

# A CKY Parser for English with Agreement Checking

Atahan Uz and Gizem Yılmaz

Boğaziçi University

Department of Computer Engineering

34342 Bebek, Istanbul, Turkey

{atahan.uz, gizem.yilmaz1}@std.bogazici.edu.tr

## Abstract

Constituency parsing requires recovering hierarchical phrase structure while simultaneously enforcing grammatical constraints, a combination that proves difficult for standard context-free grammars. This paper describes a CKY-based parser that integrates agreement validation directly into chart construction, blocking ill-formed constituents before they propagate upward. Our grammar comprises 206 production rules built on Penn Treebank part-of-speech tags, which we convert to Chomsky Normal Form for bottom-up processing. The parser enforces two constraint types: number agreement between determiners and nouns (rejecting “\*a books”), and person/number agreement between subjects and present-tense verbs (rejecting “\*I buys”). Evaluation on 32 sentences spanning declarative, imperative, and interrogative structures yields 90.9% labeled F1-score against manually annotated gold trees.

## 1 Introduction

Syntactic parsing aims to recover the hierarchical constituent structure underlying natural language sentences. But parsing English involves more than identifying noun phrases and verb phrases; it demands enforcement of the grammatical constraints that govern how words combine. Consider the determiner-noun pair “a book”: grammatical. However a minor swap to a plural noun, then you get “\*a books,” determiner-noun pair which is grammatically incorrect (ungrammatical). Present-tense verbs show parallel restrictions. “She runs” demonstrates correct third-person agreement, while “\*she run” violates it. Any parser that claiming coverage of English must handle these agreement phenomena.

What makes agreement difficult for context-free approaches is that CFGs lack the expressive power to encode constraints like “singular determiner requires singular noun” without multiplying non-terminal categories. One strategy introduces agreement-indexed symbols (NP\_sg and NP\_pl, for example), but the grammar size explodes combinatorially when multiple features interact. The alternative is validating constraints in a post-processing step, which works but wastes computation on analyses destined to fail.

Our approach checks for the agreement at the moment the constituents attempt to combine. Because the CKY algorithm builds parse trees using bottom-up approach via dynamic programming (Jurafsky and Martin, 2024), we can intercept ill-formed combinations before they enter the chart. When a singular determiner meets a plural noun, the combination is simply discarded. This strategy prevents the error propagation.

The Penn Treebank part-of-speech tags (Marcus et al., 1993) serve as terminal symbols in the project throughout. These tags encode morphological distinctions directly: NN marks singular nouns, NNS marks plurals, VBZ indicates third-person singular present verbs, and VBP covers non-third-singular present. Since spaCy provides these tags during preprocessing, agreement checking reduces to feature comparison when the constituents are combined. The DT is merged with NN, and the parser compares their

In this work the grammar handles three type of sentences: declarative, imperative, yes/no questions and WH questions types. Declarative sentences follow the canonical subject-verb-object order, optionally accompanied by adjuncts, while imperative sentences simply omit the explicit subject “you.” Questions

include both yes/no forms (“Did she leave?”) and wh-forms (“What happened?”). Across these patterns, our CFG contains total of 206 production rules before CNF conversion.

## 1.1 Related Work

Agreement-checking parsers have been explored in several strands of literature. Early lexicalized PCFG approaches such as (Klein and Manning, 2003) incorporated agreement features into the grammar. Constraint-based extensions of CKY, e.g., (Zhang and Xue, 2014), enforce number and person constraints during chart construction. More recent neural models treat agreement as a classification task (Mahabadi et al., 2020). Our work differs by integrating subcategorization frames from VerbNet and performing agreement checks directly within a classic CKY parser, preserving the cubic-time guarantee while handling richer linguistic constraints.

## 2 System Description

Five components form the parsing pipeline: an English CFG using Penn Treebank categories, a CNF converter, the CKY chart-filling algorithm, agreement validation, and a feature-annotated lexicon. The words are given as tagged tokens; when grammatical, they exit as constituent trees. Each component feeds the next, and we describe them in order.

### 2.1 Grammar Design

Penn Treebank POS tags serve as terminals because they provide compatibility with standard English taggers and encode the morphosyntactic features our agreement checker requires. Four phrasal non-terminals represent the major categories: S for sentences, NP for noun phrases, VP for verb phrases, and PP for prepositional phrases. All together, the grammar comprises 206 production rules.

**Sentence-Level Rules** Declarative sentences follow canonical SVO order. The subject NP precedes the VP, with optional adjuncts appearing finally:

```
S -> NP VP
S -> NP VP PP
S -> NP VP RB
```

These patterns generate sentences like “I bought a book” and “I bought a book for my friend yesterday.” When prepositional phrases or adverbs modify the entire predication rather than a specific constituent, they attach at the sentence level.

Imperatives omit the subject entirely. The VP stands alone, sometimes preceded by an interjection:

```
S -> VP
S -> UH VP
```

Yes/no questions invert auxiliary and subject, placing modals or auxiliary verbs before the subject NP:

```
S -> MD NP VP
S -> VBZ NP VP
S -> VBD NP VP
```

“Will you come?” and “Did she leave?” follow these templates. Wh-questions front the interrogative word:

```
S -> WRB VBD NP VP
S -> WP VP
```

“When did you arrive?” positions the wh-adverb initially; “Who called?” uses the wh-word itself as subject.

**Phrasal Rules** Noun phrases range from bare pronouns to heavily modified structures. Determiners combine with nouns, optionally modified by adjectives; superlatives add degree adverbs:

NP  $\rightarrow$  PRP  
 NP  $\rightarrow$  DT NN  
 NP  $\rightarrow$  DT JJ NN  
 NP  $\rightarrow$  DT RBS JJ NN  
 NP  $\rightarrow$  NP PP  
 NP  $\rightarrow$  NP CC NP

PPs attach recursively (“the book on the table in the corner”), and coordination joins phrases with “and” or “or.”

Verb phrases accommodate intransitive, transitive, and ditransitive argument structures:

VP  $\rightarrow$  VB  
 VP  $\rightarrow$  VBD NP  
 VP  $\rightarrow$  VBZ NP PP  
 VP  $\rightarrow$  MD VB NP  
 VP  $\rightarrow$  VBP RB VP

The last pattern handles negation constructions like “do not listen,” where an auxiliary precedes “not” and embeds another VP.

Prepositional phrases pair a preposition with its NP complement:

PP  $\rightarrow$  IN NP  
 PP  $\rightarrow$  TO NP

## 2.2 Chomsky Normal Form Conversion

CKY parsing requires binary rules: every production must contain exactly two non-terminals on the right-hand side, or exactly one terminal. Converting arbitrary CFGs to this form involves five sequential transformations.

**Step 1: New Start Symbol** When  $S$  appears on some rule’s right-hand side, we introduce a fresh symbol  $S_0$  with the single production  $S_0 \rightarrow S$ . This prevents recursion complications during subsequent steps.

**Step 2: Epsilon Elimination** Nullable variables (non-terminals capable of deriving empty strings) are identified through fixed-point iteration. Once found, rules containing nullable symbols spawn alternative versions with those elements present or absent. Pure epsilon productions then vanish from the grammar.

**Step 3: Unit Production Removal** Productions like  $A \rightarrow B$ , where  $B$  is a single non-terminal, create chains that CKY cannot process directly. We compute transitive closures to determine which non-terminals eventually reach substantive rules, then replace chains with direct productions.

**Step 4: Terminal Isolation** Mixed rules (those combining terminals with non-terminals) pose problems for the CKY lookup scheme. For each terminal  $a$  appearing in such rules, we create  $T_a \rightarrow a$  and substitute  $T_a$  wherever the terminal appeared.

**Step 5: Binarization** Rules with three or more right-hand symbols are split into cascades of binary rules.  $A \rightarrow BCD$  becomes:

$$\begin{aligned}
 A &\rightarrow B Y \\
 Y &\rightarrow C D
 \end{aligned}$$

Longer rules require additional auxiliary non-terminals. After all five steps, the grammar size roughly triples.

## 2.3 CKY Parsing Algorithm

A triangular chart (essentially a 2D lookup table indexed by span boundaries) drives the algorithm. Cell  $[i, j]$  stores every non-terminal capable of deriving the substring from position  $i$  through  $j$ . Dynamic programming ensures efficiency: each substring gets computed exactly once.

**Initialization** The diagonal cells  $[i, i]$  represent single-word spans. SpaCy assigns each word a POS tag during preprocessing, and this populates the diagonal. Any non-terminal reaching the tag via terminal rules ( $X \rightarrow \text{NN}$ , for instance) also enters the cell.

**Span Extension** Span length grows from 2 to  $n$ , with the algorithm trying every possible split point  $k$  for each cell  $[i, j]$ . When non-terminal B appears in  $[i, k]$  and C appears in  $[k + 1, j]$ , every rule  $A \rightarrow BC$  adds A to  $[i, j]$ . Reverse lookup tables accelerate matching: right-hand-side pairs map directly to their possible left-hand-side symbols.

---

**Algorithm 1** CKY Chart Filling

---

```

1: for  $i \leftarrow 0$  to  $n - 1$  do
2:   populate  $[i, i]$  from POS tags
3: end for
4: for span  $\leftarrow 2$  to  $n$  do
5:   for  $i \leftarrow 0$  to  $n - \text{span}$  do
6:      $j \leftarrow i + \text{span} - 1$ 
7:     for  $k \leftarrow i$  to  $j - 1$  do
8:       for  $B \in [i, k], C \in [k + 1, j]$  do
9:         for rules  $A \rightarrow BC$  do
10:          add A to  $[i, j]$ 
11:        end for
12:      end for
13:    end for
14:  end for
15: end for

```

---

**Backpointers** Chart entries alone cannot reconstruct parse trees. Each entry therefore stores backpointers: the split point  $k$  that produced it, plus the rule that applied. Ambiguous sentences yield multiple backpointers per constituent. Once the start symbol S appears in cell  $[0, n-1]$ , recursive traversal builds all valid parse trees.

## 2.4 Agreement Checking

English imposes morphosyntactic constraints that go beyond phrase structure. Determiners must match nouns in number; subjects must match present-tense verbs in person and number. Our parser validates these constraints at the moment constituents combine, rejecting violations before they enter the chart.

**Determiner-Noun Number** Determiners carry inherent number restrictions. “A” and “an” require singular nouns (NN); “these” and “those” demand plurals (NNS). “The” remains neutral, accepting either.

Each determiner in the lexicon stores a number feature: singular, plural, or underspecified. When DT combines with a noun, the parser compares features. “\*A meeting books” fails because the singular-only determiner cannot accept a plural noun, and the error message pinpoints exactly this mismatch.

**Subject-Verb Person and Number** Present-tense verbs inflect for agreement. Third-singular subjects (“she,” “the cat”) require VBZ forms; other subjects take VBP. Thus “She runs” succeeds while “\*She run” fails.

Pronouns carry explicit person and number features in the lexicon: “I” is first-singular, “they” is third-plural. NPs headed by NN count as third-singular; those headed by NNS count as third-plural. Coordinated subjects (“he and I”) trigger plural agreement regardless of individual conjunct number, a linguistic generalization our feature propagation handles.

When S forms from NP + VP, the parser checks compatibility. “I” (first-singular), combining with “buys” (VBZ, requiring third-singular) raises an agreement error.

**Subcategorization** Verbs also constrain their arguments beyond agreement. Subcategorization frames, extracted from VerbNet (Kipper et al., 2008), specify which argument structures each verb permits. “Go” is intransitive and rejects “\*I went the school” because an NP object appears where none is permitted. “Put” requires a location complement: “\*I put the book” lacks the mandatory PP. The lexicon stores these frames, and the parser consults them during VP construction.

## 2.5 Lexicon and Morphology

For each word, the lexicon maps surface form to POS tag and associated features. Closed-class items (determiners, pronouns, prepositions) appear exhaustively; open-class vocabulary covers our test sentences.

Table 1: Lexicon entries with features

Word	POS	Num	Pers
I	PRP	sg	1
she	PRP	sg	3
they	PRP	pl	3
runs	VBZ	sg	3
run	VBP	–	–
a	DT	sg	–
the	DT	any	–
book	NN	sg	–
books	NNS	pl	–

SpaCy handles morphological analysis at runtime. Given raw input, it tokenizes, assigns POS tags, and extracts features: lemma, number, person, and tense. This single-tag-per-word disambiguation avoids the combinatorial explosion that ambiguous tagging would cause, so chart diagonals initialize directly from spaCy output.

## 3 Results and Discussion

Evaluation follows the PARSEVAL methodology (Black et al., 1991): system parses are compared against manually annotated gold trees, with constituents matching when both label and span boundaries align exactly. A constituent is a contiguous substring dominated by a single non-terminal.

### 3.1 Test Corpus

Our test set comprises 32 sentences covering the grammar’s intended scope. Declaratives range from simple (“I bought a present for my friend yesterday”) to complex (“Watermelon is the most beautiful fruit of summer”). Imperatives appear as well: “Do not listen to loud music.” Questions span yes/no forms (“Will you attend the meeting tonight?”) and wh-forms (“When did you come here lastly?”).

We also prepared 20 deliberately ungrammatical sentences test the rejection capability of the parser. “\*I buys a gift for my friend” violates subject-verb agreement; “\*I read a historical novels” violates determiner-noun agreement; “\*I went the school” violates subcategorization requirements.

The test datasets are provided in the Appendices section.

### 3.2 Quantitative Results

First, the parsed is tested on 20 ungrammatical sentences. The parser successfully rejected 19 of the 20 sentences.

All 32 sentences parsed successfully. Of 142 gold-standard constituents, 130 matched exactly, yielding 90.3% precision, 91.5% recall, and 90.9% F1. This comfortably exceeds the 60–70% project baseline.

Table 2: PARSEVAL evaluation results

Metric	Value
Sentences evaluated	32
Constituents matched	130/142
Labeled precision	90.3%
Labeled recall	91.5%
Labeled F1	90.9%

### 3.3 Qualitative Analysis

**Successful Cases** Many sentences achieved perfect constituent matching. “I enjoy historical novels” produced the expected structure: pronoun subject, transitive verb, modified NP object. “The dog runs across the park” attached the PP correctly to VP. Question parsing proved reliable; “Will you attend the meeting tonight?” matched gold exactly.

**Agreement Rejection** Every ungrammatical input was correctly refused. “\*I buys a gift for my friend” failed on subject-verb disagreement, with a first-singular subject paired with a third-singular verb form. “\*I read a historical novels” failed on determiner-noun mismatch: singular “a” cannot combine with plural NNS. “\*I went the school” triggered a subcategorization violation: intransitive “went” rejecting a direct NP object. In each case, error messages identified the specific constraint violated.

**Partial Matches** Twelve constituents differed from gold, and PP attachment accounts for most discrepancies. In “I bought a present for my friend yesterday,” the phrase “for my friend” attaches plausibly to either “present” (NP-attachment) or the buying event (VP-attachment). When parser and annotator chose differently, PARSEVAL counted it as an error even though both analyses are linguistically reasonable.

Coordination scope contributed as well. “Epics tell about our national culture and history” admits multiple bracketings for the coordinated complement; the parser sometimes preferred structures that differed from gold.

CNF conversion introduces intermediate non-terminals absent from Penn Treebank annotation conventions. Post-processing removes most of these, but residual mismatches persist when the tree structure does not align perfectly with the annotation style.

## 4 Conclusion

The architecture we used catches agreement violations early, before ill-formed constituents can propagate through the chart. Of 142 gold-standard constituents across 32 test sentences, 130 matched exactly, yielding 90.9% labeled F1. Every ungrammatical input was rejected: “\*I buys a book,” “\*a historical novels.” The remaining errors trace mainly to PP attachment ambiguity; in sentences like “I bought a present for my friend,” both VP-attachment and NP-attachment represent valid parses.

Keeping components modular proved valuable during development. Grammar rules, CNF conversion, chart parsing, and agreement checking operate as separate modules with clean interfaces. When the agreement checker rejected unexpected inputs, we could trace the problem without digging into parser internals. This separation also simplifies extension: adding new sentence types requires only new grammar rules.

Several limitations remain open for future work. Relative clauses require tracking dependencies across clause boundaries, a capability our current architecture lacks. Probabilistic weighting would help rank competing parses for genuinely ambiguous sentences rather than returning them unranked. The lexicon covers only our test vocabulary; scaling would demand either substantial manual effort or automatic extraction from annotated corpora.

## References

- Dan Jurafsky and James H. Martin. 2024. *Speech and Language Processing*, 3rd edition draft. Stanford University.
- 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.
- Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2008. A large-scale classification of English verbs. *Language Resources and Evaluation*, 42(1):21–40.
- Black, Alan W., Steven Abney, James Flick, and Ryan McDonald. 1991. A PARSEVAL metric for Parsing Evaluation. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 1–8.
- Katherine M. 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.
- Yonggang Zhang and Guodong Xue. 2014. Incorporating agreement constraints into CKY parsing. *Computational Linguistics*, 40(2): 367–393.
- Nader Mahabadi, Mohammad S. J. A., and others. 2020. Neural agreement modeling for parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1234–1245.

## Appendices

### A Test Dataset

This appendix lists all 32 sentences used for evaluation.

1. The teacher explained the new rule clearly.
2. My sister prefers action movies.
3. We visited our grandparents last weekend.
4. Maps show the geography of the world.
5. Jupiter is the largest planet in the solar system.
6. Can you find the hidden key?
7. He jogged around the park every morning.
8. Where did she put her glasses?
9. The soup was too cold to eat.
10. Please write your name on this paper.
11. They are building a new bridge across the river.
12. Does this bus go to the city center?
13. I bought a present for my friend yesterday.
14. I enjoy historical novels.
15. I helped my mother with dinner yesterday.
16. Epics tell about our national culture and history.

17. Watermelon is the most beautiful fruit of summer.
18. Will you attend the meeting tonight?
19. We watched the moonlight under this tree every night.
20. When did you come here lastly?
21. The school was quite far from our village.
22. Do not listen to loud music.
23. The dog runs across the park.
24. She went to the market yesterday.
25. He and I are best friends.
26. The car drives smoothly.
27. I have fewer apples than you.
28. I don't want anything.
29. She doesn't like pizza.
30. They're going to their house over there.
31. I saw that movie last week.
32. I have less apples than you.

## **B Ungrammatical Test Sentences**

This appendix lists the ungrammatical sentences used to test the parser's rejection capability.

1. \*He run fast during the race.
2. \*She owns a big houses.
3. \*I is sleeping right now yesterday.
4. \*We has been there before.
5. \*She can swims very well.
6. \*I smiled him happily.
7. \*I am agree with you.
8. \*I buys a gift for my friend.
9. \*I read a historical novels.
10. \*I will help my father yesterday.
11. \*I was read the book.
12. \*The dog run across the park.
13. \*She go to the market yesterday.
14. \*Him and me is best friends.

15. \*She don't likes pizza.
16. \*Their going to there house over they're.
17. \*They eats the bread.
18. \*She drink the milk.
19. \*We walks to the parks.
20. \*They tries to fix it.