14.5
Supplemental: Context free grammars: The CYK Algorithm

14.5.1
CYK: Problem statement, basic idea, and an example

Parsing

We saw regular languages and context free languages.

Most programming languages are specified via context-free grammars. Why?

- CFLs are sufficiently expressive to support what is needed.
- At the same time one can “efficiently” solve the parsing problem: given a string/program w, is it a valid program according to the CFG specification of the programming language?

CFG specification for C

```
<relational-expression> ::= <shift-expression>
                          | <relational-expression> < <shift-expression>
                          | <relational-expression> > <shift-expression>
                          | <relational-expression> <= <shift-expression>
                          | <relational-expression> >= <shift-expression>

<shift-expression> ::= <additive-expression>
                     | <shift-expression> << <additive-expression>
                     | <shift-expression> >> <additive-expression>

<additive-expression> ::= <multiplicative-expression>
                        | <additive-expression> + <multiplicative-expression>
                        | <additive-expression> - <multiplicative-expression>

<multiplicative-expression> ::= <cast-expression>
                              | <multiplicative-expression> * <cast-expression>
                              | <multiplicative-expression> / <cast-expression>
                              | <multiplicative-expression> % <cast-expression>

<cast-expression> ::= <unary-expression>
                    | ( <type-name> ) <cast-expression>

<unary-expression> ::= <postfix-expression>
                     | ++ <unary-expression>
                     | -- <unary-expression>
                     | <unary-operator> <cast-expression>
                     | sizeof <unary-expression>
                     | sizeof <type-name>

<postfix-expression> ::= <primary-expression>
                       | <postfix-expression> [ <expression> ]
                       | <postfix-expression> ( {<assignment-expression>}* )
```

Algorithmic Problem

Given a CFG $G = (V, T, P, S)$ and a string $w \in T^*$, is $w \in L(G)$?

- That is, does S derive w?
- Equivalently, is there a parse tree for w?

Simplifying assumption: G is in Chomsky Normal Form (CNF)

- Productions are all of the form $A \to BC$ or $A \to a$.
  If $\epsilon \in L$ then $S \to \epsilon$ is also allowed.
  (This is the only place in the grammar that has an ε.)
- Every CFG G can be converted into CNF form via an efficient algorithm
- Advantage: parse tree of constant degree.

Algorithmic Problem

Given a CFG $G = (V, T, P, S)$ and a string $w \in T^*$, is $w \in L(G)$?

- That is, does S derive w?
- Equivalently, is there a parse tree for w?

Simplifying assumption: G is in Chomsky Normal Form (CNF)

- Productions are all of the form $A \rightarrow BC$ or $A \rightarrow a$.
  If $\epsilon \in L$ then $S \rightarrow \epsilon$ is also allowed.
  (This is the only place in the grammar that has an ε.)
- Every CFG G can be converted into CNF form via an efficient algorithm
- Advantage: parse tree of constant degree.

Towards Recursive Algorithm

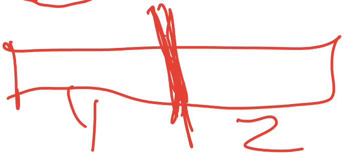
CYK Algorithm = Cocke-Younger-Kasami algorithm

Assume G is a CNF grammar.

S derives $w \iff$ one of the following holds:

- $|w| = 1$ and $S \to w$ is a rule in P
- $|w| > 1$ and there is a rule $S \to AB$ and a split $w = uv$ with $|u|, |v| \geq 1$ such that A derives u and B derives v

Observation: Subproblems generated require us to know if some non-terminal A will derive a substring of w.



Towards Recursive Algorithm

CYK Algorithm = Cocke-Younger-Kasami algorithm

Assume G is a CNF grammar.

S derives w $\iff$ one of the following holds:

- $|w| = 1$ and $S \to w$ is a rule in P
- $|w| > 1$ and there is a rule $S \to AB$ and a split $w = uv$ with $|u|, |v| \geq 1$ such that A derives u and B derives v

Observation: Subproblems generated require us to know if some non-terminal A will derive a substring of w.

Example

$S \rightarrow \epsilon \mid AB \mid XB$
$Y \rightarrow AB \mid XB$
$X \rightarrow AY$
$A \rightarrow 0$
$B \rightarrow 1$

$XB \rightarrow AYB$

$Y \rightarrow 01 \mid AYB$

$Y \rightarrow 01 \mid 0Y1$

$L(Y) = \{0^i 1^i \mid i \geq 1\}$

Question:

- Is $\mathbf{000111}$ in $L(G)$?
- Is $\mathbf{00011}$ in $L(G)$?

Order of evaluation for iterative algorithm: increasing order of substring length.

$\{\epsilon, 01, 0(0^i 1^i)1 \mid i \geq 1\} = \{0^i 1^i \mid i \geq 0\}$

Example: 000111

$$S \rightarrow \epsilon \mid AB \mid XB$$
$$Y \rightarrow AB \mid XB$$
$$X \rightarrow AY$$
$$A \rightarrow 0$$
$$B \rightarrow 1$$

Input:	0	0	0	1	1	1

Example: **000111**

$S \to \epsilon \mid AB \mid XB$
$Y \to AB \mid XB$
$X \to AY$
$A \to 0$
$B \to 1$

Len=1	A	A	A	B	B	B
Input:	**0**	**0**	**0**	**1**	**1**	**1**

Example: **000111**

$S \to \epsilon \mid AB \mid XB$
$Y \to AB \mid XB$
$X \to AY$
$A \to 0$
$B \to 1$

Len=3		X				
Len=2			Y			
Len=1	A	A	A	B	B	B
Input:	**0**	**0**	**0**	**1**	**1**	**1**

Example: **000111**

$$S \rightarrow \epsilon \mid AB \mid XB$$
$$Y \rightarrow AB \mid XB$$
$$X \rightarrow AY$$
$$A \rightarrow 0$$
$$B \rightarrow 1$$

$Y \rightarrow XB$

$S \rightarrow XB$

Len=4		Y,S				
Len=3		X				
Len=2			Y			
Len=1	A	A	A	B	B	B
Input:	**0**	**0**	**0**	**1**	**1**	**1**

X B

Example: **000111**

$S \to \epsilon \mid AB \mid XB$
$Y \to AB \mid XB$
$X \to AY$
$A \to 0$
$B \to 1$

Len=5	X					
Len=4		Y,S				
Len=3		X				
Len=2			Y			
Len=1	A	A	A	B	B	B
Input:	**0**	**0**	**0**	**1**	**1**	**1**

Example: **000111**

$S \rightarrow \epsilon \mid AB \mid XB$
$Y \rightarrow AB \mid XB$
$X \rightarrow AY$
$A \rightarrow \mathbf{0}$
$B \rightarrow \mathbf{1}$

Len=6	S					
Len=5	X					
Len=4		Y,S				
Len=3		X				
Len=2			Y			
Len=1	A	A	A	B	B	B
Input:	**0**	**0**	**0**	**1**	**1**	**1**

Example II: 00111

$S \to \epsilon \mid AB \mid XB$
$Y \to AB \mid XB$
$X \to AY$
$A \to 0$
$B \to 1$

Input:	0	0	1	1	1

Example II: **00111**

$$S \rightarrow \epsilon \mid AB \mid XB$$
$$Y \rightarrow AB \mid XB$$
$$X \rightarrow AY$$
$$A \rightarrow 0$$
$$B \rightarrow 1$$

Len=1	A	A	B	B	B
Input:	**0**	**0**	**1**	**1**	**1**

Example II: **00111**

$S \rightarrow \epsilon \mid AB \mid XB$
$Y \rightarrow AB \mid XB$
$X \rightarrow AY$
$A \rightarrow 0$
$B \rightarrow 1$

Len=3	X				
Len=2		Y			
Len=1	A	A	B	B	B
Input:	**0**	**0**	**1**	**1**	**1**

Example II: **00111**

$$S \to \epsilon \mid AB \mid XB$$
$$Y \to AB \mid XB$$
$$X \to AY$$
$$A \to 0$$
$$B \to 1$$

Len=4	Y,S				
Len=3	X				
Len=2		Y			
Len=1	A	A	B	B	B
Input:	**0**	**0**	**1**	**1**	**1**

Example II: **00111**

$S \rightarrow \epsilon \mid AB \mid XB$
$Y \rightarrow AB \mid XB$
$X \rightarrow AY$
$A \rightarrow 0$
$B \rightarrow 1$

Len=5					
Len=4	Y,S				
Len=3	X				
Len=2		Y			
Len=1	A	A	B	B	B
Input:	**0**	**0**	**1**	**1**	**1**

THE END

...

(for now)