problems with optimizing the size of an output regular term tree. By using our polynomial time algorithm, we have shown the regular term tree language with infinite edge labels is polynomial time inductively inferable from positive data.

We can give membership and MINL algorithms to the following two classes of regular term trees with no edge label: $\mathcal{RTTL}_1 = \{L(g) \mid g \in \mathcal{RTT}$ and the number of children of every vertex in g is not $2\}$ and $\mathcal{RTTL}_2 = \{L(g) \mid g \in \mathcal{RTT}$ and (i) for every pair of vertices in t whose degrees are more than 2, there exists a vertex of degree 2 on the path between them, and (ii) there is no vertex of degree 3 in t such that the distance between any leaf and the vertex is at least $2\}$. Therefore the classes are polynomial time inductively inferable from positive data. But it is an open question whether or not the class of all regular term tree languages with no edge label is polynomial time inductively inferable from positive data.

References

1. T. R. Amoth, P. Cull, and P. Tadepalli. Exact learning of unordered tree patterns from queries. *Proc. COLT-99, ACM Press*, pages 323–332, 1999.
2. D. Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Science*, 21:46–62, 1980.
3. H. Arimura, T. Shinohara, and S. Otsuki. Polynomial time algorithm for finding finite unions of tree pattern languages. *Proc. NIL-91, Springer-Verlag, LNAI 659*, pages 118–131, 1993.
4. S. Goldman and S. Kwek. On learning unions of pattern languages and tree patterns. *Proc. ALT-99, Springer-Verlag, LNAI 1720*, 1720:347–363, 1999.
5. S. Matsumoto, Y. Hayashi, and T. Shoudai. Polynomial time inductive inference of regular term tree languages from positive data. *Proc. ALT-97, Springer-Verlag, LNAI 1316*, pages 212–227, 1997.
6. T. Miyahara, T. Shoudai, T. Uchida, T. Kuboyama, K. Takahashi, and H. Ueda. Discovering new knowledge from graph data using inductive logic programming. *Proc. ILP-99, Springer-Verlag, LNAI 1634*, pages 222–233, 1999.
7. T. Miyahara, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda. Polynomial time matching algorithms for tree-like structured patterns in knowledge discovery. *Proc. PAKDD-2000, Springer-Verlag, LNAI 1805*, pages 5–16, 2000.
8. T. Miyahara, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda. Discovery of frequent tree structured patterns in semistructured web documents. *Proc. PAKDD-2001, Springer-Verlag, LNAI 2035*, pages 47–52, 2001.
9. S. Nestorov, S. Abiteboul, and R. Motwani. Extracting schema from semistructured data. *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 295–306, 1998.
10. T. Shinohara. Polynomial time inference of extended regular pattern languages. In *Springer-Verlag, LNCS 147*, pages 115–127, 1982.
11. T. Shoudai, T. Miyahara, T. Uchida, and S. Matsumoto. Inductive inference of regular term tree languages and its application to knowledge discovery. *Information Modelling and Knowledge Bases XI, IOS Press*, pages 85–102, 2000.
12. K. Wang and H. Liu. Discovering structural association of semistructured data. *IEEE Trans. Knowledge and Data Engineering*, 12:353–371, 2000.

Piecewise and Local Threshold Testability of DFA

A.N. Trahtman

Bar-Ilan University, Dep. of Math. and CS,
52900, Ramat Gan, Israel
trakht@macs.biu.ac.il

Abstract. The necessary and sufficient conditions for an automaton to be locally threshold testable are found. We introduce the polynomial time algorithm to verify local threshold testability of the automaton of time complexity $O(n^5)$ and an algorithm of order $O(n^3)$ for the local threshold testability problem for syntactic semigroup of the automaton. We modify necessary and sufficient conditions for piecewise testability problem for deterministic finite automaton and improve the Stern algorithm to verify piecewise testability for the automaton. The time complexity of the algorithm is reduced from $O(n^5)$ to $O(n^2)$. An algorithm to verify piecewise testability for syntactic semigroup of the automaton of order $O(n^2)$ is presented as well.

The algorithms have been implemented as a C/C^{++} package.

Keywords: automaton, locally threshold testable, piecewise testable, locally testable, transition graph, syntactic semigroup, algorithm

Introduction

The concept of local testability was introduced by McNaughton and Papert [11] and by Brzozowski and Simon [5]. Local testability can be considered as a special case of local l-threshold testability for $l = 1$.

Locally testable automata have a wide spectrum of applications. Regular languages and picture languages can be described by help of a strictly locally testable languages [4], [9]. Local automata (a kind of locally testable automata) are heavily used to construct transducers and coding schemes adapted to constrained channels [1]. Locally testable languages are used in the study of DNA and informational macromolecules in biology [8].

Kim, McNaughton and McCloskey [10] have found necessary and sufficient conditions of local testability and a polynomial time algorithm for local testability problem based on these conditions. The realization of the algorithm is described by Caron [6]. A polynomial time algorithm for local testability problem for semigroups was presented in [17].

The locally threshold testable languages were introduced by Beauquier and Pin [2]. These languages generalize the concept of locally testable language and have been studied extensively in recent years. An important reason to study locally threshold testable languages is the possibility of being used in pattern

R. Freivalds (Ed.): FCT 2001, LNCS 2138, pp. 347–358, 2001.

recognition [13]. Stochastic locally threshold testable languages, also known as $n-grams$ are used in pattern recognition, particular in speech recognition, both in acoustic-phonetics decoding as in language modeling [19].

For the state transition graph Γ of an automaton, we consider some subgraphs of the cartesian product $\Gamma \times \Gamma$ and $\Gamma \times \Gamma \times \Gamma$. In this way, necessary and sufficient conditions for a deterministic finite automaton to be locally threshold testable are found. It gives a positive answer on Caron's question [7]. We present here $O(n^5)$ time algorithm to verify local threshold testability of the automaton based on this characterization. By n is denoted here the sum of the nodes and edges of the graph of the automaton (n can be also considered as the product of the number of states by the size of the alphabet).

The local threshold testability problem for semigroup is, given a semigroup, to decide, if the semigroup is locally threshold testable or not. We present a polynomial time algorithm for this problem of order $O(n^3)$. By n is denoted here the size of the semigroup.

A language is piecewise testable iff its syntactic monoid is J-trivial [15].

Stern [14] modified these necessary and sufficient conditions and described a polynomial time algorithm to verify piecewise testability of deterministic finite automaton of order $O(n^5)$. The algorithm was implemented by Caron [6]. Our aim is to reduce the last estimation.

We modify necessary and sufficient conditions [15], [14] for the piecewise testability problem and describe an algorithm to verify piecewise testability of deterministic finite automaton and of his syntactic semigroup of order $O(n^2)$.

Necessary and sufficient conditions of local testability [10] are considered in this paper in terms of reachability in the graph $\Gamma \times \Gamma$. New version of $O(n^2)$ time algorithm to verify local testability based on this approach will be presented too.

The considered algorithms have been implemented as a part of C/C^{++} package TESTAS (testability of automata and semigroups).

Notation and Definitions

Let Σ be an alphabet and let Σ^+ denote the free semigroup on Σ. If $w \in \Sigma^+$, let $|w|$ denote the length of w. Let k be a positive integer. Let $i_k(w)$ $[t_k(w)]$ denote the prefix [suffix] of w of length k or w if $|w| < k$. Let $F_{k,j}(w)$ denote the set of factors of w of length k with at least j occurrences. A language L [a semigroup S] is called **l-threshold k-testable** if there is an alphabet Σ [and a surjective morphism $\phi : \Sigma^+ \to S$] such that for all $u, v \in \Sigma^+$, if $i_{k-1}(u) = i_{k-1}(v)$, $t_{k-1}(u) = t_{k-1}(v)$ and $F_{k,j}(u) = F_{k,j}(v)$ for all $j \leq l$, then either both u and v are in L or neither is in L $[u\phi = v\phi]$.

An automaton is **l-threshold k-testable** if the automaton accepts a l-threshold k-testable language [the syntactic semigroup of the automaton is **l-threshold k-testable**].

A language L [a semigroup, an automaton] is **locally threshold testable** if it is l-threshold k-testable for some k and l.

Piecewise testable languages are the finite boolean combinations of the languages of the form $A^*a_1A^*a_2A^*...A^*a_kA^*$ where $k \geq 0$, a_i is a letter from the alphabet A and A^* is a free monoid over A.

$|\Gamma|$ denotes the number of nodes of the graph Γ.

Γ^i denotes the direct product of i copies of the graph Γ.

The edge $\mathbf{p}_1, ..., \mathbf{p}_n \to \mathbf{q}_1, ..., \mathbf{q}_n$ in Γ^i is labeled by σ iff for each i the edge $\mathbf{p}_i \to \mathbf{q}_i$ in Γ is labeled by σ.

A maximal strongly connected component of the graph will be denoted for brevity as **SCC**, a finite deterministic automaton will be denoted as **DFA**. Arbitrary **DFA** is not necessary complete. A node from an SCC will be called for brevity as an **SCC − node**. **SCC − node** can be defined as a node that has a right unit in transition semigroup of the automaton.

The graph with trivial SCC is called **acyclic**.

If an edge $\mathbf{p} \to \mathbf{q}$ is labeled by σ then let us denote the node $\mathbf{q}$ as $\mathbf{p}\sigma$.

We shall write $\mathbf{p} \succeq \mathbf{q}$ if the node $\mathbf{q}$ is reachable from the node $\mathbf{p}$ or $\mathbf{p} = \mathbf{q}$.

In the case $\mathbf{p} \succeq \mathbf{q}$ and $\mathbf{q} \succeq \mathbf{p}$ we write $\mathbf{p} \sim \mathbf{q}$ (that is $\mathbf{p}$ and $\mathbf{q}$ belong to one SCC or $\mathbf{p} = \mathbf{q}$).

The **stabilizer** $\Sigma(\mathbf{q})$ of the node $\mathbf{q}$ from Γ is the subset of letters $\sigma \in \Sigma$ such that any edge from $\mathbf{q}$ labeled by σ is a loop $\mathbf{q} \to \mathbf{q}$.

Let $\Gamma(\Sigma_i)$ be the directed graph with all nodes from the graph Γ and edges from Γ with labels only from the subset Σ_i of the alphabet Σ.

So, $\Gamma(\Sigma(\mathbf{q}))$ is a directed graph with nodes from the graph Γ and edges from Γ that are labeled by letters from stabilizer of $\mathbf{q}$.

A semigroup without non-trivial subgroups is called **aperiodic** [2].

Let ρ be a binary relation on semigroup S such that for $a, b \in S$ $a\rho b$ iff for some idempotent $e \in S$ $ae = a$, $be = b$. Let λ be a binary relation on S such that for $a, b \in S$ $a\lambda b$ iff for some idempotent $e \in S$ $ea = a$, $eb = b$.

The unary operation x^ω assigns to every element x of a finite semigroup S the unique idempotent in the subsemigroup generated by x.

1 The Necessary and Sufficient Conditions of Local Threshold Testability

Let us formulate the result of Beauquier and Pin [2] in the following form:

Theorem 11 *[2] A language L is locally threshold testable if and only if the syntactic semigroup S of L is aperiodic and for any two idempotents e, f and elements a, u, b of S we have*

$$eafuebf = ebfueaf \tag{1.1}$$

Lemma 12 *Let the node $(\mathbf{p}, \mathbf{q})$ be an SCC-node of Γ^2 of a locally threshold testable DFA with state transition graph Γ and suppose that $\mathbf{p} \sim \mathbf{q}$.*

Then $\mathbf{p} = \mathbf{q}$.

Proof. The transition semigroup S of the automaton is finite and aperiodic [12]. Let us consider the node $(\mathbf{p},\mathbf{q})$ from SCC X of Γ^2. Then for some element $e \in S$ we have $\mathbf{q}e = \mathbf{q}$ and $\mathbf{p}e = \mathbf{p}$. In view of $\mathbf{q}e^i = \mathbf{q}$, $\mathbf{p}e^i = \mathbf{p}$ and finiteness of S we can assume e is an idempotent. In the SCC X for some a, b from S we have $\mathbf{p}a = \mathbf{q}$ and $\mathbf{q}b = \mathbf{p}$. Hence, $\mathbf{p}eae = \mathbf{q}$, $\mathbf{q}ebe = \mathbf{p}$. So $\mathbf{p}eaebe = \mathbf{p} = \mathbf{p}(eaebe)^i$ for any integer i. There exists a natural number n such that in the aperiodic semigroup S we have $(eae)^n = (eae)^{n+1}$. From theorem 11 it follows that for the idempotent e, $eaeebe = ebeeae$. We have $\mathbf{p} = \mathbf{p}eaebe = \mathbf{p}(eaeebe)^n = \mathbf{p}(eae)^n(ebe)^n = \mathbf{p}(eae)^{n+1}(ebe)^n = \mathbf{p}(eae)^n(ebe)^n eae = \mathbf{p}eae = \mathbf{q}$. So $\mathbf{p} = \mathbf{q}$.

Theorem 13 *For DFA* $\mathbf{A}$ *with state transition graph* Γ *the following three conditions are equivalent:*

1)$\mathbf{A}$ *is locally threshold testable.*

2)If the nodes $(\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)$ *and* $(\mathbf{q},\mathbf{r},\mathbf{t}_1,\mathbf{t})$ *are SCC-nodes of* Γ^3 *and* Γ^4, *correspondingly, and*

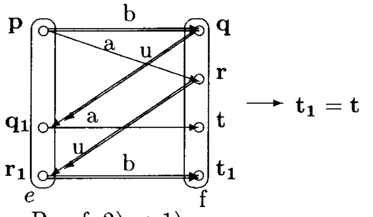
$(\mathbf{q},\mathbf{r}) \succeq (\mathbf{q}_1,\mathbf{r}_1)$, $(\mathbf{p},\mathbf{q}_1) \succeq (\mathbf{r},\mathbf{t})$, $(\mathbf{p},\mathbf{r}_1) \succeq (\mathbf{q},\mathbf{t}_1)$ *holds in* Γ^2
then $\mathbf{t} = \mathbf{t}_1$.

3)If the node $(\mathbf{w},\mathbf{v})$ *is an SCC-node of the graph* Γ^2 *and* $\mathbf{w} \sim \mathbf{v}$ *then* $\mathbf{w} = \mathbf{v}$.

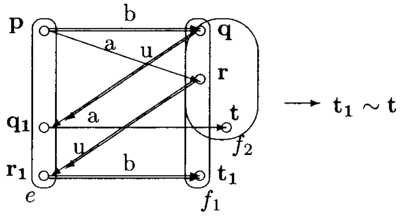
If the nodes $(\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)$, $(\mathbf{q},\mathbf{r},\mathbf{t})$, $(\mathbf{q},\mathbf{r},\mathbf{t}_1)$ *are SCC-nodes of the graph* Γ^3 *and*

$(\mathbf{q},\mathbf{r}) \succeq (\mathbf{q}_1,\mathbf{r}_1)$, $(\mathbf{p},\mathbf{q}_1) \succeq (\mathbf{r},\mathbf{t})$, $(\mathbf{p},\mathbf{r}_1) \succeq (\mathbf{q},\mathbf{t}_1)$ *hold in* Γ^2,
then $\mathbf{t} \sim \mathbf{t}_1$.

2)



3)



Proof. 2) → 1):

Let us consider the nodes $\mathbf{z}ebfueaf$ and $\mathbf{z}eafuebf$ where $\mathbf{z}$ is an arbitrary node of Γ, a, u, b are arbitrary elements from transition semigroup S of the automaton and e, f are arbitrary idempotents from S. Let us denote

$\mathbf{z}e = \mathbf{p}$, $\mathbf{z}ebf = \mathbf{q}$, $\mathbf{z}eaf = \mathbf{r}$, $\mathbf{z}eafue = \mathbf{r}_1$, $\mathbf{z}ebfue = \mathbf{q}_1$, $\mathbf{z}ebfueaf = \mathbf{t}$, $\mathbf{z}eafuebf = \mathbf{t}_1$.

By condition 2), we have $\mathbf{t} = \mathbf{t}_1$, whence $\mathbf{z}ebfueaf = \mathbf{z}eafuebf$. Thus, the condition $eafuebf = ebfueaf$ (1.1) holds for the transition semigroup S. By theorem 11, the automaton is locally threshold testable.

1) → 3):

If the node $(\mathbf{w},\mathbf{v})$ belongs to some SCC of the graph Γ^2 and $\mathbf{w} \sim \mathbf{v}$ then by lemma 12 local threshold testability implies $\mathbf{w} = \mathbf{v}$.

The condition $eafuebf = ebfueaf$ ((1.1), theorem 11) holds for the transition semigroup S of the automaton. Let us consider nodes $\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{t},\mathbf{q}_1,\mathbf{r}_1,\mathbf{t}_1$ satisfying the condition 3). Suppose

$$(\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)e = (\mathbf{p},\mathbf{q}_1,\mathbf{r}_1),\ (\mathbf{q},\mathbf{r},\mathbf{t})f_2 = (\mathbf{q},\mathbf{r},\mathbf{t}),\ (\mathbf{q},\mathbf{r},\mathbf{t}_1)f_1 = (\mathbf{q},\mathbf{r},\mathbf{t}_1)$$

for some idempotents $e, f_1, f_2 \in S$, and

$$(\mathbf{p},\mathbf{q}_1)a = (\mathbf{r},\mathbf{t}),\ (\mathbf{p},\mathbf{r}_1)b = (\mathbf{q},\mathbf{t}_1),\ (\mathbf{q},\mathbf{r})u = (\mathbf{q}_1,\mathbf{r}_1)$$

for some elements $a, b, u \in S$. Then $\mathbf{p}eaf_2 = \mathbf{p}eaf_1$ and $\mathbf{p}ebf_2 = \mathbf{p}ebf_1$.

We have $\mathbf{t}_1 f_2 = \mathbf{p}eaf_1uebf_1f_2$. By theorem 11, $\mathbf{p}ebf_jueaf_j = \mathbf{p}eaf_juebf_j$ for $j = 1, 2$. So we have $\mathbf{t}_1 f_2 = \mathbf{p}eaf_1uebf_1f_2 = \mathbf{p}ebf_1ueaf_1f_2$. In view of $\mathbf{p}ebf_2 = \mathbf{p}ebf_1$ and $f_i = f_if_i$ we have $\mathbf{t}_1 f_2 = \mathbf{p}ebf_2f_2ueaf_1f_2$. By theorem 11, $\mathbf{t}_1 f_2 = \mathbf{p}e(bf_2)f_2ue(af_1)f_2 = \mathbf{p}e(af_1)f_2ue(bf_2)f_2$. Now in view of $\mathbf{p}eaf_2 = \mathbf{p}eaf_1$ let us exclude f_1 and obtain $\mathbf{t}_1 f_2 = \mathbf{p}eaf_2uebf_2 = \mathbf{t}$. So $\mathbf{t}_1 f_2 = \mathbf{t}$. Analogously, $\mathbf{t}f_1 = \mathbf{t}_1$.

Hence, $\mathbf{t}_1 \sim \mathbf{t}$. Thus 3) is a consequence of 1).

3) → 2):

Suppose that $(\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)e = (\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)$, $(\mathbf{q},\mathbf{r},\mathbf{t},\mathbf{t}_1)f = (\mathbf{q},\mathbf{r},\mathbf{t},\mathbf{t}_1)$, for some idempotents e, f from transition semigroup S of the automaton and

$$(\mathbf{p},\mathbf{q}_1)a = (\mathbf{r},\mathbf{t}),\ (\mathbf{p},\mathbf{r}_1)b = (\mathbf{q},\mathbf{t}_1),\ (\mathbf{q},\mathbf{r})u = (\mathbf{q}_1,\mathbf{r}_1)$$

for some elements a, u, $b \in S$. Therefore

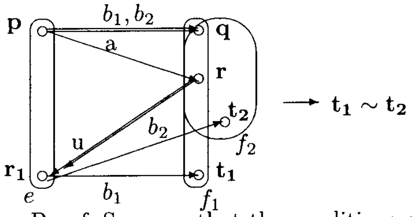
$$(\mathbf{p},\mathbf{q}_1)eaf = (\mathbf{p},\mathbf{q}_1)af = (\mathbf{r},\mathbf{t})$$
$$(\mathbf{p},\mathbf{r}_1)ebf = (\mathbf{p},\mathbf{r}_1)bf = (\mathbf{q},\mathbf{t}_1)$$
$$(\mathbf{q},\mathbf{r})u = (\mathbf{q},\mathbf{r})fue = (\mathbf{q}_1,\mathbf{r}_1)$$

for idempotents e, f and elements a, u, $b \in S$.

For $f = f_1 = f_2$ from 3) we have $\mathbf{t} \sim \mathbf{t}_1$. Notice that $(\mathbf{t}_1,\mathbf{t})f = (\mathbf{t}_1,\mathbf{t})$. The node $(\mathbf{t}_1,\mathbf{t})$ belongs to some SCC of the graph Γ^2 and $\mathbf{t} \sim \mathbf{t}_1$, whence by lemma 12, $\mathbf{t} = \mathbf{t}_1$.

Lemma 14 *Let the nodes* $(\mathbf{q},\mathbf{r},\mathbf{t}_1)$ *and* $(\mathbf{q},\mathbf{r},\mathbf{t}_2)$ *be SCC-nodes of the graph* Γ^3 *of a locally threshold testable DFA with state transition graph* Γ. *Suppose that* $(\mathbf{p},\mathbf{r}_1) \succeq (\mathbf{q},\mathbf{t}_1)$, $(\mathbf{p},\mathbf{r}_1) \succeq (\mathbf{q},\mathbf{t}_2)$ *in the graph* Γ^2 *and* $\mathbf{p} \succeq \mathbf{r} \succeq \mathbf{r}_1$.

Then $\mathbf{t}_1 \sim \mathbf{t}_2$.



Proof. Suppose that the conditions of the lemma hold but $\mathbf{t}_1 \not\sim \mathbf{t}_2$.

We have $(\mathbf{p},\mathbf{r}_1)e = (\mathbf{p},\mathbf{r}_1)$, $(\mathbf{q},\mathbf{r},\mathbf{t}_1)f_1 = (\mathbf{q},\mathbf{r},\mathbf{t}_1)$, $(\mathbf{q},\mathbf{r},\mathbf{t}_2)f_2 = (\mathbf{q},\mathbf{r},\mathbf{t}_2)$, for some idempotents e, f_2, f_2 from the transition semigroup S of the automaton and

$$(\mathbf{p},\mathbf{r}_1)b_1 = (\mathbf{q},\mathbf{t}_1),\ (\mathbf{p},\mathbf{r}_1)b_2 = (\mathbf{q},\mathbf{t}_2),\ \mathbf{p}a = \mathbf{r},\ \mathbf{r}u = \mathbf{r}_1$$

for some elements a, u, b_1, $b_2 \in S$.

If $\mathbf{t}_1 f_2 \sim \mathbf{t}_2$ and $\mathbf{t}_2 f_1 \sim \mathbf{t}_1$ then $\mathbf{t}_2 \sim \mathbf{t}_1$ in spite of our assumption. Therefore let us assume for instance that $\mathbf{t}_1 \not\sim \mathbf{t}_2 f_1$. (And so $\mathbf{t}_1 \neq \mathbf{t}_2 f_1$). This gives us an opportunity to consider $\mathbf{t}_2 f_1$ instead of $\mathbf{t}_2$. So let us denote $\mathbf{t}_2 = \mathbf{t}_2 f_1$, $f = f_1 = f_2$. Then $\mathbf{t}_2 f = \mathbf{t}_2$, $\mathbf{t}_1 f = \mathbf{t}_1$ and $\mathbf{t}_1 \not\sim \mathbf{t}_2$. Now

$$\mathbf{p}ea f = \mathbf{p}af = \mathbf{r}$$
$$(\mathbf{p},\mathbf{r}_1)eb_1f = (\mathbf{p},\mathbf{r}_1)b_1f = (\mathbf{q},\mathbf{t}_1)$$
$$(\mathbf{p},\mathbf{r}_1)eb_2f = (\mathbf{p},\mathbf{r}_1)b_2f = (\mathbf{q},\mathbf{t}_2)$$
$$\mathbf{r}u = \mathbf{r}ue = \mathbf{r}_1$$

Let us denote $\mathbf{q}_1 = \mathbf{q}ue$ and $\mathbf{t} = \mathbf{q}_1af_1$. Then

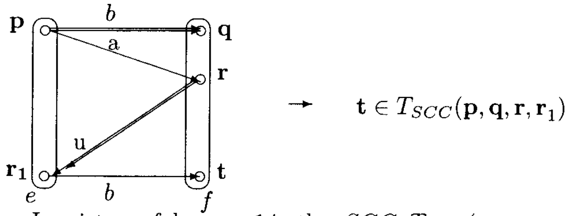
$(\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)e = (\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)$, $(\mathbf{q},\mathbf{r})ue = (\mathbf{q}_1,\mathbf{r}_1)$, $(\mathbf{q},\mathbf{r},\mathbf{t},\mathbf{t}_i)f = (\mathbf{q},\mathbf{r},\mathbf{t},\mathbf{t}_i)$

So the node $(\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)$ is an SCC-node of the graph Γ^3, the nodes $(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{t}_i)$ are SCC-nodes of the graph Γ^4 for $i = 1,2$ and we have $(\mathbf{q},\mathbf{r}) \succeq (\mathbf{q}_1,\mathbf{r}_1)$, $(\mathbf{p},\mathbf{q}_1) \succeq (\mathbf{r},\mathbf{t})$ and $(\mathbf{p},\mathbf{r}_1) \succeq (\mathbf{q},\mathbf{t}_i)$ for $i = 1,2$.

Therefore, by theorem 13, (2), we have $\mathbf{t}_1 = \mathbf{t}$ and $\mathbf{t}_2 = \mathbf{t}$. Hence, $\mathbf{t}_1 \sim \mathbf{t}_2$, contradiction.

Définition 15 *For any four nodes $\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1$ of the graph Γ of a DFA such that $\mathbf{p} \succeq \mathbf{r} \succeq \mathbf{r}_1$, $\mathbf{p} \succeq \mathbf{q}$ and the nodes $(\mathbf{p},\mathbf{r}_1)$, $(\mathbf{q},\mathbf{r})$ are SCC- nodes, let $T_{SCC}(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1)$ be the SCC of Γ containing the set*

$T(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1) := \{t \mid (\mathbf{p},\mathbf{r}_1) \succeq (\mathbf{q},\mathbf{t})$ and $(\mathbf{q},\mathbf{r},\mathbf{t})$ is an SCC-node$\}$



In virtue of lemma 14, the SCC $T_{SCC}(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1)$ of a locally threshold testable DFA is well defined (but empty if the set $T(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1)$ is empty). Lemma 14 and theorem 13 (3) imply the following theorem

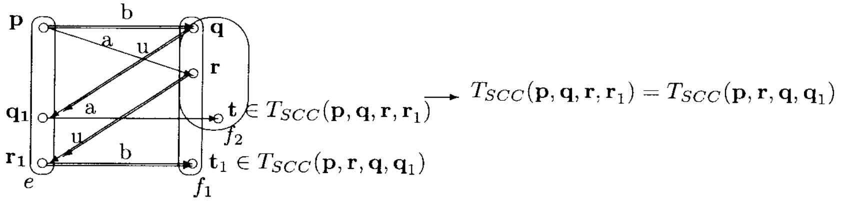
Theorem 16 *A DFA $\mathbf{A}$ with state transition graph Γ is locally threshold testable iff*

1)for every SCC-node $(\mathbf{p},\mathbf{q})$ of Γ^2 $\mathbf{p} \sim \mathbf{q}$ implies $\mathbf{p} = \mathbf{q}$ and

2)for every five nodes $\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{q}_1,\mathbf{r}_1$ of the graph Γ such that

- *the non-empty SCC $T_{SCC}(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1)$ and $T_{SCC}(\mathbf{p},\mathbf{r},\mathbf{q},\mathbf{q}_1)$ exist,*
- *the node $(\mathbf{p},\mathbf{q}_1,\mathbf{r}_1)$ is an SCC-node of the graph Γ^3,*
- *$(\mathbf{q},\mathbf{r}) \succeq (\mathbf{q}_1,\mathbf{r}_1)$ in Γ^2,*

holds $T_{SCC}(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1) = T_{SCC}(\mathbf{p},\mathbf{r},\mathbf{q},\mathbf{q}_1)$.



Let us go to the semigroup case.

Lemma 17 *Elements s,t from semigroup S belong to subsemigroup eSf where e and f are idempotents if and only if $s\rho t$ and $s\lambda t$.*

The proof follows from the definitions of ρ and λ.

Theorem 18 *A language L is locally threshold testable if and only if the syntactic semigroup S of L is aperiodic and for any three elements s, u, t of S such that $s\rho t$ and $s\lambda t$ we have*

$$sut = tus \qquad (1.2)$$

The proof follows from theorem 11 and lemma 17.

2 An Algorithm to Verify Local Threshold Testability of DFA

A linear depth-first search algorithm finding all SCC (see [16]) will be used.

By n will be denoted the sum of the nodes and edges of the graph.

2.1 To Check the Reachability on an Oriented Graph

For a given node $\mathbf{q_0}$, we consider depth-first search from the node. First only $\mathbf{q_0}$ will be marked. Every edge is crossed two times. Given a node, the considered path includes first the ingoing edges and then the outgoing edges. After crossing an edge in the positive direction from the marked node $\mathbf{q}$ to the node $\mathbf{r}$ we mark $\mathbf{r}$ too. The process is linear in the number of edges (see [10] for details).

The set of marked nodes forms a set of nodes that are reachable from $\mathbf{q_0}$. The procedure may be repeated for any node of the graph G.

The time of the algorithm for all pairs of nodes is $O(n^2)$.

2.2 To Verify Local Threshold Testability of DFA

Let us find all SCC of the graphs Γ, Γ^2 and Γ^3 and mark all SCC-nodes ($O(n^3)$ time complexity).

Let us recognize the reachability on the graph Γ and Γ^2 and form the table of reachability for all pairs of Γ and Γ^2. The time required for this step is $O(n^4)$.

Let us check the conditions of lemma 12. For every SCC-node $(\mathbf{p},\mathbf{q})$ $(\mathbf{p} \neq \mathbf{q})$ from Γ^2 let us check the condition $\mathbf{p} \sim \mathbf{q}$. A negative answer for any considered node $(\mathbf{p},\mathbf{q})$ implies the validity of the condition. In opposite case the automaton is not locally threshold testable. The time of the step is $O(n^2)$.

For every four nodes $\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1$ of the graph Γ, let us check the following conditions (see 15): $\mathbf{p} \succeq \mathbf{r} \succeq \mathbf{r}_1$ and $\mathbf{p} \succeq \mathbf{q}$. In a positive case, let us form SCC $T_{SCC}(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1)$ of all nodes $\mathbf{t} \in \Gamma$ such that $(\mathbf{p},\mathbf{r}_1) \succeq (\mathbf{q},\mathbf{t})$ and $(\mathbf{q},\mathbf{r},\mathbf{t})$ with $(\mathbf{p},\mathbf{r}_1)$ are SCC-nodes. In case that SCC T_{SCC} is not well defined the automaton is not threshold testable. The time required for this step is $O(n^5)$.

For every five nodes $\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{q}_1,\mathbf{r}_1$ from Γ we check now the second condition of theorem 16. If non-empty components $T_{SCC}(\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{r}_1)$ and $T_{SCC}(\mathbf{p},\mathbf{r},\mathbf{q},\mathbf{q}_1)$

exist, the node $(\mathbf{p}, \mathbf{q}_1, \mathbf{r}_1)$ is an SCC-node of the graph Γ^3 and $(\mathbf{q}, \mathbf{r}) \succeq (\mathbf{q}_1, \mathbf{r}_1)$ in Γ^2, let us verify the equality $T_{SCC}(\mathbf{p}, \mathbf{q}, \mathbf{r}, \mathbf{r}_1) = T_{SCC}(\mathbf{p}, \mathbf{r}, \mathbf{q}, \mathbf{q}_1)$. If the answer is negative then the automaton is not threshold testable. A positive answer for all considered cases implies the validity of the condition of the theorem. The time required for this step is $O(n^5)$.

The whole time of the algorithm to check the local threshold testability is $O(n^5)$.

3 Verifying Local Threshold Testability of Finite Semigroup

The algorithm is based on the theorem 18.

Let s_i be an element of the semigroup S, $n = |S|$.

- For any $s \in S$ let us find s^ω and check the aperiodity. The semigroup S is not locally threshold testable for non-aperiodic S.
- Let us form a binary square table L $[R]$ of the size n in the following way: For any $i, j \leq n$ suppose $L_{i,j} = 1$ $[R_{i,j} = 1$ $]$ if there exists an idempotent $e \in S$ such that $es_i = s_i$ and $es_j = s_j$ $[$ $s_ie = s_i$ and $s_je = s_j$ $]$. In opposite case $L_{i,j}[R_{i,j}] = 0$.
  The step has order $O(n^3)$.
- Let us find the intersection $\rho \cap \lambda$ and form a binary square table LR: $LR_{i,j} = L_{i,j}R_{i,j}$.
- For any triple $s_i, s_j, s_k \in S$ where $s_i(\rho \cap \lambda)s_j$, let us check the condition $s_is_ks_j = s_js_ks_i$. The validity of the condition for any triple of elements implies local threshold testability of S. In opposite case S is not locally threshold testable.
  The step has order $O(n^3)$.

The algorithm to verify local threshold testability of the semigroup S has order $O(n^3)$.

4 An Algorithm to Verify Local Testability of DFA

We present now necessary and sufficient conditions of local testability of Kim, McNaughton and McCloskey ([10]) in the following form:

Theorem 41 *([10]) A DFA with state transition graph Γ and transition semigroup S is locally testable iff the following two conditions hold:*

1)For any SCC-node $(\mathbf{p}, \mathbf{q})$ from Γ^2 such that $\mathbf{p} \sim \mathbf{q}$ we have $\mathbf{p} = \mathbf{q}$.

2)For any SCC-node $(\mathbf{p}, \mathbf{q})$ from Γ^2 such that $\mathbf{p} \succ \mathbf{q}$ and arbitrary element s from S we have $\mathbf{p}s \succeq \mathbf{q}$ is valid iff $\mathbf{q}s \succeq \mathbf{q}$.

The theorem implies

Corollary 42 *A DFA with state transition graph Γ over alphabet Σ is locally testable iff the following two conditions hold:*

1)For any SCC-node $(\mathbf{p}, \mathbf{q})$ from Γ^2 such that $\mathbf{p} \sim \mathbf{q}$ we have $\mathbf{p} = \mathbf{q}$.

2)For any node $(\mathbf{r}, \mathbf{s})$ and any SCC-node $(\mathbf{p}, \mathbf{q})$ from Γ^2 such that $(\mathbf{p}, \mathbf{q}) \succeq (\mathbf{r}, \mathbf{s})$, $\mathbf{p} \succeq \mathbf{q}$, $\mathbf{s} \succeq \mathbf{q}$ and for arbitrary σ from Σ we have $\mathbf{r}\sigma \succeq \mathbf{s}$ is valid iff $\mathbf{s}\sigma \succeq \mathbf{s}$.

In [10], a polynomial time algorithm for local testability problem was considered. Now we present another simplified version of such algorithm with the same time complexity.

Let us form a table of reachability on the graph Γ ($O(n^2)$ time complexity).

Let us find Γ^2 and all SCC-nodes of Γ^2.

For every SCC-node $(\mathbf{p}, \mathbf{q})$ $(\mathbf{p} \neq \mathbf{q})$ from Γ^2 let us check the condition $\mathbf{p} \sim \mathbf{q}$. ($O(n^2)$ time complexity). If the condition holds then the automaton is not locally testable (42).

Then we add to the graph new node $(\mathbf{0}, \mathbf{0})$ with edges from this node to every SCC-node $(\mathbf{p}, \mathbf{q})$ from Γ^2 such that $\mathbf{p} \succeq \mathbf{q}$. Let us consider first-depth search from the node $(\mathbf{0}, \mathbf{0})$.

We do not visit edges $(\mathbf{r}, \mathbf{s}) \rightarrow (\mathbf{r}, \mathbf{s})\sigma$ from the graph Γ^2 such that $\mathbf{s}\sigma \not\succeq \mathbf{s}$ and $\mathbf{r}\sigma \not\succeq \mathbf{s}$. In the case that from the two conditions $\mathbf{s}\sigma \succeq \mathbf{s}$ and $\mathbf{r}\sigma \succeq \mathbf{s}$ only one is valid the algorithm stops and the automaton is not locally testable.

In the case of absence of such cases on the path the automaton is locally testable. ($O(n^2)$ time complexity).

The whole time of the algorithm to check the local testability is $O(n^2)$.

5 Piecewise Testability

The following result is due to Simon:

Theorem 51 *[15] Let L be a regular language over the alphabet Σ and let Γ be the minimal automaton accepting L. The language L is piecewise testable if and only if the following conditions hold*

(i) Γ is a directed acyclic graph;

(ii) for any subset Σ_i from alphabet Σ each connected component of the graph $\Gamma(\Sigma_i)$ has a unique maximal state.

Lemma 52 *Let the state transition graph Γ of some DFA be acyclic. Suppose that for some subset Σ_i of the alphabet Σ the graph $\Gamma(\Sigma_i)$ has two distinct connected maximal nodes.*

Then for some node $\mathbf{p}$ from Γ the graph $\Gamma(\Sigma(\mathbf{p}))$ has also two distinct connected maximal nodes where the node $\mathbf{p}$ is one of these maximal nodes.

Proof. Suppose that the states $\mathbf{q}$ and $\mathbf{p}$ are distinct maximal nodes in some connected component X of the graph $\Gamma(\Sigma_i)$. The graph Γ of a piecewise testable deterministic finite automaton has only trivial SCC (theorem 51), whence $\mathbf{p} \not\succeq \mathbf{q}$

or $\mathbf{q} \not\succ \mathbf{p}$. Suppose that $\mathbf{q} \not\succ \mathbf{p}$ and let us consider the graph $\Gamma(\Sigma(\mathbf{p}))$. $\Gamma(\Sigma_i) \subset \Gamma(\Sigma(\mathbf{p}))$, whence the node $\mathbf{p}$ is a maximal node of the graph $\Gamma(\Sigma(\mathbf{p}))$. The states $\mathbf{q}$ and $\mathbf{p}$ are connected in the graph $\Gamma(\Sigma(\mathbf{p}))$ too. The node $\mathbf{q}$ or some successor of $\mathbf{q}$ is a maximal node in the same component of $\Gamma(\Sigma(\mathbf{p}))$ and this maximal node is not equal to $\mathbf{p}$ because $\mathbf{q} \not\succ \mathbf{p}$.

Last lemma gives us opportunity to present necessary and sufficient conditions for piecewise testability in the following form:

Theorem 53 *Let L be a regular language over the alphabet Σ and let Γ be a minimal automaton accepting L. The language L is piecewise testable if and only if the following conditions hold*

(i) Γ is a directed acyclic graph;

(ii) for any node $\mathbf{p}$ the maximal connected component C of the graph $\Gamma(\Sigma(\mathbf{p}))$ such that $\mathbf{p} \in C$ has a unique maximal state.

Let us formulate the Simon's result in the following form:

Theorem 54 . *Finite semigroup S is piecewise testable iff S is aperiodic and for any two elements x, $y \in S$ holds*

$$(xy)^\omega x = y(xy)^\omega = (xy)^\omega$$

6 An Algorithm to Verify Piecewise Testability for Automata

- Check that the graph Γ is acyclic. If not then the automaton is not piecewise testable and the procedure stops.
- For any state $\mathbf{p}$, let us compute
  - the stabilizer $\Sigma(\mathbf{p})$.
  - the graph $\Gamma(\Sigma(\mathbf{p}))$.
    The node $\mathbf{p}$ is a maximal node of some SCC of the graph $\Gamma(\Sigma(\mathbf{p}))$ and our aim is to find some more maximal nodes of the component or to prove their absence.
  - non-oriented copy N of the graph $\Gamma(\Sigma(\mathbf{p}))$ and maximal connected component C of N that contains $\mathbf{p}$. A linear depth-first search algorithm is used for to find C.
  - for any node $\mathbf{r}$ ($\mathbf{r} \neq \mathbf{p}$) from C let us do the following:
    If $\mathbf{r}$ has no outgoing edges in the graph $\Gamma(\Sigma(\mathbf{p}))$ or all its outgoing edges are loops then the automaton is not piecewise testable and the procedure stops.

Theorem 61 *Let L be a regular language over the alphabet Σ and let Γ be a minimal automaton accepting L. The considered algorithm to verify the piecewise testability of the language L has order $O(n^2)$ where n is the sum of nodes and edges of the graph Γ.*

Proof. The number of graphs $\Gamma(\Sigma(\mathbf{p}))$ we consider is not greater than the number of nodes of the graph Γ. The process of finding of the graph $\Gamma(\Sigma(\mathbf{p}))$ is linear in the number of nodes of the graph Γ.

Depth-first search algorithm is linear in n too.

The process of excluding loops in the graph $\Gamma(\Sigma(\mathbf{p}))$ is linear in the number of nodes of $\Gamma(\Sigma(\mathbf{p}))$.

The finding of node without outgoing edges is linear in the number of nodes.

7 An algorithm to Verify Piecewise Testability for Semigroups

The algorithm is based on the theorem 54.

- For any $x \in S$ let us find x^ω equal to x^i such that $x^i = x^{i+1}$. If x^ω is not found then the semigroup S is not piecewise testable.
  For given element x, the step is linear in the size of semigroup.
- For any pair $x, y \in ,S$, let us check the condition $(xy)^\omega x = y(xy)^\omega = (xy)^\omega$.
  The validity of the condition for any pair of elements implies piecewise testability of S. In opposite case S is not piecewise testable.

The algorithm under consideration has order $O(n^2)$ where $n = |S|$.

References

1. M.-P. Beal, J. Senellart, On the bound of the synchronization delay of local automata, *Theoret. Comput. Sci.* 205, **1-2**(1998), 297-306.
2. D. Beauquier, J.E. Pin, Factors of words, *Lect. Notes in Comp. Sci.* Springer, Berlin, **372**(1989), 63-79.
3. D. Beauquier, J.E. Pin, Languages and scanners, *Theoret. Comp. Sci.* 1, **84**(1991), 3-21.
4. J.-C. Birget, Strict local testability of the finite control of two-way automata and of regular picture description languages, *J. of Alg. Comp.* 1, **2**(1991), 161-175.
5. J.A. Brzozowski, I. Simon, Characterizations of locally testable events, *Discrete Math.* 4, (1973), 243- 271.
6. P. Caron, LANGAGE: A Maple package for automaton characterization of regular languages, Springer, *Lect. Notes in Comp. Sci.* **1436**(1998), 46-55.
7. P. Caron, Families of locally testable languages, *Theoret. Comput. Sci.*, **242**(2000), 361-376.
8. T. Head, Formal languages theory and DNA: an analysis of the generative capacity of specific recombinant behaviors, *Bull. Math. Biol.* **49**(1987), 4, 739-757.
9. F. Hinz, Classes of picture languages that cannot be distinguished in the chain code concept and deletion of redundant retreats, *Springer, Lect. Notes in Comp. Sci.* **349**(1990), 132-143.
10. S. Kim, R. McNaughton, R. McCloskey, A polynomial time algorithm for the local testability problem of deterministic finite automata, *IEEE Trans. Comput.* **40**(1991) N10, 1087-1093.
11. R. McNaughton, S, Papert, *Counter-free automata* M.I.T. Press. Mass., 1971.

12. J. Pin, Finite semigroups and recognizable languages. An introduction, Semigroups and formal languages, *Math. and Ph. Sci.* **1466**(1995), 1-32.
13. J. Ruiz, S. Espana, P. Garcia, Locally threshold testable languages in strict sense: Application to the inference problem. Springer, *Lect. Notes in Comp. Sci* **1433**(1998), 150-161.
14. J. Stern, Complexity of some problems from the theory of automata. Inf. and Control, 66(1985), 163-176.
15. I. Simon, Piecewise testable events, Springer, *Lect. Notes in Comp. Sci.*, **33**(1975), 214-222.
16. R.E. Tarjan, Depth first search and linear graph algorithms, *SIAM J. Comput.* **1**(1972), 146-160. *J. of Comp. System Sci.* **25**(1982), 360-376.
17. A.N. Trahtman, A polynomial time algorithm for local testability and its level. *Int. J. of Algebra and Comp.* v. 9, **1**(1998), 31-39.
18. A.N. Trahtman, A precise estimation of the order of local testability of a deterministic finite automaton, Springer, *Lect. Notes in Comp. Sci.* **1436**(1998), 198-212.
19. E. Vidal, F. Casacuberta, P. Garcia, Grammatical inference and automatic speech recognition. In *speech recognition and coding* Springer, 1995, 175-191.

Compositional Homomorphisms of Relational Structures

(Modeled as Multialgebras)

Michał Walicki*, Adis Hodzic[1], and Sigurd Meldal[2]

[1] University of Bergen, Dept. of Informatics, P.Box 7800, 5020 Bergen, Norway,
{michal,adis}@ii.uib.no
[2] CalPoly, Dept. of CS, 1 Grand Ave., San Luis Obispo, CA 93407, USA,
smeldal@phoenix.calpoly.edu

Abstract. The paper attempts a systematic study of homomorphisms of relational structures. Such structures are modeled as multialgebras (i.e., relation is represented as a set-valued function). The first, main, result is that, under reasonable restrictions on the form of the definition of homomorphism, there are exactly nine compositional homomorphisms of multialgebras. Then the comparison of the obtained categories with respect to the existence of finite limits and co-limits reveals two of them to be finitely complete and co-complete. Without claiming that compositionality and categorical properties are the only possible criteria for selecting a definition of homomorphism, we nevertheless suggest that, for many purposes, these criteria actually *might* be acceptable. For such cases, the paper gives an overview of the available alternatives and a clear indication of their advantages and disadvantages.

1 Background and Motivation

In the study of universal algebra, the central place occupies the pair of "dual" notions of congruence and homomorphism: every congruence on an algebra induces a homomorphism into a quotient and every homomorphism induces a congruence on the source algebra. Categorical approach attempts to express *all* (internal) properties of algebras in (external) terms of homomorphisms. When passing to relational structures or power set structures, however, the close correspondence of these internal and external aspects seems to get lost.

The most common, and natural, generalisation of the definition of homomorphism to relational structures says:

Definition 1.1 *A set function $\phi : \underline{A} \to \underline{B}$,*[1] *where both sets are equipped with respective relations $R^A \subseteq \underline{A}^n$ and $R^B \subseteq \underline{B}^n$, is a (weak) homomorphism iff*

$$\langle x_1 ... x_n \rangle \in R^A \;\Rightarrow \langle \phi(x_1) ... \phi(x_n) \rangle \in R^B$$

* The first author gratefully acknowledges the financial support from the Norwegian Research Council.

[1] Underlying sets will be used to indicate the "bare, unstructured sets" as opposed to power sets or other sets with structure. For the moment, one may ignorte this notational convention.

R. Freivalds (Ed.): FCT 2001, LNCS 2138, pp. 359–371, 2001.