Finite-State Transducers

The following problems relate to finite state transducers, a computational device or “machine” that is used a lot in computational linguistics. A simple example is shown below:

1.fst

The machine has a finite set of states (hence the name “finite state”) designated here with circles with numbers in them. There is a specified initial state; we will always note that state with the number “0”. And there are one or more final states, which are conventionally notated with a double instead of single circle. And there may be states that are neither initial nor final. So in the machine above, there are four states, one initial state (0) and one final state (3).

Between the states there are arcs, and these arcs are labeled with pairs of symbols from an alphabet. An alphabet can be any finite set of symbols; here we will just use letters. The symbol on the left of the “:” on the arc is the input symbol, and the one on the right is the output symbol.

The machine is called a transducer because it transduces strings of symbols — e.g. a word — into other strings of symbols. How that works is as follows. Let us say you have a string aab, and imagine I have a little pointer that points to where I am currently in that string; initially it will point to the beginning letter. You start in the initial state of the machine, and you ask: given the letter where my little pointer is pointing, is there any arc that matches that first letter on its input label? In this case the answer is “yes”: there is an arc labeled “a:a” that leaves that state, and goes to state 1. So I do three things:

• Output — from the arc’s output label — the symbol “a”
• Move the machine from state 0 to state 1
• Move my little pointer to the second letter of the string (so it now points at the second a.)
I then continue the process from state one and the second position of the string. In this case there is also an arc, labeled “a” on the input and “b” on the output, and my little input pointer is pointing at an “a”, so I:

- Output — from the arc’s output label — the symbol “b”
- Move the machine from state 1 to state 2
- Move my little pointer to the third letter of the string “b”

Now I’m looking at the “b” and I ask the same questions and by this point you should be able to see that I can:

- Output — from the arc’s output label — the symbol “b”
- Move the machine from state 2 to state 3
- Move my little pointer to the next position — which is the end of the string.

Now, notice I’ve reached the end of the string and I am in state 3, which is a final state. If, and only if, both conditions hold — I’ve used up my input and I am in a final state — then the machine has successfully read the input and successfully output a string. In this case it read aab and output abb.

If I had given it a string such as abb the machine above would fail to match the input, and hence would fail to give any output.

**Problem 1:** At which state will abb fail in 1 fst?

The machine we discussed is rather uninteresting since it only allows one input and output. But transducers can have more than one input and output. The following machine allows for a couple of input strings eat and baa, and three output strings:
Note that at state 4 there are two possible arcs to take on an input symbol a.

**Problem 2:** What are the output(s) for eat? What are the output(s) for baa in 2.fst?

Finite-state machines can have *loops* which are arcs or sequences of states and arcs that lead you back to a state that you’ve been to before. The following machine has nothing but loops: there is one state, which is both initial and final. You can read any string over the alphabet \{a, e, i, o, u, k, t, b\}. For example, you can read \(aeeeeeektlliuiuiuiu\) by simply reading a symbol, moving along the arc back to the initial state, and repeating the process. Once you’ve used up the string you will be in the initial state again, but since this is also a final state, the operation was successful.
For this machine nearly all of the arcs map symbols onto themselves, but there are some exceptions and some cases where a given input symbol might map to more than one possible output.

**Problem 3:** What are the output(s) for *kat* given the 3.fst? What are all the possible outputs for *kak*.

Here is a more complicated machine:
Problem 4: In 4.fst, what are the output(s) for:

- aabk
- aæk
- bæk
- kaek

Now that you are familiar with how FST’s work, you are ready to help us solve a problem related to a simple cypher that was developed by the late Quinneas E. Dogil and found among his papers. Actually what was left was a sample of text and a transducer that implements the cypher. The transducer seems to have been run word by word on the input text, to produce examples like the text below.

The text is as follows and the cypher FST is on the final page:

```
lipg mtbgf anc mykfn symgs akb hzg lrlfhs njbpxlh light hn qlas xinhafnh a oyq orhabn xinvfakf czn uafghw anc wycavmhfc qi qlf yjebushahb qhmh acj iyn ajf xjfmhfc kpmj
  oiq gy ajf knxmxfcz n a jifmhc xakaj gqyshahc glflshfg qlmh orhabn hj anw orhabn mi xinvfakfc anc mi wycavmhfc xrn uinx kncpfg gy ajf iyh hn a jifmh nhlijofaifc he qhmh grg gy frkf xitf qi wycavmhfh a yghabn he qhmh lafjc ad a lannmj lyshax ycnmf lig qhbs fgl fyghhfrq qfagh uakfas qhmh qhmh orhabn axlh uakf zb zd achbxflshfg lahanx anc yjbufg qhmh gy mlbpcj wj qlas
  nzf zn a urgxfg mnsf gy xrn oih wycavmhfh gy xrn oih xins-fvgmhf gy xrn oih frjfbq qlas jhbfnc qlf njmhf iyn uakanx anc wync glb mbgpxjfc fygh frkf xinsfvgmhf zb lrg abkfh hzg yibg yjfgi qi amc hj wyhgmvh qlf gijgc gaqj uahhij oihf oig uinxf tytftrfg qhmh gy mrw fygh nzf zb xrn oykflh ligxh qhmh qlfw wac fygh zb zd lig bd qlf uakanx thrflg qi ny wycavmhfhc qfagh qlf bnoanaslfq gijj glavli qhfw glb lipxlg fygh frkf qhps lrg mi oirjw ankmnnvfc zb zd thrflg lig bd qf ny fygh wycavmhfhc qi qlf jifmhc qpsi tytmnanx nyobgf bd qhmh jfbt qhfsf finbgfc wync gy qrif zngfmsfc
```
Unfortunately the transducer as drawn was rather messed up, and there were a bunch of missing input arc labels, indicated in the figure with hash marks.

Problem 5: What is the text and and what are the missing input labels on the arcs?