A Generator for Turing Machine Simulating Programs
– User's Manual –

Peter R.J. Asveld & Eerke A. Boiten

Department of Computer Science, Twente University of Technology
P.O. Box 217, 7500 AE Enschede, The Netherlands

By means of some sample dialogues we show the use of a program to generate Berkeley Pascal programs from Turing machine descriptions such that these Pascal programs simulate the behavior of the corresponding Turing machines.

1. Generating a Program to Simulate a Turing Machine

Starting from a description of a Turing machine – according to the grammar given in the file "`turgra`" (See also Appendix 1) – in a file named "`filenm.inp`" (i.e., six letters followed by a "." and "`inp`") the C-shell script "`gensim`" generates a simulating program for this Turing machine. An example of such an input file has been included in Appendix 2. First "`gensim`" generates a Pascal program "`filenm.p`" which is compiled immediately by invoking the Berkeley Pascal compiler. The resulting executable code can be found in the file "`filenm`". Thus typing "`filenm`" results in starting the simulation of the Turing machine described in "`filenm.inp`" provided that this description contains no errors. The grammar in "`turgra.txt`" is based on the conventions introduced in [2] and used in [1].

An example with "`filenm.inp`" equal to "`palind.inp`" gives rise to the following dialogue. The Turing machine described in the file "`palind.inp`" decides the set of all palindromes over $\{a,b\}$ of odd length greater than or equal to three having an a as central letter. This example as well as a few other ones is in the directory "`EXAMPLES`".

```
1-> ls -l EXAMPLES
total 14
-rw-r--r--  1 infprja      1884 Nov  7 13:38 abcopy.inp
-rw-r--r--  1 infprja       720 Nov  7 13:38 capmir.inp
-rw-r--r--  1 infprja      2372 Nov  7 13:38 copysl.inp
-rw-r--r--  1 infprja       461 Nov  7 13:38 lepaex.inp
-rw-r--r--  1 infprja      2267 Nov  7 13:38 palind.inp
-rw-r--r--  1 infprja      2306 Nov  7 13:38 paline.inp
-rw-r--r--  1 infprja       274 Nov  7 13:38 shiftl.inp
2-> cp EXAMPLES/palind.inp .
3-> gensim palind.inp

Turing Machine Simulator Generator  running on 22 Oct 86

No errors. Generate Pascal program(y/n)? y
Pascal program will be generated.
Output of Turing machine on screen(y/n)? y
```

```
Generation starts.
This program will now be compiled.

Simulator for palind is ready.  Type: palind

4-> palind
Initialisation :
Print configurations           (y/n) ? y
Print transitions              (y/n) ? n
Give maximum length of computation :1000
Print length and # cells of computation    (y/n) ? y
Give contents of tape (without rightmost  #"s) :
#abbabba
************************************************************
q00   #abbabba#
           ^
q01   #abbabba
          ^
q01   #abbabba
         ^
q01   #abbabba
        ^
q01   #abbabba
       ^
q01   #abbabba
      ^
q01   #abbabba
     ^
q01   #abbabba
    ^
q02   1abbabba
     ^
q03   1abbabba
    ^
q04   1#bbabba
    ^
q05   1#bbabba
     ^
q05   1#bbabba
      ^
q05   1#bbabba
       ^
q05   1#bbabba
        ^
q05   1#bbabba
         ^
q05   1#bbabba#
           ^
q06   1#bbabba
          ^
q10   1#bbabb#
         ^
```

```
q11   1#bbabb
           ^
q11   1#bbabb
          ^
q11   1#bbabb
         ^
q11   1#bbabb
        ^
q11   1#bbabb
       ^
q11   1#bbabb
      ^
q03   1#bbabb
       ^
q07   1##babb
       ^
q08   1##babb
        ^
q08   1##babb
         ^
q08   1##babb
          ^
q08   1##babb
           ^
q08   1##babb#
            ^
q09   1##babb
           ^
q10   1##bab#
           ^
q11   1##bab
          ^
q11   1##bab
         ^
q11   1##bab
        ^
q11   1##bab
       ^
q03   1##bab
        ^
q07   1###ab
        ^
q08   1###ab
         ^
q08   1###ab
          ^
q08   1###ab#
           ^
q09   1###ab
          ^
q10   1###a#
           ^
q11   1###a
          ^
q11   1###a
         ^
q03   1###a
          ^
q04   1####
```

```
          ^
q05   1#####
           ^
q06   1####
          ^
q15   1###
         ^
q15   1##
          ^
q15   1#
           ^
q15   1#
          ^
q16   ##
          ^
q17   ##
           ^
q17   #1
           ^

Computation took  60 steps and  9 cells.
Machine stopped ; contents of tape :
      #1#
        ^

Once more (y/n) ? y
Initialisation :
Print configurations           (y/n) ? n
Print transitions              (y/n) ? n
Give maximum length of computation :1000
Print length and # cells of computation    (y/n) ? y
Give contents of tape (without rightmost  #"s) :
#bbabbabb
************************************************************

Computation took  72 steps and  10 cells.
Machine stopped ; contents of tape :
      #0#
        ^

Once more (y/n) ? n
5->  . . .
. . .
10-> palind
Initialisation :
Print configurations           (y/n) ? y
Print transitions              (y/n) ? n
Give maximum length of computation :1000
Print length and # cells of computation    (y/n) ? y
Give contents of tape (without rightmost  #"s) :
#bab
************************************************************
q00   #bab#
          ^
```

```
q01   #bab
        ^
q01   #bab
       ^
q01   #bab
      ^
q01   #bab
     ^
q02   1bab
     ^
q03   1bab
      ^
q07   1#ab
      ^
q08   1#ab
       ^
q08   1#ab
        ^
q08   1#ab#
         ^
q09   1#ab
        ^
q10   1#a#
        ^
q11   1#a
       ^
q11   1#a
      ^
q03   1#a
       ^
q04   1##
       ^
q05   1###
        ^
q06   1##
       ^
q15   1#
      ^
q15   1#
     ^
q16   ##
     ^
q17   ##
      ^
q17   #1
      ^

Computation took  24 steps and  5 cells.
Machine stopped ; contents of tape :
      #1#
        ^
Once more (y/n) ? n
11-> . . .
```

After quitting the simulator we can restart the simulator by retyping "`filenm`" (See command: `10-> palind`). The entire dialogue appears on the screen and is also saved in a file named "`filenm.LOG`" (In case of our example: "`palind.LOG`"), which

may be consulted afterwards by means of the commands

```
more filenm.LOG
```

or

```
view filenm.LOG
```

2. Error Messages

Omitting a file name after the command "`gensim`" or providing the wrong filename results in a corresponding warning on the screen.

Much more interesting is the case in which the description of the Turing machine in the file named "`filenm.inp`" contains errors, viz. syntax errors with respect to the grammar given in "`turgra.txt`". First we consider a sample dialogue.

```
16-> cp EXAMPLES/paline.inp .
17-> gensim paline.inp

Turing Machine Simulator Generator  running on 22 Oct 86

Errors :          8
18-> . . .
```

Now the dialogue is very short indeed. The generator detected 8 errors in the description of the Turing machine. These errors are collected in a file named "`filenm.lst`" which may be read by the usual commands

```
more filenm.lst
```

or

```
view filenm.lst
```

Of course as long as "`filenm.inp`" contains errors no Pascal program is generated. As soon as the file "`filenm.inp`" is syntax error-free the command

```
gensim filenm.inp
```

does not only results in a Berkeley Pascal program "`filenm.p`" and the corresponding executable code in "`filenm`" but a complete description of the Turing machine will be put into "`filenm.lst`".

The command "`gensim filenm.inp`" creates, apart from the files "`filenm.p`", "`filenm`", "`filenm.lst`" and "`filenm.LOG`", a file called "`filenm.dlt`" which consists of a coded version of the transition function.

3. Files

The Pascal sources of the generator ("gen.p") and related Pascal files are in the directory "src.p". The compiled version of "gen.p" which is in the file "gen" and the C-shell script "gensim" are of course in the directory "~/bin". As mentioned before some example input files are in the directory "EXAMPLES".

4. History

A first version of this generator has been written by the second author in DEC VAX Pascal. The first author modified this version, converted it to Berkeley Pascal and wrote the C-shell script "gensim".

References

1. P.R.J. Asveld, *Complexiteit van Berekeningen,* (1998), Third edition, Department of Computer Science, Twente University of Technology, Enschede, The Netherlands (Lectures notes; in Dutch).
2. H.R. Lewis & C.H. Papadimitriou, *Elements of the Theory of Computation,* (1981) Prentice-Hall, Englewood Cliffs, N.J. Second edition (1988).

Appendix 1

Grammar for Creating Input Files for the "gensim" Script

```
input → 'MACHINE' machinename alphabet state start delta
        tapelength

machinename → {Pascal identifier of maximal 20 characters}

alphabet → 'ALPHABET' capitals letters digits more

capitals → 'CAPITALS' | 'NOCAPITALS'

letters → 'LETTERS' | 'NOLETTERS'

digits → 'DIGITS' | 'NODIGITS'

more → 'NOMORE' | 'MORE' alphsym CHAIN ','

alphsym → {All characters except '(', ')', ',', '>' and ' '}

state → 'STATE' standard morestates

standard → 'NOSTANDARD' | 'STANDARD' {Integer in between 0 and
           maxstate}

morestates → 'MORE' stateid CHAIN ',' | NOMORE

stateid → {Any string of 1-3 letters or digits not equal to
```

```
              'H'}
start  →  'START'  stateid
delta  →  'DELTA'  transition  SEQ
transition  →  'ON (' oldstate ',' headsym ') TO (' newstate ','
               action ')'
oldstate  →  'ANY' | stateid
headsym  →  'ANY' | alphsym
newstate  →  'H' | 'SELF' | stateid
action  →  'SELF' | movement | alphsym
movement  →  '>L' | '>R'
tapelength  →  'TAPELENGTH' {Integer in between 1 and
               maxtapelength}
```

Comments on Syntax

All symbols may be separated by spaces or returns. Correct parsing for “illegal” alphsyms is not guaranteed. There are some other restrictions:

- The `stateid` in "`start`", "`oldstate`" and "`newstate`" ought to be defined beforehand, either by means of the "`standard`" option which introduces stateids ’`q00`’ up to ’`q99`’, or explicitly using "`morestates`".
- The `alphsym` in "`headsym`" and "`action`" ought to be defined beforehand, either by using one of the options "`capitals`", "`letters`" or "`digits`" which add capitals, letters and digits to the machine alphabet respectively, or explicitly using "`more`".
- ’`SELF`’ in "`newstate`" or "`action`" can only be used if the corresponding "`oldstate`" or "`headsym`" respectively equals
- The total number of states cannot exceed `maxstate`; each state can only be defined once.
- The total number of symbols in the alphabet cannot exceed `maxalpha`; each symbol can only be defined once.
- Transitions may be defined more than once: the last definition counts. For all combinations of states and symbols a newstate and an action ought to be defined. In case there are transitions that need not be defined at all, adding a first line of the form

  ```
  ON (ANY,ANY) TO (H,SELF)
  ```

  after ’`DELTA`’ will suffice.

The names `maxstate`, `maxalpha` and `maxtapelength` refer to constants in the Pascal program “`gen.p`”. Their current values are 100, 100 and 10000 respectively.

Appendix 2

Example of an Input File for the "gensim" Script

```
MACHINE  palindrome
ALPHABET NOCAPITALS
         NOLETTERS
         NODIGITS
MORE     a,b,0,1
STATE    STANDARD 18
         NOMORE
START    q00
DELTA    ON (q00,ANY) TO (q01,>L)
         ON (q01,a  ) TO (q01,>L)
         ON (q01,b  ) TO (q01,>L)
         ON (q01,#  ) TO (q02,1 )
         ON (q01,0  ) TO (q01,0 )
         ON (q01,1  ) TO (q01,1 )
         ON (q02,ANY) TO (q03,>R)
         ON (q03,a  ) TO (q04,# )
         ON (q03,b  ) TO (q07,# )
         ON (q03,#  ) TO (q12,>L)
         ON (q03,0  ) TO (q03,0 )
         ON (q03,1  ) TO (q03,1 )
         ON (q04,ANY) TO (q05,>R)
         ON (q05,a  ) TO (q05,>R)
         ON (q05,b  ) TO (q05,>R)
         ON (q05,#  ) TO (q06,>L)
         ON (q05,0  ) TO (q05,0 )
         ON (q05,1  ) TO (q05,1 )
         ON (q06,a  ) TO (q10,# )
         ON (q06,b  ) TO (q12,# )
         ON (q06,#  ) TO (q15,>L)
         ON (q06,0  ) TO (q06,0 )
         ON (q06,1  ) TO (q06,1 )
         ON (q07,ANY) TO (q08,>R)
         ON (q08,a  ) TO (q08,>R)
         ON (q08,b  ) TO (q08,>R)
         ON (q08,#  ) TO (q09,>L)
         ON (q08,0  ) TO (q08,0 )
         ON (q08,1  ) TO (q08,1 )
         ON (q09,a  ) TO (q12,# )
         ON (q09,b  ) TO (q10,# )
         ON (q09,#  ) TO (q12,>L)
         ON (q09,0  ) TO (q09,0 )
         ON (q09,1  ) TO (q09,1 )
```

```
          ON (q10,ANY) TO (q11,>L)
          ON (q11,a  ) TO (q11,>L)
          ON (q11,b  ) TO (q11,>L)
          ON (q11,#  ) TO (q03,>R)
          ON (q11,0  ) TO (q11,0 )
          ON (q11,1  ) TO (q11,1 )
          ON (q12,a  ) TO (q12,# )
          ON (q12,b  ) TO (q12,# )
          ON (q12,#  ) TO (q12,>L)
          ON (q12,0  ) TO (q12,0 )
          ON (q12,1  ) TO (q13,# )
          ON (q13,ANY) TO (q14,>R)
          ON (q14,a  ) TO (q14,a )
          ON (q14,b  ) TO (q14,b )
          ON (q14,#  ) TO (q14,0 )
          ON (q14,0  ) TO (H  ,>R)
          ON (q14,1  ) TO (q14,1 )
          ON (q15,a  ) TO (q15,a )
          ON (q15,b  ) TO (q15,b )
          ON (q15,#  ) TO (q15,>L)
          ON (q15,0  ) TO (q15,0 )
          ON (q15,1  ) TO (q16,# )
          ON (q16,ANY) TO (q17,>R)
          ON (q17,a  ) TO (q17,a )
          ON (q17,b  ) TO (q17,b )
          ON (q17,#  ) TO (q17,1 )
          ON (q17,0  ) TO (q17,0 )
          ON (q17,1  ) TO (H  ,>R)
TAPELENGTH 100
```