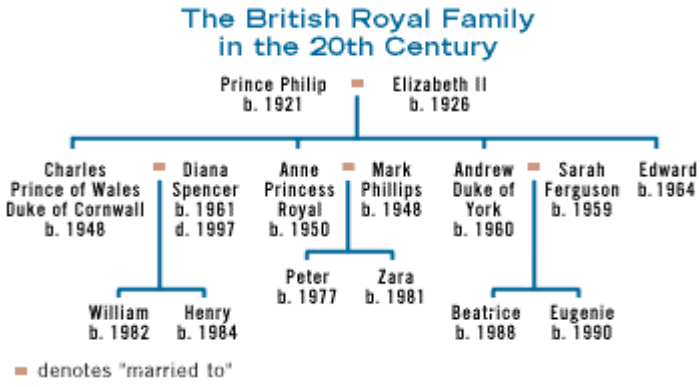
Kapitel 2.
Grundbegriffe: Fakten, Anfragen, Regeln

Anhand eines konkreten (nicht-linguistischen) Gegenstandes wollen wir die ersten Schritte in Prolog angehen, unser erstes kleines Prolog-Programm namens familie.pl verfassen und einige wichtige Grundbegriffe kennenlernen. Die Problemstellung ist so ausgewählt, daß die ihr zugrundeliegende Wissenstruktur völlig einfach nachzuvollziehen ist und wir uns somit darauf konzentrieren können, sie in eine Form zu bringen, die von Prolog verarbeitet werden kann. (Aus diesem Grund findet sich das folgende Beispiel übrigens in fast allen Prolog-Einführungen). Obwohl das Programm zunächst völlig unlinguistisch aussieht, wird in den (Haus)Aufgaben der Tatsache Rechnung getragen, dass die Analyse von Verwandtschaftsbeziehungen und deren Bezeichnung ein weit verbreitetes linguistsches Untersuchungsfeld in der kontrastiven Linguistik ist.

Es geht konkret um einen Familienstammbaum wie den folgenden:



Aus diesem Stammbaum läßt sich eine Vielzahl von unterschiedlicher Information entnehmen, nämlich

1. Information zu Eigenschaften einzelner Personen
2. Information hinsichtlich der Beziehungen, die zwischen einzelnen Personen bestehen.

Zu Punkt eins: Über eine jede Person kann ausgesagt werden, welchen Geschlechts sie ist: Philip ist männlich, Diana ist weiblich usw. und wann sie geboren wurde. (Übrigens: der Stammbaum ist nicht aktuell – auf die Tatsache, dass Familienmitglieder verstorben sind, gehen wir aber nicht ein, und welche Rolle Rittmeister Hewitt in dem Ganzen hatte, wird auch nicht berücksichtigt).

Zu Punkt zwei: Über die Beziehungen zwischen einzelnen Personen können Aussagen gemacht werden: Anne ist mit Mark verheiratet, Sarah ist mit Andrew verheiratet, Charles ist Elternteil von Henry, Peter ist Sarahs Bruder, Eugenie ist Tochter von Andrew, Elisabeth ist Großmutter von Beatrice usw.

Diese und andere Daten sind alle in dem o.a. Stammbaum enkodiert, und wir können sie daraus entnehmen. So gesehen stellt dieser Stammbaum eine Art Mini-Wissensbasis dar. Im folgenden nun soll in kleinen Schritten demonstriert und erklärt werden, wie man diese Daten in genau die Form bringt, die Prolog verarbeiten kann, und wie man somit Anfragen an Prolog stellen kann, die das Programm sinnvoll beantwortet. Es geht also darum, das Wissen so zu kodieren, daß auf eine Anfrage wie z.B. "Wer ist der Großvater von William" die Antwort "Philip" geliefert wird. Dabei soll gezeigt werden, wie man Prolog dazu bringt, mithilfe von Regeln aus einer Reihe von Fakten andere Fakten abzuleiten.

2.1. Fakten

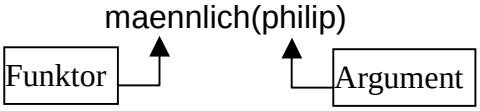
Der erste Schritt besteht darin, Fakten einzugeben. Ein Prolog-Fakt ist eine Feststellung, daß ein Objekt eine bestimmte Eigenschaft hat (*Philip ist männlich*), oder daß zwischen zwei oder mehr Objekten eine bestimmte Beziehung (Relation) besteht (*Elisabeth ist die Mutter von Edward*). Das bedeutet, daß Fakten Aussagen über Eigenschaften von Einzelindividuen oder Beziehungen zwischen Einzelindividuen sind. Dazu gehören in unserem Stammbaum beispielsweise die Geschlechtszugehörigkeit der einzelnen Personen.

Die Geschlechtszugehörigkeit der Personen im Stammbaum

Umgangsprachlich würde man das so ausdrücken: Philip ist männlich, Elizabeth ist weiblich, Charles ist männlich, Diana ist weiblich usw. Leider ist Prolog nicht in der Lage, umgangssprachliche Ausdrücke zu verarbeiten. Eine Aussage wie "Philip ist männlich" muß erst in eine andere Form gebracht werden, um von Prolog verarbeitet werden zu können. 'Männlich' drückt die Eigenschaft oder das Attribut aus, welches dem Individuum Philip zugeschrieben wird. In Prolog wird dieses Fakt auf die folgende Weise notiert:

maennlich(philip).

Das heißt, daß der Ausdruck "Philip ist männlich" in eine eine Funktor-Argument-Struktur übersetzt wird, in der die Eigenschaft oder das Attribut den Funktor stellt und das Individuum das Argument dieses Funktors. Der Funktor wird auch als 'Prädikatsname' bezeichnet, dazu mehr s.u. Der Funktor steht auf der linken Seite, das Argument steht in Klammern rechts daneben. Zu achten ist dabei auf die Groß/Kleinschreibung: es werden hier nur Kleinbuchstaben verwendet:



Analog verfahren wir bei den Geschlechtszugehörigkeit der anderen Personen: "Elizabeth ist weiblich" wird in Prolog durch weiblich(elisabeth) repräsentiert; "Andrew ist männlich" durch maennlich(andrew) usw.

Wenn Ihr das Skript am PC durcharbeitet, könnt Ihr jetzt die **Aufgaben 1 - 3** am Kapitelende angehen

B) Die Beziehungen zwischen einzelnen Personen im Stammbaum

In diesem Abschnitt geht es nicht um Eigenschaften einzelner Personen, sondern um die Beziehungen zwischen jeweils zwei Personen. Wir betrachten zunächst nur die Elternteil- und die Ehegatten-Beziehung, da wir, wie wir weiter unten sehen werden, alle anderen Verwandschaftsbeziehungen (Vater, Mutter, Schwester, Tante usw. usf.) aus diesen beiden ableiten können.

Auch hier beginnen wir damit, die einzelnen Beziehungen zunächst umgangssprachlich zu formulieren, also z.B. "Philip ist Elternteil von Anne". Dieser Satz muß, analog zu dem Satz "Philip ist männlich", auch wieder in eine Funktor-Argument-Struktur übersetzt werden, um von Prolog verarbeitet werden zu können. Er unterscheidet sich aber von "Philip ist männlich" dadurch, daß hier nicht eine Eigenschaft des Einzelindividuums Philip dargestellt ist, sondern die Beziehung, die zwischen den Einzelindividuen Philip und Anne besteht. In diesem Fall fungiert die jeweilige Beziehung als Funktor, welcher dann nicht ein, sondern zwei Argumente hat: elternteil(philip,anne). Genauso verfährt man bei "Elizabeth ist Elternteil von Charles": elternteil(elisabeth,charles). Dabei ist zu beachten, daß die einzelnen Argumente jeweils durch Kommata voneinander getrennt werden.

Ein wichtiger Punkt noch: die Frage nach der Reihenfolge der Argumente. Diese wird nämlich von dem jeweiligen Programmierer festgelegt. Im obigen Fall steht zuerst das Elternteil, danach das Kind, und das entspricht auch in etwa dem umgangssprachlichen Satz "Elisabeth ist Elternteil von Charles", in welchem auch zuerst das Elternteil und dann das Kind genannt ist. Etwas anders verhält es sich bei der Ehegatten-Beziehung: diese ist symmetrisch, d.h. man könnte sowohl "Charles ist Ehegatte von Diana" als auch "Diana ist Ehegatte von Charles" sagen, also könnte das erste Argument entweder der Ehemann oder die Ehefrau sein. In einem solchen Fall liegt es im Ermessen des Programmierers, die Reihenfolge zu bestimmen. Wenn man sich allerdings einmal festgelegt hat, muß die Reihenfolge konsequent durchgehalten werden. Wird also die Ehegatten-Beziehung in Prolog durch die folgende (abstrahierte) Struktur ehegatte(Frau,Mann) eingeführt, kann man nicht an späterer Stelle die Reihenfolge umdrehen und ehegatte(Mann,Frau) daraus machen.

Wenn Ihr das Skript am PC durcharbeitet, könnt Ihr jetzt die **Aufgaben 4 - 7** am Kapitelende angehen

2.2. Anfragen

Neben den Fakten sind die Anfragen die zweite Form der "Aussage" in einem Logikprogramm. Anfragen sind ein Mittel zur Gewinnung von Information aus Logikprogrammen. Während ein Fakt feststellt, daß ein Objekt eine bestimmte Eigenschaft hat oder daß eine bestimmte Beziehung zwischen Objekten besteht, fragt eine Anfrage danach, ob es zutrifft, daß ein Objekt eine bestimmte Eigenschaft hat oder eine bestimmte Beziehung zwischen Objekten besteht.[5] Auf unser Stammbaum-Beispiel bezogen: eine mögliche Anfrage wäre z.B. *Trifft es zu, daß Elizabeth ein Elternteil von Andrew ist ?*:

?- elternteil(elizabeth,andrew).

Da wir diese Aussage als Fakt in die Wissensbasis haben einlesen lassen, lautet die entsprechende Prolog-Antwort:

Yes

Weitere möglich Fragen an unsere Wissensbasis wären z.B.

?- weiblich(sarah).

Yes

?- elternteil(anne, zara).

Yes

Die Anfrage

?- elternteil(henry, anne).

würde allerdings als Antwort

No

erhalten — es ist klar, warum: diese Anfrage läßt sich nicht über ein Fakt in der Wissenbasis beweisen, da es ein solches Fakt gar nicht gibt.

All diese Anfragen beziehen sich ganz konkret auf bestimmte Einzelindividuen. Was aber ist zu tun, wenn wir beispielsweise nicht wissen wollen, ob Charles mit einer bestimmten Person Diana verheiratet ist, sondern ob er überhaupt verheiratet ist, und gegebenenfalls mit wem? In diesem Falle muß die Anfrage mit einer VARIABLEN formuliert werden, die für einen bestimmten, aber noch nicht bekannten Wert steht.

?- verheiratet(charles, X).

Diese Anfrage veranlaßt Prolog, in der Wissensbasis 'nachzugucken', ob es einen Wert für die Variable X gibt, welcher die Aussage verheiratet(charles, X) wahr macht. Einen solchen Wert gibt es in der Tat, entsprechend lautet die Antwort

X=diana

Das heißt, Prolog gibt als Antwort nicht nur an, ob die Anfrage zutrifft oder nicht, sondern auch welcher Wert für die Variable die Aussage wahr macht. Die Schreibmarke bleibt in diesem Falle hinter dem letzten ausgegebenen Zeichen stehen. Durch Eingabe eines Semikolons ';' wird das System veranlaßt, nach weiteren Lösungen zu suchen. Werden keine weiteren Lösungen gefunden (was bei diesem Beispield der Fall ist), erscheint die Meldung No.

VARIABLE in Logikprogrammen (LOGISCHE VARIABLE) unterscheiden sich grundlegend von Variablen in imperativen Sprachen.[6] Eine logische Variable ist ein Name, der im Zuge der Berechnung bestimmten Objekten zugeordnet werden kann. Man sagt dann, die Objekte seien an die Variable GEBUNDEN. Diese Verwendung von Variablen läßt sich mit der Funktion von Pronomina in natürlichen Sprachen vergleichen.

---

[5] Wir werden später sehen, daß Fakten und Anfragen nur Sonderfälle von Regeln sind.

[6] In einer Sprache wie z.B. Pascal ist eine Variable ein reservierter Speicherplatz, an dem durch eine explizite WERTZUWEISUNG ein Wert eines bestimmten Typs gespeichert werden kann. Der Speicherinhalt einer Variablen bleibt solange erhalten, bis er durch eine erneute Wertzuweisung verändert worden ist.

Syntaktisch werden Variable durch Zeichenketten bezeichnet, die mit einem Großbuchstaben oder einem Unterstreichungsstrich beginnen.

Beispiele für die Schreibweise von Variablen:

Nominalphrase, NP, Np, X, _singt usw.

Variable, die mit dem Unterstreichungsstrich '_' beginnen, sind sog. “anonyme” Variable, die wir verwenden können, wenn es im Zusammenhang auf den Wert der Variablen nicht ankommt. Die Bindung an einen konkreten Wert wird bei der Verwendung einer anonymen Variablen nicht ausgegeben. Um beispielsweise herauszufinden, ob Charles verheiratet ist, kann die Anfrage lauten

?- verheiratet(charles,_gattin).
Yes

An dieser Stelle lohnt ein Hinweis darauf, daß bei den anonymen Variablen zu differenzieren ist zwischen einer Variable, die mit einem Unterstrich beginnt (z.B. _X, _ehegatte, _xyz usw.) und dem einfachen Unterstrich '_'. Am besten wird der Unterschied an einem Beispiel verdeutlicht: die komplexe Suchanfrage (siehe dazu mehr weiter unten)

?- maennlich(_), ehegatte(charles,_).

wird als Ergebnis die Ausgabe

Yes

bekommen - die anonyme Variable '_' ist in diesem Fall natürlich an verschiedene Werte gebunden, d.h. sie hat unterschiedliche Referenz. Im Gegensatz dazu die Anfrage:

?- maennlich(_person), ehegatte(charles,_person).

Hier lautet die Antwort

No.

Der Grund: Bei der anonyme Variable '_person' ist es zwar egal, welchen Wert sie erhält, aber dieser muß für beide Beweisziele identisch sein. Da wir in unserem Stammbaum aber die Ehefrauen als zweites Argument aufgeführt haben, wird es ausgeschlossen sein, einen Wert zu finden, der sowohl als Argument von maennlich/1 als auch als zweites Argument von ehegatte/2 auftritt.

2.3. Regeln

In unserer Prolog-Wissensbasis ist nun dargestellt, welche Personen im Stammbaum männlich oder weiblich sind, wer mit wem verheiratet ist und wer Elternteil von wem ist. Wir können entsprechende Anfragen an das Programm stellen. Es stecken aber noch mehr Informationen in dem Stammbaum, z.B.: wer ist Mutter von wem, wer ist Enkel von wem, welche Personen sind miteinander verschwägert usw. Um auch diese Daten in unsere Wissensbasis zu bekommen, könnten wir theoretisch so weitermachen wie gehabt: wir geben lauter Fakten ein wie z.B. mutter(elisabeth,charles) oder enkel(henry,philip) oder bruder(peter,zara).

Es gibt aber eine Lösung, die wesentlich eleganter (und viel weniger schreibintensiv) ist: Man leitet diese Beziehungen aus den bereits eingegebenen Beziehungen und Eigenschaften der Personen ab. 'Ableiten' heißt in diesem Fall, daß wir nicht mehr Aussagen über Einzelindividuen machen, sondern auf der Basis der bereits bekannten Fakten allgemeingültige Regeln formulieren.

Als Beispiel betrachten wir uns die Beziehung kind_von. Es wäre ziemlich mühsam, alle kind_von Beziehungen in der Form von Fakten wie kind(Kind,Elternteil)[7] einzugeben. Außerdem, und das ist der entscheidende Punkt, wissen wir ja, daß zwischen der Beziehung kind_von und der Beziehung elternteil_von ein systematischer Zusammenhang besteht, insofern die kind_von-Beziehung gewissermaßen die Umkehrung der elternteil_von-Beziehung. ist, und diese ist ja bereits in der

[7] Die Argumente dieses Ausdrucks sind die beiden Metavariable 'Kind' und 'Elternteil'. An dieser Stelle könnte natürlich auch einfach kind(K,E) oder kind(X,Y) stehen. Durch die Verwendung dieser Variablen aber wird gleich ein Hinweis auf den Charakter bzw. die Art des Argumentes gegeben. Es bietet sich übrigens immer an, auch bei der Programmierung sog. 'sprechende' Variable zu verwenden, d.h. Variable, die mehr oder weniger selbst dokumentieren , wofür sie stehen. Das fördert die Lesbarkeit von Programmen ungemein.

Prolog-Wissensbasis. Das wollen wir uns zunutze machen, indem wir die folgende Regel formulieren: *Eine beliebige Person K ist Kind von einer beliebigen Person E falls die Person E Elternteil von der Person K ist*. Diese umgangssprachliche Äußerung ist klar. Wie wir allerdings sehen, geht es hier nicht mehr um bestimmte Einzelpersonen, also Charles, Andrew und so weiter, sondern um eine allgemeingültige Aussage. Für das Programm bedeutet das soviel wie: "Wenn wahr ist, daß E Elternteil von K ist, dann ist auch wahr, daß K Kind von E ist." Da die elternteil_von Beziehungen unseres Stammbaumes gänzlich in die Prolog-Wissensbasis eingegeben wurden, kann Prolog leicht überprüfen, für welche Personen die kind_von Beziehung zutrifft. Die Frage ist nur, wie man einen solchen Ausdruck in Prolog übersetzt. Dazu betrachten wir uns den Satz nochmal zeilenweise:

1. *Eine beliebige Person K ist Kind von einer beliebigen Person E*

2. *falls*

3. *die Person E Elternteil von der Person K ist.*

In der ersten Zeile wird ausgesagt: eine beliebige Person K ist Kind von einer beliebigen Person E. Hier handelt es sich um die kind_von Beziehung zwischen zwei Personen, allerdings wird hier keine Aussage über ein Einzelindividuum gemacht, sondern eine Aussage, die allgemeine Gültigkeit hat. Die Notation in Prolog sieht für diesen Fall vor, daß die Argumente in der Funktor-Argument-Struktur als Variable wiedergegeben werden. Diese sind durch Großbuchstaben gekennzeichnet. Die erste Zeile würde also wie folgt wiedergeben:

1. kind(K,E).

In der zweiten Zeile finden wir das Wort *falls*. Dieses ist in Prolog durch Doppelpunkt/Bindestrich repräsentiert (der übrigens eine abstrahierte Form des umgekehrten Implikationspfeiles aus der Logik ('←') darstellt):

2. :–

In der dritten Zeile schließlich findet sich wieder die Beziehung elternteil_von, allerdings auch wieder nicht für Einzelindividuen formuliert:

3. elternteil(E,K).

Von äußester Bedeutsamkeit ist die Tatsache, daß die Variablen der dritten Zeile in engem Bezug zu denen der ersten Zeile stehen – es macht auch umgangsprachlich wenig Sinn, zu sagen: Eine Person K ist Kind von einer Person E falls die Person X Elternteil von der Person Y ist. Die kind_von Beziehung ist, wie gesagt, die Umkehrung der elternteil_von Beziehung, es müssen also dieselben Variablen verwendet werden, damit Prolog die richtigen Zuordnungen treffen kann. Unser umgangsprachlicher Ausdruck hat in Prolog demnach die folgende Form:

Umgangsprachlich	**Prolog**
Eine beliebige Person K ist Kind von einer beliebigen Person E	kind(K,E)
falls	:–
die Person E Elternteil von der Person K ist.	elternteil(E,K).

Der Prolog Ausdruck wird in eine Zeile geschrieben:
kind(K,E):- elternteil(E,K).
Damit ist die kind_von Beziehung aus der elternteil_von Beziehung abgeleitet. Bei der Erstellung von Regeln ist es extrem wichtig, zu prüfen, welche Information vorhanden ist, um sich dann zu überlegen, wie man davon ausgehend neue Information produziert. Dabei sollte man sich immer zunächst umgangsprachlich klarmachen, wie die Dinge eigentlich liegen.

Dazu das nächste Beispiel: wir wollen eine Regel für die mutter_von Beziehung schreiben. Zuerst formulieren wir wieder umgangssprachlich auf der Basis des Bekannten, was eine Mutter eigentlich ist: *Eine beliebige Person M ist Mutter von einer beliebigen Person K falls die Person M weiblich ist und die Person M Elternteil von der Person K ist.* Die Konjunktion *und* wird in Prolog durch ein Komma repräsentiert, so daß wir diesen Satz zeilenweise wie folgt in Prolog übersetzen:

Umgangsprachlich	Prolog
Eine beliebige Person M ist Mutter einer beliebigen Person K	mutter(M,K)
falls	:–
die Person M weiblich ist	weiblich(M)
und	,
die Person M Elternteil von der Person K ist.	elternteil(M,K).

Die mutter_von Beziehung hat in Prolog also diese Form:
mutter(M,K):- weiblich(M),elternteil(M,K).

Wenn Ihr das Skript am PC durcharbeitet, könnt Ihr jetzt die **Aufgaben 8 und 9a** am Kapitelende angehen

Kommen wir zu einem weiteren Beispiel – die Beziehung grossmutter/2. Erneut machen wir uns zunächst umgangssprachlich klar, wie man diese Beziehung durch eine allgemeine Regel erfassen kann. Eine mögliche Formulierung lautet: *Eine beliebige Person G ist Großmutter einer beliebigen Person K, falls die Person G weiblich ist und Elternteil einer Person E, die ihrerseits Elternteil der Person K ist.*

Umgangsprachlich	Prolog
Eine beliebige Person G ist Großmutter einer beliebigen Person K	grossmutter(G,K)
falls	:–
die Person G weiblich ist	weiblich(G)
und	,
die Person G Elternteil einer beliebigen Person E ist	elternteil(G,E).
und	,
die Person E Elternteil der Person K ist.	elternteil(E,K).

Zeilenweise:

grossmutter(G,K):– weiblich(G), elternteil(G,E), elternteil(E,K).

Bevor wir noch weitere Verwandschaftsbeziehungen in Form von Regeln eingeben, erfolgen erst einige grundsätzliche Informationen über Regeln im allgemeinen.

Die allgemeine Form einer Regel

Eine Regel besteht normalerweise aus einem Regelkopf und einem Regelrumpf:

mutter(M,K)	:-	weiblich(M), elternteil(M,K).
Regelkopf		Regelrumpf

verallgemeinert:

a	:-	b_1, b_2, : . . b_n.

wobei a und alle b *Beweisziele* sind. a ist der Kopf der Regel, b_1, b_2 - b_n bilden den Rumpf.

Ein solcher Ausdruck ist eine KLAUSEL (im Grunde genommen eine Hornklausel, siehe dazu das Skript zum Grundkurs *Mathematische und logische Grundlagen der Linguistik*).

Die Regel für mutter/2 besagt, daß der Kopf mutter(M,K) erfüllbar ist, wenn die Bedingungen im Rumpf insgesamt erfüllbar sind, wenn sich also sowohl weiblich(M) als auch elternteil(M,K) für zwei Werte von M und K "beweisen" lassen – z.B. durch das Auffinden entsprechender Fakten in der Wissensbasis.

Fakten und Anfragen können nunmehr als Sonderfälle von Regeln aufgefaßt werden. Ein Fakt ist ein Prädikat, das immer erfüllbar ist. Das ist gleichbedeutend mit einer Regel, deren Rumpf immer erfüllt

wird. In der Tat gibt es in Prolog ein vordefiniertes Prädikat, das immer erfüllt wird. Es handelt sich um das argumentlose true/0. Eine Anfrage mit true/0 ist immer erfolgreich.[8]

?- true.

Yes

Ein Fakt wie weiblich(diana) ist gleichbedeutend mit einer Regel, deren Rumpf true ist: weiblich(diana) **:-** true.

Der Rumpf kann in diesem Fall per Konvention weggelassen werden. In der internen Wissensbank werden jedoch alle Fakten als Regeln behandelt, deren Rumpf aus dem Prädikat true besteht. Man kann dies leicht mit einem vordefinierten Systemprädikat prüfen, das einen direkten Zugriff auf diese Wissensbank erlaubt. Es handelt sich um das Prädikat clause/2, dessen erstes Argument für den Kopf einer Regel steht und dessen zweites für den Rumpf. Vorausgesetzt der Fakt weiblich(diana) ist in der Datenbank enthalten, erhalten wir folgendes:

?- clause(weiblich(diana), Rumpf).

Rumpf = true

Eine Anfrage ist ebenfalls eine Sonderform einer Regel: diese besteht nur aus dem Rumpf.

Prädikat

Ein wichtiger Begriff soll in diesem Zusammenhang eingeführt werden, nämlich der Begriff PRÄDIKAT. Zunächst erfolgt die Definition, dann die Erläuterung:

> **Prädikat**
>
> *Ein Prädikat ist die Menge aller Klauseln, deren Köpfe denselben Funktor (auch: Prädikatsnamen) und die gleiche Stelligkeit haben.*

Alle Prolog-Ausdrücke, die wir kennengelernt haben, also die Regeln einerseits und die Fakten und die Anfragen als Sonderformen der Regel andererseits, haben die Form von KLAUSELN. Die Stelligkeit einer Klausel bezieht sich auf die Zahl der Argumente, die die Klausel hat: maennlich(charles) hat ein Argument, nämlich 'charles', und ist einstellig. ehegatte(philip,elizabeth) hat zwei Argumente, nämlich 'philip' und 'elizabeth', ist also zweistellig, wie beispielsweise auch kind(K,E). Alle Klauseln, die denselben Funktor und dieselbe Stelligkeit haben, bilden ein Prädikat. Die korrekte Schreibweise dafür gibt den Prädikatsnamen (also den Funktor) und, dahinter durch einen Schrägstrich abgeteilt, die Stelligkeit wieder. Die Prädikate in der Datei familie.pl lauten also maennlich/1, weiblich/1, ehegatte/2, kind/2 usw. Die Grenze der Argumentzahl ist übrigens nach oben offen (bei zuvielen Argumenten verliert man allerdings rasch den Überblick); es gibt aber auch nullstellige Prädikate (dazu mehr weiter unten).

Der Gültigkeitsbereich von Variablen

Der Gültigkeitsbereich einer Variablen ist die Regel, in der sie vorkommt, d.h. gleichnamige Variablen, die in verschiedenen Regeln vorkommen, werden als verschiedene Variable behandelt. Betrachten wir dazu die beiden folgenden Regeln:

1. kind_von(P1,P2) :– elternteil(P2, P1).
2. mutter(P1,P2) :– weiblich(P1), elternteil(P1,P2).

Die Variablen P1 und P2 kommen sowohl in Regel 1 als auch in Regel 2 vor. In Regel 2 taucht die Variable P1 zweimal auf: in weiblich(P1) und elternteil(P1,P2). Hier muß tatsächlich Identität vorliegen. Hingegen handelt es sich in kind_von(P1,P2) (in Regel 1) und mutter(P1,P2) (in Regel 2) trotz des gleichen Namens um verschiedene Variable – ihr jeweiliger Gültigkeitsbereich endet bei der individuellen Regel, also bei dem Punkt.[9]

---

[8] Das Gegenstück zu true/0 ist fail/0, eine Prädikat, das immer fehlschlägt. Zur Notation (Funktor/n) siehe den Abschnitt über Prädikate

[9] Um einen Namenskonflikt zu vermeiden, werden bei Verwendung beider Regeln in einem tatsächlich Beweis die namensgleichen Variablen von Prolog so umbenannt, daß sie nicht mehr verwechselt werden können.

Typen von Anfragen

- *Anfragen zur Verifizierung eines Sachverhalts*. Sie verhalten sich wie Entscheidungsfragen (*ja*/*nein*-Fragen), die fragen, ob ein bestimmter Sachverhalt zutrifft oder nicht. Sie enhalten keine ungebundenen (freien) Variablen:
  ?- ehegatte(andrew, sarah).
  fragt, ob Andrew mit Sarah verheiratet ist.
- *Suchanfragen* Diese fragen nach Werten, die an eine oder mehrere Variable "gebunden" werden sollen. Sie entsprechen den Wortfragen. Sie fordern das System dazu auf, einen Wert für eine oder mehrere ungebundene Variable zu finden. Die Anfrage
  ?-elternteil(X,william).
  fragt, wer Elternteil von William ist. Die Antwort wird an die Variable X gebunden.
- *Anfragen zur Ausführung einer Aktion, Befehle*. Sie fordern das System auf, eine bestimmte Operation auszuführen:
  ?- consult(familie).
  ist eine Aufforderung, eine Datei namens familie zu lesen und die darin enthaltenen Fakten und Regeln der aktuellen Wissensbasis hinzuzufügen.

Komplexe Anfragen

Mehrere Anfragen können logisch durch *und* oder *oder* verknüpft werden. Man kann also den Antwortbereich einschränken, indem man überprüfen läßt, ob mehrere Bedingungen gleichzeitig zutreffen. Die *und*-Verknüpfung wird durch ein Komma, die *oder*-Verknüpfung durch ein Semikolon ausgedrückt. Die Anfrage

?- elternteil(Person,_) , weiblich(Person).

fragt nach allen Personen in der Wissensbasis (Variable Person), die zwei Bedingungen erfüllen:

1. Sie müssen Elternteil von irgend jemandem sein (anonyme Variable _).
2. Sie müssen weiblichen Geschlechts sein.

Mit anderen Worten, es wird nach allen Müttern gefragt.

Wenn Ihr das Skript am PC durcharbeitet, könnt Ihr jetzt die **Aufgabe 9b** am Kapitelende angehen

Mit diesem Wissen wollen wir erneut auf das Modell der Prologprogrammierung aus dem ersten Kapitel zurückkommen, welches nun wie folgt konkretisiert werden kann:

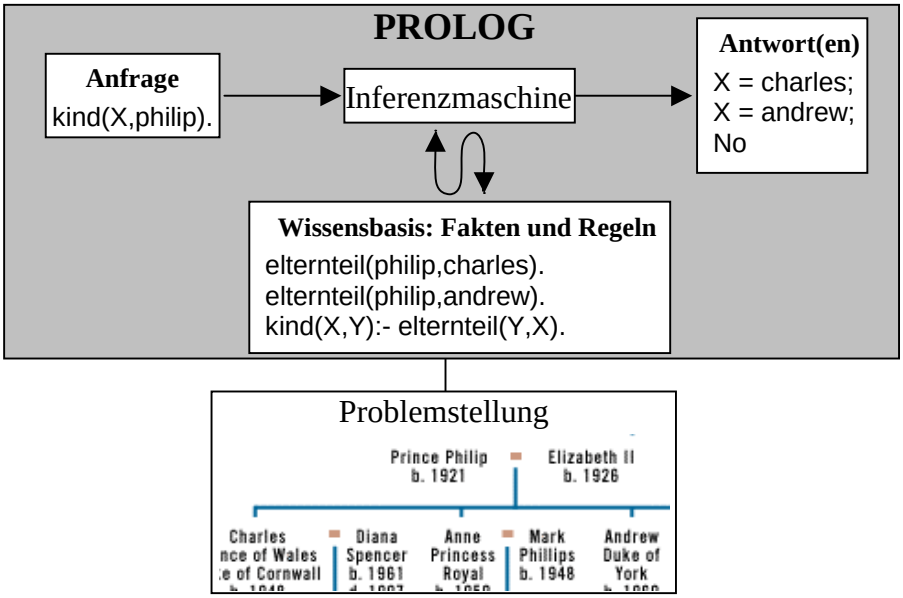


Abb. 1.2. Modell der Prologprogrammierung am Beispiel 'Stammbaum'

Diese Graphik ist nun gut nachzuvollziehen. Die 'Problemstellung' ist in diesem Fall der Stammbaum der englischen Royals. Die in diesem enthaltene Information ist in eine Reihe von Prolog-Fakten und Regeln übersetzt worden, welche in die Prolog-Wissensbasis eingelesen worden sind. Wie wir gesehen haben, ist dieser Vorgang die zentrale Aufgabe der Programmierer. Grundvoraussetzung für eine erfolgreiche Übersetzung ist, daß man sich zunächst Klarheit verschafft über die Art und Weise, die Problemstellung in Aussagen zu erfassen, aus denen durch entsprechende Regeln möglichst ökonomisch andere Aussagen abzuleiten sind. Eine Anfrage, wie in Abbildung 1.2. kind(X,philip), aktiviert die Prolog-Interferenzmaschine, welche systematisch die Wissensbasis durchforstet, um eine geeignete Antwort zu finden. In diesem Fall geht es um die verschiedenen Möglichkeiten, die Variable X an einen Wert zu binden um so die Anfrage zu beweisen. In der Wissensbasis der Abbildung 1.2. bieten sich dafür genau zwei Möglichkeiten an, die entsprechenden Variablenbinddungen werden als Antwort ausgegeben.

2.4. Faustregeln zur Übersetzung von natürlichsprachigen Ausdrücken in Prolog.

Zum Abschluß der zweiten Kapitels sollen nun noch einige Faustregel zur Überführung natürlich-sprachlicher Ausdrücke in Prolog-Ausdrücke erfolgen. Wie genau die sprachlichen Mittel aussehen, mit denen Prolog arbeitet, also eine genauere Spezifikation der formalen Seite dieser Programmier-sprache, folgt weiter unten.

Wir haben gesehen, daß ein Prologprogramm aus einer Menge von Klauseln besteht, die entweder Fakten wiedergeben, d.h. Einzelaussagen über Eigenschaften von Objekten bzw. Beziehungen zwischen Objekten einerseits, und Regeln, d.h. Aussagen über Eigenschaften von Klassen von Objekten bzw. Beziehungen zwischen Klassen von Objekten, die gelten, falls bestimmte Bedingungen erfüllt sind andererseits.

Es ist allerdings nicht immer ganz einfach, Ausdrücke der Alltagssprache, mit welchen wir bestimmte Sachverhalte beschreiben würden, in äquivalente Prologklauseln zu übersetzen.

In dem Satz *Das Pferd ist im (= in dem) Stall* ist einigermaßen klar, daß mit *das Pferd* ein bestimmtes Individuum gemeint ist und mit *dem Stall* ein bestimmter Ort. Es handelt sich in beiden Fällen um Objekte, die wir durch Konstante (z.B. Atome, s.u.) ausdrücken, und die in der Klausel als Argumente vorkommen werden. Die Präposition *in* stellt eine Beziehung zwischen diesen Objekten her, wird also durch einen Funktor in einer Funktor-Argument-Struktur wiederzugeben sein. Ein möglicher Prologausdruck dafür ist also in(pferd, stall).

In dem Satz *Das Pferd ist ein Säugetier* bezeichnet der Ausdruck *das Pferd* nicht ein einzelnes Individuum, sondern eine Klasse, nämlich die Klasse der Pferde im Allgemeinen. Ebenso meint *ein Säugetier* nicht ein individuelles Tier sondern Säugetiere im Allgemeinen. Zwischen beiden besteht eine allgemeine, “regelhafte” Beziehung: jedes Ding, das ein Pferd ist ist auch ein Säugetier. *Pferd* und *Säugetier* sind hier Allgemeinbegriffe und werden als Prädikate wiedergegeben: pferd(X) bzw. saeugetier(X). Der Satz gibt eine Regel wieder: saeugetier(X) :- pferd(X).

Im folgenden werden einige “Faustregeln” zur Übersetzung von alltagsprachlichen Ausdrücken in Prolog gegeben:

- Eigennamen wie *Bart* oder *Lisa* werden zu Konstanten: bart, lisa. Symbolische Konstante werden üblicherweise durch Atome wiedergegeben. Sie können aber auch durch Zeichenketten oder allgemein durch Strukturen ausgedrückt werden (zu den Begriffen ATOM, ZEICHENKETTE und STRUKTUR später mehr).
- Substantive wie *Katze*, oder *Tisch*, die Allgemeinbegriffe bezeichnen, werden zu einstelligen Prädikaten: katze(ophelia),tisch(tisch25).

- Substantive wie *Vater, Kind*, oder *Schwester*, die Beziehungen bezeichnen, werden zu zweistelligen Prädikaten: vater(homer, bart), kind(lisa, marge), schwester(lisa, maggie).[10]
- Bei generischen Substantiven wie Ding, Sache, Umstand können auch Variable gemeint sein. Gleiches gilt für Pronomina. *Wer rastet, der rostet* wäre als rostet(Person):-rastet(Person) wiederzugeben.
- Adjektive wie *rot* oder *bescheiden*, die Eigenschaftsbegriffe bezeichnen, werden zu einstelligen Prädikaten: rot(zora), bescheiden(lisa). Bei Ausdrücken wie *Fredi ist ein weißer Elephant* ist zu bedenken, daß durch die Nominalphrase zwei Eigenschaften ausgedrückt werden: weiss(fredi) und elephant(fredi). *Waldi ist dumm, faul, und gefräßig* ist eine Zusammenfassung von 3 Einzelfakten:
  dumm(waldi).
  faul(waldi).
  gefraessig(waldi).
- Die Komparative von Eigenschaftswörtern bezeichnen Relationen und sind demgemäß als mehrstellige Prädikate darzustellen: kleiner(lisa, bart), schwerer(blei, eisen).
- Es gibt auch Adjektive, die Beziehungen bezeichnen, z.B. *abhängig, unterlegen, zuträglich:* unterlegen(bart,lisa)
- Intransitive Verben wie *spazierengehen* oder *schlafen* werden zu einstelligen Prädikaten: spazierengehen(claudio), schlafen(theodor).
- Transitive Verben wie *sehen* oder *tragen* werden zu zweistelligen Prädikaten: sehen(bart, lisa), tragen(marge, kleid).
- Bitransitive Verben wie *geben* werden zu dreistelligen Prädikaten: geben(lisa, bart, buch) ist die Wiedergabe von *Lisa gibt Bart ein Buch*.
- Bei Passivsätzen kommt es darauf an, ob das Agens (das, was im Aktiv als Subjekt erscheint) ausgedrückt wird oder nicht. Wird das Agens ausgedrückt, bietet es sich an, den Satz ins Aktiv zu transformieren und dann zu übersetzen, z.B. *Bart wird von Lisa unterstützt = Lisa unterstützt Bart* → unterstuetzt(lisa, bart). Wird das Agens nicht ausgedrückt, verhält sich das Verb wie ein intransitives Verb. Am einfachsten verwendet man das Partizip als Prädikat: *die Socken werden gewaschen* → gewaschen(socken).
- Das Hauptverb wird gewöhnlich zum Kopf einer Regel, die übrien Bestandteile gehören zum Rumpf.
- Das Verb *sein* in seinen verschiedenen Formen hat gewöhnlich keine Prologentsprechung, statt dessen wird das darauf folgende Substantiv, Adjektiv, oder Partizip zum Hauptprädikat: *Homer ist ein Idiot*: idiot(homer).
- Präpositionen bezeichnen Relationen (zwei- und mehrstellig): auf(buch, tisch) für *Das Buch liegt (ist, steht, befindet sich) auf dem Tisch* in(anzug, schrank)*: der Anzug ist im Schrank*; zwischen(bremen, frankfurt, hamburg) für *Bremen liegt zwischen Frankfurt und Hamburg*.

Wenn Ihr das Skript am PC durcharbeitet, könnt Ihr jetzt die **Aufgabe 10** am Kapitelende angehen

[10]Solche Substantive können jedoch auch als Namen *(der Vater, das Kind, die Schwester*, z.B. in einem (nicht besonders geglückten) Ausdruck wie 'Die Mutter liebt den Vater': liebt(mutter,vater) oder als einstellige Prädikate verwendet werden (*Hans ist Vater geworden:* vater(hans)).

Übungen zu Kapitel 2

Aufgabe l: Übersetzt (auf Papier) die folgenden Sätze in geeignete Prolog-Fakten.
Charles ist reich.
Sarah ist rothaarig.
Henry ist minderjährig.
Anne ist geschieden.

Aufgabe 2: Übersetzt (auf Papier) die folgenden Prolog-Fakten in umgangsprachliche Sätze:
moppelig(andrew).
unverheiratet(edward).
klein(elizabeth).

Aufgabe 3: Legt eine Datei familie.pl und gebt in diese die Fakten über die Geschlechts-zugehörigkeit der einzelnen Personen des Stammbaumes ein

Aufgabe 4: Übersetzt (auf Papier) die folgenden Sätze in geeignete Prolog-Fakten:
Charles liebt Polo.
Eugenie und Beatrice sind Geschwister.
Sarah schreibt Kinderbücher.

Aufgabe 5: Übersetzt (auf Papier) die folgenden Prolog-Fakten in umgangssprachliche Sätze:
liiert_mit(charles,camilla).
besitzt(elizabeth,hunde).
beruf(andrew,pilot).

Aufgabe 6: Warum wird es bei den folgenden Prolog-Fakten (mit Bezug auf den konkreten Stammbaum) Komplikationen geben?
elternteil(philip,charles).
elternteil(elisabeth,andrew).
elternteil(anne,zara).
elternteil(henry,diana).
elternteil(andrew,beatrice).

Aufgabe 7: Vervollständigt in der Datei familie.pl die Daten über die Elternteil_von- und die Ehegatten-Beziehungen der einzelnen Personen des Stammbaumes. Laßt anschließend den Inhalt der Datei mit consult/1 in die Prolog-Wissensbasis einlesen.

Aufgabe 8: Wie würdet Ihr (auf Papier) die folgenden Ausdrücke in Prolog wiedergeben:
Magda mag Max
Mädchen mögen mollige Männer

Aufgabe 9a: Erweitert die Datei familie.pl um die Regeln für die Prädikate kind_von/2, mutter/2, und vater/2 und überprüft diese anschließend.

Aufgabe 9b: Erweitert die Datei familie.pl um die Regeln für die Prädikate schwester/2, bruder/2, cousin/2, cousine/2, tante/2, onkel/2, schwager/2, schwaegerin/2, großvater/2 und großmutter/2 und überprüft diese anschließend.

Aufgabe 10: Legt eine Datei namens glueck.pl an und übersetzt die folgende Information in Prolog: *Ursula ist hübsch. Norbert ist reich und gutaussehend. Karla ist reich und stark. Pierre ist stark und gutaussehend. Bruno ist gütig und stark. Alle Männer lieben hübsche Frauen. Alle reichen Männer sind glücklich. Jeder Mann, der eine Frau liebt, die ihn liebt, ist glücklich. Jede Frau, die einen Mann liebt, der sie liebt, ist glücklich. Karla liebt jeden Mann, der sie liebt. Ursula liebt jeden Mann, der sie liebt, vorausgesetzt, daß er entweder reich und gütig oder gutaussehend und stark ist.*

(a) Zeigt mit Prolog, wer hier glücklich ist.
(b) Fügt eine plausible Regel hinzu, die alle glücklich macht.
(c) Sucht in der neuen Version alle Fälle von gegenseitiger Zuneigung, einseitiger Zuneigung und von Rivalitäten (zwei oder mehr Personen lieben die gleich Person).