

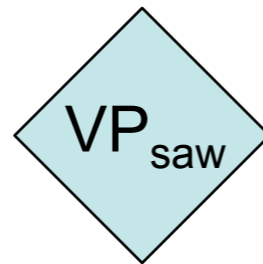
Language and Statistics II

Lecture 14: More parsers!

Charniak (1997) - in brief

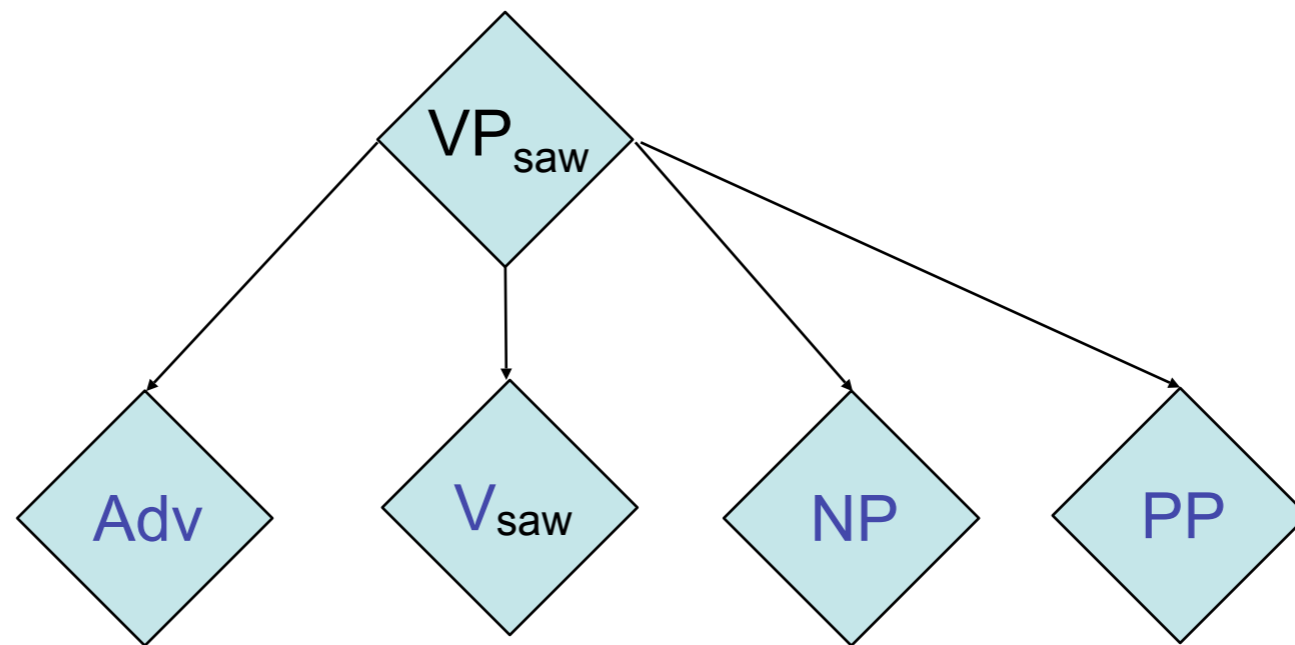
- Generally similar to Collins
- Key differences:
 - Used an additional 30 million words of unparsed text in training
 - Rules not fully markovized: pick full nonterminal sequence, then lexicalize each child independently

Charniak (1997) - in brief



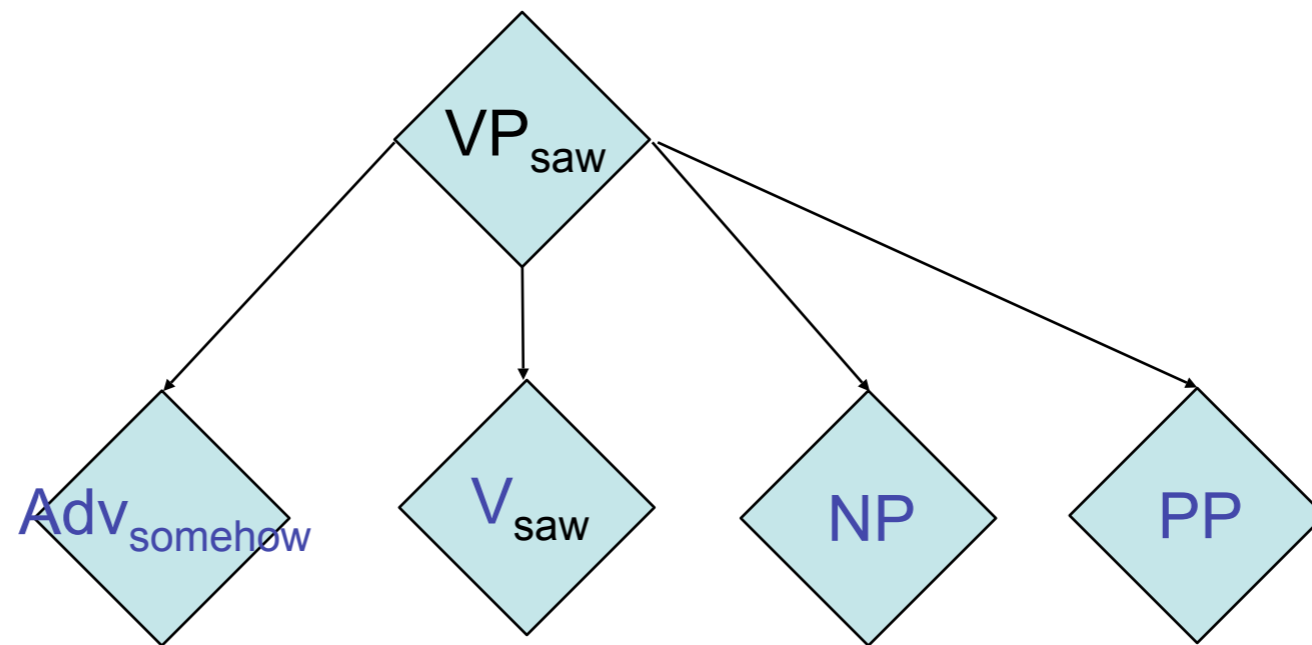
Charniak (1997) - in brief

$VP_{\text{saw}} \rightarrow \text{Adv } \underline{V} \text{ NP PP}$



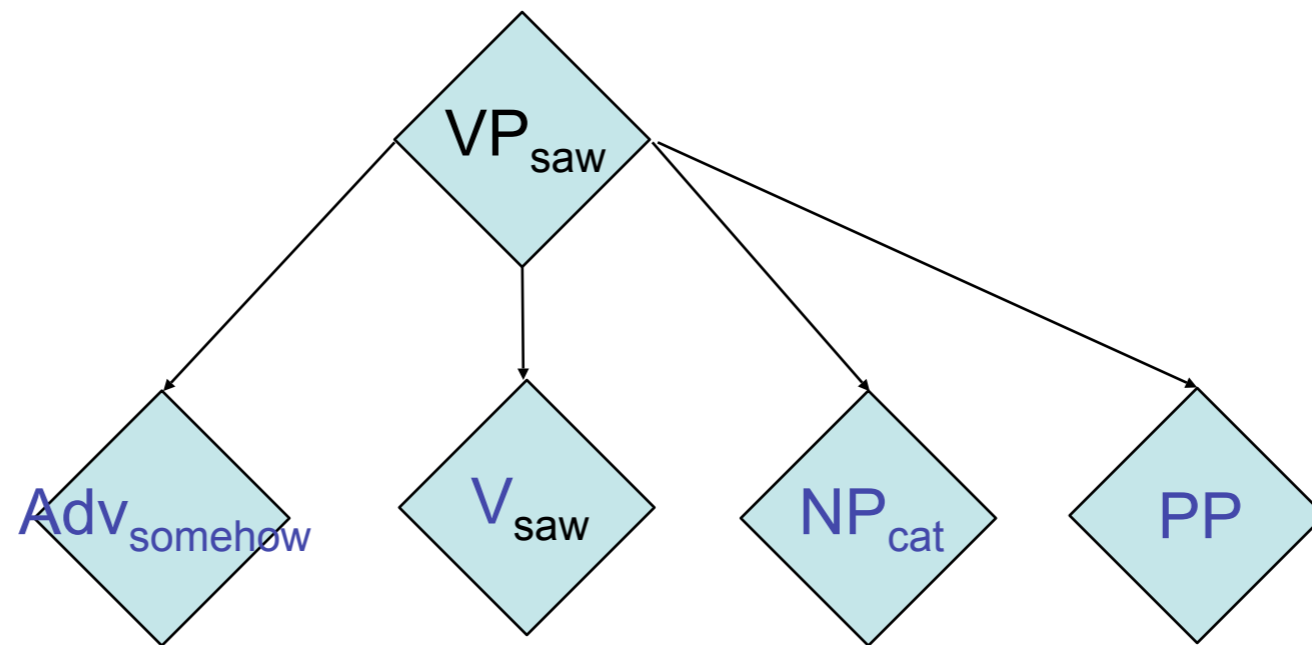
Charniak (1997) - in brief

p(somehow | VP_{saw}, Adv)



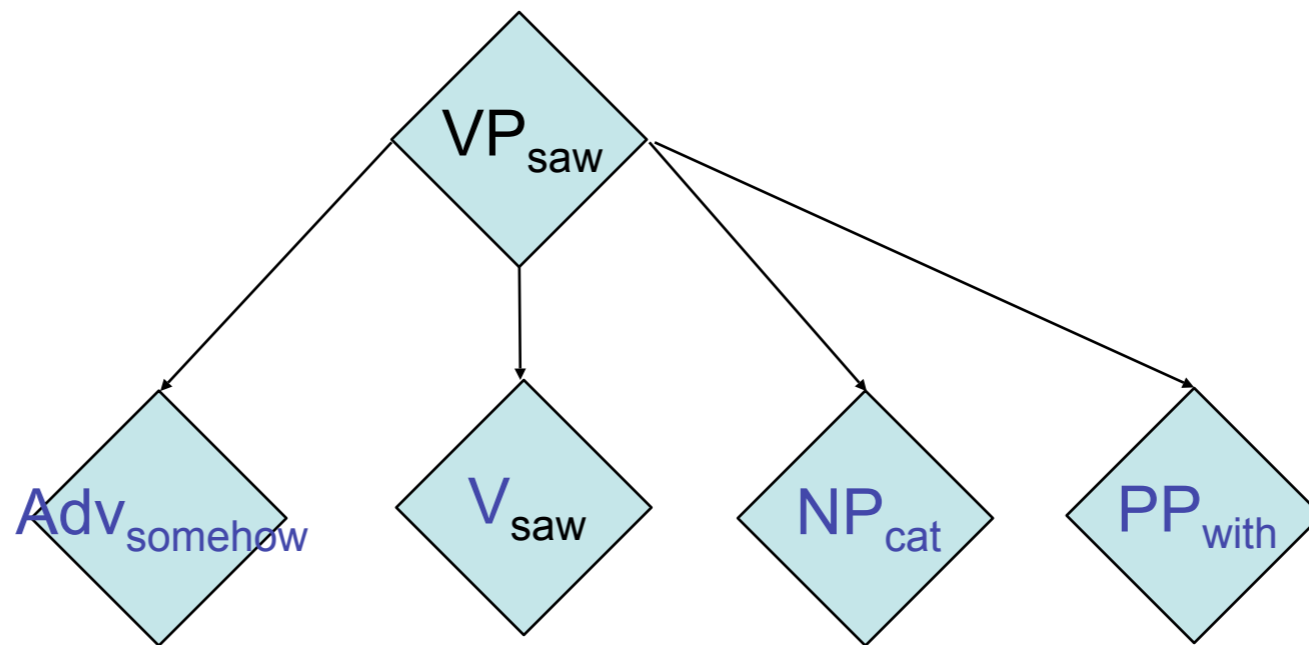
Charniak (1997) - in brief

$p(\text{cat} \mid \text{VP}_{\text{saw}}, \text{NP})$



Charniak (1997) - in brief

p(with | VP_{saw}, PP)



Charniak (2000)

- Uses grandparents (Johnson '98 transformation)
- Markovized children (like Collins)
- Bizarre probability model:
 - Smoothed estimates at many backoff levels
 - Multiply them together
 - “Maximum entropy inspired”
 - Kind of a product of experts (untrained)

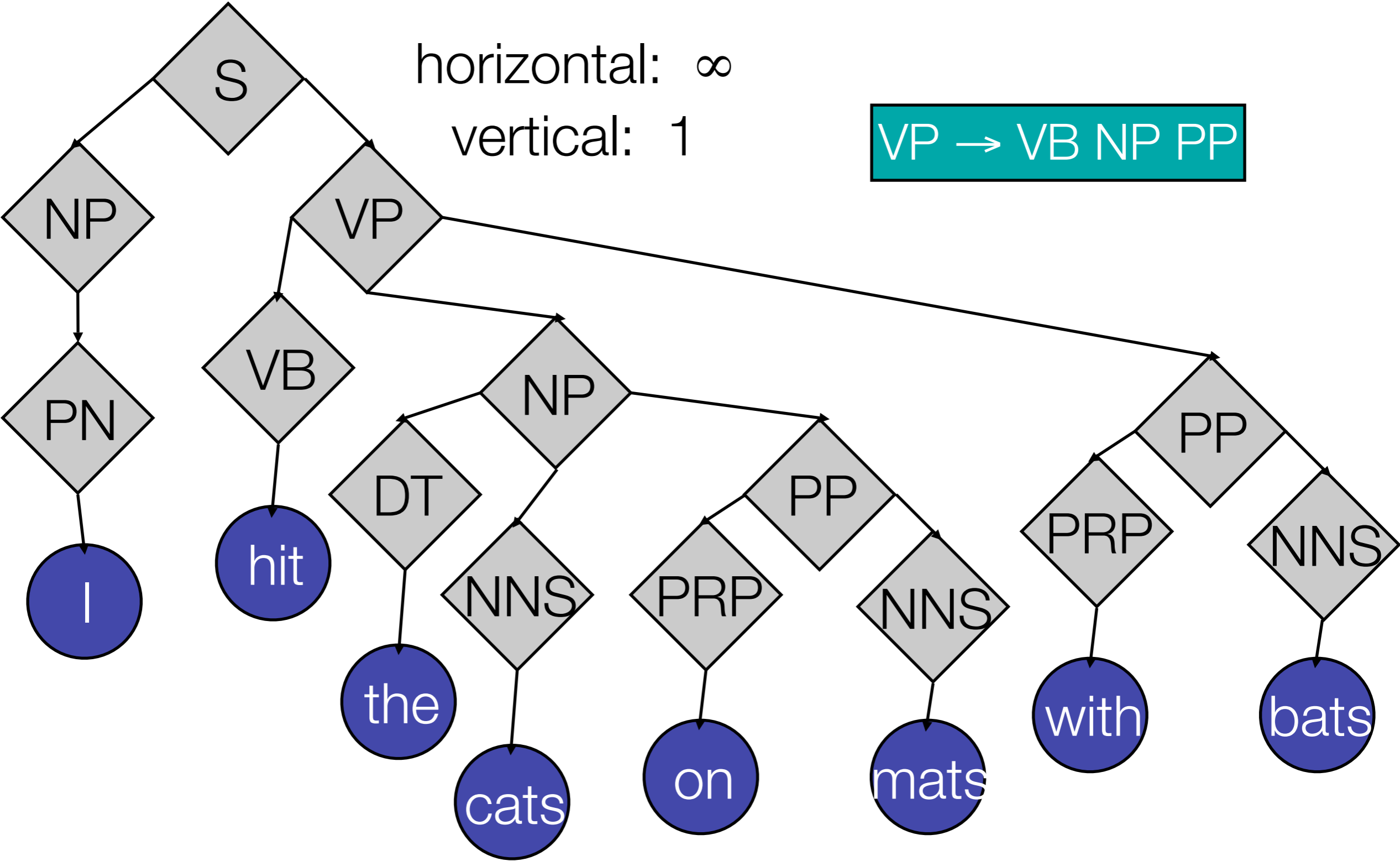
Comparison

| | | labeled recall | labeled precision | average crossing brackets |
|----------|---------|----------------|-------------------|---------------------------|
| Collins | Model 1 | 87.5 | 87.7 | 1.09 |
| | Model 2 | 88.1 | 88.3 | 1.06 |
| | Model 3 | 88.0 | 88.3 | 1.05 |
| Charniak | 1997 | 86.7 | 86.6 | 1.20 |
| | 2000 | 89.6 | 89.5 | 0.88 |

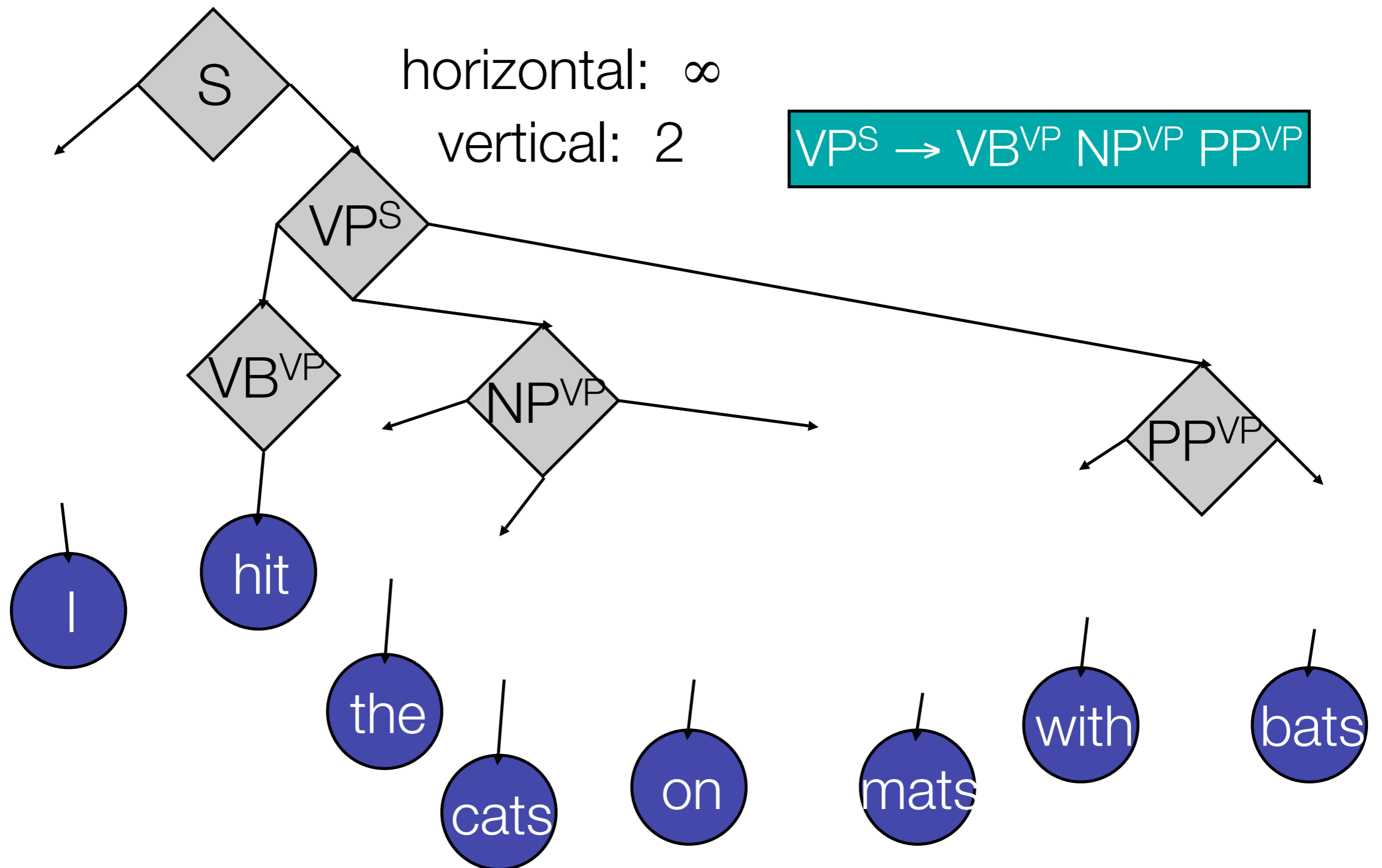
Klein and Manning (2003)

- By now, lexicalization was kind of controversial
 - So many probabilities, such expensive parsing: is it necessary?
- Goal: reasonable unlexicalized baseline
 - What tree transformations make sense?
 - Markovization (what order?)
 - Add all kinds of information to each node in the treebank
- Performance close to Collins model, much better than earlier unlexicalized models

Markovization



Markovization



Markovization

- More vertical Markovization is better
 - Consistent with Johnson (1998)
- Horizontal 1 or 2 beats 0 or ∞
- Used (2, 2), but if sparse “back off” to 1

Other Tree Decorations

- Mark nodes with only 1 child as UNARY
- Mark DTs (determiners), RBs (adverbs) when they are only children
- Annotate POS tags with their parents
- Split IN (prepositions; 6 ways), AUX, CC, %
- NPs: temporal, possessive, base
- VPs annotated with head tag (finite vs. others)
- DOMINATES-V
- RIGHT-RECURSIVE NP

Comparison

| | | labeled recall | labeled precision | average crossing brackets |
|----------|---------|----------------|-------------------|---------------------------|
| Collins | Model 1 | 87.5 | 87.7 | 1.09 |
| | Model 2 | 88.1 | 88.3 | 1.06 |
| | Model 3 | 88.0 | 88.3 | 1.05 |
| Charniak | 1997 | 86.7 | 86.6 | 1.20 |
| | 2000 | 89.6 | 89.5 | 0.88 |
| K&M | 2003 | 86.3 | 85.1 | 1.31 |

Probabilistic Automata

- FSA is to regular grammar as ___ is to context-free grammar
- Nondeterministic PDAs are more expressive than deterministic ones.
- Can define **probabilistic** PDAs, too.
- The correspondence isn't as direct as for WFSA's, and the theoretical construct isn't a perfect fit to the models, but the idea is related.

Parsers as Automata

- Move left to right.
- Eat words as you go, deciding what to do with them.
 - Think of “scan,” “predict,” and “complete” actions in an Earley parser.
 - Think of “shift” and “reduce” actions.
 - Actions modeled empirically!
- No dynamic programming; use generalized search instead.
 - Greedy methods often called “deterministic” parsing.

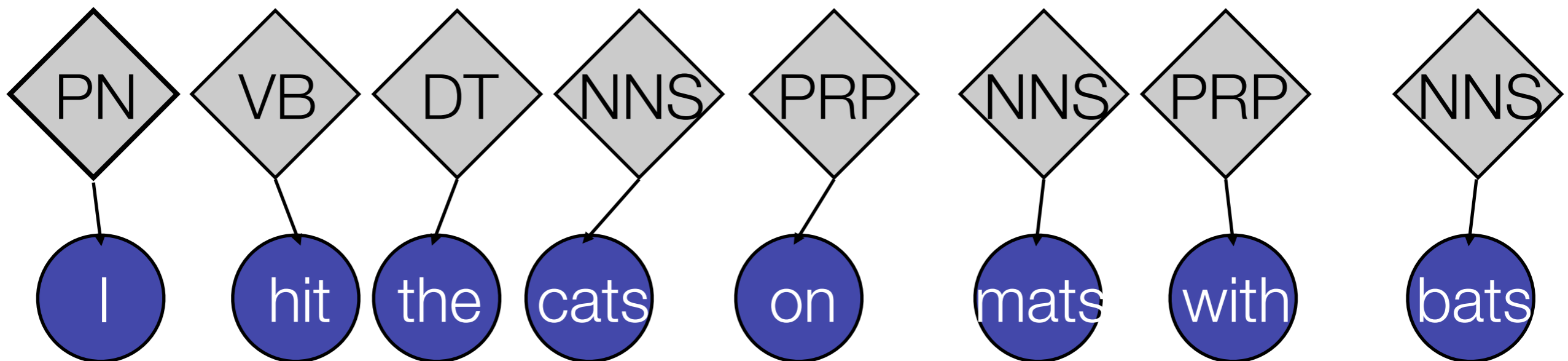
Ratnaparkhi (1998)

- Tagging, then chunking, then parsing (3 passes)
- Log-linear model: $p(\text{next action} \mid \text{history})$
 - Features include lots of context, the CFG rule, words, tags, etc.
- Beam search
- Results:
 - $O(n)$ observed runtime!
 - A little worse on performance than Collins Model 1.
- See also: Magerman (1995; decision trees); Chelba & Jelinek (1998; MLE); Sagae & Lavie (2005, SVMs); Nivre et al. (2006; SVMs)

Ratnaparkhi (1998)

Build:

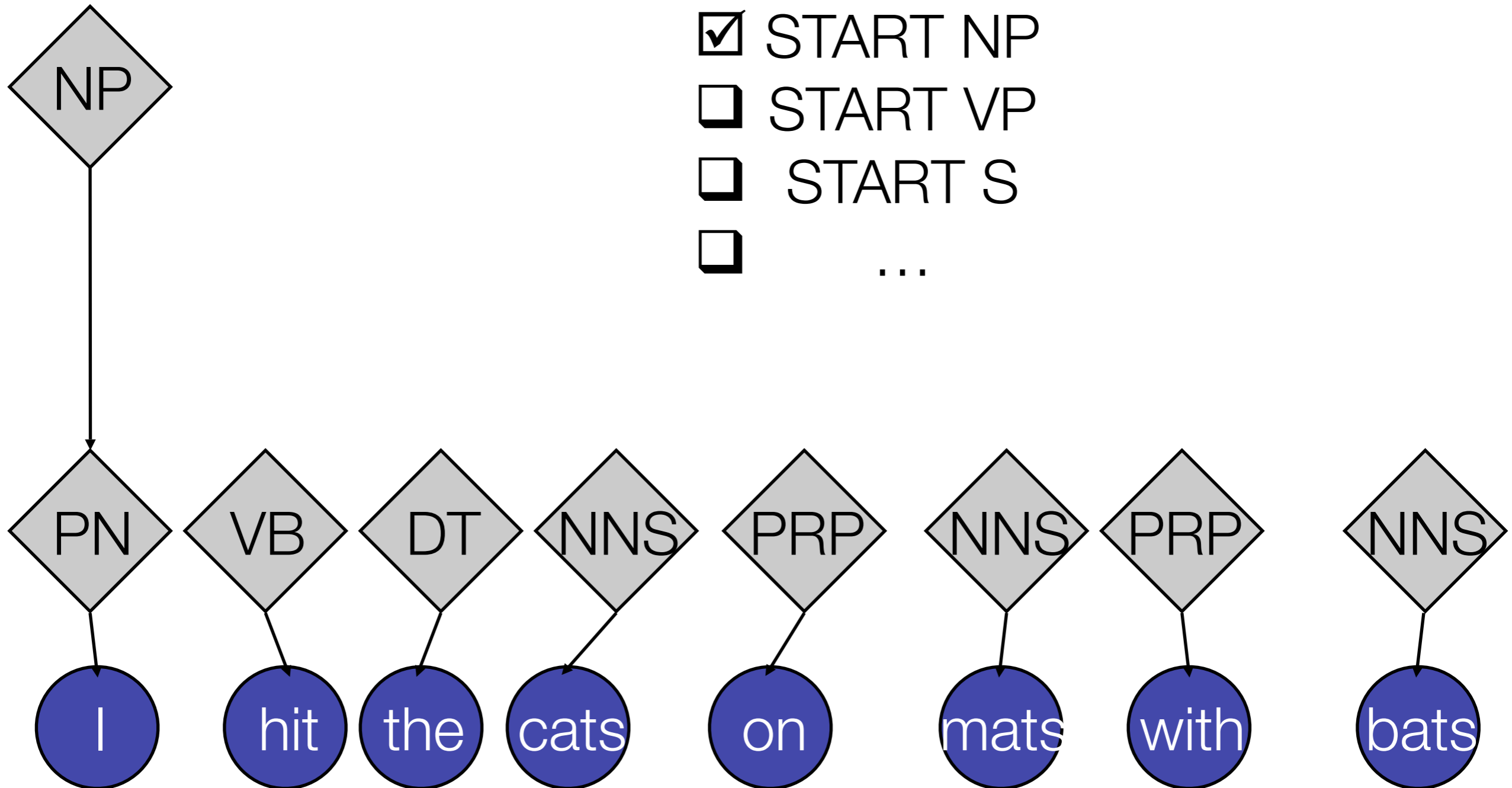
- START NP
- START VP
- START S
- ...



Ratnaparkhi (1998)

Build:

- START NP
- START VP
- START S
- ...

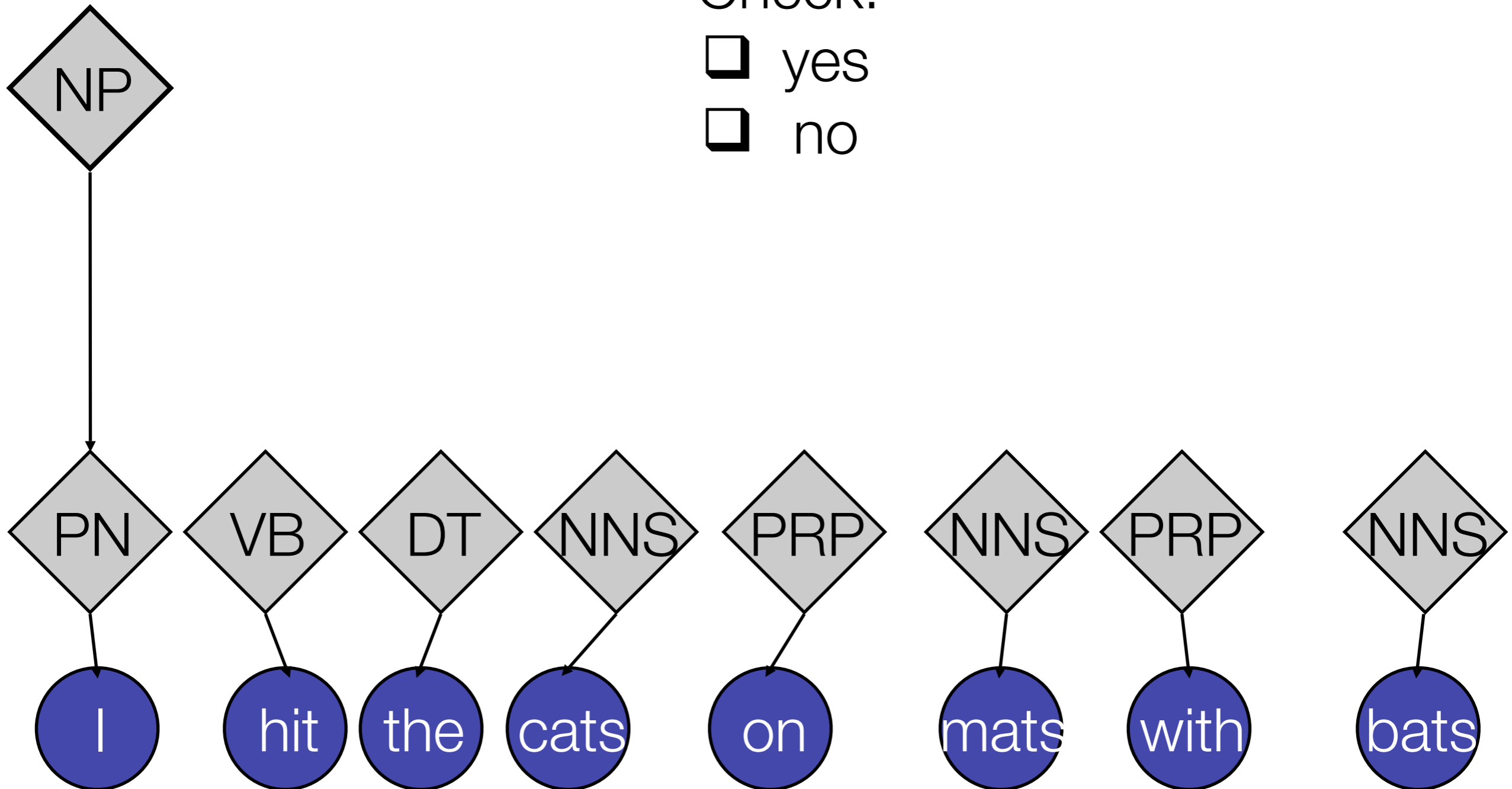


Ratnaparkhi (1998)

Check:

yes

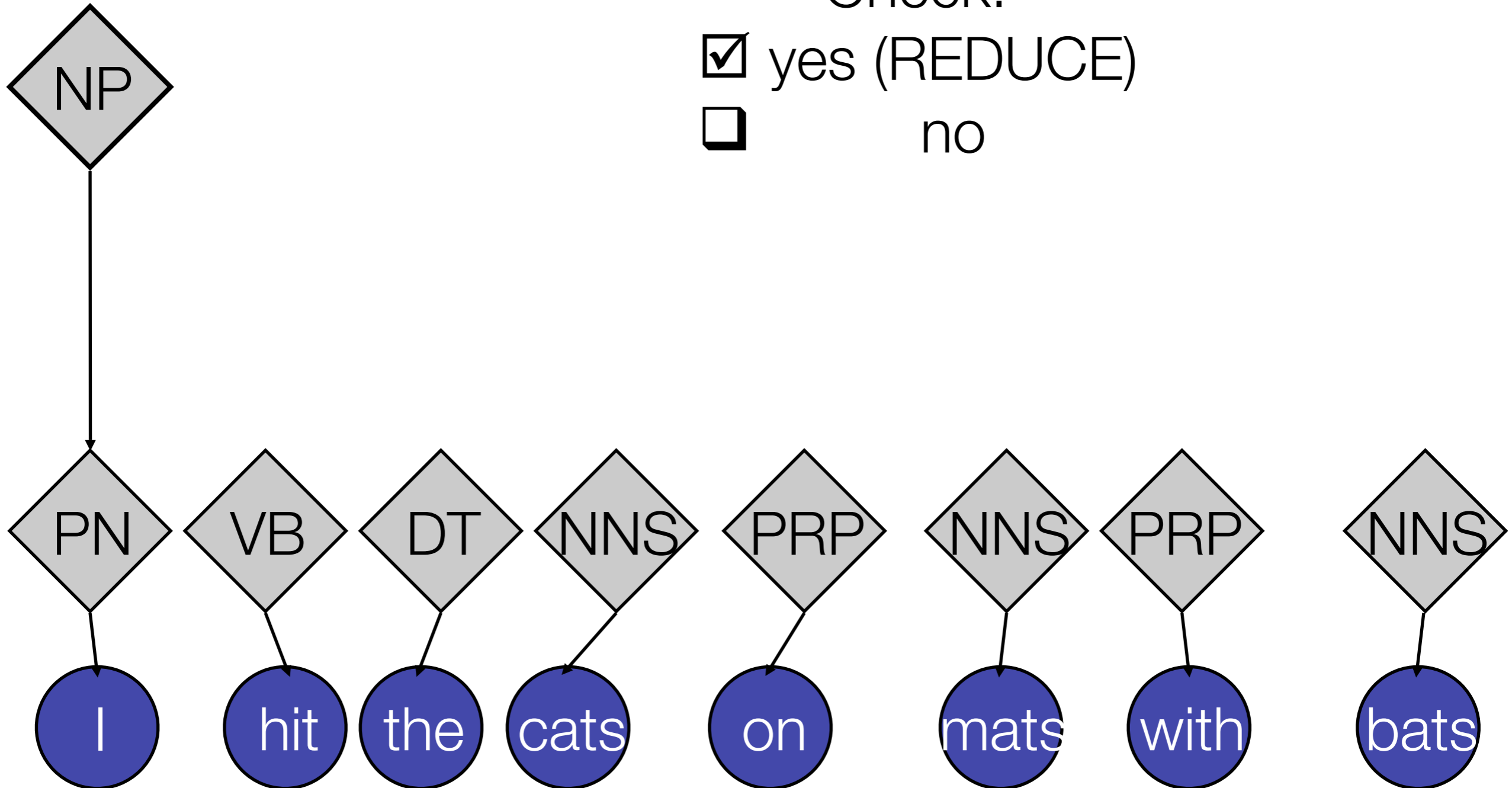
no



Ratnaparkhi (1998)

Check:

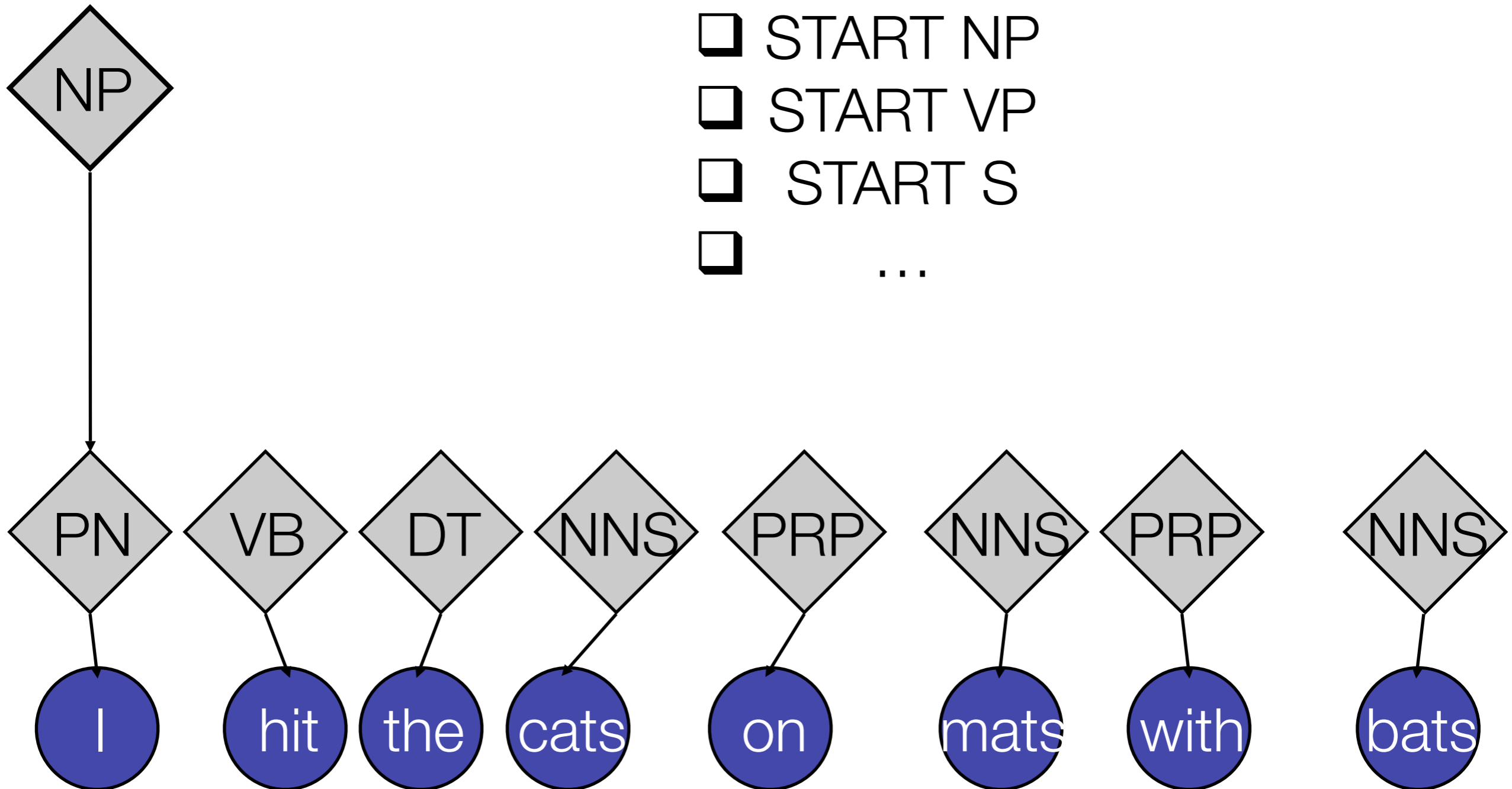
- yes (REDUCE)
- no



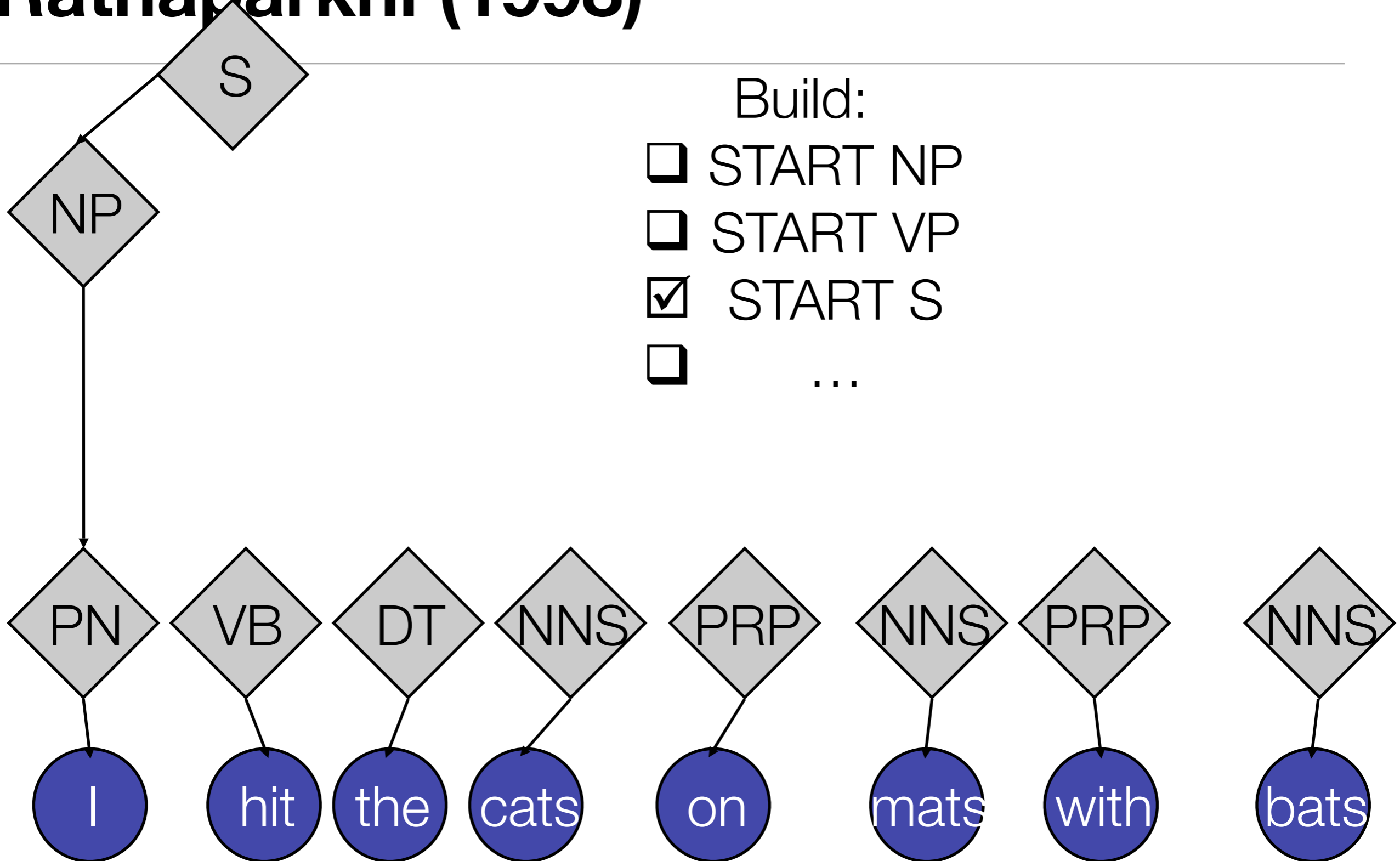
Ratnaparkhi (1998)

Build:

- START NP
- START VP
- START S
- ...



Ratnaparkhi (1998)

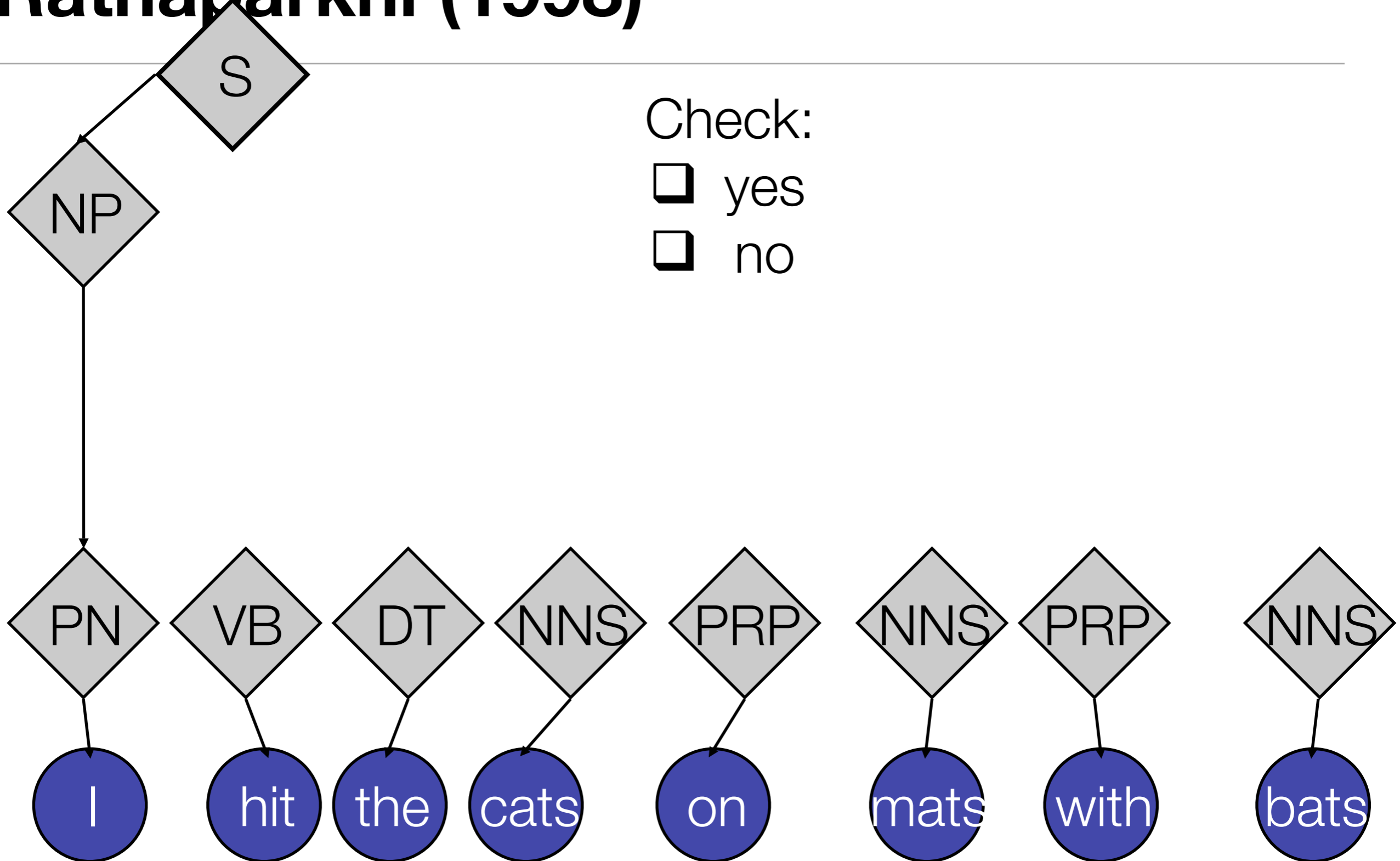


Ratnaparkhi (1998)

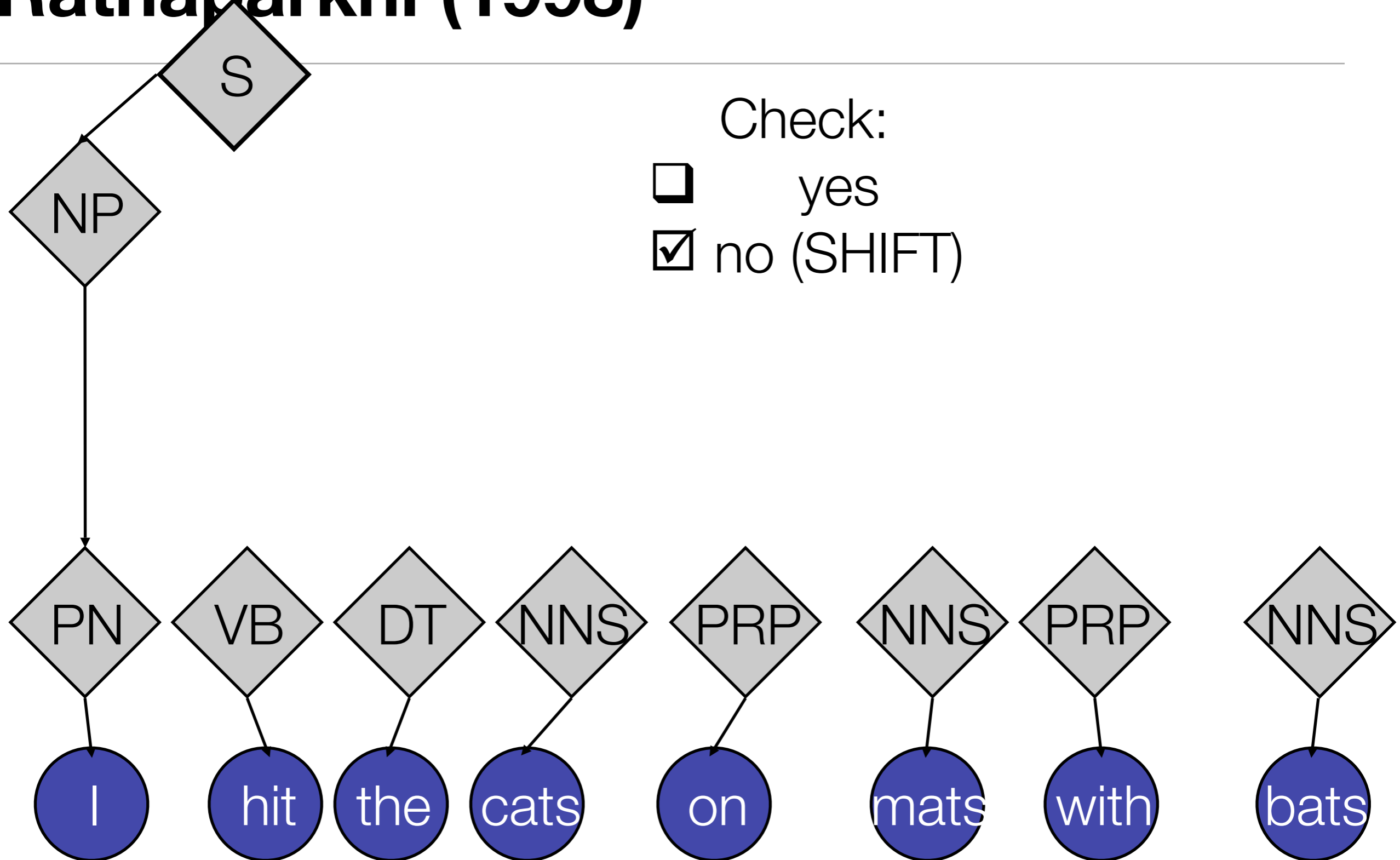
Check:

yes

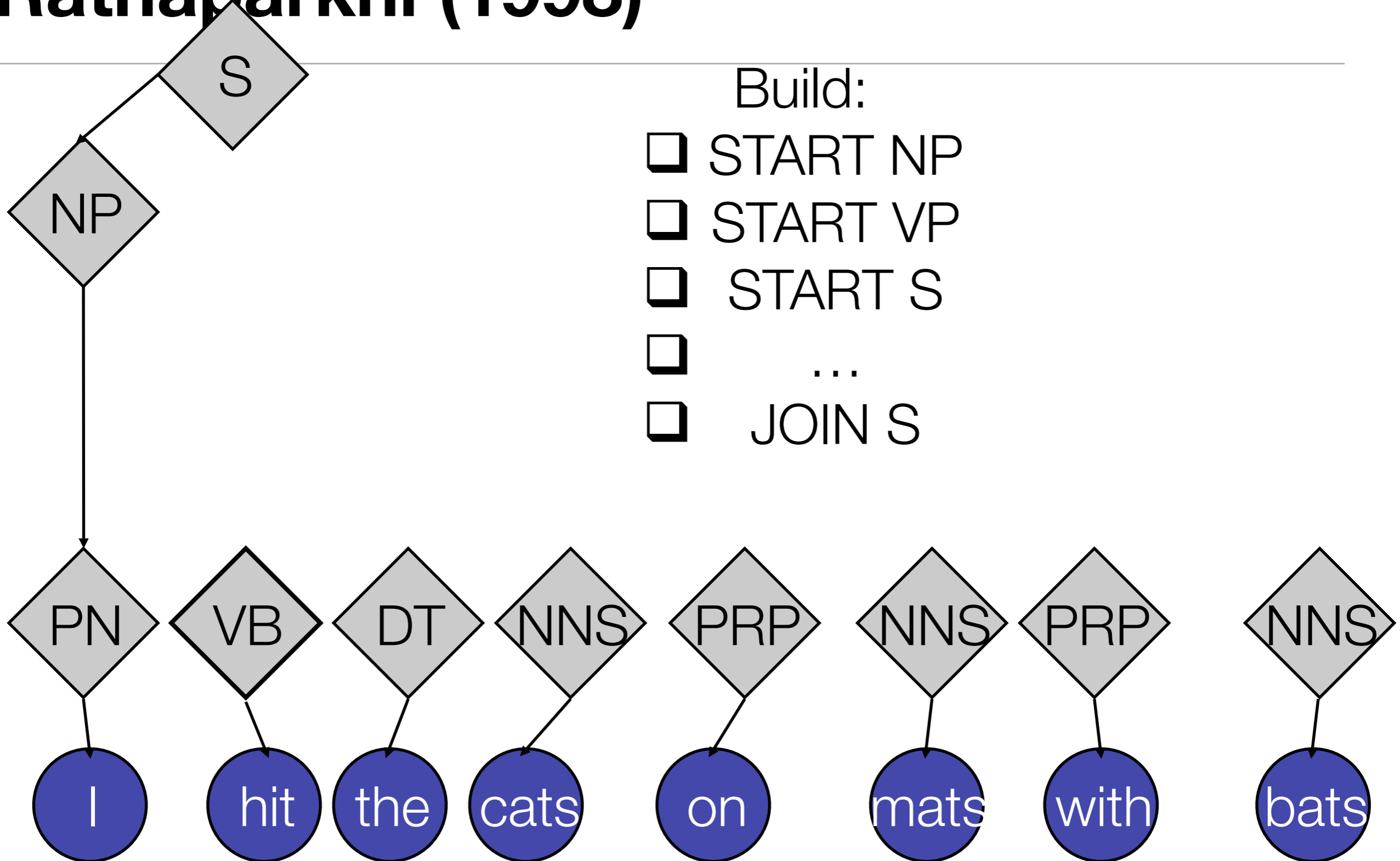
no



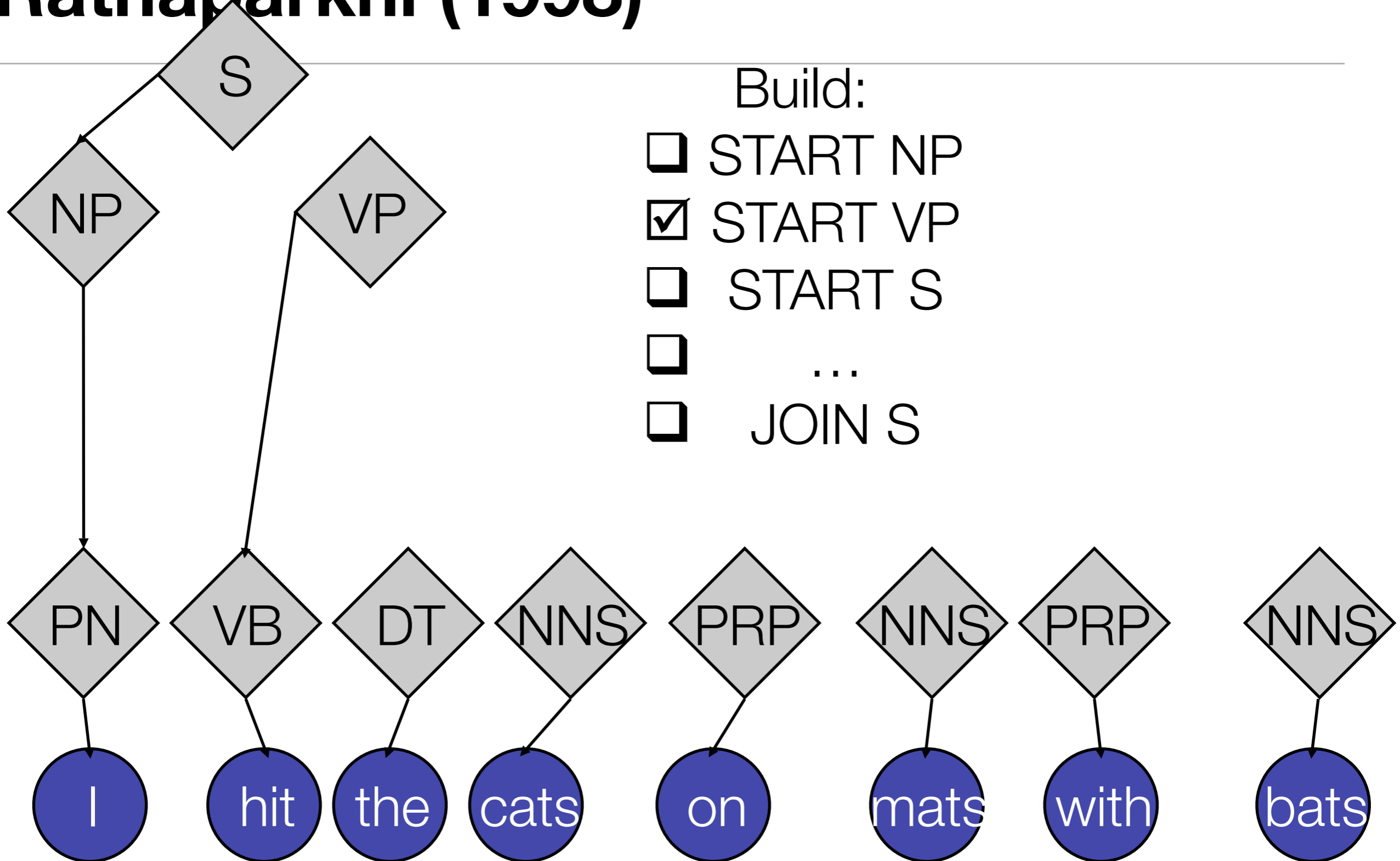
Ratnaparkhi (1998)



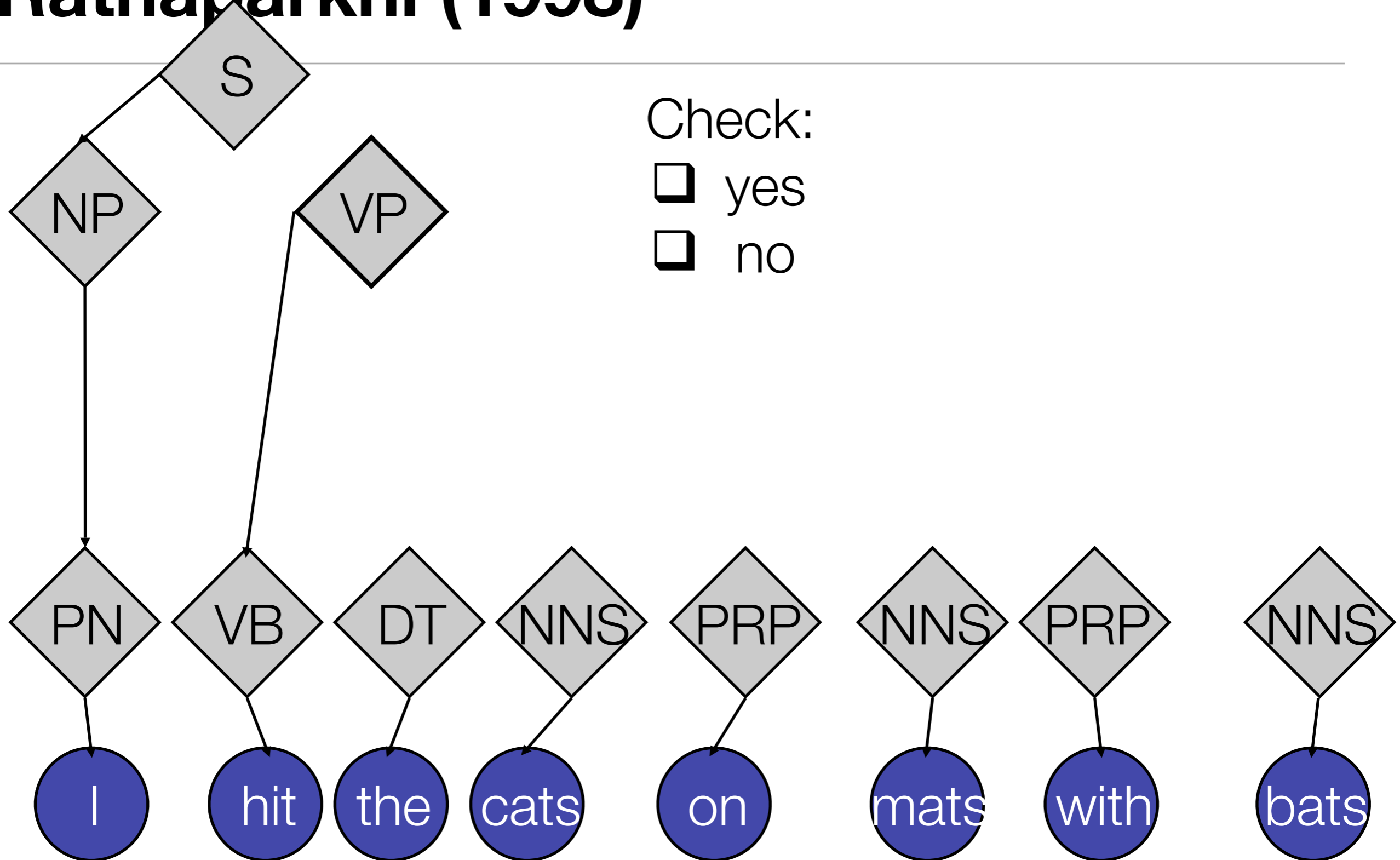
Ratnaparkhi (1998)



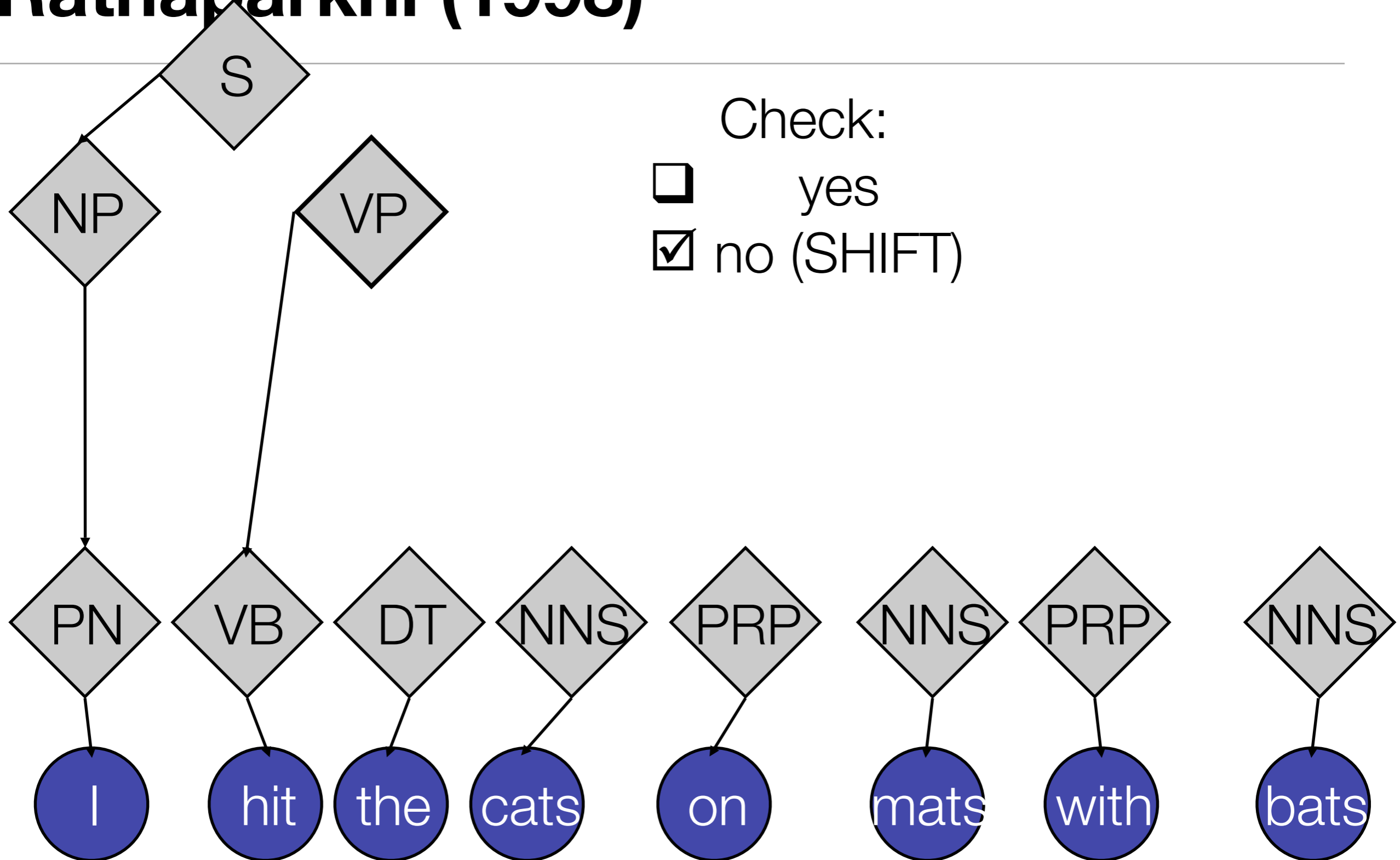
Ratnaparkhi (1998)



Ratnaparkhi (1998)



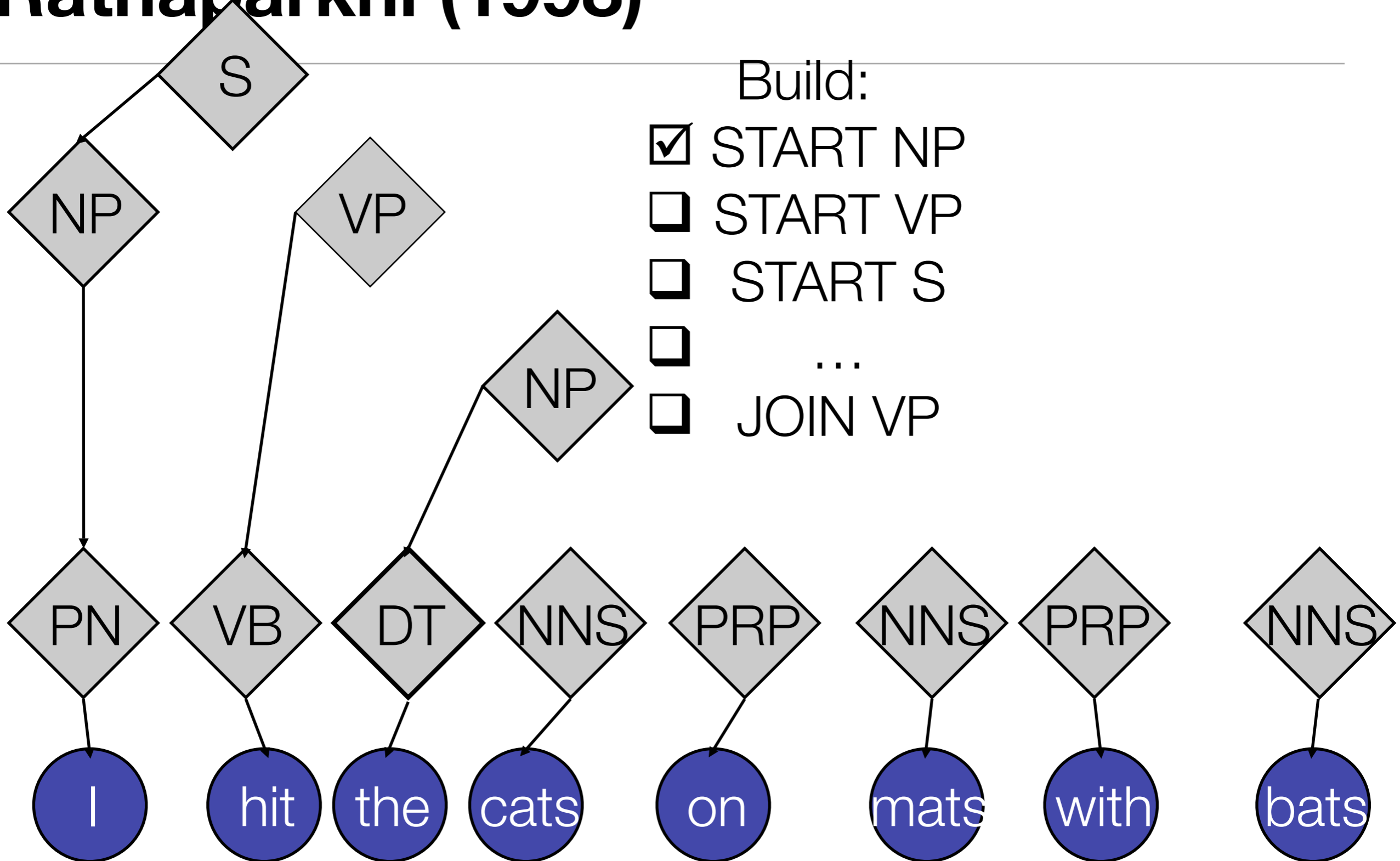
Ratnaparkhi (1998)



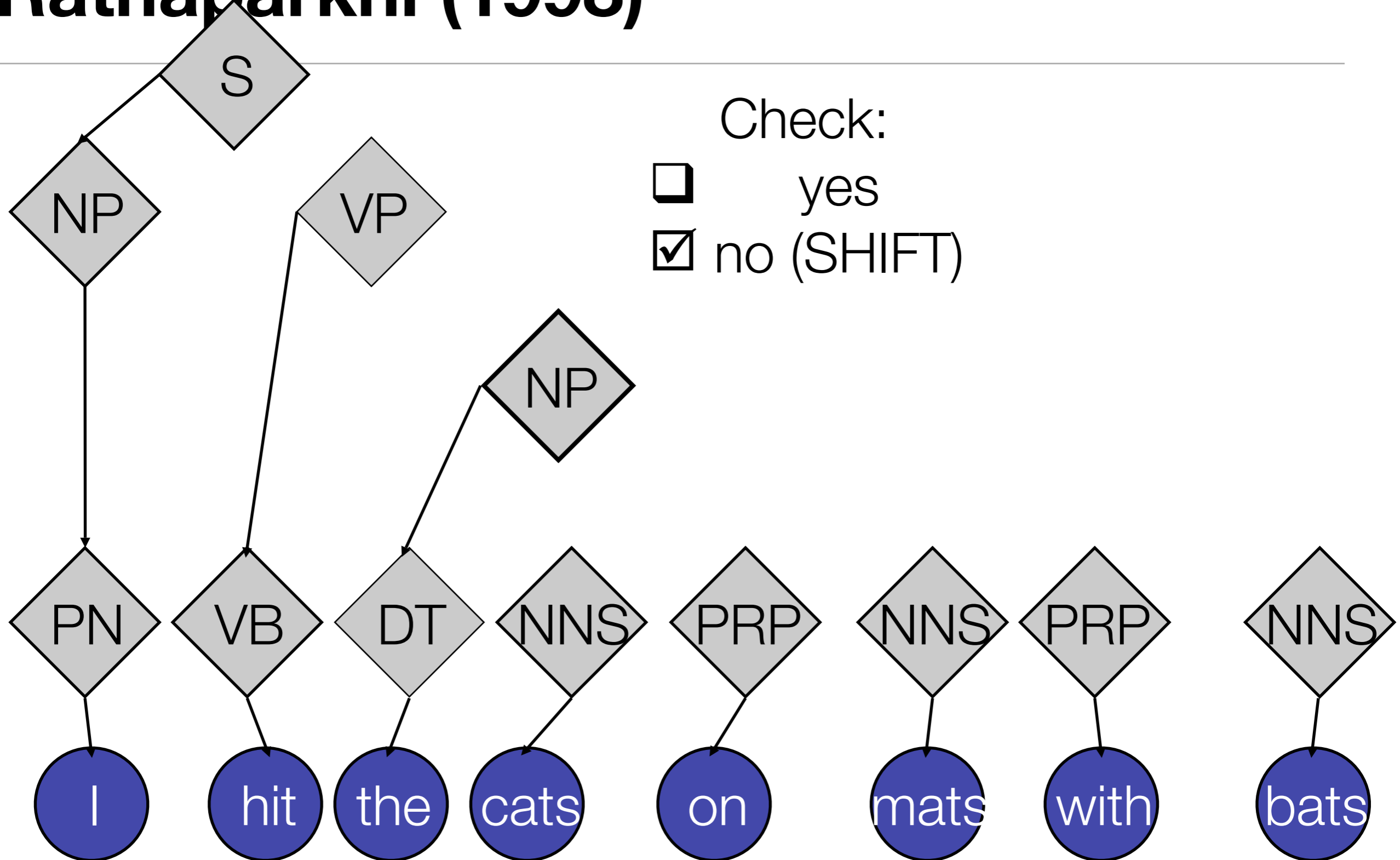
Ratnaparkhi (1998)

Build:

- START NP
- START VP
- START S
- ...
- JOIN VP



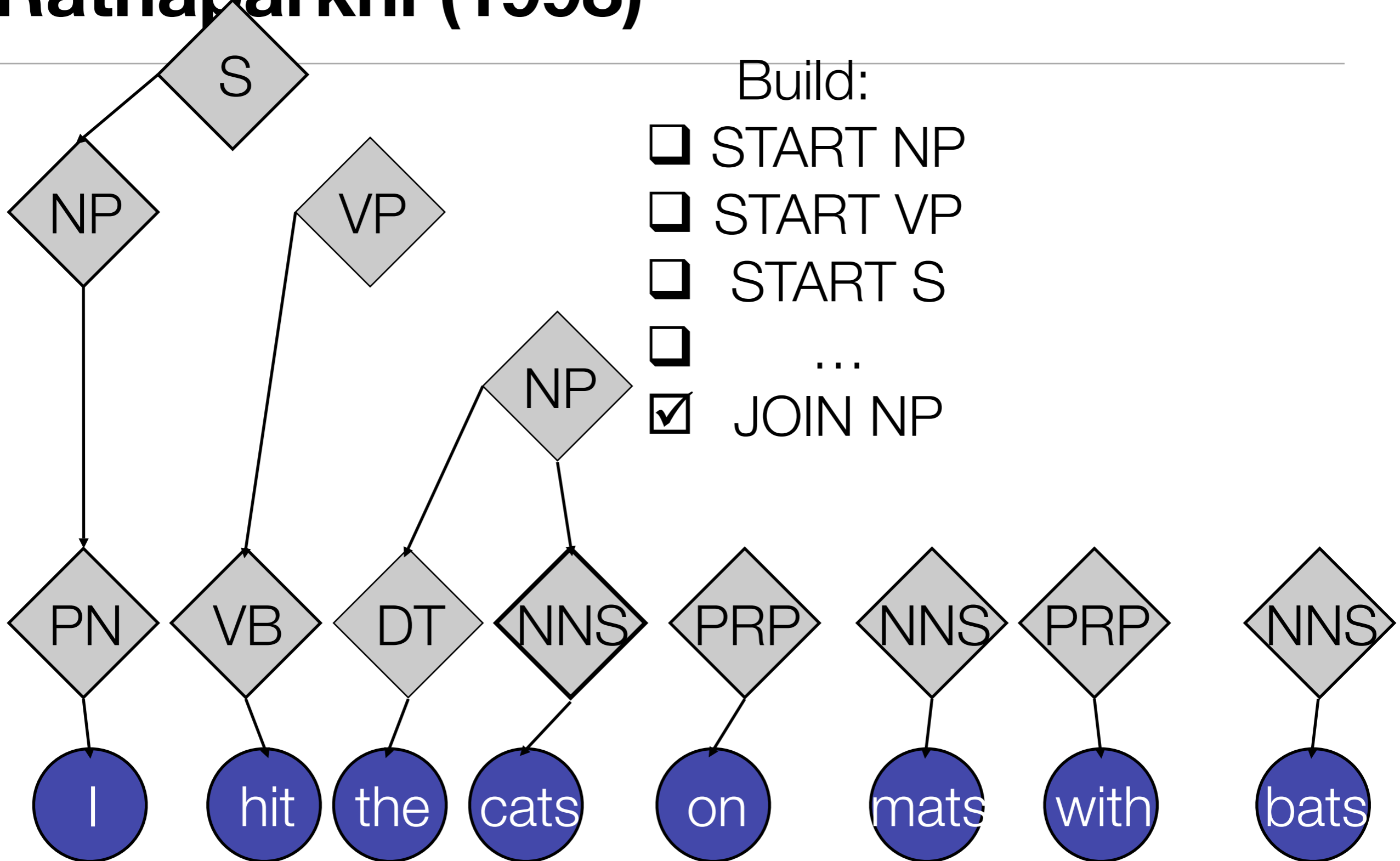
Ratnaparkhi (1998)



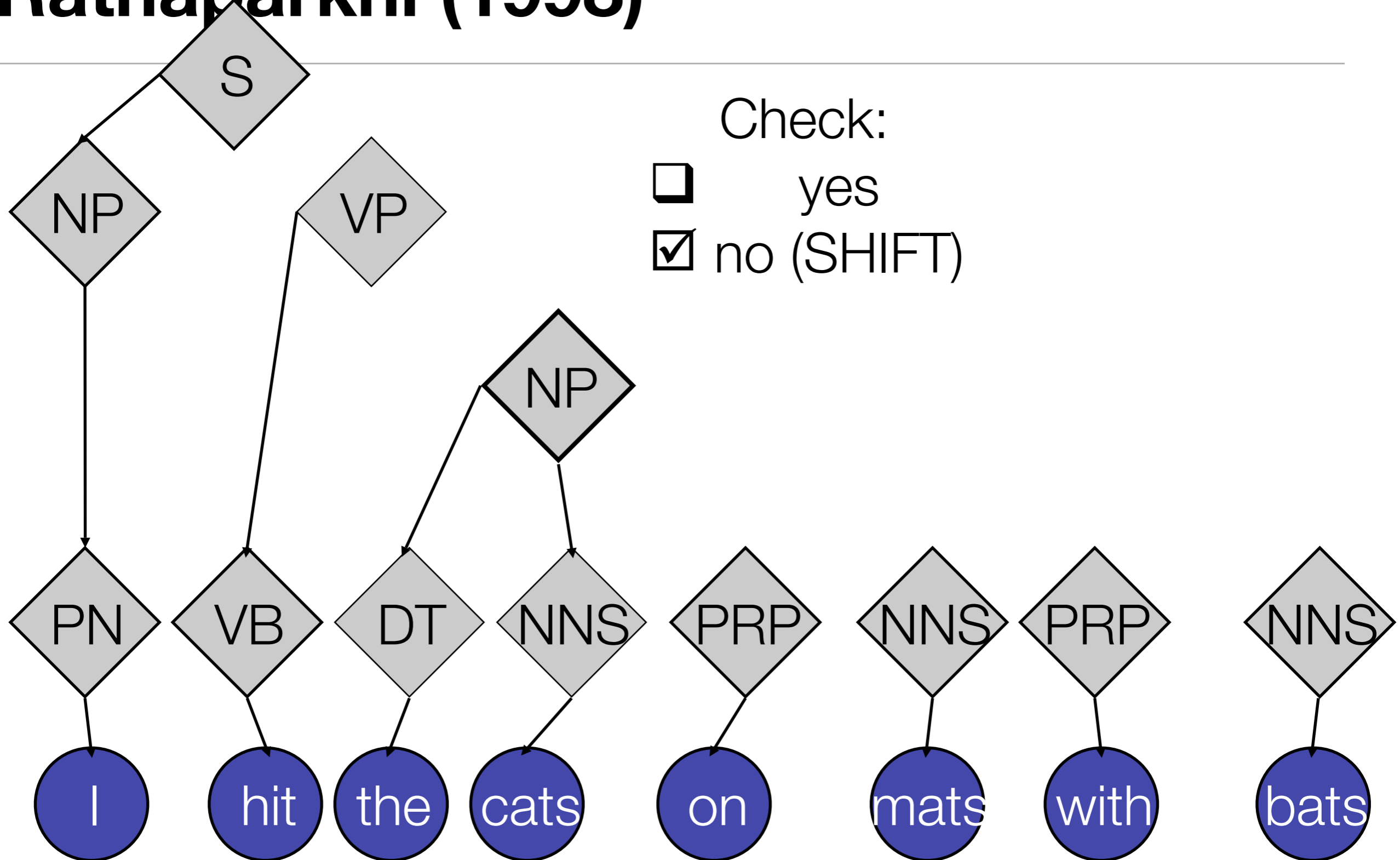
Ratnaparkhi (1998)

Build:

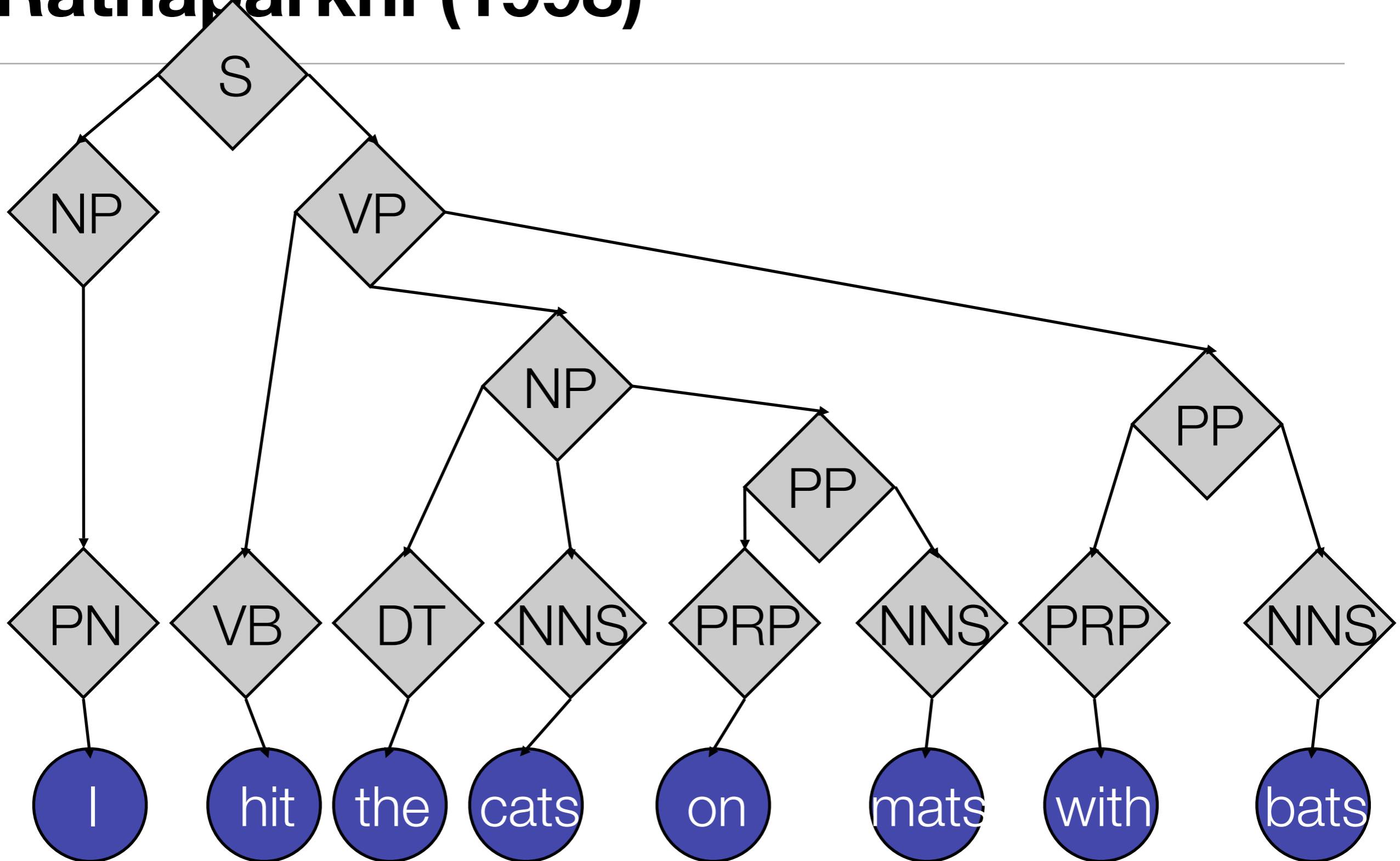
- START NP
- START VP
- START S
- ...
- JOIN NP



Ratnaparkhi (1998)



Ratnaparkhi (1998)



Finkel, Kleeman, and Manning (2008)

- CRF-CFG
 - Expensive training; SGD with parallelization
 - Four machines = three days per pass over training data
- Features are rule-local, but can look at the entire yield sentence
- Distributional similarity features from large unannotated dataset, too
-

Finkel et al.: Features

Table 1: Lexicon and grammar features. w is the word and t the tag. r represents a particular rule along with span/split information; ρ is the rule itself, r_p is the parent of the rule; w_b , w_s , and w_e are the first, first after the split (for binary rules) and last word that a rule spans in a particular context. All states, including the POS tags, are annotated with parent information; $b(s)$ represents the base label for a state s and $p(s)$ represents the parent annotation on state s . $ds(w)$ represents the distributional similarity cluster, and $lc(w)$ the lower cased version of the word, and $unk(w)$ the unknown word class.

| Lexicon Features | Grammar Features | |
|------------------------------------|--------------------------------------|--|
| t | | Binary-specific features |
| $b(t)$ | ρ | |
| $\langle t, w \rangle$ | $\langle b(p(r_p)), ds(w_s) \rangle$ | $\langle b(p(r_p)), ds(w_{s-1}, ds w_s) \rangle$ |
| $\langle t, lc(w) \rangle$ | $\langle b(p(r_p)), ds(w_e) \rangle$ | PP feature: |
| $\langle b(t), w \rangle$ | unary? | if right child is a PP then $\langle r, w_s \rangle$ |
| $\langle b(t), lc(w) \rangle$ | simplified rule: | VP features: |
| $\langle t, ds(w) \rangle$ | base labels of states | if some child is a verb tag, then rule, |
| $\langle t, ds(w_{-1}) \rangle$ | dist sim bigrams: | with that child replaced by the word |
| $\langle t, ds(w_{+1}) \rangle$ | all dist. sim. bigrams below | |
| $\langle b(t), ds(w) \rangle$ | rule, and base parent state | Unaries which span one word: |
| $\langle b(t), ds(w_{-1}) \rangle$ | dist sim bigrams: | |
| $\langle b(t), ds(w_{+1}) \rangle$ | same as above, but trigrams | $\langle r, w \rangle$ |
| $\langle p(t), w \rangle$ | heavy feature: | $\langle r, ds(w) \rangle$ |
| $\langle t, unk(w) \rangle$ | whether the constituent is “big” | $\langle b(p(r)), w \rangle$ |
| $\langle b(t), unk(w) \rangle$ | as described in (Johnson, 2001) | $\langle b(p(r)), ds(w) \rangle$ |

Finkel et al.: Results

| Model | P | R | F ₁ | Exact | Avg CB | 0 CB | P | R | F ₁ | Exact | Avg CB | 0 CB |
|----------------|-----------------------------|------|----------------|-------|--------|------|--------------------------|------|----------------|-------|--------|------|
| | test set – length ≤ 40 | | | | | | test set – all sentences | | | | | |
| Petrov 2007 | – | – | 88.8 | – | – | – | – | – | 88.3 | – | – | – |
| generative | 83.5 | 82.0 | 82.8 | 25.5 | 1.57 | 53.4 | 82.8 | 81.2 | 82.0 | 23.8 | 1.83 | 50.4 |
| generative-all | 83.6 | 82.1 | 82.8 | 25.2 | 1.56 | 53.3 | – | – | – | – | – | – |
| discriminative | 85.1 | 84.5 | 84.8 | 29.7 | 1.41 | 55.8 | 84.2 | 83.7 | 83.9 | 27.8 | 1.67 | 52.8 |
| feature-based | 89.2 | 88.8 | 89.0 | 37.3 | 0.92 | 65.1 | 88.2 | 87.8 | 88.0 | 35.1 | 1.15 | 62.3 |

Table 4: Test set results, training on sentences of length ≤ 40 from the Penn treebank. The *generative-all* results were trained on all sentences regardless of length

(Mild) Context Sensitivity

Many more expressive formalisms have been made probabilistic:

- Attribute-Value Grammar (Abney)
- Tree Adjoining Grammar (Resnik, *inter alia*)
 - “Spine” TAG (Collins)
- Lexical-Functional Grammar (Riezler, *inter alia*)
- Tree Insertion Grammar (Hwa)
- Combinatory Categorical Grammar (Curran and Clark)
- Head-driven Phrase Structure Grammar (Tsuji'i)

Lots of emphasis on speed; sometimes **stochastic process** not possible (one solution: log-linear models).

Finite-State Parsing

Yes, really!

Imagine an FST that inserts brackets. Apply it repeatedly. (Basic idea motivated and described in Roche, 1997.)

- Lots of theoretical work on approximating (P)CFGs with (W)FSAs (see Nederhof, 2001).
- Abney (2000) - partial parsing
- Eisner & Smith (2005) - dependency length constraints → regular language

Reranking

- Really want **non-local** features to influence parsing decisions.
 - Hard to get this into PCFGs, as we've seen.
- Collins (2000): re-rank the top n parses from a standard parser (>89%)
- Huang and Chiang (2005): exact n -best parses from CKY (or similar) parser
- Charniak & Johnson (2005): log-linear model for reranking, using Huang & Chiang's method for n -best list → even better!