Turing Machine Format Explanation

Adam Yedidia

April 26, 2016

This document explains how to understand generated Turing machines, which can be found at:

`parsimony/src/tm/tm2/tm2_files`
or
`parsimony/src/tm/tm4/tm4_files`

depending on whether the machine's alphabet size is 2 or 4.

The following is an example of the file format of the single-tape, 4-symbol Turing machine output by the compiler.

Let us suppose that the Turing machine has three states. The four symbols in the Turing machine's alphabet are "_," "`1`," "`E`," and "textttH" The first state's name is "`StateOne`," the second state's name is "`StateTwo`," and the third state's name is "`StateThree`." The first tape's name is "`TapeA`," and the second tape's name is "`TapeB`."

`StateOne` is associated with `TapeA`. `StateTwo` is associated with `TapeB`. `StateThree` is associated with `TapeA`.

Each line defining the Turing machine's behavior in a given state is written as follows:

```
[symbol read] -> [next state]; [direction]; [symbol written]
```

In the actual Turing machine text file, [next state] is replaced by the state that the Turing machine would enter if it read the symbol [symbol read], [direction] is one of {`R, L, -`} and indicates the direction the head

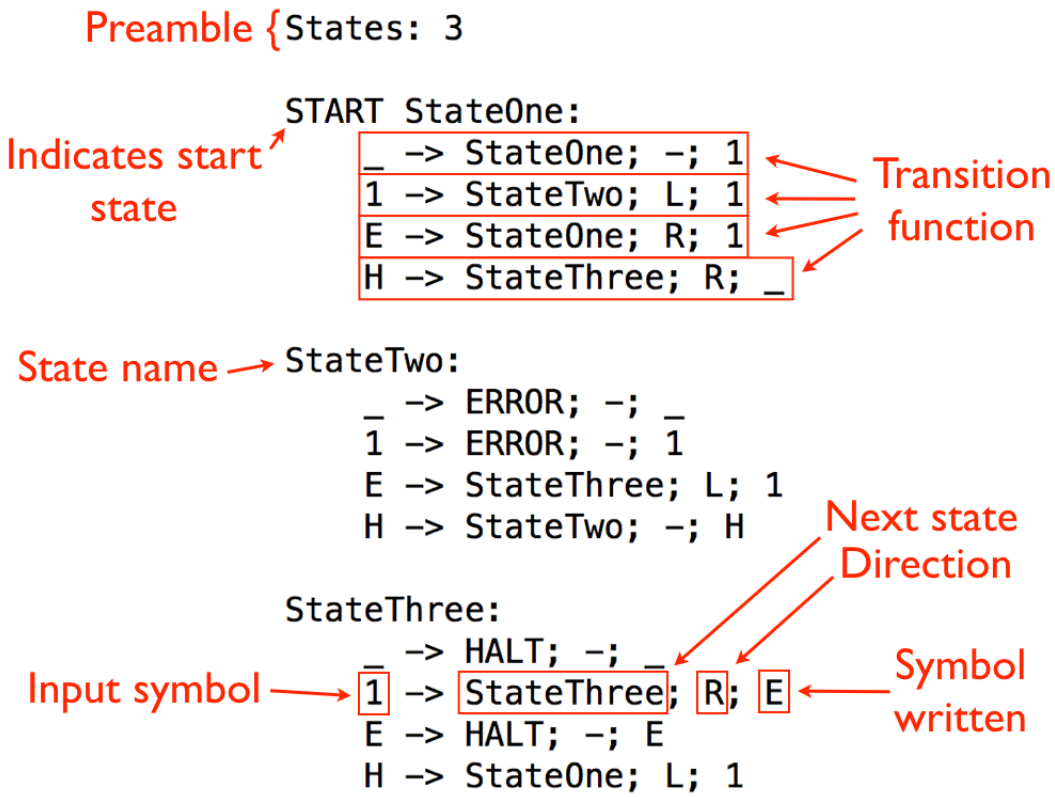

Figure 1: This is an example of a multi-tape Turing machine file. The notes in red are explanations of the meanings of the various parts of the Turing machine file.

would move in, and [symbol written] is one of {`_, 1, E`} and indicates the symbol that is written on the tape. Note that in the higher-level, multi-tape 3-symbol Turing machine, as indicated by the – symbol, the head can remain in place, (unlike the lowest-level Turing machine around which the Busy Beaver function is defined, where it cannot).

Figure 1 shows an example of such a Turing machine file.