# HHMM Parsing with Limited Parallelism

**Tim Miller**
Department of Computer Science
and Engineering
University of Minnesota, Twin Cities
tmill@cs.umn.edu

**William Schuler**
University of Minnesota, Twin Cities
and The Ohio State University
schuler@ling.ohio-state.edu

## Abstract

Hierarchical Hidden Markov Model (HHMM) parsers have been proposed as psycholinguistic models due to their broad coverage within human-like working memory limits (Schuler et al., 2008) and ability to model human reading time behavior according to various complexity metrics (Wu et al., 2010). But HHMMs have been evaluated previously only with very wide beams of several thousand parallel hypotheses, weakening claims to the model's efficiency and psychological relevance. This paper examines the effects of varying beam width on parsing accuracy and speed in this model, showing that parsing accuracy degrades gracefully as beam width decreases dramatically (to $2\%$ of the width used to achieve previous top results), without sacrificing gains over a baseline CKY parser.

## 1 Introduction

Probabilistic parsers have been successful at accurately estimating syntactic structure from free text. Typically, these systems work by considering entire sentences (or utterances) at once, using dynamic programming to obtain globally optimal solutions from locally optimal sub-parses.

However, these methods usually do not attempt to conform to human-like processing constraints, e.g. leading to center embedding and garden path effects (Chomsky and Miller, 1963; Bever, 1970). For systems prioritizing accurate parsing performance, there is little need to produce human-like errors. But from a human modeling perspective, the success of globally optimized whole-utterance models raises the question of how humans can accurately parse linguistic input without access to this same global optimization. This question creates a niche in computational research for models that are able to parse accurately while adhering as closely as possible to human-like psycholinguistic constraints.

Recent work on incremental parsers includes work on Hierarchical Hidden Markov Model (HHMM) parsers that operate in linear time by maintaining a bounded store of incomplete constituents (Schuler et al., 2008). Despite this seeming limitation, corpus studies have shown that through the use of grammar transforms, this parser is able to cover nearly all sentences contained in the Penn Treebank (Marcus et al., 1993) using a small number of unconnected memory elements.

But this bounded-memory parsing comes at a price. The HHMM parser obtains good coverage within human-like memory bounds only by pursuing an 'optionally arc-eager' parsing strategy, nondeterministically guessing which constituents can be kept open for attachment (occupying an active memory element), or closed for attachment (freeing a memory element for subsequent constituents). Although empirically determining the number of parallel competing hypotheses used in human sentence processing is difficult, previous results in computational models have shown that human-like behavior can be elicited at very low levels of parallelism (Boston et al., 2008b; Brants and Crocker, 2000), suggesting that large numbers of active hypotheses are not needed. Previously, the HHMM parser has only been evaluated on large beam widths, leaving this aspect of its psycholinguistic plausibility untested.

In this paper, the performance of an HHMM parser will be evaluated in two experiments that

vary the amount of parallelism allowed during parsing, measuring the degree to which this degrades the system's accuracy. In addition, the evaluation will compare the HHMM parser to an off-the-shelf probabilistic CKY parser to evaluate the actual run time performance at various beam widths. This serves two purposes, evaluating one aspect of the plausibility of this parsing framework as a psycholinguistic model, and evaluating its potential utility as a tool for operating on unsegmented text or speech.

## 2 Related Work

There are several criteria a parser must meet in order to be plausible as a psycholinguistic model of the human sentence-processing mechanism (HSPM).

Incremental operation is perhaps the most obvious. The HSPM is able to process sentences incrementally, meaning that at each point in time of processing input, it has some hypothesis of the interpretation of that input, and each subsequent unit of input serves to update that hypothesis.

The next criterion for psycholinguistic plausibility is processing efficiency. The HSPM not only operates incrementally, but in standard operation it does not lag behind a speaker, even if, for example, the speaker continues speaking at extended length without pause. Standard machine approaches, such as chart parsers based on the CKY algorithm, operate in worst-case cubic run time on the length of input. Without knowing where an utterance or sentence might end, such an algorithm will take more time with each successive word and will eventually fall behind.

The third criterion is a reasonable limiting of memory resources. This constraint means that the HSPM, while possibly considering multiple hypotheses in parallel, is not limitlessly so, as evidenced by the existence of garden path sentences (Bever, 1970; Lewis, 2000). If this were not the case, garden-path sentences would not cause problems, as reaching the disambiguating word would simply result in a change in the favored hypothesis. In fact, garden path sentences typically cannot be understood on a first pass and must be reread, indicating that the correct analysis is attainable and yet not present in the set of parallel hypotheses of the first pass.

While parsers meeting these three criteria can claim to not violate any psycholinguistic constraints, there has been much recent work in testing psycholinguistically-motivated parsers to make forward predictions about human sentence processing, in order to provide positive evidence for certain probabilistic parsing models as valid psycholinguistic models of sentence processing. This work has largely focused on correlating measures of parsing difficulty in computational models with delays in reading time in human subjects.

Hale (2001) introduced the *surprisal* metric for probabilistic parsers, which measures the log ratio of the total probability mass at word $t-1$ and word $t$. In other words, it measures how much probability was lost in incorporating the next word into the current hypotheses. Boston et al. (2008a) show that surprisal is a significant predictor of reading time (as measured in self-paced reading experiments) using a probabilistic dependency parser. Roark et al. (2009) dissected parsing difficulty metrics (including surprisal and entropy) to separate out the effects of syntactic and lexical difficulties, and showed that these new metrics are strong predictors of reading difficulty.

Wu et al. (2010) evaluate the same Hierarchical Hidden Markov Model parser used in this work in terms of its ability to reproduce human-like results for various complexity metrics, including some of those mentioned above, and introduce a new metric called *embedding difference*. This metric is based on the idea of embedding depth, which is the number of elements in the memory store required to hold a given hypothesis. Using more memory elements corresponds to center embedding in phrase structure trees, and presumably correlates to some degree with complexity. Average embedding for a time step is computed by computing the weighted average number of required memory elements (weighted by probability) for all hypotheses on the beam. Embedding difference is simply the change in this value when the next word is encountered.

Outside of Wu et al., the most similar work from a modeling perspective is an incremental parser implemented using Cascaded Hidden Markov Models (CHMMs) (Crocker and Brants, 2000). This model is superficially similar to the Hierarchical Hidden Markov Models described below in that it relies on multiple levels of interdependent HMMs to account for hierarchical structure in an incremental model. Crocker and Brants use the system to parse ambiguous sentences (such

as *the athlete realized his goals were out of reach*) and examine the relative probabilities of two plausible analyses at each time step. They then show that the shifting of these two probabilities is consistent with empirical evidence about how humans perceive these sentences word by word.

However, as will be described below, the HHMM has advantages over the CHMM from a psycholinguistic modeling perspective. The HHMM uses a limited memory store containing only four elements which is consistent with many estimates of human short term memory limits (Cowan, 2001; Miller, 1956). In addition to modeling memory limits, the limited store acts as a fixed-depth stack that ensures linear asymptotic parsing time, and a grammar transform allows for wide coverage of speech and newspaper corpora within that limited memory store (Schuler et al., 2010).

## 3 Hierarchical Hidden Markov Model Parser

Hidden Markov Models (HMMs) have long been used to successfully model sequence data in which there is a latent (hidden) variable at each time step that generates the observed evidence at that time step. These models are used for such applications as part-of-speech tagging, and speech recognition.

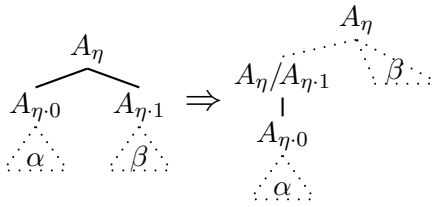
Hierarchical Hidden Markov Models (HHMMs) are an extension of HMMs which can represent sequential data containing hierarchical relations. In HHMMs, complex hidden variables may output evidence for several time steps in sequence. This process may recurse, though a finite depth is required to make any guarantees about performance. Murphy and Paskin (2001) showed that this model could be framed as a Dynamic Bayes Network (DBN), so that inference is linear on the length of the input sequence.

In the HHMM parser used here, the complex hidden variables are syntactic states that generate sub-sequences of other syntactic states, eventually generating pre-terminals and words. This section will describe how the trees must be transformed, and then mapped to HHMM states. This section will then continue with a formal definition of an HHMM, followed by a description of how this model can parse natural language, and finally a discussion of what different aspects of the model represent in terms of psycholinguistic modeling.

### 3.1 Right-Corner Transform

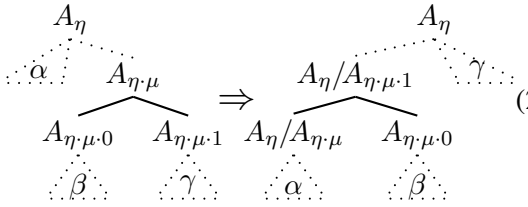
In order to parse with an HHMM, phrase structure trees need to be mapped to a hierarchical sequence of states of nested HMMs. Since Murphy and Paskin showed that the run time complexity of the HHMM is exponential on the depth of the nested HMMs, it is important to minimize the depth of the model for optimal performance. In order to do this, a tree transformation known as a *right-corner transform* is applied to the phrase structure trees comprising the training data, to transform right-expanding sequences of complete constituents into left-expanding sequences of incomplete constituents $A_\eta/A_\mu$, consisting of an instance of an active constituent $A_\eta$ lacking an instance of an awaited constituent $A_\mu$ yet to be recognized. This transform can be defined as a synchronous grammar that maps every context-free rule expansion in a source tree (in Chomsky Normal Form) to a corresponding expansion in a right-corner transformed tree:[1]

- Beginning case: the top of a right-expanding sequence in an ordinary phrase structure tree is mapped to the bottom of a left-expanding sequence in a right-corner transformed tree:



(1)

- Middle case: each subsequent branch in a right-expanding sequence of an ordinary phrase structure tree is mapped to a branch in a left-expanding sequence of the transformed tree:



(2)

- Ending case: the bottom of a right-expanding sequence in an ordinary phrase structure tree

[1] Here, $\eta$ and $\mu$ are tree node addresses, consisting of sequences of zeros, representing left branches, and ones, representing right branches, on a path from the root of the tree to any given node.

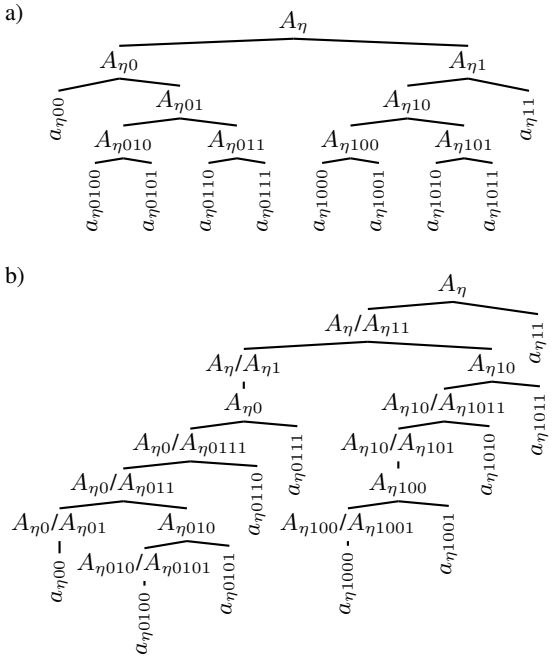
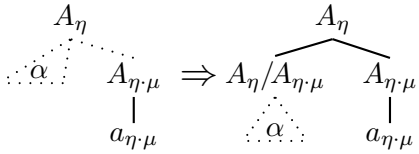



Figure 1: Sample right-corner transform of schematized tree before (a) and after (b) application of transform.

is mapped to the top of a left-expanding sequence in a right-corner transformed tree:



(3)

The application of this transform is exemplified in Figure 1.

### 3.2 Hierarchical Hidden Markov Models

Right-corner transformed trees are mapped to random variables in a Hierarchical Hidden Markov Model (Murphy and Paskin, 2001).

A Hierarchical Hidden Markov Model (HHMM) is essentially a factored version of a Hidden Markov Model (HMM), configured to recognize bounded recursive structures (i.e. trees). Like HMMs, HHMMs use Viterbi decoding to obtain sequences of hidden states $\hat{s}_{1..T}$ given sequences of observations $o_{1..T}$ (words or audio features), through independence assumptions in a transition model $\Theta_{\mathrm{A}}$ and an observation model $\Theta_{\mathrm{B}}$ (Baker, 1975; Jelinek et al., 1975):

$$\hat{s}_{1..T} \stackrel{\text{def}}{=} \underset{s_{1..T}}{\operatorname{argmax}} \prod_{t=1}^{T} \mathsf{P}_{\Theta_{\mathrm{A}}}(s_t \mid s_{t-1}) \cdot \mathsf{P}_{\Theta_{\mathrm{B}}}(o_t \mid s_t) \tag{4}$$

HHMMs then factor the hidden state transition $\Theta_{\mathrm{A}}$ into a reduce and shift phase (Equation 5), then into a bounded set of depth-specific operations (Equation 6):

$$\mathsf{P}_{\Theta_{\mathrm{A}}}(s_t|s_{t-1}) = \sum_{r_t} \mathsf{P}_{\Theta_{\mathrm{R}}}(r_t|s_{t-1}) \cdot \mathsf{P}_{\Theta_{\mathrm{S}}}(s_t|r_t\, s_{t-1}) \tag{5}$$

$$\stackrel{\text{def}}{=} \sum_{r_t^{1..D}} \prod_{d=1}^{D} \mathsf{P}_{\Theta_{\mathrm{R},d}}(r_t^d|\, r_t^{d+1} s_{t-1}^d s_{t-1}^{d-1}) \cdot \mathsf{P}_{\Theta_{\mathrm{S},d}}(s_t^d|r_t^{d+1} r_t^d\ s_{t-1}^d s_t^{d-1}) \tag{6}$$

which allow depth-specific variables to reduce (through $\Theta_{\text{R-Rdn},d}$), transition ($\Theta_{\text{S-Trn},d}$), and expand ($\Theta_{\text{S-Exp},d}$) like tape symbols in a pushdown automaton with a bounded memory store, depending on whether the variable below has reduced ($r_t^d \in R_G$) or not ($r_t^d \notin R_G$):[2]

$$\mathsf{P}_{\Theta_{\mathrm{R},d}}(r_t^d \mid r_t^{d+1} s_{t-1}^d s_{t-1}^{d-1}) \stackrel{\text{def}}{=} \begin{cases} \text{if } r_t^{d+1} \notin R_G : [\![ r_t^d = \mathbf{r}_\top ]\!] \\ \text{if } r_t^{d+1} \in R_G : \mathsf{P}_{\Theta_{\text{R-Rdn},d}}(r_t^d \mid r_t^{d+1}\ s_{t-1}^d\ s_{t-1}^{d-1}) \end{cases} \tag{7}$$

$$\mathsf{P}_{\Theta_{\mathrm{S},d}}(s_t^d \mid r_t^{d+1} r_t^d\ s_{t-1}^d s_t^{d-1}) \stackrel{\text{def}}{=} \begin{cases} \text{if } r_t^{d+1} \notin R_G,\ r_t^d \notin R_G : [\![ s_t^d = s_{t-1}^d ]\!] \\ \text{if } r_t^{d+1} \in R_G,\ r_t^d \notin R_G : \mathsf{P}_{\Theta_{\text{S-Trn},d}}(s_t^d \mid r_t^{d+1} r_t^d\ s_{t-1}^d s_t^{d-1}) \\ \text{if } r_t^{d+1} \in R_G,\ r_t^d \in R_G : \mathsf{P}_{\Theta_{\text{S-Exp},d}}(s_t^d \mid s_t^{d-1}) \end{cases} \tag{8}$$

where $s_t^0 = \mathbf{s}_\top$ and $r_t^{D+1} = \mathbf{r}_\bot$ for constants $\mathbf{s}_\top$ (an incomplete root constituent), $\mathbf{r}_\bot$ (a complete lexical constituent) and $\mathbf{r}_\top$ (a null state resulting from reduction failure) s.t. $\mathbf{r}_\bot \in R_G$ and $\mathbf{r}_\top \notin R_G$.

Right-corner transformed trees, as exemplified in Figure 1(b), can then be aligned to HHMM states as shown in Figure 2, and used to train an HHMM as a parser.

Parsing with an HHMM simply involves processing the input sequence, and estimating a most likely hidden state sequence given this observed input. Since the output is to be the best possible parse, the Viterbi algorithm is used, which keeps track of the highest probability state at each time step, where the state is the store of incomplete syntactic constituents being processed. State transitions are computed using the models above, and each state at each time step keeps a back pointer to the state it most probably came from. Extracting the highest probability parse requires extracting

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

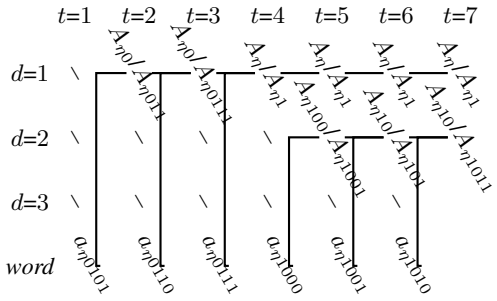



Figure 2: Mapping of schematized right-corner tree into HHMM memory elements.

the most likely sequence, deterministically mapping that sequence back to a right-corner tree, and reversing the right-corner transform to produce an ordinary phrase structure tree.

Unfortunately exact inference is not tractable with this model and dataset. The state space is too large to manage for both space and time reasons, and thus approximate inference is carried out, through the use of a beam search. At each time step, only the top $N$ most probable hypothesized states are maintained. Experiments described in (Schuler, 2009) suggest that there does not seem to be much lost in going from exact inference using the CKY algorithm to a beam search with a relatively large width. However, the opposite experiment, examining the effect of going from a relatively wide beam to a very narrow beam has not been thoroughly studied in this parsing architecture.

## 4 Optionally Arc-eager Parsing

The right-corner transform described in Section 3.1 saves memory because it transforms any right-expanding sequence with left-child subtrees into a left-expanding sequence of incomplete constituents, with the same sequence of subtrees as right children. The left-branching sequences of siblings resulting from this transform can then be composed bottom-up through time by replacing each left child category with the category of the resulting parent, within the same memory element (or depth level). For example, in Figure 3(a) a left-child category NP/NP at time $t$=4 is composed with a noun *new* of category NP/NNP (a noun phrase lacking a proper noun yet to come), resulting in a new parent category NP/NNP at time $t$=5 replacing the left child category NP/NP in the topmost $d$=1 memory element.

This in-element composition preserves elements of the bounded memory store for use in processing descendants of this composed constituent, yielding the human-like memory demands reported in (Schuler et al., 2008). But whenever an in-element composition like this is hypothesized, it isolates an intermediate constituent (in this example, the noun phrase 'new york city') from subsequent composition. Allowing access to this intermediate constituent — for example, to allow 'new york city' to become a modifier of 'bonds', which itself becomes an argument of 'for' — requires an analysis in which the intermediate constituent is stored in a separate memory element, shown in Figure 3(b). This creates a local ambiguity in the parser (in this case, from time step $t$=4) that may have to be propagated across several words before it can be resolved (in this case, at time step $t$=7). This is essentially an ambiguity between arc-eager (in-element) and arc-standard (cross-element) composition strategies, as described by Abney and Johnson (1991). In contrast, an ordinary (purely arc-standard) parser with an unbounded stack would only hypothesize analysis (b), avoiding this ambiguity.[3]

The right-corner HHMM approach described in this paper relies on a learned statistical model to predict when in-element (arc-eager) compositions will occur, in addition to hypothesizing parse trees. The model encodes a mixed strategy: with some probability arc-eager or arc-standard for each possible expansion. Accuracy results on a right-corner HHMM model trained on the Penn Wall Street Journal Treebank suggest that this kind of optionally arc-eager strategy can be reliably statistically learned.

By placing firm limits on the number of open incomplete constituents in working memory, the Hierarchical HMM parser maintains parallel hypotheses on the beam which predict whether each constituent will host a subsequent attachment or not. Empirical results described in the next section

[3]It is important to note that neither the right-corner nor left-corner parsing strategy by itself creates this ambiguity. The ambiguity arises from the decision to use this optionally arc-eager strategy to reduce memory store allocation in a bounded memory parser. Implementations of left-corner parsers such as that of Henderson (2004) adopt a arc-standard strategy, essentially always choosing analysis (b) above, and thus do not introduce this kind of local ambiguity. But in adopting this strategy, such parsers must maintain a stack memory of unbounded size, and thus are not attractive as models of human parsing in short-term memory (Resnik, 1992).

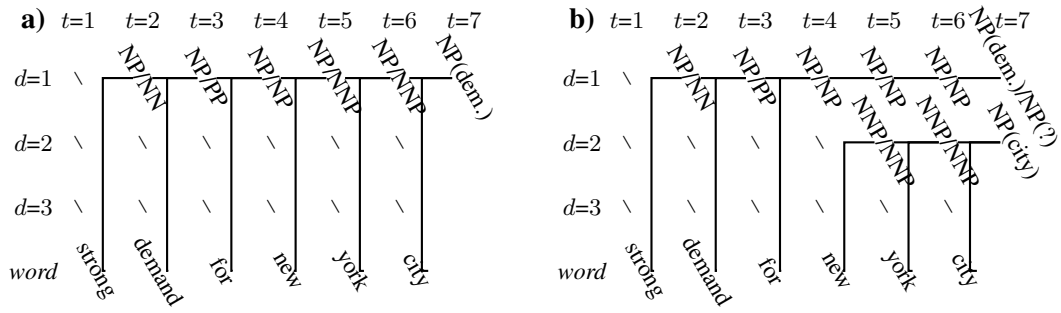



Figure 3: Alternative analyses of 'strong demand for new york city ...': a) using in-element composition, compatible with 'strong demand for new york city is ...' (in which the demand is for the city); and b) using cross-element (or delayed) composition, compatible with either 'strong demand for new york city is ...' (in which the demand is for the city) or 'strong demand for new york city bonds is ...' (in which a forthcoming referent — in this case, bonds — is associated with the city, and is in demand). In-element composition (a) saves memory but closes off access to the noun phrase headed by 'city', and so is not incompatible with the '...bonds' completion. Cross-element composition (b) requires more memory, but allows access to the noun phrase headed by 'city', so is compatible with either completion. This ambiguity is introduced at $t$=4 and propagated until at least $t$=7. An ordinary, non-right-corner stack machine would exclusively use analysis (b), avoiding ambiguity.

show that this added demand on parallelism does not substantially degrade parsing accuracy, even at very narrow beam widths.

## 5 Experimental Evaluation

The parsing model described in Section 3 has previously been evaluated on the standard task of parsing the Wall Street Journal section of the Penn Treebank. This evaluation was optimized for accuracy results, and reported a relatively wide beam width of 2000 to achieve its best results. However, most psycholinguistic models of the human sentence processing mechanism suggest that if the HSPM does work in parallel, it does so with a much lower number of concurrent hypotheses (Boston et al., 2008b). Viewing the HHMM parsing framework as a psycholinguistic model, a necessary (though not sufficient) condition for it being a valid model is that it be able to maintain relatively accurate parsing capabilities even at much lower beam widths.

Thus, the first experiments in this paper evaluate the degradation of parsing accuracy depending on beam width of the HHMM parser. Experiments were conducted again on the WSJ Penn Treebank, using sections 02-21 to train, and section 23 as the test set. Punctuation was included in both training and testing. A set of varied beam widths were considered, from a high of 2000 to a low of 15. This range was meant to roughly correspond to the range of parallelism used in other similar experiments, using 2000 as a high end due to its usage in previous parsing experiments. However, it should be noted that in fact the highest value of 2000 is already an approximate search – preliminary experiments showed that exhaustive search with the HHMM would require more than 100000 elements per time step (exact values may be much higher but could not be collected because they exhausted system memory).

The HHMM parser was compared to a custom built (though standard) probabilistic CKY parser implementation trained on the CNF trees used as input to the right-corner transform, so that the CKY parser was able to compete on a fair footing. The accuracy results of these experiments are shown in Figure 4.

These results show fairly graceful decline in parsing accuracy with a beam width starting at 2000 elements down to about 50 beam elements. This beam width is much less than $1\%$ of the exhaustive search, though it is around $1\%$ of what might be considered the highest reasonable beam width for efficient parsing. The lowest beam widths attempted, 15, 20, and 25, result in accuracy below that of the CKY parser. The lowest beam width attempted, 15, shows the sharpest decline in accuracy, putting the HHMM system nearly 8 points below the CKY parser in terms of accuracy.

This compares reasonably well to results by

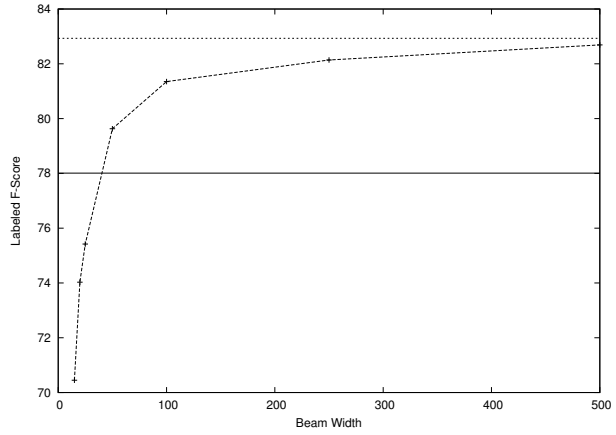



Figure 4: Plot of parsing accuracy (labeled F-score) vs. beam widths for an HHMM parser (curved line). Top line is HHMM accuracy with beam width of 2000 (upper bound). The bottom line is CKY parser results. Points correspond to beam widths of 15, 20, 25, 50, 100, 250, and 500.

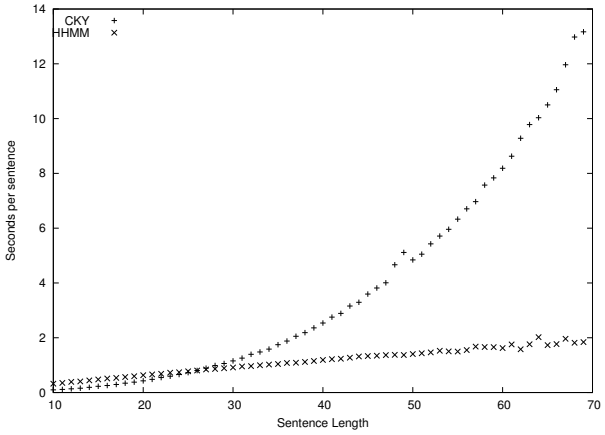



Figure 5: Plot of parsing time vs. sentence length for HHMM and CKY parsers.

Brants and Crocker (2000) showing that an incremental chart-parsing algorithm can parse accurately with pruning down to $1\%$ of normal memory usage. While that parsing algorithm is difficult to compare directly to this HHMM parser, the reduction in beam width in this system to 50 beam elements from an already approximated 2000 beam elements shows similar robustness to approximation. Accuracy comparisons should be taken with a grain of salt due to additional annotations performed to the Treebank before training, but the HHMM parser with a beam width of 50 obtains approximately the same accuracy as the Brants and Crocker incremental CKY parser pruning to 3% of chart size. At $1\%$ pruning, Brants and Crocker achieved around 75% accuracy, which falls between the HHMM parser at beam widths of 20 and 25.

Results by Boston et al. (2008b) are also difficult to compare directly due to a difference in parsing algorithm and different research priority (that paper was attempting to correlate parsing difficulty with reading difficulty). However, that paper showed that a dependency parser using less than ten beam elements (and as few as one) was just as capable of predicting reading difficulty as the parser using 100 beam elements.

A second experiment was conducted to evaluate the HHMM for its time efficiency in parsing. This experiment is intended to address two questions: Whether this framework is efficient enough to be considered a viable psycholinguistic model, and whether its parsing time and accuracy remain competitive with more standard cubic time parsing technologies at low beam widths. To evaluate this aspect, the HHMM parser was run at low beam widths on sentences of varying lengths. The baseline was the widely-used Stanford parser (Klein and Manning, 2003), run in 'vanilla PCFG' mode. This parser was used rather than the custom-built CKY parser from the previous experiment, to avoid the possibility that its implementation was not efficient enough to provide a realistic test. The HHMM parser was implemented as described in the previous section. These experiments were run on a machine with a single 2.40 GHz Celeron CPU, with 512 MB of RAM. In both implementations the parser timing includes only time spent actually parsing sentences, ignoring the overhead incurred by reading in model files or training.

Figure 5 shows a plot of parsing time versus sentence length for the HHMM parser for a beam width of 20. Sentences shorter than 10 words were not included for visual clarity (both parsers are extremely fast at that length). At this beam width, the performance of the HHMM parser (labeled F-score) was $74.03\%$, compared to $71\%$ for a plain CKY parser. As expected, the HHMM parsing time increases linearly with sentence length, while the CKY parsing time increases super-linearly. (However, due to high constants in the run time complexity of the HHMM, it was not a priori clear that the HHMM would be faster for any sentence of reasonable length.)

The results of this experiment show that the HHMM parser is indeed competitive with a probabilistic CKY parser, in terms of parsing efficiency, even while parsing with higher accuracy. At sentences longer that 26 words (including punctuation), the HHMM parser is faster than the CKY parser. This advantage is clear for segmented text such as the Wall Street Journal corpus. However, this advantage is compounded when considering unsegmented or ambiguously segmented text such as transcribed speech or less formal written text, as the HHMM parser can also make decisions about where to put sentence breaks, and do so in linear time.[4]

## 6 Conclusion and Future Work

This paper furthers the case for the HHMM as a viable psycholinguistic model of the human parsing mechanism by showing that performance degrades gracefully as parallelism decreases, providing reasonably accurate parsing even at very low beam widths. In addition, this work shows that an HHMM parser run at low beam widths is competitive in speed with parsers that don't work incrementally, because of its asymptotically linear runtime.

This is especially surprising given that the HHMM uses parallel hypotheses on the beam to predict whether constituents will remain open for attachment or not. Success at low beam widths suggests that this optionally arc-eager prediction is something that is indeed relatively predictable during parsing, lending credence to claims of psycholinguistic relevance of HHMM parsing.

Future work should explore further directions in improving parsing performance at low beam widths. The lowest beam value experiments presented here generally parsed fairly accurately when they completed, but were already encountering problems with unparseable sentences that negatively affected parser accuracy. The large accuracy decrease between beam sizes of 20 and 15 is likely to be mostly due to the lack of any correct analysis on the beam when the sentence is completed.

It should be noted, however, that no adjustments were made to the parser's syntactic model with these beam variations. This syntactic model was optimized for accuracy at the standard beam width of 2000, and thus contains some state splittings that are beneficial at wide beam widths, but at low beam widths are redundant and prevent otherwise valid hypotheses from being maintained on the beam. For applications in which speed is a priority, future research can evaluate tradeoffs in accuracy that occur at different beam widths with a coarser-grained syntactic representation that allows for more variation of hypotheses even on very small beams.

[4] It does this probabilistically as a side effect of the parsing, by choosing an analysis in which $r_t^0 \in R_G$ (for any $t$).

## Acknowledgments

This research was supported by National Science Foundation CAREER/PECASE award 0447685. The views expressed are not necessarily endorsed by the sponsors.

## References

Steven P. Abney and Mark Johnson. 1991. Memory requirements and local ambiguities of parsing strategies. *J. Psycholinguistic Research*, 20(3):233–250.

James Baker. 1975. The Dragon system: an overivew. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 23(1):24–29.

Thomas G. Bever. 1970. The cognitive basis for linguistic structure. In J.R̃. Hayes, editor, *Cognition and the Development of Language*, pages 279–362. Wiley, New York.

Marisa Ferrara Boston, John T. Hale, Reinhold Kliegl, Umesh Patil, and Shravan Vasishth. 2008a. Parsing costs as predictors of reading difficulty: An evaluation using the Potsdam Sentence Corpus. *Journal of Eye Movement Research*, 2(1):1–12.

Marisa Ferrara Boston, John T. Hale, Reinhold Kliegl, and Shravan Vasishth. 2008b. Surprising parser actions and reading difficulty. In *Proceedings of ACL-08: HLT, Short Papers*, pages 5–8, Columbus, Ohio, June. Association for Computational Linguistics.

Thorsten Brants and Matthew Crocker. 2000. Probabilistic parsing and psychological plausibility. In *Proceedings of COLING '00*, pages 111–118.

Noam Chomsky and George A. Miller. 1963. Introduction to the formal analysis of natural languages. In *Handbook of Mathematical Psychology*, pages 269–321. Wiley.

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

Matthew Crocker and Thorsten Brants. 2000. Wide-coverage probabilistic sentence processing. *Journal of Psycholinguistic Research*, 29(6):647–669.

John Hale. 2001. A probabilistic earley parser as a psycholinguistic model. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 159–166, Pittsburgh, PA.

James Henderson. 2004. Lookahead in deterministic left-corner parsing. In *Proc. Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 26–33, Barcelona, Spain.

Frederick Jelinek, Lalit R. Bahl, and Robert L. Mercer. 1975. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 21:250–256.

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

Richard L. Lewis. 2000. Falsifying serial and parallel parsing models: Empirical conundrums and an overlooked paradigm. *Journal of Psycholinguistic Research*, 29:241–248.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

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

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

Philip Resnik. 1992. Left-corner parsing and psychological plausibility. In *Proceedings of COLING*, pages 191–197, Nantes, France.

Brian Roark, Asaf Bachrach, Carlos Cardenas, and Christophe Pallier. 2009. Deriving lexical and syntactic expectation-based measures for psycholinguistic modeling via incremental top-down parsing. *Proceedings of the 2009 Conference on Empirical Methods in Natural Langauge Processing*, pages 324–333.

William Schuler, Samir AbdelRahman, Tim Miller, and Lane Schwartz. 2008. Toward a psycholinguistically-motivated model of language. In *Proceedings of COLING*, pages 785–792, Manchester, UK, August.

William Schuler, Samir AbdelRahman, Tim Miller, and Lane Schwartz. 2010. Broad-coverage incremental parsing using human-like memory constraints. *Computational Linguistics*, 36(1).

William Schuler. 2009. Parsing with a bounded stack using a model-based right-corner transform. In *Proceedings of NAACL*, pages 344–352, Boulder, Colorado.

Stephen Wu, Asaf Bachrach, Carlos Cardenas, and William Schuler. 2010. Complexity metrics in an incremental right-corner parser. In *Proceedings of the 49th Annual Conference of the Association for Computational Linguistics*.