After each Gibbs iteration, we sample each s_c directly using binomial-Beta conjugacy. We re-sample the DP concentration parameters α_c with the auxiliary variable procedure of West (1995).

Decoding We compute the rule score of each tree fragment from a single grammar sample as follows:

$$\theta_{c,e} = \frac{n_{c,e}(\boldsymbol{z}) + \alpha_c P_0(e|c)}{n_{c,\cdot}(\boldsymbol{z}) + \alpha_c} \tag{5}$$

To make the grammar more robust, we also include all CFG rules in P_0 with zero counts in $\boldsymbol{n}$. Scores for these rules follow from (5) with $n_{c,e}(\boldsymbol{z}) = 0$.

For decoding, we note that the derivations of a TSG are a CFG parse forest (Vijay-Shanker and Weir, 1993). As such, we can use a Synchronous Context Free Grammar (SCFG) to translate the 1-best parse to its derivation. Consider a unique tree fragment e_i rooted at X with frontier γ, which is a sequence of terminals and non-terminals. We encode this fragment as an SCFG rule of the form

$$[X \rightarrow \gamma\ ,\ X \rightarrow i, Y_1, \ldots, Y_n] \tag{6}$$

where $Y_1, \ldots, Y_n$ is the sequence of non-terminal nodes in γ.[8] During decoding, the input is rewritten as a sequence of tree fragment (rule) indices $\{i, j, k, \ldots\}$. Because the TSG substitution operator always applies to the leftmost frontier node, we can deterministically recover the monolingual parse with top-down re-writes of $\diamondsuit$.

The SCFG formulation has a practical benefit: we can take advantage of the heavily-optimized SCFG decoders for machine translation. We use `cdec` (Dyer et al., 2010) to recover the Viterbi derivation under a DP-TSG grammar sample.

5 Experiments

5.1 Standard Parsing Experiments

We evaluate parsing accuracy of the Stanford and DP-TSG models (Table 6). For comparison, we also include the Berkeley parser (Petrov et al., 2006).[9] For the DP-TSG, we initialized all b_s with fair coin tosses and ran for 400 iterations, after which likelihood stopped improving.

	Leaf Ancestor		Evalb			
	Corpus	Sent	LP	LR	F1	EX%
PA-PCFG	0.793	0.812	68.1	67.0	67.6	10.5
DP-TSG	0.823	0.842	75.6	76.0	75.8	15.1
Stanford	0.843	0.861	77.8	79.0	78.4	17.5
Berkeley	**0.880**	**0.891**	**82.4**	**82.0**	**82.2**	**21.4**

Table 6: Standard parsing experiments (test set, sentences $\leq$ 40 words). All parsers exceed 96% tagging accuracy. Berkeley and DP-TSG results are the average of three independent runs.

We report two different parsing metrics. *Evalb* is the standard labeled precision/recall metric.[10] *Leaf Ancestor* measures the cost of transforming guess trees to the reference (Sampson and Babarczy, 2003). It was developed in response to the non-terminal/terminal ratio bias of Evalb, which penalizes flat treebanks like the FTB. The range of the score is between 0 and 1 (higher is better). We report micro-averaged (whole corpus) and macro-averaged (per sentence) scores.

In terms of parsing accuracy, the Berkeley parser exceeds both Stanford and DP-TSG. This is consistent with previous experiments for French by Seddah et al. (2009), who show that the Berkeley parser outperforms other models. It also matches the ordering for English (Cohn et al., 2010; Liang et al., 2010). However, the standard baseline for TSG models is a simple parent-annotated PCFG (PA-PCFG). For English, Liang et al. (2010) showed that a similar DP-TSG improved over PA-PCFG by 4.2% F1. For French, our gain is a more substantial 8.2% F1.

5.2 MWE Identification Experiments

Table 7 lists overall and per-category MWE identification results for the parsing models. Although DP-TSG is less accurate as a general parsing model, it is more effective at identifying MWEs.

The predominant approach to MWE identification is the combination of lexical association measures (surface statistics) with a binary classifier (Pecina, 2010). A state-of-the-art, language independent package that implements this approach for higher order n-grams is `mwetoolkit` (Ramisch et al., 2010).[11] In Table 8 we compare DP-TSG to both

[8] This formulation is due to Chris Dyer.

[9] Training settings: right binarization, no parent annotation, six split-merge cycles, and random initialization.

[10] Available at http://nlp.cs.nyu.edu/evalb/ (v.20080701).

[11] Available at http://multiword.sourceforge.net/. See §A.2 for