind in einem Verzeichnis organisiert, das den Namen eines Versuchs trägt.

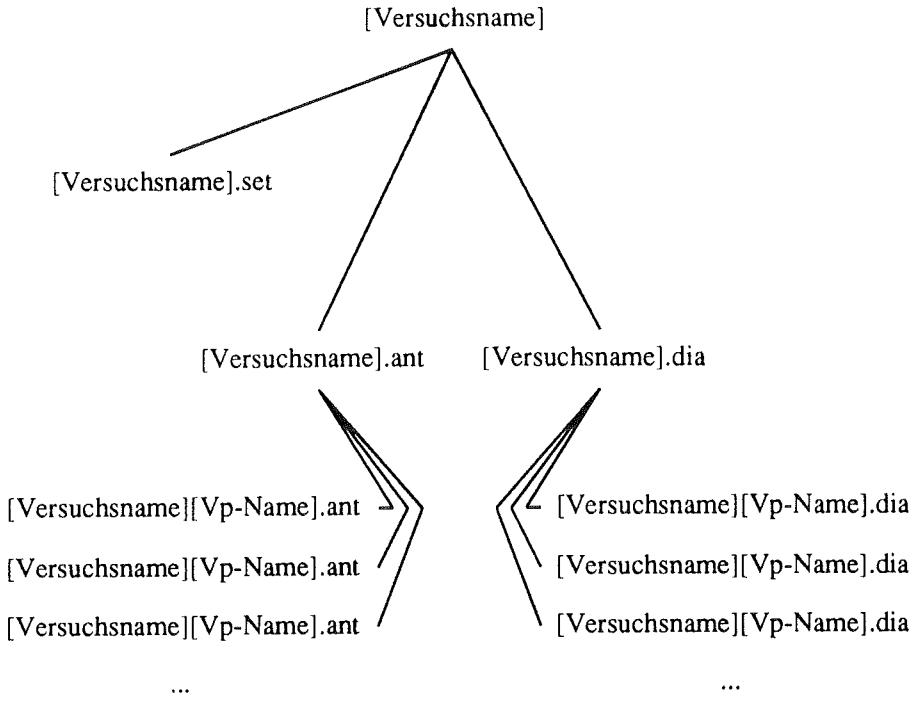


Abbildung 10: Dateiorganisation

Es enthält:

- eine Wissensstruktur in der Datei "[Versuchsname].set" zur Steuerung der Diagnose
- ein Unterverzeichnis "[Versuchsname].dia", das die Diagnoseprotokolle aufnimmt
- optional ein Unterverzeichnis "[Versuchsname].ant", das Antwortprotokolle der Präsentationsoberfläche aufnimmt

Als Ergebnisdatei für Simulationen (siehe Kap. 3.1.1) kann eine beliebige Datei im in Kapitel 3.3.3 beschriebenen Format angegeben werden.

3.3.1 Wissensstruktur Das Diagnoseprogramm arbeitet mit einer Wissensstruktur, die es aus der Datei "[Versuchsname]/[Versuchsname].set" einliest.

Diese Datei enthält eine Referenzliste, durch die die Aufgaben der Wissensstruktur beginnend mit eins und ohne eine Zahl zu überspringen durchnumeriert werden:

1 Aufgabenname_x

2 Aufgabenname_y

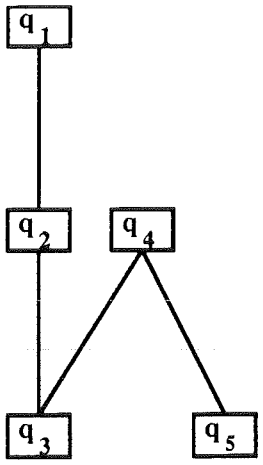
...

|Q| Aufgabenname_z

Der Referenzliste folgt eine Formatangabe

```
Format:    |κ|  X  |Q|  (|Q'|),
```

wobei |κ| X |Q| Anzahl der Zeilen und Anzahl der Spalten einer Matrix angibt, die als Menge von Wissenszuständen interpretiert wird (siehe Abb. 11).



Relation	Menge	Matrix	komprim. Matrix
	0	0 0 0 0 0	0
	{5}	0 0 0 0 1	16
	{3}	0 0 1 0 0	4
	{3, 5}	0 0 1 0 1	20
	{2, 3, 5}	0 1 1 0 1	22
	{1, 2, 3, 5}	1 1 1 0 1	23
	{1, 2, 3}	1 1 1 0 0	7
	{2, 3}	0 1 1 0 0	6
	{3, 4, 5}	0 0 1 1 1	28
	{2, 3, 4, 5}	0 1 1 1 1	30
	Q	1 1 1 1 1	31

Abbildung 11: Verschiedene Darstellungen einer Wissensstruktur

Eine Spalte der Matrix repräsentiert dabei die jeweilige Aufgabe entsprechend der Referenzliste, eine Zeile einen Wissenszustand. Eine Eins in Spalte i, Zeile j legt fest, daß Aufgabe i von einer Person mit Wissenszustand j gelöst werden kann. Eine Null steht für das Gegenteil.

Die Matrix kann auch in komprimierter Form, in Integer-Darstellung, vorliegen. In diesem Fall wurde ein boolscher Zeilenvektor in die Integer-Zahl umgewandelt, die er repräsentiert (siehe Unnewehr, 1991). Die Formatangabe |Q'| gibt dann die ursprüngliche Breite der Matrix an.

Die Anzahl |Q| ist auf maximal 32 Fragen beschränkt. Diese Grenze wird evtl. in einer späteren Version der Prozeduren aufgehoben.

Die Wissensstruktur kann mit dem Hilfsprogramm `r2set` (s. Anh. B) aus einer Relationendatei erstellt werden.

3.3.2 Protokoll Mit Hilfe des Protokolls können unterbrochene oder abgebrochene Diagnosen wieder aufgenommen werden. Außerdem kann das Protokoll zur statistischen Auswertung der Diagnosen verwandt werden. Es enthält die Nummer des Schritts n, die beiden in der Einführung vorgestellten Zufallsvariablen R und Q, die Näherung an den Wissenszustand der geprüften Person (soweit berechnet), sowie Kommentare, Pause- und Fehlermeldungen. Es kann beispielsweise wie in Abb. 12 aussehen:

```
/* Assessment started: Mon Jun 29 17:46:41 1992
/* Structure: jmp/jmp.set
/* Procedure: asslike 2.0
/* VP: jmp/jmp.dia/jmp.test.dia
/* n - R - Q - K (assessed)
1 1 2 0
/* Assessment stopped.
/* Assessment started: Mon Jun 29 17:59:39 1992
2 0 5 0
3 0 1 8
```

Abbildung 12: Beispiel einer Protokolldatei

Das Protokollbeispiel ist wie folgt zu verstehen: Am 29. Juni wurde als erstes die zweite Frage der Wissensstruktur "jmp" von der Versuchsperson "test" richtig beantwortet. Die likelihoodbasierte Diagnoseprozedur konnte mit dieser Information noch keinen Wissenszustand diagnostizieren. Nachdem die Diagnose unterbrochen und wieder aufgenommen wurde, wurde die fünfte Frage gestellt und falsch beantwortet. Nachdem auch die erste Frage falsch beantwortet wurde, stellte die Prozedur die Diagnose: Zustand Nummer 8 der Wissensstruktur. Die Wissensstruktur "jmp" wurde bereits in Kap. 3.3.1 als Beispiel verwandt. Das Protokoll ist Ergebnis des Kommunikationsbeispiels aus Kap. 3.2.2.

Das Protokoll befindet sich jeweils in der Datei [Versuchsname]/[Versuchsname].dia/[Versuchsname][Vp-Name].dia.

3.3.3 Simulationsergebnis Simulationen können durchgeführt werden, indem die entsprechende Diagnoseprozedur mit der Simulationsoption unter Angabe der Ergebnisdatei gestartet werden (siehe Kap. 3.1.1). Die Ergebnisdatei muß ein Formular mit den folgenden Einträgen enthalten:

1. `structure`: Dateiname der Wissensstruktur entspr. Kap. 3.3
2. `procedure`: don't care
3. `#assessments`: Anzahl der Simulationsläufe

4. `#questions`: Anzahl der Fragen pro Simulationslauf
5. `learning`: 0 (geplant als künftige Erweiterung)
6. `b`: Wahrscheinlichkeit für Flüchtigkeitsfehler in Prozent (0–100)
7. `g`: Wahrscheinlichkeit für Glückstreffer in Prozent (0–100)
8. `saving steps`: Anzahl der Simulationsläufe, nach denen die Daten gesichert werden sollen
9. `started`: don't care
10. `to end` : don't care

Die mit "don't care" beschriebenen Daten werden nach jedem Sicherungsschritt mit den aktuellen Werten überschrieben.

Methodisch wird bei der Simulation so vorgegangen, daß aus den Wissenszuständen der angegebenen Wissensstruktur "Anzahl der Simulationsläufe" mal ein Zustand per Zufall ausgewählt und als Zustand einer simulierten Person festgelegt wird. Dieser Person werden "Anzahl der Fragen" mal Fragen gestellt, wobei sie jeweils mit einer Wahrscheinlichkeit von "b" einen Flüchtigkeitsfehler macht bzw. mit einer Wahrscheinlichkeit von "g" die Antwort rät.

An das Formular angehängt wird bei den Sicherungen eine Ergebnismatrix im Format [Anzahl der gestellten Fragen+1] X [$|Q|$+1], wobei der Wert x in Zeile i, Spalte j für das Ereignis steht "bei x Simulationsläufen hatte nach i-1 Fragen der einzige markierte Wissenszustand eine Distanz von j-1 zum tatsächlichen Wissenszustand der simulierten Person". Diese Distanz wurde in Kap. 1 (siehe Def. 2) als Diagnosefehler definiert. Die maximale Anzahl der Fehler ist determiniert durch die Anzahl der Aufgaben $|Q|$.

In die Berechnung gehen ausschließlich die Fälle ein, in denen die Diagnoseprozedur nur einen Wissenszustand markiert hat, d.h. in denen ein "vorläufiges Ergebnis" berechnet wurde.

Abbildung 13 gibt ein Beispiel. Dabei wurde eine Simulation mit der likelihoodbasierten Diagnoseprozedur und der Schach-Wissensstruktur durchgeführt, die sich unter dem Namen `Chess.set` im Verzeichnis `Chess` befand. Von den Zuständen der Schach-Struktur wurde per Zufall 10000 mal ein Zustand ausgewählt und als Wissenszustand einer simulierten Person festgelegt. Jeder Person wurden 50 Fragen gestellt, wobei sie mit einer Wahrscheinlichkeit von 0.1 zu jeder Frage die richtige Antwort riet bzw. einen Flüchtigkeitsfehler machte. Nach jeweils 500 Durchläufen wurden die Daten gesichert. Die Simulation dauerte etwa 15 Stunden.

Als Ergebnis findet sich z.B. in Zeile 8, Spalte 2 der Matrix der Wert 92. Das ist so zu verstehen, daß nach 7 Fragen bei 92 simulierten Versuchspersonen der diagnostizierte Wissenszustand eine Distanz von 1 zu dem tatsächlichen Zustand der Person aufwies.

```
structure: Chess/Chess.set
procedure: asslike 2.0
#assessments: 10000
#questions: 50
learning: 0
b: 10
g: 10

saving steps: 500
started: Fri Jun 19 21:49:22 1992
to end : Sat Jun 20 12:20:46 1992

Format: 51 X 17
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
180 92 87 33 15 8 2 0 0 0 0 0 0 0 0 0 0
1704 1150 840 346 146 55 18 4 3 0 0 0 0 0 0 0 0 0
3307 2181 1275 522 180 64 15 6 1 0 0 0 0 0 0 0 0 0
3682 2218 995 349 95 32 9 2 0 0 0 0 0 0 0 0 0
3931 2071 829 276 76 30 8 3 0 0 0 0 0 0 0 0 0
4527 2229 761 248 78 18 9 0 0 0 0 0 0 0 0 0 0
4982 2133 643 189 49 17 5 0 0 0 0 0 0 0 0 0 0
5364 2001 587 154 41 14 2 0 0 0 0 0 0 0 0 0 0
5893 1809 537 139 33 8 2 0 0 0 0 0 0 0 0 0 0
6324 1629 479 101 32 7 0 0 0 0 0 0 0 0 0 0 0
6779 1428 409 86 27 5 0 0 0 0 0 0 0 0 0 0 0
7155 1271 350 69 25 3 0 0 0 0 0 0 0 0 0 0 0
7520 1133 289 47 15 1 0 0 0 0 0 0 0 0 0 0 0
7832 1038 235 51 12 0 0 0 0 0 0 0 0 0 0 0 0
8143 893 186 34 8 1 0 0 0 0 0 0 0 0 0 0 0
8418 791 145 27 6 0 0 0 0 0 0 0 0 0 0 0 0
8677 697 125 26 4 0 0 0 0 0 0 0 0 0 0 0 0
8849 600 95 20 3 0 0 0 0 0 0 0 0 0 0 0 0
9022 523 74 20 1 0 0 0 0 0 0 0 0 0 0 0 0
9180 459 50 16 1 0 0 0 0 0 0 0 0 0 0 0 0
9290 388 45 7 1 0 0 0 0 0 0 0 0 0 0 0 0
9397 368 38 6 1 0 0 0 0 0 0 0 0 0 0 0 0
9477 312 30 4 1 0 0 0 0 0 0 0 0 0 0 0 0
9560 264 28 2 1 0 0 0 0 0 0 0 0 0 0 0 0
9622 228 21 2 1 0 0 0 0 0 0 0 0 0 0 0 0
9674 186 22 3 1 0 0 0 0 0 0 0 0 0 0 0 0
9716 164 17 1 0 0 0 0 0 0 0 0 0 0 0 0 0
9769 127 12 1 1 0 0 0 0 0 0 0 0 0 0 0 0
9813 112 12 1 1 0 0 0 0 0 0 0 0 0 0 0 0
9844 82 8 1 0 0 0 0 0 0 0 0 0 0 0 0 0
...
```

Abbildung 13: Beispiel einer Simulationsergebnisdatei

Wie bereits erwähnt gingen in die Berechnung nur die Fälle ein, in denen ein "vorläufiges" Ergebnis berechnet wurde. Während der ersten 6 Fragen wurden noch keine Ergebnisse berechnet.

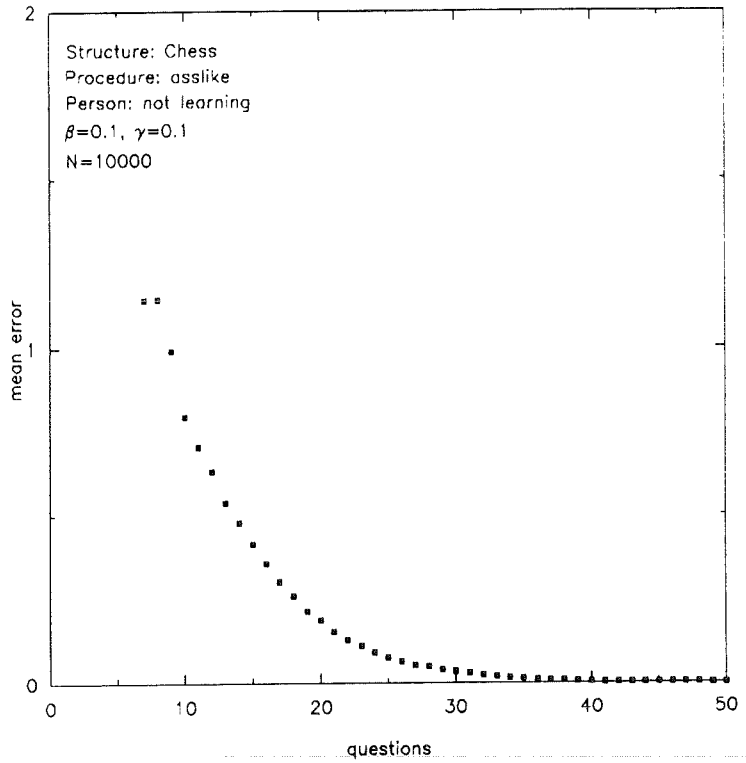


Abbildung 14: Beispiel einer Simulationsauswertung

Berechnet man für jede Zeile der Matrix die mittlere Distanz, so gelangt man zu dem in Abb. 14 dargestellten Ergebnis. Eine ausführliche Simulationsstudie mit verschiedenen Wissensstrukturen, Diagnoseprozeduren und Parametern der simulierten Population ist in Vorbereitung.

4 Fehlerbehandlung

Wenn während der Diagnose Fehler auftreten, wird zur Information des Benutzers eine entsprechende Fehlermeldung auf den Bildschirm (stderr) bzw. auf den Diagnosemonitor ausgegeben (s. Kap. 3.1.2, Fehlermeldungen). Die gleiche Fehlermeldung wird in die Protokolldatei geschrieben, wenn der Fehler nicht das Öffnen und Schreiben der Protokolldatei betrifft (s. Kap. 3.3.2, Protokoll).

Im Groben kann zwischen Kommunikationsfehlern (1. - 5.), Fehlern der Dateischnittstelle (6. - 8.), Software-Fehlern (9. - 10.) und internen Fehlern (11. - 13.) unterschieden werden.

Folgende Fehler können auftreten:

1. `internet error`
2. `connecting error`
3. `receiving error`
4. `sending error`
5. `wrong answer: [answer]`
6. `wrong filename: [filename]`
7. `no access: [filename]`
8. `wrong format: [filename]`
9. `undefined set`
10. `undefined element`
11. `empty marker`
12. `history overflow`
13. `memory overflow`

Die Beseitigung der Fehler geschieht mit Hilfe der Checkliste in Anhang A.

5 Betriebliche Kennwerte

5.1 Systemverhalten Die Protokolldatei wird während der Diagnose immer auf dem neuesten Stand gehalten. Nach einem Abbruch des Diagnoseprozesses (z.B. durch Systemausfall) kann die Diagnose deshalb durch einfaches Neustarten fortgesetzt werden.

5.2 Laufzeit Das Laufzeitverhalten des unitären Diagnoseprozesses wurde im Rahmen einer Komplexitätsanalyse ausführlich untersucht (siehe Unnewehr, 1990a). Die Komplexität hängt von der Anzahl der Aufgaben $|Q|$ und der Wissenszustände $|\kappa|$ der Wissensstruktur des Wissensgebietes ab (siehe Tabelle 1).

Prozedur	Komplexität
assunit	$O\left(\|Q\| * \|\kappa\| + \|Q\|^2\right)$

Tabelle 1: Komplexität

Als besonders zeitkritisch erwies sich die Berechnung der ersten Aufgaben. Auf einer Sun 3/140 mit 8 MB Hauptspeicher wurden bei konstantem $|Q|$=27 die in Tabelle 2 enthaltenen Laufzeiten (in sec.) gemessen:

$\|\kappa\|$	$t(\kappa$ laden)	$t(Q_1) - t$(laden)	$t(Q_2) - t(Q_1)$
100	0	0	0
500	0	1	1
1000	1	2	2
5000	2	7	6
10000	5	12	10
50000	25	56	44
100000	53	118	86

Tabelle 2: Laufzeiten (in sec)

Entsprechend der Komplexität steigt die Laufzeit ungefähr linear an. Bei einer Wissensstruktur mit 5000 Wissenszuständen benötigte die Prozedur 2 sec. zum Laden, dazu 7 sec. zum Berechnen der ersten Frage und 6 sec. zum Berechnen der zweiten Frage.

5.3 Speicherbedarf Der Diagnoseprozeß hat einen Grundbedarf von ca. 120 KB Hauptspeicher und einen Zusatzbedarf für einige speicherintensiven Datenstrukturen, der dynamisch belegt wird. Es handelt sich dabei um drei Marker und eine Wissensstruktur. Für $|\kappa|$ Wissenszustände benötigt ein Marker 4*$|\kappa|$ byte Speicher. Der Gesamtspeicherbedarf S läßt sich mit Formel 11 abschätzen.

$$S = 120 \text{ KB} + 16 * |\kappa| \text{ B} \tag{11}$$

Der Speicherbedarf ist dabei unabhängig von der Anzahl der Fragen $|Q|$ der Wissensstruktur, da diese Anzahl auf maximal 32 Fragen beschränkt ist. Jeder Wissenszustand belegt intern genau 4 byte Speicherplatz (siehe auch Kap. 3.3.1).

Einige Beispiele gibt folgende Tabelle:

$\|\kappa\|$	KB
100	122
500	128
1000	136
5000	300
10000	280
50000	920
100000	1720

Tabelle 3: Speicherbedarf

Ist nicht genügend Hauptspeicher vorhanden, tritt Fehler 13 auf (s. Kap. 4, Fehlerbehandlung).

Die Wissensstrukturdatei belegt in der komprimierten Form etwa 4*$|\kappa|$ B Speicherplatz, während für die Protokolldatei nur ein geringer Speicheraufwand benötigt wird.

Das Arbeiten mit Wissensstrukturen > 100 000 Zuständen wird wegen der ungünstigen Antwortzeiten und des hohen Speicherbedarfs nicht empfohlen.

5.4 Sonstige Ressourcen Der Diagnoseprozeß belegt einen Internet-Port. Als Defaultwert ist Port 1025 eingestellt. Wenn mehrere Diagnosen in einem Netz parallel arbeiten sollen, müssen ihnen mit der Startoption –p unterschiedliche Ports zugewiesen werden (siehe Kap. 3.1.1).

6 Tests

Die Diagnoseprozeduren können mit den in Tabelle 4 aufgeführten Testschritten auf ihre korrekte Funktion hin überprüft werden.

Testschritt	Testfall	Ergebnis
1	Diagnose starten	-
2	Diagnose erneut starten	Fehler 1
3	falschen Protokolldateinamen senden	Fehler 5
4	Protokolldatei sperren und Namen senden	Fehler 7
5	Strukturdatei sperren und Namen senden	Fehler 7
6	falsche Datei als Protokolldatei installieren und Namen senden	Fehler 8
7	falsche Datei als Strukturdatei installieren und Namen senden	Fehler 8
8	Hauptspeicher belegen und zu große Struktur laden	Fehler 13
9	Diagnose entsprechend Kommunikationsbeispiel in Kap. 3.2.2 durchführen	siehe Kap. 3.2.2
10	unitäre Diagnose mit nicht wohlgraduierter Wissensstruktur durchführen	Fehler 11

Tabelle 4: Testschritte

Literaturverzeichnis

Albert, D., Schrepp, M. & Held, T. (1992). *Construction of Knowledge Spaces for Problem Solving in Chess — Two Experimental Investigations.* Bericht aus dem Psychologischen Institut der Universität Heidelberg, Bericht Nr. 71, Heidelberg: Universität Heidelberg.

Doignon, J.-P. & Falmagne, J.-C. (1985). Spaces for the assessment of knowledge. *International Journal of Man-Machine Studies*, **23**, 175–196

Dowling, C.-E. (im Druck-a). Applying the Basis of a Knowledge Space for Controlling the Questioning of an Expert. *Journal of Mathematical Psychology*

Dowling, C.-E. (im Druck-b). On the Irredundant Generation of Knowledge Spaces. *Journal of Mathematical Psychology*

Falmagne, J.-C. & Doignon, J.-P. (1988a). A Markovian Procedure for Assessing the State of a System. *Journal of Mathematical Psychology*, **32** (3), 232–258

Falmagne, J.-C. & Doignon, J.-P. (1988b). A class of stochastic procedures for the assessment of knowledge. *British Journal of Mathematical and Statistical Psychology*, **41**, 1–23

Held, T. (1991). Eine KMS-Oberfläche zur Versuchssteuerung – Hinweise zur Benutzung. *Arbeitsbericht aus dem Projekt "Wissensstruktur" (Az. Lu 385/1–1) im Schwerpunktprogramm "Wissenspsychologie" der Deutschen Forschungsgemeinschaft.*

Koppen, M. & Doignon, J.P. (1990). How to Build a Knowledge Space by Querying an Expert. *Journal of Mathematical Psychology*, **34**, 311–331

Lukas, J. (1990). Algorithmen zur Berechnung bestimmter Eigenschaften von Relationen und eine Anwendung auf die Untersuchung von Wissensstrukturen. *Arbeitsbericht aus dem Projekt "Wissensstruktur" (Az. Lu 385/1–1) im Schwerpunktprogramm "Wissenspsychologie" der Deutschen Forschungsgemeinschaft.*

Unnewehr, J. (1990a). Komplexitätsanalylse verschiedener Algorithmen zur Wissensdiagnose. *Arbeitsbericht aus dem Projekt "Wissensstruktur" (Az. Lu 385/1–1) im Schwerpunktprogramm "Wissenspsychologie" der Deutschen Forschungsgemeinschaft.*

Unnewehr, J. (1990b). Testbericht: Simulation der Wissensdiagnose mit verschiedenen Algorithmen und Wissensräumen. *Arbeitsbericht aus dem Projekt "Wissensstruktur" (Az. Lu 385/1–1) im Schwerpunktprogramm "Wissenspsychologie" der Deutschen Forschungsgemeinschaft.*

Unnewehr, J. (1991). Software-Dokumentation: Eine Markov-Prozedur zur Wissensdiagnose nach Falmagne & Doignon (1988). *Arbeitsbericht aus dem Projekt "Wissensstruktur" (Az. Lu 385/1–1) im Schwerpunktprogramm "Wissenspsychologie" der Deutschen Forschungsgemeinschaft.*

van Leeuwe, J.F.J (1974). Item Tree Analysis. Nederlands Tijdschrift voor de Psychologie, **29**, 475–484

Weiss, D.-J. (1983). New Horizons in Testing. Latent Trait Test Theory and Computerized Adaptive Testing. New York: Academic Press.

Anhang A: Checkliste für Fehler

1. `internet error`

   Ursache: Der Internet-Port ist belegt, evtl. weil der Diagnoseprozeß bereits läuft.
   Abhilfe: Nachsehen, ob der Diagnoseprozeß bereits läuft. Wenn ja: warten, bis die Diagnose zu Ende ist. Wenn nein: Warten, bis der Port frei wird oder entspr. Kap. 5.4 einen anderen Port belegen.

2. `connecting error`

   Ursache: Fehler der Internet-Verbindung.
   Abhilfe: Der Diagnoseprozeß befindet sich im Zustand "Begin". Name der Protokoll-Datei senden und Diagnose wieder aufnehmen.

3. `receiving error`

   Ursache: Fehler der Internet-Verbindung.
   Abhilfe: Der Diagnoseprozeß befindet sich im Zustand "Begin". Name der Protokoll-Datei senden und Diagnose wieder aufnehmen.

4. `sending error`

   Ursache: Fehler der Internet-Verbindung.
   Abhilfe: Der Diagnoseprozeß befindet sich im Zustand "Begin". Name der Protokoll-Datei senden und Diagnose wieder aufnehmen.

5. `wrong answer: [answer]`

   Ursache: Falsche Nachricht vom Präsentationsprozeß.
   Abhilfe: Der Diagnoseprozeß befindet sich im Zustand "Begin". Name der Protokoll-Datei senden und Diagnose wieder aufnehmen. Aufgabennamen nur mit SOLVED, FAILED oder PAUSE beantworten.

6. `wrong filename: [filename]`

   Ursache: Der vom Präsentationsprozeß übermittelte Name der Protokolldatei entspricht nicht den in Kapitel 3.3.2 festgelegten Konventionen der Dateischnittstelle.
   Abhilfe: Richtigen Namen senden.

7. `no access: [filename]`

   Ursache: Die Wissensstrukturdatei oder die Protokolldatei kann nicht geöffnet werden, weil entweder der vom Präsentationsprozeß übermittelte Name der Protokolldatei einen falschen oder unvollständigen Pfad

enthält, oder weil die entsprechende Datei nicht existiert oder geschützt ist.
Abhilfe: Verzeichnis und Pfad entsprechend den Konventionen in Kapitel 3.3 anlegen und Datei mit richtigem Namen an der richtigen Stelle plazieren. Schreib- und Leseschutz entfernen. Namen der Protokolldatei mit vollständigem Pfad senden.

8. `wrong format:[filename]`

   Ursache: Der Inhalt der Datei hat ein falsches Format.
   Abhilfe: Inhalt wie in Kapitel 3.3 beschrieben ändern oder Datei neu anlegen.

9. `undefined set`

   Ursache: Ein errechneter Wissenszustand existiert nicht.
   Abhilfe: Protokolldatei sichern und löschen, neu beginnen.

10. `undefined element`

    Ursache: Eine Aufgabe eines errechneten Wissenszustands existiert nicht.
    Abhilfe: Protokolldatei sichern und löschen, neu beginnen.

11. `empty marker`

    Ursache: Für den unitären Algorithmus nach Doignon und Falmagne wurde eine Wissensstruktur verwandt, die nicht wohlgraduiert ist.
    Abhilfe: Likelihoodbasierte Diagnoseprozedur oder wohlgraduierte Wissensstruktur verwenden.

12. `too much questions`

    Ursache: Bei der Diagnose mußten zu viele Fragen gestellt werden (tritt im Normalfall nicht auf).
    Abhilfe: Protokolldatei sichern und löschen, neu beginnen.

13. `memory overflow`

    Ursache: Die Wissensstruktur paßt nicht in den freien Hauptspeicher.
    Abhilfe: Konkurrierende Prozesse beenden, kleinere Wissensstruktur oder Rechner mit mehr Hauptspeicher verwenden.

Anhang B: Hilfsprogramme

NAME

dist – find sets with minimal symmetric set difference

SYNOPSIS

dist [**–fair**] [**–h**] *filename1 filename2*

DESCRIPTION

dist computes the minimal symmetric set difference between a set in *filename1* and some set in *filename2*.

The files are coded with a format description **Format:** *a* **X** *b*, where *a* is the number of rows and *b* is the number of columns of a following matrix. The number of columns is equal in *filename1* and *filename2*. A row is a binary coded set with a '0' for an element not contained, a '1' for an element contained in the set and an 'x' for a don't care. A don't care results in not taking into account that element for calculating the symmetric set difference.

For coding the files an alternative format can be used, where the binary coded sets are compressed to 16 bit integers (PC) or 32 bit integers (SUN). In this integer format no don't cares are allowed. The files begin with a format description **Format:** *a* **X** *b* (*c*), where *a* is the number of rows, *b* is the number of integer columns and *c* is the number of bits used to code a set. *c* is equal in *filename1* and *filename2*.

To transform PC format (e.g. *.bga files) into boolean format use **int2bool**(l). To transform relations into one of the above described formats use **r2set**(l).

dist outputs the row number of a set in *filename1* followed by the binary code of this set, the minimal symmetric set difference between this set and a set in *filename2*, the row number and code of that set in *filename2* and the column numbers of the elements which made the difference.

OPTIONS

–fair Do not respect sets in *filename2*, which include the set in *filename1*.

–h Additionally output a frequency distribution of computed symmetric set differences.

DIAGNOSTICS

file access error

A file cannot be found, opened or accessed.

file format error

A file does not match the format conventions described above.

memory overflow error

The data set to be loaded from *filename1* or from *filename2* is too large.

SEE ALSO

int2bool(l) **r2set**(l)

NAME

int2bool – convert rows of integer into boolean

SYNOPSIS

int2bool

DESCRIPTION

int2bool converts rows of integer into rows of boolean. The rows of integer must be given from stdin headed with a format description **Format:** *a* **X** *b* (*c*), where *a* is the number of rows, *b* is the number of integer in a row and *c* is the number of boolean to gain from a row. Integer may have 16 bit PC format (e.g. *.bga files) or 32 bit sun format.

int2bool works with **r2set**(l) output. **int2bool** output can be used with **dist**(l).

EXAMPLES

int2bool < *filename*

DIAGNOSTICS

file format error

The input does not match the format conventions described above.

memory overflow error

The input is too large.

SEE ALSO

dist(l), **r2set**(l)

NAME

presproc – sample presentation tool for knowledge assessment

SYNOPSIS

presproc

DESCRIPTION

You have to start a knowledge assessment process first to use this program. For instruction see Unnewehr (1992). **presproc** displays a presentation window under sunview. Fill in hostname and internet port of the assessment process and the name of the assessment protocol file (consider Unnewehr (1992) for name conventions), then press button **BEGIN** to connect with the assessment process. Answer the questions by using the buttons **YES, NO** and **PAUSE.** Caution: presproc uses **tel**(l) for internet communication (make shure **tel**(l) is installed).

DIAGNOSTICS

internet error

No socket resource was available (for details see SunOs 4.0, *Network Programming).*

connecting error

The port is not served by a receiver.

unknown host: *hostname*

The host specified by the parameter *hostname* does not exist.

sending error

Error while sending the message via internet.

receiving error

Error while waiting for a message.

connection aborted

The peer went down or aborted the connection.

SEE ALSO

tel(l)

Unnewehr, J. (1992). *Benutzerhandbuch - Prozeduren zur Wissensdiagnose.* Bericht aus dem Psychologischen Institut der Universitaet Heidelberg, Bericht Nr. 74, Heidelberg: Universitaet Heidelberg.

NAME

r2set – convert a certain class of relations into sets

SYNOPSIS

r2set [**–boolean**]

DESCRIPTION

r2set converts a class of relations (see Lukas, 1990) into sets to be used for knowledge assessment. The relations must be given from stdin in form *x* -> *y*, where *x* and *y* may be numbers of questions for the knowledge assessment.

The result is coded with a format description **Format:** *a* **X** *b*, where *a* is the number of rows and *b* is the number of columns of a following matrix. A row is a binary coded set with a '0' for an element not contained and a '1' for an element contained in the set.

If the option **-boolean** is not given, an alternative format will be used, where the binary coded sets are compressed to 32 bit integers. The output begins with a format description **Format:** *a* **X** *b* (*c*), where *a* is the number of rows, *b* is the number of integer columns and *c* is the number of bits used to code a set.

To transform the integer format into boolean format use **int2bool**(l). The output of **r2set** can be used by **dist**(l).

Depending on the number of sets **r2set** needs a lot of time. To abort with saving the sets computed so far use **kill** *PID*. To abort with data loss use **kill -9** *PID*.

OPTIONS

–boolean

Code sets in boolean format instead of integer format.

DIAGNOSTICS

file access error

A file cannot be found, opened or accessed.

file format error

A file does not match the format conventions described above.

memory overflow error

The data set to be constructed is too large.

SEE ALSO

dist(l) **int2bool**(l)

Lukas, J. (1990). *Algorithmen zur Berechnung bestimmter Eigenschaften von Relationen und eine Anwendung auf die Untersuchung von Wissensstrukturen.* Arbeitsbericht aus dem Projekt "Wissensstruktur" (Az. Lu 385/1-1) im Schwerpunktprogramm "Wissenspsychologie" der Deutschen Forschungsgemeinschaft.

NAME

tel – send message and receive answer via internet

SYNOPSIS

tel [**–h** *hostname*] [**–p** *portnumber*] *message*

DESCRIPTION

tel sends the message *message* via internet port *portnumber* to host *hostname*. Default host is the same host, default port is port 1025. The received answer is written to stdout.

OPTIONS

–h Send to host *hostname*.

–p Send via port *portnumber*.

DIAGNOSTICS

internet error
No socket resource was available (for details see SunOs 4.0, *Network Programming).*

connecting error
The port is not served by a receiver.

unknown host: *hostname*
The host specified by the parameter *hostname* does not exist.

sending error
Error while sending the message via internet.

receiving error
Error while waiting for a message.

connection aborted
The peer went down or aborted the connection.

SEE ALSO

presproc(l)

Bisher erschienene Diskussionspapiere

Diskussionspapier Nr. 1: **GROEBEN, N.:** Vom behavioralen zum epistemologischen Subjektmodell: Paradigmawechsel in der Psychologie? September 1975

Diskussionspapier Nr. 2: **MÖBUS, C.& SIMONS, H.:** Zur Fairness psychologischer Intelligenztests gegenüber ethnischen und sozialen Gruppen: Kritik klassischer Konzepte. Oktober 1975

Diskussionspapier Nr. 3: **WOTTAWA, H.:** Skalenprobleme bei probabilistischen Meßmodellen. März 1976

Diskussionspapier Nr. 4: **Treiber, B. & Petermann, F.:** Zur Interaktion von Lernermerkmalen und Lehrmethoden: Rekonstruktion und Normierung des ATI-Forschungsprogramms. April 1976

Diskussionspapier Nr. 5: **MÖBUS, C. & WALLASCH, R.:** Zur Erfassung von Hirnschädigungen bei Kindern: Nichtlineare Entscheidungsregeln auf der Basis von Veränderungsmessungen. August 1976

Diskussionspapier Nr. 6: **SCHEELE, B. & GROEBEN, N.:** Voraussetzungs- und zielspezifische Anwendung von Konditionierungs- vs. kognitiven Lerntheorien in der klinischen Praxis. Dezember 1976

Diskussionspapier Nr. 7: **MÖBUS, C.:** Zur Analyse nichtsymmetrischer Ähnlichkeitsurteile: Ein dimensionales Driftmodell, eine Vergleichshypothese, TVERSKY's Kontrastmodell und seine Fokushypothese. Juni 1977

Diskussionspapier Nr. 8: **Simons, H. & Möbus, C.:** Veränderung von Berufschancen durch Intelligenztraining. Juli 1977

Diskussionspapier Nr. 9: **Braunmühl, C. v. & Grimm, H.:** Zur Kommunikationspsychologie: Über Versuche der methodischen Konstitution eines genuin humanwissenschaftlichen Forschungsansatzes zur Entwicklung der Verständigungsfähigkeit. November 1977

Diskussionspapier Nr. 10: **Hofer, M.:** Entwurf einer Heuristik für eine theoretisch geleitete Lehrer- und Erzieherbildung. November 1977

Diskussionspapier Nr. 11: **Scheibler, D. & Schneider, W.:** Probleme und Ergebnisse bei der Evaluation von Clusteranalyse-Verfahren. Juni 1978

Diskussionspapier Nr. 12: **Scheele, B.:** Kognitions- und sprachpsychologische Aspekte der Arzt-Patient-Kommunikation. September 1978

Diskussionspapier Nr. 13: **Treiber, B. & Schneider, W.:** Mehrebenenanalyse sozialstruktureller Bedingungen schulischen Lernens. Oktober 1978

Diskussionspapier Nr. 14: **Ahrens, H.-J. & Kordy, H.:** Möglichkeiten und Grenzen der theoretischen Aussagekraft von multidimensionalen Skalierungen bei der Untersuchung menschlicher Informationsverarbeitung. Teil I: Formale und wissenschaftstheoretische Grundlagen. März 1979

Diskussionspapier Nr. 15: **Groeben, N.:** Entwurf eines Utopieprinzips zur Generierung psychologischer Konstrukte. Juni 1979

Diskussionspapier Nr. 16: **Weinert, F.E. & Treiber, B.:** School Socialization and Cognitive Development. Juni 1979

Diskussionspapier Nr. 17: **Gundlach, H.:** Inventarium der älteren Experimentalapparate im Psychologischen Institut Heidelberg sowie einige historische Bemerkungen. 1978

Diskussionspapier Nr. 18: **Scheele, B. & Groeben, N.:** Zur Rekonstruktion von subjektiven Theorien mittlerer Reichweite. Eine Methodik-Kombination von halbstandardisiertem Interview (einschließlich Konfrontationstechnik) und Dialog-Konsens über die Theorie-Rekonstruktion mittels der Struktur-Lege-Technik (SLT). Dezember 1979

Diskussionspapier Nr. 19: **Gloger-Tippelt, G.:** Subjektive Theorien von Frauen über ihre erste Schwangerschaft: Theoretische Konzepte und methodische Möglichkeiten. Januar 1980

Diskussionspapier Nr. 20: **Kämmerer, A.:** Das Konzept 'psychotherapeutische Strategie' am Beispiel des Problemlösens. Juli 1980

Diskussionspapier Nr. 21: **Scheele, B.:** (unter Mitarbeit von B. Tuschen und C. Maier): Subjektive Theorien über Ironie - als Heuristik für einen wissenschaftlichen Hypothesenkorpus. August 1980

Diskussionspapier Nr. 22: **Treiber, B.:** Erklärung von Förderungseffekten in Schulklassen durch Merkmale subjektiver Unterrichtstheorien ihrer Lehrer. Oktober 1980

Diskussionspapier Nr. 23: **Röhrle, B. & Kommer, D.:** Handlungstheoretische Betrachtungen zur primären Prävention psychischer Störungen. Februar 1981

Diskussionspapier Nr. 24: **Voigt, F.:** Die Entwicklung des Zahlbegriffs. Teil I: Entwicklungslinien des Zahlbegriffs im Vorschulalter: Übersicht über theoretische Probleme und empirische Untersuchungen, mit einer Bibliographie zur Zahlbegriffsentwicklung. Teil II: Entwicklungslinien des Zahlbegriffs im Vorschulalter: Deskriptive Untersuchung des kindlichen Zahlverständnisses und verwandter Konzepte. April 1981. Teil III: Trainingsstudien zum Erwerb konkreter Operationen (unter besonderer Berücksichtigung von Modellen der Invarianzaufgabe). Teil IV: Die Trainierbarkeit ordinaler und kardinaler Konzepte und ihre Beziehung zum Zahlbegriff. Juli 1982

Diskussionspapier Nr. 25: **Schneider, G. & Weimer, E.:** Aspekte der Kategorisierung städtischer Umwelt - Eine empirische Untersuchung. Juni 1981

Diskussionspapier Nr. 26: **Schneider, W. & Scheibler, D.:** Zur Evaluation numerischer Klassifikation: Probleme beim Vergleich von Clusteranalysen. August 1981

Diskussionspapier Nr. 27: **Drinkmann, A. & Groeben, N.**: Techniken der Textorganisation zur Verbesserung des Lernens aus Texten: Ein metaanalytischer Überblick. November 1981

Diskussionspapier Nr. 28: **Graumann, C.F.**: Theorie und Geschichte. November 1982, Historische Reihe Nr. 1

Diskussionspapier Nr. 29: **Woodward, W.R.**: From the Science of Language to Völkerpsychologie: Lotze, Steinthal, Lazarus and Wundt. November 1982, Historische Reihe Nr. 2

Diskussionspapier Nr. 30: **Sommer, J.**: Dialogische Forschungsmethoden. Dezember 1982

Diskussionspapier Nr. 31: **Wintermantel, M. & Christmann, U.**: Textverarbeitung: Empirische Untersuchung zum Verstehen einer Personbeschreibung. Januar 1983

Diskussionspapier Nr. 32: **Schmalhofer, F.**: Text Processing with and without Prior Knowledge: Knowledge- versus Heuristic-Dependent Representations. Februar 1983

Diskussionspapier Nr. 33: **Métraux, A.**: Victor de l'Aveyron oder Zum Streit zwischen Kulturalisten und Biologisten am Anfang des 19. Jahrhunderts. Mai 1983, Historische Reihe Nr. 3

Diskussionspapier Nr. 34: **Graumann, C.F.**: Wundt - Bühler - Mead - Zur Sozialität und Sprachlichkeit menschlichen Handelns. Mai 1983, Historische Reihe Nr. 4

Diskussionspapier Nr. 35: **Gundlach, H.**: Folk Psychology and Social Psychology oder Das Los des Ausdrucks 'Völkerpsychologie' in den englischen Übersetzungen der Werke Wundts. Mai 1983, Historische Reihe Nr. 5

Diskussionspapier Nr. 36: **Woodward, W.R.**: Hermann Lotze's Concept of Function: Its Kantian Origin and its Impact on Evolutionism in the United States. Mai 1983, Historische Reihe Nr. 6

Diskussionspapier Nr. 37: **Schneider, G.**: Reflexivität als Grenzproblem einer kognitiven Psychologie. August 1983

Diskussionspapier Nr. 38: **Geuter, U.**: 'Gleichschaltung' von oben? Universitätspolitische Strategien und Verhaltensweisen in der Psychologie während des Nationalsozialismus. Oktober 1983, Historische Reihe Nr. 11

Diskussionspapier Nr. 39: **Kruse, L.**: Drehbücher für Verhaltensschauplätze oder: Scripts for Settings. Dezember 1983

Diskussionspapier Nr. 40: **Graumann, C.F.**: The individualisation of the social and the desocialisation of the individual - Floyd H. Allport's Contribution to Social Psychology -. Mai 1984, Historische Reihe Nr. 10

Diskussionspapier Nr. 41: **Kruse, L. & Graumann, C.F.**: Environmental Psychology in Germany. November 1984

Diskussionspapier Nr. 42: **Kany, W. & Schneider, G.**: Ein linguistisch fundiertes inhaltsanalytisches System zur Erfassung des referentiellen und prädikativen Gehalts verbaler Daten. Mai 1985

Diskussionspapier Nr. 43: **Hormuth, S.E.**: Methoden für psychologische Forschung im Feld: Erfahrungsstichprobe, Autophotographie und Telefoninterview. Februar 1985

Diskussionspapier Nr. 44: **Haeberle, E.J.**: Die Anfänge der Sexualwissenschaft in Berlin. April 1985, Historische Reihe Nr. 12

Diskussionspapier Nr. 45: **Schmalhofer, F. & Schäfer, I.**: Lautes Denken bei der Wahl zwischen benannt und beschrieben dargebotenen Alternativen. Juni 1985

Diskussionspapier Nr. 46: **Zielinski, W. & Rott, C.H.**: Analyse der Entwicklung des Wortleseprozesses bei erfolgreichen und schwachen Lesern der Grundschule. Februar 1986

Diskussionspapier Nr. 47: **Waller, M.**: Metasprachliche Entwicklung: Forschungsgegenstand, Schwerpunkte, Desiderate und Perspektiven der empirischen Forschung. Juli 1986

Diskussionspapier Nr. 48: **Gundlach, H.**: Inventarium der älteren Experimentalapparate im Psychologischen Institut Heidelberg sowie einige historische Bemerkungen (zweite, vermehrte Auflage). September 1986, Historische Reihe Nr. 9

Diskussionspapier Nr. 49: **Klüpfel, J. & Graumann, C.F.**: Ein Institut entsteht - Zur Geschichte der Institutionalisierung der Psychologie an der Universität Heidelberg -. Oktober 1986, Historische Reihe Nr. 13

Diskussionspapier Nr. 50: **Drinkmann, A.**: Private und öffentliche Self-Consciousness: Eine Zwischenbilanz ihrer empirischen Bewährung. Oktober 1986

Diskussionspapier Nr. 51: **Blickle, G. & Groeben, N.**: Gegen einen objektivistisch halbierten Kognitivismus: Kognitiv-konstruktives Sprachverstehen und nicht-paradoxe Wirkungen von Lob und Tadel. November 1986

Diskussionspapier Nr. 52: **Scheele, B. & Groeben, N.**: Eine Dialog-Konsens-Variante der Ziel-Mittel-Argumentation. Dezember 1986

Diskussionspapier Nr. 53: **Batz, W.-D., Bickes, C., Bickes, H., Busse, D. & Lörch, B.**: Konzeptuelle Strukturen in der Sprache des Vorurteils. Dezember 1986

Diskussionspapier Nr. 54: **Röhrle, B.**: Soziale Netzwerke und Unterstützung. Januar 1987

Diskussionspapier Nr. 55: **Sommer, J.**: Der Signifikanztest in der psychologischen Forschung. Ein Falsifikationsinstrument im Sinne des Kritischen Rationalismus? März 1987

Diskussionspapier Nr. 56: **Batz, W.-D.**: Kodierung und Repräsentation - über hypothetische Mechanismen in Gedächtnistheorien. Dezember 1987

Diskussionspapier Nr. 57: **Bastine, R.**: Psychotherapeutische Prozeßanalyse. September 1987

Diskussionspapier Nr. 58: **Amelang, M. & Krüger, C.**: Kindesmißhandlung. November 1989

Diskussionspapier Nr. 59: **Amelang, M.**: An Investigation of the Factorial Structure and External Validity of Social Intelligence. Dezember 1987

Diskussionspapier Nr. 60: **Bastine, R.**: Klinische Psychodiagnostik. März 1988

Diskussionspapier Nr. 61: **Waller, M.**: Die Entwicklung der Beurteilung fehlerhafter Äußerungen - Eine Pilotstudie. Juni 1988

Diskussionspapier Nr. 62: **Schahn, J. & Holzer, E.**: Untersuchungen zum individuellen Umweltbewußtsein. August 1989

Diskussionspapier Nr. 63: **Stössel, A. & Scheele, B.**: Nomothetikorientierte Zusammenfassung Subjektiver Theorien zu übergreifenden Modalstrukturen. Januar 1990

Diskussionspapier Nr. 64: **Aschenbrenner, K.M., Laier, R. & Albert, D.**: Wichtigkeit als Wissen über die Variation der Merkmalsattraktivität bei der Verhaltenswahl. Dezember 1989

Diskussionspapier Nr. 65: **Albert, D., Gertzen, H., Bürgy, R., Bannert, M. & Schneyer, Th.**: Abruf semantisch strukturierter Informationen beim binären Wählen zwischen beschriebenen Alternativen. Dezember 1989

Diskussionspapier Nr. 66: **Albert, D., Lages, M., Gertzen, H. & Aschenbrenner, K.M.**: Beeinflussen Struktureigenschaften von Wissen das Wahlverhalten? Dezember 1989

Diskussionspapier Nr. 67: **Gertzen, H., Bettinger, C., Körner, Chr. & Albert, D.**: Bewertende Vergleiche und Informationsabruf in Abhängigkeit von beurteilter Dimensionswichtigkeit bei unvollständig beschriebenen Alternativen. Dezember 1989

Diskussionspapier Nr. 68: **Kane, G., Rotter, B. & Waller, M.**: Konstruktion und Erprobung einer Entwicklungsskala zur Erfassung vorsprachlich-gestischer Äußerungen bei geistig behinderten Kindern. Ergebnisse einer Pilotstudie. Januar 1991

Diskussionspapier Nr. 69: **Krüger, C. & Amelang, M.**: Arbeitslosigkeit und Kriminalität. Mai 1991

Diskussionspapier Nr. 70: **Groeben, N. & Erb, E.**: Reduktiv-implikative versus elabrativ-prospektive Menschenbildannahmen in psychologischen Forschungsprogrammen. Dezember 1991

Diskussionspapier Nr. 71: **Albert, D., Schrepp, M., Held, Th.**: Construction of Knowledge Spaces for Problem Solving in Chess - Two Experimental Investigations. März 1992

Diskussionspapier Nr. 72: **Kany, W., Waller, M.**: Desiderate einer entwicklungspsychologischen Theorie des Spracherwerbs: Eine Positionsbestimmung gegenüber der nativistischen Auffassung Chomskys. Februar 1992

Diskussionspapier Nr. 73: **Kadijk, M.**: Plotting Activations in Neural Networks. Oktober 1992

Diskussionspapier Nr. 74: **Unnewehr, J.**: Benutzerhandbuch Prozeduren zur Wissensdiagnose. Dezember 1992

Diskussionspapier Nr. 75: **Erb, Egon:** Die Kontraststruktur menschlichen Denkens zwischen Dogmatismus als kurzschlüssiger Polarisierung und polarer Integration als Entwicklungsziel. Dezember 1992