Efficient Parsing for Word Structure

Anna Maria Di Sciullo
Université du Québec à Montréal
C.P. 8888, Succursale Centre Ville,
Montréal, Qc H3C 3P8, Canada
di_scuillo.anne-marie@uqam.ca

Sandiway Fong
NEC Research Institute
4 Independence Way,
Princeton, NJ 08540, USA
sandiway@research.nj.nec.com

Abstract

We describe an efficient parser for morphological analysis for a theory of X′-selection within the Strict Asymmetry framework. The bottom-up parser is based on a full LR(1) analyzer, modified to handle the processing of both overt and covert morphemes in a locally deterministic and efficient manner.

1 Introduction

We describe an efficient parser for morphological analysis in the Strict Asymmetry framework. The parser makes direct use of X′-phrase structure rules to handle a variety of examples from derivational morphology. The theory captures basic facts with respect to configurational asymmetries between prefixes and suffixes as well as the overtness or covertness of affixes given independently motivated cross-linguistic parameters. Moreover, we distinguish between internal and external prefixation in the geometry of word structure. The implemented system is based on a non-deterministic full LR(1) analyzer. We show how the bottom-up machine can be constrained to handle the processing of both covert and overt morphemes in a locally deterministic and computationally efficient manner.

2 A Theory of Morphology Based on Asymmetrical Relations

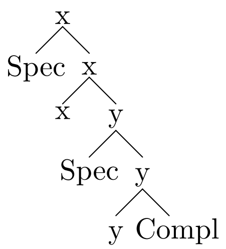
We take derived words to be sets of relations, the main property of which is asymmetry, as proposed in (Di Sciullo, 1999). The asymmetrical nature of derived words is a consequences of the nature of the operations of the grammar, structure building and linking operations. The operations generate and relate elements in asymmetrical relations. A derivation crashes if it is not constituted of asymmetrical relations at every step.

(1) Grammatical relations are asymmetrical

This theory makes correct predictions with respect to the restrictions on derivational morphology. This is quite an unexpected result, as the general practice in the area is to take morphology to be the locus of irregularity in grammar. We show that a theory of grammatical relations, based on (1), leads to a direct account of the restricted nature of morphological relations, thus making the so-called irregularities to follow in a principled way. We also show that cross-linguistic variation with respect to the overt/covert-ness of affixes follows in a principled way.

2.1 Suffixation

(2)



Considering the complete set of category deriving affixes, we expand the idea proposed in (Di Sciullo, 1995) that category deriving affixes, such as -*er*, -*able* and -*ize* project an X′-structure and include the X′-structure of the root they shift with. This is depicted in (2), using x and y for the affix and the root respectively, Spec and Compl for Argument (A) feature positions.

Typical restrictions on affix-root combinations provide empirical evidence that asymmetrical relations restrict derivational morphology. These restrictions pertain to the selectional and linking properties of derivational affixes. We thus take derivational affixes to differ with respect to whether or not the root they combine with projects an A-Spec and/or A-Compl. Derivational affixes also differ with respect to the Linking of the Spec they project to the Spec or the Compl of the root they combine with. The Linking relation is subject to the requirement that every *non* A-Spec must be linked to an A-Spec or A-Compl; whereas an A-Spec or A-Compl may remain unlinked in word structure.

This theory makes correct predictions with respect to word formation independently of the categorial properties of affixes and roots. This runs counter to the traditional practice in Derivational Morphology which restricts the combination of affixes and roots on a categorial base by means of strict subcategorization features such as -*able*: [V __], -*er*: [V __], -*ize*: [N,A __], as in (Anderson, 1992) and (Lieber, 1992). C-, i.e. categorial, selection for derivational affixes gives rise to overgeneration in morphology. It makes wrong predictions with respect to the facts because it does not express the strict asymmetrical property of morphological relations. Strict Asymmetry Theory makes the correct predication, as we illustrate with productive adjectival, nominal and verbal derivation. The facts are summarized below.

(3) a. wash*able*/lov*able*
(root: A-Spec & A-Compl)

b. *fall*able*/*arriv*able*
(root: no A-Spec)

c. *shin*able*/*snor*able*
(root: no A-Compl)

(4) -*able*

a. Selects for A-Spec & A-Compl

b. Links to A-Compl

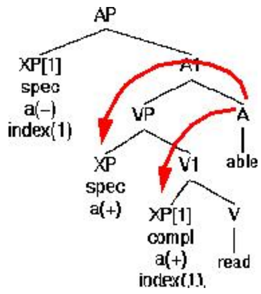


Figure 1: Example of analysis for suffix -*able*

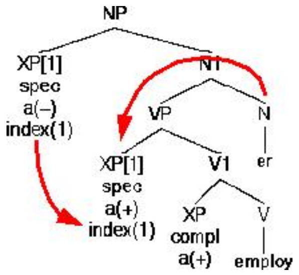


Figure 2: Example of analysis for suffix -*er*

For example, figure 1 illustrates the analysis obtained by the implemented system when parsing the example *readable*. For convenience, the two features `spec` and `compl` are used to indicate (configurationally defined) specifiers and complements, respectively. The feature `a(`±`)` encodes the A/non-A distinction. For instance, the lexical entry for *read* will contain two `a(+)` values, one for the specifier and one for the complement, whereas the corresponding entry for *shine* will just contain a single `a(+)` for the specifier. Finally, linking is indicated via the feature `index(`i`)`: a non A-position linked to an A-position will share a common index value i.

(5) a. kill*er*/hitt*er*/produc*er*
(root: A-Spec & A-Compl)

b. swimm*er*/box*er*/dream*er*
(root: no A-Compl)

c. *fall*er*/*arriv*er*/*depart*er*
(root: no A-Spec)

(6) -*er*

a. Selects for A-Spec

b. Links to A-Spec

Given the constraints in (6), figure 2 illustrates the parse assigned to *employer*.

(7) a. *sister*ize*/*friend*ize*/*equal*ize*
(root: A-Spec & A-Compl)

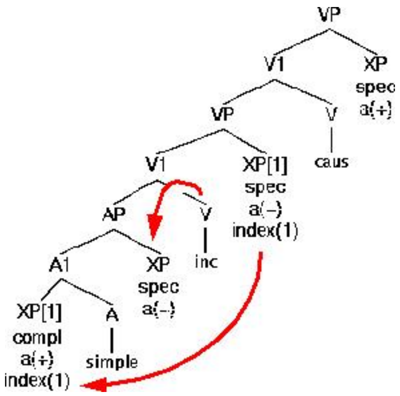


Figure 3: Example of analysis for suffix *-ify*

b. solid*ify*/cert*ify*/liqu*ify*
(root: no A-Spec)

c. union*ize*/computer*ize*/system*ize*

(no A-Spec nor A-Compl)

(8) *-ize/-ify*
a. Selects for non A-Spec
b. Links to A-Compl

Figure 3 shows the parse assigned to *simplify*. Note that -*ify* is decomposed into a pair of abstract morphemes, `caus` (causative) and `inc` (inchoative). Under the current implementation, `caus` directly selects for `inc`. In turn, the lexical entry for `inc` states the complement domain constraints given in (8). Note that specifiers are placed to the right of the head in figure 3. (Di Sciullo and Fong, 2000) show that right specifiers are necessary to preserve efficient prediction during left-to-right parsing.

2.1.1 Zero Derivation

Languages differ with respect to the overt/covert-ness of verbal affixes. This is the case for English and French, as in examples (9)-(10). Either the verbal inchoative suffix alone or the causative-inchoative combination of verbal affixes are overt in both French and English, as in (10), or only in French, as in (9). We observe that only one overt verbal suffix is necessary and sufficient when the root is a noun, as in (11), whereas both verbal suffixes are necessary when the root is adjectival, as in (12).

(9)	a.	fin-ir	‘to end’
	b.	début-er	‘to begin’
(10)	a.	form-al-i-(z)-e-r	‘to formalize’
	b.	public-i-(z)-e-r	‘to publicize’

(11) * fin-i-(z)-e-r/*début-i-(z)-e-r

(12) a. * form-al-i-r/*public-i-r
b. * form-al-e-r/*public-e-r

,

The overt/covert nature of the verbal suffix follows from a difference in strength of the INFL, or inflection, parameter (V to I parameter). The values of the parameter are strong in Romance, while they are weak in English. Assuming that a strong value for a given parameter gives rise to PF visibility in morphological objects, we correctly predict the facts.

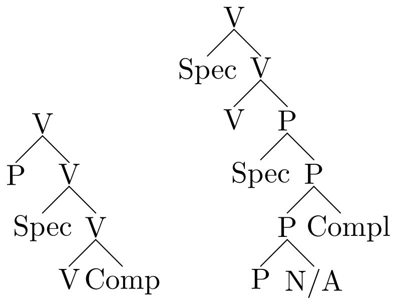
2.2 Prefixation

We assume that prefixes are part of adjunct structures. Thus, they differ from suffixes with respect to the asymmetrical relation they are a part of. We also assume that adjunct are subject to the Adjunct Identification Condition (AIC), as proposed in (Di Sciullo, 1997). According to the Adjunct Identification Condition, an adjunct must identify an unspecified feature of the category it adjoins to. This identification restricts the composition and linking relations a prefix is subject to. If there restrictions do not obtain the derivation crashes. There is empirical evidence to the effect that the composition and linking relations are strictly asymmetrical.

Considering the complete set of prepositional prefixed verbal constructions in French, we shown in (Di Sciullo, 1994) that some prefixes, such as *re-* and *un-*, are adjunct to maximal verbal projection while other prefixes, such as *in-* and *a-*, are adjuncts to the minimal verbal projection.

For concreteness, let us assume here the structures in (13), where the prepositional prefix (P) is adjoined either outside (13a) or inside (13b) a verbal projection. We assume, as in (Chomsky, 1995; Chomsky, 2001) that the grammar includes no maximal or minimal categories primitives, but that these notions are derived from the configurations and that X′ positions are virtual within word-structure. In the structure in (13a), the prefix is adjoined to a maximal projection and in (13b) the prefix is adjoined to a minimal projection.

(13)



(a) (b)

The structure in (13a) corresponds to verbal structures including external prefixes such as *un-* and *re-*, and the structure in (13b) corresponds to verbal structures including internal prefixes such as *en-*.

This configurational asymmetry does not manifest itself directly on the basis of verbal structure with one prefix only, as it is the case in the examples in (14) and (15). The asymmetry reveals itself when considering multi prefix verbal structures, as evidenced below.

(14) a. to re-load
b. to un-load

(15) a. to en-code
b. to en-large

In fact, the structural asymmetry accounts for several properties of prefixed verbal constructions. First, it accounts for the fact that external prefixes must precede internal prefixes in verbal structures, as illustrated in (16). It also captures the fact that prefixed denominal and deadjectival verbs differ from prefixed verbs in that the former may not take an external prefix without an intervening internal prefix, as evidenced in (17). Moreover, these structural differences account for the fact that external prefixes may be iterated and co-occur, as is the case for PP adjuncts, but not internal prefixes which are more closely related to the argument structure of the projection they adjoin to, this is illustrated in (18). In fact, external prefixes may not affect the argument structure of the verbal projection, whereas internal prefixes may do so, as shown in (19). The adjunction of an iterative prefix to a transitive verb neither will give rise to argument structure alternation nor will add an internal argument to the verbal projection, as in (19a). The adjunction of a directional prefix may have this effect, as in (19b).

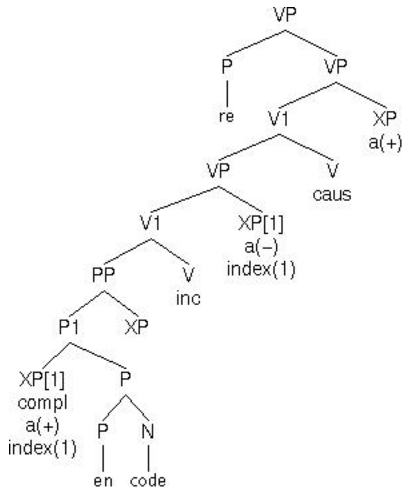


Figure 4: Example of external and internal prefix analysis: *re-encode*

(16) a. re-en-code
b. * en-re-load

(17) a. re-en-large
b. * en-re-large

(18) a. re-re-code
b. * en-en-code

(19) a. to (re)-close the door
b. to en-close a card in the envelope

The prefixes differ with respect to the AIC. Thus, *re-* identify by Linking an unspecified feature of a maximal projection; whereas prefixes such as *en-* identify by Linking an unspecified feature of a minimal projection. This accounts for the restrictions observed above with respect to the composition of prefixes and roots. Figure 4 illustrates the parse obtained for (16a), *re-encode.*

2.2.1 Zero Derivation

While external prefixes must generally be overt, (20), languages differ with respect to the overt/covert-ness of internal prefixes to V, (21), and to nouns or adjectives in derived verbal structures, (22).

(20)	a.	refaire	'to redo'
	b.	défaire	'to undo'
(21)	a.	apporter	'to carry'
	b.	emporter	'to carry from'
(22)	a.	a-tiède-ir	'to cool down'
	b.	en-boite-er	'to box'

The overt/covert nature of the prefix follows from a difference in strength of the P parameter. The values of the parameter are strong in Romance, while they are weak in English. Assuming that a strong value for a given parameter gives rise to PF visibility in morphological objects, we correctly predict the facts pertaining to variation in the covert/covertness property of prepositional prefixes in verbal structures.

3 Algorithm

We assume the framework of the full one-symbol lookahead LR(1) parsing algorithm, as introduced by (Knuth, 1965), for the grammar of X′-structure. The bottom-up, left-to-right parser is implemented in PROLOG, using the default backtracking mechanism to split the computation at LR conflict points. We also assume in this paper that the input word has undergone a pre-processing initial segmentation phase; that is, the input to the parser consists of roots and overt morphemes. For a fully integrated approach to efficient morphological analysis, a packed-forest data structure along the lines of (Tomita, 1986) can be used to minimize redundant computation in the case of ambiguity in initial segmentation.[1] In this paper, we restrict our attention to the elimination of LR(1) table conflicts to render deterministic the parsing of X′-morphological structure.

3.1 Non-Determinism

The X′-grammar generates a 117 state LR(1) machine. The associated LR(1) table consists of a list of possible instructions, or actions of type SHIFT or REDUCE, for the machine to take for a state and given input morpheme. The LR(1) algorithm makes theoretically maximal use of the input morpheme in discriminating between actions. In fact, lookahead is effective in reducing the number of conflicts for 71% of the states. The table in figure 5 summarizes the percentage of table entries that contain zero, one and two conflicts.[2] An entry with zero conflicts is a deterministic entry. For comparision, the corresponding statistics for the weaker LALR(1) method, which merges states with a common nucleus differing only in the lookahead set, are also given. In the next two subsections, we discuss the complete elimination of LR conflicts through extra-grammatical constraints for overt morphemes, reserving the treatment of covert morphemes for section 4.

No. of	Percentage of table entries	
conflicts	LR(1)	LALR(1)
0	13%	8%
1	50%	58%
2	37%	34%

Figure 5: LR(1) Conflict Statistics

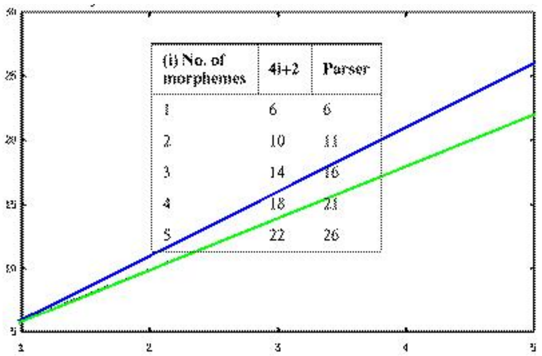
3.2 Deterministic Suffix Computation

An important goal for the construction of a computational model of grammar is that an implementation should track linguistic constraints as closely as possible without introducing extraneous computational, i.e. non-linguistically justified, choice points. In this section, we describe how extra-grammatical constraints can be imposed on the LR(1) analyzer to eliminate all computational choice points outside of the theory.

In the case of suffixes and root forms, each morpheme locally projects X′-phrase structure. Hence, an LR parser must perform a single SHIFT (in the case of an overt head morpheme) plus three reduce actions in order to construct an (empty) specifier and phrases X′ and XP, as required by X′-syntax. Since

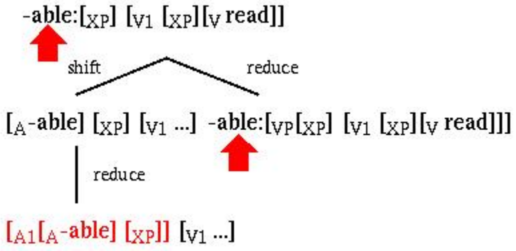
[1] Future versions of the analyzer will employ this data structure.

[2] There are no entries with three or more conflicts.



(i) No. of morphemes	4i+2	Parser
1	6	6
2	10	11
3	14	16
4	18	21
5	22	26

Figure 6: Minimum suffixation steps



-able:[XP] [V1 [XP][V read]]
shift
reduce
[A-able] [XP] [V1 ...]
-able:[VP[XP] [V1 [XP][V read]]]
reduce
[A1[A-able] [XP]] [V1 ...]

Figure 7: *Read-able* SHIFT/REDUCE conflict

maximal projects headed by individual morphemes recurse, there will no charge for the complement, except in the case of the last empty complement at the bottom of the tree. The only other action to be counted is the accepting computation signalling parse completion.[3] Therefore, the minimum number of LR computational steps is given by $4i + 2$ where i is the number of morphemes in the analysis of the word. As figure 6 suggests, the machine described so far requires linearly more computational steps than the (theoretical) minimum demanded by linguistic theory. To be more precise, it introduces a single extraneous computational choice point for each suffix.

For example, figure 7 illustrates the garden pathing of the LR machine on the example *read-able*. Consider the state of computation represented by -able:$[_{\mathrm{XP}}\ ][_{\mathrm{V1}}\ [_{\mathrm{XP}}\ ][_{\mathrm{V}}\ \text{read}]]$, shown at the top of the diagram. Here, the suffix *-able* is the current input morpheme. On the stack are two phrases $[_{\mathrm{XP}}\ ]$ and $[_{\mathrm{V1}}\ [_{\mathrm{XP}}\ ][_{\mathrm{V}}\ \text{read}]]$. The pre-computed LR(1) table for the X′-grammar indicates a SHIFT/REDUCE conflict. In other words, at this point the analyzer has the option of shifting *-able* onto the stack, as represented by the left branch of computation, or performing a reduction on the two phrases residing on the stack, producing the maximal phrase $[_{\mathrm{VP}}\ [_{\mathrm{XP}}\ ][_{\mathrm{V1}}\ [_{\mathrm{XP}}\ ][_{\mathrm{V}}\ \text{read}]]]$, as shown on the right.

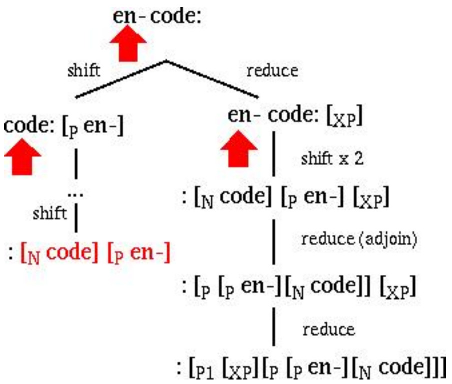


Figure 8: *En-code* SHIFT/REDUCE conflict

The left branch of computation will hit a dead end at the next step. Once *-able* has been shifted onto the stack, the LR(1) table instructs the analyzer to combine $[_{\mathrm{A}}$ -able] and $[_{\mathrm{XP}}\ ]$, i.e. make $[_{\mathrm{XP}}\ ]$ the complement of *-able*. This reduction fails to complete due to the selectional constraints stated earlier in (4). Thus only the right branch proceeds. Nevertheless, the analyzer has incurred a charge of one LR computation step. This charge can be avoided and the parser made deterministic by imposing the conflict resolution rule in (23).

(23) **Suffix Resolution Rule:**
Select REDUCE (over SHIFT) when $[_{\mathrm{XP}}\ ]$ is on the top of the stack.

Alternatively, we might try to restate (23) in terms of a constraint on stack depth; that is, stack depth must not exceed two. However, as we will see in the discussion on prefixation below, stack depth discipline cannot be maintained, and thus (23) is preferred.

3.3 Deterministic Prefix Computation

Garden pathing also occurs in LR prefix computation. Figure 8 illustrates the problem for the directional prefix *en-* in the analysis of *en-code*. At the top, we have both *en-* and *code*

[3] Of course, the accept action is simply another SHIFT action; i.e. SHIFT $, where $ is the sentinel.

in the input. There is a choice between shifting *en-* or generating an empty category $[_{\mathrm{XP}}\ ]$. The correct choice is represented by the right branch of the computation tree. Both input items have to be shifted before a series of two reduce actions can be performed to produce $[_{\mathrm{P1}}\ [_{\mathrm{XP}}\ ][_{\mathrm{P}}\ [_{\mathrm{P}}$ en-$][_{\mathrm{N}}$ code$]]]$. Note that the stack depth after the double shift is three, and thus the limit of two proposed in the previous section cannot be used. We propose the following prefix resolution rule:

(24) **Prefix Resolution Rule:**
Select REDUCE (over SHIFT) when a prefix is on the top of the stack.

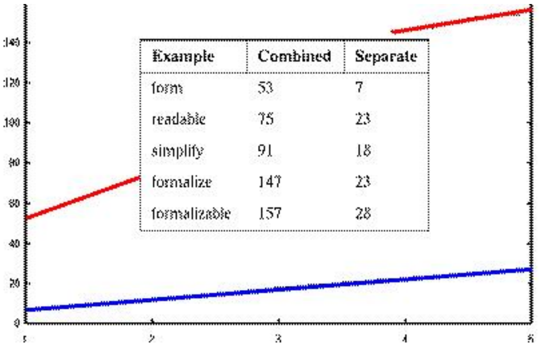
However, there are two kinds of prefixes. Rule (24) only applies to internal prefixation. When the prefix is external, as in the case of *re-* shown earlier in figure 4, the adjunction is external to all X′-Spec/head structure. Hence, a SHIFT must be preferred. We revise (24) as follows:

(25) **Prefix Resolution Rule:** (Revised)
When a prefix is on the top of the stack, select REDUCE (over SHIFT) if the prefix is internal, and the reverse when it is external.

Rule (25), together with (23), eliminate all extraneous computational choice points, and thus allows overt word structure to be analyzed deterministically.

4 Separating Overt/Covert Morphemes

In the previous section, we have restricted our attention to making deterministic the processing of overt suffixes and prefixes. Empty categories constitute a powerful linguistic device that can severely impact computational efficiency. In terms of the LR(1) parsing model, which derives much of its ability to discriminate between competing actions on the basis of the identity of the next input morpheme, the possible interspersion of covert morphemes dilutes much of its discriminatory power. In this section, we show how a modified LR-parser that interleaves the separate processing of covert and overt morphemes can maintain computational efficiency.



Example	Combined	Separate
form	53	7
readable	75	23
simplify	91	18
formalize	147	23
formalizable	157	28

Figure 9: Combined verus separate overt/covert morpheme processing

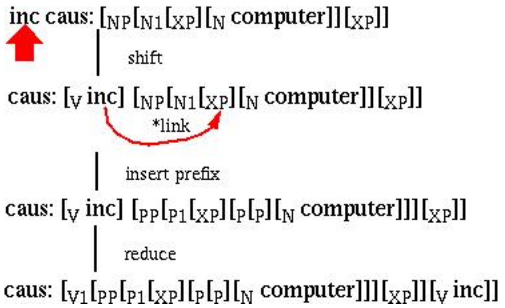

inc caus: $[_{\mathrm{NP}}[_{\mathrm{N1}}[_{\mathrm{XP}}][_{\mathrm{N}}$ computer$]][_{\mathrm{XP}}]]$
shift
caus: $[_{\mathrm{V}}$ inc$]$ $[_{\mathrm{NP}}[_{\mathrm{N1}}[_{\mathrm{XP}}][_{\mathrm{N}}$ computer$]][_{\mathrm{XP}}]]$
*link
insert prefix
caus: $[_{\mathrm{V}}$ inc$]$ $[_{\mathrm{PP}}[_{\mathrm{P1}}[_{\mathrm{XP}}][_{\mathrm{P}}[_{\mathrm{P}}][_{\mathrm{N}}$ computer$]]][_{\mathrm{XP}}]]$
reduce
caus: $[_{\mathrm{V1}}[_{\mathrm{PP}}[_{\mathrm{P1}}[_{\mathrm{XP}}][_{\mathrm{P}}[_{\mathrm{P}}][_{\mathrm{N}}$ computer$]]][_{\mathrm{XP}}]][_{\mathrm{V}}$ inc$]]$

Figure 10: Example of empty prefix insertion

Figure 9 illustrates the large difference in parsing performance between two LR(1) machines: one where all morphemes, overt and covert, are processed together, and a second machine where the LR parser is restricted to dealing only with overt morphemes. This way, no covert suffixes and (internal) prefixes can clutter the landscape seen by the lookahead mechanism, and its predictive power, as seen in figure 9, can thereby be preserved. Covert heads are added inline as required in a manner to be made clear immediately below.

4.1 The Insertion of Empty Prefixes

In the current model, complement domains are always to the left of heads that selects for them. Hence, a complement can be checked for compliance with selectional requirements of a head as soon as the relevant head has been shifted. To take a concrete example, consider the case of *computerize* when `inc` is shifted onto the stack, as shown in figure 10.

The next step is the reduce action that merges $[_V$ inc] with the NP headed by *computer*. However, `inc` imposes requirements (8) on its complement, which must be satisfied before the reduce action can go through. The derivation fails at this point because Linking cannot be satisfied by *computer*, which does not have an A-Compl. As a result the LR machine transfers control to an external routine that performs prefix insertion to repair the parse. This routine adjoins $[_N$ computer] to the prefix P and re-projects local X′-structure. Next, the LR machine re-tries the reduce action, now succeeding because P has provided an A-Compl target.

4.2 The Insertion of Empty Suffixes

Covert suffixes `inc` and `caus` are also inserted by a external procedure. However, one distinction should be clear. For cases like *simplify*, see figure 3, `inc` and `caus` are overt with respect to the LR machine. In other words, initial segmentation produces *simple*+`inc`+`caus`. In the case of zero noun to verb conversion, as in *bottle*, the input is unchanged under initial segmentation, and thus `inc` and `caus` are inserted (and projected) covertly. Assuming the conditions outlined in section 2.1 have been met, the external procedure may stack covert morphemes (modulo language parameterization) before returning control to the LR machine, as is necessary for cases of external prefixation combined with zero `inc` and `caus`, e.g. *re-bottle*.

5 Conclusions

We have described a general-purpose morphological engine for word structure in the Strict Asymmetry framework. We have shown how a generalized LR(1) implementation can be made deterministic through extra-grammatical processes of two kinds: (1) conflict resolution rules that allow the LR engine to deterministically process overt suffixation and prefixation, and (2) external routines that interject covert heads in tandem with overt parsing. The resulting LR machine is a deterministic and efficient engine, building phrases in time linear with respect to the number of overt morphemes.

Acknowledgements

This work is supported in part by funding from the Social Sciences and Humanities Research Council of Canada to the Asymmetry Project, grant number 214-97-0016, and by the NEC Research Institute.

References

S. Anderson. 1992. *A-Morphous Morphology.* Cambridge University Press.

N.A. Chomsky. 1995. *The Minimalist Program.* MIT Press.

N.A. Chomsky. 2001. *Beyond Explanatory Adequacy.* Number 20 in MIT Occasional Papers in Linguistics. MITWPL.

A.-M. Di Sciullo and S. Fong. 2000. Asymmetry, zero morphology and tractability. In *Proceedings of the 15th Pacific Asia Conference on Language, Information and Computation (PACLIC-15)*, pages 61–72, Hong Kong.

A.-M. Di Sciullo. 1994. Prefixes and suffixes. In C. Parodi, C. Quicoli, M. Saltarelli, and M.-L. Zubizarreta, editors, *Romance Linguistic in Los Angeles: Selected Papers from the XXIV Linguistic Symposium on Romance Languages held at USC and UCLA.* Georgetown University Press.

A.-M. Di Sciullo. 1995. X′ selection. In J. Roorick and L. Zaring, editors, *Phrase Structure and the Lexicon*, pages 77–107. Kluwer.

A.-M. Di Sciullo. 1997. Prefixed verbs and adjunct identification. In A.-M. Di Sciullo, editor, *Projections and Interface Conditions: Essays on Modularity*, pages 52–74. Oxford University Press.

A.-M. Di Sciullo. 1999. Local asymmetry. In *Papers from the UPenn/MIT Roundtable on the Lexicon*, volume 35, pages 25–49. MIT Working Papers in Linguistics.

D. E. Knuth. 1965. On the translation of languages from left to right. *Information and Control*, 8(6):607–639.

R. Lieber. 1992. *Deconstructing Morphology.* University of Chicago Press.

M. Tomita. 1986. *Efficient Parsing for Natural Language: A Fast Algorithm for Practical Systems.* Kluwer.