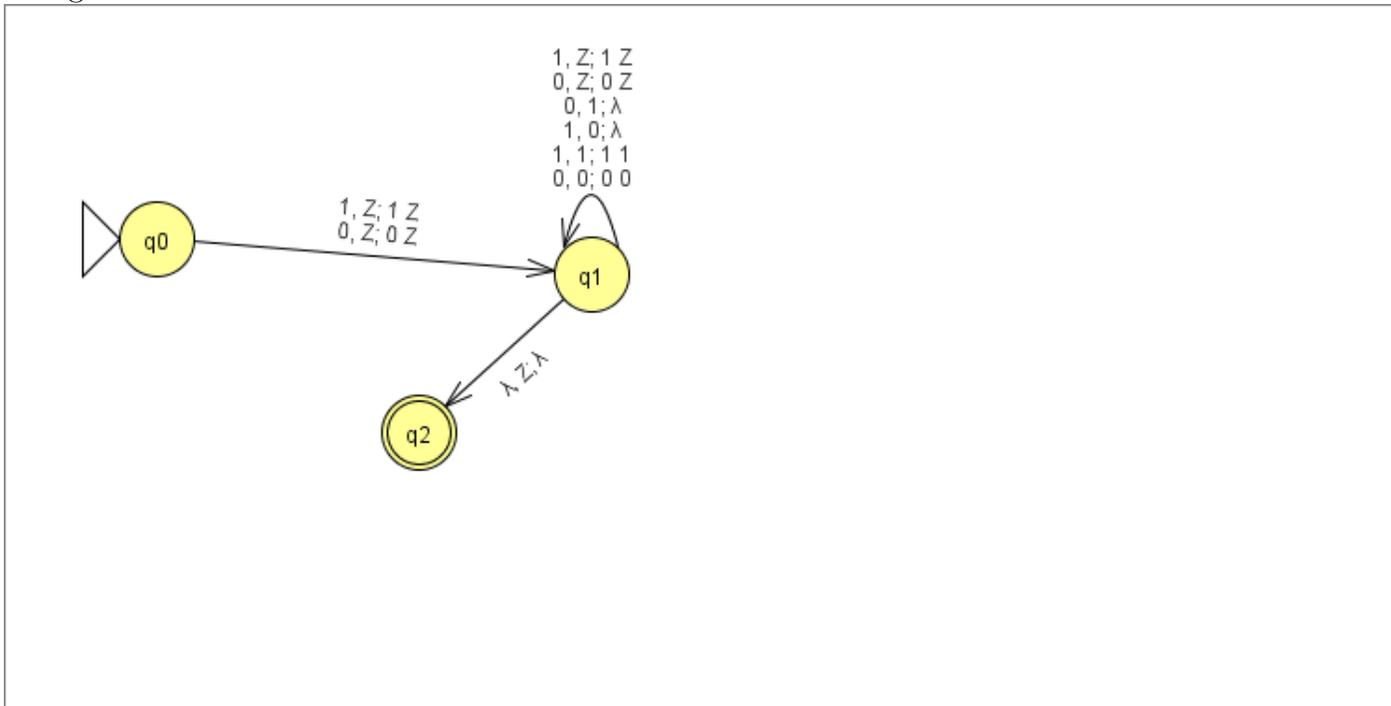


## Converting a PDA to a CFG

The following non-deterministic PDA accepts strings that contain the same number of 0s and 1s.

The goal is to convert this over to a CFG.



### Solution

The solution to this PDA conversion is quite long, but the individual steps are the same as any conversion and we will step through them in JFLAP.

When you click the Convert to Grammar option, you can pick any of the transitions and the corresponding context free grammar rules are generated.

For any of the transitions that pop a symbol off the stack, there is only one grammar rule.

For instance here we see the rule generated when we take care of the  $1, 0 : \lambda$  transition that goes from  $q_1$  back to  $q_1$ . This results in the rule  $(q_1 0 q_1) \rightarrow 1$

Step

Step To Completion

What's Left?

Export

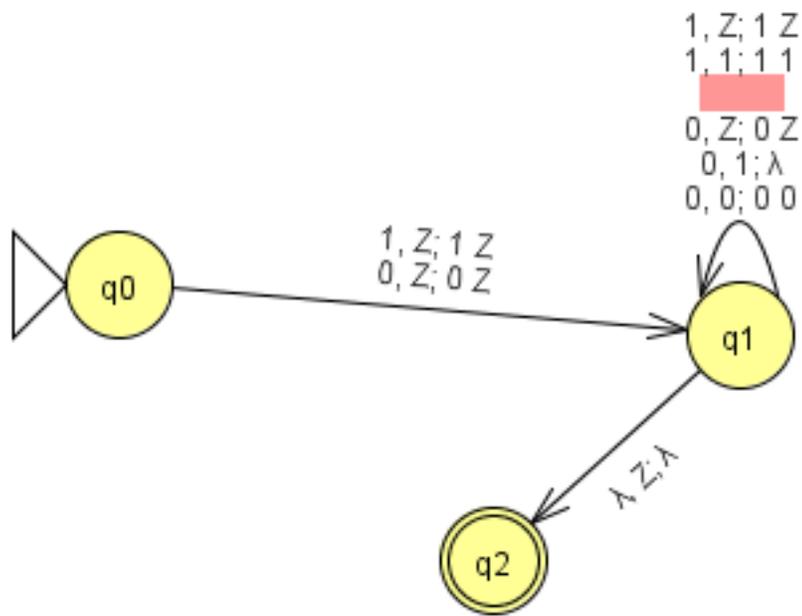
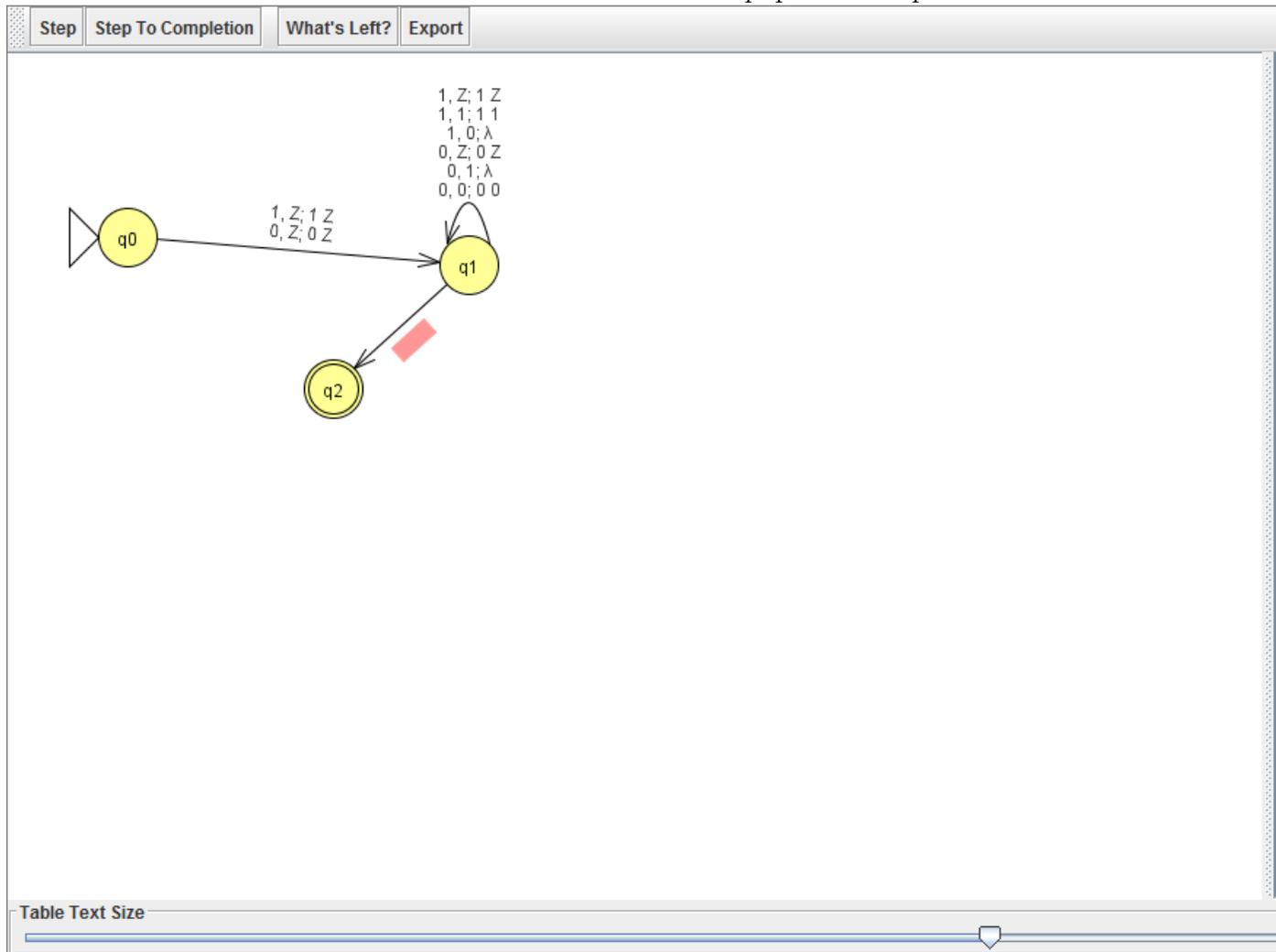


Table Text Size

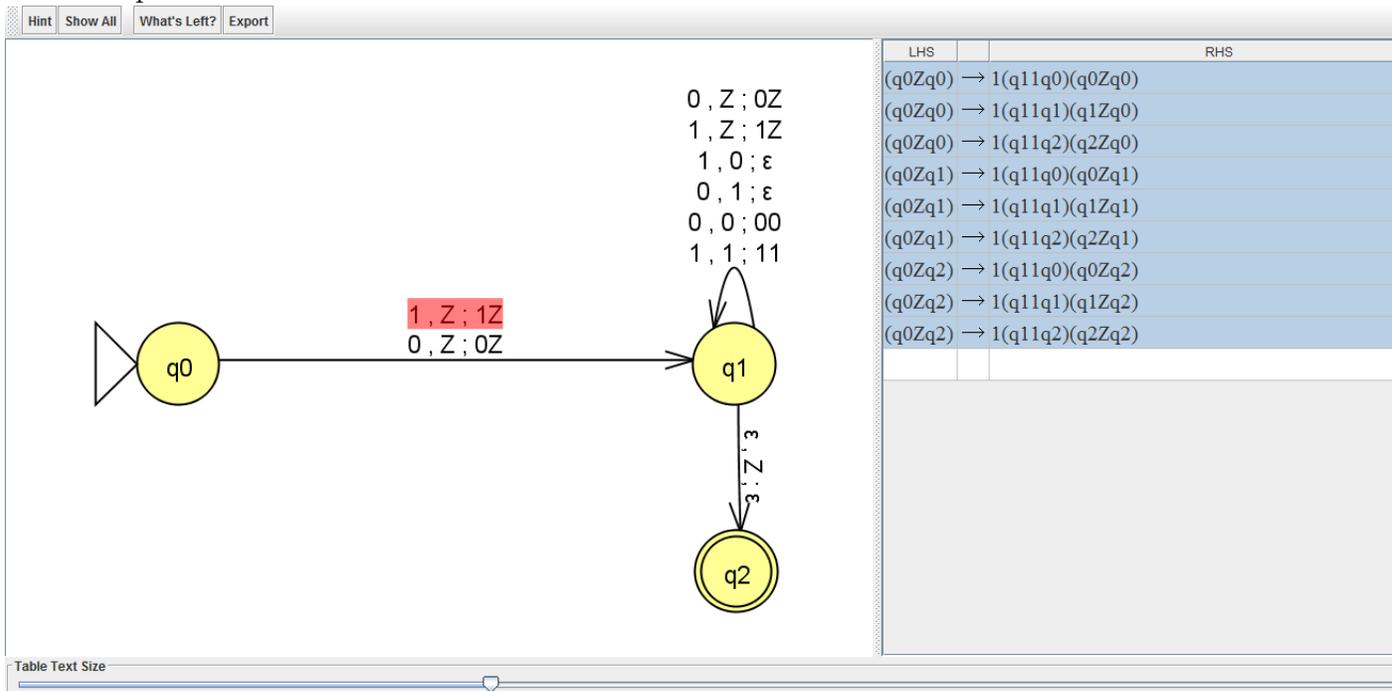
Similar to this if we click on the other transition that do a pop we end up with



When we click on the  $1, Z; 1Z$  transition the algorithm brute force computes all possible stack combinations. Since we know there is a state transition to  $q_1$  this means the following will be generated

$(q_0Zq_0) \rightarrow 1(q_{11}q_0)(q_0Zq_0)$   
 $(q_0Zq_0) \rightarrow 1(q_{11}q_1)(q_1Zq_0)$   
 $(q_0Zq_0) \rightarrow 1(q_{11}q_2)(q_2Zq_0)$   
 $(q_0Zq_0) \rightarrow 1(q_{11}q_0)(q_0Zq_1)$   
 $(q_0Zq_0) \rightarrow 1(q_{11}q_1)(q_1Zq_1)$   
 $(q_0Zq_0) \rightarrow 1(q_{11}q_2)(q_2Zq_1)$   
 $(q_0Zq_0) \rightarrow 1(q_{11}q_0)(q_0Zq_2)$   
 $(q_0Zq_0) \rightarrow 1(q_{11}q_1)(q_1Zq_2)$   
 $(q_0Zq_0) \rightarrow 1(q_{11}q_2)(q_2Zq_2)$

It is possible that not all of these are useful rules, but the brute force algorithm just considers all possible combinations.



Upon clicking the export button, these rules are trimmed. The trimming gets rid of all rules that cannot lead to a string of terminals. While it is tedious to discuss this for every single rule, we show a few examples of rules that are retained and rules that are not.

For instance  $(q_0Zq_2) \rightarrow 1(q_{11}q_1)(q_1Zq_2)$  is retained on account of the fact that the symbol  $(q_{11}q_1)$  has a rule that produces the single terminal 0 and  $(q_1Zq_2)$  can produce an  $\epsilon$ . So, beginning with  $(q_0Zq_2)$  it is possible to generate the string 10.

A rule that is not retained for instance is  $(q_1Zq_2) \rightarrow 0(q_{10}q_2)(q_0Zq_2)$ . Now we have already shown that  $(q_0Zq_2)$  is capable of generating a string. The question is whether  $q_{10}q_2$  can generate one. You should be able to go through the rules and convince yourself that it

does not.

Unfortunately this trimming and exporting of the rules currently crashes in JFLAP8, but JFLAP7 manages to do the exporting and trimming of the rules quite successfully.

The final step also involves removing some of the cumbersome notation and creating normal variables. The final solution is

Table Text Size		
LHS		RHS
S	→	1BK
S	→	0AK
A	→	1
B	→	0
K	→	$\epsilon$
K	→	0AK
K	→	1BK
A	→	0AA
B	→	1BB