Fourth International Derive TI-89/92 Conference
Liverpool John Moores University, July 12 – 15, 2000

Some Applications of Post and Turing Machines in Mathematics Teaching

Nelson Urrego
Pontificia Universidad Javeriana, Columbia
Email: nurrego@javeriana.edu.co

Abstract

This paper tries to give a presentation for non-experts about the concept of Turing Machine and Post Machine, and some examples of algorithms that can be represented by means of these models of computation will be given. In addition, we want to emphasize the historical importance of Turing Machines as well as its use like pedagogical tool in Arithmetic teaching.

Introduction

Towards principles of century XX a particular interest arose to solve problems that involved the algorithm notion. For instance, the Hilbert's tenth problem [3], [4] for finding a mechanical procedure in order to determine whether or not a polynomial Diophantine equation with integer coefficients has a solution in integers; or the well known *Decision Problem* [6] (Entscheidungsproblem) for discovering a method in order to establish the truth or falsity of any statement in the predicate calculus.

Even the term *algorithm* was introduced in Middle Age to define a mechanical and finite procedure for solving a particular problem, this concept (algorithm) required a theoretical criteria in order to determine several rules by which such procedures were governed. This task was not simple and the algorithm concept couldn't receive a mathematical definition because of its intuitive sense. However in 1936 the English mathematician Alan Turing [6] and the logician Emil Post [5] independently introduce the theoretical tools needed to give a precision to the algorithm concept and contributed to the sprouting of the Computing Theory. These tools are called Turing Machines and Post Machines.

Particularly, Turing showed how the intuitive concept of algorithm can be detailed by a model of the computing process where any algorithm is divided in a sequence of simple and primitive steps. Such computing model was used for attacking the *Decision Problem* and show that this was impossible to solve.

Turing Machines

Alan Turing considers that every algorithm can be thought as an abstract device (Turing Machine). This device consists of a tape divided in cells and, in principle, can be extended as much as it is required towards the left or the right. Also consists of an alphabet of symbols $\Sigma = \{S_0, S_1,\ldots,S_n\}$ such that every cell contains one and only one of these symbols. In addition we have a finite number of internal states $q_1, q_2, \ldots, q_m$. Finally a reading head is in capacity to scan a cell and to carry out one of the following operations:

(1) Move one cell to the right
(2) Move one cell to the left
(3) Halt the computing
(4) Write S_i in place of whatever is in the scanned cell $(1 \le i \le n)$

The *program* of instructions that the reading head is to follow can be specified in different ways: by a set of quadruples (Table 1) or by a flow graph (Figure 1) as it is indicated bellow:

State	Scanned Symbol	Action	Next State	Description action
q_j	S_i	L	q_k	Move to the left and go to state q_k
q_j	S_i	R	q_k	Move to the right and go to state q_k
q_j	S_i	S_j	q_k	Print symbol S_j in place of symbol S_i and go to state q_k

Table 1. Quadruples in Turing Machines

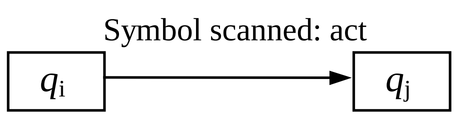


Figure 1. Flow graph. The reading head goes from state q_i to state q_j.

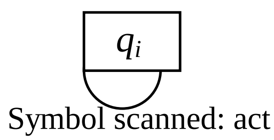


Figure 2. Flow graph. The reading head goes from state q_i to state q_j.

Example 1

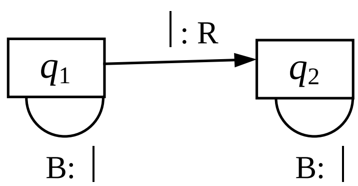
The Turing Machine **M** writes two symbols $|$.

Set of quadruples for machine **M**

$q_1\mathrm{B}\,|\,q_1$ $\qquad$ $q_2\mathrm{B}\,|\,q_2$

$q_1\,|\,\mathrm{R}q_2$ $\qquad$ $q_2\,|\,|\,q_2$

Flow graph for Machine **M**



An integer n can be represented by an unbroken string of n+1 symbols $|$ on an otherwise blank tape. The n-uple $\langle x_1,\ldots,x_n\rangle$ can be represented by n unbroken strings on an otherwise blank tape and separated one by one by a blank cell.

Example 2

The number 3 may be represented on the tape as follows:

...			|	|	|	|					...

Example 3

The expression $\langle 0,2,1\rangle$ can be represented on the tape as follows:

	⇩								
. . .	|		|	|	|		|	|	. . .

Here the arrow represents the reading head.

Post Machines

A Post Machine [8] is a particular case of a Turing Machine. The difference is based fundamentally in the notation and the fact that the alphabet of a Post Machine contains only one symbol. A Post Machine consists of a potentially infinite tape divided in cells. Each cell can be empty or contain a symbol $|$. There is a finite control or reading head that scans a cell and it is capable to write a symbol $|$ if the cell is empty or to erase a symbol otherwise.

The reading head is capable to carry out different operations, which follow the next notation:

1) i. $\rightarrow$ j
   Instruction i, the reading head must to move one cell to the right and go to instruction j.

2) i. $\leftarrow$ j
   Instruction i, the reading head must to move to the left and go to instruction j.

3) i. β j
   Instruction i, the reading head erases the symbol $|$ and go to instruction j.

4) i. $|$ j
   Instruction i, the head reading prints the symbol $|$ and go to instruction j.

5) i. ? $\begin{array}{l} \nearrow j_1 \\ \searrow j_2 \end{array}$

   Instruction i, if the scanned cell is empty, the reading head go to instruction j_1. On the other hand, if the scanned cell contains a symbol $|$, the reading head go to instruction j_2.

6) i. Stop.
   Instruction i, the Machine stops.

A nonempty set of instructions in order to solve a given problem is called a *program* for a Post Machine. Natural numbers can be represented on the tape in the same way as in Turing Machines, as well as the n-tuples.

Computable Functions in Turing and Post Machines

An arithmetic function *f* is said to be *computable* in a Turing Machine or *Turing computable* if there is a Turing Machine **M** that computes *f*.

Bellow we give examples of *Turing computable* functions. Here the reading head starts scanning the leftmost symbol.

Example 4

The successor function: $s(x) = x + 1$ is Turing Computable. The machine $\mathbf{M_1}$ computes $s(x)$

$\mathbf{M_1}$

$q_0 \mid L\, q_0$
$q_0\, B \mid q_1$

Example 5

The predecessor function:

$$\mathrm{Pd}(x) = \begin{cases} x-1 \text{ if } x > 0 \\ 0 \quad\;\; \text{if } x = 0 \end{cases}$$

is Turing Computable. The machine $\mathbf{M_2}$ computes $\mathrm{Pd}(x)$:

$\mathbf{M_2}$

$q_0 \mid B\, q_0$
$q_0\, B\; D\, q_1$
$q_1 \mid \; \mid q_0$
$q_1\, B \mid q_2$

Example 6

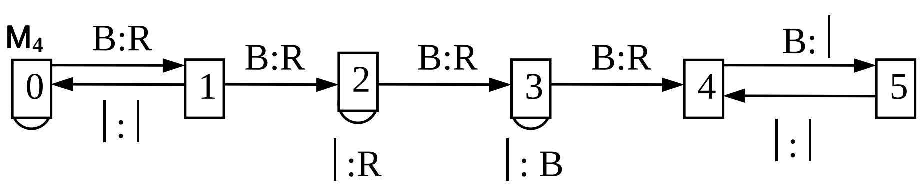
The function $\left[\frac{x}{2}\right]$, the largest integer $\leq \frac{x}{2}$ is Turing computable. The machine $\mathbf{M_3}$ computes it:

$\mathbf{M_3}$

$q_0 \mid S_2\; q_1$	$q_2\, S_2 \mid q_5$	$q_0\, B\;\, L\, q_5$
$q_1\, S_2\;\, R\, q_1$	$q_2\, B\;\, L\;\, q_3$	
$q_1 \mid \;\, R\;\, q_1$	$q_3 \mid \;\, L\;\, q_3$	
$q_1\, B\;\;\, L\;\, q_2$	$q_3\, S_2 \mid q_4$	
$q_2 \mid \;\, B\, q_2$	$q_4 \mid \;\, R\, q_0$	

Example 7

The function $I^3_2(x_1, x_2, x_3) = x_2$ can be computed by the machine $\mathbf{M_4}$:



$|$: B

Example 8

The function sum(x,y) =$x + y$ can be computed by the machine M_5 :

M_5

q_1 $\vert$ R q_1	q_3 $\vert$ B q_3	q_5 $\vert$ L q_5
q_1 B $\vert$ q_2	q_3 B L q_4	q_5 B R q_6
q_2 $\vert$ R q_2	q_4 $\vert$ B q_4	
q_2 B L q_3	q_4 S_0 I q_5	

An arithmetic function f is called *computable* in a Post machine if there is a program for a Post machine that computes f.

The examples 11-15 illustrate some Computable functions in Post Machines with their correspond programs. Here the reading head starts scanning the leftmost symbol.

Example 11

The successor function: s(x) = x + 1

$1\ ?\begin{array}{l}\nearrow 3\\ \searrow 2\end{array}$ 3. $|$ 4

2. $\leftarrow$ 1 4. Stop.

Example 12

The zero function Z(x) = 0

1. $\rightarrow$ 2 4. $\leftarrow$ 5

2. $?\begin{array}{l}\nearrow 4\\ \searrow 3\end{array}$ 5. $?\begin{array}{l}\nearrow 4\\ \searrow 6\end{array}$

3. β 1 6. Stop.

Example 13

The function $I_2^3(x_1, x_2, x_3) = x_2$

1. β 2 4. $\rightarrow$ 5 7. β 8 10. $\leftarrow$ 11 13. $?\begin{array}{l}\nearrow 14\\ \searrow 12\end{array}$

2. → 3	5. ? $\begin{matrix}\nearrow 6\\ \searrow 4\end{matrix}$	8. → 9	11. ? $\begin{matrix}\nearrow 12\\ \searrow 10\end{matrix}$	14. → 15
3. ? $\begin{matrix}\nearrow 4\\ \searrow 1\end{matrix}$	6. → 7	9. ? $\begin{matrix}\nearrow 10\\ \searrow 7\end{matrix}$	12. ← 13	15. Stop.

Example 14

The Function $Sg(x) = \begin{cases} 1 \text{ si } x>0 \\ 0 \text{ si } x=0 \end{cases}$

1. → 2	4. ? $\begin{matrix}\nearrow 6\\ \searrow 5\end{matrix}$	7.? $\begin{matrix}\nearrow 6\\ \searrow 8\end{matrix}$
2. ? $\begin{matrix}\nearrow 8\\ \searrow 3\end{matrix}$	5. β 3	8. ← 9
3. → 4	6. ← 7	9. Stop.

Example 15

The function sum(*x*,*y*) =*x* + *y*

1. → 2	4. → 5	7. β 8	10. ← 11	13.Stop.
2. ? $\begin{matrix}\nearrow 3\\ \searrow 1\end{matrix}$	5. ? $\begin{matrix}\nearrow 6\\ \searrow 4\end{matrix}$	8. ← 9	11. ? $\begin{matrix}\nearrow 12\\ \searrow 10\end{matrix}$	
3. $\mid$ 4	6. ← 7	9. β 10	12. → 13	

Pedagogical tools in Post Machines and Turing Machines

Possibly, the interesting thing, from the point of view of Mathematics Didactic, is that the students from early levels can devise each time more efficient algorithms for solving arithmetic problems or representing more complex arithmetic functions.

This allows to students manipulate some primitive arithmetic concepts in order to devise mechanical procedures in Turing Machines and Post Machines and at the same time they are becoming familiar with theoretical computing models. The problem of designing a Program for a Post Machine that represents the quotient between two integers was presented to a group of first semester mathematics

students. After working during one week, these students devised a procedure that represents the well-known *division algorithm* in a Post Machine. The interesting part is that they do not know such procedure and some how they "re-discovered" the *division algorithm*[1].
At this point, the implementation of computing simulator is very important. They allow to visualize the transition of the reading head for complex algorithms, and they simplify the assimilating of abstract computing models such as those described in this paper.

In the Internet we can find different of these simulators, for instance, TURING MACHINE SIMULATOR (Professor John Kennedy, Santa Monica College, Figure 4) or VisualTuring (Christian Chiran, Figure 3). Until the moment, we have not found in the Internet a simulator for Post Machines. However, professors Patricia Hernández, Alvaro Duque S.J., Jorge García, Enrique Ruiz and Nelson Urrego (Pontificia Universidad Javeriana) have designed a prototype for a Post Machine Simulator (Figure 5).

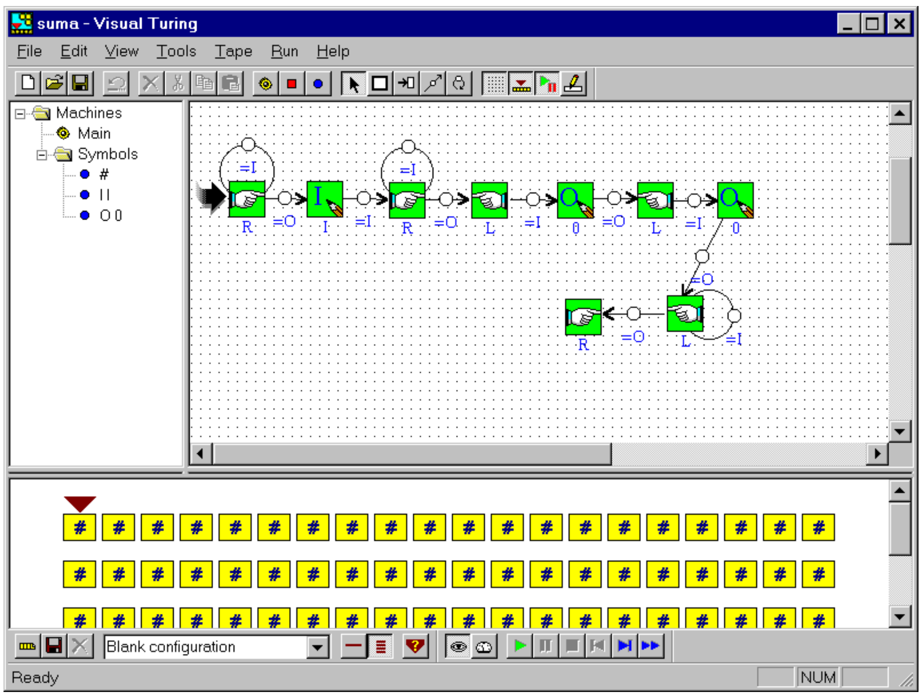


figure 3: Visual Turing Version 1.0.

Working with Turing Machines and Post Machines can stimulate to the students, of different levels, in the acquisition of creative and logic abilities by means of the design of procedures for the calculation of different functions and arithmetic problems. The manipulation of primitive arithmetic procedures allows to conjugate both the early recognition of computing models and the approaching of Mathematics in a non-conventional way.

[1] They, of course, required 165 instructions for this program.

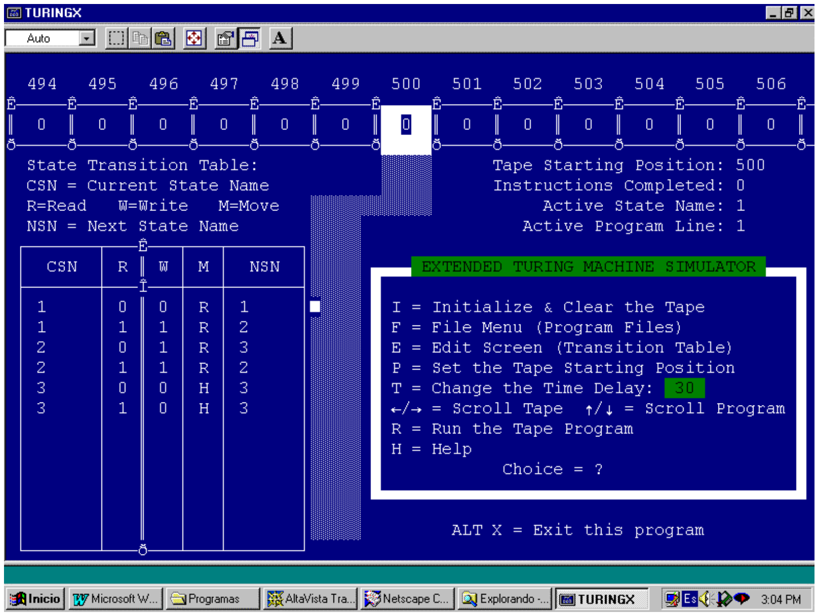


figure4: Turing machine simulator.

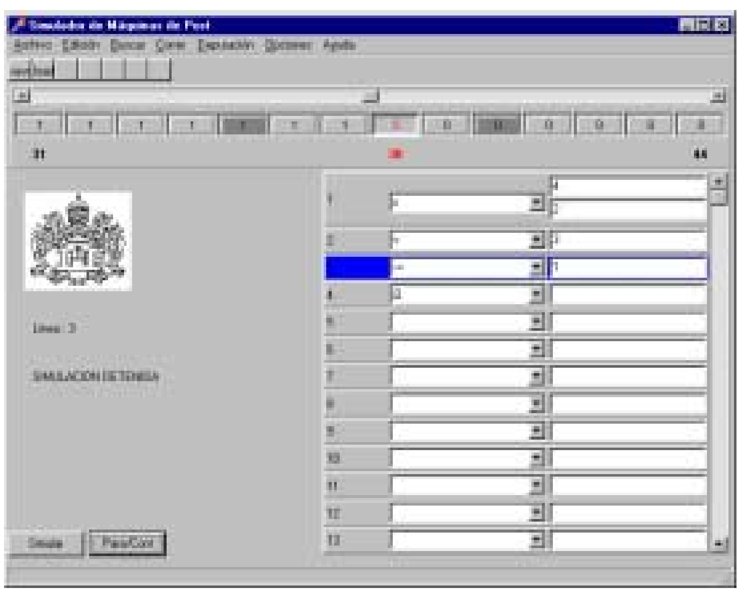


figure 5: Post Machine Simulator

References

1. Boolos & Jeffrey, *Computability and Logic*. Cambridge University Press. 3rd ed. 1989.

2. Davis Martin, *Computability and Unsolvability*. Dover Publications, New York 1958.

3. Hilbert David, *Mathematichse Probleme, Vortag, gehalten auf dem internationalen Mathematiker-Kongress zu Paris 1900*. Nachrichten Akad. Wiss. Göttingen, Math.-Phys. Kl. (1900) 253-297. English translation: Bull. Amer. Math. Soc., 8 (1901-1902) 437-479.

4. Matiyasevic Yuri, *The relation of systems of equations in words and their lengths to Hilbert's tenth problem* (Russian). Issledovaniya po Konstruktivnoi Matematike i Matematiceskoi Logike II. Vol. 8, pp. 132-144

5. Post Emil, *Finite Combinatory Processes. Formulation 1*. Journal of Symbolic Logic, Vol 1. Pp 103-105. 1936.

6. Turing Alan, *On Computable numbers with an applications to the Enscheidungsproblem.* Proc. London Math. Soc., ser 2, Vol. 45, pp. 161-228.

7. Urrego Nelson, *Introducción a la Teoría de Recursión y al Concepto de Calculabilidad.* Memorias del VII Encuentro de la Escuela Regional de Matemáticas, Medellín 1999.

8. Uspenski V., *La Máquina de Post*. Editorial Mir, Moscú, 1980.