Context-Free Grammar - Exercise

Problem:

Write a context-free grammar for L = $\{a^n b^m c^k : n+2m=k, n\geq 0, m\geq 0\}$.

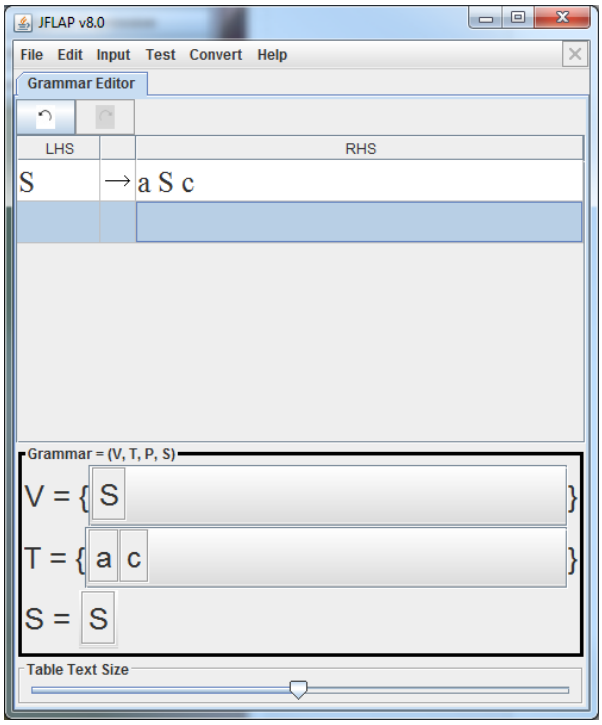
Solution:

First, we examine the kinds of words produced by this set. One way to do that is to tabulate the different values of n, m, and k and list the corresponding words for the different values.

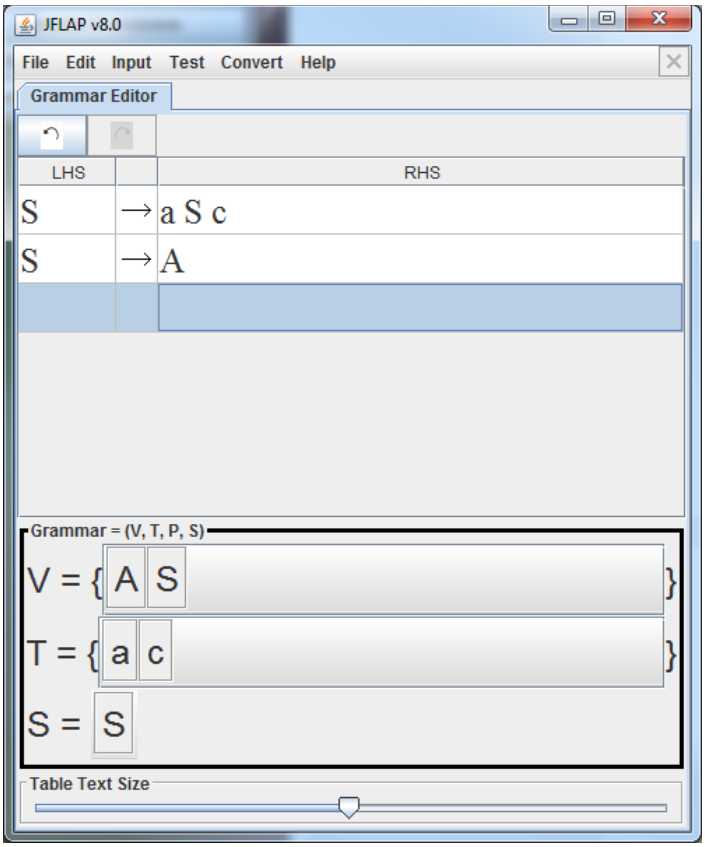
n	*m*	*k*	*w*
0	0	0	λ
0	1	2	bcc
1	0	1	ac
1	1	3	abccc
1	2	5	abbccccc
			…

We see that for each a in the word, there is a c produced. For each b in the word, there are two b's. Also, a's, b's, and c's appear in that sequence; i.e., cccba is not accepted.

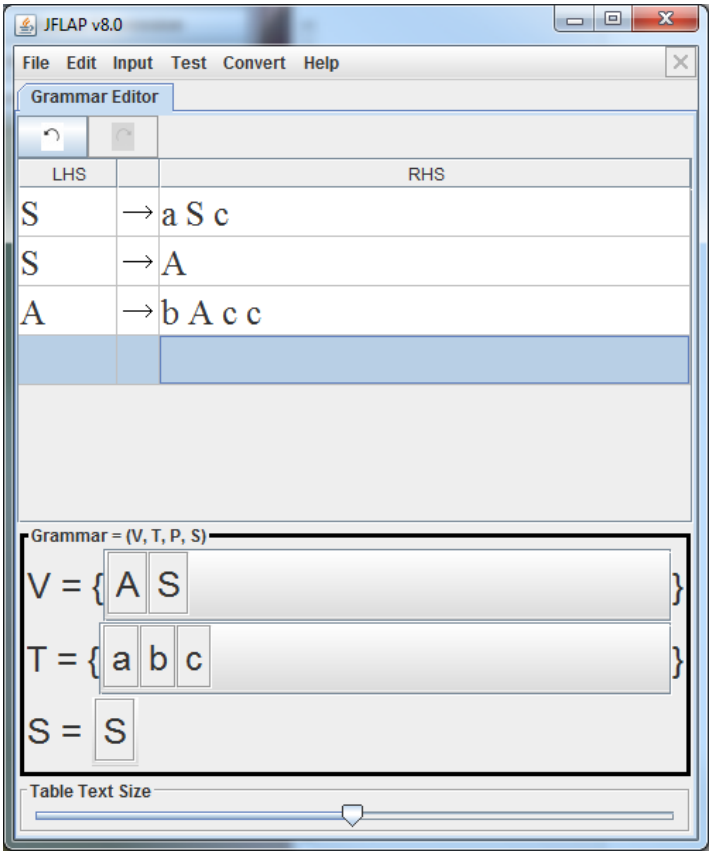
Open JFLAP and click on *Grammar*. Notice that the resulting window has a LHS column and a RHS column. Each row refers to one rule in the context-free grammar to be created. Using S as the start symbol, fill the first row with S → aSc like so:



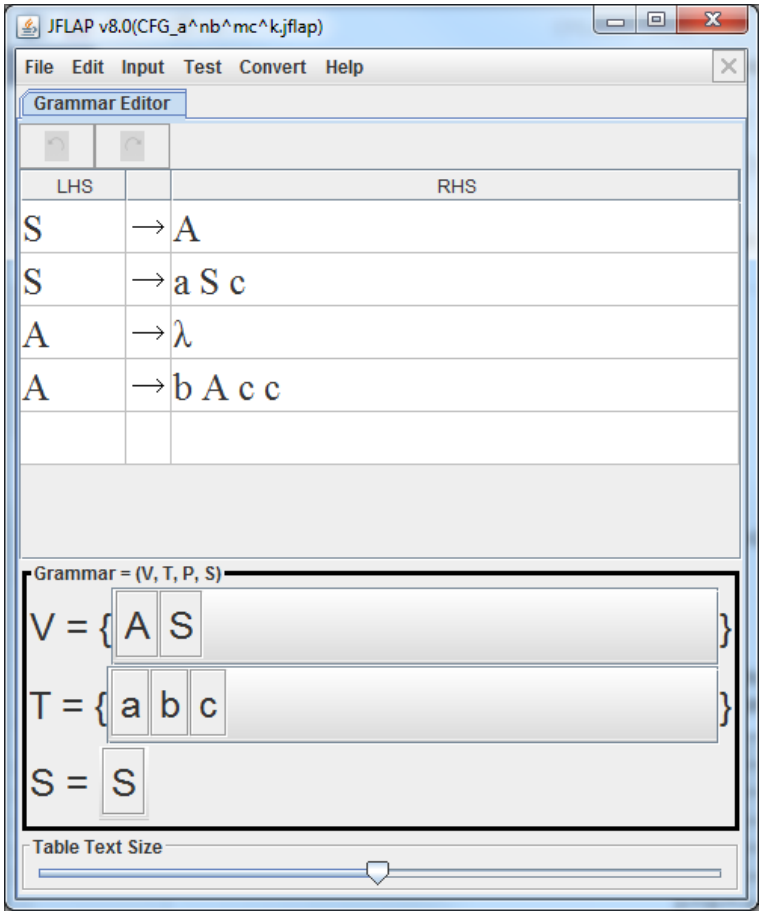
The S in the RHS allows the rule to become recursive. This rule states that for every a created, there is a c at the end of the word created. Next, we introduce a production rule to allow b's to be created. Note that once b's are being produced, we cannot create any more a's. We do this by adding a second rule, S → A.



Next, we work on letting b's be created making sure that there are two c's for every b, A → bAcc. This rule is also recursive allowing an infinite number of b's to be created.

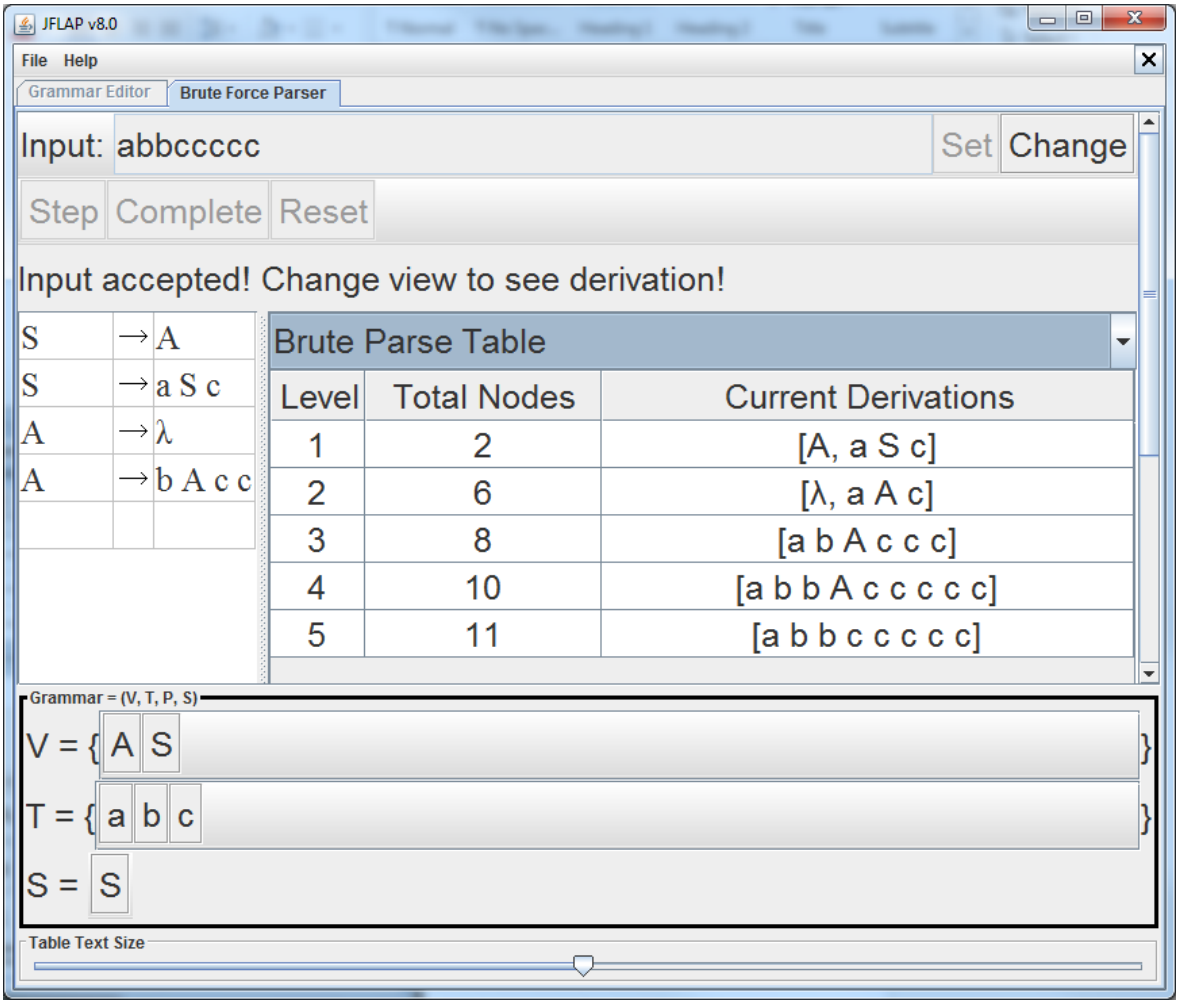


Last, we add a rule to terminate the derivation, A → λ.



That completes the grammar. Next, we test for grammar type with *Test* > *Test for Grammar Type*. JFLAP reports that it is a context-free grammar.

Lastly, we may test an input string. Click *Input* > *Brute Force Parse* and enter *abbccccc* as the input. Click *Complete*. JFLAP reports this string as *accepted*. The brute parse table also shows the 5 derivation levels and the progression from S to the test input.


JFLAP v8.0
File Help
Grammar Editor
Brute Force Parser
Input: abbccccc
Set
Change
Step
Complete
Reset
Input accepted! Change view to see derivation!
S → A
S → a S c
A → λ
A → b A c c
Brute Parse Table
Level
Total Nodes
Current Derivations
1
2
[A, a S c]
2
6
[λ, a A c]
3
8
[a b A c c c]
4
10
[a b b A c c c c c]
5
11
[a b b c c c c c]
Grammar = (V, T, P, S)
V = { A S }
T = { a b c }
S = S
Table Text Size