Turing Machines

Nicholas Geis

February 5, 2015

Disclaimer: This portion of the notes does not talk about Cellular Automata or Dynamical Systems, it talks about turing machines, however this will lay the ground work for something else that I will write about in the coming weeks where we will look at Turing Completeness in the Game of Life[1]. So with that, let's begin.

Turing Machine

The first question to ask is, what is a turing machine? I will start with a simple explanation of a turing machine first before I go into a more rigorous definition. A turing machine, in essence, is a mathematically model of a machine that mechanically operates on a tape. It has four main components:

1. **Tape**: An arbitrarily long[2] piece of tape with cells placed one right after another. Each cell contains a symbol from some given finite alphabet which will always contain a blank state.

2. **Operating Head**: Now this is the machine itself. It can read and write or rewrite data on the tape and it can move one cell at a time, either to the left or to the right, on the tape.

3. **State Register**: The machine needs to be in a state in order to know what to do. This part tells the machine what state it is in from the finitely many states available to it.

4. **Table of Instructions**: These are the instructions that the machine follows according to the state it is in and the value of the cell it reads. The state of the head and the cell it reads will tell the machine to do the following in this particular sequence:

   (a) Either erase or write a symbol (replacing a_j with $a_{j,1}$), and then

   (b) Move the head either to the left or the right[3], and then

   (c) Assume the same or a new state as prescribed.

[1] We will not be doing this. Sorry for the disappointment.

[2] "Long" here is an infinite length tape with only a finite length string of information thus that string is preceded an succeeded by an infinite length tape containing only blank cells.

[3] Some newer concepts for Turing Machines include a neutral state where the head will not move at all.

For example, suppose the machine head is on a cell with a value 1. Its state is 23[4] . If the head is in state 23 and it reads a 1, then it will change that 1 to a 0, move left, and then change its state to 17. Each step follows the instructions specified in the state and then updates and continues from there until we have our desired output.

Now this is the very simple definition for one. With a loose understand of what the machine is now, lets give it a more rigorous definition. This definition goes as follows: a one tape Turing Machine can be formally defined as a 7-tuple[5],

$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, B, F \rangle$$

where M is our machine and the elements go as follows:

1. Q: The finite set of states of the finite control.
2. Σ: The finite set of input symbols.
3. Γ: The complex set of tape symbols, $\Sigma \subset \Gamma$.
4. δ: The transition function. The arguments of $\delta(q, X)$ are a state q and a tape symbol X. The value of $\delta(q, X)$, if it is defined, is a triple (p, Y, D) where:
   (a) p is the next state, in Q.
   (b) Y is the symbol, in Γ, written in the cell being scanned, replacing whatever symbol was there.
   (c) D is a direction, either L or R, and telling us the direction in which the head moves.
5. q_0: The initial starting state, and $q_0 \in Q$.
6. B: The blank symbol. This symbol is in Γ but not in Σ. The blank appears initially in all but the finite number of initial cells that hold input symbols.
7. F: The set of final or accepting states, and $F \subset Q$.

In the previous example, we had a $q_0 = 23$. The transition function took the input $\delta(23, 1)$ and outputted $(17, 0, L)$. From that we see that $23, 17 \in Q$ and that $B, 0, 1 \in \Gamma$, where B is our blank state. In the stated example, we never defined F and we did not define δ well either. We will go over a better example once I introduce a few more ideas and notations for Turing Machines.

[4] 23 is just an arbitrary number I chose.
[5] An n-tuple is a sequence, or ordered list, of n elements.

Instantaneous Description

Now you may be thinking that this is great and all, but how do we define an infinitely long tape with a finite portion of non-blank cells of information? The way to describe the initial tape is called an *instantaneous description*. Each instantaneous description (or ID) contains an infinitely long string of blank cells at the start and the end as previously stated. So to capture everything, we will start with the leftmost non-blank entry and end with rightmost non-blank entry. To describe the position of the head and the state of the head in this notation, we put the state, q_i, to the left of the symbol it happens to be on. To show a transition between each update of the machine we will use a $\vdash$. For example, we have that $\Sigma = \{X_1, X_2, ..., X_n\}$, $Q = \{q, p\}$, $q_0 = q$, and $\delta(q, X_i) = (p, Y, L)$, where $1 < i < n$.

$$X_1X_2 \dots X_{i-1}qX_iX_{i+1} \dots X_n \ \vdash \ X_1X_2 \dots pX_{i-1}YX_{i+1} \dots X_n.$$

That is the notation for describing the strings of symbols on the tape. With this we can create a transition diagram for the machine.

Transition Diagram

A *transition diagram* is similar to the phase space we created for SDS in the first week of this class. Basically, it will be a visual representation of how each head state interacts with each cell value as it goes from the starting state, q_0, to the final state, F. A more formal definition of a transition diagram goes as follows:

1. It consists of a set of nodes corresponding to the states of the Turing Machine.
2. There is an arc from state q to state p with $X\backslash Y \ D$, where X is the initial state of the cell, Y is the new state of the cell, and D is the direction the head moves and is pictorially represented with $\leftarrow$ and $\rightarrow$.
3. We represent the starting state with "start" and an arrow entering that state.
4. And we denote the accepting states as double circles.

The only information that one cannot directly read from a transition diagram is which are blank cells. In most cases, B will continue to represent that state when it is not stated. An example will definitely help anyone who is confused by this. I will take an example from *Introduction to Automata Theory, Languages, and Computation* by John Hopcroft, Rajeev Motwani, and Jeffrey Ullman[6].

[6]This book proved to be incredibly helpful in understanding the machines and this paper is mostly written entirely out of it. Chapter 8 in particular is great for turing machines. If anyone has more questions, definitely check this book for the answer. You can find a free copy on the internet.

Example

In this example we will be looking at a code starting with n 0's and followed by n 1's. And we will be changing those 0's to X's and the 1's to Y's. The process to do this goes as follows:

1. Start at the leftmost 0 and change it to an X.
2. Check the symbol to the right and if it is not a 1, keep moving to the right until you reach one.
3. Once you reach a 1, change it to a Y and then move back left until you hit an X.
4. Once you reach an X, keep it an X move right. If there is a 0, then change that 0 to an X and move right again.
5. Repeat steps 1-4 until all 0's are X's and 1's are Y's.

That is the loose description of the process. However, we want to run this in a turing machine so it must follow the structure of the machine. We will start by defining all the components to the machine.

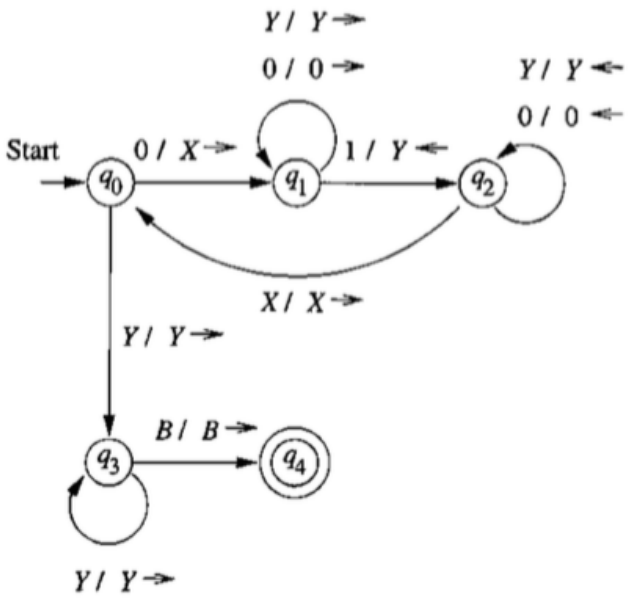
1. Q: $\{q_0, q_1, q_2, q_3, q_4\}$
2. Σ: $\{0, 1\}$
3. Γ: $\{0, 1, X, Y, B\}$
4. δ: The transition function is described by the following table:

State \ Symbol	0	1	X	Y	B
q_0	(q_1, X, R)	-	-	(q_3, Y, R)	-
q_1	$(q_1, 0, R)$	(q_2, Y, L)	-	(q_1, Y, R)	-
q_2	$(q_2, 0, L)$	-	(q_0, X, R)	(q_2, Y, L)	-
q_3	-	-	-	(q_3, Y, R)	(q_4, B, R)
q_4	-	-	-	-	-

   Remember that δ inputs both head state and the symbol in the cell it reads and outputs a new state and symbol and moves to either the left or right.

5. q_0 is the start state.
6. B is the blank symbol.
7. F: a_4 is the final or accepting state.

That is each component of a turing machine and now lets look at what the transition diagram looks like for this particular example. It looks like this



This probably looks a bit complicated but it truly is not too bad. I will break it down step by step with an example code, lets say 0011. So for all transition diagrams we start at "start." So we start at the leftmost symbol and we are in state q_0. The ID looks as follows

$$q_0 0011.$$

Now, we run one step. So the transition diagram says that if the head reads a 0 and is in q_0 it will change the symbol to an X, change to state q_1, and move right. That happens, so we get

$$q_0 0011 \vdash X q_1 011.$$

We continue to follow the diagram and its instructions. So the next instruction is if the head is in state q_1 and it reads 0 then it keeps the zero and remains in the same state except it moves to the right. Thus it will look like

$$X q_1 011 \vdash X 0 q_1 11.$$

We can continue to follow the instructions given from the transition diagram and it will eventually complete what we want it to do. This is the complete sequence it will go through

$$q_0 0011 \vdash X q_1 011 \vdash X 0 q_1 11 \vdash X q_2 0 Y 1 \vdash q_2 X 0 Y 1 \vdash X q_0 0 Y 1 \vdash X X q_1 Y 1$$

$$X X q_1 Y 1 \vdash X X Y q_1 1 \vdash X X q_2 Y Y \vdash X q_2 X Y Y \vdash X X q_0 Y Y \vdash X X Y q_3 Y \vdash X X Y Y q_3$$

At that last point we reach a blank cell so then it will run it in one last time and then stop. So from our input of 0011 we get an output of $XXYY$ which is the output we wanted from the description above.

References

[1] Hopcroft, John, Rajeev Motwani, and Jeffrey Ullman. *Introduction to Automata Theory, Languages, and Computation.* Stanford: Pearson Eduction, 2001. Web

[2] http://en.wikipedia.org/wiki/Turing_machine. Wikipedia, January 29, 2015.