Unscrambling English word order

Allan Ramsay & Helen Seville
Centre for Computational Linguistics
UMIST, PO Box 88, Manchester M60 1QD, England
allan/heleng@ccl.umist.ac.uk

Abstract

We propose a treatment of 'extraposition' which allows items to be assimilated directly even when they appear far from their canonical positions. This treatment supports analyses of a number of phenomena which are otherwise hard to describe. The approach requires a generalisation of standard chart parsing techniques.

1 Extraposition in English

It is widely accepted that sentences such as

1 *I saw the girl who your brother said he fancied.*

2 *The soup was OK, but the main course I thought was awful.*

involve items (*'who'*, *'the main course'*) being found far away from their normal positions (as the complement of *'fancied'* and the subject of *'was awful'*). It seems likely that the modifiers *'in the park'* and *'with all my heart'* in

3 *In the park I met Arthur.*

4 *I believed with all my heart that she loved me.*

are also 'out of position', since you would normally expect VP-modifying PPs of this kind to appear immediately to the right of the modified VP (so that the canonical versions of these sentences would have been *'I met Arthur in the park'* and *'I believed that she loved me with all my heart.'*). There are various reasons for moving things around in this way – moving *'who'* to the left in (1) provides an easy way of picking out the boundary of the relative clause; moving *'the main course'* and *'in the park'* in (2) and (3) puts them into thematically/informationally more prominent positions; and moving the sentential complement *'that she loved me'* to the *right* in (4) reduces the attachment ambiguity that arises in the alternative form.

This is all well-known, and is treated in most grammatical frameworks by hallucinating an item in the canonical position, and then remembering that hallucination up to the point at which the out-of-place item is encountered. Exactly how the hallucination is remembered varies from one framework to another, with unification grammars generally carrying information about it on a category-valued feature (usually called *slash*). The main problem with this approach is that it is difficult to control the situations in which 'traces' of this kind get proposed. (Johnson and Kay, 1994) suggest using 'sponsors' in order to license the introduction of traces, where a sponsor is some item of the required kind that has already been found, and which is hence potentially going to cancel with the trace.

If your parser works from left→right then this will work for items which have been left-shifted, but clearly it cannot work for *right*-shifted items, since the sponsor will not have been found at the time when it is needed. Thus we cannot use a sponsor to justify hallucinating an S-comp for *'believed'* in (4), or for the heavy-NP-shifts in

5 *He gave up his job.*

6 *He built on that spot the most appallingly ugly house.*

In any case, the notion that some item has been left- or right-shifted fails to account for cases of 'intraposition':

7 *I believe Betty is a fool.*

8 *Betty, I believe, is a fool.*

9 *Betty is, I believe, a fool.*

It is at least plausible that (8) and (9) are variants on (7). They're made out of the same words, they have the same truth conditions: the only trouble is that part of the sentence seems to be in the wrong place.

This is analogous to the situation in (2) and (3), where items were moved to the front to make them more prominent. It seems as though in the current case the words *'I believe'* have been shifted into the middle, and parenthesised, to make them *less* prominent. We will show how to deal with this by adapting

our treatment of more orthodox cases of extraposition: in particular, the resulting analysis of (8) and (9) will *not* require us to treat the words *'I believe'* as a phrase.

2 Direct mapping

We start by reconsidering the simple case of extraposition of *'who'* in (1). Suppose we are trying to parse this with a head-corner chart parser. When we encounter *'fancied'*, we will start looking for an NP immediately to the right. We won't find one, so we will be tempted to insert a trace. Before doing so, we will look for a sponsor, and we will have no trouble finding one, namely *'who'* (and possibly others, such as *'the girl'* and *'your brother'*, depending on exactly how the search is implemented). Having introduced the trace, we will build the sentence *'he fancied* $\sqcup$*'*, remembering that there was a trace; and then *'your brother said he fancied* $\sqcup$*'*, again remembering that there was a trace. At this point we will see that we've got a sentence with a trace adjacent to something which could cancel it, so we'll produce the relative clause *'who your brother said he fancied'*.

Why go through this rigmarole? Why go to the bother of finding the sponsor, and then remembering that you have hallucinated something, and then cancelling the hallucination against the sponsor? *Why not just pick up the word* 'who' *as the object of* 'fancied'*?*

There are two obvious reasons why we might not want to include *'who'* immediately. Firstly, the distribution of traces and their fillers is rather restricted in English. We need to be able to state the constraints on what can actually happen, and that may be easier this way. Secondly, most parsing algorithms (certainly most chart-based parsing algorithms) depend upon indexing items by their start and end points, and only combining things if they are in the right place.

We will deal with the practical parsing issue first. Once we have the revised approach to combining items together, it will be easier to see how we can state and utilise the linguistic constraints. The resulting approach will smoothly support linguistic descriptions, and computational analyses, of cases like (8) and (9).

3 Compact and non-compact phrases

Consider the following rather general description of what a chart parser does:

> Fundamental rule of chart parsing (FRCP)
> If X and Y are adjacent structures and X is unsaturated and Y will help saturate it then merge them to form a new, possibly saturated, structure X' covering the combined spans of X and Y.

Of course this has to be tightened up a little to ensure that X and Y are actually adjacent in the required manner – that if X expects Y to be on its right then the end point of X and the start of Y must be identical, and vice versa if Y should be to the left of X; and it may have to be adapted to cope with grammars where items can be combined because one of them modifies the other as well as because one is an argument of the other. Nonetheless, this rather general statement characterises the standard operation of a chart-parser (Kay, 1973; Kaplan, 1973). The difference between using such a parser bottom-up or top-down lies in the source of incomplete edges – in a top-down parser, incomplete edges arise from considering rules which might be expanded to obtain desirable items, in a bottom-up one they arise from considering items that trigger rules by matching some element of the right-hand side.

This algorithm will not work if the language we are dealing with has completely free word order (presumably with enough morphology to help sort out what goes with what), since under these conditions the items that need to be combined may not be adjacent. In a situation like this, we would need to restate the rule as follows:

> Fundamental rule of chart parsing (revised) (FRCP′)
> If X and Y are nonoverlapping structures and X is unsaturated and Y will help saturate it then merge them to form a new, possibly saturated, structure X' covering the combined spans of X and Y.

Suppose, for instance, we took a scrambled version of an English sentence:

10 *him she hard hitting is*

Using FRCP′, we would look for items that could be combined, and all we would find is *'hitting'* and *'him'* (assuming for simplicity that *'hit'* is a simple transitive verb). So we would combine these into a VP containing these two words. After that there are two choices: we might combine *'is'* and *'hitting him'* to get *'is hitting him'*, or we might combine *'hitting him'* and *'hard'* to get *'hitting him hard'*. The first of these would combine further with *'she'* to produce *'she is hitting him'*, but would go no further. The second would combine with *'is'* to make *'is hitting him hard'*, and then with *'she'* to make *'she is hitting him hard'*. *There are no other ways of grouping these words together into phrases* [1].

[1] The way we have grouped them makes it look as though we have reconstructed the declarative sentence *'she is hitting*

To use the new version of the rule, we would need to compare each new phrase with each non-overlapping existing phrase, rather than comparing them with adjacent existing phrases. When we compared new items with adjacent old ones, we could use the start and end positions of the two items as an indexing mechanism for making sure that we only compared items that had some chance of combining. We no longer have such an index, but we can at least introduce a rapid test for whether two items are overlapping.

We will encode the 'span' of a phrase as a bit string with a `1` in position N of the string if the phrase contains the N'th word in the sentence being parsed. Thus the phrase *'hitting him'*, containing the first and fourth words of the input sequence, has `01001` as its span (the 'first' bit in a bit-string like this is actually the rightmost, so that you have to read this from right to left). The span of *'is'* is then `10000`. It is easy to test whether two items are non-overlapping: is their bitwise-& zero? Clearly this is the case here. It is equally easy to compute the span of the result, namely take their bitwise-OR, which in this case is `11001`. When we add *'hard'*, the span of the resulting edge is `11101`, and then when we finally add *'she'* we get `11111`.

We will need two further notions. We need to know whether a phrase has any holes in it – whether we have skipped over intervening material in constructing it. Clearly English is not entirely free-word order, and you can't make phrases by picking words from here and there in a sentence as you please. Keeping track of whether a phrase is 'compact' in this sense is one of the mechanisms for ensuring the actual rules of English are obeyed. If the leftmost word included in a phrase is at position I and the rightmost is at position J, the phrase will be compact if its span is equal to `((1 << (J-I))-1) << I`[2], which we can compute extremely easily. We will mark phrases as $+compact$ or $-compact$ depending on the outcome of this computation.

We also need to distinguish between the compact start and end positions of the phrase and its extreme start and end points. Any headed phrase will have a compact core, made up of those words and phrases which have been found adjacent to the head. It may also have outliers. Thus if we analyse (1) by this approach, at some point we will construct the $-compact$ clause *'who he fancied'*. This will have a compact core, consisting of *'he fancied'*, with *'who'* as an outlier. Its extreme start will therefore be 4, the start of *'who'* (we count from 0), whereas its compact start is 8, the start of *'he'*. The extreme and compact ends are both 10 – the position immediately after the word *'fancied'*.

4 Linear-precedence constraints

FRCP′ is fine for languages with completely free word order. There are not many such languages – English certainly is not one. We therefore need to be able to put constraints on what can actually turn up where.

It is true that English does seem to have a canonical word order which is based on adjacency. For simple sentences complements do occur adjacent to heads, and modifiers do occur adjacent to targets:

11 *Charles kissed Diana.*

12 *The cat sat on the mat.*

Such facts can be captured either by using rewrite rules or by embedding them in complex lexical descriptions, where the lexical item itself carries the required information about what it can combine with. We will take a version of the latter approach, defining *'kissed'*, for instance, as $sign(cat(verb), args([np/right, np/left])$ and *'on'* as $sign(cat(prep), arg([np/right]), target(vp/left))$. These say that *'kissed'* is a verbal item which requires one NP to its right and another to its left to saturate it, and that *'on'* is a preposition which requires an NP to its right, and which will modify a VP once it is itself saturated. These lexical entries are backed up by the following very skeletal rules:

$$sign(X, args(T)) \Longrightarrow sign(X, args([H/right|T])), H$$
$$sign(X, args(T)) \Longrightarrow H, sign(X, args([H/left|T]))$$
$$X \Longrightarrow X, sign(cat(...), target(X/left))$$
$$X \Longrightarrow sign(cat(...), target(X/right)), X$$

These rules lie somewhere between pure categorial grammar (Wood, 1993) and HPSG (Pollard and Sag, 1988; Pollard and Sag, 1994; Sag and Wasow, 1999) – the rules relating to modifiers are not usually part of categorial grammar, and HPSG tends to have more modes of combination (distinguishing, for instance, between subjects and other arguments, or between specifiers and other adjuncts). Nonetheless, the general framework is familiar, and the argument below will apply to any similar analysis.

The positional information encoded in these rules clearly relates to their canonical positions. When we say that *'kissed'* requires an NP to its right, we are referring to what happens in cases like (11), not to cases like

him hard', and it might appear as though *'is she hitting him hard'* is also a possibility. The example, however, relates to a language with English vocabulary but completely free word order, so that what we are investigating is the way that words group together, not some underlying intended sentence of real English.

[2] `A << B` denotes the result of left shifting `A` by `B` places. The computation specified here produces a set of `J-I` 1's, offset to the left by `I`.

13 *Charles danced with Eliza, but Diana he kissed.*

and when we say that the PP that results from saturating *'on'* modifies a VP to its left we are referring to cases like (12), not to

14 *On the mat the cat sat.*

(or even *'On the mat sat the cat.'*, where the expectation that the subject will appear to the left of the verb has also been violated.)

It seems as though we need the standard FRCP to cope with the canonical cases; the weakened FRCP′ to cope with cases where the phrase occurs in some unexpected position; *and something else to constrain the unexpected positions which are actually possible.*

The constraints on what can be moved around take two forms. Firstly, we have say whether something can be moved at all, which we do by introducing a polar-valued feature called *moved*: items which appear away from their canonical positions are marked $moved(right)$ or $moved(left)$, depending on the direction in which they have been shifted. Arguments and targets which aren't allowed to move will be marked $-moved$ by the item that subcategorises for them.

Secondly, we have to specify where those items that can move are allowed to get to. We do this by using linear precedence rules (LP-rules), most of which place constraints on immediate local subtrees. Thus we can say things like

$$\{A,B,C\} : +wh@A \& -wh@B \rightarrow start@A < start@B$$

to capture the fact that if A and B are local subtrees of C, then if A is WH-marked and B is not then A must precede B (A's start must be before B's). Note that the signature of the rule mentions C, even though in this case the body does not.

The facts about extraposition are captured by rules which specify the circumstances under which a local subtree can (or must)be $+moved$. The key constraints for trees representing structures with verbal heads are as follow:

$$\{A,B,C\} : -wh@A \& -aux@C \& A = subject@C \rightarrow -moved@A$$

(if A is the subject of C, where A is not WH-marked and C is not an auxiliary, then A may not be moved)

$$\{A,B,C\} : moved@A = right \rightarrow \exists X(X \in dtrs@C \; \& \; start@core@C < start@X) \; \& \; start@X < start@A)$$

(if A has been *right*-shifted, then C had better have some other daughter X between C's head and A. The full rule says that C must be heavier than X, where we take 'C is heavier than X' to mean that C covers more words than X, so that this rule covers (4), (5) and (6)).

There are a number of other such rules, of which the most complex relates to *'that'*-clauses (denoted by $+comp$). The description of such clauses comes in two parts, one to say that $-wh$ phrases may not be extracted and one to say that $+wh$ phrases must be extracted.

(i) $\{A,B,C\} : C \in clause \; \& \; +comp@C \; \& \; +compact@C \rightarrow -wh@B$

(ii) $\{A,B,C\} : C \in clause \; \& \; +comp@C \; \& \; -compact@C \rightarrow +wh@B$

The first part of this says that if combining A and B produces a *'that'*-clause C, then if C is $+compact$ (so nothing has been extracted from it) then it had better be $-wh$ (in other words, nothing properly inside it can be $+wh$). If on the other hand C is $-compact$ then there must be something extracted from it, in which case the item which has been extracted must mark it as $+wh$. These rules cover the (un)acceptability of

15 *I know that she loves me.*

16 ** I know me that she loves.*

17 *who I know that she loves.*

18 ** I know that who she loves.*

5 Intraposition

The rules in Section 4 provide a reasonable account of simple extraposition (both left and right) from clauses. We now return to

7 *I believe Betty is a fool.*

8 *Betty, I believe, is a fool.*

9 *Betty is, I believe, a fool.*

Suppose we use FRCP′, with no LP-rules, to analyse (8). We will get, among other things, the phrases and part phrases shown in Fig. 1 (the commas are treated as lexical items, so that *'Betty'* starts at 0, the first comma at 1, and *'I'* at 2, and so on).

The first couple of steps are straightforward: *'a fool'* results from combining *'a'* and *'fool'*. It has no holes in it, its extreme start and end are the same as its compact start and end. Then *'is a fool'* results from combining *'is'* and *'a fool'*, and again all the pieces are in the right place, so the extreme start and end are the same as the compact start and end and the phrase is $+compact$.

At step 3, *'Betty'* is integrated as the subject of *'is a fool'*. The result starts at 0, since that's where

	phrase	start	end	xstart	xend	compact
1	a fool	6	8	6	8	+
2	is a fool	5	8	5	8	+
3	Betty is a fool	5	8	0	8	−
4	believe Betty is a fool	3	4	0	8	−
5	I believe Betty is a fool	2	4	0	8	−
6	,I believe Betty is a fool	0	4	0	8	−
7	,I believe Betty is a fool,	0	8	0	8	+

Figure 1: Analysis of (8)

'Betty' starts, and is *−compact*, since it does not include all the intervening words.

At 4 this *−compact* sentence becomes the complement of *'believe'*. The result is again *−compact*, since it fails to include the word *'I'* or the two commas which appear between its start and end points. The compact core is now the word *'believe'*, so the compact start and end are 3 and 4.

At 5, the VP *'believe Betty is a fool'* combines with *'I'* to produce *'I believe Betty is a fool'*. The two commas then combine with this phrase, marking it as being parenthetical and, when the second comma is included, finally marking it as *+compact*.

Similar structures would be created during the processing of (9), with the only difference being that *'is a fool'* would be the first *−compact* phrase found. Apart from that the analysis of (9) would be identical to the analysis of (8).

There are two problems with this approach to sentences of this kind: (i) because we obtain identical syntactic analyses of (7), (8) and (9), then any compositional semantics will assign all three the same interpretation. This is not entirely wrong: I cannot fairly say any of these sentences unless I do believe that Betty is a fool. But it is also clearly not entirely right, since it misses the difference in emphasis. We will not discuss this any further here. (ii) because we are not applying the LP-rules, we get rather a large number of analyses. Without LP-rules, we get a single analysis of *'I believe Betty is a fool'*, having constructed 23 partial and complete edges. For *'Betty, I believe, is a fool'* we get three analyses (including the correct one) having constructed 101 edges. Most of these arise from the presence of the commas, since we have to allow for the possibility that each of these commas is either an opening or closing bracket, or a conjunction in a comma-separated list of conjuncts. Others arise from the fact that we have removed *all* the LP-rules, so that we are treating English as having completely free word order. Case marking still provides some constraints on what can combine with what, so that in the current case *'I'* is the only possible subject for *'believe'* and *'Betty'* is the only possible subject for *'is'*. If we had been dealing with

19 *Betty, Fred believes, is a fool*

then we would have had six analyses from 107 edges, with the new ones arising because we had assigned *'Fred'* as the subject of *'is'* and *'Betty'* as the subject of *'believes'*.

Clearly we need to reinstate the general LP-rules, whilst allowing for the cases we are interested in. These cases are characterised in two ways: (i) some word that requires a sentence has occurred in a context where a 'split' sentence is available, and (ii) this word is adjacent to a parenthetical comma. The statement of this rule is rather long-winded, but the result is to provide a single analysis of (8) from 66 edges, and a single analysis of (9) from 70 edges.

6 *'more X than Y'*

Most cases of extraposition in English involve sentences, but there are a number of other phenomena where items seem to have been shifted around. Consider for instance the following examples:

20 *George ate more than six peaches.*

21 *Harriet ate more peaches than pears.*

22 *Ian ate more peaches than Julian.*

In (20), *'more than six'* looks like a complex determiner. How many peaches did George eat? More than six. The easiest way to analyse this is by assuming that *'more'* subcategorises for a '*than*-phrase'.

In (21) and (22), however, the *than*-phrase seems to have become disjointed. It still seems as though *'more'* heads a complex determiner, since (22) would support the answer *'more than Julian'* to the question *'How many peaches did Ian eat?'* [3].

We therefore introduce lexical entries for *'more'* and *'than'* which look roughly as follows:

[3] though (21) does not seem to support *'more than pears'* as an answer to *'How many peaches did Harriet eat?'*. The problem seems to be that NP complements to *'than'* are actually elliptical (see below), and it seems to be harder to recover the ellipsed sentence *'More than she ate pears'* than to recover *'More than Julian ate peaches'* or *'more than Julian ate'*.

$$\left[\begin{array}{l}\text{sign}\\ \\ \\ \\ \\ \\ \\ \end{array}\left[\begin{array}{l}\text{phon } \textit{`more'}\\ \text{syn}\left[\begin{array}{l}\text{nonfoot}\left[\text{head}\left[\text{cat specifier}\right]\right]\\ \ldots\end{array}\right]\\ \text{foot}\left[\text{wh} <>\right]\\ \text{subcat}\left\langle\left[\text{sign}\left[\begin{array}{l}\text{phon } \textit{`than'}\\ \text{subcat} <>\end{array}\right]\right]\right\rangle\\ \text{semantics} \ldots\end{array}\right]\right]$$

$$\left[\begin{array}{l}\text{sign}\\ \\ \\ \\ \\ \\ \end{array}\left[\begin{array}{l}\text{phon } \textit{`than'}\\ \text{syn}\left[\begin{array}{l}\text{nonfoot}\left[\text{head}\left[\text{cat than}\right]\right]\\ \ldots\end{array}\right]\\ \text{foot}\left[\text{wh} <>\right]\\ \text{subcat}\left\langle \text{X}\right\rangle\\ \text{semantics} \ldots\end{array}\right]\right]$$

The entry for *'more'* says that it will make a specifier if it finds a saturated phrase headed by *'than'*. The entry for *'than'* says that it will make a phrase of the required type so long as it finds some argument X. We know very little about X. In (20) it is a number, in (21) and (22) it appears to be an NP. In fact, as (Pulman, 1987) has shown, the best way to think about these examples is by regarding them as elliptical for the sentences

23 *Harriet ate more peaches than she ate pears.*

24 *Ian ate more peaches than Julian ate peaches.*

Other kinds of elliptical phrase are permitted, as in

25 *Keith ate more peaches than Lucy did.*

or even

26 *Martha ate more ripe than unripe peaches.* [4]

We therefore allow arbitrary phrases as the argument to *'than'*. All we need now are the LP-rules describing when arguments of *'than'* should be extraposed. These simply say that if you are combining the determiner *'more'* with a *'than'*-phrase, then if the sole daughter of the *'than'*-phrase is a number then it must not be shifted, and if it is not then it must be right-shifted.

$$\begin{array}{lll}\text{(i)} \{A,B,C\} & : A \in \textit{det}\&\textit{phon}@A = \textit{`more'} \\ & \quad \& \; \textit{cat}@B = \textit{than} \& \; \textit{dtrs}@B = <D> \\ & \quad \& \; (D \in \textit{num or } D \in \textit{adj}) \\ & \rightarrow \; -\textit{moved}@B \\ \text{(ii)} \{A,B,C\} & : A \in \textit{det}\&\textit{phon}@A = \textit{`more'} \\ & \quad \& \; \textit{cat}@B = \textit{than} \\ & \quad \& \; \textit{not}(\textit{dtrs}@B = <D> \\ & \qquad \& \; (D \in \textit{num or } D \in \textit{adj})) \\ & \rightarrow \; \textit{moved}@B = \textit{right}\end{array}$$

With these LP-rules, we get appropriate structural analyses for (20)–(25). We do not, however, currently have a treatment of ellipsis. We therefore cannot provide sensible semantic analyses of (21) and (22), since we cannot determine what sentences *'peaches'* and *'Julian'* are elliptical for (imagine, for instance, trying to decide whether *'Eagles eat more sparrows than crows'* meant *'Eagles eat more sparrows than crows eats sparrows'* or *'Eagles eat more sparrows than eagles eat crows'*).

If the structure of *'more peaches than pears'* involves a displaced *'than'*-phrase, then it seems very plausible that the same is true for

27 *Nick wrote a more elegant program than Olive.*

28 *Peter wrote a more elegant program than that.*

This is given further support by the acceptability of examples like

29 *A program more elegant than that would be hard to find.*

where the *'than'*-phrase is adjacent to the modified adjective *'elegant'* rather than to the noun *'program'* which is modified by the whole phrase *'more elegant than that'*.

Frustratingly, it just does not seem possible to reuse the lexical entry above for *'more'* to cope with these cases. In (20)–(25), *'more'* made a determiner when supplied with an appropriate *'than'*-phrase. For (27)–(29) it needs to make something which will combine with an adjective/adverb to produce an intensified version of the original. We therefore need the following entry:

$$\left[\begin{array}{l}\text{sign}\\ \\ \\ \\ \\ \\ \\ \\ \end{array}\left[\begin{array}{l}\text{phon } \textit{`more'}\\ \text{syn}\left[\begin{array}{l}\text{nonfoot}\left[\text{head}\left[\text{cat intensifier}\right]\right]\\ \ldots\end{array}\right]\\ \text{foot}\left[\text{wh} <>\right]\\ \text{subcat}\left\langle\left[\text{sign}\left[\begin{array}{l}\text{phon } \textit{`than'}\\ \text{subcat} <>\end{array}\right]\right]\right\rangle\\ \text{target}\left[\text{sign}\left[\text{syntax}\left[\text{head}\left[\text{cat adj}\right]\right]\right]\right]\\ \text{semantics} \ldots\end{array}\right]\right]$$

[4] Note that in this case the argument of *'than'* is *not* displaced.

This needs a *'than'*-phrase to saturate it, and once it is saturated it will combine with an adj (adjective or adverb) to make a new adj. There are two questions to be answered: should such a complex adj appear to the left or right of its target, and should the *'than'*-phrase be extraposed or not?

(28) and (29) show that these questions are intimately connected. If the *'than'*-phrase is right-shifted, then the resulting modifier appears to the left of its target (28); if it is not, then the modifier appears to the right (29). This is exactly what is predicted by (Williams, 1981)'s suggestion that head-final modifiers generally appear to the left of their targets (*'a quietly sleeping man'*) whereas non-head-final ones appear to the right (*'a man sleeping quietly'*). All we need to do is to make right-shifting of the *'than'*-phrase optional, and to invoke Williams' rule, using the compact core of the modifier. Thus the compact modifier *'more elegant than that'* from (29) is not head final, since the whole thing is compact but the head, *'elegant'*, is not the last word; the non-compact one *'more elegant ... than that'* from (28) *is* head final, since this time *'elegant'* is the last word in the compact core *'more elegant'*. Hence *'more elegant than that'* follows its target and *'more elegant ... than that'* precedes it. No new LP-rules are required, and no changes to the general rule for locating modifiers are required.

7 Conclusions

We have shown how retrieving displaced items directly, rather than positing a trace of some kind and then cancelling it against an appriate item when one turns up, can provide treatments of left- *and* right-extraposition which display the advantages that (Johnson and Kay, 1994) obtain for left-extraposition. This approach to extraposition can be extended to deal with 'intraposition' and to cases where items have been extracted from non-clausal items. In order to avoid overgeneration, we needed to introduce a set of LP-rules which are applied as phrases are constructed in order to ensure that items have not been shifted to unacceptable positions. The extra computation required for checking the LP-rules has no effect on the complexity of the parsing process, since they simply add a constant (and fairly small) extra set of steps each time a new edge is proposed. As a rough performance guide, the grammar generates five analyses for

30 *He built on that site a more unattractive house than the one which he built in Greenwich.*

on the basis of 237 edges (the different global analyses arise from the attachment ambiguities for the various modifiers), and takes 4.1 seconds to do so (compiled Sicstus on a Pentium 350). This sentence contains a right-shifted NP, which itself contains a *'more ... than ...'* construction and also a relative clause with a left-shifted WH-pronoun, and hence could be expected to cause problems for approaches using sponsors, while

8 *Betty, I believe, is a fool.*

takes 0.27 seconds. The worst case complexity analysis for this kind of approach is fairly awful ($o(P^2 \times 2^{2(N-1)})$ where P is the number of unsaturated edges in the initial chart and N is the length of the sentence (Ramsay, in press)). In practice the LP-rules provide sufficient constraints on the generation of non-compact phrases for performance to be generally acceptable on sentences of about twenty words.

References

M Johnson and M Kay. 1994. Parsing and empty nodes. *Computational Linguistics*, 20(2):289–300.

R M Kaplan. 1973. A general syntactic processor. In R. Rustin, editor, *Natural language processing*, pages 193–241, New York. Algorithmics Press.

M Kay. 1973. The MIND system. In R. Rustin, editor, *Natural Language Processing*, pages 155–188, New York. Algorithmics Press.

C J Pollard and I A Sag. 1988. *An Information Based Approach to Syntax and Semantics: Vol 1 Fundamentals.* CSLI lecture notes 13, Chicago University Press, Chicago.

C J Pollard and I A Sag. 1994. *Head-driven Phrase Structure Grammar.* Chicago University Press, Chicago.

S G Pulman. 1987. Events and VP-modifiers. In B.G.T. Lowden, editor, *Proceedings of the Alvey Sponsored Workshop On Formal Semantics in Natural Language Processing*, Colchester. University of Essex.

A M Ramsay. in press. Parsing with discontinuous phrases. *Natural Language Engineering.*

I A Sag and T Wasow. 1999. *Syntactic theory: a formal introduction.* CSLI, Stanford, Ca.

E Williams. 1981. On the notions 'lexically related' and 'head of a word'. *Linguistic Inquiry*, 12:254–274.

M M Wood. 1993. *Categorial Grammars.* Routledge, London.