Linguistically Informed Statistical Models of Constituent Structure for Ordering in Sentence Realization

Eric RINGGER[1], Michael GAMON[1], Robert C. MOORE[1],
David ROJAS[2], Martine SMETS[1], Simon CORSTON-OLIVER[1]

[1]Microsoft Research
One Microsoft Way
Redmond, Washington 98052, USA
{ringger, mgamon, bobmoore, msmets, simonco}@microsoft.com

[2]Butler Hill Group, LLC
& Indiana University Linguistics Dept.
1021 East 3rd Street, MM 322
Bloomington, Indiana 47405, USA
drojas@indiana.edu

Abstract

We present several statistical models of syntactic constituent order for sentence realization. We compare several models, including simple joint models inspired by existing statistical parsing models, and several novel conditional models. The conditional models leverage a large set of linguistic features without manual feature selection. We apply and evaluate the models in sentence realization for French and German and find that a particular conditional model outperforms all others. We employ a version of that model in an evaluation on unordered trees from the Penn TreeBank. We offer this result on standard data as a reference-point for evaluations of ordering in sentence realization.

1 Introduction

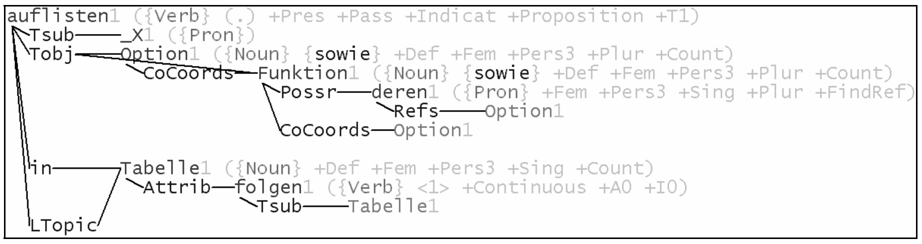
Word and constituent order play a crucial role in establishing the fluency and intelligibility of a sentence. In some systems, establishing order during the sentence realization stage of natural language generation has been accomplished by hand-crafted generation grammars in the past (see for example, Aikawa et al. (2001) and Reiter and Dale (2000)). In contrast, the Nitrogen (Langkilde and Knight, 1998a, 1998b) system employs a word n-gram language model to choose among a large set of word sequence candidates which vary in constituent order, word order, lexical choice, and morphological inflection. Nitrogen's model does not take into consideration any non-surface linguistic features available during realization.

The Fergus system (Bangalore and Rambow, 2000) employs a statistical tree model to select probable trees and a word n-gram model to rank the string candidates generated from the best trees. Like Nitrogen, the HALogen system (Langkilde, 2000; Langkilde-Geary, 2002a, 2002b) uses word n-grams, but it extracts the best-scoring surface realizations efficiently from a packed forest by constraining the search first within the scope of each constituent.

Our research is carried out within the Amalgam broad coverage sentence realization system. Amalgam generates sentence strings from abstract predicate-argument structures (Figure 1), using a pipeline of stages, many of which employ machine-learned models to predict where to perform specific linguistic operations based on the linguistic context (Corston-Oliver et al., 2002; Gamon et al., 2002a, 2002b; Smets et al., 2003). Amalgam has an explicit ordering stage that determines the order of constituents and their daughters. The input for this stage is an unordered tree of constituents; the output is an ordered tree of constituents or a ranked list of such trees. For ordering, Amalgam leverages tree constituent structure and, importantly, features of those constituents and the surrounding context. By separately establishing order within constituents, Amalgam heavily constrains the possible alternatives in later stages of the realization process. The design allows for interaction between ordering choices and other realization decisions, such as lexical choice (not considered in the present work), through score combinations from distinct Amalgam pipeline stages.

Most previous research into the problem of establishing order for sentence realization has focused on English, a language with fairly strict word and constituent order. In the experiments described here we first focus on German and French which present novel challenges.[1] We also describe an English experiment involving data from the Penn Treebank. Our ultimate goal is to develop a model that handles all ordering phenomena in a unified and elegant way across typologically diverse languages. In the present paper, we explore the space of possible models and examine some of these closely.

[1] For an overview of some of the issues in determining word and constituent order in German and French, see (Ringger et al., 2003).



```
auflisten1 ({Verb} (.) +Pres +Pass +Indicat +Proposition +T1)
 Tsub────_X1 ({Pron})
 Tobj────Option1 ({Noun} {sowie} +Def +Fem +Pers3 +Plur +Count)
          CoCoords──Funktion1 ({Noun} {sowie} +Def +Fem +Pers3 +Plur +Count)
                     Possr───deren1 ({Pron} +Fem +Pers3 +Sing +Plur +FindRef)
                              Refs────Option1
                     CoCoords──Option1

 in──────Tabelle1 ({Noun} +Def +Fem +Pers3 +Sing +Count)
          Attrib──folgen1 ({Verb} <1> +Continuous +A0 +I0)
                   Tsub────Tabelle1
 LTopic
```

Figure 1: Abstract predicate-argument structure (NLPWin logical form) for the German sentence:
In der folgenden Tabelle werden die Optionen sowie deren Funktionen aufgelistet.
(*The options and their functions are listed in the following table.*)

2 Models of Constituent Order

In order to develop a model of constituent structure that captures important order phenomena, we will consider the space of possible joint and conditional models in increasing complexity. For each of the models, we will survey the independence assumptions and the feature set used in the models.

Our models differ from the previous statistical approaches in the range of input features. Like the knowledge-engineered approaches, the models presented here incorporate lexical features, parts-of-speech, constituent-types, constituent boundaries, long-distance dependencies, and semantic relations between heads and their modifiers.

Our experiments do not cover the entire space of possible models, but we have chosen significant points in the space for evaluation and comparison.

2.1 Joint Models

We begin by considering joint models of constituent structure of the form $P(\pi,\rho)$ over ordered syntax trees π and unordered syntax trees ρ. An ordered tree π contains non-terminal constituents *C*, each of which is the parent of an ordered sequence of daughters ($D_1,...,D_n$), one of which is the head constituent *H*.[2] Given an ordered tree π, the value of the function $unordered\_tree(\pi)$ is an unordered tree ρ corresponding to π that contains a constituent *B* for each *C* in π, such that

$$daughters(B) = unordered\_set(daughters(C))$$
$$= \{D_1,...,D_n\}$$

again with $H = D_i$ for some *i* in $(1..n)$. The hierarchical structure of ρ is identical to that of π.

We employ joint models for scoring alternative ordered trees as follows: given an unordered syntax tree ρ, we want the ordered syntax tree $\hat{\pi}$ that maximizes the joint probability:

$$\hat{\pi} = \arg\max_{\pi} P(\pi,\rho) = \arg\max_{\pi:\rho=unordered\_tree(\pi)} P(\pi) \qquad (1)$$

As equation (1) indicates, we can limit our search to those trees π which are alternative orderings of the given tree ρ.

Inter-dependencies among ordering decisions within different constituents (e.g., for achieving parallelism) make the global sentence ordering problem challenging and are certainly worth investigating in future work. For the present, we constrain the possible model types considered here by assuming that the ordering of any constituent is independent of the ordering within other constituents in the tree, including its daughters; consequently,

$$P(\pi) = \prod_{C \in constits(\pi)} P(C)$$

Given this independence assumption, the only possible ordered trees are trees built with non-terminal constituents computed as follows: for each $B \in constits(\rho)$,

$$C^* = \arg\max_{C:B=unordered\_set(C)} P(C)$$

In fact, we can further constrain our search for the best ordering of each unordered constituent *B*, since *C*'s head must match *B*'s head:

$$C^* = \arg\max_{\substack{C:B=unordered\_set(C)\\ \&head(B)=head(C)}} P(C)$$

Thus, we have reduced the problem to finding the best ordering of each constituent of the unordered tree.

Now if we wish to condition on some feature $x = f(\rho)$, then we must first predict it as follows:

$$C^* = \arg\max_{\substack{C:B=unordered\_set(C)\\ \&head(B)=head(C)}} P(x)P(C|x)$$

If *x* is truly a feature of ρ and does not depend on any particular ordering of any constituent in ρ, then $P(x)$ is constant, and we do not need to compute it in practice. In other words,

$$C^* = \arg\max_{\substack{C:B=unordered\_set(C)\\ \&head(B)=head(C)}} P(C|x) \qquad (2)$$

Hence, even for a joint model $P(C)$, we can condition on features that are fixed in the given unordered tree ρ without first predicting them. The joint models described here are of this form.

[2] All capital Latin letters denote constituents, and corresponding lower-case Latin letters denote their labels (syntactic categories).

For this reason, when we describe a distribution $P(C|x)$, unless we explicitly state otherwise, we are actually describing the part of the joint model that is of interest. As justified above, we do not need to compute $P(x)$ and will simply present alternative forms of $P(C|x)$.

We can factor a distribution $P(C|x)$ in many different ways using the chain rule. As our starting point we adopt the class of models called Markov grammars.[3] We first consider a left-to-right Markov grammar of order *j* that expands *C* by predicting its daughters $D_1,...,D_n$ from left-to-right, one at a time, as shown in Figure 2: in the figure. D_i depends only on (D_{i-j}, …, D_{i-1}), and the parent category C., according to the distribution in equation (3).

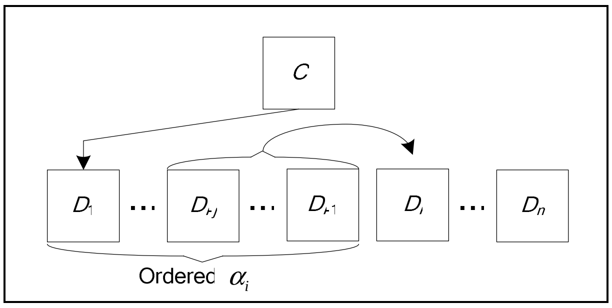


Figure 2: Left-to-right Markov grammar.

$$P(C|h)=\prod_{i=1}^{n} P\left(d_i \middle| d_{i-1},...,d_{i-j},c,h\right) \tag{3}$$

In order to condition on another feature of each ordered daughter D_i, such as its semantic relation ψ_i to the head constituent *H*, we also first predict it, according to the chain rule. The result is the *semantic Markov grammar* in equation (4):

$$P(C|h)=\prod_{i=1}^{n}\begin{bmatrix} P\left(\psi_i \middle| d_{i-1},\psi_{i-1},...,d_{i-j},\psi_{i-j},c,h\right) \\ \times P\left(d_i \middle| \psi_i,d_{i-1},\psi_{i-1},...,d_{i-j},\psi_{i-j},c,h\right)\end{bmatrix} \tag{4}$$

Thus, the model predicts semantic relation ψ_i and then the label d_i in the context of that semantic relation. We will refer to this model as Type 1 (T1).

As an extension to model Type 1, we include features computed by the following functions on the set α_i of daughters of *C* already ordered (see **Figure 2**):

- Number of daughters already ordered (size of α_i)
- Number of daughters in α_i having a particular label for each of the possible constituent labels {NP, AUXP, VP, etc.} (24 for German, 23 for French)

We denote that set of features in shorthand as $f(\alpha_i)$. With this extension, a model of Markov order *j* can potentially have an actual Markov order greater than *j*. Equation (5) is the extended model, which we will refer to as Type 2 (T2):

$$P(C|h)=\prod_{i=1}^{n}\begin{bmatrix} P\left(\psi_i \middle| d_{i-1},\psi_{i-1},...,d_{i-j},\psi_{i-j},c,h,f(\alpha_i)\right) \\ \times P\left(d_i \middle| \psi_i,d_{i-1},\psi_{i-1},...,d_{i-j},\psi_{i-j},c,h,f(\alpha_i)\right)\end{bmatrix} \tag{5}$$

As an alternative to a left-to-right expansion, we can also expand a constituent in a head-driven fashion. We refer the reader to (Ringger et al., 2003) for details and evaluations of several head-driven models (the missing "T3", "T4", and "T6" in this discussion).

2.2 Conditional Models

We now consider more complex models that use additional features. We define a function $g(X)$ on constituents, where the value of $g(X)$ represents a set of many lexical, syntactic, and semantic features of *X* (see section 5.2 for more details). No discourse features are included for the present work. We condition on

- $g(B)$, where *B* is the unordered constituent being ordered
- $g(H)$, where *H* is the head of *B*
- $g(P_B)$, where P_B is the parent of *B*, and
- $g(G_B)$, where G_B is the grandparent of *B*.

These features are fixed in the given unordered tree ρ, as in the discussion of equation (2), hence the resulting complex model is still a joint model.

Up until this point, we have been describing joint generative models that describe how to generate an ordered tree from an unordered tree. These models require extra effort and capacity to accurately model the inter-relations among all features. Now we move on to truly conditional models by including features that are functions on the set β_i of daughters of *C* yet to be ordered. In the conditional models we do not need to model the interdependencies among all features. We include the following:

- Number of daughters remaining to be ordered (size of β_i)
- Number of daughters in β_i having a particular label

As before, we denote these feature sets in shorthand as $f(\beta_i)$. The resulting distribution is represented in equation (6), which we will refer to as Type 5 (T5):

$$\begin{aligned} &P\left(C \middle| g(H),g(B),g(P_B),g(G_B)\right) \\ &=\prod_{i=1}^{n}\begin{bmatrix} P\left(\psi_i \middle| \begin{matrix} d_{i-1},\psi_{i-1},...,d_{i-j},\psi_{i-j},c,h, \\ g(H),g(B),g(P_B),g(G_B),f(\alpha_i),f(\beta_i)\end{matrix}\right) \\ \times P\left(d_i \middle| \begin{matrix} \psi_i,d_{i-1},\psi_{i-1},...,d_{i-j},\psi_{i-j},c,h, \\ g(H),g(B),g(P_B),g(G_B),f(\alpha_i),f(\beta_i)\end{matrix}\right)\end{bmatrix}\end{aligned} \tag{6}$$

All models in this paper are nominally Markov order 2, although those models incorporating the additional feature functions $f(\alpha_i)$ and $f(\beta_i)$ defined in Section 2.2 can be said to have higher order.

[3] A "Markov grammar" is a model of constituent structure that starts at the root of the tree and assigns probability to the expansion of a non-terminal one daughter at a time, rather than as entire productions (Charniak, 1997 & 2000).

2.3 Binary conditional model

We introduce one more model type called the *binary conditional model*. It estimates a much simpler distribution over the binary variable σ called "sort-next" with values in {*yes*, *no*} representing the event that an as-yet unordered member *D* of β_i (the set of as-yet unordered daughters of parent *C*, as defined above) should be "sorted" next, as illustrated in Figure 3.

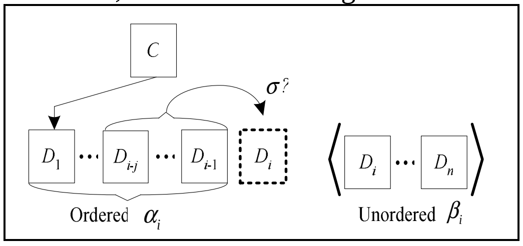


Figure 3: Binary conditional model.

The conditioning features are almost identical to those used in the left-to-right conditional models represented in equation (6) above, except that d_i and ψ_i (the semantic relation of *D* with head *H*) appear in the conditional context and need not first be predicted. In its simple form, the model estimates the following distribution:

$$P\left(\sigma_i \middle| \begin{matrix} d_i, \psi_i, d_{i-1}, \psi_{i-1}, ..., d_{i-j}, \psi_{i-j}, c, h, \\ g(H), g(B), g(P_B), g(G_B), f(\alpha_i), f(\beta_i) \end{matrix}\right) \quad (7)$$

In our shorthand, we will call this Type 7 (T7). We describe how to apply this model directly in a left-to-right "sorting" search later in the section on search.

3 Estimation

We estimate a model's distributions with probabilistic decision trees (DTs).[4] We build decision trees using the WinMine toolkit (Chickering, 2002). WinMine-learned decision trees are not just classifiers; each leaf is a conditional probability distribution over the target random variable, given all features available in training; hence the tree as a whole is an estimate of the conditional distribution of interest. The primary advantage of using decision trees, is the automatic feature selection and induction from a large pool of features.

We train four models for German and French each. One model is joint (T1); one is joint with additional features on the set of daughters already ordered (T2); one is conditional (T5). In addition, we employ one binary conditional DT model (T7), both with and without normalization (see equation (8)).

[4] Other approaches to feature selection, feature induction, and distribution estimation are certainly possible, but they are beyond the scope of this paper. One experiment using interpolated language modeling techniques is described in (Ringger et al., 2003)

4 Search

4.1 Exhaustive search

Given an unordered tree ρ and a model of constituent structure ***O*** of any of the types already presented, we search for the best ordered tree π that maximizes $P_O(\pi)$ or $P_O(\pi|\rho)$, as appropriate, with the context varying according to the complexity of the model. Each of our models (except the binary conditional model) estimates the probability of an ordering of any given constituent *C* in π, independently of the ordering inside other constituents in π. The complete search is a dynamic programming (DP) algorithm, either left-to-right in the daughters of *C* (or head-driven, depending on the model type). The search can optionally maintain one non-statistical constraint we call Input-Output Coordination Consistency (IOCC), so that the order of coordinated constituents is preserved as they were specified in the given unordered tree. For these experiments, we employ the constraint.

4.2 Greedy search for binary conditional model

The binary conditional model can be applied in a left-to-right "sorting" mode (Figure 3). At stage *i*, for each unordered daughter D_j, in β_i, the model is consulted for the probability of $\sigma_j = yes$, namely the probability that D_j should be placed to the right of the already ordered sister constituents α_i. The daughter in β_i with the highest probability is removed from β_i to produce β_{i+1} and added to the right of α_i to produce α_{i+1}. The search proceeds through the remaining unordered constituents until all constituents have been ordered in this greedy fashion.

4.3 Exhaustive search for binary conditional model

In order to apply the binary conditional model in the exhaustive DP search, we normalize the model at every stage of the search and thereby coerce it into a probability distribution over the remaining daughters in β_i. We represent the distribution in "equation" (7) in short-hand as $P(\sigma|d,\psi,\Gamma_i)$, with Γ_i representing the contextual features for the given search hypothesis at search stage *i*. Thus, our normalized distribution for stage *i* is given by equation (8). Free variable *j* represents an index on unordered daughters in β_i, as does *k*.

$$P\left(D_j \middle| d_j, \psi_j, \Gamma_i\right) = \frac{P\left(\sigma_j = yes \middle| d_j, \psi_j, \Gamma_i\right)}{\sum_{k=1}^{\|\beta_i\|} P\left(\sigma_k = yes \middle| d_k, \psi_k, \Gamma_i\right)} \quad (8)$$

This turns out to be the decision tree analogue of a Maximum Entropy Markov Model (MEMM) (McCallum et al., 2000), which we can refer to as a DTMM.

5 Experiments

5.1 Training

We use a training set of 20,000 sentences, both for French and German. The data come from technical manuals in the computer domain. For a given sentence in our training set, we begin by analyzing the sentence as a surface syntax tree and an abstract predicate argument structure using the NLPWin system (Heidorn, 2000). By consulting these two linked structures, we produce a tree with all of the characteristics of trees seen by the Amalgam ordering stage at generation run-time with one exception: these training trees are properly ordered. The training trees include all features of interest, including the semantic relations among a syntactic head and its modifiers. We train our order models from the constituents of these trees. NLPWin parser output naturally contains errors; hence, the Amalgam training data is imperfect.

5.2 Selected Features

A wide range of linguistic features is extracted for the different decision tree models. The number of selected features for German reaches 280 (out of 651 possible features) in the binary conditional model T7. For the French binary conditional model, the number of selected features is 218 (out of 550). The binary conditional models draw from the full set of available features, including:

- lexical sub-categorization features such as transitivity and compatibility with clausal complements
- lemmas (word-stems)
- semantic features such as the semantic relation and the presence of quantificational operators
- length of constituent in words
- syntactic information such as the label and the presence of syntactic modifiers

5.3 Evaluation

To evaluate the constituent order models in isolation, we designed our experiments to be independent of the rest of the Amalgam sentence realization process. We use test sets of 1,000 sentences, also from technical manuals, for each language. To isolate ordering, for a given test sentence, we process the sentence as in training to produce an ordered tree π (the reference for evaluation) and from it an unordered tree ρ. Given ρ, we then search for the best ordered tree hypothesis $\hat{\pi}$ using the model in question.

We then compare π and $\hat{\pi}$. Because we are only ordering constituents, we can compare π and $\hat{\pi}$ by comparing their respective constituents. For each C in π, we measure the *per-constituent edit distance D*, between C and its counterpart C' in $\hat{\pi}$ as follows:

1. Let d be the edit distance between the ordered set of daughters in each, with the only possible edit operators being *insert* and *delete*
2. Let the number of *moves* $m = d/2$, since insertions and deletions can be paired uniquely
3. Divide by the total number of daughters: $D = m / \left| daughters(C) \right|$

This metric is like the "Generation Tree Accuracy" metric of Bangalore & Rambow (2000), except that there is no need to consider cross-constituent moves. The total score for the hypothesis tree $\hat{\pi}$ is the mean of the per-constituent edit distances.

For each of the models under consideration and each language, we report in Table 1 the average score across the test set for the given language. The first row is a baseline computed from randomly scrambling constituents (mean over four iterations).

Model	German	French
Baseline (random)	35.14 %	34.36 %
T1: DT joint	5.948%	3.828%
T2: DT joint with $f(\alpha_i)$	5.852%	4.008%
T5: DT conditional	6.053%	4.271%
T7: DT binary cond., greedy search	3.516%	1.986%
T7: DT normalized binary conditional, exhaustive search	3.400%	1.810%

Table 1: Mean per-constituent edit distances for German & French.

5.4 Discussion

For both German and French, the binary conditional DT model outperforms all other models. Normalizing the binary conditional model and applying it in an exhaustive search performs better than a greedy search. All score differences are statistically significant; moreover, manual inspection of the differences for the various models also substantiates the better quality of those models with lower scores.

With regard to the question of conditional versus joint models, the joint models (T1, T2) outperform their conditional counterparts (T5). This may be due to a lack of sufficient training data for the conditional models. At this time, the training time of the conditional models is the limiting factor.

There is a clear disparity between the performance of the German models and the performance of the French models. The best German model is twice as bad as the best French model. (For a discussion of the impact of modeling German verb position, please consult (Ringger et al., 2003).)

	Baseline (random)	Greedy, IOCC	Greedy	DP, IOCC	DP
Total Sentences	2416	2416	2416	2416	2416
Mean Tokens/Sentence	23.59	23.59	23.59	23.59	23.59
Time/Input (sec.)	n/a	0.01	0.01	0.39	0.43
Exact Match	0.424%	33.14%	27.53%	33.53%	35.72%
Coverage	100%	100%	100%	100%	100%
Mean Per-Const. Edit Dist.	38.3%	6.02%	6.84%	5.25%	4.98%
Mean NIST SSA	-16.75	74.98	67.19	74.65	73.24
Mean IBM Bleu Score	0.136	0.828	0.785	0.817	0.836

Table 2: DSIF-Amalgam ordering performance on WSJ section 23.

6 Evaluation on the Penn TreeBank

Our goal in evaluating on Penn Tree Bank (PTB) data is two-fold: (1) to enable a comparison of Amalgam's performance with other systems operating on similar input, and (2) to measure Amalgam's capabilities on less domain-specific data than technical software manuals. We derive from the bracketed tree structures in the PTB using a deterministic procedure an abstract representation we refer to as a Dependency Structure Input Format (DSIF), which is only loosely related to NLPWin's abstract predicate-argument structures.

The PTB to DSIF transformation pipeline includes the following stages, inspired by Langkilde-Geary's (2002b) description:

A. Deserialize the tree
B. Label heads, according to Charniak's head labeling rules (Charniak, 2000)
C. Remove empty nodes and flatten any remaining empty non-terminals
D. Relabel heads to conform more closely to the head conventions of NLPWin
E. Label with logical roles, inferred from PTB functional roles
F. Flatten to maximal projections of heads (MPH), except in the case of conjunctions
G. Flatten non-branching non-terminals
H. Perform dictionary look-up and morphological analysis
I. Introduce structure for material between paired delimiters and for any coordination not already represented in the PTB
J. Remove punctuation
K. Remove function words
L. Map the head of each maximal projection to a dependency node, and map the head's modifiers to the first node's dependents, thereby forming a complete dependency tree.

To evaluate ordering performance alone, our data are obtained by performing all of the steps above except for (J) and (K). We employ only a binary conditional ordering model, found in the previous section to be the best of the models considered. To train the order models, we use a set of 10,000 sentences drawn from the standard PTB training set, namely sections 02–21 from the Wall Street Journal portion of the PTB (the full set contains approx. 40,000 sentences). For development and parameter tuning we used a separate set of 2000 sentences drawn from sections 02–21.

Decision trees are trained for each of five constituent types characterized by their head labels: adjectival, nominal, verbal, conjunctions (coordinated material), and other constituents not already covered. The split DTs can be thought of as a single DT with a five-way split at the top node.

Our DSIF test set consists of the blind test set (section 23) of the WSJ portion of the PTB. At run-time, for each converted tree in the test set, all daughters of a given constituent are first permuted randomly with one another (scrambled), with the option for coordinated constituents to remain unscrambled, according to the Input-Output Coordination Consistency (IOCC) option. For a given unordered (scrambled) constituent, the appropriate order model (noun-head, verb-head, etc.) is used in the ordering search to order the daughters. Note that for the greedy search, the input order can influence the final result; therefore, we repeat this process for multiple random scramblings and average the results.

We use the evaluation metrics employed in published evaluations of HALogen, FUF/SURGE, and FERGUS (e.g., Calloway, 2003), although our results are for ordering only. Coverage, or the percentage of inputs for which a system can produce a corresponding output, is uninformative for the Amalgam system, since in all cases, it can generate an output for any given DSIF. In addition to processing time per input, we apply four other metrics: exact match, NIST simple string accuracy (the complement of the familiar word error rate), the IBM Bleu score (Papineni et al., 2001), and the intra-constituent edit distance metric introduced earlier.

We evaluate against ideal trees, directly computed from PTB bracketed tree structures. The results in Table 2 show the effects of varying the IOCC parameter. For both trials involving a greedy search, the results were averaged across 25 iterations. As should be expected, turning on the input-output faithfulness option (IOCC) improves the performance of the greedy search. Keeping coordinated material in the same relative order would only be called for in applications that plan discourse structure before or during generation.

7 Conclusions and Future Work

The experiments presented here provide conclusive reasons to favor the binary conditional model as a model of constituent order. The inclusion of linguistic features is of great value to the modeling of order, specifically in verbal constituents for both French and German. Unfortunately space did not permit a thorough discussion of the linguistic features used. Judging from the high number of features that were selected during training for participation in the conditional and binary conditional models, the availability of automatic feature selection is critical.

Our conditional and binary conditional models are currently lexicalized only for function words; the joint models not at all. Experiments by Daumé *et al* (2002) and the parsing work of Charniak (2000) and others indicate that further lexicalization may yield some additional improvements for ordering. However, the parsing results of Klein & Manning (2003) involving unlexicalized grammars suggest that gains may be limited.

For comparison, we encourage implementers of other sentence realization systems to conduct order-only evaluations using PTB data.

Acknowledgements

We wish to thank Irene Langkilde-Geary and members of the MSR NLP group for helpful discussions. Thanks also go to the anonymous reviewers for helpful feedback.

References

Aikawa T., Melero M., Schwartz L. Wu A. 2001. Multilingual sentence generation. In *Proc. of 8th European Workshop on NLG*. pp. 57-63.

Bangalore S. Rambow O. 2000. Exploiting a probabilistic hierarchical model for generation. In *Proc. of COLING*. pp. 42-48.

Calloway, C. 2003. Evaluating Coverage for Large Symbolic NLG Grammars. In *Proc. of IJCAI 2003*. pp 811-817.

Charniak E. 1997. Statistical Techniques for Natural Language Parsing, In *AI Magazine*.

Charniak E. 2000. A Maximum-Entropy-Inspired Parser. In *Proc. of ACL*. pp.132-139.

Chickering D. M. 2002. The WinMine Toolkit. Microsoft Technical Report 2002-103.

Corston-Oliver S., Gamon M., Ringger E., Moore R. 2002. An overview of Amalgam: a machine-learned generation module. In *Proc. of INLG*. pp.33-40.

Daumé III H., Knight K., Langkilde-Geary I., Marcu D., Yamada K. 2002. The Importance of Lexicalized Syntax Models for Natural Language Generation Tasks. In *Proc. of INLG*. pp. 9-16.

Gamon M., Ringger E., Corston-Oliver S. 2002a. Amalgam: A machine-learned generation module. Microsoft Technical Report 2002-57.

Gamon M., Ringger E., Corston-Oliver S., Moore R. 2002b. Machine-learned contexts for linguistic operations in German sentence realization. In *Proc. of ACL*. pp. 25-32.

Heidorn G. 2000. Intelligent Writing Assistance. In *A Handbook of Natural Language Processing,,* R. Dale, H. Moisl, H. Somers (eds.). Marcel Dekker, NY.

Klein D., Manning C. 2003. "Accurate Unlexicalized Parsing." In *Proceedings of ACL-03*.

Langkilde I. 2000. Forest-Based Statistical Sentence generation. In *Proc. of NAACL*. pp. 170-177.

Langkilde-Geary I. 2002a. An Empirical Verification of Coverage and Correctness for a General-Purpose Sentence Generator. In *Proc. of INLG*. pp.17-24.

Langkilde-Geary, I. 2002b. *A Foundation for General-purpose Natural Language Generation: Sentence Realization Using Probabilistic Models of Language*. PhD Thesis, University of Southern California.

Langkilde I., Knight K. 1998a. The practical value of n-grams in generation. In *Proc. of 9th International Workshop on NLG*. pp. 248-255.

Langkilde I., Knight K. 1998b. Generation that exploits corpus-based statistical knowledge. In *Proc. of ACL and COLING*. pp. 704-710.

McCallum A., Freitag D., & Pereira F. 2000. „Maximum Entropy Markov Models for Information Extraction and Segmentation." In *Proc. Of ICML-2000*.

Papineni, K.A., Roukos, S., Ward, T., and Zhu, W.J. 2001. Bleu: a method for automatic evaluation of machine translation. IBM Technical Report RC22176 (W0109-022).

Reiter E. and Dale R. 2000. *Building natural language generation systems*. Cambridge University Press, Cambridge.

Ringger E., Gamon M., Smets M., Corston-Oliver S. and Moore R. 2003 Linguistically informed models of constituent structure for ordering in sentence realization. Microsoft Research technical report MSR-TR-2003-54.

Smets M., Gamon M., Corston-Oliver S. and Ringger E. (2003) The adaptation of a machine-learned sentence realization system to French. In *Proceedings of EACL*.