6 Entwicklung und Implementierung des vorgestellten Modells

6.1 TFS als experimentelles System

Constraintbeschreibungen — d.h. Typenhierarchien und Typenconstraints — lassen sich als Programm auffassen: Ähnlich wie im Paradigma der *logischen Programmierung* logische Beschreibungen eines Gegenstandsbereiches durch Hornformeln als Programm interpretiert werden, lassen sich Typenhierarchien und Typenconstraints direkt zur Berechnung der Strukturen verwenden, die durch sie beschrieben werden. Der "Mechanismus", der diesen Berechnungen zugrunde liegt, beruht auf einem mathematischen Kalkül und wurde im Abschnitt 1.5 *"Resolution"* beschrieben.

Um Constraintbeschreibung und Kalkül auf einem Computer ausführen zu können, müssen diese in eine Form gebracht werden, die durch einen Rechner manipuliert werden kann. Eine Möglichkeit ist es, den Kalkül direkt in ein Programm umzusetzen, das die Constraintbeschreibung interpretiert. Ein Beispiel für eine solche informatische Umsetzung ist TFS (*The Typed Feature Structure Representation Formalism*)[330], das System, das für die Umsetzung der vorgestellten Grammatik verwendet wurde.

Der vorgestellte Kalkül läßt sich also als Spezifikation der "Programmiersprache TFS", die in den letzten Kapiteln angegebene Grammatik als "TFS-Programm" verstehen. Die Implementierung war demnach schon Thema der vorangehenden Betrachtungen, so daß sich die Frage nach der Motivation des vorliegenden Kapitels stellt. Die Antwort ergibt sich aus der Komplexität der vorgestellten Grammatik: TFS folgt bei seinen Berechnungen einer sehr allgemeinen Strategie und kann Abhängigkeiten zwischen den Merkmalen, die sich aus den Constraints und der Art der Anfragen ergeben, nicht berücksichtigen. Daraus resultiert in den meisten Fällen ein sehr zeit- und speicheraufwendiges Backtracking. Obwohl TFS bei unbegrenzter Zeit und unbegrenztem Speicher die meisten Constraintbeschreibungen berechnen kann,[331] lassen sich in der Praxis daher nur kleine, experimentelle Grammatiken wirklich ausführen.

TFS ist damit als experimentelles System zu verstehen: Es macht die Rolle von Constraintsystemen als Formalismus zur eleganten, deklarativen Spezifikation von Grammatiken deutlich, indem es den Zusammenhang zwischen Spezifikation und formaler Semantik aufzeigt. Beschreibungen kleiner

330 vgl. Emele und Zajac 1990a, Emele und Zajac 1990b, Emele 1991, Emele 1993, Emele 1994 und Zajac 1992.

331 TFS benutzt keine faire Strategie bei der Traversierung der Ableitungsräume, sondern Tiefensuche. Bestimmte Formen der Rekursion können daher zu Endlosableitungen führen.

Sprachausschnitte können direkt überprüft werden, ohne durch zusätzliches strategisches Wissen verfälscht zu werden. Für die praktische Anwendung von Grammatiken bei der Analyse und Generierung von Sprache hingegen empfiehlt sich die Verwendung spezieller, für diese Zwecke optimierter Systeme wie ALE[332] oder PAGE[333].

6.2 Strategien für die Implementierung komplexer Grammatiken

Die Grammatik des Japanischen, die in dieser Arbeit entwickelt wurde, ist zu komplex, um direkt durch TFS interpretiert werden zu können. Es wurden deshalb verschiedene Strategien angewandt, um den Aufwand der Berechnungen zu verringern. Alle Strategien beziehen sich auf die Implementierung des Ableitungskalküls in TFS. Diese beruht auf Backtracking: Der Zustandsraum, der durch eine Anfrage und den Kalkül aufgespannt wird, wird durch Tiefensuche traversiert. Enthält die Anfrage mehrere Teilprobleme, kommt es daher zu einer Multiplikation ihres Zeitaufwands. Ein Satz mit einem Adjektiv, einem Adverb und einem Verb beispielsweise enthält drei morphologisch komplexe Formen. Auch wenn jede dieser Formen einfach ist, ist der Aufwand bei der Analyse des Gesamtsatzes erheblich.

Um den Rechenaufwand gering zu halten, bieten sich zwei Ansätze an:

1. Vereinfachung der Anfrage,
2. Vereinfachung der Constraintbeschreibung.

Bei der Entwicklung von Grammatiken empfiehlt sich die Zerlegung in Subsysteme, die getrennt oder inkrementell entwickelt werden.

6.2.1 Vereinfachung der Anfrage

Für die Berechnung einfacher Probleme ist die Rechenkapazität von TFS ausreichend. Einzelne Aspekte unserer Grammatik lassen sich damit durch TFS alleine berechnen. Es können verschiedene Strategien angewandt werden, um Anfragen einfach zu halten:

332 vgl. Carpenter und Penn 1999

333 Die Module, aus denen sich das PAGE-System zusammensetzt, werden in Uszkoreit et al. 1994, Kiefer und Fettig 1995, Kiefer und Scherf 1994, Krieger und Schäfer 1994a, Krieger und Schäfer 1994b, Krieger und Schäfer 1994c und Backofen und Weyers 1994 dokumentiert.

1. *Beschränkung auf ein Problem pro Anfrage*
   Morphologische Formen, syntaktische Schemata etc. lassen sich anhand von Anfragen untersuchen, die neben dem Aspekt von Interesse keine weiteren aufwendigen Probleme enthalten.

2. *Zerlegung komplexer Probleme in Teilprobleme*
   Diese Strategie ist ein Spezialfall der zuletzt genannten: Komplexe Anfragen können in mehrere Anfragen zerlegt werden. Beispielsweise lassen sich Satzgefüge in mehreren Schritten analysieren oder die Berechnung morphologischer Strukturen von der Berechnung syntaktisch-semantischer trennen. In beiden Fällen ist zur Integration der Ergebnisse ein weiterer Schritt notwendig, der als *"Vorberechnen von Teilstrukturen"* an späterer Stelle erklärt wird.

3. *Hinzunahme zusätzlicher Informationen*
   Die Anfragen an TFS können durch zusätzliche Informationen angereichert werden. Da vorgegebene Informationen nicht mehr durch Backtracking berechnet werden müssen, läßt sich mit diesem Mittel der Suchraum verkleinern und damit der Rechenaufwand unter Umständen drastisch verringern. Sollen beispielsweise komplexe Sätze untersucht werden, können Teile ihrer morphologischen oder syntaktischen Struktur vorgegeben werden.

6.2.2 Vereinfachung der Constraintbeschreibung

Der Aufwand von Berechnungen durch TFS resultiert nicht nur aus der Struktur der Anfrage, sondern auch aus der Anzahl alternativer Typen und Typenconstraints, die für eine Berechnung zu berücksichtigen sind, d.h. der Komplexität der Grammatik. Zur Vereinfachung der Grammatik relativ zu einer Anfrage lassen sich verschiedene Ansätze verfolgen:

1. *Vorberechnung endlicher Probleme*

   (a) *Auflösung von Rekursionen durch Betrachtung endlicher Fälle*
   Rekursionen können als mächtiges und elegantes Abstraktionsmittel bei der Theoriebildung bezeichnet werden. In bestimmten Fällen kann jedoch eine Abbildung rekursiver auf endliche Strukturen die Berechnung vereinfachen. Subkategorisierungsmengen beispielsweise haben selten mehr als fünf Elemente und können daher auch als Merkmalstrukturen mit den Merkmalen ELEMENT1, ELEMENT2, ..., ELEMENT5 dargestellt werden. Rekursive Constraints, die sich auf derart reduzierte Strukturen beziehen, lassen

sich vorberechnen, so daß sich die Berechnungen zur Laufzeit entsprechend vereinfachen.[334]

(b) *Vorberechnung von Teilstrukturen*
Die Heuristik *"Speichern statt Berechnen"* aus der Algorithmik kann auch auf linguistische Probleme angewendet werden: Teile von Constraintbeschreibungen lassen sich in manchen Fällen durch vorberechnete Constraints ersetzen. Bei komplexen Anfragen beispielsweise kann die Berechnung in mehreren Phasen geschehen. So kann die Grammatik in einem ersten Schritt dazu verwendet werden, ein Vollformlexikon der in der Anfrage vorkommenden Wörter zu generieren, das anschließend von der Syntax zur Analyse der Satzstruktur verwendet wird. Ebenso lassen sich natürlich einzelne Phrasen oder Teilsätze getrennt analysieren oder generieren.

2. *Verwendung von Ausschnitten der Grammatik*
Das Verfahren des *"dynamischen Ladens"* aus der Informatik läßt sich ebenfalls auf TFS anwenden: Werden nur die lexikalischen Zeichen, Schemata und Prinzipien geladen, die für eine Anfrage notwendig sind, kann u.U. eine erhebliche Vereinfachung der Berechnung erzielt werden.

6.2.3 Entwicklung von Subsystemen

Bei der Entwicklung einer Constraintbeschreibung lassen sich weitere Strategien anwenden. In den meisten Fällen können Teilaspekte der Beschreibung getrennt oder aber inkrementell entwickelt werden: Syntax und Morphologie beispielsweise lassen sich weitgehend unabhängig voneinander modellieren und untersuchen; die Semantik kann inkrementell auf der Syntax aufgebaut werden. Bei der Integration dieser Subsysteme bzw. der inkrementellen Weiterentwicklung können Strategien wie die Vorberechnung von Strukturen, die Hinzunahme von Informationen etc. verfolgt werden.

Um die Anwendung dieser Strategien bei der Entwicklung der Grammatik und ihrer Verifikation anhand eines Korpus von Beispielsätzen automatisieren zu können, wurde TFS in ein "Meta-System" integriert. In unserem Fall wurde dazu die Programmiersprache *Expect*[335] verwendet: Komplexe Probleme können damit durch *Expect*-Programme in Unterprobleme zerlegt

334 Diese Strategie wird auch in der Verbmobilgrammatik von Melanie Siegel angewandt.

335 TFS wird als Lisp-Image ausgeliefert; die Lisp-Sourcen sind nicht zugänglich. Da TFS keine Programmierschnittstelle hat, wurde in *Expect* ein Wrapper geschrieben, der mit TFS über die Benutzershell kommuniziert. Auf dieser Weise wurde es möglich, TFS wie eine normale Funktion von *Expect*-Programmen aus aufzurufen.

werden; die Teilprobleme können entweder an TFS weitergegeben, oder direkt von Expect berechnet werden; die Integration der Teilergebnisse wird — gesteuert wiederum von *Expect*-Programmen — von TFS vorgenommen.

6.3 Entwicklung der japanischen Grammatik

Die Grammatik des Japanischen, die in den vorangehenden Kapiteln vorgestellt wurde, kann nur für einfache Anfragen direkt durch TFS interpretiert werden. Schon ein Großteil der Beispielsätze und morphologischen Formen aus Anhang A sind für die direkte Berechnung durch TFS zu aufwendig. Bei der Entwicklung der japanischen Grammatik wurden deshalb schon zu einem frühen Zeitpunkt die im letzten Abschnitt vorgestellten Strategien angewandt. Der Darstellung der verschiedenen Stadien dieser Entwicklung im Zusammenhang mit den angewandten Strategien dient die folgende Auflistung:

6.3.1 Verbmorphologie

Verbformen und Morphe

Ausgehend von den Verbformen des *"The Complete Japanese Verb Guide"* (CJVG, Ishii et al. 1989)[336], der traditionellen Gramatik (vgl. Lewin 1990) und der *"Japanische[n] Morphosyntax"* (Rickmeyer 1995) wurde ein Lexikon von Morphen erstellt. Die Beschränkungen bezüglich der Kombination und Realisierung der Morphe wurde durch morphologische Subkategorisierung ausgedrückt; die möglichen Kombinationen der Morphe bei der Bildung komplexer Formen entsprechend dieser Beschränkungen wurden durch ein morphologisches Schema modelliert.

Zur Berechnung der so entstandenen Morphologie war die Kapazität von TFS ausreichend; Strategien mußten zu diesem Zeitpunkt noch nicht angewandt werden.

336 Der *"The Complete Japanese Verb Guide"* (CJVG, Ishii et al. 1989) erfüllt alle Voraussetzungen, um als Korpus für eine relativ vollständige Beschreibung der japanischen Verbformen zu dienen: Für die häufigsten Verben des modernen Japanisch listet er alle wichtigen Grundformen auf. (Die Tabellen aus Anhang B wurden den Tabellen des CJVG nachempfunden und vermitteln damit eine Vorstellung von seinen Verbeinträgen.) Da die Auswahl der Verben und Verbformen repräsentativ für das moderne Japanisch ist, lassen sich nahezu alle Verbformen aus seinen Tabellen ableiten: Wird ein Verb nicht aufgelistet, läßt es sich anhand eines nach gleichen Regeln flektierenden erklären; wird eine Verbform nicht in der Tabelle dargestellt, läßt sie sich aus der Kombination ihrer Grundformen ableiten.

Morphe und morphologische Formprimitiven

Die Morphe wurden in morphologische Formprimitiven (Kern und Clitics) zerlegt und ihre Struktur systematisch beschrieben. Um das morphologische Schema auch zur Ableitung der Morphe verwenden zu können, wurde die Hierarchie der Stämme erweitert.

Bei der Ableitung der morphologischen Beschreibung diente eine Menge repräsentativer Verben und Verbformen — beide wurden aus dem *"The Complete Japanese Verb Guide"* entnommen — zur Verifikation der entwickelten Beschreibung: Alle relevanten Verbformen wurden mit dem entwickelten Modell generiert und mit den Formen aus dem CJVG verglichen. Anhand der ermittelten Fehler wurde anschließend die Beschreibung korrigiert. Dieser Vorgang wurde so lange wiederholt, bis alle Formen korrekt generiert wurden. Auch wenn es sich bei den so überprüften Formen nur um einfache Grundformen handelt, folgt aus ihrer korrekten Ableitung die Richtigkeit des gesamten morphologischen Modells: Komplexe Verbformen leiten sich aus der rekursiven Kombination der Grundformen ab. Durch "strukturelle Induktion" kann damit aus der Korrektheit der Grundformen auf die Korrektheit der komplexen Formen geschlossen werden.

Bei der Ableitung des morphologischen Modells wurde *Expect* nur zur Automatisierung der Generierung und zur Überprüfung der Verbformen verwendet: Die zugrunde gelegten Verbformen sind relativ einfach strukturiert — sie bestehen aus maximal drei Morphemen — und lassen sich damit noch durch TFS berechnen.

Die Analyse und Generierung komplexerer Formen ist jedoch zu zeit- und speicheraufwendig für TFS. Für ihre Berechnung wurde daher erstmals eine der im letzten Abschnitt beschriebenen Strategien angewandt: die Formen wurden inkrementell berechnet. Ein Expect-Programm benutzte die morphologische Beschreibung, um Formen aus zwei morphologischen Primitiven abzuleiten. Die abgeleiteten Merkmalstrukturen wurden in ein temporäres Lexikon geschrieben und zur Generierung von Formen der Länge drei verwendet usw. Dieser Vorgang wurde so lange wiederholt, bis die Verbformen vollständig abgeleitet waren. Während die morphologische Beschreibung von TFS zur Generierung, Analyse und Vervollständigung von Zeichen verwendet werden kann, ist die Expect-basierte Ableitung richtungsabhängig: sie läßt sich nur zur Analyse von Formen verwenden.

6.3.2 Syntax

Die syntaktischen und semantischen Modifikationen, die mit der agglutinierenden Verbmorphologie einhergehen, lassen sich am besten anhand eines Korpus untersuchen. Aus diesem Grund wurden zu jeder morphologischen

Form aus verschiedenen Grammatiken und Wörterbüchern[337] repräsentative Beispielsätze gesucht.[338] Um die Beispielsätze modellieren und untersuchen zu können, wurde ein syntaktisch-semantisches Modell benötigt. Die Entwicklung des syntaktischen Teils dieses Modells war Ziel der zweiten Arbeitsphase.

Bei der Entwicklung der Syntax wurde zuerst von stark vereinfachten Beispielsätzen und einem ebenso grundlegenden Vollformlexikon ausgegangen. Die so entstandene Grammatik wurde anhand komplizierterer Beispielsätze nach und nach erweitert, bis alle syntaktischen Phänomene, die in den Beispielsätzen vorkommen, behandelt werden konnten.

Für die Berechnung einfacher Sätze war die Kapazität von TFS ausreichend; bei komplexen Sätzen hingegen mußte die syntaktische Struktur zumindest teilweise vorgegeben werden.

6.3.3 Semantik

In der nächsten Entwicklungsphase wurde die Syntax um eine einfache kompositionale Semantik erweitert: Die Lexikoneinträge wurden durch semantische Beschreibungen ergänzt und die Schemata und Prinzipien so erweitert, daß sie diese kombinieren konnten. Bei komplexen Sätzen mußte wie bei der Ableitung der Syntax die syntaktische Struktur teilweise vorgegeben werden.

6.3.4 Integration von Morphologie, Syntax und Semantik

Erweiterung der morphologischen Zeichen um syntaktische und semantische Merkmale

Um das syntaktisch-semantische Verhalten der Verbformen zu modellieren, wurden die morphologischen Lexikoneinträge durch entsprechende Merkmale angereichert: Die Verbstämme wurden mit Merkmalen versehen, die ihr Defaultverhalten modellieren; die Suffixe mit Spezifikationen der Modifikationen dieser Merkmale, die durch sie bewirkt werden. Das morphologische Schema und die morphologischen Prinzipien wurden entsprechend modifiziert. Das Ziel dieser Entwicklungsphase war es, aus den Einträgen der morphologischen Primitiven das Vollformlexikon ableiten zu können, das für die Syntax und Semantik verwendet wurde.

Wie bei der Entwicklung der Morphologie wurden die Formen unter Zuhilfenahme von *Expect* inkrementell berechnet.

337 Eine Liste der verwendeten Grammatiken und Wörterbüchern findet sich im Anhang A auf Seite 287.
338 vgl. Anhang A.

Integration des morphologischen Modells

Morphologie, Syntax und Semantik konnten nun in einer gemeinsamen Beschreibung integriert werden. Mit der gewonnenen Grammatik und TFS ließen sich einfache Sätze mit einfachen morphologischen Formen direkt generieren und analysieren; komplexe Sätze und Sätze mit komplexen morphologischen Formen wurden wiederum durch ein hybrides System aus *Expect* und TFS analysiert. Es wurde folgendermaßen vorgegangen:

1. Die atomaren syntaktischen Zeichen wurden aus dem Satz extrahiert und mittels der morphologischen Komponente inkrementell berechnet. Die Ergebnisse wurden in ein "Vollformlexikon" für den entsprechenden Satz gespeichert.

2. Unter Zuhilfenahme des generierten Lexikons wurde der Satz syntaktisch und semantisch analysiert. Bei komplexen Sätzen mußte die syntaktische Struktur vorgegeben werden.

Damit das Expect-Programm die syntaktischen und morphologischen Strukturen erkennen konnte, mußten diese in den Anfragen vorgegeben werden.

Motivation dieses Vorgehens war die Entwicklung und Verifikation des entwickelten sprachlichen Modells, nicht die Anwendung der Grammatik zur Analyse oder Generierung. Auch wenn das hybride System aus TFS und *Expect* damit besser als experimentelle "Entwicklungsumgebung" für Grammatiken beschrieben wird, soll im nächsten Abschnitt seine Eignung als "computerlinguistisches System" bewertet werden.

6.4 Kritik des benutzten Verfahrens

Um das dargestellte Vorgehen zu bewerten, lassen sich zwei grundlegende Dimensionen heranziehen:

1. *Deklarative Sicht*:
   Die deklarative Eleganz der entstandenen linguistischen Beschreibung im Zusammenhang mit dem durch sie beschriebenen Sprachausschnitt.

2. *Anwendungsperspektive*:
   Die Eignung des entwickelten Gesamtsystems zur Anwendung für linguistische Berechnungen, d.h. das konkrete Laufzeitverhalten bei der Analyse (Parsen) bzw. Synthese (Generierung) von sprachlichen Formen unter Berücksichtigung von Zeit- und Speicheraufwand.

Schon das Thema der Arbeit betont den zuerstgenannten, deklarativen Aspekt: Die japanische Verbmorphologie soll im Zusammenhang mit der Semantik — und damit auch der Syntax — computerlinguistisch *beschrieben* werden. Eine Betonung der Deklarativität als Ziel einer ersten Arbeitsphase ist jedoch auch bei anwendungsorientiertem Vorgehen vorteilhaft: Durch das Erstellen einer deklarativen Beschreibung und die anschließende Überarbeitung nach prozeduralen Gesichtspunkten ergeben sich im Allgemeinen effizientere, robustere und leichter erweiterbare Systeme als bei der direkten Ausrichtung auf Anwendbarkeit.

Im Folgenden sollen kurz die Einschränkungen hinsichtlich der direkten Anwendbarkeit dargestellt werden, die sich aus der Verwendung eines deklarativen Systems wie TFS ergeben. Anschließend jedoch soll dargelegt werden, warum dieses Vorgehen, verglichen mit der Verwendung anwendungsorientierter Systeme, trotzdem Vorteile hat.

TFS und die japanische Grammatik

Typenhierarchien und Constraints erlauben eine elegante Spezifikation von linguistischen Sachverhalten. Diese Eleganz resultiert aus der deklarativen Form der Darstellung und ihrer mathematisch definierten Semantik. Die "algorithmisierte Form" dieser mathematischen Semantik — der in Kapitel 1 beschriebene Kalkül — wird von TFS unter Anwendung der Tiefensuche direkt zur Ableitung von Lösungen angewandt. TFS erfüllt damit — im Gegensatz zu anwendungsorientierten Systemen — alle Anforderungen, die aus theoretischer Sicht an deklarative Systeme gestellt werden.

Aus eben diesen Qualitäten ergeben sich jedoch Nachteile bei der Anwendung von TFS zur Analyse oder Generierung von Sprache: Berechnungen sind aufgrund der allgemeinen Ableitungsstrategie extrem zeit- und speicheraufwendig, so daß TFS nur für relativ kleine Constraintbeschreibungen benutzt werden kann.

In dem hier beschriebenen Ansatz wurde dieses Problem mittels eines hybriden Systems aus TFS und *Expect* umgangen. Die daraus resultierende Aufteilung der computerlinguistischen Berechnungen auf TFS und *Expect* und die Vereinfachung der Berechnungen durch die im letzten Abschnitt vorgestellten Strategien führten jedoch zu erheblichen Einschränkungen des Gesamtsystems:

1. Bei komplexen Sätzen oder morphologischen Formen kann das erstellte System nur zur Analyse benutzt werden.

2. Bei komplexen Eingaben muß die morphologisch-syntaktische Struktur vorgegeben werden. Das Erkennen der syntaktischen Struktur — sonst

eine der wichtigsten Aufgaben der syntaktischen Analyse — ist zu komplex. Die Aufgabe von TFS beschränkt sich also darauf, zu überprüfen, ob die vorgegebene Struktur mit der Grammatik verträglich ist.

3. Durch die Zerlegung in Teilsysteme, das Vorgeben von Informationen etc. kommen wesentliche Aspekte der Grammatik nicht zur Anwendung. Das Verhalten der grammatischen Spezifikation kann ohne diese äußeren Einschränkungen für komplexe Sätze nicht experimentell überprüft werden.

Für die konkrete Anwendung ist das System aus TFS, *Expect* und der japanischen Grammatik damit nicht geeignet: Weder zur Generierung noch zur Analyse läßt es sich direkt verwenden. Anders verhält es sich mit der entwickelten Grammatik: Als deklarative Beschreibung ist sie von TFS unabhängig und kann auf jedem anderen computerlinguistischen System ohne allzu großen Aufwand implementiert werden.

Anwendungsbezogene Systeme

Um trotz der syntaktischen und morphologischen Kombinatorik zu einem akzeptablen Laufzeitverhalten zu gelangen, werden anwendungsorientierte computerlinguistische Systeme, wie z.B. PAGE oder ALE, für die Generierung oder Analyse von Sätzen optimiert. Diese Optimierung besteht in der festen Integration bestimmter Berechnungsstrategien in das betreffende System. Diese Strategien jedoch führen genau wie die im letzten Abschnitt vorgestellten zu einer Abweichung des Systems von der deklarativen Semantik der linguistischen Beschreibungssprache. Im Gegensatz zu der Vorgehensweise, der bei der Entwicklung der dargestellten Grammatik gefolgt wurde, kann diese Abweichung nicht vom Autor der Grammatik kontrolliert werden. Selbst bei der Betrachtung eines kleinen Ausschnittes eines sprachlichen Modells erfolgt die Ableitung nicht entsprechend dessen deklarativer Semantik, die damit nicht experimentell überprüft werden kann.

Damit wird der Vorteil des in dieser Arbeit verfolgten Ansatzes deutlich: Durch die Möglichkeit, über die Verwendung von Strategien bewußt zu entscheiden, kann die Semantik von Grammatiken besser überprüft werden, als bei der Verwendung eines Systems mit fest integrierten Strategien. Bei bewußtem Einsatz der Strategien ist es daher leichter, zu einer korrekten Spezifikation der linguistischen Sachverhalte zu kommen.

Zusammenfassung

Die Implementierung unseres Systems dient also nicht der Anwendung, sondern der Ableitung der vorgestellten Grammatik und sollte deshalb auch nicht an anwendungsbezogenen Kriterien gemessen werden. Die Anwendung der Grammatik — und damit verbunden die Optimierung hinsichtlich der Generierung oder Analyse von Sprache — läßt sich besser im Rahmen einer gesonderten Betrachtung diskutieren und bildet damit ein interessantes Thema für die Fortführung dieser Arbeit.