4CS4-6: Theory of Computation (Universal Turing Machines)

Lecture 21: April, 08, 2020

Prof. K.R. Chowdhary *: Professor of CS*

21.1 Universal Turing Machine

Alan M. Turing described a *Universal Turing Machine* (UTM), as a single ordinary Turing machine that could perform all operations performed by an ordinary Turing machine, even in case the ordinary machine was more complicated than this Universal machine. In other words, the UTM imitates the operations of an ordinary TM. This ability to make a relatively simple machine to act like a more complicated machine (the universal TM) is achieved by giving the complicated instructions to the simple machine. In particular, a UTM is provided on one part of its tape a complete symbolic description of a machine it is expected to imitate. Then, the UTM stores on another part of its tape (for instance on alternate squares) a copy of the tape that would be on the imitated machine, and makes the changes on this part of the tape that the imitated machine would make. The remaining part of the tape must be used for intermediate scratch work, for instance, to record what state the imitated machine is in. The internal structure of the UTM has to include instructions to use the various kinds of data, and to move back and forth between the different parts of its tape.

Since the method of storing all this information on one tape is rather complicated, the internal structure of the UTM is also rather complicated, requiring a large number of states. Therefore, a simplification could be to use all this information on multi-tapes, which requires lesser complicated instructions, but this alternative machine has no any capability of higher computing power in terms of what can be computable on a TM.

Let the ordinary Turing machine is defined as 6-tuple, $M = (Q, \Sigma, \Gamma, \delta, s, H)$, which is a special purpose computer, designed to solve a particular problem. It is possible to design a reprogramable Turing machine called Universal Turing Machine M'. Given an input description of any ordinary Turing machine M and a string w, the UTM M' can simulate the computation of M. To construct such a UTM, we first choose a standard way of describing Turing machine without any loss of generality. We assume that,

$$Q = \{q_1, q_2, ..., q_n\} \text{ and}$$
$$\Gamma = \{a_1, a_2, ..., a_m\}.$$

where, q_1 is start state, and q_n is single accepting state, and a_1 is representing blank character. Now select an encoding of states and symbols such that q_1 is 1, q_2 is 11 etc, and $a_1 = 1$, $a_2 = 11$ and so on. The symbol 0 will be used as separator among the fields on the UTM. Since, start state, final states, and other states are specified, Turing machine M can be described completely by the details of its transition function δ. The transition function δ can also be encoded according to this scheme with arguments and the result in some prescribed sequence. For example, a transition $\delta(q_1, a_3) = (q_4, a_2, R)$ may appear as follows.

$$...1011101111011010...,$$

where, 1 is for q_1, 111 is for a_3, 1111 is for q_4, 11 is for a_2, and 1 is for R. All these are separated by 0s. Having this, any Turing machine can be encoded using finite encoding on $\{0, 1\}^+$. In addition, given a finite encoding of Turing machine, it can be decoded also.

Apart from this description of M, a UTM has an input alphabet sequence string also in $\{0, 1\}^+$. To simplify the structure of information on the tape of a UTM, we use a Multi-tape machine as UTM, as shown in Fig. 21.1.

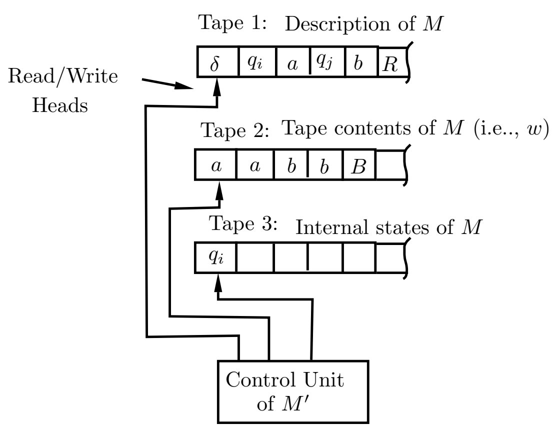

Figure 21.1: Universal Turing machine to simulate TM as M

For any input string w and an ordinary TM as input, $M = (Q, \Sigma, \Gamma, \delta, s, H)$ the corresponding UTM M' will comprise following parts.

1. Tape-1 will contain encoded definition of M,
2. Tape-2 will contain the original input w and modified tape contents (Γ^*) of M, and
3. Tape-3 will contain the state(s) of M.

The UTM M' looks at current state q on tape-3 and current current symbol a tape-2, to determine the configuration of M. Then looks for corresponding $\delta(q, a)$ on tape-1 to determine the result (q', b, L) or (q', b, R) from tape-1. Based on this result, the current symbol on tape-2 is modified to b, tape makes a move to R or L, and the state is modified simultaneously on tape-3 to q'. This process is repeated until there is halting state reached on tape-3.

Finally, a 3-tape TM can always be simulated using single tape, where different areas are selected on the tape to represent the contents of tape-1 and tape-2 and tape-3, finite control will cause the transitions as per those performed on 3-tape TM. In fact, there will be need of virtual head positions in these three areas on single tape, which corresponds to three heads of 3-tape TM.

A UTM can be designed to simulate the computations of an arbitrary TM M. To do so, input to UTM must contain representation of the machine M and input w to be processed by M. The Fig. 21.2 shows such a universal Turing machine.

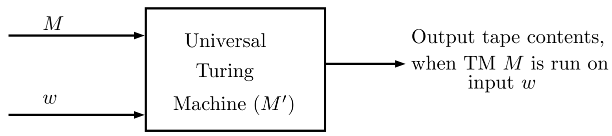

Figure 21.2: Universal Turing machine block diagram

Let there is TM M that accepts by halting. The UTM M' for this is: Input string = $R(M)w$, where $R(M)$ is representation of M. The UTM has two outputs.

1. Output-1: *Accept* (indicates that M halts with input w),
2. Output-2: *loops*, i.e., M does not halt with input w, i.e. computation of M' does not terminate.

The machine M' is called universal TM, as computation of any Turing machine can be simulated by M' (See Fig. 21.3).

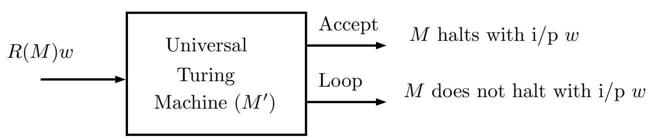

Figure 21.3: Universal Turing machine with two outputs

21.1.1 Representation of a TM

Because of the ability to encode arbitrary symbols as strings over $\{0, 1\}$, we consider Turing machine with inputs $\Sigma = \{0, 1\}$ and tape symbols $\Gamma = \{0, 1, B\}$. The states of M are assumed to be $\{q_0, q_1, \ldots, q_n\}$. The TM M is defined by its transition function: $\delta(q_i, a) = (q_j, b, d)$; $q_i, q_j \in Q$; $a, b \in \Gamma$; and $d \in \{L, R\}$.

We can have encoding for a Turing machine M as follows for alphabets, states, transition function, etc.

Let $en(z)$ denote the encoding of z. Hence, Transition $\delta(q_i, a) = (q_j, b, d)$ is encoded by string:

$$en(q_i)0en(a)0en(q_j)0en(b)0en(d).$$

The symbol *0* separates the components of δ.

Symbol	Encoding
0	1
1	11
B	111
q_0	1
q_1	11
...	...
q_n	1^{n+1}
L	1
R	11

A representation of machine M is constructed from encoded transitions. Two consecutive 0s separate one transition from other, e.g., $\delta(q,a) = (q', b, R)$ and $\delta(q', a) = (q'', b, L)$. Beginning and end of complete representation are defined by three 0s. Consider the following Transitions for the Turing machine M:

Transition	*Encoding*
$\delta(q_0, B) = (q_1, B, R)$	101110110111011
$\delta(q_1, 0) = (q_0, 0, L)$	1101010101
$\delta(q_1, 1) = (q_2, 1, R)$	110110111011011
$\delta(q_2, 1) = (q_0, 1, L)$	1110110101101

The machine M is represented by encoded string as follows:

000101110110111011 00110101010100 11011011101101100
1110110101101000.

21.1.2 Simulation of Universal TM on 3-tape TM

Tape-1 holds $R(M)w$, tape-3 simulates computations of of M for input w, and Tape-2 is used as working tape. Here, $R(M)$ is representation of TM M, discussed earlier, called encoding of TM M. If input w is not of the form $R(M)w$ for deterministic TM M and string w on tape-1, the M' moves to right forever.

The working of 3-tape machine, which acts as UTM is as follows: first w is copied from tape-1 to tape-3, with tape head at begin of w. So, the tape-3 is initial configuration of M with input w. Encoding of q_0, i.e., 1 is written to tape-2, to indicate that machine M is in state q_0. For future steps, let next state is q_j. The steps to be followed for each transition are as follows:

1. Transition of M is simulated on tape-3. The transition is determined by symbol scanned on tape-3 and state encoded on tape-2. Let these are a and q_i.
2. Tape-1, which holds the representation M, is scanned for a and q_i as first two components of a transition. If not found, M' halts by rejecting input.
3. If tape-1 consists the encoded information for above, i.e., $\delta(q_i, a) = (q_j, b, R)$, then,

i. q_i replaced by q_j on tape-2.

ii. b is written on tape 3, and tape head on tape-3 is moved for direction given in R.

Go back to step 1, and carry on computation by simulating M.