A Syntactic Time-Series Model for Parsing Fluent and Disfluent Speech*

Tim Miller
University of Minnesota
tmill@cs.umn.edu

William Schuler
University of Minnesota
schuler@cs.umn.edu

Abstract

This paper describes an incremental approach to parsing transcribed spontaneous speech containing disfluencies with a Hierarchical Hidden Markov Model (HHMM). This model makes use of the right-corner transform, which has been shown to increase non-incremental parsing accuracy on transcribed spontaneous speech (Miller and Schuler, 2008), using trees transformed in this manner to train the HHMM parser. Not only do the representations used in this model align with structure in speech repairs, but as an HMM-like time-series model, it can be directly integrated into conventional speech recognition systems run on continuous streams of audio. A system implementing this model is evaluated on the standard task of parsing the Switchboard corpus, and achieves an improvement over the standard baseline probabilistic CYK parser.

1 Introduction

Disfluency is one obstacle preventing speech recognition systems from being able to recognize spontaneous speech. Perhaps the most challenging aspect of disfluency recognition is the phenomenon of speech repair, which involves a speaker realizing a mistake, cutting off the flow of speech, and then continuing on, possibly retracing and replacing part of the utterance to that point. This paper will describe a system which applies a syntactic model of speech repair to a time-series parsing model, and evaluate that system on the standard Switchboard corpus parsing task.

The speech repair terminology used here follows that of Shriberg (1994). A speech repair consists of a *reparandum*, an *interruption point*, and the *alteration*. The reparandum contains the words that the speaker means to replace, including both words that are in error and words that will be retraced. The interruption point is the point in time where the stream of speech is actually stopped, and the repairing of the mistake can begin. The alteration contains the words that are meant to replace the words in the reparandum.

2 Background

Historically, research in speech repair has focused on acoustic cues such as pauses and prosodic contours for detecting repairs, which could then be excised from the text for improved transcription. Recent work has also looked at the possible contribution of higher-level cues, including syntactic structure, in the detection of speech repair. Some of this work is inspired by Levelt's (1983) investigation of the syntactic and semantic variables in speech repairs, particularly his well-formedness rule, which states that the reparandum and alteration of a repair typically have the same consititutent label, similar to coordination.

Hale et al. (2006) use Levelt's well-formedness rule to justify an annotation scheme where unfinished categories (marked X-UNF) have the UNF label appended to all ancestral category labels all the way up to the top-most constituent beneath an EDITED label (EDITED labels denoting a reparandum). They reason that this should prevent grammar rules of finished constituents in the corpus from corrupting the grammar of unfinished constituents. While this annotation proves helpful, it also leads to the unfortunate result that a large reparandum requires several special repair rules to be applied, even though the actual error is only happening at one point.

Intuitively, though, it seems that an error is only occurring at the end of the reparandum, and that until that point only fluent grammar rules are being applied by the speaker. This intuition has also been confirmed by empirical studies (Nakatani and Hirschberg, 1994), which show that there is no obvious error signal in speech up until the moment of interruption. Although speakers may retrace much of the reparandum for clarity or other reasons, ideally the reparandum contains nothing but standard grammatical rules until

*The authors would like to thank the anonymous reviewers for their input. This research was supported by National Science Foundation CAREER/PECASE award 0447685. The views expressed are not necessarily endorsed by the sponsors.

the speech is interrupted.[1]

Another recent approach to this problem (Johnson and Charniak, 2004) uses a tree-adjoining grammar (TAG) approach to define a mapping between a source sentence possibly containing a repair, and a target, fluent sentence. The use of the TAG channel model is justified by the putative crossing dependencies seen in repairs like *. . . a flight to Boston, uh, I mean, to Denver. . .* where there is repetition from the reparandum to the repair. Essentially, this model is building up the reparandum and alteration in tandem, based on these crossing dependencies. While this is an interesting model, it focuses on detection and removal of EDITED sections, and subsequent parsing of cleaned up speech. As such, it introduces challenges for integrating the system into a real-time speech recognizer.

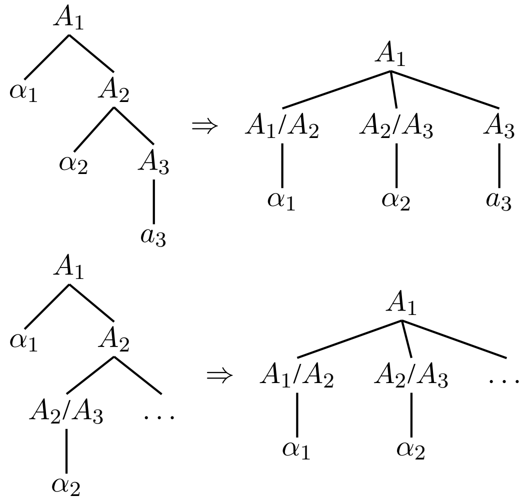
Recent work by Miller and Schuler (2008) showed how a probabilistic grammar trained on trees modified by use of the *right-corner transform* can improve parsing accuracy over an unmodified grammar when tested on the Switchboard corpus. The approach described here builds on that work in using right-corner transformed trees, and extends it by mapping them to a time-series model to do parsing directly in a model of the sort used in speech recognition. This system is shown to be more accurate than a baseline CYK parser when used to parse the Switchboard corpus. The remainder of this section will review the right-corner transform, followed by Section 3, which will step through an extended example giving details about the transform process and its applicability to the problem of processing speech repairs.

2.1 Right-corner transform

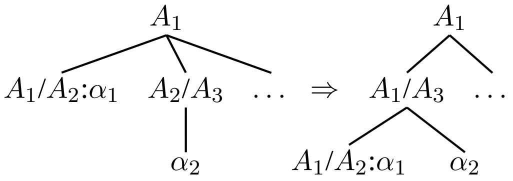
The right-corner transform rewrites syntax trees, turning all right branching structure into left branching structure, and leaving left branching structure as is. As a result, constituent structure can be explicitly built from the bottom up during incremental recognition. This arrangement is well-suited to recognition of speech with errors, because it allows constituent structure to be built up using fluent speech rules until the moment of interruption, at which point a special repair rule may be applied.

Before transforming the trees in the grammar into right-corner trees, trees are binarized in the same manner as Johnson (1998b) and Klein and Manning (2003).[2] Binarized trees are then transformed into right-corner trees using transform rules similar to those described by Johnson (1998a). In this transform, all right branching sequences in each tree are transformed into left branching sequences of symbols of the form A_1/A_2, denoting an incomplete instance of category A_1 lacking an instance of category A_2 to the right.[3]

Rewrite rules for the right-corner transform are shown below, first flattening right-branching structure:



then replacing it with left-branching structure:



Here, the first two rewrite rules are applied iteratively (bottom-up on the tree) to flatten all right branching structure, using incomplete constituents to record the original nonterminal ordering. The third rule is then applied to generate left branching structure, preserving this ordering.

An important property of this transform is that it is reversible. Rewrite rules for reversing a right-corner transform are simply the converse of those shown above.

Because this process turns right expansion into left expansion (leaving center expansion as the only stack consumer), right-corner transformed trees also require less stack memory than ordinary phrase structure trees. This key property of the right-corner transform is exploited in the mapping of transformed training trees to a time-series model. This property will be examined further in Section 5.

3 Speech Repair Example

A substantial example of a speech repair from the Switchboard corpus can be seen in Figures 1 and 2,

[1] One objection to this claim is the case of multiple nested repairs. In this case, though, we presume that all reparanda were originally intended by the speaker to be fluent at the time of generation.

[2] For the details of the particular binarization process used here, see Miller and Schuler (2008).

[3] Here, all A_i denote nonterminal symbols, and all α_i denote subtrees; the notation $A_1{:}\alpha_1$ indicates a subtree α_1 with label A_1; and all rewrites are applied recursively, from leaves to root.

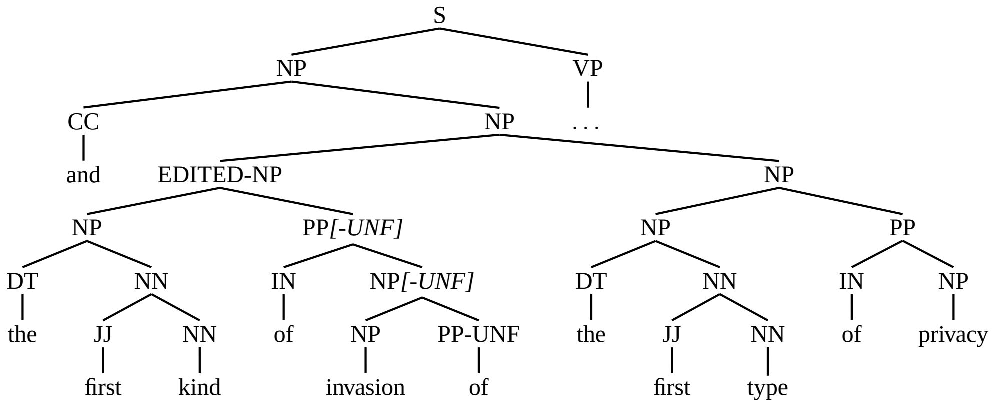


Figure 1: Binarized tree repair structure, with the -UNF propagation as in Hale et al. (2006) shown in brackets.

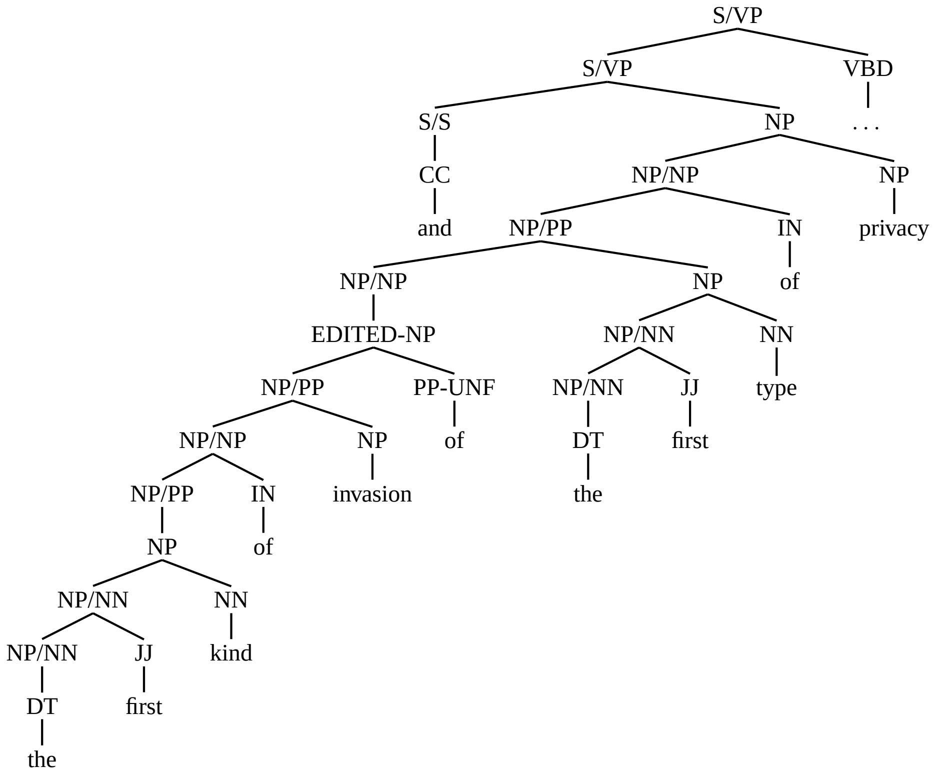


Figure 2: Right-corner transformed tree with repair structure

in which the same repair and surrounding context is shown after the preliminary binarization process, and after the right-corner transform. Figure 1 also shows, in brackets, the augmented annotation described above from Hale et al. (2006). This scheme consisted of adding -X to an EDITED label which produced a category X (daughter annotation), as well as propagating the -UNF label at the right corner of the tree up through every parent below the EDITED root.

3.1 Re-annotation of speech repair

Before applying the right-corner transform, some corpus pre-processing steps are applied — Figure 1 shows an example tree after these changes. These steps aim to improve on the default annotation of repair in the Switchboard corpus by making the representation more in line with linguistic as well as processing desiderata.

The standard annotation practice of having the EDITED label as the category at the root of a reparandum does not represent any proposed linguistic phenomenon, and it conflates speech repairs of different categories. As a result, the parser is unable to make use of information about which syntactic category the reparandum was originally intended to be. This information would be useful in the example discussed here, since an unfinished NP is followed by a completed NP. The daughter annotation used by Hale et al. fixes this annotation to allow the parser to have access to this information. The annotation scheme in this paper also makes use of the daughter annotation.

In addition, trees are modified to reduce the arity of speech repair rules, and to change the model of how repairs occur. The default Switchboard has, e.g. an NP reparandum (headed by EDITED) and the NP alteration represented as siblings in the syntax tree. This annotation seems to implicitly model all repairs as stopping the process of producing the current constituent, then starting a new one. In contrast, the representation here models repairs of this type as a speaker generating a partial noun phrase, realizing an error, and then continuing to generate *the same noun phrase*. This representation scheme thus represents the beginning of an effort to distinguish between repairs that are changing the direction of the utterance and those that are just fixing a local mistake.

3.2 Right-corner transform model of speech repair

The right corner transform is then applied to this example in Figure 2. The resulting tree representation is especially valuable because it models human production of speech repairs well, by not applying any special rule until the moment of interruption.

In the example in Figure 1, there is an unfinished constituent (PP-UNF) at the end of the reparandum. This standard annotation is deficient because even if an unfinished consituent like PP-UNF is correctly recognized, and the speaker is essentially in an error state, there may be several partially completed constituents above — in Figure 1, the NP, PP, and NP above the PP-UNF. These constituents need to be completed, but using the standard annotation there is only one chance to make use of the information about the error that has occurred — the 'NP → NP PP-UNF' rule. Thus, by the time the error section is completed, there is no information by which a parsing algorithm could choose to reduce the topmost NP to EDITED (or EDITED-NP) other than independent rule probabilities.

The approach used by Hale et al. (2006) works because the information about the transition to an "error state" is propagated up the tree, in the form of the -UNF tags. As the parsing chart is filled in from the bottom up, each rule applied is essentially coming out of a special repair rule set, and so at the top of the tree the EDITED hypothesis is much more likely. However, this requires that several fluent speech rules from the data set be modified for use in a special repair grammar, which not only reduces the amount of available training data, but violates our intuition that reparanda are usually fluent up until the actual edit occurs.

The right corner transform works in a different way, by building up constituent structure from left to right. In Figure 2, the same repair is shown as it appears in the training data for this system. With this representation, the problem noticed by Hale and colleagues has been solved in a different way, by incrementally building up *left-branching* rather than right-branching structure, so that only a single special error rule is required at the end of the constituent. As seen in the figure, all of the structure beneath the EDITED-NP label is built using rules from the fluent grammar. It is only at one point, when the PP-UNF is found, that a repair rule is applied and the EDITED-NP section is found. The next step in the process is that the NP essentially restarts (the NP/NP label), and the subsequent words start to build up what will be the NP alteration in a fluent manner.

To summarize, while the -UNF propagation scheme often requires the entire reparandum to be generated from a speech repair rule set, this scheme only requires one special rule, where the moment of interruption actually occurred. This reduces the number of special speech repair rules that need to be learned and saves more potential examples of fluent speech rules, and therefore potentially makes better use of limited data.

4 Mapping to an HHMM

This section describes how a corpus of trees transformed as above can be mapped to a time-series model called a Hierarchical Hidden Markov Model (HHMM) in order to incorporate parsing into speech decoding. This suggests that this approach can be used in applications using streaming speech input, unlike other parsing approaches which are cubic time on input length at best, and require input to be presegmented.

This section will begin by showing how HHMMs can model linguistic structure by extending standard Hidden Markov Models (HMMs) used in speech recognition, and will follow with a description of how right-corner transformed trees can be mapped to this model topology.

4.1 Hierarchical HMMs

In general, the hidden state in an HMM can be as simple or complex as necessary. This can include factorizing the hidden state into any number of interdependent random variables modeling the sub-states of the complex hidden state. A Hierarchical Hidden Markov Model is essentially an HMM with a specific factorization that is useful in many domains — the hidden state at each time step is factored into d random variables which function as a stack, and d additional boolean random variables which regulate the operations of the stack through time. The boolean random variables are typically marginalized out when performing inference on a sequence.

While the vertical direction of the hidden sub-states (at a fixed t) represents a stack at a single point in time, the horizontal direction of the hidden sub-states (at a fixed d) can be viewed as simple HMMs at depth d, taking direction from the HMM above them and controlling those below them. This interpretation will be useful when formally defining the transitions between the stack elements at different time steps below.

Formally, HMMs characterize speech or text as a sequence of hidden states q_t (which may consist of speech sounds, words, and/or other hypothesized syntactic or semantic information), and observed states o_t at corresponding time steps t (typically short, overlapping frames of an audio signal, or words or characters in a text processing application). A most likely sequence of hidden states $\hat{q}_{1..T}$ can then be hypothesized given any sequence of observed states $o_{1..T}$, using Bayes' Law (Equation 2) and Markov independence assumptions (Equation 3) to define a full $\mathsf{P}(q_{1..T} \mid o_{1..T})$ probability as the product of a *Language Model (Θ_L)* prior probability and an *Observation Model (Θ_O)* likelihood probability:

$$\hat{q}_{1..T} = \operatorname*{argmax}_{q_{1..T}} \mathsf{P}(q_{1..T} \mid o_{1..T}) \tag{1}$$

$$= \operatorname*{argmax}_{q_{1..T}} \mathsf{P}(q_{1..T}) \cdot \mathsf{P}(o_{1..T} \mid q_{1..T}) \tag{2}$$

$$\stackrel{\mathrm{def}}{=} \operatorname*{argmax}_{q_{1..T}} \prod_{t=1}^{T} \mathsf{P}_{\Theta_\mathrm{L}}(q_t \mid q_{t-1}) \cdot \mathsf{P}_{\Theta_\mathrm{O}}(o_t \mid q_t) \tag{3}$$

Language model transitions $\mathsf{P}_{\Theta_\mathrm{L}}(q_t \mid q_{t-1})$ over complex hidden states q_t can be modeled using synchronized levels of stacked-up component HMMs in a Hierarchic Hidden Markov Model (HHMM) (Murphy and Paskin, 2001). HHMM transition probabilities are calculated in two phases: a 'reduce' phase (resulting in an intermediate, marginalized state f_t), in which component HMMs may terminate; and a 'shift' phase (resulting in a modeled state q_t), in which unterminated HMMs transition, and terminated HMMs are re-initialized from their parent HMMs. Variables over intermediate f_t and modeled q_t states are factored into sequences of depth-specific variables — one for each of D levels in the HMM hierarchy:

$$f_t = \langle f_t^1 \ldots f_t^D \rangle \tag{4}$$

$$q_t = \langle q_t^1 \ldots q_t^D \rangle \tag{5}$$

Transition probabilities are then calculated as a product of transition probabilities at each level, using level-specific 'reduce' Θ_F and 'shift' Θ_Q models:

$$\mathsf{P}(q_t \mid q_{t-1}) = \sum_{f_t} \mathsf{P}(f_t \mid q_{t-1}) \cdot \mathsf{P}(q_t \mid f_t\; q_{t-1}) \tag{6}$$

$$\stackrel{\mathrm{def}}{=} \sum_{f_t^1 .. f_t^D} \prod_{d=1}^{D} \mathsf{P}_{\Theta_\mathrm{F}}(f_t^d \mid f_t^{d+1} q_{t-1}^d q_{t-1}^{d-1}) \cdot \prod_{d=1}^{D} \mathsf{P}_{\Theta_\mathrm{Q}}(q_t^d \mid f_t^{d+1} f_t^d\; q_{t-1}^d q_t^{d-1}) \tag{7}$$

with f_t^{D+1} and q_t^0 defined as constants.

Shift and reduce probabilities are now defined in terms of finitely recursive FSAs with probability distributions over transition, recursive expansion, and final-state status of states at each hierarchy level. In simple HHMMs, each intermediate state variable is a boolean switching variable $f_t^d \in \{\mathbf{0}, \mathbf{1}\}$ and each modeled state variable is a syntactic, lexical, or phonetic state q_t^d. The intermediate variable f_t^d is true (equal to $\mathbf{1}$) with probability 1 if there is a transition at the level immediately below d and the stack element q_{t-1}^d is a final state, and false (equal to $\mathbf{0}$) with probability 1 otherwise:[4]

$$\mathsf{P}_{\Theta_\mathrm{F}}(f_t^d \mid f_t^{d+1} q_{t-1}^d q_{t-1}^{d-1}) \stackrel{\mathrm{def}}{=} \begin{cases} \text{if } f_t^{d+1}{=}\,\mathbf{0} : [f_t^d{=}\,\mathbf{0}] \\ \text{if } f_t^{d+1}{=}\,\mathbf{1} : \mathsf{P}_{\Theta_\text{F-Reduce}}(f_t^d \mid q_{t-1}^d, q_{t-1}^{d-1}) \end{cases} \tag{8}$$

where $f^{D+1} = \mathbf{1}$ and $q_t^0 = \mathbf{ROOT}$.

Shift probabilities at each level are defined using level-specific transition $\Theta_\text{Q-Trans}$ and expansion $\Theta_\text{Q-Expand}$ models:

$$\mathsf{P}_{\Theta_\mathrm{Q}}(q_t^d \mid f_t^{d+1} f_t^d\; q_{t-1}^d q_t^{d-1}) \stackrel{\mathrm{def}}{=} \begin{cases} \text{if } f_t^{d+1}{=}\,\mathbf{0},\ f_t^d{=}\,\mathbf{0} : [q_t^d{=}\,q_{t-1}^d] \\ \text{if } f_t^{d+1}{=}\,\mathbf{1},\ f_t^d{=}\,\mathbf{0} : \mathsf{P}_{\Theta_\text{Q-Trans}}(q_t^d \mid q_{t-1}^d q_t^{d-1}) \\ \text{if } f_t^{d+1}{=}\,\mathbf{1},\ f_t^d{=}\,\mathbf{1} : \mathsf{P}_{\Theta_\text{Q-Expand}}(q_t^d \mid q_t^{d-1}) \end{cases} \tag{9}$$

where $f^{D+1} = \mathbf{1}$ and $q_t^0 = \mathbf{ROOT}$. This model is conditioned on final-state switching variables at and immediately below the current HHMM level. If there is no final state immediately below the current level (the first case above), it deterministically copies the current HHMM state forward to the next time step. If there is a final state immediately below the current level (the second case above), it transitions the HHMM state at the current level, according to the distribution $\Theta_\text{Q-Trans}$. And if the state at the current level is final (the third case above), it re-initializes this state given the state at the level above, according to the distribution $\Theta_\text{Q-Expand}$. The overall effect

[4] Here $[\cdot]$ is an indicator function: $[\phi] = 1$ if ϕ is true, 0 otherwise.

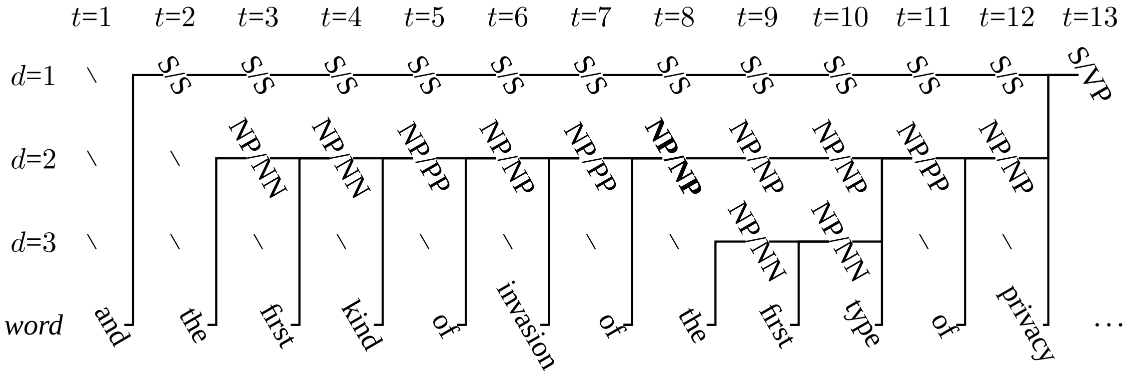


Figure 3: Sample tree from Figure 2 mapped to q_t^d variable positions of an HHMM at each stack depth d (vertical) and time step t (horizontal). Values for final-state variables f_t^d are not shown. Note that some nonterminal labels have been omitted; labels for these nodes can be reconstructed from their children. This includes the EDITED-NP nonterminal that occurs as a child of the NP/NP at t=8, d=2, indicated in boldface.

is that higher-level HMMs are allowed to transition only when lower-level HMMs terminate. An HHMM therefore behaves like a probabilistic implementation of a pushdown automaton (or 'shift-reduce' parser) with a finite stack, where the maximum stack depth is equal to the number of levels in the HHMM hierarchy.

4.2 Mapping trees to HHMM derivations

Any tree can now be mapped to an HHMM derivation by aligning the nonterminals with q_t^d categories. First, it is necessary to define rightward depth d, right index position t, and final (right) child status f, for every nonterminal node A in a tree, where:

- d is defined to be the number of right branches between node A and the root,
- t is defined to be the number of words beneath or to the left of node A, and
- f is defined to be **0** if node A is a left (or unary) child, **1** otherwise.

Any binary-branching tree can then be annotated with these values and rewritten to define labels and final-state values for every combination of d and t covered by the tree. This process simply copies stacked up constituents over multiple time steps, while other constituents are being recognized. Coordinates $d, t \leq D, T$ that are not covered by the tree are assigned label '$-$', and $f = \mathbf{1}$. The resulting label and final-state values at each node now define a value of q_t^d and f_{t+1}^d for each depth d and time step t of the HHMM (see Figure 3). Probabilities for HHMM models $\Theta_{\text{Q-Expand}}$, $\Theta_{\text{Q-Trans}}$, and $\Theta_{\text{F-Reduce}}$ can then be estimated from these values directly. Like the right-corner transform, this mapping is reversible, so q and f values can be taken from a hypothesized most likely sequence and mapped back to trees (which can then undergo the reverse of the right-corner transform to become ordinary phrase structure trees).

5 HHMM Application to Speech Repair

While the HHMM parser described above can produce the same output as a standard probablistic CYK parser (the most likely parse tree), the different parsing strategy of an HHMM parser and the close connection of this system with a probabilistic model of semantics present potential benefits to the recognition of disfluent speech.

First, by using a depth-limited stack, this model better adheres to psycholinguistically observed short term memory limits that the human parser and generator are likely to obey (Cowan, 2001; Miller, 1956). The use of a depth-limited stack is enabled by the right-corner transform's property of transforming right expansions to left expansions, which minimizes stack usage. Corpus studies (Schuler et al., 2008) suggest that broad coverage parsing can be achieved via this transform using only four stack elements. In practical terms this means that the model is less likely than a standard CYK parser to spend time and probability mass on analyses that conflict with the memory limits humans appear to be constrained by when generating and understanding speech.

Second, this model is part of a more general framework that incorporates a model of referential semantics into the parsing model of the HHMM (Schuler et al., in press). While the framework evaluated in this paper models only the syntactic contribution to speech repair, there are some cases where syntactic cues are not sufficient to distinguish disfluent from fluent utterances. In many of these cases, semantic information is the only way to tell that an utterance contains a repair.[5] A recognition system that incorporates referential semantics with syntax should improve the accuracy of speech repair recognition as it has been shown to increase recognition of entities in fluent speech recognition (Schuler and Miller, 2005).

Finally, the semantic model just described, as well as the mechanics of the HHMM parser on a right-corner transformed grammar, combine to form a model that accounts for two previously distant aspects of speech processing: referential semantics and speech repair. From the generative view of language processing, the model starts with a desired referent, and based on that referent selects the appropriate syntactic structures, and within those it selects the appropriate lexical items to unambiguously describe the referent. In the semantic sense, then, the model is operating in a top-down fashion, with the referent being the driving force for the generation of syntax and words. However, since the model is also working in a left-to-right fashion on a highly left-branching grammar, there is also a bottom-up composition of constituents, which models the phenomenon of speech repair naturally and accurately.

6 Evaluation

The evaluation of this system was performed on the Switchboard corpus of transcribed conversational speech, using the *mrg* annotations in directories 2 and 3 for training, and the files sw4004.mrg to sw4153.mrg in directory 4 for evaluation, following Charniak and Johnson (2001). In addition to testing the HHMM parser on the Switchboard corpus, the experiment testing a CYK parser from Miller and Schuler (2008) was replicated, with slightly better results due to a change in the evaluation script[6] and small changes in the binarization process (both of these changes affect the baseline and test systems).

The input to the system consists of the terminal symbols from the trees in the corpus section mentioned above. The terminal symbol strings are first pre-processed by stripping punctuation and empty categories, which could not be expected from the output of a speech recognizer. In addition, any information about repair is stripped from the input, including partial words, repair symbols,[7] and interruption point information. While an integrated system for processing and parsing speech may use both acoustic and syntactic information to find repairs, and thus may have access to some of this information about where interruptions occur, this testing paradigm is intended to evaluate the use of the right-corner transform in a time-series model on parsing speech repair. To make a fair comparison to the CYK baseline of Hale et al. (2006), the recognizer was given correct part-of-speech tags as input along with words.

System	Parseval F	EDIT F
Baseline	**63.43**	**41.82**
CYK (H06)	71.16	41.7
RCT	**73.21**	**61.03**
HHMM+RCT	**77.15**	**68.03**
TAG-based model (JC04)	–	79.7

Table 1: Baseline results are from a standard CYK parser with binarized grammar. We were unable to find the correct configuration to match the baseline results from Hale et al. RCT results are on the right-corner transformed grammar (transformed back to flat treebank-style trees for scoring purposes). HHMM+RCT results are for the HHMM parser described in this paper. CYK and TAG lines show relevant results from related work.

The results presented here use two standard metrics for assessing accuracy of transcribed speech with repairs. The first metric, Parseval F-measure, takes into account precision and recall of all non-terminal (and non pre-terminal) constituents in a hypothesized tree relative to the gold standard. The second metric, EDIT-finding F, measures precision and recall of the words tagged as EDITED in the hypothesized tree relative to those tagged EDITED in the gold standard. F score is defined as usual, $2pr/(p+r)$ for precision p and recall r.

Table 1 shows the results of experiments using the model of speech repair described in this paper. The 'Baseline' result shows the accuracy of the binarized grammar at parsing the Switchboard test set. The 'RCT' result shows the accuracy of parsing when the right-corner transform is performed on the trees in the training set prior to training. Finally, the 'HHMM+RCT' results shows the accuracy of the HHMM parser system described in this paper, trained on right-corner trees mapped to the random variables at each time step.

These results show an improvement over the standard CYK parsing algorithm, in both overall parsing accuracy and EDIT-finding accuracy. This shows that the HHMM parser, which is more applicable to speech input due to its asymptotic linear time com-

[5] For example, the sentence "The red…uh…blue box" is more likely to be considered a repair in a context with single colored boxes, whereas the sentence "The big…uh…blue box" is less likely to be considered a repair in the same context, although the two sentences have the same syntactic structure.

[6] Specifically, we switched to using the evalb tool created by Sekine and Collins (1997).

[7] The Switchboard corpus has special terminal symbols indicating e.g. the start and end of the reparandum.

plexity, does not need to sacrifice any accuracy to do so, and indeed improves on accuracy for both metrics under consideration.

7 Conclusion

The work described here has extended previous work for recognizing disfluent speech to an incremental model, moving in a direction that holds promise for eventual direct implementation in a speech recognizer.

The next step in this work will attempt to apply this parsing model to unsegmented utterances. In its current form, this model assumes pre-segmented utterances, but segmenting speech into sentence-like units is itself a difficult and unsolved research problem (Roark et al., 2006). This framework may be extended to perform sentence segmentation along with parsing, with relatively small changes to the system.

The time series model for parsing described here can then be extended to actual speech by making the word level a hidden variable, and adding a pronunciation model which generates the observed acoustic evidence.

Extending this model to actual speech adds some complexity, since disfluency phenomena are difficult to detect in an audio signal. However, there are also advantages in this extension, since the extra phonological variables and acoustic observations contain information that can be useful in the recognition of disfluency phenomena.

References

Charniak, Eugene and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *2nd Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 118–126.

Cowan, Nelson. 2001. The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences*, 24:87–185.

Hale, John, Izhak Shafran, Lisa Yung, Bonnie Dorr, Mary Harper, Anna Krasnyanskaya, Matthew Lease, Yang Liu, Brian Roark, Matthew Snover, and Robin Stewart. 2006. PCFGs with syntactic and prosodic indicators of speech repairs. In *Proceedings of the 45th Annual Conference of the Association for Computational Linguistics (COLING-ACL)*.

Johnson, Mark and Eugene Charniak. 2004. A tag-based noisy channel model of speech repairs. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04)*, pages 33–39, Barcelona, Spain.

Johnson, Mark. 1998a. Finite state approximation of constraint-based grammars using left-corner grammar transforms. In *Proceedings of COLING/ACL*, pages 619–623.

Johnson, Mark. 1998b. PCFG models of linguistic tree representation. *Computational Linguistics*, 24:613–632.

Klein, Dan and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430.

Levelt, William J.M. 1983. Monitoring and self-repair in speech. *Cognition*, 14:41–104.

Miller, Tim and William Schuler. 2008. A unified syntactic model for parsing fluent and disfluent speech. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL '08)*.

Miller, George A. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63:81–97.

Murphy, Kevin P. and Mark A. Paskin. 2001. Linear time inference in hierarchical HMMs. In *Proc. NIPS*, pages 833–840.

Nakatani, C. and J. Hirschberg. 1994. A corpus-based study of repair cues in spontaneous speech. *The Journal of the Acoustic Society of America*, 95:1603–1616.

Roark, Brian, Yang Liu, Mary Harper, Robin Stewart, Matthew Lease, Matthew Snover, Izhak Safran, Bonnie Dorr, John Hale, Anna Krasnyanskaya, and Lisa Yung. 2006. Reranking for sentence boundary detection in conversational speech. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP'06)*.

Schuler, William and Tim Miller. 2005. Integrating denotational meaning into a DBN language model. In *Proceedings of the 9th European Conference on Speech Communication and Technology / 6th Interspeech Event (Eurospeech/Interspeech'05)*, pages 901–904, Lisbon, Portugal.

Schuler, William, Samir AbdelRahman, Tim Miller, and Lane Schwartz. 2008. Toward a psycholinguistically-motivated model of language. In *Proceedings of COLING*, Manchester, UK.

Schuler, William, Stephen Wu, and Lane Schwartz. in press. A framework for fast incremental interpretation during speech decoding. *Computational Linguistics*.

Sekine, Satoshi and Michael Collins. 1997. Evalb bracket scoring program.

Shriberg, Elizabeth. 1994. *Preliminaries to a Theory of Speech Disfluencies*. Ph.D. thesis, University of California at Berkeley.