Shift-Reduce
Phrase-Structure Parsing

Miguel Ballesteros
7-11
Overview

- Shift-reduce transition based process
- A Stack:
  - holds the partial parse trees already built
- A Buffer (or queue)
  - holds the incoming words with POS
- Actions
  - SHIFT, REDUCE-BINARY-L/R, REDUCE-UNARY
Overview

- The parsing process starts with all words in the buffer.
- In each step, there is a deterministic decision that decides which action to take given the parsing state.
- We use a classifier that decides what to do given the parsing state.
  - Parsing state: Stack and Buffer.
Example

\[ S \rightarrow \text{NP VP} \]
\[ S \rightarrow \text{Aux NP VP} \]
\[ S \rightarrow \text{VP} \]
\[ \text{NP} \rightarrow \text{Det NOM} \]
\[ \text{NOM} \rightarrow \text{Noun} \]
\[ \text{NOM} \rightarrow \text{Noun NOM} \]
\[ \text{VP} \rightarrow \text{Verb} \]
\[ \text{VP} \rightarrow \text{Verb NP} \]
\[ \text{Det} \rightarrow \text{that | this | a | the} \]
\[ \text{Noun} \rightarrow \text{book | flight | meal | man} \]
\[ \text{Verb} \rightarrow \text{book | include | read} \]
\[ \text{Aux} \rightarrow \text{does} \]
Example

- Book that flight.

Stack

Buffer

Book that flight
Example

- Book that flight.

Stack

Buffer

Book that flight

SHIFT
Example

- Book that flight.

Stack

Buffer

(Book)

that flight
Example

- Book that flight.

Stack

Buffer

(Book)

that flight

REDUCE. Verb → book
Example

- Book that flight.

Stack

(Book)

Buffer

that flight

It could also be Noun → Book

REDUCE. Verb → book
Example

- Book that flight.

Stack

Buffer

(Verb Book) that flight
Example

- Book that flight.

Stack

Buffer

that flight

(Verb Book)

SHIFT
Example

- Book that flight.

Stack

Buffer

(Verb book) (that)

flight
Example

- Book that flight.

Stack

Buffer

(Verb book) (that)

flight

Reduce, Det → that
Example

• Book that flight.

Stack

Verb book) Det that)

Buffer

flight
Example

- Book that flight.

Stack

(Verb book) (Det that)

Buffer

flight

SHIFT
Example

- Book that flight.

Stack

Buffer

(Verb book) (Det that) (flight)
Example

- Book that flight.

Stack

Buffer

(Verb book) (Det that) (flight)

Reduce. Noun → flight
Example

- Book that flight.

Stack

| (Verb book) | (Det that) | (Noun flight) |

Buffer
Example

- Book that flight.

Stack                                   Buffer

(Verb book) (Det that) (Noun flight)

Reduce. NOM → Noun
Example

- Book that flight.

Stack

Buffer

(Verb book) (Det that) (NOM (Noun flight))
Example

- Book that flight.

**Stack**

```
(Verb book)  (Det that)  (NOM (Noun flight))
```

**Buffer**

Reduce. NP → Det NOM
Example

- Book that flight.

Stack

Buffer

(Verb book) (NP (Det that) (NOM (Noun flight)))
Example

- Book that flight.

Stack

| (Verb book) (NP (Det that) (NOM (Noun flight))) |

Buffer

Reduce. VP → Verb NP
Example

- Book that flight.

Stack

(Buffer)

(Verb book) (Det that) (Noun flight))

Buffer
Example

- Book that flight.

Stack

Buffer

(VP (Verb book) (NP (Det that) (NOM (Noun flight))))

Reduce. S → VP
Example

- Book that flight.

Stack

Buffer

(S (VP (Verb book) (NP (Det that) (NOM (Noun flight)))))
Example

- Book that flight.

Is S in the stack? YES! Done.
Example

- Flight a meal (?)  Let's try with something “ungrammatical”

Stack

Buffer

Flight a meal
Example

- Flight a meal (?)  
  
Let's try with something “ungrammatical”

Stack

Buffer

Flight a meal

SHIFT
Example

• Flight a meal (?)

Let's try with something “ungrammatical”

Stack

Buffer

a meal

(Flight)
Example

- Flight a meal (?)  Let's try with something “ungrammatical”

Stack

Buffer

REDUCE. Noun → Flight
Example

- Flight a meal (?)  
  Let's try with something “ungrammatical”

Stack

Buffer

(Noun Flight)

a meal
Example

- Flight a meal (?)

Let's try with something “ungrammatical”

Stack

Buffer

(Noun Fligh)

a meal

Reduce. NOM → Noun
Example

- Flight a meal (?)  
  Let's try with something “ungrammatical”

Stack

Buffer

(NOM (Noun Flight))
a meal
Example

- Flight a meal (?)  
  Let's try with something “ungrammatical”

Stack

Buffer

\[(\text{NOM (Noun Flight)})\]

SHIFT
Example

- Flight a meal (?)  
  Let's try with something “ungrammatical”

Stack

Buffer

(NOM (Noun Flight)) (a)

meal
Example

- Flight a meal (?)

Let's try with something “ungrammatical”

Stack

Buffer

Reduce. Det → a

(NOM (Noun Flight)) (a)

meal
Example

- Flight a meal (?)  
  Let's try with something "ungrammatical"

Stack

Buffer

(NOM (Noun Flight)) (Det a)  
meal
Example

- Flight a meal (?)  

Let's try with something “ungrammatical”

Stack

Buffer

(NOM (Noun Flight)) (Det a)

meal

SHIFT
Example

- Flight a meal (?)  

Let's try with something “ungrammatical”

Stack

Buffer

(NOM (Noun Flight)) (Det a) (meal)
Example

- Flight a meal (?)  Let's try with something “ungrammatical”

Stack

Buffer

(NOM (Noun Flight)) (Det a) (meal)

Reduce
Noun → Meal
Example

- Flight a meal (?) Let's try with something “ungrammatical”

Stack

Buffer

(NOM (Noun Flight)) (Det a) (Noun (meal))
Example

- Flight a meal (?)

Let's try with something “ungrammatical”

Stack         Buffer

\[(\text{NOM (Noun Flight)}) (\text{Det a}) (\text{Noun (meal)})]\]

Reduce

NOM → Noun
Example

• Flight a meal (?)  Let's try with something “ungrammatical”

Stack

Buffer

(NOM (Noun Flight)) (Det a) (NOM (Noun (meal)))
Example

• Flight a meal (?) Let's try with something “ungrammatical”

Stack

Buffer

\[
\text{(NOM (Noun Flight)) (Det a) (NOM (Noun (meal)))}
\]

Reduce

NP \rightarrow \text{Det NOM}
Example

- Flight a meal (?) Let's try with something “ungrammatical”

Stack

Buffer

\[ \text{(NOM (Noun Flight)) (NP(Det a) (NOM (Noun (meal))))} \]

Now we have:

NP at the top of the stack.
NOM NP at the top of the stack
We are done, but S is not at the top of the stack.

**We reject the input.**
Shift-Reduce Parser

Start with the sentence to be parsed in an input buffer.

- a "shift" action corresponds to pushing the next input symbol from the buffer onto the stack
- a "reduce" action occurs when we have a rule’s RHS on top of the stack. To perform the reduction, we pop the rule’s RHS off the stack and replace it with the terminal on the LHS of the corresponding rule.

- (When either "shift" or "reduce" is possible, choose one arbitrarily.)

If you end up with only the Start symbol on the stack, then success!
If you don’t, and you cannot and no "shift" or "reduce" actions are possible, backtrack.
Running time

- Linear in the length of the sentence. \( O(n) \).

- In terms of memory, it is also cheap.
Problems

• Unable to deal with empty categories: termination problem, unless rewriting empties as constituents is somehow restricted (but then it’s generally incomplete)

• Useless work: locally possible, but globally impossible.

• Repeated work: anywhere there is common substructure.

• Backtracking makes the parsing problem not linear anymore.
  However, in real parsing, this is not the case.