Rheinische Friedrich-Wilhems-Universität Bonn
Institut für Kommunikationsforschung und Phonetik

Hauptseminar Linguistische Wissensformate
Prof. Dr. W. Lenders, Dr. B. Schröder

Frames
Scripts and Plans

Sebastian Marius Kirsch

Wintersemester 2003/2004

Inhaltsverzeichnis

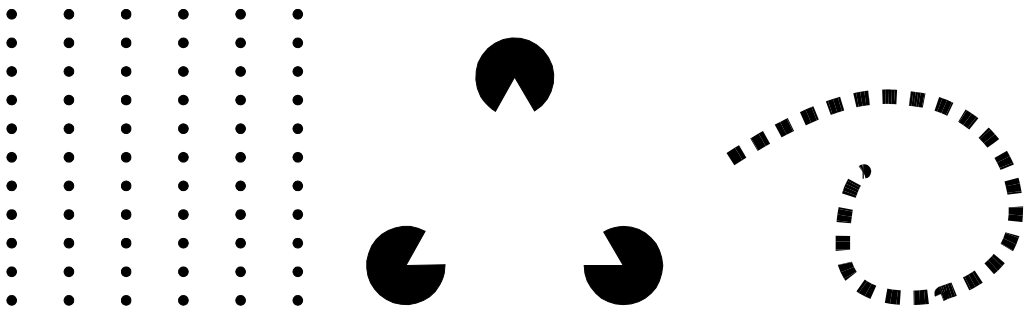
1 Frames

1.1 Einführung

- Frames sind ein universelles Wissensrepräsentationsformat für die Modellierung von Weltwissen. Frames besitzen keine festgelegte Semantik.
- Grundlage ist Gestaltpsychologie, Prototypentheorie der kognitiven Psychologie

  Gestaltpsychologie und die Prototypentheorie der kognitiven Psychologie sind Teile der Wahrnehmungspsychologie. Die Gestaltpsychologie geht zurück auf die Arbeiten von Max Wertheimer et. al. im Jahre 1935 und ist als Gegenbewegung zum Strukturalismus in der Wahrnehmungspsychologie zu sehen. Im Gegensatz zu den Strukturalisten, die Wahrnehmungserscheinungen in kleine Einheiten zu untergliedern versuchten, ging die Gestaltpsychologie davon aus, dass der Mensch Szenen immer als Ganzes, eben als „Gestalt“ wahrnimmt. Wir übernehmen von der Gestaltpsychologie z. B. die Gestaltgesetze (siehe Abbildung 1.1.)

  Die Prototypentheorie ist einer von drei Ansätzen der kognitiven Psychologie, um Wahrnehmungsvorgänge zu erklären. Sie geht davon aus, dass Prototypen zur Erkennung von bereits oft gesehenen Objekten eingesetzt werden. Die *template theory* hingegen nimmt an, dass gesehene Muster mit Vorlagen (*templates*) verglichen werden, während die *theory of distinctive features* Erkennungsmerkmale einsetzt, um Bildverarbeitung zu erklären.



(a) Gesetz der Nähe

(b) Gesetz der Geschlossenheit

(c) Gesetz der Prägnanz

Abbildung 1.1: Gestaltgesetze

- Marvin Minsky: Toshiba Professor of Media Arts and Sciences, MIT AI Lab, MIT Media Lab

  Marvin Minsky ist einer der großen Namen in der künstlichen Intelligenz: Gründer und langjährige Leitung des MIT AI Labs, Erbauer des ersten Neurocomputers, zahlreiche Veröffentlichungen, u. a. „Perceptrons“ und „The Society of Mind“, beide mit Seymour Papert.

- erstes Paper von 1974, sehr allgemein/bruchstückhaft

  Das erste Paper üeber Frames[1] ist sehr allgemein gehalten und legt nur allgemeine Ideen dar, was denn dieses „Ding genannt Frame“ sein könnte und welche Anforderungen eine solche Datenstruktur erfüllen muesste.

- Implementationen: KRL (Xerox, 1977), FRL (MIT, 1977) und andere

  In den Beispielen werde ich auf die am MIT selbst von R. Bruce Roberts und Ira P. Goldstein entwickelte *Frame Representation Language* FRL eingehen.

1.2 Grundlagen

- Datenstruktur, um stereotype Situationen zu repräsentieren

  Im Paper von Minsky wird der Einsatz von Frames zur Modellierung von Vorgängen im visuellen System, bei der Sprachverarbeitung und zur Modellierung von Situationen und Aktionen dargelegt.

  Praktisch eingesetzt wurden Frames in unterschiedlichen Gebieten, sowohl in der Wissensrepräsentation (WHOSIS, eine Datenbank mit Namen und Adressen der Angehörigen des AI LAB, NUDGE, ein Terminplaner, TRIPPER, eine Datenbank mit Reiseinformationen), als auch in der Diskursmodellierung und Verarbeitung natürlicher Sprache (COMEX, zur Modellierung von Diskursstrukturen, und PAL, ein natürlichsprachiges Frontend zu NUDGE.)

- Frames bilden ein hierarchisches Netzwerk, dessen höhere Level fest sind, während niedrigere Level *Terminale/Slots* haben, die nach Bedarf gefüllt werden.

- Den Inhalt von Slots bilden *Sub-Frames*

- Slots enthalten Default-Belegungen

- Erwartungen und Beschränkungen an die Subframes, die einen Slot füllen können

  Die Information eines Frames ist demnach auf zwei Arten kodiert: Einmal in der Struktur des Frames selbst und zum Zweiten an die Anforderungen, die an Subframes gestellt werden. Dies wird klarer werden, wenn wir Beispiele sehen.

- Suchprozess nach einem Frame, dessen Terminale vorhandene Subframes aufnehmen und zu einem größeren Ganzen zusammenbauen können

Dieser Suchprozess wird in diesem Vortrag nicht behandelt – aber wir kennen ihn schon aus dem vorherigen Vortrag von Fabio Fracassi über Merkmalsstrukturen: es ist Unifikation.

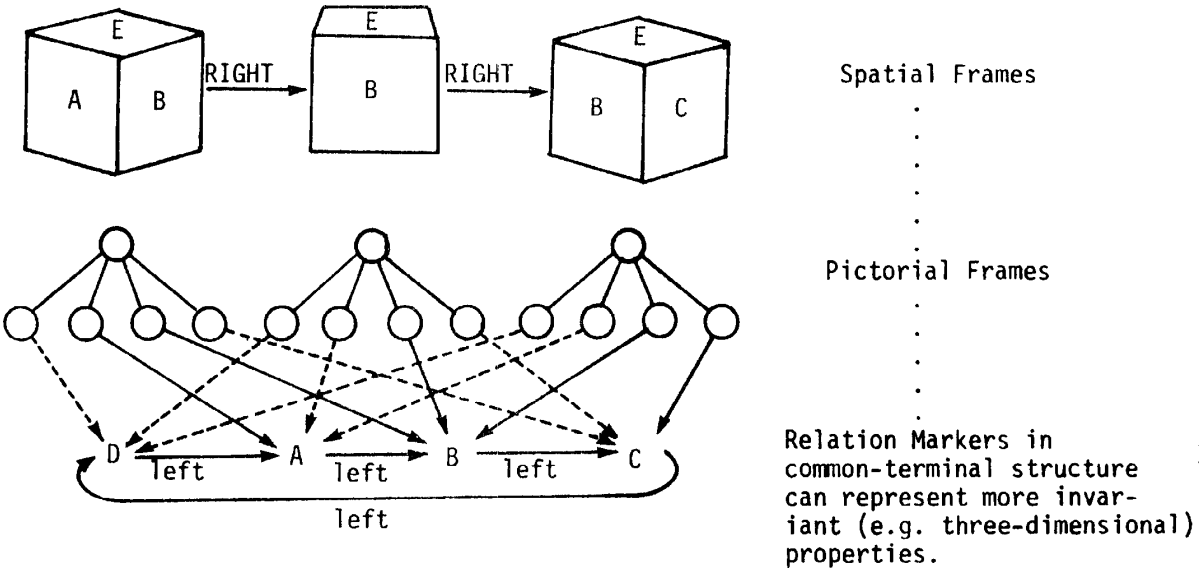
1.3 Frame-Systeme

- Mehrere Frames bilden ein *Frame-System*
- Frames eines Systems haben die gleichen Terminale
- erklärt, wie Information koordiniert wird

  Die Mächtigkeit von Frames zeigt sich erst, wenn man nicht einzelne Frames betrachtet, sondern ganze Frame-Systeme. Bei Frame-Systemen enthalten mehrere *top-level*-Frames die selben Sub-Frames in ihren Terminalen; Änderungen, die bei der Verarbeitung eines *top-level*-Frames an einem Sub-Frame ausgeführt werden, sind so für alle Frames des Systems sofort verfügbar.
- über viele Probleme wird auf mehreren Ebenen nachgedacht

  Minsky illustriert dies am Beispiel eines Automechanikers, der ein Problem sucht: Dieser wird mehrere verschiedene Sichtweisen des Autos haben, z. B. eine Unterteilung in funktionale Komponenten, eine des elektrischen Systems, eine mechanische Sicht etc. – diese *top-level*-Frames zeigen jedoch auf die gleichen Sub-Frames, nämlich die Komponenten des Autos, das gerade untersucht wird. Informationen, die bei der Untersuchung einer Komponente gewonnen wurden, sind in allen Sichtweisen sofort verfügbar.
- Information muss nicht neu berechnet werden, sondern nur neuen Terminalen zugeordnet werden

1.4 Beispiel: Frames im visuellen System



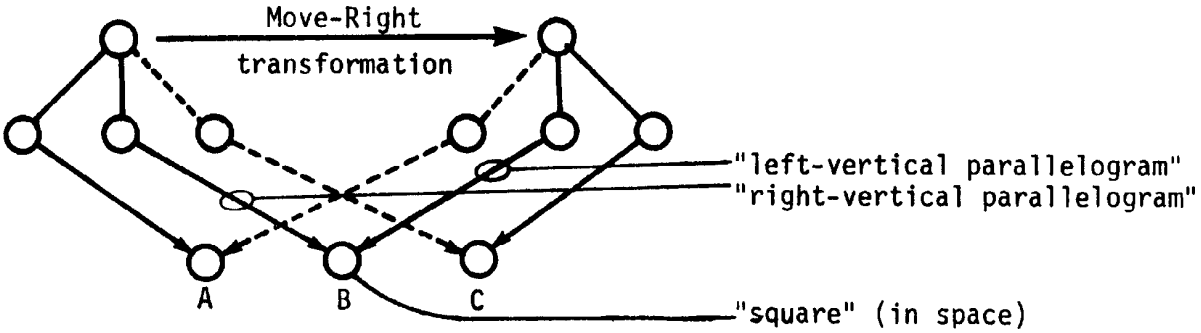
- Flächen-Frame füllt Slots des Würfel-Frames

  Die Verarbeitung erfolgt auf zwei verschiedenen Ebenen: Einer symbolischen, auf der ein Würfel als dreidimensionales Gebilde modelliert wird, und einer visuellen, auf der die aktuelle Ansicht des Würfels modelliert wird. Slots, die auf beiden Ebene für das selbe Objekt stehen, werden vom selben Frame gefüllt.

- Drehung des Würfels: Subframes werden neuen Slots zugeordnet

  Bei der Drehung des Würfels muss Information nicht neu berechnet werden, da der Mensch als Beobachter im Laufe seiner Entwicklung Wissen über perspektivische Transformationen gesammelt hat und so das Ergebnis einer solchen Transformation in Teilen vorhersagen kann. Aus diesem Grund werden bei einer Drehung des Würfels die Frames auf der symbolischen Ebene, die für eine Facette des Würfels stehen, nur neuen Slots auf der visuellen Ebene zugeordnet.

1.5 Frame-Transformationen



- gelernte Regeln bauen Frames um

- Neuberechnung wird vermieden

  Im Falle der perspektivischen Transformationen geht die Entwicklungspsychologie davon aus, dass ein Kind im Alter von acht bis neun Jahren das nötige Wissen hat, um die Veränderungen bei der Drehung von Gegenständen im Geist nachvollziehen und vorhersagen zu können. Gleichzeitig erklärt die Tatsache, dass es sich hier um gelernte Transformationen handelt, die relativ geringe Performanz des Menschen bei solchen Aufgaben – die meisten Menschen kommen sehr schnell durcheinander, wenn sie mehrmals hintereinander Drehungen etc. eine Würfels im Kopf nachvollziehen und das Ergebnis vorhersagen sollen.

  Obwohl derlei gelernte Transformationen weniger mächtig als z.B. ein a-priori-Modell der Perspektive sind, verkürzen sie die Verarbeitungszeit drastisch, und helfen, Neuberechnungen zu vermeiden.

- ähnlich zu Transformationsoperatoren in der Transformationsgrammatik

- logisches Denken und „gesunder Menschenverstand“ scheint ähnliches Vorgehen wie linguistische Transformationen zu benutzen

- ähnliche Mechanismen beim Denken wie bei der Kommunikation?

  Dies ist natürlich eine sehr breite und gewagte Aussage – aber unabhängig davon, ob der vorgestellte Formalismus in der Tat dem entspricht, der im menschlichen Gehirn benutzt wird, impliziert er prinzipielle Charakteristiken der Informationsverarbeitung im Gehirn. Er legt dar, dass mehrere Ebenen der Informationsverarbeitung entsprechend dem selben Formalismus ablaufen können und gibt damit eine Richtung für weitere Forschungen in dieser Richtung vor.

1.6 Frames und Sprache

- Frames, Constraints und Default-Belegungen können benutzt werden, um sprachliche Vorgänge zu beschreiben

  Frame-Datenstrukturen können auch benutzt werden, um Sätze, Satzfragmente etc. zu beschreiben. Constraints übernehmen dabei die Rolle von Selektionsbeschränkungen; Default-Belegungen erklären *lookahead*-Phänomene beim Sprachverstehen.

- generative Grammatik entspricht der äusseren Erscheinung der sprachlichen Frames

  Ähnlich wie das angesprochene Modell für visuelle Informationsverarbeitung, das auf mehreren Ebenen arbeitet (hier die visuelle und die symbolische Ebene,) arbeitet auch die Transformationsgrammatik mit einer Tiefenstruktur und einer Oberflächenstruktur. Eine ähnliche Trennung zwischen Ebenen ließe sich auch mit Frames realisieren.

- Operatoren der Transformationsgrammatik haben Ähnlichkeit mit Frame-Transformationen

Zum Zeitpunkt der Veröffentlichung des grundlegenden Papers[1] über Frames waren Transformationsgrammatiken eine aktuelle linguistische Theorie; auch wenn sie inzwischen als „überholt“ gelten, musste sich eine neue Theorie über das Sprachverständnis an ihnen messen. Minsky schlägt den Bogen zu den damals aktuellen Entwicklungen in der Linguistik, indem er ähnliche Mechanismen auch in seiner Theorie anbietet.

- Idee: linguistische Aktivität umfasst größere Strukturen, als durch eine Grammatik beschrieben werden kann

  Minsky nimmt an, dass Frames in der Sprachverarbeitung den Rahmen eines Satzes schnell sprengen und in größere Einheiten eingebettet werden – zum Beispiel Geschichten, Kontexte, Sprechakte.

- Syntax und Semantik sind keine Dichotomie, sondern zwei Extreme eines kontinuierlichen ganzen

  Die klassische Trennung zwischen Syntax und Semantik lehnt Minsky ab; er plädiert eher für einen Verarbeitungs-„Mischmasch“, der auf beiden Ebenen gleichzeitig abläuft.

- Nichtverstehen eines Satzen kann auf das Fehlschlagen eines semantischen Prozesses hindeuten

  Als Beispiel führt Minsky die Sätze

  > „colorless green ideas sleep furiously“

  und

  > „furiously sleep ideas green colorless“

  an. Beide Sätze sind sinnlos, aber sie sind auf verschiedenen Ebenen sinnlos: Der erste Satz ist syntaktisch korrekt, aber semantisch nicht interpretierbar, während der zweite auch auch syntaktischer Ebene nicht verarbeitet werden kann.

1.7 Frames und Sprache II

- bei nichtgrammatischen Sätzen findet kein Subframe genügend Fragmente, um gefüllt zu werden
- deshalb kann auch kein höherer Frame dem Satz eine Bedeutung zuordnen

  Im Gegensatz dazu verletzen semantisch nicht verarbeitbare Sätze Constraints, also Selektionsbeschränkungen, und können deshalb nicht in einen höheren Frame eingebettet werden.

- niedrige Level der syntaktischen Verarbeitung sind *verb-driven* (oder *head driven*)
- Bedeutung des Verbs tritt bei höheren Leveln in den Hintergrund

1.8 Frame Representation Language

- entwickelt 1977 von Roberts und Goldstein (MIT)
- spezialisierte Datenstruktur und LISP-Funktionen
- implementiert Defaults, Vererbung, Constraints, *procedural attachment* und Annotationen
- Quellcode (in MACLISP) erhältlich unter [6]

  Für MACLISP ist leider seit vielen Jahren kein Interpreter mehr verfügbar; eine Portierung der FRL nach Common Lisp ist mir nicht bekannt. Der Quelltext der FRL kann unter [6] heruntergeladen werden; das Handbuch[4] und ein Primer[5] sind ebenfalls verfügbar.

1.9 FRL Beispiele I: Frames erzeugen

FASSERT legt einen neuen Frame an, hier einen Frame mit Namen MINSKY, der Informationen über die Person Marvin Minsky enthält. Der Frame besteht aus mehrere Slots, wie NAME und ADDRESS. Jeder Slot kann mehrere Facetten haben; wir sehen in den ersten Beispielen nur die $VALUE-Facette, die den Wert eines Slots enthält. Die ersten drei Slots enthalten Strings als Wert, während der Slot INTEREST mit dem Wert REPRESENTATION auf einen Subframe verweist.

```
(FASSERT MINSKY
    (NAME         ($VALUE ( |Marvin Minsky| )))
    (ADDRESS      ($VALUE ( |545 Technology Square, R...
    (PUBLICATIONS ($VALUE ( |A Framework for Represen...
    (INTERESTS    ($VALUE ( REPRESENTATION ))))
```

Mit dem Befehl FPUT wird einem Frame ein neues Datum hinzugefügt; im folgenden Beispiel wird der $VALUE-Facette des Slots INTEREST im Frame MINSKY ein neues Datum hinzugefügt, nämlich ROBOTICS; gleichzeitig wird ein sogenanntes Label „SOURCE:“ angelegt, das den Wert RBR erhält. Dieses Label gibt die Quelle der Information an; RBR steht hier wohl für R. Bruce Roberts, einen der Autoren der FRL.

```
(FPUT 'MINSKY 'INTEREST '$VALUE 'ROBOTICS 'SOURCE: 'RBR)
```

Nach dem FPUT-Befehl sieht der Frame wie folgt aus:

```
(MINSKY
    ...
    (INTERESTS    ($VALUE ( REPRESENTATION )
                          ( ROBOTICS (SOURCE: RBR)))))
```

1.10 FRL Beispiele II: Daten extrahieren

Der Befehl FGET extrahiert Daten aus Frames; dabei werden im Allgemeinen nur die Inhalte der Facetten extrahiert, nicht die Labels.

```
(FGET 'MINSKY 'INTERESTS '$VALUE)
    -> ( REPRESENTATION ROBOTICS )
```

Will man den Inhalt eines Labels extrahieren, muss dieses speziell angegeben werden.

```
(FGET 'MINSKY 'INTERESTS '$VALUE 'ROBOTICS 'SOURCE:)
    -> ( RBR )
```

1.11 FRL Beispiel III: Vererbung

Vererbung wird in der FRL über einen speziellen Slot, den AKO-Slot gelöst. AKO steht hier für „a kind of“, gibt also an, dass ein Frame eine Art anderer Frame ist. (In der Programmiersprache Perl wird Vererbung ähnlich gelöst; mit dem Unterschied, dass die Variable dort nicht AKO, sondern @ISA, also „is a“ heisst.)

```
(FASSERT MIT-AI
    (STREET ($VALUE   ( |545 Technology Square| )))
    (CITY   ($VALUE   ( |Cambridge| )))
    (OFFICE ($DEFAULT ( 817 )))
    (ZIP    ($VALUE   ( |02139| ))))

(FASSERT MINSKY
    (NAME         ($VALUE ( |Marvin Minsky| )))
    (AKO          ($VALUE ( MIT-AI )))
    (OFFICE       ($VALUE ( 821 )))
    (PUBLICATIONS ($VALUE ( |A Framework for Represen...
    (INTERESTS    ($VALUE ( REPRESENTATION ))))

(FGET 'MINSKY 'CITY '$VALUE)
    -> ( |Cambridge| )
```

2 Scripts and Plans

2.1 Einführung

- Repräsentation von stereotypen Abläufen

  Während Frames ein allgemein einsetzbares Format sind, sind Scripts nur für den eingeschränkten Bereich der Repräsentation von sozialen Abläufen konzipiert.

- Roger Schank: John Evans Professor Emeritus der Informatik, Bildung und Psychologie, Northwestern University; Robert Abelson: Professur Emeritus der Psychologie, Yale University

  Roger Schank gründete 1989 das *Institute for Learning Sciences* an der Northwestern University und ist heute mit seiner Firma Socratic Arts im Bereich E-Learning aktiv. Robert Abelson ist im Übrigen nicht Harold Abelson (MIT, „The Structure and Interpretation of Computer Programs“, etc.) zu verwechseln.

- zwei Klassen von Wissen: spezialisiertes Wissen und allgemeines Wissen

  Allgemeines Wissen könnte man auch als „logisches Denkvermögen“ oder „gesunden Menschenverstand“ bezeichnen – die Fähigkeit, aufgrund allgemeiner logischer Regeln Schlüsse ziehen zu können. Spezialisiertes Wissen hingegen muss diesen Regeln nicht genügen und kann nur in bestimmten Situationen angewandt werden.

- ein grosser Teil der alltäglichen Interaktion zwischen Menschen läuft nach gelernten Schemata, sog. *Scripts* ab

  Diese Scripte, die teilweise regelrechten „Ritualen“ gleichen, sind stark kulturell abhängig – ein Script, das in Mitteleuropa funktioniert, könnte in Russland oder China zu Verwirrungen führen.

- Scripts sparen Verarbeitungszeit und schränken Erwartungen bei häufig erlebten Situationen ein

  Soziale Interaktion läeuft oft innerhalb von Sekundenbruchteilen ab und erlaubt es deshalb nicht, komplett von Grund auf durchgeplant zu werden. Scripts erlauben akzeptable Performanz in diesen Situationen.

- angewendet in SAM (Script Applier Mechanism) von Cullingford, Lehnert, Gershman, Carbonell (1975): Programm liest Geschichten, generiert Zusammenfassungen in mehreren Sprachen und beantwortet Fragen zu den Geschichten

2.2 Beispiel: Geschichten

- Scripts ermöglichen es, auch bei unvollständig erzählten Geschichten den kompletten Handlungsrahmen zu synthetisieren

  Dieses Verfahren wurde in SAM angewendet, um Geschichten zusammenzufassen – SAM synthetisiert aus den Hinweise in der Geschichte das komplette Script und erzeugt dann eine Zusammenfassung des Scripts.

- verläuft eine Geschichte entlang eines Scripts, werden fehlende Passagen kaum wahrgenommen.

- Beispiel:

  > John went to a restaurant. He asked the waitress for coq au vin. He paid the check and left.

  Diese Geschichte enthält alle Schlüsselelemente, die für einen Restaurantbesuch wichtig sind: Den passenden Ort, den Akt des Bestellens und den Akt des Bezahlens. Bei der Instantiierung eines passenden Scripts treten deshalb keine Schwierigkeiten auf.

- fehlen wichtige Schlüsselszenen, die die Instantiierung eines Scripts hervorrufen, führt dies zu Verwirrung:

  > John went to a restaurant. He saw a waitress. He went home.

  In dieser Geschichte bleiben wichtige Fragen offen: Hat John die Bedienung nur gesehen, oder hat er auch bei ihr bestellt? Hat er etwas gegessen, hat er bezahlt? So bleibt unklar, ob in dieser Geschichte ein „erfolgreicher“ Restaurantbesuch beschrieben wird.

2.3 Beispiel: Geschichten ohne Script

- verweist eine Geschichte auf kein Script, so ist sie oft vollkommen unverständlich:

  > John was walking on the street. He thought of cabbages. He picked up a shoe horn.

  Es ist nicht möglich, aus diesen Informationen ein Script zu instantiieren, das einen übergreifenden Handlungsrahmen liefert; so bleibt die Geschichte unsinnig.

- Geschichten müssen nicht auf ein Script verweisen, um verständlich zu sein:

  > John wanted a newspaper. He found one on the street. He read it.

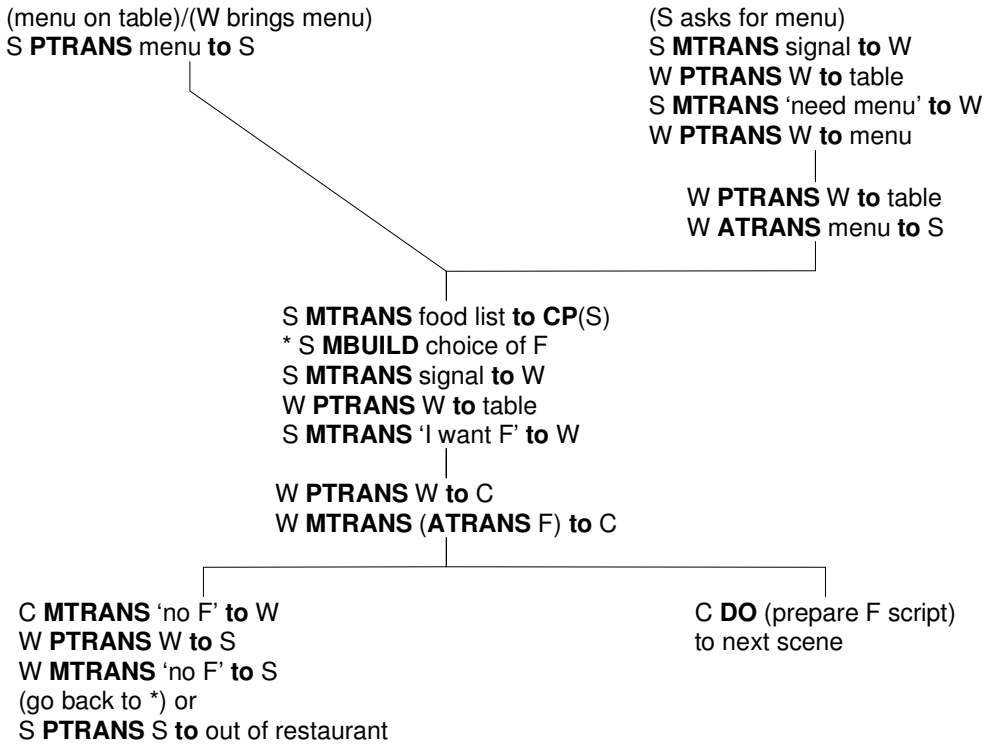
  Diese Geschichte ist auch ohne Verweis auf ein Script verständlich, da die einzelnen Elemente logisch aufeinander folgen: Der zweite Satz ist eine Mittel zur Erfüllung des im ersten Satz ausgesprochenen Wunsches, und die Voraussetzungen für den dritten Satz wurden durch die Wunscherfüllung geschaffen.

- illustriert Unterschied zwischen logischem Denken und scriptgesteuerter sozialer Interaktion

  Soziale Interaktion muss nicht unbedingt logisch sein, aber wenn ein passendes Script gelernt wurde, ist sie verständlich. Logische Abläufe hingegen benötigen keine Scripts.

2.4 Das Restaurant-Script

Die folgende Abbildung stellt einen kurzen Ausschnitt aus dem Script für einen Restaurant-Besuch dar. Wir sehen zwei Verzweigungen im Script, Vorbedingungen für die einzelnen Äste (Speisekarte auf dem Tisch oder nicht), und eine Reihe von primitiven Aktionen wie PTRANS (*physical transportation*), MTRANS (*mental transportation*), MBUILD (Erstellen eines Plans) und DO (Ausführen eines anderen Scripts.)



2.5 Instantiierung von Scripts

- Scripts werden durch ihre Header instantiiert

  Zu einem gegebenen Zeitpunkt können durchaus mehrere Scripts gleichzeitig aktiv sein, und ein Geschehnis in einer Geschichte kann die Ausführung eines Scripts beenden und zur Instantiierung eines anderen Scripts führen.

- Verschiedene Typen von Headern:

Precondition Header (PH) Voraussetzungen, die für die Abarbeitung eines Scripts erfüllt sein müssen (Beispiel Restaurant: Akteur ist hungrig)

Instrumental Header (IH) aktuelles Script ist nur „Mittel zum Zweck“, bietet den Hintergrund für eine andere Aktion

Locale Header (LH) Script wird durch eine bestimmte Örtlichkeit aktiviert (Beispiel: Restaurant)

Dieser Header ist mit Vorsicht zu geniessen – ein Paketbote, der in einem Restaurant ein Paket ausliefert, will dort nur selten direkt zu Mittag essen.

Internal Conceptualization Header (ICH) Ein beliebiges Konzept einer Geschichte kann ein neues Script instantiieren.

2.6 Pläne

- In Fällen, bei denen kein Script greift, wird ein allgemeinerer Mechanismus benutzt, der „Plan“.

  Ein Plan setzt nicht auf die Anwendung eines vorgefertigten Handlungsablaufs zum Erreichen eines Ziels, sondern auf das Zusammensetzen von mehreren primitiven Aktionen, die zum Erreichen führen.

- Verhältnis zwischen Plänen und Scripts ist nicht klar

- Pläne, die mehrfach ausgeführt wurden, können zu einen Script werden, müssen aber nicht

- genauso kann eine Situation, die nur ein einziges Mal erlebt wurde, zu einem Script internalisiert werden.

  Der Autor führt zur Illustration das Beispiel seiner kleinen Tochter an, die mit ihm ein neues Auto abholt und ihn vorher fragt, ob er denn nun auch einen neuen Schlüsselanhänger bekommt. Der Autor berichtet, dass er stutzte, kurz nachdachte, und sich dann erinnerte, dass seine Tochter mit ihm bis jetzt nur einmal ein neues Auto abgeholt hat, und er dort in der Tat einen neuen Schlüsselanhänger bekommen hat. Obwohl diese Situation nur einmal erlebt wurde, ist für die Tochter der Punkt „Schlüsselanhänger bekommen“ Teil des „Auto abholen“-Scripts.

- Spezialfall: Benannte Pläne beschreiben allgemeine Strategien zur Problemlösung

  Ein Beispiel für eine solche Strategie ist „Wenn ich etwas wissen will, frage ich jemanden, der es weiss.“

- Primitive **MBUILD** in Scripts verweist auf das Erstellen eines Plans

  Diese Primitive taucht auch im Restaurant-Skript auf, wenn der Akteur entscheidet, was er essen will.

2.7 Elemente von Plänen

- Pläne dienen dem Erreichen eines Ziels. Im vorliegenden Formalismus gibt es zwei Arten von Zielen:

  I-goals *instrumental goals*, einfache und stereotype Ziele, oft durch instrumentelle Scripts erreicht

  D-goals *delta goals*, Ziele, die Veränderungen beschreiben, die für das Erreichen des Hauptziels nötig sind.

- **D-goals** haben keinen eigenen Sinn, sondern dienen nur dem Erreichen der höheren Ziele.

- Arten von **D-goals**:
  - **D-KNOW** – das Ziel, etwas zu wissen
  - **D-PROX** – das Ziel, in die Nähe von etwas zu kommen
  - **D-CONT** – das Ziel, Kontrolle über etwas zu erlangen
  - **D-SOCCONT** – das Ziel, „soziale Kontrolle“ zu erlangen; Beispiel: Jemand, der Theaterkarten kauft, kontrolliert dadurch nicht das Theater. Er hat aber im sozialen Kontext Kontrolle darüber, dass er die Theatervorstellung besuchen kann.
  - **D-AGENCY** – das Ziel, jemand anderen dazu zu bringen, etwas zu erledigen.

2.8 Benannte Pläne

- Schritt zwischen Script und Plan

  Ein benannter Plan ist nicht spezifisch genug für ein Script, da er noch Ziele enthält, die für jeden benannten Plan eigens aufgelöst werden müssen. Ein Teil der Planungsarbeit wird jedoch schon im Vorfeld geleistet.

- Beispiel:

$$\mathbf{USE}(X) = \mathbf{D}-\mathbf{KNOW}(\mathbf{LOC}(X)) + \mathbf{D}-\mathbf{PROX}(X) + \mathbf{I}-\mathbf{PREP}(X) + \mathrm{DO}$$

  Mit anderen Worten: „Wenn man etwas benutzen will, muss man zuerst wissen, wo es sich befindet, sich dann dorthin bewegen, die Benutzung vorbereiten und danach ausführen.“

- allgemeine Problemlösungsstragien werden so unter einem bestimmten Namen abrufbar gemacht

- Probleme mit rekursiven Plänen $\mathbf{ASK} \rightarrow \mathbf{FIND} \rightarrow \mathbf{ASK} \rightarrow \mathbf{FIND} \rightarrow \ldots$

  Rekursion ist auch in logischen Programmiersystemen wie Prolog ein Problem, und Schleifenerkennung ist ein aktuelles Forschungsthema.

2.9 Planboxen

- Planboxen werden zur Auflösung von Zielen benutzt; typischerweise kann jedes Ziel durch verschiedene Planboxen aufgelöst werden.
- Elemente einer Planbox sind die folgenden Felder:

  ACT enthält die Aktion, die ausgeführt wird

  UP ist die *uncontrollable precondition*, eine Vorbedingung, die vom Akteur nicht beeinflusst oder herbeigeführt werden kann.

  CP die *controllable precondition* kann von Akteur (zum Beispiel durch Erzeugung eines Unterziels) herbeigeführt werden, wenn sie nicht schon erfüllt ist.

  MP die *mediating precondition* bezieht sich auf die Interaktion zwischen zwei Akteuren

  RES bezeichnet das Resultat nach der Ausführung der Planbox.

- Planbox für **ASK**: löst **D-KNOW** auf

ACT	X **MTRANS Q? to** Y
	X überträgt die Frage an Y
CP	X **BE**(**PROX**(Y))
	X muss sich in der Nähe von Y befinden
UP	Y knows Q
	Y muss die Antwort wissen
MP	Y wants to **MTRANS** Q **to** X
	Y muss auch gewillt sein, X die Antwort zu verraten
RES	Y **MTRANS** Q **to** X
	Als Resultat überträgt Y die Antwort an X

2.10 Fazit

- zwei Beispiele für frühe Formalismen zur Speicherung von menschlichem Wissen

  Die Struktur von Wissensformat hat sich seit den siebziger Jahren stark verändert – durch Fortschritte in der Psychologie, Neurologie und der mathematischen Logik, wie auch durch schnellere Rechner und höhere Speicherkapazitäten.

- erfolgreiche Programme (vgl. SAM) auf Basis dieser Formalismen
- praktische Anwendbarkeit ist trotzdem beschränkt

  Wie viele Projekte aus der Frühzeit der KI konnte SAM in einem eng beschränkten Anwendungsbereich erstaunliche Resultate erbringen, versagt jedoch, sobald dieser Bereich verlassen wird.

- haben heute an Bedeutung verloren
- Verhältnis zwischen Formalismen und dem menschlichen Denken an sich ist ungeklärt

Literaturverzeichnis

[1] Marvin A. Minsky: *A Framework for Representing Knowledge.* In: Patrick Henry Winston (ed.): The Psychology of Computer Vision. New York 1975. S. 211-277. ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-306.pdf

[2] Roger C. Schank, Robert P. Abelson: *Scripts, Plans, Goals and Understanding.* An Inquiry into Human Knowledge Structures. New York et al. 1977. Chapters 3-4.

[3] Wolfgang Wahlster: *Frames.* In: Skript zur Vorlesung 'Einführung in die künstliche Intelligenz', Sommersemester 2002, Universität des Saarlandes, Saarbrücken. http://w5.cs.uni-sb.de/ss02/materialien/KI-2002-131-138.pdf

[4] R. Bruce Roberts, Ira P. Goldstein: The FRL Primer. AI Memo 408, Juli 1977. ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-408.pdf

[5] R. Bruce Roberts, Ira P. Goldstein: The FRL Manual. AI Memo 409, Juli 1977. ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-409.pdf

[6] FRL: Frame Representation Language. http://www-2.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/kr/systems/frames/frl/frl.tgz