CSE211-Formal Languages and Automata Theory

U3L7 – Programming Techniques for Turing Machine

Dr. P. Saravanan
School of Computing
SASTRA Deemed University

Agenda

- Showing how a TM computes.
- Indicating that TM's are as powerful as conventional computers.
- Even some extended TM's can be simulated by the original TM

Programming Techniques for TM’s

SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 OF THE UGC ACT, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

8.3.1 Storage in the State

– **Technique**:

use the finite control of a TM to hold a finite amount of data, in addition to the state (which represents a position in a TM “program”).

– Method:

think of the state as [*q*, *A*, *B*, *C*], for example, when think of the finite control to hold three data elements *A*, *B*, and *C*. See the figure in the next page (Figure 8.13)

Programming Techniques for TM's

Figure 8.13

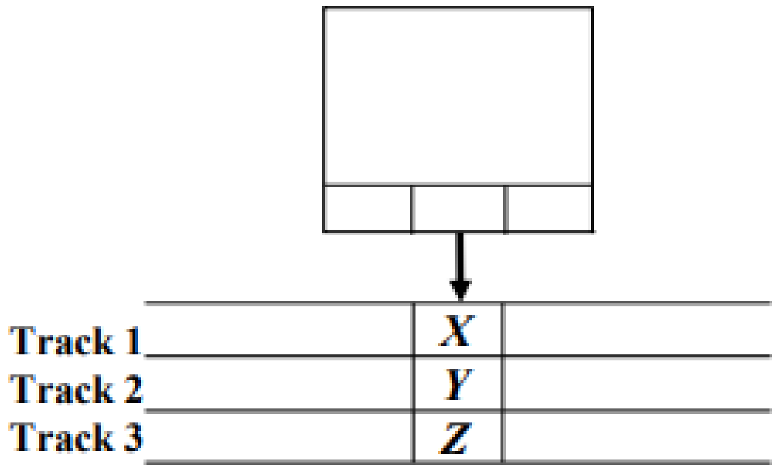


Figure 8.13. A TM viewed as having finite control storage and multiple tracks.

Programming Techniques for TM's

8.3.1 Storage in the State

– **Example 8.6 (cont'd):**

The transition function δ is as follows.

- $\delta([q_0, B], a) = ([q_1, a], a, R)$ for $a = 0, 1$. --- *Copying the symbol it scanned.*
- $\delta([q_1, a], \overline{a}) = ([q_1, a], \overline{a}, R)$ where $\overline{a}$ is the complement of $a = 0, 1$. --- *Skipping symbols which are complements of the 1st symbol read (stored in the state as a).*

SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 OF THE UGC ACT, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

8.3.2 Multiple Tracks

- We may think the tape of a TM as composed of several tracks.
- For example, if there are three tracks, we may use the tape symbol [*X*, *Y*, *Z*] (like that in Figure 8.13).
- **Example 8.7** --- see the textbook. The TM recognizes the non-CFL language

  $$L = \{wcw \mid w \text{ is in } (\mathbf{0} + \mathbf{1})^{+}\}.$$

- Why does not the power of the TM increase in this way?

  Answer: just a kind of *tape symbol labeling*.

Programming Techniques for TM's

- Subroutines
  - The concept of subroutine may also be implemented for a TM.
- Example 8.8 --- design a TM to perform multiplication on the tape in a way of transformation as follows:

  $0^m10^n1 \Rightarrow 0^{mn}$

Example 8.8: We shall design a TM to implement the function "multiplication." That is, our TM will start with 0^m10^n1 on its tape, and will end with 0^{mn} on the tape. An outline of the strategy is:

1. The tape will, in general, have one nonblank string of the form $0^i10^n10^{kn}$ for some k.

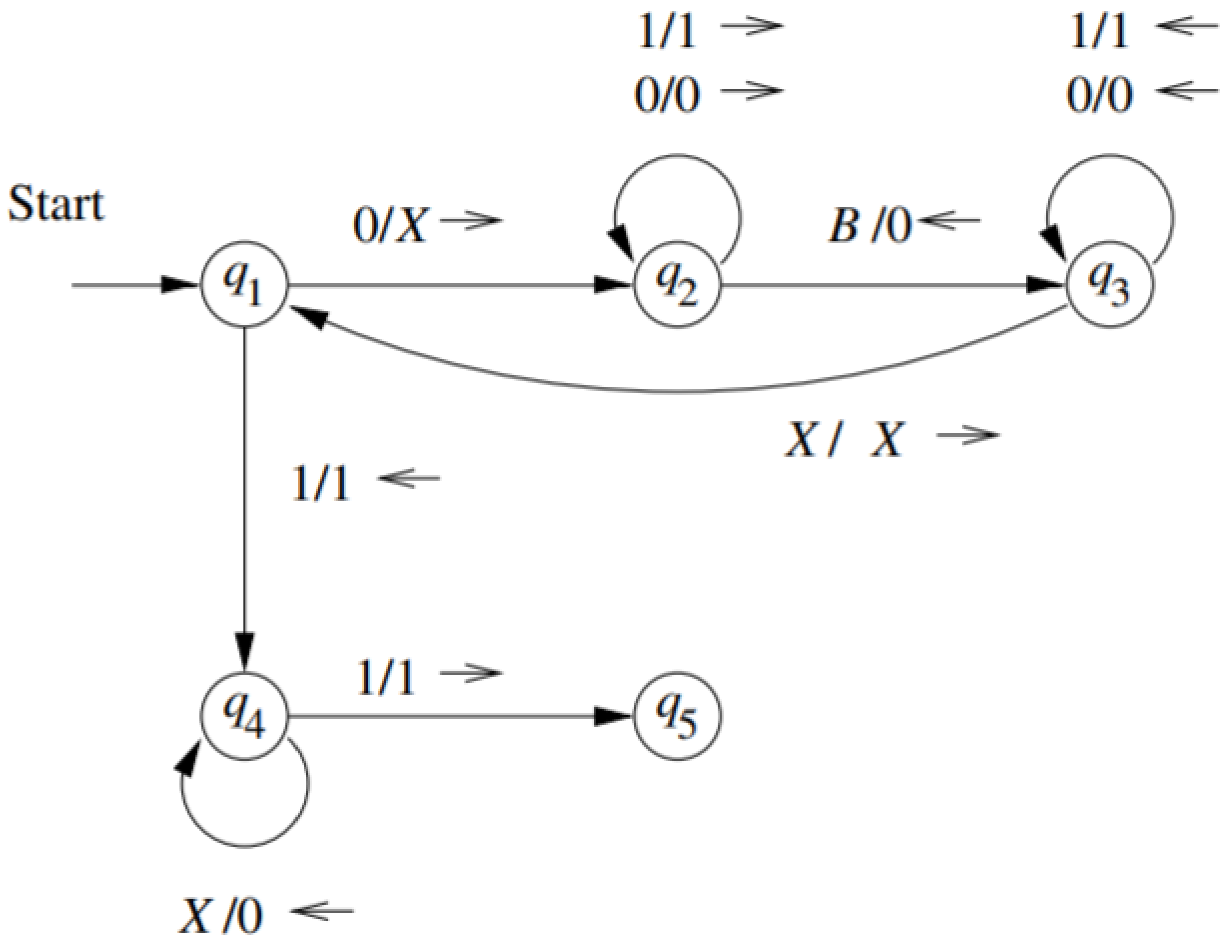
2. In one basic step, we change a 0 in the first group to B and add n 0's to the last group, giving us a string of the form $0^{i-1}10^n10^{(k+1)n}$.

3. As a result, we copy the group of n 0's to the end m times, once each time we change a 0 in the first group to B. When the first group of 0's is completely changed to blanks, there will be mn 0's in the last group.

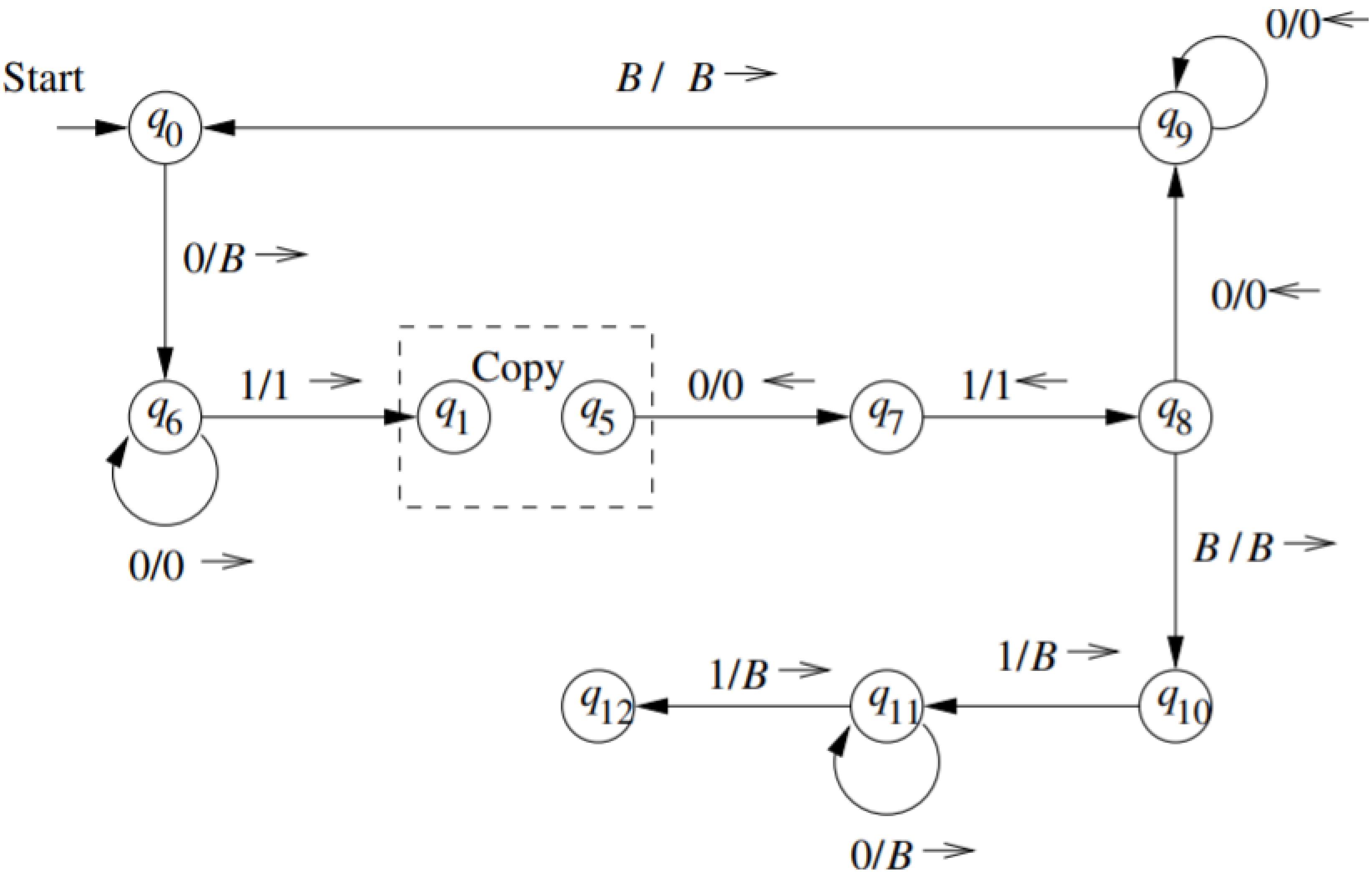
4. The final step is to change the leading 10^n1 to blanks, and we are done.

- The heart of this algorithm is a subroutine which we call Copy This sub routine helps implement step (2) above copying the block of n 0’s to the end More precisely Copy converts an ID of the form $0^{m-k}1q_5 0^n 1 0^{kn}$ to ID $0^{m-k}1q_1 0^n 1 0^{(k-1)n}$

Subroutine copy



The complete multiplication



References

- John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman, *Introduction to Automata Theory*, Languages, and Computation, Pearson, 3rd Edition, 2011.
- Peter Linz, An Introduction to Formal Languages and Automata, Jones and Bartle Learning International, United Kingdom, 6th Edition, 2016.

Next Class: **Unit III**

Extensions to the Basic TM

Thank you.