Top-down Chart Parsing: the Earley algorithm

Data Structures and Algorithms for Computational Linguistics III
(ISCL-BA-07)

Çağrı Çöltekin
ccoltekin@sfs.uni-tuebingen.de

University of Tübingen
Seminar für Sprachwissenschaft

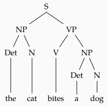
Winter Semester 2022/23

version: 367ab35 @2023-01-25

Parsing so far

- We can formulate parsing as
  - Top-down: begin with the start symbol, try to *produce* the input string to be parsed
  - Bottom up: begin with the input, and try to *reduce* it to the start symbol
- Another aspect of a parser is its directionality. Two choices are:
  - Directional: parses processes the input left to right (right to left is also possible, but rarely used)
  - Non-directional: order is not important, typically require all input to be in memory before processing

Top-down parsing as search



S	→	NP VP
NP	→	Det N
VP	→	V NP
VP	→	V
Det	→	a
Det	→	the
N	→	cat
N	→	dog
V	→	bites

Earley algorithm

- Earley algorithm is a top down (and left-to-right) parsing algorithm
- It allows arbitrary CFGs
- Keeps record of constituents that are
  - predicted: using the grammar (top-down)
  - in-progress: with partial evidence
  - completed: based on input seen so far

  at every position in the input string
- Time complexity is $O(n^3)$

Earley chart entries (states or items)

Earley chart entries are CF rules with a 'dot' on the RHS representing the state of the rule

- $A \rightarrow \bullet\alpha[i,i]$ predicted without any evidence (yet)
- $A \rightarrow \alpha \bullet \beta[i,j]$ partially matched
- $A \rightarrow \alpha\beta \bullet [i,j]$ completed, the non-terminal A is found in the given span

Earley algorithm: an informal sketch

1. Start at position 0, predict S
2. Predict all possible states (rules that apply)
3. Read a word
4. Update the table, advance the dot if possible
5. Go to step 2
6. If we have a completed S production at the end of the input, the input it recognized

Earley algorithm: three operations

- **Predictor** adds all rules that are possible at the given state
- **Completer** adds states from the earlier chart entries that match the completed state to the chart entry being processed, and advances their dot
- **Scanner** adds a completed state to the next chart entry if the current category is a pre-terminal symbol, and the terminal symbol (word) matches

Earley parsing example (chart[0])

0 she 1 saw 2 a 3 duck 4

state	rule	position	operation
0	γ → •S	[0,0]	initialization
1	S → •NP VP	[0,0]	predictor
2	S → •Aux NP VP	[0,0]	predictor
3	NP → •Det N	[0,0]	predictor
4	NP → •NP PP	[0,0]	predictor
5	NP → •Prn	[0,0]	predictor

Note: the chart[0] is independent of the input.

S	→	NP VP
S	→	Aux NP VP
NP	→	Det N
NP	→	Prn
NP	→	NP PP
VP	→	V NP
VP	→	V
VP	→	VP PP
PP	→	Prp NP
N	→	duck
N	→	park
V	→	duck
V	→	ducks
V	→	saw
Prn	→	she | her
Prp	→	in | with
Det	→	a | the
Aux	→	does | has

Earley parsing example (chart[1])

0 she 1 saw 2 a 3 duck 4

state	rule	position	operation
6	Prn → she •	[0,1]	scanner
7	NP → Prn •	[0,1]	completer
8	S → NP •VP	[0,1]	completer
9	NP → NP •PP	[0,1]	completer
10	VP → •V NP	[1,1]	predictor
11	VP → •V	[1,1]	predictor
12	VP → •VP PP	[1,1]	predictor
13	PP → •Prp NP	[1,1]	predictor

S → NP VP
S → Aux NP VP
NP → Det N
NP → Prn
NP → NP PP
VP → V NP
VP → V
VP → VP PP
PP → Prp NP
N → duck
N → park
V → duck
V → ducks
V → saw
Prn → she | her
Prp → in | with
Det → a | the
Aux → does | has

Earley parsing example (chart[2])

0 she 1 saw 2 a 3 duck 4

state	rule	position	operation
14	V → saw •	[1,2]	scanner
15	VP → V •NP	[1,2]	completer
16	VP → V •	[1,2]	completer
17	S → NP VP •	[0,2]	completer
18	NP → •Det N	[2,2]	predictor
19	NP → •NP PP	[2,2]	predictor
20	NP → •Prn	[2,2]	predictor

S → NP VP
S → Aux NP VP
NP → Det N
NP → Prn
NP → NP PP
VP → V NP
VP → V
VP → VP PP
PP → Prp NP
N → duck
N → park
V → duck
V → ducks
V → saw
Prn → she | her
Prp → in | with
Det → a | the
Aux → does | has

Earley parsing example (chart[3])

0 she 1 saw 2 a 3 duck 4

state	rule	position	operation
21	Det → a •	[2,3]	scanner
22	NP → Det •N	[2,3]	completer

S → NP VP
S → Aux NP VP
NP → Det N
NP → Prn
NP → NP PP
VP → V NP
VP → V
VP → VP PP
PP → Prp NP
N → duck
N → park
V → duck
V → ducks
V → saw
Prn → she | her
Prp → in | with
Det → a | the
Aux → does | has

Earley parsing example (chart[4])

0 she 1 saw 2 a 3 duck 4

state	rule	position	operation
23	N → duck •	[3,4]	scanner
24	V → duck •	[3,4]	scanner
25	NP → Det N •	[2,4]	completer
26	VP → V NP •	[1,4]	completer
27	S → NP VP •	[0,4]	completer

S → NP VP
S → Aux NP VP
NP → Det N
NP → Prn
NP → NP PP
VP → V NP
VP → V
VP → VP PP
PP → Prp NP
N → duck
N → park
V → duck
V → ducks
V → saw
Prn → she | her
Prp → in | with
Det → a | the
Aux → does | has

Earley parsing: summary

- Complexity (asymptotic) is the same as CKY
  - time complexity: $O(n^3)$
  - space complexity: $O(n^2)$
- Our example shows recognition, we need to maintain back links for parsing
- Again, the Earley chart stores a parse forest compactly, but extracting all trees may require exponential time

Summary

- The Earley parser is a top-down parser with bottom-up filtering (or, you can also view it the other way around)
- The parser improves over a backtracking parser by
  - dynamic programming: not re-computing the subtrees
  - filtering: not generating hypotheses (predictor) that cannot match at a given input position
- It can process any CFG (no need for CNF)
- There is a nice relation between CKY and Earley: you can view Earley as binarizing the grammar (converting to CNF) 'on the fly'

Next:

- Dependency parsing
- Reading suggestion: **jurafsky2009**

An exercise

Construct the CKY and Earley charts for the sentence below

`The duck she saw is in the park`

Recommended grammar:

S	→ NP VP	PP	→ Prp NP
NP	→ Det N	N	→ park
NP	→ Prn	N	→ duck
NP	→ NP PP	V	→ is
NP	→ NP S	V	→ saw
VP	→ V NP	Prn	→ she
VP	→ V	Prp	→ in
VP	→ VP PP	Det	→ the

Acknowledgments, references, additional reading material