INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA

UMI[®]
800-521-0600

An Inferencing Strategy for Resolving Quantifier Scope Ambiguities

by
Walid S. Saba, B.Sc., M.Sc.

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Ottawa-Carleton Institute of Computer Science

School of Computer Science

Carleton University
Ottawa, Ontario
April 10, 1999

© 1999, Walid S. Saba

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-42806-0

Canada

The undersigned hereby recommend to
The Faculty of Graduate Studies and Research

Acceptance of the thesis,

**An Inferencing Strategy for Resolving
Quantifier Scope Ambiguities**

submitted by

Walid S. Saba, B.Sc., M.Sc.

in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

for External Examiner

Director, School of Computer Science

Thesis Supervisor

Carleton University
31 May 1999

TO THE MEMORY OF SABA ABRAHAM SABA

*who inspired pride, humility, and uncompromising
goodness.*

Abstract

When building natural language systems, computational linguists must ultimately be concerned with pragmatics, i.e., with designing decision procedures that draw on contextual and commonsense knowledge to select the most 'plausible' interpretation from a multitude of alternatives that syntactic and semantic analysis deem equally valid. A classic example is the practical problem of choosing the most plausible quantifier scope ordering, in formulating database queries, which remains one of the most challenging problems in computational linguistics. Despite overwhelming evidence suggesting that quantifier scope is a phenomenon that must be treated at the pragmatic level, most computational treatments of scope ambiguities have thus far been a collection of syntactically motivated preference rules. This work investigates the thesis that, in a circumscribed environment, very effective procedures for resolving quantifier scope ambiguities at the pragmatic level can be attained. In particular, we present a process that views resolving quantifier scope ambiguities as an inferencing procedure that utilizes commonsense world knowledge.

In designing the proposed scope resolution algorithm we have tried to address three major challenges: *(i)* the scope algorithm must account for a number of linguistic phenomena that involve quantification; *(ii)* the scope algorithm must account for the apparent differences in scope preferences that are due to variations in the linguistic context; and *(iii)* while we expect the scope resolution algorithm to utilize some commonsense world knowledge, our goal was to design a highly effective procedure that does not suffer from the bottleneck of storing and reasoning with a vast amount of background knowledge, as such procedures can be intractable.

In addition to achieving these design goals, we believe our approach raises some interesting questions from the psycholinguistic standpoint, which are briefly discussed throughout the thesis.

Acknowledgements

I am indebted first and foremost to my advisor Dr. Jean-Pierre Corriveau for his continuous support and guidance, and for encouraging me to not fear conducting provoking and uncompromising research. I am also grateful to Dr. Lokendra Shastri of the University of California, Berkeley, for the insight he provided as my external examiner.

This thesis has also improved considerably due to the valuable feedback of Dr. Robert Stainton, Dr. Marie-Odile Junker, Dr. Franz Oppacher, Dr. Caroline Barrière and Dr. Stan Szpakowicz, and I am grateful to all of them. I must also thank Dr. Richard A. Frost, who few years ago gently planted the *computational intelligence* bug in the mind of a drifter, and Dr. Mohsen Rais-Ghasem, Rod Davison, Dr. Douglas Young, Dr. Bill Tephenthart, Dr. Wlodek Zadrozny, Dr. Dimitris Phoukas, and Hassan Afyouni for their encouragement.

Above all, I want to thank my wife Norma, and my two children, Shaker and Samar, who understood the reason behind all those long hours in solitude, and for the lovely interruptions that I needed every now and then.

Finally, to my parents: the scope of your love and support cannot be quantified!

Table of Contents

Abstract	iv
List of Tables	ix
List of Figures	x
List of Appendices	xi
1 Introduction	1
1.1 QUANTIFIER SCOPE AMBIGUITIES	2
1.2 EVIDENCE OF THE INFERENTIAL ASPECT OF QUANTIFIER SCOPE.....	6
1.2.1 A Combinatorial Puzzle.....	6
1.2.2 The Transformational Puzzle.....	8
1.2.3 Beyond Syntax and Semantics.....	9
1.3 OUR THESIS	10
1.4 THE SCOPE OF THIS INVESTIGATION	11
1.5 STRUCTURE OF THIS THESIS	12
2 Quantificational Constraints.....	14
2.1 RELATIONS AND CARDINALITY CONSTRAINTS	14
2.2 QUANTIFICATIONAL CONSTRAINTS AND QUANTIFIER SCOPE.....	15
2.3 CONTEXT AND POSSIBLE WORLDS	16
2.4 MODAL AND TEMPORAL ASPECTS OF QUANTIFICATIONAL CONSTRAINTS	16
2.4.1 The Role of Modality	16
2.4.2 Temporal Aspects	18
2.5 REASONING WITH QUANTIFICATIONAL CONSTRAINTS	19
2.5.1 Quantificational Constraints as Logical Constraints.....	19
A Brief Interim Summary.....	24
3 A Numerical Scoping Algorithm.....	26
3.1 QUANTIFICATIONAL CONSTRAINTS AS NUMERICAL CONSTRAINTS	26
3.1.1 A Space of Plausible Situations	28
3.1.2 The Linguistic Context.....	30
3.2 A NUMERICAL MEASURE FOR THE PLAUSIBILITY OF A READING	35
3.3 APPLYING THE SCOPING ALGORITHM.....	37
3.3.1 A Possible Explanation for the Transformational Puzzle.....	37
3.3.2 Generalized Quantifiers.....	40
3.4 COMPLEX FUNCTIONAL DEPENDENCIES	41
3.4.1 Simple Functional Dependencies.....	41

3.4.2	<i>Branching Quantifiers</i>	43
3.5	EXTENDING THE SCOPING ALGORITHM TO ACCOUNT FOR THE TEMPORAL AND MODAL ASPECTS.....	45
3.5.1	<i>The Relation Between Modality and the Temporal Aspect</i>	46
3.6	A BRIEF SUMMARY	49
4	Representational and Computational Issues	51
4.1	TYPICAL CARDINALITIES	51
4.2	QUANTIFICATIONAL CONSTRAINTS	52
4.3	TYPICAL ELEMENTS	54
4.4	TYPICAL ELEMENTS, INHERITANCE AND QUANTIFICATIONAL CONSTRAINTS	55
4.4.1	<i>A Brief Summary</i>	57
4.5	THE OVERALL PROCESS	58
4.6	SUMMARY	60
5	Implementation	61
5.1	SCOPE OF THE IMPLEMENTATION.....	61
5.1.1	<i>Testing the Scoping Algorithm</i>	61
5.1.2	<i>Function (Process) vs. Data</i>	61
5.1.3	<i>The Knowledge Representation</i>	62
5.1.4	<i>Extending the Knowledge Representation</i>	64
5.2	THE PARSER	65
5.3	SEMANTIC ANALYSIS	68
5.3.1	<i>Logical Expressions</i>	70
5.3.2	<i>Processing Prepositional Phrases</i>	71
5.3.3	<i>Processing Quantified Noun Phrases</i>	75
5.4	THE SCOPING FUNCTION	77
5.5	CONCLUDING REMARKS	79
6	Related Work	81
6.1	QUANTIFICATION IN NATURAL LANGUAGE	81
6.2	QUANTIFICATION IN FORMAL LOGIC	82
6.2.1	<i>Quantification in PTQ</i>	82
6.2.2	<i>Quantifier Scope in Montague's PTQ</i>	84
6.2.3	<i>The Theory of Generalized Quantifiers</i>	86
6.2.4	<i>Formal Properties of Generalized Quantifiers</i>	88
6.2.5	<i>The Relevance of TGQ in Quantifier Scope Ambiguity</i>	89

6.2.6	<i>How do Scope Ambiguities Arise in a Set-Theoretic Model?</i>	89
6.2.7	<i>Branching Quantifiers in Natural Language</i>	90
6.2.8	<i>Quantifiers as Procedures</i>	91
6.3	QUANTIFIER SCOPE AMBIGUITY IN COMPUTATIONAL LINGUISTICS	92
6.3.1	<i>Computational Models of Quantifier Scope Ambiguity</i>	93
6.3.2	<i>Underspecification and Incremental Interpretation</i>	94
6.3.3	<i>Rules and Preferences for Resolving Scope Ambiguities</i>	96
	<i>Allen</i>	97
	<i>Hobbs and Shieber</i>	98
	<i>Moran and Park</i>	99
	<i>Fox</i>	100
	<i>May</i>	100
	<i>Sher</i>	101
	<i>Moran and Pereira</i>	102
	<i>TEAM</i>	102
6.4	SUMMARY	103
7	Summary and Future Work	105
7.1	THE PRAGMATICS OF QUANTIFIER SCOPE	105
7.2	FUTURE WORK	108
7.2.1	<i>The Role of Time and Memory Constraints</i>	109
7.2.2	<i>A Suggested Procedure that is Sensitive to Computational Constraints</i>	110
7.2.3	<i>Further Work</i>	113
7.3	CONCLUDING REMARKS	114
	Bibliography	116
	Appendix A. An Object-Oriented Attribute Grammar	121
	Appendix B. A Detailed Example	122
	Appendix C. Sample Session with <i>Quasi</i>	124

List of Tables

Table 1. Cardinality constraints on relations and sets	27
Table 2. The interaction between QCs and the modal and temporal aspects	46
Table 3. The correspondence between syntactic and semantic objects in <i>Quasi</i>	69
Table 4. A set-theoretic interpretation of quantifiers.....	90
Table 5. Three models of negation.....	110
Table 6. The standard quantifiers using the function in (95)	112

List of figures

Figure 1	The basic mechanisms of quantifier raising and quantifying-in	3
Figure 2	The <i>BuiltOn</i> relation between houses and streets	20
Figure 3.	Using entailment in FOL to test the plausibility of a reading	23
Figure 4.	Applying the algorithm on <i>Most students speak at least two languages</i>	24
Figure 5.	The space of plausible combinations defined by a QC	29
Figure 6.	Measuring the plausibility of a scope ordering.....	38
Figure 7.	Movements of quantificational constraints on the temporal dimension	47
Figure 8.	Quantificational constraints defined in a concept hierarchy	56
Figure 9.	The overall process of resolving scope ambiguities	58
Figure 10.	A screen shot of the main <i>Quasi</i> interface	63
Figure 11.	The attributes and methods of a generic linguistic object	66
Figure 12.	The semantic analysis of <i>few MIT students submitted a paper</i>	73
Figure 13.	Prepositional phrase attachment: the simple case.....	74
Figure 14.	The implementation of the scoping functions.....	79
Figure 15.	Two derivation trees corresponding to the $\forall\exists$ and $\exists\forall$ readings.....	85
Figure 16.	An automaton corresponding to the quantifier <i>all</i>	91
Figure 17.	Quantification with time and memory constraints	111

List of Appendices

Appendix A	An Object-Oriented Attribute Grammar.....	121
Appendix B	A Detailed Example	122
Appendix C	Sample Session with <i>Quasi</i>	124

1 Introduction

It has been nearly half a century since the mathematician and computing pioneer Alan Turing envisioned a machine that could converse in English, in such a manner that would render the machine *intelligent*. The genius of Turing, in our view, was not only in his theoretical analysis of what is ultimately computable and what is not, but in his visionary analysis of what language understanding entails, so as to consider it a suitable test for intelligence. It took researchers in natural language understanding (NLU) few decades to appreciate the complexities in building systems that can comprehend natural language text. Chief among these obstacles has been *ambiguity*, in its various forms.

As far as humans are concerned, however, ambiguity is not a sin! In fact, ambiguity in natural language, as Givon (1984) has convincingly argued, exists because of our ingenious method of data compression that we employ for performance and storage effectiveness. We store several meanings (senses) of words under the same token, so as to build smaller and smaller lexicons. We also compress our natural language expressions, leaving unnecessary details and explanations out, hoping that our background knowledge and the overall context are enough to *decode* the message being conveyed. This linguistic invention, which has served us well, is a monumental challenge for machines.

Thus, while linguists and logicians *study* ambiguity, in natural language processing (NLP) ambiguity is an 'illness' that computational linguists speak of *treating*. A query to a database system must be an unambiguous logical formula that is either true or false of some situation. A particularly challenging task in formulating database queries has been the task of resolving quantifier scope ambiguities.

As will be discussed in this chapter, there seems to be ample evidence suggesting that quantifier scope is a phenomenon that must be treated at the pragmatic level. In spite of this, virtually all the computational approaches to this problem employ a number of syntactically motivated preference rules. The work presented here investigates the thesis that in a circumscribed environment, such as that of a database system, a highly effective inferencing procedure for resolving scope ambiguities can be attained. Before we discuss our strategy, in the remaining of this chapter we will define the boundaries of our investigation, and present some evidence in favor of an inferential and a pragmatic approach to resolving scope ambiguities.

1.1 QUANTIFIER SCOPE AMBIGUITIES

Excluding structural, lexical, and other types of ambiguities, a sentence with n quantifiers has potentially $n!$ distinct interpretations that are due only to the different possible scope orderings of quantifiers. As a simple example, consider the sentence in (1), which can admit the two interpretations given in 1a and 1b:

(1) *Every child ate a cookie*

1a. $\forall c(\text{Child}(c) \rightarrow \exists k(\text{Cookie}(k) \wedge \text{Ate}(c, k)))$

1b. $\exists k(\text{Cookie}(k) \wedge \forall c(\text{Child}(c) \rightarrow \text{Ate}(c, k)))$

In a truth-conditional model of interpretation (Montague, 1974; Cresswell, 1973), where the interpretation of a sentence is the set of conditions under which the sentence is true, deciding on the correct interpretation of (1) is crucial since 1a and 1b may potentially describe two distinct possible states of affairs.

What makes quantifier scope ambiguity distinct from other types of ambiguities (e.g., structural, lexical, etc.), is that it is not at all that clear at what level in the overall interpretation process should scope ambiguities be addressed. Partly a linguistic and partly a logical phenomenon, scope ambiguities have received considerable attention from both linguists and logicians where two related yet quite distinct notions of 'logical form' have developed. In both cases, the challenge was to (syntactically, or semantically) *explain* how a seemingly unambiguous sentence can admit different interpretations that are due only to the different quantifier scope orderings.

Linguists interested in a syntax-level explanation of scope ambiguities suggested that the possible distributions of quantifiers can be explained by a **covert syntactic operation**, termed quantifier raising (QR), which allowed the movement of quantified constituents as long as some rules were not violated. Since the early years of its introduction (Rodman, 1976), the optimism surrounding QR was in the hope of establishing some syntactic rules that could universally account for all possible quantifier movements (see Reinhart, 1983; May, 1985; Fox, 1995; Reinhart, 1997). Indeed, a number of interesting patterns (of movements) were observed, and *structural principles* to explain quantifier phrase movements were established, such as the linear order and *c-command* principles, *islands* constraints, and *wh*-movements.

While quantifier phrase movements *can* reduce the number of possible readings (as some permutations might be disallowed), QR is not itself a decision procedure that can predict a 'single' preferred reading. Moreover, the initial excitement surrounding QR has dwindled in recent years, as it became apparent that a single syntactic operation is not going to provide an explanation for all the predictable movements of quantifier phrases¹. As the experiments of Kurtzman and MacDonald (1993) suggest, no *structural principle* alone seem to always predict the preferred scope ordering. Other principles, such as *lexical bias* and *world-knowledge* also seem to be operative in some situations. Nevertheless, QR and similar linguistic approaches have inspired a number of scope resolution algorithms in computational linguistics (see Hobbs and Shieber, 1987; Grosz *et al.*, 1987; Allen, 1987; Moran, 1988; Moran and Pereira, 1992; Park, 1995).

Within the logicophilosophical tradition, scope ambiguities were explained by a **covert semantic operation**, known as quantifying-in (Montague, 1974; henceforth PTQ). In PTQ, due to strong compositionality, scope ambiguity had to be explained by positing a different derivation for every possible scope ordering. While it was an ingenious invention to salvage compositionality (see Thomason, 1974; Gamut, 1991), the quantifying-in rule introduced an arbitrary number of derivations for a structurally unambiguous sentence, a shortcoming that was later remedied by the 'Cooper storage' method (Cooper, 1983)².

Figure 1. The basic mechanisms of quantifier raising and quantifying-in.

¹ A recent study by Reinhart (1997) that provides an excellent historical analysis of QR, does suggest some recommendations that might yet again revive the early optimism.

² It should be noted here that the 'Cooper storage' method is not strictly compositional. Since the method requires the maintenance of a state (storage), a purely declarative denotational semantics can not be attained. See (Pereira and Pollack, 1991) for a more elaborate discussion on this subject.

Like QR, which aimed at providing a **syntactic explanation** of scope ambiguities, quantifying-in or Cooper Storage are not techniques for resolving scope ambiguities, but are a **semantic explanation** of scope ambiguities³. Figure 1 above is a very simple illustration of QR and quantifying-in. Note that both operations achieve the same purpose: reverse the scope orderings of the quantified NPs at the logical form level. (We will have more to say about quantifying-in in chapter 6.)

It is important to note that quantifying-in (and consequently Cooper storage), as well as more recent techniques for coping with scope ambiguities, such as underspecified discourse representation structures (UDRS) (Reyle, 1995), do not suggest an inferencing strategy to 'select' a single preferred reading, but are extensions to the logical formalism that would allow logical reasoning to be performed on underspecified (ambiguous) logical forms. From a logical standpoint, this is very crucial. For example, suppose we were told S, where $S = \text{Every student read a book}$, and that *Steven is a student*. Regardless of the scope ambiguity in S, the following reasoning is valid:

$$\frac{\left\{ \begin{array}{l} \forall x(\text{Student}(x) \rightarrow \exists y(\text{Book}(y) \wedge \text{Read}(x, y))) \\ \exists y(\text{Book}(y) \wedge \forall x(\text{Student}(x) \rightarrow \text{Read}(x, y))) \end{array} \right\}}{\text{Student}(\text{Steven})} \quad \therefore \exists y(\text{Book}(y) \wedge \text{Read}(\text{Steven}, y))$$

That is, regardless of whether every student read a different book (narrow scope *a*), or all the students read the same book (wide scope *a*), the conclusion 'Steven read a book' follows. Note that here we have reasoned with S by representing the ambiguity as a *disjunction* of all possible meanings⁴. From a logical point of view, therefore, both readings are equally valid, and neither can be said to be more plausible. To be sure, semanticists and logicians were never concerned with 'plausibility' (i.e., with selecting the most 'plausible' meaning), but are primarily concerned with developing an *algebra for meanings*, whatever these meanings are.

In PTQ, for example, lexical items such as 'engineer' and 'president' share the same syntactic category, and thus from the semantic point of view all that is required to know is that such items have the (extensional) **semantic type** $\langle e, t \rangle$, representing a (truth-functional) predicate that is either true or false of some entity. Thus, semanticists are not concerned with what we

³ As has also been shown by Weisler (1987), movements due to quantifier raising (QR) in generative semantics perform the same function as quantifying-in in Montague's intensional logic.

⁴ Disjunction is not the best representation of ambiguity however (see van Deemter (1996)).

know about *engineers* and *presidents*, but what we know about the tokens 'engineer' and 'president'. In other words, the fact that common knowledge says that institutions often have many engineers but one president does not concern semanticists. These differences, however, are apparently part of the reason why different scope preferences seem to be available in the following:

- (2) *Every senior engineer of a major software company attended Internet'98*
- (3) *Every president of a major software company attended Internet'98*

While many readers might contemplate a wide scope *a* in (2), they are not likely to admit a wide scope *a* in (3), which implies the existence of several presidents for a single software company. Lacking any syntactic or semantic differences between (2) and (3), we argue that readers must be relying on the encyclopedic facts that while software companies might have many senior engineers, they typically have one president. These pragmatically relevant facts, however, are irrelevant to logic and semantic analysis, and, as of yet, there is no evidence that these differences are encoded at the syntactic level⁵.

When building natural language systems, these differences are crucial however, and a treatment for the syntactic, semantic, **pragmatic**, as well as the **plausibility** of an interpretation, must be provided⁶. While for a semanticist all meanings were created equal, a query to a database system *must* ultimately be translated into an unambiguous logical form (LF). In the absence of syntactic or semantic evidence in favor of a single preferred reading, the resolution of scope ambiguity will inevitably require a decision procedure to infer, based on some pragmatic considerations, the most **plausible interpretation**. As Kameyama (1996) also argues, pragmatics must be the last resort in selecting the most plausible scope ordering, as it is also the process where anaphora resolution, prepositional attachment disambiguation, etc. must be performed.

⁵ It is precisely this point that makes the effort to develop the quantifier raising formalism of considerable importance. That is, if *structural principles* that could account for all predictable quantifier movements can be discovered, then the argument of a *universal grammar* that encodes much more than syntax would gain considerable strength.

⁶ In formal languages a similar situation arises. For example, the *syntax* of a certain programming language might allow us to construct expressions of the form *m/n*. The *semantics* of the language would replace the symbols by their meanings (*m* and *n* by their values, and */* by some division procedure). *Pragmatics* is then responsible for *making sense* of the meaning of the whole expression. For example, if *n* had a value of 0, the meaning of the whole expression, although semantically valid, would be rejected. Condition rules in Knuth's (1968) attribute grammars were one such device that took the meaning (denotation) of an expression as input and enforced some pragmatic rules. (Frost and Saba, 1991) used such condition rules in a pragmatics module for pronoun resolution.

The argument for a pragmatics-based approach to scope ambiguities is not uncontroversial, however, since algorithms based on the processing of background knowledge might be **undecidable** (Reinhart, 1997). As van Deemter (1996) notes, "the computational difficulties involved in the use of background knowledge to discard impossible or unlikely readings are notorious, since this requires the storage of, and reasoning with a vast body of encyclopedic knowledge."

As will be argued in this thesis, however, the background knowledge that we assume reasoners have recourse to in resolving quantifier scope ambiguities is minimal. We will first present our approach by making two very simple assumptions regarding the knowledge representation. In particular, it will be assumed that a quantifier scoping algorithm has access to (i) a cardinality constraint associated with every binary relation between two concepts; and (ii) the typical cardinality of every concept's extension.

In later chapters these assumptions are discussed in detail. For now, it suffices to say that whether or not these assumptions are borne out, the engineering applications of our results are clear. Specifically, it will be demonstrated that these two simple assumptions are sufficient to design a highly effective inferencing procedure for predicting quantifier scope.

1.2 EVIDENCE OF THE INFERENTIAL ASPECT OF QUANTIFIER SCOPE

Various **processing principles** have been suggested in the literature as to what might govern the process of resolving scope ambiguities, such as *structural principles*, *lexical bias*, formal properties of quantifiers (van Benthem, 1986), and *world-knowledge*.

A detailed review of the main approaches to quantifier scope ambiguities will be given in chapter 6. Our immediate concern, however, is to point out that among the various approaches to scope ambiguities little effort has been given to the development of inferential algorithms that utilize commonsense world knowledge. In this section we will present some evidence suggesting the need for such an approach.

1.2.1 A Combinatorial Puzzle

Most computational approaches to the resolution of quantifier scope ambiguities are based on applying a number of syntactically motivated rules in an incremental fashion. Typically, a

scope-neutral (Allen, 1987), or a quasi-logical form (QLF) (Alshawi, 1990) is constructed⁷. For each s-structure with n quantifiers there are initially $n!$ possible scope orderings that must be considered, as has indeed been done in some implementations (Grosz *et al.*, 1987). A number of rules are then applied to reduce the number of permutations considerably, such as *c-command* and other structural principles (Reinhart, 1997), economy of scope principles (Fox, 1995), and various logical constraints (Hobbs and Shieber, 1987; Park, 1995). Hobbs and Shieber, for example, argue that (4) has 42, as opposed to 120 ($5!$), valid readings:

- (4) *Some salesman of every department in most companies
saw a few samples of each product.*

In this approach some readings are not admitted on logical grounds; e.g., readings that result in logical forms with unbound variables are disallowed. It is also assumed that further processing on the remaining 'valid' readings is needed to select the 'plausible' readings. This incremental approach to the selection of the most likely reading is somewhat problematic however. As noted by Poesio (1996), a sentence such as (5) has "hundreds of thousands of scopally distinct readings if all permutations were considered admissible yet most people seem to deal with these sentences effortlessly."

- (5) *A politician can fool most voters on most issues most of the time, but no
politician can fool all voters on every single issue all of the time.*

From a computational as well as from a psychological stand, this *Combinatorial Explosion Puzzle* is somewhat problematic. Given the speed by which readers comprehend large linguistic fragments (Shastri and Ajjanagadde, 1993), performing a number of inferences along the way, it is highly unlikely that readers initially consider all the *syntactically* and *semantically valid* interpretations. As several authors have also suggested (Cherniak, 1992; Johnson-Laird, 1994; Coriveau, 1995), a system performing linguistic inferences must take into account real time constraints and limitations on working memory. Considering all possible readings, while it has the advantage of being an infeasible inferencing process, is neither cognitively, nor computationally plausible. A more promising explanation might be

⁷ A scope-neutral logical form is a logical form where no commitment has yet been made regarding the scope of quantifiers. Typically such decisions are made in a pragmatics module that takes the output of the semantic analysis and discourse information into consideration. In such a module, scope ambiguities, as well as a number of other ambiguities, are resolved. In recent years a number of powerful techniques for representing and reasoning with *underspecified* (i.e., ambiguous) forms have been developed. See (van Deemter and Peters, 1996) for a recent collection of papers on this subject.

one where readers initially select **the most plausible reading** given an appropriately defined context, with the caveat that such an inference could be retracted in the future.

1.2.2 The Transformational Puzzle

One of the strongest indications of the pragmatic and inferential aspect of quantifier scope is the so-called “transformational puzzle.” Consider the following:

- (6) *a. Every student submitted a paper*
b. A paper was submitted by every student
- (7) *a. Every student attended a seminar*
b. A seminar was attended by every student
- (8) *a. Every student speaks a language*
b. A language is spoken by every student

While many syntactic theories in the Chomskian tradition suggest that the active and passive forms are semantically equivalent, the reader can easily verify that the meanings implied by the two forms *gradually* become radically different. In (6), for example, the meanings implied by the passive and active forms are quite similar. In (7), however, the situation seems to be gradually changing, as both interpretations of (7*b*), implying one or several seminars, could be contemplated. Moving on to (8), the meanings implied by the passive and active forms start to be radically different. 8*a* simply states that every student speaks some language, while in 8*b* there seems to be a much stronger statement being made, namely that there is some language that every student speaks.

This ‘gradual’ (soft?) change in the salient reading seems to suggest that there is some **scalar** aspect to whatever parameters this process might be a function of. This is even more evident in examples such as these:

- (9) *Every*
 - (10) *Several*
 - (11) *Two*
- } *student(s) submitted a paper to AAAI'97*

While the only difference between the sentences (9-11) is in the determiner of the head noun phrase, it seems that the likelihood of a single paper being implied by the sentence *gradually* increases as we speak of less students, while concurrently the passive form seems to *gradually* imply a different meaning from the one implied by the active form. Lacking any syntactic or semantic explanations these examples suggest that some process that lies beyond syntax and semantics must be operative.

1.2.3 Beyond Syntax and Semantics

In (2) and (3) (repeated below) we argued that the only apparent difference seems to be in the choice of the lexical item in the head noun phrase.

(2) *Every senior engineer of a major software company attended Internet'98*

(3) *Every president of a major software company attended Internet'98*

It seems reasonable to argue, then, that the apparent differences can still be captured by semantics; by attributing the difference to the meaning of the lexical items 'engineer' and 'president'. Without debating where *word meaning* stops and *world knowledge* begins, we hope to argue that the difference between (2) and (3) is not simply a function of a single lexical item only, but a function of **the interaction between several lexical items**. Let's elaborate. First, consider the following:

(12) *John advertised a restaurant on every street*

(13) *John visited a restaurant on every street*

In compositional semantics in the Fregean tradition, e.g. (Montague, 1974), the form dictates the construction of a certain logical form (LF) thereby forcing a specific scope ordering of quantifiers. Having identical syntactic derivations, the same semantic rule would be used to construct a LF for (12) and (13), and mechanisms such as QR and quantifying-in are needed to account for all possible readings, all of which are, as far as semantics go, equally valid. However, while most readers might admit the possibility of a single restaurant being advertised on every street (by having *a* outscope *every* in (12)), they are not likely to admit a reading of (13) which implies the physical existence of the same restaurant on every street⁸.

The question, then, is what are the apparent differences in the scope preferences of (12) and (13) due to? Since the only difference here seems to be in the choice of the main verb, it is tempting to suggest that the semantics of the main verb play a crucial role, as has in fact been suggested by a number of authors (e.g., Scha, 1981; van der Does, 1994). While this is indeed a sound observation, we suggest that it is only part of the puzzle. For example, in (2) and (3), the differences can only be attributed to the difference in the head noun. To illustrate this point even further, let us consider another lexical variation on (12) and (13):

⁸ While we are discussing declarative statements, our concern is interpreting queries such as

(1) *Did John advertise a restaurant on every street?*

Essentially, the answer to (1) is (12) or its denial. To generate a query for (12), then, we must select a single reading for (1).

(14) *John visited a restaurant on every occasion*

While (13) and (14) differ only in the noun of the prepositional phrase, it is easy to see how this could also affect the choice of preferred scope orderings considerably. (The same restaurant could not be visited on more than one street, although it could be visited on more than one occasion).

What these examples seem to suggest, then, is that an appropriate scoping algorithm would have to be a function of (the meaning of) various constituents. This conclusion, as it stands, is not radically different from various proposals that have been previously suggested. In fact, several authors have suggested that the preferred scope ordering must be a function of:

1. The quantifiers and their structural positions (Reinhart, 1983; May 1985)
2. The semantics of the main verb (Scha, 1981; van der Does, 1994)
3. The cardinality conditions underlying the main relation (Sher, 1990)
4. The functional dependency implied by the main verb (Park, 1995; Hobbs, 1996)

While we agree with these observations, we want to suggest that the process does not involve just a single constituent but instead is a function of the interaction between **all** the constituents. Moreover, and to our knowledge, the specifics of how these various parameters contribute to an inferencing module for predicting the most plausible scope ordering, as well as the nature of the inferencing process itself, have never been laid out.

1.3 OUR THESIS

In this chapter we have argued that when building natural language systems some procedures for resolving quantifier scope ambiguities must be developed. Furthermore, we have provided some evidence suggesting that information regarding quantifier scope seem to be encoded at the pragmatic level.

The logical consequence of this latter hypothesis is the development of inferential procedures that utilize commonsense knowledge. The challenge in meeting this goal is in identifying the relevant world knowledge that is sufficient for such an inferencing strategy, as well as the development of effective procedures that do not suffer from the well-known bottlenecks in knowledge engineering and knowledge acquisition.

With these tasks in mind, the thesis that will be defended here can be summarized as follows:

1. Much of the information regarding quantifier scope is a function of an individual knowledge base, and is encoded at the pragmatic level.
2. The process of inferring the most plausible scope ordering is a defeasible inferencing process, as additional information could in principle result in retracting the initially inferred scope preferences.
3. In a circumscribed environment, such as that of a database system, a highly effective procedure for predicting quantifier scope at pragmatic level could be attained.

As will be demonstrated throughout this thesis, the inferencing procedure that we suggest will explain a number of quantifier scope phenomena, including the transformational puzzle, branching quantifiers, as well as account for generalized quantifiers, variations in the linguistic context, and the modal and temporal aspects of a relation.

1.4 THE SCOPE OF THIS INVESTIGATION

In developing an inferential strategy for predicting quantifier scope, we are not necessarily claiming that *all* information regarding quantifier scope is encoded at the pragmatic level, but that pragmatics must be the last resort, and that syntactic and semantic approaches do not necessarily suffice.

As such, we will not be concerned with cases where syntactic and semantic information could adequately resolve the ambiguity. Our concern is to address those particular cases that seem to escape both syntactic and semantic analysis. In a sense, then, the inferencing procedure that we will suggest could be part of a pragmatics module that takes as input $(n! - (s_1 + s_2))$ possible readings, where s_1 and s_2 are the number of readings that were rejected by syntactic and semantic analysis, respectively, and where n is the number of quantified NPs.

That is, the inferencing procedure that we propose takes as input an un-scoped (or a scope-neutral) logical form that has been arrived at by traditional syntactic and semantic analysis. In such a model, traditional syntactic and semantic approaches to quantifier scope could be applied at their respective stages. Our algorithm is invoked in those cases where the syntactic and semantic modules fail to arrive at a single *preferred* reading; in particular, in those

situations where the disambiguation seems to require an inference based on some commonsense world knowledge.

Admittedly, however, this sequential processing model (of syntax, then semantics then pragmatics) is somewhat simplistic, since conceivably a more sophisticated process that performs syntax, semantics and pragmatics in tandem could be developed. We do not address this issue in this thesis, although the results of our investigation could be used to rethink some of the proposals suggested at the syntactic and semantic levels.

Another important boundary that we drew around our investigation is the role of syntax. Since our thesis is not concerned with the analysis of quantifier scope at the syntactic level, we only consider very simple syntactic structures, although a number of syntactic forms for noun phrases were considered due to the role these linguistic variations play in the inferencing strategy.

Finally, and although our strategy assumes the existence of some ontology, the inferencing procedure itself is not committed to any particular knowledge representation. Nevertheless, our proposal for an inferential approach to resolving quantifier scope ambiguities does raise some important knowledge acquisition and knowledge representation questions that we will address in some detail in chapter 4.

1.5 STRUCTURE OF THIS THESIS

The rest of this thesis will be structured as follows:

- In chapter 2 we discuss Quantificational Constraints, which we argue is a piece of commonsense world knowledge that interpreters (i.e., readers) of natural language have recourse to, and which is sufficient to resolve scope ambiguities. In this chapter we will also show how Quantificational Constraints result in logical constraints that can be used to test the *plausibility* of a scope ordering using logical entailment.
- In chapter 3 we expand the notion of Quantificational Constraints considerably and develop a more general numeric procedure that not only can be used to predict the plausibility of a reading but also to rank all the plausible readings and suggest which, among these, is the *preferred* reading. Also in this chapter we will show how the numerical scoping algorithm can address a number of issues relating to quantifier scope,

such as accounting for generalized quantifiers, variations in the linguistic context, as well as the modal and temporal aspects.

- In chapter 4 we discuss a number of representational and computational issues leading to our discussion of the implementation in chapter 5.
- Chapter 5 discusses in some detail *Quasi*, our Java implementation of an inferential strategy for predicting the most plausible quantifier scope ordering.
- In chapter 6 we review some related work, and, finally, in chapter 7 we summarize and discuss some future work.

2 Quantificational Constraints

In chapter 1 we have made the argument that computational linguists must develop an inferential approach to resolving quantifier scope ambiguities. In this chapter we will discuss the nature of the world knowledge that might be required in this process, as well as the inferencing strategy itself.

2.1 RELATIONS AND CARDINALITY CONSTRAINTS

An important assumption that we make in this thesis is that speakers of natural language associate with a binary relation R between two concepts (represented extensionally as two sets) C_1 and C_2 a quantificational constraint (QC), that we define as:

$$(15) \quad \begin{aligned} & (QC(R, C_1, C_2) = \langle m_1, m_2 \rangle) \\ & \equiv_{df} \left(\left(\forall X \subseteq C_1 \exists y \in C_2 [R(X, y) \wedge (m_1 = |X|)] \right) \right. \\ & \quad \left. \wedge \left(\forall Y \subseteq C_2 \exists x \in C_1 [R(x, Y) \wedge (m_2 = |Y|)] \right) \right) \end{aligned}$$

where

m_1 = the number of elements of C_1 that are in relation R to the same element of C_2

m_2 = the number of elements of C_2 that the same element of C_1 could be related to by R

Intuitively, that is, $QC(R, C_1, C_2)$ is a pair of values $\langle m_1, m_2 \rangle$ where m_1 and m_2 are assumed to be either constant numeric values (e.g., *one*, *two*, *at least three*), standard linguistic quantifiers (e.g., *a*, *some*, *every*), or context-dependent (or fuzzy) linguistic quantifiers (e.g., *few*, *several*, *many*, etc.) Regarding these quantifiers, we will assume the following intuitive ordering:

$$(\text{few } C) < (\text{some } C) < (\text{several } C) < (\text{many } C) < (\text{most } C) < (\text{all } C)$$

i.e., we are suggesting that regardless of how much is *few men*, it is less than *several men*, and that in turn is less than *many men*, etc. Moreover, we will use the notation $\exists!(n)c \in C(\dots c \dots)$ instead of the longer $\exists C' \subseteq C(|C'|=n) \wedge \forall c \in C'(\dots c \dots)$ to say that there are at least n elements in C such that $(\dots c \dots)$, where n is a numeric or a linguistic quantifier.

What is potentially troubling in combining numeric and fuzzy linguistic quantifiers is that some way of comparing the two (e.g., *few men* and *3 men*) must be established. Concerning such comparisons we will assume the following:

$$(16) \quad (q\ C) = \begin{cases} 0.1\langle C \rangle < n \leq 0.2\langle C \rangle & \text{if } q = \text{few} \\ 0.2\langle C \rangle < n \leq 0.3\langle C \rangle & \text{if } q = \text{some} \\ 0.3\langle C \rangle < n \leq 0.4\langle C \rangle & \text{if } q = \text{several} \\ 0.4\langle C \rangle < n \leq 0.6\langle C \rangle & \text{if } q = \text{many} \\ 0.6\langle C \rangle < n \leq 0.9\langle C \rangle & \text{if } q = \text{most} \end{cases}$$

While these numbers are admittedly *ad-hoc*, they are simply suggested to 'transform' fuzzy linguistic quantification into some numerical value. The actual proportion itself is not crucial, as will become apparent in the next chapter.

2.2 QUANTIFICATIONAL CONSTRAINTS AND QUANTIFIER SCOPE

To motivate the use of quantificational constraints we discuss two examples. Consider the (physically located) *On* relation defined between a *house* and a *street*, and the *Eat* relation defined between a *child* and a *cookie*. We suggest that given an appropriately defined context, which we will shortly discuss, speakers of ordinary English must have a quantificational constraint defined on these relations. For the moment let us assume that these values exist in some individual's knowledge base as:

$$(17) \quad QC(Eat, Child, Cookie) = \langle 1, several \rangle$$

$$(18) \quad QC(On, House, Street) = \langle many, 1 \rangle$$

That is, we have assumed that some individual 'believes' that *typically* there are many houses located on a given street (and that no house can be on more than one street), and that a single child could eat several cookies, each of which is typically eaten by a single child. As the reader might have already observed, such quantificational constraints can only be reasonably accepted if they were defined on typical elements, since the existence of potentially an infinite number of exceptional cases makes such a representational assumption very questionable.

Fortunately, in resolving quantifier scope ambiguities, we need only consider typical elements, as will be demonstrated in chapter 4, where the representation of quantificational constraints and typical elements are discussed in some detail. For the moment, we need to point out that assuming that such quantificational constraints do exist in an individual's knowledge base, one must also consider a number of other parameters that a quantificational constraint must depend on. These are discussed next.

2.3 CONTEXT AND POSSIBLE WORLDS

The QC in (18) reflects some individual belief that a house can be physically located on one street, where there could be many. Clearly, this belief is a function of a possible world, as it is conceivable that some individual would associate a different QC in some fictional world where a house might be located on several (or all) streets. In our model, therefore, it is not crucial for a QC to have any specific value, but that such QCs *do exist* in the knowledge base. That is, we are committed to the assumption that understanding quantification *must involve the evaluation of a QC given a fully specified context*. In a sense, then, QC is a *character* (Kaplan, 1979), it needs a context to yield an interpretation. This possible need of a context of interpretation in understanding quantifiers is also suggested by (van der Does and Verkuy, 1996).

In our case, we will assume that the **context** is a pair $\langle kb, ind \rangle$ denoting a knowledge base associated with **some individual**. On this view, scope preferences are a function of a reader-based (Corriveau, 1995) value that could also change over time. For instance, it is not unreasonable to assume that by observation (by induction on the instances in the database, or by some other means), a child might come to believe that the *MarriedTo(Person, Person)* relation has the quantificational constraint $\langle 1, 1 \rangle$, i.e., that a person, if married, typically has one spouse, who also has one spouse. The same child might however at some point encounter cultures where it is not uncommon for a spouse to have more than one spouse. In that case, one would assume that the relevant QC is slightly modified. Depending on the data (the overall number of instances of the relation), the individual might even store several QCs, reflecting what is necessary, what is typical, and what is simply possible. These modal aspects are therefore very crucial to our formulation of a quantificational constraint and are discussed next. Before we turn to this discussion, however, we want to underline the individualistic character of QCs, which, in our view, is the reason why different readers tend to have different scope preferences for the *same linguistic context*, and why, over time, the *same individual* might have different scope preferences given the *same textual context*.

2.4 MODAL AND TEMPORAL ASPECTS OF QUANTIFICATIONAL CONSTRAINTS

2.4.1 The Role of Modality

Consider again the quantificational constraint given in (18), with a similar example of the (physically located) *On* relation:

(19) $QC(On, Book, Shelf) = \langle many, 1 \rangle$

(20) $QC(On, House, Street) = \langle many, 1 \rangle$

As in (19), the QC in (20) reflects some individual's belief that a book can be physically located on one shelf, where there could be many. However, there is an important difference between the two situations that is not captured by (19) and (20). While it might be *necessary* (or at least highly likely) for a house to be located on some street⁹, a book need not be on a shelf. There is a crucial distinction here that appears to affect the choice of quantifier scope. Let's consider the following:

(21) *John visited every house on some street*

(22) *John placed every book on some shelf*

While there is a tendency to assume a single street in (21) (by having *some* outscope *every*), the same does not seem to be the case in (22). Assuming it is *necessary* for a house to be located on some street, the direct reading of (21) does not add any significant information, and should be rejected on Gricean grounds¹⁰.

The case in (22) is quite different, however, since there is quite a bit of information in the reading where *every* outscopes *a* (since a book might very well be placed on a desk, or in a box, etc.) This additional dimension, that of the modality of a relation, therefore, seems to explain quantifier scope at a much deeper level. Note that when *a* precedes *every* in the surface structure, as in the following,

(23) *John visited a house on every street*

(24) *John placed a book on every shelf*

the quantifier scope ordering must be reversed in both cases on QC grounds, since neither a house nor a book can physically occupy more than one space.

In this dissertation we define the modality of a relation R , denoted to by $Mod(R)$, as follows:

$Mod(R) \in \{Nec, Typ, Pos\}$ where $Nec \supset Typ \supset Pos$

⁹ After all, a house might be located on the top of a mountain, or in the middle of the Nile river.

¹⁰ Moran (1988) also incorporates a conversational maxim of quantity in his preference rules. However, this preference must be formalized so that similar situations can be explained.

Note the **implications between the modalities**: assuming it is *necessary* for a house to be on one street, then it is *typical*, and, trivially, *possible*, although the reverse is not true. We discuss the role these modalities play in defining a quantificational constraint, and subsequently in inferring the most plausible scope ordering in the next chapter.

2.4.2 Temporal Aspects

In addition to the modality of a relation, quantificational constraints also seem to be a function of the temporal aspects. To illustrate, consider the following QC defined on the *Lift* relation between *man* and *piano*¹¹:

$$(25) \quad QC(Lift, Man, Piano) = \langle 2^+, 1 \rangle$$

This QC reflects some individual's belief that it takes *at least* two men to lift a single piano¹². Clearly, this QC assumes a single lifting event, since two men can lift more than one piano at different points in time. This temporal aspect is very crucial to quantification and has in fact been investigated extensively, most notably by (Verkuyl, 1989). To see the effect of the temporal aspect on the available scope preferences, consider the following:

(26) *Two men lifted four pianos to the second floor*

(27) *Four men lifted two pianos to the second floor*

Given the QC in (25), the two possible scope orderings in (27) would be allowed. That is, it is conceivable that of the four men, each pair of two lifted one of the pianos *at the same time*, or that the four (collectively) lifted the two pianos, one piano at a time. The reading that most readers would choose for (26), however, can only be explained by admitting at least four lifting events, since it will take at least two to lift each piano.

Scope preferences can also be a function of temporal information that is available at the syntactic level, as in the following:

(28) *Every student is sitting on a leather chair in the dean's office*

(29) *Every student sat on a leather chair in the dean's office*

¹¹ Throughout we take n^+ to mean *at least* n (or n or more), and n^- to mean *at most* n (or n or less).

¹² Again, the reader should keep in mind that these QCs are assumed to be defined on *typical* elements, since a 100 men with a handicap cannot lift a piano, nor can a million men that died in the 1800's. We will discuss the notion of a *typical element* in chapter 4.

While it is not likely that all students can be sitting on the same leather chair at the same point in time (i.e., a wide scope a in (28) does not seem to be plausible), it is quite possible that all the students have, at some point in the past, sat on the same (and only) leather chair in the dean's office. Regarding the temporal aspect of a relation, therefore, we will assume that a relation between two concepts, when referring to an event, can refer to a single ($\uparrow$) or multiple ($\uparrow\uparrow$) events. We further assume that $\uparrow\uparrow \supset \uparrow$. That is, a relation that describes multiple occurrences of an event also describes individual occurrences.

To summarize the above discussion, a quantificational constraint, which is a piece of commonsense knowledge that is associated with a binary relation defined on two concepts, can be viewed as a function that takes a context and returns a pair of cardinality constraints. The context is a possible world (in our case an individual's knowledge base) as well as the modal and temporal aspects of the relation. In the next section we will demonstrate how this simple formulation can be used in an inferencing strategy for predicting the most plausible scope ordering(s).

2.5 REASONING WITH QUANTIFICATIONAL CONSTRAINTS

Our goal in this section is twofold. First, we will try to 'tie' the quantificational constraint, which is a numeric constraint defined on two concepts and a binary relation, with the quantifiers of first-order logic. Second, we will present a simple algorithm that shows how quantificational constraints can be used in an inferencing process to determine the plausibility of readings.

2.5.1 Quantificational Constraints as Logical Constraints

In this section we will show how quantificational constraints, which are cardinality constraints on a binary relation, can be used to define a set of logical constraints that can be used to test the plausibility of a logical form.

We proceed with an example.

Consider the binary relation *BuiltOn* between houses and streets, with the quantificational constraint $QC(BuiltOn, House, Street) = \langle many, 1 \rangle$, meaning that a house can be built on one street only, but *many* (which for now we take to mean one or more) houses can be built on the same street. This relation can be graphically depicted as shown in Figure 2 below.

Figure 2. The *BuiltOn* relation between houses and streets

Note that, in general, the relation does not involve all the houses and streets in the universe, but selected subsets from the domain of discourse, which we denote by H_i and S_j respectively, as shown in Figure 2.

What the quantificational constraint (QC) that we have assumed does is *place some conditions on how the sets can be generated in the context of the BuiltOn relation*. Specifically, QC implicitly defines an *equivalence relation* between the elements of each set. In the above example, i.e., in the context of the *BuiltOn* relation between *houses* and *streets*, elements in the set of houses are implicitly related to each other by the relation *OnSameStreet*. Now note that this relation is:

- **reflexive, since**
 $\forall h \in \text{House}(\text{OnSameStreet}(h, h))$
- **symmetric, since**
 $\forall h_1, h_2 \in \text{House}(\text{OnSameStreet}(h_1, h_2) \rightarrow \text{OnSameStreet}(h_2, h_1))$
- **transitive, since**
 $\forall h_1, h_2, h_3 \in \text{House}((\text{OnSameStreet}(h_1, h_2) \wedge \text{OnSameStreet}(h_2, h_3)) \rightarrow \text{OnSameStreet}(h_1, h_3))$

Therefore, $QC(\text{BuiltOn}, \text{House}, \text{Street})$ defines a *partition* on the set of houses and the set of streets, that can be defined as follows:

$$(30) \quad \begin{aligned} P_{\text{House}} &= \{H \subseteq \text{House} \mid (1 \leq |H| \leq \text{many}) \wedge \exists (1)s \in \text{Street}(\text{BuiltOn}(H, s))\} \\ P_{\text{Street}} &= \{S \subseteq \text{Street} \mid (1 = |S|) \wedge \exists (\text{many})h \in \text{House}(\text{BuiltOn}(h, S))\} \end{aligned}$$

Intuitively, what these definitions say is the following: in the context of the *BuiltOn* relation between *houses* and *streets*, we can only speak of subsets of houses (of size *many*) that are

related to *one* street, and of subsets of streets (of size *one*) that are related to *many* houses. Let us denote the predicates that enforce these conditions by

$$(31) \quad \begin{aligned} P_1 &= (1 \leq |H| \leq \text{many}) \wedge \exists! (1) s \in \text{Street}(\text{BuiltOn}(H, s)) \\ P_2 &= (1 = |S|) \wedge \exists! (\text{many}) h \in \text{House}(\text{BuiltOn}(h, S)) \end{aligned}$$

This is a crucial step in our formulation of a commonsense inferencing strategy for selecting the most plausible scope ordering and thus it is worthy of further elaboration. Initially, we assumed the existence of piece of background knowledge that we termed 'quantificational constraint'. As we argued above, these quantificational constraints enforce certain rules on how subsets in a given domain can be generated in a certain context. In other words, a quantificational constraint defines all the plausible ways a certain relation can be manifested.

As an example, the partitions that are implied by $QC(\text{BuiltOn}, \text{House}, \text{Street})$ are the unions of subsets that can only be generated by the predicates given in (31). How could this be used in inferential scoping algorithm?

Let us take our first look at how conditions imposed by a quantificational constraint, such as those in (31), can be used to decide on the plausibility of a reading. Consider the simple sentences:

(32) *A house is built on every street*

(33) *Every house is built on a street*

Using a simple set-theoretic interpretation¹³ the processing of the *direct readings* of (32) and (33) proceeds as follows:

$$\begin{aligned} & \text{A house is built on every street} \\ & \Rightarrow A(\text{House}, \text{house } h_i \text{ such that } h_i \text{ is built on every street}) \\ 32' & \Rightarrow A(\text{House}, \exists H \subseteq \text{House}((1 \leq |H| \leq a) \wedge \forall s \in \text{Street}(\text{BuiltOn}(H, s)))) \\ & \Rightarrow A(\text{House}, \exists H \subseteq \text{House}((1 = |H|) \wedge \forall s \in \text{Street}(\text{BuiltOn}(H, s)))) \\ & \text{Every house is built on a street} \\ 33' & \Rightarrow \text{Every}(\text{House}, \text{house } h_i \text{ such that } h_i \text{ is built on a street}) \\ & \Rightarrow \text{Every}(\text{House}, \exists H \subseteq \text{House}((1 = |H| \leq \text{many}) \wedge \exists s \in \text{Street}(\text{BuiltOn}(H, s)))) \end{aligned}$$

¹³ In a set-theoretic interpretation $A(X, Y)$ is true iff $(X \cap Y) \neq \emptyset$ and $\text{Every}(X, Y)$ is true iff $X \subseteq Y$.

Note that a direct reading of 32' (i.e., a reading where the scope of quantifiers is that implied by the surface structure) results in generating a set of houses (of size one) that are on **every** street, which is inconsistent with the conditions imposed by the quantificational constraint in (31). The predicates in (31), however, will always imply the predicate generating the set of houses in 33'. That is, according to the conditions imposed by the quantificational constraint, the set of houses that must be generated for a direct reading of (32) does not seem to be plausible, while a direct reading of 33' is quite consistent with conditions imposed by the quantificational constraint. Since a direct reading of (32) does not seem to be plausible, let us look at the reading with a wide scope *every*. The processing in this case proceeds as follows:

Every street s_i a house is built on s_i
 $\Rightarrow \text{Every}(\text{Street}, \text{street } s_i \text{ such that a house is built on } s_i)$
 $\Rightarrow \text{Every}(\text{Street}, \exists S \subseteq \text{Street} ((1 \leq |S| \leq \text{many}) \wedge \exists h \in \text{House}(\text{BuiltOn}(h, S))))$

The translation in this case could be accepted, since the predicate generating the set of streets in this case is always implied by the corresponding predicate in (31). The truth-value obtained here is *true*, just in case on every street at least one house is built, which is a more plausible reading of (32).

What has happened here, essentially, is that we have managed to find a **systematic way of testing the plausibility of a certain reading**, by selecting readings that are *consistent with the conditions imposed by the quantificational constraint*¹⁴. The above discussion can be formalized in the procedure of Figure 3 below. Before we extend the procedure considerably we consider another example where the quantificational constraint seems to allow more plausible situations than the restricted QC of the previous example. In Figure 4 below we make the following assumptions in the processing of *Most students speak at least two languages*:

- $QC(\text{Speak}, \text{Student}, \text{Language}) = \langle \text{all}, \text{few} \rangle$
- $\exists L \subseteq \text{Language} ((1 \leq |L| \leq \text{few}) \wedge \forall l \in L (\text{Speak}(s, l))) \equiv \exists! (\text{few } L) \in \text{Language} (\text{Speak}(s, l))$
- *Existential import, i.e., we are assuming sets have at least one element*
- $(\forall x \in X (P(x))) \equiv \exists! (\text{all}) x \in X (P(x)) \rightarrow \dots \rightarrow \exists! (\text{few}) x \in X (P(x)) \rightarrow \dots \rightarrow \exists! (1) x \in X (P(x))$

¹⁴ The reader can easily verify that using the same line of reasoning both readings of 'Every house was built on two streets' seem to be equally implausible. However, in 'Many houses were built on two streets' (i.e., if we simply replace *every* by *many*), a reading with wide scope *two* seems to be quite plausible: there are 2 streets, where on both (or, collectively on each) of them *many* houses were built! It is important to note however that when we say 'the reader can verify' we are speaking of *plausibility* and not *validity*. That is, the result of this 'verification' will depend on the quantificational constraint(s) the reader has in mind, as it should be evident by now that different QCs result in different scope preferences.

Given a sentence $S = [{}_S[{}_{NP}[_{Det} Q_1]X][{}_{VP}[_V R][{}_{NP}[_{Det} Q_2]Y]]$

- Construct two logical forms LF_1 and LF_2 corresponding to the direct and indirect readings respectively, as follows
 $LF_1 = Q_1(X, \exists X' \subseteq X(P^\triangleright))$ where $P^\triangleright = (1 \leq |X'| \leq Q_1) \wedge (Q_2)_Y(R(X', Y))$
 $LF_2 = Q_2(Y, \exists Y' \subseteq Y(P^\triangleleft))$ where $P^\triangleleft = (1 \leq |Y'| \leq Q_2) \wedge (Q_1)_X(R(X, Y'))$
- $QC(R, C_1, C_2) = \langle m_1, m_2 \rangle$ is the relevant QC if $isA(X, C_1)$ and $isA(Y, C_2)$
- $QC(R, C_1, C_2) = \langle m_1, m_2 \rangle$ defines two partitions
 $P_{C_1} = \{X \subseteq C_1 | P_1\}$ where $P_1 = (1 \leq |X| \leq m_1) \wedge \exists! (m_2) y \in C_2 (R(X, y))$
 $P_{C_2} = \{Y \subseteq C_2 | P_2\}$ where $P_2 = (1 \leq |Y| \leq m_2) \wedge \exists (m_1) x \in C_1 (R(x, Y))$
- $r_1 = \text{'direct reading is plausible'}$ $r_2 = \text{'indirect reading is plausible'}$
 if (S is in active form) $r_1 = (P_1 \rightarrow P^\triangleright)$
 $r_2 = (P_2 \rightarrow P^\triangleleft)$
 else $r_1 = (P_2 \rightarrow P^\triangleright)$
 $r_2 = (P_1 \rightarrow P^\triangleleft)$

Figure 3. Using entailment in FOL to test the plausibility of a reading

That is, we have assumed that some individual believes that a given language can be spoken by any number of students (i.e., *all*), and that very few languages are spoken by the same student. Second, we have used a simple notation for 'there exists a few X' which hides all the details of the inequalities defined in (16). Finally, we are assuming *existential import* since empty sets can be treated outside the scope algorithm altogether. This, of course, makes the last assumption valid.

Two points should be made regarding the example in Figure 4. First, unlike the previous example of the *BuiltOn* relation between houses and streets, the logical constraints imposed by $QC(Speak, Student, Language)$ seem to allow many more plausible situations.

For instance, while we could not speak of a single house on more than one street, it seems quite plausible to speak of any number of students speaking any number of languages¹⁵. Secondly, while both the direct and indirect readings are plausible, the meanings implied by

¹⁵ Here we have assumed that the $QC(Speak, Student, Language)$ reflects what is *possible*, and therefore anything that is possible will be considered by the algorithm to be plausible. Plausibility would have a different interpretation if we started with a QC that reflected what is *typical* or *necessary*. There will be more to say about this in the next chapter.

each are quite different. In keeping with our requirement that a computational system must ultimately decide on a single meaning, a method must be devised so that the scoping algorithm can select a **preferred reading** when more than one is plausible.

$$\begin{aligned}
 &S = \text{Most students speak at least 2 languages} \\
 &\bullet \text{ LF}_1 = \text{Most}(\text{Student}, \exists S \subseteq \text{Student}(P^\triangleright)) \\
 &\text{LF}_2 = 2^+(\text{Language}, \exists L \subseteq \text{Language}(P^\triangleleft)) \\
 &\text{where} \\
 &P^\triangleright = (1 \leq |S| \leq \text{most}) \wedge \exists (2^+) \in \text{Language}(\text{Speak}(s, l)) \\
 &P^\triangleleft = (1 \leq |L| \leq 2^+) \wedge \exists (\text{most}) s \in \text{Student}(\text{Speak}(s, l)) \\
 &\bullet \text{ Quantificational constraint is } QC(\text{Speak}, \text{Student}, \text{Language}) = \langle \text{all}, \text{few} \rangle \\
 &\bullet QC \text{ defines two partitions } P_{C_1} \text{ and } P_{C_2} \text{ as follows} \\
 &P_{C_1} = \{S \subseteq \text{Student} \mid P_1\} \\
 &P_{C_2} = \{L \subseteq \text{Language} \mid P_2\} \\
 &\text{where} \\
 &P_1 = (1 \leq |S| \leq \text{all}) \wedge \exists (\text{few}) l \in \text{Language}(\text{Speak}(s, l)) \\
 &P_2 = (1 \leq |L| \leq \text{few}) \wedge \exists (\text{all}) s \in \text{Student}(\text{Speak}(s, l)) \\
 &\bullet \text{ Sentence is in active form} \\
 &\Rightarrow r_1 = (P_1 \rightarrow P^\triangleright) = \left(\begin{array}{l} (1 \leq |S| \leq \text{all}) \wedge \exists (\text{few}) l \in \text{Language}(\text{Speak}(S, l)) \\ \rightarrow (1 \leq |S| \leq \text{most}) \wedge \exists (2^+) \in \text{Language}(\text{Speak}(S, l)) \end{array} \right) = \text{true} \\
 &\Rightarrow r_2 = (P_2 \rightarrow P^\triangleleft) = \left(\begin{array}{l} (1 \leq |L| \leq \text{few}) \wedge \exists (\text{all}) s \in \text{Student}(\text{Speak}(s, L)) \\ \rightarrow (1 \leq |L| \leq 2^+) \wedge \exists (\text{most}) s \in \text{Student}(\text{Speak}(s, L)) \end{array} \right) = \text{true} \\
 &\therefore (r_1 \wedge r_2), \text{ both the direct and indirect readings are plausible}
 \end{aligned}$$

Figure 4. Applying the scoping algorithm on *Most students speak at least two languages*

2.5.2 A Brief Interim Summary

The main objective behind the above exercise was to show what has been achieved by assuming quantificational constraints on concepts and relations. Essentially, a QC imposes logical constraints on how sets could be generated in a certain **linguistic context**. Readings that are consistent with the conditions imposed by the relevant QC are considered plausible, while others are rejected.

Although logical reasoning is used in this inferencing process, it is very crucial to keep in mind that readings are not being rejected on logical grounds. Readings that are inconsistent with some *pragmatic considerations* are being rejected, and thus the **scoping rules do not test the validity but the plausibility** of a logical form.

In a sense, then, we are trying to deal with cognitive plausibility, while other attempts such as those of (Fox, 1995; Reinhart, 1997) could be thought of as dealing with syntactic plausibility, and those of (Hobbs and Shieber, 1987; Park, 1995) as dealing with logical/semantic plausibility.

Finally, it should be noted that although we have managed to use the conditions imposed by quantificational constraints to determine the plausibility of a reading, we do not yet have a decision procedure that infers the *most* plausible (or the preferred) reading. For example, while both readings in Figure 4 were considered plausible, it is not readily obvious how we could determine the *most* plausible reading, which might be crucial when the two readings are not **equally** plausible, i.e., when the two readings have different meanings.

Thus far, plausibility has been determined by logical entailment, and thus we cannot easily determine the *degree* to which some reading is plausible, since we cannot speak of one implication being *stronger* than another. Fortunately, however, the simple algorithm described above can be used as a starting point to discover a more descriptive numerical algorithm, which we discuss in the next chapter.

3 A Numerical Scoping Algorithm

In the previous chapter we used the logical constraints imposed by a quantificational constraint and logical entailment to describe a simple scoping algorithm. In this chapter we will extend the scoping algorithm considerably. In doing so we will move from an algorithm that uses logical constraints and entailment to one that uses numerical constraints and simple numerical operations.

The numerical method has several advantages over the logical method. In particular, it (i) simplifies the modeling of generalized quantifiers (e.g., *at most six*); (ii) provides a method by which we could rank the plausible readings and select a *preferred* reading; and (iii) facilitates accounting for the linguistic context where the size of the quantified sets plays an important role.

3.1 QUANTIFICATIONAL CONSTRAINTS AS NUMERICAL CONSTRAINTS

When modeled in a logical process, quantificational constraints were thought of as imposing some logical constraints on how subsets from two domains can be related. In this section we will proceed from these logical constraints to discover **a set of numerical constraints**. Consider again the conditions imposed by a quantificational constraint:

$$\begin{aligned}
 & \text{A quantificational constraint } QC(R, C_1, C_2) = \langle m_1, m_2 \rangle \\
 & \text{defines two partitions} \\
 (34) \quad & P_{C_1} = \{X \subseteq C_1 \mid P_1\} \text{ where } P_1 = (1 \leq |X| \leq m_1) \wedge \exists!(m_2)y \in C_2 (R(X, y)) \\
 & P_{C_2} = \{Y \subseteq C_2 \mid P_2\} \text{ where } P_2 = (1 \leq |Y| \leq m_2) \wedge \exists!(m_1)x \in C_1 (R(x, Y))
 \end{aligned}$$

Informally, the predicate P_1 in (34) is enforcing the constraint imposed by $QC(R, C_1, C_2) = \langle m_1, m_2 \rangle$ that there could be up to m_2 C_2 that could be related by R to any subset of C_1 of size up to m_1 . Stated yet in other words, P_1 in (34) says there must be (m_2/m_1) C_2 for every C_1 , and similarly for P_2 . More formally, therefore, P_1 and P_2 can be stated as:

$$\begin{aligned}
 (35) \quad & P_{C_1} = \{X \subseteq C_1 \mid P_1\} \text{ where } P_1 = \exists!(n_1)y \in Y(R(X, y)), \quad 1 \leq n_1 \leq (m_2 / m_1) |C_1| \\
 & P_{C_2} = \{Y \subseteq C_2 \mid P_2\} \text{ where } P_2 = \exists!(n_2)x \in X(R(x, Y)), \quad 1 \leq n_2 \leq (m_1 / m_2) |C_2|
 \end{aligned}$$

Intuitively, what P_1 in (35) says is the following: m elements from C_1 could always be related by R to n elements of C_2 , as long as $1 \leq n \leq (m_2/m_1)m$; where (m_2/m_1) is integer division.

To get a feel of how these predicates enforce the conditions imposed by a quantificational constraint, let us consider $QC(Lift, Person, Piano) = \langle 2^+, 1 \rangle$, reflecting some individual's belief that it takes at least two men to lift a piano. According to (34), the subsets of *men* and *pianos* that can be related to each other by the *lift* relation must be generated as shown in table 1 below.

<i>if there are</i>	<i>they can lift</i>	<i>to lift</i>	<i>there must be</i>
1 <i>man</i>	$\left(1 / 2^+\right) 1 = 0 \text{ pianos}$	1 <i>piano</i>	$\left(2^+ / 1\right) 1 = 2^+ \text{ men}$
2 <i>men</i>	$\left(1 / 2^+\right) 2 = 1^- \text{ pianos}$	2 <i>pianos</i>	$\left(2^+ / 1\right) 2 = 4^+ \text{ men}$
3 <i>men</i>	$\left(1 / 2^+\right) 3 = 1^- \text{ pianos}$	3 <i>pianos</i>	$\left(2^+ / 1\right) 3 = 6^+ \text{ men}$
4 <i>men</i>	$\left(1 / 2^+\right) 4 = 2^- \text{ pianos}$	4 <i>pianos</i>	$\left(2^+ / 1\right) 4 = 8^+ \text{ men}$
<i>etc.</i>	<i>etc.</i>	<i>etc.</i>	<i>etc.</i>

Table 1. Cardinality constraints on relations and sets

For example, if there are four men, then we could speak of up to (*at most*) two pianos being lifted. On the other hand, if we were speaking of three pianos being lifted, then there must be *at least* six men, and so on.

The exact same reasoning can also be used to redefine the predicates of the direct and indirect readings of a sentence which were given in the algorithm of Figure 3, as:

$$\begin{aligned}
 LF_1 &= Q_1 \left(X \subseteq C_1, \{x \in X \mid P^D\} \right) \\
 LF_2 &= Q_2 \left(Y \subseteq C_2, \{y \in Y \mid P^I\} \right) \\
 (36) \quad &\text{where} \\
 P^D &= \exists! (n_1) y \in Y (R(X, y)), \quad 1 \leq n_1 \leq (Q_2 / Q_1) |X| \\
 P^I &= \exists! (n_2) x \in X (R(x, Y)), \quad 1 \leq n_2 \leq (Q_1 / Q_2) |Y|
 \end{aligned}$$

Having redefined the predicates imposed by a quantificational constraint and those of the direct and indirect reading of a sentence in numerical terms we can now re-interpret logical entailment (which was used as a test for the plausibility of a scope ordering) numerically.

Before we proceed, however, we need to recall that the direct reading is considered plausible if $(P_1 \rightarrow P^D)$, whereas the indirect reading is considered plausible when $(P_2 \rightarrow P^I)$. Using (34) and (35) we can now easily proceed with transforming the entailment rules of Figure 3 into

numerical comparisons. In (37) below we consider the case of the active form of a sentence and detail the transformation of $r_1 = \text{'direct reading is plausible'}$, which is true if $(P_1 \rightarrow P^\circ)$.

$$\begin{aligned}
 & \text{let } r_1 = (P_1 \rightarrow P^\circ) = \text{direct reading is plausible} \\
 & \equiv \exists! \left(1 \leq n \leq \frac{m_2}{m_1} |C_1| \right) y \in Y(R(x, y)) \rightarrow \exists! \left(1 \leq n \leq \frac{Q_2}{Q_1} |X| \right) y \in Y(R(x, y)) \\
 & \text{let } n_1 = \frac{m_2}{m_1} |C_1|, n_2 = \frac{Q_2}{Q_1} |X|, \text{ and } P(y) = R(x, y) \quad (x \text{ is a free variable in } P_1 \text{ and } P^\circ) \\
 (37) \quad & \text{then} \\
 & r_1 = \exists! (1 \leq n \leq n_1) y \in Y(P(y)) \rightarrow \exists! (1 \leq n \leq n_2) y \in Y(P(y)) \\
 & \equiv (|P| = \text{at least } n_1) \rightarrow (|P| = \text{at least } n_2) \\
 & \equiv (n_1 \geq n_2) \\
 & \therefore r_1 = 1 \leq \left(\frac{Q_2}{Q_1} |X| \right) \leq \left(\frac{m_2}{m_1} |C_1| \right)
 \end{aligned}$$

Transforming the entailment rules for the case of an indirect reading and those of the passive form are almost identical to (37), and need not be repeated here. The two rules of entailment in Figure 3 can now be replaced by the following rules:

$$\begin{aligned}
 (38) \quad & r_1 \equiv 1 \leq \left(\frac{Q_2}{Q_1} |X| \right) \leq \left(\frac{m_2}{m_1} |C_1| \right) \\
 & r_2 \equiv 1 \leq \left(\frac{Q_1}{Q_2} |Y| \right) \leq \left(\frac{m_1}{m_2} |C_2| \right)
 \end{aligned}$$

These new rules can be used to do much more than test the plausibility of a reading, which the simple implication rules of Figure 3 did. Before that can be demonstrated, however, we need to explain how the inequalities in (38) should be interpreted. This is done next.

3.1.1 A Space of Plausible Situations

Consider (again!) the quantificational constraint $QC(\text{BuiltOn}, \text{House}, \text{Street}) = \langle \text{many}, 1 \rangle$, which could be defined on the *BuiltOn* relation between houses and streets. This QC defines more than one pair of possible combinations, i.e., it does not say that we can only speak of *many* houses on *one* street, but of any number of houses (1 up to *many*) on a single street.

Similarly, $QC(\text{Speak}, \text{Student}, \text{Language}) = \langle \text{all}, \text{few} \rangle$ says that any number of students (between 1 and *all*) could speak any number of languages (1 up to a *few*). In other words, the

numerical constraint imposed by a QC is not a single pair of values, but a 2-dimensional space. We can also see this from the conditions that are imposed by a quantificational constraint as given in (35):

According to (34), c_1 and c_2 , the number of C_1 and C_2 elements respectively, can be defined as follows

$$c_1 = \left(\frac{m_1}{m_2} |C_2| \right) \quad c_2 = \left(\frac{m_2}{m_1} |C_1| \right)$$

What quantificational constraints do, therefore, is define a 2-dimensional space of plausible combinations of quantifiers. The plausible points in that space, for some very common quantificational constraints, are depicted graphically in figure 5 below, where the shaded area represents all the points $(Q1, Q2)$ such that $Q1(x)Q2(y)[R(x,y)]$ is a plausible scope ordering.

Figure 5. The space of plausible combinations defined by a quantificational constraint.

As is intuitively clear from figure 5, a *many-to-many* quantificational constraint allows many more plausible situations than a more restricted (e.g., *many-to-1*) QC. Given this interpretation of quantificational constraints, the conditions in (38) should be read as¹⁶:

$$(39) \quad \begin{aligned} r_1 = 1 &\leq \left(\frac{Q_2}{Q_1} |X| \right) \leq \left(\frac{[1, \dots, m_2]}{[1, \dots, m_1]} |C_1| \right) \\ r_2 = 1 &\leq \left(\frac{Q_1}{Q_2} |Y| \right) \leq \left(\frac{[1, \dots, m_1]}{[1, \dots, m_2]} |C_2| \right) \end{aligned}$$

¹⁶ In relational (Dullea and Song, 1997) as well as in object data models (Rumbaugh *et al.*, 1991), cardinality constraints are also modeled by two pairs of values, where each pair essentially defines a range $[min, \dots, max]$.

3.1.2 The Linguistic Context

Note that in (39) r_1 and r_2 are functions of the sets C_1 and C_2 , corresponding to the general concepts in the knowledge base, as well as $X \subseteq C_1$ and $Y \subseteq C_2$ that correspond to the noun phrases in a given sentence. Depending on the linguistic context, the size of X and Y could be a small fraction of the original concepts, since noun phrases do not always refer to simple nouns, but can in general be very complex, involving any number of adjectives, prepositional phrases, relative clauses, etc. As it turns out, these variations in the linguistic context play a critical role in the apparent scope preferences, as will be demonstrated shortly. The following example demonstrates how the various sets in (39) are identified in a specific linguistic context. First, a sentence is parsed producing the following structure¹⁷:

$$\text{Parse}(\text{Every student at MIT submitted a paper on quantification to ACL'95})$$

$$= \left[\begin{array}{l} S \left[\begin{array}{l} NP_1 \left[\begin{array}{l} Det \text{ "Every"} \\ TP \left[\begin{array}{l} N_h \text{ "student"} \\ PP \text{ "at MIT"} \end{array} \right] \end{array} \right] \end{array} \right] \\ \\ VP \left[\begin{array}{l} V \text{ "submitted"} \\ NP_2 \left[\begin{array}{l} Det \text{ "a"} \\ TP \left[\begin{array}{l} N_{vp} \text{ "paper"} \\ PP \text{ "on quantification"} \end{array} \right] \end{array} \right] \\ PP \text{ ["to ACL'95"]} \end{array} \right] \end{array} \right]$$

Given this parse, the relevant concepts and quantificational constraint can be identified as follows:

$$\begin{aligned} C_1 &= \text{GetConcept}(\text{GetHeadNoun}(NP_1)) = \text{GetConcept}(\text{"student"}) = \text{Student} \\ C_2 &= \text{GetConcept}(\text{GetHeadNoun}(NP_2)) = \text{GetConcept}(\text{"paper"}) = \text{Paper} \\ R &= \text{GetRelation}(\text{GetVerb}(VP)) = \text{GetRelation}(\text{"submitted"}) = \text{Submit} \\ \therefore QC &= \text{GetQC}(R, C_1, C_2) = \text{GetQC}(\text{Submit}, \text{Student}, \text{Paper}) \end{aligned}$$

Note, therefore, that the sets corresponding to the relevant knowledge base concepts, i.e., C_1 and C_2 are always associated with the nouns, not the noun phrases, which are not always simple nouns but could be modified by adjectives, prepositional phrases, relative clauses,

¹⁷ As will be discussed in more detail in the next chapter, prepositional phrase (PP) attachments are hard-coded in our system. That is, in the current implementation all possible PP attachments are considered valid readings that our scope resolution algorithm could take as input.

etc. Note also that the function $\text{GetConcept}(N)$ retrieves the relevant concept from the knowledge base based on the lexical item N ¹⁸. In retrieving the relevant concept (meaning) for a lexical item we have not been concerned with lexical ambiguity. Our ‘dictionary’ is a simple hashtable where a single concept is associated with every lexical item.

Having identified the two concepts and the relation, the relevant quantificational constraint can be retrieved. However, while the quantificational constraint might be defined on the *Submit* relation between *Student* and *Paper*, the linguistic context here refers to a specific set of students and papers. That is, the sets X and Y in (39) can be identified as follows:

$$\begin{aligned} X &= \text{Student} \cap (\text{Student at MIT}) \\ Y &= \text{Paper} \cap (\text{Paper on quantification}) \\ \therefore (|X| \leq |C_1|) \wedge (|Y| \leq |C_2|) \end{aligned}$$

From basic set theory it follows that the extension of a modified concept will always be a fraction of the size of the original concept. That is, the size of the sets referred to in the linguistic context will always be a fraction of the size of the concept in the KB. (39), therefore, can be further simplified to the following:

$$(40) \quad \begin{aligned} r_1 = 1 &\leq \left(\frac{Q_2}{Q_1} \frac{1}{a} \right) \leq \left(\frac{v_2}{v_1} = \frac{[1, \dots, m_2]}{[1, \dots, m_1]} \right) \text{ where } |Y| = \frac{1}{b} |C_2| \text{ and } b \geq 1 \\ r_2 = 1 &\leq \left(\frac{Q_1}{Q_2} \frac{1}{b} \right) \leq \left(\frac{v_1}{v_2} = \frac{[1, \dots, m_1]}{[1, \dots, m_2]} \right) \text{ where } |X| = \frac{1}{a} |C_1| \text{ and } a \geq 1 \end{aligned}$$

That is, the size of the concept implied by the linguistic context is always a fraction of (i.e., is always less than or equal to) the size of the original concept in the knowledge base. Note that in a void context, *student* and *advisor* in *Every student has an advisor* refers to all the *students* and all the *advisors* in the database, and a and b in (40) are both 1.

This also brings us to the question of why are the parameters a and b in (40) thought of as capturing the ‘linguistic context’ while thus far they actually only capture the ‘sentential context’? The reason for this is that while for now these parameters are functions of the immediate (sentential context), in a scaled-up solution where larger linguistic fragments are

¹⁸ Lexical ambiguity is not an issue we want (or need) to be concerned with here. However, in *Every student and faculty at MIT submitted a paper to ACL* the function $\text{GetConcept}(\text{"student and faculty"})$ is not a simple lookup for a lexical item, but is the least upper bound (*lub*) of *Student* and *Faculty* in the concept hierarchy.

considered, the values of these parameters might be computed as functions that are dependent on extra-linguistic parameters. In particular, these parameters will have to reflect a wider context when the interaction between quantification and reference resolution is considered. While this is not an issue we need to be concerned with here, when processing large fragments of text it is rarely the case that the *reference* of a concept is to all the instances in the universe (or in the domain of discourse), but to a subset that has been properly circumscribed by the surrounding context (or, as Cooper (1995) calls it “the background situation”). Clearly, the issue here is the interaction between quantifier scope and reference resolution, which is beyond the scope of this investigation. This is indeed part of the future work that we plan to investigate, and will be briefly discussed in chapter 7.

We now turn to the question of what do the inequalities in (40) mean? Specifically, what does it mean to compare ratios that include ranges of values? To get at a feel for what these inequalities might mean in some linguistic context, we consider two examples (the process of applying these rules will be explained in some detail here as these rules will be repeatedly applied throughout the remainder of the thesis):

- (41) a. *Every student attended a seminar*
 b. *Every student of Dr. Miller attended a seminar on Chaos Theory*

A reasonable quantificational constraint that is relevant to these examples might be $QC(Attend, Student, Seminar) = \langle 1^+, 1^+ \rangle$, stating that any number of students can attend the same seminar, and that any number of seminars can be attended by the same student. Applying the rules in (39) on (41a) now yields the following:

$$\begin{aligned}
 r_1 &= \left(1 \leq \frac{Q_2}{Q_1} |X| \leq \frac{m_2 = 1^+}{m_1 = 1^+} |C_1| \right) \\
 &= \left(1 \leq \frac{a}{every} |Student| \leq \frac{m_2 = [1, \dots, all]}{m_1 = [1, \dots, all]} |Student| \right) \\
 &= \left(\frac{1}{all} \leq \frac{m_2 = [1, \dots, all]}{m_1 = [1, \dots, all]} \right) = true, \forall m_1, m_2 \\
 41a' \Rightarrow \\
 r_2 &= \left(1 \leq \frac{Q_1}{Q_2} |Y| \leq \frac{m_1 = 1^+}{m_2 = 1^+} |C_2| \right) \\
 &= \left(1 \leq \frac{every}{a} |Seminar| \leq \frac{m_1 = [1, \dots, all]}{m_2 = [1, \dots, all]} |Seminar| \right) \\
 &= \left(\frac{all}{1} \leq \frac{m_1 = [1, \dots, all]}{m_2 = [1, \dots, all]} \right) = true \text{ only when } (m_1, m_2) = (all, 1)
 \end{aligned}$$

Note that a and b in the inequalities in (40) are both 1 here, since the nouns in the simple sentence of 41a are not modified, and thus their denotations (sets) are equal to those of the original concepts in the knowledge base. What is more important to note here is the result of applying the inequality tests, keeping in mind that the direct and indirect readings are considered to be plausible whenever r_1 and r_2 are true, respectively. What 41a' suggests is that the direct reading seems to be plausible for any combination of m_1 and m_2 , while the indirect reading is consistent with only one combination in that space of plausible situations, namely ($Q_1=all, Q_2=1$). This situation corresponds to a state of affairs where indeed there is a single *seminar* that *every student* attended, which is in the realm of possibilities. However, given the relevant background knowledge (i.e., the relevant QC), it would seem that the direct reading, implying that *every student attended 'some' seminar* is more plausible. Thus, while both the direct and indirect readings are in fact *plausible*, it is clear that the two readings are not **equally** plausible. In such situations we suggest that the reading with a narrow scope a must be the **preferred reading**.

Let us now look at the situation in 41b, which is a bit more interesting due to the additional linguistic context. Intuitively, while both, the direct and indirect reading should still be plausible, in this situation one would expect a wide scope a to have a higher degree of plausibility than was the case in 41a. Applying (39) on 41b results in the following:

$$\begin{aligned}
 r_1 &= \left(1 \leq \frac{Q_2}{Q_1} |X| \leq \frac{1^+}{1^+} |C_1| \right) \\
 &= \left(1 \leq \frac{a}{every} |StudentOfDrMiller| \leq \frac{[1,...,all]}{[1,...,all]} |Student| \right) \\
 &= \left(\frac{1}{all} \frac{1}{s} \leq \frac{m_2 = [1,...,all]}{m_1 = [1,...,all]} \right) = true, \forall m_1 m_2 \\
 r_2 &= \left(1 \leq \frac{Q_1}{Q_2} |Y| \leq \frac{1^+}{1^+} |C_2| \right) \\
 41b' \Rightarrow &= \left(1 \leq \frac{every}{a} |SeminarOnChaosTheory| \leq \frac{[1,...,all]}{[1,...,all]} |Seminar| \right) \\
 &= \left(\frac{all}{1} \frac{1}{m} \leq \frac{m_1 = [1,...,all]}{m_2 = [1,...,all]} \right) = true \text{ for } m \text{ combinations out of } [1,...,all] \times [1,...,all]
 \end{aligned}$$

where

$$|StudentOfDrMiller| = \frac{1}{s} |Student|, s \geq 1$$

$$|SeminarOnChaosTheory| = \frac{1}{m} |Seminar|, m \geq 1$$

As in 41a, the direct reading of 41b is quite plausible. Unlike the situation in 41a', however, in 41b' there seems to be many more plausible situations that the indirect reading is consistent with, due to the additional linguistic context. The scope function seems to predict that the likelihood of a single seminar increases as more modifiers are used in the noun phrase referring to *seminar*.

In order to appreciate the various dependencies that are captured by the inequalities in (39), however, it is very important to note that the contribution of the linguistic context does not always yield the same result. The reason for this is that the inequalities in (39) are not only a function of the linguistic context, but of *the linguistic context in conjunction with the conditions imposed by the quantificational constraint*. To illustrate this point even further, consider the following example:

(42) *Every student of Dr. Miller gave a seminar on Chaos Theory*

Let us assume $QC(Give, Student, Seminar) = \langle 1, 1^+ \rangle$, which is a reasonable quantificational constraint in this context. That is, we assume that only one student gives the same seminar, but that the same student can give many (up to *all*) seminars. Applying the rules of (39) on (42) yields the following (where *s* and *m* are the same as those in 41b'):

$$\begin{aligned}
 r_1 &= \left(1 \leq \frac{a}{every} |StudentOfDrMiller| \leq \frac{1^+}{1} |Student| \right) \\
 &= \left(\frac{1}{all} \frac{1}{s} \leq \frac{m_2 = [1, \dots, all]}{m_1 = [1]} \right) = true, \forall m_1 m_2 \\
 42' \Rightarrow \\
 r_2 &= \left(1 \leq \frac{every}{a} |SeminarOnChaosTheory| \leq \frac{1}{1^+} |Seminar| \right) \\
 &= \left(\frac{all}{1} \frac{1}{m} \leq \frac{m_1 = [1]}{m_2 = [1, \dots, all]} \right) = false \forall m_1 m_2
 \end{aligned}$$

Apparently, the situation in (42) is quite different from the one in (41), as no amount of modification to the noun phrase referring to *seminar* seems to make the indirect reading of (42) plausible. That is, regardless of how small the set of seminars gets to be, the indirect reading would still be implausible, as r_2 in 42' could never be made *true*. We leave it to the reader to verify that the modifier '*of Dr. Miller*' in '*A student of Dr. Miller gave few seminars on Chaos Theory*' **does** contribute positively to the plausibility of a direct reading.

3.2 A NUMERICAL MEASURE FOR THE PLAUSIBILITY OF A READING

The point of the above examples was to show how the inequalities in (39) should be interpreted. From the examples discussed thus far it seems that a suitable measure of plausibility for a reading is the number of ways the inequality

$$r_1 = 1 \leq \left(\frac{Q_2}{Q_1} \frac{1}{a} \right) \leq \left(\frac{v_2}{v_1} = \frac{[1, \dots, m_2]}{[1, \dots, m_1]} \right)$$

holds, and similarly for the indirect reading. However, this is the same as saying that a good measure of plausibility is the number of ways the following inequality holds:

$$\left(\frac{Q_2}{Q_1} \frac{1}{a} \right) \leq (v_1 \div v_2) \text{ where } (v_1 \div v_2) = \left\{ v = \frac{x}{y} \mid (x, y) \in (v_1 \otimes v_2) \right\}$$

where $(v_1 \div v_2)$ is the set of x/y for all (x, y) in the cross-product $v_1 \otimes v_2$. The quantifiers on the left-hand side (LHS) of the inequality might also define a range, which is a single value in the case of simple (numeric or linguistic) quantifiers. In general, however it could refer to a range $[n1, \dots, n2]$ (e.g., $[2, \dots, all]$ for *at least 2*), and thus, the degree of plausibility is the number of ways in which the LHS range can be less than the RHS range. In measuring the degree to which one interval is less than some other interval, we are interested in the following situations:

In (a), every value in the range I is less than any possible value in J . In this case, the inequalities in (39) hold for all possible combinations of m_1 and m_2 , and the measure of plausibility is 1. In (b) we have the reverse situation, since no value in the range I is less than any value in J . In this case the measure of plausibility is 0. In (c) through (f), the measure of

plausibility is some value between 0 and 1. The following formula can be used to compute a measure of plausibility $p(I,J)$ given two intervals I and J ¹⁹:

$$p(I,J) = \left(Pr(I,J)|J| + \sum_{i=1}^{|I \cap J|} (|J| - Pr(J,I) - i + 1) \right) \left(\frac{1}{|I||J|} \right)$$

where

$$Pr(I,J) = |\{i \in I \mid i < J_{\min}\}|$$

Here $Pr(I,J)$ is the number of elements in the interval I that are strictly less than (i.e., they *precede*) all the elements in J . Note that for the two special cases of (a) and (b) above we have the following:

$$(a) \Rightarrow p(I,J) = \left(|I||J| + \sum_{i=1}^0 (...) \right) \left(\frac{1}{|I||J|} \right) = \frac{|I||J|}{|I||J|} = 1$$

$$(b) \Rightarrow p(I,J) = \left(0|J| + \sum_{i=1}^0 (...) \right) \left(\frac{1}{|I||J|} \right) = \frac{0}{|I||J|} = 0$$

Prior to applying the function that computes a measure of plausibility it is assumed that the appropriate ranges (intervals) have been constructed. In figure 6 below we detail this process in a scoping algorithm that takes a sentence as input and returns a measure of plausibility for the direct and indirect readings. There are several important points that should be made regarding this algorithm. In particular:

1. The function $\text{GetConcept}(N)$ was briefly discussed above, and there will be more to say about these knowledge representation issues in chapters 5 and 6.
2. The details of how a and b are computed in step 1d are discussed in chapter 6.
3. In step 2 we assign numeric values to the linguistic quantifiers based on the size of the sets they quantify, as discussed in (16).
4. In steps 3 and 4 we used two functions that compute the Cartesian division of two ranges and the degree to which a range is contained in another range. The functions are rather simple to implement and are discussed in chapter 6.

¹⁹ In van Eynde (1990) time intervals are similarly used to model tense and temporal aspects. In a slightly different context (Shubert *et al.*, 1983; see also Geller, 1994), a numerical method was also proposed for very efficient hierarchical reasoning in a semantic network by associating a pair of values (a,b) with every node N , where a is the pre-order traversal number of N and b is the maximum pre-order number below N . With such an arrangement, a node is a subclass of another node if it has a smaller interval. In such a network, intervals do not overlap, and thus a node is either a subclass of another node or it is not.

5. Finally, the algorithm assumes a sentence with a binary relation between two quantified noun-phrases. We will later show how this can be easily generalized.

3.3 APPLYING THE SCOPING ALGORITHM

In the next few sections we examine the scoping algorithm by considering examples that highlight various aspects of the quantifier scope phenomena.

3.3.1 A Possible Explanation for the Transformational Puzzle

Besides predicting the plausible scope orderings, the scoping rules in (39) seem to provide an explanation for the so-called *transformational puzzle* which was discussed earlier. In particular, the scoping rules can determine when a transformation is *meaning-preserving*, i.e., when the passive and active forms imply the same meaning.

To illustrate this, consider the following, where we make the same assumptions regarding the quantificational constraint as those made in (41):

$$\begin{aligned}
 &\text{Every student gave a seminar} \\
 (43) \quad r_1 &= \left(\frac{Q_2}{Q_1} |X| \leq \frac{m_2}{m_1} |C_1| \right) = \left(\frac{1}{all} \leq \frac{m_2 = [1, \dots, all]}{m_1 = [1]} \right) = true, \forall m_1 m_2 \\
 r_2 &= \left(\frac{Q_1}{Q_2} |Y| \leq \frac{m_2}{m_1} |C_2| \right) = \left(\frac{all}{1} \leq \frac{m_1 = [1]}{m_2 = [1, \dots, all]} \right) = false, \forall m_1 m_2
 \end{aligned}$$

Regardless of whether the sentence is in active or passive form, the only plausible reading in (43) seems to be the one with narrow scope *a*. We say **the transformation is meaning-preserving** since regardless which form is uttered, one reading (and thus one meaning) will always be selected.

This can also occur in situations where the indirect reading seems to be the only plausible reading:

$$\begin{aligned}
 &2 \text{ men lifted a piano to the second floor} \\
 &QC(Person, Lift, Piano) = \langle 2^+, 1 \rangle \\
 (44) \quad r_1 &= \left(\frac{1}{2} \leq \frac{m_2 = [1]}{m_1 = [2, \dots, all]} \right) = false, \forall m_1 m_2 \text{ except for } m_1 = 2 \\
 r_2 &= \left(\frac{2}{1} \leq \frac{m_1 = [2, \dots, all]}{m_2 = [1]} \right) = true, \forall m_1 m_2
 \end{aligned}$$

- Let p_1 = 'degree of plausibility of direct reading'
 p_2 = 'degree of plausibility of indirect reading'
1. Given a $S = [{}_S [{}_{NP} [{}_{Det} Q_1] X] [{}_{VP} V [{}_{NP} [{}_{Det} Q_2] Y]]]$
 - 1a. $QC(R, C_1, C_2) = \langle m_1, m_2 \rangle$ is the relevant quantificational constraint iff

$$\begin{aligned} \text{GetRelation}(V) &= R; \\ \text{GetConcept}(X) &= C_1; \\ \text{GetConcept}(Y) &= C_2 \end{aligned}$$
 - 1b. $a = \frac{|C_1|}{|X|}; b = \frac{|C_2|}{|Y|}$
 - 1c. $v_1 = [1, \dots, m_1]; v_2 = [1, \dots, m_2]; q_i = \text{ParseAndCreateRange}(Q_i)$
 2. Assign values to q_1, q_2, v_1, v_2 according to the size of X, Y, C_1 and C_2 respectively. (see (16))
 3. $R_1^D = (q_2 \div a q_1); R_1^I = (q_1 \div b q_2);$
 $R_2^D = (v_2 \div v_1); R_2^I = (v_1 \div v_2)$
 where
 $(v_1 \div v_2) = \left\{ v = \frac{a}{b} \mid (a \in v_1) \wedge (b \in v_2) \right\}$
 $a(Range) = \{ (a \bullet v) \mid v \in Range \}$
 4. $p_1 = 0 \leq p(R_1^D, R_2^D) \leq 1$
 $p_2 = 0 \leq p(R_1^I, R_2^I) \leq 1$
 where

$$p(R_1, R_2) = \frac{\sum_{i=1}^{|R_1|} \left| \{ v \in R_2 \mid R_1[i] \leq v \} \right|}{|R_1| |R_2|}$$

Figure 6. Measuring the plausibility of the direct and indirect scope orderings.

As in (43), one reading will always be selected, although unlike (43), it is the indirect reading of (44), 'a piano was lifted by two men to the second floor', that will be selected. What happens when the two readings are plausible? This could happen in two ways: the two readings could be plausible under the exact same conditions, or they may both be plausible, but not equally so. In the former case, it is immaterial which reading is selected since both readings would have the same meaning²⁰. The important question that must be answered

²⁰ In sentences where the passive and active forms have the same meaning, such as 'a crime was committed by a teenager' or 'John committed a crime', scope ambiguities do not arise and the choice of which form the sentence might take is perhaps simply a matter of bringing into focus a certain topic.

now is the following: what should happen when the direct and indirect readings are equally plausible, although they imply different meanings? Before we suggest an answer to this question, let us consider this situation with an example. Consider the following:

- (45) *a. Three MIT students submitted a paper on Data Mining to ACL'95*
b. A paper on Data Mining was submitted to ACL'95 by three MIT students

Assuming that on average one or two co-author the same paper, applying the rules in (39) on (45a and b) results in the following:

$$\begin{aligned}
 45a' \Rightarrow r_1 &= \left(1 \leq \frac{a}{3} |MITStudent| \leq \frac{1}{2^-} |Student| \right) \\
 &= \left(\frac{1}{3} \frac{1}{s} \leq \frac{m_2 = [1]}{m_1 = [1,2]} \right) = true, \forall m_1, m_2, \text{ where } |MITStudent| = \frac{1}{s} |Student| \\
 45b' \Rightarrow r_1 &= \left(1 \leq \frac{3}{a} |DataMiningPaper| \leq \frac{2^-}{1} |Paper| \right) \\
 &= \left(\frac{3}{1} \frac{1}{p} \leq \frac{m_1 = [1,2]}{m_2 = [1]} \right) = true, \forall m_1, m_2, \text{ where } |DataMiningPaper| = \frac{1}{p} |Paper|
 \end{aligned}$$

In (45) both, the active and the passive forms are equally plausible, although the meaning implied by each is quite different. In these situations we argue that the simple heuristic suggesting that the preferred meaning should perhaps be **the meaning implied by the direct reading** seems reasonable²¹.

This heuristic is very similar in spirit to the *topic principle* of (Katz, 1980) and the *thematic principle* of (Grimshaw, 1990), where the topic or the main agent of the sentence is considered to be the main focus and thus would preferably have wide scope.

It should be noted, however, that these two principles tell the same story only in this particular situation, namely when the two readings are equally plausible **and** they imply different meanings. The reader can easily verify that thematic and topic principles do not apply to other situations, as all the previous examples suggest. The various possibilities discussed above can be graphically summarized as shown below:

²¹ We are not brave enough to discuss *intention* here. We are merely suggesting that when two readings that are equally plausible convey different meanings, it is perhaps preferable to stick with the intention of the author, whatever that might be.

Note that the diagonal represents the cases where both readings are *equally* plausible, and thus, according to our heuristic, the points along the diagonal as well as those in the area labeled A represent the cases where the direct reading is considered the preferred reading. The points in area B (excluding those points on the diagonal) represent the cases where the indirect reading is considered the preferred reading.

3.3.2 Generalized Quantifiers

The numerical formulation of the scoping rules in (39) allows us to easily model any generalized quantifier (Barwise and Cooper, 1981), such as '*many but less than six*', '*at most three*', etc. As the following example suggests, a very subtle change in scope preferences is sometimes due to a slight variation in the choice of the quantifier:

- (46) *Every student at MIT speaks 2 foreign languages*
 (47) *Every student at MIT speaks at least 2 foreign languages*

According to the scoping rules in (39), the direct reading in (46) should be the preferred reading. However, it seems that the addition of '*at least*' to the quantifier 2 in (47) changes the situation slightly, as can be seen in the following:

$$\begin{aligned}
 r_1 &= \left(\frac{2}{all} \frac{1}{s} \leq \frac{m_2 = [1, \dots, all]}{m_1 = [1, \dots, all]} \right) = true \forall m_1, m_2 \\
 46' \Rightarrow r_2 &= \left(\frac{all}{2} \frac{1}{l} \leq \frac{m_1 = [1, \dots, all]}{m_2 = [1, \dots, all]} \right) = true \text{ for } l(m_1, m_2) \\
 r_1 &= \left(\frac{[2, \dots, all]}{all} \frac{1}{s} \leq \frac{m_2 = [1, \dots, all]}{m_1 = [1, \dots, all]} \right) = true \text{ for half of } (m_1, m_2) \\
 47' \Rightarrow r_2 &= \left(\frac{all}{[2, \dots, all]} \frac{1}{l} \leq \frac{m_1 = [1, \dots, all]}{m_2 = [1, \dots, all]} \right) = true, \text{ for } 2l(m_1, m_2) \\
 \text{where } |StudentAtMIT| &= \frac{1}{5} |Student|; \quad |ForeignLanguage| = \frac{1}{l} |Language|
 \end{aligned}$$

In 46', i.e., without the addition of *at least* to the quantifier 2, the indirect reading does not seem to be plausible. By qualifying the quantifier 2 with *at least*, however, it seems that the indirect reading implying

(48) *There are at least 2 foreign languages that every student at MIT speaks*

is quite plausible, as shown in 47'. Intuitively, therefore, 47' predicts that qualifying 2 with *at least* suggests that we might have (at least!) 2 **specific** languages in mind.

3.4 COMPLEX FUNCTIONAL DEPENDENCIES

Thus far we have only considered sentences involving a binary relation between two quantified noun phrases²². However, sentences typically involve more than a pair of quantifiers, and more complex **functional dependencies** must be considered. In this section we will start with simple extensions to the model and work our way into more complex situations involving **branching-quantifiers**. Before we proceed, however, we need to restate Frege's initial formulation of quantifiers, namely that quantifiers are linear in that an expression such as $(Q_1a(Q_2b))(Q_3c)R(a,c)$, where $Q_i \in \{\forall, \exists\}$, is interpreted as true in a model A with universe D , iff²³

$$(49) \quad \left[\begin{array}{l} q1 \ a' \text{'s in } D \text{ are such that for each one of them} \\ \quad \left[\begin{array}{l} q2 \ b' \text{'s in } D \text{ are such that for each one of them} \\ \quad \left[q3 \ c' \text{'s in } D \text{ are such for each one of them } R(a,b,c) \right] \end{array} \right] \end{array} \right]$$

Thus, in general, for any two linearly-ordered pairs of quantifiers $(Q_1a)(Q_2b)(\dots e\dots)$, the expression e refers to Q_1a and to $(Q_1Q_2)b$. It is this dependency that must be captured to appropriately explain the scope preferences in situations where complex functional dependencies are involved.

3.4.1 Simple Functional Dependencies

Before we discuss branching quantifiers we first illustrate the relationship between functional dependencies and quantifier scope with simple examples. Consider the following:

²² We must note here that it suffices to formulate our algorithm on binary relations only (corresponding to transitive verbs at the syntactic level), since n -ary relation can be reduced to a set of binary relations.

²³ See Sher (1990).

- (50) $\left\{ \begin{array}{l} a. \text{ A scientist} \\ b. \text{ A scientist from every country} \end{array} \right\} \text{ was consulted by two senators}$

A reasonable quantificational constraint that is relevant in this context might be $QC(Consult, Scientist, Senator) = \langle 1^+, 1^+ \rangle$, stating that the same scientist could be consulted by *many* senators and that the same senator might consult *many* scientists²⁴. With this background, applying the rules in (39) on (50a) yields the following:

$$(51) \quad \begin{aligned} r_1 &= \left(\frac{2}{a} \leq \frac{m_2 = [1, \dots, all]}{m_1 = [1, \dots, all]} \right) = true, \forall m_1 m_2 (m_2 \geq 2m_1) \\ r_2 &= \left(\frac{a}{2} \leq \frac{m_1 = [1, \dots, all]}{m_2 = [1, \dots, all]} \right) = true, \forall m_1 m_2 (m_2 \leq 2m_1) \end{aligned}$$

According to (51), both readings of (50a) are quite plausible, and thus by the simple heuristic that we argued for above, the direct reading of (50a) should be preferred. In (50b), however, we have a relation between two quantified NPs, one of which (the head NP) involves scope ambiguities. Thus, at a very high-level, (50b) is processed as

$$\begin{aligned} &\text{scope}(A \text{ scientist from every country was consulted by two senators}) \\ &= \text{scope}_1(\text{scope}_2(A \text{ scientist from every country}) \text{ was consulted by two senators}) \end{aligned}$$

where *scope* is a second-order function that repeatedly applies the scoping rules of (39) on constituents involving a binary relation between two quantified noun phrases. Assuming that a scientist can be from one country only, where there could be many, we can apply *scope*₂ on 'a scientist from every country' as shown below:

$$(52) \Rightarrow \quad \begin{aligned} r_1 &= \left(\frac{all}{1} \leq \frac{m_2 = [1]}{m_1 = [1, \dots, all]} \right) = false, \forall m_1 m_2 \\ r_2 &= \left(\frac{1}{all} \leq \frac{m_1 = [1, \dots, all]}{m_2 = [1]} \right) = true, \forall m_1 m_2 \end{aligned}$$

²⁴ Clearly in applying the scoping rules we are treating linguistic quantifiers as numeric values. In fact, linguistically, there is no difference between '3 students' and 'several students'; both answer the question 'how many students'. However, we should keep in mind that unlike '3 scientists' and '3 senators', for example, 'many scientists' and 'many senators' do not refer to the same values. These differences can be very crucial as the following example demonstrates:

(1) *Most rebel leaders read a Che Guevara book on guerrilla warfare*

In (1), the modifiers on 'book' as well as the fact that *many people* is a much larger value than *many books* in $QC(Person, Read, Book) = \langle many, many \rangle$ combine to make the indirect reading the more plausible reading.

By reversing the scope ordering of '*a scientist from every country*', the phrase '*a scientist*' might potentially refer to *many* (as opposed to *a* single) scientist, one from every country. In processing of the entire sentence, therefore, we must have the following:

$$\begin{aligned} & \text{scope}(A \text{ scientist from every country was consulted by two senators}) \\ &= \text{scope}_1(\text{scope}_2(A \text{ scientist from every country}) \text{ was consulted by two senators}) \\ &= \text{scope}_1\left(\text{Every } \left(\dots \text{country } c, a \text{ scientist from } c \dots\right) \text{ was consulted by } \right. \\ & \quad \left. \text{Two } (\dots \text{senators} \dots) \right) \end{aligned}$$

That is, in applying the scope function at the (outermost) sentential level we are speaking of two senators that consulted *several* scientists. Moreover, unlike the situation of (50a), in (50b) the possibility of a single scientist does not exist. The final scope resolution of the entire sentence is then done as follows:

$$\begin{aligned} & \text{Every } (\dots \text{Scientist } \dots) \text{ was consulted by two senators} \\ & QC(\text{Consult}, \text{Senator}, \text{Scientist}) = \langle 1^+, 1^+ \rangle \\ 52' \Rightarrow & \quad r_1 = \left(\frac{2}{all} \leq \frac{m_2 = [1, \dots, all]}{m_1 = [1, \dots, all]} \right) = \text{true}, \forall m_1 m_2 \text{ except for } m_2 = 1 \\ & \quad r_2 = \left(\frac{all}{2} \leq \frac{m_1 = [1, \dots, all]}{m_2 = [1, \dots, all]} \right) = \text{false}, \forall m_1 m_2 \text{ except for } m_2 \in \{1, 2\} \end{aligned}$$

In (52') the direct reading of the passive is accepted, and the final reading of (50b) is '*from every country there is a scientist that some two senators consulted with*'. What is important to note here is that unlike (50a), the addition of the PP '*from every country*' makes the indirect reading of (50b) implausible. That is, while it is quite plausible for two senators to consult with a specific scientist, it is not likely that they would be able to consult with a (specific) scientist, who is a *scientist from every country*. Unlike '*a scientist*', therefore, '*a scientist from every country*' does not refer to a single scientist, but to several (one from every country). Since the scoping rules of (39) are sensitive to the relative size of the quantified sets, these subtle functional dependencies can be easily accounted for in our method. As we will show in the next section, this process can be generalized even further to account for more complex functional dependencies.

3.4.2 Branching Quantifiers

A more complex type of functional dependency points to a phenomenon that has occupied logicians for a number of years, namely the problem of "branching quantifiers." Hintikka (1974) was the first to point out that quantifiers are not always linear as has been initially

suggested by Frege. Instead, it was suggested that in (53), for example, there are two pairs of quantifiers that are *independent in regard to scope*.

(53) *An author of every novel was mentioned in an article by every critic*

While it is true that some quantifiers are branching, extensive studies by (Fauconnier, 1975; Barwise, 1979; May, 1989; Sher, 1990), among others, have not settled the issue of “under what conditions do quantifiers branch?” In fact, (Gurevich, 1988) has shown that the problem is NP-hard, which suggests that a computationally tractable solution must rely on some “heuristics.” Let us see what might be happening in (53). The relevant relations are depicted graphically below.

Note that the intermediate relation, *MentionedIn*, forms a *many-to-many* nucleus, which, as we have seen above, allows any possible scope ordering, while the relations on the two end points seem to be more restricted. Another way of stating this is that it seems that the intermediate relation that glues together the entire sentence suggests that once the scope ordering of both pairs has been decided, the two pairs are independent. The scoping of the entire sentence must proceed as follows:

$$\begin{aligned}
 & \text{scope}(\text{An author of every novel was mentioned in an article by every critic}) \\
 &= \text{scope}_0 \left(\begin{array}{l} \text{scope}_1 (\text{an author of every novel}) \text{ was mentioned in} \\ \text{scope}_2 (\text{an article by every critic}) \end{array} \right) \\
 &= \text{scope}_0 \left(\begin{array}{l} \text{Every}(\dots \text{author} \dots) \text{ was mentioned in} \\ \text{Every}(\dots \text{article} \dots) \end{array} \right)
 \end{aligned}$$

Applying the scoping rules in (39) in **scope1** and **scope2** results in reversing the scope ordering in both cases (since, with respect to our hypothesized QCs, we cannot speak of one author, who is the author of every novel, and of one article which is written by every critic). Once the constituent's scope have been determined, the scope of the entire sentence can be determined in two equivalent ways, since as we have seen above, this is the situation where

the two readings are equally plausible. The reason the two pairs of quantifiers are branching in this case, therefore, is a function of the intermediate relation and the quantifiers that resulted in noun phrases of each constituent (which in turn is a function of the scoping at the constituent level). That is, in **scope0** we use

$$QC(MentionedIn, Author, Article) = \langle many, many \rangle \\ \text{and } (Q_1 = Every) \text{ and } (Q_2 = Every)$$

Note that if we replace '*every novel*' by '*a novel*' the pairs of quantifiers would not be branching and one and only one ordering would be accepted at the top-most level. Finally, we must make two important remarks regarding the issue of complex functional dependencies and branching quantifiers. First, sentences involving multiple quantified term phrases can have quite a complex syntax and a number of important issues are typically addressed at the syntactic level prior to the resolution of quantifier scope ambiguities. In this section we have glossed over these issues and, as we have done throughout this dissertation, we have assumed that the proper syntactic and semantic analysis has been done, resulting in a logical form which we take as the input to our process. That is, the purpose of the above discussion was to demonstrate that the inferencing strategy we have suggested could be scaled-up to handle complex functional dependencies. A detailed example of how a non-trivial sentence can be processed using the algorithm described here is shown in appendix B.

In conclusion, while the higher-order **scope** function discussed above is admittedly too simplistic to address the many complex syntactic forms that could involve branching quantifiers, it nevertheless demonstrates that the problem of branching quantifiers is simply a composite of pair-wise quantifier scope ambiguities, each of which can be resolved using the inferencing strategy that we have developed here.

3.5 EXTENDING THE SCOPING ALGORITHM TO ACCOUNT FOR THE TEMPORAL AND MODAL ASPECTS

Throughout this chapter we have assumed quantificational constraints that implicitly reflect the conditions under which a relation can **possibly** hold between two concepts **at a specific point in time**. While this is adequate to explain quantifier scope preferences in most cases, there are situations that we briefly discussed in chapter 2 where the scoping algorithm must take the modal and temporal aspects into consideration. In this section we discuss a minor extension to the inferencing strategy to address this issue.

3.5.1 The Relation Between Modality and the Temporal Aspect

Consider again the following quantificational constraints involving the (physically located) *On* relation, defined on two different pairs of concepts:

$$(54) \quad QC(On, House, Street) = \langle many, 1 \rangle$$

$$(55) \quad QC(On, Book, Shelf) = \langle many, 1 \rangle$$

Both of these QCs state that neither a house, nor a book can be in more than one physical location at the same time. However, as we briefly mentioned in chapter 2, there is an important difference between these two relations that is not captured by (54) and (55). While it might be *necessary* for a house to be located on a single street, a book need not be on a shelf. In other words, while it is *possible*, or may even be *typical*, it is not necessary for a book to be on a shelf. Another difference is that a book, typically, and unlike a house, can over time change its physical location. In other words, the QC in (55) might be *many-to-many* if we were speaking of *books on shelves* at different points in time. Table 2 below illustrates the possible QCs for the *On* relation between books/shelves and houses/streets, along with the temporal and modal dimensions.

# of shelves that a book	is typically on	can possibly be on	must be on
at some specific point in time	?	1	0
at various points in time	?	many	0
# of streets that a house	Is typically on	can possibly be on	must be on
at some specific point in time	1	1	1
at various points in time	1	1	1

Table 2. The interaction between QCs and the modal and temporal aspects.

Note that while a book can only be on *one* shelf at a *specific* point in time, at various points in time a book *can be* on many shelves. What is interesting to note here is that in (54), it seems that *m2*, which must necessarily be 1, is 1 at all times. This should not be surprising since from modal logic (Gamut, 1991) we know that

$$\begin{aligned} \Box(\phi) &\equiv \forall t \forall w(\phi), \text{ i.e., } \phi \text{ is necessarily true if it is true in **every** possible world} \\ \Diamond(\phi) &\equiv \exists t \exists w(\phi), \text{ i.e., } \phi \text{ can possibly be true if it is true in **some** possible world} \\ &\text{where } \langle t, w \rangle \text{ is a possible world} \end{aligned}$$

That is, **necessity can be modeled by universal quantification over time**, while *possibility* is modeled by existential quantification over time. Stated in other words, a necessary relation

is true *iff* it is true in every possible world (a place and time index.) Given a quantificational constraint $\langle m_1, m_2 \rangle$, therefore, neither m_1 nor m_2 can change their values over time if either one is known to be necessary. For example, in

$$QC(On, House, Street) = \langle many, 1 \rangle$$

$$QC(Woman, MotherOf, Child) = \langle 1, many \rangle$$

both QCs are the most general, i.e., they cannot move up to *many-to-many* if we consider the relations at various points in time: since it is necessary for a house to be on some street, and it is also necessary for a child to have a (biological) mother; a house is *always* on a single street, and a child will *always* have the same (one) mother. These restrictions do not apply to other quantificational constraints, however, as the examples in figure 7 below illustrate²⁵.

Figure 7. Movements of quantificational constraints on the temporal dimension.

²⁵ Again we repeat that QCs are reader-based, and so are the suggested movements in figure 7.

What the above seems to suggest, therefore, is that we should start with a default QC that reflects what is *possible at a specific point in time* and proceed as follows:

- $QC(R, C_1, C_2) = \langle m_1, m_2 \rangle$
- $Mod(m_1) = mod_1; Mod(m_2) = mod_2$
- if $((mod_1 \neq Nec') \wedge (Tense(R) = 'past')) m_1 = (m_1)^+$
- if $((mod_2 \neq Nec') \wedge (Tense(R) = 'past')) m_2 = (m_2)^+$
- proceed with the scoping algorithm as before ...

To illustrate the need for taking the effect of the modal and temporal aspects into consideration, consider the following examples:

(56) *An American battalion led the attack on several fronts*

(57) *An American battalion is leading the attack on several fronts*

There are a couple of interesting points that should be made here. First, in both (56) and (57) the scope ordering in '*the attack on several fronts*' must be reversed, since the same attack can not happen on several fronts. Instead, 'the attack' here refers to several attacks, one on each of the several fronts. The scoping in (56) and (57) must therefore proceed as follows:

$$\begin{aligned}
 & \mathbf{scope}(An\ American\ battalion [led/ tense] the\ attack\ on\ several\ fronts) \\
 &= \mathbf{scope}_0 \left(\begin{array}{l} \mathbf{scope}_1 (An\ American\ battalion) [led/ tense] \\ \mathbf{scope}_2 (the\ attack\ on\ several\ fronts) \end{array} \right) \\
 &= \mathbf{scope}_0 \left(\begin{array}{l} A (... Battalion ...) [led/ tense] \\ Several (... Attack ...) \end{array} \right)
 \end{aligned}$$

In order to apply the final scoping function we require a quantificational constraint that is defined on the *Lead* relation between *battalion* and *attack*. The default QC is the one that reflects how the conditions under which the relation can happen at a specific point in time. The tense differences in (56) and (57) are then used to decide if the QC should be updated, as follows:

$$\begin{aligned}
 & \text{Default } QC(Lead, Battalion, Attack) = \langle few, l \rangle \\
 56' \Rightarrow & \text{ Since } ((Mod(m_2) \neq Nec') \wedge (Tense(R) = 'past')) \text{ then } m_2 = (m_2)^+ \\
 & \therefore QC \text{ used is } QC(Lead, Battalion, Attack) = \langle few, l^+ \rangle \\
 57' \Rightarrow & \text{ Since } (Tense(R) = 'present') \text{ default QC is used}
 \end{aligned}$$

That is, it is quite possible for the same battalion to lead the attack on several fronts, if done at various points in time, but the same battalion can not, at the same point in time, lead the

attack on several fronts. The different QCs indeed result in different scope preferences, where the direct reading of (57) is not considered plausible:

$$\begin{aligned}
 56' \Rightarrow r_1 &= \left(\frac{\text{several}}{1} \frac{1}{b} \leq \frac{m_2 = [1, \dots, \text{all}]}{m_1 = [1, \dots, \text{few}]} \right) = \text{true, for most } (m_1, m_2) \\
 57' \Rightarrow r_1 &= \left(\frac{\text{several}}{1} \frac{1}{b} \leq \frac{m_2 = [1]}{m_1 = [1, \dots, \text{few}]} \right) = \text{false, } \forall m_1 m_2
 \end{aligned}$$

3.6 A BRIEF SUMMARY

Kurtzman and MacDonald (1993) stated that "for psycholinguists, the interest in quantifier scope ambiguities lies in the operations that comprehenders employ to resolve the ambiguities." In this section we have made a suggestion as to what these operations might be, and what parameters they might depend on.

However, it should be evident by now that since our method is based on the assumption that readers associate a commonsense constraint on a relation between two concepts, the scoping function cannot be validated analytically. Moreover, since all possible combinations of quantifiers, quantificational constraints, and variations of the linguistic context are infinite, all we can do here is consider some examples that highlight the main features of the method we have proposed.

In this chapter we tried to provide evidence suggesting that an inferencing strategy for predicting quantifier scope could be attained. What we have proposed seems to suggest a method by which one could hope to:

1. Explain how different individuals might have different scope preferences;
2. Predict the set of plausible readings and the preferred reading among them;
3. Account for variations in the linguistic context and generalized quantifiers;
4. Provide an explanation for the transformational puzzle;
5. Account for complex functional dependencies, including branching quantifiers;

The need for a reader-based approach in the overall task of text comprehension has been argued for by a number of authors, most notably by Corriveau (1995). In this dissertation we have a much narrower view of this phenomenon. In particular, our aim was to simply

suggest a cognitively plausible model that would explain how different individuals seem to have different scope preferences, given the same linguistic context, as has also been demonstrated by the experiments of Kurtzman and MacDonald (1993).

To model these individual preferences, we have suggested an algorithm that depends on a set of individual beliefs as to how relations between concepts are manifested, with respect to their cardinality constraints. This individual parameter, which we termed *quantificational constraint*, also captures the different modalities (*necessity*, *typicality*, and *possibility*) that different individuals might associate with a relation. Clearly, such an approach cannot be validated *analytically*, but has to be validated experimentally, since our approach does not aim at predicting what is syntactically or logically *valid*, but what is **plausible**, given a set of individual beliefs.

Regardless of how accurately the process we have suggested captures the process we might be employing in resolving scope ambiguities, the process seems to explain how different individuals might arrive at different scope preferences. Moreover, the fact that the scoping algorithm assigns a numerical measure for the plausibility of a reading seems to provide an explanation for the so-called ‘combinatorial puzzle’ where hundreds of thousands of possible readings might be available. In particular, our approach suggests that readers might not necessarily select a single reading, but could arrive at an interpretation which is a set of meanings which contains all the plausible meanings that have a measure of plausibility beyond a certain threshold.

We must note, however, that although we have only touched the surface regarding a number complex phenomena, such as branching quantifiers, the method we proposed seems to provide an adequate avenue for modeling the linguistic context and capturing complex semantic and functional dependencies.

Finally, we must note that in presenting our inferencing strategy we have glossed over some representational and computational issues that could have some impact on the implementation of our proposed method. In the next chapter we discuss these issues in some detail. The details of the implementation are deferred until chapter 5.

4 Representational and Computational Issues

In the last chapter we presented an inferencing strategy for predicting the most plausible quantifier scope ordering, using a contextually computed value which we termed *quantificational constraint*. Along the way we have made a number of assumptions and glossed over a number of representational and computational issues, which we will discuss in some detail in this chapter. The details of our implementation are discussed in the next chapter.

4.1 TYPICAL CARDINALITIES

In describing the scoping algorithm of figure 6 we have accounted for the linguistic context by assuming the algorithm has access to the typical size of a concept's extension. What we like to argue here is that our notion of a *typical cardinality of a set* is not problematic in the context of a database system.

First, in a database environment, computing the *actual* number of elements of a certain entity type is trivial. However, it is not the *actual* size of the concept at a specific point in time that we are interested in, but the *typical size* at all times, since this value is dynamically changing, and any single value obtained at a specific point in time might be misleading. How could we arrive at such a value?

A good measure of this value might be a running average that is updated periodically, where the period chosen is a function of the semantics of the database and the (extension of the) entity in question. For example, the typical cardinality of the concept *Student* in some university database could be the average size of this set that is maintained for the lifetime of the database, and updated periodically. A reasonable period in this case might be a semester, since in that period the cardinality of the set remains more or less constant. Depending on their semantics, different concepts (entities) will have different periods. For example, a reasonable period to update the running average of the cardinality of *Professor* in a university database might be an academic year. Finally, in a certain context (environment), the database could perhaps be 'bootstrapped' with these typical cardinalities, as part of the initial knowledge engineering effort.

Once the typical size of a concept is identified, linguistic quantifiers such as *all*, *most*, *many*, *few*, etc. can be assigned values as described earlier in (16). Note that using this model a

noun phrase such as '*many students*' would always have the intuitively correct interpretation that is dependent on the context. That is, '*many students*' should reflect a different number of students depending on the context (database); i.e., on whether the student's database is that of Carleton University, Shanghai University, or the Ministry of Education in Germany.

In conclusion, therefore, we can state that, with minimal knowledge engineering effort, procedures to access and periodically update the typical cardinalities of concepts can be easily attained in a database environment.

4.2 QUANTIFICATIONAL CONSTRAINTS

Clearly, a major ontological claim that we make in this thesis is that readers of natural language have access to a quantificational constraint that defines the conditions under which a relation between two concepts can occur. Admittedly, from a psycholinguistic standpoint, such a claim might be questionable. In particular, a valid question might be 'where do these quantificational constraints come from?' One possible answer that we might suggest to this question is that quantificational constraints are *learned* (by being told, by experience, by observation, etc.), as part of learning the concepts and the relations themselves. Our reasoning here is that it is not obvious to us how else could a child 'comprehend' the *BuiltOn* relation between houses and streets, without learning the conditions under which this relation can be manifested. Therefore, while we have assumed that QCs statically exist in the database, we are implicitly assuming that (as with most symbolic approaches) the issue of acquiring/learning these QCs is a knowledge acquisition issue which is beyond the scope of this dissertation.

Thus, while we are not making any claims as to how quantificational constraints might be acquired, we have committed ourselves to the postulate that such quantificational constraints *do* exist in the knowledge base. We argue below that this is quite a reasonable assumption to make, particularly in the context of a database system. Secondly, while this is not our concern here, accepting such a claim does raise an important question regarding the demarcation line between *word meaning* and *world knowledge*. For example, an important question that should now be answered is which concept (representing the meaning of some lexical item) 'owns' this (interactive) piece of knowledge reflecting that many houses can be on a single street, but that the same house is not *typically* located on more than one street? We also briefly discuss this issue below.

Regarding the above postulate, we argue that regardless of how QCs are acquired (or learned), assuming the existence of these static constraints in a computational environment is quite reasonable. In particular, in a natural language interface to a database system, for example, assuming the existence of such quantificational constraints is not problematic. Indeed, relational (Dullea and Song, 1997) as well as more recent object-oriented data models (Rumbaugh *et al.*, 1991) require such cardinality constraints to be defined to enforce referential integrity. Moreover, in relational data models necessary relations have to be explicitly stated, and are typically modeled by specifying that a certain value cannot be NULL²⁶. For example, the following simple data definition in SQL (structured query language) can define the quantificational constraint that we have assumed on the *LocatedOn* relation between houses and streets:

```
CREATE TABLE LocatedOn
( House_ID          INTEGER,
  Street_ID         INTEGER NOT NULL,
  PRIMARY KEY       (House_ID),
  FOREIGN KEY       (StreetID) REFERENCES Streets
)
```

The fact that *m2* in the *many-to-1* quantificational constraint is necessary is captured by the NOT NULL clause, which says that this relation cannot have a house that is not located on some street. The SQL statement also implicitly enforces a *many-to-1* relationship since *House_ID* is the primary key, which in turn means that it must be unique in that relation, i.e., there can be no more than one *LocatedOn* entry for the same house. Another advantage for the implementation of our method in a circumscribed environment, such as that of a database system, is that databases typically reflect a possible world (a *context*). That is, a data model typically reflects a specific view of the world, where business rules are enforced for some organizational reason. Other data models might define a completely different set of cardinality constraints reflecting a different set of business rules.

The second issue to consider pertains to where quantificational constraints are to be defined (i.e., on lexical items, concepts, or categories). This is very important for it is related a) to the question of the number of QCs that need to be defined in a knowledge base, b) to how exceptions are to be handled, and c) to the maintenance of these QCs. As should be evident

²⁶ In databases, NULL is a value that can roughly be taken to mean 'unknown', although this should not be confused with the 'unknown' truth-value in multi-valued logics.

by now, thus far QCs have been assumed to be defined on relations between *concepts*. Moreover, and although we have used a first-order logic representation of logical forms, our approach is not based on that representation nor have we assumed any knowledge representation or specific inheritance hierarchies.

This initial simple approach could be somewhat problematic however, since it would require r QCs to be defined, where r is the number of relations in the database (or the number of *links* in an entity-relationship model or in a semantic network). This number can potentially be very large, and consequently, the retrieval and maintenance of QCs can become quite a challenge. In our view, a more reasonable representational scheme can be devised. We discuss these enhancements in the following sections.

4.3 TYPICAL ELEMENTS

A quantificational constraint is a pair of values that are defined on a triple: a binary relation and two concepts. When defining a QC, therefore, we are implicitly stating the conditions under which a certain relation can occur between the **typical** elements of two sets. For example, in stating that it takes at least two people to lift a single piano, i.e., $QC(Lift, Person, Piano) = \langle 2^+, 1 \rangle$ we are clearly assuming that we are speaking of a typical piano (i.e., of a piano that is not extraordinarily large), and of people that are alive, reasonably healthy, etc., in short, we are speaking of **typical elements**²⁷.

Note, however, that what is 'typical' is a function of some features that are relevant to the quantificational constraint. For example, in the context of lifting pianos, a typical piano is a piano that has a typical *size/weight*, regardless of what *color* it is. In the same context, a typical person is a healthy person with a certain *age*. In the context of speaking languages, for example, a typical person is one that speaks one or two languages (as opposed to 100 languages), and the *age* feature in this context is really irrelevant! As will be discussed below, a quantificational constraint will be implemented as a function of a subset of features of the two concepts²⁸.

²⁷ Hobbs (1996) formally describes a slightly different notion of typical elements of a set, where each set is assumed to have a typical element with properties that are shared by all the elements in the set. Our notion of typical elements differs in one important way in that there is no single typical element that is always associated with a set. In particular, what is typical in one context is not necessarily typical in another.

²⁸ The terms 'concept' and 'feature' are used here in the same manner they are used in symbolic artificial intelligence and knowledge representation research. See, for example, the collection of papers in Sowa (1991).

When dealing with typical elements, however, an important question arises, namely, what to do about exceptions. Fortunately, as far as the scoping algorithm is concerned, we need not be concerned with the exceptional cases, as long as we can avoid multiple inheritance (see Sowa (1991)). The reason for this is that by assuming a certain quantificational constraint we are not defining what relations can hold between various concepts, but merely defining the **possible** conditions that that relation occurs under, if and **whenever** it occurs. That is, quantificational constraints are only used to examine the plausibility of a scope ordering in a certain linguistic context, and **not** the plausibility of the proposition itself. For example, in

- (58) *An American general that was killed in the American Civil war was leading the attack on every front in the 1991 Gulf War.*

the most **plausible scope ordering** can be determined, regardless of the fact that (58) does not describe a **plausible state of affairs**.

4.4 TYPICAL ELEMENTS, INHERITANCE AND QUANTIFICATIONAL CONSTRAINTS

The notion of typical elements can be combined with inheritance in order to define quantificational constraints in an effective manner. Note that the simple representation we have suggested thus far allows us to apply the scope resolution algorithm using QCs that are statically defined; i.e., QCs that have been defined once and for all. Without exploiting inheritance, the amount of data that might have to be stored can be tremendous. To illustrate this point further, consider the following example:

```

class Lift extends Relation
{
  QC(Man p, Piano n) =  $\langle 2^+, 1 \rangle$ 
  QC(Woman p, Piano n) =  $\langle 2^+, 1 \rangle$ 
  QC(Man p, Book b) =  $\langle many, many \rangle$ 
(59)  QC(Man p, Magazine b) =  $\langle many, many \rangle$ 
      QC(Man p, Journal b) =  $\langle many, many \rangle$ 
      QC(Woman p, Book b) =  $\langle many, many \rangle$ 
      QC(Woman p, Magazine b) =  $\langle many, many \rangle$ 
      QC(Woman p, Journal b) =  $\langle many, many \rangle$ 
}

```

The representation of the *Lift* relation in (59) reflects some individual's belief that, typically, it takes at least two *man* or *woman* to lift a single piano, and that any number of *men* or *women* can lift any number of *books*, *magazines*, *journals*, etc. As the reader can easily

verify, the cases that must be considered (one for every combination of a relation and two concepts) can lead to a combinatorial explosion. By exploiting inheritance, (59) can be represented by something like the following:

```

class Lift extends Relation
{
  tense;    // past, present, future
  QC(Person p, Piano n) =  $\langle 2^+, 1 \rangle$ 
  QC(Person p, Publication b) =  $\langle \text{many}, \text{many} \rangle$ 
}
  
```

(60)

Thus, inheritance is used for representational convenience only, and the same result could be achieved by encoding all the combinations individually. The type hierarchy that is implicitly assumed by the definition in (60) is shown in figure 8 below.

Figure 8. Quantificational constraints defined in a concept hierarchy.

Note, however, that by incorporating inheritance we would have to answer the question of what to do in order to account for situations where the context refers to a **relevant specialization of the typical element**. That is, there are situations where a quantificational restriction that is defined on typical elements must be specialized. To illustrate this point further consider the following:

(61) *Five young girls lifted two pianos to the second floor.*

The default quantificational constraint given in (60) states that it takes two (typical) persons to lift a single piano. Perhaps, however, it takes more than two *young girls* to lift a piano, and the direct reading in (61) should not be selected. In other words, we must allow specialized

concepts to re-evaluate the quantificational constraint based on a number of **relevant** features. Thus, instead of a statically defined QC, the QCs defined in (60) could be defined as dynamically computed functions of the relevant features, as follows:

```

class Lift extends Relation
{
  public QC getQC(Person p, Piano n)
  {
    if (typeOf(p) == Man) return (2+, 1);
    else return func(Age(p), Sex(p), Weight(n), ...etc.);
  }

  public QC getQC(Person p, Publication b)
  {
    return (many, many);
  }
}

```

Note that regardless of the (sub)type of *Person* and *Publication*, it is assumed that any number of *Person* can *lift* any number of *Publications*. However, when lifting pianos, the default quantificational constraint returned is a function of the default QC and some of the relevant features of the two concepts.

This formulation of computing QCs as functions of relevant features, could also suggest a method by which we can handle examples such as *every child carried a fake piano*, where presumably a single child can carry several **fake** pianos at once. Of course, there are larger questions here that are beyond the scope of this dissertation, namely the issue of whether a *fake piano* ‘is a’ *piano* (or, how much is a *fake piano* a piano, and how much is it an entirely different **type** of an object, e.g., a *toy*). In other words, computing the appropriate QC in this context will depend on how ‘fake’ contributes to the classification of the concept *fake piano*, which is a question that is beyond the scope of this work.

4.4.1 A Brief Summary

In the above sections we discussed the main issues regarding the knowledge representation that is implicitly assumed by the scoping algorithm of chapter 3. Before we discuss the details of our implementation in chapter 5, we briefly discuss the overall process that we have adopted in testing our approach.

4.5 THE OVERALL PROCESS

Our implementation is concerned with testing the algorithm we suggested in chapter 3. However, prior to applying the scoping rules an input sentence must be parsed and the appropriate semantic objects must be constructed, leading a scope neutral logical form. The overall process that we adopted can be described as follows:

1. The system takes a (declarative) sentence as input.
2. The sentence is parsed, and according to the syntactic rule that succeeds, a corresponding semantic rule is applied. The semantic rule constructs a semantic object of the compound as a function of the semantic objects of the constituents.
3. At the *VP* level, prepositional attachments are resolved, following a strategy that will be described in the next chapter.
4. Once prepositional attachments are resolved, scope ambiguities of the constituent noun phrases (if any) are resolved.
5. At the top level, a logical form for the sentence is constructed, and a top-level scoping function is applied on the entire sentence, whose constituents are assumed to have already been scoped.

This process can be graphically depicted as shown in figure 9 below. Note that in the case of the head NP (NP_1), prepositional phrase attachments are not problematic, since they can only attach to the head NP. In the VP, however, PPs in NP_2 can either attach to the NP or to the relation. The details of this step are discussed in the next chapter.

Figure 9. The overall process of resolving scope ambiguities.

Finally, note that since it is only at the *VP* node that the scope ambiguities of *NP2* can be resolved, the meaning of *NP2* cannot be said to be a function of its constituents only. This raises an important question that we need not delve into in any detail here, except to point out that quantifier scope ambiguities present a serious challenge to compositional semantics. First, it seems that there is real difficulty in accepting that the meaning of a constituent can be determined solely as a function of its components. For example, in the following, the noun phrase '*a restaurant on every street*' can apparently have different meanings due to slight variations in the context outside the NP:

John $\left\{ \begin{array}{l} \text{visited} \\ \text{advertised} \end{array} \right\} \text{a restaurant on every street}$

It is only at the *VP* node that we can determine the possible attachments of '*on every street*'. Decisions regarding quantifier scope will differ considerably, depending on which attachment has been made. Beyond the sentential level, it also seems that a correct interpretation of a sentence (at least with respect to scope preferences), might also be a function of the surrounding linguistic context. For example, consider again sentence (41a) with some additional discourse:

- (62) *Every student attended a seminar. It was on the application of machine learning in information retrieval, and was given by an AI researcher at NRC.*

In 41a' (page 31), we processed the first sentence in isolation, where we only considered the **local** linguistic context at the sentential level. In the absence of discourse information the scoping rules suggested that the reading with narrow scope *a* is the most plausible reading. The indirect reading (wide scope *a*) was considered to be plausible in one situation only, namely in a situation where we actually had one seminar that all students attended. Although this possibility was not likely, given the sentence in isolation, it appears that subsequent information suggests that this unlikely scenario is the actual scenario at hand.

As Zdrozny (1992) observes, therefore, it seems that compositionality is either unattainable, if it is strict compositionality where the meaning of a constituent is a function of its components **only**, or it is a vacuous definition, since otherwise all semantic theories are compositional in the weakest sense. Strong compositionality might perhaps be achieved at larger linguistic boundaries, however. For example, the semantics of a paragraph (see Zdrozny and Jensen (1991)), might be a function of the meanings of its parts only, although

even here it might be argued that discourse information beyond a paragraph might play a crucial role. Another example where the overall (global) context plays an important role is in capturing the correct linguistic context in the scoping rules. For example, in a context such as the following

I was at MIT yesterday. Every student in Computer Science submitted a paper to ACL'95.

the scoping algorithm must treat '*student*' as an anaphoric term that should be resolved with '*MIT student in Computer Science*' in order to appropriately capture the linguistic context.

We will not be concerned here with this larger process of incorporating discourse information and retracting previous inferences. We only point out that what is plausible at the sentential level must eventually be unified with the plausibility of larger linguistic fragments, which could force a reassessment of previously inferred scope preferences. This overall process, therefore, must ultimately include a pragmatic module that integrates anaphor resolution with the resolution of scope ambiguities (Kameyama, 1996), using a *defeasible* inferencing process.

4.6 SUMMARY

In this chapter we have discussed some general issues that relate to an implementation of our proposed strategy for predicting the most plausible quantifier scope ordering. We have made some knowledge representation assumptions that seemed to be quite reasonable in a database environment, and suggested how such a representation could be implemented.

We also discussed the overall process that we followed in our implementation, and briefly discussed some larger issues that have to be dealt with when building a larger pragmatic module that should incorporate a defeasible inferencing strategy, as well as integrate the inferences involved in predicting the most plausible quantifier scope ordering and anaphor (reference) resolution.

In the next chapter we discuss our implementation in some detail. A sample session with the system is given in appendix C.

5 Implementation

In this chapter we discuss in some detail the implementation of the model proposed in this thesis. We describe the main features of *Quasi* (**Q**uantifier scope inferencing algorithm), a Java implementation of the scoping algorithm discussed in the previous two chapters. A sample session with the system is given in appendix B.

5.1 SCOPE OF THE IMPLEMENTATION

5.1.1 Testing the Scoping Algorithm

This thesis is concerned with proposing a new strategy for resolving quantifier scope ambiguities in a computational environment, where the system must decide on an unambiguous logical form. In order to test our inferential strategy, a simple natural language 'processor' must be built. We emphasize the word 'processor' here, since for our purposes it is not necessary to build a natural language 'interpreter.' The difference between the two is the difference between a logical form and its evaluation against some database. In other words, a natural language query such as

(63) *Did John receive a passing grade in every course?*

is said to be fully interpreted only when the system can verify the corresponding (formal) query against some database, and an answer (in this case, *yes* or *no*) is generated. Thus, given input in the form of a natural language query, a natural language interpreter generally performs a number of operations, including syntactic and semantic analysis, and finally the evaluation (interpretation) of an ambiguous logical form (a formal query) against some database. For our purposes, however, this last step is not required. We simply apply a scoping function on an unscoped (scope neutral) logical form, the result of which is the reading with the preferred scope ordering. In other words, a scoped logical form, and not its evaluation (interpretation), is the final output that we are concerned with here.

5.1.2 Function (Process) vs. Data

Another important aspect of our implementation is our decision to develop a general process that can be parameterized by an (individual's) knowledge base. Our concern was to suggest what the *process* of resolving quantifier scope ambiguities might be, without making any commitments regarding the *data* that this process might depend on, including quantificational constraints, which reflect an individual's biases.

Developing a generic module is a challenging software engineering task. Like any other computer program, a natural language processing program conforms with the old motto that computer scientists are familiar with, namely that a *program* = *data* + *control*. If a program is a collection of data and functions (processes), then the following argument seems to be quite reasonable:

$$\begin{aligned} \text{program} &= \text{data} + \text{functions} \\ \text{program} &= \text{solution} \\ \Rightarrow \text{solution} &= \text{data} + \text{functions} \\ \Rightarrow \text{generalized}(\text{solution}) &= \text{generalized}(\text{data} + \text{functions}) \end{aligned}$$

In other words, to generalize a solution, one could generalize the *data*, the *functions*, or, of course, both²⁹. In this thesis, we made commitments to a *process* by which the preferred quantifier scope ordering can be inferred, regardless of what some individual's quantificational constraints, or what some other parameters such as the typical cardinality constraints might be. In other words, we made a commitment to defend a hypothesis regarding a *process* and not any specific *data*. As a consequence of this, the question of what specific ontology and knowledge representation we have chosen is not critical.

The only knowledge that has been hard coded in the system is the quantificational constraints and the typical cardinalities of the sets. While the default values that have been entered reflect the author's, they could be modified by the user. That is, in order to test this data-independent process, we have allowed the user to test various situations by changing the linguistic context as well as the relevant quantificational constraints, which are the two most important parameters in the inferencing strategy that we proposed. The screen shot in Figure 10 below shows the main functions that the user can perform in *Quasi*.

5.1.3 The Knowledge Representation

As we have already stated above, in this dissertation we are not committed to any knowledge representation *per se*, and a simple entity-relationship (ER) model suffices for our purposes. That is, in inferring the preferred scope ordering for the sentence *John visited a house on every*

²⁹ Interestingly, the two most recent developments in programming languages, namely the object-oriented paradigm and the pure functional programming paradigm, offer dramatically different *views* on what the main building blocks in constructing *reusable* (abstract/generic) code should be, based on different interpretations of that equation. In the object-oriented paradigm the emphasis is on *data* abstraction, where in the functional programming paradigm the emphasis is on *functional* abstraction.

street, the system need only know that *House* and *Street* are entities related to each other by the *LocatedOn* relation, which has a specific QC, in this case $\langle many, 1 \rangle$.

Figure 10. A screen shot of the *Quasi* interface.

What we want to underline here is that, in our implementation, we make no knowledge representation assumptions whatsoever: concepts need not contain any features, nor even form any sort of a hierarchy. In fact, since we are not concerned with evaluating the final scoped logical form, the entities can simply be represented as (empty) nodes, and relations as simple links. The reason for this conscious effort not to *hardcode* any specific knowledge is, again, our aim to describe and defend a *process*, and not any specific *data*. The following is a sample of how concepts and relations are represented in *Quasi*:

```
class Concept{
  int typicalCardinality = defaultTC;
  Concept( int tc ) { typicalCardinality = tc; }
};

class House extends Concept { House(int tc) {super(tc);} };
class Street extends Concept { Street(int tc) {super(tc);} };
class Book extends Concept { Book(int tc) {super(tc);} };
class Shelf extends Concept { Shelf(int tc) {super(tc);} };

class On extends Relation{
  QC getQC(House,Street) = makeQC(⟨many,1⟩,⟨typ,nec⟩)
  QC getQC(Book,Shelf) = makeQC(⟨many,1⟩,⟨typ,typ⟩)
};
```

As can be seen from this sample, the only assumptions we make are that every concept has a typical cardinality, and that a quantificational constraint is defined for every triple of two concepts and a binary relation. Both of these parameters have default values that could be modified by the user. Finally, a point must be made here regarding the typical cardinality of a set. This measure, as we discussed in chapter 4, must in practice be updated by some special function that takes into consideration the history and knowledge of a certain database. Currently, these values are not automatically (and periodically) updated by such functions, but are simply set by the user.

5.1.4 Extending the Knowledge Representation

In spite of the arguments made above, we have extended the representation in two important ways and for two different reasons. First, we have assumed that concepts are represented in a type hierarchy. The reason for this, as we argued in chapter 4, was to exploit inheritance and reduce the amount of information that has to be stored. For example, we take the following

```
(64)
class Lift extends Relation
{
  tense;    // past, present, future
  QC(Person p, Piano n) =  $\langle 2^+, 1 \rangle$ 
  QC(Person p, Publication b) =  $\langle many, many \rangle$ 
}
```

to be the representation of the *Lift* relation instead of one where a QC has to be defined for every possible subtype of *Person* and every subtype of *Piano*, and similarly for all combinations of *Person* and *Publication*.

Exploiting inheritance for storage effectiveness does not come for free, however. In particular, we must now deal with exceptions, since it cannot be assumed that all subtypes of a certain concept obey the same QC in all contexts. As discussed in chapter 4, defining QCs on concepts that form a type hierarchy requires that we consider *typical elements*, and, subsequently, define QCs as dynamically computed functions of some relevant set of features. The reason for this is that it is very restrictive to define a QC on the *Lift* relation, for example, in such a way that it would equally apply to all subtypes of *Person* and *Piano*. For instance, while some individual might believe that, in general, it will take at least two *Men* to lift a piano, they might believe that it takes at least four *LittleGirls* to lift a single piano. Thus, a very flexible representation might be one where QCs could be computed as functions

of the relevant features, and where the statically defined QC remains to be the default value. The following shows how such a representation can be implemented:

```
class Lift extends Relation
{
  public QC getQC(Person pr,Piano pn)
  {
    if(((pr.getClass()).toString())
        .compareTo("Person")==0)
      return <2+,1>;
    else return func(Age(pr),Gender(pr),Weight(pn));
  }

  public QC getQC(Person pr,Publication pb)
  {
    return <many,many>;
  }
};
```

Note that whatever the subtype of *Person* and *Publication* (i.e., man or woman, lifting a journal, a magazine, or a book, etc.), the QC is <many,many>. However, in the case of lifting a piano, we will in general (i.e., by default), assume that it takes at least two persons to lift a piano. If the type of the person was specified further, a function is used to modify the default QC using some features that could be relevant in this context.

The above discussion summarizes all the important aspects of the concept representation in *Quasi*. We discuss the representation of all these semantic objects (concepts, relations and logical forms) further later. For now, however, we turn to the syntactic and semantic analysis part of our implementation.

5.2 THE PARSER

We employed a simple top-down recursive descent parser which was implemented in a process that resembles an attribute grammar (Knuth, 1968). In an attribute grammar (AG) a number of attributes are associated with each syntactic category. These attributes are separated into *synthesized* and *inherited* attributes. Given a syntactic category *S*, whose components are *s1* and *s2*, we compute the synthesized attributes of *S* as functions of the (synthesized) attributes of the constituents, i.e., as functions of the attributes of *s1* and *s2*. The inherited attributes of *S* are those attributes whose values are passed as a *context* to the evaluation of the synthesized attributes of *S*. In a pure and declarative paradigm the

computation of synthesized attributes has access to inherited (context) attributes via the arguments to constructors³⁰ and functions, as opposed to global environment variables.

```
class LingObj
{
    Meaning    meaning;    // every syntactic object has a meaning (object)
    String     parseTree;  // every syntactic object has a parse tree
    Vector     tokens;     // the tokens that make up the syntactic object
    Vector     restOfInput; // the input tokens that are left for a subsequent syntactic object
    boolean    parsedOK;   // true, if the input starts with a set of tokens that make a C object
    Vector     input;      // the input text that 'this' linguistic object will parse

    // the constructor that, given an input string performs some initializations
    LingObj(Vector words)
    {
        input = words;
        parsedOK = false;
        etc...
    }

    public void parse( )
    {
        // this is specialized for each syntactic object
        // according to the relevant syntactic rule(s)
    }

    // this function implements the '→' (or the ' ::= ' ) functor in a BNF specification
    // the input is a list of syntactic objects that make one case/rule of constructing
    // 'this' syntactic object (e.g., NP = consistsOf (Det, TP))
    private LingObj consistsOf (LingObj lobj1, LingObj lobj2, ..., LingObj lobjn)
    {
    }

    // this function implements the 'I' functor in a BNF specification
    // the input is a list of rules (all cases) that 'this' syntactic object can apply
    // (e.g., NP = orElse (consistsOf (Det, TP), consistsOf (Det, TP, PP)))
    private Vector orElse (Rule r1, Rule r2, ..., Rule rn)
    {
    }
};
```

Figure 11. The attributes and methods of a generic linguistic object.

Attribute grammars fit very naturally into the object-oriented paradigm, where syntactic categories can be modeled as objects whose synthesized attributes are represented as their data members. In such a representation parsing can be viewed as object construction and the evaluation of synthesized attributes are performed by private methods (see appendix A for

³⁰ In object-oriented programming a constructor of an object can be viewed as a special function (method) that creates the data structure (allocates space for the object) and properly initializes the (state of the) object. It is also convenient to have more than one constructor defined, each of which can initialize the state of the object differently, depending on the context. The reader is referred to (Gosling *et al.*, 1996) for an introductory look at Java and object-oriented programming (OOP).

illustration of an object-oriented implementation of an attribute grammar). In what follows we briefly describe the strategy we adopted in *Quasi* for an object-oriented implementation of an attribute grammar. In an object-oriented implementation of an attribute grammar an object (a Java class) is defined for every syntactic category. Moreover, and since a number of generic methods (e.g., parsing methods) are used by all syntactic categories, all syntactic categories are defined as subtypes of a generic object, called *LingObj*, whose structure and methods (behavior) are common to (inherited by) all the other linguistic objects. The structure and behavior of this generic linguistic object is shown in figure 11 above.

First, note that the generic linguistic object has a number of attributes (data members) that are shared by all linguistic objects. That is, every linguistic object has a *meaning*, a *parseTree*, etc. Note also that every linguistic object has a *parse* method, the logic of which is different depending on the specific syntactic object. For terminal symbols, the *parse* method is a lexical analyzer that simply performs a dictionary lookup. For non-terminal symbols, however, the *parse* method calls the **private** methods *consistsOf* and *orElse* that roughly correspond to '::=' and '|' in a BNF specification. These two higher-order functions are used to specify the possible components in a syntactic rule (*consistsOf*), as well as all the alternative rules that a syntactic construct can apply (*orElse*). The following generic example shows how the *parse* method of a specific construct uses *consistsOf* and *orElse* to implement the syntactic rules.

```
// The class C defined below corresponds to some syntactic
// construct C that has the following BNF rules
//  $\langle C \rangle \rightarrow \langle C_1 \rangle \langle C_2 \rangle$ 
//  $\langle C \rangle \rightarrow \langle C_1 \rangle \langle C_2 \rangle \langle C_3 \rangle$ 

class C extends LingObj
{
    C1      c1;
    C2      c2;
    C3      c3;

    // initialize using the constructor of the generic linguistic object
    C(Vector input){ super(input); }

    public void parse( )
    {
        Vector cases = orElse( consistsOf(c1,c2,c3), consistsOf(c1,c2) );

        // rule - to - rule correspondence (for every syntactic rule,
        // there's a corresponding semantic rule)
        if (rulePassed(1,cases))    computeMeaning1(c1,c2,c3);
        else if (rulePassed(2,cases)) computeMeaning2(c1,c2);
        else
            failParse( );
    }
};
```

Note the use of *consistsOf* and *orElse* in implementing the BNF rules of the linguistic construct *C*. Note also the 'syntactic sugaring' that was achieved by overloading these functions so that they would appear to be functions that take a variable number of arguments. This technique is quite useful since the grammar is now very extensible. If, for example, we wanted to consider an additional rule for *C*, all we need to do is insert another *consistsOf* argument to the *orElse* call, and define a corresponding semantic rule. The alternative (which we had started with), is to have complicated if-then-else logic in every *parse* method, which would greatly complicate the task of modifying/extending the grammar.

It should also be noted here that since grammars involve recursion, the *orElse* and *consistsOf* methods could be given an argument of the same type as that of the object being parsed. This recursive object construction works fine as long cycles are avoided; i.e., as long as the terminating cases can be reached. The syntactic objects (for both terminal and non-terminal symbols) and the syntactic rules that were covered in *Quasi* are given in the BNF specification shown below (*TP* corresponds to 'term phrase'):

$$\begin{array}{lll}
 \langle SENT \rangle \rightarrow \langle NP \rangle \langle VP \rangle & \langle NP \rangle \rightarrow \langle NP_0 \rangle & \langle PP \rangle \rightarrow \langle PREP \rangle \langle NP \rangle \\
 \langle NP_0 \rangle \rightarrow \langle DET \rangle \langle TP \rangle & \langle TP \rangle \rightarrow \langle ADJS \rangle \langle NOUNS \rangle & \langle RCLAUSE \rangle \rightarrow \langle RELPRON \rangle \langle VP \rangle \\
 \langle NP \rangle \rightarrow \langle NP_0 \rangle & \langle TP \rangle \rightarrow \langle ADJS \rangle \langle NOUNS \rangle & \langle VP \rangle \rightarrow \langle VERB \rangle \langle NP \rangle \\
 \langle NP \rangle \rightarrow \langle NP_0 \rangle \langle PP \rangle & \langle NOUNS \rangle \rightarrow \langle NOUN \rangle^+ & \\
 \langle NP \rangle \rightarrow \langle NP_0 \rangle \langle RCLAUSE \rangle & \langle ADJS \rangle \rightarrow \langle ADJ \rangle^+ &
 \end{array}$$

Finally, and leading to our discussion on the semantic analysis, we note that whenever a syntactic rule succeeds, the parse is terminated and the appropriate semantic rule is called to compute the meaning of the compound from the meanings of the constituents. That is, in *Quasi* structural ambiguities were ignored by explicitly ordering the rules according to which rule should be preferred when more than one succeeds.

5.3 SEMANTIC ANALYSIS

Like syntactic analysis, semantic analysis is performed using standard techniques, in this case a somewhat compositional approach that computes the meaning of a compound as function of the meanings of the parts. However, it should be noted it is the scope neutral logical form corresponding to the entire sentence that is computed compositionally, and *not* the final interpretation, which is the final **scoped** logical form. As will be discussed shortly,

the final interpretation is a function of all the constituents, as well as the relevant quantificational constraints.

The semantic analysis starts with building a semantic object for every terminal symbol. Compound semantic objects (like clauses, logical forms, etc.) are built up according to the syntactic construction. Thus, in *Quasi* there is a semantic object (a meaning) corresponding to every syntactic object as shown in table 3 below. Like the syntactic objects, all semantic objects have a root type, called *Meaning*. The structure of these meaning objects is quite simple, however, since we were not concerned with evaluating the final logical form and question/answering *per se*.

The semantic rules in *Quasi* are used to build up complex semantic objects from the semantic objects of the constituents, according to their rules of construction. For most cases, these rules are straightforward, except for the case of prepositional phrase (PP) attachments and the processing of quantified concepts.

In the case of PPs, some procedure must be in place to decide when a PP should modify the noun, and when it occupies a modifying slot (a case role) in the relation. For quantified concepts we must consider the situation when an NP itself involves relations between two quantified concepts and potentially scope ambiguities within the NP. We discuss both of these issues in the following sections.

Syntactic Object	Semantic Object
<i>SENT</i>	<i>Logical Form (proposition)</i>
<i>VP</i>	<i>Clause</i>
<i>NP</i>	<i>Quantified Concept</i>
<i>TP</i>	<i>Modified Concept</i>
<i>NOUN (S)</i>	<i>(possibly a modified) Concept</i>
<i>ADJ (S)</i>	<i>Modifier(s)</i>
<i>PP</i>	<i>Attachment (either to an NP or to a Relation - see text)</i>
<i>RCLAUSE</i>	<i>Modifier</i>
<i>DET</i>	<i>Quantifier</i>
<i>VERB</i>	<i>Relation</i>

Table 3. The correspondence between syntactic and semantic objects in *Quasi*.

Before we turn to this discussion, however, we must discuss an important semantic object that is used to generate the logical form, namely the object *Expression*.

5.3.1 Logical Expressions

An important semantic object out of which logical forms are recursively constructed is *LogicalExpression*, which could be thought of as a well-formed formula (*wff*) in first-order logic. In constructing logical expressions we 'loosely' followed Montague's (1974) treatment of quantified term phrases which we discuss in some detail in chapter 6.

Essentially the *Expression* object takes as input a quantifier, a quantified concept, or a proper name and generates the corresponding logical expression, as shown below:

$$\text{Expression}(\text{every}) \Rightarrow \text{FORALL}(x1)[P(x1) \rightarrow Q1(x1)]$$

$$\text{Expression}(\text{a man}) \Rightarrow \text{EXISTS}(x2)[\text{MAN}(x1) \& Q2(x2)]$$

$$\text{Expression}(\text{every old man}) \Rightarrow \text{FORALL}(x1)[\text{MAN}(x1) \& \text{OLD}(x1) \rightarrow Q(x1)]$$

$$\text{Expression}(\text{a university student}) \Rightarrow \text{EXISTS}(x1)[\text{UNIVERSITY} - \text{STUDENT}(x1) \& Q(x1)]$$

First we point out that logical expressions are constructed in a *curried*³¹ fashion, which allows us to 'partially' evaluate an expression and delay the full evaluation until scope ordering commitments can be made. Note also that the expression generated depends on the quantifier and the syntactic structure of the noun phrase. For instance, noun modifiers are assumed to subtype the noun, while adjective modifiers result in a conjunction of predicates.

Note also that care must be taken in consistently assigning variables, since at higher levels where unification of expressions occurs this will be very crucial. Finally, it should be noted that unlike Montague's uniform treatment of quantified NPs, proper names in *Quasi* are simple constants.

A number of important operations (methods) are defined on the *LogicalExpression* object, the most important of which is the *Apply* method which takes an expression and unifies it with (i.e., applies it to) another expression.

Here's an illustration of this process:

³¹ This term comes from the mathematician Howard Curry who saw the utility in performing *partial evaluations* of functions. In this technique applying a function of *n* arguments on an argument returns a function of *n-1* arguments, which takes an argument and returns a function of *n-2* arguments, etc. A constant function, one that expects no arguments, is the final value (thus every object is a function!)

$$\begin{aligned}
& \text{Expression}(\text{every child}) \Rightarrow \text{FORALL}(x1) [\text{CHILD}(x1) \rightarrow Q1(x1)] \\
& \text{Expression}(\text{a cookie}) \Rightarrow \text{EXISTS}(x2) [\text{COOKIE}(x2) \& Q2(x2)] \\
& \text{Expression}(\text{ate}) \Rightarrow \text{ATE}(x, y) \\
& \text{Expression}(\text{ate (a cookie)}) \\
& = \text{Apply}(\text{Expression}(\text{a cookie}), \text{Expression}(\text{ate})) \\
& = \text{EXISTS}(x2) [\text{COOKIE}(x2) \& Q2(x2)] [\text{ATE}(x, y)] \\
& = \text{EXISTS}(x2) [\text{COOKIE}(x2) \& \text{ATE}(x, x2)] \\
& \text{Expression}(\text{every child (ate (a cookie))}) \\
& = \text{Apply}(\text{Expression}(\text{every child}), \text{Expression}(\text{ate (a cookie)})) \\
& = \text{FORALL}(x1) [\text{CHILD}(x1) \rightarrow Q1(x1)] [\text{EXISTS}(x2) [\text{COOKIE}(x2) \& \text{ATE}(x, x2)]] \\
& = \text{FORALL}(x1) [\text{CHILD}(x1) \rightarrow \text{EXISTS}(x2) [\text{COOKIE}(x2) \& \text{ATE}(x, x2)]]
\end{aligned}$$

There are a number of important observations to be made here. First, while the process of constructing logical expressions is straightforward, care must be taken in variable binding and unification. Second, note that the *Apply* method might replace a 1-place predicate with a 2-place predicate, in which case the variables must be inserted in the right slot. This happens when unifying expressions at the clausal level where the binary relation has been identified.

Finally, and of immediate concern to us here, is that scope orderings are determined by the order in which expressions are applied.

For example, the last expression computed above could have been generated as

$$\begin{aligned}
& \text{Expression}((\text{ate (a cookie)}) (\text{every child})) \\
& = \text{Apply}(\text{Expression}(\text{ate (a cookie)}), \text{Expression}(\text{every child})) \\
& = \text{EXISTS}(x2) [\text{COOKIE}(x2) \& \text{ATE}(x, x2)] [\text{FORALL}(x1) [\text{CHILD}(x1) \rightarrow Q1(x1)]] \\
& = \text{EXISTS}(x2) [\text{COOKIE}(x2) \& \text{FORALL}(x1) [\text{CHILD}(x1) \rightarrow \text{ATE}(x1, x2)]]
\end{aligned}$$

In *Quasi*, expressions for individual (simple) quantified concepts are immediately generated, although these expressions are not unified (applied) until scope decisions have been made. This step will occur after prepositional attachments have been resolved, and the complete logical form has been constructed (see figure 9, p. 57). Roughly speaking, therefore, the scope neutral logical form is the set of all possible combinations by which we could *Apply* (unify) expressions, since each application represents a scope ordering commitment.

5.3.2 Processing Prepositional Phrases

First, let us consider a simple sentence to illustrate the basic strategy that we follow in building up complex semantic objects in *Quasi*. In figure 12 below we detail the recursive construction of semantic objects in the analysis of *few MIT students submitted a paper*. Note

that MIT was ultimately attached as a modifier to the concept *Student*, since, syntactically, it was processed as a noun modifier. Prepositional phrases that appear in the head noun phrase also result in a modifier, much like the case of an adjective or a noun modifier³².

This simple case of prepositional phrases is considered first by processing the sentence *few students from MIT submitted a paper* in figure 13 below. Semantically, a PP is an *ATTACHMENT*. The decision as to where this attachment belongs must be made at a higher-level. In the case of a head NP, the decision is made at the NP level where there are no relations that the PPs can attach too. An example of how the PP in the head noun phrase results in a modifier is shown in figure 13 below.

The more interesting case, however, is when the PPs are part of the verb phrase's NP, where a PP could in principle attach either to the NP or to the relation. In these situations decisions regarding PP attachment must be delayed until the construction of the *Clause* semantic object, which corresponds to a VP, since it is at the clausal level that all the information needed to decide on the attachment of PPs is available. Regarding this step, we have adopted a very simple strategy that can be summarized as follows:

- At the clausal level, a recursive traversal of all attachments to the quantified concept of the VP is made. An attachment can be made to one NP only, but could also be an attachment for the relation.
- From the bottom-up (or structurally, from the right-most PP), a search for an attachment 'candidate' is performed. The 'next' candidate for attachment is *np1*, the NP one level higher than the attachment, followed by *np2*, the NP one level higher than *np1*, and so on. As soon as a relation with a candidate NP is found, the attachment is made. (This simple heuristic says that if an attachment is possible on several NPs, the nearest NP is the most likely NP this attachment modifies.)

The possibility of a reading with an attachment of a PP to the relation is also tested with every attachment.

³² Recall that the scoping algorithm takes modifiers (both adjectives and nouns) into consideration, since they affect the size of the quantified concept, which turns out to be an important measure that captures the role of the linguistic context. In our implementation each adjective and noun modifier reduces the size of the concept by a certain ratio. We have also incorporated the heuristic that noun modifiers reduce the size of the concept more so than adjective modifiers.

$$\begin{aligned}
& \text{LogicalForm}(\text{Sentence}(\text{few MIT students submitted a paper})) \\
&= \text{meaningOf} \left(\begin{array}{l} \text{meaningOf}(\text{NP}(\text{few MIT students})), \\ \text{meaningOf}(\text{VP}(\text{submitted a paper})) \end{array} \right) \\
&= \text{meaningOf} \left(\begin{array}{l} \text{meaningOf} \left(\begin{array}{l} \text{meaningOf}(\text{DET}(\text{few})), \\ \text{meaningOf}(\text{TP}(\text{MIT students})) \end{array} \right), \\ \text{meaningOf} \left(\begin{array}{l} \text{meaningOf}(\text{VERB}(\text{submitted})), \\ \text{meaningOf}(\text{NP}(\text{a paper})) \end{array} \right) \end{array} \right) \\
&= \text{meaningOf} \left(\begin{array}{l} \text{meaningOf} \left(\begin{array}{l} \text{Quantifier } [\text{few}] \\ \text{Concept } [[\text{MIT}]\text{Student}] \end{array} \right), \\ \text{meaningOf} \left(\begin{array}{l} \text{Relation } [\text{Submit}], \\ \text{meaningOf} \left(\begin{array}{l} \text{meaningOf}(\text{DET}(\text{a})), \\ \text{meaningOf}(\text{TP}(\text{paper})) \end{array} \right) \end{array} \right) \end{array} \right) \\
&= \text{meaningOf} \left(\begin{array}{l} \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier } [\text{few}] \\ \text{Concept } [[\text{MIT}]\text{Student}] \end{array} \right], \\ \text{meaningOf} \left(\begin{array}{l} \text{Relation } [\text{Submit}], \\ \text{meaningOf} \left(\begin{array}{l} \text{Quantifier } [\text{a}], \\ \text{Concept } [\text{paper}] \end{array} \right) \end{array} \right) \end{array} \right) \\
&= \text{meaningOf} \left(\begin{array}{l} \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier } [\text{few}] \\ \text{Concept } [[\text{MIT}]\text{Student}] \end{array} \right], \\ \text{meaningOf} \left(\begin{array}{l} \text{Relation } [\text{Submit}], \\ \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier } [\text{a}] \\ \text{Concept } [\text{paper}] \end{array} \right] \end{array} \right) \end{array} \right) \\
&= \text{meaningOf} \left(\begin{array}{l} \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier } [\text{few}] \\ \text{Concept } [[\text{MIT}]\text{Student}] \end{array} \right], \\ \text{Clause} \left[\begin{array}{l} \text{Relation } [\text{Submit}] \\ \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier } [\text{a}] \\ \text{Concept } [\text{paper}] \end{array} \right] \end{array} \right] \end{array} \right) \\
&= \text{LogicalForm} \left[\begin{array}{l} \text{Relation } [\text{Submit}] \\ \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier } [\text{few}] \\ \text{Concept } [[\text{MIT}]\text{Student}] \end{array} \right] \\ \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier } [\text{a}] \\ \text{Concept } [\text{paper}] \end{array} \right] \end{array} \right]
\end{aligned}$$

Figure 12. The semantic analysis of *few MIT students submitted a paper*.

$$\begin{aligned}
& \text{LogicalForm}(\text{Sentence}(\text{few students at MIT submitted a paper})) \\
&= \text{meaningOf} \left(\text{meaningOf}(\text{NP}(\text{few students at MIT})), \right. \\
&\quad \left. \text{meaningOf}(\text{VP}(\text{submitted a paper})) \right) \\
&= \text{meaningOf} \left(\text{meaningOf} \left(\text{meaningOf}(\text{DET}(\text{few})), \right. \right. \\
&\quad \left. \left. \text{meaningOf}(\text{TP}(\text{students at MIT})) \right), \right. \\
&\quad \left. \text{meaningOf}(\text{VERB}(\text{submitted})), \right. \\
&\quad \left. \text{meaningOf}(\text{NP}(\text{a paper})) \right) \\
&= \text{meaningOf} \left(\text{meaningOf} \left(\text{Quantifier} [\text{few}], \right. \right. \\
&\quad \left. \left. \text{meaningOf} \left(\text{meaningOf}(\text{TP}(\text{students})), \right. \right. \right. \\
&\quad \left. \left. \left. \text{meaningOf}(\text{PP}(\text{at MIT})) \right) \right) \right), \\
&\quad \left. \text{meaningOf} \left(\text{Relation} [\text{Submit}], \right. \right. \\
&\quad \left. \left. \text{meaningOf} \left(\text{meaningOf}(\text{DET}(\text{a})), \right. \right. \right. \\
&\quad \left. \left. \left. \text{meaningOf}(\text{TP}(\text{paper})) \right) \right) \right) \\
&= \text{meaningOf} \left(\text{meaningOf} \left(\text{Quantifier} [\text{few}], \right. \right. \\
&\quad \left. \left. \text{meaningOf} \left(\text{Concept} [\text{Student}], \right. \right. \right. \\
&\quad \left. \left. \left. \text{meaningOf} \left(\text{meaningOf}(\text{PREP}(\text{at})), \right. \right. \right. \right. \\
&\quad \left. \left. \left. \left. \text{meaningOf}(\text{NP}(\text{MIT})) \right) \right) \right) \right), \\
&\quad \left. \text{meaningOf} \left(\text{Relation} [\text{Submit}], \right. \right. \\
&\quad \left. \left. \text{meaningOf} \left(\text{Quantifier} [\text{a}], \right. \right. \right. \\
&\quad \left. \left. \left. \text{Concept} [\text{paper}] \right) \right) \right) \\
&= \text{meaningOf} \left(\text{meaningOf} \left(\text{Quantifier} [\text{few}], \right. \right. \\
&\quad \left. \left. \text{meaningOf} \left(\text{Concept} [\text{Student}], \right. \right. \right. \\
&\quad \left. \left. \left. \text{meaningOf} \left(\text{meaningOf}(\text{PREP}(\text{at})), \right. \right. \right. \right. \\
&\quad \left. \left. \left. \left. \text{QuantifiedConcept} [\text{MIT}] \right) \right) \right) \right), \\
&\quad \left. \text{meaningOf} \left(\text{Relation} [\text{Submit}], \right. \right. \\
&\quad \left. \left. \text{meaningOf} \left(\text{QuantifiedConcept} [\text{a}], \right. \right. \right. \\
&\quad \left. \left. \left. \text{Concept} [\text{paper}] \right) \right) \right) \\
&= \text{meaningOf} \left(\text{meaningOf} \left(\text{Quantifier} [\text{few}], \right. \right. \\
&\quad \left. \left. \text{meaningOf} \left(\text{Quantifier} [\text{few}], \right. \right. \right. \\
&\quad \left. \left. \left. \text{Concept} [\text{Student}] \right. \right. \right. \\
&\quad \left. \left. \left. \text{Attachment} \left[\begin{array}{l} \text{Prep} [\text{at}] \\ \text{QuantifiedConcept} [\text{MIT}] \end{array} \right] \right) \right) \right), \\
&\quad \left. \text{Clause} \left[\begin{array}{l} \text{Relation} [\text{Submit}] \\ \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier} [\text{a}] \\ \text{Concept} [\text{paper}] \end{array} \right] \end{array} \right] \right) \\
&= \text{meaningOf} \left(\text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier} [\text{few}] \\ \text{Concept} [\text{Student}] \\ \text{nModifiers} [\text{MIT}] \end{array} \right] \right. \\
&\quad \left. \text{Clause} \left[\begin{array}{l} \text{Relation} [\text{Submit}] \\ \text{QuantifiedConcept} \left[\begin{array}{l} \text{Quantifier} [\text{a}] \\ \text{Concept} [\text{paper}] \end{array} \right] \end{array} \right] \right)
\end{aligned}$$

Figure 13. Prepositional phrase attachment: the simple case.

To illustrate how this process works, let us consider the sentence *few students at MIT submitted a paper on quantification to ACL-95*. Prior to making any decisions regarding PP attachments, the processing will proceed exactly as shown in figure 13 above. The crucial

step, however, is during constructing the *Clause* object from the meanings of the relation and the associated NP. At this stage, we would have the following situation:

$$= \text{meaningOf} \left(\begin{array}{l} \text{meaningOf} \left(\begin{array}{l} \text{meaningOf}_1 \left(\begin{array}{l} \text{Quantifier} \text{ [few]}, \\ \text{Concept} \text{ [Student]} \\ \text{Attachment} \left[\begin{array}{l} \text{Prep} \text{ [at]} \\ \text{QuantifiedConcept} \text{ [MIT]} \end{array} \right] \end{array} \right) \\ \text{meaningOf} \left(\begin{array}{l} \text{Relation} \text{ [Submit]}, \\ \text{meaningOf}_2 \left(\begin{array}{l} \text{Quantifier} \text{ [a]}, \\ \text{Concept} \text{ [Paper]} \\ \text{Attachment}_1 \left[\begin{array}{l} \text{Prep} \text{ [on]} \\ \text{QuantifiedConcept} \text{ [Quantification]} \end{array} \right] \\ \text{Attachment}_2 \left[\begin{array}{l} \text{Prep} \text{ [to]} \\ \text{QuantifiedConcept} \text{ [ACL95]} \end{array} \right] \end{array} \right) \end{array} \right) \end{array} \right) \end{array} \right)$$

As explained above, when processing the head NP (when applying the function *meaningOf*₁ above), the PP is attached as a modifier on the NP. It is in the second case, when computing *meaningOf*₂, that decisions regarding the attachment(s) must be made, since they could in principle attach either to the relation or to the NP of the VP. Part of this process entails a search for the following relations:

On(Paper, Subject) // since *ISA*(Quantification, Subject)
To(Paper, Event) // since *ISA*(ACL95, Conference) $\wedge$ *ISA*(Conference, Event)

The first attachment will be on the NP if the *On* relation exists between *Paper* and *Subject*, and similarly for the second attachment. Essentially, the reasoning behind this simple strategy is that a PP describes a relation between two noun phrases, if such a relation exists, otherwise, it is a modifier for the relation. Note that two different readings will be generated if the attachment could be made to both the relation and the NP. For example, in *John advertised a house on every street*, the final scope decisions differ considerably depending on whether the PP attaches to 'a house' or 'advertised'.

5.3.3 Processing Quantified Noun Phrases

When prepositional phrases attach to the noun phrase, the possibility arises of quantifier scope ambiguity within the NP itself. The following shows the structure (ignoring the methods) of a quantified concept in *Quasi*:

5.4 THE SCOPING FUNCTION

Thus far we discussed how *Quasi* parsed and built up semantic objects, the most important of which is a scope-neutral logical form. The following shows the steps followed in *Quasi* in determining the set of plausible scope orderings of a sentence:

```

Sentence      sent = new Sentence("several senators consulted with
                                a scientist from every country");

LogicalForm   lf      = sent.getLogicalForm( );
Vector        plausibleReadings = lf.scope( );
LogicalForm   preferredReading = (LogicalForm)
                                plausibleReading.elementAt(0);

```

A sentence object is first created, which entails parsing the sentence, and, if successful, creating all the semantic objects recursively leading into the semantic object corresponding to the sentence, namely its logical form. The *scope* method on the logical form object is then called returning all the plausible readings in decreasing order with respect to their measure of plausibility.

Therefore, one of the important methods in the *LogicalForm* object is *scope*, which returns the set of all plausible scope orderings as a result. The following shows the overall structure and the main functions in the *LogicalForm* object:

```

class LogicalForm extends Meaning
{
    Relation      relation;
    QuantifiedConcept qConcept1;
    QuantifiedConcept qConcept2;

    Vector        plausibleReadings;

    // a logical form is made up from an NP and a Clause
    LogicalForm(QuantifiedConcept qc, Clause c)
    {
        relation = c.relation;
        qConcept1 = qc;
        qConcept2 = c.qConcept;
    }

    public Vector scope( )
    {
        double r1 = scopeTop( relation, scopeQC(qConcept1), scopeQC(qConcept2));
        double r2 = scopeTop( relation, scopeQC(qConcept2), scopeQC(qConcept1));
        addResult(r1, r2);
    }

    private QuantifiedConcept scopeQC(QuantifiedConcept qc);
};

```

Note that the *scope* function recursively applies a private scope method on all the constituent quantified noun phrase, since, as we discussed in chapter 3, the noun phrases themselves could involve quantifier scope ambiguities. Also note that eventually the *scope* function calls a top-level scoping algorithm which computes a measure of plausibility for each possible scope ordering. The result of each call is a measure of plausibility, which is a real value between 0 and 1.

Before we discuss this top-level scoping algorithm, which is the implementation of the algorithm described in chapter 3, let us look at the *scopeQC* function that recursively scopes all NPs that have attachments (which in turn could have attachments). This function, as discussed above, is only called after the attachments have been resolved. That is, an attachment on an NP at this point is assumed to be a genuine attachment, corresponding to some relation. The *scopeQC* function is defined as follows:

```
private QuantifiedConcept scopeQC(QuantifiedConcept qc)
{
    // an NP with no PPs has no scope ambiguities
    if (qc.noAttachment()) return qc;

    // a Proper Name has no scope ambiguities
    if (qc.isProperName()) return qc;

    Relation r = (qc.getAttachment()).getRelation();
    QuantifiedConcept qcAttached = (qc.getAttachment()).getQC();

    // measure the plausibility of the direct and indirect readings
    double r1 = scopeTop(relation, qc, scopeQC(qcAttached));
    double r2 = scopeTop(relation, scopeQC(qcAttached), qc);

    // store both plausibilities for subsequent use
    qc.addResults(r1, r2);

    if ( r1 > r2 ) return qc;
    else return qc.indirectReading();
}
```

This function returns the scope ordering of the quantified concept with the best plausibility measure, where those quantified concepts that do not involve scope ambiguities are considered the only plausible readings. To measure the plausibility of a reading, like the top-most *scope* function, this function also calls the *scopeTop* function. This function implements the algorithm described in figure 6 (chapter 3, p. 36). The function and related (private) functions are shown in figure 14 below. Some of the details of these functions have been omitted for clarity.

5.5 CONCLUDING REMARKS

In this chapter we discussed in some detail the program *Quasi*, which is a Java implementation of the quantifier scope inferencing strategy that we described in chapters 2 and 3. Our main goal in the implementation of *Quasi* was to demonstrate that a commonsense inferencing strategy for predicting the most plausible scope ordering is feasible. For this verification to be valid it was also important not to *hard code* any data in the process, but to build a generic *process* that, as we view it, depends only on individual preferences.

Beyond the generalization of the solution, we have also tried to apply some good software engineering principle in order to build an extensible program where it would be sufficiently easy to extend the grammar and the semantic analysis component. For example, with minimal effort *Quasi* should be extended to incorporate anaphora and pronoun resolution.

```
private double computePlausibilityMeasure(Vector range1, Vector range2)
{
    double pl = 0.0;
    int summationPart = 0;
    int sz1 = range1.size();
    int sz2 = range2.size();
    int intersectSz = (intersect(range1, range2)).size();
    int pr12 = precede(range1, range2);
    int pr21 = precede(range2, range1);

    for (int i = 1; i < intersectSz - 1; i++)
        summationPart = summationPart + sz2 * pr21 * i + 1;
    return ((pr12 * sz2) + summationPart * (1 / (double) (sz1 * sz2)));
}

private int precede(Vector range1, Vector range2)
{
    int range2Min = ((Integer)range2.elementAt(0)).intValue();
    int pr = 0;

    for (int i = 1; i < intersectSz - 1; i++)
    {
        int oneValue = ((Integer)range1.elementAt(i)).intValue();
        if (oneValue < range2Min) pr++;
    }
    return pr;
}
```

Figure 14. The implementation of the scoping functions.

Also, a question/answering capability on top of *Quasi* is already being added, in order to investigate quantification issues beyond scope ambiguities, such as generic quantification and quantification over large and potentially infinite domains. For this reason, and although

it was not technically necessary to implement a generic logical expression module in *Quasi*, the implementation was partly done for possible future extensions, where an SQL query can be generated to be executed against some database.

Regarding the various modules in *Quasi*, we should note that while there is little novelty in the syntactic and semantic analysis components, the pragmatic module (the scoping functions) required us to make some careful considerations in both the syntactic and semantic analysis components. A particularly challenging task was the incorporation of a (weak) compositional approach with delaying decisions regarding prepositional phrase attachments and quantifier scope until the processing of larger linguistic constructs.

6 Related Work

The work we described in this dissertation was motivated in part by the fact that while there seems to be ample evidence that quantifier scope is a phenomenon that must be treated at the pragmatic level, there has been little effort in that regard. Drawing on the logical and linguistic research, computational approaches to quantifier scope ambiguities have for the most part been inspired by syntactic and semantic treatments of the problem³³. In this chapter we will review the main computational approaches, starting with a brief review of the linguistic and logical models of quantification which have inspired the computational models.

6.1 QUANTIFICATION IN NATURAL LANGUAGE

The invention of variables and quantification is generally attributed to the genius of Frege, who, according to Dummett (1973), “resolved, for the first time in the whole history of logic, the problem which has foiled the most penetrating minds that had given their attention to the subject.” To appreciate the expressive power of quantifiers and variables, consider the task of expressing the mathematical fact that “every natural number multiplied by 2 is an even natural number.” This expresses a property of objects in an infinite domain, namely the domain of all natural numbers. Such properties are typically established using an axiomatic system (e.g., algebra and number theory), and mathematical induction.

Surprisingly, however, induction is a problematic phenomena since it requires us to have faith in a general principle stating that “regularities of the past will continue into the future” (Goertzel, 1993). In mathematical reasoning, such regularities are accepted since a mathematical object, such as “2”, for example, cannot be expected to exhibit a different behavior, or take on different properties in different contexts. In everyday commonsense reasoning, and in particular in language understanding, however, such regularities are not accepted so easily. Unlike properties attributed to mathematical objects, properties of *objects of thought* are context-dependent, and subject to revisions, exceptions, and other irregularities. In fact, whatever inductive generalizations we often make, must always involve some sort of “jumping to conclusions” (Flach, 1992). This problem, which has been

³³ The ‘problem’, for a computational linguist, is not quantifier scope ambiguity *per se*, but *resolving* the ambiguities in a computer program. As we stated in chapter 1, ambiguity itself is not a sin, but an ingenious invention that we have developed and dealt with quite effectively. It has nevertheless been an *obstacle* in building computer programs that can comprehend natural language text.

termed by McCarthy (1987) as the problem of “generality in artificial intelligence”, is central to the study of commonsense reasoning and natural language understanding.

This last point, concerning the relation of quantification to commonsense reasoning, is central to our treatment of quantification in natural language. While the need for variables and quantification is generally accepted, the role of commonsense knowledge in resolving quantifier scope ambiguities has largely been ignored. For the most part, computational models *that do consider* the modeling of quantification do so by using the models of formal logic where the “form”, and not the linguistic context, dictate the scope of quantifiers. While a number of significant results have been achieved, the problem of resolving quantifier scope ambiguity remained largely unresolved. In the remainder of this chapter we discuss the prevailing models of quantification, all of which have their roots in formal logic. The incorporation of these models in computational linguistics will also be discussed.

6.2 QUANTIFICATION IN FORMAL LOGIC

Virtually all computational models of quantification are based on some variation of Montague’s (1974) “proper treatment of quantification” (henceforth PTQ), and the theory of generalized quantifiers (Barwise and Cooper, 1981). In this section we briefly review the essential features of both approaches.

6.2.1 Quantification in PTQ

In PTQ³⁴, quantifiers are modeled as functions that take two properties and return a truth-value as a result, where a property is an object of type $(e \rightarrow t)$, i.e., a function that takes an entity and returns a truth-value³⁵. For example, the meaning of *every* is a function that takes two properties, P and Q , and returns true if whenever P is true of some entity so is Q , and false otherwise.

Montague’s main contribution was his ingenious use of the lambda calculus that allowed him to propose a uniform treatment for all noun phrases. The problem here was to explain

³⁴ Montague’s semantic program can be overwhelming and it would be inappropriate to review the program here. The reader can consult the excellent introduction to Montague semantics in (Dowty et al., 1981). The entire semantic program of Richard Montague is described in (Thomason, 1974). For a less philosophical exposition, the computational linguist could consult “Making computational sense of Montague’s intensional logic” (Hobbs and Rosenchein, 1978).

³⁵ Here we ignore the intensional version of PTQ where an object of type a has an intensional type $(s \rightarrow a)$, where s is a possible world (world-time pair) denoting a context of interpretation.

how noun phrases such as *John*, *every man* and *some man that talks* seem to function the same in the following:

$$(65) \left\{ \begin{array}{l} \textit{John} \\ \textit{Every man} \\ \textit{Some man that talks} \end{array} \right\} \textit{walks}$$

Seeking generality³⁶, Montague formulated a method by which *John* and *every man* could denote objects of the same type, and consequently, this required a method by which one could assign meanings to partial expressions, such as *every man*. As noted by Partee (1974), this expression can roughly be modeled in FOPL as $\forall x(\textit{Man}(x) \rightarrow \dots)$, which is not a well-formed formula (*wff*). Montague resolved this problem by utilizing the tools of the lambda calculus to generalize FOPL expressions. To see how this is done, consider the following:

$$(66) \textit{every man walks} \Rightarrow \forall x(\textit{Man}(x) \rightarrow \textit{Walk}(x))$$

$$(67) \textit{every man} \Rightarrow \lambda Q[\forall x(\textit{Man}(x) \rightarrow Q(x))]$$

$$(68) \textit{every} \Rightarrow \lambda P \lambda Q[\forall x(P(x) \rightarrow Q(x))]$$

In (66) we have a straightforward translation of *every man walks* into FOPL. Using lambda abstraction, a more general expression can be obtained by abstracting from the formula the property of “walking.” The result, given in (67), is a function, call it *every man*, that takes a property and returns true if the property is true of *every man* and false otherwise. Finally, a most general function can be obtained by another lambda abstraction over the property of “being a man” resulting in a function that takes two properties and returns a truth value, which is the meaning of *every*. A similar process is applied on sentences with definite noun phrases (NPs):

$$(69) \textit{John walks} \Rightarrow \textit{Walk}(\textit{john}')$$

$$(70) \textit{John} \Rightarrow \lambda P[P(\textit{john}')]]$$

In the above, English expressions are translated into expressions of intensional logic (IL), according to a well-defined set of rules. Note that *john'* is an expression of IL that rigidly designates a person, while *John* is an expression of English. What is more important to note here is that the meaning of *John* given in (70) and that of *every man* given in (67) have the same (semantic) type, $((e \rightarrow t) \rightarrow t)$, i.e., both are functions that take a property and return a

³⁶ Montague believed that “there is no theoretical difference between formal and natural languages.”

truth-value³⁷. Forming one semantic type for the syntactic category of noun phrases was the main goal of this formulation. The table below shows how partial meanings (functions) compose to form an expression of type t (i.e., a statement that is either true or false.)

Expression	Translation	semantic type	description
<i>every</i>	$\lambda P \lambda Q [\forall x (P(x) \rightarrow Q(x))]$	$(e \rightarrow t) \rightarrow ((e \rightarrow t) \rightarrow t)$	Function that takes a property and returns a function that takes a property and returns a truth-value.
<i>every man</i>	$\lambda Q [\forall x (\text{Man}(x) \rightarrow Q(x))]$	$(e \rightarrow t) \rightarrow t$	Function that takes a property and returns a truth-value.
<i>every man walks</i>	$\forall x (\text{Man}(x) \rightarrow \text{Walk}(x))$	t	Truth value

It is also important to note here that Montague's treatment of quantification is central to the entire semantic program in PTQ, since as can be seen from the lambda abstractions described above, the meaning of a quantifier in Montague's PTQ is a "template" for the interpretation of any English expression whatsoever. This also implies that the issue of quantifier scope is not directly addressed by this formulation since P and Q could in turn involve the use of other quantifiers. For example, in

(71) *Every young man loves a woman that is cheerful*

the property Q in this instance of 'every P is Q ' is the property of "loving a woman that is cheerful" which is different from the property of "loving every woman that is cheerful." A *direct* translation would therefore force a specific scope ordering that simply mirrors the syntactic structure, which is clearly inadequate, as discussed below.

6.2.2 Quantifier Scope in Montague's PTQ

An important aspect of Montague's system is his adherence to the rule-to-rule correspondence between syntax and semantics. For every syntactic rule there's a corresponding semantic rule that evaluates the meaning of the compound as a function of the meanings of the parts. In such a model, the semantic contribution of a certain constituent is the same in every context, which is the essence of compositionality in the Fregean tradition. Furthermore, in such a model a sentence is ambiguous *iff* it involves a lexical or structural ambiguity. Clearly, this presents a technical problem since this begs the question of how

³⁷ Montague's uniform treatment was not uncontroversial however (See Partee (1992)).

could we explain semantic ambiguities of structurally *and* lexically unambiguous sentences? For example, consider the sentence (72) that admits the two interpretations given in 72a and 72b:

(72) *Every man loves a woman*

72a. $\forall x[Man(x) \rightarrow \exists y[Woman(y) \wedge Love(x, y)]]$

72b. $\exists x[Woman(x) \wedge \forall y[Man(y) \rightarrow Love(y, x)]]$

Clearly, the truth conditions (meanings) of the two possible interpretations can be quite distinct. In order to account for this ambiguity, Montague devised a rule (the 'quantifying-in' rule) that admits multiple derivations trees for structurally unambiguous sentences. Accounting for scope ambiguity by admitting multiple derivations essentially amounts to admitting multiple logical forms, which is also the standard practice in computational linguistics (Hobbs and Shieber, 1987); Alshawi, 1990; Pereira and Pollack, 1991). Note that, in PTQ, *one* syntactic tree is still assumed (i.e., the sentence is *not* assumed to be structurally ambiguous), and the two derivation trees are obtained by applying different *semantic rules*. This is quite different from the quantifier raising mechanism (Reinhart, 1983; Fox, 1995; Reinhart, 1997), where the ambiguity is assumed to be structural. In a sense, then, PTQ treats scope ambiguities as semantic ambiguities, whereas in QR they are syntactic.

Applying the quantifying-in rule on the sentence in (72) results in the two derivations shown in figure 15 below. The derivation tree which mirrors the structure of the sentence (figure 15a) is called the *direct* reading, while that of figure 15b is called the *indirect* reading (Gamut, 1991) obtained by the rule of quantifying-in.

Figure 15. Two derivation trees corresponding to the $\forall\exists$ and $\exists\forall$ readings.

The interpretation of the derivation tree in figure 15a in PTQ would proceed as follows:

every man $\Rightarrow \lambda Q[\forall x[Man'(x) \rightarrow Q(x)]]$
a woman $\Rightarrow \lambda Q[\exists x[Woman'(x) \wedge Q(x)]]$
loves $\Rightarrow \lambda x \lambda y Love'(x, y)$
loves a woman $\Rightarrow \exists y[Woman'(y) \wedge Love'(x, y)]$
every man loves a woman $\Rightarrow \forall x[Man'(x) \rightarrow \exists y[Woman'(y) \wedge Love'(x, y)]]$

The interpretation of the derivation tree in figure 15*b* proceeds in the same manner, however, the noun phrase *every man* is quantified-inside the scope of *a woman* in the last step, yielding the following translation:

every man $\Rightarrow \lambda Q[\forall x[Man'(x) \rightarrow Q(x)]]$
a woman $\Rightarrow \lambda Q[\exists x[Woman'(x) \wedge Q(x)]]$
loves her_i $\Rightarrow \lambda y Love'(x_i, y)$
every man loves her_i $\Rightarrow \forall x[Man'(x) \rightarrow \lambda y Love'(x, y)]$
a woman every man loves her_i $\Rightarrow \exists y[Woman'(y) \wedge \forall x[Man'(x) \rightarrow Love'(x, y)]]$

While it is ingenious to 'invent' such a rule to salvage compositionality, as L.T.F. Gamut (1991) notes, Montague does not address the issue of *resolving* the ambiguity, i.e., of choosing between the candidate derivation trees. To be sure, this was never the goal of Montague's program, as has been noted by (Thomason, 1974; see the "Introduction") and (Dowty *et al.*, 1981). Montague was simply developing an algebra of meanings, whatever the nature of people's employment of these meanings. In computational linguistics, however, deciding between the two derivation trees of figure 15 (or, between the two equivalent logical forms) is a central issue, which was the focus of this dissertation.

In the next section we will briefly discuss the theory of generalized quantifiers and its relevance to issues of quantifier scope ambiguities.

6.2.3 The Theory of Generalized Quantifiers

The theory of generalized quantifiers (Barwise and Cooper, 1981; van Benthem, 1986) models a quantifier Q as binary relation between two properties A and B , much like the treatment of quantification in PTQ³⁸. In the theory of generalized quantifiers, however, the treatment is purely extensional, and thus a quantifier Q is modeled as a relation over two sets, A and B , where Q is a function of the cardinalities of the set A , called the *restriction set*,

³⁸ For a comparison between Montague's PTQ and the treatment of quantifiers using the theory of generalized quantifiers see (Cooper, 1987).

and the intersection set $A \cap B$. Using a set-theoretic model, an intuitive definition of the common quantifiers in natural language could therefore be easily defined as follows:

$$\text{every}(X, Y) =_{df} \begin{cases} \text{true}, & \text{if } X \subseteq Y \\ \text{false}, & \text{otherwise} \end{cases}$$

$$\text{some}(X, Y) =_{df} \begin{cases} \text{true}, & \text{if } (X \cap Y) \neq \emptyset \\ \text{false}, & \text{otherwise} \end{cases}$$

$$\text{no}(X, Y) =_{df} \begin{cases} \text{true}, & \text{if } (X \cap Y) = \emptyset \\ \text{false}, & \text{otherwise} \end{cases}$$

$$n(X, Y) =_{df} \begin{cases} \text{true}, & \text{if } |X \cap Y| = n \\ \text{false}, & \text{otherwise} \end{cases}$$

Therefore, *every man walks* is true if the set of men is contained in the set of things that walk. Similarly, *no man is running* is true if the set of those that are running contains no men. The appeal of the set-theoretic formulation is that one can easily define the meaning of general quantifiers in natural language. For example, assume that $n = |X|$ and $m = |X \cap Y|$, then,

$$\text{most}(X, Y) = \begin{cases} \text{true}, & \text{if } (m < 2n) \\ \text{false}, & \text{otherwise} \end{cases}$$

$$\text{at least three but less than five}(X, Y) = \begin{cases} \text{true}, & \text{if } (m \geq 3) \wedge (m < 5) \\ \text{false}, & \text{otherwise} \end{cases}$$

From a linguistic standpoint, the schema $Q(X, Y)$, where Q is a natural language quantifier and X and Y are the sets representing the extensional meanings of two term phrases, corresponds to

$$(73) \quad ({}_S ({}_{NP} ({}_{DET} Q) ({}_{TP} X)) ({}_{VP} Y))$$

Thus, it is assumed here that a syntactic analysis has already determined the set corresponding to the verb phrase Y , and that the sets X and Y are of the same domain (so that they could have an intersection set). Indirectly, therefore, this interpretation avoids the problem of quantifier scope ambiguity, by assuming it has been resolved. Below we will discuss this point further.

6.2.4 Formal Properties of Generalized Quantifiers

The main goal of developing a theory of generalized quantifiers was to classify the quantifiers of natural language into various classes according to a number of formal properties that they exhibit. By viewing quantifiers as binary relations over sets, one can use relational theory to derive some formal properties of the quantifiers.

Extensive work in this area has been done, most notably by (Barwise and Cooper, 1981) and (van Benthem, 1986). As a representative set of the various formal properties³⁹ consider the following where Q is a quantifier, and X , Y , and Z are sets:

Persistence (left increasing monotonicity) $Q(X, Y) \wedge X \subseteq Z \rightarrow Q(Z, Y)$

(holds for $Q = a, some, at least n, \dots$)

Right increasing monotonicity $Q(X, Y) \wedge Y \subseteq Z \rightarrow Q(X, Z)$

(holds for $Q = every, all, the, a, most, \dots$)

Left decreasing monotonicity $Q(X, Y) \wedge Z \subseteq X \rightarrow Q(Z, Y)$

(holds for $Q = every, no, the, at most n, \dots$)

Reflexivity $Q(X, X)$

(holds for $Q = the, all, every, both, \dots$)

Symmetry $Q(X, Y) \rightarrow Q(Y, X)$

(holds for $Q = a, no, some, exactly n, \dots$)

According to the symmetry property, for example, if *Some man from Detroit owns a 1965 Corvette* is true, then “the set of those who own a 1965 Corvette” must include *Some man from Detroit*. While these formal properties do not address the problem of quantifier scope ambiguity directly, they are important in that any proposal for a new model for quantification must ensure the validity of the above and other similar formal properties.

In proposing an inferencing procedure for resolving scope ambiguities we assumed the standard semantics of quantifiers, and thus these formal properties still hold. However, quantifiers at the implementational level could be modeled by procedures that are much more descriptive than the functions discussed in section 6.2.3 (page 89). By suggesting new procedures for quantifiers one must ensure that the formal properties discussed above are preserved. We discuss this issue in some detail in the next chapter.

³⁹ For detailed discussions on these and other formal properties see van Benthem (1986), Peres (1990), Barwise and Cooper (1981), and May (1989).

6.2.5 The Relevance of TGQ in Quantifier Scope Ambiguity

The relational model of quantification as suggested by the theory of generalized quantifiers does not directly address the problem of quantifier scope ambiguity. As was also pointed out by (van Benthem, 1986), in the schema $Q(X,Y)$, where Q is a quantifier, it is assumed that the two sets X and Y have been “identified” and that the result of quantification is the application of a set-theoretic operation corresponding to “one” quantifier, namely Q . Moreover, the two sets involved in the operation are assumed to be in the same domain. However, quantifier scope ambiguities arise when we express a relation between “two quantified” sets of different domains. That is, we encounter scope ambiguities in a schema $R(Q_1X, Q_2Y)$, where R is a binary relation, corresponding to the main verb in the sentence. Linguistically, this schema corresponds to

$$({}_S ({}_{NP} ({}_{DET} Q_1) ({}_{TP} X)) ({}_{VP} ({}_V R) ({}_{NP} ({}_{DET} Q_2) ({}_{TP} Y))))$$

This argument about “determining” the sets to be quantified suggests that there is a dual purpose for quantifiers in natural language. In fact Freeman and Stedmon (1986) make this point by arguing that quantifiers have two main functions: determining and quantifying. The first function is to determine “what is to be quantified” while the second function is to determine “how much is to be quantified.” It is in the first function that scope ambiguities are encountered. The dual purpose of quantifiers, while generally overlooked, is also consistent with our view that quantifiers operate at two different levels: (a) a semantic (or knowledge) level, where the issue is to determine “what” is to be quantified, and (b) an implementational level, where the issue is to determine “how much” is to be quantified. Scope ambiguities arise at the first (conceptual level), while at the implementational level the issues that must be dealt with are related to generic quantification, and quantifying over large and potentially infinite domains. In this work we have focused almost exclusively on scope ambiguities at the conceptual level. In chapter 7, some preliminary investigations of quantification at the implementational level and future work in that regard are briefly discussed.

6.2.6 How do Scope Ambiguities Arise in a Set-Theoretic Model?

As mentioned above, scope ambiguities in the relational model of TGQ are not addressed. The reason for this is that issues of quantifier scope ambiguity arise prior to obtaining the relational schema $Q(X,Y)$, where Q is a quantifier and X and Y are sets. To illustrate this point further, consider the following:

(74) *A child ate every cookie*

(75) *Every child ate a cookie*

In order to apply the TGQ model, we have to generate two overlapping sets, i.e., two sets in the same domain. The first set in both of these examples is the set of children. The second set, however, is “the set of children that ate every cookie” in (74), and “the set of children that ate at least one cookie” in (75). The relation that underlies the sentences in (74) and (75) can be expressed as follows:

$$R = \{ \langle x, y \rangle \mid \text{Child}(x) \wedge \text{Cookie}(y) \wedge \text{Ate}(x, y) \}$$

The relation R is therefore all pairs (x, y) of a child, x , that “ate” some cookie, y . Using R table 4 below illustrates the interpretation of (74) and (75) in a set-theoretic model, according to the theory of generalized quantifiers.

SENTENCE	SCHEMA	X (SET 1)	Y (SET 2)	VALUE
<i>A child ate every cookie</i>	$A(X, Y)$	<i>Child</i>	$\{x \mid ((x, y) \in R) \wedge (\{y \mid (x, y) \in R\} = \text{Cookie})\}$	$(X \cap Y) \neq \emptyset$
<i>Every child ate a cookie</i>	$Every(X, Y)$	<i>Child</i>	$\{x \mid (x, y) \in R\}$	$X \subseteq Y$

Table 4. A set-theoretic interpretation of quantifiers

In both cases, the first set in the $Q(X, Y)$ schema is the set of children. However, the second set is determined in a completely different manner. In the first sentence, the second set is the set of children that ate “every cookie” while in the second sentence the second set is the set of children that ate a (“at least one”) cookie. Note that this essentially means that we have committed ourselves to a particular scope ordering. The final interpretation, which is simply $(\text{child} \cap Y) \neq \emptyset$ and $\text{child} \subseteq Y$ respectively, is just an extensional set-theoretic operation. Essentially, what this analysis suggests is that the set-theoretic interpretation according to TGQ is assumed to operate **after** syntactic and semantic processing has determined what sets are involved in the quantification, i.e., after scope ambiguities have been resolved.

6.2.7 Branching Quantifiers in Natural Language

In addition to a number of formal properties that the theory of generalized quantifiers was concerned with, logicians have for some years been occupied by the notion of “branching quantifiers.” As we discussed in chapter 3 (section 3.4), the main issue regarding branching

quantifiers was to determine under what conditions quantifiers branch. As we showed in chapter 3, this question is a generalization of the scope ambiguity question, where an additional complexity is added due to the complicated functional dependencies that could occur when dealing with more than two quantifiers.

6.2.8 Quantifiers as Procedures

A final point on the theory of generalized quantifiers that is more relevant to some future work that we intend to pursue is the notion of “quantifiers as automata” that (van Benthem, 1986) has proposed. van Benthem presents an analysis of quantifiers individually viewed as an automaton that takes two sets as inputs and returns a truth value as a result. The complexity of each automaton is clearly a function of the complexity of the quantifier. For example, the following could be an example of an automaton corresponding to the quantifier *all* applied on sets X and Y :

Figure 16. An automaton corresponding to the quantifier *all*.

In Figure 16, the state S is considered to be a final state, w is considered an element of $X \cap Y$, y is an element of Y and x' is an element of $X - Y$, i.e., an element that is in X but not in Y . The automaton is fed elements from X , and must end in the final state S if $All(X, Y)$ is true. Clearly, the automaton will end in the final state *iff* every element it processes from the set X is an element of Y , which is the intended set-theoretic interpretation of $All(X, Y)$.

Other quantifiers, for example *most*, being a context-dependent quantifier, would require a more complex state machine than that of *every*. Computationally, *most* (and similar fuzzy, or context-dependent quantifiers, such as *few*, *several*, *many*, etc.) requires a store (a state variable), since in computing $most(X, Y)$ the relative ratio of $|X \cap Y|$ to $|Y|$ must be stored somewhere for later processing. A finite-state-machine would not suffice for the quantifier *most*, although a push-down automaton (e.g., a Post machine or, of course, a Turing machine), would.

The issue of the *computational* effectiveness of verifying quantification over large sets is very crucial here. That is, this discussion of extensional verification brings to the forefront questions concerning limitations on working memory and time constraints. This subject, which we plan to investigate further in the future, will be briefly discussed in the next chapter. For the moment, however, there are two important points regarding van Benthem's notion of *quantifiers as procedures* that must be noted:

1. While we agree with the notion of *quantifiers as procedures*, we suggest that quantifiers do not simply operate on extensional sets, but are also dependent on contextual and computational factors.
2. The view of quantifiers as procedures inevitably brings to the forefront the "verificationist" objection to Frege's notion of quantifiers⁴⁰, since the standard model cannot explain quantification over large and potentially infinite domains⁴¹.

The first observation has to do with our view that resolving quantifier scope ambiguities is an inferencing process that is dependent on commonsense background knowledge.

The second point is intended to address the issue of cognitive plausibility. That is, how is it that we tend to quantify over large and potentially infinite domains within limited time and memory constraints. Clearly, the purely extensional view suggested by the automaton in figure 16 does not explain this adequately.

6.3 QUANTIFIER SCOPE AMBIGUITY IN COMPUTATIONAL LINGUISTICS

Having summarized the theoretical foundations of quantification in the previous sections we now turn to the treatment of quantifier scope in computational linguistics⁴².

⁴⁰ See (Dummett, 1973, pp. 512-541) for an interesting discussion on this subject.

⁴¹ Recall that in logical semantics $\forall xP(x)$ is modeled as $P(x_1) \wedge P(x_2) \dots \wedge P(x_n)$, for all x_i in X . The "verificationist" objection questions how an exhaustive (and thus computationally effective) verification of a property can be performed on a large (or even potentially infinite) domain. This is also related to the general problem of "generality in AI" raised by McCarthy (1987) and later addressed by theories of non-monotonic reasoning (See Thanassas, 1992).

⁴² Readers versed in the philosophical and logical literature on quantification will very clearly see this brief review as, at best, incomplete. A large body of work on quantification as it relates to intensionality and opaque contexts (*de re* and *de dicto* readings), most notably by Carnap, W. V. Quine, and subsequently by Richard Montague, David Kaplan, and Barbara Partee, among others, have been ignored. Our concern here was limited to quantifier scope ambiguities in **extensional** contexts.

6.3.1 Computational Models of Quantifier Scope Ambiguity

Virtually all natural language systems that give an adequate treatment to quantification base their model of quantification on the models of formal logic, and consequently all computational models are based on some variation of Montague's PTQ or the theory of generalized quantifiers (TGQ). The issue of resolving quantifier scope ambiguities, which as we discussed above was not directly addressed in Montague's PTQ or in the theory of generalized quantifiers, is, however, a practical problem that requires an *efficient* solution in computational models.

Quantifier scope ambiguity presents a challenge to computational models that are based on compositional semantics. First, semantically, the problem translates in computational terms into obtaining multiple logical forms for the same lexically unambiguous syntactic structure. Second, quantifier scope ambiguity also presents a challenge at the pragmatic level, in deciding on the most plausible scope ordering (i.e., on the most plausible logical form) from a multitude (typically $n!$) of semantically valid logical forms. As argued in (Grosz *et al.* 1987), computationally, this can be an intractable problem.

In computational linguistics, a number of techniques have been suggested. Typically, a sentence is first translated into a scope-neutral (Allen, 1987) or a *quasi* LF (Alshawhi, 1990) with no scope ordering commitments. A number of syntactic and semantic rules are then *incrementally* applied to reduce the number of possible readings. Computational models for incremental interpretation must be carefully crafted if one is to stay committed to compositionality. Elaborate attempts at such models are given in (Pereira and Pollack, 1991) and (Harper, 1992). These models do not directly address the problem of scope ambiguity however. Instead, they provide a *computational framework* for the resolution of quantifier scope (and other) ambiguities.

Regarding the resolution of quantifier scope ambiguity, several algorithms that apply a number of syntactically-motivated “preference rules” have been suggested. Instead of inferring a most plausible reading, these methods suggest a combinatorial approach where less likely readings are incrementally disallowed.

Most of this work is described in (Allen, 1987; Hobbs & Shieber, 1978; Moran, 1988; Moran and Pereira, 1992; and Park, 1995). In summary, computational linguistics work that

attempts to address the problem of quantifier scope ambiguity can be grouped into two general categories:

1. Computational models to support a compositional and model-theoretic interpretation of *underspecified* (or "quasi", or "scope neutral") logical forms⁴³.
2. Algorithms for resolving quantifier scope ambiguities.

The work in the first category is not directly concerned with describing general procedures for resolving scope ambiguities. Instead, the goal of the work is to define a computational framework that would support an incremental (Pereira and Pollack, 1991) (sometimes referred to as *conditional* (Pollack and Pereira, 1988)) interpretation process. While such models are important, and indeed we have adopted such techniques in our implementation, it is the work in the second category that is the one most relevant to our proposal of an inferential strategy for resolving scope ambiguities.

In the following sections, we briefly review models of incremental interpretation of underspecified (i.e., not fully committed logical forms), as well as the proposed algorithms for resolving quantifier scope ambiguities.

6.3.2 Underspecification and Incremental Interpretation

The need to devise a conditional (or incremental) interpretation process on an underspecified logical form is due to the fact that a fully unambiguous logical form cannot in practice be attained until larger linguistic fragments have been processed, and until various inferences that use some background knowledge have been performed to resolve various types of ambiguities, including pronoun resolution and quantifier scope.

Typically such inferences require access to background knowledge and discourse information of larger linguistic fragments. In such models, the strategy is to isolate the syntactic and semantic aspects of the analysis from the pragmatic and inferential aspects of language understanding, where contextual world knowledge might be brought into the analysis at some point.

⁴³ The debate between model-theoretic semanticists and transformational grammarians regarding the role of logical form is an important one. Logical Form is a representation that is one step beyond the surface structure, and yet not quite a meaning. Since most interpretation issues appear at the level of logical form, the debate has always been *what is in the realm of syntax, and what is considered semantics?* This also relates to the whole question concerning the *autonomy of syntax* is clearly beyond the scope of this dissertation.

Allen's scope-neutral logical form (1987) and Cooper storage (1983) were some early suggestions on how to delay commitments regarding quantifier scope until further information was available. In recent years, more sophisticated techniques have been suggested, such as quasi logical forms (QLFs) (Alshawhi and van Ejck, 1989; Alshawhi, 1990), and Underspecified Discourse Representation Structures (UDRSs) (Reyle, 1995). These ideas were initially popularized by work on the SRI Core Language Engine (CLE) (Moran, 1988; Pollack and Pereira, 1988; Pereira, 1989; Pereira and Pollack, 1991; and Moran and Pereira, 1992). The basic idea behind these techniques can be illustrated by the following example:

(76) *Every pilot controls a jet*

The standard notation for representing quantified term phrases in computational linguistics models that are based on formal logic is as follows:

(77) **Quant**(*variable, quantifier, restriction*)

where *variable* is the variable of that binds the quantifier *quantifier*, and *restriction* is a sort (type) restriction over which the variable can range. Using this notation, the logical expression representing the quantified term phrase *every man* is the following:

Quant(*x, every, Man*)

Note that in our implementation we have used a similar notation using the semantic object *LogicalExpression* to represent quantified expressions and variable binding.

A scope-neutral logical form representing the sentence in (76) can now be defined as follows:

(*Control*, {**Quant**(*x, every, Pilot*), **Quant**(*y, a, Jet*)}, ϕ)

where the first set is a non-ordered set of quantified term phrases (no scope-commitments), and ϕ is a set of assumptions and conditions that are discharged by a pragmatics module when resolving ambiguities, including resolving the scope orderings (the assumptions/conditions might contain a reference to some entity introduced in the discourse, which is then used to resolve an anaphor.) The final disambiguated logical form must have an empty set of discharged (i.e., applied or processed) conditions, and an empty set of yet-to-

be-ordered quantified term phrases. Thus the following are two possible disambiguated logical forms corresponding to the $\exists\forall$ and $\forall\exists$ readings of (76):

$$\begin{aligned} & \{Control, \{Quant(x, \text{every}, Pilot), Quant(y, a, Jet)\}, \{ \} \} \\ & \{Control, \{Quant(y, a, Jet), Quant(x, \text{every}, Pilot)\}, \{ \} \} \end{aligned}$$

Different conditions/assumptions are discharged by different sets of rules. Thus, there are rules to resolve quantifier scope ambiguities, reference resolution, etc. These rules typically utilize sort (type) information and might have access to semantic information of the main verb, as well as discourse information. The application of these rules might also update the global context (in cases where new references might be introduced, for example).

All the proposals for an initial (underspecified) logical form assume that each utterance is completely specified and disambiguated (i.e., its assumptions set has been completely discharged) before the next utterance is considered (Kameyama, 1996). Thus these models do not allow for large linguistic fragments to be considered, and while the previous discourse can be considered, subsequent discourse, which might be required (Grosz *et al.*, 1987), is not considered.

Although in this dissertation we did not specifically suggest an overall defeasible inferencing strategy, we have suggested a scoping algorithm that can be readily incorporated into such a strategy. By numerically ranking the plausibility of all possible scope orderings, additional information could be used to retract previous inferences in two ways: (i) by iterating, in decreasing order of the measure of plausibility, through the possible readings and selecting the first scope ordering that is consistent with the discourse information; or (ii) by incorporating the discourse information into the computation of the measure of plausibility.

The development of such an overall strategy is beyond the scope of this work, although, as we discuss in the next chapter, we plan to investigate this issue further in the future.

6.3.3 Rules and Preferences for Resolving Scope Ambiguities

The computational work has, for the most part, been rule-based, where the rules incorporated have been syntactically-motivated. While most of these approaches generally share the same strategy, in the following sub-sections we will review the main proposals individually.

Allen

Allen (1987) suggests that "while not all scoping can be determined independently of context, certain sentences have strong preferences toward a particular scoping." These 'preferences' are based on some linguistic data, and are of course subjective, which is our main objection to this approach to quantifier scope. As an example of a simple rule of preferences, the following is the order of quantifiers that seem to take wider scope:

the > each > wh-terms > every, all, some, a

So, for example, based on the above observation, a simple rule in a quantifier scope resolution algorithm would be "give all definite NPs the widest scope." Thus, in a sentence such as "*The man saw every dog*" it is very likely that we are always speaking of a particular man (that saw every dog.)

The general preference rules that are suggested by Allen however are more controversial when quantifiers in prepositional attachments and relative clauses are considered. The final algorithm suggested has preference rules that are applied in a specific order. Since multiple rules might match a certain linguistic construction, the order for applying the rules becomes crucial. Furthermore, the algorithm suggests that if a subsequent rule contradicts a previous rule, the scoping determined by the first rule applied is maintained. While most of the rules suggested work in some cases, there is almost always an example where these rules seem to predict a reading that is not the one most readers would consider. For example, consider the following:

- (78) *John knows several programmers that work for a text retrieval company in Ottawa*
- (79) *John knows several programmers that started a text retrieval company in New York*
- (80) *John appointed a manager for several programmers on the RAD project*
- (81) *The committee included every director of an AI research group*
- (82) *The panel invited every columnist in a major newspaper*
- (83) *John visited a restaurant on a main street in every city*
- (84) *A union representative consulted with each executive at General Motors*
- (85) *John knows three engineers who started two text retrieval companies that were bought by a US company*

These examples were chosen to show how most rules, and in fact several at once, might be broken if the linguistic context is not considered. For example, one preference 'rule' that is strongly suggested is that a quantifier in a relative clause must depend on (i.e., must have narrow-scope relative to) the quantifier of its head noun. However, the examples in (78) and (79) seem to suggest that this preference does not always capture our intuitions. Most

readers, based on their knowledge of programmers and software companies would choose a reading for (78) and (79) where one text retrieval company was implied (i.e., a reading where *a* outscopes the quantifier of its head noun.)

Similarly, (80), (81) and (82) suggest that quantifiers in a PP do not always outscope the quantifier of the NP they modify. We argue that the most likely readings for these examples imply, *several managers*, *several AI research groups*, and *several major newspapers*, respectively. (83) demonstrates that a quantifier *can be* raised two levels (have wide scope over two preceding quantifiers), as most readers would not accept a reading in (83) that implies there is some restaurant on some main street which (itself) is in every city, and thus *every* would have the widest scope. (84) shows that *each* does not necessarily outscope other quantifiers, and (85) shows that several rules can fail at once: a quantifier can be raised out of more than one clause, and a quantifier in a relative clause can outscope the quantifier of its head NP.

The reason for pointing out how some of these suggested preference rules (that are syntactically-motivated) fail in some cases is to underline our main thesis that purely linguistic considerations do not always capture some pragmatic and inferential aspects of quantifier scope.

Hobbs and Shieber

In their algorithm for generating quantifier scopings (Hobbs and Shieber, 1987) were not concerned with determining a specific scope ordering *per se*, but in "suppressing readings that are spurious for purely structural reasons, that is, for reasons that follow from the general relationship between the structure of the sentences and the structure of their logical forms and independent of the meanings of the particular sentences." In other words, their algorithm does not *select* a preferred scope ordering, but eliminates what they term *invalid* readings. They specifically mention that they "are not concerned with infelicity of certain readings due to lexical semantic or world knowledge." In addition to eliminating some readings based on grammatical rules, some readings are also eliminated based on the resulting translation into logical form (LF). For example, in

(86) *Most students talked to a friend of every professor about his work*

every professor must outscope *a friend* since otherwise the pronoun *his* would not be bound in the logical form. Thus a certain reading is not allowed based on what is known as the

unbound variable constraint (UVC). According to the algorithm of (Hobbs and Shieber, 1987), therefore, the following sentence does not have all 120 possible readings (which is $5!$, where 5 is the number of quantifiers), but 42 "valid" readings:

(87) *Some representative of every department in most companies saw a few samples of each product*

As we argued in chapter 1, our main objection to this incremental elimination of invalid readings is that this combinatorial approach is neither cognitively, nor computationally plausible. Moreover, we cannot escape the need for an inferencing module that utilizes commonsense knowledge since the number of remaining readings that are syntactically and logically "valid" is still not useful, particularly in the context of a query to a database. As we suggested throughout this thesis, a preferred strategy is one that directly infers, based on some background knowledge, the most plausible reading.

Moran and Park

The approaches of Moran (1988) and Park (1995) are based on a combination of a general set of preference rules (much like the algorithm proposed by Allen (1987)), as well as an algorithm that eliminates syntactically and logically invalid readings (a la Hobbs and Shieber (1987)).

In Moran's proposal several additional criteria are applied, such as the *conversational maxim of quantity*, which is used to suggest readings that carry the maximum amount of information. In chapter 3 we discussed this strategy and suggested what are the factors that might motivate this preference, namely the modality of a relation. Moreover, in our proposal no special rule had to be added in order to account for this preference. That is, the approach that we suggested seems to capture this preference, as well as several others, in a very simple, generic and computationally effective formula.

In Park (1995) there is an interesting addition to most of the proposals suggested in the literature, namely the consideration of the functional dependency that is implied by the underlying relation. While Park's observation is sound, there is no suggestion on how this semantic dependency can be used *systematically* in an inferencing process. Much of what we suggested in our proposal is based on exploiting the semantics of the underlying relations through the use of quantificational constraints in the inferencing procedure suggested in chapter 3.

Fox

While most comprehensive treatments of quantification and quantifier scope ambiguity have been made within the framework of logical semantics, a number of studies in the tradition of transformational grammar have provided some interesting insights.

Fox (1995) suggests that the two scope shifting operations (SSO) in transformational grammar, namely quantifier raising (QR) and quantifier lowering (QL) obey what he termed the "economy" principle. According to this principle, a quantifier is not raised (or lowered) from a constituent if the movement does not produce a logical form that is not already implied by the candidacy set that has already been determined.

The essence of what Fox suggests is a way of eliminating redundant readings. While his rules are not designed to eliminate (logically) invalid readings, as is done in the algorithm of (Hobbs and Shieber, 1987), these two combinations can be used to reduce the number of possible readings considerably.

Here we should reiterate what is our view of the relation between the syntactic, semantic, and pragmatic approaches to scope ambiguities. As we argued above, the linguistic techniques of eliminating readings based on economy principles and the restrictions of some movements, as well as the logical/semantic approaches that eliminate logically invalid readings, will not always guarantee the selection of a single preferred reading, and a pragmatic module that infers the most plausible reading must thus be the last resort.

May

While the work of (May, 1989) is not presented in a computational context, but rather within the general framework of logical form in transformational grammars, several important insights can be readily incorporated into an algorithm for resolving quantifier scope ambiguities. In particular, May (1989) has suggested some important conditions on logical forms that must be satisfied to correctly predict all the valid collective and distributive readings in sentences involving *resumptive* quantifiers. The problem is one of predicting all the valid situations in which the scope ordering suggesting a distributive reading of (88) is selected (May, 1989):

(88) *Two detectives solved two crimes*

Based on the possible scopal interpretations (88) has two interpretations that suggest a total of four crimes that have been solved, two by each detective (the direct reading), or two

crimes that have been solved (the indirect reading.) In interpreting (88) however we must insure that we do not accept readings that do not conform with the intuitive interpretation, such as the situations depicted graphically below:

While in both (a) and (b) we have a total of 2 detectives that (collectively) solved a total of 2 crimes, both of these situations do not satisfy (88), since we require two detectives to be involved in the solution of (at least) two crimes.

The problem gets even more complicated when the semantics of the underlying relations impose further constraints on all the valid distributive readings. For example, while we can allow the distributive reading in (89), the distributive reading in (90) should be blocked since 2 houses can not (in any way) be spread over three streets (given our common notions of houses and streets):

(89) *John visited 3 houses on 2 streets*

(90) *John visited 2 houses on 3 streets*

May (1989) stops short of involving the semantics of the underlying relation in the defining the conditions under which the collective and distributive readings can occur.

Sher

Sher (1990) is perhaps the most relevant investigation to our work. Although the focus of the investigation was primarily quantifier branching, the conclusions drawn are that a proper formulation of quantification in natural language might have to involve looking at the semantics of the underlying binary relations that "glue" together quantified term phrases. Specifically, Sher (1990) looks at the possibility of investigating various cardinality conditions on the relation, such as *one-to-many*, *each-to-all*, etc., and attempts to suggest under what conditions do quantifiers seem to branch.

While the general thrust of Sher (1990) is in line with what we propose in this work, we have precisely defined all the ways in which cardinality conditions (which we termed

quantificational constraints) effect quantifier scope for simple as well as more complex functional dependencies, including quantifier branching.

Moran and Pereira

The work of (Moran and Pereira, 1992) was the outgrowth of several years of research into the problem of quantifier scope and related ambiguities in logical form (Pereira and Pollack, 1991; Pollack and Pereira, 1988; Moran, 1988).

The scoping algorithm presented in (Moran and Pereira, 1992) is a collection of rules and 'preferences' that have been compiled from the literature, with a strong influence from (Allen, 1987) and (Hobbs and Shieber, 1987). While the authors do suggest the need for an inferencing module that could draw on domain knowledge in resolving scope ambiguities, the details of this proposal have been left for future work. Thus, neither the nature of the domain knowledge that would be required, nor the details of the inferencing process itself have been discussed. Since this was precisely our focus in this dissertation, it is our view that the main contribution we have made is to suggest for the first time what the nature of a commonsense inferencing strategy for resolving scope ambiguities might be.

TEAM

The treatment of quantifier scope in TEAM (Grosz *et al.*, 1987) is also similar to the proposals of (Moran and Pereira, 1992). In TEAM all $n!$ valid readings are initially considered, and multiple *pragmatic* paths over the candidate logical forms are then assumed to eliminate unlikely readings.

Possible scope orderings are eliminated in TEAM by "quantifier specialists", which are essentially special heuristic rules responsible for enforcing syntactic, preference, and some pragmatic rules. These rules have the same general flavor of those suggested by (Allen, 1987; Hobbs and Shieber, 1987; and Moran, 1988), namely there are rules that eliminate scope orderings that result in readings subsumed by others, readings that result in an unbound variable, etc. A set of preference rules that uses a pre-defined order of *strong* quantifiers is also used (*each*, for example, is taken to be stronger than *every*, etc.). A combination of (a) the surface structure information, (b) an order of quantifiers with numeric strength assigned to each, with (c) a number of syntactic and preference rules are all used in tandem to decide on the most plausible scope ordering.

As the authors correctly note however, the procedure for determining scope orderings still fails to predict the most plausible reading in many cases since the pragmatic (inferencing) procedure is rather weak. For example, in the following

- (91) *Who is the commander of every ship*
- (92) *Who is the commander of every soldier*

the difference is in the *CommanderOf* relation between a person and a ship, and a person and soldiers. While there is a different commander for every ship, there is typically one commander for a large number of soldiers. It is precisely this type of pragmatic and background knowledge that we exploit in our proposal for an inferencing procedure.

6.4 SUMMARY

Our brief review of the major theoretical and computational approaches to modeling quantifier scoping and its ambiguities in natural suggests that several aspects of this problem remains largely unresolved. In our view, this is due in part to the following:

While there is ample evidence that some quantifier scope information must be encoded at the pragmatic level, the main approaches to resolving quantifier scope ambiguities has consisted in the use of syntactically motivated preference rules, and semantic/logical validity. While these rules do help in eliminating a number of readings, this general approach to scope ambiguity suffers from the following problems:

- 1) The preference rules suggested do not always capture our intuition of what the preferred reading is.
- 2) The incremental approach of eliminating invalid or redundant readings is neither computationally nor cognitively plausible, since this approach can lead to a combinatorial explosion of possible (i.e., of syntactically and semantically valid) scope orderings that must be considered. Moreover, this approach does not guarantee the selection of a preferred reading.
- 3) A number of issues related to quantification have been addressed in isolation, and specific rules to handle each situation have been suggested. Besides

violating the Occam's Razor principle (the simplest solution should be preferred), this may lead into an unmanageably large and potentially conflicting set of rules.

In this dissertation, we have suggested a very simple and generic procedure that seems to address these concerns. Amongst these, perhaps the most serious claim that we have made here is to suggest that our approach does provide an adequate answer to the concern raised in 1. Regarding this point, we argue that since our approach is essentially reader-based, the scope ordering suggested by our algorithm should always reflect the reader's intuition. In fact, the algorithm we suggested is entirely dependent on individual biases as opposed to an *ad hoc* set of rules that may or may not reflect an individual preference.

7 Summary and Future Work

Freeman and Stedmon (1986) suggested that quantification might be "the topic which holds the most immediate promise for working at where the three disciplines (logic, philosophy, and psychology) may interpenetrate." Semanticists and logicians have indeed given much attention to quantification, and various proposals have been made to explain the scope ambiguity phenomenon. The pragmatic aspect of this problem, however, has not received much attention. Since such pragmatic considerations are essential in a computational system, this was almost exclusively our focus in this investigation. In this chapter we summarize the arguments made in the previous chapters and discuss some future work.

7.1 THE PRAGMATICS OF QUANTIFIER SCOPE

Our conviction that scope ambiguity is a phenomenon that must, at least in part, and as a last resort, be treated at the pragmatic level has been guided by two factors. First, we agree with Givon's (1984) observation that natural language expressions are ambiguous as a result of our ingenious method of data compression for storage and performance effectiveness. Based on this observation, it would seem that speakers of natural language rely on the hearer's (reader's) intuitions and commonsense knowledge in 'working out the details' that were omitted. Until it can be demonstrated that such knowledge is encoded at the syntactic and/or semantic levels, an inferential and commonsense reasoning approach to resolving scope ambiguities (among others, such as anaphor), must be developed.

The second observation that has inspired our approach to quantifier scope ambiguity is based on a comparison with other types of ambiguities in natural language. Sentences involving lexical, structural, anaphoric and prepositional attachment ambiguities are not inherently ambiguous in the sense that scope ambiguities are. Given an appropriately defined context, readers seem to be able to resolve all these types of ambiguities in the same manner; i.e., readers do not seem to have **individual preferences** as to how lexical, structural or anaphoric ambiguities should be resolved.

We argue that the reason for this is that the disambiguation process in these situations does not depend on any individual parameters! This, as we have argued throughout this dissertation, does not seem to be the case when resolving scope ambiguities.

To further argue this point, consider the following examples involving lexical, structural, and anaphoric ambiguities:

- a. *John likes to* $\left\{ \begin{array}{l} \text{play} \\ \text{drink} \end{array} \right\}$ *gin*
- b. $\left\{ \begin{array}{l} \text{John saw a man with a telescope} \\ \text{John drinks tea with lemon} \end{array} \right\}$
- c. *John saw a man robbing a supermarket.*
He immediately called the police.

In (a-c) it seems that an appropriate context that would resolve the ambiguity is, respectively, (i) common knowledge that we *play* a game (and not a drink); (ii) a picture shedding some light on the actual state of affairs; and (iii) common knowledge that a man robbing a supermarket does not call the police, but a witness might very well do so. In neither case does the required context involve any individual biases, which might lead to individual preferences in how the ambiguity should be resolved.

As we have argued above, however, a fully specified context for resolving scope ambiguities does include an individual bias (a quantificational constraint, reflecting an individual belief), which is the reason why different individuals seem to have different preferences as to which is the most plausible quantifier scope ordering.

What this suggests, therefore, is that ultimately we need to develop formalisms that can go beyond testing the syntactic and semantic validity of a scope ordering. Since it is neither cognitively nor computationally plausible that we consider all **valid scope orderings**, an inferential method that could decide on the most **plausible scope ordering** must be developed, where plausibility is a function of the beliefs of some individual. This, undoubtedly, takes the scope ambiguity phenomenon outside the realm of syntax and semantics into the realm of pragmatics and commonsense reasoning.

Developing inferencing strategies that utilize commonsense world knowledge has generally been considered problematic for two important reasons. First, unlike syntax and semantics, pragmatics does not deal with validity, but with plausibility, which can be undecidable. Second, and equally important, a pragmatic module that performs inferences utilizing commonsense world knowledge might require the storage of, and reasoning with, a massive amount of background knowledge.

Regarding the first concern, we argue that human reasoning *is* undecidable, in the formal sense. As noted by (Shastri and Ajjanagadde, 1993), much of language understanding is inferencing, and in our view, we do not employ formal deductive reasoning only in this process, but a combination of deductive, abductive (analogical), inductive, as well as probabilistic (uncertainty) reasoning. What sound and complete models of reasoning can we employ to model these complex processes? Results from mathematical logic (see Enderton (1972)) have established that once we step outside first-order logic (even into second-order modal logic) we can not assume a sound and complete model of reasoning. Therefore, unless it can be demonstrated that first-order logic is an adequate model for human reasoning⁴⁴, to successfully build systems that can interpret natural language text, we must travel along that (undecidability) road.

This is not to say that we should not be able to formally describe whatever process we conceive of. Everything that is ultimately executed on a digital machine can be formally described. What we *are* saying however is that the (logical) tools that we currently have at our disposal to formally describe the complex reasoning that language understanding requires are inadequate.

In that sense, we agree with Barwise's (1989) observation that the time has come to enrich our logical formalisms to cope with the enormous problems that we face in the study of language, intelligence and the mind. In particular, we believe that the logical formalisms that are currently available must be complemented by a number of algorithms and processes that could be used to model a number of complex linguistic phenomena. In this context, we believe the work presented here was a step in that direction.

It is therefore in that spirit (of enriching the currently available logical formalisms) that we conceived of and developed the work presented here. Moreover, it is our conviction that this work must continue, as numerous problems in the study of language await us. There is little doubt that in order to tackle these problems, some novel extensions have to be made to the set of logical tools that are currently available.

Regarding the second point, namely the problems associated with storing and reasoning with a vast amount of background knowledge, we have tried to keep the amount of background

⁴⁴ There are numerous contributions that argue against the adequacy of first-order logic as a model for human reasoning. See, for example, Corriveau (1995, pp. 34-35), Lakoff (1987), Chermiak (1992) and Fauconnier (1994).

knowledge that is required in the scope resolution process to a minimum. In fact, the only major ontological commitment that we make is to assume that a quantificational constraint is defined on every relation between two concepts. Moreover, and since the entire process has been reduced to a simple numerical algorithm, the 'reasoning' process that emerged is computationally very effective.

Regardless of its simplicity, however, we believe that the method we described in this dissertation suggests a novel and computationally very effective procedure for predicting the most plausible scope ordering. Beyond suggesting a possible solution to this urgent and very practical problem, the method we propose raises a number of interesting questions concerning the relationship between commonsense reasoning and language understanding, which, undoubtedly, will take 'some' time to investigate.

7.2 FUTURE WORK

Before we proceed in discussing a number of proposals that we plan to investigate in the future, we must note that most of the arguments we make in this section are, at best, highly speculative.

There are two major extensions envisioned to the work presented here. First, the method we proposed in this dissertation for resolving quantifier scope ambiguity is focused primarily at the conceptual level (or in Newell's terminology, the knowledge level). That is, we have considered quantifier scope at the logical form level, without any consideration to the processing of quantifiers at the processing (or implementational) level. However, at the implementational level, quantifiers are essentially procedures (van Benthem, 1986) that operate on sets.

At this level of processing, there are a number of issues to consider, such as (i) verifying the truth of a proposition that might involve quantifying over large and (potentially) infinite domains; and (ii) the interaction between quantification and reference resolution and the associated memory search and recall operations.

This overall approach to quantification at both the conceptual and implementational levels is what started the investigation we described here. In (Saba and Corriveau, 1995), we discussed some of the issues that must be addressed at both levels, and subsequent work, which resulted in this dissertation, focused almost exclusively at the conceptual (logical

form) level. In the remainder of this section, we will discuss some of the ideas that we started investigating and that we hope to develop further.

7.2.1 The Role of Time and Memory Constraints

From the semantic point of view, the meaning of *every*(P, Q) is $\forall x(P(x) \rightarrow Q(x))$. Computationally, however, the meaning of *every* is a procedure that attempts to verify that every element of $\mathbf{P}$ is also an element of $\mathbf{Q}$, where $\mathbf{P}$ and $\mathbf{Q}$ are the two sets that correspond to the predicates P and Q , respectively.

We refer to this procedure as the meaning of *every* at the implementational level. Furthermore, we argue that in addition to the linguistic context and background knowledge, the meaning of quantifiers at the implementational level is also dependent on time and memory constraints. For example, consider the following:

- (93) *Cubans prefer rum over vodka.*
- (94) *Students in CS404 work in groups.*

Our intuitive reading of these sentences suggests that in (93) we have an implicit *most*, while in (94) it is more reasonable to assume an implicit *all*. We argue that such inferences are dependent on time constraints and constraints on working memory. That is, since the set of students in CS404 is a much smaller set than the set of *Cubans*, it is conceivable that we are able to perform an exhaustive search over the set of all students in CS404 to verify (94) within some time and memory constraints. In (93), however, we are most likely performing a *generalization* based on few examples that are currently activated in short-term memory (STM).

What this suggests, therefore, is that there must be some relation between generic quantification and commonsense reasoning on the one hand, and quantifying over large (or infinite) domains. In fact, our suggestion that time and memory constraints play an important role in defining the boundaries of what is being quantified is based on our view of properties and their negation.

We suggest that there are three ways to conceive of properties and of their negation, as shown in table 5 below.

P	$\neg P$	P	$\neg P$	P	$\neg P$
T	F	T	F	T	F
F	T	F	T	F	T
$-$	$-$	$-$	T	$-$	F
(a)		(b)		(c)	

Table 5. Three models of negation.

In (a), we take the view that if we have no information regarding $P(x)$, then, we cannot decide whether $\neg P(x)$ is true or false. In (b), we take the view that if P cannot be confirmed of some entity x , then $P(x)$ is assumed to be false⁴⁵. In (c), however, we take the view that if there is no evidence to negate $P(x)$, then assume $P(x)$. Note that model (c) essentially allows one to “generalize”, given no evidence to the contrary - or, given an overwhelming positive evidence. Of course, formally speaking, we are interested in defining the exact circumstances under which models (a) through (c) might be appropriate.

We believe that the three models are used, depending on the context, time, and memory constraints. In model (c), we believe that assigning a truth value for a certain property $P(x)$ is a function of the following:

$np(P,x)$ number of positive instances satisfying $P(x)$
 $nn(P,x)$ number of negative instances satisfying $P(x)$
 $cf(P,x)$ the degree to which P is “generally” believed of x .

It is assumed here that cf is a value $v \in \{\perp\} \cup [0,1]$. That is, a value that is either undefined, or a real value between 0 and 1. We also suggest that this value is constantly modified through a feedback mechanism, as more examples are experienced⁴⁶.

7.2.2 A Suggested Procedure that is Sensitive to Computational Constraints

In modeling quantifiers as procedures that are sensitive to time and memory constraints we are interested in interpreting statements of the form *every* C P (the set-theoretic counterpart of the *wff* $\forall x(C(x) \rightarrow P(x))$), where C has an indeterminate cardinality. Verifying *every*(C,P) is depicted graphically in Figure 17 below, where it is assumed that the property P is generally attributed to members of the concept C with certainty $cf(C,P)$, and where $cf(C,P)=0$

⁴⁵ Note that this is the *Closed World Assumption* (CWA), typically assumed in database systems.

⁴⁶ This is similar to the dynamic reasoning process suggested by Wang (1994).

represents the fact that P is *not* generally assumed of objects in C . On the other hand, a value of cf near 1 represents a strong bias towards believing P of C at face value. In the former case, the processing depends little, if at all, on our general belief, but more on the actual instances.

In the latter case, and especially when faced with time and memory constraints, more weight might be given to prior stereotyped knowledge that we might have accumulated. Based on these arguments, an algorithm that implements a procedure for the quantifier *every* is suggested below:

Figure 17. Quantification with time and memory constraints.

1. An attempt at an exhaustive verification of all the elements in the set C is first made, within a fixed amount of time and some memory constraints (this is the default meaning of “every”).
2. If time and/or memory capacity allow the processing of all the elements in C , then the result is “true” iff $np = |C|$ (that is, if every C P), and “false” otherwise.
3. If time and/or memory constraints do not allow an exhaustive verification, then we will attempt making a decision based on the evidence at hand, where the evidence is based on cf , nn , np (a suggested function is given below).
4. In 3, cf is computed from C elements that are currently active in short-term memory, if any, otherwise cf is the current value in the knowledge base.

5. The certainty factor cf can be updated based on the new result/evidence.

In 3, the final output is determined as a function F that could be defined as follows:

$$(95) \quad F_{\langle C, P \rangle}(nn, np, \varepsilon, cf, \omega) = (np > \varepsilon \cdot nn) \wedge (cf(C, P) \geq \omega)$$

where ε and ω are quantifier-specific parameters. In the case of *every*, the function in (95) states that, in the absence of time and memory resources to process *every*(C, P) exhaustively, the result of the process is *true* if there is an overwhelming positive evidence (high value for ε), and if there is some prior stereotyped belief supporting this inference (i.e., if $cf > \omega > 0$). This essentially amounts to processing *every*(C, P) as *most*(C, P), which seems reasonable in situations such as those in (93)⁴⁷.

Note that the modality of a relation (discussed in chapter 3) could also be added as an additional parameter, since these generalizations (from *every* to *most*, for example), should perhaps not be allowed when a relation is *obligatory* (necessary). That is, in some situations it is probably necessary that the verification be exhaustive, which would be the situation corresponding to (b) in table 5.

Note also that if *most* was the quantifier we started with, then the function in (95) and the above procedure can be applied, although smaller values for ε and ω will be assigned. These parameters should resemble the ratios suggested in (16) in chapter 3. Finally, it should be noted that the above function is a generalization of all quantifiers as defined by the theory of generalized quantifiers. For example, in table 6 below we suggest what the parameters in (95) should be for the standard definitions⁴⁸.

Quantifier	np	nn	ε
<i>every</i>	$np = C $	$nn = 0$	$\varepsilon > 0$
<i>some</i>	$np > 0$	$nn < C $	$\varepsilon > 0$
<i>no</i>	$np = 0$	$nn = C $	$\varepsilon < 0$

Table 6. The standard quantifiers using the function in (95).

⁴⁷ Strzalkowski (1989) proposed a similar solution in the context of treating *generics* in natural language, although the cardinality of the set was considered as the only relevant parameter.

⁴⁸ We therefore agree with (Fauconnier, 1994) that logic is an idealized abstract model that works under well-constrained situations (called models, in formal logic), but in general fails to capture the idiosyncratic nature of cognition.

Note also that since our formulation of quantifiers as procedures that are sensitive to time and memory constraints is a generalization of the standard definition, the formal properties generally attributed to generalized quantifiers (which we briefly discussed in chapter 6) will still hold true.

7.2.3 Further Work

Our interest in further investigation of quantification at the implementational level which we started in (Saba and Corriveau, 1995) is part of our interest in developing an overall strategy for commonsense reasoning in reference resolution and resolving scope ambiguities. As we argued earlier, quantifiers serve a dual purpose, namely to *determine what is to be quantified* (the 'determining' function), and to *determine how much is quantified* (the 'quantifying' function). Scope ambiguities occur when determining what is to be quantified, i.e., at the conceptual level, prior to the evaluation of a logical form against some data/knowledge base. Addressing this issue has been our focus in this dissertation.

The implementational level addresses the issue of verifying/evaluating a disambiguated logical form against some data/knowledge base. At this level, as we argued above, the exact nature of the procedures that implement the behavior of the various quantifiers becomes critical. For example, the verification of properties that are universally quantified must occur in a time/memory constrained model, so that generalizations and uncertainty reasoning can be explained.

Another important aspect of investigating quantification at the implementational level is its strong relation to reference resolution, which essentially is a memory search and retrieval process that must conform to some computational constraints. For example, consider the interaction between quantification and reference resolution in the following:

- (96)
$$\left\{ \begin{array}{l} a1. \text{ John's report on Japanese professionals is remarkable.} \\ b1. \text{ David has four very hardworking children.} \end{array} \right\}$$
- $$\left\{ \begin{array}{l} a2. \text{ Everyone works at least 14 hours.} \\ b2. \text{ They work at least 14 hours.} \end{array} \right\}$$

Note that *everyone* (or the pronoun *they*) must be resolved against some set of entities that has been determined by the previous discourse. Regardless of the reference resolution process itself, there is an additional complexity of determining the verification criteria in

each case. For example, 96a1 might force *everyone* (or *they*) to be processed in a time/memory constrained procedure that would perhaps result in interpreting *everyone* as *most*. In 96b1, however, time and memory constraints would most likely allow an exhaustive verification to be performed.

Admittedly, such proposals are highly speculative at this point. However, early investigations in this regard (see (Saba and Corriveau, 1995)), as well as some early work on reference resolution (Frost and Saba, 1991) suggest that an overall inferencing process integrating procedures for handling quantification and reference resolution might be a worthwhile endeavor.

Finally, we have to acknowledge a very important remark that was made by an AAAI-97 reviewer of (Saba and Corriveau, 1997). The use of quantificational constraints in an inferencing procedure, as discussed in this thesis, is essentially a form of a constraint satisfaction procedure. Therefore, the possibility exists of either under- or over-constraining by the algorithm. What is worth investigating further here, is the relationship between individual beliefs (i.e., biases) and between over- and under- constraining by the scoping procedure.

7.3 CONCLUDING REMARKS

In this dissertation we have investigated a central problem in computational linguistics, namely the resolution of quantifier scope ambiguities. While most approaches to this problem incorporate decision procedures that implement a number of syntactic and semantic rules, we have suggested that these combinatorial approaches do not suffice. Quantifier scope ambiguity seems to be encoded at the pragmatic level, and as such, inferencing procedures that utilize commonsense world knowledge must be developed.

We described an inferencing procedure that makes two simple assumptions regarding the knowledge representation, and we showed that the result is a highly effective procedure that, in addition to providing an elegant treatment for quantifier scope, seems to explain a number of related phenomena that have traditionally been studied in isolation.

Finally, we have suggested that the resolution of scope ambiguities at the conceptual level is but one part of the puzzle, and another inferencing process that operates at the implementational level must also be developed. This process would have to interact with

inferential procedures that are employed in reference resolution. We aim to continue this investigation of building an overall inferencing strategy for the treatment of quantification and reference resolution.

At this early stage, we believe that this integration must be made in a time and memory constrained model that we hope can help us explain several phenomena, such as uncertainty reasoning and generalization (stereotyping).

Our motivation in continuing these investigations is our belief that, ultimately, a natural language understanding system must somehow incorporate a number of reasoning strategies, including inductive, deductive, probabilistic, and commonsense reasoning. If the experience of few decades of research in NLP tells us something, it should be that developing natural language understanding systems will not be achieved by implementing simple and one-dimensional (psychological, logical, linguistic) models. Instead, our conviction is that several reasoning strategies have to be integrated, knowing fully well that integrating such reasoning strategies into one language processing system will no doubt prove to be a very challenging task.

With this realization of the difficult road ahead, we cannot but reiterate how it was truly amazing of Turing to accurately predict nearly half a century ago what the most challenging task in building intelligent machines might be: understanding spoken language!

Bibliography

- Allen, J. (1987), *Natural Language Understanding*, Benjamin/Cummings Pub., Menlo Park, CA.
- Alshawi, H. (1990), Resolving Quasi Logical Forms, *Computational Linguistics*, 16, 133-144.
- Alshawi, H. and van Eijck, J. (1989), Logical Forms in the CORE Language Engine, *In Proceedings of the 27th Annual Meeting of the Assoc. for Computational Linguistics.*, pp. 25-32.
- Barwise, J. (1989), *The Situation in Logic*, CLSI Publications, Stanford, CA.
- Barwise, J. (1979), On Branching Quantifiers in English, *J. of Philosophical Logic*, 8, 47-80.
- Barwise, J. and Cooper, R. (1981). Generalized Quantifiers and Natural Language, *Linguistics and Philosophy*, 4, 159-219.
- van Benthem, J. (1986), *Essays in Logical Semantics*, D. Reidel Publishing Company, Dordrecht, Holland.
- Cherniak, C. (1992), *Minimal Rationality*, MIT Press.
- Cooper, R. (1995), The Role of Situations in Generalized Quantifiers, In L. Shalom (Ed.), *Handbook of Contemporary Semantic Theory*, Blackwell.
- Cooper, R. (1987), Meaning Representation in Montague Grammar and Situation Semantics, *Computational Intelligence*, 3, 35-44.
- Cooper, R. (1983). *Quantification and Syntactic Theory*, D. Reidel Publishing Company, Dordrecht, Netherlands.
- Corriveau, J-P. (1995). *Time-Constrained Memory: A Reader-Based Approach to Text Comprehension*, Lawrence Erlbaum Associates, NJ.
- Cresswell, M. J. (1973), *Logics and Languages*, Methuen & Co Ltd., London.
- van Deemter, K. and Peters, S. (1996) (Eds.), *Semantic Ambiguity and Underspecification*, 55-76, CSLI Publications, Stanford, CA.
- van Deemter, K. (1996), Towards a Logic of Ambiguous Expressions, In K. van Deemter and S. Peters (1996).
- van der Does, J. (1994), On Complex Plural Noun Phrases, In Kanazawa *et al.* (Eds.), *Dynamics, Polarity, and Quantification*, 81-118, CLSI Pub., Stanford, CA.
- van der Does, J. and Verkuyl, H. (1996), Quantification and Predication, In K. van Deemter and S. Peters (1996).
- Dowty, D. R., Wall, R. E. and Peters, S. (1981), *Introduction to Montague Semantics*, D. Reidel Publishing Company.

- Dullea, J. and Song, I-Y. (1997), An Analysis of Cardinality Constraints in Redundant Relationships, In *Proceedings of the Conference on Information and Knowledge Management (CIKM-97)*, 270-276, Las Vegas, Nevada.
- Dumett, M. (1973), *Frege: Philosophy of Language*, Harvard University Press.
- Enderton, H. B. (1972), *A Mathematical Introduction to Logic*, Academic Press.
- van Eynde, F. (1990), The Semantics of Tense and Aspect, In M. Filgueiras *et al.* (Eds.), *Proceedings of EAIA '90, Lecture Notes in Artificial Intelligence*, Vol. 476, 158-184, Springer-Verlag, NY.
- Fauconnier, G. (1994), *Mental Spaces: Aspects of meaning construction in natural language*, Cambridge University Press.
- Fauconnier, G. (1975), Do Quantifiers Branch?, *Linguistic Inquiry*, 6, 555-567.
- Flach, P. A. (1992), An Analysis of Various Forms of 'Jumping to Conclusions', In K. P. Jantke (Ed.), *Lecture Notes in Artificial Intelligence*, Subseries of *Lecture Notes in Computer Science*, 642, Springer-Verlag, 170-186.
- Fox, D. (1995), Economy and Scope, *Natural Language Semantics*, 3, 283-341.
- Freeman, N. H. and Stedmon, A. J. (1986). How Children Deal with Natural Language Quantification, In I. Kurcz, et al (Eds.), *Knowledge and Language*, 21-48, North Holland.
- Frost, R. A. and Saba, W. S. (1991), Executable Attribute Grammars and Knowledge Base Systems, In *Proceedings of the 4th UNB Artificial Intelligence Symposium*, New Brunswick, Canada, 471-482.
- Gamut, L.T.F. (1991), *Logic, Language, and Meaning - Volume 2: Intensional Logic and Logical Grammar*, University of Chicago Press.
- Geller, J. (1994), Advanced Update Operations in Massively Parallel Knowledge Representation, In H. Kitano and J. Hendler (Eds.), *Massively Parallel Artificial Intelligence*, 74-101, AAAI Press/MIT Press, Menlo Park, CA.
- Givon, T. (1984), Deductive Vs. Pragmatic Processing in Natural Language, In W. Kintsch, J. R. Miller and P. G. Polson (Eds.), *Methods and Tactics in Cognitive Science*, 137-189, LEA Accosiates, NJ.
- Goertzel, B. (1993), *The Structure of Intelligence: A New Mathematical Model of Mind*, Springer Verlag.
- Gosling, J. *et al.* (1996), *The Java Programming Language*, SunSoft Press.
- Grimshaw, J. (1990), *Argument Structure*, Cambrdige, MA: MIT Press.
- Grosz, B., Appelt, D., Martin, P., and Pereira, F.C.N. (1987), TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces, *Artificial Intelligence*, 32, 173-243.
- Gurevich, Y. (1988), Logic and the Challenge of Computer Science, In E. Borger (Ed.), *Trends in Theoretical Computer Science*, 1-58, Computer Science Press.

- Harper, M. P. (1992), Ambiguous Noun Phrases in Logical Form, *Computational Linguistics*, 18(4), 419-465.
- Hintikka, J. (1974), Quantifiers vs. Quantification Theory, *Linguistic Inquiry*, 5 153-177. Originally appeared in *Dialectica*, 27, 329-358.
- Hobbs, J. R. (1996), Monotone Decreasing Quantifiers in a Scope-Free Logical Form, In K. van Deemter and S. Peters (1996).
- Hobbs, J. R. and Rosenchein, M. (1978), Making Computational Sense of Montague's Intensional Logic, *Artificial Intelligence*, 287-306.
- Hobbs, J. R. and Shieber. (1987), An Algorithm for Generating Quantifier Scopings, *Computational Linguistics*, 13(1&2), 47-55.
- Johnson-Laird, P. N. (1994). Mental Models and Probabilistic Thinking, *Cognition*, 50, 198-209.
- Kameyama, M. (1996), Infeasible Semantics and Defeasible Pragmatics, In M. Kanazawa et al. (Eds.), *Quantifiers, Deduction, and Context*, 111-170, CSLI Publications, Stanford, CA.
- Kaplan, D. (1979). On the Logic of Demonstratives, *Journal of Philosophical Logic*, 8, pp. 81-98.
- Katz, J. (1980), Chomsky on meaning, *Language*, 56, 1-41.
- Knuth, D. (1968), Semantics of Context-Free Languages, *Mathematical Systems Theory*, 2(2), 127-145.
- Kurtzman, H. and MacDonald, M. (1993), Resolution of Quantifier Scope Ambiguities, *Cognition*, 48, 243-279.
- Lakoff, G. (1987), *Women, Fire, and Dangerous Things - What Categories Reveal About the Mind*, Chicago University Press.
- May, R. (1989), Interpreting Logical Form, *Linguistics and Philosophy*, 12(4), 387-436.
- May, R. (1985), *Logical Form: Its Structure and Derivation*, MIT Press, Cambridge, Mass.
- McCarthy, J. (1987), Generality in Artificial Intelligence, *Communications of the ACM*, 30(8).
- Montague, R. (1974). Proper Treatment of Quantification in Natural Language, In Thomason (1974).
- Moran, D. (1988), Quantifier Scoping in the SRI Core Language Engine, In *Proceedings of the 26th Annual Meeting of the Assoc. for Comp. Linguistics*, 33-40.
- Moran, D. and Pereira F. C. N. (1992), Quantifier Scoping, In H. Alshawi (Ed.), *The Core Language Engine*, MIT Press, Cambridge, MA.

- Park, J. (1995), Quantifier Scope and Constituency, In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics* (ACL '95), ACL/MIT Press, 205-212.
- Partee, B. (1992), Syntactic Categories and Semantic Type, In M. Rosner and R. Johnson (Eds.), *Computational Linguistics and Formal Semantics*, 31-96, Cambridge University Press.
- Partee, B. (1974), Opacity and Scope, In M. K. Munitz and P. K. Unger (Eds.), *Semantics and Philosophy*, 81-101, New York University Press.
- Pereira, F. C. N. (1989), A Calculus for Semantic Composition and Scoping, In *Proceedings of the 27th Annual Meeting of the ACL* (ACL '98), 152-160.
- Pereira, F. C. N. and Pollack, M. E. (1991), Incremental Interpretation, *Artificial Intelligence*, 50, 37-82.
- Peres, J. (1990), Basic Aspects of the Theory of Generalized Quantifiers, In M. Filgueiras et al. (Eds.), *Lecture Notes in Artificial Intelligence, Subseries of Lecture Notes in Computer Science* (EAI-90), Vol. 476, Springer-Verlag, 141-157.
- Poesio, M. (1996), Semantic Ambiguity and Perceived Ambiguity, In K. van Deemter and S. Peters (1996).
- Pollack, M. E. and Pereira, F. C. N. (1988), An Integrated Framework for Semantic and Pragmatic Interpretation, In *Proceedings of the 26th Annual Meeting of Association for Computational Linguistics* (ACL '88), 75-86.
- Reinhart, T. (1997), Quantifier Scope: How Labor is Divided between QR and Choice Functions, *Linguistics and Philosophy*, 20(4), 335-397.
- Reinhart, T. (1983), *Anaphora and Semantic Interpretation*, Chicago University Press.
- Reyle, U. (1995), On Reasoning with Ambiguities, In *Proceedings of the 6th Meeting of the Association for Computational Linguistics, European Chapter*, Dublin, Ireland.
- Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, B. and Loernsen, W. (1991), *Object-Oriented Modeling and Design*, Prentice Hall.
- Rodman, R. (1976), Scope Phenomena, Movement "Transformations", and Relative Clauses, in B. Partee (Ed.), *Montague Grammar*, Academic Press, New York.
- Saba, W. S. and Corriveau, J.-P. (1997), A Pragmatic Treatment of Quantification in Natural Language, In *Proceedings of 1997 National Conference of the American Association for Artificial Intelligence- AAAI-97*, RI, USA.
- Saba, W. S. and Corriveau, J.-P. (1995), Quantification and Cognitive Constraints in Natural Language Understanding, In *Proceedings of SBIA '95: The XII Brazilian Symposium on Artificial Intelligence. Lecture Notes in Artificial Intelligence*, Vol. 991, Springer-Verlag, 331-342.

- Scha, R. (1981), Distributive, Collective and Cumulative Quantification. In J. Groendijk *et al.* (Eds.), *Formal Methods in the Study of Language (Part I & II)*, 483-512, Mathematical Tracts, Amsterdam.
- Shastri, L. and Ajjanagadde, V. (1993). From Simple Associations to Systematic Reasoning: A Connectionist Representation of Rules, Variables, and Dynamic Bindings Using Temporal Synchrony, *Behavioral & Brain Sciences*, 16, 417-494.
- Sher, G. (1990), Ways of Branching Quantifiers, *Linguistics and Philosophy*, 13(4), pp. 393-422.
- Schubert, L. K. et al. (1983), Determining Type, Part, Color and Time Relationships, *Computer*, 16(10), 53-60.
- Sowa, J. F. (1991), (Ed.), *Principles of Semantic Networks*, Morgan Kaufmann, San Mateo, CA.
- Strzalkowski, T. (1989), Extra-Sentential Dependencies, Meaning Representation, and Generics, In *Proceedings of 4th Portuguese Conference on Artificial Intelligence (EPLA '89)*, *Lecture Notes in Artificial Intelligence*, Vol. 390, Springer-Verlag, 210-221.
- Thanassas, D. (1992), *The Phenomenon of Commonsense Reasoning: nonmonotonicity, action and information*, Ellis Horwood Limited.
- Thomason, R. (1974) (Editor, with an introduction), *Formal Philosophy: Selected Papers of Richard Montague*, Yale University Press.
- Verkuyl, H. (1989), Aspectual Classes and Aspectual Composition, *Linguistics and Philosophy*, 12, 39-94.
- Wang, P. (1994), From Inheritance Relation to Non-Axiomatic Logic, *International Journal of Approximate Reasoning*.
- Weisler, S. (1987), Quantification, Ellipses, and Logical Form, In J. L. Garfield (ed.), *Modularity in Knowledge Representation and Natural Language Understanding*, 151-172, MIT Press, Cambridge, Mass.
- Zadrozny, W. (1992), On Compositional Semantics, In *Proceedings of COLING-92*, 23-28.
- Zadrozny, W. and Jensen K. (1991), Semantics of Paragraphs, *Computational Linguistics*, Vol. 17, No. 2, 171-209

Appendix A. An Object-Oriented Attribute Grammar

The following is an example of the specification of an attribute grammar (AG) for one syntactic construct, *Sent*. *Sent* has two synthesized attributes: *meaning* and *parseTree*, which are computed as functions of the context (represented by inherited attributes) and the synthesized attributes of its constituents, i.e., as functions of the context and the synthesized attributes of *NP* and *VP*.

Syntactic Rule	Attribute Evaluation (Semantic) Rules
$\langle Sent \rangle \rightarrow \langle NP \rangle \langle VP \rangle$	$Sent \uparrow meaning = func_1(Sent \downarrow context, NP \uparrow meaning, VP \uparrow meaning)$ $Sent \uparrow parseTree = func_2(NP \uparrow parseTree, VP \uparrow parseTree)$ $NP \downarrow context = Sent \downarrow context$ $VP \downarrow context = (Sent \downarrow context, NP \uparrow meaning)$

The following is an example of how such an AG specification could be implemented in an object-oriented language, in this case Java.

```

class Sent extends LingObj
{
    Meaning    meaning;
    ParseTree  parseTree;
    boolean    parsedOK;
    Vector     context;

    NP         np;
    VP         vp;

    Sent(Vector input, Vector cntxt)
    {
        super(input);
        context = cntxt;
    }
    LingObj lobj = consistsOf(context, np, vp);
    if (lobj.parsedOK)
    {
        meaning = func1(context, np.meaning, vp.meaning);
        parsedOK = true;
        parseTree = func2(np.parseTree, vp.parseTree);
    }
    private Meaning func1(Meaning npMeaning, Meaning vpMeaning) { ... }
    private ParseTree func2(ParseTree npParseTree, ParseTree vpParseTree) { ... }
};

```

Appendix B. A Detailed Example

In this appendix we detail the process of resolving the scope ambiguities in a non-trivial sentence. Consider the following sentence:

Some salesman of every department in most companies saw a few samples of each product.

In an incremental approach, where all 5! Readings are initially considered, the sentence has 120 valid readings. However, in our approach the most plausible reading is selected at every step in such a way that there is always a single preferred reading selected.

The processing of the example given above proceeds as shown in the following steps.

$$\begin{aligned}
 & \text{scope} \left(\begin{array}{l} \text{Some salesman of every department in most} \\ \text{companies saw a few samples of each product} \end{array} \right) \\
 &= \text{scope}_2 \left(\begin{array}{l} \text{scope}_1 (\text{Some salesman of every department in most companies}) \\ \text{Saw} \\ \text{scope}_1 (\text{a few samples of each product}) \end{array} \right) \\
 &= \text{scope} \left(\begin{array}{l} \text{scope}_1 \left(\begin{array}{l} \text{Some salesman} \\ \text{of} \\ \text{scope}_3 (\text{every department in most companies}) \end{array} \right) \\ \text{Saw} \\ \text{scope}_2 \left(\begin{array}{l} \text{a few samples} \\ \text{of} \\ \text{each product} \end{array} \right) \end{array} \right) \\
 &= \text{scope} \left(\begin{array}{l} \text{scope}_1 \left(\begin{array}{l} \exists m (\text{Salesman}(m) \wedge Q(m)) \\ \text{of} \\ \text{scope}_3 \left(\begin{array}{l} \text{every department} \\ \text{in} \\ \text{most companies} \end{array} \right) \end{array} \right) \\ \text{Saw} \\ \text{scope}_2 \left(\begin{array}{l} \text{a few samples} \\ \text{of} \\ \text{each product} \end{array} \right) \end{array} \right) = \text{scope} \left(\begin{array}{l} \text{scope}_1 \left(\begin{array}{l} \exists m (\text{Salesman}(m) \wedge Q(m)) \\ \text{of} \\ \text{scope}_3 \left(\begin{array}{l} \forall d (\text{Department}(d) \rightarrow Q(d)) \\ \text{in} \\ \exists (most) \neg (\text{Company}(c) \rightarrow Q(c)) \end{array} \right) \end{array} \right) \\ \text{Saw} \\ \text{scope}_2 \left(\begin{array}{l} \exists (aFew) s (\text{Sample}(s) \rightarrow Q(s)) \\ \text{of} \\ \forall p (\text{Product}(p) \rightarrow Q(p)) \end{array} \right) \end{array} \right)
 \end{aligned}$$

$$= \text{scope}_{\text{Saw}} \left(\text{scope}_1 \left(\begin{array}{l} \exists m (\text{Salesman}(m) \wedge Q(m)) \\ \text{of} \\ \exists! (\text{most})_c \left(\begin{array}{l} \text{company}(c) \rightarrow \\ \forall d (\text{department}(d) \wedge \text{In}(d, c) \rightarrow Q(d)) \end{array} \right) \end{array} \right) \right)$$

$$= \text{scope}_{\text{Saw}} \left(\begin{array}{l} \exists! (\text{most})_c \left(\begin{array}{l} \text{company}(c) \rightarrow \\ \forall d \left(\begin{array}{l} \text{department}(d) \wedge \text{In}(d, c) \\ \rightarrow \exists m (\text{Salesman}(m) \wedge Q(m)) \end{array} \right) \end{array} \right) \\ \forall p \left(\begin{array}{l} \text{Product}(p) \rightarrow \\ \exists! (\text{aFew})_s (\text{Sample}(s) \wedge \text{Of}(s, p) \rightarrow Q(s)) \end{array} \right) \end{array} \right)$$

$$= \forall p \left(\begin{array}{l} \text{Product}(p) \rightarrow \\ \exists! (\text{aFew})_s \left(\begin{array}{l} \text{Sample}(s) \wedge \text{Of}(s, p) \rightarrow \\ \exists! (\text{most})_c \left(\begin{array}{l} \text{company}(c) \rightarrow \\ \forall d \left(\begin{array}{l} \text{department}(d) \wedge \text{In}(d, c) \\ \rightarrow \exists m (\text{Salesman}(m) \wedge \text{Saw}(m, s)) \end{array} \right) \end{array} \right) \end{array} \right) \end{array} \right)$$

Note that at the various steps the scoping algorithm invoked by the **scope** function calls uses various quantificational constraints as follows:

scope₁ uses $QC(\text{Of}, \text{Salesman}, \text{Department}) = \langle \text{many}, 1 \rangle$

scope₂ uses $QC(\text{Of}, \text{Sample}, \text{Product}) = \langle \text{many}, 1 \rangle$

scope₃ uses $QC(\text{In}, \text{Department}, \text{Company}) = \langle \text{many}, 1 \rangle$

Appendix C. Sample Session with *Quasi*

List of Examples Tested in *Quasi*

- every engineer attended a seminar
- a senior engineer gave 2 seminars
- every senior engineer attended a conference at MIT
- a senior engineer attended every seminar at MIT
- John visited a house on every street
- John advertised a house on every street
- John advertised a house on every occasion
- John visited a house on every occasion
- John visited every house on some street
- John placed a book on every shelf
- John placed every book on some shelf
- every senior engineer of a software company attended ACL98
- every senior president of a software company attended ACL98
- every student submitted a paper
- a paper was submitted by every student
- every student speaks a foreign language
- a foreign language is spoken by every student
- every MIT student submitted a paper to AAAI98
- several MIT student students submitted a paper to AAAI98
- 2 MIT student students submitted a paper to AAAI98
- a paper was submitted to AAAI98 by 2 MIT students
- 2 men lifted every piano
- 2 men lifted 4 pianos
- 4 men lifted 2 pianos
- 4 men are lifting two pianos
- 3 men lifted 2 pianos
- 3 MIT students submitted a paper on quantification to ACL98
- a paper on quantification was submitted to ACL98 by 3 MIT students
- a scientist was consulted by two senators
- a scientist from every country was consulted by two senators
- an author of every novel was mentioned in an article by every critic
- 4 young girls lifted 2 pianos

A Detailed Trace of several Examples

Note:

The typical cardinality of all concepts was assumed to be 100 by default. Each noun modifier reduces the cardinality by 15% and each adjective by 10%. This is done to capture the effect of the linguistic context, as explained in chapter 3.

Input: every senior engineer gave a seminar

-> Sentence([every, senior, engineer, gave, a, seminar])

-> Parse succeeded... Remaining input = []

-> Scope Neutral Logical Form:

```
[
  FORALL(x1) [ENGINEER(x1) & SENIOR(x1) -> EXISTS(x2) [SEMINAR(x2) & Gave(x1,x2)]]
  EXISTS(x1) [SEMINAR(x1) & FORALL(x2) [ENGINEER(x2) & SENIOR(x2) -> Gave(x2,x1)]]
]
```

-> Applying the scoping function

- Accounting for the linguistic context...
- Retrieving the relevant QC...
- Retrieved QC(Give,Student,Seminar)=<1,1+>
- Constructing appropriate ranges...
 - For m1=1 lower bound = 1.0; upper bound = 1.0
 - For m2=1+ lower bound = 1.0; upper bound = 100.0
 - For Q1=every lower bound = 100.0; upper bound = 100.0
 - For Q2=a lower bound = 1.0; upper bound = 1.0
- Constructed appropriate ranges...
- Multiplying ranges by appropriate (linguistic context) ratios...
- Dividing ranges...
- Done preparing ranges...
- Computing plausibility measure...
 - PLAUSIBILITY of direct reading = 1.0
 - PLAUSIBILITY of indirect reading = 0.11

-> Completed scoping function....

-> Direct reading is more plausible.

-> Retrieving the most plausible reading:

```
FORALL(x1) [ENGINEER(x1) & SENIOR(x1) -> EXISTS(x2) [SEMINAR(x2) & Gave(x1,x2)]]
```

Input: a senior engineer gave 2 seminars

-> Sentence([a, senior, engineer, gave, 2, seminars])

-> Parse succeeded... Remaining input = []

-> Scope Neutral Logical Form:

```
[
  EXISTS(x1)[ENGINEER(x1) & SENIOR(x1) & E!(2:)(x2)[SEMINAR(x2) & Gave(x1,x2)]];
  E!(2:)(x1)[SEMINAR(x1) & EXISTS(x2)[ENGINEER(x2) & SENIOR(x2) & Gave(x2,x1)]]
]
```

-> Applying the scoping function

- Accounting for the linguistic context...
- Retrieving the relevant QC...
- Retrieved QC(Give,Student,Seminar)=<1,1+>
- Constructing appropriate ranges...
 - For m1=1 lower bound = 1.0; upper bound = 1.0
 - For m2=1+ lower bound = 1.0; upper bound = 100.0
 - For Q1=every lower bound = 100.0; upper bound = 100.0
 - For Q2=a lower bound = 1.0; upper bound = 1.0
- Constructed appropriate ranges...
- Multiplying ranges by appropriate (linguistic context) ratios...
- Dividing ranges...
- Done preparing ranges...
- Computing plausibility measure...
 - PLAUSIBILITY of direct reading = 1.0
 - PLAUSIBILITY of indirect reading = 1.0

-> Completed scoping function....

-> Both readings are (equally) plausible.

-> Retrieving the most plausible reading (direct reading):

```
EXISTS(x1)[ENGINEER(x1) & SENIOR(x1) & E!(2:)(x2)[SEMINAR(x2) & Gave(x1,x2)]]
```

Input: John visited a house on every street

-> Sentence([John, visited, a, house, on, every, street])

-> Parse succeeded... Remaining input = []

-> Scope Neutral Logical Form:

```
[
  EXISTS(x1)[HOUSE(x1) & FORALL(x2)[STREET(x2) & On(x1,x2)->Visited(John,x1)]]
  FORALL(x1)[STREET(x1)->EXISTS(x2)[HOUSE(x2) & On(x2,x1) & Visited(John,x2)]]
]
```

-> Applying the scoping function

- Accounting for the linguistic context...
- Retrieving the relevant QC...
- Retrieved QC(Give,Student,Seminar)=<1+,1>
- Constructing appropriate ranges...
 - For m1=1+ lower bound = 1.0; upper bound = 100.0
 - For m2=1 lower bound = 1.0; upper bound = 1.0
 - For Q1=a lower bound = 1.0; upper bound = 1.0
 - For Q2=every lower bound = 100.0; upper bound = 100.0
- Constructed appropriate ranges...
- Multiplying ranges by appropriate (linguistic context) ratios...
- Dividing ranges...
- Done preparing ranges...
- Computing plausibility measure...
 - PLAUSIBILITY of direct reading = 0.0
 - PLAUSIBILITY of indirect reading = 1.0

-> Completed scoping function....

-> Indirect reading is more plausible.

-> Retrieving the most plausible reading:

```
FORALL(x1)[STREET(x1)->EXISTS(x2)[HOUSE(x2) & On(x2,x1) & Visited(John,x2)]]
```

Input: a paper was submitted to AAAI98 by 2 MIT students

-> Sentence([a, paper, was, submitted, to, AAAI98, by, 2, MIT, students])

-> Parse succeeded... Remaining input = []

-> Scope Neutral Logical Form:

```
[
  EXISTS(x1)[PAPER(x1) & E!(2:)(x2)[MIT-STUDENT(x2) & Submitted(x2,x1)]]
  E!(2:)(x1)[MIT-STUDENT(x1) & EXISTS(x2)[PAPER(x1) & Submitted(x1,x2)]]
]
```

-> Applying the scoping function

- Accounting for the linguistic context...
- Retrieving the relevant QC...
- Retrieved QC(Give,Student,Seminar)=<2-,1>
- Constructing appropriate ranges...
 - For m1=2- lower bound = 1.0; upper bound = 2.0
 - For m2=1 lower bound = 1.0; upper bound = 1.0
 - For Q1=a lower bound = 1.0; upper bound = 1.0
 - For Q2=2 lower bound = 2.0; upper bound = 2.0
- Constructed appropriate ranges...
- Multiplying ranges by appropriate (linguistic context) ratios...
- Dividing ranges...
- Done preparing ranges...
- Computing plausibility measure...
 - PLAUSIBILITY of direct reading = 1.0
 - PLAUSIBILITY of indirect reading = 1.0

-> Completed scoping function....

-> Both readings are (equally) plausible.

-> Retrieving the most plausible reading (direct reading):

```
EXISTS(x1)[PAPER(x1) & E!(2:)(x2)[MIT-STUDENT(x2) & Submitted(x2,x1)]]
```

Input: every MIT student submitted a paper to AAAI98

-> Sentence([every, MIT, student, submitted, a, paper, to, AAAI98])

-> Parse succeeded... Remaining input = []

-> Scope Neutral Logical Form:

```
[
  FORALL(x1) [MIT-STUDENT(x1) -> EXISTS(x2) [PAPER(x2) & Submitted(x1,x2)]]
  EXISTS(x1) [PAPER(x1) & FORALL(x2) [MIT-STUDENT(x2) -> Submitted(x2,x1)]]
]
```

-> Applying the scoping function

- Accounting for the linguistic context...
- Retrieving the relevant QC...
- Retrieved QC(Give,Student,Seminar)=<2-,1>
- Constructing appropriate ranges...
 - For m1=2- lower bound = 1.0; upper bound = 2.0
 - For m2=1 lower bound = 1.0; upper bound = 1.0
 - For Q1=every lower bound = 100.0; upper bound = 100.0
 - For Q2=a lower bound = 1.0; upper bound = 1.0
- Constructed appropriate ranges...
- Multiplying ranges by appropriate (linguistic context) ratios...
- Dividing ranges...
- Done preparing ranges...
- Computing plausibility measure...
 - PLAUSIBILITY of direct reading = 1.0
 - PLAUSIBILITY of indirect reading = 0.0

-> Completed scoping function....

-> Direct reading is more plausible.

-> Retrieving the most plausible reading:

```
FORALL(x1) [MIT-STUDENT(x1) -> EXISTS(x2) [PAPER(x2) & Submitted(x1,x2)]]
```