Language and Statistics II

Lecture 12: Modern Parsing

Noah Smith

Lecture Outline

- PCFGs
  - Final comments about Collins parser
  - Charniak Parsers, in brief
  - Klein and Manning (2003)
- Probabilistic automata for parsing
  - Ratnaparkhi (1998)
- Dependency parsing: models, algorithms
- Reranking
- Other topics: up & down the Chomsky hierarchy

Other Details

- Smoothing: deleted interpolation.
- Unknown words: every type with count ≤ 5 became UNK
- Tagging is not a separate stage; it is just part of the parse.

Further Refinements

- Base noun phrases
  - Labeled “NPB”
  - First-order Markov model for children of head!
- Coordinators (“and”) predicted **together** with the later argument.
- Punctuation treated similarly (see the 2003 paper)

Charniak (1997)

- Similar setup.
  - Lexicalized PCFG, factored model for rules
  - Tags don’t travel up the tree as in Collins
  - Tagging part of parsing
  - Deleted interpolation for smoothing
- Used an additional 30 million words of unannotated data.

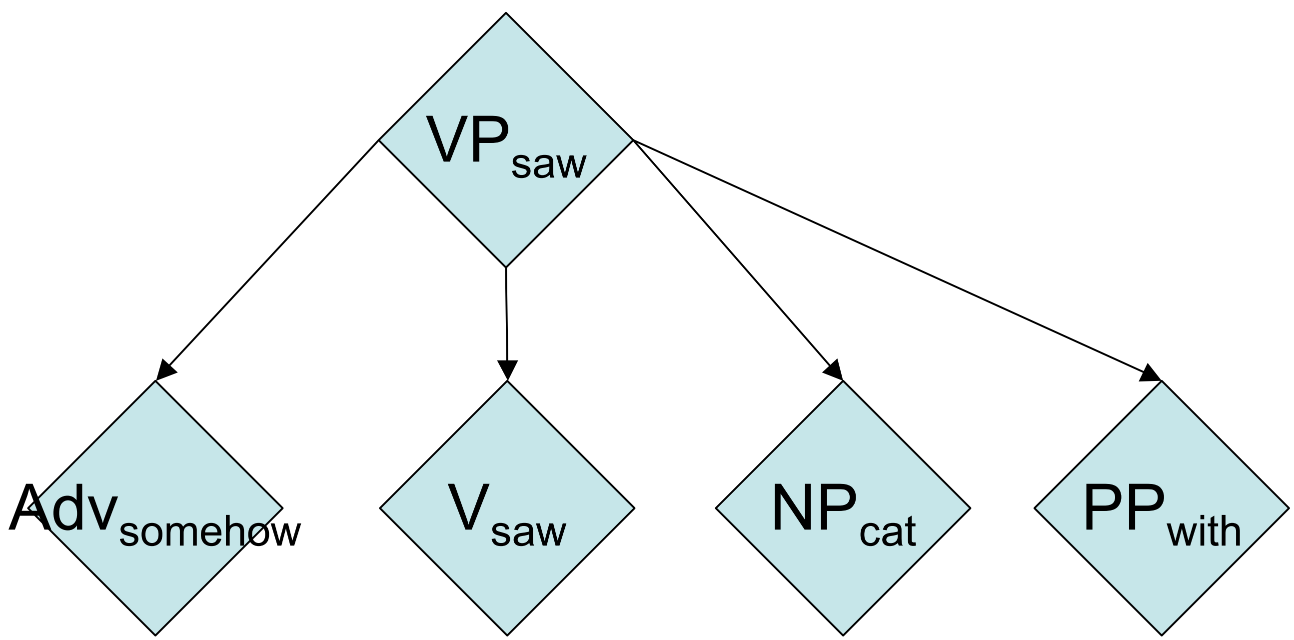
Charniak (1997)

$p(\text{Adv } \underline{V}_{saw} \text{ NP PP} \mid VP_{saw}, S)$

$p(\text{somehow} \mid VP_{saw}, \text{Adv})$

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

$p(\text{with} \mid VP_{saw}, \text{PP})$



Charniak (2000)

- The 2000 parser is “maximum entropy inspired.”
- Uses grandparents.
- It is closer to Collins’ model (Markovized children), but the estimation is bizarre.
  - Smoothed, backed-off probabilities are multiplied together - almost like a **product of experts**.

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

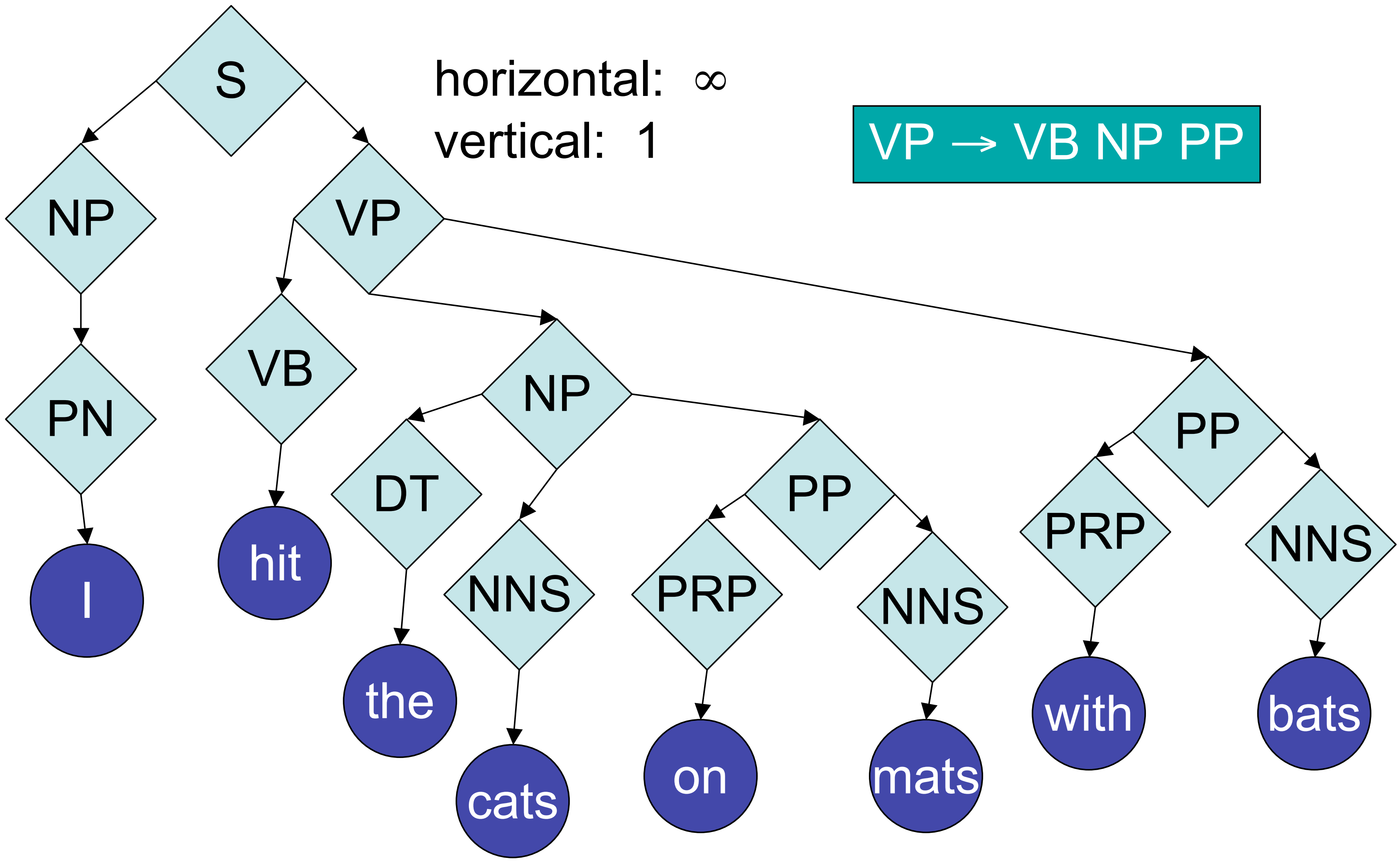
Klein and Manning (2003)

- By now, lexicalization was kind of controversial
- 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

horizontal: ∞
vertical: 1

VP → VB NP PP



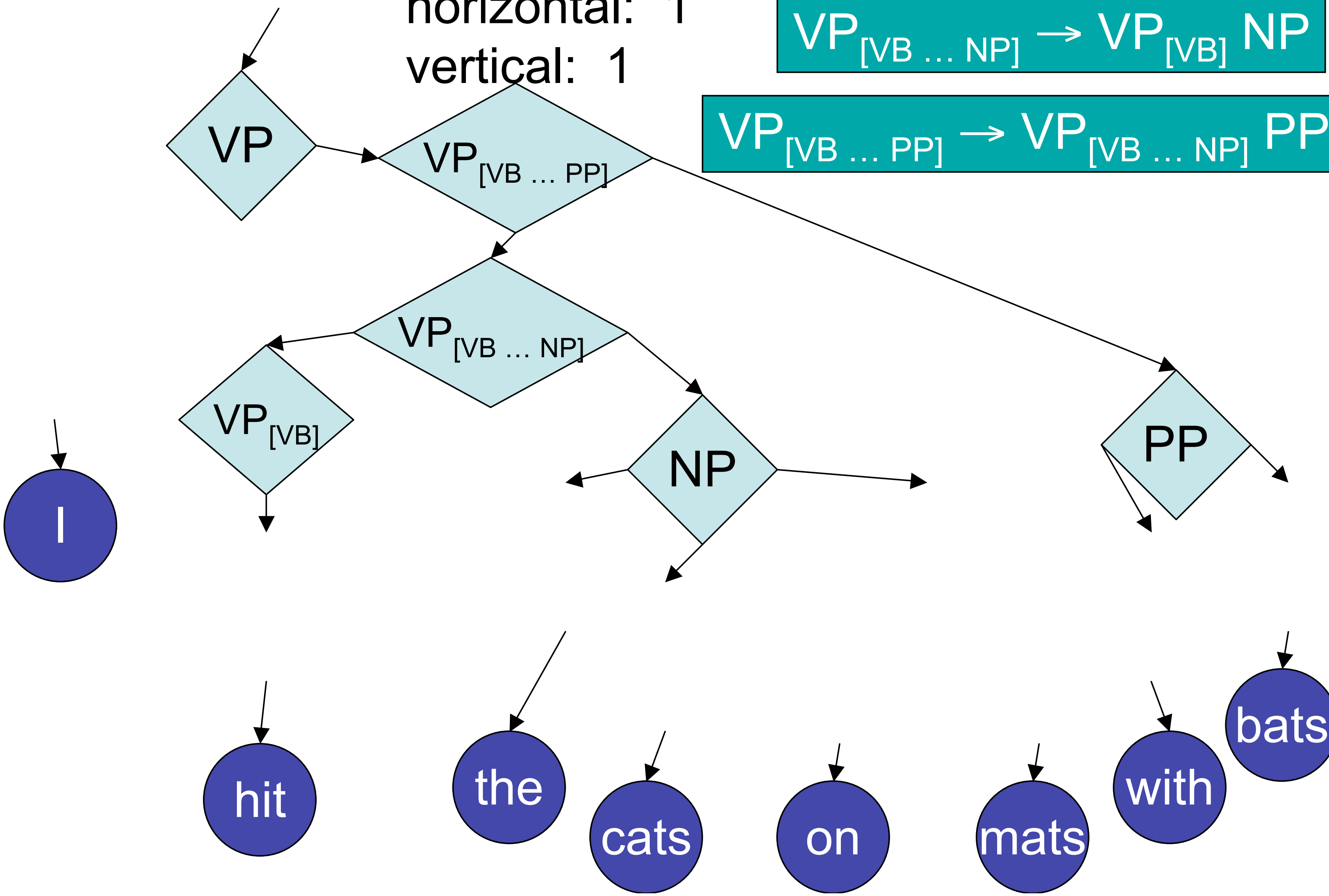
Markovization

horizontal: 1
vertical: 1

$VP_{[VB]} \rightarrow VB$

$VP_{[VB \ldots NP]} \rightarrow VP_{[VB]}\ NP$

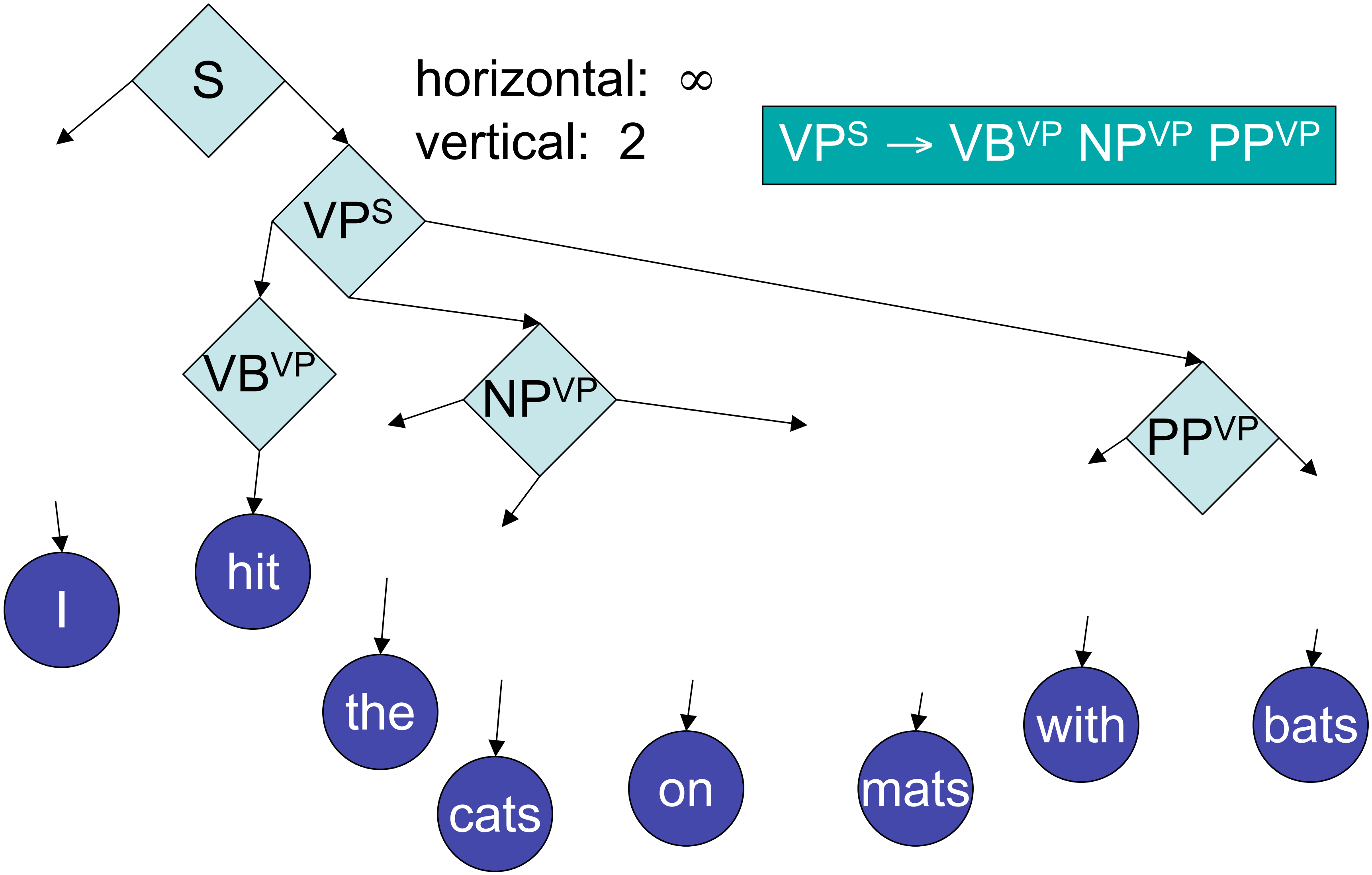
$VP_{[VB \ldots PP]} \rightarrow VP_{[VB \ldots NP]}\ PP$



Markovization

horizontal: ∞
vertical: 2

$VP^{S} \rightarrow VB^{VP}\ NP^{VP}\ PP^{VP}$



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 Annotations

- 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 WFSAs, 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)

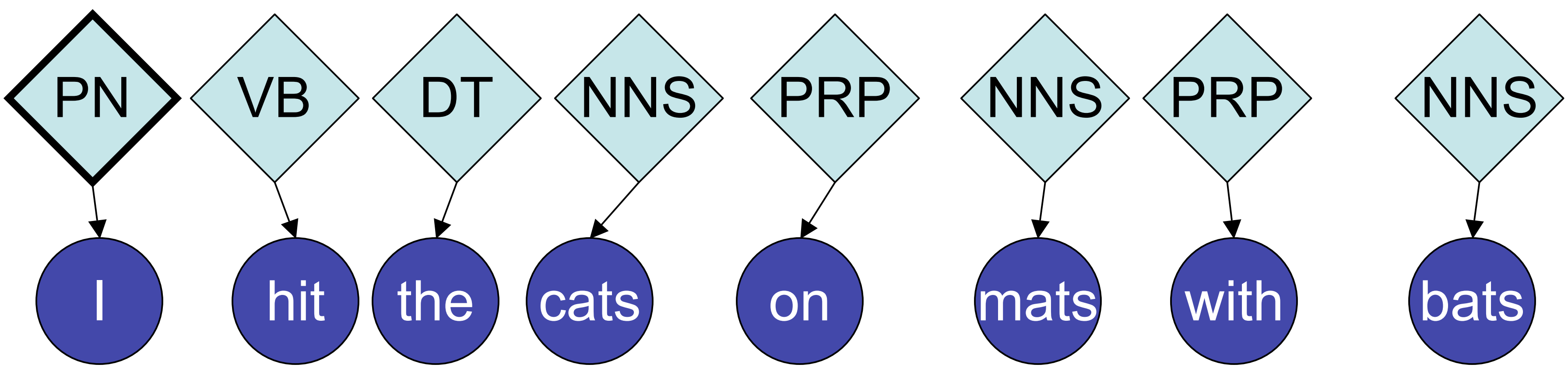
- Tagging, then chunking, then parsing (3 passes)
- Log-linear model: p(next action | history)
  - Features include lots of context, the CFG rule, words, tags, etc.
- Beam search
- Results:
  - O(n) observed runtime!
  - A little worse on performance than Collins Model 1.

- See also: Magerman (1995; decision trees); Chelba & Jelinek (1998; MLE); Sagae & Lavie (2005, SVMs); Nivre et al. (2006; SVMs)

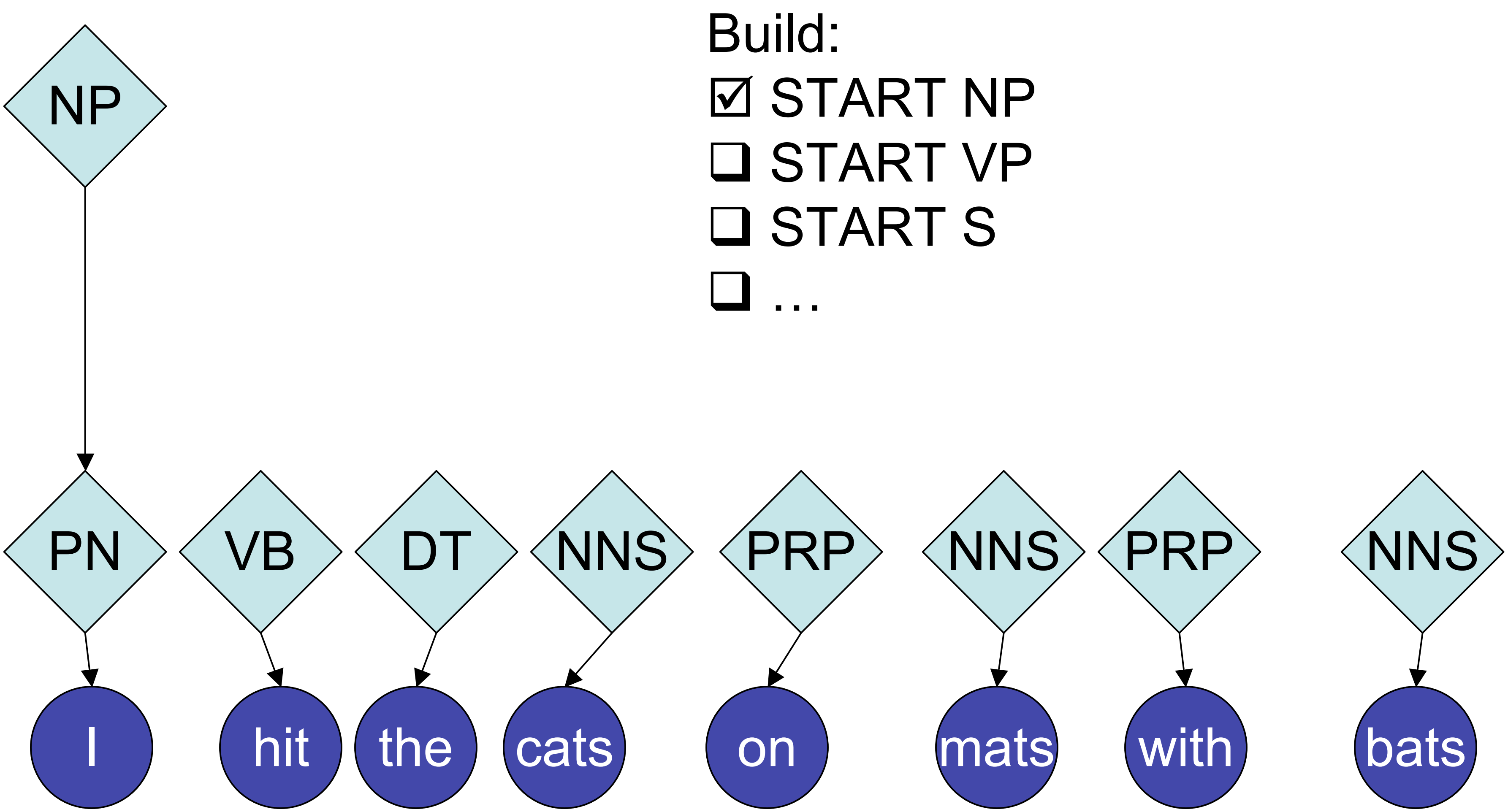
Ratnaparkhi (1998)

Build:

- ❑ START NP
- ❑ START VP
- ❑ START S
- ❑ …

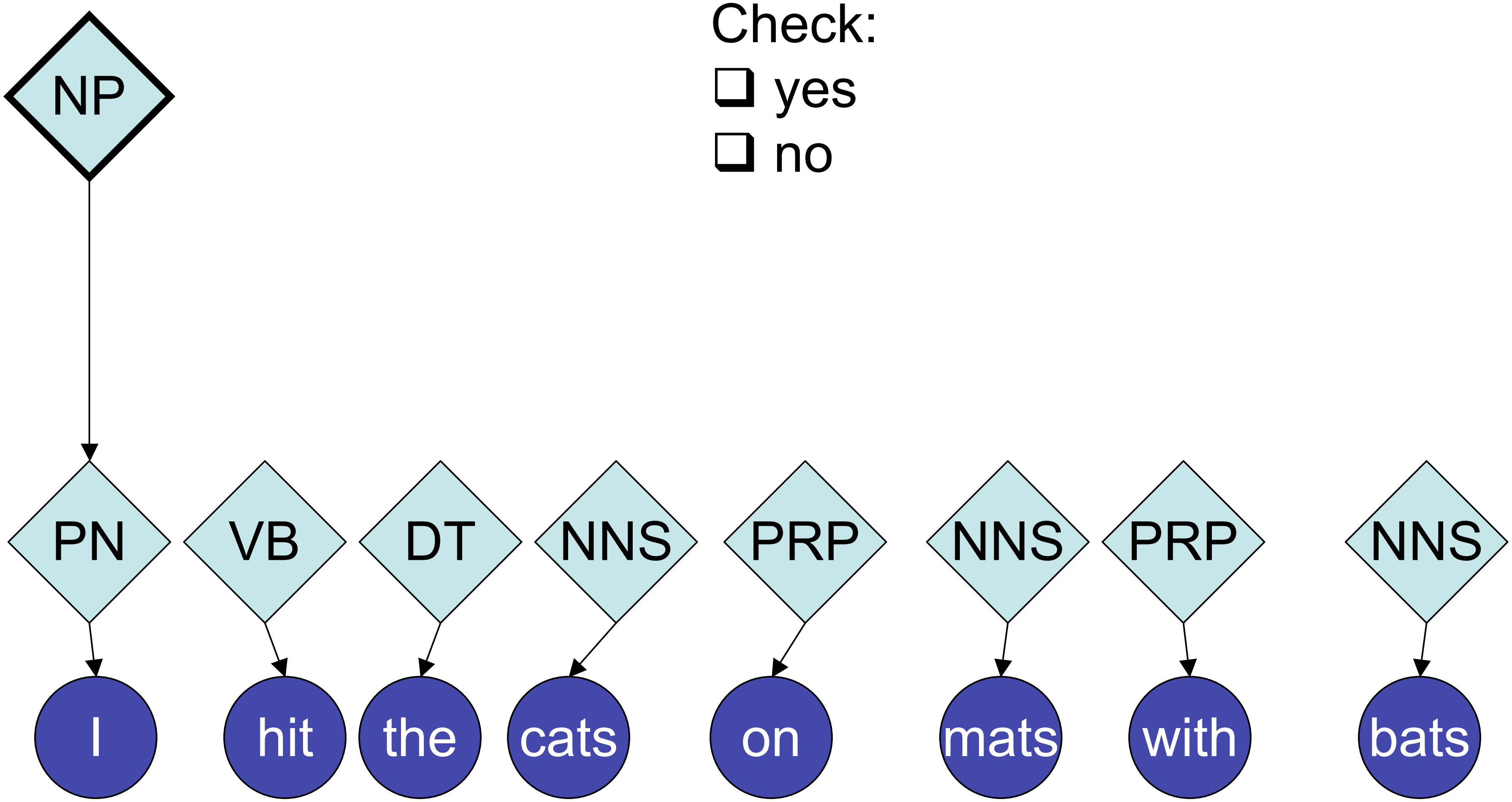


Ratnaparkhi (1998)



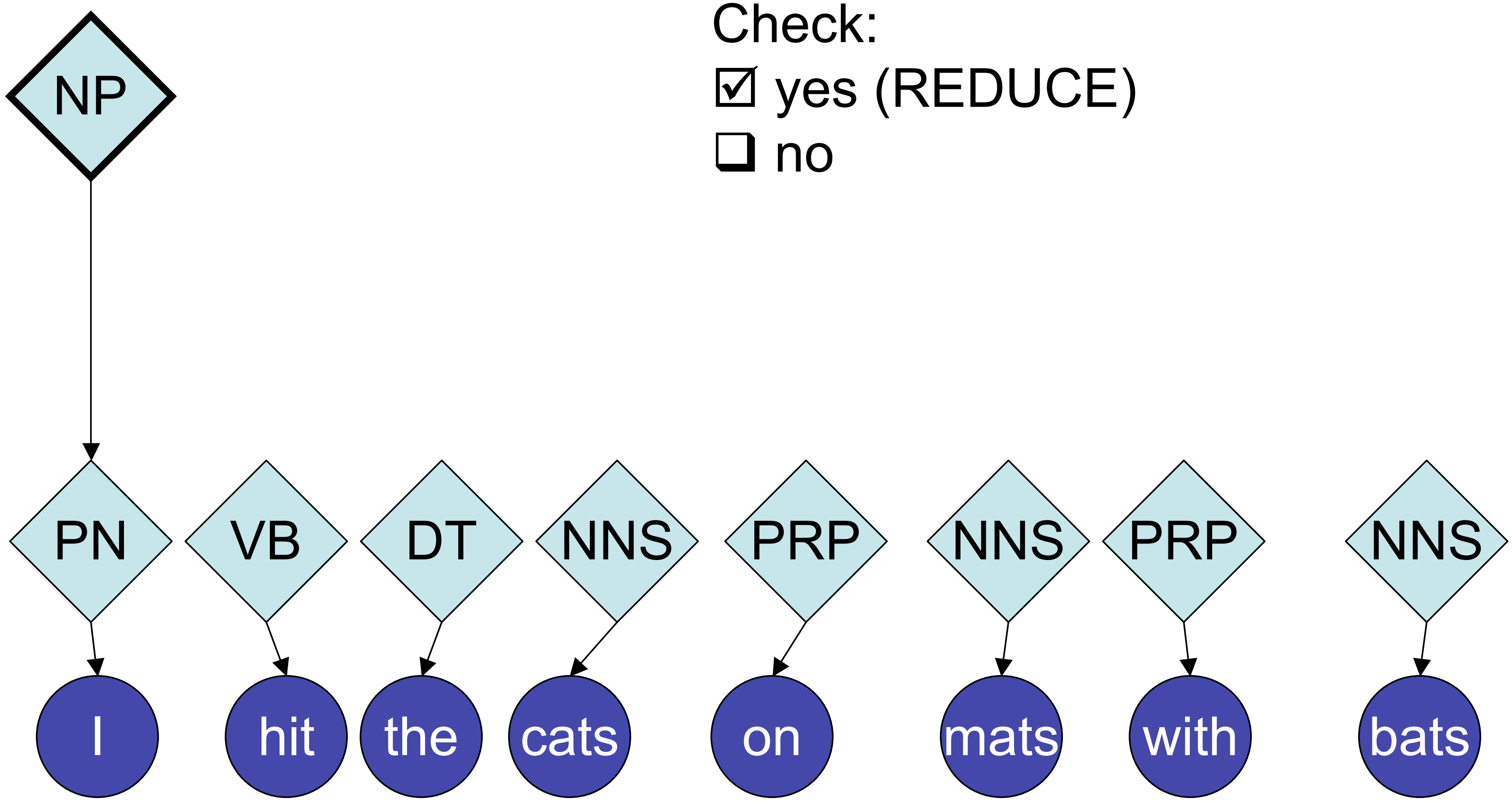
Ratnaparkhi (1998)

Check:
- ❑ yes
- ❑ no

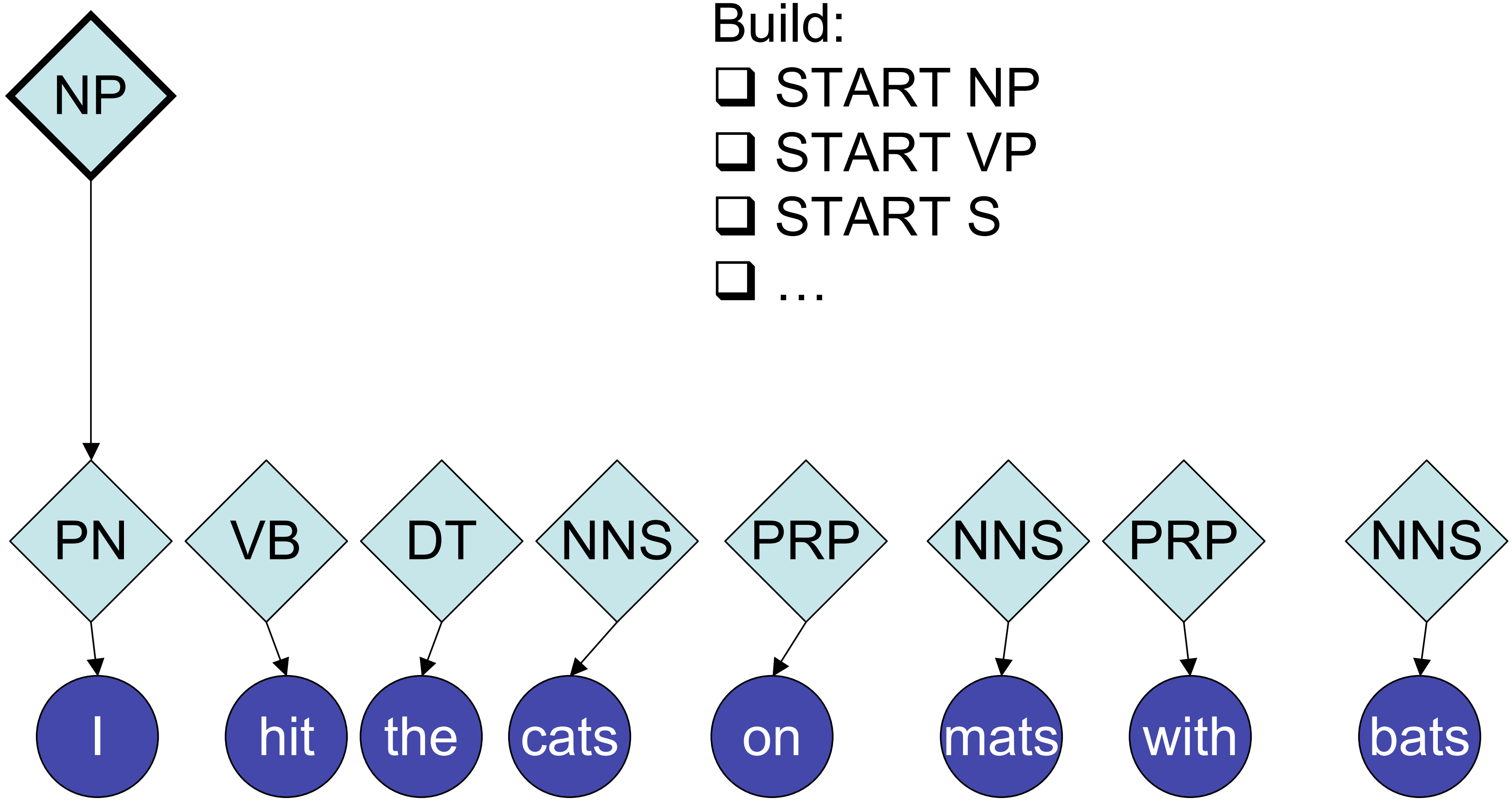


Ratnaparkhi (1998)

Check:
☑ yes (REDUCE)
❑ no



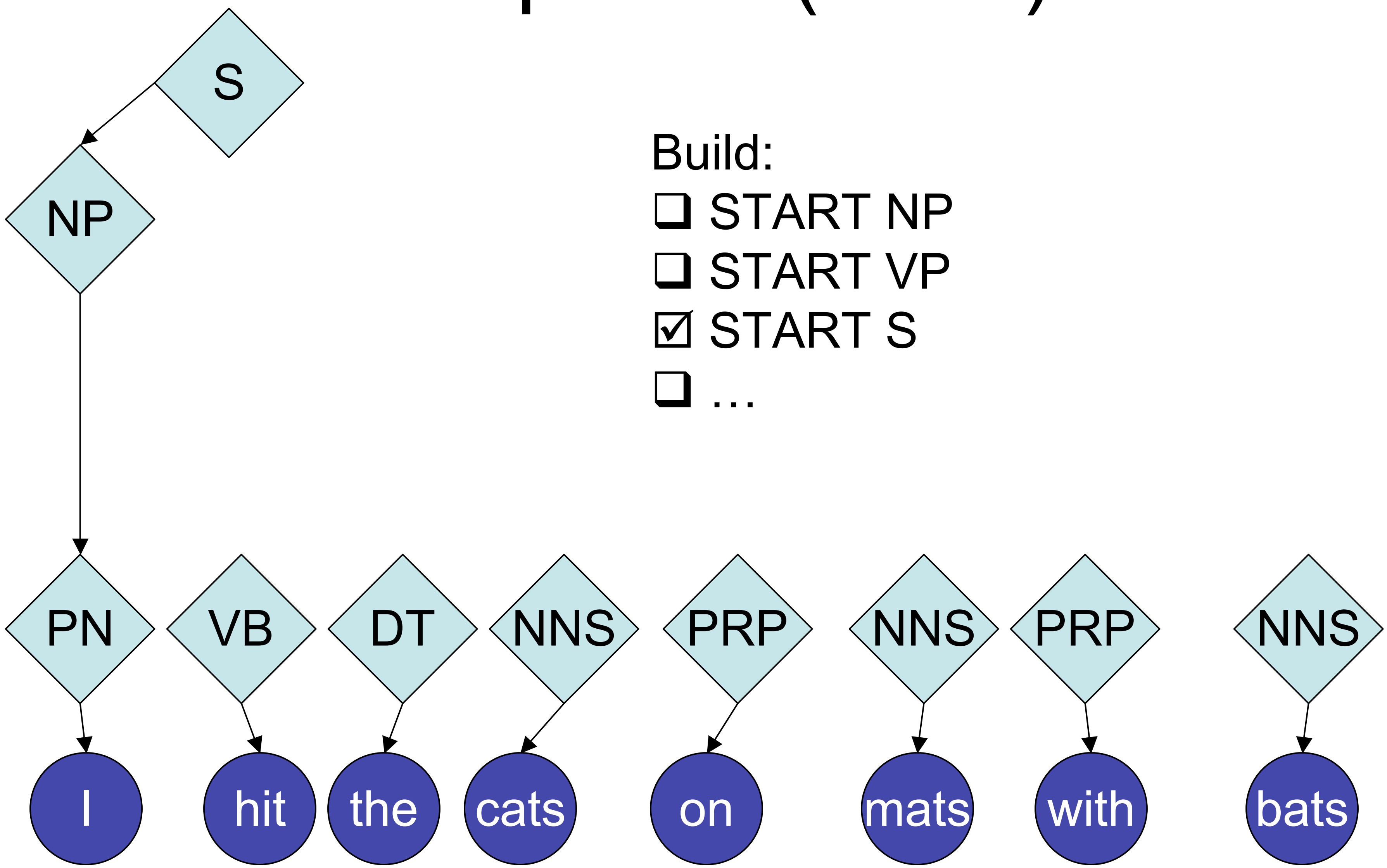
Ratnaparkhi (1998)



Build:

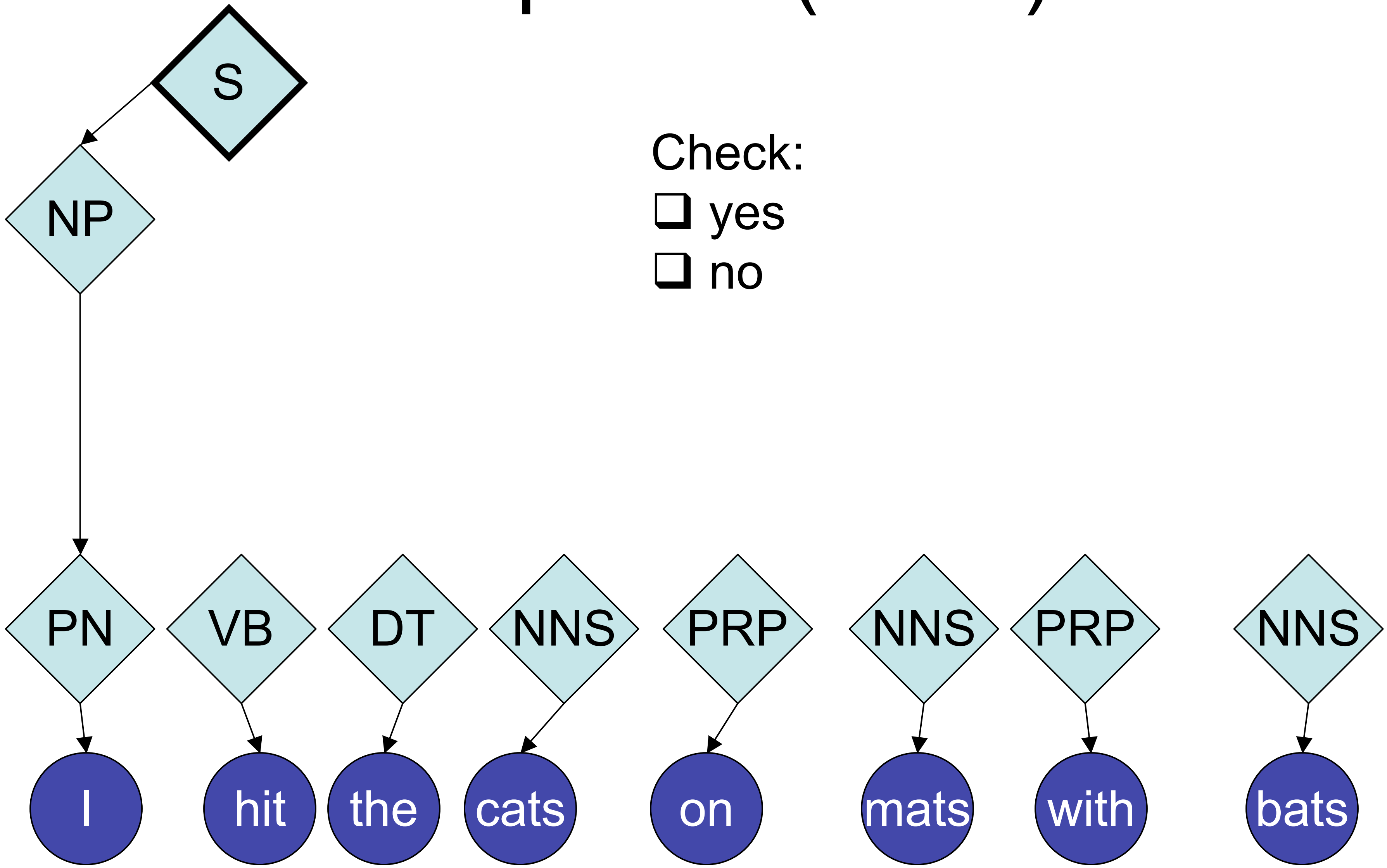
- ❑ START NP
- ❑ START VP
- ❑ START S
- ❑ …

Ratnaparkhi (1998)



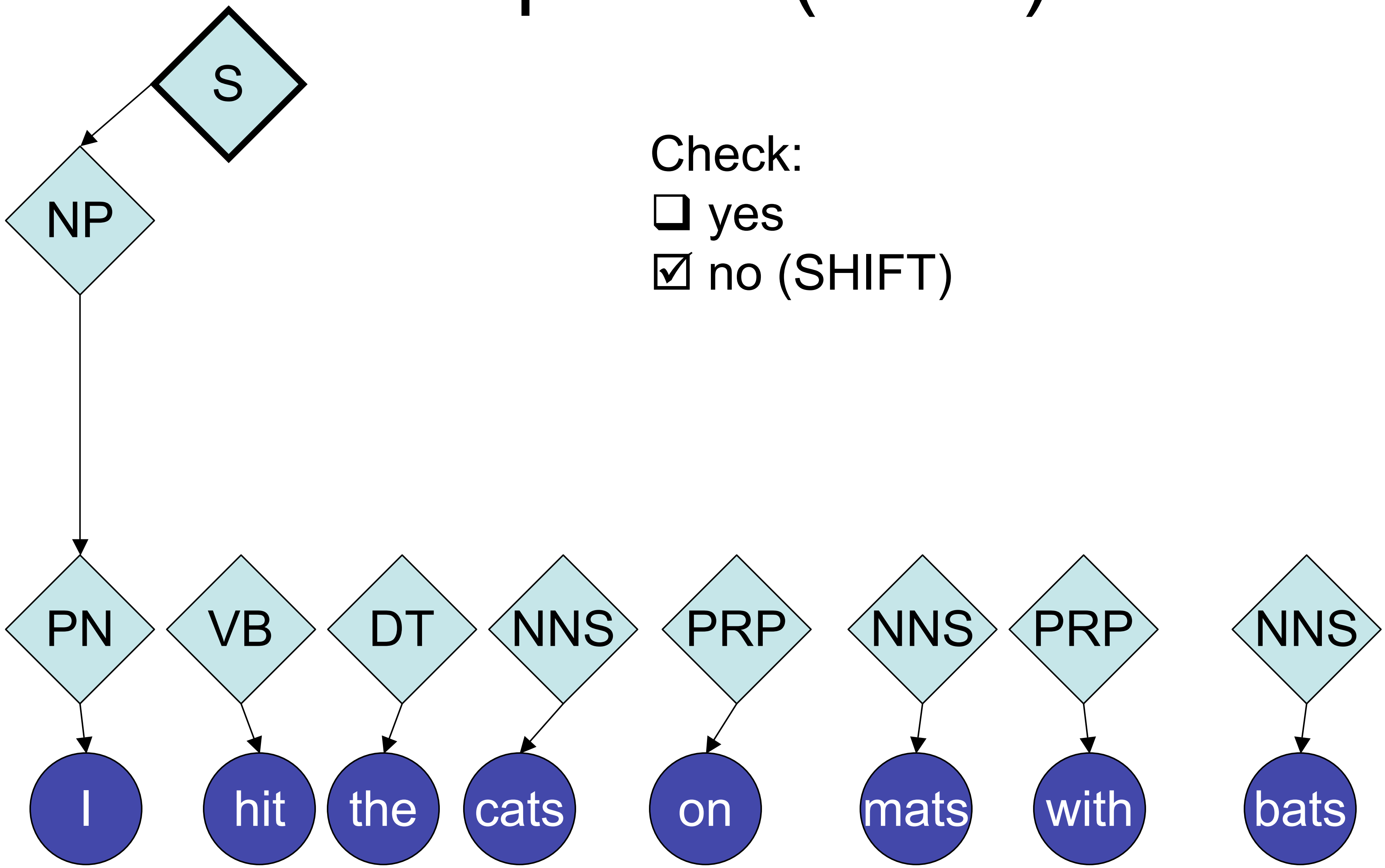
Build:
- ☐ START NP
- ☐ START VP
- ☑ START S
- ☐ …

Ratnaparkhi (1998)



Check:
- ❑ yes
- ❑ no

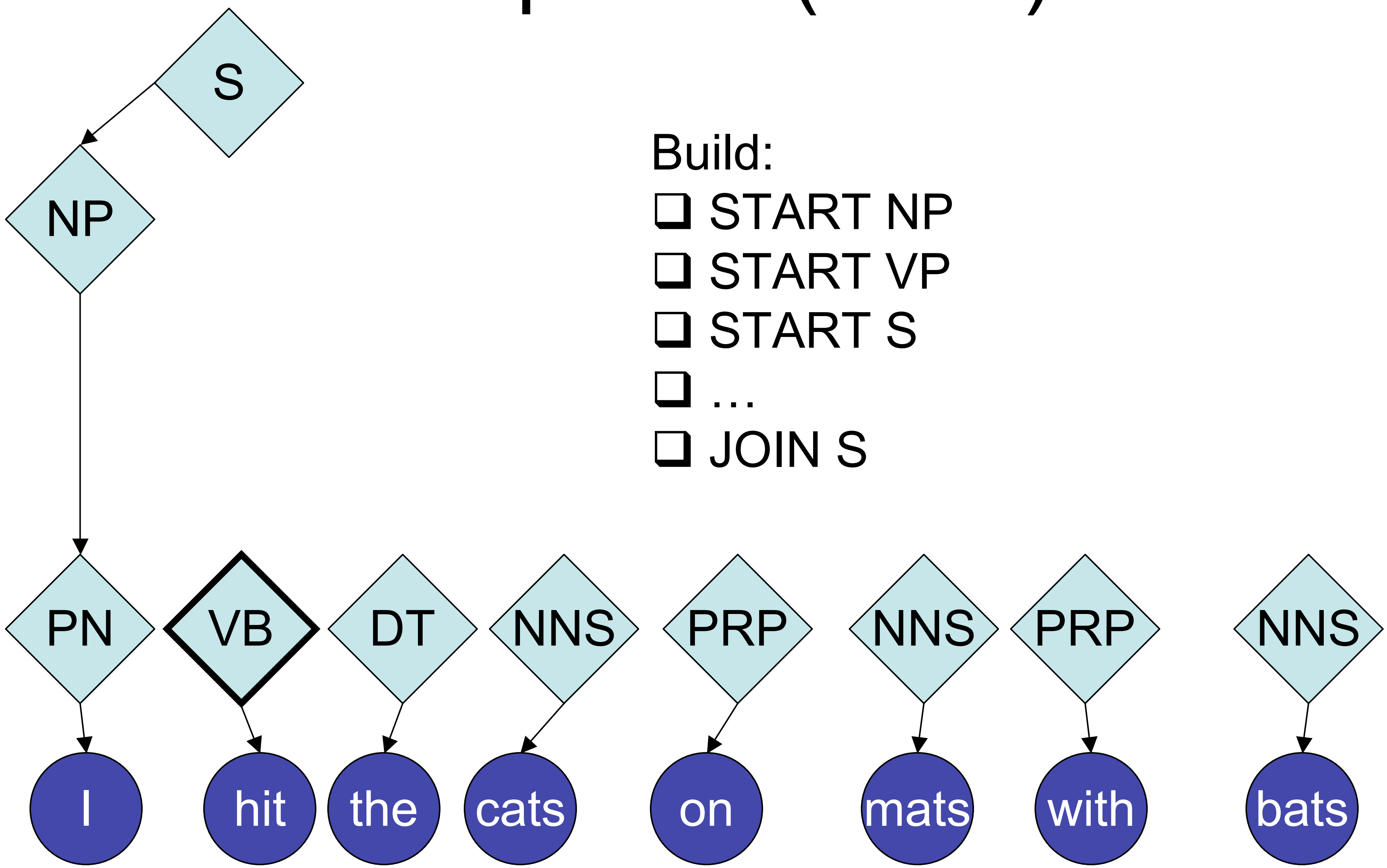
Ratnaparkhi (1998)



Check:

- ❑ yes
- ☑ no (SHIFT)

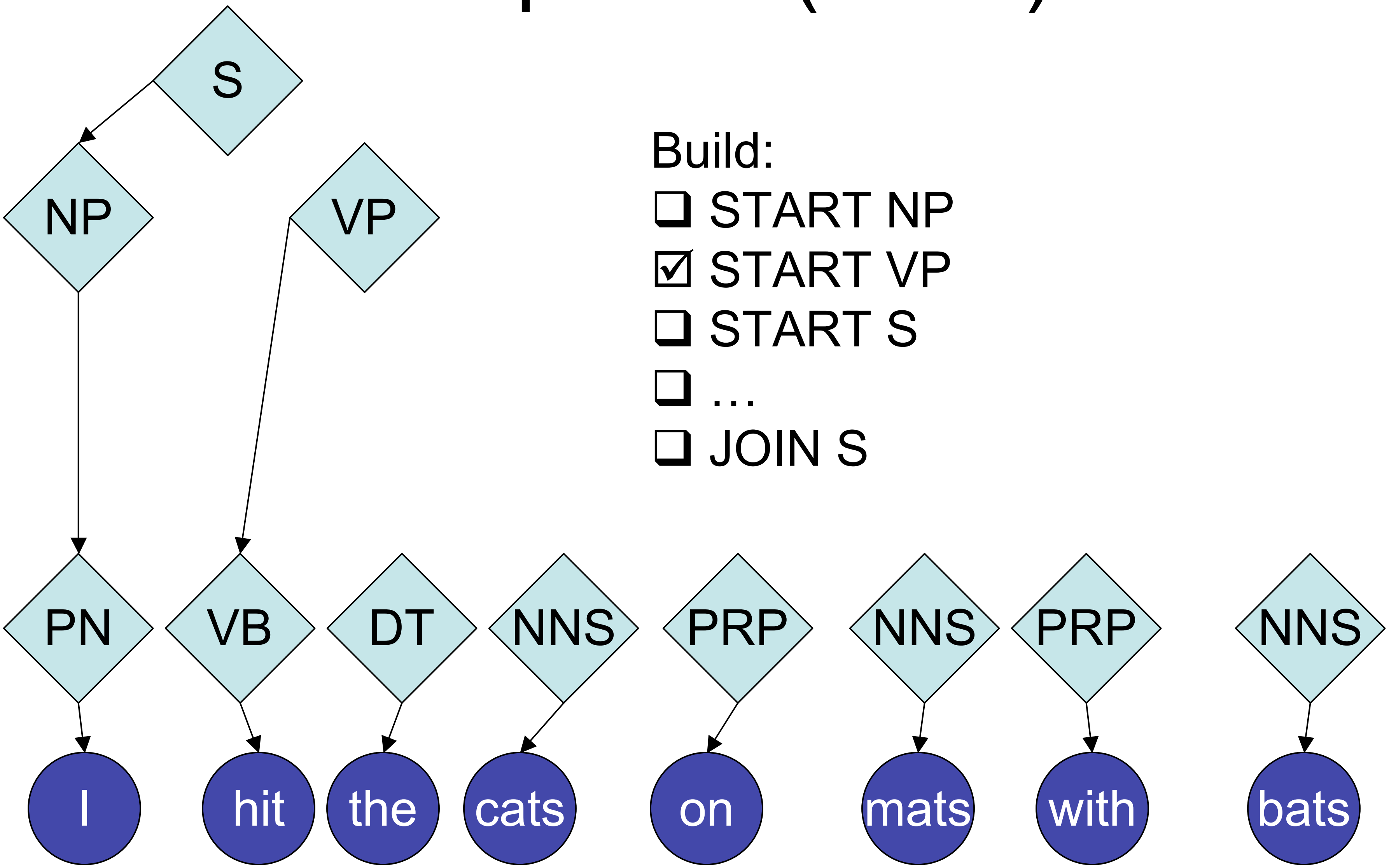
Ratnaparkhi (1998)



Build:

- START NP
- START VP
- START S
- …
- JOIN S

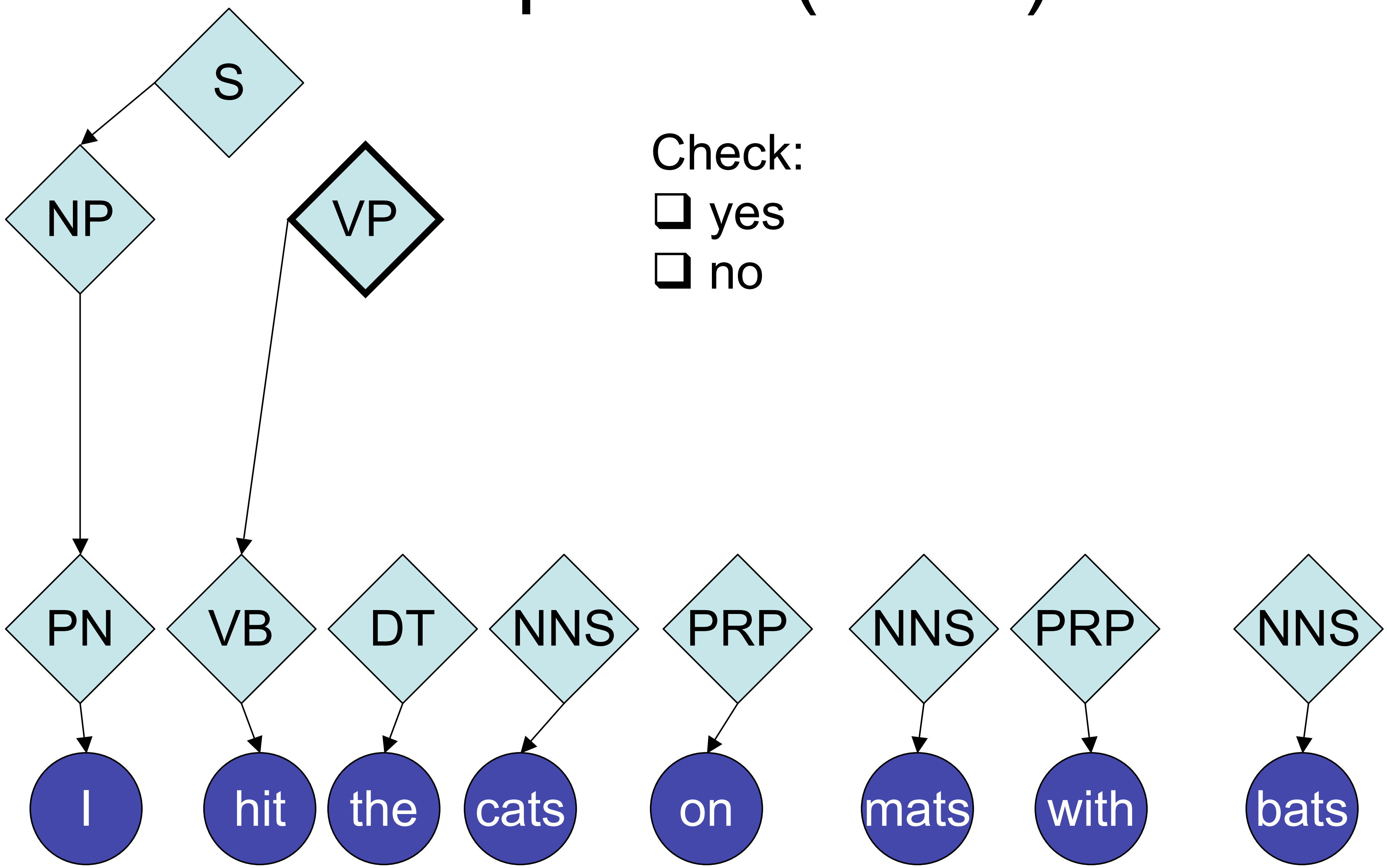
Ratnaparkhi (1998)



Build:

- [ ] START NP
- [x] START VP
- [ ] START S
- [ ] …
- [ ] JOIN S

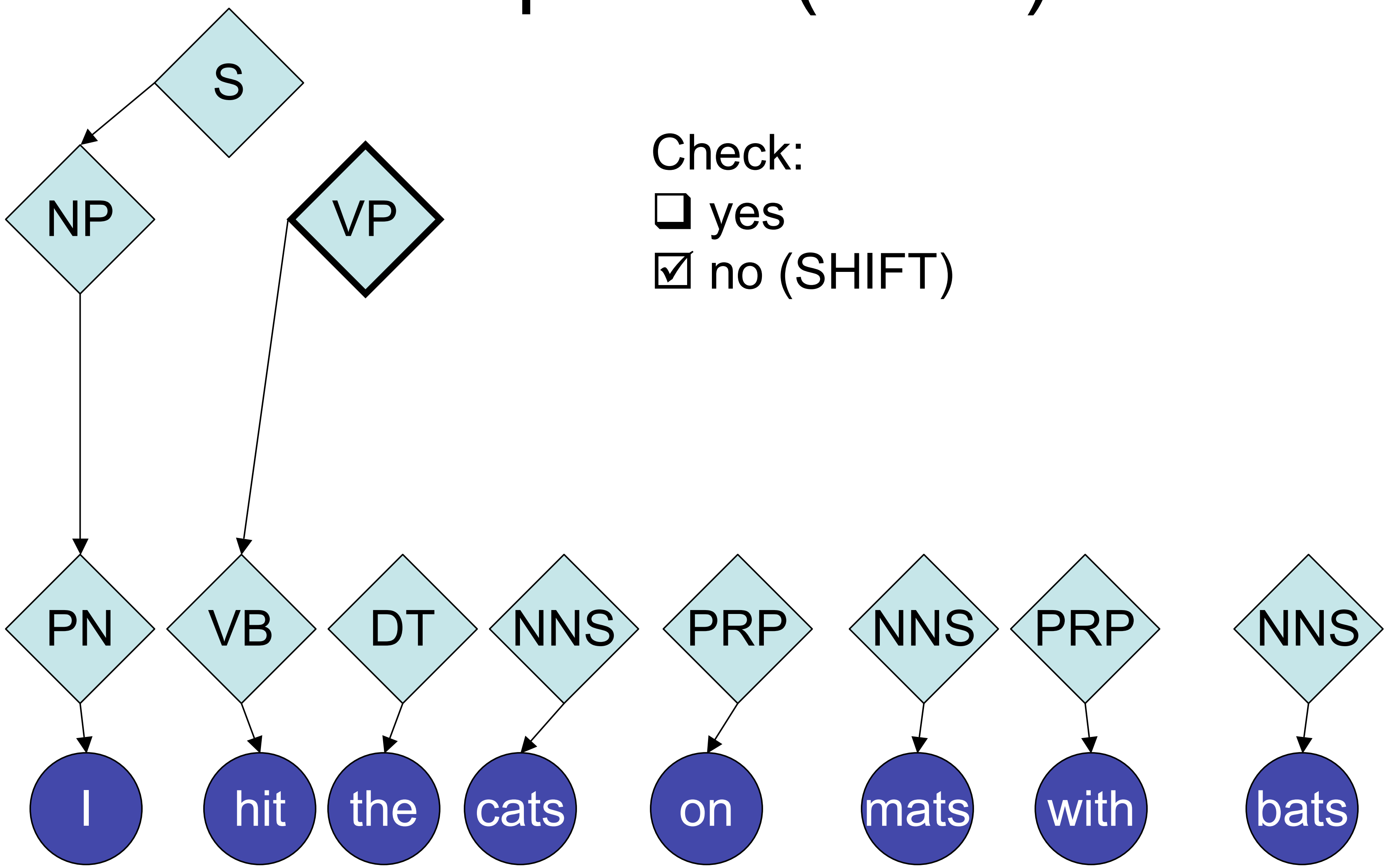
Ratnaparkhi (1998)



Check:

- ❑ yes
- ❑ no

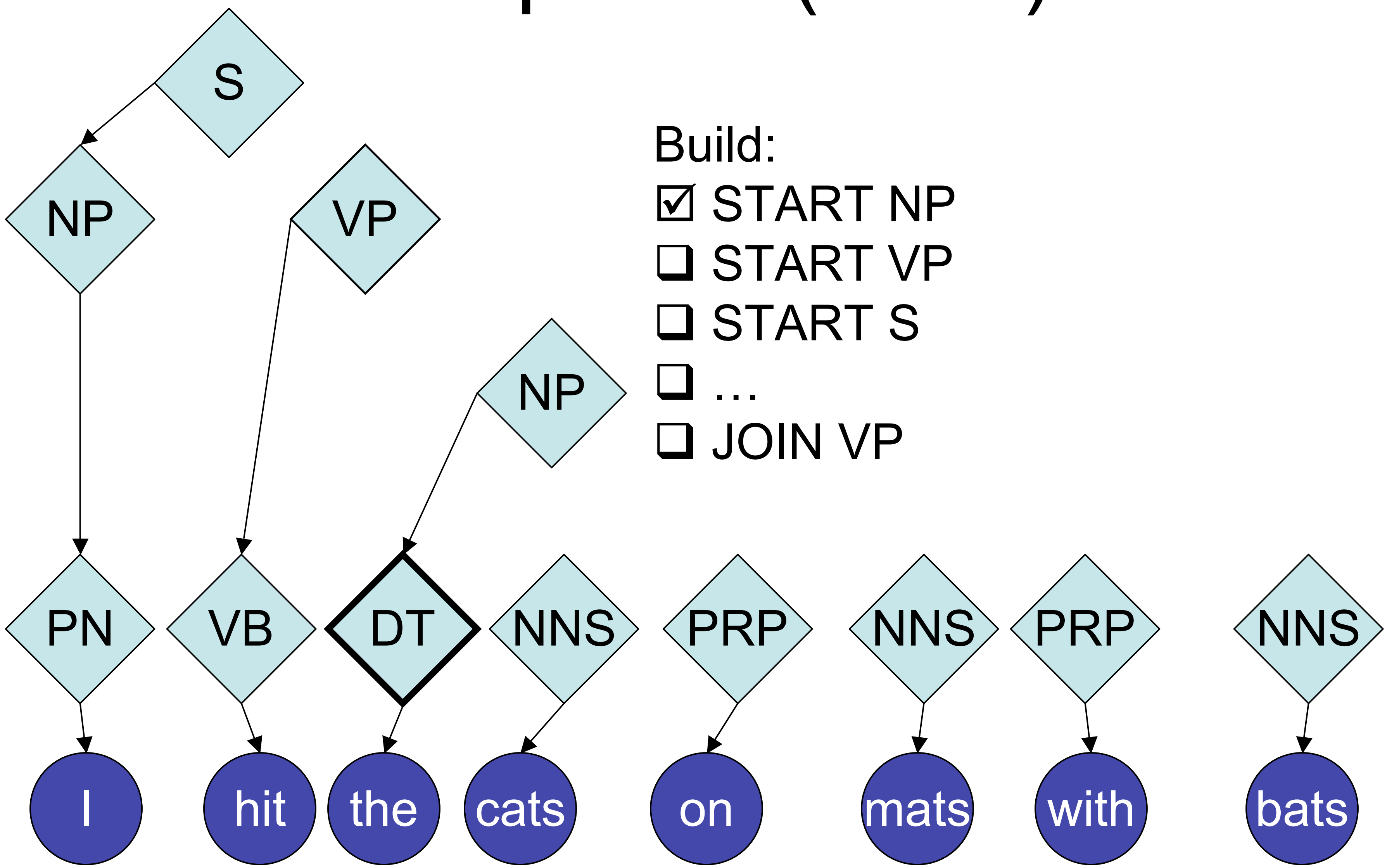
Ratnaparkhi (1998)



Check:

- ❑ yes
- ☑ no (SHIFT)

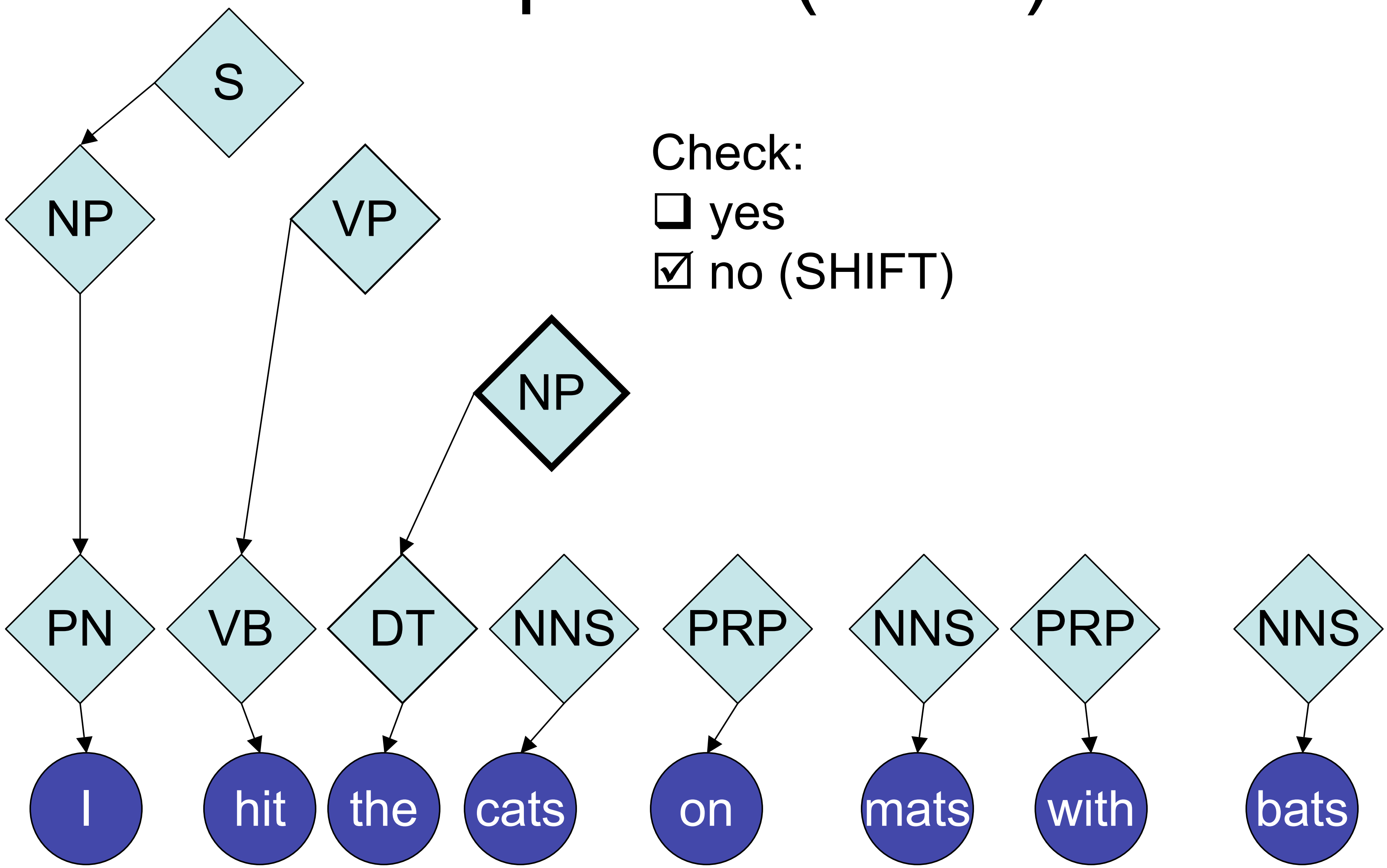
Ratnaparkhi (1998)



Build:

- ☑ START NP
- ☐ START VP
- ☐ START S
- ☐ …
- ☐ JOIN VP

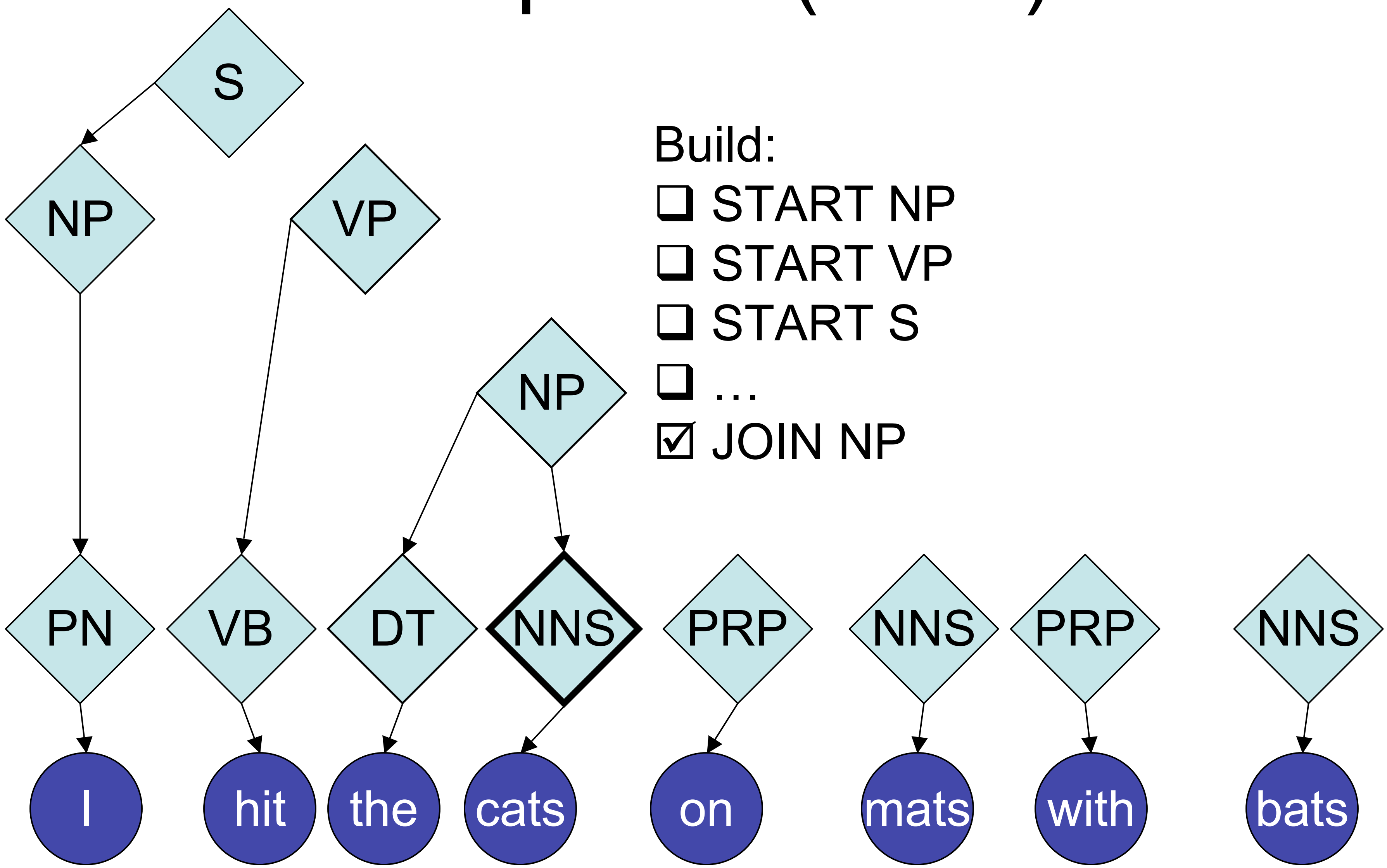
Ratnaparkhi (1998)



Check:

- ❑ yes
- ☑ no (SHIFT)

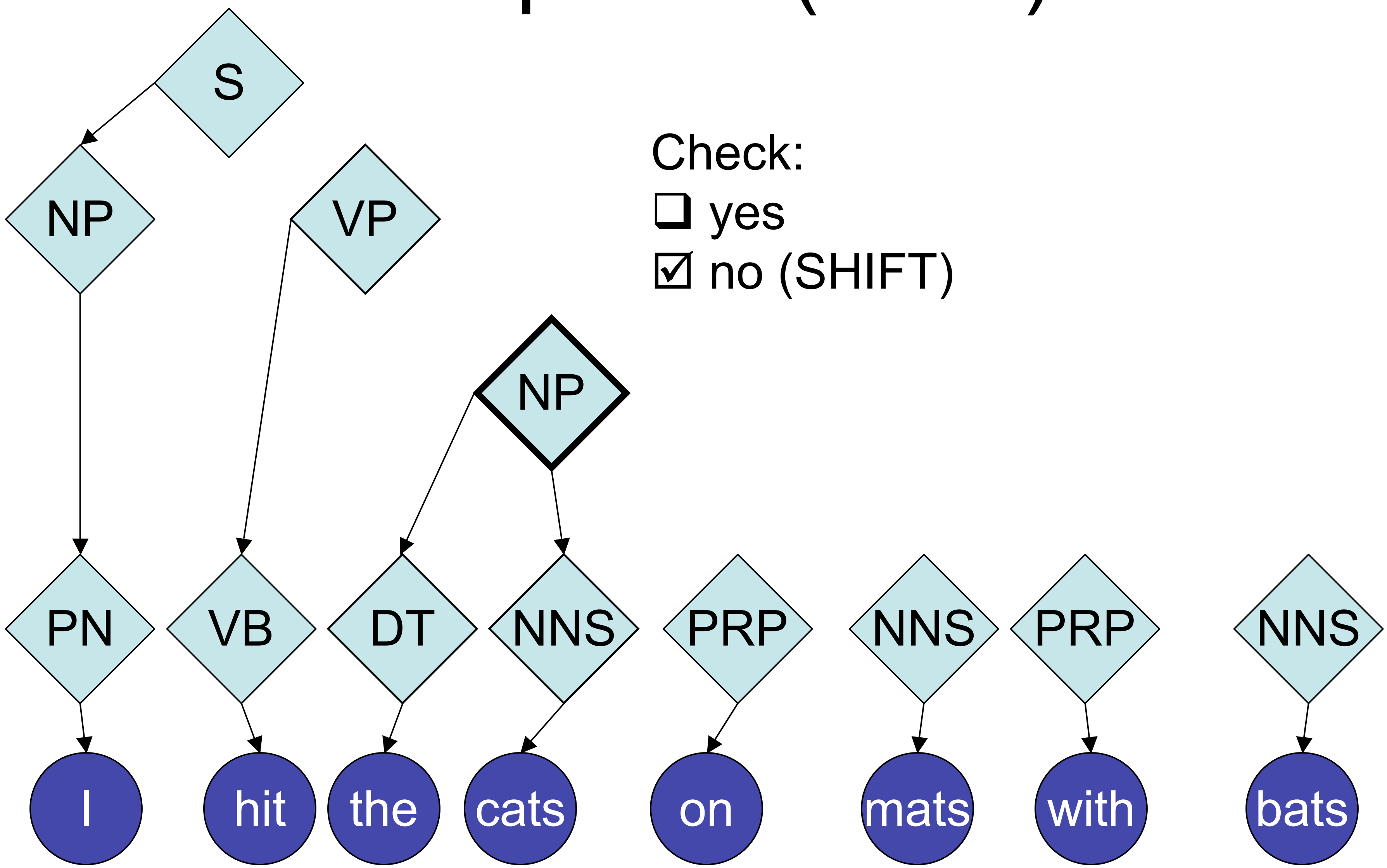
Ratnaparkhi (1998)



Build:

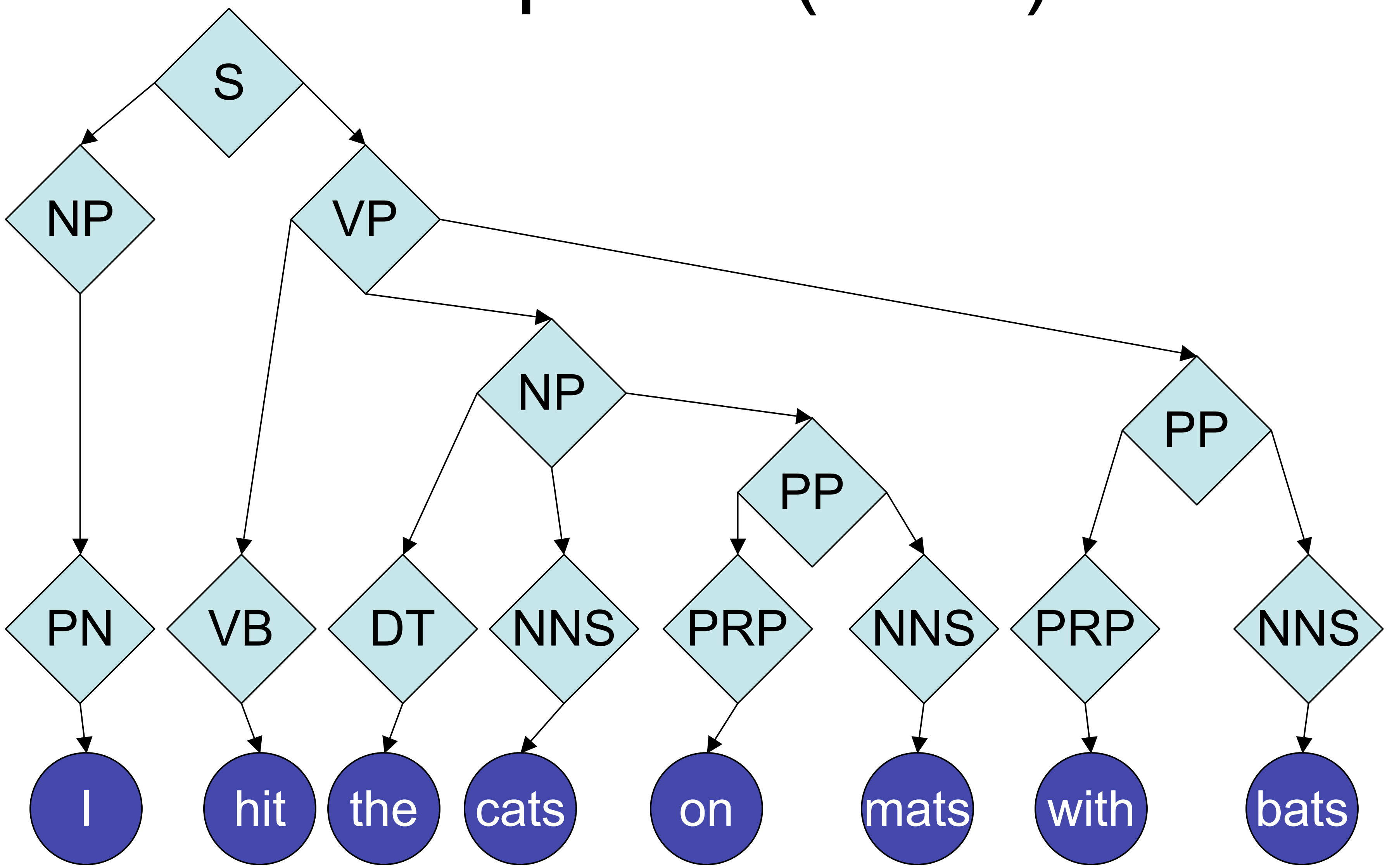
- ❑ START NP
- ❑ START VP
- ❑ START S
- ❑ …
- ☑ JOIN NP

Ratnaparkhi (1998)

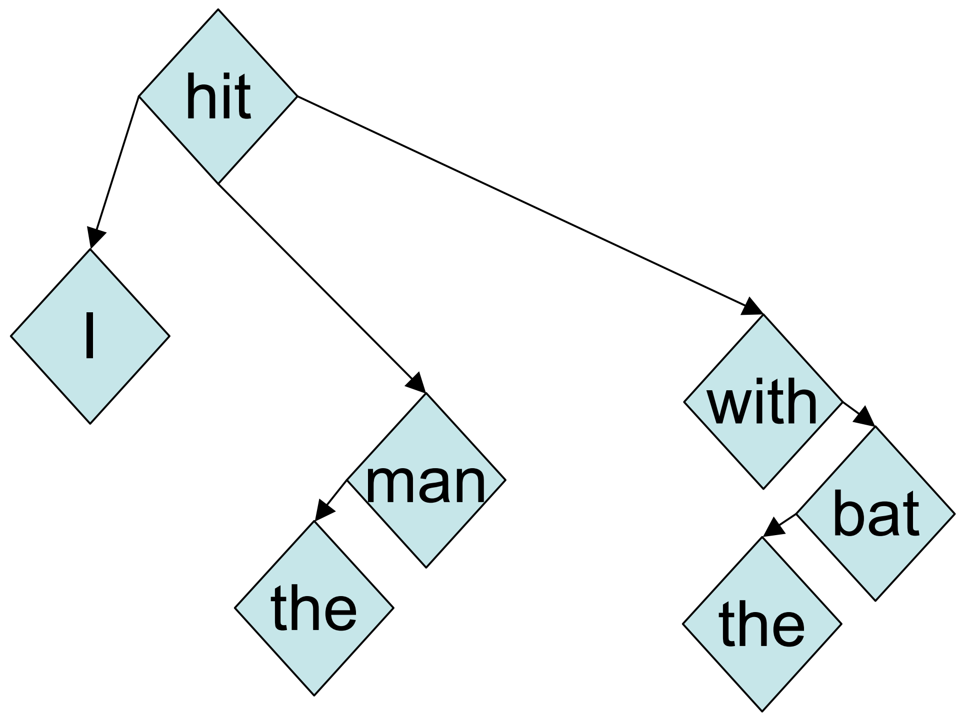


Check:
❑ yes
☑ no (SHIFT)

Ratnaparkhi (1998)



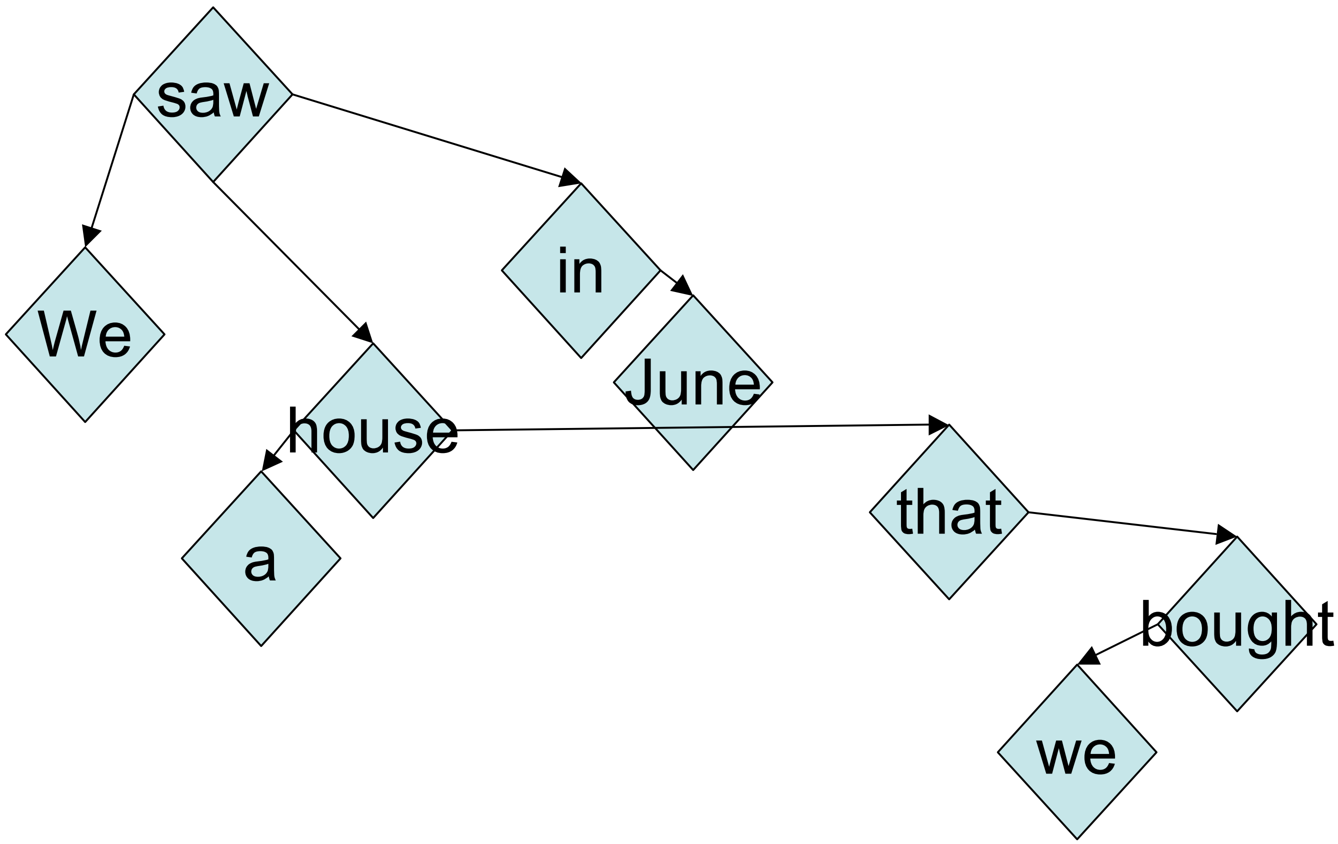
Dependencies



Dependency Parsing

- Very influential in structural and European linguistics
  - Tesniere (1959); Mel'cuk (1988), *inter alia*
  - Captures lexical relationships easily
  - Projective dependency grammar is context-free (Gaifman, 1965)
- Link Grammar (Sleator and Temperley, 1992); later made probabilistic
  - Syntax is an **undirected** planar graph, possibly cyclic!
  - Cubic-time parsing
- Evaluation: Lin (1995) - attachment accuracy
- Generative model: Eisner (1996)
  - Simple, projective dependency grammars parseable in cubic time
  - Several models presented, most notably the **recursive generation model**, which arguably inspired the generative model in Collins (1997)
- 2006: CoNLL shared task (13 languages!)

Nonprojective Dependencies



Nonprojective Dependency Parsing

- Arguably really important for some languages
  - Free word order (Czech)
  - Crossing dependencies (Dutch?)
- McDonald et al., 2005: nonprojective parsing is a **minimum-cost spanning tree** problem!
  - Need to generalize to **directed** trees.
- Cost of a tree = sum of edge costs
- Independence assumptions?
- State-of-the-art for many languages when trained **discriminatively**. (We'll come back to this!)
- Later added second-order features (two edges) and approximate search algorithm (optimal is NP-hard).

(Mild) Context Sensitivity

Many more expressive formalisms have been made probabilistic:

- Tree Adjoining Grammar (Resnik, *inter alia*)
- Lexical-Functional Grammar (Riezler, *inter alia*)
- Tree Insertion Grammar (Hwa)
- Combinatory Categorial 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!

Wait, I'm Confused!

- We will come back to all this "discriminative" training stuff.
- For now, the key message is:
  - Parsing is harder than anyone thought it would be.
  - All kinds of tradeoffs: sparseness, independence assumptions, speed