Syntactic and Dependency Parsing

Natalie Parde, Ph.D.

Department of Computer Science

University of Illinois at Chicago

CS 421: Natural Language Processing

Fall 2019

Many slides adapted from Jurafsky and Martin (https://web.stanford.edu/~jurafsky/slp3/), Ben-Gurion University's NLP course (https://www.cs.bgu.ac.il/~michaluz/seminar/CKY1.pdf).

What is syntactic parsing?

The process of automatically recognizing sentences and assigning syntactic (grammatical) structure to them.

Why is syntactic parsing useful?

- Lots of reasons!
  - Grammar checking
    - Sentences that can't be parsed may be grammatically incorrect (or at least hard to read)
  - Semantic analysis
  - Downstream applications
    - Question answering
    - Information extraction

What courses were **taught by UIC CS assistant professors in 2019**?

Subject = courses …don't return a list of UIC CS assistant professors!

Parsing algorithms are one of the core tools for analyzing natural language.

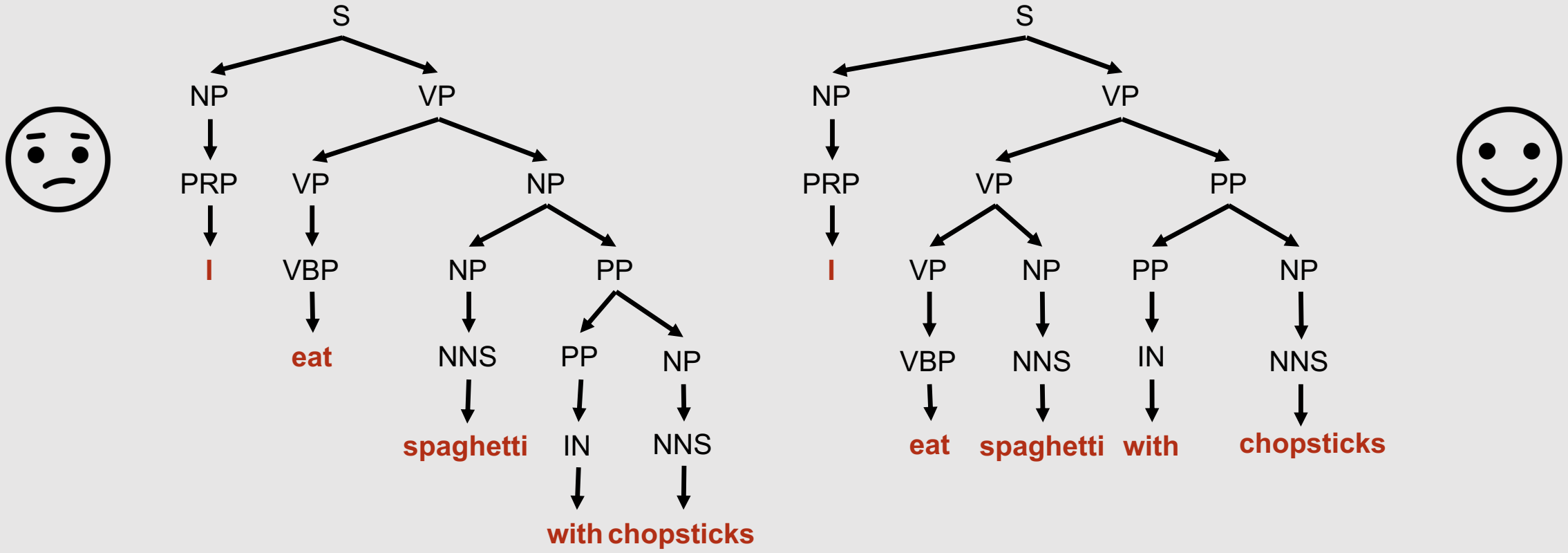
- Parsing algorithms automatically describe the syntactic structure of sentences in terms of **context-free grammars** (such as those discussed last week)
- This can be viewed as a **search problem**:
  - Given the set of all possible parse trees, find the correct parse tree for this sentence.

Recognition vs. Parsing

- **Recognition:** Deciding whether a sentence belongs to the language specified by a formal grammar.
- **Parsing:** Producing a parse tree for the sentence based on that formal grammar.
- Both tasks are necessary for generating correct syntactic parses!
  - Failure to accurately recognize whether a sentence can be parsed will lead to **misparses**, which will in turn lead to additional errors in downstream applications.
- Parsing is more “difficult” (greater time complexity) than recognition

Remember, language is ambiguous!

Input sentences may have many possible parses



There are many ways to generate parse trees.

Top-Down Parsing:

- Goal-driven
- Builds parse tree from the start symbol down to the terminal nodes

Bottom-Up Parsing:

- Data-driven
- Builds parse tree from the terminal nodes up to the start symbol

Naïve approach: Enumerate all possible solutions

Dynamic programming approach: Save partial solutions in a table, and use this information to reduce search time

These approaches can be implemented naïvely, or using more advanced techniques.

Top-Down Parsing

- Assume that the input can be derived by the designated start symbol ***S***
- Find the tops of all trees that can start with ***S***
  - Look for all production rules with ***S*** on the left-hand side
- Find the tops of all trees that can start with those constituents
- (Repeat recursively until POS categories at bottom of tree are reached)
- Trees whose leaves fail to match all words in the input sentence can be rejected, leaving behind trees that represent successful parses

Top-Down Parsing: Example

Input Sentence:

Book that flight.

Grammar:

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP

Lexicon:

Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

Top-Down Parsing: Example

Book that flight.

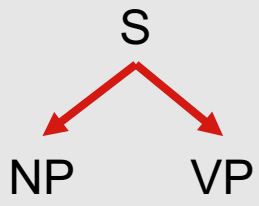
S S S

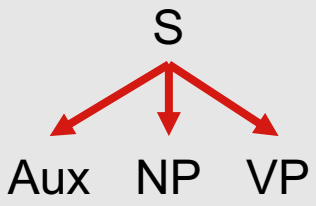
S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP

Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP

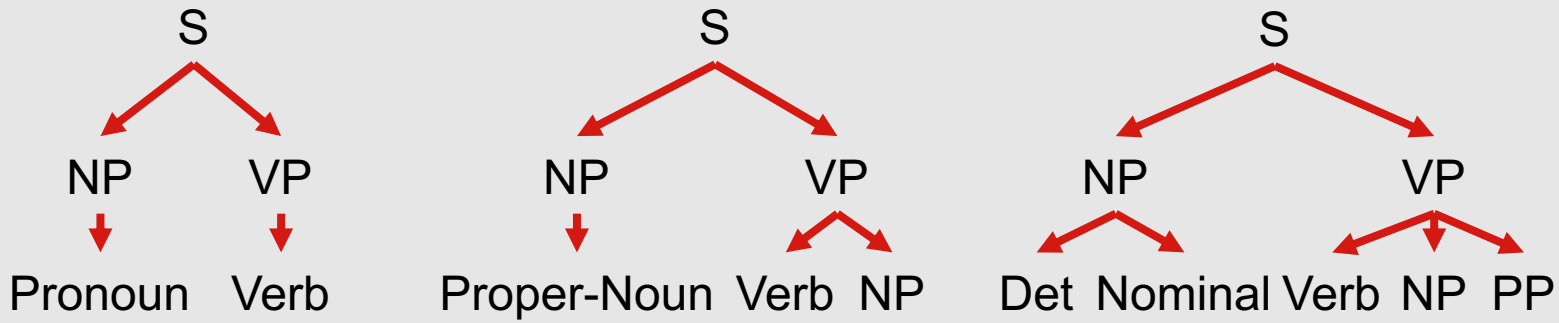


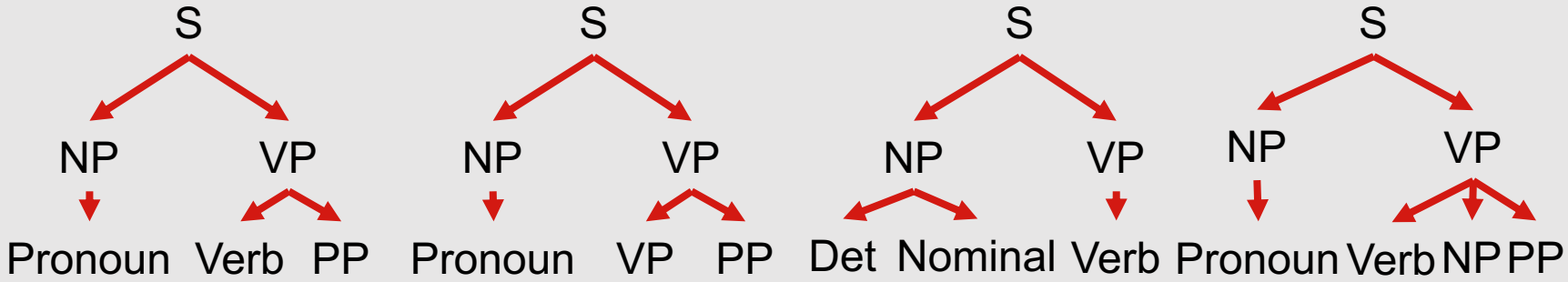


Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP



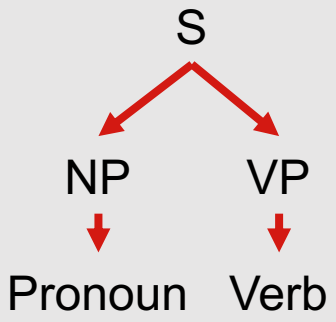


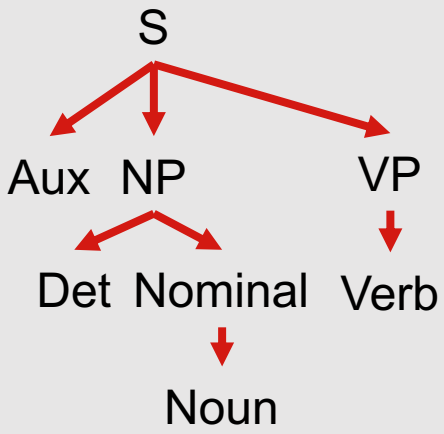
…and many more!

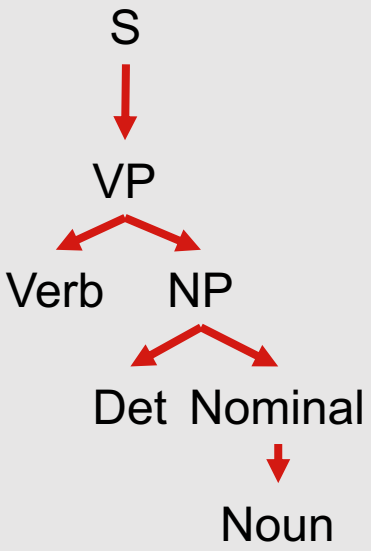
Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP





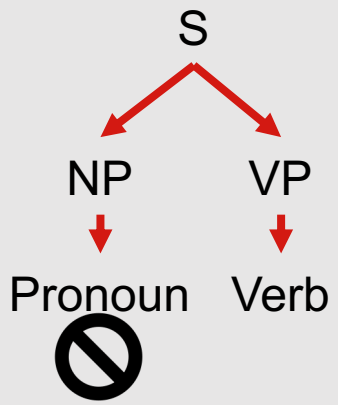


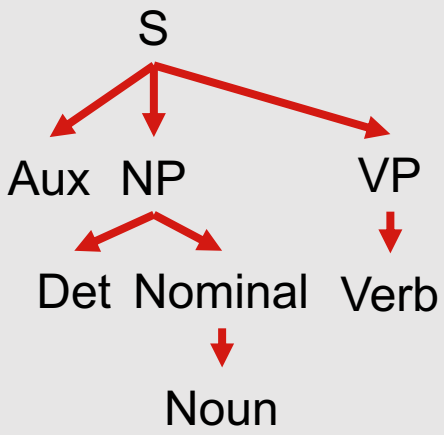
…and many, many more not shown!

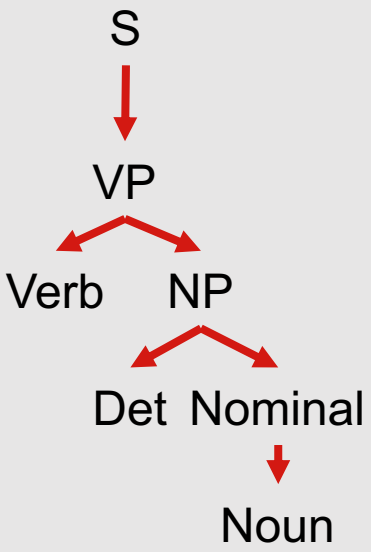
Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP







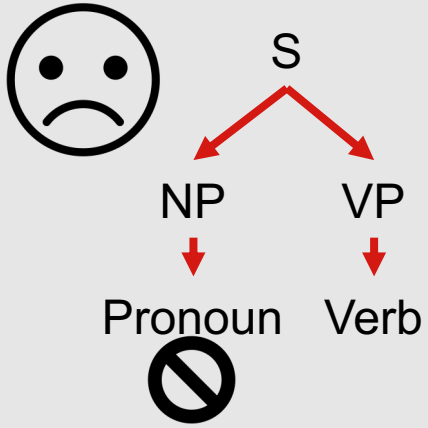
Det → that | this | a
Noun → book | flight | meal | money
Verb → **book** | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

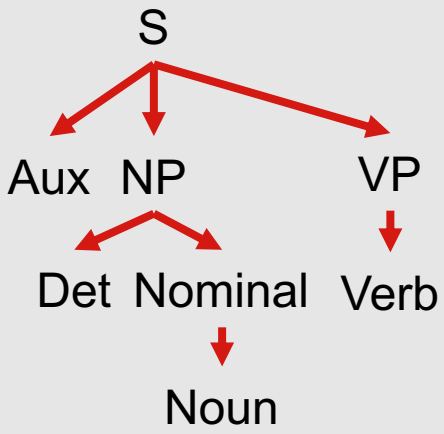
…and many, many more not shown!

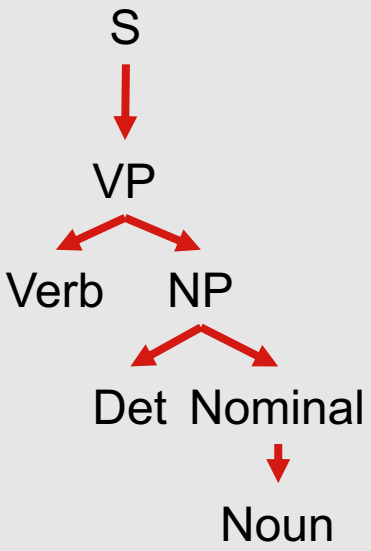
Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP







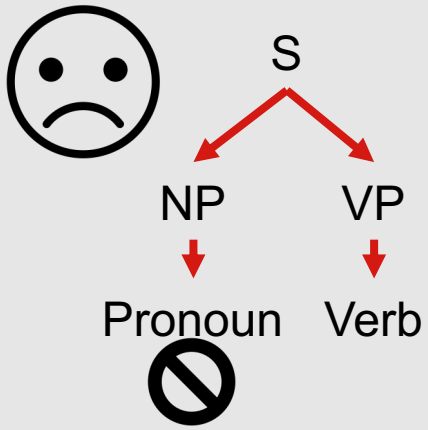
Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

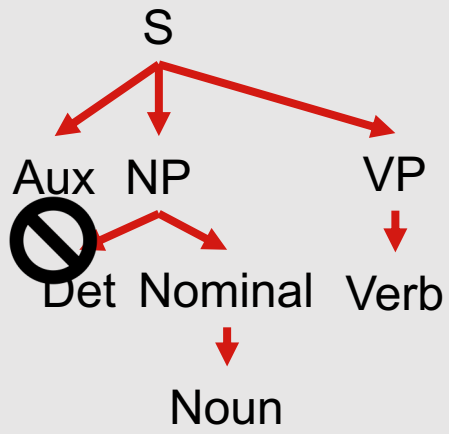
…and many, many more not shown!

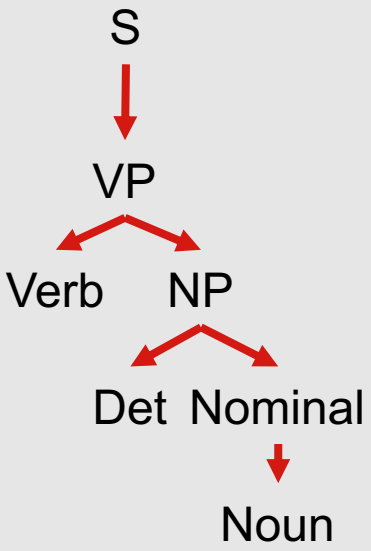
Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP







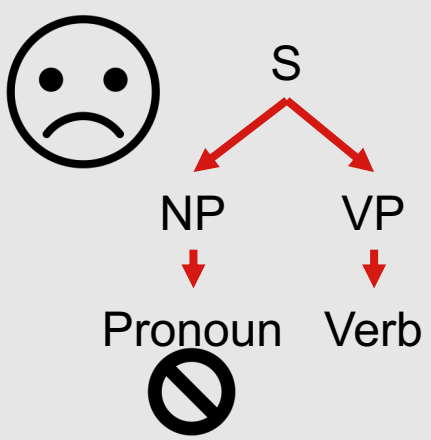
Det → **that** | this | a
Noun → book | **flight** | meal | money
Verb → **book** | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

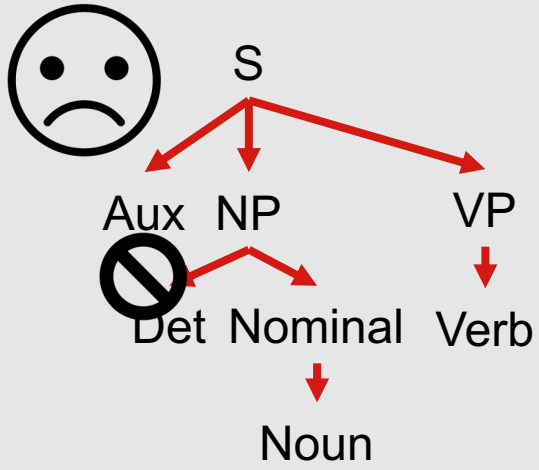
…and many, many more not shown!

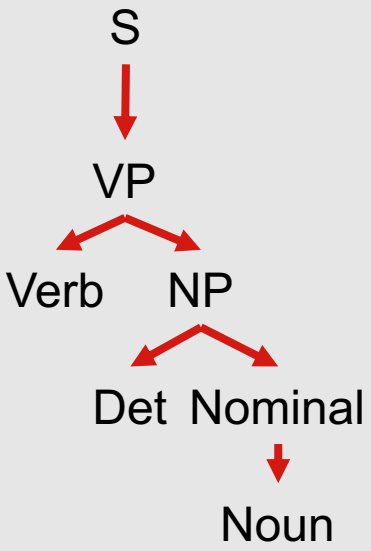
Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP







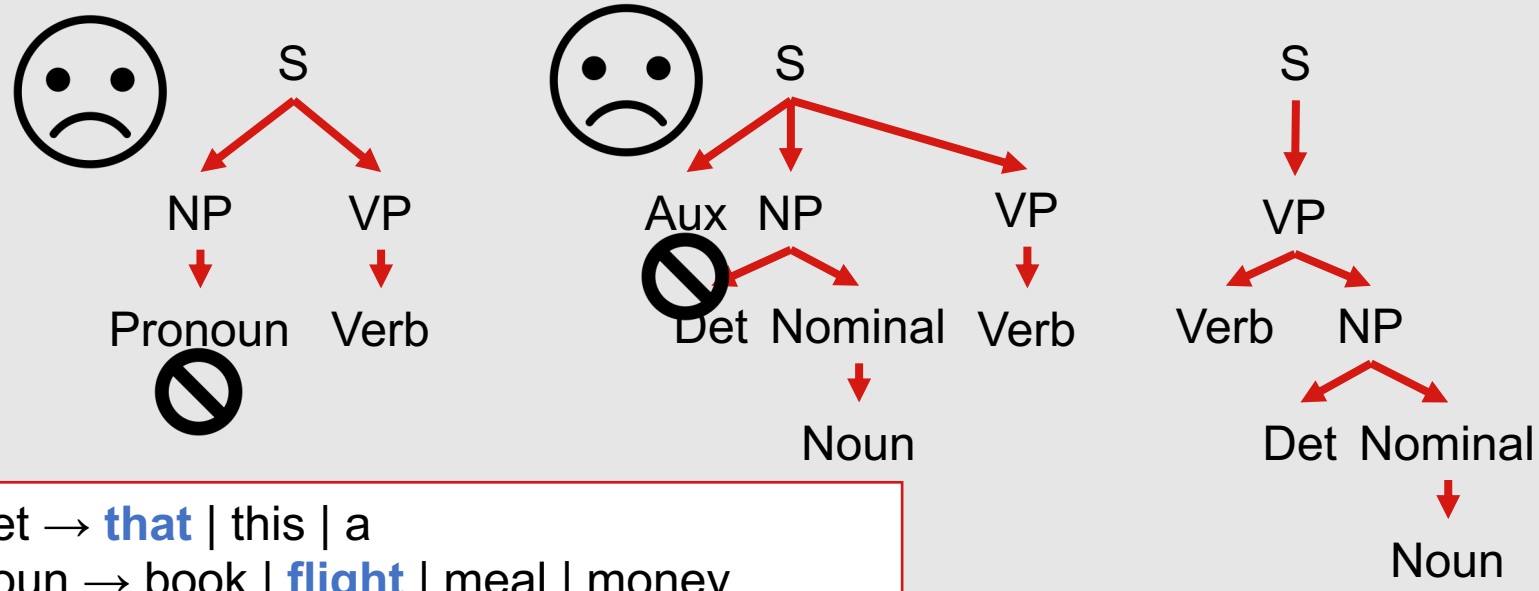
Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

…and many, many more not shown!

Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP



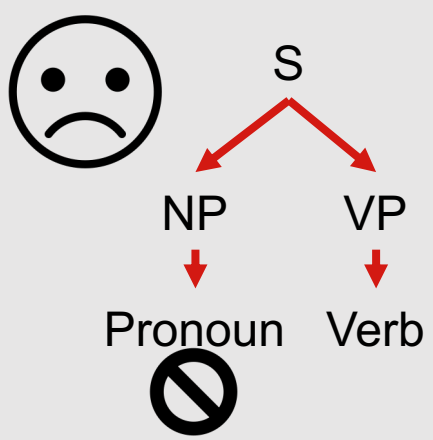
Det → **that** | this | a
Noun → book | **flight** | meal | money
Verb → **book** | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

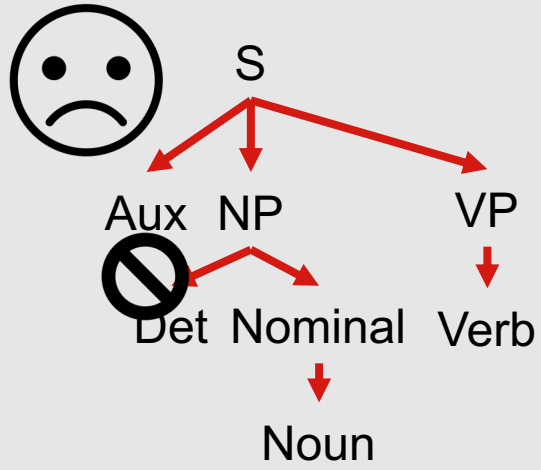
…and many, many more not shown!

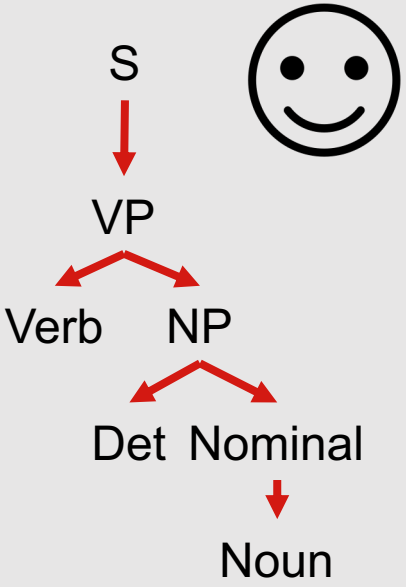
Top-Down Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP







Det → **that** | this | a
Noun → book | **flight** | meal | money
Verb → **book** | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

…and many, many more not shown!

Bottom-Up Parsing

- Earliest known parsing algorithm!
- Starts with the words in the input sentence, and tries to build trees from those words up by applying rules from the grammar one at a time
  - Looks for places in the in-progress parse where the righthand side of a production rule might fit
- Success = parser builds a tree rooted in the start symbol ***S*** that covers all of the input words

Bottom-Up Parsing: Example

Input Sentence:

Book that flight.

Grammar:

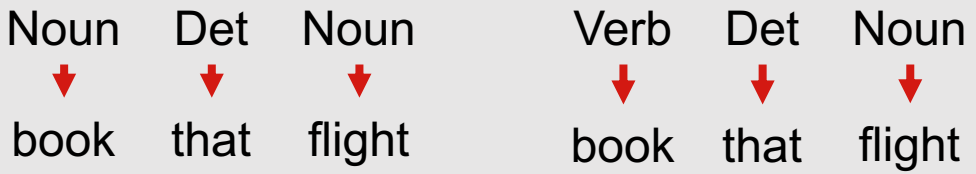
S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP

Lexicon:

Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

Bottom-Up Parsing: Example

Book that flight.

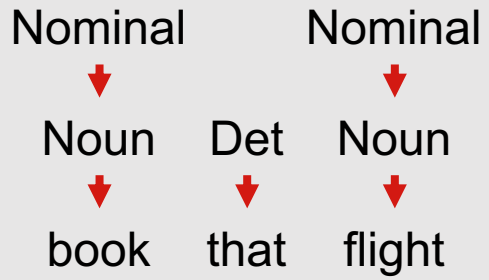


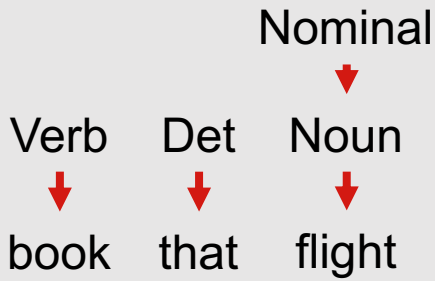
Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near | through

Bottom-Up Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP

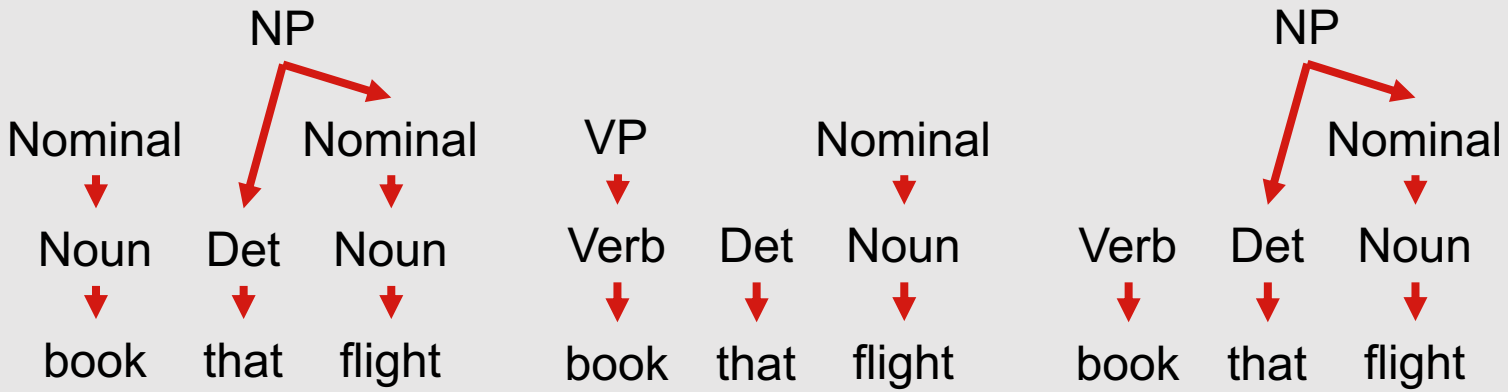




Bottom-Up Parsing: Example

Book that flight.

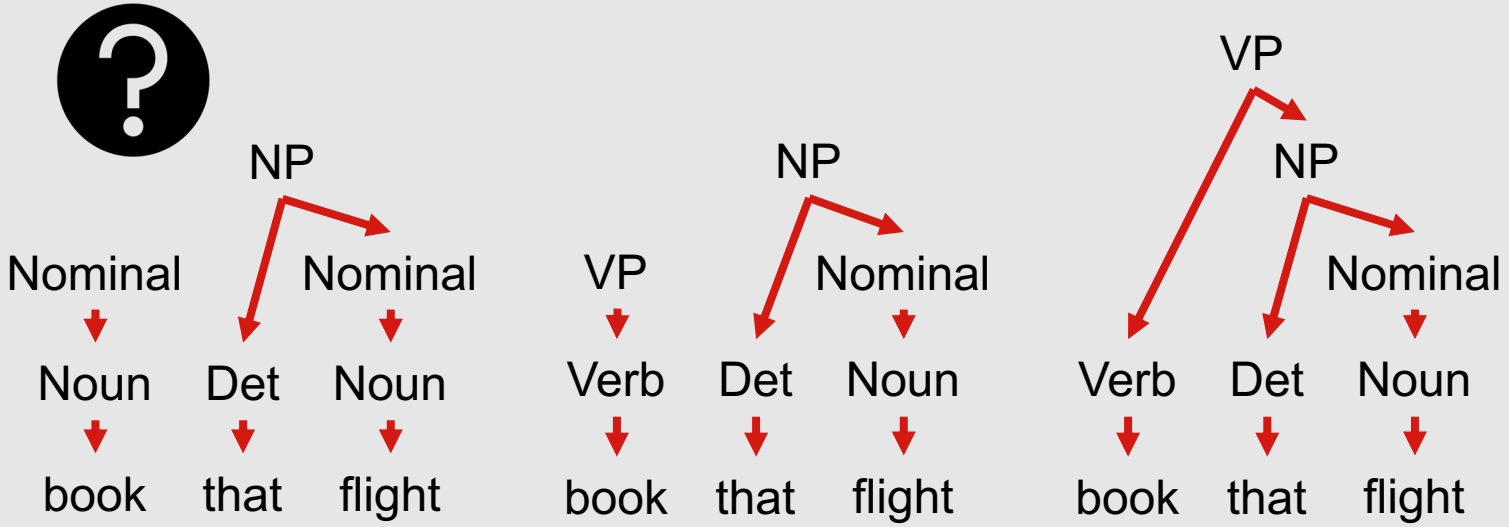
S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP



Bottom-Up Parsing: Example

Book that flight.

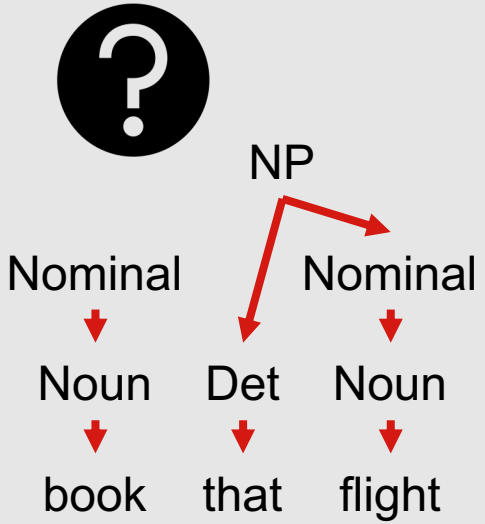
S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP

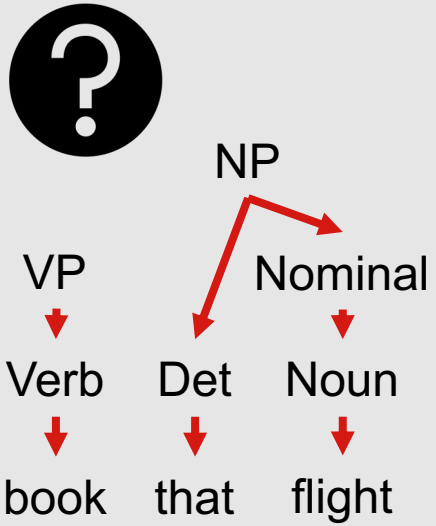


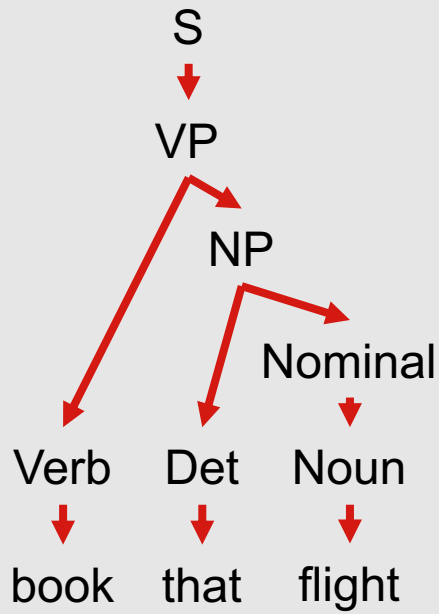
Bottom-Up Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP



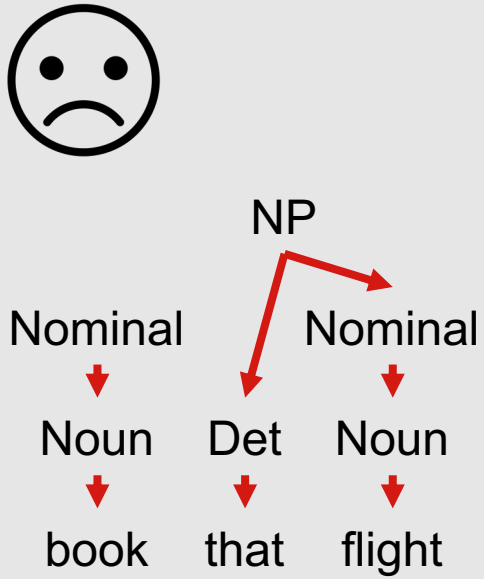


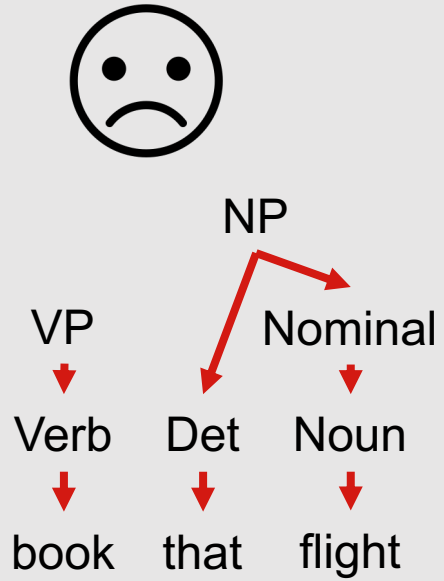


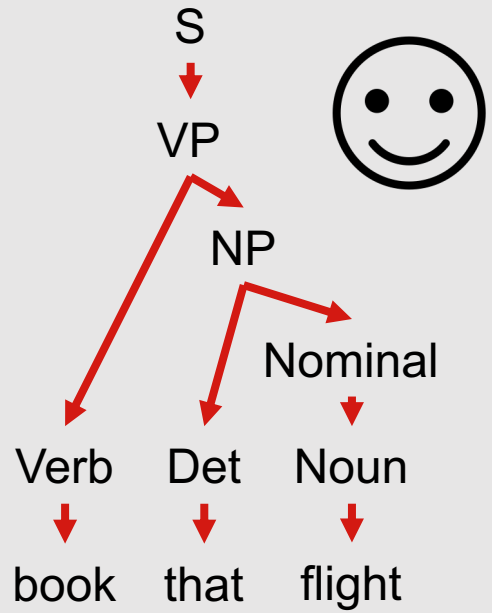
Bottom-Up Parsing: Example

Book that flight.

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP







Top-Down vs. Bottom-Up Parsing

Top-Down Parsing

- Pros:
  - Never wastes time exploring trees that cannot result in a sentence
  - Never explores subtrees that cannot fit into a larger valid (i.e., results in a sentence) tree
- Cons:
  - Spends considerable effort on trees that are not consistent with the input

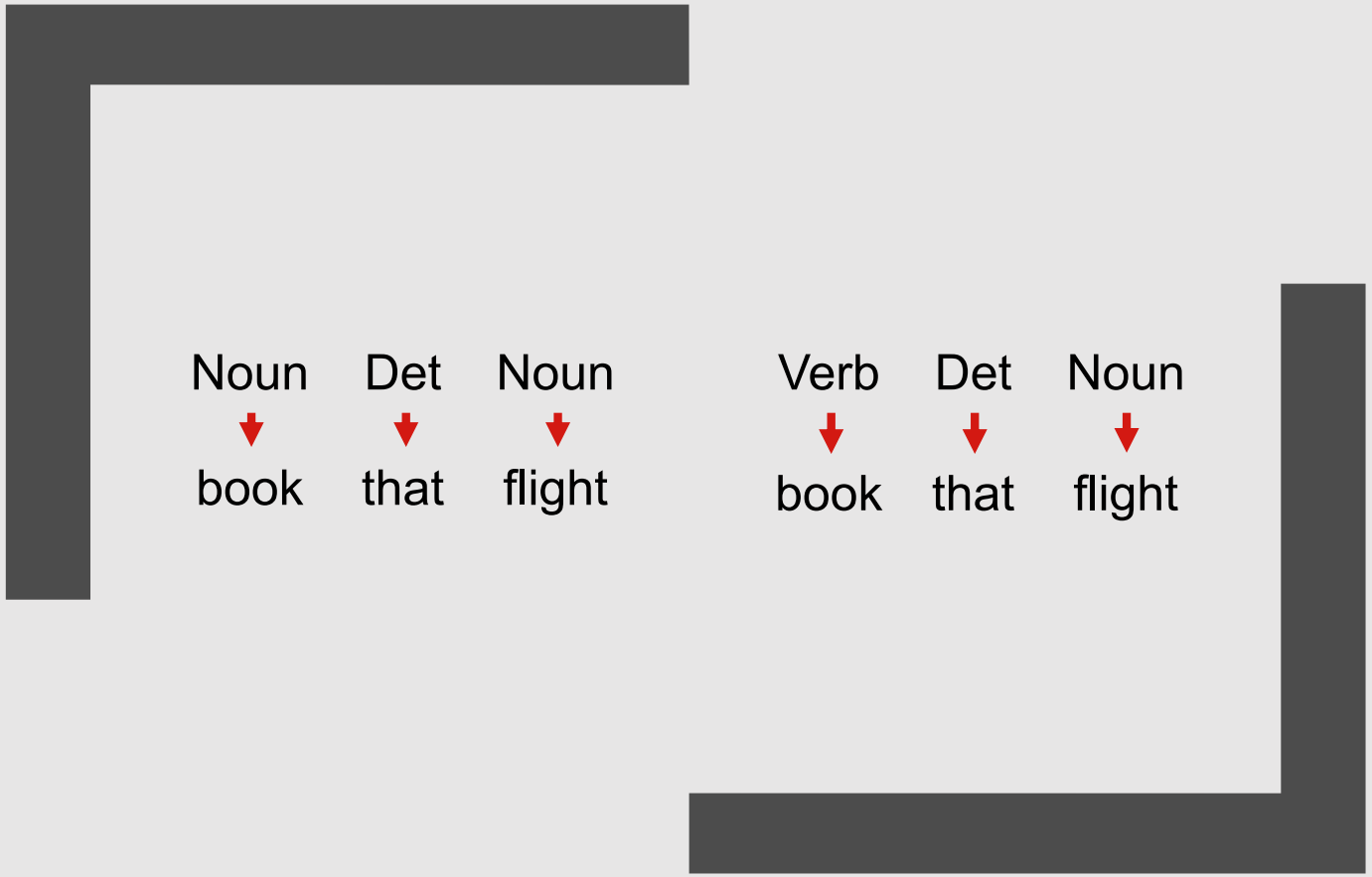
Bottom-Up Parsing

- Pros:
  - Never suggests trees that are inconsistent with the input
- Cons:
  - Generates many trees and subtrees that cannot result in a valid sentence (according to production rules specified by the grammar)

More about ambiguity….

- **Structural Ambiguity:** Occurs when a grammar allows for more than one possible parse for a given sentence
- Two Forms:
  - **Attachment Ambiguity:** Occurs when a constituent can be attached to a parse tree at more than one place
    - I eat spaghetti **with chopsticks**.
  - **Coordination Ambiguity:** Occurs when different sets of phrases can be conjoined by a conjunction
    - I grabbed a muffin from the table marked "nut-free scones **and** muffins," hoping I'd parsed the sign correctly.

Remember …local ambiguity can also exist.



- Det → that | this | a
- Noun → **book** | flight | meal | money
- Verb → **book** | include | prefer
- Pronoun → I | she | me
- Proper-Noun → Houston | NWA
- Aux → does
- Preposition → from | to | on | near | through

All of this ambiguity can lead to really complex search spaces!

- **Backtracking** approaches expand the search space incrementally, systematically exploring one state at a time
- When they arrive at trees inconsistent with the input, they return to an unexplored alternative
- However, in doing so they tend to discard valid subtrees …this means that time-consuming work needs to be repeated
- More efficient approach?
  - **Dynamic programming**

Dynamic Programming Parsing Methods

- **Tables store subtrees for constituents as they are discovered**
- Solves:
  - Re-parsing problem
  - (Partially) ambiguity problem, since the table implicitly stores all possible parses

Dynamic Programming Parsing Methods

- Most widely used methods:
  - Cocke-Kasami-Younger (**CKY**) algorithm
  - **Earley** algorithm
  - **Chart parsing**

CKY Algorithm

- One of the earliest recognition and parsing algorithms
- **Bottom-up dynamic programming**
- Standard version can only recognize CFGs in **Chomsky Normal Form** (CNF)

Chomsky Normal Form

- Grammars are restricted to production rules of the form:
  - $A \rightarrow B\ C$
  - $A \rightarrow w$
- This means that the righthand side of each rule must expand to either two non-terminals or a single terminal
- Any CFG can be converted to a corresponding CNF grammar that accepts exactly the same set of strings as the original grammar!

How does this conversion work?

- Three situations we need to address:
  1. Production rules that mix terminals and non-terminals on the righthand side
  2. Production rules that have a single non-terminal on the righthand side (**unit productions**)
  3. Production rules that have more than two non-terminals on the righthand side
- Situation #1: **Introduce a dummy non-terminal that covers only the original terminal**
  - INF-VP → to VP could be replaced with INF-VP → TO VP and TO → to
- Situation #2: **Replace the non-terminals with the non-unit production rules to which they eventually lead**
  - A → B and B → w could be replaced with A → w
- Situation #3: **Introduce new non-terminals that spread longer sequences over multiple rules**
  - A → B C D could be replaced with A → B X1 and X1 → C D

CNF Conversion: Example

- S → NP VP
- S → Aux NP VP
- S → VP
- NP → Pronoun
- NP → Proper-Noun
- NP → Det Nominal
- Nominal → Noun
- Nominal → Nominal Noun
- Nominal → Nominal PP
- VP → Verb
- VP → Verb NP
- VP → Verb NP PP
- VP → Verb PP
- VP → VP PP
- PP → Preposition NP

Original	CNF
S → NP VP	S → NP VP
S → Aux NP VP	S → X1 VP
	X1 → Aux NP
S → VP	S → book | include | prefer
	S → Verb NP
	S → X2 PP
	X2 → Verb NP
	S → Verb PP
	S → VP PP

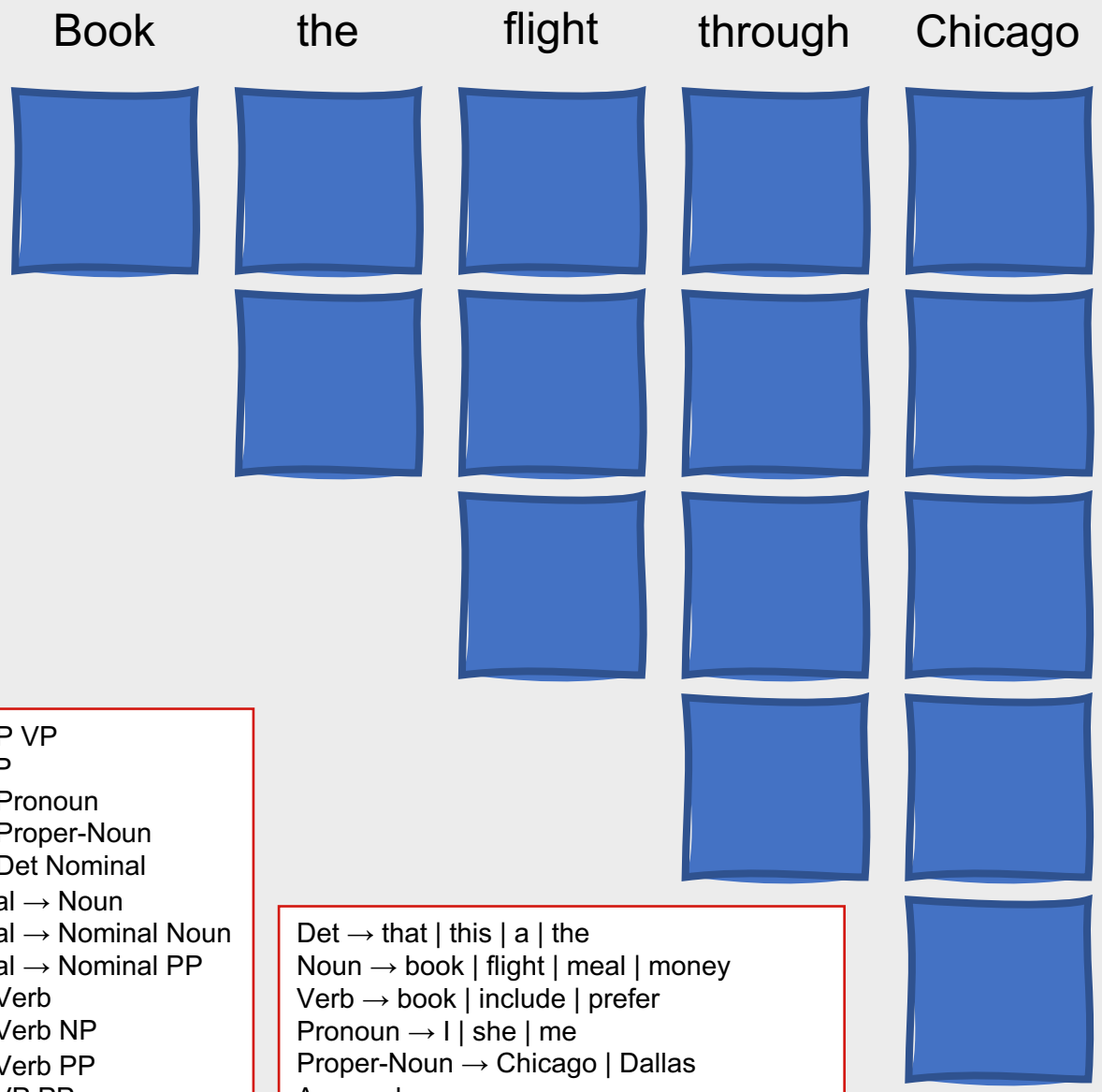
CKY Algorithm

- With the grammar in CNF, each non-terminal node above the POS level of the parse tree will have exactly two children
- Thus, a two-dimensional matrix can be used to encode the tree structure
- For sentence of length *n*, work with upper-triangular portion of (*n*+1) x (*n*+1) matrix
- Each cell [*i*,*j*] contains a set of non-terminals that represent all constituents spanning positions *i* through *j* of the input
  - Cell that represents the entire input resides in position [0,*n*]

CKY Algorithm

- Non-terminal entries: For each constituent [*i*,*j*], there is a position, *k*, where the constituent can be split into two parts such that $i < k < j$
  - [*i*,*k*] must lie to the left of [*i*,*j*] somewhere along row *i*, and [*k*,*j*] must lie beneath it along column *j*
- To fill in the parse table, we proceed in a bottom-up fashion so when we fill a cell [*i*,*j*], the cells containing the parts that could contribute to this entry have already been filled

CKY Algorithm: Example



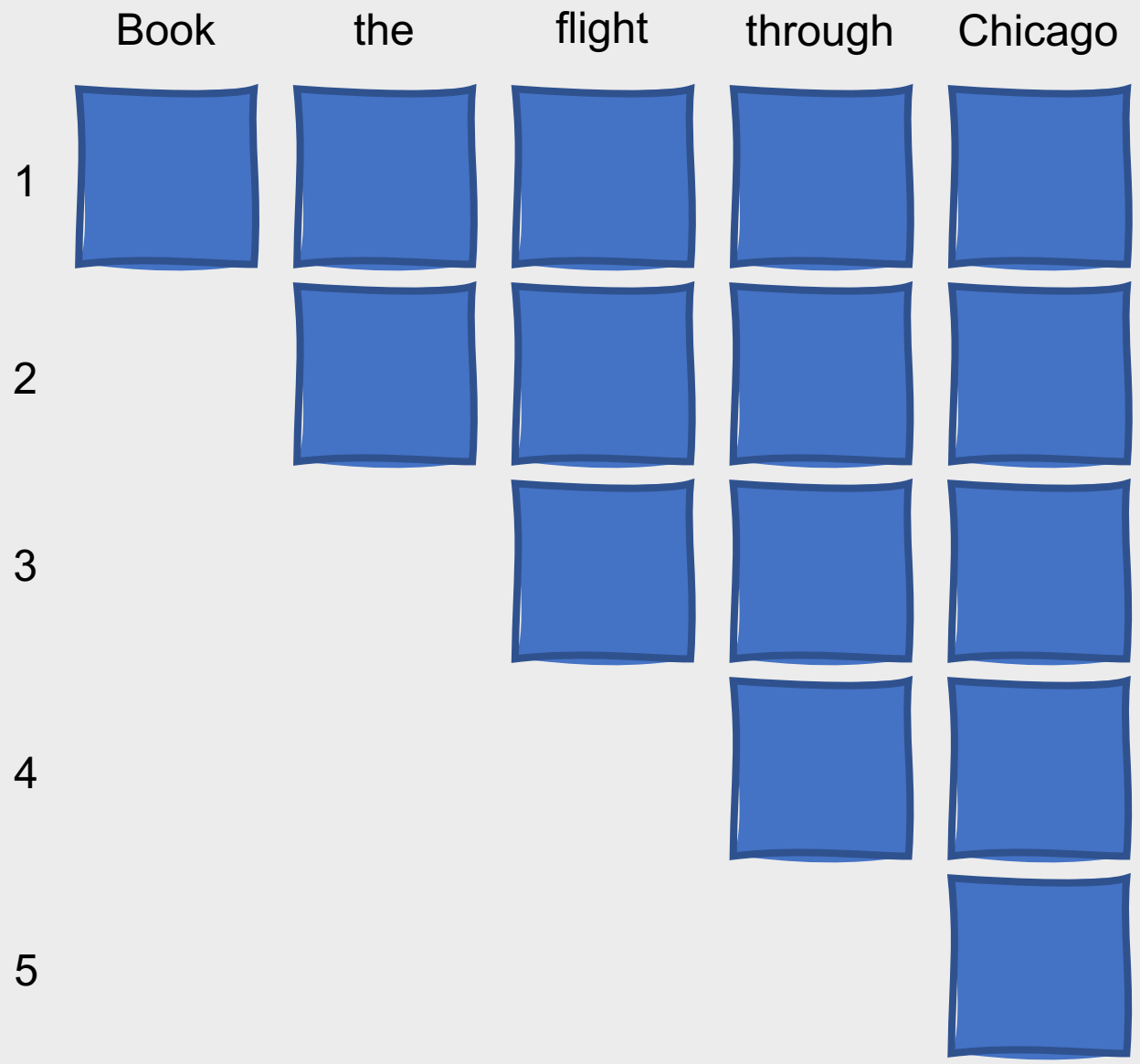
S → NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

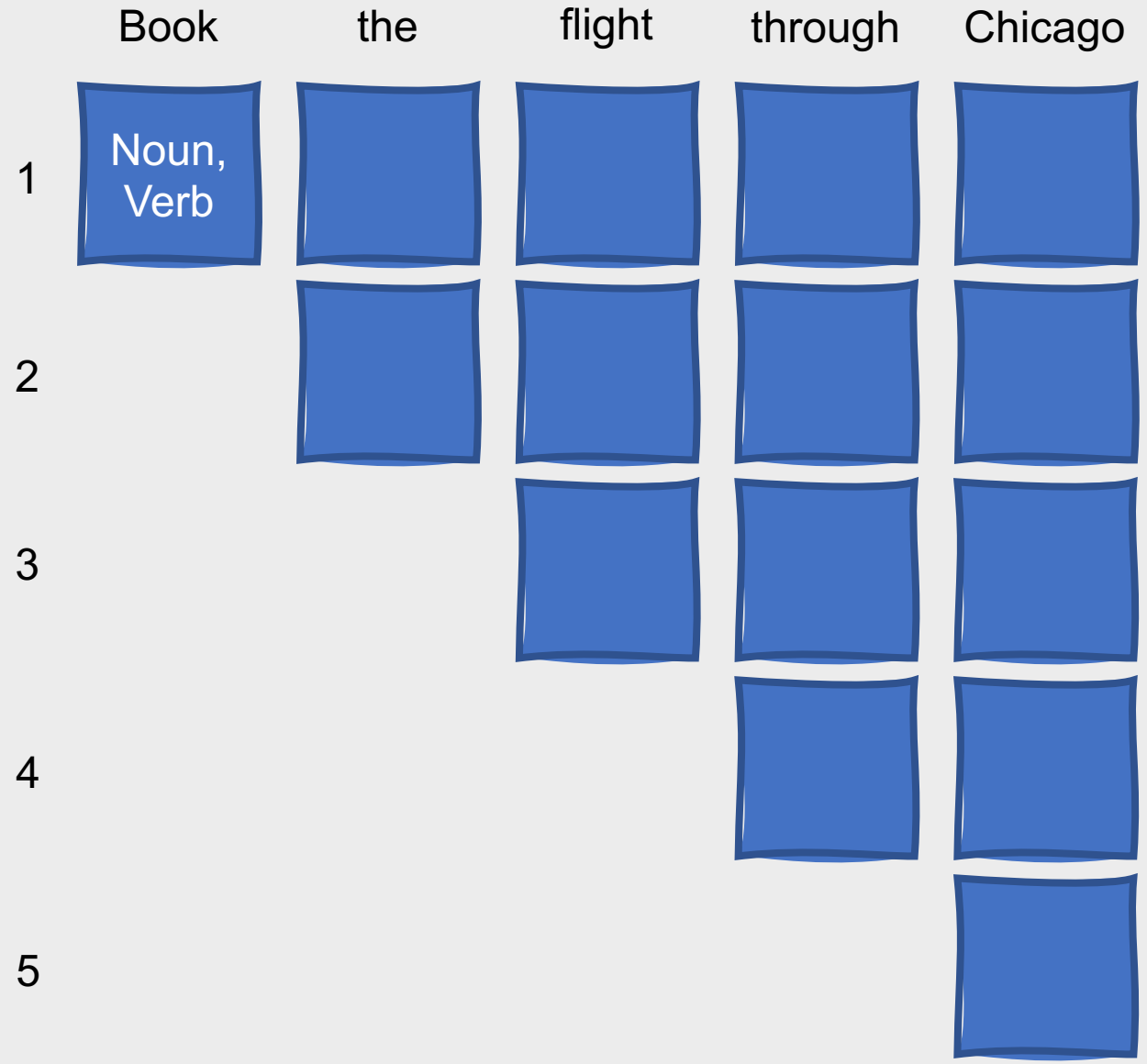
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → **book** | flight | meal | money
Verb → **book** | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

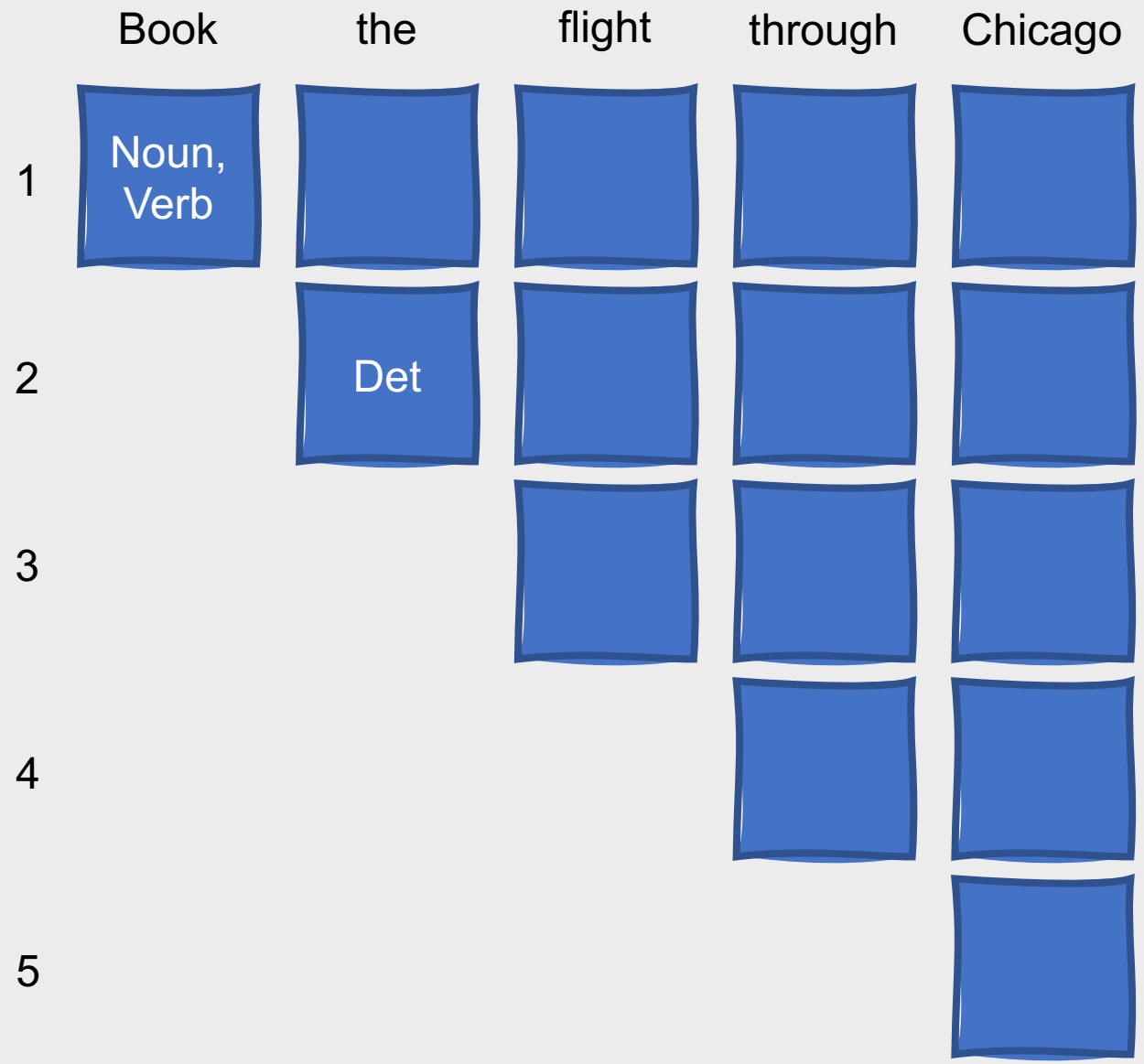
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | **the**
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

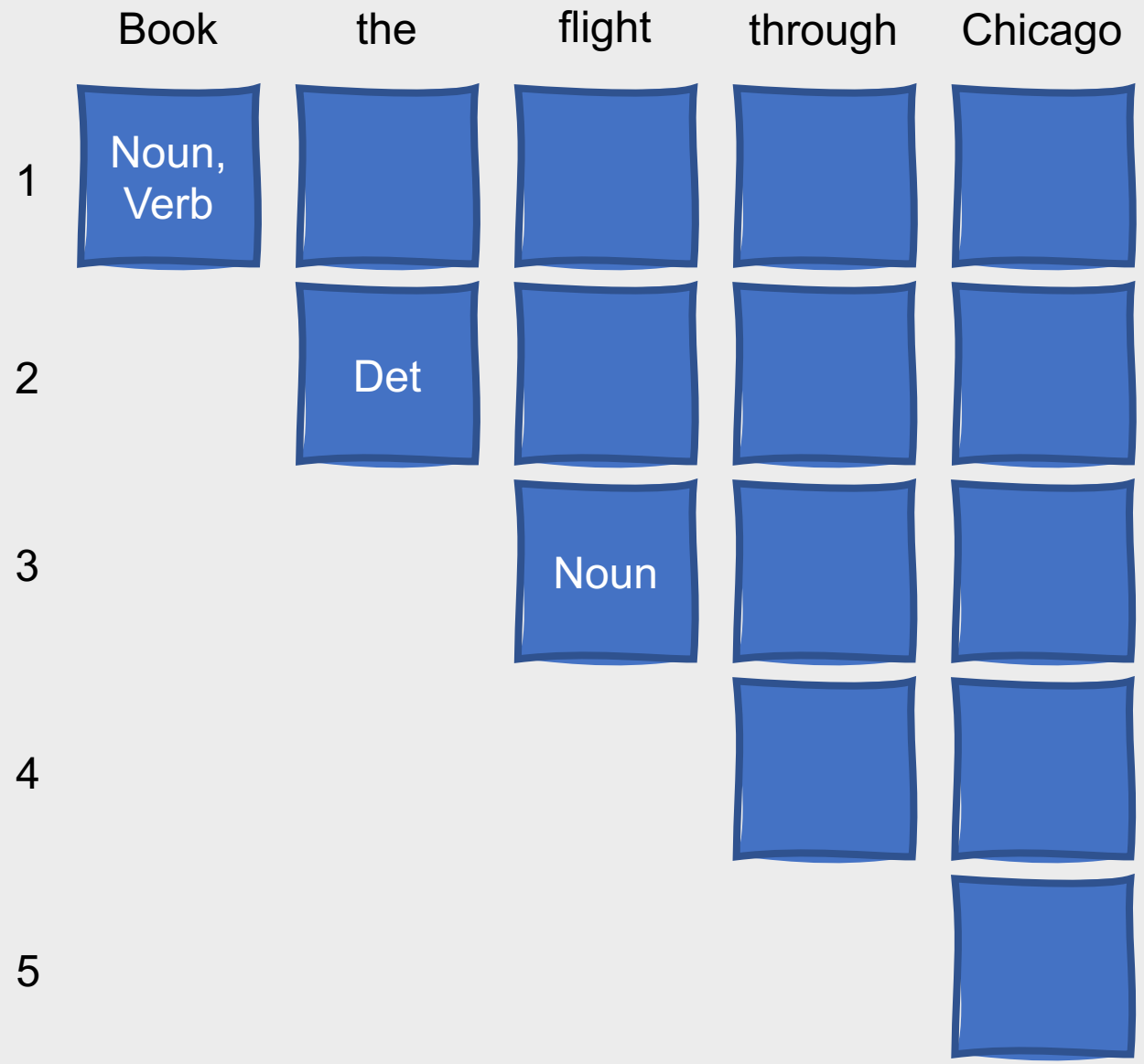
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | **flight** | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

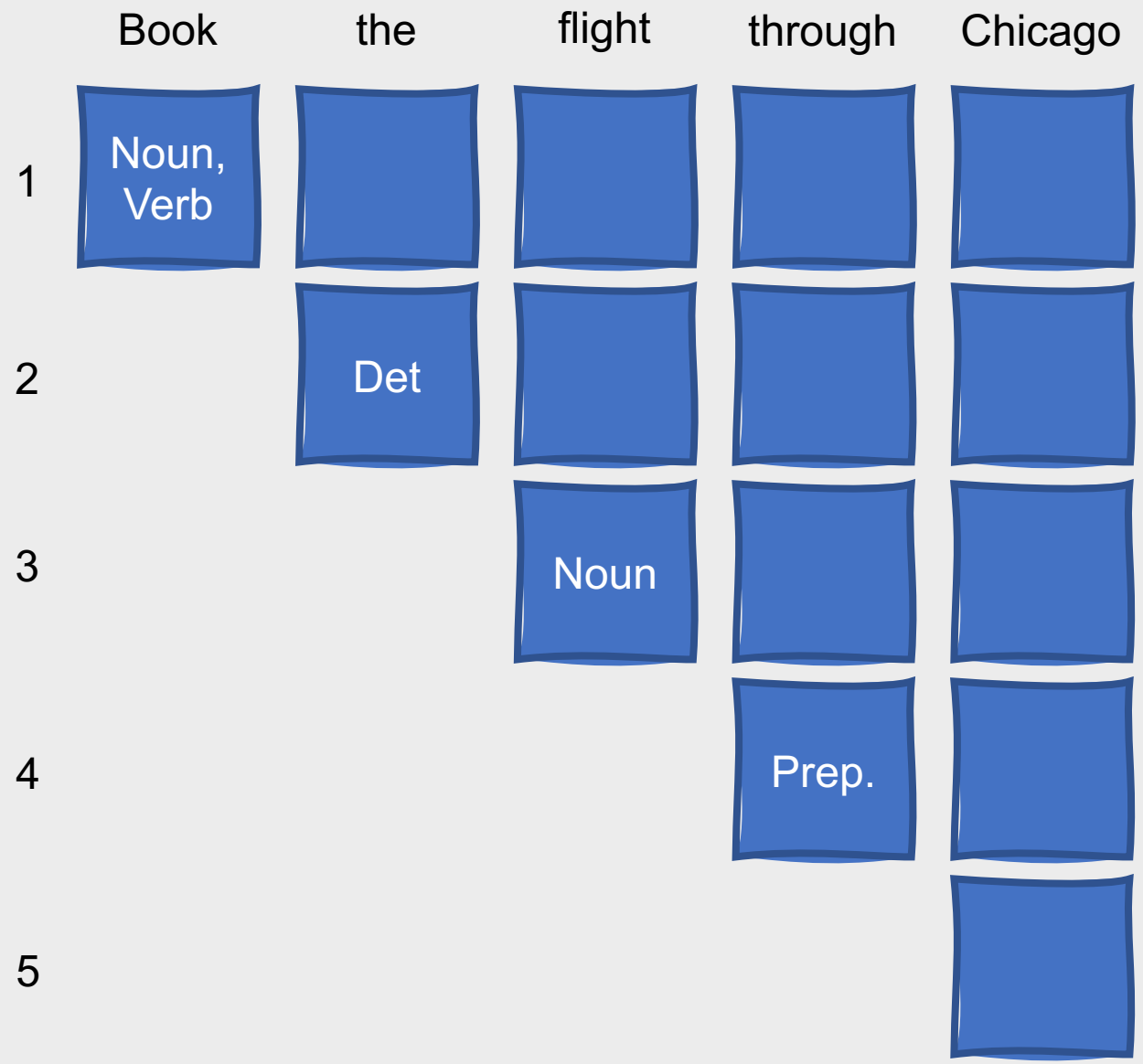
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | **through**

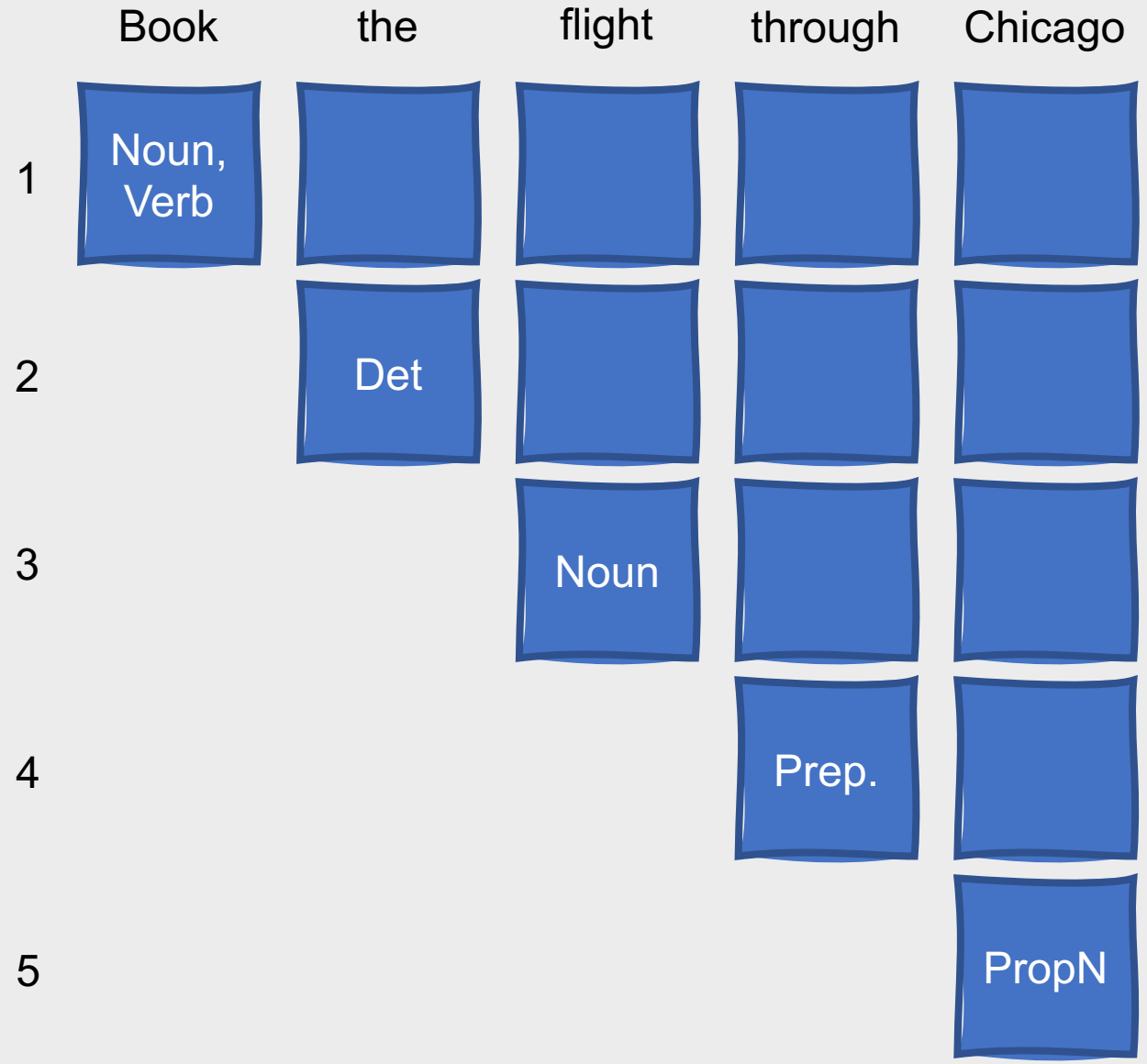
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → **Chicago** | Dallas
Aux → does
Preposition → from | to | on | near | through

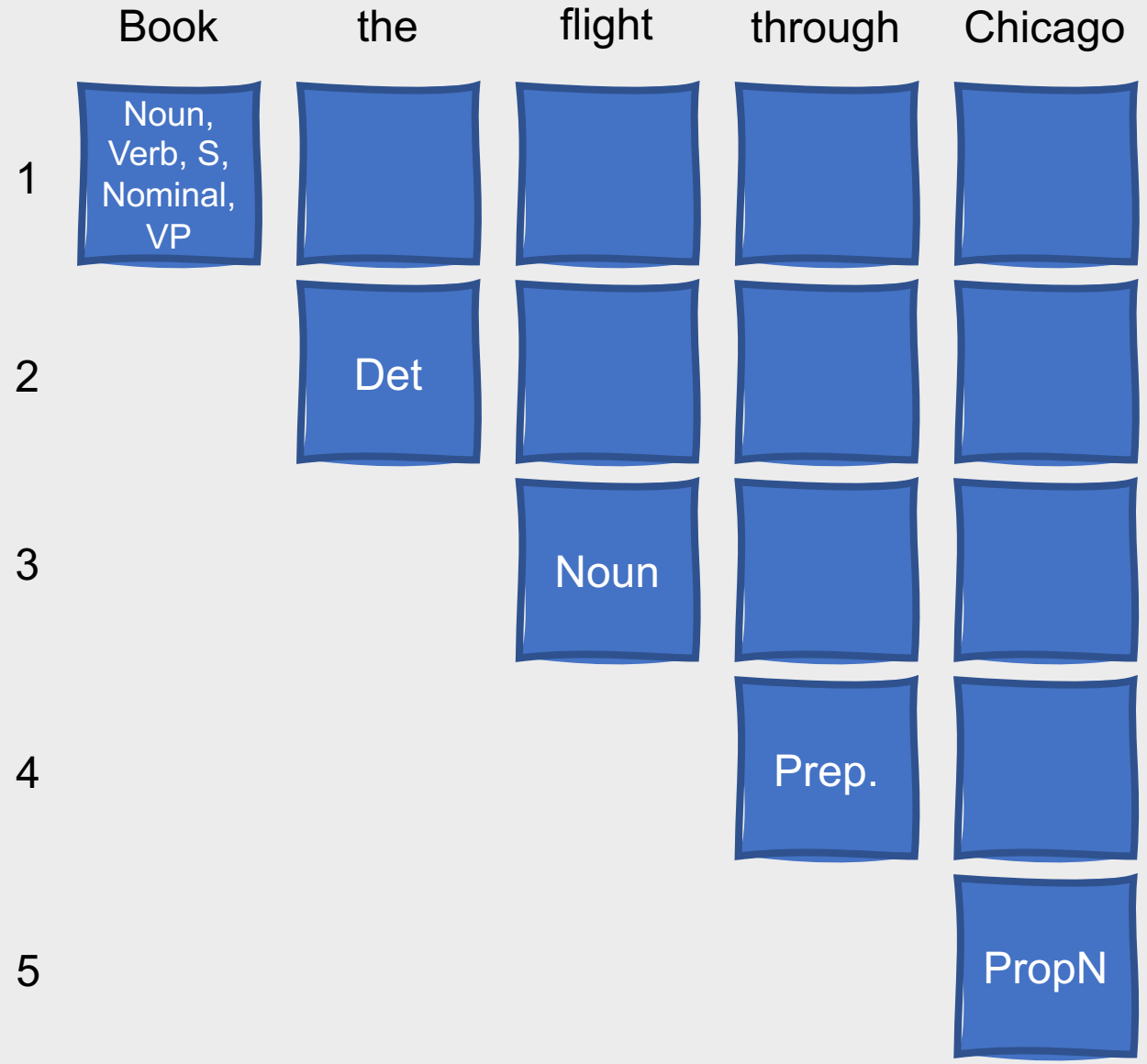
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

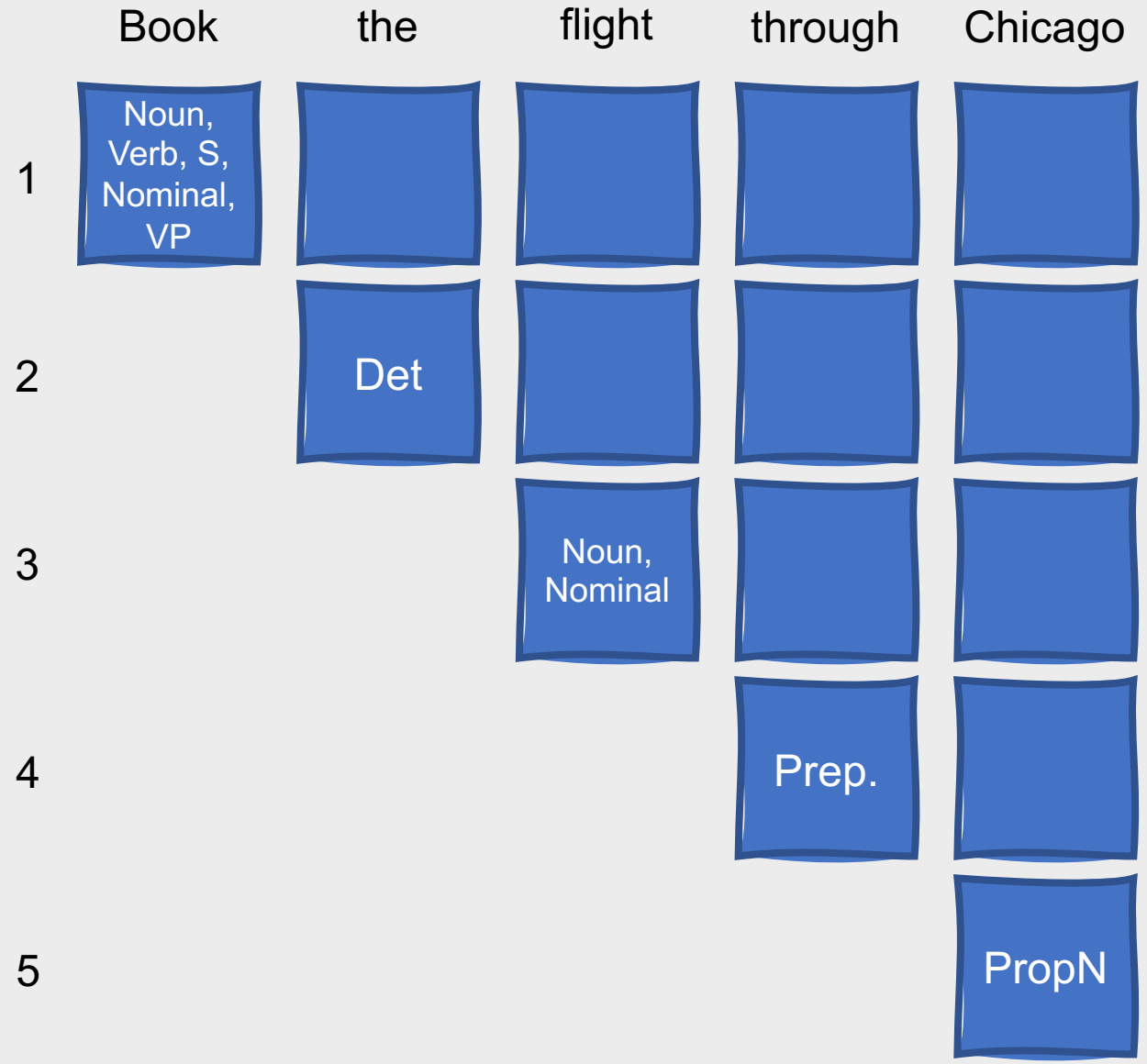
S → NP VP
S → VP → Verb → **book** | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → **book** | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → **book** | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

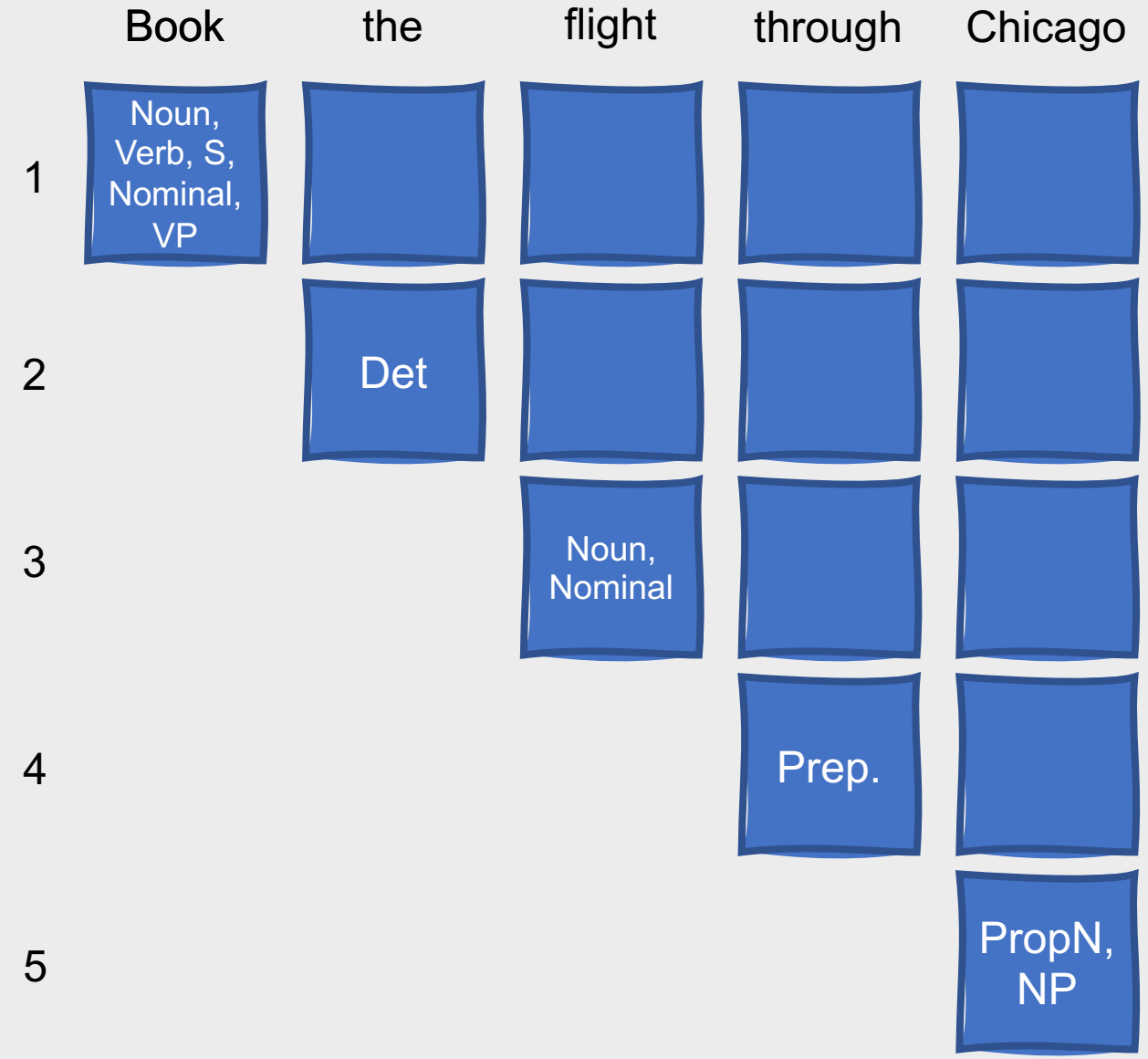
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → **book** | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

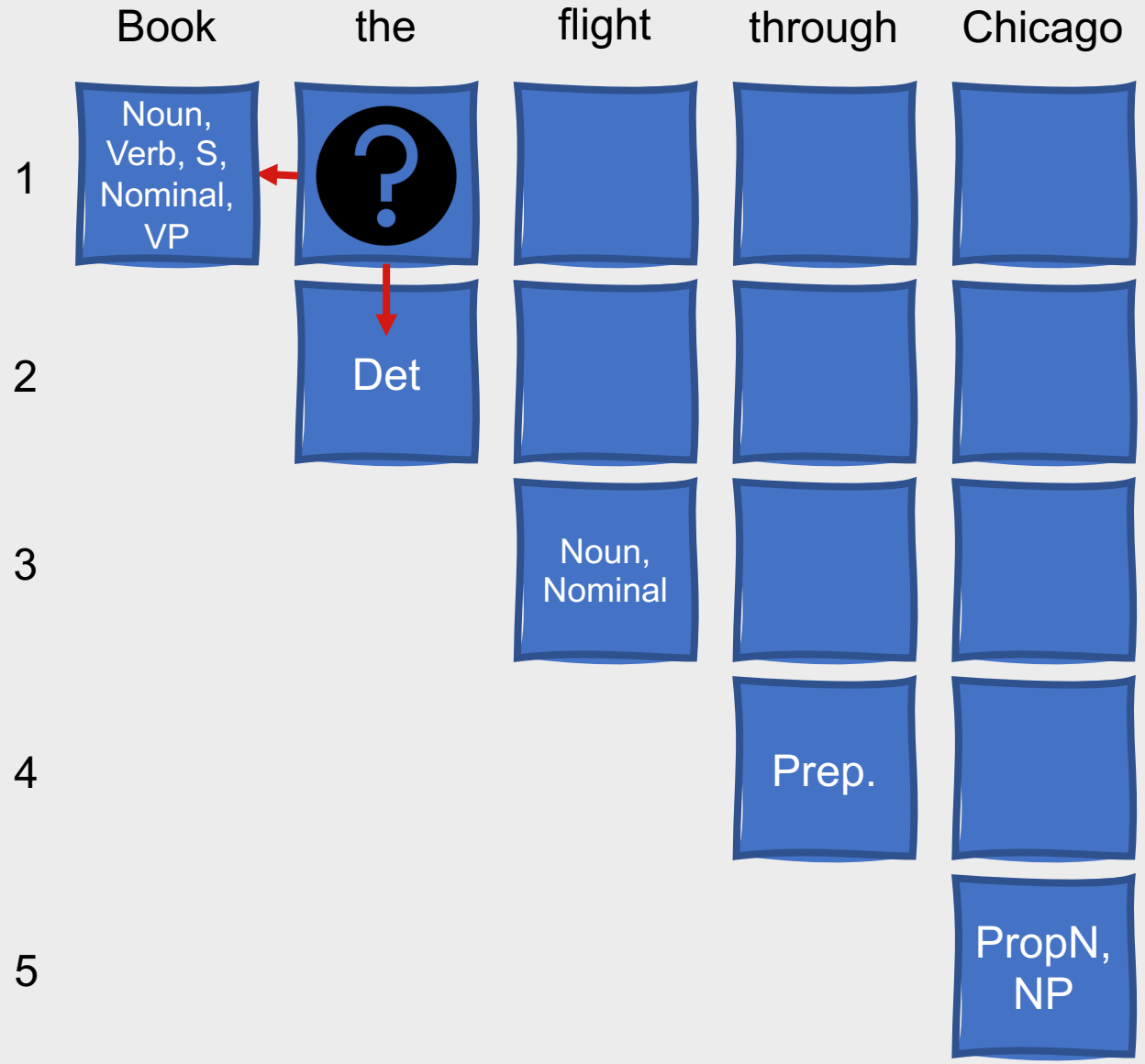
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → **Chicago** | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

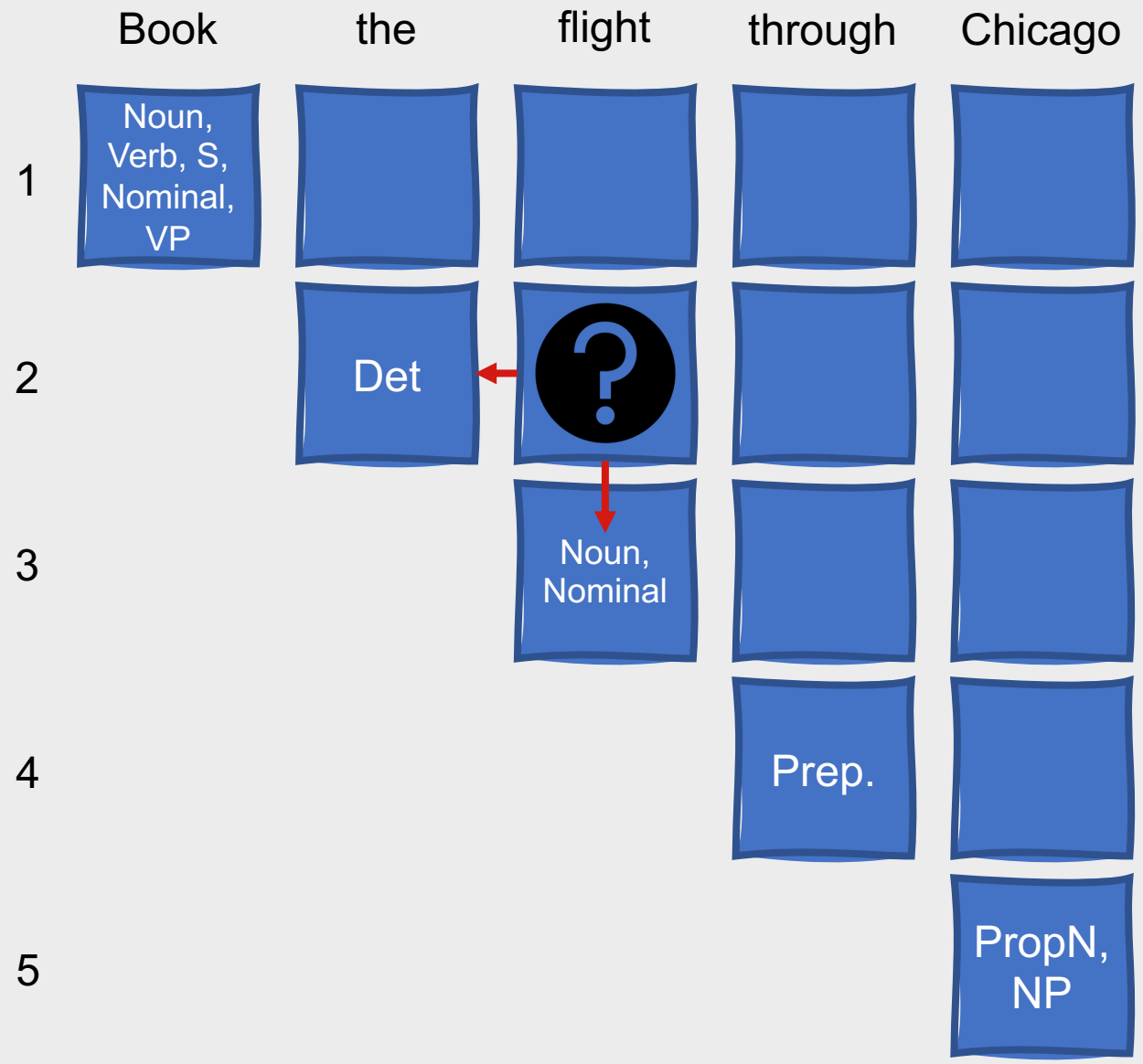
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

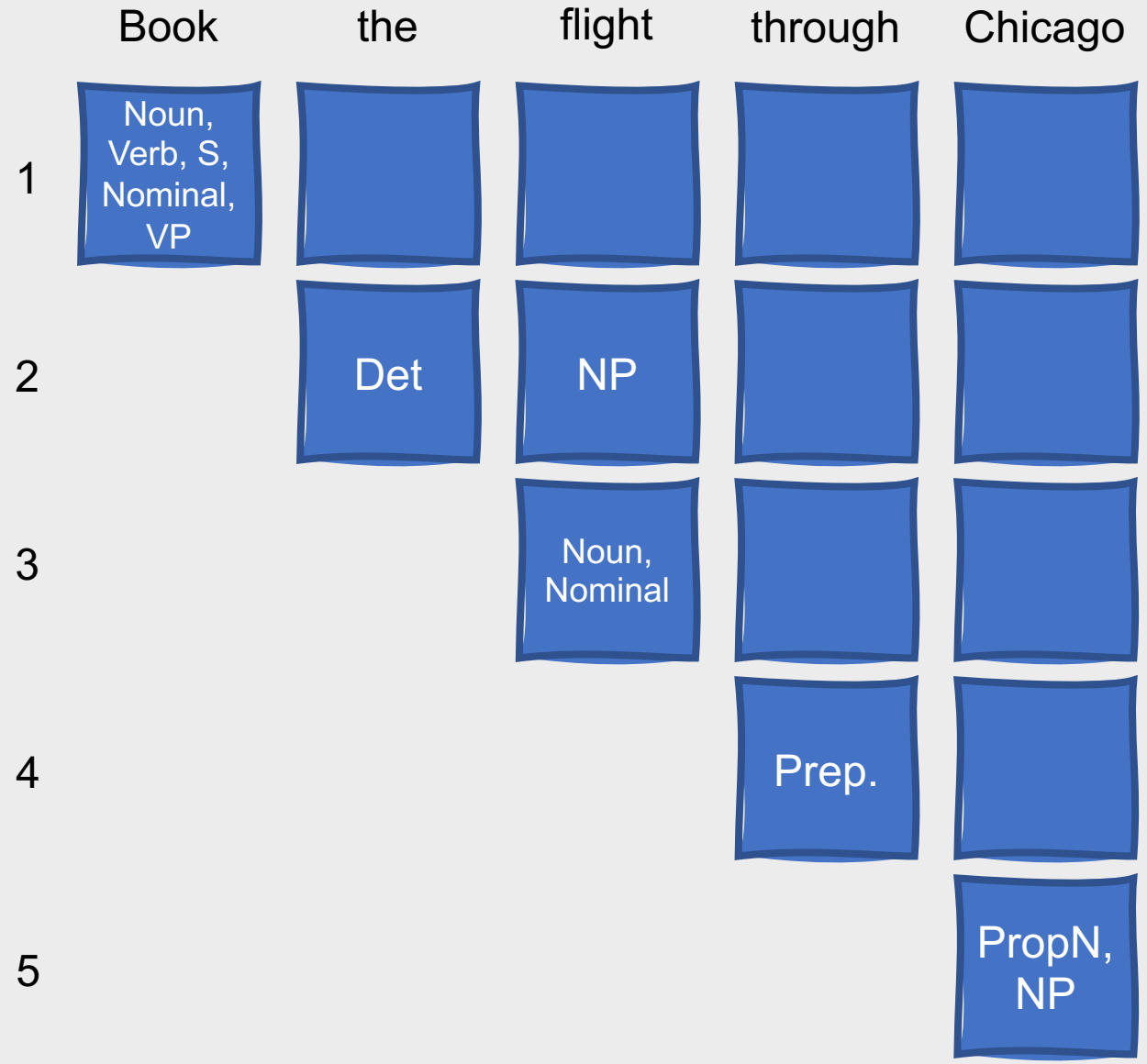
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

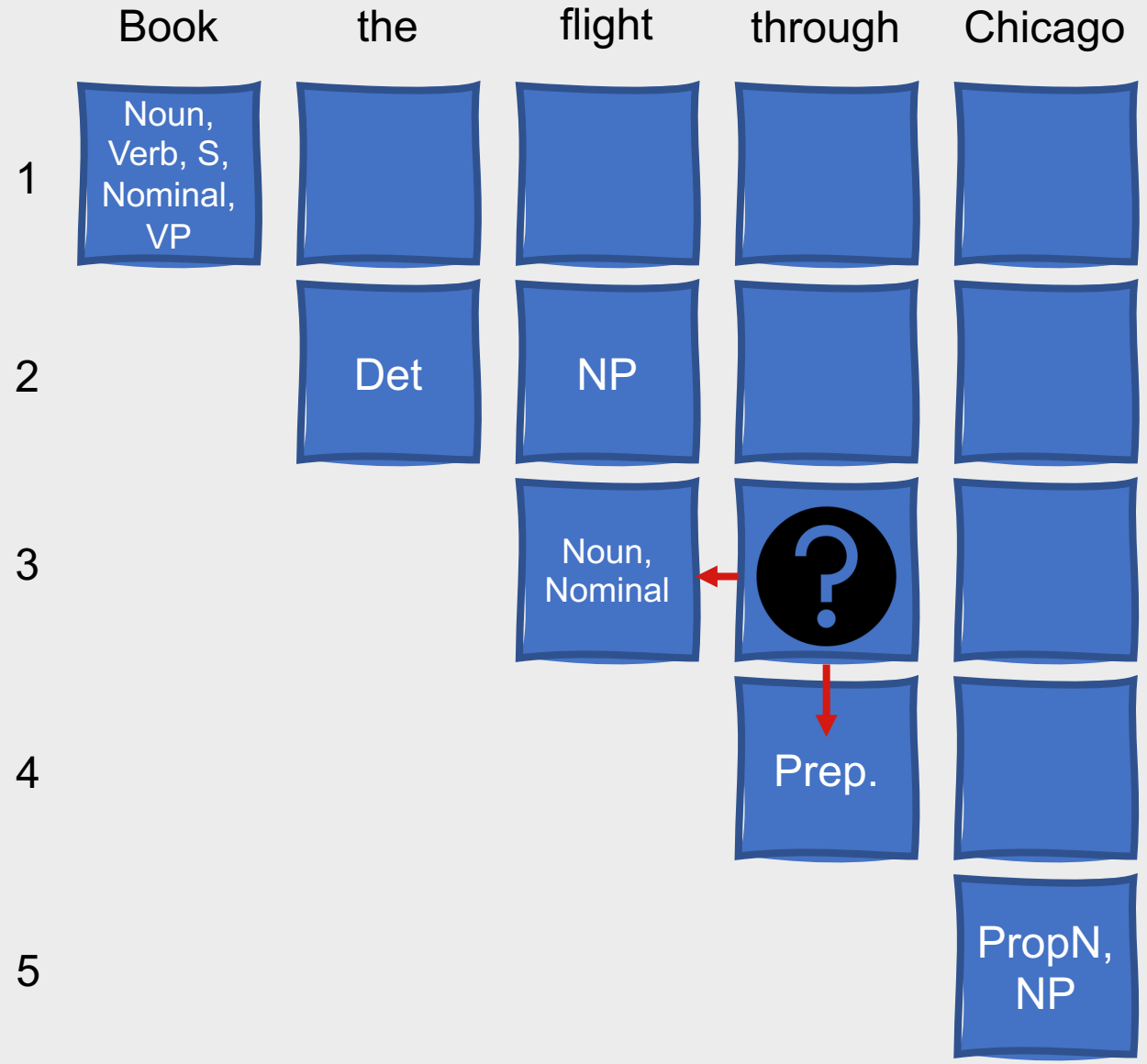
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → **Det Nominal**
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

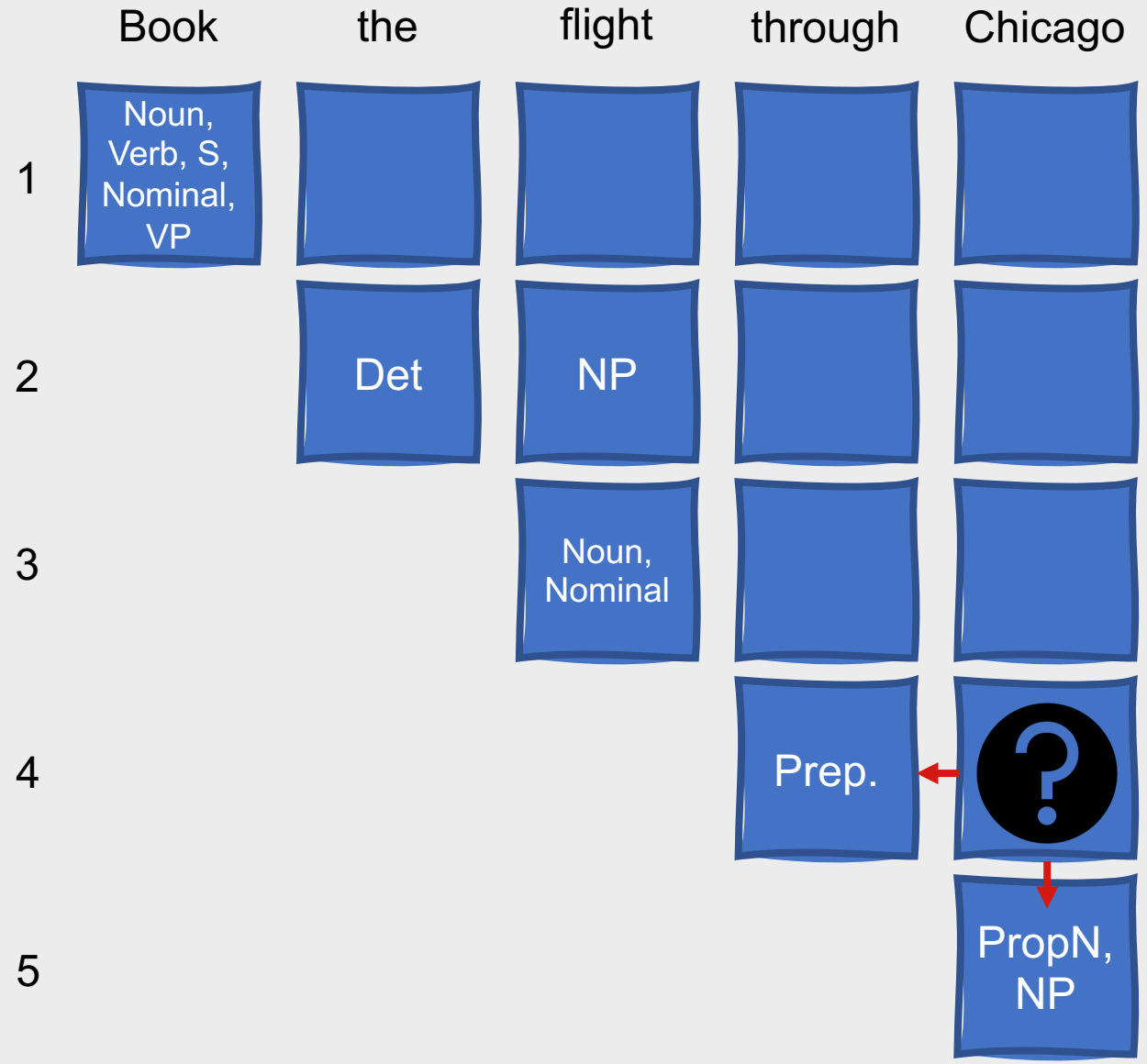
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

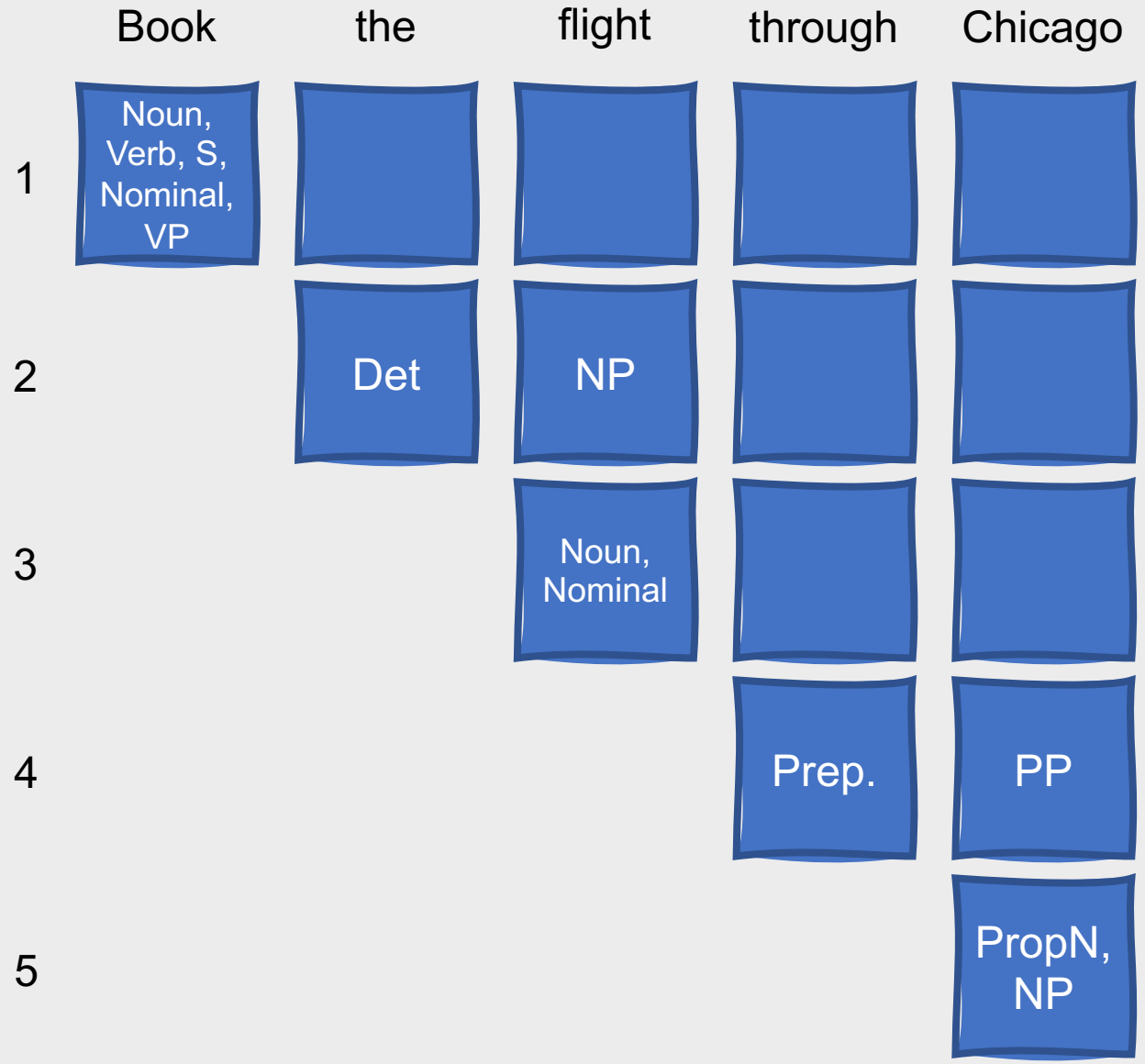
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

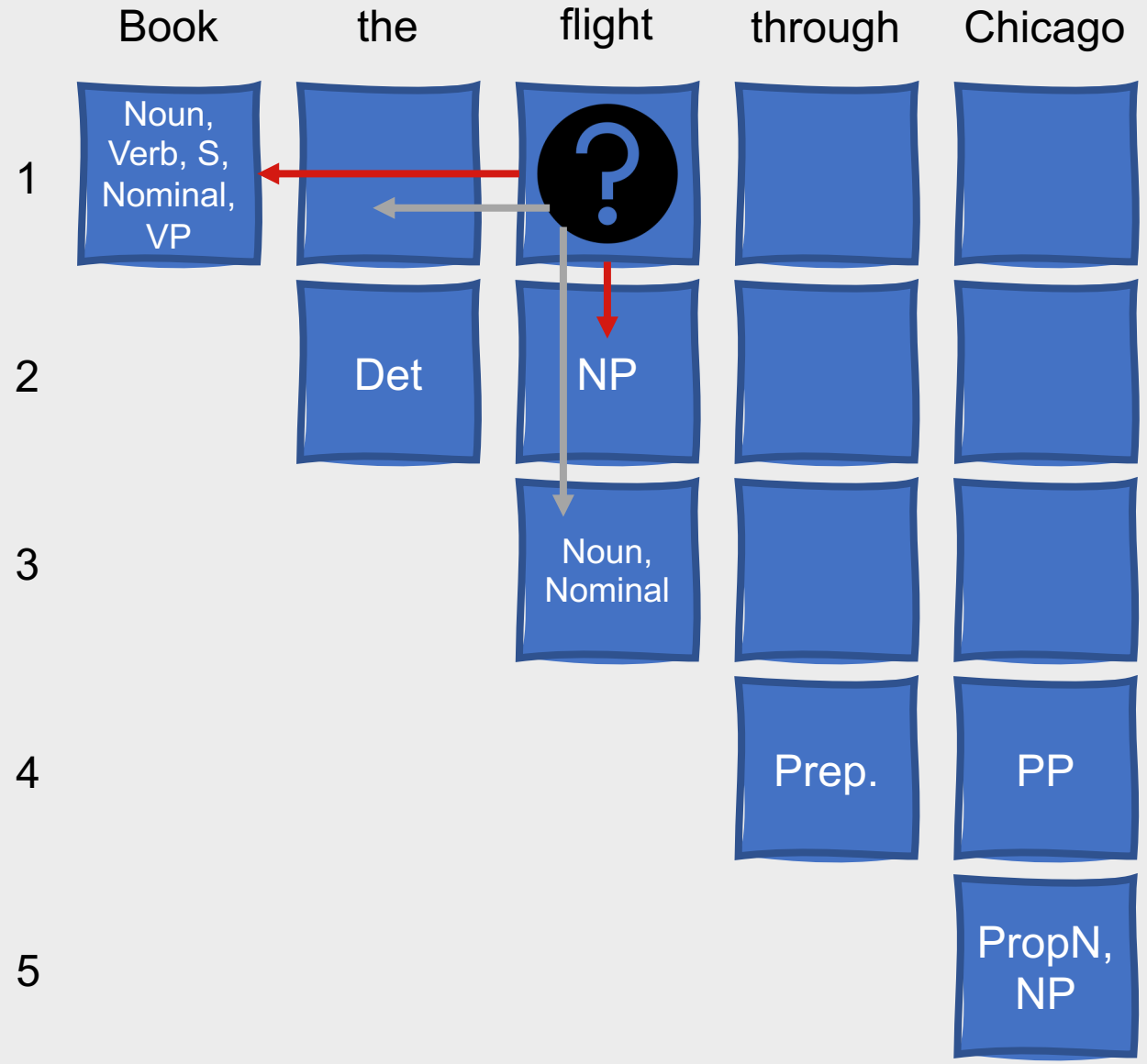
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → **Preposition NP**



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

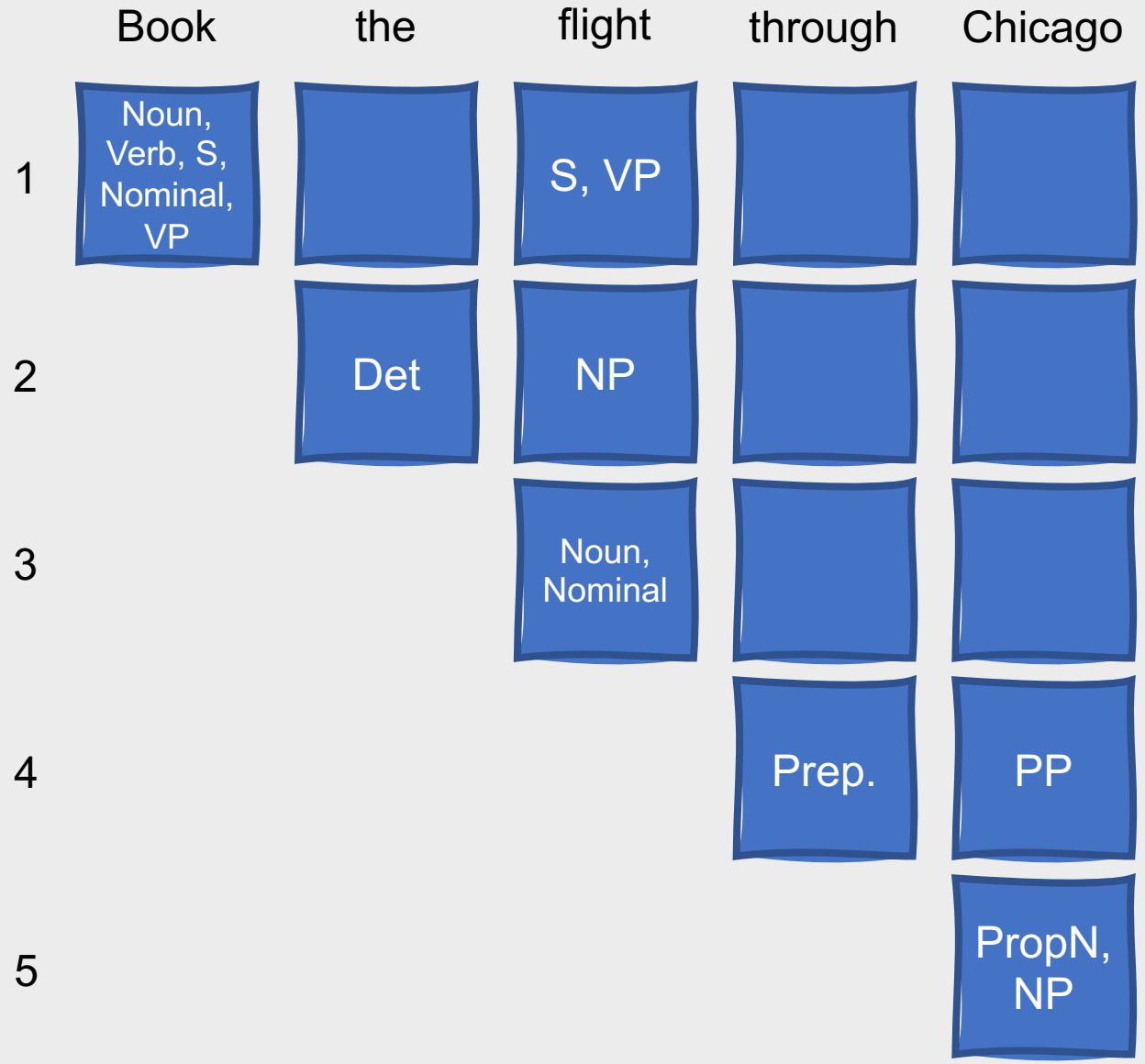
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

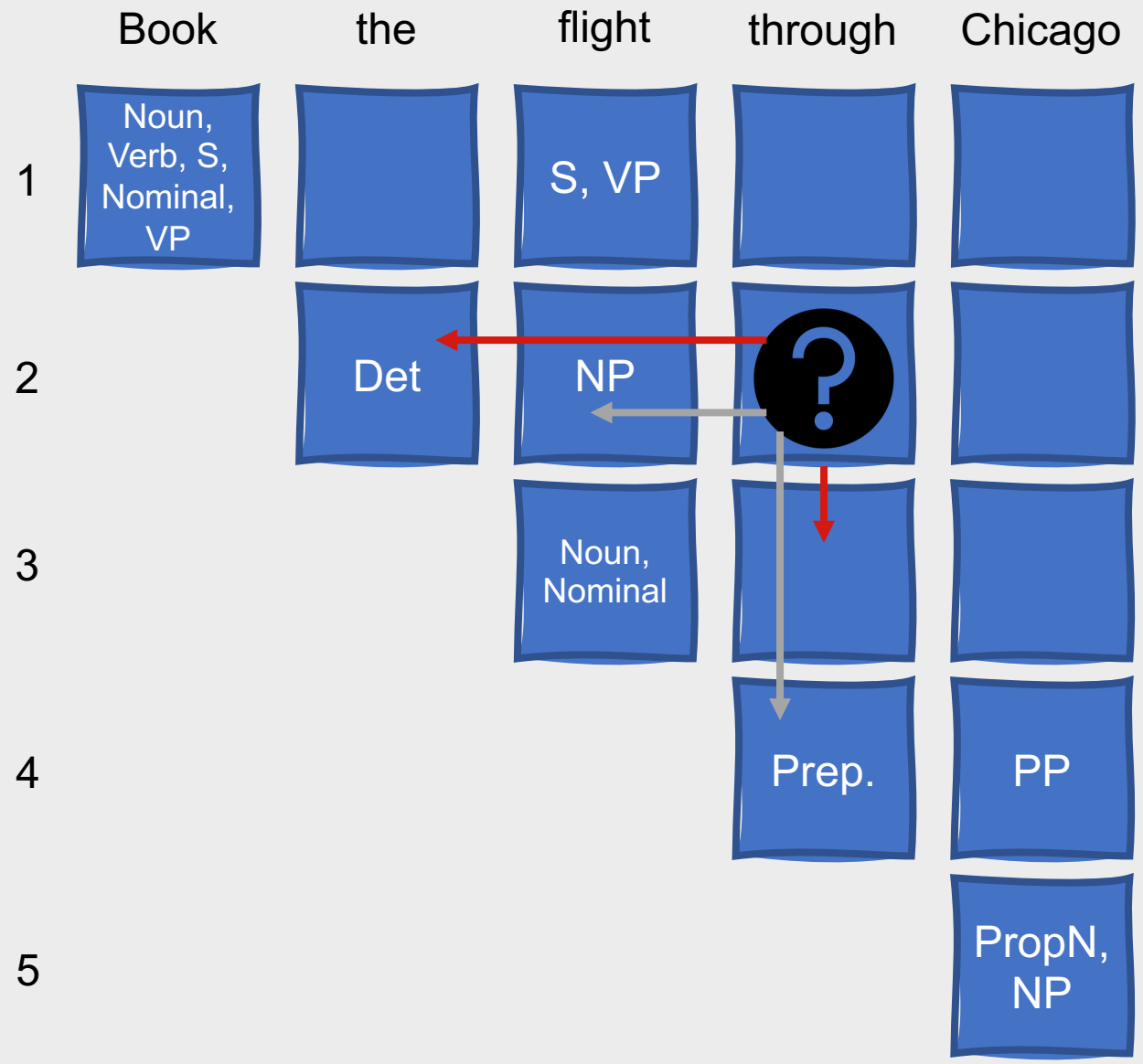
S → NP VP
S → **VP** → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → **Verb NP**
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

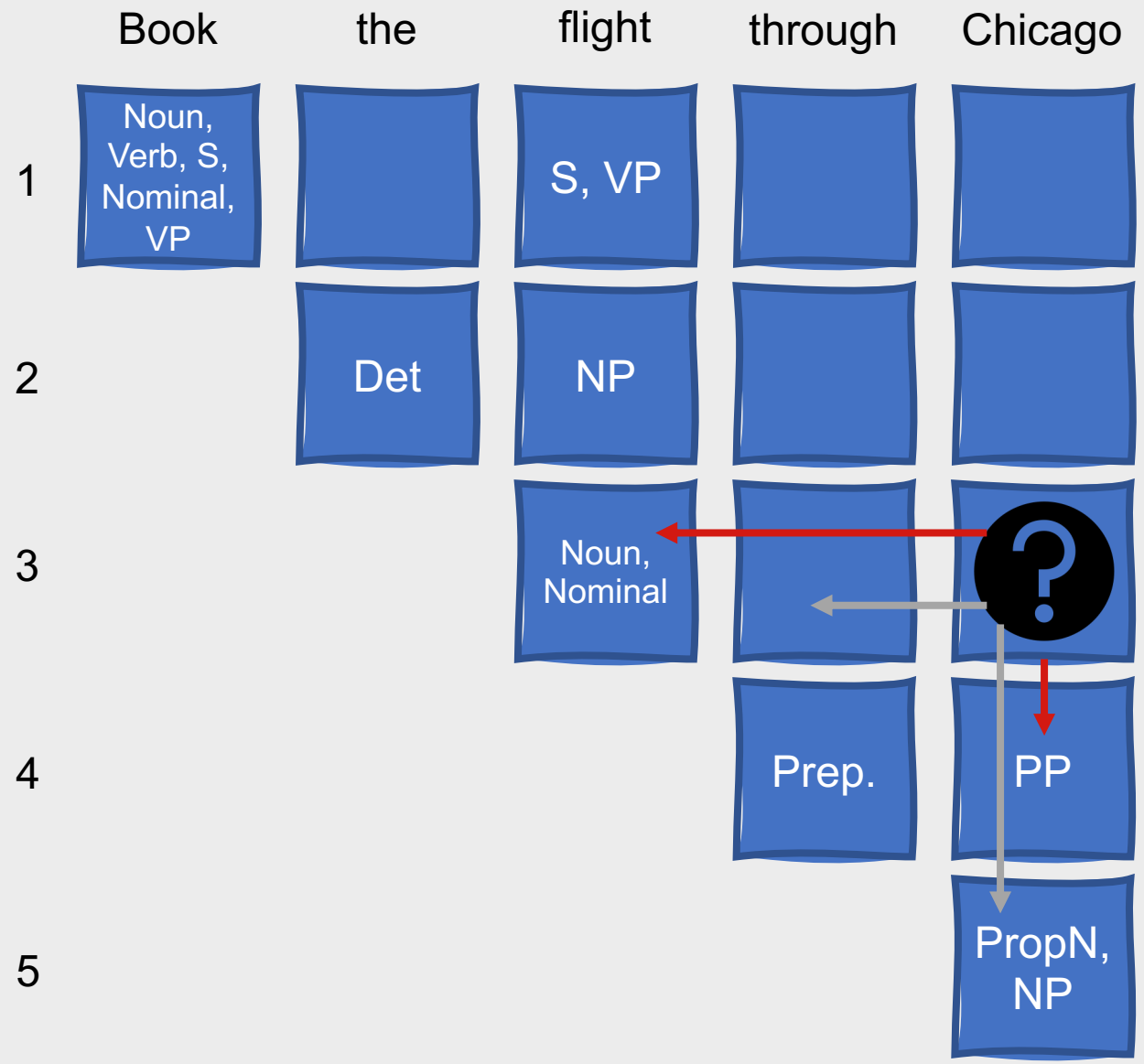
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

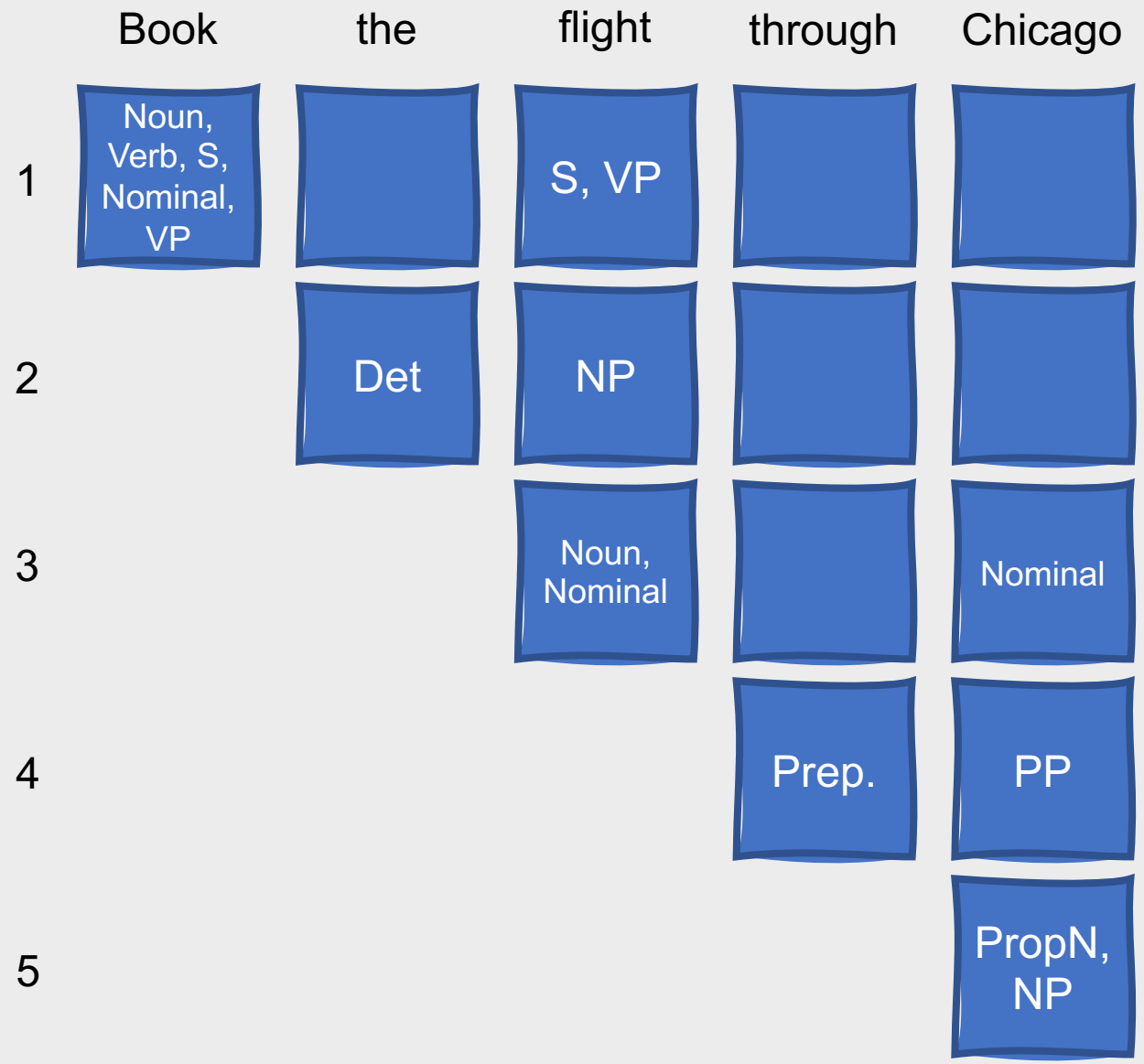
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

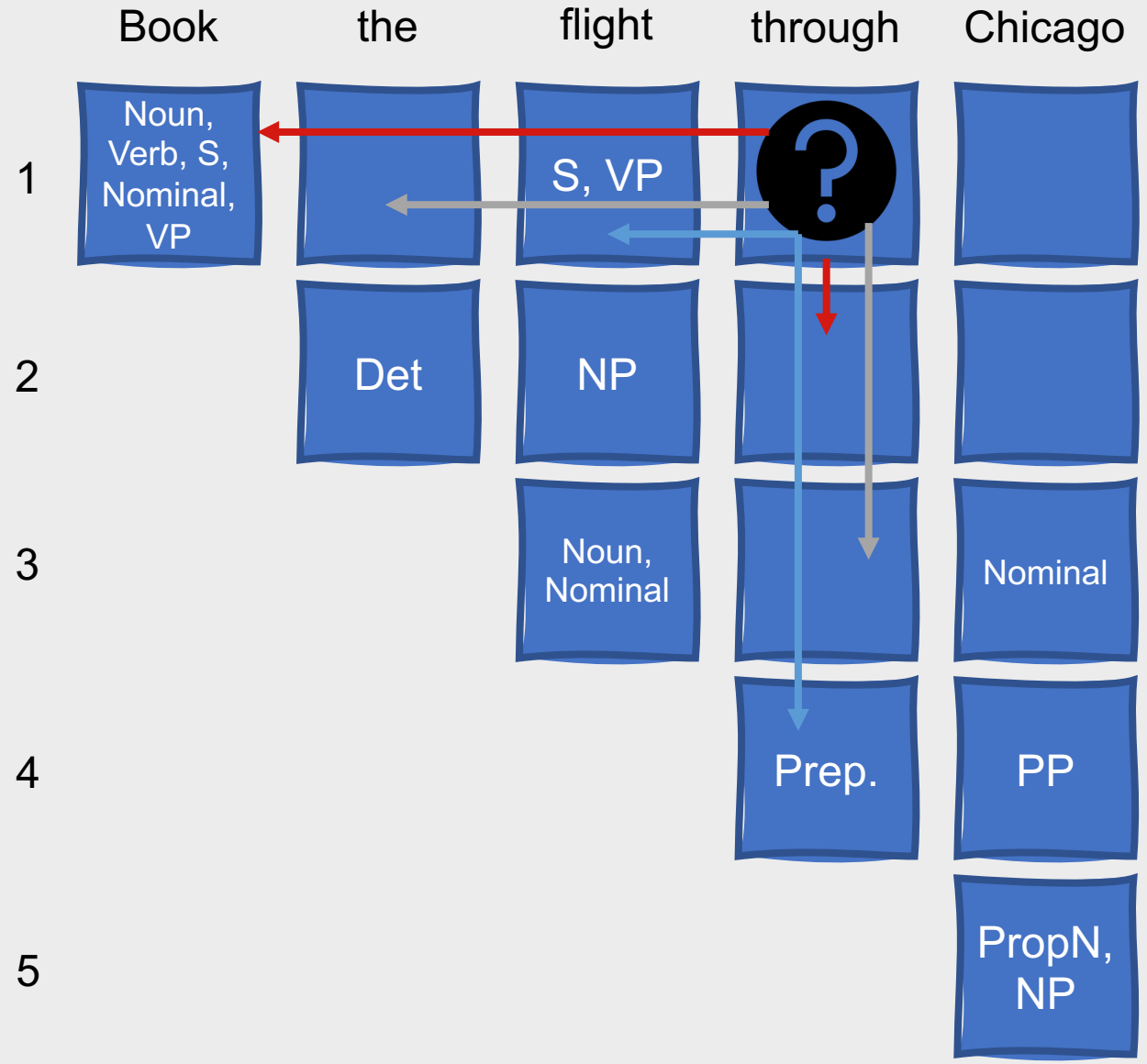
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → **Nominal PP**
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

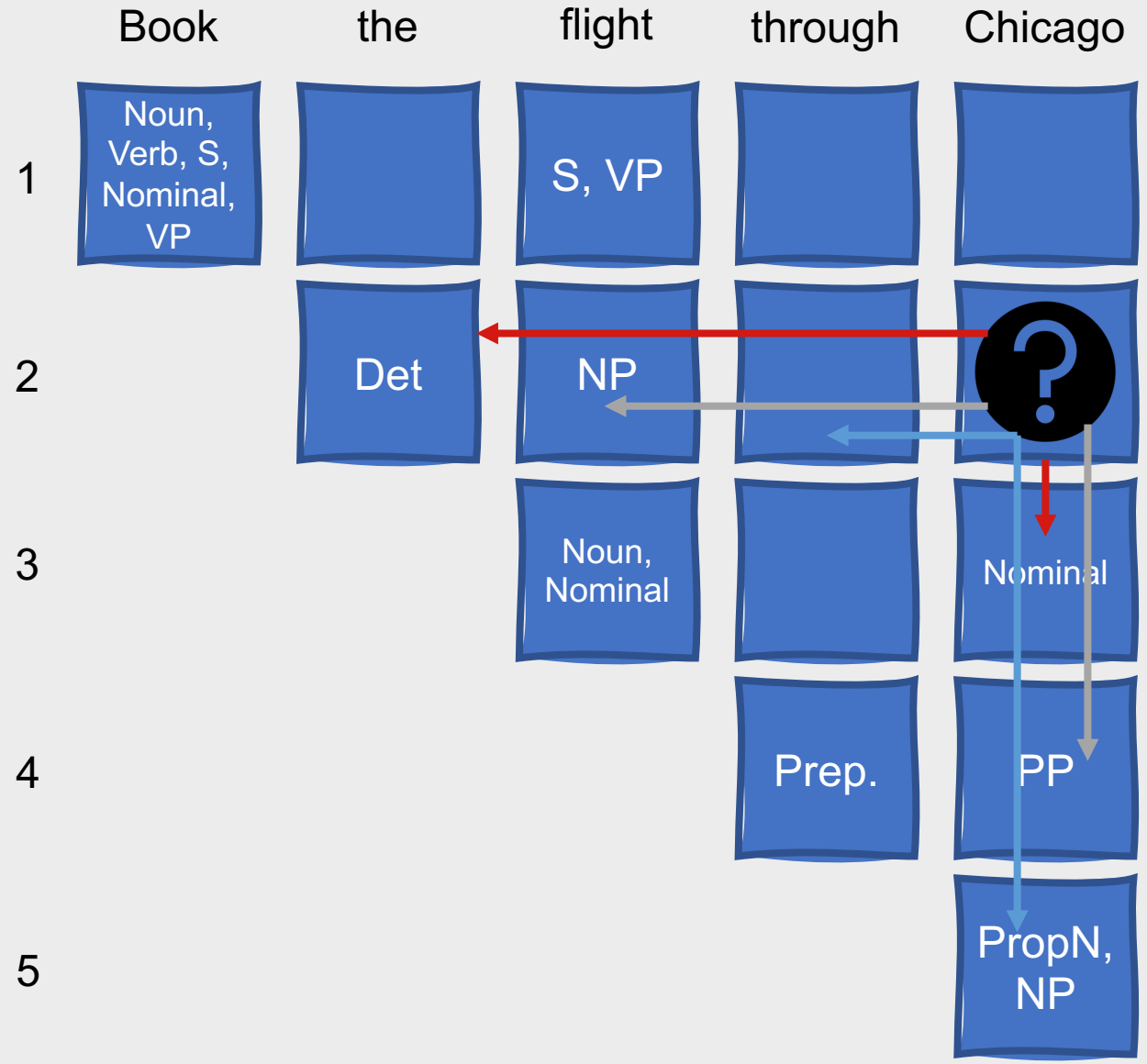
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

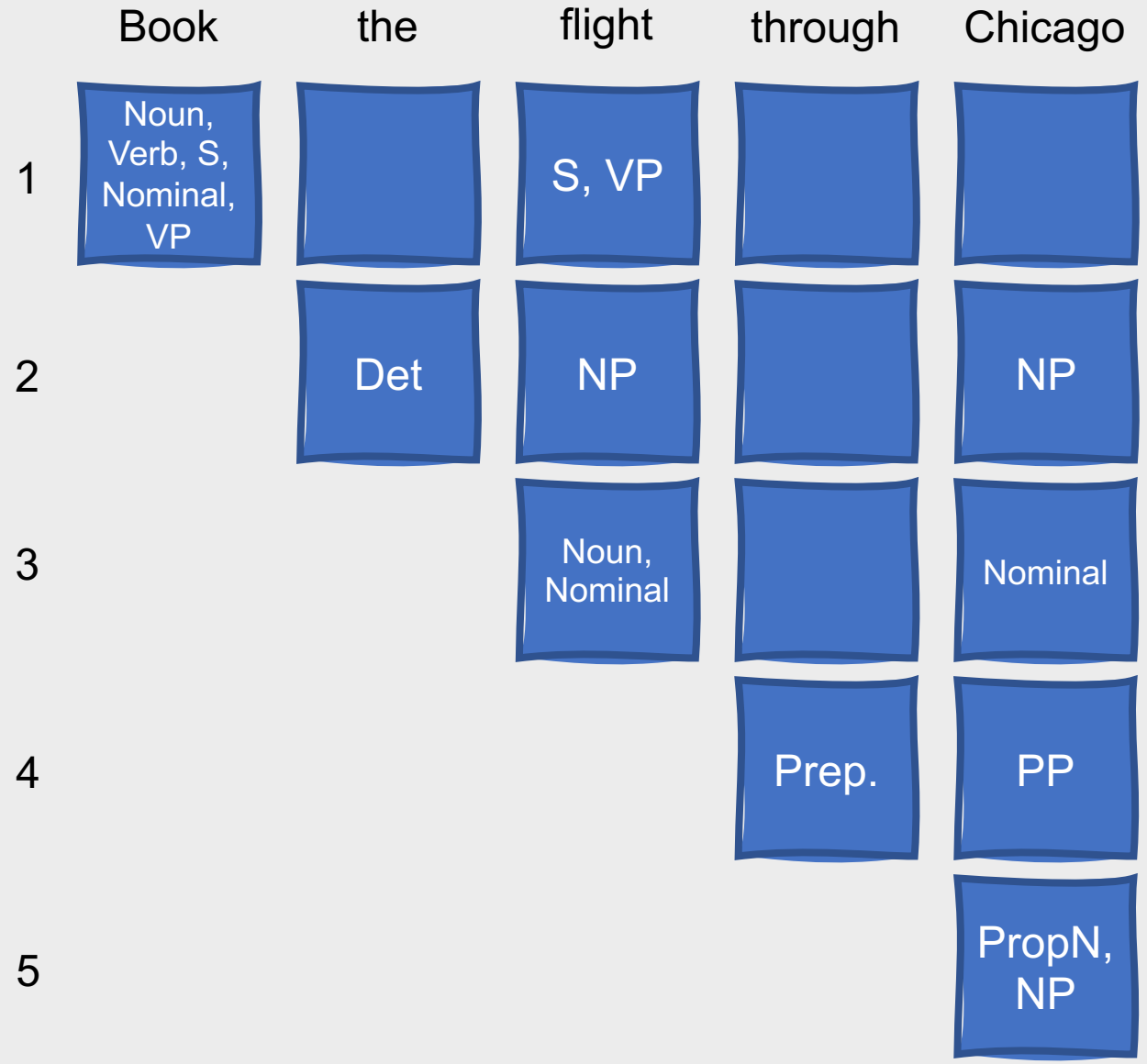
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

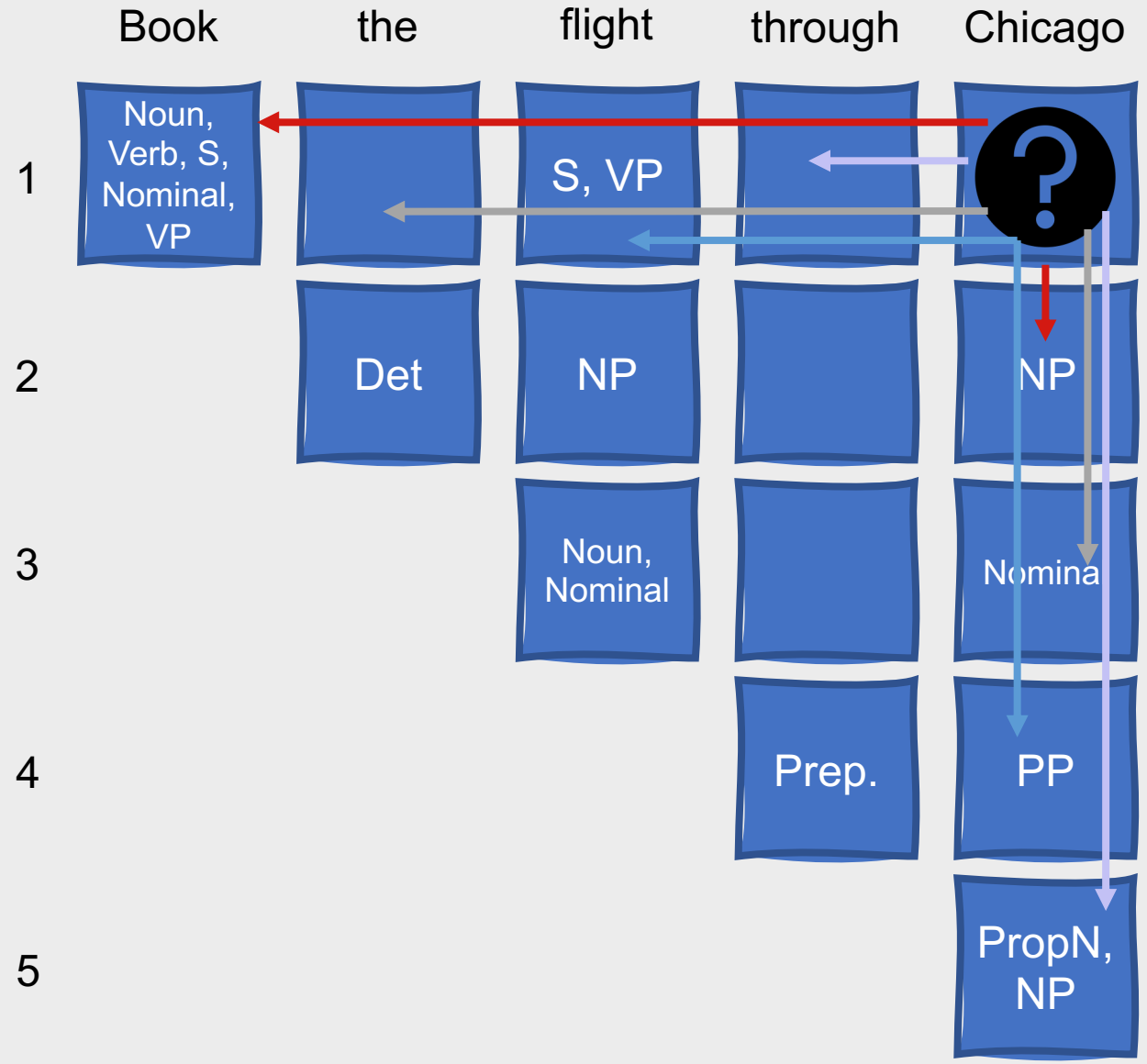
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → **Det Nominal**
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

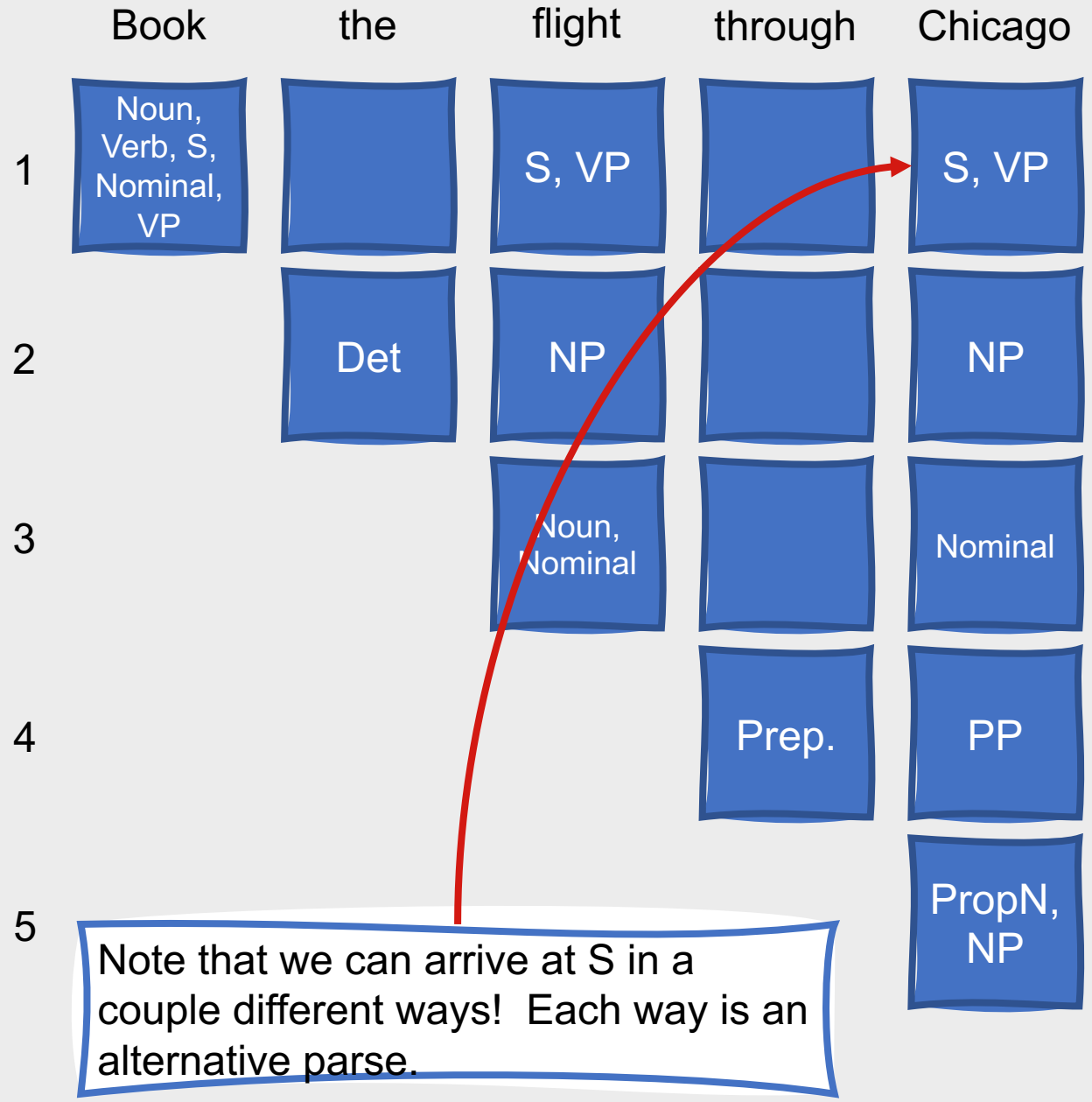
S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

S → NP VP
S → **VP** → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → **Verb NP**
VP → Verb PP
VP → **VP PP**
PP → Preposition NP



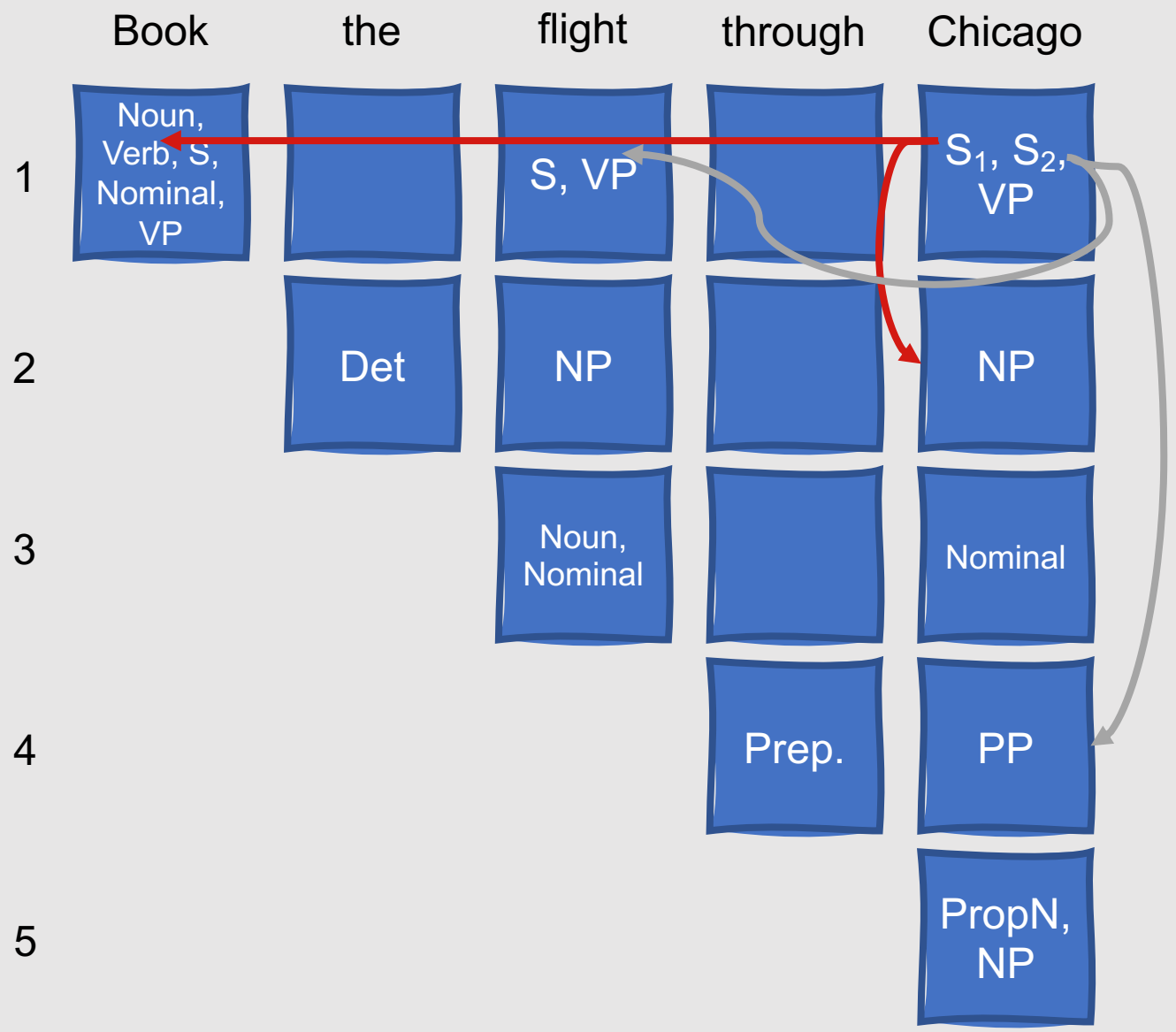
CKY Algorithm

- The example we just saw functions as a **recognizer** …for it to succeed (i.e., find a valid sentence according to this grammar), is simply needs to find an S in cell [0,n]
- To return all possible parses, we need to make two changes to the algorithm:
  - Pair each non-terminal with pointers to the table entries from which it was derived
  - Permit multiple versions of the same non-terminal to be entered into the table
- Then, we can choose an S from cell [0,n] and recursively retrieve its component constituents from the table

CKY Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Chicago | Dallas
Aux → does
Preposition → from | to | on | near | through

S → NP VP
S → VP → Verb → book | include | prefer
NP → Pronoun → I | she | me
NP → Proper-Noun → Chicago | Dallas
NP → Det Nominal
Nominal → Noun → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb → book | include | prefer
VP → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



CKY Complexity

Time Complexity: $O(n^3)$

Space Complexity: $O(n^2)$

How can we improve upon CKY?

- Fill table in a single sweep over the input words, using a top-down approach
- Table is length *n*+1, where *n* is equivalent to the number of words
- Table entries contain three types of information:
  - A subtree corresponding to a single grammar rule
  - Information about the progress made in completing the subtree
  - The position of the subtree with respect to the input
- This is called Earley parsing

In Earley parsing, table entries are known as states.

- States include structures called **dotted rules**
- A • within the righthand side of a state's grammar rule indicates the progress made towards recognizing it
- A state's **position with respect to the input is represented by two numbers**, indicating (1) where the state begins, and (2) where its dot lies

Example States

- Input: Book that flight.
- S → • VP, [0,0]
  - Top-down prediction for this particular kind of S
  - First 0: Constituent predicted by this state should begin at the start of the input
  - Second 0: Dot lies at the start of the input as well
- NP → Det • Nominal, [1,2]
  - NP begins at position 1
  - Det has been successfully parsed
  - Nominal is expected next
- VP → V NP •, [0,3]
  - Successful discovery of a tree corresponding to a VP that spans the entire input

Earley Algorithm

- An Earley parser moves through the *n*+1 sets of states in a chart in a left-to-right fashion, processing the states within each set in order
- At each step, one of three operators is applied to each state depending on its status
  - Predictor
  - Scanner
  - Completer
- This results in the addition of new states to the end of either the current or next set of states in the chart
- States are never removed
- The algorithm never backtracks
- The presence of S $\rightarrow \alpha$ •, [0,*n*] in the list of states in the last chart entry indicates a successful parse

Earley Operators: Predictor

Predictor

- Creates new states representing top-down expectations
- Applied to any state that has a non-terminal immediately to the right of its dot (as long as the non-terminal is not a POS category)
- New states are placed into the same chart entry as the generating state
- They begin and end at the same point in the input where the generating state ends

S → • VP, [0,0]

- VP → • Verb, [0,0]
- VP → • Verb NP, [0,0]
- VP → • Verb NP PP, [0,0]
- VP → • Verb PP, [0,0]
- VP → • VP PP, [0,0]

Earley Operators: Scanner

- Used when a state has a POS category to the right of the dot
- Examines input and incorporates a state corresponding to the prediction of a word with a particular POS into the chart
- Does so by creating a new state from the input state with the dot advanced over the predicted input category
- VP → • Verb NP, [0,0]
  - Since category following the dot is a part of speech (Verb)….
  - Verb → book •, [0,1]

Earley Operators: Completer

- Applied to a state when its dot has reached the right end of the rule
- Indicates that the parser has successfully discovered a particular grammatical category over some span of input
- Purpose: Find and advance all previously created states that were looking for this grammatical category at this position in the input
- New states are created by copying the older state, advancing the dot over the expected category, and installing the new state in the current chart entry
- NP → Det Nominal •, [1,3]
  - What incomplete states end at position 1 and expect an NP?
  - VP → Verb • NP, [0,1]
  - VP → Verb • NP PP, [0,1]
  - So, add VP → Verb NP •, [0,3] and the new incomplete VP → Verb NP • PP, [0,3] to the chart

Earley Algorithm: Example

Chart	State	Rule	Start, End	Operator
0	S0	$\gamma \rightarrow \bullet$ S	0, 0	Start State
0	S1	S $\rightarrow \bullet$ NP VP	0, 0	Predictor
0	S2	S $\rightarrow \bullet$ VP	0, 0	Predictor
0	S3	NP $\rightarrow \bullet$ Det Nominal	0, 0	Predictor
0	S4	VP $\rightarrow \bullet$ Verb	0, 0	Predictor
0	S5	VP $\rightarrow \bullet$ Verb NP	0, 0	Predictor

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer

S → NP VP
S → VP
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
VP → Verb
VP → Verb NP

Earley Algorithm: Example

Chart	State	Rule	Start, End	Operator
0	S0	$\gamma \rightarrow \bullet$ S	0, 0	Start State
0	S1	S $\rightarrow \bullet$ NP VP	0, 0	Predictor
0	S2	S $\rightarrow \bullet$ VP	0, 0	Predictor
0	S3	NP $\rightarrow \bullet$ Det Nominal	0, 0	Predictor
0	S4	VP $\rightarrow \bullet$ Verb	0, 0	Predictor
0	S5	VP $\rightarrow \bullet$ Verb NP	0, 0	Predictor
1	S6	Verb $\rightarrow$ book $\bullet$	0, 1	Scanner
1	S7	VP $\rightarrow$ Verb $\bullet$	0, 1	Completer
1	S8	VP $\rightarrow$ Verb $\bullet$ NP	0, 1	Completer
1	S9	S $\rightarrow$ VP $\bullet$	0, 1	Completer
1	S10	NP $\rightarrow \bullet$ Det Nominal	1, 1	Predictor

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer

S → NP VP
S → VP
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
VP → Verb
VP → Verb NP

Earley Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer

S → NP VP
S → VP
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
VP → Verb
VP → Verb NP

Chart	State	Rule	Start, End	Operator
0	S0	γ → • S	0, 0	Start State
0	S1	S → • NP VP	0, 0	Predictor
0	S2	S → • VP	0, 0	Predictor
0	S3	NP → • Det Nominal	0, 0	Predictor
0	S4	VP → • Verb	0, 0	Predictor
0	S5	VP → • Verb NP	0, 0	Predictor
1	S6	Verb → book •	0, 1	Scanner
1	S7	VP → Verb •	0, 1	Completer
1	S8	VP → Verb • NP	0, 1	Completer
1	S9	S → VP •	0, 1	Completer
1	S10	NP → • Det Nominal	1, 1	Predictor
2	S11	Det → that •	1, 2	Scanner
2	S12	NP → Det • Nominal	1, 2	Completer
2	S13	Nominal → • Noun	2, 2	Predictor
2	S14	Nominal → • Nominal Noun	2, 2	Predictor

Earley Algorithm: Example

Det → that | this | a | the
Noun → book | flight | meal | money
Verb → book | include | prefer

S → NP VP
S → VP
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
VP → Verb
VP → Verb NP

Chart	State	Rule	Start, End	Operator
0	S0	γ → • S	0, 0	Start State
0	S1	S → • NP VP	0, 0	Predictor
0	S2	S → • VP	0, 0	Predictor
0	S3	NP → • Det Nominal	0, 0	Predictor
0	S4	VP → • Verb	0, 0	Predictor
0	S5	VP → • Verb NP	0, 0	Predictor
1	S6	Verb → book •	0, 1	Scanner
1	S7	VP → Verb •	0, 1	Completer
1	S8	VP → Verb • NP	0, 1	Completer
1	S9	S → VP •	0, 1	Completer
1	S10	NP → • Det Nominal	1, 1	Predictor
2	S11	Det → that •	1, 2	Scanner
2	S12	NP → Det • Nominal	1, 2	Completer
2	S13	Nominal → • Noun	2, 2	Predictor
2	S14	Nominal → • Nominal Noun	2, 2	Predictor
3	S15	Noun → flight •	2, 3	Scanner
3	S16	Nominal → Noun •	2, 3	Completer
3	S17	NP → Det Nominal •	1, 3	Completer
3	S18	Nominal → Nominal • Noun	2, 3	Completer
3	S19	VP → Verb NP •	0, 3	Completer
3	S20	S → VP •	0, 3	Completer

Which states participate in the final parse?

Chart	State	Rule	Start, End	Operator
0	S0	$\gamma \rightarrow \bullet$ S	0, 0	Start State
0	S1	S $\rightarrow \bullet$ NP VP	0, 0	Predictor
0	S2	S $\rightarrow \bullet$ VP	0, 0	Predictor
0	S3	NP $\rightarrow \bullet$ Det Nominal	0, 0	Predictor
0	S4	VP $\rightarrow \bullet$ Verb	0, 0	Predictor
0	S5	VP $\rightarrow \bullet$ Verb NP	0, 0	Predictor
1	S6	Verb $\rightarrow$ book $\bullet$	0, 1	Scanner
1	S7	VP $\rightarrow$ Verb $\bullet$	0, 1	Completer
1	S8	VP $\rightarrow$ Verb $\bullet$ NP	0, 1	Completer
1	S9	S $\rightarrow$ VP $\bullet$	0, 1	Completer
1	S10	NP $\rightarrow \bullet$ Det Nominal	1, 1	Predictor
2	S11	Det $\rightarrow$ that $\bullet$	1, 2	Scanner
2	S12	NP $\rightarrow$ Det $\bullet$ Nominal	1, 2	Completer
2	S13	Nominal $\rightarrow \bullet$ Noun	2, 2	Predictor
2	S14	Nominal $\rightarrow \bullet$ Nominal Noun	2, 2	Predictor
3	S15	Noun $\rightarrow$ flight $\bullet$	2, 3	Scanner
3	S16	Nominal $\rightarrow$ Noun $\bullet$	2, 3	Completer
3	S17	NP $\rightarrow$ Det Nominal $\bullet$	1, 3	Completer
3	S18	Nominal $\rightarrow$ Nominal $\bullet$ Noun	2, 3	Completer
3	S19	VP $\rightarrow$ Verb NP $\bullet$	0, 3	Completer
3	S20	S $\rightarrow$ VP $\bullet$	0, 3	Completer

As with CKY, the example algorithm acted as a recognizer.

- We can retrieve parse trees by adding a field to store information about the completed states that generated constituents
- How to do this?
  - Have the Completer operator add a pointer to the previous state onto a list of constituent states for the new state
  - When an S is found in the final chart, just follow pointers backward

Which states participate in the final parse?

Chart	State	Rule	Start, End	Operator (Backward Pointer)
0	S0	$\gamma \rightarrow \bullet$ S	0, 0	Start State
0	S1	S $\rightarrow \bullet$ NP VP	0, 0	Predictor
0	S2	S $\rightarrow \bullet$ VP	0, 0	Predictor
0	S3	NP $\rightarrow \bullet$ Det Nominal	0, 0	Predictor
0	S4	VP $\rightarrow \bullet$ Verb	0, 0	Predictor
0	S5	VP $\rightarrow \bullet$ Verb NP	0, 0	Predictor
1	S6	Verb $\rightarrow$ book $\bullet$	0, 1	Scanner
1	S7	VP $\rightarrow$ Verb $\bullet$	0, 1	Completer
1	S8	VP $\rightarrow$ Verb $\bullet$ NP	0, 1	Completer
1	S9	S $\rightarrow$ VP $\bullet$	0, 1	Completer
1	S10	NP $\rightarrow \bullet$ Det Nominal	1, 1	Predictor
2	S11	Det $\rightarrow$ that $\bullet$	1, 2	Scanner
2	S12	NP $\rightarrow$ Det $\bullet$ Nominal	1, 2	Completer
2	S13	Nominal $\rightarrow \bullet$ Noun	2, 2	Predictor
2	S14	Nominal $\rightarrow \bullet$ Nominal Noun	2, 2	Predictor
3	S15	Noun $\rightarrow$ flight $\bullet$	2, 3	Scanner
3	S16	Nominal $\rightarrow$ Noun $\bullet$	2, 3	Completer (S15)
3	S17	NP $\rightarrow$ Det Nominal $\bullet$	1, 3	Completer (S11, S15)
3	S18	Nominal $\rightarrow$ Nominal $\bullet$ Noun	2, 3	Completer
3	S19	VP $\rightarrow$ Verb NP $\bullet$	0, 3	Completer (S6, S17)
3	S20	S $\rightarrow$ VP $\bullet$	0, 3	Completer (S19)

Summary: Syntactic Parsing

- **Syntactic parsing** is the process of automatically determining the grammatical structure of an input sentence
- Language is ambiguous, so **sentences can have multiple grammatically-correct parses**
- Parsing can be performed using either a **top-down** or **bottom-up** approach
- Common algorithms for syntactic parsing include:
  - **CKY**
  - **Earley**