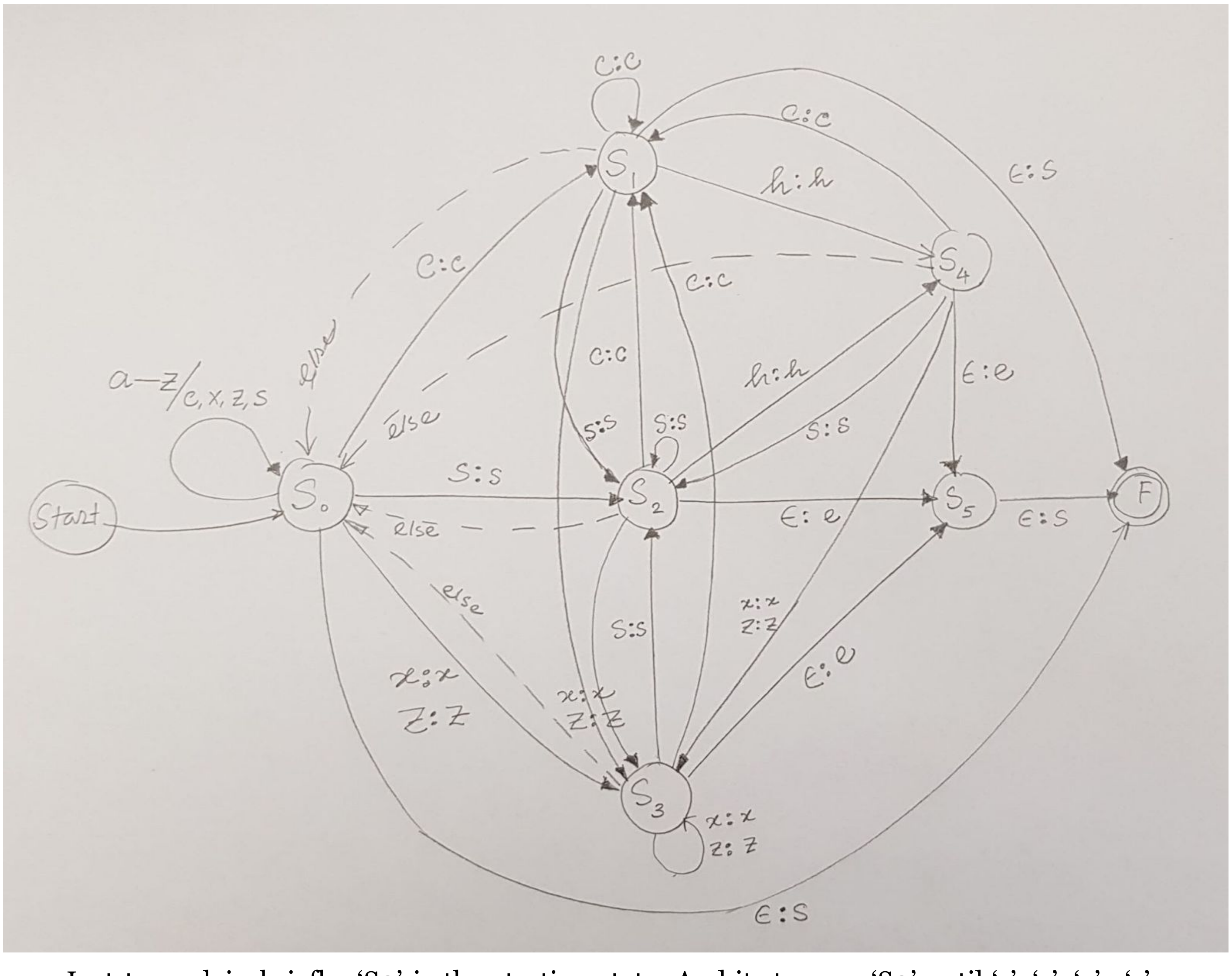
Report on Exercise 1: Designing FST for pluralization

Sanad Saha 933 620 612

Given pluralize.fst only added a single ‘s’ at the end of the words to make those plural. And pluralize2.fst contains Finite State Transducer which adds either ‘s’ or ‘es’, or both after a word; which is incorrect.

Because according to English Grammar we should add ‘-es’ to pluralize only when the word finish with the following letters: ‘s’, ‘x’, ‘z’, ‘sh’ and ‘ch’. So, in these 5 cases we should put ‘es’ at the ending of a word and for all other words we should just put ‘s’ to pluralize. So, I modified the FST which was provided in the pluralize2 and designed the following FST.



Just to explain briefly, ‘S0’ is the starting state. And it stays on ‘S0’ until ‘s’, ‘c’, ‘x’ , ‘z’ appears. I’ve tried to separate the cases where the word finishes with the letters which

would take them to ‘S1’, ‘S2’ and ‘S3’ to pluralize with ‘-es’ and if the word doesn’t finish with any of these cases; then it would go directly from ‘s0’ to the final state and add a ‘-s’ in the process.
If we find the letter ‘c’ or ‘s’ we become cautious [goto state s1, s2 respectively] if the next letter is ‘h’ and if that is also the last letter we can add ‘-es’. If the letter is ‘x’ or ‘z’ we move from s1 to s3;
On high level state 0 deals all letters but c, x, s, z. State 1 deals with ‘ch’, state 2 deals with ‘s’ and ‘sh’, state 3 deals with ‘x’ and ‘z’.
If there are more letters after ‘x’, ‘z’, ‘s’, ‘sh’, ‘ch’, then we come back to respective states or reset from state 0 accordingly which can be seen from the figure provided above.
After designing the FST it was programmed using carmel; which has been provided in ‘pluralize2.fst’.