Kapitel 1.
Einleitung: Computerlinguistik und Prolog

1.1. Computerlinguistik

Die Bezeichnung COMPUTERLINGUISTIK hat sich in jüngerer Zeit als Sammelbegriff für eine Vielzahl ganz unterschiedlicher Projekte und Programme etabliert, bei denen es im weitesten Sinne um die Verarbeitung von Sprache und sprachlichen Daten auf Computern geht. Darunter fallen so unterschiedliche Anwendungen wie Synthese und Analyse gesprochener Sprache; Programme zur maschinellen Übersetzung; Programme zur Modellierung sprachlicher Kompetenz auf den verschiedenen sprachlichen Ebenen und so weiter. Für ein Teilgebiet der Computerlinguistik findet sich auch die Bezeichnung *linguistische Datenverarbeitung*, wobei der Computer im wesentlichen als Hilfsmittel zur Verarbeitung von Sprachdaten dient, beispielsweise zur Erstellung von Wortlisten (Konkordanzen,[1] Häufigkeitswörterbücher, rückläufige Wörterbücher[2] etc.) oder zur statistischen Analyse und Beschreibung von sprachlichen Ausdrücken in Texten.

Im vorliegenden Skript ist der Schwerpunkt allerdings anders gelagert: Hier soll die Computerlinguistik als Teildisziplin der Linguistik verstanden werden, deren Ziel die Implementierung linguistischer Theorien oder Teiltheorien auf Computern ist. Ein konkretes Beispiel dafür ist die Implementierung einer Phrasenstrukturgrammatik. In diesem Zusammenhang ist die Computerlinguistik ein wissenschaftlicher Ansatz, der im Spannungsfeld von Disziplinen wie der Linguistik, der Informatik und der künstlichen Intelligenz, der Wissensrepräsentation, den Kognitionswissenschaften usw. angesiedelt ist. Eine genau Trennung ist bei diesen Fachrichtungen nicht möglich, da die Grenzen zwischen ihnen fließend sind. Was nun ist die primäre Zielsetzung der Computerlinguistik in diesem Sinne, welchem Zweck also dient die Umsetzung linguistischer Theorien auf Computern? Im Grunde werden zwei primäre Aufgabenbereiche verfolgt:

- Zum einen geht es um das eher praktische Ziel, auf der Basis von bestimmten Grammatiktypen und -fragmenten Programmsysteme zur maschinellen Sprachverarbeitung zu entwickeln, die zu einem bestimmten Nutzen eingesetzt werden können. Dazu gehören beispielsweise Systeme zur maschinellen Übersetzung, Dialogsysteme, Systeme für die computergestützte Lexikographie, Systeme zur Stil- und Strukturanalyse, automatisches Erstellen von Inhaltsangaben und so weiter. Dieser Aufgabenbereich wird gemeinhin zur ANGEWANDTEN COMPUTERLINGUISTIK gezählt.
- Zum anderen, im Bereich der THEORETISCHEN COMPUTERLINGUISTIK, geht es darum, diejenigen (im wesentlichen algorithmischen)[3] linguistischen Theorien und Modelle, die die Grundlage der Implementierung darstellen, auf ihre Adäquatheit hin zu überprüfen und sie im Sinne der Modellverbesserung zu modifizieren. Hier erfüllt der Computer eine wichtige Funktion, da auch hochgradig komplexe Modelle zu entsprechend aufbereiteten Datenmengen wie beispielsweise Lexika in Beziehung gesetzt und mit diesen gestestet und ausprobiert werden können. Nicht zu vergessen ist bei diesem Ansatz die Tatsache, daß er die Linguisten zwingt, präzise und systematisch zu arbeiten: es ist hier nicht möglich, bestimmte (möglicherweise unbequeme) Probleme im Formalismus zu vernachlässigen oder deren Lösung 'auf später' zu verschieben.

---

[1] Alphabetische Verzeichnisse der (inhaltlich verschiedenen) Wörter in einem Text mit Angabe der Belegstellen.

[2] Alphabetische Verzeichnisse der Wörter einer Sprache, jedoch nach dem Ende eines Wortes; zuerst kommen alle Wörter, die auf den Buchstaben 'a' enden, dann die mit 'b', etc. Sind sind nützlich beispielsweise für morphologische Analysen, denn alle Wörter mit der gleichen Endung finden sich an der gleichen Stelle.

[3] Hier lohnt ein Hinweis auf die Tatsache, daß die Bezeichnung 'algorithmische Linguistik' im Grunde dem entspricht, was durch das engl. *computational linguistics* ausgedrückt wird. Die Bezeichnungen Computerlinguistik und *computational linguistics* sind keine 1–zu–1 Entsprechungen voneinander: *computational* bedeutet soviel wie 'berechenbar' oder eben 'algorithmisch'.

1.2. Prolog

Der Name *Prolog* ist abgeleitet aus *Programming in Logic*. Diese Bezeichnung ist durch die Tatsache begründet, daß die Verarbeitung von Daten in Prolog auf einem Mechanismus (nämlich dem sog. 'Resolutionsverfahren') fußt, der auf einem Teilbereich der Prädikatenlogik basiert. Durch die Eignung für den oben diskutierten. Aufgabenbereich erfreut sich Prolog in Bereichen wie der Künstlichen Intelligenz und der algorithmischen Linguistik weiter Verbreitung. Prolog unterscheidet sich in seiner Arbeitsweise ganz wesentlich von konventionellen Programmiersprachen wie *BASIC*, *C* oder *Pascal*. Letztere sind sog. *imperative* Sprachen (vgl. IMPERATIV = 'Befehlsform'). Sie werden so genannt, weil Programme in diesen Sprachen aus Anweisungsfolgen bestehen, mit denen dem Computer im Detail mitgeteilt wird, welche Aktionen in welcher Reihenfolge auszuführen sind, um ein Problem zu lösen.

Prolog ist im Gegensatz dazu eine *deklarative* (bzw. *prädikative*[4]) Sprache. 'Deklarativ' deshalb, weil ein Programm nicht aus einer Anweisungsfolge, sondern aus einer detaillierten *Beschreibung* (durch "Aussagesätze", vgl. engl. *declarative clause*) des Problems besteht. Programmieren wird bei einer deklarativen (prädikativen) Sprache als Beweisen in einem System von Tatsachen und Schlußfolgerungen aufgefaßt. In Prolog formuliert man sein eigenes Wissen über das Problem, und der Computer versucht, mithilfe dieses Wissens selbständig eine Lösung des Problems zu finden.

Informell könnte man den Unterschied wie folgt darstellen:

Imperativ/Prozedural

1) Führe X aus
2) Führe Y aus
3) Gehe zurück zu 1) usw.

Deklarativ/Prädikativ

X ist wahr.
Y ist wahr.
Wenn X und Y wahr sind, ist auch Z wahr.

In der Sprache der Logik, auf der Prolog basiert, besteht das Wissen aus einer Menge von *wahren Aussagen* und *Regeln*, die wahre Aussagen in wahre Aussagen überführen. Das heißt, das Prolog in der Lage ist, über Regeln aus bekannter Information neue Information abzuleiten. Gibt der Benutzer eine *Behauptung* ein, so versucht das Prolog-System diese auf der Grundlage des verfügbaren Wissens zu beweisen.

Prolog ist aus mehreren Gründen gut für linguistische Zwecke geeignet:

1. Prolog ist primär keine Sprache zur Durchführung von Berechnungen als Manipulation von Zahlen, sondern eine Sprache zur Manipulation von einfachen und komplexen Symbolen; Zahlen sind nur eine besondere Art von Symbolen. Es liegt in der Hand des Anwenders, wie bestimmte Symbole zu interpretieren sind. Beispielsweise ist ein Prolog-Ausdruck wie np(det(der),n(mann)) für Prolog nur ein komplexes Symbol ohne weitere Bedeutung. Im Zusammenhang mit einem Syntaxanalyseprogramm kann dieser Ausdruck jedoch zur Repräsentation der Phrasenstruktur der Wortfolge *der Mann* verwendet werden.
2. Mit Bezug auf die weiter oben angesprochene Zielsetzung der theoretischen Computerlinguistik, also der Implementierung linguistischer Theorien, kann folgendes gelten: Eine Grammatik wird aufgefaßt als Theorie einer Einzelsprache, insofern die Grammatik beispielsweise u.a. spezifiziert, welche Eigenschaften Wortfolgen aufweisen müssen, damit diese als grammatisch gelten können. Eine Grammatik besteht in der Regel aus einer Menge von FAKTEN (beispielsweise über die Zugehörigkeit von Wörtern zu Wortarten: *der* ist ein Artikel, *mann* ist ein Nomen) und REGELN (beispielsweise bezüglich der Zusammensetzung von Sätzen aus Syntagmen: z.B S $\rightarrow$ NP, VP). Dies gilt unabhängig davon, ob man die Grammatik zur Analyse von Ausdrücken oder zu ihrer Synthese (oder Generierung) verwenden will.

[4] Der *Duden "Informatik"* (S. 547) verwendet anstelle von "deklarativ" die Bezeichnung "prädikativ".

Worum es bei Prolog in erster Linie geht, ist also die präzise Umsetzung der der jeweiligen Problemstellung zugrundeliegenden Informationsstrukturen in entsprechende Fakten und Regeln, die Verarbeitung dieser Fakten und Regeln wird dann automatisch vom Programm übernommen und muß nicht, wie bei einem imperativen Programm, schrittweise angegeben werden.

1.2.1. DIE ARBEITSWEISE VON PROLOG

Ein PROLOGPROGRAMM, bzw. allgemein ein LOGIKPROGRAMM, besteht aus einer Menge von Aussagen, welche die Eigenschaften von Objekten und die Beziehungen oder Relationen zwischen diesen Objekten definieren. Die Aussagen eines Prologprogrammes sind entweder Aussagen über individuelle Gegebenheiten (z.B. *Fido ist ein Hund*, *Fido ist größer als Miezie* oder *Das Genus von 'Mädchen' ist Neutrum*), oder Verallgemeinerungen von solchen Aussagen (z.B. *Menschen sind sterblich*, *Hunde sind größer als Katzen*, *eine Sprache mit Verbendstellung weist auch Postposition auf*). Aussagen über individuelle Gegebenheiten werden FAKTEN genannt, die Verallgemeinerungen sind REGELN.

Die Menge der Fakten und Regeln, die ein Logikprogramm ausmachen, sind in einer internen Datenbank gespeichert, die auch WISSENSBASIS genannt wird. Diese Wissensbasis ist sozusagen eine 'Mini-Welt', ein Modell eines Ausschnitts aus der Wirklichkeit.

Die 'Berechnung' eines Logikprogrammes ist die ABLEITUNG der Menge der Aussagen, die aus dem Programm gefolgert werden können. Ein Prologsystem muß also über einen Mechanismus zur Ableitung von Aussagen aus einem Programm verfügen. Dieser Mechanismus basiert, wie schon erwähnt, auf einem spezifischen Beweisverfahren der Prädikatenlogik (worauf an dieser Stelle nicht näher eingegangen werden soll). Man nennt diesen Mechanismus INFERENZMASCHINE, von engl. *inference engine*, wobei Inferenz hier als Schlußfolgerung zu verstehen ist.

Stellt nun ein Benutzer eine entsprechend formulierte Anfrage an Prolog, so wird die Inferenzmaschine aktiv und versucht, diese Anfrage durch die systematische Abarbeitung von Aussagen aus der Wissensbasis zu beweisen und auf diese Art zu einer (oder mehrerer) Antwort(en) zu gelangen. Der Programmablauf bei Prolog kann zunächst wie folgt schematisch dargestellt werden:

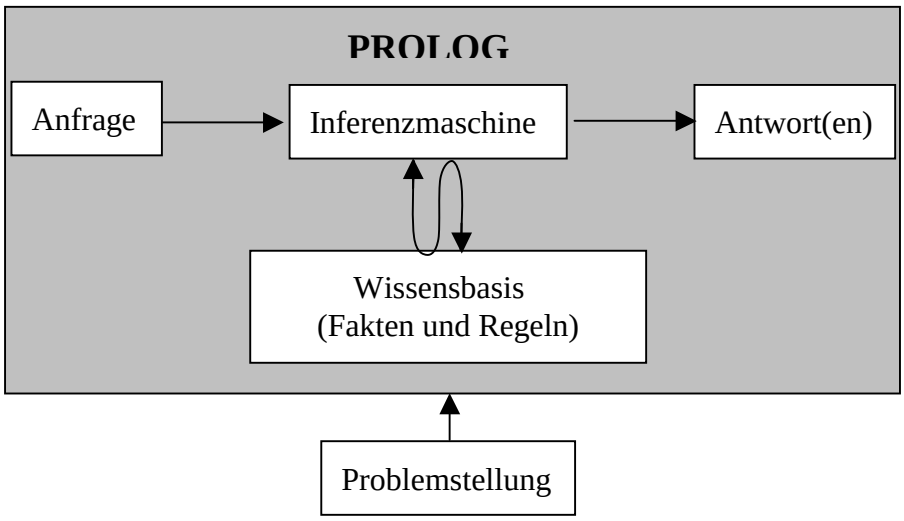

Abb. 1.1. Modell der Prologprogrammierung

Diese Abbildung ist so zu verstehen: eine (beliebige) Problemstellung – z.B. eine Phrasenstrukturgrammatik für das Englische – wird als Menge von Regeln und Fakten in Prolog-Aussagen übersetzt. Dieser Punkt ist die zentrale Aufgabe der Programmierer. Wird eine Anfrage an das Programm gestellt, versucht die Inferenzmaschine auf der Basis eines prädikatenlogischen Algorithmus diese Anfrage durch die Abarbeitung der Wissensbasis zu 'beweisen', um so eine bzw. mehrere mögliche Antworten zu erhalten. Diese Darstellung wird sehr viel anschaulicher, wenn sie im nächsten Kapitel an einem konkreten Beispiel erneut aufgegriffen wird

Dieses Skript soll eine erste elementare Einführung in bestimmte Fragestellungen der theoretischen Computerlinguistik in dem weiter oben spezifizierten Sinne sein. Dabei werden zwei Ziele verfolgt:

- Einerseits geht es um die Vermittlung von Grundkenntnissen der Programmiersprache Prolog, die sich für die oben genannten Aufgaben gut eignet und in der modernen Computerlinguistik bzw. der Künstlichen Intelligenz recht verbreitet ist.
- Zum anderen geht es darum, anhand konkreter Programmierbeispiele exemplarisch einige der zentralen Probleme und Fragestellungen der theoretischen Computerlinguistik nachzuvollziehen und in der Praxis umszusetzen.

Idealerweise sind diese beiden Zielsetzungen so miteinander verknüpft, daß sich die Aneignung der Prolog-Grundlagen aus den Problemstellungen der computerlinguistischen Implementierungen ergibt. Diesem Anspruch versucht das Skript, so weit es geht, gerecht zu werden. Für die notwendige Sicherheit im Umgang mit Prolog müssen aber auch bestimmte Übungen durchgeführt werden, die mehr oder weniger den Status von Fingerübungen haben und deren Nutzen für die zu erstellenden Programme eher mittelbar ist.