CKY Algorithm

Miguel Ballesteros

Algorithms for NLP Course.
7-11

Carnegie Mellon

Some adapted slides by Dor Altshuler and some of Chris's last year slides

CKY

- Kasami, 1965
- Younger, 1967
- Cocke and Schwartz, 1970

CKY Algorithm

- Cocke-Kasami-Younger algorithm.
- Recognition vs. Parsing:
  - Recognition - deciding the membership in the language
  - Parsing – Recognition+ producing a parse tree for it
- Parsing is more “difficult” than recognition (time complexity)
- **CKY:** bottom-up dynamic programming.

Remembering Context Free Grammars

- CFG
  - G = (V, T, P, S)
    - V: a finite set of variables, non-terminal symbols.
    - T: a finite set of terminal symbols (equiv. To Σ in FSAs)
    - P: a set of context free production rules, each of the form
      - $A \rightarrow \alpha$, where $A \in V$, $\alpha \in (V \cup T)^*$
    - S: a start non-terminal $S \in V$
      - Single variables or terminals are denoted by **X, Y, Z**

Remembering ϵ-productions

- Formally, context-free grammars are allowed to have empty productions (ϵ = the empty string):

    VP → V NP

    NP → DT Noun

    NP → ϵ

- These can always be eliminated without changing the language generated by the grammar:
- The grammar above becomes
    - VP → V NP   VP → V ϵ   NP → DT Noun

- The second production rule does not make a lot of sense, then …

    VP → V NP

    VP → V

    NP → DT Noun

CKY

- Chart parsing algorithm.
- One of the earliest recognition and parsing algorithms
- The standard version of CKY can only recognize languages defined by context-free grammars with "Binary Branching"

- Based on a "dynamic programming" approach:
  - Build solutions compositionally from sub-solutions
- Uses the grammar directly.

Remembering
Chomsky Normal Form

- Each production is of the form:

  (i) $A \rightarrow BC$

  (ii) $A \rightarrow a$

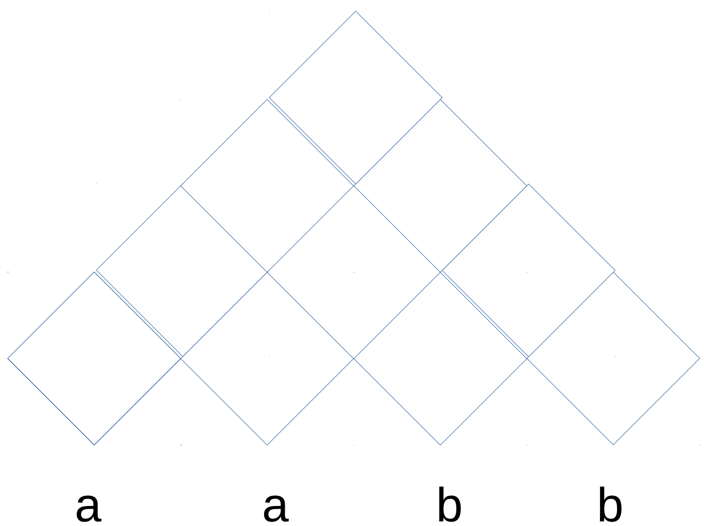
  There you have your “binary branching”

  – We saw that Any context-free language is generated by a context-free grammar in CNF.

CKY

- It is also possible to extend the CKY algorithm to handle some grammars which are not in CNF
  - Harder to understand.

CKY Algorithm for deciding CFL



$S \rightarrow \epsilon \mid AB \mid XB$

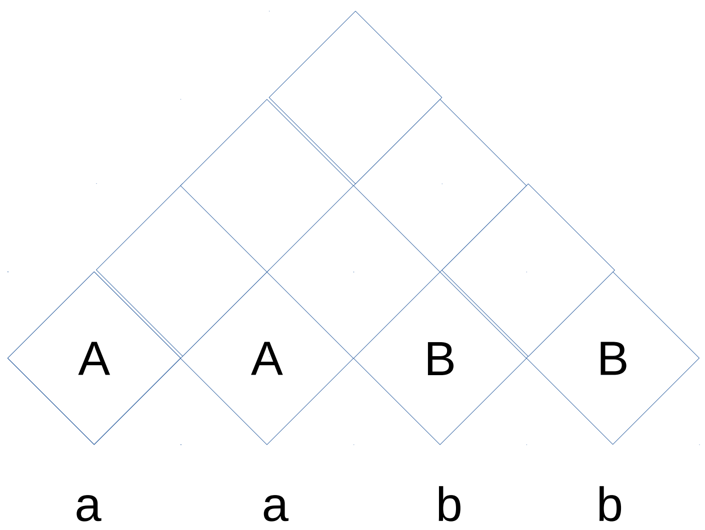
$T \rightarrow AB \mid XB$

$X \rightarrow AT$

$A \rightarrow a$

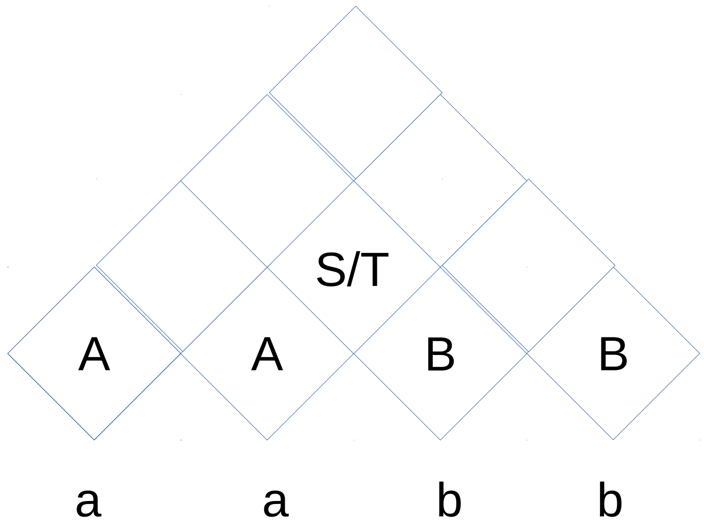
$B \rightarrow b$

CKY Algorithm for deciding CFL



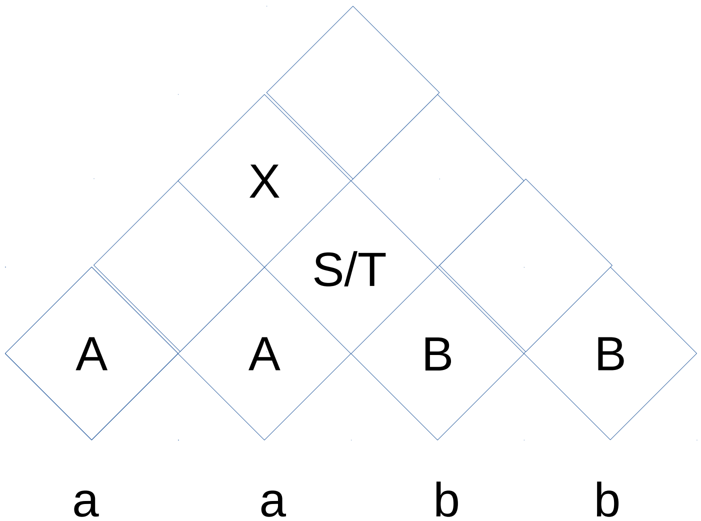
S → ϵ | AB | XB
T → AB | XB
X → AT
A → a
B → b

CKY Algorithm for deciding CFL



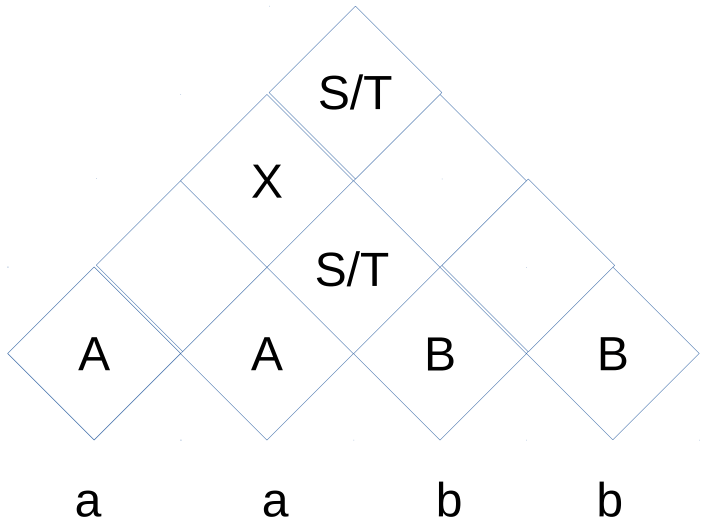
$S \rightarrow \epsilon \mid AB \mid XB$
$T \rightarrow AB \mid XB$
$X \rightarrow AT$
$A \rightarrow a$
$B \rightarrow b$

CKY Algorithm for deciding CFL



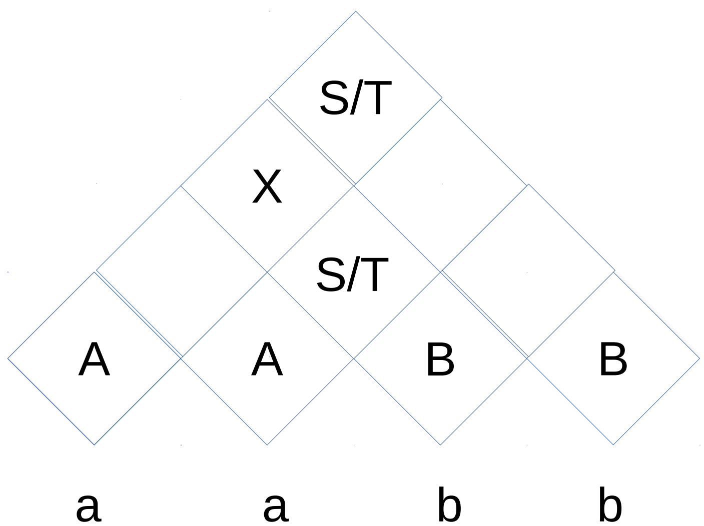
S → ϵ | AB | XB
T → AB | XB
X → AT
A → a
B → b

CKY Algorithm for deciding CFL



S → ϵ | AB | XB
T → AB | XB
X → AT
A → a
B → b

CKY Algorithm for deciding CFL



S → ϵ | AB | XB
T → AB | XB
X → AT
A → a
B → b

S in top square? Yes → a a b b belongs to the language :-)

CKY Algorithm

- function CKY (word w, grammar P) returns table

  for i ← **from** 1 **to** LENGTH(w) **do**

  table[i-1, i] ← {A | A → wi ∈ P }

  for j ← **from** 2 **to** LENGTH(w) **do**

  for i ← **from** j-2 **down to** 0 **do**

  for k ← i + 1 **to** j – 1 **do**

  table[i,j] ← table[i,j] ∪ {A | A → BC ∈ P,
  B ∈ table[i,k], C ∈ table [k,j] }

- If the start symbol S ∈ table[0,n] then **w ∈ L(G)**

CKY

- Observation: any portion of the input string spanning i to j can be split at k, and structure can then be built using sub-solutions spanning i to k and sub-solutions spanning k to j .
- Meaning: Solution to problem [i, j] can constructed from solution to sub problem [i, k] and solution to sub problem [k ,j].

CKY

- Once it has considered sequences of length 1, it goes on to sequences of length 2, and so on.
- For subsequences of length 2 and greater, it considers every possible partition of the subsequence into two halves, and checks to see if there is some production **A -> BC** such that **B** matches the first half and **C** matches the second half. If so, it records **A** as matching the whole subsequence.
- Once this process is completed, the sentence is recognized by the grammar if the entire string is matched by the start symbol: **S**

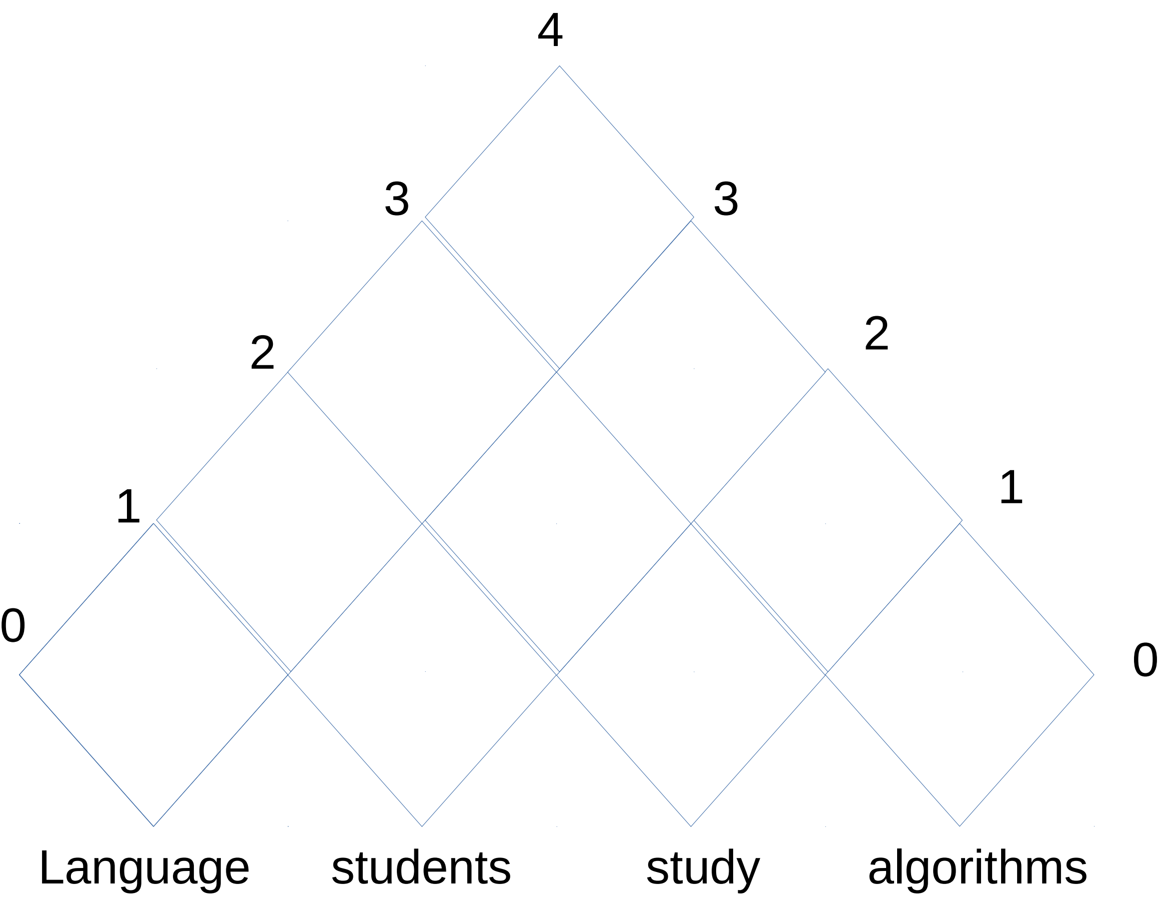
CKY

- Observation: any portion of the input string spanning i to j can be split at k, and structure can then be built using sub-solutions spanning i to k and sub-solutions spanning k to j .

- Meaning: Solution to problem [i, j] can constructed from solution to sub problem [i, k] and solution to sub problem [k ,j].

Phrase structure parsing
CKY algorithm

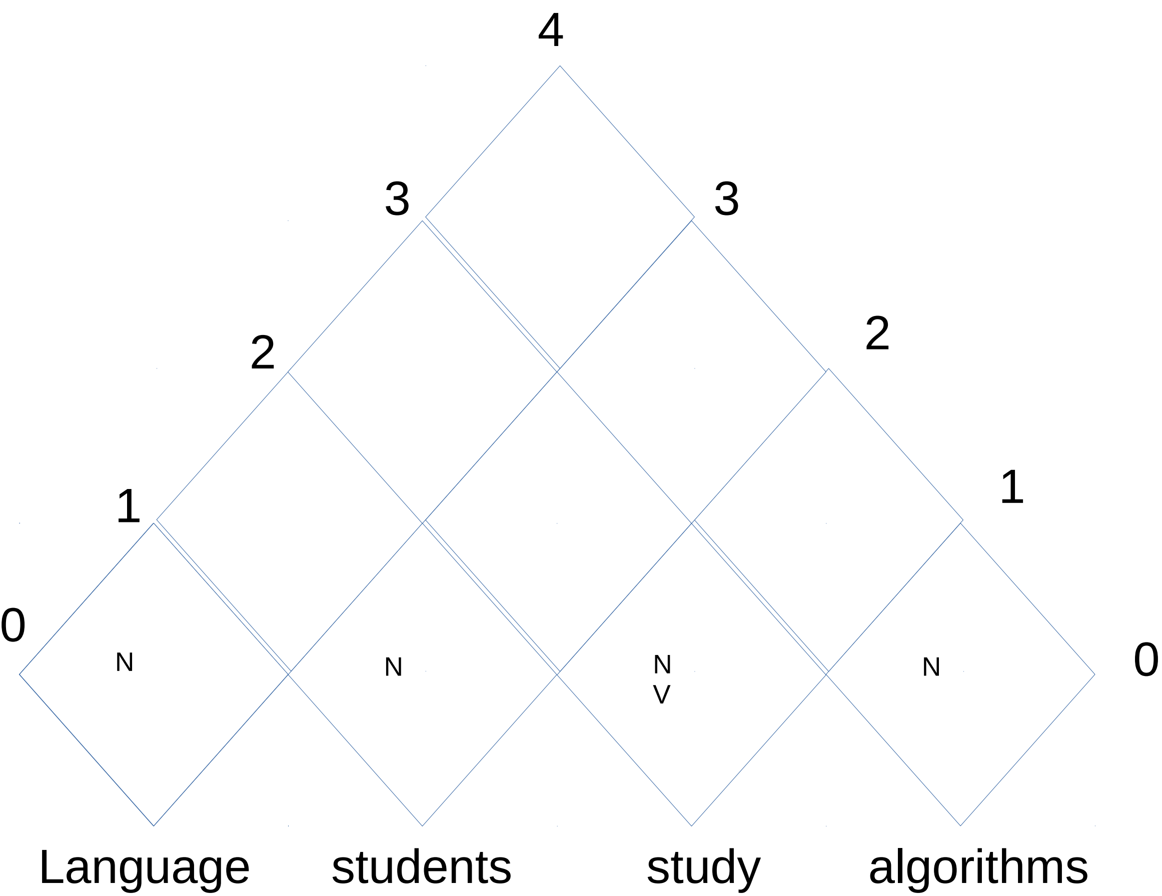
- Bottom-up dynamic programming by building a chart/triangle.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

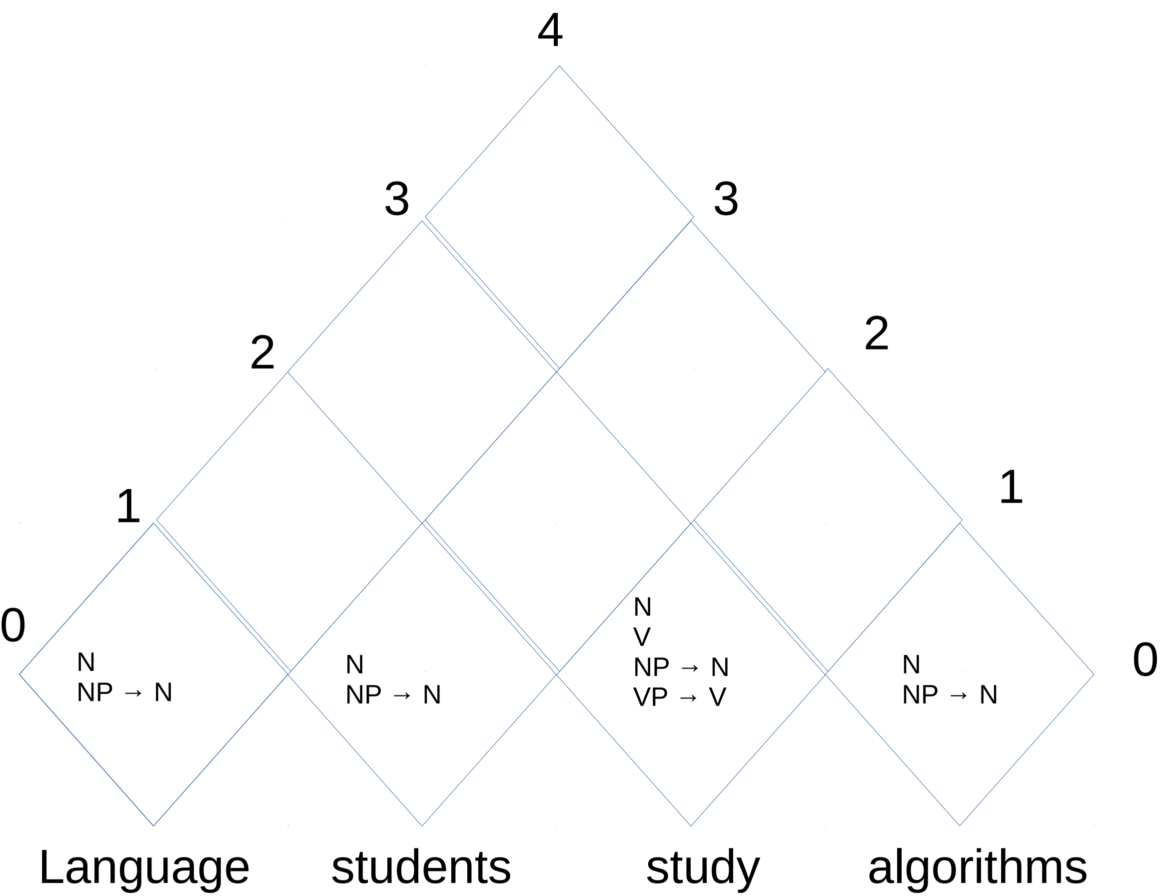
- First, lexicon rules.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing CKY algorithm

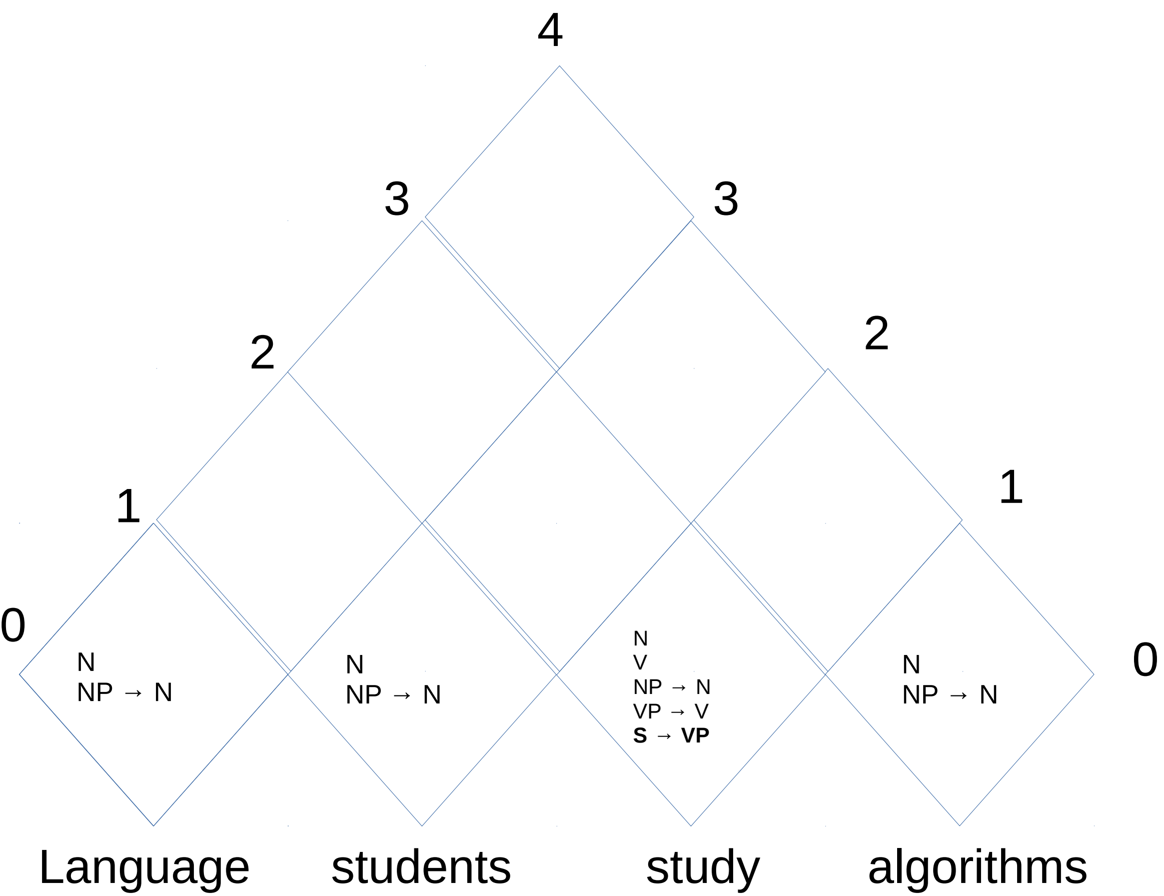
- Unary rules.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

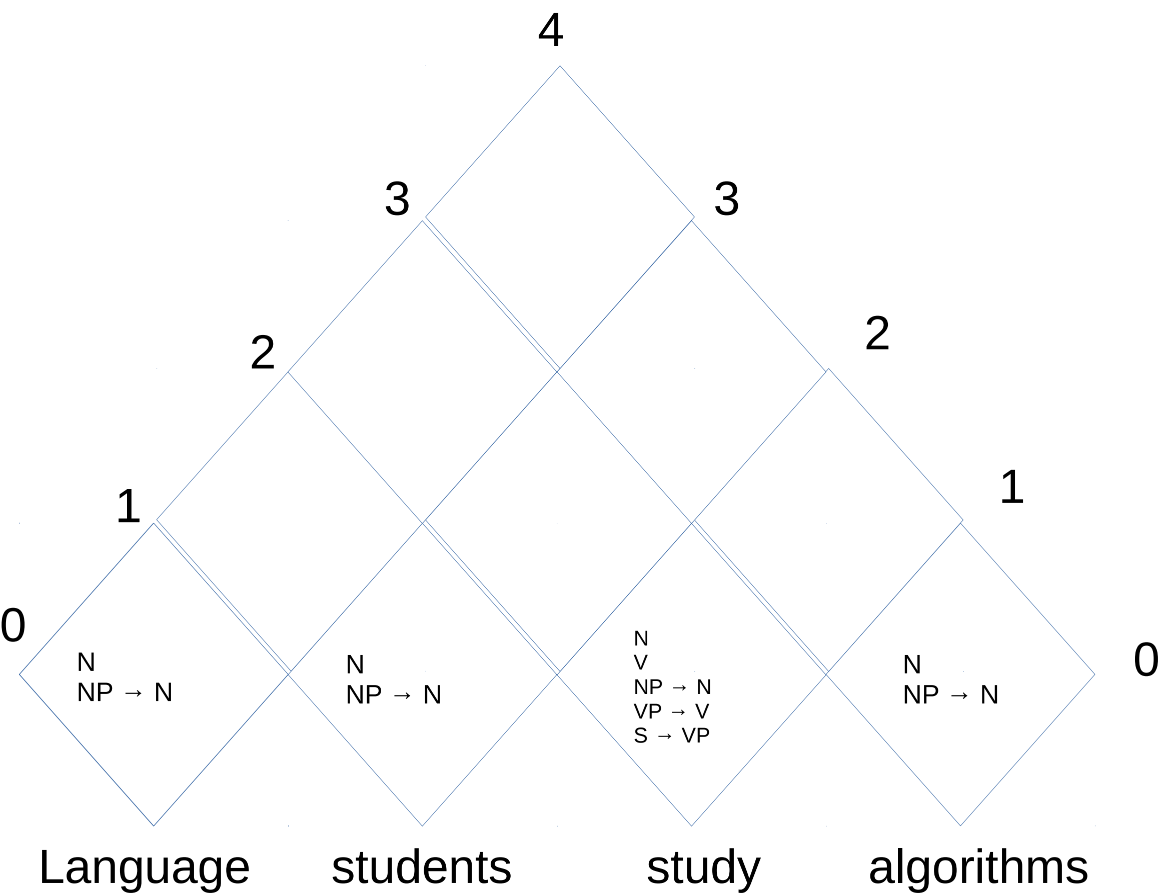
- More unary rules ?



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

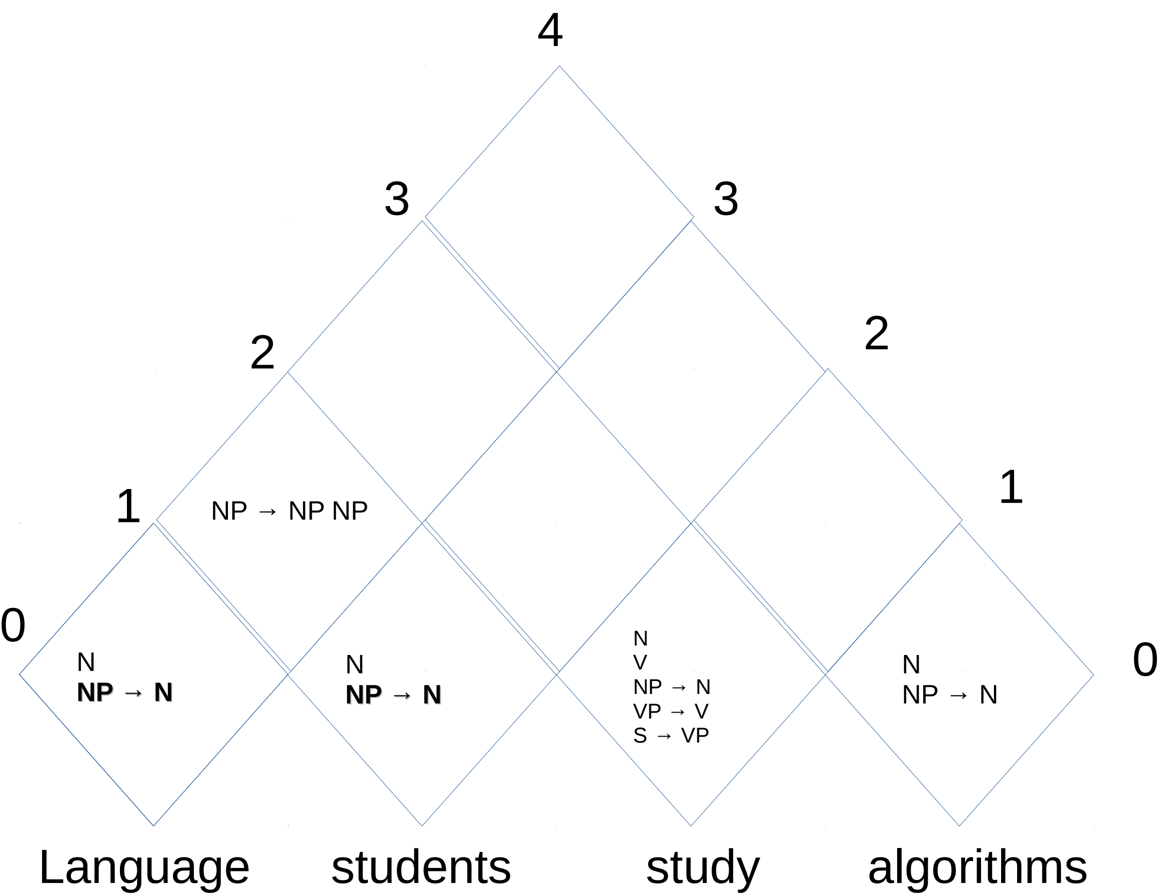
- Now, binary rules



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

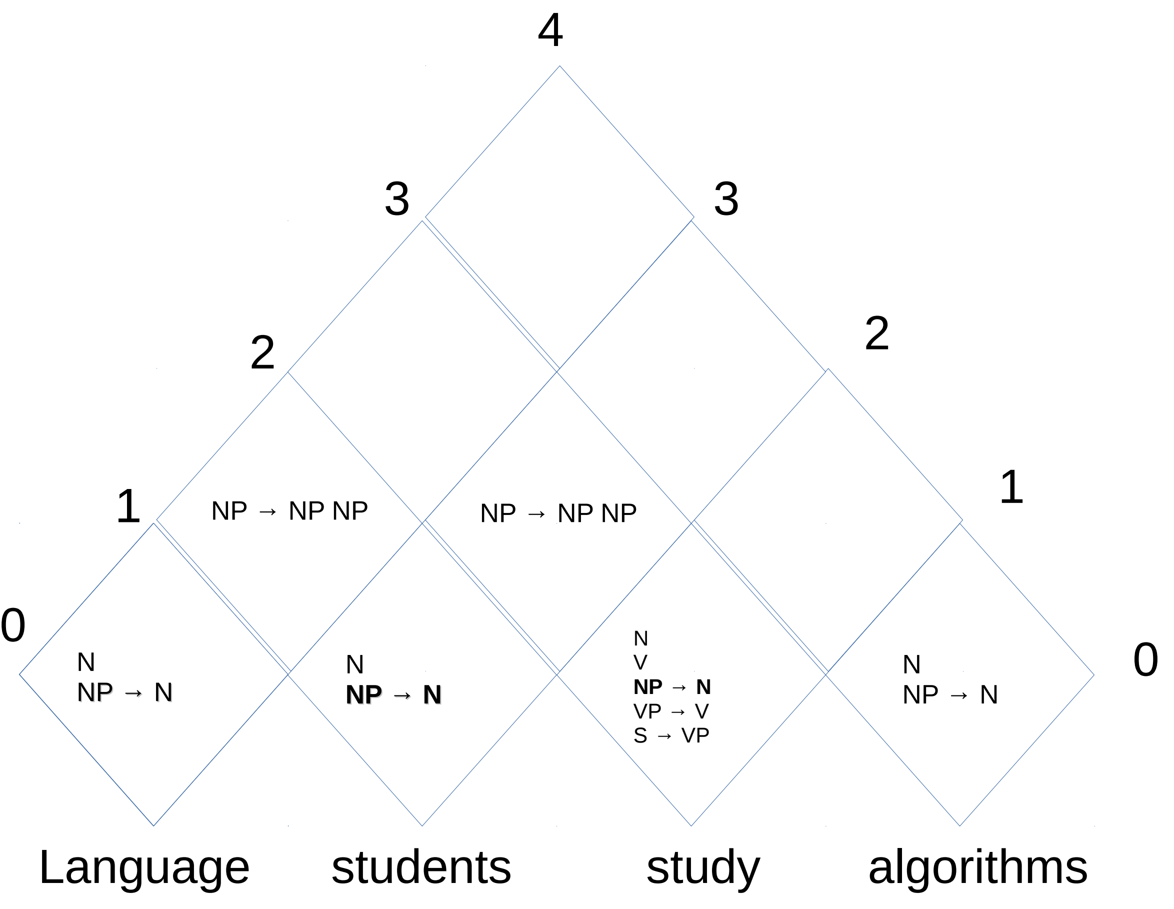
- Now, binary rules



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

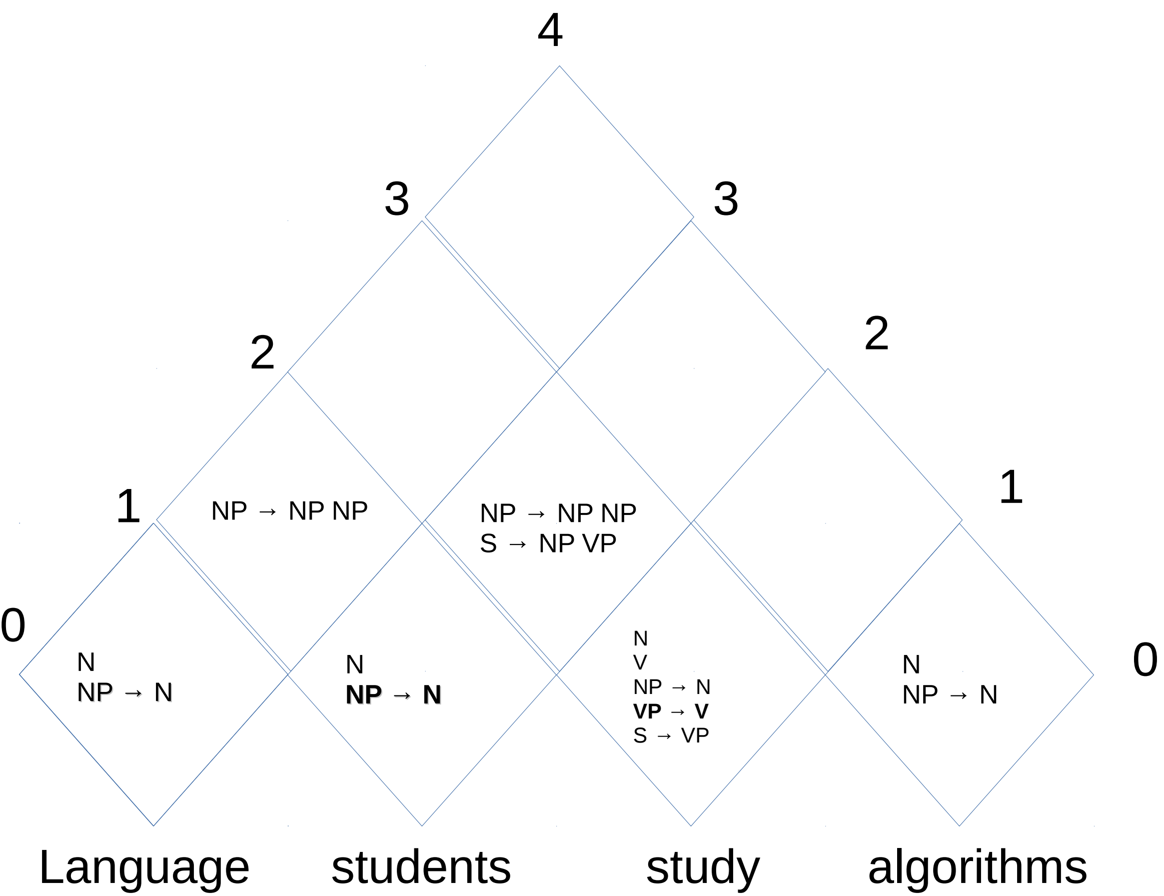
- Now, binary rules



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing CKY algorithm

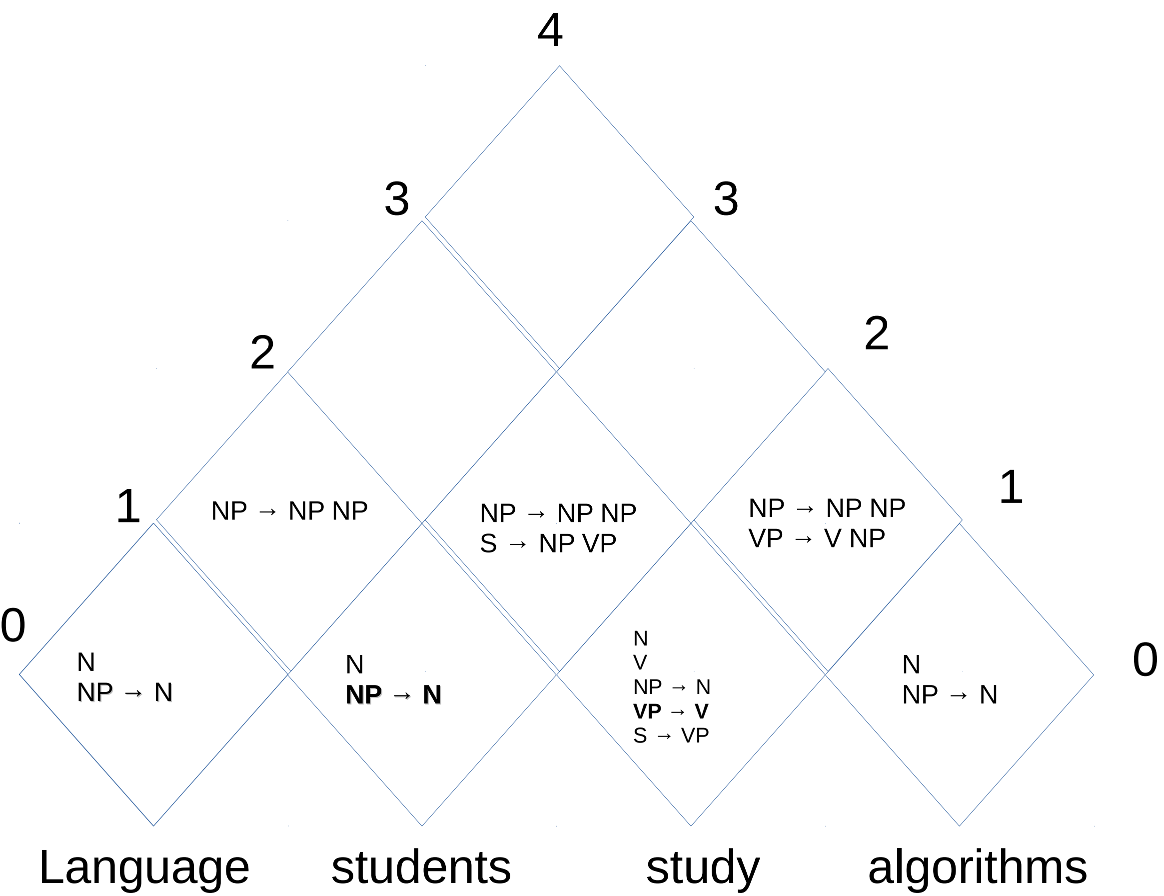
- Now, binary rules



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing CKY algorithm

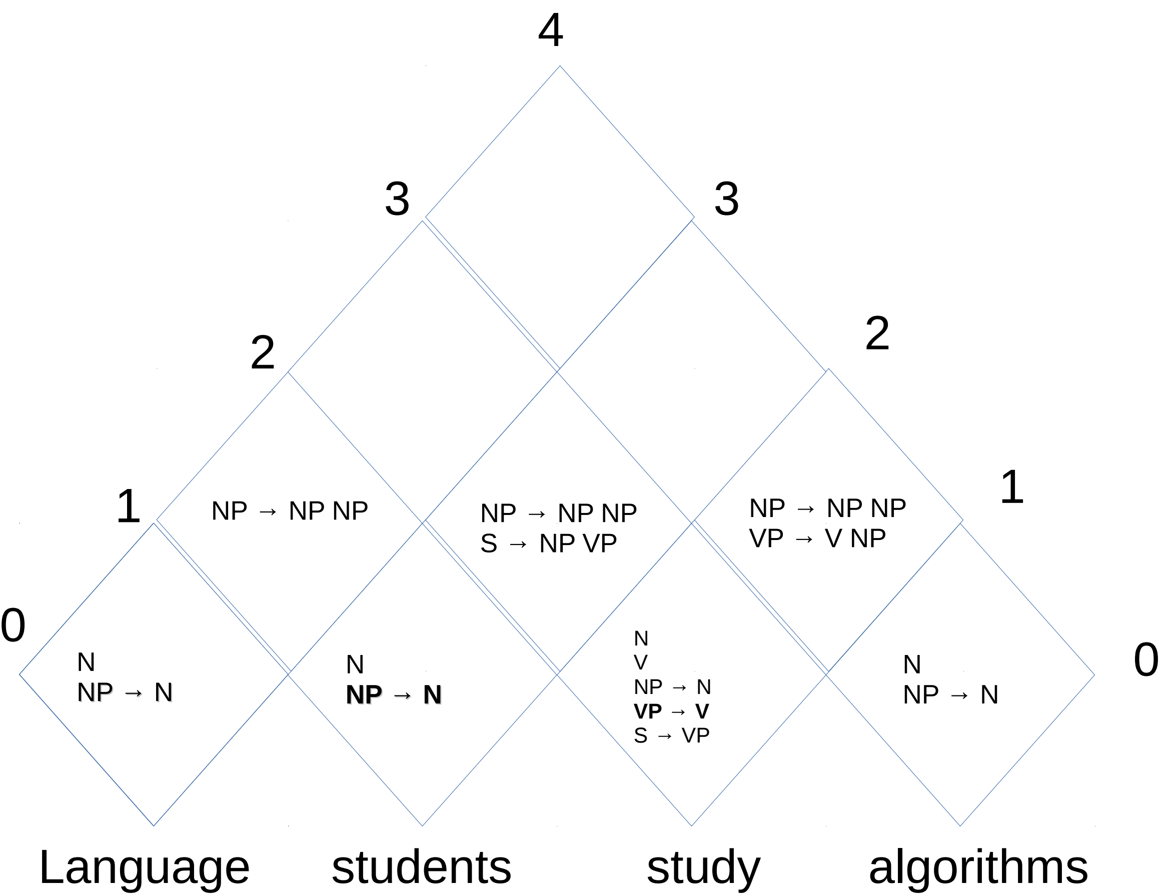
- Now, binary rules



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

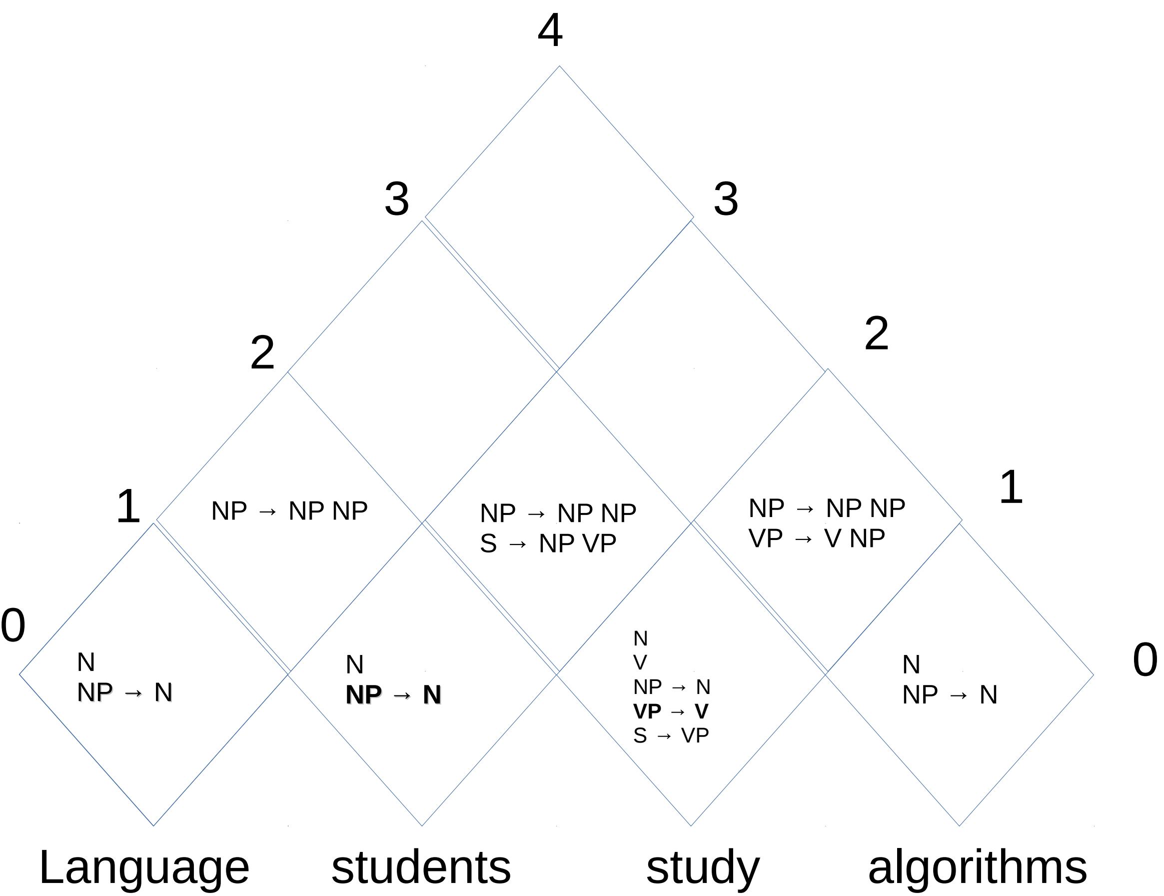
- Now, binary rules again.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

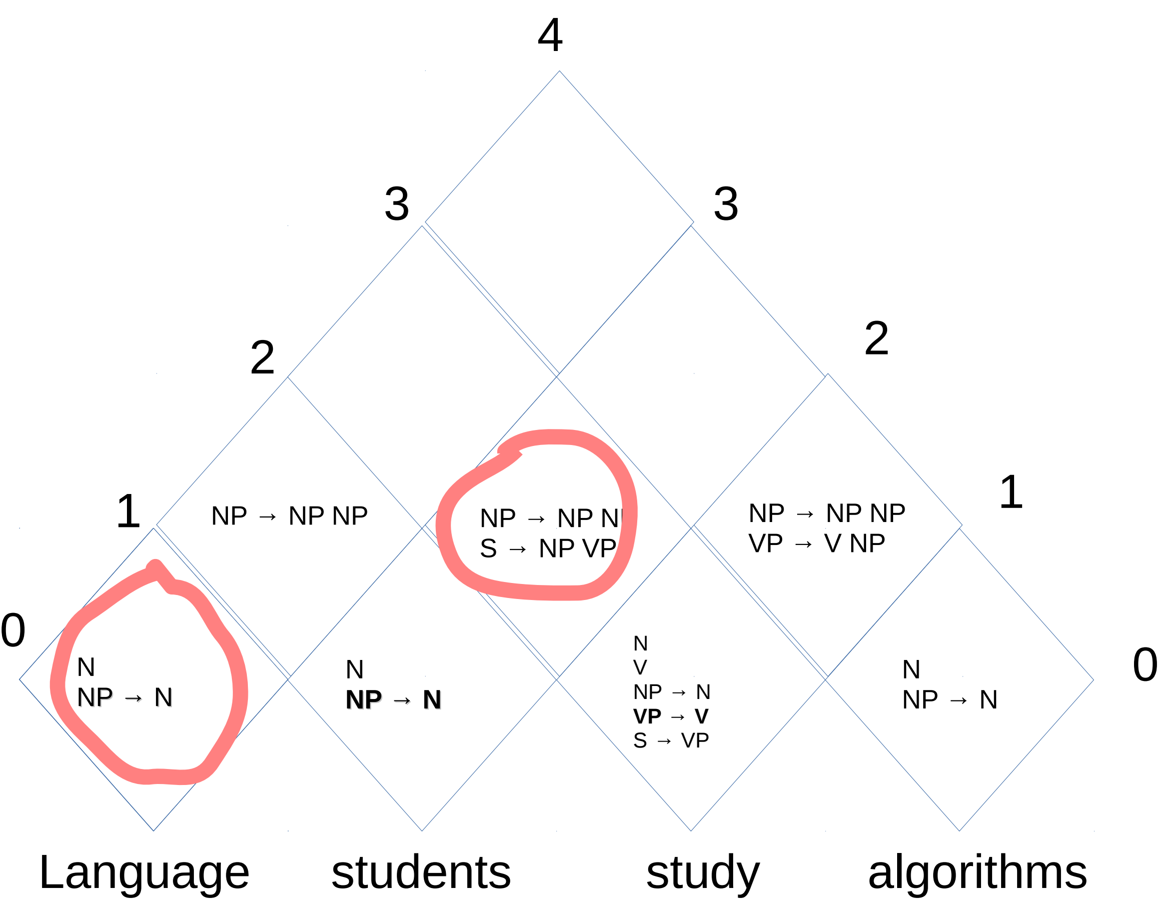
- Now, binary rules again.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

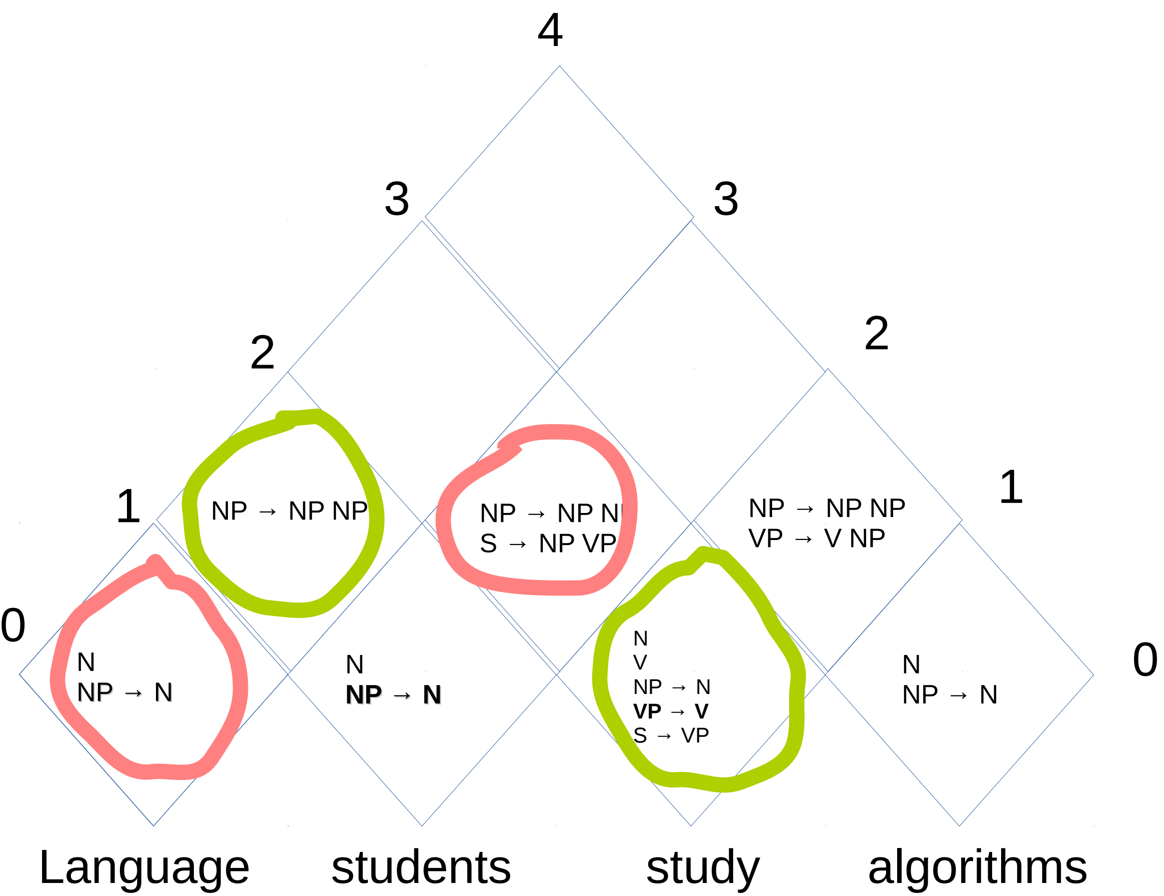
- Now, binary rules again.



- S → NP VP 0.7
- S → VP 0.3
- VP → V NP 0.5
- VP → V 0.3
- **NP → NP NP 0.3**
- NP → NP PP 0.4
- NP → N 0.3
- PP → P NP 1.0
- N → students 1.0
- N → study 0.4
- V → study 0.6
- N → Algorithms 1.0
- N → Language 1.0

Phrase structure parsing
CKY algorithm

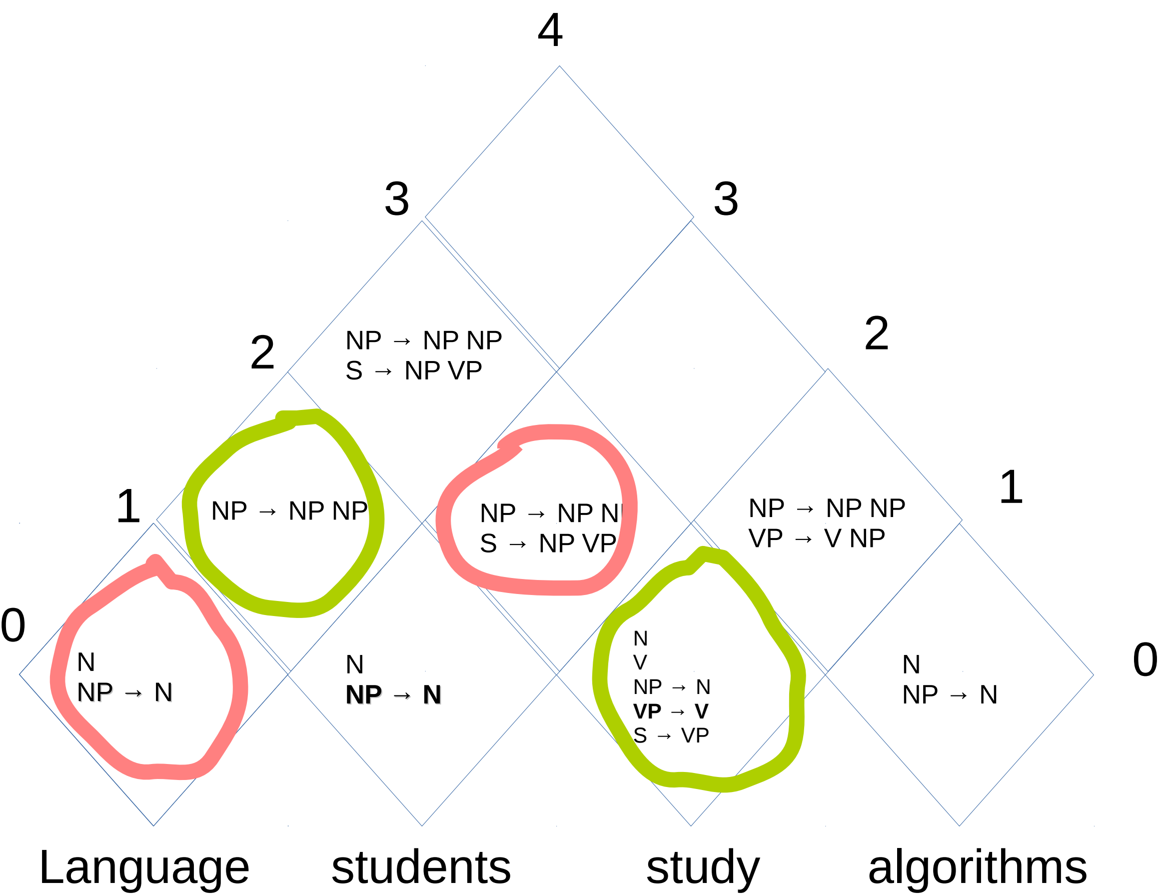
- Now, binary rules again.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

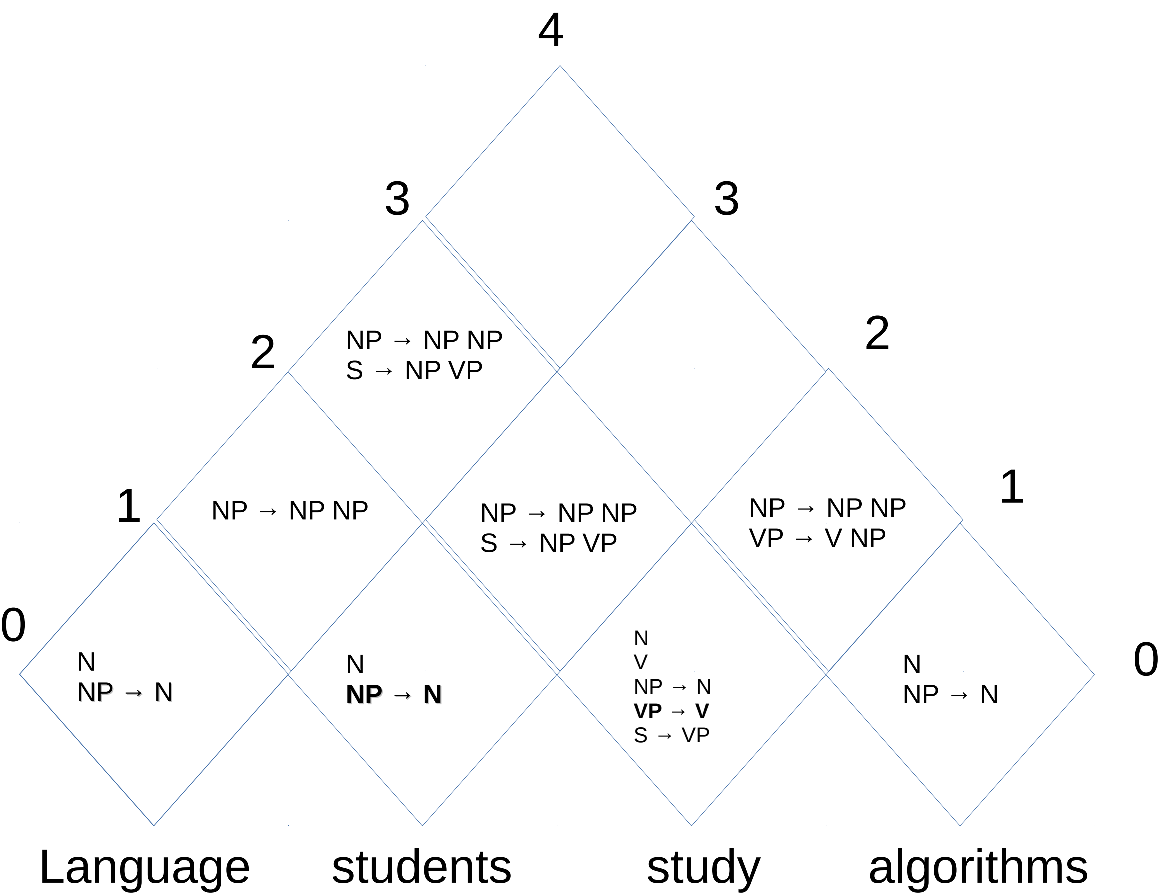
- Now, binary rules again.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

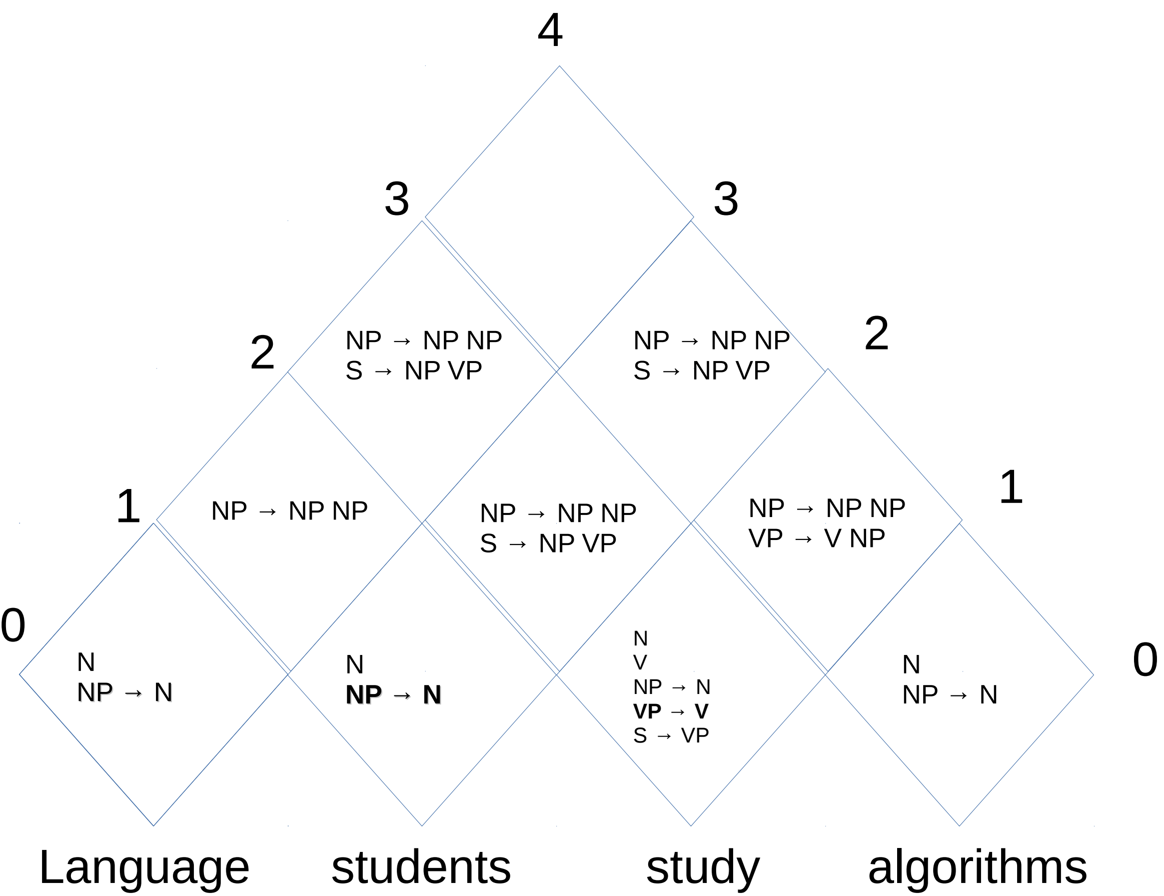
- Now, binary rules again.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

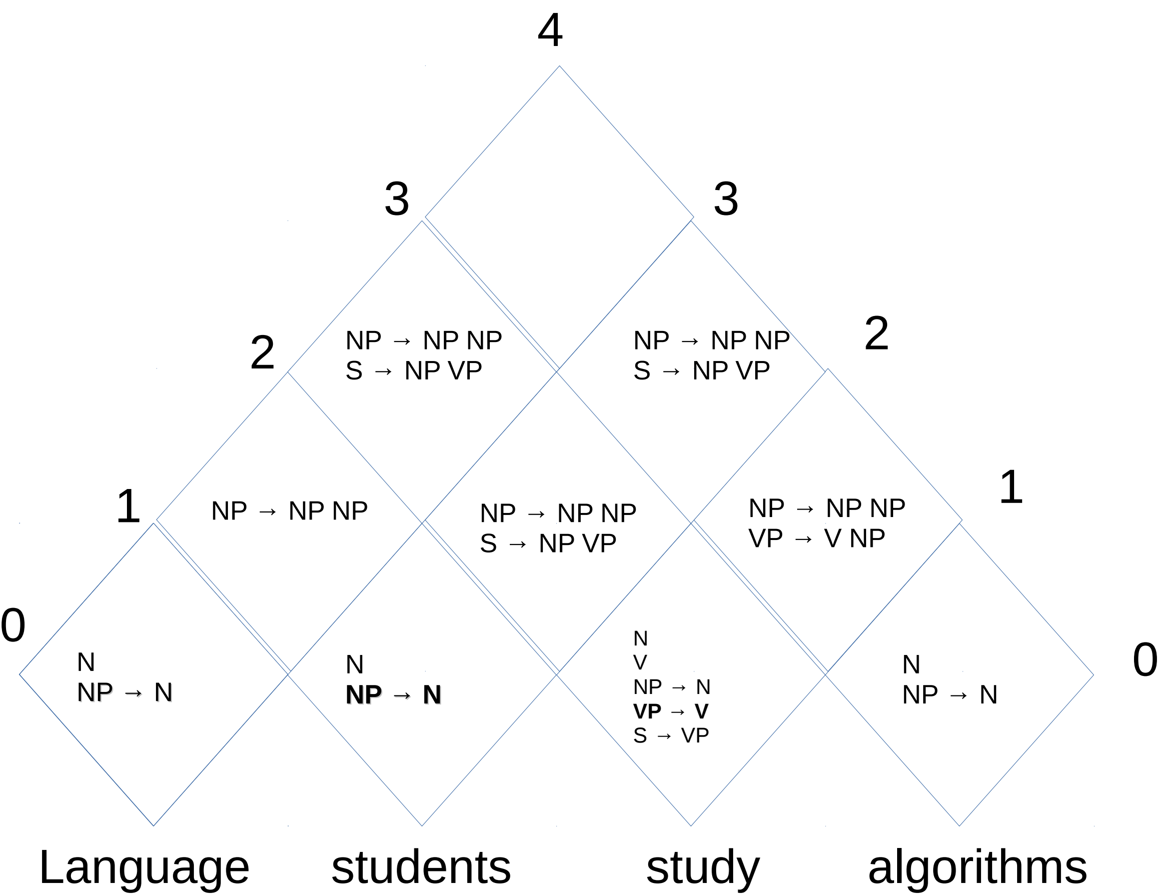
- Now, binary rules again.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

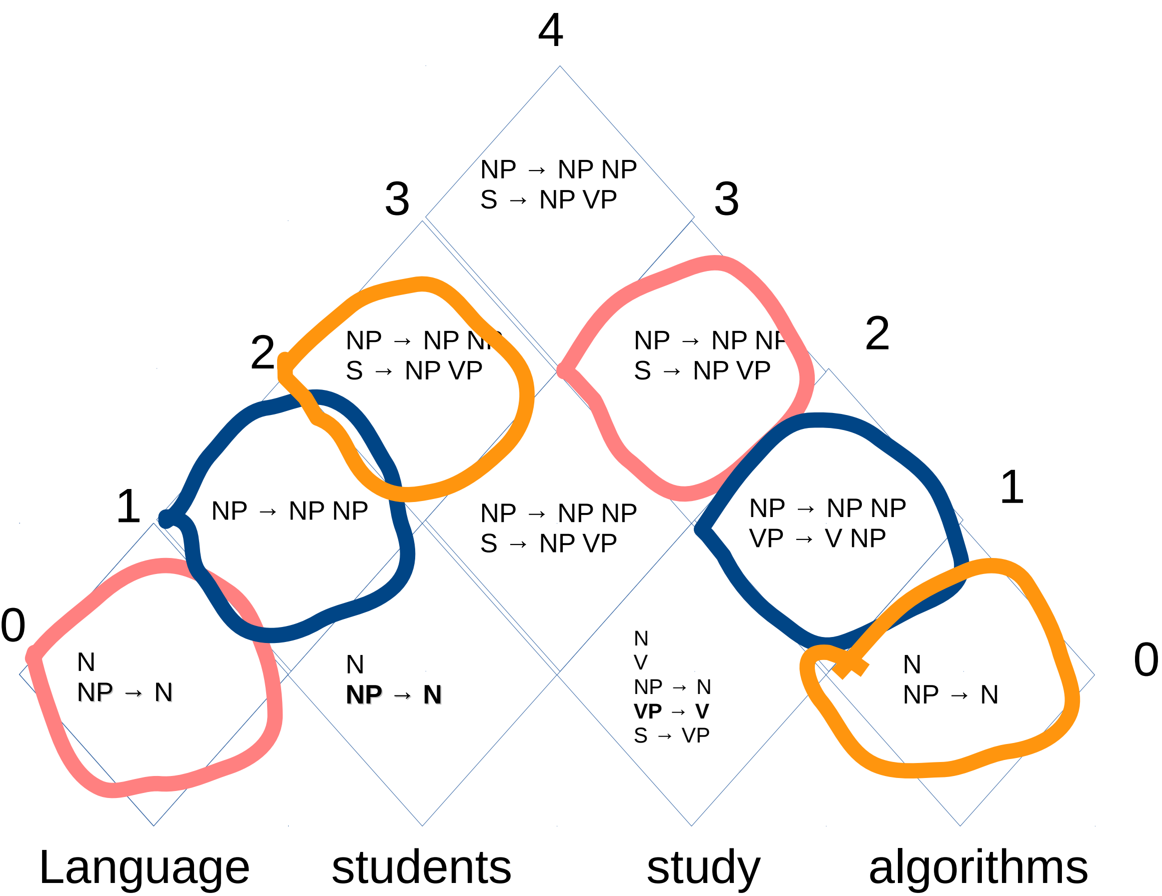
- And the last square...



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

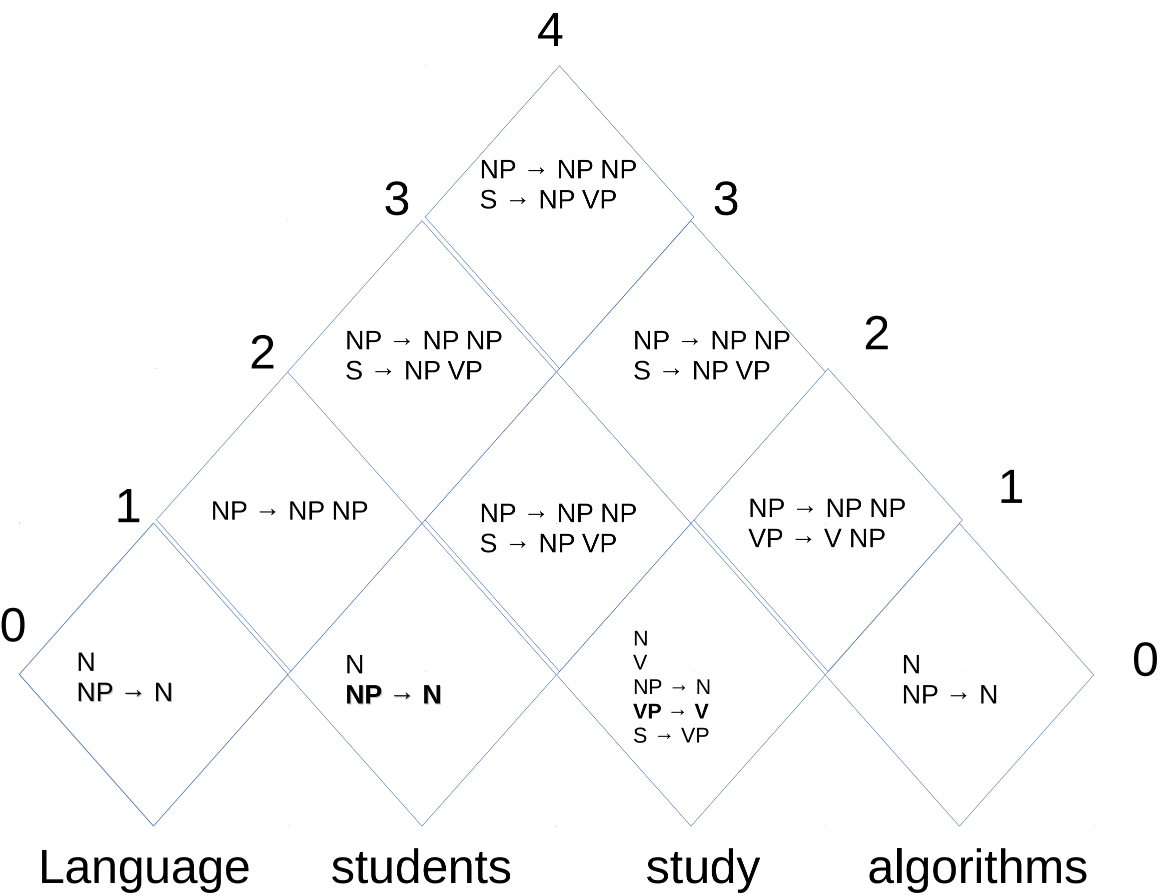
- And the last square...



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

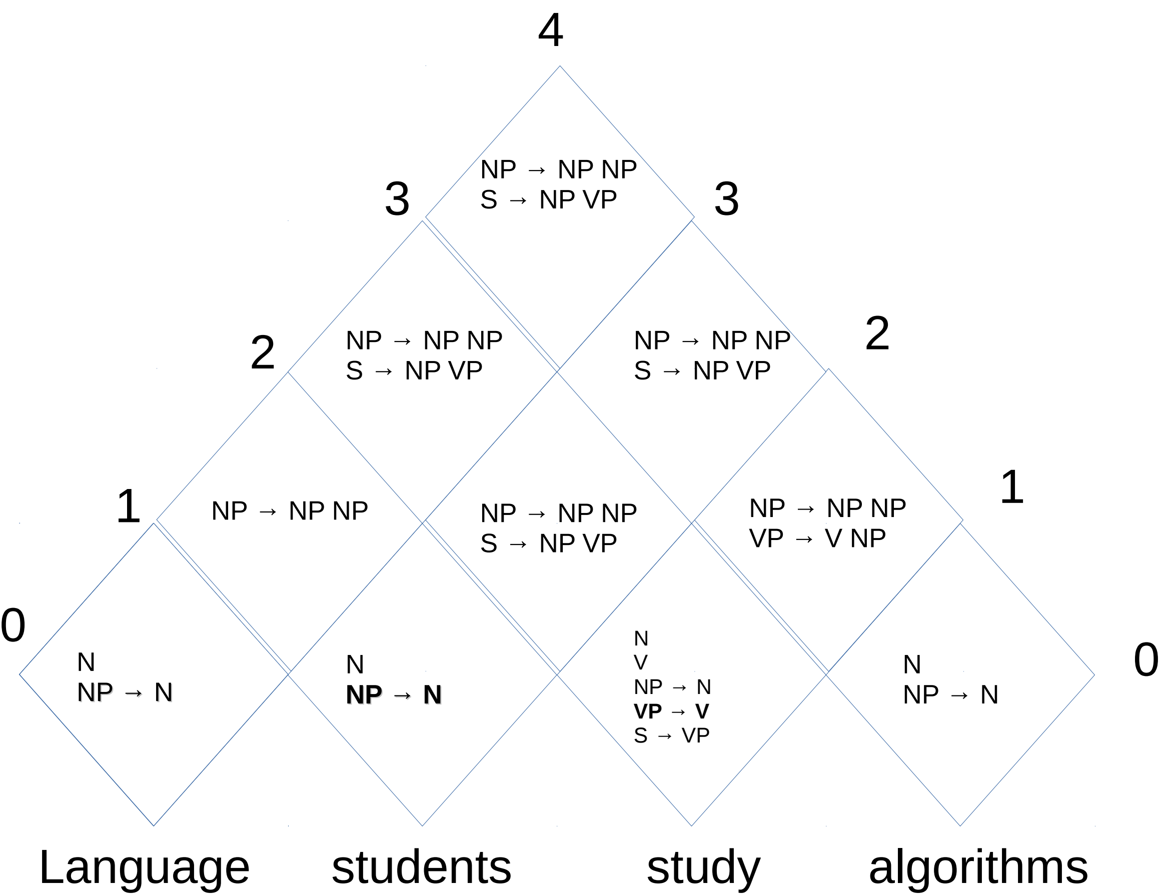
- Is S in the last square?



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

Phrase structure parsing
CKY algorithm

- Is S in the last square? YES! The sentence is recognized.



- S → NP VP
- S → VP
- VP → V NP
- VP → V
- NP → NP NP
- NP → NP PP
- NP → N
- PP → P NP
- N → students
- N → study
- V → study
- N → Algorithms
- N → Language

CKY for Parsing

- What about ambiguity? What we have provided is only a recognition algorithm,
  - it just says whether or not a sentence can be generated by a grammar, but it does not say what the parses are.
- In order to recover parses, we must encode additional information in the cells: **backpointers**
  - They point back to the two nonterminals in a split that prompt the entry of some new nonterminal

CKY for Parsing

- Whenever CKY adds a new nonterminal to **table[i, j]**, the algorithm must place in this table a pointer to the table entries from which it was derived, **table[i, k]** and **table[k, j]**.
- In order to accommodate ambiguity, it can be the case that the same nonterminal in a particular cell would have to be listed more than once.
  - If we have a sentence that has multiple parses, then the final entry for the sentence nonterminal in table[0, n] must contain more than one instance for S.
  - Each of these entries must have different backpointers from each other, because they tell us that the S was put together in different ways.

CKY for Parsing

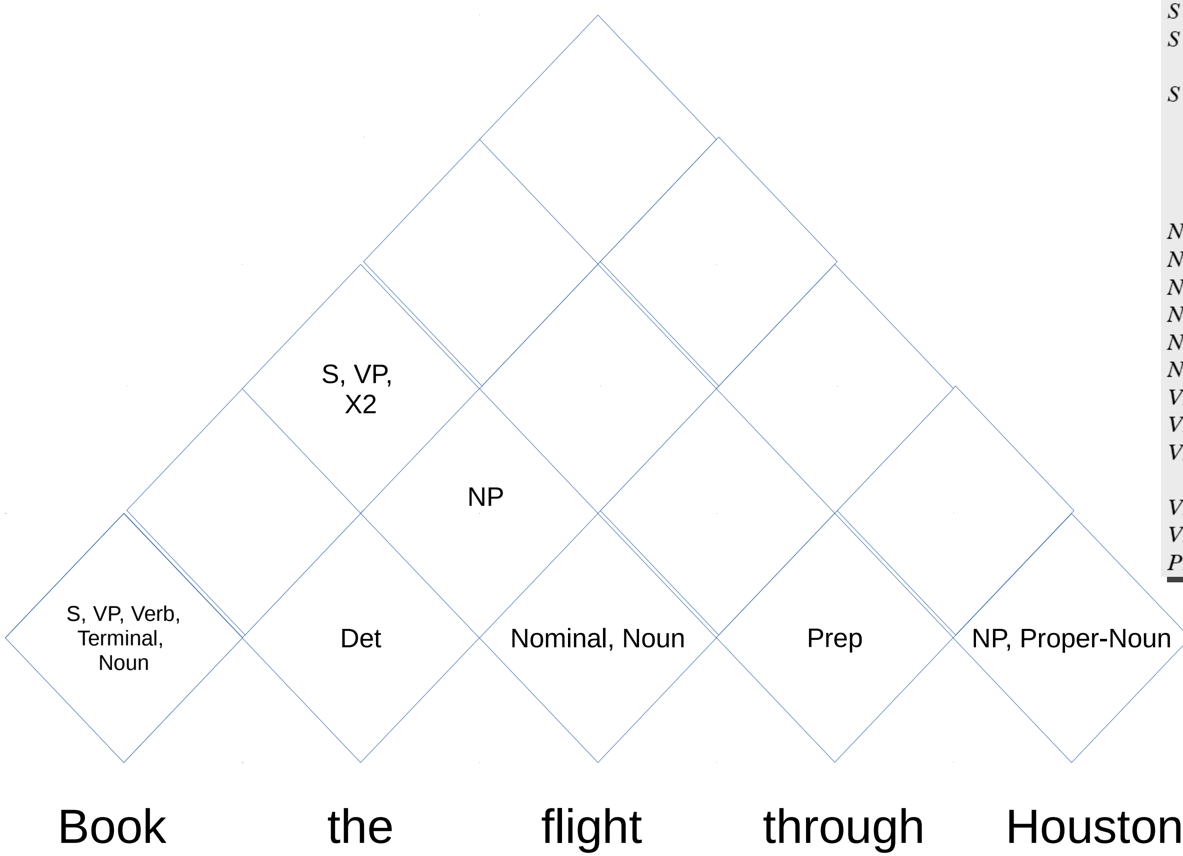
- The running time of the algorithm changes:
  - we have to make a difference of what would otherwise be the same nonterminal name in a particular cell.

CKY for Parsing

- To recover a particular parse, one could then trace through any sequence of backpointers, starting with all S entries in table[0, n].

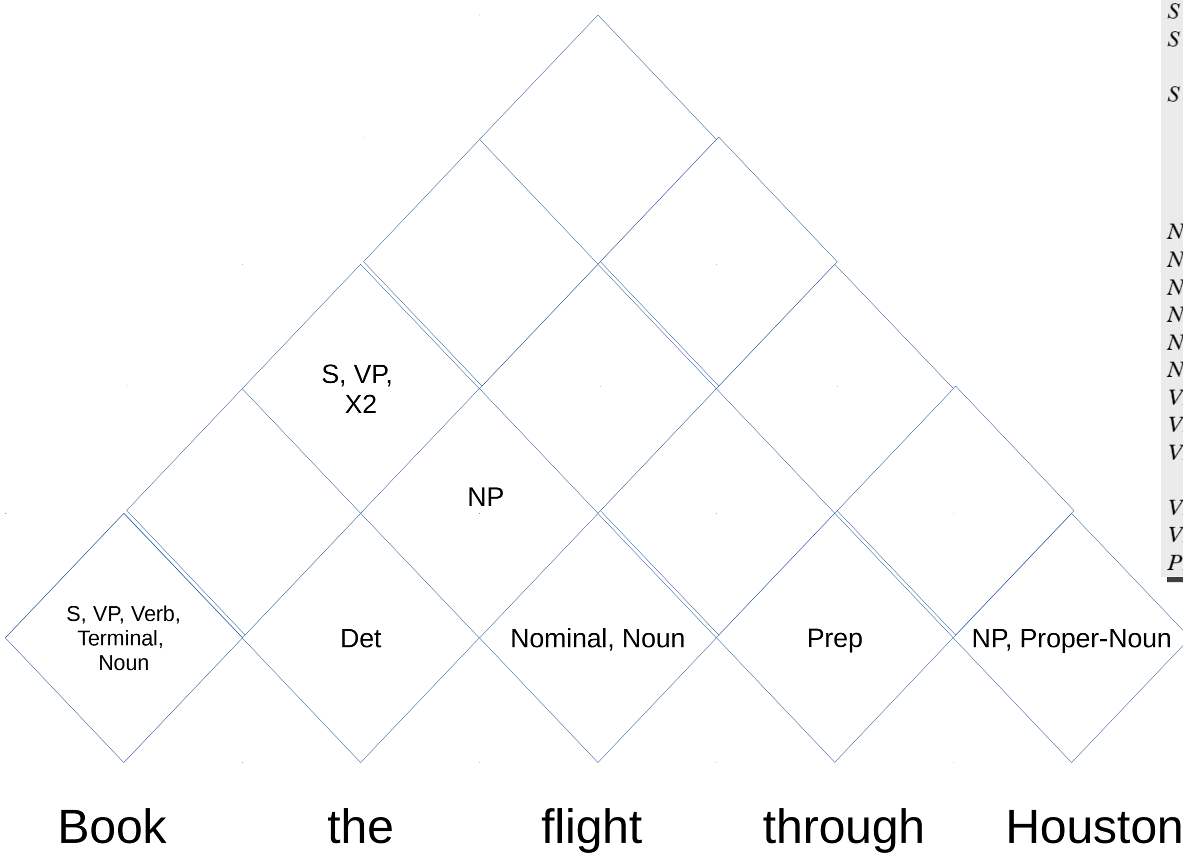
Note that if one had to recover all possible parses, then this would entail an exponential amount of time again, since we have shown that there can be an exponential number of parses

CKY for Parsing



$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

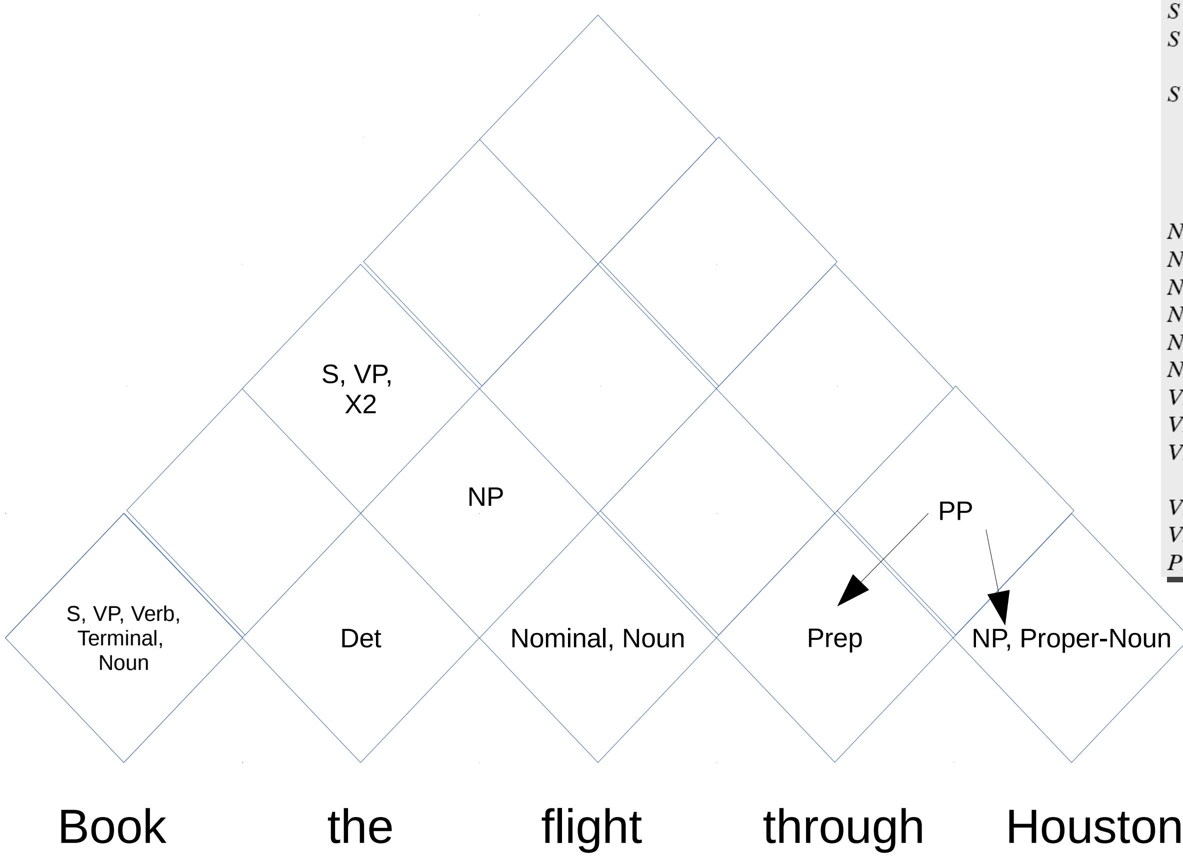
CKY for Parsing



$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

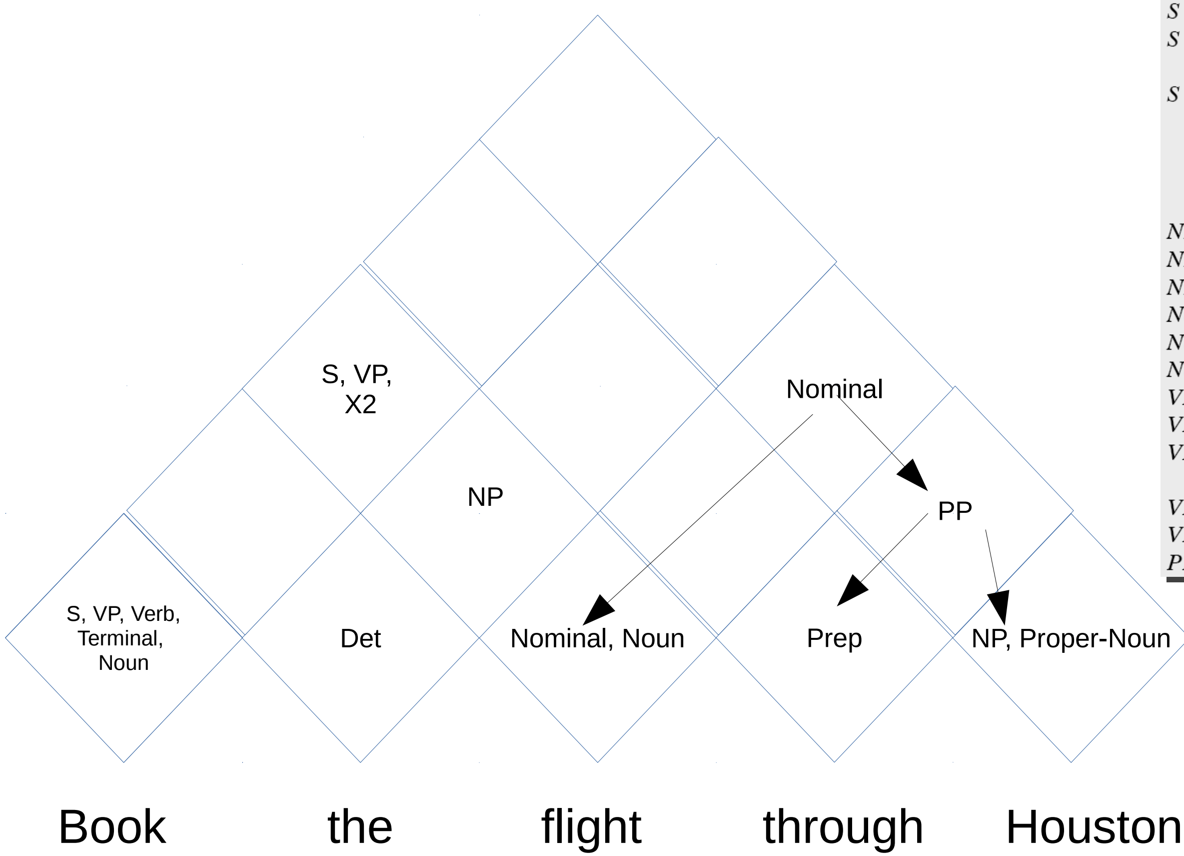
Let's fill Houston column, and see how the backpointers look like

CKY for Parsing



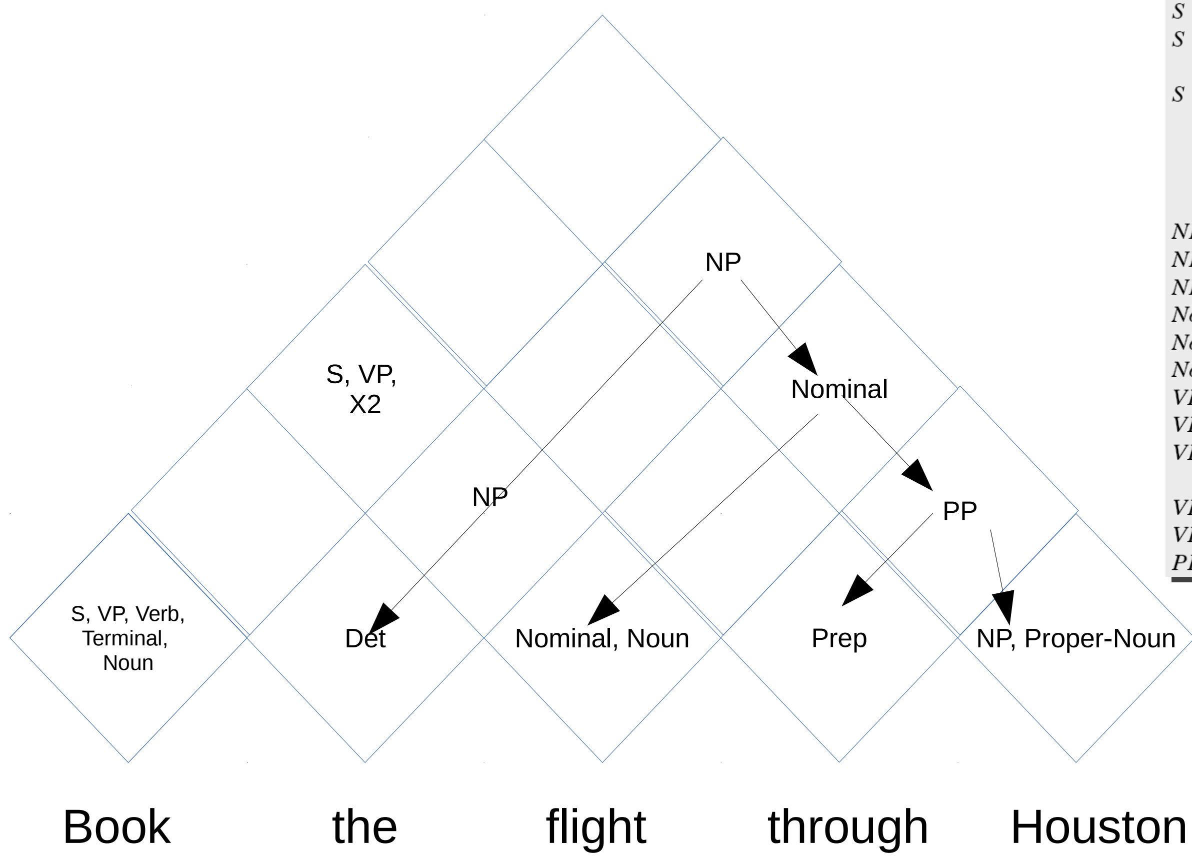
$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

CKY for Parsing



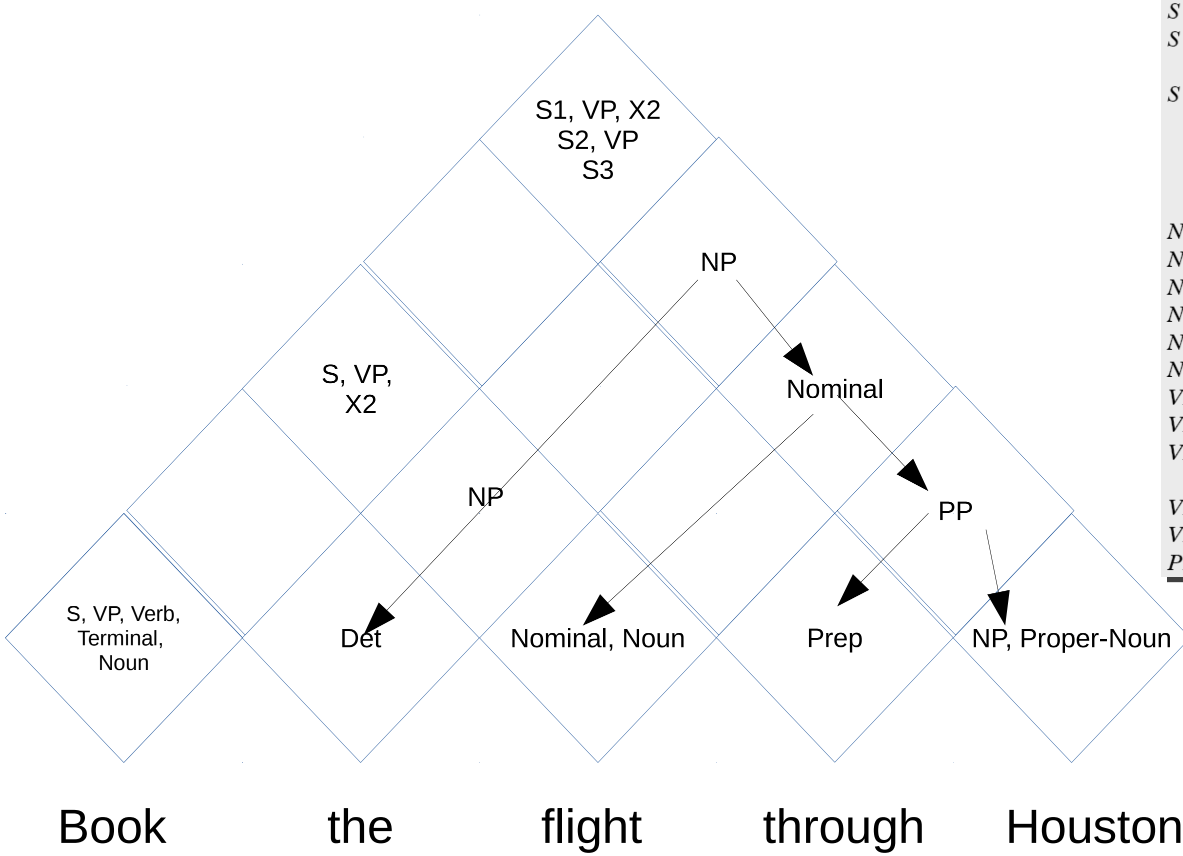
$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

CKY for Parsing



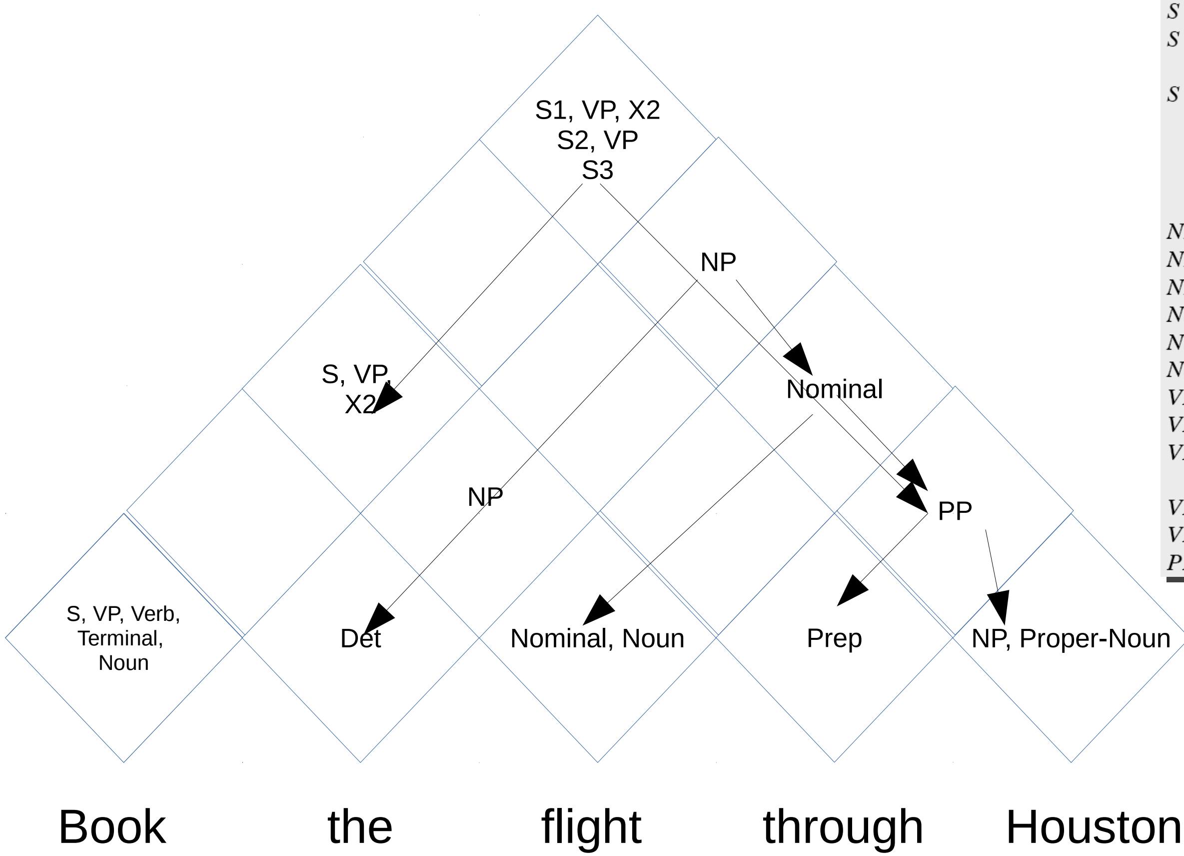
$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

CKY for Parsing



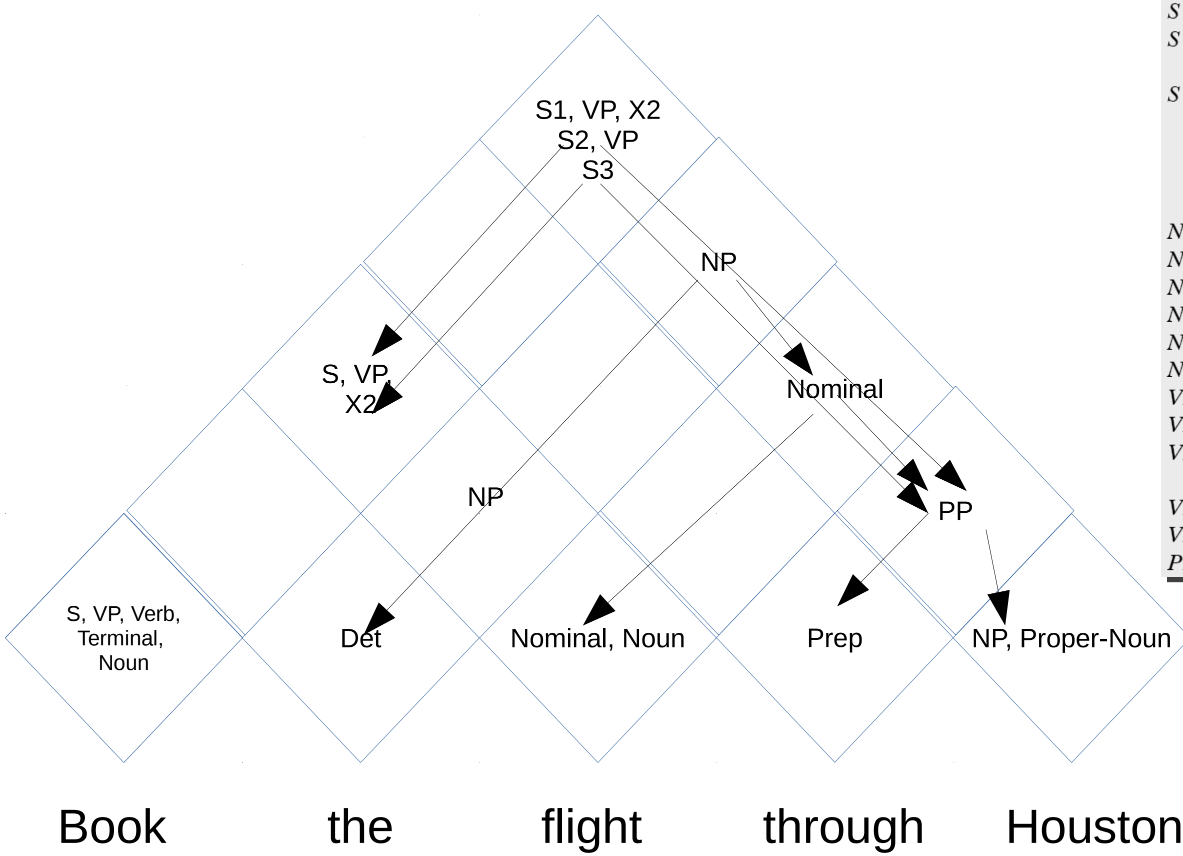
$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

CKY for Parsing



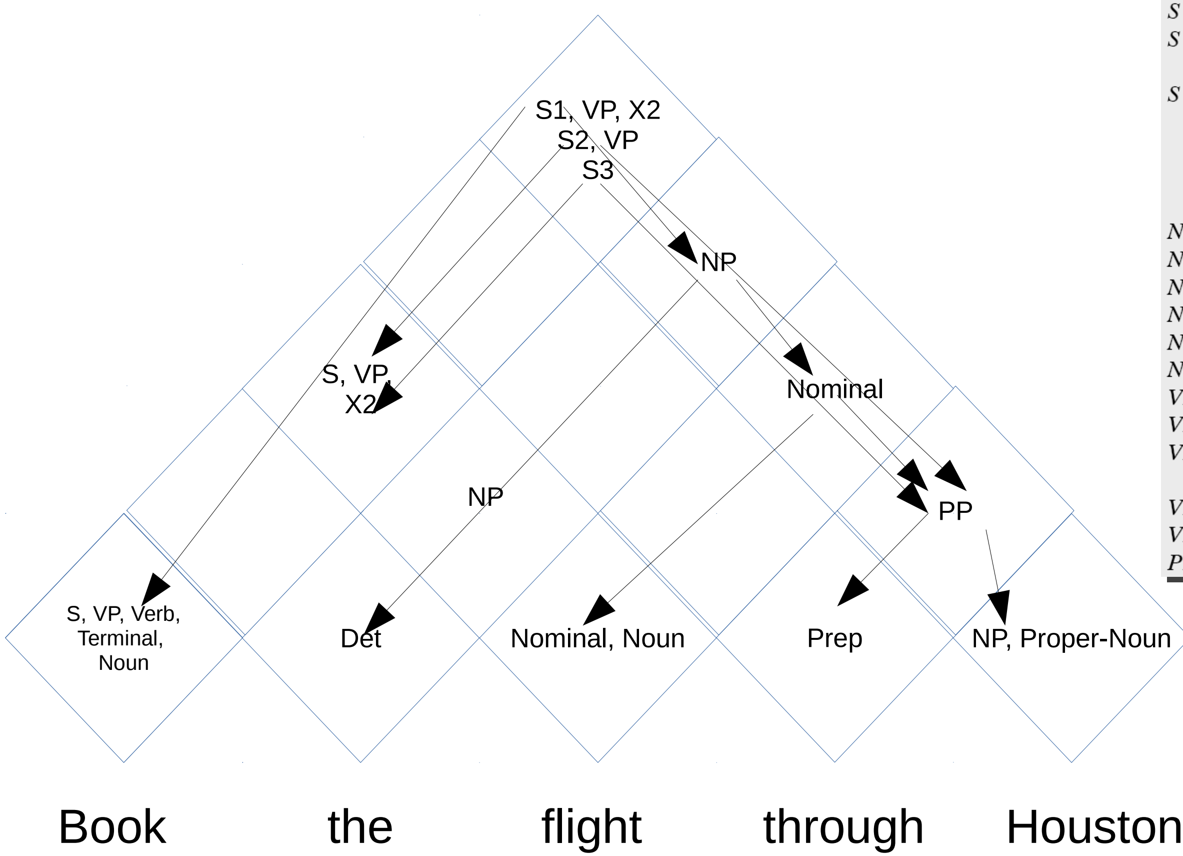
$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

CKY for Parsing



$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

CKY for Parsing



$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

Recognition vs Parsing

- Returning the full parse requires storing more in a cell than just a node label.
- We also require back-pointers to constituents of that node.
- We could also store whole trees, but less space efficient.
- For parsing, we must add an extra step to the algorithm: **follow pointers and return the parse.**

Ambiguity

- Efficient Representation of Ambiguities
- Local Ambiguity Packing :
  - a Local Ambiguity - multiple ways to derive the same substring from a non-terminal
  - All possible ways to derive each non-terminal are stored together
  - When creating back-pointers, create a single back-pointer to the "packed" representation
- Allows to efficiently represent a very large number of ambiguities (even exponentially many)
- Unpacking - producing one or more of the packed parse trees by following the back-pointers.

CKY Complexity

- function CKY (word w, grammar P) returns table

  for i ← **from** 1 **to** LENGTH(w) **do**

  table[i-1, i] ← {A | A → wi ∈ P }

  for j ← **from** 2 **to** LENGTH(w) **do**

  for i ← **from** j-2 **down to** 0 **do**

  for k ← i + 1 **to** j – 1 **do**

  table[i,j] ← table[i,j] ∪ {A | A → BC ∈ P,
  B ∈ table[i,k], C ∈ table [k,j] }

- If the start symbol S ∈ table[0,n] then **w ∈ L(G)**

CKY Complexity

- Time complexity:
  - Three nested “for” loop each one of O(n) size.
  - Time complexity = **$O(n^3)$**
- Space complexity:
  - A table of size n*n
  - Space complexity – **$O(n^2)$**