CSE 322
Introduction to Formal Models in Computer Science
Cocke-Kasami-Younger Algorithm example

The following algorithm, which is described on pages 262-3 of the 2nd edition of Sipser's text (pages 240-241 of the 1st edition), is an $O(n^3)$ algorithm (originally given by Cocke, Kasami and Younger) to determine whether or not a Chomsky normal form grammar G generates a given string w of length n.

The main idea of the algorithm on input $w = a_1 \cdots a_n$ is to build a table that for each pair (i, j) with $1 \le i \le j \le n$ stores which variables of the grammar could possibly generate the substring $a_i \cdots a_j$ of w. This is easy when $i = j$ and the algorithm starts there and builds up the table for longer and longer strings until the algorithm has computed which symbols can generate all of w. This general technique of solving a recursive search bottom up is called *dynamic programming*.

```
On input $w = a_1 \cdots a_n$:
| If $w = \varepsilon$ and $S \to \varepsilon$ is a rule, accept.                [handle $w = \varepsilon$ case]
|  For $i = 1$ for $n$:                                         [examine each substring of length 1]
|      For each variable $A$:
|          Test whether $A \to a_i$ is a rule.
|          If so, place $A$ in $table[i, i]$
| For $\ell = 2$ to $n$                                          [$\ell$ is the length of the substring]
|     For $i = 1$ to $n - \ell + 1$:                              [$i$ is the start position of the substring]
|         Let $j = i + \ell - 1$                                  [$j$ is the end position of the substring]
|         For $k = i$ to $j - 1$                                  [$k$ is the position just before the split]
|             For each rule $A \to BC$:
|                 If $table[i, k]$ contains $B$ and $table[k + 1, j]$ contains $C$ put $A$ in $table[i, j]$.
| If $S$ is in $table[1, n]$, accept. Otherwise reject
```

Figure 1: Cocke-Kasami-Younger Algorithm

We consider an example of how the algorithm works on the next couple of pages

Consider the following Chomsky Normal Form grammar:

$$\begin{array}{lcl} S & \rightarrow & AT \mid AU \mid \varepsilon \\ T & \rightarrow & UB \mid b \\ U & \rightarrow & AT \mid UT \\ A & \rightarrow & a \\ B & \rightarrow & b \end{array}$$

We now show the computation of the tableau for the Cocke-Kasami-Younger algorithm on input $aaabbb$.

We begin with the single symbols:

6						B, T
5					B, T	
4				B, T		
3			A			
2		A				
1	A					
	1	2	3	4	5	6

Now we fill in the table for each entry just below the diagonal, looking for rules whose right hand side is composed of an element of the cell just to the left, followed by an element of the cell just above. The only right hand side that is of this form is AT.

6						B, T
5					B, T	$\emptyset$
4				B, T	$\emptyset$	
3			A	S, U		
2		A	$\emptyset$			
1	A	$\emptyset$				
	1	2	3	4	5	6

Now we fill in the next range. Note that the general form of the right-hand sides of rules for a cell one looks for involves a series of combinations beginning with the combination of the entry just to that cell's left with the highest entry in that cell's column and moving simulataneous leftward in the row and downward in the column.

In this case, most of the pairs involve empty cells ($\emptyset$) as one of the elements and so nothing can be generated. The combination of AU from the cells (2,2) and (3,4) together generate an S in cell (2,4). The combinations UT and UB obtained from cells (3,4) and (5,5) generate U and T, respectively in entry (3,5).

6						B,T
5					B,T	$\emptyset$
4				B,T	$\emptyset$	$\emptyset$
3			A	S,U	U,T	
2		A	$\emptyset$	S		
1	A	$\emptyset$	$\emptyset$			
	1	2	3	4	5	6

For the next layer, UT and UB are both obtained from cells (3,5) and (6,6) and generate U and T respectively in cell (3,6). Also, AT and AU both obtained from (2,2) and (3,5) generate U and S in cell (2,5).

6						B,T
5					B,T	$\emptyset$
4				B,T	$\emptyset$	$\emptyset$
3			A	S,U	U,T	U,T
2		A	$\emptyset$	S	S,U	
1	A	$\emptyset$	$\emptyset$	$\emptyset$		
	1	2	3	4	5	6

Now an AU obtained from (1,1) and (2,5) together generate an S in cell (1,5). An AT and AU obtained from (2,2) and (3,6) generate U and S in cell (2,6). Also, a UB obtained from (2,5) and (6,6) generates a T in cell (2,6), a UT obtained from the same pair of cells is another way to generate S in (2,6).

6						B,T
5					B,T	$\emptyset$
4				B,T	$\emptyset$	$\emptyset$
3			A	S,U	U,T	U,T
2		A	$\emptyset$	S	S,U	S,T,U
1	A	$\emptyset$	$\emptyset$	$\emptyset$	S	
	1	2	3	4	5	6

Finally, an AT and an AU from cells (1,1) and (2,6) creates S and U in cell (1,6). Since S is in this cell the input $aaabbb$ is generated by the grammar.

6						B,T
5					B,T	$\emptyset$
4				B,T	$\emptyset$	$\emptyset$
3			A	S,U	U,T	U,T
2		A	$\emptyset$	S	S,U	S,T,U
1	A	$\emptyset$	$\emptyset$	$\emptyset$	S	S,U
	1	2	3	4	5	6