AENSI Journals

Australian Journal of Basic and Applied Sciences

ISSN:1991-8178

Journal home page: www.ajbasweb.com

A Universal Turing Machine Design Approach For English To Tamil Language Conversion

[1]A.Maheshwari and [2]M.A. DoraiRangaswamy

[1]*Research Scholar, Sathyabama University, Chennai, India.*
[2]*Professor, AVIT University, Chennai, India.*

ARTICLE INFO

Article history:
Received 19 August 2014
Received in revised form
19 September 2014
Accepted 29 November 2014
Available online 15 December 2014

Keywords:
Tamil words; Delta rule; FSA; MATLAB; Transitions; UTM

ABSTRACT

Turing Machines are the most powerful computational machines. Turing machines are equivalent to algorithms, and are the theoretical basis for modern computers. Still it is a tedious task to create and maintain Turing Machines for all the problems. The Universal Turing Machine (UTM) is a solution to this problem. A UTM simulates any other TM, thus providing a single model and solution for all the computational problems. A universal Turing machine is one which can accept the description of another Turing machine as input and simulate the operation of that Turing machine. The creation of UTM is very tedious because we need to devise encoding scheme for Tamil language to serve all TM's. Also many of the existing tools do not support the creation of UTM which makes the task very difficult to accomplish. Hence a Universal Turing Machine is developed using MATLAB platform to recognize Tamil language. In this paper, MATLAB has been used for implementation and found most successful and widely used tool for visualization and simulation.

© 2014 AENSI Publisher All rights reserved.

To Cite This Article: A.Maheshwari and M. A. DoraiRangaswamy, A Universal Turing Machine Design Approach For English To Tamil Language Conversion**.** ***Aust. J. Basic & Appl. Sci.,*** *8(18): 572-577, 2014*

INTRODUCTION

Turing Machine as a machine with a finite number of control states and an infinite tape, bounded at the left and stretching off to the right. The tape is divided into cells, each of which can hold one symbol. The input of the machine is a string $w=w_1w_2\ldots w_n$, followed by an infinite sequence of blanks B. TM can read and write symbols on the tape as it pleases. Transition function of TM is

δ (qi , a) = (qj , b, R)

Turing Machine M computes a function f if, when the given input w is in the domain of f, the machine halts in its accept state, with f(w) written on the tape. For example, add two numbers, i.e., f(m,n) = m+n, then the numbers m and n are to be placed on the tape as 0^m10^n, where 1 is a separator for the numbers m and n. Once processing is completed and the TM halts then the tape would have the contents as $0^{(m+n).}$

II. Existing system:

Turing Machines are the most powerful computational machines. Turing machines are equivalent to algorithms, and are the theoretical basis for modern computers. Still it is a tedious task to create and maintain Turing Machines for all the problems. The Universal Turing Machine (UTM) or simply a universal machine is a solution to this problem.

III. Proposed system:

In proposed system to Develop a Universal Turing Machine for implementing the concept of more symbols in the input alphabet as well as the tape alphabet. It is very useful of rural people. TM can be used as a language recognizer. Turing Machine recognizes all the languages, CFL, CSL, and Type 0. A string W is written on the tape, with blanks filling out the unused portions. The machine enters a initial state q0, with the read write head position on the left most symbol of W. If after a sequence of moves, the Turing Machine enters the final state and halts, then w is considered to be accepted. A language is accepted by Turing Machine is known as Recursively enumerable language (RE), being enumerable means, TM precisely list all the strings of that language. Although TM may precisely list all the strings of RE (Type 0) language, it may not be able to decide every such language. Deciding a language requires that TM halt on a final state for every w.

Corresponding Author: A.Maheshwari, Research Scholar, Sathyabama University, Chennai, India
E-mail: 78mahee@gmail.com

IV. Turing machine model:

The Turing machine is a simple and powerful mathematical model of computation that can be used as language acceptor, language translator, and function evaluator. The languages that the Turing machine can compute are recursively enumerable.

The graphical model of the Universal Turing machine is shown in Fig 1.

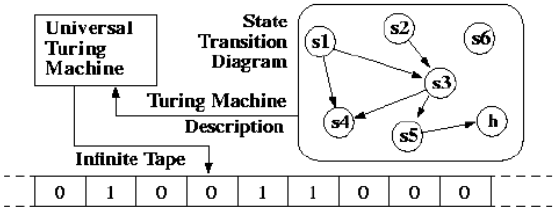


Fig. 1: Turing Machine Model.

The model consists of an input output relation that the machine computes. The input is given in binary form on the machine's tape, and the output consists of the contents of the tape when the machine halts.

At every step, the current state and the character read on the tape determine the next state the FSM will be in, the character that the machine will output on the tape (possibly the one read, leaving the contents unchanged), and which direction the head moves in, left or right.

The problem with Turing Machines is that a different one must be constructed for every new computation to be performed, for every input output relation.

This is why we introduce the notion of a universal turing machine (UTM), which along with the input on the tape, takes in the description of a machine M. The UTM can go on then to simulate M on the rest of the contents of the input tape. A universal turing machine can thus simulate any other machine.

V. Literature review:

(Alan Rosen and David Rosen, 2007) Turing machines over sets with atoms, also known as nominal sets. Our main result is that deterministic machines are weaker than nondeterministic ones; in particular, P6=NP in sets with atoms.

(Eric Gramond and Susan H.Rodger, 1999) Display the running process of any user-specified Turing machine by loading a Turing Machine specification file. The data structure that records the specification is initialize when the specification file is loaded, during which the check for grammar soundness is performed. After the user then input the string into the text field at the bottom of the window and "run", the Turing Machine start the recognizing process to determine whether the input can be accepted by the Turing Machine.

In (Jos C. M. Baeten, 2011) the authors have proposed reactive Turing machines (RTMs), extending classical Turing machines with a process-theoretical notion of interaction, and use it to define a notion of executable transition system.

In (Jainendra Singh, 2013) the implementation of the unrestricted grammar in to recursively enumerable language for JFLAP platform. Automata play a major role in compiler design and parsing. The class of formal languages that work for the most complex problems belongs to the set of Recursively Enumerable Language (REL).RELs are accepted by the type of automata as Turing Machine.

In (Maheshwari.A and Dr.M.A. Doraiswamy, 2011) Automata play a major role in compiler design and parsing. Turing Machines are the most powerful computational machines. Turing machines are equivalent to algorithms, and are the theoretical basis for modern computers. Still it is a tedious task to create and maintain Turing Machines for all the problems. The Universal Turing Machine (UTM) or simply a universal machine is a solution to this problem.

In (Sumitha C.H, 2011) an automaton is a mathematical model for a finite state machine (FSM). A FSM is a machine that has a set of input symbols and transitions and jumps through a series of states according to a transition function. Automata play a major role in compiler design and parsing. Turing Machines are the most powerful computational machines. They possess an infinite memory in the form of a tape, and a head which can read and change the tape, and move in either direction along the tape or remain stationary. Turing Machines are equivalent to algorithms, and are the theoretical basis for modern computers.

In (Tirtharaj Dash, 2012) analyze mathematically a possibility that the Universal Quantum Turing Machine (UQTM) is able to compute faster than any other classical models of computation. Basically we focused on comparative study on computation power of Universal Turing Machine (UTM) and UQTM. Namely, in the equal, we tried to show that the UQTM can solve any NP-complete problem in polynomial time. The result analysis showed that UQTM is faster for any computation.

Australian Journal of Basic and Applied Sciences, 8(18) December 2014, Pages: 572-577

VI. *Comparison With Previous Models:*

A finite automaton (FA) is a simple idealized machine used to recognize patterns within input taken from some character set (or alphabet) C. The job of an FA is to *accept* or *reject* an input depending on whether the pattern defined by the FA occurs in the input. A finite automaton consists of:

- a finite set S of N states
- a special start state
- a set of final (or accepting) states
- a set of transitions T from one state to another, labeled with chars in C Pushdown automata differ from finite state machines in two ways:
- Pushdown automata choose a transition by indexing a table by input signal, current state, and the symbol at the top of the stack. This means that those three parameters completely determine the transition path that is chosen. Finite state machines just look at the input signal and the current state: they have no stack to work with. Pushdown automata add the stack as a parameter for choice.
- Pushdown automata can also manipulate the stack, as part of performing a transition. Finite state machines choose a new state, the result of following the transition. The manipulation can be to push a particular symbol to the top of the stack, or to pop off the top of the stack. The automaton can alternatively ignore the stack, and leave it as it is. The choice of manipulation (or no manipulation) is determined by the transition table.

Table 1: Comparison with previous models.

Device	Separate Input?	Read/Write Data Structure	Deterministic by default?
FA	*Yes*	*None*	*Yes*
PDA	*Yes*	*LIFO Stack*	*No*
TM	*No*	*1-way infinite tape. One cell access per step.*	*Yes (but will also allow crashes)*

In TM Initially, the input, which is a finite-length string of symbols chosen from the input alphabet, is placed on the tape. All other tape cells, extending infinitely to the left and right, initially hold a special symbol called the blank. The blank is a tape symbol but not an input symbol, and there may be other tape symbols besides the input symbols and the blank, as well. There is a tape head symbol scanned. In one move, the Turing Machine will work as following steps.

a. The next state optionally may be the same as the current state.

b. Write a tape symbol in the cell scanned. This tape symbol replaces whatever symbol was in that cell. Optionally, the symbol written may be the same as the symbol currently there.

c. Move the tape head left or right. In our formalism we require a move, and do not allow the head to head to remain stationary. This restriction does not constrain what a Turing Machine(TM) can compute, since any sequence of moves with a stationary head could be condensed, along with the next tape head could be condensed, along with the next tape Head move, into a single state change, a new tape symbol, and a move left or right.

VII. *Universal Turing Machine:*

- Universal Turing Machine M_u is an automaton. Input of UTM is an description of any Turing Machine M and a string w. UTM can simulate the computation of M for the input w. If there is a transition

$$\delta (q_i , a_j) = (q_k , a_l, R)$$

- Then the binary representation for the transition is given as $0^i1o^j10^k10^l10^m$
- The binary code for the Turing Machine M which has transitions $t_1,t_2,\ldots..,t_n$ is represented as

111 t_1 11 t_2 11 t_3 11 ……11 t_n 111

- For example:

Let M = ({q_1,q_2,q_3,q_4,q_5}, { வா, ங், க, c, o, m, e}, { வா, ங், க, c, o, m, e, B}, δ ,q_1, B, {q_5})

- Moves defined as

$\delta(q_1,$ வா$) = (q_2, c, R)$

$\delta(q_2,$ ங்$) = (q_3, o, R)$

$\delta(q_3,$ க$) = (q_4, m, R)$

$\delta(q_4, B) = (q_5, e, S)$

- Solution:

1. Let the binary representation for the states {q_1,q_2,q_3,q_4,q_5} be {0,00,000,0000,00000}
2. The alphabet { வா, ங், க, c, o, m, e, B} be {0, 00, 000, 0000, 00000, 0^6, 0^7, 0^8}
3. The direction {L, R, S} be {0, 00, 000}
4. The transitions are represented as

Transitions Binary Representation

$\delta(q_1, வா) = (q_2, c, R)$ 01010010^4100 →A
$\delta(q_2, ங்) = (q_3, o, R)$ 00100100010^5100 →B
$\delta(q_3, க) = (q_4, m, R)$ 000100010^410^6100 →C
$\delta(q_4, B) = (q_5, e, S)$ $000010^810^510^71000$ →D

5. The problem instance < M, வாங்க > is represented as 11 A 11 B 11 C 11 D 111 வாங்க

VIII. *Organization Of Universal Turing Machine:*

A UTM simulates any other TM, thus providing a single model and solution for all the computational problems. A UTM TU is an automaton that, given as input the description of any Turing Machine TM and a string w, can simulate the computation of M on w. It reduces the memory usage when compared to using multiple TMs.

The transition function is the core part of a UTM. The UTM works on the basis of the rules defined in it. The transition function δ is defined as

$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ (1)

The UTM is represented as

$TU = (Q, \sum, \Gamma, \delta, q0, B, F)$ (2)

Where,
Q is the set of all internal states,
∑ is the input alphabet,
Γ is a finite set of symbols called the tape alphabet,
δ is the transition function,
q0 is the initial state
B ε Γ is a special symbol called the blank,
F ε Q is the set of all final states.

A. Working Of Utm:

The Universal Turing Machine has a control unit and three tapes.

- Tape1 will keep an encoded definition (Transitions) of M.
- Tape 2 will contain the tape contents of M.
- Tape 3 contains the internal state of M.
- The behavior of the M is as follows:

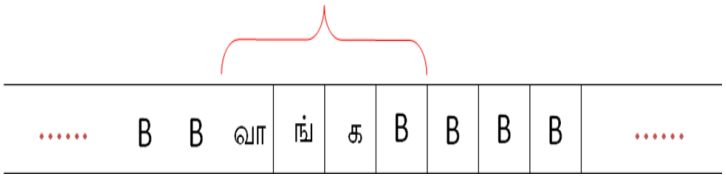
(i) Check the format of Tape 1.
(ii) Initialize Tape 2 to contain w.
(iii) Initialize Tape 3 to hold a single 0 representing the initial state q_1.
(iv) When Tape 3 holds 05, then it is said to reach the final state and the machine can halt.
(v) Let 'வா' be the symbol currently scanned by Tapehead 2.
(vi) Let 0 be the contents of Tape 3 which indicates the state.
(vii) Scan Tape 1 from the left end, looking for a string 0101
(a) if no such string is found then halt and reject.
(b) if the string is found, then let the suffix be $0^210^410^211$. Put 0^2 on Tape 3, print c on Tape 2 and move head in the right direction.

- M_u accepts <M, w> iff M accepts w.
- It is also true that if M runs forever on w then M_u runs forever on <M, w> and If M halts on w without accepting, Mu also halts on w without accepting.

RESULTS AND DISCUSSION

For a deterministic Turing machine with m symbols in the alphabet such that | ∑ | = m and total number of states n, m X n transitions are possible.

Input String:



Output String:

Australian Journal of Basic and Applied Sciences, 8(18) December 2014, Pages: 572-577

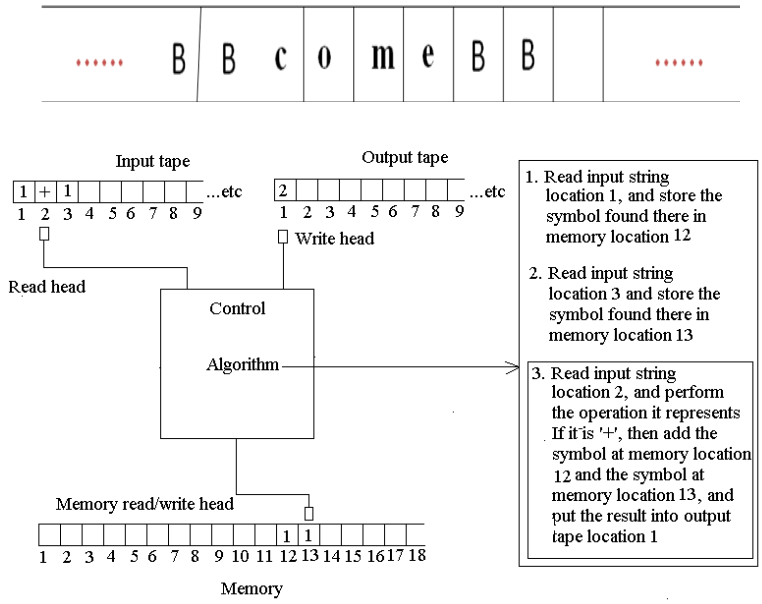


Fig. 2: Overview of translation process.

The overview of translation process is shown in Fig2. A UTM with n states, | ∑ | = m and p possible directions branches into m X n X p states for execution. A problem that can be solved with a multi tape turing machine with m tapes in O (n) moves can be done with a UTM in O(nm) moves.

X. *Experimental Setup:*

Machine translation is a key technology in language processing that can play a lead role in information access and communication. Building translation system from and to Tamil helps the Tamil community not only in Tamil nadu but also countries like Singapore, Malaysia and Srilanka, where Thamizh is one of the many official languages used. All over the world in accessing the information in Tamil.

Sentence forms that can be translated by

S = NP + (PP)n +VP

(eg.) A waiter waits on the guests.

NP = (adj)n + N

(eg.) His dress is very neat.

NP = NP 'wh' S

(eg) Where did you go?

PP = prep + NP

(eg.) The cat is in.

VP = V + adv

(eg.) Run fast.

S1 that S2

(eg.) He is so weak so that he lost his attendance.

If S1 , S2

(eg.) If he is regular, then he will get good marks.

Tokenization is splitting the words from a given English sentence. Parsing is obtaining morphemes. Morphemes are the parts of the meaning of a word. There are two types of morphemes.

(i) Lexical morpheme

(ii) Grammatical morpheme

Lexical morpheme or Root morpheme is of four groups namely Nominal, Verbal, Adjectival and Adverbial. After obtaining the root words from parser output, the equivalent Thamizh word is obtained from the dictionary. Then the tense, case morphs are retrieved from the tables according to the parser output. The actions that should be carried out are

1. Copies the input from input tape.
2. Positions head 1 at the beginning of the string in input tape.
3. At a time only one character is recognized.
4. All Tamil words are stored in the memory tape.
5. Convert the Tamil words to the corresponding English words.

Australian Journal of Basic and Applied Sciences, 8(18) December 2014, Pages: 572-577

Conclusion And Future Works:

Turing machines are the most powerful computational machines. A Universal Turing Machine simulates any other Turing Machine, thus providing a single model and solution for all the computational problems. The Turing Machines provide an abstract model to all the problems. This paper describes the working of a Turing Machine as well as a Universal Turing Machine for Context Free Languages.

The Turing Machines differ from all other automata as it can work with recursively Enumerable Languages. The language $a^nb^nc^n$ is a recursively enumerable language which cannot be implemented using a PDA but can be done using a T.M. This requires more storage than for Context Free Languages and hence the Turing Machines with the infinite tapes, extendable in both directions are used for this.

The future work includes:

i. Developing a Universal Turing Machine for South Indian Languages.

ii. Implement the concept of universality by including more symbols in the input alphabet as well as the tape alphabet.

REFERENCES

Alan Rosen and David Rosen, 2007. The Turing Machine Revisited, *MCon Inc*.

Eric Gramond and Susan H. Rodger, 1999. Using JFLAP to interact with theorems in automata theory. *ACM Portal Proc. in SIGCSE,* 336-340.

Jos, C.M., Baeten, Bas Luttik and Paul van Tilburg, 2011. Reactive Turing Machines, Springer-Verlag Berlin Heidelberg.

Jainendra Singh, Dr. S.K. Saxena, 2013. Implementation of Unrestricted Grammar in To the Recursively Enumerable Language Using Turing Machine, the International Journal of Engineering and Science, 3: 56-59, ISSN: 2319 – 1813 ISBN, 2319-1805.

Maheshwari, A., M.A. DoraiRangaswamy, 2010. Implementation of Context free languages in Universal Turing Machines, in the National Conference on Control, Communication and Information Technology NCCCIT of SRM Easwari Engineering College on 8th – 10th December

Maheshwari, A., M.A. DoraiRangaswamy, 2011. Implementation of Context free languages Using Universal Turing Machines, in the National Conference on Cloud and Scientific Computing organized by Dr.MGR University on 28th March.

Maheshwari, A., M.A. DoraiRangaswamy, 2011. Implementation of Context free languages in Universal Turing Machines., in the International Conference on Smart Technologies for Materials, Communication, Controls, Computing & Energy ICST organized by VEL TECH University on 5^{th} – 7^{th} January.

Maheshwari, A., M.A. DoraiRangaswamy, 2011.Implementation of Context free languages in Universal Turing Machines in the International Conference on Electronics Computer Technology ICECT, organized by IEEE, Kanyakumari on 8^{th} – 10^{th} April.

Maheshwari, A., M.A. DoraiRangaswamy, 2012. Universal turing machine implementation, in the international conference on advances in electrical/ electronics, information, communication and bio-informatics organized by prathyusha institute of technology and management on 24^{th} – 25^{th} january.

Maheshwari, A. and Dr. M.A. Doraiswamy, 2011. Implementation of Context Free Languages in Universal Turing Machines in the CIIT International Journal of Automation and Autonomous System, April 2011. Print: ISSN 0974 – 9659 & Online: ISSN, 0974-9551**.**

Sumitha, C.H., Member, ICMLC and Krupa Ophelia Geddam, 2011. Implementation of Recursively Enumerable Languages in Universal Turing Machine, International Journal of Computer Theory and Engineering, 3: 1793-8201.

Tirtharaj Dash, Tanistha Nayak, 2012. Quantum Finite Automata: a Language Acceptor Model, International Journal of Advanced Research in Computer Science and Software Engineering, 9: ISSN, 2277-128X.