Theory of Formal Languages (Sequential Operations of Turing Machines)

Lecture 10: March. 23/27, 2021

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

10.1 Sequential Operation of Turing Machines

A Turing machine is designed to accomplish a single task, and it corresponds to an algorithm. Such Turing machines can be combined to operate in sequential order in sufficient numbers to perform complex operations. In such cases, the output produced by a Turing machine on the tape becomes input for next Turing machine. For example, two Turing machines Z and S can be operated sequentially, one after other, so that whatever is the current tape contents, it will be made zero by Z Turing machine, followed to this, it is incremented by 1 by the S Turing machine. In this situation, the final state of the first machine (Z) becomes the initial state of S, as shown in Fig. 10.1, and its symbolic representation is shown in Fig. 10.2. It is necessary that input is provided in proper format, else it would be garbage-in-garbage-out situation.

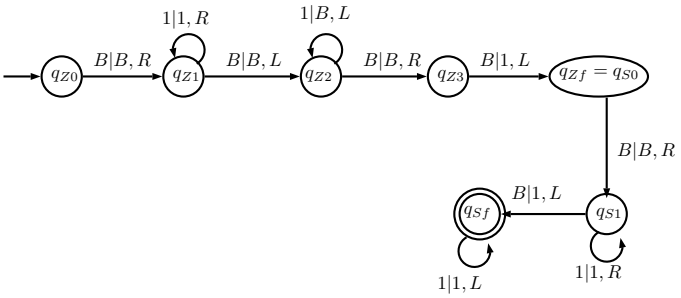

Figure 10.1: Sequential operation of TMs Z and S

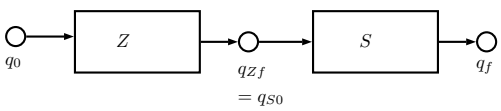

Figure 10.2: Symbolic representation of operation of TMs Z and S in sequence

The TMs Z and S can be used as components of more complex Turing machines. The component machines are called Macros – the concept used in assembly and C languages. Following are some of the commonly used macros [sudkamta07].

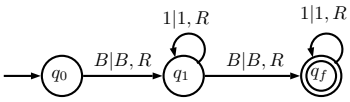

Figure 10.3: TM for MR_2

MR_i (Move Right) Macro This macro moves the tape head right through i consecutive natural numbers on the tape[1]. For example, MR_2 macro is defined by the Turing machines shown in Fig. 10.3, where tape-head is advanced by two numbers (arguments) on the tape, such that it is positioned at the begin of 3rd argument.

A move macro does not effect the tape contents to the left of initial position of tape head. The initial and final configuration for MR_2 are as follows:

Initial configuration: $B\bar{n_1}Bq_0\bar{n_2}B\bar{n_3}B\bar{n_4}B$

Final configuration: $B\bar{n_1}B\bar{n_2}B\bar{n_3}Bq_f\bar{n_4}B$

For a macro, it is necessary that proper input is provided, e.g., for MR_2 it is necessary that at least two natural numbers exists after the tape head. A macro never effects the contents outside the bounds of that macro. Henceforth, we will indicate tape head position by underscore, (e.g., $\underline{B}$ or $\underline{n_i}$).

ML_i (Move left) Macro The ML_i macro is defined in the similar way as MR_i, except that it moves the tape head to left. The initial and final configurations of ML_2 macro are as follows.

Initial configuration: $B\bar{n_1}B\bar{n_2}B\bar{n_3}B\bar{n_4}\underline{B}$

Final configuration: $B\bar{n_1}B\bar{n_2}\underline{B}\bar{n_3}B\bar{n_4}B$

FR and FL (Find right and find left) Macros The find right macro moves the tape-head into a position to process the first natural number in right position. The Following is an example of find right macro.

Initial configuration: $\underline{B}B^i\bar{n}B \quad i \geq 0$

[1]A natural number on tape is sequence of 1s separated from both sides by Bs.

Final configuration: $B^i\underline{B}\bar{n}B$

Note that in the initial configuration, there are B^i blank symbols between head position and the natural number at right, after head moves right to the first natural number, it points to the blank position just before the natural number. Similarly, find left macro moves the tape-head to find the first left natural number from head position.

Initial configuration: $B\bar{n}B^i\underline{B} \quad i \geq 0$

Final configuration: $\underline{B}\bar{n}B^iB$

E_k **(Erase) Macro** The erase macro erases a sequence of k natural numbers, and halts with tape head in original position.

Initial configuration: $\underline{B}\bar{n_1}B\bar{n_2}B...B\bar{n_k}B \quad k \geq 1$

Final configuration: $\underline{B}B.....B$

$CPY_{k,i}$**(Copy through** i **numbers) Macro** The copy macro produces a copy of specified number of integers, in the area of tape that is assumed to be blank. The macro $CPY_{k,i}$ expects total $k+i$ integers on the tape, and when it is executed, it copies first k numbers of integers following to original $k+i$ integers, making total $2k+i$ integers when operation is computed. For example, $CPY_{3,2}$ copies first three numbers following to 5th position, making total 8 numbers, and $CPY_{3,0}$ copies first three numbers at the position after 3rd number, making total six numbers. Following is the configuration before and after $CPY_{k,i}$ macro.

Initial: $\underline{B}\bar{n_1}B\bar{n_2}B...B\bar{n_k}B\bar{n}_{k+1}B...B\bar{n}_{k+i}BB...B \quad k \geq 1$

Final: $\underline{B}\bar{n_1}B\bar{n_2}B...B\bar{n_k}B\bar{n}_{k+1}B...B\bar{n}_{k+i}B\bar{n_1}B\bar{n_2}B...B\bar{n_k}B$

T **(Translate) macro** The translate macro changes the location of the first natural number so that it is to the right of tape head position. The computation is terminated with the tape head in the position it occupied in the beginning, with translated string (natural number) to its immediate right.

Initial Configuration: $\underline{B}B^i\bar{n}B \quad i \geq 0$

Final Configuration: $\underline{B}\bar{n}B^iB$

A (Add) Macro The Add macro adds the two consecutive naturals after the tape head, and replaces these by a single number equal to sum of these two. The initial and final configurations of the Turing machines are:

Initial Configuration: $\underline{B}\bar{m}B\bar{n}B$

Final Configuration: $\underline{B}\overline{m+n}B$

In the following example, we illustrate the implementation of *swap* operation for two integers using the macros discussed above.

Example 10.1 *Construct a sequence of macros to exchange two integer numbers m and n available on the tape of TM.*

The initial and final configurations are as given below.

Initial configuration: $\underline{B}\bar{m}B\bar{n}B$

Final configuration: $\underline{B}\bar{n}B\bar{m}B$

The Fig. 10.4 show the six macros sequentially performed to swap the two numbers m and n.

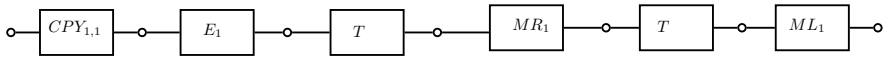

Figure 10.4: Sequential operations of Turing machines to swap two numbers

The change in configuration after each macro is performed, to swap two numbers, are shown below.

Initial configuration: $\underline{B}\bar{m}B\bar{n}B$

Configuration after $CPY_{1,1}$: $\underline{B}\bar{m}B\bar{n}B\bar{m}B$

Configuration after E_1: $\underline{B}B^mB\bar{n}B\bar{m}B$

Configuration after T: $\underline{B}\bar{n}B^mBB\bar{m}B$

Configuration after MR_1: $B\bar{n}\underline{B}B^mB\bar{m}B$

Configuration after T: $B\bar{n}\underline{B}\bar{m}B^mBB$

Configuration after ML_1: $\underline{B}\bar{n}B\bar{m}B^mBBB$

We note that m and n are swapped, and tape-head points to blank symbol just before n.□

Example 10.2 *Construct sequential TMs to multiply a natural number n by 2.*

Initial configuration: $\underline{B}\bar{n}B$

Configuration after $CPY_{1,0}$: $\underline{B}\bar{n}B\bar{n}B$

Configuration after (Add) A: $\underline{B}\overline{n+n}B$ (This is final configuration)

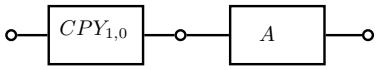

Figure 10.5: Sequential operations of Turing machines to multiply a number by 2

The Fig. 10.5 shows the construction of TM to add two numbers.□