Dependency-based machine translation and parallel parsing without the projectivity and edge-factoring assumptions. A white paper.

Matthias Buch-Kromann
ISV Computational Linguistics Group
Copenhagen Business School
mbk.isv@cbs.dk

Abstract

In this white paper, we review the theoretical evidence about the computational efficiency of dependency parsing and machine translation without the widely used, but linguistically questionable assumptions about projectivity and edge-factoring. On the basis of the heuristic local optimality parser proposed by (Buch-Kromann, 2006), we propose a common architecture for monolingual parsing, parallel parsing, and translation that does not make these assumptions. Finally, we describe the elementary repair operations in the model, and argue that the model is potentially interesting as a model of human translation.

1 Introduction

Dependency grammar has attracted much attention in computational linguistics in recent years, in particular in the fields of parsing, machine translation, and word alignment. However, in most of the dependency-based systems proposed within these fields, computational efficency has been ensured by central assumptions about either projectivity or edge-factoring in the grammar model: the assumption about *projectivity* (continuity, context-freeness) stipulates that crossing dependencies are not allowed; and the assumption about *edge-factoring* states that dependencies are created independently of each other.

In dependency-based parsing, for example, projectivity has played a central role in systems based on chart parsing such as (Eisner, 1996; Collins, 1997) or linear time shift-reduce parsing such as (Nivre and Nilsson, 2005); and edge-factored models have played a pivotal role in systems based on the Minimum Spanning-Tree algorithm such as (McDonald et al., 2005; McDonald and Satta, 2007; Smith and Smith, 2007; Koo et al., 2007). In machine translation and word alignment, context-freeness (which entails projectivity) has played a pivotal role in most dependency-based systems, including (Fox, 2002; Fox, 2005; Quirk et al., 2005; Ding and Palmer, 2005; Ding, 2006; Smith and Eisner, 2006), whereas the edge-factoring assumption has not played any role so far, and may be unlikely to do so because the Minimum Spanning Tree algorithm is difficult to generalize to the parallel word-aligned trees used in machine translation and alignment.

The projectivity and edge-factoring assumptions are known to be problematic from a linguistic point of view, and they are therefore best seen as a way of trading linguistic expressiveness for computational efficiency. The projectivity assumption is violated by many linguistic phenomena, including topicalizations, scramblings, and extrapositions. For example, in the Danish Dependency Treebank (Kromann, 2003), 0.9% of all dependencies are non-projective, and 15% of all sentences contain a non-projective dependency. Since a non-projective word order in the source language often translates into an entirely different word order in the target language, errors with respect to non-projective dependencies can be ex-

pected to result in meaning-disturbing target language word orders, and non-projective dependencies are therefore more important than their relatively small contribution to precision and recall in monolingual parsing suggests.

The edge-factoring assumption has been studied extensively by (McDonald and Satta, 2007), who have shown that it is unlikely that the edge-factoring assumption can be loosened in a non-projective grammar model without making exact parsing computationally intractable. In their paper, they examine the effect of making the grammar model sensitive to a word's arity (the number of dependents of the word), horizontal k-neighbourhood (defined as the k nearest siblings to the left and right), or vertical k-neighbourhood (defined as all transitive parents or children in the dependency tree that are at most k dependencies away). They then prove that the parsing problem becomes NP-hard if a non-projective grammar model is made sensitive to either the arity, or the horizontal or vertical 1-neighbourhood, a result that elegantly generalizes the NP-hardness results for exact non-projective dependency parsing proven by (Neuhaus and Bröker, 1997) and (Buch-Kromann, 2006, p. 15).

The important lesson here is that if unconstrained non-projectivity is coupled with a probabilistic grammar model that is capable of taking even a minimal notion of linguistic context into account, then the corresponding exact parsing problem will be NP-hard. The complexity problems can be expected to become even worse if the grammar models are extended so that they can handle parallel texts and deal with important linguistic phenomena such as secondary dependencies, gapping coordinations, anaphora, and punctuation. For this reason, it is relevant to ask what can be done to escape the difficult trade-off between computational efficiency and linguistic adequacy, and whether we can learn anything from human parsing and translation, where we know the problem has been solved.

In parsing, there are several reasonable approaches that one could take. (McDonald and Satta, 2007) suggest that one could attempt to identify classes of non-projective structures that can be parsed with chart-parsing algorithms, a proposal that may be inspired by the work on pseudo-projective parsing by (Kahane et al., 1998) and the non-projectivity constraints proposed by (Nivre, 2006). (Smith and Smith, 2007) suggest that Minimum Bayes-risk decoding can be used to reduce any non-projective parsing problem to the maximum directed spanning tree problem, even if the original model is not edge-factored. And (Nivre and Nilsson, 2005) has proposed a solution where non-projective dependencies are encoded within a projective dependency tree by means of a gap-threading technique.

In this paper, we will pursue a fourth approach that builds on the psycholinguistic model of human parsing proposed by (Buch-Kromann, 2006) within the dependency framework Discontinuous Grammar. The underlying philosophy is that instead of compromising on the linguistic adequacy in order to make exact parsing computationally efficient, it is preferable to compromise on the exactness, ie, to try to identify heuristic algorithms that often succeed in finding an optimal or near-optimal solution, but are not guaranteed to always do so. This strategy has independent psycholinguistic merit, because psycholinguistic experiments on strong garden-path constructions (Frazier and Rayner, 1982) show that the human parser is sometimes incapable of escaping from a suboptimal analysis, ie, human parsing is heuristic rather than exact.

Compared with the other proposals, our processing model has the advantage that it builds directly on Discontinuous Grammar, a linguistically sophisticated dependency framework which covers secondary dependencies, gapping coordinations, discourse and anaphora, morphology, and punctuation. However, since our model is rather complex and has not been tested empirically on a large scale, it is not clear that it will in the end turn out to be more fruitful than the other approaches. Our goal in this paper is merely to argue that our proposal is a promising line of research within a coherent research programme.

The paper is structured as follows. In section 2, we define monolingual and bilingual dependency analyses in more detail. In section 3, we describe our abstract notion of probabilistic dependency grammars, and how monolingual parsing, bilingual parsing, and translation can be defined in this framework. In section 4, we out-

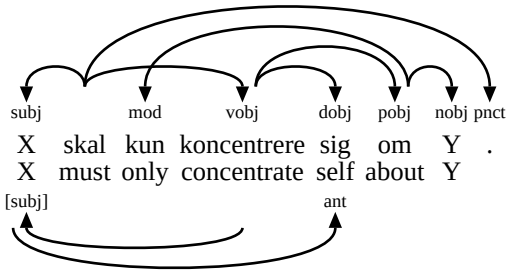

Figure 1: Monolingual dependency analysis with primary dependencies (above the sentence), and secondary dependencies and antecedent edges (below the sentence).

line the psycholinguistically motivated heuristic parsing algorithm proposed by (Buch-Kromann, 2006). In section 5, we discuss how this parsing model can be generalized to translation and parallel parsing. In section 6, we present our conclusions, and in section 7, we present our directions for future research.

2 Monolingual and bilingual dependency analyses[1]

In its simplest form, a monolingual dependency analysis D for a text t consists of dependency edges that link the words in the text so that the dependency edges form a tree. Each dependency edge $g \xrightarrow{r} d$ encodes a complement or adjunct relation between a word g (the *governor*) and a complement or adjunct phrase headed by the word d (the *dependent*), where the edge label r specifies the complement or adjunct role.

As an illustration of dependency trees, the dependency tree for the Danish sentence "X skal kun koncentrere sig om Y" ("X has to concentrate only on Y") is shown in the top part of Figure 1. The dependency arcs are drawn as arrows that go from the governor to the dependent, with the dependency role shown below the head of the arrow. The dependencies may be non-projective, ie, the dependency tree may contain crossing branches, as in the dependency between "only" and "about". As an illustration of dependency roles, Figure 2 shows the most important complement and adjunct roles in Discontinuous Grammar and the associated Danish Dependency Treebank (Kromann, 2003).

In more linguistically sophisticated dependency analyses, the primary tree structure formed by the complement and adjunct edges (the *primary dependency edges*) is supplemented by edges that encode secondary linguistic relations, such as secondary dependencies, antecedents for anaphora and ellipses, and replacement and antecedent relations in gapping coordinations. The bottom part of Figure 1 shows the secondary linguistic relations in the analysis: the "[subj]" edge indicates that in addition to being the subject of "must," the phrase headed by "X" also functions as a secondary subject of "concentrate," which has no subject in the primary dependency structure; and the "ant" edge indicates that the phrase headed by "X" functions as the antecedent for the reflexive pronoun "self." (Buch-Kromann, 2006) provides a detailed account of many other aspects of monolingual dependency analyses, including word order, gapping coordinations, punctuation, secondary dependencies, and an account of discourse structure and anaphora. In this paper, we will focus on probabilistic dependency grammars that are capable of dealing with non-projective dependencies and taking a minimal notion of linguistic context into account, but leave all the other phenomena to a later paper.

Monolingual dependency analyses can be generalized to parallel dependency analyses, as described in (Buch-Kromann, 2007b; Buch-Kromann, 2007a). By a *parallel (bilingual) dependency analysis*, we mean a tuple $A = (D, D', W)$ where D is a dependency analysis of the source text, D' is a dependency analysis of the target text, and W is a set of word alignments. Each word alignment $w \leftrightarrow w'$ in W is assumed to encode a translational correspondence between the word clusters w and w' in the source text and target text, ie, the word alignment encodes the intuition that the subset w of words in the source text corresponds roughly in meaning or function to the subset w' of words in the target text. The translations may contain additions or deletions, ie, w and w' may be empty. Finally, in order to ensure that the source and target structures have a certain degree of syntactic parallelism, the parallel dependency analyses must be well-formed with respect to translation units; the

[1] Sections 2 and 3 are partly based on (Buch-Kromann, 2007a).

Complement roles		Adjunct roles	
aobj	adjectival object	**appa**	parenthetical apposition
avobj	adverbial object	**appr**	restrictive apposition
conj	conjunct of coordinator	**coord**	coordination
dobj	direct object	**list**	unanalyzed sequence
expl	expletive subject	**mod**	modifier
iobj	indirect object	**modo**	dobj-oriented modifier
lobj	locative-directional obj.	**modp**	parenthetical modifier
nobj	nominal object	**modr**	restrictive modifier
numa	additive numeral	**mods**	subject-oriented mod.
numm	multiplicative numeral	**name**	additional proper name
part	verbal particle	**namef**	additional first name
pobj	prepositional object	**namel**	additional last name
possd	possessed in genitives	**pnct**	punctuation modifier
pred	subject/object predicate	**rel**	relative clause
qobj	quotation object	**title**	title of person
subj	subject	**xpl**	explification (colon)
vobj	verbal object		

Figure 2: The main dependency roles in the dependency framework Discontinuous Grammar.

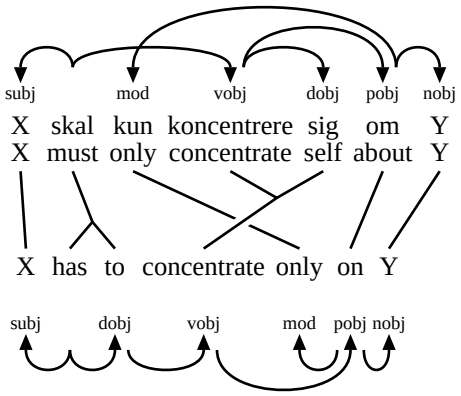

Figure 3: Parallel dependency analysis consisting of a word alignment and two monolingual dependency analyses (secondary linguistic relations not shown).

definition is outside the scope of this paper, and the reader is referred to (Buch-Kromann, 2007b; Buch-Kromann, 2007a) for the details.

Figure 3 is an example of a parallel dependency analysis, based on the annotation conventions used in the Copenhagen Danish-English Dependency Treebank (Buch-Kromann et al., 2007). The source language dependency analysis is shown in the top part of the graph above the source text, the target language dependency analysis is shown in the bottom part below the target text, and word alignments are shown as lines that connect word clusters in the source text with corresponding word clusters in the target text.

3 An abstract notion of parsing, parallel parsing and translation

From an abstract point of view, a *parallel probabilistic dependency grammar* can be viewed as a probability measure $P(A)$ on the space $\mathcal{A}$ of all conceivable parallel dependency analyses; similarly, a *monolingual probabilistic dependency grammar* can be viewed as a probability measure $P_0(A)$ on the space $\mathcal{A}_0$ of all conceivable monolinguial dependency analyses. In this setting, monolingual parsing, bilingual parsing, and translation can be reduced to the problem of optimizing the probability measures $P(A)$ or $P_0(A)$ with different side conditions.

In *monolingual parsing*, we know a text t and need to find the most probable monolingual dependency analysis, $\mathrm{Parse}(t)$, that matches t. That is, we must find:

$$\mathrm{Parse}(t) = \arg\max_{\substack{A\in\mathcal{A}_0 \\ Y(A)=t}} P(A)$$

where $Y(A)$ denotes the text associated with the monolingual analysis A.

Similarly, in *bilingual parsing* (or *parallel parsing*) — which is a crucial step if we want to turn a parallel corpus into a parallel dependency treebank — we know a source text t and a target text t', and need to find the most probable parallel dependency analysis, $\mathrm{Parse}(t, t')$, that matches the given source and target texts t, t'. That is, we

must find:

$$\mathrm{Parse}(t, t') = \arg\max_{\substack{A \in \mathcal{A} \\ Y(A)=t \\ Y'(A)=t'}} P(A)$$

where $Y(A)$ denotes the source text associated with A, and $Y'(A)$ the target text. Finally, in *machine translation*, we know a source text t and need to find the most probable parallel dependency analysis, $\mathrm{Trans}(t)$, that matches t. That is, we must find:

$$\mathrm{Trans}(t) = \arg\max_{\substack{A \in \mathcal{A} \\ Y(A)=t}} P(A).$$

Once we have computed $\mathrm{Trans}(t)$, it is easy to compute the optimal translation by extracting the target text from $\mathrm{Trans}(t)$ by means of Y'.

Several problems must be solved in order to build a functioning parallel parser or machine translation system that uses these ideas to circumvent the linguistic limitations of projectivity and edge-factoring: we must (a) formulate a linguistically sensible notion of parallel dependency analyses and parallel probabilistic dependency grammars; (b) specify a method for inducing such grammars from parallel corpora and/or parallel dependency treebanks; and (c) identify computationally efficient optimization algorithms for translation and parallel parsing that normally succeed in finding optimal or near-optimal translations and parallel parses.

(Buch-Kromann, 2006) and (Buch-Kromann, 2007a) have argued that it is possible to define a linguistically reasonable probabilistic dependency model for monolingual and parallel texts, and that it is possible to construct a probabilistic dependency grammar by training the dependency model on a monolingual or parallel dependency treebank. In the following, we will therefore assume that we already have a probabilistic dependency grammar in the form of a probability measure $P(A)$ (defined on the space $\mathcal{A}$ of all conceivable bilingual dependency analyses), and that all that remains is to find computationally efficient optimization algorithms that often return optimal or near-optimal solutions in the translation task, and in the monolingual and parallel parsing tasks. Ie, in this paper we will focus on (c), and largely ignore (a) and (b).

4 Local optimality parsing as a model of human parsing

In this section, we will sketch the local optimality parsing algorithm proposed within the Discontinuous Grammar framework by (Kromann, 2001; Buch-Kromann, 2006) as a solution to the monolingual parsing problem. In section 5, we will then sketch how this algorithm can be generalized to parallel parsing and translation.

The local optimality parsing algorithm is based on local search, an algorithm that has been applied successfully to other NP-hard computational problems, such as the Travelling Salesman Problem (Papadimitriou and Steiglitz, 1982; Johnson and McGeoch, 1997). Local search has been used as the basis for other parsing algorithms as well, including (Lewis, 1999), (Daum and Menzel, 2002) and the second parsing stage in (McDonald and Pereira, 2006).

The simplest version of the local optimality parsing algorithm (LOP) proposed in (Kromann, 2001) and (Buch-Kromann, 2006, chapters 1 and 7) is shown in Figure 4. The algorithm is based on local search, and builds up analyses incrementally from the empty analysis by reading words one at a time. After a word has been read, the algorithm attempts to improve the probability of the current partial analysis by applying a number of parsing operations that are capable of adding or deleting edges in the current analysis. The analysis that results in the largest improvement in terms of probability (or more generally cost, if the grammar is a cost measure on $\mathcal{A}$) is chosen as the new current analysis, and the parsing operations are applied again until no more improvements are possible. The subroutine improve(A) searches for an improvement A' that can be obtained by applying $\pi_1, \ldots, \pi_m$ to A, where the parsing operation π_i maps an analysis $A \in \mathcal{A}$ into a subset $\pi_i(A)$ of $\mathcal{A}$, which is called the π_i-*neighbourhood* of A. The algorithm can be characterized as a serial parsing algorithm with repair that processes the input incrementally in a left-right fashion, just like humans. (Buch-Kromann, 2006, chapter 7.4) has shown that the parsing algorithm is compatible with the findings from a wide range of psycholinguistic experiments, if it is restricted by a certain set of processing constraints that fall out-

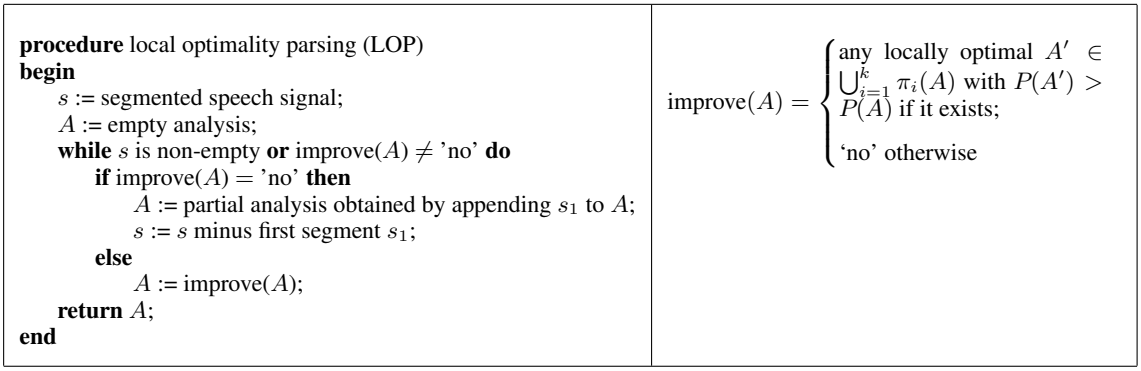

Figure 4: The algorithm for local optimality parsing (LOP) and the definition of improve(A), from (Buch-Kromann, 2006, p. 28).

side the scope of this paper.

Local optimality parsing is a general algorithm which leads to different parsers depending on the particular choice of parsing operations. The simplest choice is to use k-change operations, which are allowed to change up to k arbitrary dependencies in one parsing operation.[2] k-change operations are very powerful, since every parsing operation can be described as a k-change operation for sufficiently large k. However, k-change operations with $k > 1$ are also quite inefficient because they do not exploit the fact that the grammar tends to be local, ie, if there is an error in the analysis at a particular node, the error is unlikely to be resolved by changing the structure at a node that is very far away from the error node in the graph structure. The size of a k-change neighbourhood is roughly n^k, where n is the length of the input, and it is therefore no surprise that k-change operations are quite inefficient even for small k.

The key to efficient local optimality parsing is therefore to design parsing operations that can exploit the locality of the grammar. (Buch-Kromann, 2006, chapter 7.3) has proposed an alternative family of parsing operations, called k-*error operations*, which are sequences of elementary parsing operations as in k-change, but with the restriction that each new elementary parsing operation must correct an error that was introduced by the preceding elementary parsing operation (the elementary parsing operations do not necessarily improve the analysis, they may temporarily introduce new errors and make the analysis globally worse, as long as the final analysis is better than the original analysis). Buch-Kromann also proposes a detailed inventory of elementary parsing operations for adding, deleting or changing a node's lexeme, landing site, complement governor, or adjunct governor, as well as a number of specialized operations that deal with secondary dependencies and gapping coordinations. With this inventory, k-error parsing has a worst-case time complexity of $O(n \log^{3k+1} n)$ under linguistically plausible assumptions about balancedness and island constraints.[3]

In Discontinuous Grammar, errors are defined in terms of the probabilities of the individual generation decisions in the generative language model (Buch-Kromann, 2006, chapter 6.2), which assign a local probability to each decision at each individual word: if the local probability falls below a certain threshold that may depend on the kind of generative step taken (eg, the 1% fractile for complement frame decisions), then the local generation decision is defined to be an error. In this way, a generative probabilistic language model gives rise to localized probabilities at individual words, which can be used to pinpoint the type and precise location of errors in the graph.[4]

[2]The second parsing stage in (McDonald and Pereira, 2006) is an instance of local optimality parsing with 1-change, except that the local search starts with the best projective parse of the sentence rather than the empty analysis; Pereira and McDonald state that their 1-change algorithm is frequently trapped in local optima that are not global optima, ie, 1-change operations are helpful, but not powerful enough.

[3]Without any assumptions, the worst-case time complexity becomes $O(n^{4k+1} \log n)$, ie, in a local optimality parser, island constraints are a necessary precondition for almost-linear parsing.

[4]Because local optimality parsing is incremental, it is important that the grammar assigns a non-zero probability to

Name	Violation	Cost
no gov.	no governor	\$5
no lsite	no landing site	\$5
bad case	bad case agreement	\$5
bad cframe	bad complement frame	\$5
inverted	inverted sentence	\$0.5

Figure 5: A simple toy grammar. Cost $\geq$ \$1 indicates error, cost $<$ \$1 indicates preference.

To demonstrate how a local optimality parser with k-error parsing works, we will for simplicity assume the toy grammar shown in Figure 5, rather than a probabilistic grammar trained on a large dependency treebank. The grammar assigns costs to different linguistic violations at particular nodes. Eg, it assigns a cost of \$5 to a word which lacks a governor or a landing site, or where there is a problem with case agreement or with the compatibility between the complement structure and the chosen complement frame. It also assigns a cost of \$0.5 to inverted sentences, thereby indicating a preference for non-inverted sentences. Costs at or above \$1 indicate errors, costs below \$1 are non-errors that indicate preferences.

Figure 6 shows the individual parsing steps in a local optimality parse of the topicalized Danish sentence "John tror jeg vi møder" ("I believe we will meet John"). In step 1, the parser reads the first word, which is marked as an error node (shown with a red oval) because the word lacks a governor and a landing site, resulting in a total cost of \$10. In step 2, the parser reads the second word, and marks it as an error node in the same way. In step 3, the parser manages to analyze "John" as the subject of "believe," thereby reducing the cost with \$10. In step 4, the parser reads the next word, which is again an error node. In step 5a, the nominative marking of "I" means that the parser chooses to analyze "I" as a subject of "believe" (rather than a direct object), which triggers temporary complement frame errors at the two subjects "John" and "I," turning "John" into an error node; and in step 5b, the parser resolves the error at "John" by reanalyzing "John" as the direct object of "believe." In step 6, the parser reads the next word "we," which is analyzed as a temporary landed node of "believe" in step 7. In step 8, the parser reads the next word "meet." In step 9a, "meet" is analyzed as the verbal object of "believe," which triggers complement frame errors at "John" and "meet," but the error is resolved in step 9b by reanalyzing "John" as the direct object of "meet." Finally, in step 10, "we" is reanalyzed as the subject of "meet."

Preliminary studies of local optimality parsing by (Kromann, 2001) and (Buch-Kromann, 2006, chapter 7.5) have shown that local optimality parsing seems capable of parsing a wide range of non-projective constructions in German, but fails in at least some garden-path constructions. However, the lack of a large-scale computational implementation so far means that local optimality parsing has not been tested on a large scale yet. But while it is too early to draw any final conclusions, the preliminary evidence seems to suggest that local optimality parsing is an interesting parsing model with respect to both computational efficiency, psycholinguistic plausibility, and linguistic adequacy.

5 Extending the LOP algorithm to translation and parallel parsing

In this section, we will tentatively sketch how we can extend the local optimality parsing algorithm to the translation and parallel parsing tasks. We will assume that we already have a bilingual dependency grammar that assigns local probabilities to every partial bilingual dependency analysis, so that all we need to do is to (a) generalize the local optimality algorithm to the parallel case, and (b) specify a set of elementary operations for manipulating translation units and word alignments in addition to the existing elementary operations for monolingual parsing. Note that k-error operations do not require any further generalization because they are specified by means of elementary operations and grammar errors, which also make sense in the bilingual case.

partial dependency analyses as well as dependency analyses of complete texts and sentences. (Buch-Kromann, 2006) introduces a wide range of mechanisms for dealing with partial analyses, including temporary landing sites (so that dependents can get a landing site before their governor has appeared in the input) and time-dependent dynamic cost functions (so that errors are temporarily or permanently ignored if there is reason to believe that missing structure is going to show up in the input soon, or a repair attempt has been made either recently or many times without success). However, the details are outside the scope of this paper.

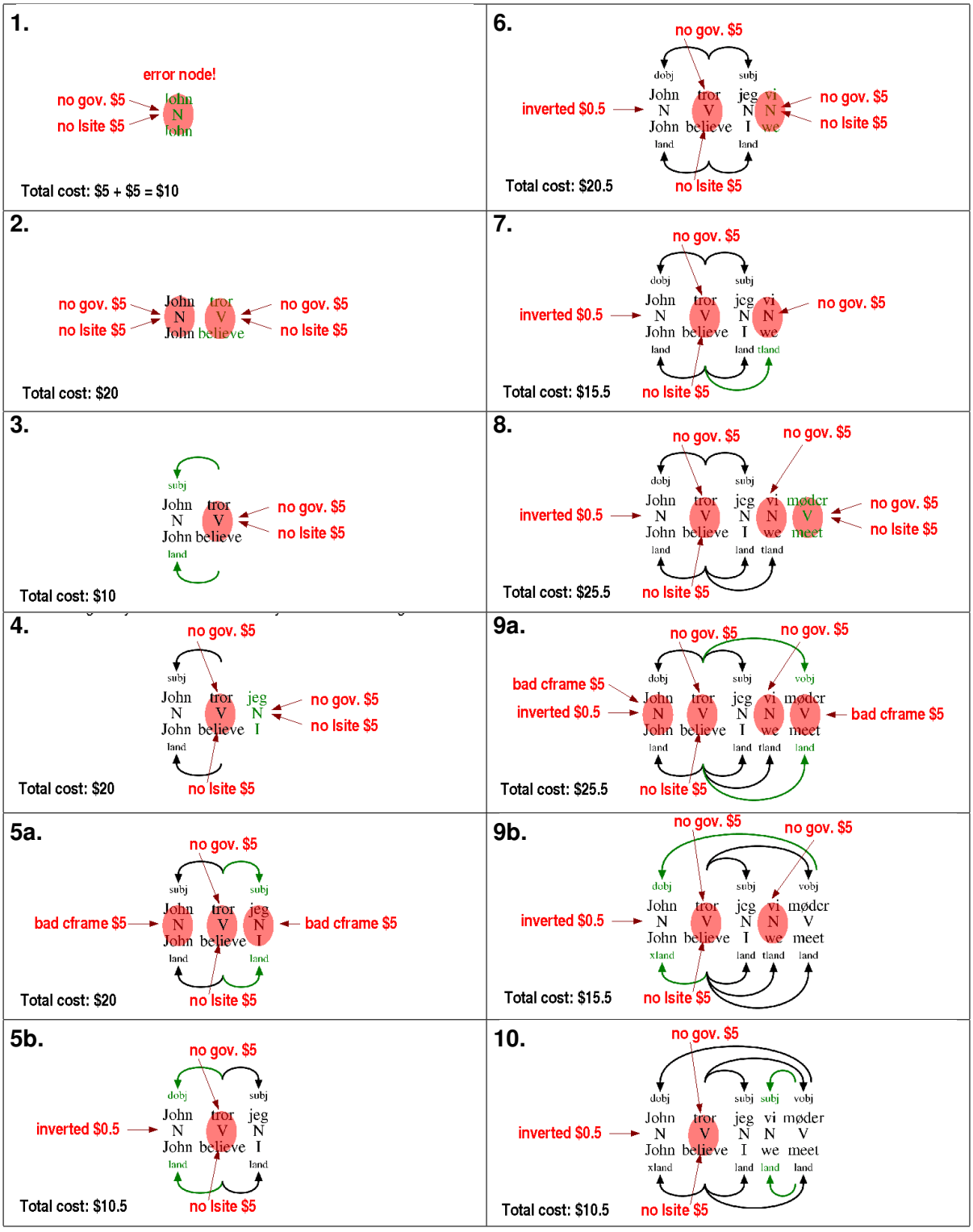

Figure 6: A local optimality parse of the topicalized Danish sentence "John tror jeg vi møder" (I believe we will meet John") with the toy grammar in Figure 5.

```
procedure multi-channel local optimality processing (MLOP)
begin
    s := segmented speech signal with channels s_1, ..., s_n;
    A := empty analysis;
    while s is non-empty or improve(A) ≠ 'no' do
        if improve(A) = 'no' then
            for i = 1 to n do
                A_i := partial analysis obtained by appending first segment of s_i to A;
                A_i^* := the best analysis within {A_i, improve(A_i)};
            i^* := any i that optimizes the probability of A_i^*;
            A := A_{i^*};
            s := s without first segment of s_{i^*};
        else
            A := improve(A);
    return A;
end
```

Figure 7: The algorithm for multi-channel local optimality processing (MLOP).

The local optimality algorithm has one input channel (the source text) in the translation task, but two input channels (the source and target texts) in the parallel parsing task. (Buch-Kromann, 2006, p. 299) has proposed a generalization of the LOP algorithm in Figure 4 to multi-channel parsing in the case where the channels share a common timeline (as in multi-speaker dialogue); however, this algorithm does not address the parallel parsing task where there is no common timeline between the channels. We therefore propose to generalize the LOP algorithm by means of the multi-channel optimality processing algorithm (MLOP) shown in Figure 7. The MLOP algorithm solves the multi-channel problem by examining the set of all analyses that can be produced by reading a word from one of the channels and applying the improve-operation; it then chooses the best resulting analysis as the new analysis.[5] [6]

When designing the elementary operations needed for translation and parallel parsing, we must create a set of elementary operations that allows every parallel analysis to be created by means of a sequence of elementary operations and lexical access operations that add a single word from one of the input channels. The identification of these elementary operations is still work in progress, but in the following, we will describe the most important elementary operations required for this purpose.

In parallel parsing, the *ualign-add* operation identifies translation units for parallel phrases.

[5] From an engineering point of view, it is not necessarily desirable to let the MLOP algorithm start with the empty analysis. For example, inspired by the approximate dependency parsing algorithm by (McDonald and Pereira, 2006), one could initialize a MLOP parallel parser with a parallel analysis composed of the best Giza++ word alignment and the best MST parses of the source and target text, instead of the empty graph. The MLOP parallel parser would then apply improve(A) until no further improvements were possible. Similarly, one could initialize an MLOP translator with a partial analysis consisting of the best MST parse of the source text. The MLOP algorithm would then apply improve(A) until it had created a locally optimal translation. While it is not really clear that parallel parsing has an equivalent in human language processing, translation does have a human equivalent; and here the use of a non-incremental algorithm would invalidate the algorithm's psycholinguistic plausibility. It is unclear which approach would work best in terms of avoiding local minima.

[6] From an engineering point of view, it is also possible to create a simple translation algorithm by using the second stage in the generative procedure for parallel texts described in (Buch-Kromann, 2007a) as a translation algorithm, and make the generative decisions on the basis of greedy search or beam search. However, this algorithm is outside the scope of this paper.

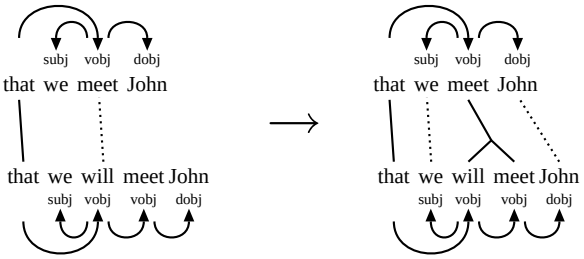

Figure 8: ualign-operation in parallel parsing.

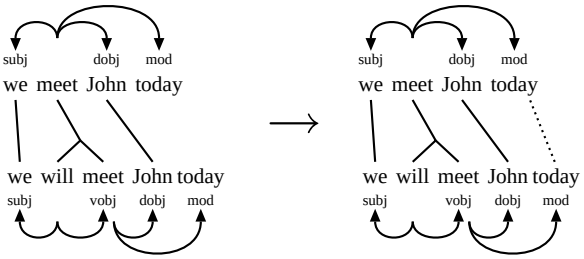

Figure 9: malign-operation in parallel parsing.

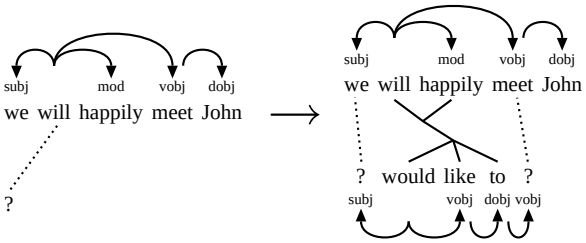

Figure 10: utrans-operation in translation.

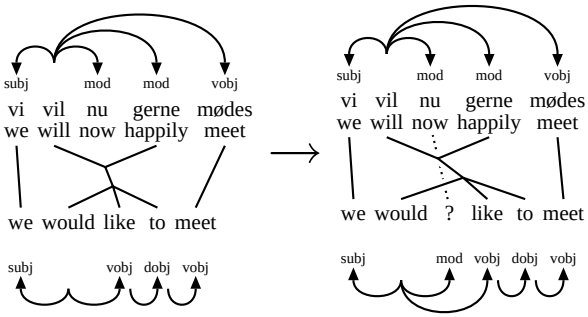

Figure 11: mtrans-operation in translation.

It takes two nodes n, n' that have been phrase-aligned, and aligns the underlying structures by selecting a translation unit that is compatible with the source and target dependency subtrees at n, n'. The resulting parallel arguments in the translation unit are then marked as phrase-aligned. The reverse operation is called *ualign-del*. Figure 8 shows a typical input (left) and output (right) from the ualign-add operation. The top sentence in each graph is an English gloss of a Danish sentence, and the bottom sentence is the English translation. In the example, "meet" and "will" have been phrase-aligned (shown with dotted lines), and the ualign-operation has identified the two phrases by means of the translation unit "meet ↔ will meet."

The *malign-add* operation identifies parallel adjuncts. It takes an adjunct that modifies a translation unit, and attempts to find a corresponding parallel adjunct in the other language that modifies the same translation unit. The two adjuncts are then marked as phrase-aligned in the resulting analysis. Figure 9 shows a typical malign-add operation. In the example, the malign-operation identifies the instances of "today" in the top and bottom graph as parallel adjuncts of the translation unit "meet ↔ will meet," and aligns them by means of a phrase alignment.

In translation, the *utrans*-operation selects a translation unit at a phrase-aligned node in the source dependency tree and creates the corresponding subtree in the target analysis; the parallel arguments of the translation unit are encoded by means of phrase alignments (shown with dotted lines). Figure 10 shows a typical utrans-operation; here, the utrans-operation selects the translation unit "will happily ↔ would like to" for the word "will" in the source dependency tree, and creates the corresponding target structure.

The *mtrans*-operation projects a parallel adjunct from the source analysis to the target analysis, by identifying an adjunct role and an adjunct governor within the target translation unit. Figure 11 shows a typical mtrans-operation within the translation unit "vil gerne ↔ would like to"; here, the mtrans-operation transfers the source mod-adjunct "nu" of "vil" into a parallel mod-adjunct of "would."

The *itrans*-operation creates an inserted target unit in the target analysis, ie, a target unit that is not aligned to anything within the source analysis. Figure 12 shows a typical itrans-operation; here, the itrans-operation creates a comma (the target unit) as a punctuation adjunct at the word "today" in the target analysis.

The *lsite*-operation is an extension of the *lsite*-operation in monolingual parsing which identifies a word's landing site and word order rela-

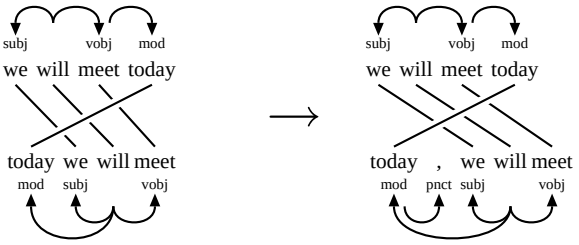

Figure 12: itrans-operation in translation.

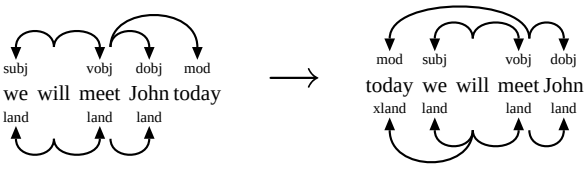

Figure 13: lsite-operation in translation.

tive to the other landed nodes at the landing site.[7] The operation identifies a landing site for a word that lacks a landing site, and specifies the word's position in the word order relative to the other landed nodes at that landing site, thereby defining the word's global word order position. Figure 13 shows the surface tree and deep tree for the target dependency analysis in a parallel analysis (the source analysis is not shown). The word "today," which initially lacks a landing site, is identified as a landed node of "will" (ie, it is extracted from its governor "meet' to "will"), and it is placed before the landed subject of "will," resulting in the inverted word order shown on the right.

The complete preliminary inventory of elementary operations for translation and parallel parsing, including the inverse deletion operations, is shown in Figure 14. Although we have not calculated the theoretical time complexity of the elementary operations, we suspect that the time complexity will end up being of the form $\log^k n$ under linguistically reasonable assumptions about balancedness and island constraints (important for lsite). If this is true, then the almost-linear time complexity will be preserved when we generalize local optimality parsing with k-error to translation and parallel parsing. The crucial question therefore is whether the elementary operations are powerful enough to allow the parsing algorithm to avoid getting trapped in local optima that are not globally near-optimal — an empirical question that remains to be answered. Our description of the local optimality algorithm for translation and parallel parsing, and the inventory of elementary operations, should therefore be seen as a preliminary blueprint for the system that we intend to implement.

Most of the algorithms used in machine translation and parsing today, are either globally parallel (eg, chart parsing, beam search, etc.) or non-destructive (eg, Nivre's deterministic dependency parser), with a few exceptions such as (Germann et al., 2001). In contrast, the parsing and translation model presented in this paper is based on a serial repair algorithm that is capable of changing analyses destructively, like the parsing model proposed by (Dyke and Lewis, 2003; Lewis and Vasishth, 2005), which is arguably the most detailed and sophisticated model of human language processing in current psycholinguistic research. Moreover, it is known from empirical studies of human translation processes (Alves, 2003; Jakobsen, 2006; Buchweitz and Alves, 2006) that expert human translators make frequent revisions of their translations, ie, that repair processes are an integral aspect of human translation. This suggests that our generalized local optimality algorithm coupled with our elementary operations and k-error may be potentially interesting as a model of human translation.

6 Conclusions

In this paper, we have argued that our proposed model of monolingual parsing, bilingual parsing, and translation is based on a coherent research programme with promising properties in terms of linguistic and psycholinguistic adequacy, and computational efficiency, although the model still remains to be evaluated on a large scale. We have argued that parsing and translation can be viewed as optimization problems, and that the NP-hardness results for these optimization problems makes it reasonable to look for heuristic algorithms that compromise on exactness in order to gain a better balance between linguistic adequacy and computational efficiency. We

[7] The theory of landing sites and word order in Discontinuous Grammar is presented in detail in (Buch-Kromann, 2006, chapters 2.4, 4.2). The relative word order of the landed nodes at a landing site is defined by the input in monolingual and bilingual parsing, and in the source analysis in translation, so reordering of landed nodes can only take place in the target analysis within the translation task.

	delete	add	change
parallel parsing			
tunit alignment	ualign-add	ualign-del	*
modifier alignment	malign-add	malign-del	*
translation			
tunit transfer	utrans-add	utrans-del	*
modifier transfer	mtrans-add	mtrans-del	*
insertion	itrans-add	itrans-del	*
word order	lsite-add	lsite-del	*

Figure 14: Preliminary inventory of elementary operations needed for translation and parallel parsing.

have outlined how the Discontinuous Grammar framework account for monolingual and bilingual grammars, and how it attempts to model human parsing. Finally, we have sketched how the local optimality parsing algorithm proposed within Discontinuous Grammar can be generalized to translation and parallel parsing, and described the elementary operations that may be needed within this framework.

7 Future research directions

In future research, we will attempt to create a large-scale implementation of the proposed monolingual parser, bilingual parser, and translation system that can be used to assess the linguistic adequacy and computational efficiency of the system, and to compare the system with alternative translation models. We will also attempt to examine the psycholinguistic validity of the model by holding it up against the growing body of evidence about human translation processes. There are many unresolved theoretical questions that need to be addressed, such as how to define a probabilistic dependency model for partial analyses with dynamic costs, cf. (Buch-Kromann, 2006, 98–103), and how to extend and refine the probabilistic dependency models and the elementary operations so that they can deal with linguistic phenomena such as secondary dependencies, gapping coordinations, discourse relations, anaphora and ellipses, and punctuation.

8 Acknowledgements

The work presented in this paper is supported by a grant from the Danish Research Council for the Humanities.

References

Fábio Alves, editor. 2003. *Triangulating Translation: Perspectives in process oriented research*. Number 45 in Benjamins Translation Library. John Benjamins Publishing Company.

Matthias Buch-Kromann, Jürgen Wedekind, and Jakob Elming. 2007. The Copenhagen Danish-English Dependency Treebank. http://www.id.cbs.dk/∼mbk/ddt-en.

Matthias Buch-Kromann. 2006. Discontinuous Grammar. A dependency-based model of human parsing and language learning. Dr.ling.merc. dissertation, Copenhagen Business School. http://www.id.cbs.dk/∼mbk/thesis.

Matthias Buch-Kromann. 2007a. Breaking the barrier of context-freeness. Towards a linguistically adequate probabilistic dependency model of parallel texts.

Matthias Buch-Kromann. 2007b. Computing translation units and quantifying parallelism in parallel dependency treebanks. In *Proc. of Linguistic Annotation Workshop, ACL-2007 (see errata on http://www.isv.cbs.dk/∼mbk/pub/2007-law.html)*.

Augusto Buchweitz and Fábio Alves. 2006. Cognitive adaptation in translation: an interface between language direction, time, and recursiveness in target text production. *Letras de Hoje*, 41(2).

Michael Collins. 1997. Three generative, lexicalized models for statistical parsing. In Philip R. Cohen and Wolfgang Wahlster, editors, *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 16–23, Somerset, New Jersey. Association for Computational Linguistics.

Michael Daum and Wolfgang Menzel. 2002. Parsing natural language using guided local search. In

F. van Harmelen, editor, *Proceedings 15th European Conference on Artificial Intelligence, Lyon, France, 2002*, pages 435–439.

Yuan Ding and Martha Palmer. 2005. Machine translation using Probabilistic Synchronous Dependency Insertion Grammars. In *Proc. ACL-2005*.

Yuan Ding. 2006. *Machine translation using Probabilistic Synchronous Dependency Insertion Grammars*. Ph.D. thesis, Univ. of Pennsylvania.

J.A. Van Dyke and R.L. Lewis. 2003. Distinguishing effects of structure and decay on attachment and repair: A cue-based parsing account of recovery from misanalyzed ambiguities. *Journal of Memory and Language*, 49:285–316.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING-96*, pages 340–345.

Heidi J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proc. of EMNLP 2002*, pages 304–311.

Heidi J. Fox. 2005. Dependency-based statistical machine translation. In *Proc. of 2005 ACL Student Workshop*.

Lyn Frazier and Keith Rayner. 1982. Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology*, 14:178–210.

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast decoding and optimal decoding for machine translation. In *Proceedings of ACL 2001*.

Arnt Lykke Jakobsen. 2006. Research methods in translation — Translog. In Kirk P.H. Sullivan and Eva Lindgren, editors, *Computer Keystroke Logging and Writing*, pages 95–105. Elsevier.

D. S. Johnson and L. A. McGeoch. 1997. The Traveling Salesman Problem: A case study in local optimization. In E. H. L. Aarts and J. K. Lenstra, editors, *Local search in combinatorial optimization*, pages 215–310. John-Wiley and Sons, Ltd.

Sylvain Kahane, Alexis Nasr, and Owen Rambow. 1998. Pseudo-projectivity: a polynomially parsable non-projective dependency grammar. In *COLING-ACL'98, Montreal*, pages 646–52.

T. Koo, A. Globerson, X. Carreras, and M. Collins. 2007. Structured prediction models via the Matrix-Tree Theorem. In *Proc. of EMNLP-2007*.

Matthias T. Kromann. 2001. Optimality parsing and local cost functions in Discontinuous Grammar. In *Proceedings of the Joint Conference on Formal Grammar and Mathematics of Language (FGMOL-01), Helsinki, August 10-12, 2001. Electronic Notes in Theoretical Computer Science 53*.

Matthias T. Kromann. 2003. The Danish Dependency Treebank and the DTAG treebank tool. In Joakim Nivre and Erhard Hinrichs, editors, *Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT 2003)*. Växjö University Press.

R.L. Lewis and S. Vasishth. 2005. An activation-based model of sentence processing as skilled memory retrieval. *Cognitive Science*, 29:375–419.

R. L. Lewis. 1999. Specifying architectures for language processing: Process, control, and memory in parsing and interpretation. In M. Crocker, M. Pickering, and C. Clifton, editors, *Architectures and mechanisms for language processing*. Cambridge University Press.

R. McDonald and F. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proc. of EACL-2006*.

R. McDonald and G. Satta. 2007. On the complexity of non-projective data-driven dependency parsing. In *Proceedings of International Conference on Parsing Technologies, 2007*.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP 2005*.

Peter Neuhaus and Norbert Bröker. 1997. The complexity of recognition of linguistically adequate dependency grammars. In Philip R. Cohen and Wolfgang Wahlster, editors, *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 337–343, Somerset, New Jersey. Association for Computational Linguistics.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective dependency parsing. In *Proceedings of ACL 2005*.

Joakim Nivre. 2006. Constraints on non-projective dependency parsing. In *Proc. EACL*, pages 73–80.

Christos H. Papadimitriou and Kenneth Steiglitz. 1982. *Combinatorial optimization. Algorithms and complexity*. Dover Press.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proc. ACL-2005*.

David A. Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proc. of HLT-NAACL Workshop on Statistical Machine Translation*, pages 23–30.

David A. Smith and Noah A. Smith. 2007. Probabilistic models of nonprojective dependency trees. In *Proc. of EMNLP-2007*.