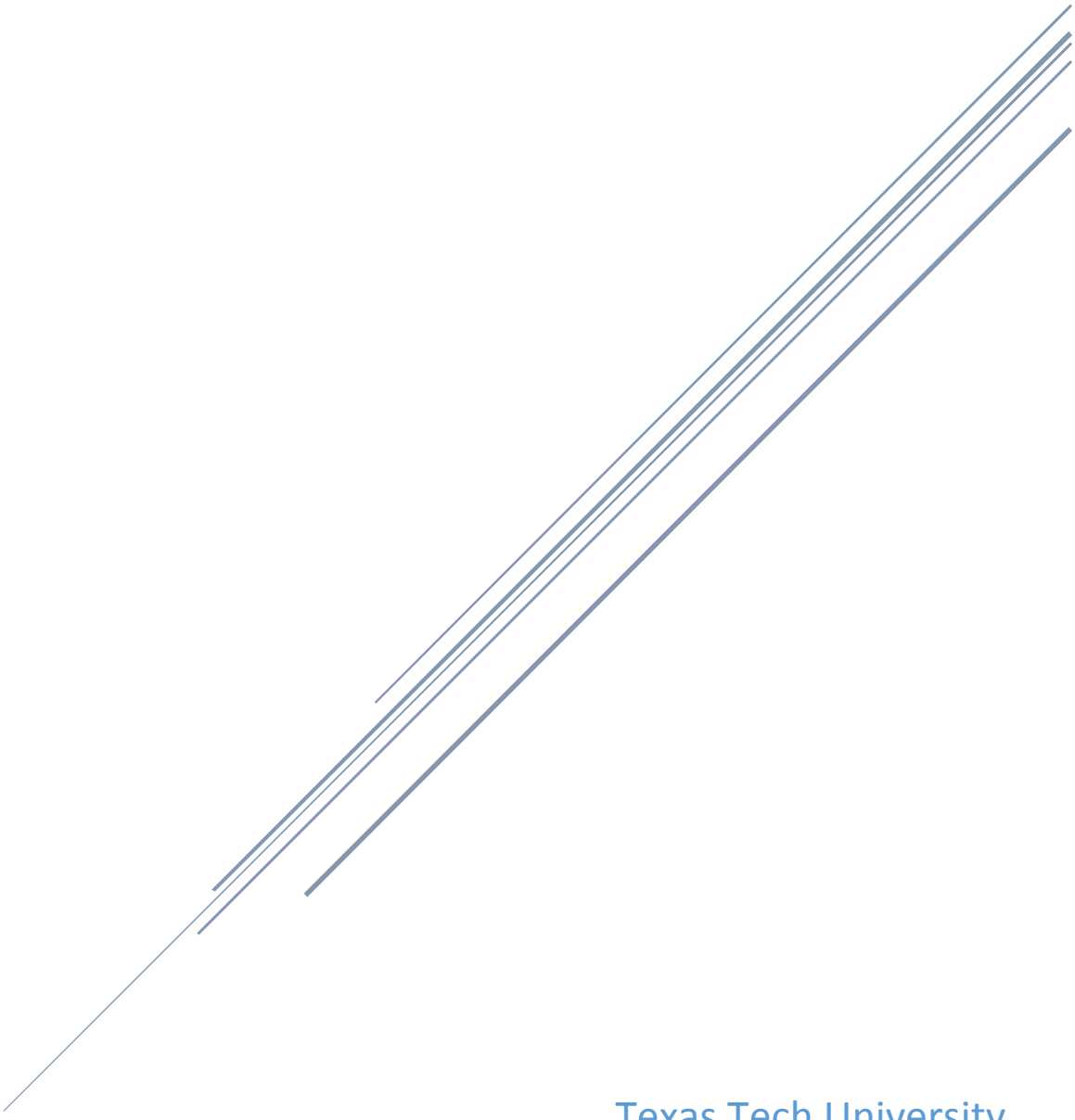
TURING MACHINE INTERPRETER

Instruction Manual



Texas Tech University
Independent Study

Table of Contents

1 Introduction

This program was written to aid in the understanding of Turing Machines. It does so by allowing the user to execute Turing Machines and trace their execution. It was specifically designed to allow the execution of Turing Machines as they are defined in Turing's paper *On Computable Numbers, with an Application to The Entscheidungsproblem*.

Though the interpreter is limited to only the printable ASCII characters as symbols on the tape, any Turing Machine presented in the paper can be executed with this interpreter. In fact, some of the machines presented in the paper are presented in executable format as an appendix to this document.

2 Turing Machine File Format

A text file is used to describe an input Turing Machine.

```
30

1
b
//Computes the rational number .010101010101.....
b | none | P0, R | c
c | none | R | e
e | none | P1, R | f
f | none | R | b
```

Figure 1: Sample Turing Machine

An input file consists of the following:

a) The first line contains an integer specifying the number of iterations to execute the machine.
b) The second line specifies the initial tape of the machine.
c) The third line contains an integer specifying the initial position of the head of the machine.
d) The fourth line specifies the initial m-config of the machine.
e) The remaining lines contain the rules of the machine or a comment, one on each line. Blank lines are ignored.
    i. A line is specified to be a comment if it starts with two forward slashes (//).

2.1 Tape

The initial tape is a sequence of characters and blanks ('_'). The initial tape specified in the file begins at index 1. It is possible for a Turing Machine to step past the beginning of the tape during execution. Anything before and after what is specified by the initial tape is assumed to be blank.

2.2 RULES

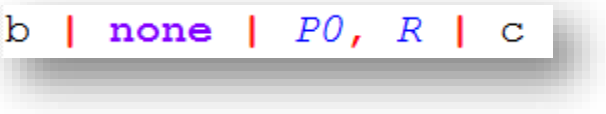


Figure 2: Sample Rule

Every rule consists of four parts separated by a pipe (|):

a) The first part is the name of the m-config or the m-function declaration.
b) The second part is the specification of the scanned symbols. This is a, possibly empty, list of symbols, variable names, and keywords.
c) The third part is the specification of the actions to perform. This is a, possibly empty, list of print commands, erase commands, and move commands.
d) The fourth part specifies the target m-config or m-function.

2.2.1 Rule Name

The first part of a rule is the name of the m-configuration, specified as, possibly zero-ary, m-function declaration. For an overview of how these m-functions work please see section *"4. Abbreviated Tables"* starting on page 235 of Turing's paper.

An m-function declaration is of the form:

```
<name>(<var1>, <var2>, ..., <varn>)
```

For a zero-ary m-function, the parentheses can be dropped.

2.2.2 Scanned Symbols

The list of scanned symbols is specified using keywords (none, any), variable names, and single characters each separated by a comma.

- An empty scanned symbol list matches anything.
- *'none'* matches only a bank square ('_').
- *'any'* matches any symbol except a blank.
- A symbol matches that symbol on a square.
- A variable matches whatever symbol the variable is instantiated with.
- The scanned symbols list matches a symbol if any element of the list matches the symbol.

Matching of rules goes from top to bottom. If two rules meet the selection criteria (correct m-config and scanned symbol), the rule that occurs highest in the file is chosen.

2.2.3 Actions

The list of actions is specified as a list of commands in one of three forms (Print, Erase, and Move) each separated by a comma. The actions are performed one at a time in the order they are specified.

2.2.3.1 Print

Print actions are specified by a capital *'P'* followed by the symbol or variable to print. This action overwrites the currently scanned square on the tape with the specified symbol. If 'P' is followed by something longer than a single symbol, that is not a variable, only the first character is printed.

2.2.3.2 Erase

The erase action is specified by a capital *'E'*. This action erases the symbol on the currently scanned square leaving a blank or '_'.

2.2.3.3 Move

A move action is specified by either a capital *'L'* or *'R'*. *'L'* moves the scanned square once to the left and 'R' moves it once to the right.

2.2.4 Target

The rule target specifies the next m-configuration the machine will move to. This is specified in terms of instantiated m-functions or the keyword *'halt'*. When instantiating an m-function the user can make use of the *'scanned'* keyword, which is substituted by the currently scanned symbol during execution.

2.3 A Brief Example

The following Turing machine copies the initially scanned symbol to the end of the tape and then halts.

```
1   1000
2   #aabc
3   1
4   b
5
6   b |  |  | copy( halt )
7
8   //print current scanned symbol at end of tape
9   copy( alpha ) |  |  | printEnd( scanned , alpha )
10
11  //print a character at end of tape
12  printEnd( alpha, beta ) | none | Palpha | beta
13  printEnd( alpha, beta ) |      |   R    | printEnd( alpha, beta )
```

Figure 3: Brief Example Turing Mschine

Below is the output of the execution trace. Each step in the execution of the machine is illustrated by the current tape, the current m-config, and an indicator showing which square is currently scanned.

```
C:\Turing Machine>TuringMachineInt.exe "Example 1.tm" arrow-trace

#aabc
¦
b

#aabc
¦
copy(halt)

#aabc
¦
printEnd(#,halt)

#aabc
 ¦
 printEnd(#,halt)

#aabc
  ¦
  printEnd(#,halt)

#aabc
   ¦
   printEnd(#,halt)

#aabc
    ¦
    printEnd(#,halt)

#aabc_
     ¦
     printEnd(#,halt)

#aabc#
     ¦
     halt
```

3 USING THE PROGRAM

```
C:\Turing Machine>TuringMachineInt.exe <fileName> <outputFlag> <interactive>
```

Figure 4: Command Line Format

The Turing Machine Interpreter is used by passing three command-line arguments.

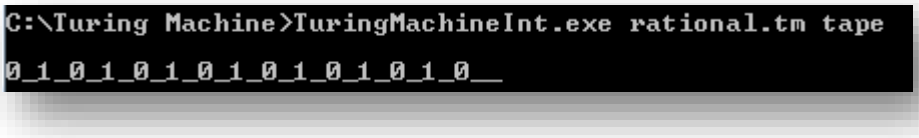
a) The first argument, *<filename>*, specifies the location of the Turing Machine file to execute.
b) The second argument, *<outputFlag>*, specifies the desired output. This argument is optional. It's default value is "arrow-state"
c) If the user would like to use the interpreter in *interactive mode* the third argument, *<interactive>*, is passed '-i'. This argument is optional and by default the interpreter does not run in interactive mode.

3.1 OUTPUT FLAG

The output flag is used to specify what information to display.

Possible Values:

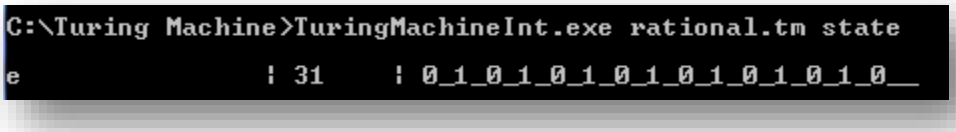
1. tape:
   The flag 'tape' is used to show the contents of the tape after the specified number of iterations have been executed.



2. f-squares:
   The flag 'f-squares' is used to show the contents of the f-squares (as defined by Turing on page 235 of his paper) in the tape after the specified number of iterations have been executed.

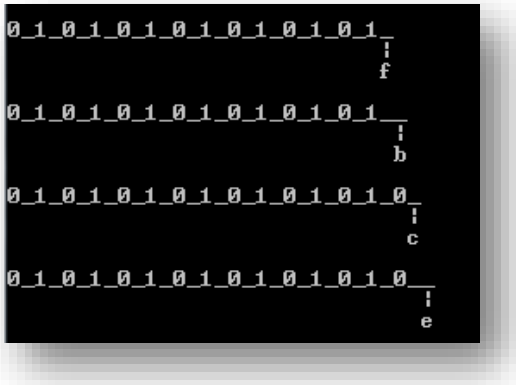
3. arrow-state:
   The flag 'arrow-state' is used to show the final state of the Turing Machine, in an easy to read format, after the specified number of iterations have been executed. The format shows the final tape, the final m-config, and an indicator showing which square is scanned.

4. state:
   The flag 'state' is used to show the final state of the Turing Machine, in a simpler format, after the specified number of iterations have been executed. The format shows final m-config, the index of which square is scanned, and the final tape.

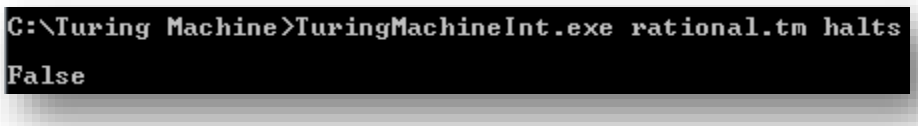


5. arrow-trace:
   The flag ‘arrow-trace’ is used to show the execution trace of the machine. Each step in the execution of the machine is illustrated by the formatted ‘arrow-state’.

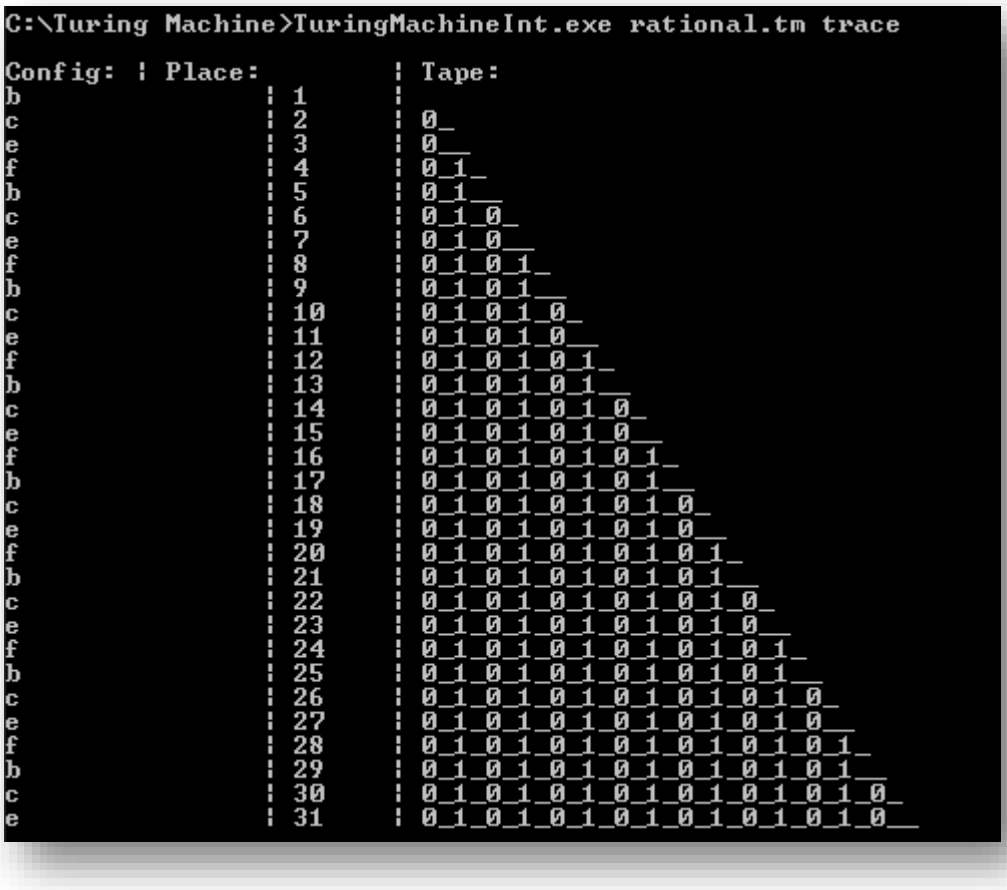
⋮



6. halts:
   The flag ‘halts’ is used to show whether the machine reaches the built-in *‘halt’* m-config. Outputs ‘true’ if *‘halt’* is reached and false otherwise.



7. trace:
   The flag ‘trace’ is used to show the execution trace of the machine. Each step in the execution of the machine is illustrated by the simple ‘state’ format.

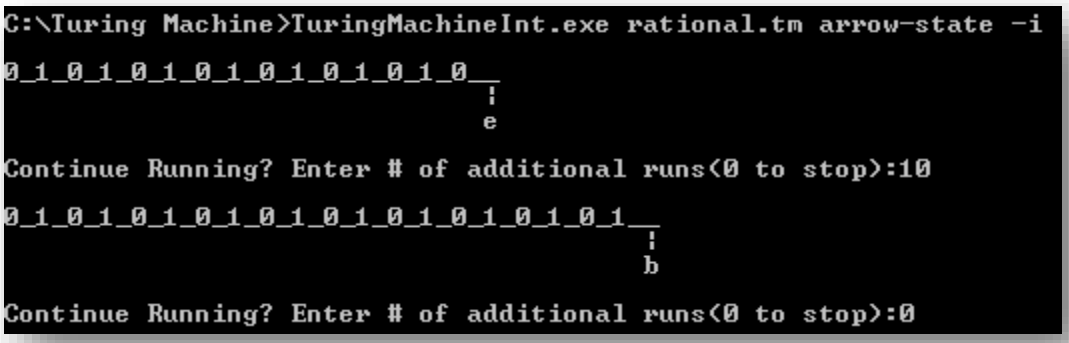


8. utm-out:
   The flag ‘utm-out’ is used to show the contents of the tape that would be the output of Turing’s Universal Machine as defined on pages 241 through 246.

3.2 INTERACTIVE MODE

Interactive mode allows the user to continuously execute the input Turing machine, seeing the partial output between executions. To start the interpreter in interactive mode, include the third command line argument "-i". If the "-i" is not provided, the machine will execute only the number of iterations specified in the file.

When started in interactive mode, the interpreter will execute the given Turing machine the number of times specified in the file. It will then ask the user to enter the next number of iterations to continue execution. This repeats until the user enters some number less than 1 or the Turing machine reaches the *'halt'* state or the target m-config is not defined.



4 APPENDIX

4.1 MACHINE WHICH COMPUTES A RATIONAL NUMBER

Presented on page 233.

```
30

1
b
//Computes the rational number .010101010101..
b | none | P0, R | c
c | none | R | e
e | none | P1, R | f
f | none | R | b
```

Modified and presented again on page 234.

```
30

1
b
//Computes the rational number .010101010101...
b | none | P0       | b
b | 0    | R, R, P1 | b
b | 1    | R, R, P0 | b
```

4.2 Machine Which Computes an Irrational Number

Presented on page 234.

```
100

1
b
//Computes the Irrational Number .0010110111011110111110111111.....
//Sets up the beginning of the tape.
b | | Pe, R, Pe, R, P0, R, R, P0, L, L | o

//Marks all of the 1's with x
o | 1 | R, Px, L, L, L | o
o | 0 | | q

//Goes to the end and prints a '1'.
q | 0, 1 | R, R | q
q | none | P1, L | p

//Goes to the first x and then Handles erasing the 'x' that marks a '1'.
p | x | E, R | q
p | e | R | f
p | none | L, L | p

//Tacks a '0' to the end.
f | any | R, R | f
f | none | P0, L, L | o
```