Semantische Netze

Marc Kirchner
<mail@marc-kirchner.de>

18. Juli 2002

Zusammenfassung

Diese Ausarbeitung gibt einen Überblick über verschiedene Arten semantischer Netze, deren Eigenschaften und Verwendungszwecke.

Inhaltsverzeichnis

1 Einführung

Semantische Netze stellen eine grafische Form der Wissenrepräsentation dar. Zur Visualisierung werden Knoten und Verbindungen zwischen diesen Knoten verwendet. Verschiedene Formen von semantischen Netzen werden schon lange in den Bereichen der Psychologie, Linguistik und Philosophie eingesetzt. Erste rechnergestützte Implementierungen entstanden im Umfeld der Künstlichen Intelligenz, anfänglich vor allem mit dem Ziel, natürliche Sprache zu verstehen und zu verarbeiten. Heute sind die Anwendungen semantischer Netze breit gefächert.

Allen Arten von semantischen Netzen ist ein grundliegender syntaktischer Aufbau gemeinsam. Semantische Netze sind

- gerichtete Graphen, deren
- Komponenten von verschiedenen, wohldefinierten Typen

sind. Das heißt, sowohl jeder Knoten als auch jede Verbindung ist von einem bestimmten Typ.

Semantische Netze sind in den verschiedensten Formen und Farben zu finden. Sie unterscheiden sich voneinander vor allem in zwei Punkten:

- Manchen semantischen Netzwerktypen liegt eine formale Definition zugrunde, anderen nicht.
- Die zur Verfügung stehende Menge an Knoten- und Verbindungstypen variiert von Netztyp zu Netztyp beträchtlich.

Anhand dieser Kriterien kann man die folgenden Netzwerktypen unterscheiden. Diese Typen sind meist selbst noch in Untertypen gegliedert.

1. definitional networks
2. assertional networks
3. implicational networks
4. executable networks
5. learning networks
6. hybrid networks

2 Definitional Networks

Der Schwerpunkt von Definitional Networks liegt im Aufbau von Onthologien und Typspezifikationen, insbesondere von Klassen- und Generalisierungshierarchien. Daher basieren solche Netze auf IS-A und subtype-Relationen. Oft verwendete Relationstypen sind:

- IS-A, AKO (a kind of) zur Darstellung von Vererbungs- bzw. Generalisierungshierarchien
- PART-OF, HAS-A-PART zur Darstellung der Aggregation
- MEMBER-OF, INSTANCE-OF für die Zwecke der Individualisierung.

Diese Sachverhalte sind in Abb. 1 veranschaulicht. Die Vererbungshierarchie

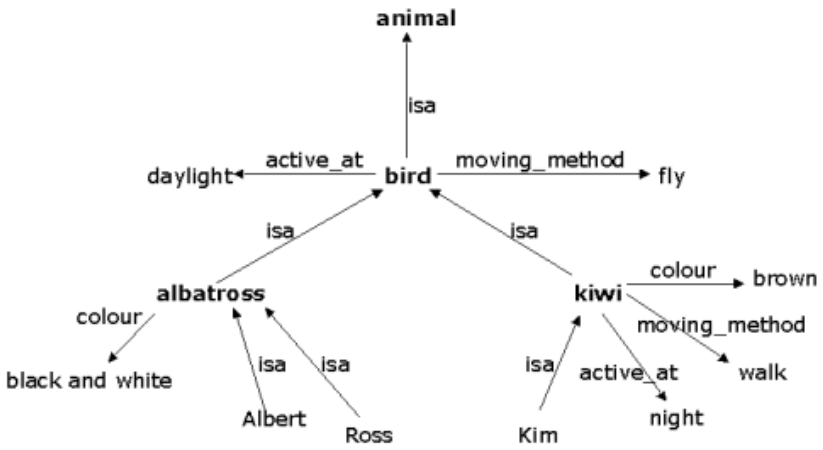

Abbildung 1: Vererbungshierarchie [Bra01]

spiegelt die Verwandschaftsverhältnisse zwiscehn Albatrossen, Kiwis, Vögeln und Tieren wider. Sowohl Albatrosse als auch Kiwis sind Vögel und ein Vogel ist ein Tier. Ross und Albert sind Albatrosse, Kim ein Kiwi.

Von den Knoten abgehende Verbindungen die nicht vom Typ IS-A sind zeigen Attribute des Knotens an. So wird mit `colour` beispielsweise die Färbung eines Albatross auf `schwarz und weiß` festgelegt oder die `moving_method` des Kiwis auf `walk` festgelegt. An dieser Stelle ist eine weitere Eigenschaft definitionaler Netzwerke zu erkennen: Attributwerte können als

- default-Werte oder
- cancelling-Werte

verwendet werden. Dies bedeutet, dass Attributwerte übergeordneter Knoten automatisch in die Unterklassen vererbt werden, die Werte jedoch in den Unterklassen überschrieben werden können. Ein Beispiel hierfür wäre das Attribut `moving_method` der Knoten `bird` und `kiwi`. In `bird` wird festgelegt, dass sich ein Vogel fliegend fortbewegt. Dieser Wert wird an den Kiwi vererbt und muss daher von diesem überschrieben werden, da diese Tatsache schlicht und einfach nicht zutrifft. Anders beim Albatross: hier trifft der geerbte Wert zu, wird dementsprechend nicht lokal überschrieben und wird daher verwendet.

2.1 Inferenz in Definitional Networks

Man unterscheidet in Definitional Networks drei Inferenzansätze. In implementierten Systemen werden meist Mischformen dieser Ansätze verwendet.

2.1.1 Deduktive Inferenz

Die deduktive Inferenz beantwortet Fragen der Form

> "Auf welche Art und Weise bewegt sich Ross?"

folgendermaßen:

> "Ross ist ein Albatross, ein Albatross ist ein Vogel und ein Vogel fliegt. Daher fliegt auch jeder Albatross und daher fliegt auch Ross."

Wird in diesem Falle also nach einem Attributwert gefragt, der nicht direkt am Knoten definiert ist, so erfolgt ein erneutes Stellen der selben Frage eine IS-A-Hierarchiestufe höher. Dies entspricht einem Propagieren der Werte entgegen der Richtung der IS-A-Hierarchie. Ist der Attributwert direkt am Knoten definiert, ist kein weiteres Verfolgen der IS-A-Hierarchie notwendig.

Beispiel in Prolog

Das folgende Beispiel führt innerhalb eines kleinen Netzwerks deduktive Inferenz entlang der IS-A-Hierarchie durch.

```
%
% deduktive Inferenz in einem definitional
% network, mit cancelling-Werten
%
```

```
% mail@marc-kirchner.de
%

fact(F) :-
        F, !.
fact(F) :-
        F=..[Relation, Entity1, Entity2],
        is_a(Entity1, SuperEntity),
        SuperFact=..[Relation, SuperEntity, Entity2],
        fact(SuperFact).

% Datenbasis
is_a(bird, animal).
is_a(kiwi, bird).
is_a(albatros, bird).
is_a(kim, kiwi).
is_a(albert, albatros).

moving_method(bird, fly).
moving_method(kiwi, walk).

colour(kiwi,  brown).
colour(albatros, black_and_white).
```

Abfragen erfolgen beispielsweise mit:

```
   :- fact(moving_method(bird, X)).
X = fly

:- fact(moving_method(albatros, X)).
X = fly

:- fact(moving_method(kiwi, X)).
X = walk
```

2.1.2 Induktive Inferenz

Induktive Inferenz schließt aus Eigenschaften, die für einzelne Objekte gelten auf Eigenschaften der Gesamtheit. Ein Beispiel hierfür wäre:

> "Sowohl ein Albatross als auch ein Kiwi besitzen Federn. Damit besitzen alle Subtypen des Typs `bird` Federn. Also besitzt auch `bird` Federn."

Diese Vorgehensweise entspricht - entgegen der deduktiven Inferenz - einem Propagieren der Attribute und ihrer Werte in Richtung der IS-A-Hierarchie.

In Anbetracht der Unvollständigkeit des Wissens das mit einem Definitional Network dargestellt wird, ist es verständlich, dass diese Art des Schließens die immense Gefahr voreiliger Schlüsse in sich birgt.

2.1.3 Analoge Inferenz

Analoge Inferenz bezeichnet eine Schlussweise, die, ausgehend von einer bereits erarbeiteten Lösung eines Problems oder einer beantworteten Fragestellung, den zuvor erarbeiteten Lösungsweg verwendet um das aktuelle Problem zu lösen, bzw. die aktuelle Frage zu beantworten.

2.2 Anwendungsbeispiele

2.2.1 Parsing natürlicher Sprache

1961 verwendete Silvio Ceccato ein Definitional Network um einen Parser für natürlichsprachliche Texte zu konstruieren. Hierbei stellte das Definitional Network Entscheidungshilfen zur Verfügung, die es dem Parser erlaubten, auch auf syntaktisch schwierigem Gelände gut zu arbeiten.

2.2.2 KL-ONE

Knowledge Language One (KL-ONE) ist ein System der deskriptiven Logik, das eine Art von Definitional Networks verwendet, um Konzepte zu definierern (s. Abb 2).

3 Assertional Networks

Im Vergleich zu Definitional Networks sind Assertional Networks dazu gedacht, Aussagen und logische Sacherverhalte darzustellen. Grob gesprochen handelt es sich hier also um einen grafischen Ansatz zum Betreiben von Logik.

1889 entwickelte Gottlob Frege eine grafische Baumnotation für die vollständige Prädikatenlogik erster Stufe. Unabhängig davon entwickelte Charles S. Peirce 1880 eine algebraische Notation, die, abgesehen von einigen Symboländerungen (Peano, 1889), bis heute verwendet wird. Peirce war mit seinem Werk nicht zufrieden und begann schon bald nach einer anderen, grafischen Notation zu suchen, die “die Atome und Moleküle der Logik” offen-

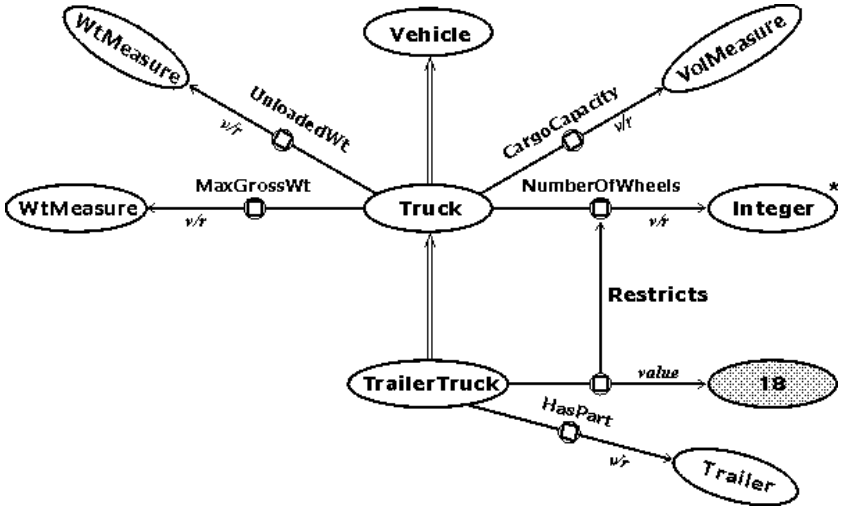

Abbildung 2: Konzeptdefinition `Truck` und `TrailerTruck` in KL-ONE [Sow01]

sichtlicher machen sollte. Hierzu entwickelte er die sogenannten "relationalen Graphen".

3.1 Relationale Graphen

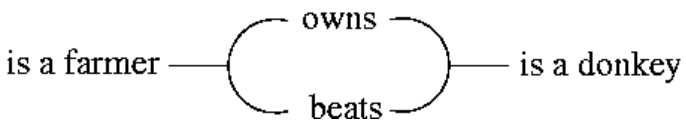

Abbildung 3: relationaler Graph [Sow01]

Relationale Graphen bestehen aus mehreren sogenannten "Identitätslinien" an deren Zweigen beschreibende Prädikate angebracht sind. Damit kann die Aussage des Satzes aus Abb. 3 dargestellt werden als

$$(\exists x)(\exists y)(farmer(x) \wedge donkey(y) \wedge owns(x,y) \wedge beats(x,y)) \qquad (1)$$

Aus Gl. 1 ist ersichtlich, dass ein relationaler Graph lediglich zur Darstellung zweier logischer Operatoren geeignet ist: dem Existenzquantor $\exists$ und der Konjunktion $\wedge$. Negation, Disjunktion, Implikation und der Allquantor können auf diese Art und Weise nicht dargestellt werden. Diese Lösung war

natürlich unbefriedigend. Wäre es möglich gewesen, die Negation darzustellen, so hätten sich alle weiteren Operatoren durch algebraische Umformungen in Verbindung nit der Negation darstellen lassen. 1897 löste Peirce das Problem, indem er das Konstrukt eines Ovals einführte das die Negation des gesamten in ihm eingeschlossenen Subgraphen darstellte. Damit kam Peirce von den relationalen Graphen ab und gründete die existentiellen Graphen.

3.2 Existentielle Graphen

Die Ausdrucksfähigkeit der existentiellen Graphen umfasst die gesamte Prädikatenlogik der ersten Stufe. Mit ihnen ist es möglich, Sätze wie

"If a farmer owns a donkey, then he beats it."

darzustellen:

$$(\forall x)(\forall y)(farmer(x) \wedge donkey(y) \wedge owns(x,y) \rightarrow beats(x,y)) \qquad (2)$$

Gl. 2 wird durch Umformen zu:

$$\neg(\exists x)(\exists y)(farmer(x) \wedge donkey(y) \wedge owns(x,y) \wedge \neg beats(x,y)) \qquad (3)$$

Gl. 3 kann durch den existentiellen Graphen in Abb. 4 dargestellt werden.

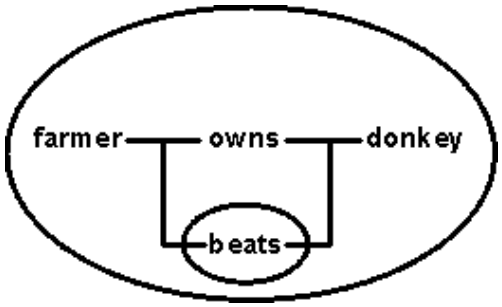

Abbildung 4: Existential graph [Sow01]

3.3 Konzeptuelle Graphen

Konzeptuelle Graphen (CGs) sind existentiellen Graphen auf den ersten Blick sehr ähnlich (Abb. 5) und die offensichtlichen Unterschiede sind eher kosmetischer Natur: Anstatt Ovalen verwenden CGs Rechtecke und die Negation wird explizit dargestellt. Bei näherer Betrachtung unterschieden sich die Modelle jedoch in weiteren Punkten.

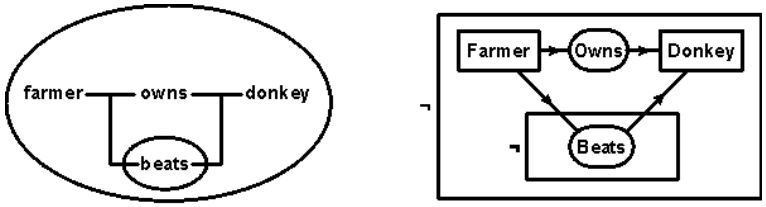

Abbildung 5: existentieller und konzeptueller Graph [Sow01]

3.3.1 Aufbau einfacher Konzeptueller Graphen

Konzeptuelle Graphen sind zweiteilige (bipartite) Graphen, das heißt, sie bestehen aus zwei Arten von Knoten und Verbindungen zwischen Knoten führen immer von einer Knotenart zur anderen. Die beiden Knotenarten sind:

- Konzepte und
- Relationen

3.3.2 Konzepte

Ein Konzept besteht aus zwei Entitäten: seinem Konzepttyp und dem Referenten, der als Instanz des Konzepttyps betrachtet werden kann (Abb. 6). Konzepte können außerdem aus lediglich einem Konzepttyp bestehen. So

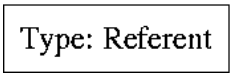

Abbildung 6: Konzeptdefinition [Pet]

stellen die Beispiele in Abb. 7 folgende Aussagen dar:

> "Es gibt eine Person, deren Name John ist."

und

> "Es gibt einen Bus."

Abbildung 7: Konzeptdefinition [Pet]

3.3.3 Relationen

Relationen werden durch drei voneinander abhängige Eigenschaften definiert. Jede Relation hat ähnlich einem Konzept einen Typ. Beispiele für Typen sind unter anderem:

- Agnt (Agent)
- Rcpt (Recepient)
- Ptnt (Patient)
- Dest (Destination)

Ist der Typ einer Relation festgelegt, so folgt hieraus die Valenz der Relation. Diese trifft eine Aussage darüber, wieviele Verbindungen an der Relation festgemacht sind. Handelt es sich hierbei um mehr als zwei Verbindungen, so werden die auf die Relation zeigenden Verbindungen durchnummeriert. Von einer Relation zeigt nur genau eine Verbindung weg. Die Nummerierung der Relationen ist relevant, da der Typ einer Relation festlegt, was für Konzepttypen mit der Relation verbunden werden dürfen. Diese Spezifikation stellt die Signatur einer Relation dar. Beispielsweise besitzt die Relation `Agnt` die Signatur

`<Act, Animate>`,

wobei die letzte Komponente den Typ des Konzepts spezifiziert, auf die der von der Relation wegführende Verbindung zeigt. Abb. 8 zeigt diese Eigenschaften an zwei Beispielen.

3.3.4 Beispiele Konzeptueller Graphen

Die Beispiele aus Abb. 8 seien hier kurz in englischer Sprache wiedergegeben:

> "There exists a person whose Name is John. John is an agent of go, which has a destination which is a city, the name of which is Aalborg."

In besserem Englisch:

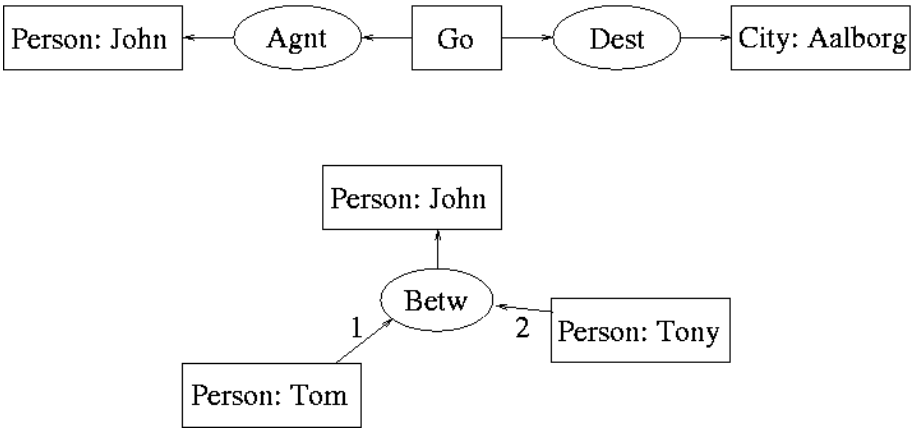

Abbildung 8: Konzeptdefinition [Pet]

"John is going to Aalborg."

Für die Relation mit einer Valenz > 2 ergibt sich:

"John is between Tom and Tony."

3.4 Weitere Beispiele

Als ein weiteres Beispiel für Assertional Networks seien hier komplexe semantische Netze zum Einsatz in der Linguistik und automatisierten Sprachverarbeitung genannt. Um der diesen Themen innewohnenden Komplexität Rechnung zu tragen, werden die Graphenelemente in viele weitere Komponenten unterteilt und mit Ergänzungen versehen. Knoten umfassen beispielsweise die Klassen Sachverhalte, Situationdeskriptoren, formale Entitäten, usw. Relationen umfassen Subordinationsrelationen, Attributierungen, Raum-Zeit-Angaben, Quantoren, Modalitäten und weitere. Mit solchen Mitteln sind unter anderem folgende Aussagen möglich:

"Die Haltestelle befindet sich seit 1988 zwischen Goethe-Strasse und Schillerplatz." (Abb. 9)

"1520 umsegelte der Portugiese Magalhaes mit seinem Schiff die Südspitze von Südamerika." (Abb. 10)

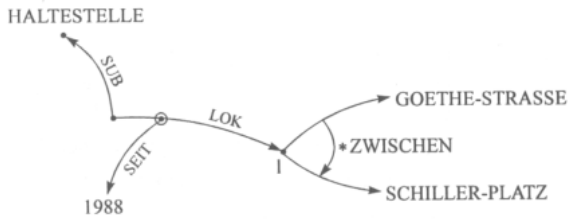

Abbildung 9: Anwendung in der Linguistik (1) [Boe]

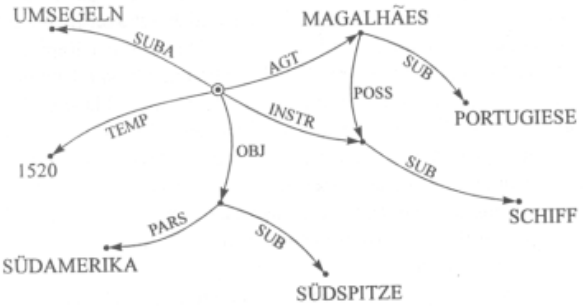

Abbildung 10: Anwendung in der Linuguistik (2) [Boe]

4 Implicational Networks

Implicational Networks stellen einen Spezialfall der behandelten Assertional Networks dar. Die grundliegende Relation in solchen Netzen ist die Implikation. Es ist zwar möglich weitere Relationstypen zu verwenden, diese spielen in Inferenzprozessen jedoch keine Rolle.

Abhängig davon, welcher Interpretationsansatz gewählt wird, werden Implicational Networks als

- belief networks,
- Kausale Netze
- Bayes-Netze oder
- Truth-Maintenance-Systeme

bezeichnet. Es ist sogar möglich, den gleichen Graphen für alle Interpretationsarten zu verwenden. So kann ein kausales Netzwerk auch als ein Bayes-Netz aufgefasst werden, dessen Wahrscheinlichkeiten aus den beiden diskre-

ten Werten 0 und 1 bestehen. Ein Beispiel für ein einfaches kausales Netz zeigt Abb. 11.

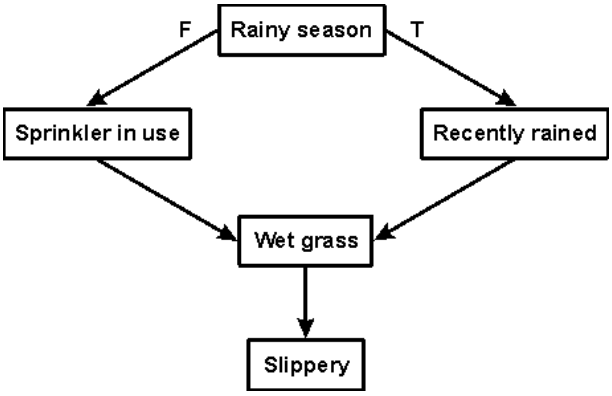

Abbildung 11: Implicational network [Sow01]

4.1 Schließen in Implicational Networks

Innerhalb eines Graphen wie Abb. 11 können verschiedene Arten von Inferenz betrieben werden.

4.1.1 Logische Inferenz

Im Bereich der logischen Inferenz seien Truth-Maintenance-Systeme genannt, die basierend auf Annahmen (ATMS) oder der Gültigkeit aller positiven und der Ungültigkeit aller negativen Vorbedingungen (JTMS) von Fakten in der Lage sind, innerhalb großer Wissensmengen Wissenskonsistenz zu realisieren. Ein TMS propagiert erhaltene Wahrheitswerte vorwärts und rückwärts durch sein Netzwerk. Damit ist es mit TMS möglich, aus vorhandenen Fakten neues Wissen zu extrahieren oder auch widersprüchliche Aussagen innerhalb eines Netzes zu entdecken. Die von TMS praktizierte Art nichtmonotonen Schließens wird “belief revision” genannt.

4.1.2 Probabilistische Inferenz

Auch im Bereich der probabilistischen Inferenz arbeiten TMS. Im Gegensatz zur logischen Inferenz können Wahrscheinlichkeiten, bzw. Wahrheitswerte mehr als zwei diskrete Werte annehmen; sie bewegen sich im Intervall [0..1]. Dies führt zu einer höheren Komplexität und höherem Berechnungsaufwand,

stellt aber Methoden zur Verfügung, mit denen Implicational Networks näher an der Wirklichkeit modelliert werden können.

5 Andere Typen semantischer Netze

5.1 Executable Networks

Die charakteristische Eigenschaft von Executable Networks besteht darin, dass sie in der Lage sind, ihre eigene Struktur zu verändern. Es existieren drei Mechanismen, die in diesem Zusammenhang häufig verwendet werden:

Message passing bezeichnet das Weitergeben von Daten zwischen Knoten. Daten sind hier beispielsweise Marker, Tokens oder Trigger.

Attached procedures sind Programme, die zu einem Knoten gehören und Berechnungen auf den Daten eines Knotens durchführen.

Graphentransformationen durch Programme, die - ähnlich einer chemischen Reaktion - Subgraphen bilden und diese wieder zusammensetzen.

5.1.1 Beispiele

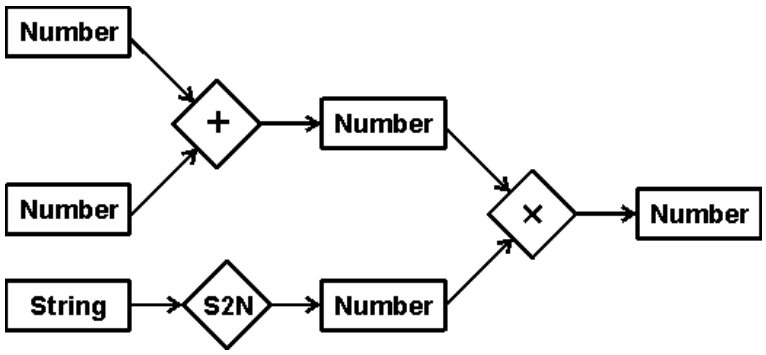

Abbildung 12: Datenflussdiagramm [Sow01]

Einfache Beispiele für Executable Networks sind Datenflussdiagramme (Abb. 12) oder Petri-Netze (Abb. 13). Beide Beispiele werden normalerweise nicht als semantische Netze bezeichnet, aber ihnen ähnlich Techniken werden in prozeduralen semantischen Netzen verwendet. Auch konzeptuelle Graphen erlauben ein Ersetzen von Relationen durch Aktoren, die eine funktionale Erweiterung darstellen.

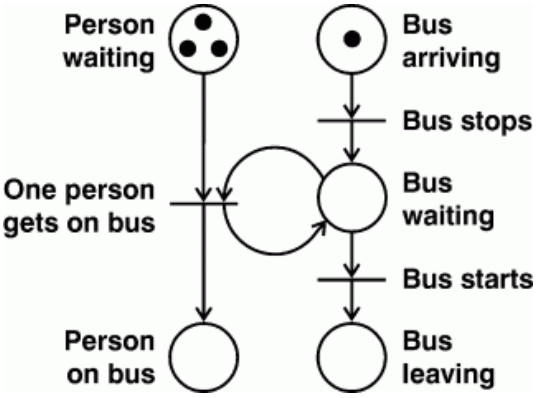

Abbildung 13: Petri-Netz [Sow01]

5.2 Learning Networks

Lernende Netzwerke antworten auf die Präsentation neuer Information mit einer Änderung an ihrer internen Wissensrepräsentation. Dies geschieht im Allgemeinen auf drei mögliche Arten:

Hinzufügen heißt, die neu erhaltene Information in eine Netzwerkdarstellung zu bringen und dieses Netzwerk ohne weitere Änderungen an das bestehende Netzwerk anzufügen. Dies ist die einfachste Form eines lernenden Netzwerks.

Gewichtsanpassung stellt eine weitere Möglichkeit dar, auf äußere Anforderungen zu reagieren. Im Bereich der Implicational Networks könnte dies beispielsweise eine Anpassung der Wahrscheinlichkeiten der Implikationen bedeuten.

Restrukturierung ist die komplexeste Form des Lernens in Netzwerken und beinhaltet ähnlich den Executable Networks eine Änderung der Netzwerkstruktur selbst.

5.2.1 Beispiel

1975 verwendete P. Winston eine Version von relationalen Graphen, um unter Zuhilfenahme eines Programms ein Definitional Network zu erzeugen, das eine Klassifikation eingegebener Daten erstellte.

5.3 Hybrid Networks

Die Unified Modeling Language (UML) kann entlang den Kategorien von semantischen Netzen klassifiziert werden und stellt in dieser Hinsicht ein hybrides Netzwerk dar: Objektdiagramme können als Definitional Networks, Aktivitätsdiagramme und Statecharts als Executable Networks betrachtet werden.

Literatur

[Sow01] Sowa, J.F., *Semantic Networks*, 2001, http://www.jfsowa.com/pubs/semnet.htm.

[Pet] Peterson, U., *Online Course in Knowledge Representation using Conceptual Graphs*, http://www.hum.auc.dk/cg/index.html.

[Boe] Böhm, R., *Wissensrepräsentationssysteme - Semantische Netze*, http://www.inf.hs-zigr.de/ boehm/mr98/lauke/WRS_SN.html.

[Bra01] Bratko, I., *Prolog Programming for Artificial Intelligence*, 3. Ausgabe, 2001.