Adrien Protzel

* 1. 1. JFLAP
     2. Written: There are 4 tapes, input on 1. Tapes 2-4 holds a, b, and c respectively. Q0 loops through tape 1 to the right, turning each letter into an x. Each letter is also pushed to their respective tapes 2-4 then moved right. Once the end of tape 1 is reached with a blank, it is stopped, all other tapes are moved left, and it continues to q1. Q1 loops all tapes 2-4 to the left while they see a letter and not a blank. If all inputs on all states are blank, it stops and moves to q2 and succeeds. This assumes that tape 1 has reached its end, tape 2-4 have equal number of letters and have ended all at blanks.
     3. Formal:

Q=(q0,q1,q2)

Σ=(a,b,c)

「=(1,2,3,4)

δ={na(w)=nb(w)=nc(w): w∊{a,b,c}\*}

S=q0

Blank=[]

F={q2}

* + - 1. (q0,a[][][])->(q0,xa[][],RRRR)
      2. (q0,b[][][])->(q0,x[]b[],RRRR)
      3. (q0,c[][][])->(q0,x[][]c,RRRR)
      4. (q0,[][][][])->(q1,[][][][],SLLL)
      5. (q1,[]abc)->(q1,[][][][],SLLL)
      6. (q1,[][][][])->(q2,[][][][],SSSS)
  1. O(n). It counts tape 1 linearly and uses tapes 2-4 as stacks to sort out the a,b,c. Once tape 1 is through, it continues for a number of times equal to the shortest tape length.
  2. 1. JFLAP
     2. Written: Starting at q0, with inputs a/b/c change to x, then right to q1/q2/q3 respectively. Loop through their own states q1/q2/q3, passing over x and a/b/c which got them to their states respectively. Each state q1/q2/q3 branches off into two states each, on inputs of either of two letters in the language, excluding the first letter that brought it to that current state. Changing the new letter to x and then moving right to q1->q4/q5, q2->q6/q7, q3->q8/q9. In each state, they have similar loops like q1/q2/q3 which pass over the previously recorded letter inputs and x, moving right. From each of the q4-9 a final input of a letter not recorded in ‘abc’ is taken, changed to x, moved left and then transitioned to q10. Q10 loops over x,a,b,c changing nothing and moving left until it reaches blank, changes nothing and moves right and moves to q0. Q0 then loops on input x, changing to blank and moving right, repeating q1-10 loop as needed. Finally moving to q16 on blank move right from q0.
     3. Formal:

Q=(q0,q1,q2,q3,q4,q5,q6,q7,q8,q9,q10,q16)

Σ=(a,b,c)

δ={na(w)=nb(w)=nc(w): w∊{a,b,c}\*}

S=q0

Blank=[]

F={q16}

* + - 1. (q0,x)->(q0,[],R)
      2. (q0,a)->(q1,x,R)
      3. (q0,b)->(q2,x,R)
      4. (q0,c)->(q3,x,R)
      5. (q1,a)->(q1,a,R)
      6. (q1,x)->(q1,x,R)
      7. (q2,b)->(q2,b,R)
      8. (q2,x)->(q2,x,R)
      9. (q3,c)->(q3,c,R)
      10. (q3,x)->(q3,x,R)
      11. (q1,b)->(q4,x,R)
      12. (q1,c)->(q5,x,R)
      13. (q2,a)->(q6,x,R)
      14. (q2,c)->(q7,x,R)
      15. (q3,a)->(q8,x,R)
      16. (q3,b)->(q9,x,R)
      17. (q4,xab)->(q4,xab,R)
      18. (q5,xac)->(q5,xac,R)
      19. (q6,xab)->(q6,xab,R)
      20. (q7,xbc)->(q7,xbc,R)
      21. (q8,xac)->(q8,xac,R)
      22. (q9,xbc)->(q9,xbc,R)
      23. (q4,c)->(q10,x,L)
      24. (q5,b)->(q10,x,L)
      25. (q6,c)->(q10,x,L)
      26. (q7,a)->(q10,x,L)
      27. (q8,b)->(q10,x,L)
      28. (q9,a)->(q10,x,L)
      29. (q10,xabc)->(q10,xabc,L)
      30. (q10,[])->(q0,[],R)
      31. (q0,[])->(q16,[],R)
  1. O(n^3). It uses a loop only when there are a certain number of n. The machine can then stop mid way if there aren't enough inputs, which also means the language fails as there aren't enough equal letters to continue the machine. It reads linearly and marks read letters as x but needs to loop back as it tries to read ‘abc’ as matches instead of counting out as a multitape machine.

1. 1. Written: It has an input and makes an output equal to the square of the input n. It will read in one input and output twice the number of items then moves right and to the next state.
   2. Formal:

Q=(q0,q1,q2,q3,q4,q5,q6,q7)

Σ=(1)

「=(1,[],x)

S=q0

Blank=[]

F={q7}

1. (q0,1)=(q1,x,R)
2. (q1,1)=(q2,x,R)
3. (q2,1)=(q3,x,R)
4. (q3,1)=(q4,1,R)
5. (q4,1)=(q5,1,R)
6. (q5,1)=(q6,[],L)
7. (q6,x)=(q7,x,S)