International Journal of Soft Computing 11 (3): 131-135, 2016
ISSN: 1816-9503

About Realization of Some Turing Machine is Finite-State and Store Machines with the Exit

N.K. Zarubina and V.P. Dobritsa
Southwest State University, St. 50 Years of October, 94, 305040 Kursk, Russia

Abstract: The concept of the store automatic machine with an exit from the point of view of computability theory is entered. The comparative analysis of the entered concept with automatic machine with store memory from the point of view of formal languages and grammars is carried out. Examples of automatic realization of some cars of Turing by means of finite-state and store machines with exits are reviewed. It is proved that the circle of the tasks realized by the store automatic machine with an exit is wider, than at the finite-state machine with an exit, but already, than at Turing's machine.

Key words: Finite-state machine, the store automatic machine, the store automatic machine with an exit, Turing's machine, Russia

INTRODUCTION

In the theory of automatic machines and computability theory interest is attracted reducibility (extent of automatic transformations (Bayrasheva, 1982; Korneeva, 2011) for finite-state machines, T-degree for Turing (Rogers, 1972) machine). It is known that possibilities of the car of Turing much ample opportunities of finite-state machines. Therefore studying of automatic machines which will be by the opportunities between finite-state machines and Turing's machine attracts interest. Such automatic machine is the store automatic machine which unlike the finite-state machine has "memory" an opportunity to address stack ("store") storage. At the same time we can write down an element in a stack, get an element from a stack and just read out an element, without any actions over a stack. Those in the store automatic machine transition depends not only on current state and an entrance symbol but also on an element which is at stack top.

The concept of the store automatic machine (the automatic machine with store memory, the SM- automatic machine, the SM-converter) has appeared in the 60th of the last century (Shutzenberger, 1963; Lewis and Stearns, 1968; Akho and Ullman, 1978) and is so far well studied, but only as the automatic machine without an exit (the "recognizing" automatic machine) from the point of view of formal languages and grammars. At the same time the question about the reducibility from the point of view of the theory of a computability still remains open. Therefore we will give generalization of a concept of the store automatic machine, addition in him an output tape.

We will give exact definition of a store automatic machine (Sipser, 2012). Definition 1; The Store Automatic Machine (SAM) is called the six:

$$(Q, \Sigma, \Gamma, \delta, q_0, F)$$

Where:

Q = Final set of states
Σ = Entrance alphabet
Γ = Store alphabet
δ = $Q\times(\Sigma\cup\Lambda)\times(\Gamma\cup\Lambda)^{\neg}Q\times(\Gamma\cup\Lambda)$ function of transitions
Λ = Empty symbol
q_0 = The initial state
F = Set of final states

Thus, the store automatic machine is set by system of teams of a look:

$$q_i, a, A \rightarrow q_j, B \quad (1)$$

Where:

q_i = Current state of SAM
a = An entrance symbol (can be)
A = The current top symbol of store (stack)
q_i = A new state
B = A new top symbol of store
$\neg$ = The external symbol which isn't belonging to any of alphabets of SAM

As can be seen from the definition 1 this automatic machine without an exit he is intended only for recognition (in him there is no output film and respectively, the output alphabet; and this automatic machine, as a rule has to finish work in one of the

Corresponding Author: N.k. Zarubina, Southwest State University, St. 50 Years of October, 94, 304050 Kursk, Russia

allocated final states). However, from the point of view of the theory of automatic machines, a set of final states and the allocated initial state not obligatory attributes of the automatic machine. Besides, in automatic machines there are output alphabet and the film. According to this remark, we will give definition of the store automatic machine with an exit (Zarubina *et al*., 2016a).

The store automatic machine with an exit

Definition 2: We will call the store automatic machine with an exit (SAMwE): The five:

$$(Q, \Sigma, \Sigma', \Gamma, \delta)$$

Where:

Q = Final set of states
Σ = entrance alphabet
Σ' = output alphabet
Γ = store alphabet
δ = $Q\times(\Sigma\cup\Lambda)\times(\Gamma\cup\Lambda)\neg Q\times(\Sigma'\cup\Lambda)\times(\Gamma\cup\Lambda)$, function of transitions, an empty symbol

Those the team from system of teams for the store automatic machine with an exit looks as follows:

$$q_i, a, A \rightarrow q_j, B \qquad (2)$$

Where:

q_i = Current state of SAMwE
a = An entrance symbol (can be Λ)
A = The current top symbol of store (can be Λ)
q_j = A new state
b = An output symbol (can be Λ)
B = A new top symbol of store
$\neg$ = The external symbol which isn't belonging to any of alphabets of SAMwE

Thus, SAM is a special case of SAMwE if to put that the output alphabet $\Sigma' = \phi$ and on an exit always moves an empty symbol Λ. By analogy with the finite-state machine with an exit the store automatic machine with an exit can be set by means of the focused count. At the same time each condition of SAMwE is the count's top and each team of a look (Eq. 2) is implemented by an edge between tops of q_i and q_j (Fig. 1). As already it has been noted earlier, we can execute for one operation one of the 3rd of actions over a stack. For convenience of reasonings we will accept the following designations: If we write down an element in a stack, then in team (Eq. 2) we write instead of To 2 symbols written down and previous. Example: q_i, a, A$\neg q_j$, b, BA (recorded the symbol "B" at the previous value in a stack A). If we get a symbol from a stack, instead of B in Eq. 2, we write an empty symbol. Example: q_i, a, A$\neg q_j$, b, Λ (have got a symbol from a stack A).

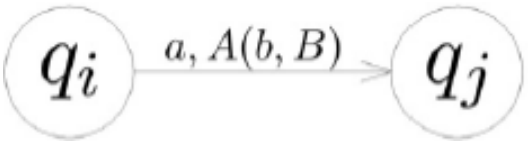

Fig. 1: Implementation of the command (Eq. 2) by means of the count

If we do" t make any changes to a stack, then the team has an appearance: q_i, a, A$\neg$ q_j, b, A where instead B the top symbol of a stack just corresponds. By analogy with the standard terms in the theory of automatic machines, we will enter some more definitions.

Definition 3: We will call the Initial Store Automatic Machine with an Exit (ISAMwE) SAMwE with the set initial state.

Definition 4: We will call the Determined Store Automatic Machine with an Exit (DSAMwE) SAMwE in which for the left part of team of a look (Eq. 2), there is the only right part of team of a look (Eq. 2).

Definition 5: We will call the Nondeterministic Store Automatic Machine with an Exit (NSAMwE) SAMwE in which for the left part of team of a look (Eq. 2) there can be >1 right part of team of a look (Eq. 2).

We will notice that already on these concepts finite-state machines and store automatic machines differ. Finite-state machines have property that the finite-state determined machine is equivalent some determined and naturally, on the contrary. And here for store automatic machines this property doesn't take place (Meduna, 2000). Nondeterministic store automatic machines are more powerful than the determined store automatic machines. As, it was already noted above, store automatic machines are a special case of store automatic machines with an exit, therefore the same ratio takes place and for them. Further all reasonings in article will concern DSAMwE. We will consider on examples as various machine of Turing can be realized by finite-state and store machines with an exit.

MATERIALS AND METHODS

Realization of some MT finite-state and store machines with an exit: It is obvious that owing to the thesis of

Chercha, we realize any algorithm realized by the finite-state machine or the store automatic machine also by Turing's machine. Therefore, interest is attracted by the return realization what types of machine of Turing can be realized by various automatic machines. At the same time, once again to pay attention to distinctions in concepts SAMwE and SM-converters, we will consider not standard tasks on calculation of predicates of a look:

$$P(x)=x\in\{a^n b^n \mid n>0\}$$

or

$$Q(x)=x\in\{a^n b^n \mid n>0\}$$

which are given when studying SM-automatic machines.

Summation of final number of composed: Formulation of a task: on the input tape MT we have final number composed in an unary numeral system (where, n of units is n-1), divided among themselves by one zero. It is necessary to receive on an output tape the sum composed (where already n of units correspond to number n).

We will set MT according to definition: the set of entrance states Q = {q_0, q_1, ..., q_8} the entrance alphabet coincides with output state $\Sigma = \Sigma' = \{0, 1\}$ and function of transitions δ: $Q\times\Sigma\to Q\times\Sigma'\times\{L, R, N\}$ is set by Table 1 (q_8-a final state). This machine of Turing can be realized by both the finite-state machine and the store automatic machine with an exit. And in both cases because of existence of a separate output tape function of transitions will be set by smaller number of states.

Realization by the finite-state machine with an exit: The state graph of the finite-state machine realizing the sum of final number of composed is represented in Fig. 2. Thus, this MT is implemented by the finite-state machine with an exit in which a set of states Q = {q_0, q_1, q_2} the entrance alphabet $\Sigma = \{0, 1\}$ the output alphabet $\Sigma' = \{0, 1\}$ and function of transitionsδ: $Q\times\Sigma\to Q\times\Sigma'$ is set by Table 2 (q_2-a final state).

Realization by the store automatic machine with an exit: It is obvious that this algorithm through SAMwE can be realized by analogy with the previous realization by the finite-state machine having just put that the stack is always empty (i.e., the stack won't be involved). However, one of the purposes to show operation of the store automatic machine in operation. Therefore, we will give realization of SAMwE with use of store. We will set the

Table 1: Function of transitions of the machine of Turing for an example 1

Transition	0	1
q_0	q_6, 0, N	q_1, 0, R
q_1	q_2, 0, R	q_1, 1, R
q_2	q_6, 0, L	q_1, 0, R
q_3	q_4, 1, L	-
q_4	q_5, 0, R	q_4, 1, L
q_5	-	
q_6	q_7, 0, L	-
q_7	q_8, 0, R	q_7, 0, L

Table 2: Function of transitions of the finite-state machine for an example

Transition	0	1
q_0	q_2, Λ	q_1, Λ
q_1	q_0, Λ	q_1, 1

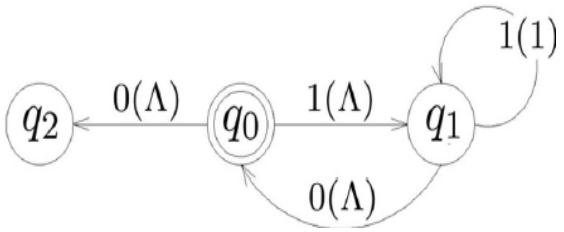

Fig. 2: A state graph of the finite-state machine for an example 1

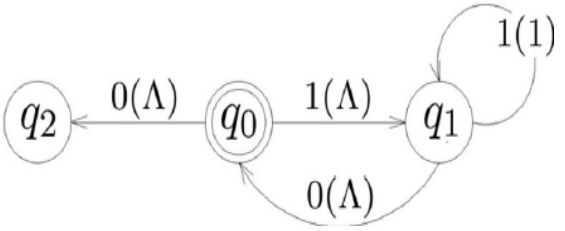

Fig. 3: State graph of the store automatic machine for an example 1

store automatic machine with an exit according to definition. Let the set of states Q = {q_0, q_1, q_3} entrance, output and store alphabets coincide $\Sigma = \Sigma' = \{0, 1\}$ and function of transitions δ: $Q\times(\Sigma\cup\Lambda)\times(\Gamma\cup\Lambda)\to Q\times(\Sigma'\cup\Lambda)\times(\Gamma\cup\Lambda)$ is set by Table 3 (q_2 a final state). The state graph of this SAMwE is represented in Fig. 3. Here and further for reduction of record of work of SAMwE we will designate through and the top symbol of a stack (if the specification what the top symbol is isn't necessary; he can also be an empty symbol).

"Copy": sequence: The formulation of the problem: on the input tape recorded sequence of any number of units. It should be on the output tape to record the original sequence and its "copy", separated by a single 0. The set of entrance conditions of turing machine Q = {q_0, q_1,..., q_{11}} the entrance alphabet coincides with output $\Sigma = \Sigma' = \{0, 1\}$ and function of transitions δ: $Q\times\Sigma\to Q\times\Sigma'\times\{L, R, N\}$ is set by Table 4 (q_{11}-a final state).

Table 3: Function of transitions of SAMwE for an example 1

Transitions	Actions
q_0, 1, A→q_0, Λ, 1, A	All units from an entrance tape register in a stack. Nothing is given for an output tape. We remain is able q_0
q_0, 0, A→q_0, Λ, Λ	If on an entrance tape zero, the top unit from a stack are removed. Nothing is given for an output tape. We remain is able q_0
q_0,Λ, A→q_1, Λ, Λ	On an entrance tape elements have come to an end. The top unit from a stack is removed. Nothing is given for an output tape Transition to a state q_1
q_1,Λ, A→q_1, Λ, Λ	On an entrance tape there are no elements. The top elements of a stack on one are given for an output tape (and, respectively, are removed from a stack). We remain is able q_1
q_1,Λ, A→q_2, Λ, Λ	On an entrance tape there are no elements. The stack is empty. Transition to a final state q_2 Completion of work of SAMwE

Table 4:Function of transitions of MT for an example 2

Transition	0	1
q_0	q_8, 0, L	q_1, 0, L
q_1	q_2, 0, L	-
q_2	q_3, 1, L	-
q_3	q_4, 0, L	q_3, 1, L
q_4	q_5, 1, R	q_4, 1, L
q_5	q_6, 0, R	q_5, 1, R
q_6	q_7, 0, R	q_6, 1, R
q_7	q_0, 0, R	-
q_8	q_8, 0, L	q_9, 0, L
q_9	q_{10}, 0, L	q_9, 1, L
q_{10}	q_{11}, 0, R	q_{10}, 1, L

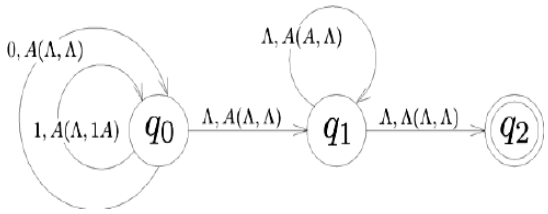

Fig. 4: State graph of the store automatic machine for an example 2

About realization by the finite-state machine with an exit: Because of this problem we need to somehow "remember" how many units contains the original sequence, the implementation of "copying" Turing machine state machine with access impossible.

Realization by the store automatic machine with an exit: Thanks to existence of "memory" (stack) in the store automatic machine, realization of the "copying" MT through SAMwE is much simpler. In this case the set of states Q = {q_0, q_1, q_2} entrance, output and store alphabets coincide Σ = Σ' = Γ = {0, 1} and function of transitions δ: Q×(Σ∪Λ)×(Γ∪Λ)→Q×(Σ'∪Λ)×(Γ∪Λ) is set by Table 5 (q_2-a final state). The state graph of this SAMwE is represented in Fig. 4.

RESULTS AND DISCUSSION

Modular difference of two numbers: Formulation of a task: on an entrance tape 2 numbers in an unary numeral system are set. On an output tape to write down result of a modular difference of the set numbers. In this Turing machine will be consistently erased extreme left and extreme right units. A set of states Q = {q_0, q_1,..., q_8} entrance and output alphabets coincide Σ = Σ' = {0, 1} and function of transitions δ: Q×Σ→Q×Σ1×{L, R, N} is set by Table 6 (q_8-a final state) (Table 6).

About realization by the finite-state machine with an exit: Realization by the finite-state machine with an exit is again impossible as it is necessary "to remember" as far as one number is more than another.

Realization by the store automatic machine with an exit: Realization of a modular difference through SAMwE is again simpler as thanks to existence of a stack the possibility by symbol comparison of the first and second stack, without the need for additional passing of a tape appears. So, the set of states Q = {q_0, q_1, q_2} the entrance alphabet Σ = {0, 1} the output alphabet coincides with store Σ = Γ = {1} and function of transitions δ: Q×(Σ∪Λ)×(Γ∪Λ)→Q×(Σ'∪Λ)×(Γ∪Λ) is set by Table 7 (q_2-a final state). We will notice that if to change operation 3 as follows q_1, 1, 1→q_1,Λ: that this SAMwE realizes an algorithm of comparison as a result of which on an output tape it will be removed bigger of two numbers which are written down on an entrance tape.

Multiplication of two numbers

Formulation of a task: on an entrance tape 2 numbers in an unary numeral system are set. It is necessary to write down the work of the set numbers on an output tape. This task realizable in MT. In a type of dimensions of function of transitions, we won't give realization of an algorithm here. For realization of a task it is necessary "to remember" 2 multipliers. Therefore this algorithm can't be realized both by the finite-state machine with an exit and the store automatic machine with an exit (SAMwE thanks to existence of a stack can remember one value, but not two. Though multiplication of any number by some concrete value means of SAMwE is feasible).

Table 5: Function of transitions of SAMwE for an example 2

Transition	Actions
q_0, 1, A→q_0, 1, 1, A	We bring each unit from the initial sequence to an output tape and the copy we write down in a stack
q_0, Λ, 1→q_1, 0, 1	When entrance symbols have ended, is brought to the output sequence 0 we leave a stack without changes. We pass into a state q_1
q_1, Λ, 1→q_1, 1, Λ	We bring all units out of a stack on an output tape
q_1, Λ, Λ→q_2, Λ, Λ	When units and in a stack end, SAMwE finishes work

Table 6: Function of transitions of MT for an example 3

Transition	0	1
q_0	q_8, 0, R	q_1, 0, R
q_1	q_2, 0, R	q_1, 1, R
q_2	q_3, 0, L	q_2, 1, R
q_3	-	q_4, 0, L
q_4	q_7, 0, L	q_5, 1, L
q_5	q_6, 0, L	q_5, 1, L
q_6	q_0, 0, R	q_6, 1, R
q_7	q_8, 0, R	q_7, 1, L

Table 7: Function of transitions of SAMwE for an example 3

Transition	Actions
q_0, 1, A→q_0, Λ, 1, A	All units of the first register in a stack. On a conclusion moves nothing
q_0, 0, A→q_1, Λ, A	The first has ended, on an entrance tape 0. The stack remains without changes, on a conclusion moves nothing Transition to a state q_1
q_1, 1, 1→q_1, Λ, Λ	On symbolical comparison of the first (in a stack) and the second (on an entrance tape)
q_1, 1,Λ→q_1, 1,Λ	The first more we bring to an output tape all remained units on an entrance tape
q_1, Λ, 1→q_1, 1, Λ	The second more we bring to an output tape all remained units from a stack
q_1, Λ, Λ→q_2, Λ, Λ	Completion of work of SAMwE

CONCLUSION

The following theorems naturally follow from the given examples.

Lemma 1: The store automatic machine for the opportunities at least isn't inferior to the finite-state machine with an exit.

Evidence: Obviously, SAMwE without involvement of a stack is a finite-state machine with an exit. Respectively, any command of the finite-state machine q_i, a→q_j, b is implemented in SAMwE q_i, a, Λ→q_j, b, Λ.

Theorem 1: The store automatic machine with an exit is more powerful (it is capable to realize a bigger circle of tasks) than the finite-state machine with an exit. The proof is obvious owing to examples 2 and 3, these articles and lemmas 1 (as in examples it is shown that there are tasks realized by SAMwE but which can't be realized means of finite-state machines with an exit).

Theorem 2: Turing's machine is more powerful than the store automatic machine with an exit. The proof follows from an example 4. Thus, in operation the concept of the store automatic machine with an output is entered, examples of its operation are shown. It is proved that by the opportunities the store automatic machine with an output is between finite-state state machines with an output and the Turing machine. Further store automatic machines with an output can be used in different application areas (in particular, in encoding (Zarubina *et al*., 2016b).

REFERENCES

Akho, A. and J. Ullman, 1978. Theory of Parse. Translation and Compilation. Vol. 1, Mir, Moscow, Pages: 613, (In Russian).

Bayrasheva, V.R., 1982. Extents of automatic transformations. Probabilistic Models Cybernetics, 18: 17-25.

Korneeva, N.N., 2011. Extents of asynchronously automatic transformations. Mathematics, 3: 30-40.

Lewis II, P.M. and R.E. Stearns, 1968. Syntax-directed transduction. J. ACM., 15: 465-488.

Meduna, A., 2000. Automata and Languages: Theory and Applications. Springer-Verlag, London, pp:916.

Rogers, H., 1972. Theory of Recursive Functions and Effective Computability. McGraw-Hill, New York, Pages: 624.

Shutzenberger, M.P., 1963. Context-free languages and pushdown automata. Inform. Control, 6: 246-264.

Sipser, M., 2012. Introduction to the Theory of Computation. 3rd Edn., Cengage Learning, Boston, pp: 458.

Zarubina, N.K., D.N. Zarubina, V.P. Dobritsa and A.A. Nozdrin, 2016a. Consecutive automatic shifrator. Series: Management, computer facilities, informatics. Medical instrument making. News of Southwest State University, pp: 36-39.

Zarubina, N.K., V.P. Dobrits and N.S. Ualiyev, 2016a. The store automatic machine with an output tape. Series: Management, computer facilities, informatics. Medical instrument making. News of Southwest State University, pp: 52-55.