NLP

Introduction to NLP

Earley Parser

Earley's Parser

- Background
  - Developed by Jay Earley in 1970
  - No need to convert the grammar to CNF
  - Left to right
- Complexity
  - Faster than $O(n^3)$ in many cases

Earley's Parser

- Looks for both full and partial constituents
- Example:
  - S → Aux . NP VP
- When reading word *k*, it has already identified all hypotheses that are consistent with words 1 to *k*–1
- Example:
  - If the parser matches NP in the example above
  - S → Aux NP . VP

Earley's Parser

- It uses a dynamic programming table, just like CKY
- Example entry in column 1
  - `[0:1] VP -> VP . PP`
  - Created when processing word 1
  - Corresponds to words 0 to 1 (these words correspond to the VP part of the RHS of the rule)
  - The dot separates the completed (known) part from the incomplete (and possibly unattainable) part

Earley's Parser

- Three types of entries
  - ‘scan’ – for words
  - ‘predict’ – for non-terminals
  - ‘complete’ – otherwise

04.04 Earley Parser

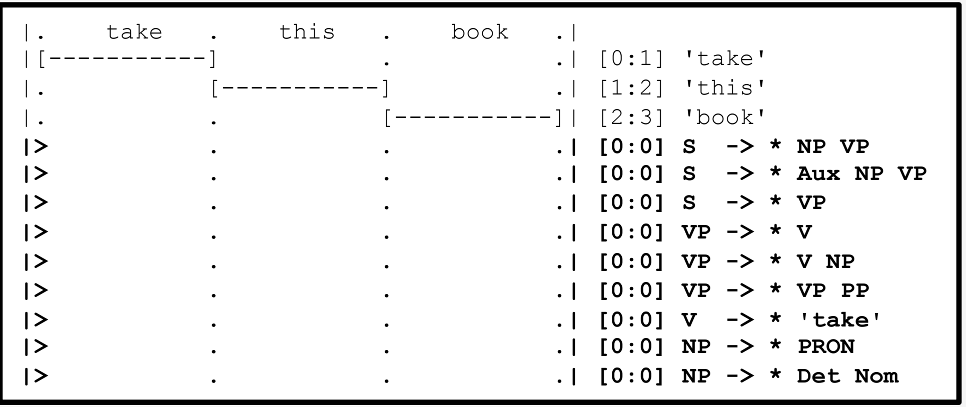
```
S -> NP VP
S -> Aux NP VP
S -> VP
NP -> PRON
NP -> Det Nom
Nom -> N
Nom -> Nom N
Nom -> Nom PP
PP -> PRP NP
VP -> V
VP -> V NP
VP -> VP PP
Det -> 'the'
Det -> 'a'
Det -> 'this'
PRON -> 'he'
PRON -> 'she'
N -> 'book'
N -> 'boys'
N -> 'girl'
PRP -> 'with'
PRP -> 'in'
V -> 'takes'
V -> 'take'
```

04.04 Earley Parser

```
|.    take    .    this    .    book    .|
|[-----------]            .            .| [0:1] 'take'
|.            [-----------]            .| [1:2] 'this'
|.            .            [-----------]| [2:3] 'book’
```

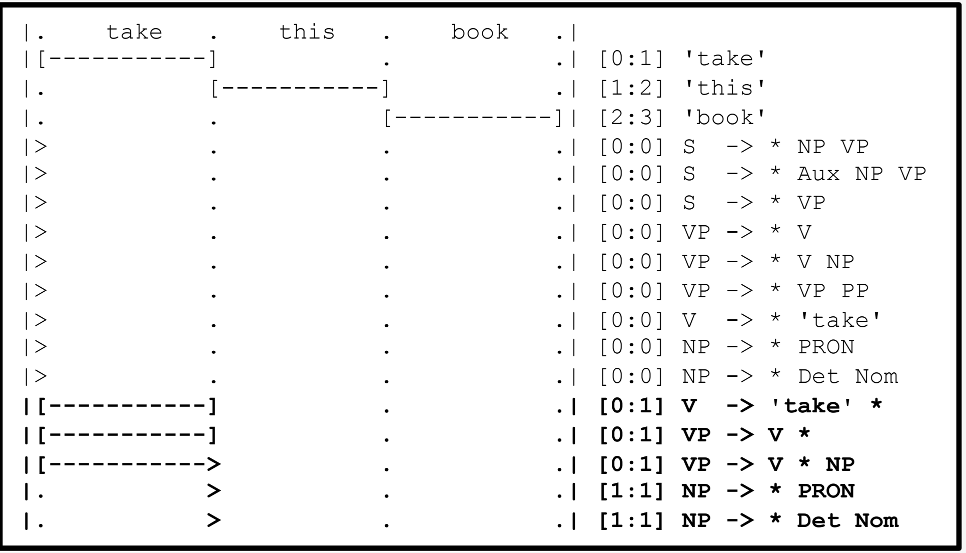
Example created using NLTK

04.04 Earley Parser



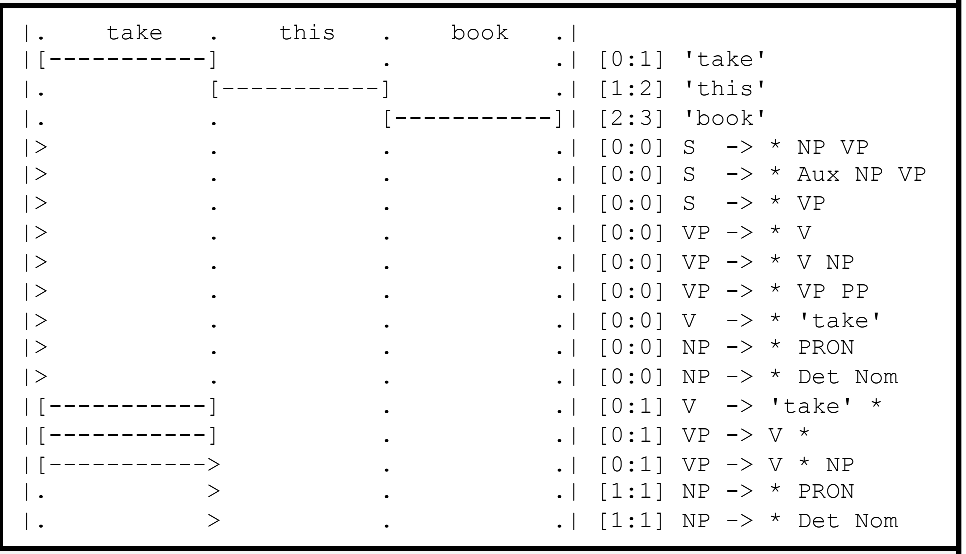
```
|.    take   .    this   .    book   .|
|[-----------]           .           .| [0:1] 'take'
|.           [-----------]           .| [1:2] 'this'
|.           .           [-----------]| [2:3] 'book'
|>           .           .           .| [0:0] S  -> * NP VP
|>           .           .           .| [0:0] S  -> * Aux NP VP
|>           .           .           .| [0:0] S  -> * VP
|>           .           .           .| [0:0] VP -> * V
|>           .           .           .| [0:0] VP -> * V NP
|>           .           .           .| [0:0] VP -> * VP PP
|>           .           .           .| [0:0] V  -> * 'take'
|>           .           .           .| [0:0] NP -> * PRON
|>           .           .           .| [0:0] NP -> * Det Nom
```

04.04 Earley Parser

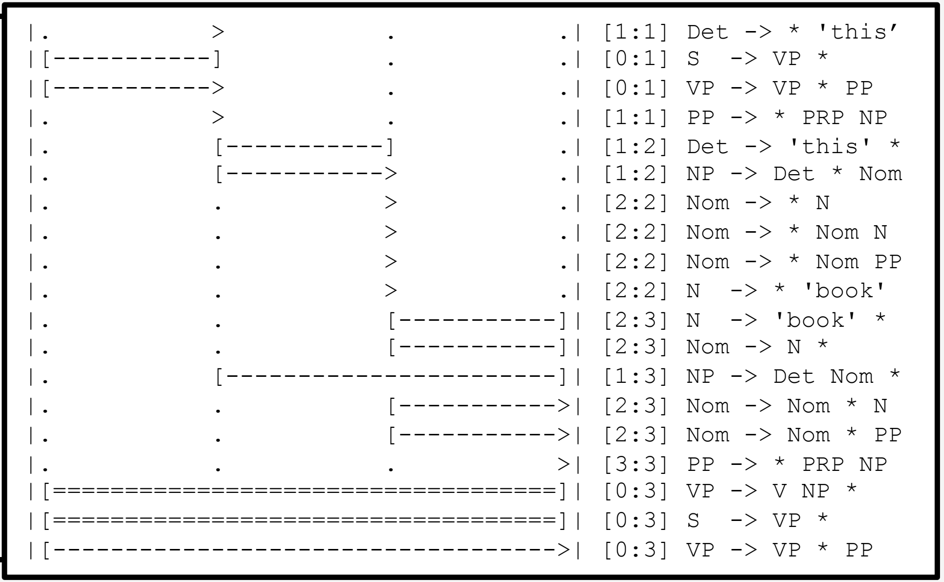


```
|.    take    .    this    .    book    .|
|[-----------]           .           .| [0:1] 'take'
|.           [-----------]           .| [1:2] 'this'
|.           .           [-----------]| [2:3] 'book'
|>           .           .           .| [0:0] S  -> * NP VP
|>           .           .           .| [0:0] S  -> * Aux NP VP
|>           .           .           .| [0:0] S  -> * VP
|>           .           .           .| [0:0] VP -> * V
|>           .           .           .| [0:0] VP -> * V NP
|>           .           .           .| [0:0] VP -> * VP PP
|>           .           .           .| [0:0] V  -> * 'take'
|>           .           .           .| [0:0] NP -> * PRON
|>           .           .           .| [0:0] NP -> * Det Nom
|[-----------]           .           .| [0:1] V  -> 'take' *
|[-----------]           .           .| [0:1] VP -> V *
|[----------->           .           .| [0:1] VP -> V * NP
|.           >           .           .| [1:1] NP -> * PRON
|.           >           .           .| [1:1] NP -> * Det Nom
```

04.04 Earley Parser



```
|.    take   .    this   .    book   .|
|[-----------]           .           .| [0:1] 'take'
|.           [-----------]           .| [1:2] 'this'
|.           .           [-----------]| [2:3] 'book'
|>           .           .           .| [0:0] S  -> * NP VP
|>           .           .           .| [0:0] S  -> * Aux NP VP
|>           .           .           .| [0:0] S  -> * VP
|>           .           .           .| [0:0] VP -> * V
|>           .           .           .| [0:0] VP -> * V NP
|>           .           .           .| [0:0] VP -> * VP PP
|>           .           .           .| [0:0] V  -> * 'take'
|>           .           .           .| [0:0] NP -> * PRON
|>           .           .           .| [0:0] NP -> * Det Nom
|[-----------]           .           .| [0:1] V  -> 'take' *
|[-----------]           .           .| [0:1] VP -> V *
|[----------->           .           .| [0:1] VP -> V * NP
|.           >           .           .| [1:1] NP -> * PRON
|.           >           .           .| [1:1] NP -> * Det Nom
```



```
|.           >           .           .| [1:1] Det -> * 'this’
|[-----------]           .           .| [0:1] S  -> VP *
|[----------->           .           .| [0:1] VP -> VP * PP
|.           >           .           .| [1:1] PP -> * PRP NP
|.           [-----------]           .| [1:2] Det -> 'this' *
|.           [----------->           .| [1:2] NP -> Det * Nom
|.           .           >           .| [2:2] Nom -> * N
|.           .           >           .| [2:2] Nom -> * Nom N
|.           .           >           .| [2:2] Nom -> * Nom PP
|.           .           >           .| [2:2] N  -> * 'book'
|.           .           [-----------]| [2:3] N  -> 'book' *
|.           .           [-----------]| [2:3] Nom -> N *
|.           [-----------------------]| [1:3] NP -> Det Nom *
|.           .           [----------->| [2:3] Nom -> Nom * N
|.           .           [----------->| [2:3] Nom -> Nom * PP
|.           .           .           >| [3:3] PP -> * PRP NP
|[===================================]| [0:3] VP -> V NP *
|[===================================]| [0:3] S  -> VP *
|[----------------------------------->| [0:3] VP -> VP * PP
```

(S (VP (V take) (NP (Det this) (Nom (N book)))))

NLP

Introduction to NLP

Issues with Context-free grammars

Agreement

- Number
  - Chen is/people are
- Person
  - I am/Chen is
- Tense
  - Chen was reading/Chen is reading/Chen will be reading
- Case
  - not in English but in many other languages such as German, Russian, Greek
- Gender
  - not in English but in many other languages such as German, French, Spanish

Combinatorial Explosion

- Many combinations of rules are needed to express agreement
  - S → NP VP
  - S → 1sgNP 1sgVP
  - S → 2sgNP 2sgVP
  - S → 3sgNP 3sgVP
  - …
  - 1sgNP → 1sgN
  - …

Subcategorization Frames

- Direct object
  - The dog ate a sausage
- Prepositional phrase
  - Mary left the car in the garage
- Predicative adjective
  - The receptionist looked worried
- Bare infinitive
  - She helped me buy this place
- To-infinitive
  - The girl wanted to be alone
- Participial phrase
  - He stayed crying after the movie ended
- That-clause
  - Ravi doesn’t believe that it will rain tomorrow
- Question-form clauses
  - She wondered where to go

CFG independence Assumptions

- Non-independence
  - All NPs
    - 11% NP PP, 9% DT NN, 6% PRP
  - NPs under S
    - 9% NP PP, 9% DT NN, 21% PRP
  - NPs under VP
    - 23% NP PP, 7% DT NN, 4% PRP
  - (example from Dan Klein)
- Lexicalized grammars
  - later

Conclusions

- Syntax helps understand the meaning of a sentence.
  - Bob gave Alice a flower
  - Who gave a flower to Alice?
  - What did Bob give to Alice?
- Context–free grammars are an appopriate representation for syntactic information
- Dynamic programming is needed for efficient parsing
  - Cubic time to find one parse
  - Still exponential time to find all parses
  - Why?

Answer

- Why does it still take an exponential time to find all parses?
  - Very simple – because the number of parses can be exponential

NLP