An Implementable Semantics for Comparative Constructions[1]

Manny Rayner and Amelie Banks[2]
Swedish Institute of Computer Science
Box 1263
S-164 28 KISTA
Sweden

We describe a comprehensive treatment of the syntax and semantics of comparative constructions based on theoretical work by Pinkham, which can be implemented in a relatively straightforward fashion within a feature-based phrase-structure grammar. Comparatives are divided up into "clausal" and "phrasal" constructions; in contrast to most previous theories, however, phrasals are not regarded as reduced forms of clausals. We begin by defining a Montagovian semantics for phrasal comparatives that directly links interpretation rules to the surface syntactic structure; we then show how this solution can be made computationally more efficient by "factoring" the interpretation through an intermediate level of representation, using a method similar to that used for dealing with scoping phenomena. Detailed examples are provided, showing how the method can correctly describe the semantics of a variety of superficially widely different comparative constructions. The ideas have all been implemented within a large-scale grammar for Swedish, a "toy" version of which is presented, along with examples of the output.

1 Introduction

1.1 Overview

This paper is written with two different, though to some extent overlapping, audiences in mind. On the practical side, we address ourselves to the natural language interface constructor, who wants to expand the coverge of her system so as to incorporate comparative constructions; the state of the art has now progressed to the point where it is generally acknowledged that this is a problem that a serious system must attempt to deal with. In the following, we will thus attempt to describe our ideas in a sufficiently concrete fashion that they can be used as an implementation guide.

Typical examples of comparatives according to the traditional classification are the following:

Clausal comparatives

1) Mary was happier in New York than John was in London.
2) John has more books than Mary has newspapers.
3) The table is longer than it is wide.

Comparative ellipsis

4) John is taller than Mary.
5) Few people run as fast as John.
6) John bought more books than Mary.
7) John was happier in New York than in London.
8) Mary had more friends than John thought.
9) John hit Mary harder than he meant to.
10) More men than women bought the book.
11) More people voted for the proposal than against.
12) John likes Mary's house better than Mary John's.

Unclear cases

13) More than 50 people signed the petition.
14) Mary needs a larger car than this Fiat.
15) John ran faster than the world record.
16) John was born in the same city as Mary.
17) John arrived before Mary.

On the theoretical side, we want to re-examine some fundamental questions about the nature of the comparative construction, and in particular about the phenomenon generally referred to as "comparative ellipsis"; by this, we mean comparative constructions like those listed above in which the element after the comparative marker is not a clause, and which traditionally have been explained in terms of various kinds of deletion rules. Here, we will argue that, in most common cases, it is by no means necessary to postulate syntactic deletion operations. We will claim, rather, that it is possible (and indeed *simpler*) to assign a syntactic representation to such constructions that closely follows the apparent surface constituent structure. We will then show how to define semantic interpretation rules for such structures, which at no level reconstruct a representation of a syntactic clause, but rather operate on the principle of directly comparing constituents of similar type. To highlight the difference between our approach and the traditional one, we will refer to constructions handled by this method as examples, not of *comparative ellipsis*, but rather of *contrastive comparison.*

Our opinion is that the issues we are raising are probably applicable to most grammatical frameworks subsumed under the *feature-based phrase structure grammar* paradigm; this includes established formalisms like LFG, GPSG, HPSG, PATR-II etc., and a vast number of more or less obscure formalisms in use around the world. At risk of offending everybody and pleasing nobody, we will make no particular attempt to align ourselves with any one of these theories; in Section 1.2, we will describe in general terms the properties we will expect a framework to possess, if it is to be capable of supporting the type of operations we will need. Basically, we will demand distinct levels of representation for constituent structure, function-argument structure, and logical form, related in the normal way; we will also require certain constructions to be treated as syntactic constituents. Most of this will be quite uncontroversial.

To clear the decks before moving into action, we now spend a couple of paragraphs making clear which questions we are *not* attempting to answer. We regard the lexical semantics of comparatives as being outside the scope of the paper; we are only interested in determining how the phrasal semantics can be derived from them. We explain exactly what we mean by this statement in Section 1.2. We will ignore complex comparative operators ("more than twice as much as"; "at least two inches longer than three times as long as"); these are well dealt with in Ballard (1988), and in any case do not seem to pose fundamental difficulties. For similar reasons, we refrain from discussing determiners with phrasal comparatives, like "more than two" or "at least half": here, the most comprehensive reference is perhaps Keenan and Stavi (1986). We also ignore certain idiosyncratic constructions ("as fast as ever"; "as much as usual"), which do not seem to fit any pattern known to us.

In our treatment of clausal comparatives, we will present two slightly different variants of a solution. The first is simple to implement, and appears capable of giving correct semantics to most normal examples; however, it fails to give both readings for the notorious Russell example, "I thought your yacht was longer than it is." The second solution, on the other hand, is capable of dealing with this kind of sentence, but needs a number of additional scoping rules. We say more about this in Section 4.

With regard to cross-linguistic coverage, we will restrict ourselves to English, Swedish, and a few other languages in which the comparative construction is syntactically similar in nature; this actually appears to include most European languages. Although we suspect that our results are probably applicable to at least some languages outside this group,[3] we shall not make any further attempt to lend credibility to this claim.

The organization of the rest of the paper is as follows. In Section 2 we review previous work: in particular, we point out what we regard as the main weaknesses of the deletion method, and summarize earlier attempts to describe phrasal comparatives by direct interpretation methods. The next two sections contain the main results of the paper.

In Section 3, we examine contrastive phrasal comparatives. We begin by defining a Montagovian semantics for these, and then show how this can be improved by "factoring" it through an intermediate level of representation, in which the "contrasting" relationship between phrases is left undefined. We then go on to show how this methodology can be used to provide a readily implementable semantics for a wide variety of contrastive constructions, which also gives a simple explanation of the ambiguity of certain comparative sentences. Section 4 then goes on to describe our treatment of clausal comparatives, which is largely adapted from the work reported in Klein (1980) and Pinkham (1985).

In Section 5 we briefly discuss noncoreference of compared objects; we also look at sentences containing the words "before" and "after," pointing out some differences between these and comparative sentences. A corpus analysis, which gives some indication of the relative frequencies of the various constructions, is presented in Section 6. In Section 7 we summarize our results.

The theoretical ideas advocated here were first developed and tested in practice within the framework of the SNACK-85 project (Rayner and Banks 1986; 1989). In Appendix 2 we give the complete program code for a "scaled-down" version of the system, which handles all the constructions described in the main body of the paper, and in Appendix 3 we show examples of the system's output.

1.2 THEORETICAL ASSUMPTIONS

1.2.1 GENERAL REMARKS ON THE THEORETICAL FRAMEWORK

In this section, we attempt to summarize the theoretical assumptions needed to support our arguments. The main body of the paper is Section 3, which deals with phrasal comparatives: here we present our theory in two separate forms.

Version one is a Montague grammar, modified slightly to allow use of generalized quantifiers. This has the advantage of theoretical elegance, but as we shall see is computationally unattractive. We consequently recast it in a form more suitable for implementation purposes, which draws its inspiration from the well-known treatment of scoping phenomena originally described in Woods et al. (1972); later elaborations and refinements have been implemented in a number of systems, notably CHAT-80 (Pereira 1983), TEAM (Grosz et al. 1986), and CLE (Moran 1988; Alshawi et al. 1989). We will continue to use this framework in Section 4, where we consider clausal comparatives, and now describe it in more detail.

Most importantly, we have three distinct levels of linguistic representation, which we refer to as *syntax, quasi-logical form,*[4] and *logical form;* we stress that these terms are merely used for convenience, and are not to be read as referring to any specific linguistic theory, Chomskian or otherwise. The levels will encode the following information:

Syntax. At this level, the structure of the sentence is represented as a phrase structure tree, where each node is associated with a set of *features*. We will make two assumptions about the feature structure. First, we distinguish a special feature, which encodes the quasi-logical form of a constituent; not all nodes need possess this feature. Second, there will be some mechanism that permits the definition of long-range dependencies, by allowing features in different constituents to share information.

Quasi-logical form (QLF). This level encodes the predicate-argument structure of the phrase; the only demand we will make is that, like any other feature value, its value can be derived entirely by phrase structure rules with feature unification. (This is in spirit similar to a compositionality restriction, but is weaker.)

Logical form (LF). At this level, the sentence is associated with an expression in a (generally higher-order) predicate calculus; the logical form of a sentence will be derivable from its quasi-logical form by application of rewriting rules representing scoping and other transformations. Since there is little agreement concerning the representation of logical forms, we now say a few words about the less usual features of our system.

First, we will find it convenient to treat events as objects, since this can often greatly simplify the semantics of adverbial modification. For example, we will represent *Gustav III ruled for 21 years* as something like

∃e.event_type(e,rule)&agent(e,gustav_iii)&duration(e,21)

We will also make extensive use of generalized quantifiers (Barwise and Cooper 1981); these will play a key role in our representation of comparatives. We will not, however, use Barwise and Cooper's notation, which involves combining a determiner and a set-descriptor to form a quantifier, but rather will take the equivalent approach of treating determiners as two-place higher-order predicates on sets. We will use $\{N \mid \phi(N)\}$ for the determiner "A number N such that $\phi(N)$": thus we write

no(rose,λx.send(john,x,mary))

for "John sent no roses to Mary,"

{N | N<5}(rose,λx.send(john,x,mary))

for "John sent less than 5 roses to Mary," and

{N | ∃N'.N<N'&{N'}
(rose,λx.rose(x)&in(x,garden))}
(rose,λx.send(john,x,mary))

for "John sent fewer roses to Mary than there were roses in the garden." Formulas like the third one above will occur frequently in Section 3, and we advise the reader to spend a minute at this point making sure that she understands the notation.

1.2.2 SPECIFIC SYNTACTIC AND SEMANTIC ASSUMPTIONS

The basic scheme, or some not too distant relative, is the one used in many large-scale implemented systems; as examples, we can quote TEAM (Grosz et al. 1987), PUNDIT (Dahl et al. 1987), TACITUS (Hobbs et al. 1988), MODL (McCord 1987), CLE (Alshawi et al. 1989), and SNACK-85 (Rayner and Banks 1986). It also has close links with theoretical work in situation semantics (Pollard and Sag 1988; Fenstad et al. 1987). We start by enumerating the basic syntactic constituents S, NP, VP, DET, AP, ADVP, PP; as far as our ontological commitments go, we need a set of THINGS, a set of EVENTS, and a set of DEGREES. We will assume that DEGREES are isomorphic to some kind of numbers.

An NP will consist of a DET and a CN (we borrow this nonstandard term for N-bar from Montague grammar). A DET will semantically correspond to a generalized determiner, and a CN to a subset of THINGS. An NP will thus become a generalized quantifier.

A VP consists of a VERB, followed by a list of VERB-ARGs (or just ARGs), by which we mean its obligatory and optional complements. An ARG is represented at QLF level as a pair ⟨*arg-type, arg-value*⟩. We will need at least the following kinds of complements: NP, PP, S-COMP (a suitable S), VP-COMP (a suitable VP), APs. Semantically, a verb will be associated with a subset of EVENTS, which we can think of as a type of event, state, or situation. The VALUE of each ARG will also be semantically represented by a suitable object, and the ARG-TYPE will translate to a relation between the verb-object and the representation of the ARG-VALUE. We will require a special kind of QLF node, called an **abstraction,** to represent VPs, APs, and other constituents whose denotation is a function from individuals to propositions; one component of the **abstraction** node will be the abstracted variable.

A scalar adjective A translates to a two-place predicate $A'(x, d)$ on OBJECTS×DEGREES, with the interpretation: the object x is A' to degree d. Similarly, a scalar adverb A translates to a two-place predicate $A'(x, d)$ on EVENTS × DEGREES, with the interpretation: the event x is A' to degree d. We will assume that degree expressions like "more," "less," or "at least twice as much as much as" translate to two-place predicates on DEGREES × DEGREES. Since we are assuming that DEGREES are numbers, it will also make sense to use a generalized quantifier as the second argument to an adjective: for example, $A(x, \{N \mid N < 2\})$ will be interpreted as "x is (less than 2) A."

We will also need to be able to form "slash categories" or something similar. In particular, the following will be required (in each example the "gap" is represented as an underscore):

S/DET (e.g. "John bought _ vases")
S/NP (e.g. "John gave _ to Mary")
S/S-COMP (e.g. "John probably believed _ ")
S/VP-COMP (e.g. "John presumably meant to _ ")

A more exact formulation of the basic question we are attacking is thus:

In terms of the above framework, state rules by which we can systematically calculate the denotation of a complex expression involving a comparative construction in terms of

the denotation of its scalar adjectives and adverbs, the denotations of its degree expressions, and the denotations of its other constituents.

2 Previous Work on Comparatives

This section will summarize relevant previous work on comparatives; we begin by listing sources. So far, there has been a surprisingly small amount of work done in computational frameworks: the only major attempts known to us that integrate syntactic and semantic processing are those reported in Ballard (1988) and Friedmann (1989). Limited syntactic/semantic coverage has appeared in some other systems (e.g. DIALOGIC [Robinson 1982] and CLE [Alshawi et al. 1989]), and the Linguistic String Project (Sager 1981) has an advanced syntactic coverage.

On the theoretical front, we have been considerably influenced by Bresnan (1973), Hankamer (1973), Pinkham (1985), and Klein (1980), especially the last three; we were also surprised to discover at a late stage of our research that unpublished work by Heim (1985) and Krifka (1987) had pursued a course almost parallel to our own. We now go on to describe what we regard as relationships between these treatments and ours.

We will start by looking at clausal comparatives, where the immediate problem is clearly to account for the apparent "missing material" in the comparative clause. An early and still very influential study of comparative constructions is that of Bresnan (1972; 1973; 1977), working within transformational grammar; the key concept is the introduction of the rule of Comparative Subdeletion, which will for example derive sentences like 1a) and 2a) by deleting the bracketed material from the base forms 1b) and 2b).

1a) John bought more books than Mary bought.
1b) John bought more books than Mary bought [x many books].

2a) John bought more books than Mary bought records.
2b) John bought more books than Mary bought [x many] records.

Although Comparative Subdeletion explains many of the facts, it makes a number of erroneous predictions. A very long and thorough criticism was advanced in Pinkham (1985), where reasons are given to believe that it is preferable to adopt an analysis that represents the "missing" material as (in English) null proforms; these are bound to the head of the comparison by a mechanism that Pinkham calls Quantifier Binding. Pinkham's view of the clausal comparative has many points in common with that proposed in Klein (1980), in the context of a GPSG/Montague grammar framework, although Klein only really considers sentences of the type in example 2); our own treatment will in this respect be similar to Pinkham's and Klein's.

Like Bresnan and Pinkham, but unlike Friedmann (1989), we will also regard the comparative clause as in general modifying the comparative head in the main clause, as opposed to treating the two clauses as parallel; we discuss further in Section 4 the reasons behind this decision.

Moving on to phrasal comparatives, there would appear to be two basic approaches. The earlier and more common one is to treat these as reduced forms of clausal comparatives; for example, Hankamer (1971) introduces a rule of "C-ellipsis," which deletes material from the comparative clause under identity with material in the main clause. Thus for example, 3a) and 4a) would be regarded as reduced forms of 3b) and 4b), where the bracketed material has been deleted:

3a) John bought more books than Mary.
3b) John bought more books than Mary [bought x many books].

4a) John bought more books than Mary thought.
4b) John bought more books than Mary thought [that he bought x many books].

In a computational treatment, this means that the deletion rule has to be "reversed," so as to reconstruct the deleted material from the phrasal remnant: this is the method used in Sager (1981) and Friedmann (1989).

However, a number of researchers have showed that there are drawbacks to the "C-ellipsis" account. Although both Hankamer and Pinkham use C-ellipsis in their theories, they also provide strong evidence for the existence of at least some comparatives that are not elliptic in nature (Hankamer 1973; Pinkham 1985). Three examples are shown in 5)–7), where Pinkham in each case argues that there are severe difficulties involved in deriving a sentence of this type from a clausal comparative; the natural choice in a transformational framework is to regard the italicized portions as base-generated phrasal.

5) I invited *more men than women.*
6) John ran *faster than the world record.*
7) Mary looks *taller than John* when she wears high heels.

Of particular interest here is Pinkham's treatment of examples like 5), which are interpreted using a rule that she calls "distributive copying." 5) receives the logical form 5a), which after distributive copying becomes 5b). Although the details of the solution are not fully worked out (in particular, the notation is never given a formal semantics), the intuitive idea is clearly that of duplicating a "shared" or "matrix" predicate, substituting in the two compared elements in a suitable way.

5a) I INVITED (MORE [q1 (q1 men), q2 (q2 women)])
5b) MORE [q1 (I INVITED q1 men), q2 (I INVITED q2 women)]

Similar suggestions have also been made in a rather different context by Keenan and Stavi (1986; 282–284); considering the semantics of comparative adjectival constructions like that in 8a), they argue convincingly that these are to be regarded as directly interpreted, rather than as reduced forms of sentences like 8b) or 8c):

8a) More male than female students passed the exam.
8b) More male students than female students passed the exam.
8c) More male students passed the exam than female students passed the exam.

We should also mention the work of McCord (1981; 1985; 1987) on "focalizers"; the key notion we borrow is the use of rewriting rules on semantic representations, to produce a logical form in which contrasted items appear at the same level. Although McCord does not attempt, as we do, to justify his operations formally, the intuitions behind them clearly overlap to a large extent with ours.

In our own treatment of phrasal comparatives, we take all these ideas to their logical conclusion: we dispense with the C-ellipsis rule altogether, and regard all nonclausal comparatives as essentially phrasal, interpreting them by a method analogous to Pinkham's distributive copying. The abstract scheme is concretely implemented in the form of rewriting rules, which are applied to an intermediate quasi-logical form. A solution along these lines, which builds on previously reported work (Banks 1986; Banks and Rayner 1987; Rayner and Banks 1988), is described in Section 3. As we complete this article, we discover that a similar approach for German comparatives has independently been suggested by Krifka (1987); Krifka also proceeds by using direct interpretation rules involving contrasting of compared constituents, although the details of his analysis are in some respects fairly different. The following is a typical example: 9a) is given the semantics represented by 9b), where $\mu d\Phi(d)$ is to be read as "the maximum d, for which $\Phi(d)$ holds," and KOMP(x, y, f) is defined by

$$\mathrm{KOMP}(x, y, \Phi) \Leftrightarrow \Phi(x) > \Phi(y)$$

9a) Ich habe ihr bessere Zahärtze empfohlen als du.
I have him better dentist recommended than you-NOM.

(I have recommended him a better dentist than you [did]).

9b) KOMP(ich, du, $\lambda x \mu d$[x hat ihr d-gute Zahärtze empfohlen]).

3 Phrasal Comparatives

3.1 Basic Ideas

As we have already indicated, our strategy for handling phrasal comparatives will be to provide direct interpretation rules, rather than to assume the existence of deletion rules and attempt to reconstruct a clausal comparative. In the preceding section, we referred to a number of authors who have pointed out weaknesses in the C-ellipsis theory; using their arguments and some of our own, we now describe our reasons for preferring our method to the traditional one.

Our fundamental argument will be Occam's razor: especially from the implementation viewpoint, assuming that phrasals are derived from clausals has little real explanatory power. Since the authors quoted above give good theoretical arguments to show that at least some phrasal comparatives are not the result of C-ellipsis, we are at any rate going to have to provide direct interpretation rules for these cases. If it then turns out that these rules can be systematically extended so as to cover the other phrasal comparatives as well, we can dispense with C-ellipsis altogether and arrive at a simpler system. We claim that this is exactly what we have done.

It is worth looking a little more closely at the case in which the comparative complement is of the form *than NP;* this is certainly the one that occurs most commonly in practice. The first point is well known, and was first described in detail in Hankamer (1971): in English (and in fact a good many other languages), it is possible to extract out of contrastive NP complements, but not from clausal comparative complements. For example, compare 10a) with 10b):

10a) Who$_i$ does John run faster than t_i?
10b) *Who$_i$ does John run faster than t_i runs?

The relevance of this observation is obvious: if contrastive NP complements really are elliptic forms of clausal complements, we would appear to need some mechanism by which something originally an extraction island could cease to be one when a reduction is performed. This would be a very unusual phenomenon, which it would certainly be preferable to avoid introducing into a syntactic theory; rather than do this, it seems much simpler to treat the two constructions as fundamentally different, as we have done.

Another fairly clear-cut piece of evidence can be found in Swedish, where a different possessive pronoun is used depending on whether the comparative is clausal or phrasal, as in 11a) and 11b). Here, the point is that *sin* refers back to the subject of the clause, while *hans* refers to a nonsubject NP in the same clause, or an NP in a different clause. Once again, this would be strange if the comparative complement in 11a) were a clausal remnant.

11a) Sven köpte fler böcker än *sin* far.
Sven bought more books than his father.

11b) Sven köpte fler böcker än *hans* far gjorde.
Sven bought more books than his father did.

However, in practice the biggest problem seems to be that it is not easy to provide a satisfactory definition of the C-ellipsis deletion operation; to be more exact, we need to define the *inverse* of this operation, so as to reconstruct the comparative clause from the reduced form. A relatively simple example is 12a), which we quoted in Rayner and Banks (1988); the sentence is also discussed at length by Friedmann. Here, a naive definition of "deletion of identical material" will result in reconstructing 12b); what we want is rather 12c), which requires an NP to be regarded as "identical" to a pronoun bound by it.

12a) Every tourist spent more in London than in New York.
12b) Every tourist spent more in London than every tourist spent in New York.
12c) Every tourist spent more in London than he spent in New York.

Friedmann's solution to the problem begins by reconstructing a form like 12b), and then makes use of rewriting rules that "merge" the two occurrences of *every tourist* into a single quantifier that has wide scope over the whole sentence; the final result is essentially 12c). This idea is, however, open to several criticisms. To begin with, it has little linguistic motivation; more important, it is difficult to see how to extend it to handle harder cases without adding more *ad hoc* rules. For example, getting the normal ("sloppy") reading of 13a) by Friedmann-like analysis seems to involve transforming it to something like 13b), which rather begs the question of how to account for the different possessive pronouns in the two clauses. On a direct-interpretation account, however, the intended reading for sentences like 13a) can be derived in a natural way; we describe how this can be done in Section 3.4.6.

13a) Few men are as loyal to their country as George (is).
13b) Few are as loyal to their country as George is loyal to his country.

The remainder of Section 3 will give a detailed description of our version of the direct-interpretation method. We start in Section 3.2 by showing how to construct a Montagovian semantics for a fragment containing phrasal constructions; the basic trick is to have a phrase structure rule that combines a syntactic structure with a pronoun-like "slot," a "filler" for the "slot," and an element of the same semantic type as the "filler," against which it is contrasted. This rule can then be linked in the normal way with a corresponding semantic rule, which defines the semantics of the composite structure in terms of the semantics of its constituents.

However, this has the same drawbacks as the original Montague treatment of scoping phenomena, namely that we are introducing phrase structure rules that lack syntactic motivation. Consequently, we introduce in Section 3.3 a computational version of the theory; here we perform interpretation in two stages, with the intermediate QLF level of structure interposed between phrase structure and logical form. At QLF level, comparative phrases are linked to the associated graded element, but *not* to the element they are compared against. A subsequent transformation of the QLF structure introduces links between compared elements, resulting in a form that can be converted into an LF. At the penultimate stage, we will have a higher-order four-place predicate, which takes the following arguments: the two contrasted elements, the way in which they are compared, and a parameterized "matrix" formula containing two "slots," one for the degree, and one for the contrasted element.

In Section 3.4 we give examples showing how our framework can be used to define the semantics of a variety of phrasal comparative complements, and also touch briefly on the question of ambiguity in phrasal comparative constructions. Finally, in Section 3.5 we say a few words about how the theoretical ideas presented above can be realized concretely within a unification-based grammar.

3.2 MONTAGOVIAN SEMANTICS FOR PHRASAL COMPARATIVES

We now proceed to show how the intuitive picture sketched out in Section 3.1 can be expressed formally in terms of Montague grammar; the technical idea is essentially inspired by the PTQ treatment of scoping phenomena[5] (Montague 1972; Dowty et al. 1982, pp. 203–215). There are three slightly different variants of the method, one for constituents of type t (clauses), and two for constituents of type $\langle e, t \rangle$ (VPs, APs). Since we are treating adverbials as properties of verb-objects, ADVPs will also fall into this second group.

For each variant, we have a rule-schema, parameterized by the syntactic type of the comparative complement; we call this constituent COMPLEMENT, and we also assume the existence of a "proform" that can stand for constituents of type COMPLEMENT. An example of such a proform will be written as **c-proform**$_i$ for some suitable **i**. We will also need a set of proforms to stand for determiners; we will write these as **q-proform**$_i$. Finally, we will assume a syntactic class COMP-OPERATOR, which will include expressions like **more, less, at least twice as much,** and so on. The denotation of an expression in COMP-OPERATOR will be a constant two-place relation between degrees.

We then add syntactic rules, to say that a constituent of type COMPLEMENT can be realized as **c-proform**$_i$, and that its denotation is a variable $\mathbf{c}_i$ of the appropriate type; similarly, a determiner can be realized as **q-proform**$_i$, and its denotation will be a variable $\mathbf{q}_i$ of the type associated with degree expressions. Since the semantic rules are all defined relative to a variable-substitution g, it naturally follows that the truth conditions for a phrase containing any of these proforms depends on the values assigned by g to the associated variables.

We are now in a position to define the syntactic rule-schema for composition of phrasal comparatives. We first take the simpler case, where the phrase is of clausal type; the input to the rule will be the following:

1) A phrase Φ of type S, containing occurrences of **c-proform**$_i$ and **q-proform**$_j$ for some **i** and **j**.
2) Two phrases Λ_1 and Λ_2 of type COMPLEMENT.
3) A phrase Ψ of type COMP-OPERATOR.

and produces as output the phrase **Comp**$(\Phi, \Lambda_1, \Lambda_2, \Psi)$ constructed by doing the following:

a) Replace **c-proform**$_i$ in Φ by the phrase Λ_1.
b) Replace **q-proform**$_j$ in Φ with the comparative operator Ψ.
c) Concatenate **than** or **as** (depending on Ψ) + Λ_2 to the end.
d) Perform necessary morphological operations (e.g. change **more big** to **bigger, less many** to **fewer,** etc.)

We let Φ' be the denotation of Φ, Ψ' the denotation of Ψ, etc. Since Φ contains instances of the pronouns **c-proform**$_i$ and **q-proform**$_j$, Φ' will contain corresponding free variables $\mathbf{c_i}$ and $\mathbf{q_j}$. It will thus make sense to abstract over these: for convenience's sake, we will refer to $\lambda \mathbf{c_i} \lambda \mathbf{q_j} \Phi'$ as Φ''. The associated semantic rule will then define the denotation of **Comp**$(\Phi, \Lambda_1, \Lambda_2, \Psi)$ to be the value of the expression

$$\Phi''(\Lambda_1', \{N \mid \Phi''(\Lambda_2', \{N' \mid \Psi'(N, N')\})\}) \qquad (*)$$

This formula is both central to the paper, and sufficiently obscure as to require some further explanation. To understand what it means in intuitive terms, we focus first on Φ''. This can be viewed in the usual way as a predicate with two argument slots: the first filled by an object denoting a phrase of type COMPLEMENT, and the second by an object denoting a degree.

We now work "from the inside out." We thus begin with the subformula $\{N' \mid \Psi'(N,N')\}$; remember that this is a generalized determiner (see Section 1.2). For example, if Ψ has the value **more,** it will be the determiner $\{N' \mid \mathbf{more}(N,N')\}$, or to express this in more usual terms **(less than** N). Working outward to the next level, the subformula $\Phi''(\Lambda_2', \{N' \mid \Psi'(N,N')\})$ is that formed by substituting the complement denotation Λ_2' in the first argument place of Φ'' and the determiner denotation $\{N' \mid \Psi'(N,N')\}$ in the second. It is consequently a condition on **N,** which means that $\{N \mid \Phi''(\Lambda_2', \{N' \mid \Psi'(N,N')\})\}$ is also a generalized determiner. The final formula is now constructed by once again substituting suitable values into the argument slots for ϕ'', these now being Λ_1' and $\{N \mid \Phi''(\Lambda_2', \{N' \mid \Psi'(N,N')\})\}$.

To summarize: what we have described here is a rule-schema that allows us to construct what could be called a *comparative S,* which *contrasts* two constituents of type COMPLEMENT.

A small extension now gives us corresponding schemas for comparative constituents of type $\langle e, t \rangle$ (this includes VPs, APs, and relative clauses); it seems that there are actually two slightly different versions. The most obvious case is the one in which contrasting takes place, as previously, between the comparative complement and some other constituent of similar type. Here, we can use the syntactic conditions for the rule above in unchanged form; since the type of Φ' is $\langle e, t \rangle$ rather than t, the semantic part of the rule has to be rewritten slightly, as the old definition would no longer make the condition on N a truth-value. We solve this problem by substituting in a value **x** in Φ'' (reducing it to a truth-value), and later abstract **x** away again at the outermost level. The denotation of **Comp**$(\Phi, \Lambda_1, \Lambda_2, \Psi)$ will then be

$$\lambda \mathbf{x}.\Phi''(\Lambda_1', \{N \mid \Phi''(\Lambda_2', \{N' \mid \Psi'(N, N')\}, \mathbf{x})\}, \mathbf{x}) \qquad (**)$$

An important special case, which we will make use of in Sections 3.4.4 and 3.4.6, occurs when the comparative complement is an NP, and the "contrast" is between it and the entity referred to by the abstracted variable in Φ. Sentences 14)–16) illustrate this. In each, we claim that contrasting is occurring within the italicized portion.

14) A man *taller than John* was chosen for the team.
15) Most people don't expect *to live as long as John (did).*
16) People *who travel to London as often as John* usually have season tickets.

One way to deal with these constructions would be to make them all equivalent with the case represented by 16): for relative clauses, it is reasonable to maintain that contrasting is between the complement, and a "gap" linked to the relative pronoun. On this account, 14) and 15) are regarded as derived from something like 14a) and 15a).

14a) A man *(who is) taller than John* was chosen for the team.
15a) Most people don't expect *(that they will) live as long as John.*

Although this approach is certainly also feasible, it seems technically simpler to dispense with the null pronouns and instead use the following semantic analogue of the idea, essentially a type-raising trick. We let Φ be a constituent of type $\langle e, t \rangle$, containing an occurrence of **q-proform**$_j$, Ψ a COMP-OPERATOR, and Λ_2 an NP; we define **Comp**$_1(\Phi, \Lambda_2, \Psi)$ to be the phrase formed by substituting Ψ for **q-proform**$_j$, and concatenating **than/as** Λ_2 to the end, and now want to define its denotation. We let $\mathbf{c_i}$ be a parameter ranging over NP denotations, **x** be a parameter ranging over individuals, and Λ_1' be the function $\lambda \mathbf{Q}.\mathbf{Q}(\mathbf{x})$; if we now define Φ'' to be the formula $\lambda \mathbf{c_i} \lambda \mathbf{q_j} \cdot \mathbf{c_i}(\Phi')$, the denotation of **Comp**$_1(\Phi, \Lambda_2, \Psi)$ is given by yet another variant on our basic formula, namely

$$\lambda \mathbf{x}.\Phi''(\Lambda_1', \{N \mid \Phi''(\Lambda_2', \{N' \mid \Psi'(N, N')\})\}) \qquad (***)$$

Readers who wish to see examples of derivations using the three interpretation schemas defined above may now want to turn to Section 3.4; when they have returned, we will proceed to describe how our theoretical picture can be converted into one that is more readily implementable.

3.3 A COMPUTATIONAL VERSION

The analysis we have just presented gives a simple and compact formalization of our theory of phrasal comparatives, but it is still not easily implementable as it stands. The reasons for this are exactly the same as those applying to the original PTQ treatment of scoping phenomena; essentially, the rules have insufficient syntactic motivation. However, the solution (as far as scoping goes) is by now standard, and has been successfully implemented in a number of well-known systems (e.g., Pereira 1983; Lesmo 1985; Grosz et al. 1986; Alshawi et al. 1989); the methodology can originally be traced back to Woods (1978).[6] The trick is to split interpretation into two stages, the first linked to the syntax and the second to the semantics; mediating between these, we have an intermediate level of representation, the *quasi-logical form.* This is produced compositionally from the syntax, and then subjected to rewriting rules before being converted into the final logical form. Normally, these rewriting rules formalize scoping transformations; here, we will also use them to describe the interpretation of nonclausal comparison. In Section 1.1, we summarized briefly the basic properties we demanded of the QLF level of formalism. We now describe those features that are specially concerned with comparative constructions.

We start by defining the various kinds of nodes we will need in the QLF. These will be the following.

Node name	*Represents*
c-complement	Comparative complement.
c-operator	Comparative operator.
c-degree	Comparative degree; includes the comparative complement and comparative operator as subnodes.
q	Placeholder; the **q-proform** of the Montagovian analysis.
c	Placeholder; the **c-proform** of the Montagovian analysis.
comparison	Corresponds to **Comp**(Φ, Λ_1, Λ_2, Ψ) and **Comp**$_1$(Φ, Λ_2, Ψ) of the Montagovian analysis. The part of the tree under the node corresponds to Φ. The subnodes are: the "contrasted material" (Λ_1), the comparative complement (Λ_2), the comparative operator (Ψ), and pointers to the relevant **q** and **c** nodes (**q_i, c_j**)

The fundamental idea should be fairly easy to grasp, if the reader has understood the previous section; the QLF is successively transformed through a number of stages, until it is in a form consonant with the Montagovian analysis of Section 3.2. We go through the stages in turn, illustrating each of them schematically; in Sections 3.4.1 and 3.4.2, we give examples of their application.

(Stage 0) In the original QLF, the comparative complement and comparative operator turn up as **c-complement** and **c-operator** nodes, modifying the **c-degree** node representing the comparative degree; the **c-complement** is not yet associated with the constituent it is contrasted against (see Figure 1).

(Stage 1) Rewriting rules replace the **c-degree** with a **q** placeholder, and save the **c-complement, c-operator,** and associated **q** in a separate location; we call this the "comparative store," by analogy with Cooper's "quantifier store" (Figure 2; Cooper 1983; Engdahl 1985).

(Stage 2) The contents of the "store" are moved upwards in the tree; at each higher node **N,** attempt to do one of i), ii) or iii). (The second and third alternatives are special cases) (Figure 3).

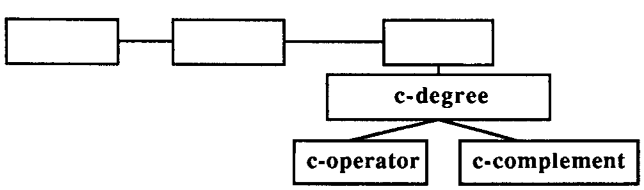

Figure 1 Original QLF

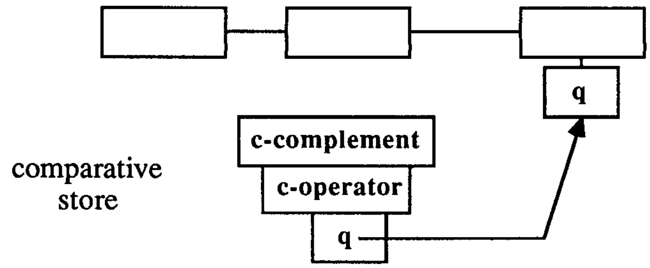

Figure 2 An item is placed in the "store"

i) Find a suitable constituent below *N* to contrast the **c-complement** against. If such a constituent is found, replace it by a **c** placeholder, and insert a **comparison** node in the tree above *N*. The **comparison** node is constructed from the following components: the **c-complement,** the contrasted constituent, the **c-operator,** and the **q** and **c** placeholders.

ii) (This is to deal with the third schema from Section 3.2) If *N* is an **abstraction** node (representing a constituent of type $\langle e, t\rangle$), insert a **c** node above *N*, a **comparison**

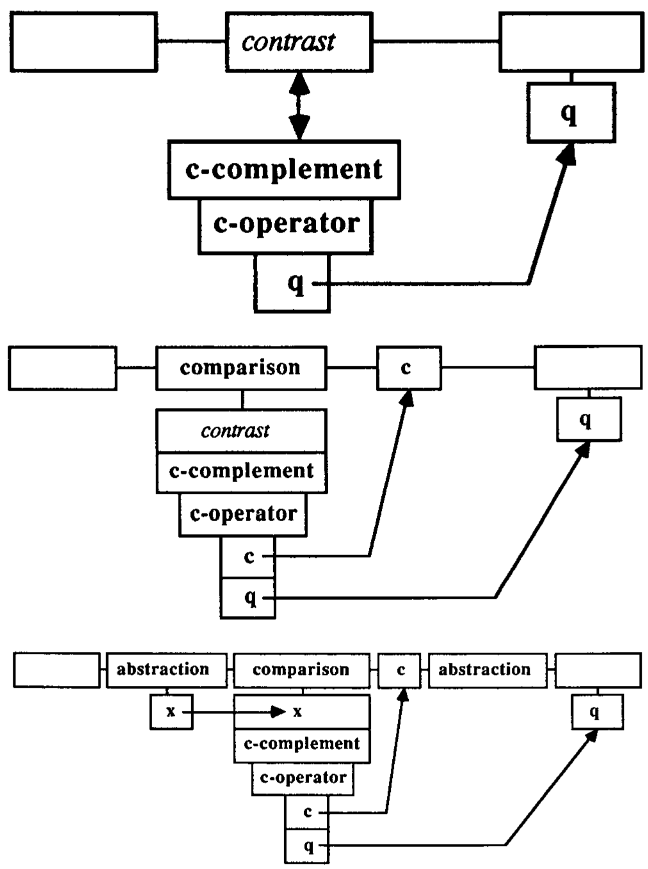

Figure 3 (a) A node has been located, which can serve as a contrast to the comparative complement. (b) The contrast node is replaced with a **c** placeholder, and a **comparison** node is inserted. (c) Special case for third schema in 3.2.

node above the **c**, and a further **abstraction** node above that. The **comparison** node is constructed from the **c-complement, c-operator, c** and **q;** the "contrast" component is that corresponding to a pronoun with associated variable **x,** where **x** is the abstracted variable in the new **abstraction** node.

iii) (See Section 3.4.3) If *N* is a constituent of type *t* or ⟨*e*, t⟩, and **c-complement** is an S-OPERATOR, insert an **s-operator** node above *N,* and a **comparison** node above that; the body of the **s-operator** node will be a **c** placeholder. The **comparison** node has the same structure as that in case i), except that the "contrasted material" slot contains the special value **it-is-true-that.**

(Stage 3) The tree is now in a form where the Montagovian analysis can be used directly; one of the formulas (*), (**), or (***) is applied to yield the logical form, the choice depending on which alternative was chosen in the preceding step. In the final stage, the portion of the QLF under the **comparison** node is duplicated twice, and the **q** and **c** placeholders instantiated in each copy in an appropriate manner (see Figure 4).

3.4. APPLICATIONS OF THE THEORY

3.4.1 NP- AND PP-COMPARATIVES

Phrasal comparative constructions in which the comparative clause is an NP or PP appear to have fundamentally the same structure, and we shall consequently consider them under one heading. We will present a typical example, first using the Montagovian framework, and then reworking it in the computational one.

The following table represents a semantic derivation for the sentence **each woman received more roses from toronto than from london;** we will analyze this as being composed of the subject **each woman,** and the comparative VP **received more roses from toronto than from london.** We will use an instance of the second schema from 3.2, with COMPLEMENT equal to PP, and accordingly write **c-proform**$_i$ as **from-there**$_i$**. Received** will be treated as a verb taking three arguments, with **received'(x, y, z)** interpreted as "**x** received **y** from **z.**" For example, **{16}(λr.rose'(r) & red'(r), λ r.received' (amelie', r, toronto'))** will correspond to **amelie received 16 red roses from toronto,** and **(no'(rose', λr.received'(amelie', r, graham'))** will correspond to **amelie received no roses from graham.**[7] (see Table 1).

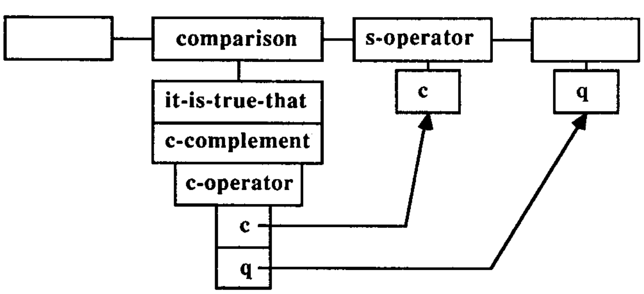

Figure 4 Special case for 3.4.3.

Table 1. Montagovian Analysis of Contrasted PPs

1. each woman	λP.each'(woman',P)
2. from toronto	λP.P(toronto')
3. from london	λP.P(london')
4. q-proform$_1$ roses	λP.q_1(rose',P)
5. received	λxλyλz.received'(x,y,z)
6. from-there$_1$	l_1
7. received from-there$_1$	λxλy.l_1(λz.received'(x,y,z))
8. received q-proform$_1$ roses from-there$_1$*	λx.q_1(rose',λy.l_1(λz.received'(x,y,z)))
9. more	λxλy.more'(x,y)
10. received more roses from toronto than from london	λx.{N I {N' I more'(N,N')} (rose',λy.received'(x,y,london'))} (rose',λy.received'(x,y,toronto'))
11. each woman received more roses from toronto than from london	each'(woman',λx.{N I {N' I more'(N,N')}(rose', λy.received'(x,y,london'))} (rose',λy.received'(x,y, toronto')))

*Remember that since q_1 is a variable ranging over determiner denotations it takes two arguments, which should both be of the type ⟨e,t⟩. Similarly, l_1 (an NP denotation) takes a single argument of the same type.

Figure 5 presents in outline the analysis of the same sentence in the computational framework of Section 3.3, and should be compared with Figs. 1–4.

3.4.2 CN COMPARATIVES

Comparative constructions of the type illustrated in 17a) have been the object of considerable controversy. The orthodox position was that they were "parallel" constructions: 17a) would thus be a reduced form of 17b).

17a) More women than men read "Playgirl."
17b) More women read "Playgirl" than men read "Playgirl."

Pinkham, as we have already mentioned, gives good reasons for supposing that this is not the case, and that the construction is in base generated phrasal (pp. 121–123). We now show how her analysis can be recast in our framework: once again, we present our derivation first in terms of Montague grammar and then in the computational version.

In Montague grammar, we instantiate the first schema from Section 3.2, this time with COMPLEMENT equal to CN; in the interests of readability we write **c-proform**$_i$ as **of-that-kind**$_i$. The derivation appears in Table 2.

The corresponding treatment in the computational framework is another direct application of the method from Section 3.3, and appears in Figure 6.

3.4.3 "S-OPERATOR" AND "V-P-OPERATOR" COMPARATIVES

Our next type of construction is illustrated in 18a), 18b) and 18c).

18a) Mary had more friends than *John had expected.*

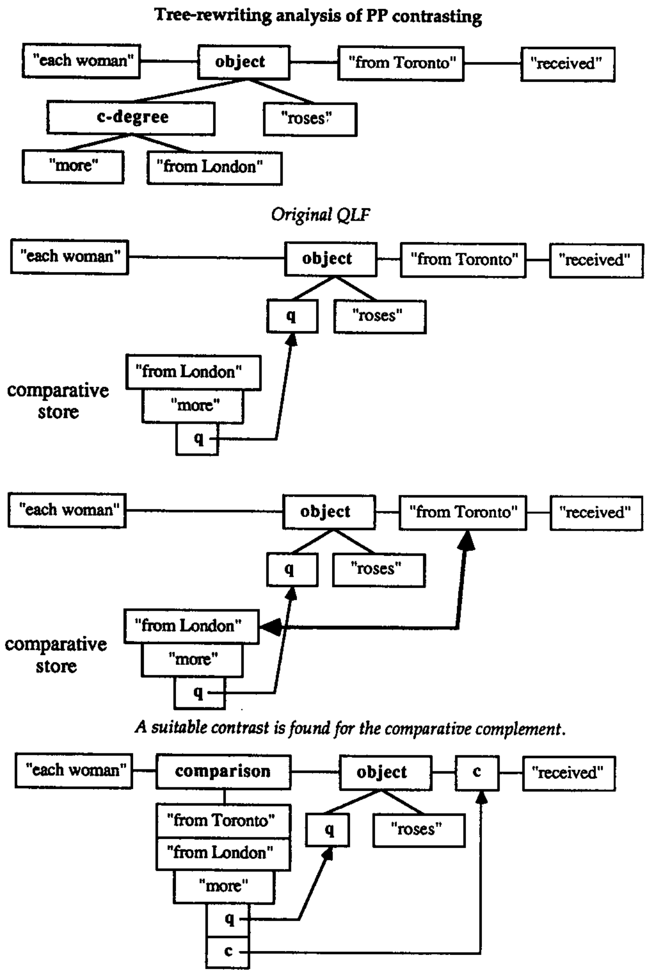

Figure 5 Tree-rewriting analysis of PP-contrasting

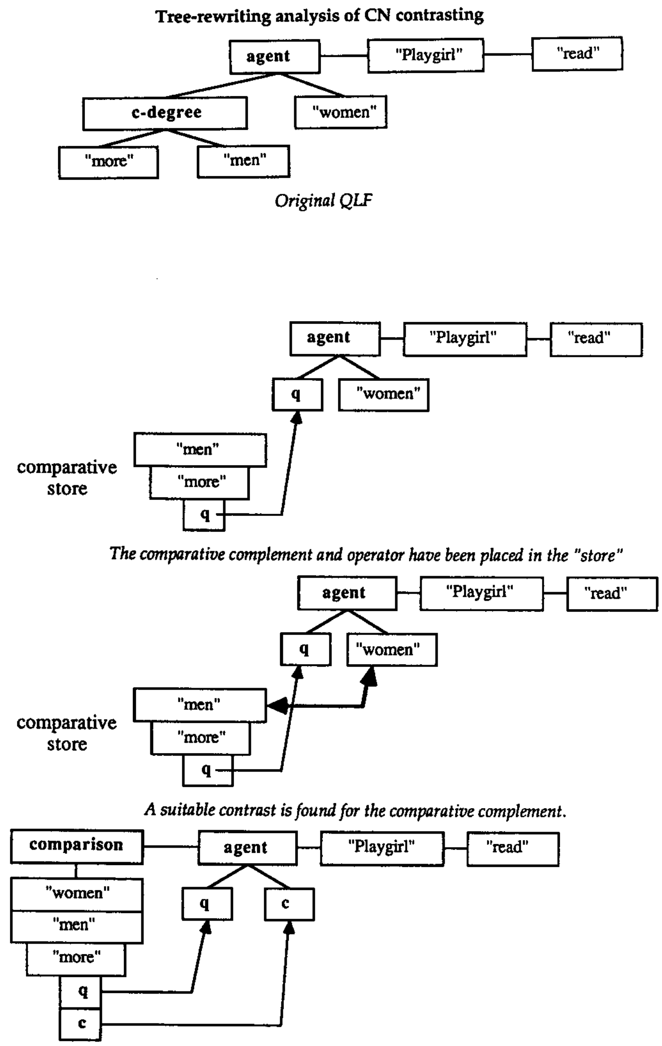

Figure 6 Tree re-writing analysis of CN-contrasting

18b) Most people ate more biscuits than *Mary said.*
18c) John's yacht was longer than *I thought.*

To handle examples like these within our framework, we need a syntactic representation that does not involve ellipsis. Our solution is to introduce a syntactic constituent that we call an S-OPERATOR: by this, we mean a constituent of the form S/S-COMP, that is to say, a clause with a missing sentential complement. The italicized portions of the sentences above are accordingly deemed to be S-OPERATORs, and the contrast is with an implicit trivial S-OPERATOR that we can write as **(it is true that)**. Note that there is an unbounded dependency here between the S and its missing constituent; we say a little more about this in Section 3.5.

Table 2. Montagovian Analysis of CN Comparative

1. woman	woman′
2. man	man′
3. "Playgirl"	λQ.Q(playgirl′)
4. read "Playgirl"	λx.read(x,playgirl′)
5. q-proform$_1$ of-that-kind$_1$	$\lambda Q.q_1(l_1,Q)$
6. q-proform$_1$ of-that-kind$_1$ read "Playgirl"	$q_1(l_1,\lambda x.read(x,playgirl'))$
7. more women than men read "Playgirl"	{N I {N′ I more′(N,N′)} (man′,λx.read(x,playgirl′))} (woman′,λx.read (x,playgirl′))

[(reads(x,y) is to be understood as "x habitually reads y")]

An example in the Montagovian framework is shown below; we use the second schema from Section 3.2, with COMPLEMENT instantiated to S-OPERATOR, and write **c-proform$_i$** as **attitude$_1$**. There are in fact two tricky technical points here, which we are forced by space limitations to gloss over.

First, we are assuming that it is possible when introducing complex categories of the form A/B to derive the associated semantic rule from those for the component categories A and B. Since it is at any rate fairly obvious in the present case that this is feasible (that is, that we can justify the semantic value of the phrase **mary said**), we shall not discuss the matter further. Second, we allow application of the denotation of an S-OPERATOR to the denotation of a VP, which formally is of the wrong type. Here, our reasoning is that since an S-OPERATOR denotes a function σ from propositions to propositions, it will naturally induce a function σ': $\lambda x.P(x) \rightarrow \lambda x.\sigma(P(x))$ from VP-denotations to VP-denotations; we will identify σ and σ' (see Table 3).

It is interesting to see that Krifka independently arrives at exactly the same conclusion; he writes our "it is true that" as the operator FAKT, and for example analyses 19a) below as 19b):

19a) Anna ist reicher als Otto glaubt.
Anna is richer than Otto thinks.
19b) KOMP(FAKT,Otto glaubt,λd.[Anna ist d-reich]

Analogously with the "S-OPERATOR" introduced above, we can also define a "VP-OPERATOR" to be a constituent of the syntactic category S/VP, i.e. something that can combine with a VP complement to yield an S; note that this includes subject NPs. We can then give a similar account for sentences like those illustrated in 20a) to c):

20a) Mary opened more bottles than *John told her to.*
20b) Mary scored more points than *I normally can.*
20c) Mary needs to get a better result than *she's so far managed.*

We sketch a Montagovian derivation for the first of these: here we use the first schema from 3.2, this time with COMPLEMENT instantiated to VP-OPERATOR. Note that we give **mary** wide scope over the whole sentence, so as to get the pronoun bindings right (see Table 4).

Readers who wish to understand the implementation details for the computational version are referred to the relevant program code in Appendix 2. Parse-tree and logical forms are given in example sentence 8 in Appendix 3.

3.4.4 CONTRASTING INSIDE ADJECTIVAL AND ADVERBIAL PHRASES

So far, all the examples of contrastive comparatives that we have shown have involved using one of our two schema to construct a phrase that is either an S or a VP. In this section, we will consider sentences where the contrasting operation is used internally within adverbial and adjectival phrases: Pinkham quotes constructions of this kind as some of the clearest instances of phrasal comparatives. We start in 21)–23) by giving some typical examples; 21) and 22) are taken from Pinkham (1985).

21) A man *taller than John* was chosen for the team.
22) He ran *faster than the world record.*
23) John needs a *spanner bigger than the No. 4.*

In each of these sentences, contrasting appears to occur within the italicized phrase; in 21), John need not have been chosen for the team, in 22) the world record certainly cannot run, and in 23) John presumably does not need the No. 4. We would, however, derive all these conclusions if contrasting were at VP or clausal level, since we would then copy the verb.

In accordance with Pinkham's suggestions, our approach will rather be to use a direct interpretation method based on predicate copying; we will in fact use the third schema from Section 3.2, defined by formula (***). This applies in a straightforward fashion to adjectival constructions like those in 21) and 23). Pinkham notes that a similar treatment is also possible for sentences like 22) if we take *fast* to be a predicate applying to the "running" event, although we imagine that some people may find this move unattractive.

In Table 5, we present a sketch of a Montagovian analysis for 21). Note that "Q tall" will be represented as $\lambda x.tall(x, Q)$, where Q is a generalized determiner, and that formula (***) is used when deriving 5 from 3 and 4.

3.4.5 CONTRASTED PREPOSITIONS

The types of contrastive comparatives described in Sections 3.4.1 to 3.4.4 appear to account for the vast majority of those that occur in practice.[8] There are also a few rare cases, one of which we here touch on briefly. As a final example, we consider the case of contrasted prepositions, as shown in 24) below.[9]

24) There are more points for the proposal than against.

Table 3. Montagovian Analysis of "S-OPERATOR" Comparative

1. most people	λQ.most'(person',Q)
2. mary said	λQ.said'(mary',Q)
3. (it is true that)	λQ.Q
4. ate	λxλy.ate'(x,y)
5. ate so-many₁ biscuits	$\lambda x.(q_1(\text{biscuit}',\lambda y.\text{ate}'(x,y)))$
6. attitude₁ ate so-many₁ biscuits	$a_1(\lambda x.(q_1(\text{biscuit}',\lambda y.\text{ate}'(x,y))))$
7. (it is true that) ate more biscuits than Mary said	λx.{N | said'(mary',{N' | more'(N,N')}(biscuit',λy.ate'(x,y))} (biscuit',λy.ate'(x,y)))
8. most people ate more biscuits than Mary said	most'(person',λx.{said'(mary', {N' | more'(N,N')} (biscuit',λy.ate'(x,y))} (biscuit',λy.ate'(x,y))))

Table 4. Montagovian Analysis of "VP-OPERATOR" Comparative

1. mary	λQ.Q(mary')
2. he$_1$	$\lambda Q.Q(x_1)$
3. john told he$_1$ to	$\lambda Q.\text{told}'(\text{john}',x_1,Q(x_1))$
4. opened so-many$_1$ bottles	$\lambda x.(q_1(\text{bottle}',\lambda y.\text{opened}'(x,y)))$
5. vp-op$_1$ opened so-many$_1$ bottles	$c_1(\lambda x.(q_1(\text{bottle}',\lambda y.\text{opened}'(x,y))))$
6. he$_1$ opened more bottles than john told he$_1$ to	$\{N \mid \text{told}'(\text{john}',x_1,\{N' \mid \text{more}'(N,N')\}(\text{bottle}',\lambda y.\text{opened}'(x_1,y))))\}(\text{bottle}',\lambda y.\text{opened}'(x_1,y))$
7. mary opened more bottles than john told her to	{N | told'(john',mary',{N' | more'(N,N')}(bottle',λy.opened'(mary,y))))} (bottle',λy.opened'(mary,y))

Table 5. Montagovian Analysis of Contrasting in AP

1. was chosen for the team	was-chosen′
2. player	player′
3. john	λQ.Q(john′)
4. q-proform$_1$ tall	λx.tall′(x,q_1)
5. taller than john	λx.tall′(x,{D | tall′(john, {D′ | more′(D,D′)})})
6. player taller than john	λx.player′(x)&tall′(x,{D | tall′(john,{D′ | more′(D,D′)})})
7. a player taller than john	λQ.a′(λx.player′(x)&tall′(x,{D | tall′(john,{D′ | more′(D,D′)})}),Q)
8. a player taller than john was chosen for the team	a′(λx.player′(x)&tall′(x,{D | tall′(john,{D′ | more′(D,D′)})}), was-chosen′)

These can be treated in a completely straightforward manner in the computational version from Section 3.3; the key step will be the replacement of the ARG-TYPE field of a suitable NP denotation with a C-placeholder. A Montagovian treatment of the construction is of course somewhat problematic with the framework we have so far adopted, since there is no sensible way to quantify over ARG-TYPE of anything similar; some people may see in this fact an argument against Montagovian formalisms. (Parse-tree and logical form in example sentence 12 in Appendix 3).

3.4.6 AMBIGUITIES IN CONTRASTIVE CONSTRUCTIONS

It is well known that phrasal comparative constructions can easily give rise to ambiguities: sentences 25)–27) illustrate three typical possibilities. Here, we will discuss how our system fares when attempting to explain these.

25) Nobody talked more to John about Carol than Mary.
26) Few men give their wives more presents than George (does).
27) John intends to buy more books than Mary.

We claim that most speakers find all these sentences ambiguous, the first in three ways, and the others in two. The simplest case is 25), where we can paraphrase the various readings as 25a–c):

25a) Nobody talked more to John about Carol than Mary did.
(Mary talks a lot to John about Carol.)

25b) Nobody talked more to John than to Mary about Carol.
(John is less popular than Mary when people want to hold conversations about Carol.)

25c) Nobody talked more to John about Carol than they did about Mary.
(Everyone prefers discussing Carol to discussing Mary when they talk with John.)

In terms of the discussion from Section 3.4.1, ambiguity arises naturally as a result of the uncertainty concerning the identity of the element that "Mary" is intended to be compared with. By successively choosing the NPs "Nobody," "John," or "Carol," the reader can easily satisfy herself that each of the readings above can be derived.

Sentence 26) also appears to be fairly clear: here, the distinction is essentially between the "strict" and "sloppy" readings, which can be represented as 26a) and 26b).

26a) Few men give their wives more presents than George gives them.

26b) Few men give their wives more presents than George gives his own wife.

On a Montagovian account, we will get 26a) if the pronoun **their** is still unbound in the denotation of the VP **give their wives q-proform$_1$ presents;** if, on the other hand, it is bound to the variable representing the subject, we will get 26b).

The hardest case definitely seems to be sentence 27). Here we have at least two readings, which we represent as 27a) and 27b):

27a) John intends to buy more books than Mary will buy.
(John's plan is that, no matter how many books Mary has, he will buy more.)

27b) John intends to buy more books than Mary intends to buy.
(John and Mary both plan to buy specific numbers of books, that number being in John's case greater.)

We can obtain the first of these using formula (***) from Section 3.2, as can be seen from the schematic derivation in Table 6.

Unfortunately, there is no very satisfactory way to derive the wide scope reading 27b). The problem is that the comparative degree will end up in our account with lower scope than the modal operator**wants,** irrespective of the level at which we carry out the contrasting operation; to derive 27b), however, it needs to have scope over the whole clause. Although one can define an *ad hoc* rule that raises the scope of the comparative quantifier, this goes very much against the spirit of our treatment; a slightly more attractive possibility may be to introduce alternative "wide-scope" versions of formulas (*), (**) and (***) from 3.2. For example, (*) would become

$$\exists N,N'.\Phi''(\Lambda_1',\{N\})\&\Phi''(\Lambda_2',\{N'\})\&\Psi'(N,N')$$

The problem is of course to find conditions defining when the wide-scope formulas are applicable. One idea might be

Table 6. "Narrow Scope" Interpretation of Sentence 27

1. buy q-proform$_1$ books	λx.q_1(book′,λy.buy′(x,y))
2. buy more books than mary	λx.{N | {N′ | more(N,N′)} (book′,λy.buy′(mary,y))} (book′,λy.buy′(x,y))
3. wants to buy more books than mary	λx.want′(x,{N | {N′ | more (N,N′)}(book′,λy.buy′(mary,y))} (book′,λy.buy′(x,y)))
4. john wants to buy more books than mary	want′(john; {N | {N′ | more(N,N′)} (book′, λy.buy′(mary,y))} (book′, λy.buy′(x,y)))

to allow their use when the comparative occurs in an embedded S or VP (as it does here), but the whole topic clearly needs more study.

3.5 IMPLEMENTATION ISSUES

We first discuss the syntactic level; the main implementation problems here are caused by various kinds of long-range dependencies. Apart from normal "WH-movement" dependencies in questions and relative clauses, we have the following:

1. Dependency between comparative head and comparative complement. Normally, the head is separated from its associated complement in surface syntax, even though it conceptually forms a constituent with it.
2. Dependencies associated with missing complements in S- and VP-operators, which can be arbitrarily deeply nested. These appear to be very similar to WH-movement dependencies.
3. "Verb anaphora" dependencies: as is well known, certain verbs (in English *do, be, can,* etc.) can act anaphorically for verb phrases. We treat a comparative complement consisting of an NP together with one of these verbs as essentially equivalent with the NP on its own, except that it is forced to contrast against an NP which is a subject in a clause whose main verb is a suitable antecedent. The verbal antecedent relationship also appears to lead to an unbounded dependency.
4. "Quantifier binding" in clausal comparatives; we discuss this in Section 4.1.

All these forms of long-range dependency are handled by application of "threading" (Pereira 1983; Kartunnen 1986). Properly speaking, each distinct type of dependency ought to be associated with a distinct feature, which will be present in relevant constituents; by unifying features in different constituents against each other, information is propagated through the tree. In the XG-grammar in Appendix 2, we have cheated a little, by using the "extraposition list" feature to handle all of 1, 2, and 4 above. Although this can potentially lead to problems when dependencies "cross," we felt that the gain in simplicity made this compromise worthwhile. It is a simple matter to reorganize the grammar using different features.

The semantic part of the grammar follows very closely the discussion in Section 3.3. The syntax-tree is first transformed into QLF form by the predicate syn_to_qlf/2; the QLF is then searched to find potential comparisons. That is, it first tries to extract a c-complement and store it, then tries to find a suitable contrast and finally constructs the comparison node and inserts it.

4. Clausal Comparatives

We borrow most of the basic principles in our treatment of clausal comparatives from Pinkham. Firstly, just as with phrasals, we assume that the comparative clause is normally a modifier to a degree expression in the main clause; this assumption is also shared by Bresnan, though not by Friedmann, who places the two clauses at the same level.

Secondly, we assume that the "missing material" in the comparative clause is a null degree expression; this may possibly occur together with a null copy of the material modified by the head degree, which can be an adjective (example 29), an adverb (30), or a CN (31, 32). If the head degree is modifying an adjective in nonpredicate position, then it is obligatory to copy the CN (32). This idea is more or less borrowed from the discussion in Pinkham (pp. 33–40). Adapting Pinkham's notation slightly, we write a null degree as Q, a null adjective or adverb as PRO_A, and a null CN as PRO_{CN}.

Our third principle is that the denotation of the COMP-OPERATOR is realized as a relation between the head degree and the Q, and appears conjoined with the comparative clause; this means that the Q originally will have wide scope over the clause it occurs in. Examples 28)–32) illustrate our analysis: in each, the first item is the sentence, the second a schematic representation of the QLF, and the third a schematic representation of the logical form. In the QLF, we enclose the comparative degree expression in parentheses.

28) John bought more books than Mary bought records.
28a) John bought (more than Mary bought Q records) books.
28b) John bought {N | Mary bought {N′} records & more (N,N′)} books.

29) John was happier in London than he was in New York.
29a) John was (more than he was Q PRO_A in New York) happy in London.
29b) John was {D | he was {D′} happy in New York & more(D,D′)} happy in London.

30) John runs faster than Mary swims.
30a) John runs (more than Mary swims Q PRO_A) fast.
30b) John runs {D | Mary swims {D′} fast & more (D,D′)} fast.

31) John bought more books than Mary could carry.
31a) John bought (more than Mary could carry Q PRO_{CN}) books.
31b) John bought {N | Mary bought {N′} books & more(N,N′)} books.

32) John bought a more expensive vase than Mary bought.
32a) John bought a (more expensive than Mary bought Q PRO_A PRO_{CN}) vase.
32b) John bought a {D | Mary bought a {D′} expensive vase & more(D,D′)} expensive vase.

It is fairly straightforward to implement a scheme of this kind in a framework that can handle long-range dependencies; in Section 4.1, we describe in more detail how this can be done. Before doing so, however, we take a brief look at the theoretical issues raised by our first and third assumptions.

With regard to the first assumption, suppose that we chose Friedmann's alternative and allowed the main and comparative clauses to be parallel. For instance, this would give example 28) above a logical form something like 28c):

28c) ∃ N,N′. John bought N books & Mary bought N′ records & more(N,N′).

At first glance, this looks simpler. The problems arise when we introduce quantified NPs: thus if we replace *John* with something like *no one*, we are obliged to have *more*(N,N′) within the scope of the quantifier representing the new NP. Of course, it is possible to require that such NPs always be given wider scope than the whole sentence (which is what Friedmann does), but to do so ignores the fact that an NP that has higher scope than the NP in which the comparative degree occurs normally has higher scope than the comparative degree as well. Making the comparative clause a modifier to the degree is from this point of view a more principled solution.

We now consider our third assumption: what scope should the "inner" degree receive, and where should the condition linking the two degrees appear? Here, there are at least two reasonable choices. Alternative A, which is the one we have chosen, is to give the inner degree wide scope over the whole comparative clause, and conjoin the condition; alternative B is to give the inner degree its "natural" scope in the comparative clause, and make the relation between the two degrees a condition on it. This would give 28) the logical form shown in 28d) below. We marginally prefer A, but the issues involved are sufficiently unclear to merit further discussion.

28d) "John bought {N|Mary bought {N′|more(N,N′)} records} books.

The drawback to A is, once again, that quantified NPs can cause problems: thus in a sentence like 33) we do not want the interpretation 33a) where most Swedes have traveled abroad the same number of times.

33) Sven has visited Japan more times than most Swedes have traveled abroad.

33a) Sven has visited Japan {N|(most Swedes have traveled abroad N′ times) & more(N,N′)} times.

To get around this, we have to postulate a rule that gives *most Swedes* wide scope over N′; this is not necessary if we choose B, where the natural scoping is 33b):

33b) Sven has visited Japan {N|(most Swedes have traveled abroad {N′|more(N,N′)} times)} times.

If we are only dealing with extensional sentences, examples like 33) would thus make it seem that B was preferable. However, serious problems arise with the well-known example 34), due to Russell:

34) I thought your yacht was longer than it is.

Here we are more or less forced to choose an interpretation that gives the "inner" degree wide scope, i.e. 34a)

34a) Your yacht is D′ long & I thought that your yacht was {D|more(D,D′)} long

If we use alternative A, it is possible to define scoping rules that produce this interpretation. However, this appears to be out of the question with alternative B; if the comparative clause is given wide scope, the degree comparison *more(D,D′)* will be outside the scope of D. Although this conclusion seems inescapable, we are still somewhat reluctant to recommend a solution that is motivated entirely by a type of example that essentially never occurs in normal applications; readers interested in concrete implementation may well consider that the practical problems involved in formulating suitable scoping rules for alternative A outweigh the theoretical loss of generality. It would appear that there is room for further research here: the reader is warned that the issues at stake are far from simple, as witness for example, Larson (1988).

4.1 IMPLEMENTATION ISSUES

Our proposed treatment of clausal comparatives can be implemented fairly simply in a framework that contains a mechanism for handling long-range dependencies; the basic idea was first described in Klein (1980) in the context of an early version of GPSG, although we have modified it considerably. As already mentioned in Section 3.5, we will assume some kind of gap-threading idea. Pinkham observes (p. 81) that Quantifier Binding respects island constraints; the most obvious way to ensure this is to propagate it using the same mechanism as is used to handle WH-movement.

A type of dependency we have already seen is that between the head of the comparative and the associated comparative clause; in this case, the problem is to pass rightward the information that a comparative clause is being looked for, which will end up as a modifier of a comparative degree expression somewhere to its left. Since the possible nature of the bound null proforms in the comparative clause depends on the structure of the associated head, the two types of dependency are interrelated; just how, we explain below. We will call the feature associated with the first kind of dependency the Q feature, and that associated with the second the **C_complement** feature. The values of both of these will as usual be pairs of the form ⟨**Gaps_in**, **Gaps_out**⟩.

Let us now examine in detail how the scheme works. A comparative degree expression on a determiner, adjective, or adverb causes a non-null value of the **C_complement** feature to move rightwards; that is, a non-null **Gaps_out** on the degree expression successively gives rise to non-null **Gaps_in** values on other constituents to its right. This continues until the **C_complement** feature is "absorbed" by a non-null comparative complement. This will be a comparative complementizer, followed by a clause with suitable **C_gaps** features.

The difficult part of the business is getting the right **Gaps_in** value for the Q feature on the comparative clause; in other words, making sure that the type of null proform that occurs in the comparative clause is one that matches the head of the comparison. To begin with, this means that the **C_complement** feature must contain information about the permissible values for the associated Q; the rules for comparative degree constituents then need to specify this information in a suitable way. The following rules are those we have implemented: we write **H** for the head of the comparison and **B** for the bound null proforms, and enumerate four separate cases.

1. If **H** is a determiner, then **B** is either
a) a Q
b) an NP whose determiner is a Q, and whose CN is a copy of the CN in the NP where **H** occurs.
2. If **H** is modifying an adjective *A* in predicate position, then **B** is a Q modifying a copy of *A*.
3. If **H** is modifying a verbal adverb *A*, then **B** is a Q modifying a copy of *A*.
4. If **H** is modifying an adjective occurring in prenominal position in an NP *N*, then **B** is a copy of *N*, where **H** has been replaced by a Q.

The rules above are a considerable simplification of Pinkham's, and there are in particular two major omissions. First, we copy constituents, instead of introducing bound proforms of type PRO_A and PRO_{CN}. This usually makes no difference, but becomes significant in sentences like 35): if we copy, getting 35a), we do not guarantee that the two occurrences of *them* refer to the same set.

35) John gave me more of them than I managed to eat.
35a) John gave me more of them than I managed to eat (Q many of them).

Secondly, and more critically for practical purposes, we have ignored the important distinction Pinkham draws between "narrow-scope" and "wide-scope" clausals (p. 86–91). This was not done for any principled reason, but simply for lack of time; implementing it in a satisfactory way is an important part of the planned continuation of our research.

5 Other Issues

This concludes our analysis, with the exception of some minor points that we have postponed until now in the interests of expositional clarity. The first concerns noncoreference of compared objects, the second the question of whether "before" and "after" can be regarded as comparatives.

5.1 NONCOREFERENCE OF COMPARED OBJECTS

Although speaker judgments tend not to be completely unanimous, there is a strong tendency to assume that comparison is always between different objects. This point becomes important when answering questions like 36).

36) Has any king ruled as long as Gustav V?

Most people find "Yes" meaning "Yes, Gustav V did" very misleading. To correct this, we make a slight adjustment in the analysis we have so far been using, so that an appropriate inequality is added when the logical form is produced from the reshaped quant tree. With this alteration 37) may be judged true even if John is present, and 38) may correctly be said of one of Mary's articles.

37) Nobody here has read as many books as John.
38) Mary has never written an article that is as bad as this one.

5.2 "BEFORE" AND "AFTER"

In English, the words *before* and *after* can display several of the features associated with comparatives. For example, sentences like 39) and 40) would appear to have interpretations which could be represented as 39a) and 40a):

39) John arrived before me.
39a) John arrived at a time T, T such that I arrived at a time T′, T before T′.

40) Henry VIII married Anne Boleyn after Catherine of Aragon.
40a) Henry VIII married Anne Boleyn at a time T, T such that he married Catherine of Aragon at a time T′, T after T′.

Given examples like these, it is tempting to conclude that *before* and *after* are the "comparative forms" of adverbials that could be represented as "at-early-time" and "at-late-time." (Indeed, we made exactly this proposal in an earlier paper). However, it must be noted that, even if this analysis is correct, these adverbials would appear to be subject to certain restrictions with regard to the comparative complements they can take. For example, examples 41a)–42a) are at best dubious, and should be contrasted with the correct 41b)–42b):

41a) *John arrived before I expected.
41b) John arrived earlier than I expected.

42a) ?Mary left before she needed to.
42b) Mary left earlier than she needed to.

It thus seems uncertain whether these words should be regarded as a special sort of comparative, or as belonging to a separate class of their own. The best way to resolve this question might perhaps be to consider cross-linguistic data; if it turns out that there are other languages that allow a full range of comparative complements to the analoguous words, it would presumably make sense to hypothesize that this was originally the case for English, and that the missing constructions have simply fallen into disuse. Lacking at the moment any such evidence, we refrain from further speculation on this subject.

6 DATA FROM CORPUS ANALYSES

To give our claims concerning relative frequencies of the various constructions some substance, we here give the results of a small corpus analysis. We took the texts of one English and one Swedish novel (Agatha Christie's . . . *And Then There Were None;* Selma Lagerlöf's *Kejsaren av*

Table 7. Data from Corpus Analysis

Type of construction	No. of examples	Frequency
1. Discourse comparative:	**37**	*33.9%*
2. Clausal:	**7**	*6.4%*
3. Contrastive:	**45**	*41.3%*
NP:	*12*	*11.1%*
NP + "anaphoric verb":	*4*	*3.7%*
PP/Adverbial:	*10*	*9.1%*
S-operator:	*5*	*4.6%*
VP-operator:	*2*	*1.8%*
Internal contrasting in AP:	*11*	*10.1%*
Other:	*1*	*0.9%*
4. Determiner/Phrasal*:	**6**	*5.5%*
5. Verb of change + comparative*:	**6**	*5.5%*
6. Inte *comparative* **än att...*:**	**4**	*3.7%*
7. Other:	**4**	*3.7%*
Total:	**109**	

*Constructions like "more than two" or "less than half."
*Constructions involving verbs like "grow" or "become" together with a comparative, as in "The room grew more cheerful" or "The road became harder." The problems involved in giving a correct semantics to such sentences derive in our opinion primarily from the verbs, and not from the comparatives.
*A construction peculiar to Swedish, as in *Det var inte tyngre än att han kunde lyfta det.* (Lit: "It was not heavier than that he could carry it").

Portugalien), and manually extracted all sentences containing comparatives. The results are summarized in Table 7.

One striking fact is immediately apparent; the second commonest type of construction (No. 1, "discourse comparative") is actually one that we fail to cover! In our defense, however, we can at any rate claim that none of the other authors we have quoted appear to do so either. By *discourse comparatives,* we mean here comparatives where the associated comparative complement (in English normally introduced by "than" or "as") is completely absent, and must be inferred from the context. Typical examples (taken from the Agatha Christie novel) follow below in sentences 43)–45).

43) The abandoned creature . . . committed a still graver sin.
44) At eight o'clock the wind was blowing more strongly.
45) I should be colder if I were dead.

If the discourse comparatives are excluded, however, we are clearly capable of dealing with the vast majority of those left. Thus we feel fairly well justified in claiming that our treatment covers most of the common cases of comparison that occur in practice.

7 Summary and Further Directions

We have presented a treatment of comparative constructions that can claim to cover most of the commonly occurring cases, and is also capable of being implemented in a reasonably standard framework. Several of the key ideas are adapted from the theoretical work of Pinkham. In particular, we divide comparatives into two separate classes, clausals and phrasals. Clausals are interpreted by a version of Pinkham's "quantifier binding," which treats the missing material in the comparative clause as null proforms bound by the comparative head (Section 4); phrasals are interpreted directly, using an adaption of Pinkham's "distributive copying." The semantics of phrasals are summarized in the key formulas (*), (**), and (***) from Section 3.2.

At the beginning of Section 1, we presented a selection of comparative constructions, classified according to a more or less traditional approach. Table 8 shows the same examples, this time with our classification.

Particularly in our analysis of contrastive phrasals, we have gone to some lengths to describe how the ideas can be implemented in a reasonably efficient way. Here, the key idea has been to use a level of representation intermediate between syntax and logical form, which we call quasi-logical form, or QLF. At QLF level, phrasal comparatives are linked to their heads, but not to the constituent they are contrasted against. Rewriting rules then reshape the QLF until it is in a form where the comparative complements are at the same level as their "correlates." We also make use of features for defining long-range dependencies. These occur in several varieties: the bound null proforms in clausal comparatives, the missing sentential and VP complements in S- and VP-operators, and finally the comparative complements, which in general are separated from their heads in surface syntax.

Weighing up the strengths and weaknesses of our approach, it is most natural to compare it with Friedmann's. Since Friedmann gives the comparative degree wide scope by default, her system needs extra rewriting rules for dealing with quantified NPs, both in the main clause and the comparative complement; we have argued that a "narrow scope" default is in general preferable. We have also claimed that it is difficult in Friedmann's syntactically oriented approach to give a formulation of the comparative deletion rule that will extend to the harder cases.

However, we have also seen, in Section 3.4.6, that our semantics runs into problems when attempting to derive the "wide-scope" reading for sentences like 27), where the comparative head occurs within the scope of a modal operator. Rather than postulate more *ad hoc* rewriting rules, we tentatively suggested that there may be alternate wide-scope versions of the key formulas from Section 3.2; this fits in well with Pinkham's remarks on wide- and narrow-scope clausals, which we refer to at the end of Section 4. Further investigation of these issues seems like one of the most important items on our agenda for future research.

Another topic that urgently needs study is the interaction between comparatives and coordination; in certain cases, illustrated in sentences 46)–49), our first impression is that problems will occur. Since these obviously depend on the

Table 8. Typical Examples of Comparatives: Our Classification

Clausal comparatives	
1) Mary was happier in New York than John was in London.	
2) John has more books than Mary has newspapers.	
3) The table is longer than it is wide.	
Contrastive phrasal comparatives	
4) John is taller than Mary.	*NP contrasting, adjective*
5) Few people run as fast as John.	*NP contrasting, adverbial*
6) John bought more books than Mary.	*NP contrasting, determiner*
7) John was happier in New York than in London.	*PP contrasting*
8) Mary had more friends than John thought.	*S-operator contrasting*
9) John hit Mary harder than he meant to.	*VP-operator contrasting*
10) More men than women bought the book.	*CN contrasting*
11) More people voted for the proposal than against.	*Preposition contrasting*
12) Mary needs a larger car than this Fiat.	*Internal contrasting in AP*
13) John ran faster than the world record.	*Internal contrasting in adverbial*
Determiner/phrasal	
14) More than 50 people signed the petition.	
Unclear cases: possibly contrastive phrasals	
15) John was born in the same city as Mary.	
16) John arrived before Mary.	
17) John likes Mary's house better than Mary John's.	

way in which we intend to handle coordination, we postpone discussion to a later paper.

46) No one could have a safer and more promising future than John.
47) Mary is more attractive than Jane and Sarah.
48) John visits the U.S. more often than Mary, but less frequently than Carol.
49) The results were better than John claimed, but worse than we had hoped.

Several other open questions also deserve mention. We have not been very specific in giving rules about where comparative complements may occur, and our treatment of phrasal contrastives also appear to allow more readings than really exist. It would certainly be desirable to find rules to eliminate these, or at least heuristics to say which readings can be regarded as unlikely. Another interesting question is whether it is possible to make the theory more compact, by collapsing the three formulas (*), (**), and (***) into one; they are so similar that this seems intuitively quite feasible. We speculate that one may be able to do this in a framework like that described in Pereira and Pollack (1988), which allows conditional interpretation.

Finally, we say a few more words about the "discourse comparatives" mentioned in Section 6. A cursory examination of the example sentences would suggest that most of the missing comparative complements are of one of the following: "than previously," "than the one just mentioned," or "than is the case." For example, taking another look at sentences 43)–45), we can postulate that the complement is as given below in italics.

43) The abandoned creature ... committed a still graver sin (*than the one just mentioned*).
44) At eight o'clock the wind was blowing more strongly (*than previously*).
45) I should be colder (*than is the case*) if I were dead.

To fill in these missing complements, it as usual seems clear that pragmatic information is needed; researchers working in discourse theory may find these problems worth investigating.

ACKNOWLEDGMENTS

We would like to thank the following people: several anonymous referees, for insightful criticism of earlier versions of this paper; James Allen, Jerry Hobbs, Fernando Pereira and Remko Scha for useful comments and general encouragement; and, last but not least, Rune Gustavsson and Siwert Sundström for helpfulness and flexibility with regard to financing our research.

REFERENCES

Alshawi, H.; Carter, D. M.; van Eijck, J.; Moore, R. C.; Moran, D. B.; Pereira, F. N. C.; Pulman, S. G.; and Smith, A. G. 1989. *Research Program in Natural Language Processing: Final Report,* SRI Technical Report.

Ballard, B. 1989. A General Computational Treatment of Comparatives for Natural Language Question Answering. *Proc. 26th ACL,* 41–48.

Banks, A. 1986. Modifiers in Natural Language. B.A. Thesis, Uppsala University.

Banks, A. and Rayner, M. 1987. Comparatives in Logic Grammars—Two Viewpoints. *Proceedings of the 2nd International Workshop on Natural Language Understanding and Logic Programming.*

Barwise, J. and Cooper, R. 1981. Generalized Quantifiers and Natural Language. *Linguistics and Philosophy*. 4:159–219.

Bowen, J. and Carvalho, A. 1988. A Logic Grammar Formalism for Pronoun Endophora. Unpublished manuscript.

Bresnan, J. 1973. Syntax of the Comparative Clause Construction in English. *Linguistic Inquiry* 4:275–343.

Cooper, R. 1983. *Quantification and Syntactic Theory*. D. Reidel, Dordrecht, Holland.

Dahl, D.; Dowding, J.; Hirschman, L.; Lang, F.; Linebarger, M.; Palmer, M.; Passonneau, R.; and Riley, L. 1989. *Integrating Syntax, Semantics, and Discourse: DARPA Natural Language Understanding Program*. R&D Status Report, Paoli Research Center, Unisys Defense Systems.

Dowty, D.; Wall, R. E.; and Peters, S. 1982. *Introduction to Montague Semantics*. D. Reidel, Dordrecht, Holland.

Engdahl, E. 1985. *Constituent Questions*. D. Reidel.

Fenstad, J. E.; Halvorsen, P. K.; Langholm, T.; and van Bentham, J. 1987. *Situations, Language and Logic*. D. Reidel, Dordrecht, Holland.

Friedmann, C. 1989. A General Computational Treatment of the Comparative. In *Proc. 27th ACL*. 161–168.

Grosz, B.; Appelt, D. E.; Martin, P. A.; and Pereira, F. C N. 1987. TEAM: An Experiment in the Design of Transportable Natural-Language Inferfaces. *Artificial Intelligence* 32(2):173–244.

Hankamer, J. 1971. Why There Are Two "than" 's in English. *CLS* 9:179–192.

Heim, I. 1985. *Notes on Comparatives and Related Matters*. Unpublished manuscript.

Hobbs, J.; Stickel, M.; Martin, P.; and Edwards, D. 1988. Interpretation as Abduction. *Proc. 26th ACL*, 95–103.

Kartunnen, L. 1986. D-PATR: A Development Environment for Unification-Based Grammars. *Proc. 11th COLING*, 74-80.

Keenan, E. L. and Stavi J. 1986. Natural Language Determiners. *Linguistics and Philosophy* 9:253–325.

Klein, E. 1980. A Semantics for Positive and Comparative Adjectives. *Linguistics and Philosophy* 4:1–45.

Klein, E. 1981. The Interpretation of Adjectival, Nominal, and Adverbial Comparatives. In: Groenendijk, J.; Janssen, T.; and Stokhof, M. (eds.) *Formal Methods in the Study of Language*. Mathematical Center Tracts, Amsterdam.

Klein, E. 1982. The Interpretation of Adjective Comparatives. *Journal of Linguistics* 18:113–136.

Krifka, M. 1987. *Bemerkung zu Vergleichskonstruktioner*. Unpublished manuscript. University of Tübingen.

Larson, R. 1988. Scope and Comparatives. *Linguistics and Philosophy* 11:1–27.

Lesmo, L. and Torasso, P. 1985. Analysis of Conjunction in a Rule-Based Parser. *Proc. 23rd ACL*, 180–187.

McCord, M. C. 1986. Focalizers, the Scoping Problem, and Semantic Interpretation Rules in Logic Grammars. In: van Carneghem, M. and Warren, D. H. D. (eds.) *Logic Programming and its Applications*. Ablex, Norwood, N.J.

McCord, M. C. 1987. Natural Language Processing in Prolog. In: Walker, A. (ed.) *Knowledge Systems and Prolog*. Addison-Wesley, Reading, MA.

Montague, R. 1974. The Proper Treatment of Quantification in English. In: Thomason, R. (ed.) *Formal Philosophy: Selected Papers of Richard Montague*. Yale University Press, New Haven, CT.

Moran D. B. 1988. Quantifier Scoping in the SRI Core Language Engine. *Proc. 26th ACI*, 33–40.

Pereira, F. N. C. 1983. *Logic for Natural Language Analysis*. SRI Technical Note No 275.

Pereira, F. N. C. and Pollack, M. E. 1988. An Integrated Framework for Semantic and Pragmatic Interpretation. *Proc. 26th ACL*, 75–86.

Pereira, F. N. C. and Shieber, S. 1985. *Prolog and Natural Language Understanding*. CSLI lecture notes.

Pinkham, J. 1985. *The Formation of Comparative Clauses in French and English*. Garland Publishing Inc., New York, NY.

C. Pollard and I. Sag. 1988. *Information-Based Syntax and Semantics CSLI*, University of Chicago Press, Chicago, IL.

Rayner, M. and Banks, A. 1986. Temporal Relations and Logic Grammars. *Proc. ECAI-86*, 2:9–14.

Rayner, M. and Banks, A. 1988. Parsing and Interpreting Comparatives. *Proc. 26th ACL*, 49–60.

Rayner, M. and Banks, A. 1989. An Overview of the SNACK-85 Natural Language System. In Plander, V. (ed.) *Proc. 5th Intl. Conference on AI and Information-Control Systems of Robots*, North-Holland.

Woods, W. 1978. Semantics and Quantification in Natural-language Question-Answering. In: Yovits, M. (ed.) *Advances in Computers*, 17:2–64. Academic Press. New York, NY.

Notes

1. Earlier versions of several of the results herein have already been published in Banks (1986), Rayner and Banks (1986), Banks and Rayner (1987), and Rayner and Banks (1988a; 1988b). Since a number of important changes have been made, this report is to be taken as superseding all five.
2. Current address: ZYX Sweden AB, Styrmansg. 6, 114 54 Stockholm.
3. It appears, for example, that Japanese allows essentially the same range of comparative complements, and assigns a parallel semantics to them.
4. We borrow this useful term from the Core Language Engine project (Alshawi *et al.* 1988; 1989).
5. Those familiar with Montague semantics will realize that the version we are using here is in several respects fairly different from that in PTQ. Most important, we more or less completely dispense with the mechanisms for dealing with intentionality; since this isn't our concern here, we feel that it would only confuse the issue. Readers who disapprove of these ideas should have no great difficulty in converting out derivations into more conventional ones.
6. There is some disagreement between applied and theoretical linguists concerning the correct division of credit here. Although Woods and his colleagues were the first people to realize the idea in practice, they made no attempt to justify their work in formal terms; this was later achieved independently by Cooper (1983), working within the Montague grammar school. Our work has to some extent been influenced by both of these sources.
7. Mean bastard.
8. Excluding "discourse comparatives": see Sections 6 and 7.
9. We would like to thank Carol Friedmann for bringing this example to our attention.

Appendix 1 Terminology

Comparative Complement: the constituent introduced by the comparison marker

Comparative Operator: expressions like *more, less, twice as much.*

Comparison Marker: words like *than, as.*

Contrasted Material: constituent in the main clause against which a phrasal comparative complement is compared

Head of the Comparison: refers to the comparative degree or determiner in the main clause

Phrasal Comparative: a comparative complement which is not a comparative clause.

Q: a null quantifier which is extraposed in the interpretation of clausal comparatives.

Appendix 2 Program Code

```
% The syntactic part of the system, written as an XG-grammar [Pereira 83]

% Internal XG stuff, used for extraposing constituents.

open_x_bracket ... close_x_bracket --> [].

extrapose(s_comp(S)) ... s_comp(S) --> [].
extrapose(vp_comp(Vp)) ... vp_comp(Vp) --> [].
extrapose(np(A,B)) ... np(np(A,B)) --> [].
extrapose(det(A)) ... det(det(A)) --> [].
extrapose(adv(A,B)) ... adv(adv(A,B)) --> [].

% Clauses

question(y_n_q(S)) -->
  extraposed_verb,
  s(S).

question(wh_q(s(subject(Subject),Vp))) -->
  question_np(Subject),
  vp(Vp).

% Swedish is a verb-second language, so a fronted WHQ forces the
% next thing to be a verb.

question(wh_q(S)) -->
  extraposed_question_element,
  extraposed_verb,
  s(S).

main_s(S) --> s(S).

main_s(S) -->
  extraposed_adv,
  extraposed_verb,
  s(S).

s(s(subject(Subject),Vp)) -->
  np(Subject),
  vp(Vp).

extraposed_verb ... verb(Verb,Mood,Voice,Frame_pattern) -->
  verb(Verb,Mood,Voice,Frame_pattern).

extraposed_question_element ... np(Question_element) -->
  question_np(Question_element).

extraposed_question_element ... preposition(Prep),np(Question_element) -->
  preposition(Prep),
  question_np(Question_element).

extraposed_adv ... adv(Adv) -->
  adv(Adv).

% The XG-pre-processor has been modified slightly so as to give a simple
% transparent treatment of bound anaphora. (This idea was borrowed from
% [Bowen & Carvalho 88]). The form with_extra_anaphor(Anaphor,Constituents)
% means that Constituents are to be parsed with Anaphor available as
% an anaphoric referent. This is used to treat verbs in comparative
% complements like "do" and "is", which (we claim, at any rate), refer back
% to some verbal antecedent, e.g.
% "Which kings reigned longer than Charles I did?"

vp(vp(Verb,Mood,trace_np(Controller),Args)) -->
  verb(Verb,Mood,Voice,Subcat_pattern),
  with_extra_anaphor(verb(Verb,Voice),

 (verb_args(Subcat_pattern,Args,Mods,trace_np(Controller),Voice),
                   verb_mods(Mods,[]))).

% NPs. As explained in the paper, NP's consist of a determiner + a CN.

question_article(Article) :-
  member(Article,[which,which_pl,what,who,how_many(_)]).

question_np(Np) -->
  np(Np),
  {Np = np(det([Article,_,_]),cn(_,Mods)),
  (question_article(Article);
   member(role(_,np(det([Article_1,_,_]),Cn)),Mods),
   question_article(Article_1))}.

np(np(det([Art,Det,Num]),
      cn(Noun,Mods))) -->
  det(det([Art,Det,Num])),
  cn(cn(Noun,Mods),Det,Num),
  {np_constraints(Art,Det,Num)},
  optionally_signal_comparative(np(det([Art,Det,Num]),
                                   cn(Noun,Mods))),
  optional_comparand.

np(name(Name)) --> name(Name).

det(det([Art,Det,Num])) -->
  article(Art,Det,Num).

det(det([[],_,_])) --> [].

cn(cn(Noun,Mods),Det,Num) -->
  optional_adj(Mods),
  optional_noun(Mods,Noun,Det,Num).

% No undetermined NP's without articles

np_constraints(Art,Det,Num) :-
  dif(Art,[]).

np_constraints([],det,_).

% If there's an adjective, then we don't necessarily need a noun.

optional_noun(Mods,Noun,Det,Num) -->
  noun(Noun,Det,Num).

optional_noun([Adj],omitted_noun,_,_) --> [].

% optionally_signal_comparative propagates rightward a request for a
comparative complement.

optionally_signal_comparative(X) ...
      comparison_signalled(Comp_operator,X,Complement) -->
  {contains_degree(X,deg(comp(Comp_operator,Complement)))}.

optionally_signal_comparative(X) -->
  {\+contains_degree(X,deg(comp(_,_)))}.

contains_degree(det([art(_,Degree),_,_]),Degree).
contains_degree(adj(_,deg(Degree)),deg(Degree)).
contains_degree(adv(_,deg(Degree)),deg(Degree)).
contains_degree(ap_comp(Ap),Degree) :-
  contains_degree(Ap,Degree).
contains_degree(cn(_,Mods),Degree) :-
  member(Mod,Mods),
  contains_degree(Mod,Degree).
contains_degree(np(Det,Cn),Degree) :-
  contains_degree(Det,Degree);
  contains_degree(Cn,Degree).

% This is discussed at the end of section 4.
% If we have article comparison ("more", "fewer"), then we can either
% extrapose an article ("I have more cars than he has motorcycles.")
% or a whole NP ("I have more friends in London than he has in New York.")

bound_null_proforms(np(det([art(Art,deg(comp(_,_))),
                          Det,Num]),_),
                    det([q(Q),Det,Num]),
                    q(Q)).

bound_null_proforms(np(det([art(Art,deg(comp(_,_))),
                          Det,Num]),
                      cn(Noun,Mods)),
                    np(det([q(Q),Det,Num]),
                      cn(Noun,Mods)),
                    q(Q)).

% If we have prenominal adjective comparison, then we have to extrapose
% a whole NP. We form it by replacing the comparative adjective by a positive
one.
```

```
bound_null_proforms(np(Det,
                       cn(Noun,[adj(Adj,deg(comp(_,_)))])),
                    np(Det,
                       cn(Noun,[adj(Adj,deg(q(Q)))])),
                    q(Q)).

% If we have predicate adjective comparison, we extrapose a similar adjective.

bound_null_proforms(ap_comp(adj(Adj,deg(comp(_,_)))),
                    ap_comp(adj(Adj,deg(q(Q)))),
                    q(Q)).

% With an adverb, a similar adverb is extraposed

bound_null_proforms(adv(Adv,deg(comp(_,_))),
                    adv(Adv,deg(q(Q))),
                    q(Q)).

optional_adj([Adj]) --> adj(Adj).
optional_adj([]) --> [].

adj(adj(Adj,[])) -->
  adjective(Adj).
adj(adj(Adj,deg(comp(Comp_operator,Complement)))) -->
  comparative_adjective(Adj,Comp_operator).

%Verb args

% Each member in the list of arguments to the verb is a term of the form
% role(Arg_type, Arg_value). See section1.2.2

%Base case. If we've parsed all the frame, we're finished

verb_args([],Mods,Mods,_,_) --> [].

verb_args([Subcat|Rest],[Arg|Next_args],Out_args,Controller,Voice) -->
  verb_arg(Subcat,Arg,Controller,Voice),
  verb_args(Rest,Next_args,Out_args,Controller,Voice).

verb_args([agent|Rest_subcat],Next_args,Out_args,Controller,passive) -->
  verb_args(Rest_subcat,Next_args,Out_args,Controller,Voice).

verb_arg(agent,role(agent,Controller),Controller,active) --> [].

%With an active verb, we parse an object by parsing an NP

verb_arg(object,role(object,Np),_,active) -->
  np(Np).

%But with a passive verb, the surface subject is the object

verb_arg(object,role(object,Controller),Controller,passive) --> [].

verb_arg(prep_object(Prep),role(prep_object,Np),_,_) -->
  preposition(Prep),
  np(Np).

verb_arg(relational_object,role(Head,np(Det,cn(person,Mods))),_,active) -->
  np(np(Det,cn(Head,Mods))).

verb_arg(s_comp,role(s_comp,S),_,active) -->
  s_comp(S).

verb_arg(ap_comp,role(ap_comp,Ap),_,active) -->
  ap_comp(Ap).

verb_arg(inf_comp,role(inf_comp,Vp),_,_) -->
  vp_comp(Vp).

s_comp(S) -->
  [att],s(S).

vp_comp(Vp) -->
  vp(Vp).

ap_comp(ap_comp(Adj)) -->
  adj(Adj),
  optionally_signal_comparative(ap_comp(Adj)),
  optional_comparand.

% A complement to a comparative. This can only be parsed if we have
% earlier detected a comparative marker, which has made its presence
% known by putting a comparative_signalled element on the extraposition
% list.

optional_comparand -->
  comparison_signalled(Comp_operator,Associated_constituent,Complement),
  {complementizer_for_comp_operator(Comp_operator,Complementizer)},
  complementizer(Complementizer),
  comparand_complement(Complementizer,Associated_constituent,Complement).

% First case: an NP or PP comparand, the simplest (and commonest)
% kind of contrastive comparative: Section 3.4.1. We may have an
% "anaphoric verb" with an NP, which is going to be available
% from the "verb anaphor stack" (see comments above).

comparand_complement(Complementizer,_,subject(Np)) -->
  optional_vad(Complementizer),
  np(Np),
  anaphoric_verb.

comparand_complement(_,_,Np) -->
  np(Np).

comparand_complement(_,_,Role) -->
  pp(Role).

% Second case: CN-comparison (section 3.4.2)

comparand_complement(_,_,Cn) -->
    cn(Cn,Det,Num).

% Third case: an "S-operator" or "VP-operator" comparand, (section 3.4.3)

comparand_complement(Complementizer,_,S_operator) -->
  optional_vad(Complementizer),
  s_operator(S_operator).

comparand_complement(Complementizer,_,Vp_operator) -->
  optional_vad(Complementizer),
  vp_operator(Vp_operator).

s_operator(s_operator(trace_s(Inner_s),S)) -->
  open_x_bracket,
  extrapose(s_comp(trace_s(Inner_s))),
  s(S),
  close_x_bracket.

vp_operator(vp_operator(trace_vp(Inner_vp),S)) -->
  open_x_bracket,
  extrapose(vp_comp(trace_vp(Inner_vp))),
  s(S),
  close_x_bracket.

% Fourth case: contrasted prepositions (section 3.4.5)

comparand_complement(_,_,role_marker(Prep)) -->
  preposition(Prep).

% Fifth case: clausal comparison (section 4)

comparand_complement(Complementizer,Associated_constituent,Clausal) -->
  optional_vad(Complementizer),
  clausal_comparative(Clausal,Associated_constituent).

clausal_comparative(clausal(Q,S),Associated_constituent) -->
  {bound_null_proforms(Associated_constituent,Null_proforms,Q)},
  open_x_bracket,
  extrapose(Null_proforms),
  s(S),
  close_x_bracket.

% Of course, the optional comparand will usually not be present!
% However, if a comparison has been signalled we have to pick it up somewhere,
% otherwise it will still be hanging around on the extraposition list
% when we reach the end of the sentence.

optional_comparand --> [].

% In Swedish, some comparative complements, may be preceded by an optional
% "vad" ("what").

optional_vad(than) --> [vad].
optional_vad(than) --> [].
optional_vad(Other) -->
  {dif(Other,than)}.

% Getting an "anaphoric verb" - see comments above.

anaphoric_verb -->
  verb(Verb,_,_,_),
  anaphor_stack(Potential_referents),
  {verb_anaphora_antecedent(Verb,Potential_referents)}.

verb_anaphora_antecedent(Verb,Potential_referents) :-
  member(Ref,Potential_referents),
  verb_anaphora_match(Verb,Ref).

verb_anaphora_match(Verb,verb(Ref_verb,active)) :-
  verb_anaphora_match_1(Verb,Ref_verb,active).

verb_anaphora_match(Verb,verb(Ref_verb,passive)) :-
  verb_anaphora_match_1(Verb,Ref_verb,passive).

% Rules for verb anaphora (grossly oversimplified - this is not
% the main subject of the paper!)

verb_anaphora_match_1('do',_,active).
verb_anaphora_match_1(have,have,active).
verb_anaphora_match_1('be_passive',_,passive).
verb_anaphora_match_1('be','be',active).
```

```
% Verb mods

verb_mods(Mods,Mods) --> [].
verb_mods([Mod|Mods],Mods) -->
    verb_mod(Mod),
    optionally_signal_comparative(Mod),
    optional_comparand.

verb_mod(Pp) --> pp(Pp).
verb_mod(Adv) --> adv(Adv).

pp(role(Prep,Np)) -->
  preposition(Prep),
  np(Np).

adv(adv(Adv,[])) -->
  adverb(Adv).
adv(adv(Adv,deg(comp(Comp_operator,Complement)))) -->
  comparative_adverb(Adv,Comp_operator).

adjective(Adj) -->
  [Word],{word(Word,Adj,adjective)}.

adjective_modifier(Mod) -->
  [Word],{word(Word,Mod,adjective_modifier)}.

adverb(Adverb) -->
  [Word],{word(Word,Adverb,adverb)}.

article(Art,Det,Num) -->
  [Word],{word(Word,Art,article(Det,Num))}.

article(art(Art,deg(comp(Comp_operator,Complement))),
        Det,Num) -->
  comparative_article(Art,Comp_operator,Det,Num).

comparative_adverb(Adverb,Comp_operator) -->
  [Word],{word(Word,Adverb,comparative_adverb(Comp_operator))}.

comparative_adverb(Adv,Mod) -->
  adjective_modifier(Mod),
  [Word],{word(Word,Adv,adverb)}.

comparative_adjective(Adj,more) -->
  [Word],{word(Word,Adj,comparative_adjective)}.

comparative_adjective(Adj,Mod) -->
  adjective_modifier(Mod),
  [Word],{word(Word,Adj,adjective)}.

comparative_article(Art,Comp_operator,Det,Num) -->
  [Word],{word(Word,Art,article(comparative,Comp_operator,Det,Num))}.

complementizer(null) --> [].

complementizer(Comp) -->
  [Word],{word(Word,Comp,complementizer)}.

name(Name) -->
  [Word],{word(Word,Name,name)}.

noun(Noun,Det,Num) -->
  [Word],{word(Word,Noun,noun(Det,Num))}.

preposition(Prep) -->
  [Word],{word(Word,Prep,preposition)}.

pronoun(Pronoun) -->
  [Word],{word(Word,Pronoun,pronoun)}.

verb(Verb,Mood,Voice,Frame_pattern) -->
  [Word],{word(Word,Verb,verb(Voice,Mood,Frame_pattern))}.


% A note on representation:

% evnt(lambda(X,P(X)) is to be read as: "An event E occurred, such that P(E)
% held.

% quant(Q,lambda(X,R(X)),lambda(Y,S(Y))), where Q is a generalized quantifier,
% is to be taken as equivalent with
% Q(lambda(X,R(X)),lambda(Y,S(Y))).
% When reading formulas, it will often be helpful to think of X and Y as in
% some sense being "the same variable"

% q(N,P) is to be read as denoting the generalized quantifier {N|P}.
% q(N,[]) represents the quantifier {N}, i.e. "exactly N"


top_sem(y_n_q(S),y_n_q(Sem)) :- !,
  s_form(S,Sem).

top_sem(wh_q(S),wh_q(Sem)) :- !,
  s_form(S,Sem).

top_sem(S,Sem) :-
  s_form(S,Sem).

s_form(S,Form) :-
  syn_to_qlf(S,Qlf),
  reduce(Qlf,Form).

% syn_to_qlf converts the syntactic form to the quasi-logical form

syn_to_qlf([],[]).

syn_to_qlf([First|Rest],[First_qlf|Rest_qlf]) :-
  syn_to_qlf(First,First_qlf),
  syn_to_qlf(Rest,Rest_qlf).

syn_to_qlf(s(Subject,Vp),Qlf) :-
  syn_to_qlf(Subject,Subj_qlf),
  syn_to_qlf(Vp,Vp_qlf),
  rewrite_qlf0(apply(Subj_qlf,Vp_qlf),Qlf).

syn_to_qlf(vp(Head,_,trace_np(X),Frame),vp(Head,X,Qlf)) :-
  syn_to_qlf(Frame,Qlf).

syn_to_qlf(role(Role,Arg),role(role_marker(Role),Arg_qlf)) :-
  syn_to_qlf(Arg,Arg_qlf).

syn_to_qlf(subject(Np),subject(Np_qlf)) :-
  syn_to_qlf(Np,Np_qlf).

syn_to_qlf(np(Det,Cn),quant(Det_qlf,Cn_qlf)) :-
  syn_to_qlf(Det,Det_qlf),
  syn_to_qlf(Cn,Cn_qlf).

syn_to_qlf(ap_comp(Adj),Qlf) :-
  syn_to_qlf(Adj,Qlf).

syn_to_qlf(trace_np(Var),trace_np(Var)).

syn_to_qlf(det([art(_,deg(Degree)),_,_]),Degree_qlf) :-
  syn_to_qlf(Degree,Degree_qlf).

syn_to_qlf(det([q(Q),_,_]),q(Q,[])).

syn_to_qlf(det([Det,D,N]),[Det,D,N]) :-
  \+functor(Det,art,_),
  \+functor(Det,q,_).

syn_to_qlf(cn(Head,Mods),cn(Head,Mods_qlf)) :-
  syn_to_qlf(Mods,Mods_qlf).

syn_to_qlf(name(Name),name(Name)).

syn_to_qlf(adj(Adj,deg(Degree)),
          abstraction(X,relation([Adj,X,Degree_qlf]))) :-
  syn_to_qlf(Degree,Degree_qlf).

syn_to_qlf(adv(Adv,deg(Degree)),
          abstraction(X,relation([Adv,X,Degree_qlf]))) :-
  syn_to_qlf(Degree,Degree_qlf).

syn_to_qlf(comp(Comp_operator,clausal(q(Q1),S)),
          q(Q,conj(relation([Comp_operator,Q,Q1]),
                   S_qlf))) :-
  syn_to_qlf(S,S_qlf).

syn_to_qlf(comp(Comp_operator,Phrasal),
          c_degree(Comp_operator,Phrasal_qlf)) :-
  \+functor(Phrasal,clausal,_),
  syn_to_qlf(Phrasal,Phrasal_qlf).

syn_to_qlf(q(Q),q(Q,[])).

% This is for "S_operators"

syn_to_qlf(trace_s(X),trace_s(X)).

% And this is for VP-operators.

syn_to_qlf(trace_vp(X),trace_vp(X)).

syn_to_qlf(role_marker(Prep),role_marker(Prep)).

syn_to_qlf(s_operator(trace_s(X),S),s_operator(X,S_operator_qlf)) :-
  syn_to_qlf(S,S_operator_qlf).

syn_to_qlf(vp_operator(trace_vp(X),S),vp_operator(X,S_qlf)) :-
  syn_to_qlf(S,S_qlf).

rewrite_qlf0(Qlf0,Qlf) :-
  contains_functor(Qlf0,c_degree/2),!,
  rewrite_qlf(Qlf0,Qlf).

rewrite_qlf0(Qlf,Qlf).

% comparison nodes can be inserted in the following places:
%
% 1. Over a "role" node in a vp
% 2. Over a "vp" node
```

```
% 3. Over an "abstraction" node (really, this should be unified with 2.)
% 4. Over an "apply" node
%
% rewrite_qlf recursively goes down the qlf: when it finds a place where
% a comparison could potentially occur, it checks to see if it can make one.
% That is, it
%
% a. tries to extract a c_complement and store it.
% b. tries to find a suitable contrast.
% c. constructs the comparison node and inserts it.

rewrite_qlf(Qlf0,Qlf) :-
  rewrite_qlf_on_subtrees(Qlf0,Qlf).

rewrite_qlf(Qlf0,Qlf) :-
  comparison_insertion_site(Qlf0),
  raise_comparative(Qlf0,Qlf1,Store),
  contrast_and_insert_comparison(Qlf1,Store,Qlf).

comparison_insertion_site([role(_,_)|_]).
comparison_insertion_site(vp(_,_,_)).
comparison_insertion_site(abstraction(_,_)).
comparison_insertion_site(apply(_,_)).

contrast_and_insert_comparison(Qlf0,Store,Qlf) :-
  Store = c_stored_item(C_complement,_,Q),
  find_contrast0(C_complement,Qlf0,Qlf1,Contrast,C,Q),
  insert_normal_comparison(Qlf1,Store,Contrast,C,Qlf).

contrast_and_insert_comparison(Qlf1,Store,Qlf) :-
  insert_s_operator_comparison(Qlf1,Store,Qlf).

contrast_and_insert_comparison(Qlf1,Store,Qlf) :-
  insert_abstracted_variable_comparison(Qlf1,Store,Qlf).

insert_normal_comparison(
    Qlf,
    c_stored_item(C_complement,C_operator,Q),
    Contrast,
    C,
    comparison(Contrast,
               C_complement,
               C_operator,
               C,
               Q,
               Qlf)).

% This is the first special case from section 3.3. The qlf
% under the comparison must be an abstraction, and the C_complement
% must be an NP denotation.

insert_abstracted_variable_comparison(
    Qlf,
    c_stored_item(C_complement,C_operator,Q),
    abstraction(X1,
                comparison(trace_np(X1),
                           C_complement,
                           C_operator,
                           C,
                           Q,
                           apply(C,Qlf)))) :-
  np_denotation(C_complement),
  Qlf = abstraction(_,_).

% This is the second special case from section 3.3. The C_complement
% must be an s_operator, and the qlf under the comparison something
% that an s_operator can be applied to - an S or VP denotation.
% The second clause is to take care of a subtle problem that arises
% if the s_operator's argument is a VP-denotation: see the remarks
% before Table 4 in section 3.4.3.

insert_s_operator_comparison(
    Qlf,
    c_stored_item(C_complement,C_operator,Q),
    comparison(s_operator(S,S),
               C_complement,
               C_operator,
               C,
               Q,
               apply(C,Qlf))) :-
  C_complement = s_operator(_,_),
  Qlf = apply(_,_).

insert_s_operator_comparison(
    Qlf,
    c_stored_item(C_complement,C_operator,Q),
    abstraction(
       X,
       comparison(s_operator(S,S),
                  C_complement,
                  C_operator,
                  C,
                  Q,
                  apply(C,(apply(trace_np(X),
                                 Qlf)))))) :-
  C_complement = s_operator(_,_),
  Qlf = vp(_,_,_).

% Raise_comparative lifts a c_degree out of a constituent, leaving a
% Q behind.

raise_comparative(X,_,_) :- var(X),!,fail.

raise_comparative(c_degree(C_operator,C_complement),
                  Q,
                  c_stored_item(C_complement,C_operator,Q)).

raise_comparative(relation(List0),
                  relation(List1),
                  Store) :-
  raise_comparative(List0,List1,Store).

raise_comparative([F0|R],[F1|R],Store) :-
  raise_comparative(F0,F1,Store).

raise_comparative([F|R0],[F|R1],Store) :-
  raise_comparative(R0,R1,Store).

raise_comparative(apply(Qlf0,X),
                  apply(Qlf1,X),
                  Store) :-
  raise_comparative(Qlf0,Qlf1,Store).

raise_comparative(apply(X,Qlf0),
                  apply(X,Qlf1),
                  Store) :-
  raise_comparative(Qlf0,Qlf1,Store).

raise_comparative(abstraction(X,Qlf0),
                  abstraction(X,Qlf1),
                  Store) :-
  raise_comparative(Qlf0,Qlf1,Store).

raise_comparative(quant(Det0,Cn_item),
                  quant(Det1,Cn_item),
                  Store) :-
  raise_comparative(Det0,Det1,Store).

raise_comparative(quant(Det,cn(Head,[Mod0])),
                  quant(Det,cn(Head,[Mod1])),
                  Store) :-
  raise_comparative(Mod0,Mod1,Store).

raise_comparative(role(Role,Quant0),
                  role(Role,Quant1),
                  Store) :-
  raise_comparative(Quant0,Quant1,Store).

raise_comparative(subject(Quant0),
                  subject(Quant1),
                  Store) :-
  raise_comparative(Quant0,Quant1,Store).

raise_comparative(vp(Head,X,Qlf0),
                  vp(Head,X,Qlf1),
                  Store) :-
  raise_comparative(Qlf0,Qlf1,Store).

rewrite_qlf_on_subtrees(apply(Np,Vp0),apply(Np,Vp)) :-
  rewrite_qlf(Vp0,Vp).
rewrite_qlf_on_subtrees(apply(Np0,Vp),apply(Np,Vp)) :-
  rewrite_qlf(Np0,Np).

rewrite_qlf_on_subtrees(vp(Head,X,Qlf0),vp(Head,X,Qlf)) :-
  rewrite_qlf(Qlf0,Qlf).

rewrite_qlf_on_subtrees([F0|R],[F|R]) :-
  rewrite_qlf(F0,F).
rewrite_qlf_on_subtrees([F|R0],[F|R]) :-
  rewrite_qlf(R0,R).

rewrite_qlf_on_subtrees(role(Role,Quant0),role(Role,Quant)) :-
  rewrite_qlf(Quant0,Quant).

rewrite_qlf_on_subtrees(subject(Quant0),subject(Quant)) :-
  rewrite_qlf(Quant0,Quant).

rewrite_qlf_on_subtrees(quant(Det,Cn0),quant(Det,Cn)) :-
  rewrite_qlf(Cn0,Cn).

rewrite_qlf_on_subtrees(cn(Head,Mods0),cn(Head,Mods)) :-
  rewrite_qlf(Mods0,Mods).

% find_contrast checks a C_complement against something that might
% be a contrastable item (or contain one). The last three arguments are: what
% the thing is replaced by (either the C, or something containing the C),
% the extracted contrasted item, and a pointer to the C. Note that the
% contrast cannot contain the Q - hence the following clause.
```

```
find_contrast0(C_complement,Qlf0,Qlf,Contrast,C,Q) :-
  find_contrast(C_complement,Qlf0,Qlf,Contrast,C),
  \+contains_id(Contrast,Q).

% There are a large number of cases. It would clearly be possible to collapse
% at least some of these.

% The first five are essentially similar:


% 1: we're looking at a role, and the C_complement is also a role.
% (John has more books in the bedroom than in the kitchen)

find_contrast(role(Role,_),
              role(Role,Quant),
              C,role(Role,Quant),C).


% 2: we're looking at a role, and the C_complement is an NP.
% (John has more books in the bedroom than the kitchen).

find_contrast(C_complement,
              role(Role,Quant),
              role(Role,C),Quant,C) :-
  np_denotation(C_complement).


% 3: we're looking at an NP, and the C_complement is an NP.

find_contrast(C_complement,
              Contrast,
              C,Contrast,C) :-
  np_denotation(C_complement),
  np_denotation(Contrast).


% 4: we're looking at a subject, and the C_complement is something
% that can constrast against the subject's NP.
% (John has more books in the kitchen than Mary).

find_contrast(C_complement,
            subject(Quant),
              subject(Quant1),Contrast,C) :-
   find_contrast(C_complement,Quant,Quant1,Contrast,C).


% 5: we're looking at a subject, and the C_complement is a subject.
% (John has more books in the kitchen than Mary does).

find_contrast(subject(_),
              subject(Quant),
              subject(C),Quant,C).


% 6: CN contrasting; we're looking at an NP, and the C_complement is a cn.
% (More men have books in the kitchen than women).

find_contrast(cn(_,_),
              quant(Det,Cn),
              quant(Det,C),Cn,C).


% 7: VP operator contrasting; we're looking at a subject, and the
% C_complement is a vp_operator.
% (John has more books in the kitchen than Mary is able to).

find_contrast(vp_operator(_,_),
            subject(Quant),
              subject(C),Quant,C).


% 8: Preposition contrasting; we're looking at a role, and the
% C_complement is a role_marker with a "contrastable" role.
% ("more women voted for the proposal than against".)

find_contrast(role_marker(Comp_role),
              role(role_marker(Role),Quant),
              role(C,Quant),role_marker(Role),C) :-
  contrastable_roles(Comp_role,Role).

contrastable_roles(X,Y) :-
  (contrastable_roles_1(X,Y);
   contrastable_roles_1(Y,X)).

% One example of a clause defining a contrastable role pair:

contrastable_roles_1(for,against).


% 9: Recursive cases:

find_contrast(C_complement,
              vp(Head,X,Qlf),
              vp(Head,X,Qlf1),
              Contrasted,C) :-
  find_contrast(C_complement,Qlf,Qlf1,Contrasted,C).

find_contrast(C_complement,
              apply(Qlf,X),
              apply(Qlf1,X),
              Contrasted,C) :-
  find_contrast(C_complement,Qlf,Qlf1,Contrasted,C).

find_contrast(C_complement,
              apply(X,Qlf),
              apply(X,Qlf1),
              Contrasted,C) :-
  find_contrast(C_complement,Qlf,Qlf1,Contrasted,C).

find_contrast(C_complement,[F|R],[F_1|R],Contrast,C) :-
  find_contrast(C_complement,F,F_1,Contrast,C).

find_contrast(C_complement,[F|R],[F|R_1],Contrast,C) :-
  find_contrast(C_complement,R,R_1,Contrast,C).

% Reduce turns the QLF into a logical form by recursively going through it.

reduce(X,X) :- var(X),!.

reduce(apply(F_qlf,Arg_qlf),LF) :-
  reduce(F_qlf,F),
  reduce(Arg_qlf,Arg),
  apply(F,Arg,LF).

reduce(abstraction(X,Qlf),λ(X,LF)) :-
  reduce(Qlf,LF).

% We have to reduce possible degree arguments to a relation.

reduce(relation(Arg_list0),LF) :-
  reduce_arg_list(Arg_list0,Arg_list),
  LF =.. Arg_list.

reduce(conj(Qlf1,Qlf2),LF) :-
  reduce(Qlf1,LF1),
  reduce(Qlf2,LF2),
  conjoin([LF1,LF2],LF).

reduce(subject(Qlf),LF) :- reduce(Qlf,LF).

reduce(quant(Det,Cn),
      λ(Vp_sem,quant(Det_sem,Cn_sem,Vp_sem))) :-
  reduce_det(Det,Det_sem),
  reduce(Cn,Cn_sem).

% "substitute(X)" denotes the higher-order function λP.P(X).
% Our simple-minded implementation of "apply" can't deal with anything more
% complicated than simple substitutions, so this has to be treated as a
% special case.

reduce(trace_np(X),substitute(X)).

reduce(name(N),substitute(N)).

reduce(vp_operator(X,Vp_op),λ(X,Vp_op_sem)) :-
  reduce(Vp_op,Vp_op_sem).

reduce(cn(Head,[]),λ(X,Head_rel)) :-
  Head_rel =.. [Head,X].

reduce(cn(Head,[Adj_qlf]),LF) :-
  Head_rel =.. [Head,X],
  reduce(Adj_qlf,Adj_LF),
  conjoin([λ(X,Head_rel),Adj_LF],LF).

reduce(vp(Head,X,Qlf),λ(X,LF)) :-
  reduce_vp(Qlf,LF,Head,Props-Props).

reduce(trace_vp(X),X).

reduce(trace_s(X),X).

reduce(role_marker(Role),Role).

reduce(s_operator(X,Qlf),λ(X,LF)) :-
  reduce(Qlf,LF).

% This is the expression (*) in section 3.2

reduce(comparison(Contrast,C_complement,C_operator,C,Q,Psil),LF) :-
  N1_condition = relation([C_operator,N,N1]),
  Psil1 = λ(Q,λ(C,Psil)),
  apply(Psil1,q(N1,N1_condition),N_condition0),
  apply(N_condition0,C_complement,N_condition),
  apply(Psil1,q(N,N_condition),Qlf0),
  apply(Qlf0,Contrast,Qlf),
  reduce(Qlf,LF).
```

```
reduce([],[]).

% reduce_vp is a special form of reduce for reducing the list of verb_args.
% The arguments are:
% 1: Qlf - a list of verb_arg nodes
% 2: Piece of LF produced by reducing it
% 3: The head verb
% 4: a d-list of lambda abstractions, which will end up as properties of the
%    event associated with the head verb.

reduce_vp([],evnt(Property),Head,Event_properties-[]) :-
  Head_rel =.. [Head,X],
  conjoin([λ(X,Head_rel)|Event_properties],Property).

reduce_vp([role(role_marker(Role),NP_qlf)|Rest_args],
         LF,Head,Event_properties-[λ(E,Role_relation)|Rest_properties]) :-
  np_denotation(NP_qlf),
  Role_relation =.. [Role,E,X],
  reduce(NP_qlf,NP_LF),
  reduce_vp(Rest_args,Rest_LF,Head,Event_properties-Rest_properties),
  apply(NP_LF,λ(X,Rest_LF),LF).

reduce_vp([role(role_marker(Role),Qlf)|Rest_args],
         LF,Head,Event_properties-[λ(E,Role_relation)|Rest_properties]) :-
  \+np_denotation(Qlf),
  Role_relation =.. [Role,E,Role_LF],
  reduce(Qlf,Role_LF),
  reduce_vp(Rest_args,LF,Head,Event_properties-Rest_properties).

reduce_vp([abstraction(X,Qlf)|Rest],
         LF,Head,Event_properties-[λ(X,Sem)|Rest_properties]) :-
  reduce(Qlf,Sem),
  reduce_vp(Rest,LF,Head,Event_properties-Rest_properties).

reduce_arg_list([],[]).
reduce_arg_list([F0|R0],[F|R]) :-
  reduce_arg(F0,F),
  reduce_arg_list(R0,R).

reduce_arg(X,X) :- var(X),!.
reduce_arg(q(N,Qlf),q(N,LF)) :- !,
  reduce(Qlf,LF).
reduce_arg(X,X).

reduce_det([a,undet,sing],ex) :- !.
reduce_det(['some',undet,sing],ex) :- !.
reduce_det([many,undet,plur],many) :- !.
reduce_det([[],undet,sing],ex) :- !.
reduce_det([[],det,sing],the) :- !.
reduce_det([which,_,_],wh) :- !.
reduce_det([which_pl,_,_],wh_pl) :- !.
reduce_det(q(N,Qlf),q(N,LF)) :- !,
  reduce(Qlf,LF).

np_denotation(quant(_,_)).
np_denotation(name(_)).
np_denotation(trace_np(_)).
```

APPENDIX 3 EXAMPLES

```
1: [vilka,kungar,regerade,l{ngre,{n,Gustav III]

    which kings reigned longer than Gustav III?


          -------  Syntactic form  ---------


wh_q(s(subject(np(det([which_pl
                       undet
                       plur])
                  cn(king
                     []))))
       vp(reign
          finite
          trace_np(A)
          [role(agent
                trace_np(A))
           adv(long
               deg(comp(more
                        name(Gustav III))))]))))


          -------  Logical form  ---------


wh_q(quant(wh_pl
           λ(A,king(A))
           λ(B,evnt(λ(C,[and
                         reign(C)
                         agent(C,B)
                         long(C
                              q(D
                                λ(B,evnt(λ(E,[and
                                             reign(E)
                                             agent(E,Gustav III)
                                             long(E
                                                  q(F
                                                    more(D,F)))])))))])))))

2: [hade,n}gon,kung,fler,s|ner,{n,Gustav Wasa]

    had any king more sons than Gustav Wasa?


          -------  Syntactic form  ---------


y_n_q(s(subject(np(det([some
                        undet
                        sing])
                   cn(king
                      []))))
        vp(have
           finite
           trace_np(A)
           [role(agent
                 trace_np(A))
            role(son
                 np(det([art(many
                             deg(comp(more
                                      name(Gustav Wasa))))
                         undet
                         plur])
                    cn(person
                       []))))]))))


          -------  Logical form  ---------


y_n_q(quant(ex
            λ(A,king(A))
            λ(B,quant(q(C
                        λ(B,quant(q(D
                                    more(C,D))
                                  λ(E,person(E))
                                  λ(F,evnt(λ(G,[and
                                               have(G)
                                               agent(G,Gustav Wasa)
                                               son(G,F)]))))))
                      λ(H,person(H))
                      λ(I,evnt(λ(J,[and
                                   have(J)
                                   agent(J,B)
                                   son(J,I)])))))))

3: [dog,Gustav Wasa,f|re,Gustav II Adolf]

    died Gustav Wasa before Gustav II Adolf?


          -------  Syntactic form  ---------


y_n_q(s(subject(name(Gustav Wasa))
        vp(die
           finite
           trace_np(A)
           [role(agent
                 trace_np(A))
            adv(at_time
                deg(comp(before
                         name(Gustav II Adolf))))])))

          -------  Logical form  ---------


y_n_q(evnt(λ(A,[and
                die(A)
                agent(A,Gustav Wasa)
                at_time(A
                        q(B
                          λ(C,evnt(λ(D,[and
                                       die(D)
                                       agent(D,Gustav II Adolf)
                                       at_time(D
                                               q(E
                                                 before(B,E)))])))))]))))

4: [vilka,kungar,var,mer,ber|mda,{n,Karl XII]

    which kings were more famous than Karl XII?
```

```
         -------  Syntactic form   ---------

wh_q(s(subject(np(det([which_pl
                       undet
                       plur])
                   cn(king
                      []))))
      vp(be
         finite
         trace_np(A)
         [role(agent
               trace_np(A))
          role(ap_comp
               ap_comp(adj(famous
                           deg(comp(more
                                    name(Karl XII))))))])))

         -------  Logical form   ---------

wh_q(quant(wh_pl
           λ(A,king(A))
           λ(B,evnt(λ(C,[and
                         be(C)
                         agent(C,B)
                         ap_comp(C
                                 λ(D,famous(D
                                            q(E
                                              famous(Karl XII
                                                     q(F,more(E,F)))))))])))))
```

```
5: [Kalle,k|pte,fler,b|cker,{n,vad,Marie,kunde,b(ra]

   Charlie bought more books than (what) Mary could carry.

         -------  Syntactic form   ---------

s(subject(name(Charlie))
  vp(buy
     finite
     trace_np(A)
     [role(agent
           trace_np(A))
      role(object
           np(det([art(many
                       deg(comp(more
                                clausal(q(B)
                                        s(subject(name(Mary))
                                          vp(be able to
                                             finite
                                             trace_np(C)
                                             [role(agent
                                                   trace_np(C))
                                              role(inf_comp
                                                   vp(carry
                                                      infinite
                                                      trace_np(D)
                                                      [role(agent
                                                            trace_np(D))
                                                       role(object
                                                            np(det([q(B)
                                                                    undet
                                                                    plur])
                                                               cn(book
[]))))])))])))))
                  undet
                  plur])
              cn(book
                 []))))])))

         -------  Logical form   ---------

quant(q(A
        [and
         more(A,B)
         evnt(λ(C,[and
                   be able to(C)
                   agent(C,Mary)
                   inf_comp(C
                            λ(D,quant(q(B,[])
                                      λ(E,book(E))
                                      λ(F,evnt(λ(G,[and
                                                    carry(G)
                                                    agent(G,D)
                                                    object(G,F)]))))))]))])
      λ(H,book(H))
      λ(I,evnt(λ(J,[and
                    buy(J)
                    agent(J,Charlie)
                    object(J,I)]))))
```

```
6: [Marie,k|pte,en,dyrare,vas,{n,vad,Kalle,k|pte]

   Mary bought a dearer vase than (what) Charlie bought.

         -------  Syntactic form   ---------

s(subject(name(Mary))
  vp(buy
     finite
     trace_np(A)
     [role(agent
           trace_np(A))
      role(object
           np(det([a
                   undet
                   sing])
              cn(vase
                 [adj(expensive
                      deg(comp(more
                               clausal(q(B)
                                       s(subject(name(Charlie))
                                         vp(buy
                                            finite
                                            trace_np(C)
                                            [role(agent
                                                  trace_np(C))
                                             role(object
                                                  np(det([a
                                                          undet
                                                          sing])
                                                     cn(vase
                                                        [adj(expensive
                                                             deg(q(B))
]))])))))))])))))

         -------  Logical form   ---------

quant(ex
      λ(A,[and
           vase(A)
           expensive(A
                     q(B
                       [and
                        more(B,C)
                        quant(ex
                              λ(D,[and
                                   vase(D)
                                   expensive(D,q(C,[]))])
                              λ(E,evnt(λ(F,[and
                                            buy(F)
                                            agent(F,Charlie)
                                            object(F,E)]))))]))])
      λ(G,evnt(λ(H,[and
                    buy(H)
                    agent(H,Mary)
                    object(H,G)]))))
```

```
7: [m}nga,turister,spenderade,fler,kronor,i,London,{n,i,New York]

   many tourists spent more kronor in London than in New York

         -------  Syntactic form   ---------

s(subject(np(det([many
                  undet
                  plur])
              cn(tourist
                 []))))
  vp(spend
     finite
     trace_np(A)
     [role(agent
           trace_np(A))
      role(object
           np(det([art(many
                       deg(comp(more
                                role(in
                                     name(New York)))))
                   undet
                   plur])
              cn(krona
                 [])))
      role(in
           name(London))]))
```

```
       -------  Logical form  ---------

quant(many
    λ(A,tourist(A))
    λ(B,quant(q(C
              λ(B,quant(q(D,more(C,D))
                        λ(E,krona(E))
                        λ(F,evnt(λ(G,[and
                                      spend(G)
                                      agent(G,B)
                                      object(G,F)
                                      in(G,New York)]))))))
             λ(H,krona(H))
             λ(I,evnt(λ(J,[and
                           spend(J)
                           agent(J,B)
                           object(J,I)
                           in(J,London)]))))))
```

8: [Gustav Wasa,hade,fler,{lskarinnor,{n,Kalle,p}stod]

Gustav Wasa had more mistresses than Charlie claimed.

```
       -------  Syntactic form  ---------

s(subject(name(Gustav Wasa))
  vp(have
    finite
    trace_np(A)
    [role(agent
          trace_np(A))
     role(mistress
          np(det([art(many
                      deg(comp(more
                               s_operator(trace_s(B)
                                   s(subject(name(Charlie))
                                     vp(claim
                                       finite
                                       trace_np(C)
                                       [role(agent
                                             trace_np(C))
                                        role(s_comp
                                             trace_s(B))]))))))
                 undet
                 plur])
             cn(person
               []))))))

       -------  Logical form  ---------

quant(q(A
      evnt(λ(B,[and
                claim(B)
                agent(B,Charlie)
                s_comp(B
                       quant(q(C,more(A,C))
                             λ(D,person(D))
                             λ(E,evnt(λ(F,[and
                                           have(F)
                                           agent(F,Gustav Wasa)
                                           mistress(F,E)]))))))))))
    λ(G,person(G))
    λ(H,evnt(λ(I,[and
                  have(I)
                  agent(I,Gustav Wasa)
                  mistress(I,H)]))))
```

9: [fler,kvinnor,{n,m{n,l{ser,H{nt i Veckan]

more women than men read 'News of the World'

```
       -------  Syntactic form  ---------

s(subject(np(det([art(many
                      deg(comp(more
                               cn(man
                                 [])))))
                 undet
                 plur])
             cn(woman
               [])))
  vp(read
    finite
    trace_np(A)
    [role(agent
          trace_np(A))
     role(object
          name(News of the World))]))
```

```
       -------  Logical form  ---------

quant(q(A
        quant(q(B
                more(A
                     B))
              λ(C,man(C))
              λ(D,evnt(λ(E,[and
                            read(E)
                            agent(E,D)
                            object(E,News of the World)]))))))
      λ(F,woman(F))
      λ(D,evnt(λ(G,[and
                    read(G)
                    agent(G,D)
                    object(G,News of the World)]))))
```

10: [Marie,verkar,l{ngre,{n,Kalle]

Mary looks taller than Charlie.

```
       -------  Syntactic form  ---------

s(subject(name(Mary))
  vp(appear
    finite
    trace_np(A)
    [role(agent
          trace_np(A))
     role(ap_comp
          ap_comp(adj(tall
                      deg(comp(more
                               name(Charlie))))))))])

       -------  Logical form  --------

evnt(λ(A,[and
          appear(A)
          agent(A,Mary)
          ap_comp(A
                  λ(B,tall(B
                           q(C
                             tall(Charlie
                                  q(D,more(C,D)))))))))])
```

11: [Kalle,springer,fortare,{n,vad,Marie,kan]

Charlie runs faster than Mary can.

```
       -------  Syntactic form  ---------

s(subject(name(Charlie))
  vp(run
    finite
    trace_np(A)
    [role(agent
          trace_np(A))
     adv(fast
         deg(comp(more
                  vp_operator(trace_vp(B)
                        s(subject(name(Mary))
                          vp(be able to
                            finite
                            trace_np(C)
                            [role(agent
                                  trace_np(C))
                             role(inf_comp
                                  trace_vp(B))]))))))))])

       -------  Logical form  ---------

evnt(λ(A,[and
          run(A)
          agent(A,Charlie)
          fast(A
               q(B
                 evnt(λ(C,[and
                           be able to(C)
                           agent(C,Mary)
                           inf_comp(C
                                    λ(D,evnt(λ(E,[and
                                                  run(E)
                                                  agent(E,D)
                                                  fast(E,q(F,more(B,F)
))])))))])))))])))
```

```
12: [fler,kvinnor,r|stade,f|r,f|rslaget,(n,emot]

    more women voted for the-suggestion than against.

           -------  Syntactic form   ---------

s(subject(np(det([art(many
                      deg(comp(more
                               role_marker(against))))
                   undet
                   plur])
              cn(woman
                 []))))
  vp(vote
     finite
     trace_np(A)
     [role(agent
           trace_np(A))
      role(for
           np(det([[]
                   det
                   sing])
              cn(suggestion
                 []))))]))
```

```
           -------  Logical form   ---------

quant(q(A
        quant(q(B,more(A,B))
              λ(C,woman(C))
              λ(D,quant(the
                        λ(E,suggestion(E))
                        λ(F,evnt(λ(G,[and
                                      vote(G)
                                      agent(G,D)
                                      against(G,F)])))))))
     λ(H,woman(H))
     λ(D,quant(the
               λ(I,suggestion(I))
               λ(J,evnt(l(K,[and
                             vote(K)
                             agent(K,D)
                             for(K,J)]))))))
```