ELSEVIER

Cognitive Science 26 (2002) 255–268

COGNITIVE SCIENCE

http://www.elsevier.com/locate/cogsci

Discovering syntactic deep structure via Bayesian statistics

Jason Eisner[*]

Department of Computer Science, Johns Hopkins University, 3400 N. Charles Street, Baltimore, MD 21218-2691, USA

Accepted 22 March 2002

Abstract

In the Bayesian framework, a language learner should seek a grammar that explains observed data well and is also *a priori* probable. This paper proposes such a measure of prior probability. Indeed it develops a full statistical framework for lexicalized syntax. The learner's job is to discover the system of probabilistic transformations (often called lexical redundancy rules) that underlies the patterns of regular and irregular syntactic constructions listed in the lexicon. Specifically, the learner discovers what transformations apply in the language, how often they apply, and in what contexts. It considers simpler systems of transformations to be more probable *a priori*. Experiments show that the learned transformations are more effective than previous statistical models at predicting the probabilities of lexical entries, especially those for which the learner had no direct evidence. © 2002 Cognitive Science Society, Inc. All rights reserved.

Keywords: Grammar induction; Bayesian learning; Transformational grammar; Lexicalized syntax

1. Introduction

How can one discover a language's deep syntax? This brief report summarizes recent work (Eisner, 2001) that attempts to do so from examples of the language's surface syntax.

In modern lexicalized theories of syntax, each word of the language lists the constructions in which it can appear. A typical construction, used for English transitive verbs, projects a subject slot to the word's left and an object slot to its right. Syntax trees are constructed by "gluing

* Tel.: +1-410-516-8775; fax: +1-410-516-6134.
E-mail address: jason@cs.jhu.edu (J. Eisner).

together" entries from this syntactic lexicon. There is no grammar except for the lexicon (and the universal "gluing" operations).

The method sketched here seeks underlying regularities in the syntactic lexicon, such as the fact that many transitive verbs in English also list intransitive and passive constructions. It explains the observed regularities by positing optional rules, such as intransitivization and passivization, that *transform* lexical entries—just as early ideas of transformational grammar posited rules to transform complete sentences. Armed with these rules of (lexicalized) deep syntax, the method can then predict new lexical entries that it has not actually observed.

While this framework is familiar in linguistic circles, it is recast here in statistical terms to allow learning from data. Probabilities are attached to the various common and uncommon constructions in the lexicon and also to the transformational rules, which are assumed to apply to entries randomly but at rates that may vary with the rule and the entry. Lexical exceptions and even families of exceptions can be seamlessly described and in principle learned within the probabilistic framework. This statistical perspective on what is learned (and why) may be helpful to linguists, psycholinguists, and engineers alike.

2. Language learning in a Bayesian framework

Any kind of language learning is a generalization problem. A learner must extrapolate an infinite language such as English—a set of grammatical sentences or syntax trees—from the finite, random subset of English that he/she/it happens to have observed. That is, the learner tries to explain the observed data as a sample from the output of an underlying generative grammar.

Why does the learner posit a grammar that generates more than just the observations? A grammar that generates exactly the observations may not be available within the learner's hypothesis space, sometimes called Universal Grammar. Or, even if it is, Occam's Razor or memory limitations may still drive the learner to prefer a more permissive grammar that is "simpler" by some measure.

To name this measure of simplicity, we may say that the learner favors grammars $\boldsymbol{\theta}$ with high *a priori* probability $p(\boldsymbol{\theta})$. The prior probability function $p(\boldsymbol{\theta})$ is conceptually a "softer" version of Universal Grammar, with $p(\boldsymbol{\theta}) > 0$ just when $\boldsymbol{\theta}$ is allowed by Universal Grammar at all.

Of course a learner also prefers grammars that account well for the observed data D. An optimal learner will combine these two preferences according to Bayes' theorem, which (for fixed D) is equivalent to seeking $\boldsymbol{\theta}$ that maximizes the product

$$p(\boldsymbol{\theta}) \cdot p(D|\boldsymbol{\theta}) \tag{1}$$

The likelihood $p(D|\boldsymbol{\theta})$ expresses the conditional probability that if the grammar really were $\boldsymbol{\theta}$, the learner would indeed have observed D. Both factors are minuscule, as there are very many possible grammars and very many possible samples from each grammar.

The goal of this paper is to improve learning under Eq. (1) by designing a sensible prior distribution $p(\boldsymbol{\theta})$ that prefers *linguistically* simpler grammars—those with strong, consistent internal structure and few lexically specific exceptions. This preference can be overwhelmed by

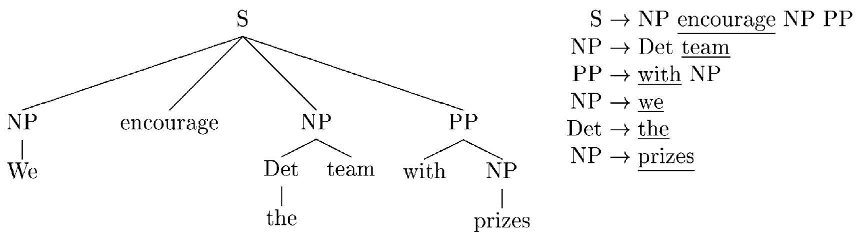


Fig. 1. Level-2 structure: a six-word sentence and the six lexical entries that combined to generate it.

sufficient evidence: where observed data D are plentiful, the $p(D|\boldsymbol{\theta})$ term will strongly favor grammars that fit those data closely,[1] even if such grammars are complex and exception-ridden. But where the evidence is uncertain, the prior $p(\boldsymbol{\theta})$ will guide the learner to favor rules rather than exceptions.

The paper's intuition comes from an old notion of the size (cost) of a lexicon. In English, nearly all transitive verbs can passivize. A grammar $\boldsymbol{\theta}$ that encodes this generalization can *derive* passive verb forms by transformation from active ones, so it has a smaller (cheaper) lexicon than a grammar $\boldsymbol{\theta}'$ that explicitly *lists* passive forms in addition to active ones. We will ensure that $p(\boldsymbol{\theta}) > p(\boldsymbol{\theta}')$, encoding a preference for small explanatory grammars over large stipulative ones.

For the expression $p(D|\boldsymbol{\theta})$ to be meaningful, the grammar $\boldsymbol{\theta}$ must assign probabilities to the numerous grammatical forms that it generates. ($p(D|\boldsymbol{\theta})$ is large if $\boldsymbol{\theta}$ assigns high probabilities to forms observed frequently in D.) That is, $\boldsymbol{\theta}$ must be a *probabilistic* generative grammar. A reasonable starting point is a probabilistic context-free grammar (PCFG).

Indeed a number of previous researchers have attempted to learn PCFGs in this Bayesian framework. However, they have used rather simple priors in the Minimum Description Length tradition, typically defining $p(\boldsymbol{\theta})$ to favor grammars whose context-free rules are few, short, nearly equiprobable, and defined over a small set of nonterminals. But in the lexicalized case (Fig. 1), we are supposing that $\boldsymbol{\theta}$ is a large lexicon listing many context-free rules for each word. (Such a lexicon allows word-specific exceptions.) This paper's version of $p(\boldsymbol{\theta})$ will gladly allow a great multitude of rules of varying length and probability, so long as they are largely predictable from one another by linguistic transformation. It favors grammars whose probabilistic rules are interrelated by a comparatively small number of transformational principles.

3. The explanatory hierarchy

Lexicalized theories of syntax include categorial grammar, LFG, LTAG, HPSG, link grammar, and minimalism. Each can be regarded as describing a language on four levels:

1. At the simplest level, a language is merely a set of strings.
2. To explain regularities in the above set, it is taken to be generated (under universal combination operations) from a more concise and perhaps finite lexicon of language-specific syntactic substructures.

3. Regularities in the above lexicon are explained, in turn, via language-specific transformations that relate its entries. These are commonly called lexical redundancy rules or metarules.
4. Finally, a language's system of transformations must be organized in a way that is acceptable, and preferably likely, under Universal Grammar.

Let us call this the explanatory hierarchy. We may regard levels 2 and 3 as the lexicalized versions of Chomsky's surface and deep structure. Level 4 corresponds roughly to Chomsky's "principles and parameters," or more closely to Becker's (1994) "patterns in metarules."

Such lexicalized theories include irregular and regular forms alike in the level-2 lexicon, deferring the account of regularities to higher levels (Aronoff, 1976; Bresnan, 1978). This uniform treatment has become popular for several reasons, both linguistic and computational (Eisner, 2001, Section 2.2). This paper emphasizes the following advantages:

- While the original role of a lexicon was to list only unpredictable information, nearly every construction in a probabilistic grammar is unpredictable. For example, virtually all English verbs passivize, but they do so at different rates. Thus, every lexical entry—even if *syntactically* regular—must list a probability in the lexicon that may be to some extent idiosyncratic.[2] Lexically specific subcategorization probabilities (Charniak, 1997) have been used by several recent statistical parsers.
- There is no need for a strict distinction between regular and irregular entries (Flickinger, 1987). Level-3 transformations can be used to account for the whole spectrum:
  - ◦ `PP`-adjunction (*We encourage the team* $\rightarrow$ *We encourage the team with prizes*) is fairly regular, though it applies to different verbs and nouns at somewhat different rates.
  - ◦ Unaccusative movement in English (*They sank the boat* $\rightarrow$ *The boat sank*) is subregular, or systematically irregular, in that it applies to only a small set of verbs.[3]
  - ◦ A transformation that applies to only a *single* word can be crafted in order to produce an exceptional entry in the lexicon. However, since all transformations and narrowly targeted ones in particular will increase the cost of the grammar (reduce $p(\boldsymbol{\theta})$), it behooves the learner to account for the data with as few of them as possible.

For homogeneity, we will presume that learners regard lexically specific transformations (such as unaccusative movement) as transformations that apply at high rates to some words and very low rates to others, just as `PP`-adjunction applies at lexically specific rates. Thus, an English speaker will treat unaccusative movement of a novel verb as a possibility that is relatively unlikely but still worth considering—whereas for more familiar verbs such as `sink` or `eat`, speakers have apparently learned that the rate of unaccusative movement is manyfold higher or lower than this "typical" rate.[4]

Notice that successive levels of the explanatory hierarchy result in more detailed *analyses* of sentences. Level 1 treats sentences as unstructured strings. But level 2 describes each string as the result of at least one tree-like derivation, from which it is possible to infer a surface semantics: e.g., that a particular `NP` is the subject of `taken`. Level 3 allows a deeper semantics by identifying the subject of (passive) `taken` with the object of (active) `take`. Finally, our weak treatment of level 4 will relate the transformations themselves, explaining right-adjunction of a PP argument to take as reflecting more general tendencies toward right-adjunction, PP-adjunction, and verbal modification in the language.

Children learn from unadorned level-1 material—namely speech (perhaps paired with non-linguistic observations). But the method to be described here supposes that the learner can already segment or parse some of this speech, so that the training samples in D are level-2 material—observed lexical entries. The learner's job is to extract higher-level generalizations from D, allowing the learner to better estimate the true probabilities of all lexical entries whether or not they appear in D. Presumably, this helps the learner parse sentences that contain novel but plausible lexical entries (Pinker, 1989) —verifying these entries' existence and feeding them back into D for further training.

Besides any light it may shed on possible human strategies, there is an engineering motivation for learning generalizations from surface syntax. Statistical parsers for English are often trained from the same level-2 trees used in the present experiments (the Penn Treebank). To parse novel sentences correctly, such parsers must frequently hypothesize probabilities for lexical entries that never appeared in these training trees. The present work hypothesizes more accurately by exploiting the linguistic insight that generalizations are transformational.

4. A concrete framework

For concreteness, the rest of this paper will commit to a particularly simple syntactic formalism, a lexicalized form of context-free grammar. While the statistical and algorithmic techniques would apply to other lexicalized formalisms,[5] it is necessary to select one in order to run actual experiments.

It is easiest to sketch the framework by example. A level-2 syntax tree and the lexical entries that produce it are shown in Fig. 1. For instance, with's lexical entry $\mathtt{PP} \rightarrow \underline{\mathtt{with}}\ \mathtt{NP}$ allows with to subcategorize for a following NP to produce a PP. The lexical entries may be regarded as context-free rules and the lexicon as a context-free grammar, from which the tree in Fig. 1 is generated in the usual recursive way. We may assume a standard probability model here (Charniak, 1997): roughly speaking, a tree is probable if its lexical entries are probable and hence likely to be chosen during generation.

Importantly, each lexical entry in Fig. 1 specifies a full argument structure for a word ($\mathtt{S} \rightarrow \mathtt{NP}\ \underline{\mathtt{encourage}}\ \mathtt{NP}\ \mathtt{PP}$), including arguments such as PP that are traditionally regarded as adjuncts.[6] Hence the level-3 grammar can include transformations that rearrange this full structure, such as heavy-shift, passivization, and unaccusative movement. Moreover, the lexicon can specify exceptions at this level of full argument structure: it need not derive $\mathtt{S} \rightarrow \mathtt{NP}\ \underline{\mathtt{put}}\ \mathtt{PP}$ from $^{*}\mathtt{S} \rightarrow \mathtt{NP}\ \underline{\mathtt{put}}$, or the (disproportionate) probability of $\mathtt{S} \rightarrow \mathtt{NP}\ \underline{\mathtt{went}}\ \mathtt{PP}$ from that of $\mathtt{S} \rightarrow \mathtt{NP}\ \underline{\mathtt{went}}$. Of course, such exceptional entries result in a more costly grammar, since by definition they cannot be fully predicted by the usual level-3 transformations. But it is demonstrably beneficial to represent them when they are well observed during training (Johnson, 1998; Eisner, 2001, Section 6.7.1).

For a parser (human or machine) to find a sentence's most probable tree, it must be able to compare probabilities of different lexical entries.

Table 1
Complete counts in training data of the S → · · · frames projected by six sample words

	encourage	question	fund	merge	repay	remove
S → To —[a] NP	1	1	5	1	3	2
S → To — NP PP	1	1	2	2	1	1
S → To AdvP — NP						1
S → To AdvP — NP PP						1
S → NP — NP.		2				
S → NP — NP PP.	1					
S → NP Md — NP	1					
S → NP Md — NP PP-TMP					1	
S → NP Md — PP PP						1
S → To — PP				1		
S → To — S	1					
S → NP — SBAR.		2				

[a] The symbol "—" stands for the position of the head word, such as encourage.

However, these probabilities must be learned from very sparse data. Table 1 shows six inflected verbs with all the sentential lexical entries they headed in the experimental training data.[7] For example, the second row says the entry S → To encourage NP PP was used once in the hand-constructed training parses, while S → To fund NP PP was used twice, and so forth.

We wish to replace the observed counts in Table 1 with estimated probabilities that can be used by a parser. The probabilities should not be strictly proportional to the luck-of-the-draw counts: that would treat unobserved entries (count = 0) as impossible (probability = 0). Rather, the poverty of the stimulus calls for a learner to "smooth" these sparse counts, treating them as samples from a more plausible, broader-based probability distribution.

Fortunately, there are clear structural relationships that allow generalization among the rows of Table 1. They boil down to the observation that a word's entries tend to have small string edit distance from one another (a finding that holds in the training data at large). That is, entries in the same column can be transformed into one another by a short sequence of so-called edit operations:

- **Insert** a single nonterminal into the RHS (right-hand side)
- **Delete** a single nonterminal from the RHS
- **Substitute** a single nonterminal for another in the RHS
- **Swap** two adjacent elements in the RHS

For example, rows 1 and 2 differ by a single PP. One might explain their numeric correlation by positing that all the counts in both rows started out in row 1, but a transformation inserted PP about one-third of the time. The same PP-insertion transformation relates many other rows as well. More generally, almost every row is at edit distance 1 from some other row, and all have small edit distance from rows 1–2.

Even the entries that might reflect actual lexical idiosyncrasies—the ability of merge to lose its object (row 10: *to merge with the hospital*), the affinity of remove for adverbial modification (row 3: *to completely/properly/surgically remove the tumor*)—are plausibly the result of simple edit transformations such as `NP`-deletion and `AdvP`-insertion that have non-negligible probability in English. In other words, these transformations just apply to `merge` and `remove` at a multiple of their "typical" rates. Such cases are typical; so we will adopt a prior under which it is cheapest for a grammar to list idiosyncratic entries that are already plausible on transformational grounds.

As Carpenter (1991) remarks, these simple edit transformations have been repeatedly invoked in lexicalized theories of syntax. They correspond to natural operations on argument structure: adding or suppressing semantic roles, changing their syntactic type, or permuting their positions as in heavy-shift.

Let us therefore take these edit operations to be the basic transformations allowed on lexical entries. It is the job of the learner to discover which of these universal transformations apply in the language, how often they apply, and in what contexts. For example, Table 1 suggests that in English $\texttt{S} \to \cdots$ entries, `PP`-insertion is common at the right edge, whereas insertion of `AdvP` (adverbs) or `Md` (modals) is more common just before the head word.

One could easily argue for a richer universal set of allowable transformations, such as transformations that manipulate argument structure more radically in a single step (English passivization), or change an entry's phrasal category `S` into `S/NP` (extraction) or into `NP` (nominalization). This would arm a learner with more possible generalizations to test against the data. However, the experiments of this paper are confined to sequences of simple edit transformations, as a basic test of the approach.

5. Transformation graphs

Attaching statistics to the level-2 grammar was simple: the lexicon is simply a probability distribution over possible lexical entries. Let us now develop statistical accounts of level-3 structure (this section) and level-4 structure (the next section).

Fig. 2 shows a "transformation graph." The graph's topology is determined by one's syntactic theory and is supposed to be the same for all languages. Its vertices are simply the lexical entries allowed universally by the theory. The arcs (directed edges) represent linguistically possible transformations—in this case, single edits and also arcs from START. The latter serve to generate lexical entries from thin air.

What must be learned are the arc probabilities, which specify which of the possible transformations are likely. These probabilities constitute the level-3 description of a particular language. From each lexical entry there are several arcs with total probability 1, representing possible transformations that will compete for the right to modify an instance of the entry. If the arc to the special node HALT is chosen, the instance is not modified any further.

The transformation graph merely describes a random process for generating lexical entries. One generates a lexical entry by taking a random walk from START. The steps of this walk in effect choose a starting lexical entry e' and a succession of transformations to apply to it before halting at some final entry e. A lexical entry is taken to be probable if a random walk is likely to halt at it: $p(e)$ is defined as the total probability of all paths of the form START, ... , e,

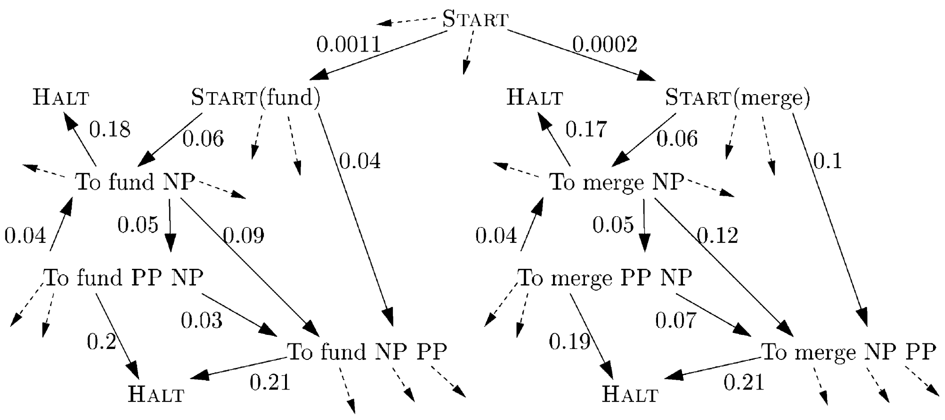


Fig. 2. A fragment of a transformation graph. "S →" is omitted at vertices to avoid visual clutter. Dashed arrows stand for other transformations not shown in this figure: in practice there are hundreds of arcs from each node (e.g., insertions of different nonterminals at different positions).

Halt. (A path's probability is the product of its arcs' probabilities, since the arcs are chosen independently.)

The level-3 transformation probabilities therefore determine the level-2 lexical entry probabilities $p(e)$. The latter can be computed by matrix inversion or (for speed) by an approximate relaxation algorithm (Eisner, 2001).[8] These probabilities can then be used for parsing.[9]

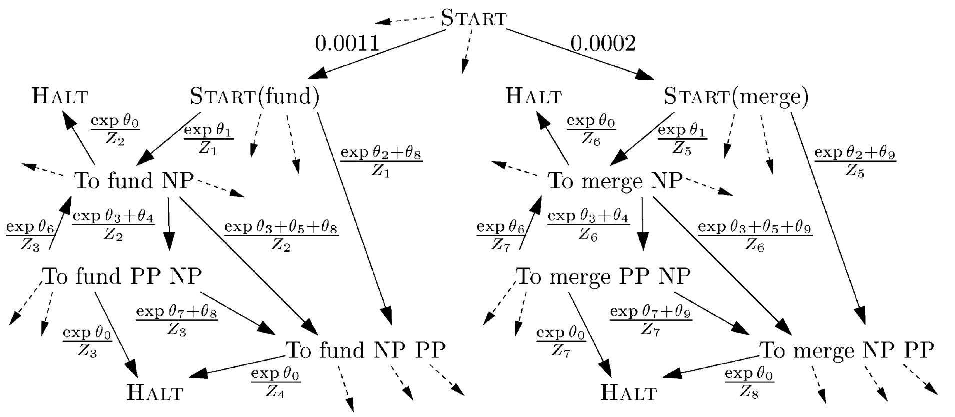
Recall that a Bayesian learner will try to maximize Eq. (1). If the observed training data D is a random sample of lexical entries, then the likelihood term $p(D|\boldsymbol{\theta})$ is proportional to the product $\prod_{e\in D} p(e)$.[10] To increase $p(D|\boldsymbol{\theta})$, the learner can increase the probabilities of arcs on paths to these observed entries. But such arcs participate in other paths as well, so this has a side effect of increasing the probabilities of *other*, related entries.

Observing e will therefore cause the learner to raise its probability estimate not only for e, but also for entries that e is likely to transform into, and perhaps for e's transformational precursors as well. These deductive and abductive generalizations (including "explaining away") are similar to those made by Bayesian networks. However, nodes in a Bayesian network represent correlated variables, rather than mutually exclusive but related values of the same variable e.

6. Transformation models

It remains to define the Bayesian prior $p(\boldsymbol{\theta})$, which pressures the learner to stick to a small and consistent set of transformations. Specifically, $p(\boldsymbol{\theta})$ should state that similar arcs in a language's transformation graph—e.g., all the PP-insertion arcs—tend to have similar probabilities. Otherwise, the learner would have no reason to posit predictive generalizations, such as a typical rate for PP-insertion in the language. It would simply adjust individual arcs' probabilities so as to maximize the probability of the training data D, leading to overfitting.

Each arc in the universal transformation graph can be characterized by its several linguistically salient features. This provides a way to identify "similar arcs." For example, all


START
0.0011
0.0002
START(fund)
HALT
$\frac{\exp\theta_0}{Z_2}$
$\frac{\exp\theta_1}{Z_1}$
$\frac{\exp\theta_2+\theta_8}{Z_1}$
To fund NP
$\frac{\exp\theta_6}{Z_3}$
$\frac{\exp\theta_3+\theta_4}{Z_2}$
$\frac{\exp\theta_3+\theta_5+\theta_8}{Z_2}$
To fund PP NP
$\frac{\exp\theta_7+\theta_8}{Z_3}$
$\frac{\exp\theta_0}{Z_3}$
To fund NP PP
HALT
$\frac{\exp\theta_0}{Z_4}$
START(merge)
HALT
$\frac{\exp\theta_0}{Z_6}$
$\frac{\exp\theta_1}{Z_5}$
$\frac{\exp\theta_2+\theta_9}{Z_5}$
To merge NP
$\frac{\exp\theta_6}{Z_7}$
$\frac{\exp\theta_3+\theta_4}{Z_6}$
$\frac{\exp\theta_3+\theta_5+\theta_9}{Z_6}$
To merge PP NP
$\frac{\exp\theta_7+\theta_9}{Z_7}$
$\frac{\exp\theta_0}{Z_7}$
To merge NP PP
HALT
$\frac{\exp\theta_0}{Z_8}$

θ_0	halts	θ_3	inserts PP	θ_6	deletes PP
θ_1	chooses To —— NP	θ_4	inserts PP before NP	θ_7	moves NP right past PP
θ_2	chooses To —— NP PP	θ_5	inserts PP before right edge	θ_8	yields To $\underline{\texttt{fund}}$ NP PP
				θ_9	yields To $\underline{\texttt{merge}}$ NP PP

Fig. 3. How the arc probabilities in Fig. 2 were determined from a vector $\boldsymbol{\theta}$ of feature weights. The Z values are chosen so that the arcs leaving each vertex have total probability 1. A formula such as $[\exp(\theta_3 + \theta_4)]/Z_2$ is best regarded as a product of e^{θ_3}, e^{θ_4}, and a normalizing factor.

PP-insertion arcs might have a certain feature in common. It is up to the learner to discover which features tend to make arcs probable or improbable in the language.

The formal solution is to define arc probabilities in terms of a smaller number of arc features, using a so-called log-linear model. Let us say that each feature i has a weight θ_i, and that each arc's probability in Fig. 2 is computed from the weights of its features *by the formulas illustrated in Fig. 3*. An arc is probable (relative to other arcs from the same entry) if its features have high total weight. Similar arcs share most of their features and so their probabilities are similar, indeed coupled.

A language is now completely determined by the values of the weights $\theta_1, \theta_2, \ldots \in \mathbb{R}$. The learner's job now is to estimate these weights, which constitute the level-4 structure of the language. The layout of nodes, arcs, and features in Fig. 3 is assumed to be universal across languages: it describes the kinds of lexical entries and transformational patterns that learners will consider.

For example, by raising θ_3, the learner can scale up the probabilities of all PP-insertion arcs, all of which include a factor of e^{θ_3}. But not all PP-insertion arcs are identical: the ones that awkwardly insert PP immediately before NP also have a factor of e^{θ_4}, so by setting $\theta_4 < 0$, the learner can model the fact that those arcs are less probable. To distinguish the probabilities of inserting PP into S $\rightarrow$ To $\underline{\texttt{fund}}$ NP and S $\rightarrow$ To $\underline{\texttt{merge}}$ NP, the learner must manipulate the only weights that differentiate those nearly identical arcs, namely the word-specific feature weights θ_8 and θ_9.

A simple and natural prior is now at hand to discourage the learner from manipulating too many weights. Let us stipulate that *a priori*, each θ_i is independently normally distributed with mean 0 and some variance σ^2. In other words, we suppose that typical languages are simple in that their exceptions are few and modest: most features tend to have weights close to zero and are therefore irrelevant. The learner can favor such languages and obtain a high value of $p(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \text{const} \cdot \exp(-\sum_i \theta_i^2/2\sigma^2)$ only by keeping $\sum_i \theta_i^2$ small. This means avoiding large weights as far as possible (Occam's Razor).

Given observed data D, a Bayesian learner simply adjusts $\boldsymbol{\theta}$ to maximize the product $p(\boldsymbol{\theta}) \cdot p(D|\boldsymbol{\theta})$. This is possible either by gradient ascent or by Expectation–Maximization.[11] Large θ_i harms $p(\boldsymbol{\theta})$, so is justified only when feature i appears on one or more arcs that can collectively explain many observations and dramatically raise $p(D|\boldsymbol{\theta})$. If one arc, this θ_i encodes an exception; if many, a generalization.

An example may help to illustrate the learning dynamic. Suppose the entry $e = \texttt{S} \rightarrow \texttt{To}\ \underline{\texttt{merge}}\ \texttt{NP PP}$ is observed more often than PP-insertion would predict. The $p(D|\boldsymbol{\theta})$ term encourages the learner to increase θ_9 (which appears just on arcs to e) until $p(e)$ matches the empirically observed probability.[12] But the $p(\boldsymbol{\theta})$ term acts as an opposing brake that tries to keep $\theta_9 \approx 0$ and thereby keep $p(e)$ to about 1 times the predicted probability. Choosing $\boldsymbol{\theta}$ to maximize $p(\boldsymbol{\theta}) \cdot p(D|\boldsymbol{\theta})$ yields a smoothed estimate of $p(e)$—a compromise between the predicted and observed probabilities. (The influence of the observed probabilities through the $p(D|\boldsymbol{\theta})$ factor depends on the number of observations: see Note 2.) Of course, it is possible that the predicted probability is too low and should itself be raised. For instance, if not only $\texttt{S} \rightarrow \texttt{To}\ \underline{\texttt{merge}}\ \texttt{NP PP}$ but also $\texttt{S} \rightarrow \texttt{To}\ \underline{\texttt{fund}}\ \texttt{NP PP}$ is more likely than the current rate of PP-insertion would predict, the prior certainly prefers increasing the single weight θ_3 to increasing both θ_8 and θ_9; so a single generalization about the rate of PP-insertion is preferred to multiple exceptions if the likelihood $p(D|\boldsymbol{\theta})$ is the same either way.

Under our prior, is the cost of a grammar $\boldsymbol{\theta}$ related to the number of entries in the lexicon? No: the grammar always includes every possible lexical entry. But many entries have extremely low probabilities; a parser will not use them unless it has no better choice. Then is the cost related to the number of high-probability entries? Again no. Expensive probabilities are simply those that deviate greatly from what the transformations would already predict. The prior considers it as unlikely *a priori* for $p(e)$ to be tiny in the above example (via $\theta_9 \ll 0$) as for $p(e')$ to be large for a transformationally implausible entry e'.

De Marcken (1996) likewise made lexical entries cheaper to list if their form could be easily derived from other lexical entries. But in his work, every new entry needed to specify its derivation and its new probability, at some cost. In transformation models, fully predictable entries come at *zero* cost. The only cost to the grammar comes from adjusting entries' probabilities to better match observed data (e.g., $\theta_9 \neq 0$), and from setting up the system of predictive transformations in the first place (e.g., $\theta_3 \neq 0$, $\theta_4 \neq 0$).

7. Experimental evaluation

In this brief report it is only possible to outline the main experiments conducted in (Eisner, 2001). Such experiments examine how well transformation models fit real data, which tests both the linguistic intuitions behind them and their engineering promise.

Lexical entries of the form $\texttt{S} \rightarrow \cdots$ were extracted from the Penn Treebank (Marcus, Santorini, & Marcinkiewicz, 1993), a collection of hand-parsed sentences from *The Wall Street*

Journal. A transformation model with simple local features was trained on 18836 of these entries (a larger version of Table 1) and then used to evaluate the probability of 973 unseen test entries.[13] Nearly half of the test entries had never been observed in training data, and 6.3% of them contained argument structures that had not even been observed with other head words. Yet, it is desirable for a model to generalize and recognize these test entries as plausible; otherwise the correct parses of test sentences would appear implausible. We may check this by measuring a simple, standard quantity called perplexity.

The perplexity of the test entries (given their head words) was 108.6, meaning that the transformation model assigned them an average (geometric mean) probability of 1/108.6. Several other models of lexical entries from the parsing literature were evaluated under the same conditions. The transformation model's perplexity was 20% lower than the best of these, a bigram model (Eisner, 1996). This means that it did a better job of predicting future lexical entries from past ones, being less surprised by them.

An even better perplexity of 102.3 was achieved by averaging the transformation model with an improved bigram model that, like the transformation model, could learn exceptions. This represents a 12% perplexity reduction over the improved bigram model alone. More significantly, the averaged model (the best model with transformations) needed only half as much training data to perform about as well as the improved bigram model (the best model without transformations).

Comparing the detailed predictions of these two best models showed that transformations helped not just on average but across the board. Transformational estimates particularly helped boost the probability of novel or rare argument structures, as hoped, and they were better able to choose which of two novel argument structures had occurred with a word. Meanwhile, there was no evident class of entries on which using transformations hurt.

Table 2 shows some of the probabilities predicted by the learned transformation model. It is possible to see where the model generalized (and how strongly), and where it took pains to fit

Table 2
Part of the probabilistic lexicon induced by transformational smoothing

p(*entry* | *head word*, S)[a]	encourage	question	fund	merge	repay	remove
S → To — NP	0.142	0.117	0.397	0.210	0.329	0.222
S → To — NP PP	0.077	0.064	0.120	0.181	0.088	0.080
S → To AdvP — NP	0.00055	0.00047	0.0011	0.00082	0.00091	0.079
S → To AdvP — NP PP	0.00018	0.00015	0.00033	0.00037	0.00026	0.050
S → NP — NP.	0.022	0.161	0.0078	0.0075	0.0079	0.0075
S → NP — NP PP.	0.079	0.0085	0.0026	0.0027	0.0026	0.0026
S → NP Md — NP	0.090	0.0021	0.0024	0.0020	0.024	0.0026
S → NP Md — NP PP-TMP	0.0018	0.00016	0.00017	0.00016	0.069	0.00019
S → NP Md — PP PP	0.00010	0.000027	0.000027	0.000038	0.000078	0.059
S → To — PP	0.0092	0.0065	0.012	0.126	0.010	0.0091
S → To — S	0.098	0.0016	0.0043	0.0039	0.0036	0.0027
S → NP — SBAR.	0.0034	0.190	0.0032	0.0032	0.0032	0.0032
Other	0.478	0.449	0.449	0.461	0.461	0.482

[a] The model actually defines p(*entry*), but for parsing we need to condition that probability on the head word (encourage) and phrasal category (S), as shown here.

the training data in Table 1. See Eisner, 2001, Section 1.5.3 for discussion and a comparison to the bigram model.

8. Discussion

Several recent papers have used heuristics to generalize lexical subcategorization frames. Briscoe and Copestake (1999) even did so along transformational lines. However, the present work appears to be the first actual statistical model of transformational syntax or lexical redundancy rules. The transformation models introduced here (Fig. 3) could be applied to any formalization of lexical entries, transformations, and features. By learning which features of a transformation make it probable or improbable in a language, the learner acquires a probabilistic syntactic lexicon that can be used for parsing and generation. Much more detail and discussion (and a review of related work) can be found in Eisner (2001).

The major difficulty with the method at present is the lack of a good optimization technique. The model reduces learning to the maximization of a precisely specified function: $p(\boldsymbol{\theta}) \cdot p(D|\boldsymbol{\theta})$. However, optimization techniques such as gradient descent and EM tend to get stuck in local maxima of this function. Moreover, the function is expensive to evaluate even for a single value of $\boldsymbol{\theta}$, since it is non-trivial to derive level-2 lexical entry probabilities from level-4 feature weights. Improving optimization is important if the technique is to be used for large-scale parsing or applied to linguistically richer transformation models.

Notes

1. If observing x is 10 times less likely under $\boldsymbol{\theta}$ than $\boldsymbol{\theta}'$, then observing x twice is 100 times less likely. With enough observations it becomes untenable to choose $\boldsymbol{\theta}$ over $\boldsymbol{\theta}'$.
2. Not to mention idiosyncratic semantic or pragmatic connotations: consider the past participle of the verb `retard`.
3. Like the subregular morphological transformation *sing* $\rightarrow$ *sang*, *ring* $\rightarrow$ *rang*, *swim* $\rightarrow$ *swam*.
4. Much of the extreme rate variation from word to word can actually be predicted by the word's semantics (Levin, 1993). Unaccusative movement is much more likely when the verb has a change-of-state semantic feature. The method described in this paper could in principle discover that correlation, allowing seamless syntactic and semantic bootstrapping (where a verb's change-of-state feature is inferred from its rate of unaccusative movement or vice-versa).
5. Provided that the formalism defines a tree's probability as a function of a product of "scores" assigned to the tree's component lexical entries. (Then transformations can redistribute fractions of an entry's score to related entries.) This is true in standard probability models for CFG and TAG, and more generally in the log-linear models that Johnson and Riezler (this issue) propose for a broader class of grammars.
6. These simple string-like lexical entries will permit a particularly simple set of transformations. But one could instead use lexical entries with more internal structure

(Joshi and Schabes, 1991): $\texttt{S} \rightarrow \texttt{NP}\,[_{\text{VP}}\,[_{\text{VP}}\,[_{\text{V}}\,\underline{\texttt{encourage}}\,]\,\texttt{NP}\,]\,\texttt{PP}\,]$. Besides cosmetically replacing the "flat" tree of Fig. 1 with a more traditional syntax tree, this would permit transformations to be sensitive to this internal structure.

7. The words were not specially chosen, except that each was required to appear with $\texttt{S} \rightarrow \texttt{To}$ — $\texttt{NP PP}$ (so that they would be related) and to appear 4–7 times in total (so that this example table would be small but not trivial). The lexical entries shown were easy to extract from parsed training data, but are overly simple from a linguistic point of view. For example, the first one would be better written as $\texttt{S}_{\texttt{inf}}\backslash\texttt{NP} \rightarrow \texttt{To}$ — $\texttt{NP}$, in order to indicate that this "sentence" is infinitival and has a missing (or silent) subject $\texttt{NP}$. Moreover, since the words shown are zero-inflected, their lexical entries include only infinitive and present-tense constructions. Morphological stemming of the data would have discovered additional entries such as $\texttt{S} \rightarrow \texttt{NP Aux}$ — $\texttt{+ PresP NP}$ (*we are encourag-ing the team*) and $\texttt{NP} \rightarrow \texttt{Det}$ — $\texttt{+ Nom}$ (*much encourage-ment*).
8. In general the graph may have infinitely many vertices, since lexical entries can be arbitrarily long. In this case, it is only possible to explore part of the graph, resulting in underestimates of the probabilities. However, when certain lexical entries are known to be of interest to the parser, one can take pains to explore some paths to them so that at least the estimates are non-zero.
9. Statistical parsers seek the parse tree maximizing $p(tree)$, which they model as a product of conditionalized versions of $p(e)$ probabilities and perhaps other probabilities representing lexical preferences (Charniak, 1997). Eisner (2001, Section 5.4) suggests that surprisingly, the transformation graph could also help estimate the latter.
10. If D was derived from a collection of parses, then its lexical entries were not chosen independently but rather during stochastic tree generation. In this case, $p(D|\boldsymbol{\theta}) = \prod_{tree \in D} p_{\boldsymbol{\theta}}(tree)$, so is proportional to a product of *conditionalized* versions of $p(e)$ probabilities as in the previous footnote.
11. See Eisner (2001). The gradient computation resembles back-propagation in neural networks; so does the E step of EM, which reconstructs the random walks probably taken in the transformation graph. The M step of EM reestimates $\boldsymbol{\theta}$ to make these random walks more likely, using standard "iterative scaling" techniques for parameter estimation in conditional log-linear models. Since these algorithms find only local maxima, it is necessary to make a good heuristic initial guess of $\boldsymbol{\theta}$.
12. Increasing $p(e)$ beyond this will tend to hurt $p(D|\boldsymbol{\theta})$, since $p(e)$ can only be increased at the expense of other lexical entries. The total probability of all lexical entries is 1.
13. A development set of 1588 entries was used to tune control parameters and to determine when to stop training. The experiment actually used a "perturbed transformation model," which uses entry-specific weights (θ_8, θ_9) slightly differently than described here (Eisner, 2001, Section 3.9).

Acknowledgments

Many thanks to Mitch Marcus and the other members of the LINC Lab at the University of Pennsylvania, where this work was conducted, and to the reviewers for helpful feedback. The work was supported in part by ARPA Grant N6600194-C-6043 "Human Language Technology."

References

Aronoff, M. (1976). *Word formation in generative grammar*. Cambridge, MA: MIT Press.

Becker, T. (1994). Patterns in metarules. In *Proceedings of the TAG+3 Workshop*, Paris.

Bresnan, J. (1978). A realistic transformational grammar. In M. Halle, J. Bresnan, & G. A. Miller (Eds.), *Linguistic theory and psychological reality*. Cambridge, MA: MIT Press.

Briscoe, T., & Copestake, A. (1999). Lexical rules in constraint-based grammar. *Computational Linguistics*, *25*(4), 487–526.

Carpenter, B. (1991). The generative power of categorial grammars and head-driven phrase structure grammars with lexical rules. *Computational Linguistics*, *17*(3), 301–313.

Charniak, E. (1997). Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence* (pp. 598–603). Menlo Park: AAAI Press/MIT Press.

De Marcken, C. (1996). *Unsupervised language acquisition*. Unpublished doctoral dissertation, MIT.

Eisner, J. (1996). *An empirical comparison of probability models for dependency grammar* (Tech. Rep. No. IRCS-96-11). Institute for Research in Cognitive Science, University of Pennsylvania.

Eisner, J. (2001). *Smoothing a probabilistic lexicon via syntactic transformations*. Unpublished doctoral dissertation, University of Pennsylvania.

Flickinger, D. P. (1987). *Lexical rules in the hierarchical lexicon*. Unpublished doctoral dissertation, Stanford University.

Johnson, M. (1998). PCFG models of linguistic tree representations. *Computational Linguistics*, *24*(4), 613–632.

Joshi, A. K., & Schabes, Y. (1991). Tree-adjoining grammars and lexicalized grammars. In M. Nivat & A. Podelski (Eds.), *Definability and recognizability of sets of trees*. Amsterdam: Elsevier.

Levin, B. (1993). *English verb classes and alternations: A preliminary investigation*. Chicago, IL: University of Chicago Press.

Marcus, M., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, *19*(2), 313–330.

Pinker, S. (1989). *Learnability and cognition*. Cambridge, MA: MIT Press.