Learning Finite State Models of Observable Nondeterministic Systems in a Testing Context

Khaled El-Fakih[1], Roland Groz[2], Muhammad Naeem Irfan[2] and Muzammil Shahbaz[3]

[1] American University of Sharjah
P. O. Box 26666 Sharjah, UAE
kelfakih@aus.edu
[2] Grenoble Universities
38402 Saint Martin d'Hères, France
{Irfan, Groz}@imag.fr
[3] Fraunhofer IESE
67663 Kaiserslautern, Germany
Muzammil.Shahbaz@iese.fraunhofer.de

Abstract. Learning models from test observations can be adapted to the case when the system provides nondeterministic answers. In this paper we propose an algorithm for inferring observable nondeterministic finite state machines (ONFSMs). The algorithm is based on Angluin *L** algorithm for learning DFAs. We define rules for constructing and updating learning queries taking into account the properties of ONFSMs. Application examples, complexity analysis and an experimental evaluation of the proposed algorithm are provided.

Keywords: Nondeterministic finite state machine, inferring algorithm, conformance testing.

1 Introduction

Inferring finite state models is an old yet still active research activity with many relevant application domains. In the last decade, it has in particular been applied in the field of software system testing [5]. Angluin [1] proposed an algorithm, called L^*, that learns a minimal DFA accepting the target regular language with a polynomial number of queries (wrt the size of alphabet and state space of the automaton). This algorithm consists in asking two types of queries, in particular, *membership* and *equivalence* queries. A membership query is asked by the learner to test whether a certain string over the given alphabet is actually in the given language. An equivalence query is asked by the learner to test whether a conjectured model is correct (i.e. equivalent or has the same language as the given language). In testing software systems, membership queries can be more adequately replaced by output queries. An output query consists in testing a system with a given sequence of inputs and observing the corresponding sequence of outputs.

Nondeterminism occurs due to various reasons such as performance, flexibility, limited controllability and abstraction. In this paper, we present an algorithm with polynomial complexity for learning observable nondeterministic finite state machines

Short Papers of the 22nd IFIP ICTSS, Alexandre Petrenko, Adenilso Simao, Jose Carlos Maldonado (eds.), Nov. 08-10, 2010, Natal, Brazil.

(ONFSM). This class corresponds to machines whose outputs to a given input string are nondeterministic, but whose state is uniquely determined by the observation of (inputs and) outputs (this type of nondeterminism is sometimes called output-nondeterminism). Since an ONFSM is structurally equivalent to a DFA on the alphabet of input/output couples, we compare the new algorithm with DFA inference. We also discuss the implications of implementing membership and output queries on output-nondeterministic systems. We rely on the "all-weather conditions" assumption [3]. To implement membership queries on I/O systems, we use filters defined by Niese [4], which take into account the fact that an output query can answer many membership queries at once.

2 Preliminaries

A *finite state machine* (FSM), simply called a *machine* throughout the paper, is a 5-tuple $N = \langle Q, I, O, h_Q, q_0 \rangle$, where Q is a finite nonempty set of states with q_0 as the initial state; I and O are input and output alphabets; and $h_Q \subseteq Q \times I \times O \times Q$ is a behavior relation. A nondeterministic FSM (NFSM) is *observable* if for each pair $(q, i) \in Q \times I$ and each output o there is at most one state $q' \in Q$ such that $(q, i, o, q') \in h_Q$. In this paper, we consider only complete (for each pair *(q,i)* there is a corresponding transition in h_Q) observable nondeterministic FSMs, or *ONFSM* for short. Fig 1 provides a small example for illustration.

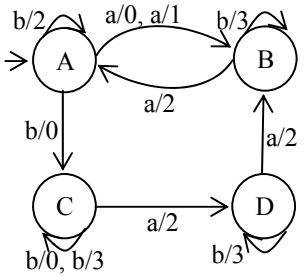

Fig 1. ONFSM over Q = {A,B,C,D}, I={a,b} and O={0,1,2,3}.

The different types of queries that will be used in the paper are defined as follows. For a DFA accepting an unknown language L over a given alphabet Σ, a *membership query* is a string $\alpha \in \Sigma^*$ to test whether α is in L. The output is either "yes" if $\alpha \in L$, or "no" otherwise. An *equivalence query* is a conjectured language L' and the output is "yes" if $L' = L$, or "no" otherwise. If the answer is no, then a counterexample α is returned, where α belongs to the symmetric difference of L and L'.

3 Inference of ONFSMs

Table 1. Initial observation table for ONFSM in Fig1.

		E	
		a	**b**
S_S	**ε/ε**	0, 1	0, 2
S_P	**a/0**	2	3
	a/1	2	3
	b/0	2	0,3
	b/2	0,1	0,2

An observation table consists of rows indexed by a prefix closed set Γ of I/O sequences, i.e., $\Gamma \subseteq (I \times O)^*$, columns indexed by suffix closed set E of input strings, and the content of a cell at the intersection of a row $\gamma \in \Gamma$ and a column $e \in E$ will be defined by $T(\gamma, e)$ belonging to the power set of O^+. A member of Γ is usually referred as α/β, or γ in short. We further restrict OT by defining $\Gamma = S_S \cup S_P$, such that $S_S \subseteq (I \times O)^*$ and $S_P = \{(\alpha i/\beta o): o \in T(\alpha/\beta, i)\}$, $\forall (\alpha/\beta \in Ss) \wedge (i \in I)$.

We define the equivalence of rows in OT with the help of the function T as follows. Two rows $\gamma_1, \gamma_2 \in \Gamma$ are *equivalent*, denoted by $\gamma_1 \cong \gamma_2$, iff $\forall e \in E$, $T(\gamma_1, e) = T(\gamma_2, e)$. A table is *closed* iff for each $\gamma_1 \in S_P$ there exists $\gamma_2 \in S_S$, such that $\gamma_1 \cong \gamma_2$. When the table is closed, then a complete *ONFSM* conjecture $C = \langle Q', I, O, h'_O, q'_0 \rangle$ can be constructed from the table as follows:

Definition 1 (Conjecture):

- $Q' = S_S$, i.e., the strings in S_S are the states of the conjecture
- $q'_0 = \varepsilon/\varepsilon \in S_S$ is the initial state of the conjecture
- For each $\alpha/\beta \in S_S$, $i \in I$ and $o \in T(\alpha/\beta, i)$; add $(\alpha/\beta, i, o, \gamma)$ into h'_O, where $\gamma \in S_S$ such that $\gamma \cong \alpha i/\beta o$

Theorem 1: If (Γ, E, T) is a closed observation table, then the conjecture $C = \langle Q', I, O, h'_O, q'_0 \rangle$, constructed as defined in Definition 1, is consistent with the finite function T. That is, for every $\alpha/\beta \in S_S$ and $e \in E$ such that $T(\alpha/\beta, e) = \{\beta_1, \ldots, \beta_d\}$, we have $(q'_0, \alpha e, \beta\beta_1, q'_1), \ldots, (q'_0, \alpha e, \beta\beta_d, q'_d) \in h'_O$.

Function Make-Closed(Γ, E, T)
Input: Observation table(Γ, E, T)
Output: Observation table(Γ, E, T)
Begin

Find $\gamma_1 \in S_P$ such that $\gamma_1 \ncong \gamma_2$, for all $\gamma_2 \in S_S$ and move γ_1 from S_P to S_S.
Extend (Γ, E, T) by adding $\alpha i/\beta o$, $\forall i \in I \wedge o \in T(\gamma_1 = \alpha/\beta, i)$, to S_P.
Ask output queries for the added rows in S_P and update (Γ, E, T) as follows:
$\forall$ added row $\alpha i/\beta o$ and $\forall e \in E$, ask output queries $\alpha i e$.
Update $T(\alpha i/\beta o, e)$ accordingly.

End

Algorithm 1: The algorithm L_N for inferring a complete ONFSM
Input: Black-box (unknown) complete ONFSM $N = \langle Q, I, O, h_O, q_0 \rangle$
Output: A complete minimal ONFSM Conjecture $C = \langle Q', I, O, h'_O, q'_0 \rangle$ that is equivalent to N
Begin
Initialize (Γ, E, T) with $S_S = \varepsilon/\varepsilon$, $S_P = \varnothing$, $\Gamma = S_S \cup S_P$, and $E = I$.
Ask output queries from (Γ, E, T) and update (Γ, E, T) as follows:
$\forall\ \alpha/\beta \in \Gamma, e \in E$, apply the output query $\alpha \cdot e$ to N.
Update $T(\alpha/\beta, e) = \{\beta_1, \ldots, \beta_d\}$, where $(q_0, \alpha, \beta, \gamma) \in h_O$ and $(\gamma, e, \beta_1, \gamma_1'), \ldots, (\gamma, e, \beta_d, \gamma_d') \in h_O$.
Update S_P as $S_P = \{i/o \mid i \in I \text{ and } o \in T(\varepsilon/\varepsilon, i)\}$ and complete (Γ, E, T) by asking output queries for all the rows in S_P.
while (Γ, E, T) is not closed
Make-Closed(Γ, E, T)
endwhile
Make the ONFSM conjecture C (as described in Definition 1)
Ask the equivalence query for C to the oracle
repeat
if the oracle replies with a counterexample α/β **then**
while α/β is a counterexample
for $j = 2$ **to** $|\alpha|$ **loop**
if $suffix^j(\alpha) \notin E$ **then**
add $suffix^j(\alpha)$ to E
construct the output queries for the new column
fill (Γ, E, T) by asking output queries
if (Γ, E, T) is not closed **then break** for loop **endif**
endif
endfor
while (Γ, E, T) is not closed
Make-Closed(Γ, E, T)
endwhile
make the conjecture C
endwhile
endif
until the oracle says "yes" to the conjecture C
Return the conjecture C from (Γ, E, T).
End

Theorem 2: If (Γ, E, T) is a closed observation table and the conjecture C is constructed, as defined in Definition 1, has n states, then any other complete ONFSM C' that has n or fewer states and also consistent with T is isomorphic to C.

4 Example

As an application example of the algorithm L_N, consider the "unknown" ONFSM N shown in Fig. 1 with four states labeled A (initial state), B, C, and D, and defined over inputs $\{a, b\}$ and outputs $\{0, 1, 2, 3\}$.

Table 2. Making the table 1 closed by moving the rows *a/0* and *b/0* to S_S

	a	**b**
ε/ε	0, 1	0, 2
a/0	2	3
b/0	2	0,3
a/1	2	3
b/2	0,1	0,2
aa/02	0,1	0,2
ab/03	2	3
ba/02	2	3
bb/00	2	0,3
bb/03	2	0,3

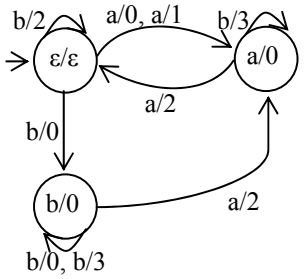

Fig 2. First ONFSM Conjecture

Table 3. Processing the counterexample *baab/0223*

	a	**b**	**ab**
ε/ε	0, 1	0, 2	03, 13
a/0	2	3	20, 22
b/0	2	0,3	23
a/1	2	3	22, 20
b/2	0,1	0,2	03, 13
aa/02	0,1	0,2	03, 13
ab/03	2	3	22,20
ba/02	2	3	23
bb/00	2	0,3	23
bb/03	2	0,3	23

Table 4. Making the table closed by adding the row *ba/02* to S_S

	a	b	ab
ε/ε	0, 1	0, 2	03, 13
a/0	2	3	20, 22
b/0	2	0,3	23
ba/02	2	3	23
a/1	2	3	22, 20
b/2	0,1	0,2	03, 13
aa/02	0,1	0,2	03, 13
ab/03	2	3	22,20
bb/00	2	0,3	23
bb/03	2	0,3	23
baa/022	2	3	22, 20
bab/023	2	3	23

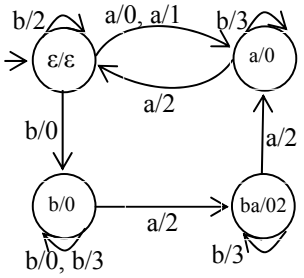

Fig 3 ONFSM Conjecture Table 4

5 Comparison of Algorithms

To have a significant set of experiments and assess the impact of various factors on the complexity (in particular $|I|$, n, m and d), we used large sets of randomly generated machines. Although L_N and L^* use different structures for observation tables, the "book-keeping" implied by the algorithms is overshadowed by the actual cost of interacting with black-box system, especially when testing nondeterministic systems.

When interacting with nondeterministic I/O systems, we may get different answers for the same input sequence. With the all-weather assumption, each output query is tried a number of times on the system, and the driver reports the set of all possible outputs. The cost of repeating the test may depend on the length of the query, but it

does not depend on the algorithm, so it will not be taken into account. The degree d of nondeterminism is defined as the average number of outgoing transitions for a given state and input. It is 1 for a deterministic machine. We will only consider values of d close to 1. For L^*, we use filters for membership queries [4], and *Suffix1by1* method to process the counterexamples [2] is used for both L^* and L_N algorithms. However, in Fig 2 data for L^* is also presented for Angluin method of processing counterexamples.

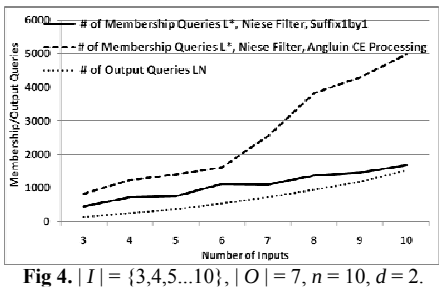

Fig 4. $|I| = \{3,4,5...10\}$, $|O| = 7$, $n = 10$, $d = 2$.

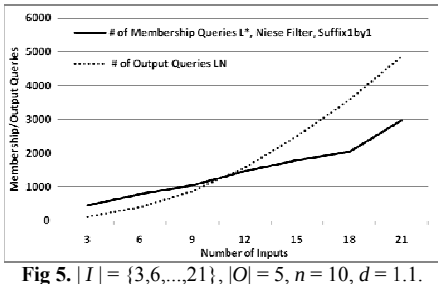

Fig 5. $|I| = \{3,6,...,21\}$, $|O| = 5$, $n = 10$, $d = 1.1$.

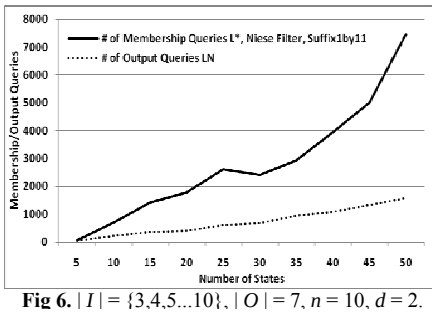

Fig 6. $|I| = \{3,4,5...10\}$, $|O| = 7$, $n = 10$, $d = 2$.

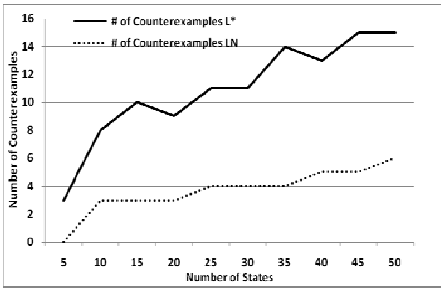

Fig 7. $|I| = 3$, $|O| = 2$, $n = \{5,10,15,...50\}$, $d = 1.1$.

6 Conclusion

Given a black-box (or unknown) ONFSM, the algorithm L_N infers an ONFSM that is equivalent to the unknown ONFSM via queries obtained using the "all-weather conditions" assumption. In most cases, L_N provides a significant improvement in performances when compared with approaches based on learning a DFA equivalent.

Possible extensions of this work involve learning NFSMs when "all-weather conditions" assumption cannot be satisfied, e.g. with limited controllability over SUT [6]. In this case learning has to be carried out using not the equivalence relation but other relations defined for NFSMs such as the separability and reduction relations.

References

1. D. Angluin: Learning Regular Sets from Queries and Counterexamples. Information and Computation: 2, 87--106 (1987).
2. M. N. Irfan, C. Oriat, R. Groz: Angluin Style Finite State Machine Inference with Non-optimal Counterexamples, Workshop on Model Inference In Testing 2010, ISSTA, Trento, Italy, 11-19 (2010).
3. R. Milner: A Calculus of Communicating Systems, Springer Verlag, ISBN 0-387-10235-3. (1980).
4. O. Neise. An Integrated Approach to Testing Complex Systems. Ph.D. Thesis, University of Dortmund (2003).
5. H. Raffelt, M: Merten, B. Steffen and T. Margaria: Dynamic Testing via Automata Learning. In International Journal on Software Tools for Technology Transfer: 307--324 (2009).
6. N. Shabaldina, K. El-Fakih, N. Yevtushenko: Testing Nondeterministic Finite State Machines with Respect to the Separability Relation. In TestCom/FATES: 305--318 (2007).