Grundlagen wissenschaftlichen Arbeitens
PS Programmiersprachen: SS 2005
Ömer Faruk Dural (e0326728)

PROLOG

Das nachfolgende Referat behandelt die Programmiersprache Prolog; Die Entwicklung und die Grundlagen von Prolog sowie einige praktische Beispiele.

1. Historisches

Prolog wurde 1972/73 von den Forschern Alain Comlmarauer und Robert Kowalsky entwickelt. In diesen Jahren wurde also die erste logikorientierte Programmiersprache in Marsielle von Colmarauer und seinen Kollegen entwickelt und zugleich implementiert.

2. Grundlagen

Prolog ist die erste logikorientierte Programmiersprache in Form eines Interpreters für natürlichsprachliche Dialogsysteme. Der Name Prolog leitet sich bezeichnenderweise aus PROgramming LOGic ab. Angewandt und weiterentwickelt wurde Prolog vor allem in den Bereichen der Linguistik und der theoretischen Informatik. In diesen Bereichen ist Prolog immer noch die weit verbreitetste Programmiersprache. Es wurde weiters verwendet, um den Algorithmus zu entwickeln, um Compiler zu schreiben und um eine effiziente Suche im Database durchzuführen.
Im Vergleich zu anderen Programmiersprachen ist Prolog relativ leicht erlernbar, da ihr nur eine Datenstruktur zur Verfügung steht, mit der gleichzeitig Prologprogramme und Daten von Prolog beschrieben werden.
Vorerst allgemein zum Logikprogramming: Das wichtigste Ziel des Logikprogramming ist, es dem Programmierer zu ermöglichen, in einer höheren Stufe als bis dato möglich zu programmieren. Die Hauptunterschiede vom Logikprogrammiermodell zu anderen Programmiermodellen sind folgende: Ein Logikprogramm arbeitet mit Realtionen und die Prädikatsparameter sind symmetrisch.
Nun genauer zum Prologprogramm: Es handelt sich hier um eine Menge prädikatenlogischer Aussagen, wobei eine Aussage einfach oder komplex sein kann.
Eine einfache Aussage ist eine Eigenschaft eines Objekts oder eine Relation zwischen Objekten, welche durch einen Term erster Ordnung dargestellt wird.
Eine komplexe Aussage besteht aus einer Verknüpfung einfacher oder komplexer Aussagen mit logischen Operatoren wie z. B. einer Konjunktion („und“), einer Disjunktion („oder“) oder der Implikation. Die Ausführung eines Programms erfolgt durch eine Anfrage an den Prolog-Interpreter, in den das Programm vorher geladen werden muss. Durch eine solche Anfrage wird die Gültigkeit einer Aussage überprüft.

3. Programmieren in Prolog

Nun einige Beispiele, wie man im Prolog programmieren kann:

Fakten:

Als Fakten werden logische Aussagen bezeichnet, welche Eigenschaften von Objekten oder Relationen zwischen Objekten definieren. Alle wahren Aussagen (ohne Vorbedingung), werden als Fakten angeschrieben:

Kind (stefanie, thomas, michael)
Kind (stefanie, thomas, lisa)

In diesem Beispiel beschreiben die Fakten folgenden Sachverhalt. Stefanie und Thomas haben zwei Kinder namens Michael und Lisa. Sie definieren eine dreistellige Relation zwischen Mutter, Vater und Kind. Das erste Argument wird hier also als Mutter interpretiert, das zweite als Vater und das dritte Argument als Kind der beiden.

Terme:

Terme sind die einzigen Datenstrukturen in Prolog. Anhand von Termen können Prologprogramme und Daten von Prolog beschrieben werden.

Regeln:

Unter Regeln versteht man Anleitungen, um aus vorhandenen logischen Aussagen (Fakten oder Regeln) neue Aussagen zu gewinnen. Eine Regel setzt sich aus einer Konklusion (Regelkopf) und Prämissen (Regelkörper) zusammen. Sie werden durch das Symbol „:-„ getrennt:

Y :- X1 X2 .... Xn

Y = Regelkopf, X1, X2, ...Xn = Regelkörper

Die Regel wird wie folgt gelesen:
Wenn X1 und X2 ...und Xn wahr sind, dann ist Y wahr.

Es ergibt sich daher Folgendes: Ein Fakt ist eine Regel ohne Körper und: Alle Regeln und Fakten mit gleichen Namen und derselben Anzahl an Argumenten beschreiben eine Prozedur.

Anfragen:

Mit Anfragen an das System kann ermittelt werden, ob eine Relation gültig ist bzw. aus den vorhandenen Regeln und Fakten abgeleitet werden kann. Die möglichen Antworten des Prolog-System sind YES – falls die Anfrage gefolgert werden kann – oder NO – falls die Anfrage nicht gefolgert werden kann. Die Ausgabe von NO bedeutet, dass eine Relation aus den vorhandenen Regeln nicht abgeleitet werden kann
Eine Anfrage kann als Regel ohne Konklusion aufgefasst werden.

?-weiblich(michael)
NO

?-kind(michael)
Yes

Atom:

Atome sind üblicherweise Zeichenketten aus Kleinbuchstaben, Großbuchstaben, Ziffern und dem Unterstrich. Die Zeichenketten beginnen mit einem Kleinbuchstaben.

Struktur:

Strukturen sind in Prolog die einzige Möglichkeit, um zusammengesetzte Datenobjekte zu erzeugen. Eine Struktur folgendermassen bestimmt:

Name der Struktur (Atom)
Argumente in runden Klammern, die wiederum Terme darstellen
Arität (Stelligkeit) Anzahl der Argumente

Variablen:

Alle Bezeichnungen, die mit einem Großbuchstaben oder Unterstrich beginnen, werden als Variable erkannt.
Variablen sind typlos, nur lokal gültig und können während der Programmabarbeitung an einen Wert und zu verschiedenen Zeiten an beliebige Terme gebunden werden. In verschiedenen Regeln bedeuten gleiche Bezeichner unterschiedliche Variablen. Sie können auch in Anfragen auftreten. Jedes Auftreten der anonymen Variable kann eine andere Variable bedeuten.

4. Unifikation

Die Unifikation ist die Bindung der Variablen zweier Terme und bestimmter Terme, sodass beide identisch sind. Der Unifikationalalgorithmus ist das Kernstück bei der Bearbeitung von Anfragen. Eine Unifikation zweier Terme T1 und T2 ist in folgenden Fällen möglich: T1 und T2 sind nur Konstanten, wenn beide gleich sind. Ist T1 eine Variable, so kann T2 beliebig sein (T1 wird zu T2 substituiert). Und: Sind T1 und T2 zusammengesetzte Terme, ist eine Unifikation nur

möglich, wenn der Hauptfunktor (Name) gleich ist, wenn beide die gleich Arität besitzen oder wenn alle Argumente, die einander entsprechen, unifizieren.

5. Bestätigen, Schätzen, Unterschiedsliste und Variablen als Platzhalter

Im Folgenden kurz zu der Programmiertechnik in Prolog:

Durch die o. g. Unifikation können Variablen als Platzhalter verwendet werden, der später gefüllt wird.
Mit der Technik des Backtrackings kann ein Ergebnis gefunden werden, wenn schon ein solches existiert. Wichtig ist hier auch, dass die Anfragen in Prolog von links nach rechts durchgeführt werden.

Schätzen und Bestätigen: Eine Guess- und Verifiy-Aussage sieht folgendermaßen aus:
„S“ wird so eingegeben, sodass Guess (S) und Verify (S) überlappen. (X,Y):-member(M,X), member(M,Y).
Es wird gesucht, ob es ein M gibt, das X beinhaltet und anschließend wird bestätigt, ob dies auch Y beinhaltet. Die Listen X und Y werden überlappt, wenn so ein M gefunden werden kann.

Variablen als Platzhalter: Variablen können nicht nur in Regeln und Anfragen verwendet werden, sondern auch als Speicherelement (Platzhalter) eines Objekts. Im Prolog können Variablen auch verwendet werden, um Terme zu speichern. Steht die Variable am Ende der Liste, wird sie als „open list“ bezeichnet, steht sie am Ende, nennt man sie „closed list“.

[a, b|X] open list
Setzt man die Variable X mit [c] zusammen, sieht die Liste so aus: [a, b, c] closed list.

Unterschiedsliste: Eine Unterschiedsliste setzt sich aus 2 Listen zusammen (z. B. A und B). Der Inhalt dieser Listen besteht aus den Elementen, die in A vorkommen. Die Elemente aus B sind nicht enthalten. Diese Listen sind entweder open oder closed sein. Geschrieben wird eine solche Liste als dl(A,B).

dl([a, b, c], [c])

6. Das Cut Prädikat

Das Cut Prädikat funktioniert folgendermassen,bei der Abarbeitung eines Programms können Zweige des Abarbeitungsbaumes im Database abgeschnitten. Sie können als Teilbäume, die keine erforderlichen oder unendlichen Suchwege enthalten, bei der Lösungssuche ausgelassen werden. Das Prädikat kann auch die prozedurale Semantik eines Prologprogramms beeinflussen. Für die Erzeugung effizienter Programme ist die Verwendung dieser Prädikate oft unbedingt

notwendig. Im Prolog wird das Cut Prädikat durch „!“ bezeichnet. Dieses Prädikat wird immer als wahr angesehen, so beeinflusst es nicht die deklatative Semantik eines Prologprogramms. Es gibt zwei verschiedene Arten von Prädikaten, das grüne und das rote Cut Prädikat. Ein in einer Prädikatsdefinition verwendetes Cut-Prädikat wird als „grün“ bezeichnet. Dies ist der Fall, wenn die prozeduerale Semantik des definierten Prädikats ohne Cut-Prädikat übereinstimmt und nur nicht erfolgreiche Suchwege abgeschnitten werden. Daher wird ein grünes Cut-Prädikat nur aus Effizienzgründen benutzt. Jedes Cut-Prädikat, das kein grünes ist, ist ein rotes.
Damit seien die wichtigsten Punkte der Programmiersprache erklärt.

7. Listen:

Im Folgenden kurz zu den Datenstrukturen in Prolog:

Listen sind jene Terme in Prolog, die am häufigsten Verwendung finden. Eine Liste ist eine geordnete Menge von Elementen. Als Elemente kommen dabei sämtliche Terme, also auch wieder Listen, in Frage. Die einfachste Möglichkeit, eine Liste, die aus drei Elementen besteht, zu schreiben, ist folgende:

[x, y, z] Einfache Liste

„[]“ stellt eine leere Liste dar
Die Situation wird speziell, wenn eine Liste einen Head und einen Tail hat ([H|T]).
Der Head ist das erste Element der Liste und der Tail ist das andere Element in der Liste.
Die leere Liste ist die einzige Konstante.

[x, y [z]] Spezielle Liste

Mittels der Unifikation können Head und Tail getrennt werden.

? = [H|T] = [x, y, z]
H : x
T : [y, z]

8. Controlling in Prolog

Ein Algorithmus setzt sich aus logic+control zusammen. „Logic“ steht hier für die Regeln und Anfragen in einem Logikprogramm. Mit „control“ kann man ausdrücken, wie ein Programm die Antwort auf eine Anfrage berechnet. Bei control ist die Anfrageordnung und die Regelordnung zu beachten, Die Anfrageordnung und die Regelordnung beeinflussen das Ergebnis.:
Die Anfrageordnung ist wichtig, da der Prolog-Interpreter die Anfrage von links nach rechts durchführt. Die Regelordnung ist wichtig, da die Ordnung der Regeln und Fakten im Search Tree (Database) relevant ist. Wenn eine Anfrage eingegeben wurde, nimmt der Prolog-Interpreter zuerst jene Anfrage, die am weitesten links steht und dazu die erste entsprechende Regel vom Search Tree im Database.

Literaturverzeichnis

1. The Craft of PROLOG (Logic Programming)
von Richard A. O'Keefe
1990, Verlag: MIT Press

2. The Practice of PROLOG (Logic Programming)
von Leon S. Sterling (Herausgeber)
Verlag: MIT Press
Erscheinungsdatum: 1. Dezember 1990

3. Prolog, Autor: Ulrich Geske
Verlag:Akademie Verlag Berlin
Erschienen: 1993

4. Grundlagen der Logischen Programmierung,
Hans-Joachim Goltz, Heinrich HERRE
Verlag: Akademie Verlag Berlin
Erschienen: 1990

5. Prolog. Logische Programmierung in der Praxis.
von Wilhelm Weisweber
1997 erschienen, 1. Auflage, Gedruckt bei Wiener Verlag, Himberg

6. Essentials of Logic Programming, Christopher John
Hogger, Oxford,
Erschienen: 1990
Verlag: Clarendon Press