Parsing (part 1)

Chapter 13 J&M'09

Parsing

We've been parsing for ages now...

- Shift-reduce : Yngve (1955)
- Left-corner : Rosenkrantz & Lewis (1970)
- Earley : Earley (1970)
- CKY : Cocke-Kasami-Younger Parsing (1965,1967,1970)

Bottom-Up Parsing: Shift-Reduce

Given a grammar Γ, a finte sequence of tokens w_1 ... w_n, a stack (initialized as empty), and a chart (recording the input, the stack and the applied rule and the alternative rule applications, at every step):

While stack not singleton and input not empty, do:

1. replace $Y_1 ... Y_n$ elements at the top of the stack with X if there is a rule $X \rightarrow Y_n ... Y_1$
   (Reduce)
2. move first token w from input to the top of the stack
   (Shift)

The chart should also keep track of alternative choice points at every step, so the parser can backtrack if needed.

Parsing

Input	**Stack**	**Actions**
Tom probably sneezed	–	initial state
probably sneezed	Tom	shift
probably sneezed	PN	reduce
probably sneezed	NP	reduce
sneezed	probably NP	shift
sneezed	ADV NP	reduce
–	sneezed ADV NP	shift
–	IV ADV NP	reduce
–	VP ADV NP	reduce
–	VP NP	reduce
–	S	reduce

Grammar

S → NP VP NP → PN PN → Tom

VP → ADV VP VP → IV IV → sneezed

ADV → probably

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift
a record	TV NP	reduce

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift
a record	TV NP	reduce
record	a TV NP	shift

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift
a record	TV NP	reduce
record	a TV NP	shift
record	DT TV NP	reduce

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift
a record	TV NP	reduce
record	a TV NP	shift
record	DT TV NP	reduce
–	record DT TV NP	shift

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift
a record	TV NP	reduce
record	a TV NP	shift
record	DT TV NP	reduce
–	record DT TV NP	shift
–	N DT TV NP	reduce

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift
a record	TV NP	reduce
record	a TV NP	shift
record	DT TV NP	reduce
–	record DT TV NP	shift
–	N DT TV NP	reduce
–	NP TV NP	reduce

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift
a record	TV NP	reduce
record	a TV NP	shift
record	DT TV NP	reduce
–	record DT TV NP	shift
–	N DT TV NP	reduce
–	NP TV NP	reduce
–	VP NP	reduce

Parsing

S → NP VP	DT → a	NP → PRP	N → record
VP → TV NP	TV → recorded	NP → DT N	PRP → I

Input	**Stack**	**Actions**
I recorded a record	–	initial state
recorded a record	I	shift
recorded a record	PRP	reduce
recorded a record	NP	reduce
a record	recorded NP	shift
a record	TV NP	reduce
record	a TV NP	shift
record	DT TV NP	reduce
–	record DT TV NP	shift
–	N DT TV NP	reduce
–	NP TV NP	reduce
–	VP NP	reduce
–	S	reduce

Parsing

```
1  rule(s,[np,vp]).
2  rule(np,[dt,n]).
3  rule(n,[n,pp]).
4  rule(pp,[p,np]).
5  rule(np,[prp]).
6  rule(vp,[tv,np]).
7  rule(vp,[iv]).
8  lex(p,in).
9  lex(dt,the).
10 lex(tv,saw).
11 lex(iv,sneezed).
12 lex(n,cat).
13 lex(n,hat).
```

```
1  ?- sr_parse([the,cat,in,the,hat,sneezed]).
2  true
```

Prolog shift-reduce parser

```
1  sr_parse(Sentence):-
2          srparse([],Sentence).
3
4  srparse([_],[]).
5
6  srparse([Y,X|MoreStack],Words):-
7         rule(LHS,[X,Y]),
8         srparse([LHS|MoreStack],Words).
9
10 srparse([X|MoreStack],Words):-
11        rule(LHS,[X]),
12        srparse([LHS|MoreStack],Words).
13
14 srparse(Stack,[Word|Words]):-
15         lex(X,Word),
16         srparse([X|Stack],Words).
```

Prolog shift-reduce analyzer (outputs a parse tree)

```
1  sr_parse(Sentence,Parse):-
2          srparse([],Sentence,[],Parse).
3
4  srparse([_],[],[X],X).
5
6  srparse([Y,X|MoreStack],Words,[Z,W|ListNodes],Parse):-
7         rule(LHS,[X,Y]),   Node =.. [LHS,W,Z],
8         srparse([LHS|MoreStack],Words,[Node|ListNodes],Parse).
9
10 srparse([X|MoreStack],Words,[Y|ListNodes],Parse):-
11        rule(LHS,[X]), Node =.. [LHS,Y],
12        srparse([LHS|MoreStack],Words,[Node|ListNodes],Parse).
13
14 srparse(Stack,[Word|Words],ListNodes,Parse):-
15         lex(X,Word),   Node =.. [X,Word],
16         srparse([X|Stack],Words,[Node|ListNodes],Parse).
```

Parsing

```
1  rule(s,[np,vp]).
2  rule(np,[dt,n]).
3  rule(n,[n,pp]).
4  rule(pp,[p,np]).
5  rule(np,[prp]).
6  rule(vp,[tv,np]).
7  rule(vp,[iv]).
8  lex(p,in).
9  lex(dt,the).
10 lex(tv,saw).
11 lex(iv,sneezed).
12 lex(n,cat).
13 lex(n,hat).
```

```
1  ?- sr_parse([the,cat,in,the,hat,sneezed],P).
2    P = s(np(dt(the), n(n(cat), pp(p(in), np(dt(the), n(cat))))),
         vp(iv(sneezed))) ;
3    false.
```

Parsing

Left-corner parsing: combines the best aspects of top-down and bottom-up parsing strategies.

- Like bottom-up parsing, use the input to avoid building irrelevant parses;
- Like top-down parsing, uses the grammar rules to avoid trying to combine phrases that cannot be combined.

Parsing

Initialization: empty the stack.
While input not empty and stack not equal to S $\rightarrow X_1 ... X_n \bullet$, do:

- Complete.
  If the top stack item is $Z \rightarrow W_1 ... W_i \bullet$ and the next item down is $X \rightarrow Y_1 ... Y_j \bullet Z K_0 ... K_n$, then replace those two items with $X \rightarrow Y_1 ... Y_j Z \bullet K_0 ... K_n$.

- Predict.
  If $X \rightarrow Y_1 ... Y_n \bullet$ is on top of the stack, and the grammar contains a rule $Z \rightarrow X K_0 ... K_n$ then replace the top item on the stack with $Z \rightarrow X \bullet K_0 ... K_n$.

- Shift.
  If the next word in the input is w and $X \rightarrow w$ is part of the grammar then shift $X \rightarrow w \bullet$ onto the top of the stack.

As usual, a chart keeps track of the states and alternative choice points so the parser can backtrack to it when parsing fails.

Parsing

Input read	**Stack**	**Action**
I recorded it	–	Initialize
recorded it	[PRN → I •]	Shift
recorded it	[NP → PRN •]	Predict
recorded it	[S → NP • VP]	Predict
it	[TV → recorded •] [S → NP • VP]	Shift
it	[VP → TV • NP] [S → NP • VP]	Predict
–	[PRN → it •] [VP → TV • NP] [S → NP • VP]	Shift
–	[NP → prn •] [VP → TV • NP] [S → NP • VP]	Predict
–	[VP → TV NP •] [S → NP • VP]	Complete
–	[S → NP VP •]	Complete

S → NP VP DT → a NP → PRN N → record
VP → TV NP TV → recorded PRN → it PRN → I

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift
cat sneezed	[N → ADJ • N] [NP → DT • N]	Predict

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift
cat sneezed	[N → ADJ • N] [NP → DT • N]	Predict
sneezed	[N → cat •] [N → ADJ • N] [NP → DT • N]	Shift

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift
cat sneezed	[N → ADJ • N] [NP → DT • N]	Predict
sneezed	[N → cat •] [N → ADJ • N] [NP → DT • N]	Shift
sneezed	[N → ADJ N •] [NP → DT • N]	Complete

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift
cat sneezed	[N → ADJ • N] [NP → DT • N]	Predict
sneezed	[N → cat •] [N → ADJ • N] [NP → DT • N]	Shift
sneezed	[N → ADJ N •] [NP → DT • N]	Complete
sneezed	[NP → DT N •]	Complete

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift
cat sneezed	[N → ADJ • N] [NP → DT • N]	Predict
sneezed	[N → cat •] [N → ADJ • N] [NP → DT • N]	Shift
sneezed	[N → ADJ N •] [NP → DT • N]	Complete
sneezed	[NP → DT N •]	Complete
sneezed	[S → NP • VP]	Predict

Parsing

$S \rightarrow NP\ VP$ $\quad DT \rightarrow a$ $\quad N \rightarrow ADJ\ N$ $\quad N \rightarrow cat$

$VP \rightarrow IV$ $\quad IV \rightarrow sneezed$ $\quad NP \rightarrow DT\ N$ $\quad ADJ \rightarrow black$

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift
cat sneezed	[N → ADJ • N] [NP → DT • N]	Predict
sneezed	[N → cat •] [N → ADJ • N] [NP → DT • N]	Shift
sneezed	[N → ADJ N •] [NP → DT • N]	Complete
sneezed	[NP → DT N •]	Complete
sneezed	[S → NP • VP]	Predict
–	[IV → sneezed •] [S → NP • VP]	Shift

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift
cat sneezed	[N → ADJ • N] [NP → DT • N]	Predict
sneezed	[N → cat •] [N → ADJ • N] [NP → DT • N]	Shift
sneezed	[N → ADJ N •] [NP → DT • N]	Complete
sneezed	[NP → DT N •]	Complete
sneezed	[S → NP • VP]	Predict
–	[IV → sneezed •] [S → NP • VP]	Shift
–	[VP → IV •] [S → NP • VP]	Predict

Parsing

S → NP VP	DT → a	N → ADJ N	N → cat
VP → IV	IV → sneezed	NP → DT N	ADJ → black

Input read	**Stack**	**Action**
a black cat sneezed	–	Initialize
black cat sneezed	[DT → a •]	Shift
black cat sneezed	[NP → DT • N]	Predict
cat sneezed	[ADJ → black •] [NP → DT • N]	Shift
cat sneezed	[N → ADJ • N] [NP → DT • N]	Predict
sneezed	[N → cat •] [N → ADJ • N] [NP → DT • N]	Shift
sneezed	[N → ADJ N •] [NP → DT • N]	Complete
sneezed	[NP → DT N •]	Complete
sneezed	[S → NP • VP]	Predict
–	[IV → sneezed •] [S → NP • VP]	Shift
–	[VP → IV •] [S → NP • VP]	Predict
–	[S → NP VP •]	Complete

Parsing

```
1  lc_parse(String) :-
2      leftcorner_recognize(s,String,[]).
3
4  leftcorner_recognize(Cat,[Word|StringIn],StringOut) :-
5      lex(WCat,Word),
6      complete(Cat,WCat,StringIn,StringOut).
7
8  complete(Cat,Cat,String,String).
9
10 complete(Cat,SubCat,StringIn,StringOut) :-
11     rule(LHS,[SubCat|Cats]),
12     matches(Cats,StringIn,String1),
13     complete(Cat,LHS,String1,StringOut).
14
15 matches([],String,String).
16 matches([Cat|Cats],StringIn,StringOut) :-
17     leftcorner_recognize(Cat,StringIn,String1),
18     matches(Cats,String1,StringOut).
```

```
?- lc_parse([the,cat,in,the,hat,sneezed]).
true ;
false.

?- lc_parse([the,in,the,hat,sneezed]).
false.
```

Parsing/Grammar profiling

```
?- time(sr_parse([the,big,cat,in,the,hat,in,the,hat,saw,tom]))
 387 inferences, 0.000 CPU in 0.000 seconds

?- time(lc_parse([the,big,cat,in,the,hat,in,the,hat,saw,tom])).
 341 inferences, 0.000 CPU in 0.000 seconds
```

Parsing/Grammar profiling (all parses)

```
1  stats_t(S,sr,Total):-
2      statistics(cputime,I1),
3      findall(S2, (S2=S, sr_parse(S)),_),
4      statistics(cputime,I2),
5      Total is I2 - I1.
6
7  stats_t(S,lc,Total):-
8      statistics(cputime,I1),
9      findall(S2, (S2=S, lc_parse(S)),_),
10     statistics(cputime,I2),
11     Total is I2 - I1.
```

```
1  ?- stats_t([the,big,cat,in,the,hat,in,the,hat,saw,tom],sr,T).
2  T = 0.
3
4  ?- stat_inf([the,big,cat,in,the,hat,in,the,hat,saw,tom],lc,T).
5  T = 0.
```

Parsing

(1) The captain said you claimed the sergeant denied the antenna in the hangar broke in the attack near the gate.

(14 parses)

Parser	**Steps**	**Time**
Shift-reduce	19129	0.01 secs
Left-corner	5453	0.01 secs

(on a 2.4 Ghz Intel Core 2 Duo with 4 GB 1067 MHz DDR4)