Zeitschrift für Pädagogische Psychologie, 4, 1990, Heft 3, 177-186

FRANZ SCHMALHOFER, STEFAN BOSCHERT & OTTO KÜHN

Der Aufbau allgemeinen Situationswissens aus Text und Beispielen

The Construction of a Situation Model from Text and Examples

Summary: It is investigated how elementary knowledge of a programming language (LISP) is acquired from a text in comparison to learning from examples. By a cognitive model it was specified how knowledge is acquired and transformed when a learner studies a text or examples. In learning from text, a propositional textbase is initially constructed. In learning from examples, at first a situational form representation (template) is constructed. From either one or both of these more peripheral knowledge representations a more general situation model of the subject domain can be built. Via such a situation model a template may be generated from text propositions and vice versa. However, such transformation processes may be error-prone and cognitively very demanding. With this cognitive model receptive and discovery learning processes can be investigated more fruitfully than without such a theoretical underpinning.
In an experiment with three conditions, subjects either learned from a text, from informationally equivalent examples, or from the combination of these two materials. As predicted, there were only differences in the peripheral knowledge representations but no differences in the situation models among the three conditions.

Zusammenfassung: Der Erwerb von elementaren Kenntnissen einer Programmiersprache (LISP) aus einem Text wurde im Vergleich zu Lernen aus Beispielen untersucht. Für die beiden Lernmaterialien wurden entsprechende Wissenserwerbs- und Transformationsprozesse in einem Modell beschrieben. Beim Textstudium wird zunächst eine propositionale Textbasis aufgebaut. Beim Studieren von Beispielen wird dagegen eine situative Formrepräsentation (Schablone) erstellt. Aus jeder dieser zuerst erstellten Wissensrepräsentationen kann dann ein allgemeines Situationsmodell erzeugt werden. Durch Transformationsprozesse können somit Textpropositionen und Schablonen über ein allgemeines Situationsmodell ineinander überführt werden. Wie ein Experiment zeigte, kann beim Studium von Text, von dazu informationsäquivalenten Beispielen bzw. der Kombination der beiden Materialien zwar ein gleichwertiger Wissensstand erworben werden, jedoch ist dieses Wissen - wie vom Modell vorhergesagt - zu einem unterschiedlichen Ausmaß in Textpropositionen, Schablonen und allgemeinen Situationsdarstellungen gespeichert.

1. Einleitung

In den 60er Jahren fand in der Pädagogischen Psychologie eine ausführliche Debatte darüber statt, ob rezeptives Lernen (Ausubel 1964) oder Entdeckungslernen (Bruner 1961) zu einem günstigeren Lernergebnis führt. Daraus entstand eine umfangreiche Forschungsliteratur. Die Ausgangsfrage konnte bisher jedoch noch nicht zufriedenstellend beantwortet werden (Lefrancois 1976), da in Abhängigkeit von der gewählten Operationalisierung die eine oder andere Lernform zu einem besseren Lernergebnis führte (Neber 1981).
Die Hauptschwierigkeit eines sinnvollen Vergleichs der beiden Lernformen besteht offensichtlich darin, theoretisch fundierte Operationalisierungen und Materialien zu finden. Auch wenn man - wie in der vorliegenden Arbeit - davon ausgeht, daß man rezeptives Lernen als Lernen aus Text und Entdeckungslernen als Lernen aus Beispielen operationalisieren kann, besteht immer noch das grundsätzliche Problem, daß Text und Beispiele wegen ihrer Verschiedenartigkeit zunächst eigentlich gar nicht vergleichbar sind.
Dieses grundsätzliche Problem kann möglicherweise jedoch dadurch gelöst werden, daß für jedes Lernmaterial eine präzise Beschreibung der jeweiligen Informationsverarbeitungsprozesse angegeben wird und die von ihnen erzeugten Wissensrepräsentationen miteinander verglichen werden. Dies kann durch eine *Kognitive Modellierung* (Schmalhofer & Wetter 1988) geleistet werden, in der das Zusammenwirken von Lernmaterialien, Informationsverarbeitungsprozessen und Wissensstrukturen in ihrer zeitlichen Abfolge spezifiziert wird.

Im folgenden wird deshalb ein *Kognitives Modell* vorgestellt, das die Informationsverarbeitungsprozesse beim Lernen aus Text und aus Beispielen beschreibt. Das Modell impliziert, daß aufgrund interner Umsetzungsprozesse durch das Studium von Text und entsprechend ausgewählten Beispielen ein gleichwertiger Wissensstand (Newell 1982) erreicht werden kann, wobei sich jedoch Unterschiede in der Repräsentation des abgespeicherten Wissens zeigen sollten. Zur Überprüfung dieser Vorhersagen wurde ein Experiment durchgeführt. Als Gegenstandsbereich der Untersuchung wurde der Erwerb von Kenntnissen der Programmiersprache LISP gewählt.

2. Lernen aus Text und Beispielen

Wissen über einen Gegenstandsbereich wie z.B. eine Programmiersprache kann aus einem Text (textbasiert) oder auch aus konkreten Beispielsituationen (situationsbasiert) erworben werden. Der prinzipielle Unterschied zwischen text- und situationsbasiertem Wissenserwerb liegt in den vorgegebenen Informationen: Während im Text meist allgemeine Aussagen gemacht werden, die ein Lernender rezipieren soll, stellen Beispiele konkrete Sachverhalte dar, mit denen das allgemein Gültige entdeckt werden kann. Beide Materialien können so implizit die gleichen Informationen enthalten, da das allgemein Gültige sowohl aus Text als auch aus Beispielen erworben werden kann. Außerdem können aus den allgemeinen Aussagen des Textes konkrete Beispiele abgeleitet und umgekehrt aus den Beispielen allgemeine Textaussagen erzeugt werden.
Mit einem Kognitiven Modell kann eine theoretische Fundierung für die Konstruktion von informationsäquivalenten Lernmaterialien (Larkin & Simon 1987) geliefert werden. Zwei verschiedenartige Lernmaterialien können dann als informationsäquivalent betrachtet werden, wenn durch die im Kognitiven Modell postulierten Verarbeitungsprozesse das gleiche allgemeingültige Situationswissen erworben wird. Falls das Kognitive Modell die menschliche Informationsverarbeitung in einer Untersuchungssituation korrekt beschreibt, würden Personen aus informationsäquivalenten Materialien tatsächlich auch das gleiche allgemeingültige Wissen erwerben.
Zusätzlich zum allgemein gültigen Situationswissen kann es jeweils noch eine Repräsentation des Textes bzw. der Beispiele geben, so daß manche Inhalte multipel repräsentiert sein können. Eine multiple Repräsentation bringt beim Problemlösen sowohl beim Menschen als auch für künstliche Systeme Vorteile mit sich. Da jede Repräsentation ein bestimmtes Wissen explizit darstellt, während sie anderes nur implizit beinhaltet, erleichtert jede Repräsentation gerade die Ausführung derjenigen Aufgaben, welche das explizit kodierte Wissen erfordern. Andere Aufgaben werden dagegen von entsprechend anderen Repräsentationen unterstützt. So werden beispielsweise Aufgaben, deren Lösung auf räumlichen Überlegungen basiert, mit einer analogen Repräsentation leichter gelöst, während verbale Beschreibungen bei Vorliegen einer propositionalen Repräsentation leichter erzeugt werden können.
Durch multiple Wissensrepräsentationen kann sich ein System somit durch einen größeren Speicheraufwand in bezug auf die Lösung bestimmter Aufgaben Berechnungsaufwand ersparen, wenn Teilergebnisse bereits im Gedächtnis vorhanden sind. Welche Teilergebnisse nun vorberechnet werden, sollte dabei von der *Zielsetzung* beim Wissenserwerb abhängen. Diese Hypothese wird durch mehrere Experimente von Schmalhofer & Glavanov (1986) gestützt, in denen sich zeigte, daß in Abhängigkeit von der Zielsetzung des Lernenden sich Lesezeiten systematisch ändern und die betreffende Repräsentation entsprechend stärker ausgebildet wird.

3. S-T-Struktur Modell

3.1. Grundannahmen des Modells

a) *Multiple Wissensrepräsentationen:* Es werden drei Wissensrepräsentationen postuliert, die im folgenden als 1) allgemeines Situationswissen oder Situationsmodell, 2) S-Struktur und 3) T-Struktur bezeichnet werden. Das allgemeine Situationswissen oder Situationsmodell (van Dijk & Kintsch 1983) repräsentiert die strukturelle Basis des Gegenstandsbereiches.

Dagegen beinhaltet die S-Struktur Eigenschaften von Situationsbeispielen, die durch Schablonen (Anderson, Farrell & Sauers 1984) dargestellt werden. Die T-Struktur oder Textbasis (Kintsch 1974) ist aus Propositionen aufgebaut, die in Texten zum Ausdruck kommen können.

b) *Text- und situationsbasierter Wissenserwerb:* Der Erwerb von allgemeinem Situationswissen kann aus natürlichsprachlichen Beschreibungen und/oder aus beispielhaften Einzelsituationen, also text- und/oder situationsbasiert erfolgen. In Abhängigkeit vom Lernmaterial entstehen dabei zuerst text- und situationsbasierte Wissensrepräsentationen, d.h. eine propositionale Textbasis oder Schablonen (T- bzw. S-Struktur). Beim Lernen aus Text und Lernen aus Beispielen werden also verschiedene periphere Wissensrepräsentationen (T- bzw. S-Struktur) aufgebaut.

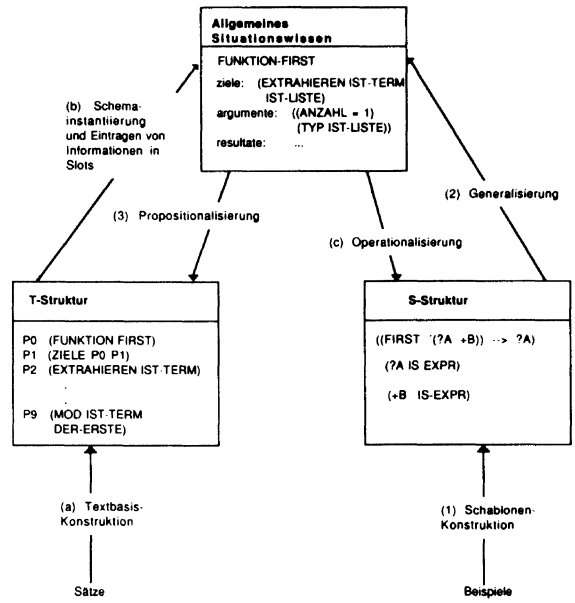

Abbildung 1: S-T-Struktur Modell

c) *Gleiches allgemeines Situationswissen bei informationsäquivalenten Lernmaterialien:* Da sowohl die T- als auch die S-Struktur weiter verarbeitet werden, kann aus text- und situationsbasiertem Lernmaterial das gleiche allgemeine Situationswissen[1] erworben werden. Dazu müssen die beiden Lernmaterialien jedoch informationsäquivalent sein.

d) *Abhängigkeit von Vorwissen und Zielsetzung:* Zur Wissensenkodierung wird stets das für den jeweiligen Bereich spezifische Vorwissen herangezogen. Beim Wissenserwerb können in Abhängigkeit von der jeweiligen Zielsetzung durch Umsetzungsprozesse zwischen S- und T-Struktur implizite Informationen des Materials in explizites Wissen umgewandelt werden.

3.2. Wissenserwerb

In Abbildung 1 wird der Wissenserwerb aus Text und aus Beispielen im S-T-Struktur Modell skizziert. Das Lernen aus Text besteht aus drei Teilschritten: zunächst erfolgt der Aufbau einer Textbasis (a) und daran anschließend der Aufbau eines allgemeinen Situationsmodells (b). Dieser Aufbau besteht im wesentlichen darin, daß Textpropositionen in das Vorwissen des Lernenden, das durch Schemata dargestellt wird, integriert werden (Kintsch & Greeno 1985). In einem dritten Schritt (c) kann daraus nun eine Schablone konstruiert werden, welche die allgemeinen strukturellen Beziehungen richtiger Beispiele darstellt. Durch das Einsetzen passender Konstanten in diese Schablonen könnten nun auch konkrete Beispiele erzeugt werden.

Beim Lernen aus Beispielen werden durch ähnlichkeitsbasierte oder erklärungsbasierte Generalisierungsprozesse (Mitchell, Keller & Kedar-Cabelli 1986) Schablonen konstruiert (1), welche die strukturellen Beziehungen richtiger Beispiele darstellen. Ähnlich zu dem schrittweisen Aufbau einer Textbasis (Kintsch & van Dijk 1978) kann dabei durch jedes neue Beispiel eine Veränderung der bereits konstruierten Schablone erfolgen. Die ähnlichkeitsbasierten Lernprozesse, die eine Schablone als Formrepräsentation konstruieren, wurden in Schmalhofer (1986a) beschrieben. Aus den Schablonen kann dann durch Heuristiken wie etwa dem «no function in identity»-Prinzip (Anderson & Thompson 1987) der Funktionalcharakter (Duncker 1935) der Schablone und damit allgemeines Situationswissen erschlossen werden (2). Das inferierte Situationswissen kann verbalisiert werden, was ein letzter Teilschritt beim Lernen aus Beispielen sein kann (3). Die spezielle Zielsetzung des Lernenden bestimmt, wel-

che dieser Verarbeitungsprozesse durchgeführt werden. Das skizzierte Modell wurde als Computersimulation implementiert (Schmalhofer, Kühn & Messamer 1989; Schmalhofer, Kühn, Messamer & Charron 1990), wodurch die Suffizienz der postulierten Wissenserwerbsprozesse nachgewiesen ist.

3.3. Wissensanwendung

Kenntnisse einer Programmiersprache können u.a. durch Satzverifikationsaufgaben, Beispielverifikationsaufgaben und durch Programmieraufgaben überprüft werden. Bei den Verifikationsaufgaben muß eine Person beurteilen, ob die vorgegebenen Sätze oder Beispiele richtig oder falsch sind. Bei Programmieraufgaben müssen Programme erstellt werden, welche eine Eingabe in eine bestimmte Ausgabe überführen. Bei der Wissensnutzung wird zuerst diejenige Repräsentation angesprochen, die der Aufgabenstellung am nächsten liegt.
Bei einer *Satzverifikationsaufgabe* wird ein vorgegebener Testsatz zuerst in Propositionen übersetzt. Diese Propositionen können dann mit den in der T-Struktur gespeicherten Propositionen verglichen werden. Falls dabei eine Übereinstimmung auftritt, hat der Testsatz einen Sinngehalt, der entweder im Lernmaterial vorkam, oder vom Lernenden während des Lesens inferiert wurde. Eine propositionale Übereinstimmung zwischen Testsatz und T-Struktur weist also darauf hin, daß es sich um einen richtigen Satz handelt. Darüber hinaus wird auch das allgemeine Situationswissen zur Verifikation eines Satzes herangezogen (Reder 1982; Schmalhofer 1986b). Dazu müssen die Propositionen des Testsatzes zuerst mit dem bis zu diesem Zeitpunkt aufgebauten Situationswissen in Beziehung gesetzt werden. Falls sich die Propositionen widerspruchsfrei einpassen lassen, ergibt sich Evidenz, daß es sich um einen richtigen Satz handelt. Falls dies nicht möglich ist, wird der Satz als falsch eingestuft. Eine detaillierte Untersuchung des Zusammenwirkens von propositionalen und situativen Gedächtnisspuren bei Satzverifikationsaufgaben wurde von Kintsch, Welsch, Schmalhofer und Zimny (1990) berichtet.
Bei *Beispielverifikationsaufgaben* wird vor allem die S-Struktur herangezogen. Für das zu verifizierende Beispiel wird zuerst überprüft, ob es in eine der gespeicherten Schablonen paßt. Falls eine solche Einpassung möglich ist, wird entschieden, daß es sich um ein korrektes Beispiel handelt. Andernfalls wird das allgemeine Situationswissen herangezogen, um die Korrektheit eines Beispiels zu überprüfen.
Programmieraufgaben werden normalerweise in Textform beschrieben. Bei der Lösung einer solchen Aufgabe wird daher zuerst eine Repräsentation in der T-Struktur ausgebildet. Dieses Wissen wird danach in allgemeines Situationswissen umgesetzt. Von hier aus wird die in der Programmieraufgabe angegebene Ein-/Ausgabespezifikation einer Schablone in der S-Struktur zugeordnet und dann in eine entsprechende Funktionseingabe umgesetzt. Falls eine solche Zuordnung nicht direkt möglich ist, kann durch Algorithmenwissen oder heuristische Problemlöseprozesse eine Dekomposition der Programmieraufgabe erfolgen (Waloszek, Weber & Wender 1986), so daß die einzelnen Teile jeweils einer Schablone zugeordnet werden können (Vorberg & Goebel 1990).

4. Experiment

In einem Experiment zum Erwerb elementarer Kenntnisse der Programmiersprache LISP wurde untersucht, ob Personen beim Lernen aus Text und dazu informationsäquivalenten Beispielen einen gleichwertigen Wissensstand erreichen. Zu diesem Zweck wurde auf der Grundlage des S-T-Struktur Modells zu einem Text informationsäquivalentes Beispielmaterial konstruiert. Versuchspersonen einer Untersuchungsbedingung erhielten in der Lernphase nur Text, Versuchspersonen der zweiten Bedingung nur Beispiele und Versuchspersonen einer dritten Bedingung (Kombinationsbedingung) sowohl Text- als auch Beispielmaterial. Das von den Versuchspersonen erworbene Wissen sollte durch Satzverifikations-, Beispielverifikations- und Programmieraufgaben geprüft werden.
Es wurde vorhergesagt, daß

a) die Versuchspersonen der Textbedingung eine stärker ausgebildete T-Struktur aufweisen als die Versuchspersonen der Beispielbedingung;

b) die Versuchspersonen der Beispielbedingung eine stärkere Schablonenrepräsentation haben als die Versuchspersonen der Textbedingung;
c) die Versuchspersonen der Text- und der Beispielbedingung sich im aufgebauten allgemeinen Situationswissen nicht unterscheiden;
d) die Versuchspersonen der Text- und der Beispielbedingung bei einer Aufgabe, die keiner der beiden peripheren Repräsentationen (S- und T-Struktur) näher liegt (Programmieraufgabe), auch ähnliche Ergebnisse erzielen würden;
e) wegen der Informationsäquivalenz der Materialien auch die Versuchspersonen der Kombinationsbedingung sich im allgemeinen Situationswissen von den beiden anderen Versuchsbedingungen nicht unterscheiden.

4.1. Methode

Die Untersuchung bestand für alle Versuchspersonen aus zwei Wissenserwerbsphasen mit jeweils anschließender Testphase. In der zweiten Wissenserwerbsphase gab es drei verschiedene Untersuchungsbedingungen.

Versuchspersonen: An der Untersuchung nahmen 80 Studenten der Physik, Mathematik, Chemie und Biologie der Universität Freiburg teil. Sie erhielten für ihre Teilnahme je 25 DM.

Instruktionsmaterial: Das Material der ersten Lernphase bestand aus 39 Sätzen über den allgemeinen Aufbau von LISP-Atomen, zusammengesetzten S-Termen und die Bedeutung von Funktionen in LISP. Das Instruktionsmaterial der zweiten Lernphase bestand für die drei Bedingungen aus 1) einem Text, 2) konkreten Beispielen und 3) einer Kombination der beiden Materialien.

Text: Der Text, der zufälligerweise ebenfalls aus 39 Sätzen bestand, war in 7 Abschnitte zu jeweils 4–11 Sätzen unterteilt. Der erste Abschnitt erklärte, wie LISP-Funktionen syntaktisch korrekt in das LISP-System eingegeben werden. Die vier darauffolgenden Abschnitte beschrieben die LISP-Funktionen LIST, FIRST, REST und EQUAL. Im sechsten Abschnitt wurde die Syntax der Funktionseingabe nochmals beschrieben, wobei besonders auf die Bedeutung des Hochkommas eingegangen wurde. Der letzte Abschnitt behandelte die Verknüpfungsmöglichkeiten der vier zuvor eingeführten Funktionen. Tabelle 1 zeigt in der oberen Hälfte einen der sieben Abschnitte des Textes.

Beispiele: In der Beispielbedingung gab es 67 Einträge, 11 Sätze, die dazu dienten, das Lernmaterial zu strukturieren und 56 Funktionsbeispiele. Dieses Lernmaterial war in 5 Abschnitte untergliedert. Die ersten vier Abschnitte zeigten jeweils die LISP-Funktionen LIST, FIRST, REST und EQUAL auf exemplarische Weise, indem entsprechende Funktionseingaben in das LISP-System und die durch das LISP-System erstellten Werte jeweils links und rechts in einer Zeile angesehen werden konnten. Jeder dieser vier gleichartig aufgebauten Abschnitte bestand aus zwei einleitenden Sätzen und 12 Eingabebeispielen in das LISP-System, von denen die ersten sieben Eingaben syntaktisch richtig aufgebaut waren. Die restlichen fünf Eingaben enthielten Fehler. Die korrekten Beispiele bestanden aus den Eingaben in das LISP-System und den daraus resultierenden Werten der Eingaben. Für alle falsch formulierten Eingaben wurde «ERROR» zurückgegeben. Der

Tabelle 1: Ausschnitte aus den Lernmaterialien der Text- und Beispielgruppe

Definition von FIRST:
Die Funktion FIRST wird verwendet, um den ersten S-Term aus einem zusammengesetzten S-Term zu extrahieren.
Die Funktion FIRST hat genau ein Argument.
Das Argument der Funktion FIRST muß ein zusammengesetzter S-Term sein. Der Wert der Funktion FIRST ist der erste S-Term des Arguments.

Beispiele für FIRST:
Anhand der folgenden Beispiele können Sie Kenntnisse der Funktion FIRST erwerben. Beachten Sie die Anführungszeichen!
Mögliche Eingaben und die dazugehörigen Ausgaben des LISP-Systems werden jeweils links und rechts in einer Zeile angegeben.

(FIRST '(A B))	→	A
(FIRST '((A B) C))	→	(A B)
(FIRST '(A (B C)))	→	A
(FIRST '((A B) (C D)))	→	(A B)
(FIRST '(A))	→	A
(FIRST (FIRST '((A B) C)))	→	A
(FIRST '(FIRST ((A B) C)))	→	FIRST
(FIRST 'A 'B)	→	ERROR
FIRST '(A B)	→	ERROR
(FIRST (A B))	→	ERROR
(FIRST 'A)	→	ERROR
(FIRST (A 'B))	→	ERROR

fünfte Abschnitt enthielt in gleicher Weise acht korrekte Beispiele für die Verknüpfung von LISP-Funktionen. Tabelle 1 zeigt in der unteren Hälfte einen typischen Abschnitt dieses Materials.

Testmaterial: In der Prüfphase wurden Satzverifikationsaufgaben, Beispielverifikationsaufgaben und Programmieraufgaben vorgegeben.

Satzverifikationsaufgaben: Um zwischen verschiedenen Wissensrepräsentationen unterscheiden zu können, wurden für jede Funktion, die in der zweiten Wissenserwerbsphase vorgestellt wurde, vier Testsätze verwendet: jeweils ein Originalsatz (O-Satz), ein paraphrasierter Satz (P-Satz), ein bedeutungsveränderter Satz (B-Satz) und ein falscher Satz (F-Satz). Während O-Sätze wörtlich im Text vorkamen, waren P-Sätze nur sinngemäß im Text enthalten. B-Sätze waren Sätze, die aus dem Text inferiert werden konnten. F-Sätze stellten eine falsche Behauptung über LISP auf.

O-Sätze und P-Sätze unterscheiden sich weder in ihrer propositionalen Darstellung noch in ihrer Richtigkeit, sondern nur auf der wörtlichen Ebene. Die Unterschiede, die bei ihrer Beantwortung auftreten, können zur Bestimmung der Stärke der wörtlichen Satzrepräsentation verwendet werden. Die wörtliche Satzrepräsentation macht einen bestimmten Anteil der T-Struktur aus. Analog kann der Unterschied zwischen P-Sätzen und B-Sätzen als Indikator für die Ausprägung der propositionalen Textrepräsentation (ebenfalls Teil der T-Struktur) dienen, da diese beiden Satzarten zum einen richtig sind und zum anderen nicht wörtlich im Text vorkommen und somit nur in der propositionalen Darstellung verschieden sind. B-Sätze und F-Sätze unterscheiden sich nur hinsichtlich ihrer Richtigkeit und können daher zur Bestimmung der Stärke des allgemeinen Situationswissens herangezogen werden. Eine detaillierte Beschreibung dieser Kontraste zur Bestimmung verschiedener Gedächtnisspuren wurde bereits in Schmalhofer (1986b) gegeben.

Beispielverifikationsaufgaben: Es wurden 20 richtige und 20 falsche Beispiele vorgegeben, zu deren korrekter Verifikation Kenntnisse verschiedener Aspekte der Programmiersprache LISP notwendig waren. Es handelte sich um Beispiele, die das Verständnis der prinzipiellen Beziehung zwischen Ein- und Ausgabe einer LISP-Funktion, das Verständnis von Hochkommas, Listenstrukturen und der Kombination von Funktionen überprüften. Tabelle 2 zeigt die exemplarischen Testsätze und Beispiele.

Programmieraufgaben: Während die Personen bei den Verifikationsaufgaben nur entscheiden

Tabelle 2: Ausschnitte aus den Testmaterialien für Satz- und Beispielverifikation

O-Satz:
Das Argument der Funktion FIRST muß ein zusammengesetzter S-Term sein.

P-Satz:
Um den ersten S-Term aus einem zusammengesetzten S-Term zu extrahieren, wird die Funktion FIRST verwendet.

B-Satz:
Das Argument der Funktion FIRST kann fünf oder auch mehr LISP-Atome enthalten.

F-Satz:
Der Wert der Funktion FIRST ist stets ein zusammengesetzter S-Term.

Richtige Beispiele:

ER (FIRST '(WEIN WEIB GESANG))	→	WEIN
QR (FIRST '(FIRST((OBERS(DORF)))))	→	FIRST
LR (FIRST (((AUA WEH)HUH)OH))	→	((AUA WEH)HUH)
GR (FIRST (FIRST (FIRST '(((A B)C)D))))	→	A
VR (EQUAL (LIST (REST '(A B C)) 'D) '((B C)D))	→	T

Falsche Beispiele:

EF (FIRST '(GAR TEN SCHLAUCH))	→	(TEN SCHLAUCH)
QF (FIRST '(FIRST ((TUTU LULU) DIDI)))	→	TUTU
LF (FIRST '(((ROT BLAU)TOT)OH))	→	((ROT BLAU))
GF (FIRST '(FIRST (FIRST '(((X Y)Z)A))))	→	X
VF (EQUAL (LIST '(REST '(X Y Z)) 'B) '((Y Z)B))	→	T

mußten, ob eine Aussage oder eine Ein-/Ausgaberelation hinsichtlich der erworbenen LISP-Kenntnisse als richtig oder falsch einzuschätzen war, mußten sie bei den Programmieraufgaben selbst eine Eingabe in das LISP-System erzeugen, bei der sie einige der gelernten Funktionen miteinander kombinieren mußten.

Geräte: Der Versuchsablauf wurde durch einen Personal Computer AT kontrolliert. Das Versuchssteuerungsprogramm wurde unter Einbindung einiger Assembler-Routinen in Turbo-Pascal programmiert (vgl. Schmalhofer, Schlei & Farin 1986).

Durchführung: Die Zuweisung der Versuchspersonen zu den Bedingungen erfolgte zufällig (Textbedingung: 27 Vpn; Beispielbedingung: 27 Vpn; Kombinationsbedingung: 26 Vpn). Die Versuchspersonen wurden über den Zweck und den Ablauf der Untersuchung unterrichtet und wurden instruiert, das Lernmaterial in ähnlicher Weise wie ein Lehrbuch bei einer Klausurvorbereitung zu studieren. In einer Übungsphase konnten sich die Versuchspersonen mit der Bedienung der Tastatur und der Darbietungsweise auf dem Bildschirm vertraut machen. Im ersten Lern- und Prüfungsabschnitt des eigentlichen Experiments gab es keine Unterschiede zwischen den Bedingungen; alle Versuchspersonen studierten zunächst 15 Minuten lang Instruktionsmaterial über Datenrepräsentationen und wurden dann mit 20 Testsätzen geprüft. Im zweiten Lernabschnitt studierten dann die Versuchspersonen der ersten Bedingung Text, die Versuchspersonen der zweiten Bedingung Beispiele und die Versuchspersonen der dritten Bedingung sowohl Text als auch Beispiele. Die Gesamtdauer dieser Lernphase betrug für alle Bedingungen jeweils 30 Minuten. Die Lernmaterialien wurden in Bildschirmfenstern derart dargeboten, daß sich die Versuchspersonen mit Hilfe der Tastatur frei innerhalb des Textes bzw. der Beispiele bewegen konnten. Sie konnten zu einem Zeitpunkt immer nur ein Item (d.h. einen Satz oder ein Beispiel) ansehen, waren jedoch in der Auswahl und der Bearbeitungszeit der Items nicht eingeschränkt.

Die zweite Testphase, die für die Versuchspersonen der drei Bedingungen wiederum identisch war, begann mit 16 Satzverifikationsaufgaben, die in zufälliger Reihenfolge am Bildschirm dargeboten wurden. Jeder Testsatz wurde von der Versuchsperson per Tastendruck angefordert. Neben anderen Aufgabenanforderungen, auf die hier nicht eingegangen werden soll (vgl. Schmalhofer, Boschert & Kühn 1987), mußte die Versuchsperson nach 5,25 Sekunden durch Drücken der entsprechenden Taste angeben, ob sie den vorgegebenen Satz als «richtig» oder «falsch» beurteilte. Die Verifikation der Beispiele wurde danach in gleicher Weise durchgeführt, jedoch betrug das Antwortintervall hier 14 Sekunden. Zum Schluß wurden noch zwei Programmieraufgaben vorgegeben, die die Versuchspersonen mit Papier und Bleistift bearbeiteten.

4.2. Ergebnisse

Eine ANOVA über den Anteil richtiger Antworten der ersten Testphase mit dem Faktor «Bedingung» ergab erwartungsgemäß keine Unterschiede ($F < 1$). Zu Beginn der zweiten Wissenserwerbsphase bestand somit nachweislich kein Wissensunterschied zwischen den drei Bedingungen.

Satzverifikationsaufgaben: Zur Auswertung der Satzverifikationsaufgaben der zweiten Testphase wurde für jede Versuchsperson und Satzart (O-, P-, B- und F-Sätze) die relative Häufigkeit der «ja»-Antworten erstellt. Der Beitrag der wörtlichen und propositionalen Textrepräsentation (Gedächtnisspur) und des allgemeinen Situationswissens (situative Gedächtnisspur) wurde entsprechend der Theorie der Signalentdeckung (Baird & Noma 1978) durch die Berechnung von d'-Werten geschätzt. Für jede Versuchsperson wurde aus den relativen Häufigkeiten der «ja»-Antworten für die vier Satzarten der entsprechende z-Wert bestimmt. Um auch für relative Häufigkeiten von 0.00 und 1.00 z-Werte zu erhalten, wurden die Werte 0.00 und 1.00 durch 1/9 bzw. 8/9 ersetzt. Die Differenzen der z-Werte der entsprechenden Satzpaare ergaben dann die d'-Werte. Für jede Versuchsperson wurde somit je ein d'-Wert für die drei Gedächtnisrepräsentationen bestimmt. Die daraus bestimmten mittleren Repräsentationsstärken von wörtlicher, propositionaler und situativer Information sind in Abbildung 2 dargestellt.

Eine 3×3 ANOVA dieser Daten mit den Faktoren Gedächtnisrepräsentation und Lernbedin-

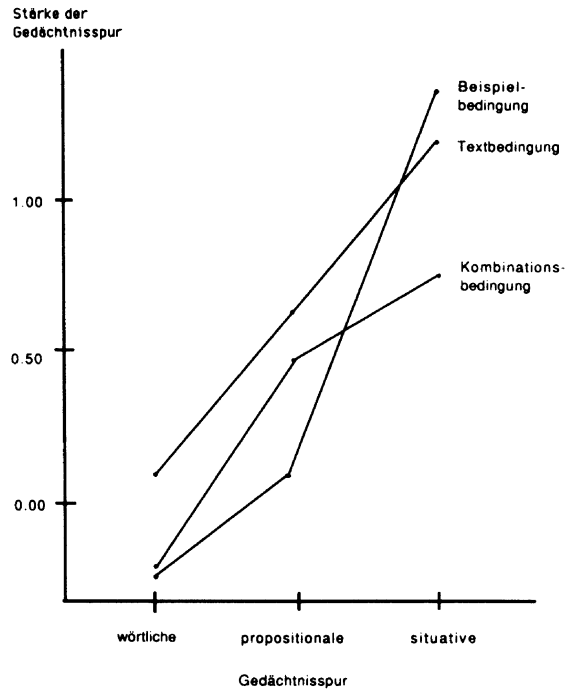

Abbildung 2: Mittlere Repräsentationsstärken von wörtlicher, propositionaler und situativer Information für die drei Bedingungen.

gung ergab Unterschiede bezüglich der Gedächtnisrepräsentation ($F(1,136)=32.52$; $MSE=0.67$; $p<0.01$)[2] und der Lernbedingung ($F(2.77)=3.76$; $MSE=0.22$; $p<0.05$). Die Interaktion war statistisch nicht signifikant ($F(3,136)=1.94$; $MSE=0.67$)[2]. Die Unterschiede in den Gedächtnisrepräsentationen lassen sich daraus erklären, daß die wörtliche Repräsentation praktisch nicht vorhanden, die propositionale nur schwach und die situative am stärksten ausgebildet war.

Eine einfaktorielle Varianzanalyse über die d'-Werte für die wörtliche Repräsentation mit dem Faktor «Bedingung» zeigte keine Unterschiede ($F<1$). Für die propositionale Repräsentation deutete sich jedoch ein Bedingungsunterschied an ($F(2,77)=2.85$; $MSE=0.47$; $p<0.10$). Darüber hinaus zeigte ein Mittelwertsvergleich zwischen der Text- und der Beispielbedingung einen signifikanten Unterschied ($t(50.7)=2.14$; $p<0.05$)[3].

Hieraus läßt sich schließen, daß die Versuchspersonen der Textbedingung wie vorhergesagt stärker propositional repräsentieren als die Versuchspersonen der Beispielbedingung. Zum Schluß wurde eine einfaktorielle Varianzanalyse über die d'-Werte für die situative Repräsentation mit dem Faktor «Lernbedingung» gerechnet, wobei weder die Hauptanalyse ($F(2.77)=2.23$; $MSE=0.82$) noch der Mittelwertsvergleich zwischen Text- und Beispielbedingung ($t(45.8)=-0.63$)[3] signifikant waren. Dieses Ergebnis zeigt, daß sich die Bedingungen, wie vorhergesagt, im aufgebauten allgemeinen Situationswissen nicht unterscheiden. Dies gilt ebenfalls für die Kombinationsgruppe, deren Wert zwar niedriger ist als die Werte der beiden anderen Bedingungen, sich jedoch statistisch nicht signifikant unterscheidet.

Beispielverifikationsaufgaben: Zur Auswertung der Beispielverifikationsaufgaben wurde für jede Versuchsperson die relative Häufigkeit der richtigen Antworten erstellt. Die Mittelwerte der Text-, Beispiel- und Kombinationsbedingung betrugen 0.61, 0.68, 0.65. Bei einer einfaktoriellen ANOVA mit dem Faktor «Bedingung» deutete sich ein Bedingungsunterschied an ($F(2,77)=2.70$; $MSE=0.01$; $p<0.10$) und der geplante Kontrast von Text- und Beispielbedingung ($t(51.8)=-2.57$; $p<0.05$)[3] zeigte, daß die Personen der Beispielbedingung eine stärkere Schablonenrepräsentation haben als die Versuchspersonen der Textbedingung.

Programmieraufgaben: Bei der Lösung der Programmieraufgaben ist den drei Lernbedingungen gemeinsam, daß nur sehr wenige Aufgaben völlig korrekt gelöst wurden. In der Text-, der Beispiel- und der Kombinationsbedingung wurden nur 12%, 13%, bzw. 9% der Aufgaben sowohl syntaktisch als auch semantisch richtig gelöst[4].

Um eine detailliertere Beurteilung der von den Versuchspersonen erzeugten Lösungen durchführen zu können, wurden für jede Lösung vier verschiedene Fehlerarten ausgezählt: Fehler bei Funktionsnamen, Fehler bei Argumenten, Weglassen von Klammern und Fehler durch Hochkommas und andere inkorrekte Zusätze (z.B. Punkte, Kommata, etc.). Die Resultate sind in Tabelle 3 dargestellt.

Eine 3×4 ANOVA mit den Faktoren Lernbedingung und Fehlerart zeigte keinen Unterschied zwischen den Lernbedingungen ($F<1$), wohingegen zwischen den verschiedenen Fehlertypen Unterschiede bestanden ($F(2,178)=8.63$; $MSE=0.05$; $p<0.01$)[2]. Die Interaktion war nicht signifikant ($F(4,178)=2.25$; $MSE=$

Tabelle 3: Relative Häufigkeit der vier Fehlerarten bei der Programmieraufgabe für die drei Lernbedingungen

	Lernbedingung		
	Text	Beispiele	Kombination
Fehlerart			
Funktionsname	.17	.15	.14
Argumentspezifikation	.11	.10	.08
Klammerstruktur	.36	.17	.23
Falsche Zusätze	.15	.19	.21

0.05; $p<0.10$)[2]. Die häufigsten Fehler waren hierbei Klammerfehler und inkorrekte Zusätze, die meisten aus zusätzlichen Hochkomma-Zeichen bestanden, während Funktionsnamen und Argumente meist richtig eingesetzt wurden. Dieses Ergebnis zeigt, daß die Versuchspersonen der Text- und Beispielbedingung wie vorhergesagt bei einer Aufgabe, die weder der propositionalen Text- noch der Schablonenrepräsentation näher liegt, annähernd gleiche Ergebnisse erzielten.

5. Diskussion

In dieser Arbeit wurde versucht, theoretisch fundierte Operationalisierungen für verschiedene Lernformen und Materialien zu erstellen, um so die relative Effektivität der einzelnen Lernformen besser untersuchen zu können. Dabei gilt es, das Problem zu lösen, wie trotz der Verschiedenartigkeit von Text und Beispielen ein sinnvoller Vergleich der Effektivität der betreffenden Lernformen überhaupt möglich ist. Ein Vergleich des Lernerfolgs erscheint nur dann sinnvoll, wenn die in den Lernmaterialien enthaltenen Informationen in einem bestimmten Sinn als äquivalent angesehen werden können. Es wurde definiert, daß Lernmaterialien dann informationsäquivalent sind, wenn daraus das gleiche allgemein gültige Wissen erworben werden kann. Diese Definition wurde durch die Erstellung eines Kognitiven Modells expliziert, welches die menschlichen Informationsverarbeitungsprozesse beschreibt. Wenn nach dem Modell aus Text und Beispielen das gleiche allgemeine Situationswissen aufgebaut wird, werden diese als informationsäquivalent angesehen. Weil bei den im Modell beschriebenen Informationsverarbeitungsprozessen auch Vorkenntnisse eine wichtige Rolle spielen, können Lernmaterialien nur für einen bestimmten Vorkenntnisstand als informationsäquivalent gelten.

In dem Experiment wurden ein Text und für den vermuteten Vorkenntnisstand der Versuchspersonen dazu informationsäquivalente Beispiele eine konstante Zeit lang studiert. Es wurde eine sehr lange Studierzeit gewählt, damit die Versuchspersonen genügend Zeit hatten, das allgemeine Situationswissen aufzubauen. Dadurch, daß alle Versuchspersonengruppen, einschließlich der Gruppe, die sowohl Text als auch Beispiele studieren konnte, das gleiche allgemeine Situationswissen erwarben, wurde das Kognitive Modell bestätigt. Sofern die Vorkenntnisse von Versuchspersonen hinreichend genau eingeschätzt werden können, kann das Kognitive Modell zur Konstruktion informationsäquivalenter Materialien eingesetzt werden.

Das Kognitive Modell kann darüber hinaus dazu verwendet werden, den Wissenserwerb aus verschiedenen Sequenzen von Texten und Beispielen vorherzusagen. Es beschreibt, wie beim Studium von Beispielen nach Text, das aus dem Text erworbene Wissen zur Erklärung der Beispiele verwendet wird. Beim Studium von Text nach Beispielen, werden nach dem Modell die aus den Beispielen generierten Hypothesen mit den allgemeinen Aussagen des Textes verglichen. Daraus lassen sich für verschiedene Vorkenntnisstände differenzierte Vorsagen der Lesezeiten und der Lernergebnisse machen, die zur Zeit durch weitere Experimente geprüft werden.

Anmerkungen

1 Bei dem aus der S-Struktur konstruierten allgemeinen Situationswissen handelt es sich aus erkenntnistheoretischer Sicht um Hypothesen. Das aus der T-Struktur konstruierte allgemeine Situationswissen kann dagegen als Faktenwissen bezeichnet werden. Trotz dieser Unterschiede betrachten wir solches Wissen aus psychologischer Sicht als gleich.

2 Es wurde eine Greenhouse-Geisser-Korrektur der Freiheitsgrade durchgeführt.

3 Es wurde eine Freiheitsgradkorrektur nach getrennter Varianzschätzung durchgeführt.

4 Eine zusätzliche Auswertung, bei der eine Lösung dann als richtig eingestuft wurde, wenn Funktionsnamen und Argumente korrekt angegeben waren («semantisch» richtige Lösungen), ergab für Text-, Beispiel- und Kombinationsbedingung relative Häufigkeiten richtiger Lösungen von 0.69, 0.70, 0.69.

Literatur

Anderson, J. R., Farrell, R. & Sauers, R. (1984). Learning to program in LISP. Cognitive Science, 1984, 8, 87-129.

Anderson, J. R. & Thompson, R. (1987). Knowledge representation in the PUPS Theory. Technical Report. Carnegie-Mellon University.

Ausubel, D. P. (1964). Some psychological and educational limitations of learning by discovery. Arithmetic Teacher, 11, 290-302.

Baird, C. B. & Noma, E. (1978). Fundamentals of scaling and psychophysics. New York: John Wiley & Sons.

Bruner, J. S. (1961). The act of discovery. Harvard Educational Review, 31, 21-32.

Dijk van, T. A. & Kintsch, W. (1983). Strategies of discourse comprehension. New York: Academic Press.

Duncker, K. (1935). Zur Psychologie des produktiven Denkens. Berlin: Springer.

Kintsch, W. (1974). The representation of meaning in memory. Hillsdale, N.J.: Erlbaum.

Kintsch, W. & van Dijk, T. A. (1978). Toward a model of text comprehension and production. Psychological Review, 85, 363-394.

Kintsch, W. & Greeno, J. G. (1985). Understanding and solving word arithmetic problems. Psychological Review, 92, 109-129.

Kintsch, W., Welsch, D., Schmalhofer, F. & Zimny, S. (1990). Sentence Memory: A theoretical analysis. Journal of Memory and Language, 29, 133-159.

Larkin, J. H. & Simon, H. A. (1987). Why a diagram is (sometimes) worth ten thousand words. Cognitive Science, 11, 65-100.

Lefrancois, G. R. (1976). Psychologie des Lernens. Berlin: Springer.

Mitchell, T. M., Keller, R. M. & Kedar-Cabelli, S. T. (1986). Explanation-based generalization: A unifying view. Machine Learning, 1, 47-80.

Neber, H. (1981). Entdeckendes Lernen. Weinheim u. Basel: Beltz.

Newell, A. (1982). The knowledge level. Artificial Intelligence, 18, 87-127.

Reder, L. M. (1982). Plausibility judgments versus fact retrieval: Alternative strategies for sentence verification. Psychological Review, 89, 250-280.

Schmalhofer, F. (1986a). The construction of programming knowledge from system explorations and explanatory text: a cognitive model. In Rollinger, R. & Horn, W. (Eds.). GWAI-86 und 2. Österreichische Artificial Intelligence Tagung. Heidelberg: Springer, 152-163.

Schmalhofer, F. (1986b). Verlaufscharakteristiken des Informationsabrufs beim Wiedererkennen und Verifizieren von Sätzen. Zeitschrift für experimentelle und angewandte Psychologie, 33, 133-149.

Schmalhofer, F. & Glavanov, D. (1986). Three components of understanding a programmer's manual: verbatim, propositional and situational representations. Journal of Memory & Language, 25, 279-294.

Schmalhofer, F., Schlei, J. & Farin, E. (1986). Eine Anpassung der Assembler-Routinen der CMU-Button-Box an Turbo Pascal. Forschungsbericht Nr. 36, Psychologisches Institut der Universität Freiburg.

Schmalhofer, F., Boschert, S. & Kühn, O. (1987). Die ersten Stunden des Erwerbs von Computerkenntnissen: Text- und situationsbasierte Akquisitions- und Transformationsprozesse. Forschungsbericht Nr. 37, Psychologisches Institut der Universität Freiburg.

Schmalhofer, F. & Wetter, Th. (1988). Kognitive Modellierung: Menschliche Wissensrepräsentation und Verarbeitungsstrategien. In Christaller, Th., Hein, H.-W. & Richter, M. M. (Hrsg.). Künstliche Intelligenz: Theoretische Grundlagen und Anwendungsfelder. Heidelberg: Springer, 245-291.

Schmalhofer, F., Kühn, O. & Messamer, P. (1989). Receptive and exploratory learing in intelligent tutoring systems. In Alexander, J. H. (Ed.). Fourth Annual Rocky Mountain Conference on Artificial Intelligence, June 8-9, 1989, Conference Proceedings, 71-79.

Schmalhofer, F., Kühn, O., Messamer, P., Charron, R. (1990). An experimental evaluation of different amounts of receptive and exploratory learning in a tutoring system. Computers in Human Behavior, 6, 1990.

Vorberg, D. & Goebel, R. (1990). Rekursionsschemata als Problemlösepläne: Das Lösen rekursiver Programmierprobleme mit LOGO. (Preprint, zu beziehen über den Autor).

Waloszek, G., Weber, G. & Wender, K. F. (1986). Entwicklung eines intelligenten LISP-Tutors (Bericht No. 2). Braunschweig: Technische Universität, Institut für Psychologie.

Dr. Franz Schmalhofer, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Erwin-Schrödinger-Straße, D-6750 Kaiserslautern, Telefon (0631) 2053465

Stefan Boschert, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Erwin-Schrödinger-Straße, D-6750 Kaiserslautern, Telefon (0631) 2053464

Otto Kühn, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Erwin-Schrödinger-Straße, D-6750 Kaiserslautern, Telefon (0631) 2053461