Analyzing Robustness of Angluin's L* Algorithm in Presence of Noise

Igor Khmelnitsky
Université Paris-Saclay
CNRS, ENS Paris-Saclay
INRIA, LMF, France
igor.khme@gmail.com

Serge Haddad
Université Paris-Saclay
CNRS, ENS Paris-Saclay
INRIA, LMF, France
haddad@lsv.fr

Lina Ye
Université Paris-Saclay
CNRS, ENS Paris-Saclay
CentraleSupélec, LMF, France
lina.ye@centralesupelec.fr

Benoît Barbot
Université Paris-Est Créteil
France
benoit.barbot@u-pec.fr

Benedikt Bollig
Université Paris-Saclay
CNRS, ENS Paris-Saclay, LMF, France
bollig@lsv.fr

Martin Leucker
Institute for Software Engineering and
Programming Languages
Universität zu Lübeck, Germany
leucker@isp.uni-luebeck.de

Daniel Neider
Carl von Ossietzky
University of Oldenburg
Germany
daniel.neider@uol.de

Rajarshi Roy
Max Planck Institute
for Software Systems
Kaiserslautern, Germany
rajarshi@mpi-sws.org

Angluin's L* algorithm learns the minimal (complete) deterministic finite automaton (DFA) of a regular language using membership and equivalence queries. Its probabilistic approximatively correct (PAC) version substitutes an equivalence query by a large enough set of random membership queries to get a high level confidence to the answer. Thus it can be applied to any kind of (also non-regular) device and may be viewed as an algorithm for synthesizing an automaton abstracting the behavior of the device based on observations. Here we are interested on how Angluin's PAC learning algorithm behaves for devices which are obtained from a DFA by introducing some noise. More precisely we study whether Angluin's algorithm reduces the noise and produces a DFA closer to the original one than the noisy device. We propose several ways to introduce the noise: (1) the noisy device inverts the classification of words w.r.t. the DFA with a small probability, (2) the noisy device modifies with a small probability the letters of the word before asking its classification w.r.t. the DFA, and (3) the noisy device combines the classification of a word w.r.t. the DFA and its classification w.r.t. a counter automaton. Our experiments were performed on several hundred DFAs.

Our main contributions, bluntly stated, consist in showing that: (1) Angluin's algorithm behaves well whenever the noisy device is produced by a random process, (2) but poorly with a structured noise, and, that (3) almost surely randomness yields systems with non-recursively enumerable languages.

1 Introduction

Discrete-event systems and their languages. Discrete-event systems [5] form a large class of dynamic systems that, given some internal state, evolve from one state to another one due to the occurrence of an event. For instance, discrete-event systems can represent a cyber-physical process whose events are triggered by a controller or the environment, or, a business process whose events are triggered by human activities or software executions. Often, the behaviors of such systems are classified as safe (aka correct, representative, etc.) or unsafe. Since a behavior may be identified by its sequence of occurred events, this leads to the notion of a language.

D. Della Monica and P. Ganty (Eds.): 13th International Symposium on
Games, Automata, Logics and Formal Verification (GandALF 22)
EPTCS 370, 2022, pp. 81–96, doi:10.4204/EPTCS.370.6

Analysis versus synthesis. There are numerous formalisms to specify (languages of) discrete-event systems. From a designer's perpective, the simpler it is the better its analysis will be. So finite automata and their languages (regular languages) are good candidates for the specification. However, even when the system is specified by an automaton, its implementation may slightly differ due to several reasons (bugs, unplanned human activities, unpredictable environment, etc.). Thus, one generally checks whether the implementation conforms to the specification. However, in many contexts, the system has already been implemented and the original specification (if any) is lost, as for instance in the framework of process mining [1]. Thus, by observing and interacting with the system, one aims to recover a specification close to the system at hand but that is robust with respect to its pathologic behaviors.

Language learning. The problem of learning a language from finite samples of strings by discovering the corresponding grammar is known as grammatical inference. Its significance was initially stated in [11] and an overview of very first results can be found in [4]. As it may not always be possible to infer a grammar that exactly identifies a language, approximate language learning was introduced in [13], where a grammar is selected from a solution space whose language approximates the target language with a specified degree of accuracy. To provide a deeper insight into language learning, the problem of identifying a (minimal) deterministic finite automaton (DFA) that is consistent with a given sample has attracted substantial attention in the literature since several decades [6, 2, 12]. An understanding of regular language learning is very valuable for a generalization to other more complex classes of languages.

Angluin's L* algorithm. Angluin's L* algorithm learns the minimal DFA of a regular language using membership and equivalence queries. Thus, one could try to adapt it to the synthesis task described above. However, for most black box systems, it is almost impossible to implement the equivalence query. Thus, its probabilistic approximatively correct (PAC) version substitutes an equivalence query with a large enough set of random membership queries. However, one needs to define and evaluate the accuracy of such an approach. Thus, here we are interested in how PAC Angluin's algorithm behaves for devices which are obtained from a DFA by introducing some noise.

Noisy learning. Most learning algorithms in the literature assume the correctness of the training data, including the example data such as attributes as well as classification results. However, sometimes noise-free datasets are not available. [10] carried out an experimental study of the noise effects on the learning performance. The results showed that generally the classification noise had more negative impact than the attribute one, i.e., errors in the values of attributes. [3] studied how to compensate for randomly introduced noise and discovered a theorem giving a bound on the smaple size that is sufficient for PAC-identification in the presence of classification noise when the concept classes are finite. Michael Kearns formalized another related learning model from statistical queries by extending Valiant's learning model [8]. One main result shows that any class of functions learnable from this statistical query model is also learnable with classification noise in Valiant's model.

Our contribution. In this paper, we study against which kinds of noise Angluin's algorithm[1] is *robust*. To the best of our knowledge, this is the very first attempt of noise analysis in the automata learning setting. More precisely, we consider the following setting (cf. Figure 1): Assume that a regular device $\mathscr{A}$ is given, typically as a black box. Due to some noise $\mathscr{N}$, the system $\mathscr{A}$ is pertubed resulting in a

[1] In this work by "Angluin's algorithm" we refer to the optimized version from [9].

not necessarily regular system $\mathscr{M}_{\mathscr{N}}$. This one is consulted by the PAC version of L^* to obtain a regular system $\mathscr{A}_E$. The question studied in this paper is whether $\mathscr{A}_E$ is closer to $\mathscr{A}$ than $\mathscr{M}_{\mathscr{N}}$, or, in other words, to which extent learning via L^* is robust against the noise $\mathscr{N}$. To this end, we introduce three kinds of

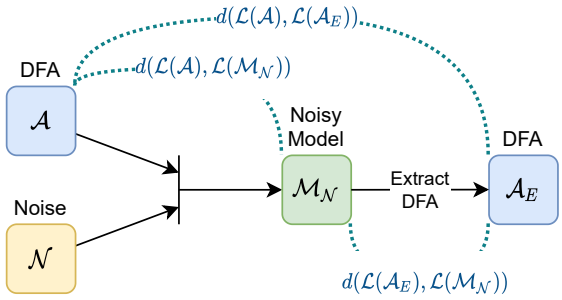

Figure 1: The experimental setup and the studied distances

noisy devices obtained from the DFA $\mathscr{A}$: (1) the noisy device is obtained by a random process from a given DFA by inverting the classification of words with a small probability, which corresponds to the classification noise in the classical learning setting, (2) the noisy device is obtained by a random process that, with a small probability, replaces each letter of a word by one chosen uniformly from the alphabet and then determines its classification based on the DFA, which corresponds to the attribute noise in the classical setting, and (3) the noisy DFA combines the classification of a word w.r.t. the DFA and its status w.r.t. a counter automaton. Our studies are based on the distribution over words that is used for generating words associated with membership queries and defining (and statistically measuring) the *distance* between two devices as the probability that they differ on word acceptance. We have performed experiments over several hundreds random DFA. We have pursued several goals along our experiments, expressed by the following questions:

- What is the threshold (in terms of distance) between pertubating the DFA or producing a device that is no more "similar to" the DFA?
- What is the impact of the nature of noise on the robustness of Angluin's algorithm?
- What is the impact of the words distribution on the robustness of Angluin's algorithm?

Due to the approximating nature of the PAC version of L^*, we had to consider the question of how to choose the accuracy of the approximate equivalence query to get a good trade-off between accuracy and efficiency. Moreover, since in most cases, Angluin's algorithm may perform a huge number of refinement rounds before a possible termination, we considered what a "good" number of rounds to stop the algorithm avoiding underfitting and overfitting is.

We experimentally show that w.r.t. the random noise, i.e., the noise introduced with a small probability in different ways, Angluin's algorithm behaves quite well, i.e., the learned DFA ($\mathscr{A}_E$) is very often closer to the original one ($\mathscr{A}$) than the noisy random device ($\mathscr{M}_{\mathscr{N}}$). When the noise is obtained using the counter automaton, Angluin's algorithm is not robust. Instead, the device $\mathscr{A}_E$ is closer to the noisy device $\mathscr{M}_{\mathscr{N}}$.

Moreover, we establish that the expectation of the length of a random word should be large enough to cover a relevant part of the set of words in order for Angluin's algorithms to be robust.

In order to understand why Angluin's algorithm is robust w.r.t. random noise we have undertaken a theoretical study establishing that almost surely the language of the noisy device ($\mathscr{M}_{\mathscr{N}}$) for case (1) and, with a further weak assumption, also for case (2) is not recursively enumerable. Considering non-recursively enumerable languages as unstructured, this means that due to the noise, the (regular) structure of $\mathscr{A}$ vanishes. This is not the case for the counter automaton setting. Altogether, to put it bluntly: the less structure the noisy device has, the better Angluin's algorithm works.

Organization. In Section 2, we introduce the technical background required for the robustness analysis. In Section 3, we detail the goals and the settings of our analysis. In Section 4, we provide and discuss the experimental results. In Section 5, we discuss randomness versus structure. Finally in Section 6, we draw our the conclusions and identify future work.

2 Preliminaries

Here we provide the technical background required for the robustness analysis.

Languages. Let Σ be an alphabet, i.e., a nonempty finite set, whose elements are called *letters*. A *word* w over Σ is a finite sequence over Σ, whose length is denoted by $|w|$. The unique word of length 0 is called the *empty word* and denoted by λ. As usual, Σ^* is the set of all words over Σ, and $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$ is the set of words of positive length. A *language* (over Σ) is any set $L \subseteq \Sigma^*$. The symmetric difference of languages $L_1, L_2 \subseteq \Sigma^*$ is defined as $L_1 \Delta L_2 = (L_1 \setminus L_2) \cup (L_2 \setminus L_1)$.

Words distribution and measure of a language. A distribution D over Σ^* is defined by a mapping $\mathbf{Pr}_D$ from Σ^* to $[0,1]$ such that $\sum_{w \in \Sigma^*} \mathbf{Pr}_D(w) = 1$. Let L be a language. Its probabilistic measure w.r.t. D, $\mathbf{Pr}_D(L)$ is defined by $\mathbf{Pr}_D(L) = \sum_{w \in L} \mathbf{Pr}_D(w)$.

Our analysis requires that we are able to efficiently sample a word according to some D. Thus we only consider distributions D_μ with $\mu \in]0,1[$, that are defined for a word $w = a_1 \dots a_n \in \Sigma^*$ by

$$\mathbf{Pr}_{D_\mu}(w) = \mu \left(\frac{1-\mu}{|\Sigma|} \right)^n .$$

To sample a random word according to D_μ in practice, we start with the empty word and iteratively we flip a biased coin with probability $1-\mu$ to add a letter (and μ to return the current word) and then uniformly select the letter in Σ.

Language distance. Given languages L_1 and L_2, their distance w.r.t. a distribution D, $d_D(L_1, L_2)$, is defined by $d_D(L_1, L_2) = \mathbf{Pr}_D(L_1 \Delta L_2)$. Computing the distance between languages is in most of the cases impossible. Fortunately whenever the membership problem for L_1 and L_2 is decidable, then using Chernoff-Hoeffding bounds [7], this distance can be statistically approximated as follows. Let $\alpha, \gamma > 0$ be an error parameter and a confidence level, respectively. Let S be a set of words sampled independently according to D, called a sampling, such that $|S| \geq \frac{\log(2/\gamma)}{2\alpha^2}$. Let $dist = \frac{|S \cap (L_1 \Delta L_2)|}{|S|}$. Then, we have

$$\mathbf{Pr}_D(|d_D(L_1, L_2) - dist| > \alpha) \ < \ \gamma.$$

Since we will not simultaneously discuss about multiple distributions, we omit the subscript D almost everywhere.

Finite Automata. A (complete) deterministic finite automaton (DFA) over Σ is a tuple $\mathscr{A} = (Q, \sigma, q_0, F)$ where Q is a finite set of states, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of final states, and $\sigma : Q \times \Sigma \to Q$ is the transition function. The transition function is inductively extended over words by $\sigma(q, \lambda) = q$ and $\sigma(q, wa) = \sigma(\sigma(q, w), a)$. The language of $\mathscr{A}$ is defined as $\mathscr{L}(\mathscr{A}) = \{w \in \Sigma^* \mid \sigma(q_0, w) \in F\}$. A language $L \subseteq \Sigma^*$ is called *regular* if $L = \mathscr{L}(\mathscr{A})$ for some DFA $\mathscr{A}$.

PAC version of Angluin's L* algorithm. Given a regular language L, Angluin's L* algorithm learns the unique mimimal DFA $\mathscr{A}$ such that $\mathscr{L}(\mathscr{A}) = L$ using only membership queries 'Does w belong to L?' and equivalence queries 'Does $\mathscr{L}(\mathscr{A}_E) = L$? and if not provide a word $w \in L \Delta \mathscr{L}(\mathscr{A}_E)$'. An abstract version of this algorithm is depicted by Algorithm 1. The main features of this algorithm are: a data structure *Data* from which `Synthetize`(*Data*) returns an automaton $\mathscr{A}_E$ and such that given a word $w \in L \Delta \mathscr{L}(\mathscr{A}_E)$, `Update`(*Data*, w) updates *Data*. The number of states of $\mathscr{A}_E$ is incremented by one after each round and so the algorithm terminates after its number of states is equal to the (unknown) number of states of $\mathscr{A}$.

The Probably Approximately Correct (PAC) version of Angluin's L* algorithm takes as input an error parameter ε and a confidence level δ, and replaces the equivalence query by a number of membership queries '$w \in L \Delta \mathscr{L}(\mathscr{A})$?' where the words are sampled from some distribution D unknown to the algorithm. Thus this algorithm can stop too early when all answers are negative while $L \neq \mathscr{L}(\mathscr{A})$. However due to the number of such queries which depends on the current round r (i.e., $\lceil \frac{log(1/\delta)+(r+1)\log(2)}{\varepsilon} \rceil$) this algorithm ensures that

$$\mathbf{Pr}_D(d_D(L, \mathscr{L}(\mathscr{A})) > \varepsilon) \; < \; \delta \, .$$

A key observation is that this algorithm could be used for every language L for which the membership problem is decidable. However since L is not necessarily a regular language, the algorithm might never stop and thus our adaptation includes a parameter *maxround* that ensures termination.

3 Robustness Analysis

3.1 Principle and goals of the analysis

Principle of the analysis. Figure 1 illustrates the whole process of our analysis. First we set the qualitative and quantitative nature of the noise ($\mathscr{N}$). Then we generate a set of random DFA ($\mathscr{A}$). Combining $\mathscr{A}$ and $\mathscr{N}$, one gets a noisy model $\mathscr{M}_{\mathscr{N}}$. More precisely, depending on whether the noise is random or not, $\mathscr{M}_{\mathscr{N}}$ is either generated off-line (deterministic noise) or on-line (random noise) when a membership query is asked during Angluin's L* algorithm. Finally we compare (1) the distances between $\mathscr{A}$ and $\mathscr{M}_{\mathscr{N}}$, and (2) between $\mathscr{A}$ and $\mathscr{A}_E$, the automaton returned by the algorithm. The aim of this comparison is to establish whether $\mathscr{A}_E$ is closer to $\mathscr{A}$ than $\mathscr{M}_{\mathscr{N}}$. In order to get a quantitative measure, we define the *information gain* as:

$$\text{Information gain} = \frac{d(\mathscr{L}(\mathscr{A}), \mathscr{L}(\mathscr{M}_{\mathscr{N}}))}{d(\mathscr{L}(\mathscr{A}), \mathscr{L}(\mathscr{A}_E))}$$

Algorithm 1: Angluin's L* algorithm

Input: L, a language unknown to the algorithm
Input: an integer *maxround* ensuring termination

`Angluin()`

Data: an integer r, a boolean b and a data structure *Data*
Output: a DFA $\mathscr{A}_E$

`Initialize(`*Data*`)`
$r \leftarrow 0$

`// The control of` *maxround* `is unnecessary when` L `is regular`
while $r <$ *maxround* **do**
    $\mathscr{A}_E \leftarrow$ `Synthetize(`*Data*`)`
    $(b, w) \leftarrow$ `IsEquivalent(`$\mathscr{A}_E$`)`
    if b **then return** $\mathscr{A}_E$
    `Update(`*Data*$, w$`)`
    $r \leftarrow r+1$
end
return `Synthesize(`*Data*`)`

We consider a **low** information gain to be in $[0, 0.9)$, a **medium** information gain to be in $[0.9, 1.5)$, and a **high** information gain to be in $[1.5, \infty)$. To make high information gain more evident, we set its threshold value as 1.5.

In addition, we also evaluate the distance between $\mathscr{A}_E$ and $\mathscr{M}_{\mathscr{N}}$ in order to study in which cases the algorithm learns in fact the noisy device instead of the original DFA.

Goals of the analysis.

- **Quantitative analysis.** The information gain highly depends on the 'quantity' of the noise, i.e., error rate. So we analyze the information gain depending of the distance between the original DFA and the noisy device and want to identify a threshold (if any) where the information gain starts to significantly increase.
- **Qualitative analysis.** Another important criterion of the information gain is the 'nature' of the noise. So we analyze the information gain w.r.t. the three noisy devices that we have introduced.
- **Impact of word distribution.** Finally, the robustness of the L* algorithm with respect to word distribution is also analyzed.

In order to perform relevant experiments, one needs to tune two critical parameters of Angluin's L* algorithm. Since the running time of the algorithm quadratically depends on the number of rounds (i.e. iterations of the loop), selecting an appropriate *maximal number of rounds* is a critical issue. We vary this maximal number of rounds and analyze how the information gain decreases w.r.t. this number. As an equivalence query is replaced with a set of membership queries whose number depends on the current round and the pair (ε, δ), it is thus interesting to study (1) what is the effect of *accuracy of the approximate equivalence queries*, i.e., the values of (ε, δ) on the ratio of executions that reach the maximal number of rounds and (2) compare the information gain for executions that stop before reaching this maximal number and the same execution when letting it run up to this maximal number.

3.2 Noise

A *random language* $R \subseteq \Sigma^*$ is determined by a random process: for each $w \in \Sigma^*$, membership $w \in R$ is determined independently at random, *once and for all*, according to some probability $\mathbf{Pr}(w \in R) \in [0,1]$. The probability $\mathbf{Pr}(w \in R)$ may depend on some parameters such as w itself and a given DFA.

We now describe the three kinds of noise that we analyze in this paper. Each type adds noise to a given DFA $\mathscr{A}$ in form of a random language R. For the first two types, *noise with output* and *noise with input*, the probability $\mathbf{Pr}(w \in R)$ of including $w \in \Sigma^*$ in R depends on w itself, $\mathscr{L}(\mathscr{A})$, and some parameter $0 < p < 1$. The third kind of noise, *counter DFA*, is actually *deterministic*, i.e., $\mathbf{Pr}(w \in R) \in \{0,1\}$ for all $w \in \Sigma^*$. In that case, the given DFA $\mathscr{A}$ determines a unique "noisy" language. Let us be more precise:

DFA with noisy output. Given a DFA $\mathscr{A}$ over the alphabet Σ and $0 < p < 1$, the random language $\mathscr{L}(\mathscr{A}^{\to p})$ flips the classification of words w.r.t. $\mathscr{L}(\mathscr{A})$ with probability p. More formally, for all $w \in \Sigma^*$,

$$\mathbf{Pr}(w \in \mathscr{L}(\mathscr{A}^{\to p})) = (1-p)\mathbf{1}_{w \in \mathscr{L}(\mathscr{A})} + p\mathbf{1}_{w \notin \mathscr{L}(\mathscr{A})}$$

where $\mathbf{1}_C$ is 1 if condition C holds, and 0 otherwise. Observe that the expected value of the distance $d(\mathscr{L}(\mathscr{A}), \mathscr{L}(\mathscr{A}^{\to p}))$ is p. Moreover, in our experiments, we observe that $\left|\frac{d(\mathscr{L}(\mathscr{A}), \mathscr{L}(\mathscr{A}^{\to p})) - p}{p}\right| < 5 \cdot 10^{-2}$ for all the generated languages.

DFA with noisy input. Given a DFA $\mathscr{A}$ over the alphabet Σ (with $|\Sigma| > 1$) and $0 < p < 1$, the random language $\mathscr{L}(\mathscr{A}^{\leftarrow p})$ changes every letter of the word with probability p uniformly to another letter and then returns the classification of the new word w.r.t. $\mathscr{L}(\mathscr{A})$. More formally, for $w = a_1 \dots a_n \in \Sigma^*$,

$$\mathbf{Pr}(w \in \mathscr{L}(\mathscr{A}^{\leftarrow p})) = \sum_{\substack{w' = b_1 \dots b_n \in \mathscr{L}(\mathscr{A}) \\ \text{s.t. } |w| = |w'|}} \prod_{1 \le i \le n} \left((1-p)\mathbf{1}_{a_i = b_i} + \frac{p}{|\Sigma| - 1} \mathbf{1}_{a_i \neq b_i} \right).$$

Counter DFA. Let $\mathscr{A}$ be a DFA over the alphabet Σ and $c : \Sigma \cup \{\lambda\} \to \mathbb{Z}$ be a function. We inductively define the function $\overline{c} : \Sigma^* \to \mathbb{Z}$ by

$$\overline{c}(\lambda) = c(\lambda) \text{ and } \overline{c}(wa) = \overline{c}(w) + c(a).$$

The counter language $\mathscr{L}(\mathscr{A}_c)$ is now given as

$$\mathscr{L}(\mathscr{A}_c) = \mathscr{L}(\mathscr{A}) \cup \{w \in \Sigma^* \mid \overline{c}(w) \le 0\}.$$

4 Experimental Evaluation

In order to empirically evaluate our ideas, we have implemented a prototype and benchmarks in Python, using the NumPy library. They are available on GitHub[2]. All evaluations were performed on a computer equipped by Intel i5-8250U CPU with 4 cores, 16GB of memory and Ubuntu Linux 18.03.

[2] `https://github.com/LeaRNNify/Noisy_Learning`

4.1 Generating DFAs

We now describe the settings of the experiments we made with three different types of noises. We choose $\mu = 10^{-2}$ for the parameter of the word distribution so that the average length of a random word is 99. All the statistic distances were computed using the Chernoff-Hoeffding bound [7] with $\alpha = 5 \cdot 10^{-4}$ as error parameter and $\gamma = 10^{-3}$ as confidence level.

The benchmarks were performed on DFA randomly generated using the following procedure. Let $M_q = 50$ and $M_a = 20$ be two parameters, which impose upper bounds on the number of states and of the alphabet, that could be tuned in future experiments. The DFA $\mathscr{A} = (Q, \sigma, q_0, F)$ on Σ is generated as follows:

- Uniformly choose $n_q \in [10, M_q]$ and $n_a \in [3, M_a]$;
- Set $Q = [0, n_q]$ and $\Sigma = [0, n_a]$;
- Uniformly choose $n_f \in [0, n_q - 1]$ and let $F = [0, n_f]$;
- Uniformly choose q_0 in Q;
- For all $(q, a) \in Q \times \Sigma$, choose the target state $\sigma(q, a)$ uniformly among all states.

The choice of M_q and M_a was inspired by observing that these values often occur when modeling realistic processes like in business process management.

4.2 Tunings

Before launching our experiments, we first tune two key parameters for both efficiency and accuracy purposes: the maximal number of rounds of the algorithm and the accuracy of the approximate equivalence query. This tuning is based on experiments over the DFA with the noisy output since the expected distance between the DFA and the noisy device is known (p), thus simplifying the tuning.

Maximal number of rounds. In order to specify a maximal number of rounds that lead to the good performances of the Angluin's Algorithm, we took a DFA with noisy output $\mathscr{A}^{\to p}$ for $p \in \{0.005, 0.0025, 0.0015, 0.001\}$. We ran the learning algorithm, stopping every 20 rounds to estimate the distance between the current DFA $\mathscr{A}_E$ to the original DFA $\mathscr{A}$. Figure 2 shows the evolution graphs of $d(\mathscr{L}(\mathscr{A}), \mathscr{L}(\mathscr{A}_E))$ w.r.t. the number of rounds according to the different values of p each of them summarizing five runs on five different DFAs. The vertical axis corresponds to the distance to original DFA $\mathscr{A}$, and the horizontal axis corresponds to the number of rounds. The red line is the distance with $\mathscr{A}^{\to p}$, and the blue line is the distance with $\mathscr{A}_E$. We observe that after about 250 rounds $d(\mathscr{L}(\mathscr{A}), \mathscr{L}(\mathscr{A}_E))$ is stabilizing except some rare peaks, which are worth further investigation. Therefore, from now on all the experiments are made with a maximum of 250 rounds. Of course this number depends on the size of $\mathscr{A}$ but for the variable size that we have chosen (between 10 and 50 states) it seems to be a good choice.

Accuracy of the approximate equivalence query. We have generated thirty-five DFA and for each of them we generated five $\mathscr{A}^{\to p}$ with different values of p. Table 1 summarizes our results with different ε and δ for the approximate equivalence query. The rows correspond to the value of the noise p, the columns correspond to the values of ε and δ (where we always choose $\varepsilon = \delta$) and each cell shows the average information gain. Looking at this table, $\varepsilon = \delta = 0.01$ and $\varepsilon = \delta = 0.005$ seem to be optimal values. We decided to fix $\varepsilon = \delta = 0.005$ for all our experiments.

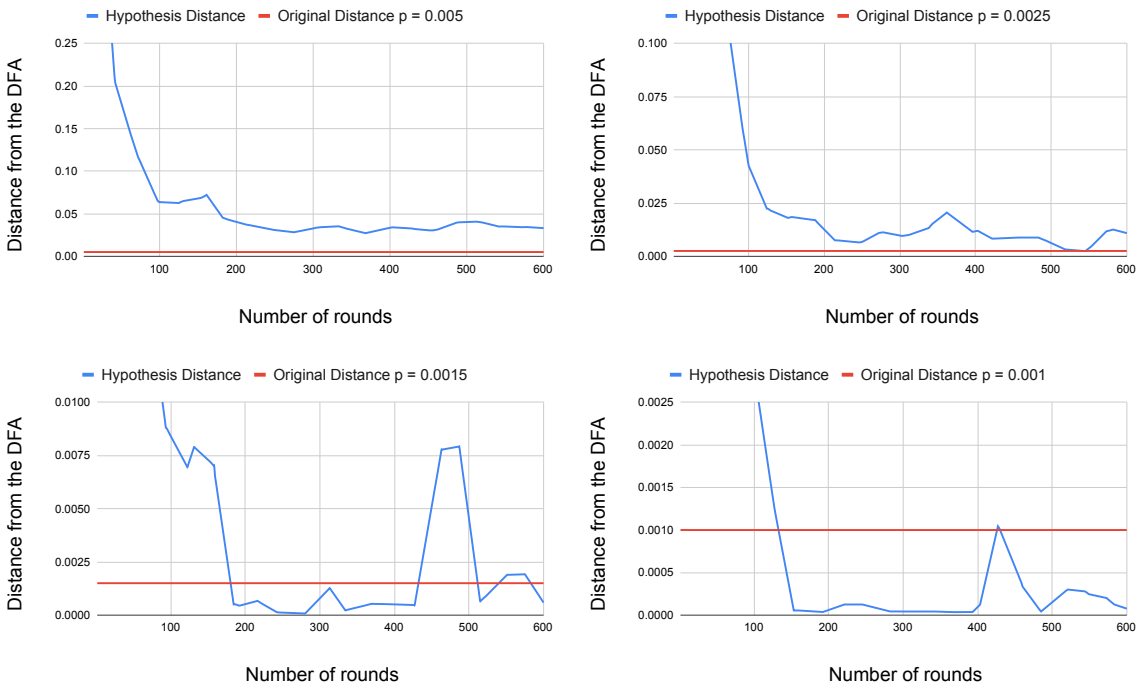

Figure 2: Number of rounds analysis

4.3 Qualitative and Quantitative analysis

For the three types of noise we have generated numerous DFA (as described shortly above), and for each DFA we have generated several noisy devices depending on the 'quantity' of noise. By computing the (average) information gain for all these experiments, we have been able to get conclusions about the effect of the nature and the quantity of the noise on the performance of Angluin's algorithm.

When Angluin's algorithm is applied to a noisy device, a corresponding random language is generated on-the-fly: once membership of a word in the target language has been determined (e.g., through a membership query), the corresponding truth value is stored and not changed anymore.

DFA with noisy output. We have generated fifty DFA, and for each such DFA $\mathscr{A}$, we have generated random languages with noisy output $\mathscr{L}(\mathscr{A}^{\to p})$ with five values for p between 0.01 and 0.001. Table 2

p \ $\varepsilon = \delta$	0.05	0.01	0.005	0.001	0.0005
0.01	0.081	0.054	0.047	0.048	0.050
0.005	0.086	0.087	0.072	0.070	0.094
0.0025	0.867	0.292	0.591	0.321	0.748
0.0015	1.401	2.933	3.082	0.980	0.710
0.001	5.334	4.524	3.594	1.811	6.440

Table 1: Evaluation of the impact of ε and δ.

summarizes the results. Recall that the expected value of $d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}^{\to p}))$ is p. We have identified a threshold for p between 0.0015 and 0.0025: if the noise is above 0.0025 the resulting DFA $\mathscr{A}_E$ has a bigger distance to the original one $\mathscr{A}$ than $\mathscr{A}^{\to p}$, and smaller if the noise is under 0.0015. Moreover, once we cross the threshold the robustness of the algorithm increases very quickly. We have also included a column that represents the standard deviation of the random variable $d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}_E))$ to assess that our conclusions are robust w.r.t. the probabilistic feature.

p	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}_E))$	$d(\mathscr{L}(\mathscr{A}^{\to p}),\mathscr{L}(\mathscr{A}_E))$	gain	standard deviation
0.01	0.12625	0.13320	0.07432	0.04102
0.005	0.04420	0.04827	0.11312	0.03366
0.0025	0.00333	0.00568	0.75031	0.00523
0.0015	0.00027	0.00174	5.52999	0.00047
0.001	0.00006	0.00103	15.75817	0.00007

Table 2: Evaluation of the algorithm w.r.t. the noisy output.

DFA with noisy input. We have generated forty-five random DFA, and for each such DFA $\mathscr{A}$, we have generated random languages $\mathscr{L}(\mathscr{A}^{\leftarrow p})$ with $p \in \{10^{-4}, 5\cdot 10^{-4}, 10^{-3}, 5\cdot 10^{-3}\}$. Contrary to the case of noisy output, p does not correspond to the expected value of $d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}^{\leftarrow p}))$. Thus we have evaluated this distance for every pair of the experiments and we have gathered the pairs whose distances belong to intervals that are described in the first column of Table 3. The second column of this table reports the number of pairs in the interval while the third one reports the average value of this distance for these pairs. Again we identify a threshold for $d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}^{\leftarrow p}))$ between 0.001 and 0.005 and once we cross the threshold the robustness of the algorithm increases very quickly.

Range	#	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}^{\leftarrow p}))$	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}_E))$	$d(\mathscr{L}\mathscr{A}^{\leftarrow p}),\mathscr{L}(\mathscr{A}_E))$	gain	standard deviation
$[0.025,1]$	36	0.04027	0.21513	0.22658	0.18	0.05279
$[0.005,0.025]$	53	0.00924	0.05416	0.06077	0.17	0.04172
$[0.002,0.005]$	33	0.00378	0.01260	0.01611	0.30	0.01783
$[0.001,0.002]$	11	0.00123	0.00030	0.00154	4.1	0.00058
$[0.0005,0.001]$	25	0.00079	0.00002	0.00082	39.5	0.00007

Table 3: Evaluation of the algorithm w.r.t. the noisy input.

Counter DFA. We have randomly generated the counter function as follows: We have uniformly chosen $c(\lambda)$ in $[0,|\Sigma|]$. Then, for all $a\in\Sigma$, $\mathbf{Pr}(c(a)=-1)=\frac{1}{4}$ and for all $0\le i\le 6$, $\mathbf{Pr}(c(a)=i)=\frac{3}{28}$.

We have generated 160 DFA. For each of them, we have generated a counter automaton (as described before). The results of our experiments are given in Table 4. Here whatever the quantity of noise the Angluin's algorithm is unable to get closer to the original DFA. Moreover the extracted DFA $\mathscr{A}_E$ is very often closer to the counter automaton $\mathscr{A}_c$ than the original DFA $\mathscr{A}$.

Thus we conjecture that when the noise is 'unstructured' and the quantity is small enough such that the word noise is still meaningful, then Angluin's algorithm is robust. On the contrary when the noise is structured, then Angluin's algorithm 'tries to learn' the noisy device whatever the quantity of noise. In

Range	#	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}_c))$	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}_E))$	$d(\mathscr{L}(\mathscr{A}_c),\mathscr{L}(\mathscr{A}_E))$	gain	standard deviation
[0.005, 0.025]	14	0.01238	0.02586	0.02053	0.47886	0.01898
[0.002, 0.005]	57	0.00245	0.00396	0.00262	0.61765	0.00298
[0.001, 0.002]	22	0.00143	0.00209	0.00121	0.68156	0.00126
[0.0005, 0.001]	20	0.00079	0.00108	0.00064	0.72481	0.00065
[0.0001, 0.0005]	44	0.00025	0.00035	0.00021	0.71054	0.00021

Table 4: Evaluation of the algorithm w.r.t. the 'noisy' counter.

Section 5, we will strengthen this conjecture establishing that in some sense noise produced by random process implies unstructured noise.

4.4 Words distribution

We now discuss the impact of word distribution on the robustness of the Angluin algorithm. The parameter μ determines the average length of a random word ($\frac{1}{\mu} - 1$). Table 5 summarizes experimental results with values of μ indicated on the first row. The other rows correspond to different values of the noise p for $\mathscr{A}^{\to p}$. The cells (at the intersection of a pair (p,μ)) contain the (average) information gain where experiments have been done over twenty-two DFA always eliminating the worst and best cases to avoid that the pathological cases perturb the average values. For values of p that matter (i.e., when the gain is greater than 1), there is clear tendency for the gain to first increase w.r.t. μ, reaching a maximum about $\mu = 0.01$ the value that we have chosen and then decrease. A possible explanation would be the following: too short words (i.e., big μ) does not help to discriminate between languages while too long words (i.e., small μ) lead to overfitting and does not reduce the noise.

p \ μ	0.001	0.005	0.01	0.05	0.1
0.01	0.059	0.067	0.078	0.184	0.317
0.005	0.078	0.130	0.134	0.559	0.966
0.0025	0.165	0.298	0.398	1.246	0.823
0.0015	0.465	0.671	2.267	2.074	1.651
0.001	1.801	10.94	8.907	3.753	2.341

Table 5: Analysis of different distributions on Σ^*

5 Random languages versus structured languages

Recall that in the precedent section, from the experimental results, we conjecture that Angluin's algorithm is robust, when the noise is random, i.e., unstructured, and its quantity is small enough, such as for DFA with noisy output and with noisy input. This is however not the case for structured counter DFA, for which Angluin's algorithm learns the noisy device itself instead of the original one whatever the quantity of noise.

In this section, we want to theoretically establish that the main factor of the robustness of the Angluin's L^* algorithm w.r.t. random noise is that almost surely randomness, in most cases, yields the perturbated

language that is unstructured. We consider a language as structured if it can be produced by some general device. Thus we identify the family of structured languages with the family of recursively enumerable languages. More precisely, we show that almost surely DFA with noisy output leads to a language that is not recursively enumerable. We then demonstrate further that with a mind condition, almost surely DFA with noisy input yields also non-recursively enumerable language. As for the counter DFA, by definition, it is clearly recursively enumerable, thus not being studied further.

The following lemma gives a simple means to establish that almost surely a random language is not recursively enumerable.

Lemma 1 *Let R be a random language over Σ. Let $(w_n)_{n\in\mathbb{N}}$ be a sequence of words of Σ^*. Let $W_n = \{w_i\}_{i<n}$ and $\rho_n = \max_{W\subseteq W_n} \mathbf{Pr}(R\cap W_n = W)$. Assume that $\lim_{n\to\infty}\rho_n = 0$. Then, for all countable families of languages $\mathscr{F}$, almost surely $R\notin\mathscr{F}$. In particular, almost surely R is not a recursively enumerable language.*

Proof Let us consider an arbitrary language L. Then, for all n, $\mathbf{Pr}(R=L)\leq\mathbf{Pr}(R\cap W_n = L\cap W_n)\leq\rho_n$. Thus, $\mathbf{Pr}(R=L)=0$ and $\mathbf{Pr}(R\in\mathscr{F}) = \sum_{L\in\mathscr{F}}\mathbf{Pr}(R=L) = 0$. ■

From Lemma 1, we immediately obtain that almost surely the noisy output perturbation of any language is not recursively enumerable. The proofs of the two next theorems use the same notations as those given in Lemma 1.

Theorem 1 *Let L be a language and $0<p<1$. Then almost surely $L^{\to p}$ is not a recursively enumerable language.*

Proof Consider any enumeration $(w_n)_{n\in\mathbb{N}}$ of Σ^* and any $W\subseteq W_n$. The probability that $L^{\to p}\cap W_n$ is equal to W is bounded by $\max(p,1-p)^n$. Thus, $\rho_n\leq\max(p,1-p)^n$ and $\lim_{n\to\infty}\rho_n = 0$. ■

We cannot get a similar result for the noisy input perturbation. Indeed consider the language Σ^*, whatever the kind of noise brought to the input, the obtained language is still Σ^*. With the kind of input noise that we study, consider the language that accepts words of odd length (see the automaton $\mathscr{A}'$ of Figure 3). Then the perturbed language is unchanged.

However given a DFA $\mathscr{A}$, we establish a mild condition on $\mathscr{A}$ ensuring that almost surely the random language $\mathscr{L}(\mathscr{A}^{\leftarrow p})$ is not recursively enumerable. We abbreviate bottom strongly connected components (of $\mathscr{A}$ viewed as a graph) by BSCC.

Definition 1 *Let $\mathscr{A} = (Q,F,\sigma,q_0)$ be a DFA. We call $\mathscr{A}$* equal-length-distinguishing *if there exist (possibly identical) BSCC $\mathscr{C},\mathscr{C}'$ of $\mathscr{A}$, $q_1\in\mathscr{C}\cap F$, $q_1'\in\mathscr{C}'\setminus F$, and $w,w'\in\Sigma^*$ such that we have $q_1=\sigma(q_0,w)$, $q_1'=\sigma(q_0,w')$, and $|w|=|w'|$.*

Theorem 2 *Let Σ be an alphabet with $|\Sigma|>1$. Let $\mathscr{A}=(Q,F,\sigma,q_0)$ be a DFA over Σ, $0<p<1$ and $\mathscr{C},\mathscr{C}'$ some BSCC of $\mathscr{A}$ (possibly equal). Assume that $\mathscr{A}$ is equal-length-distinguishing. Then almost surely $\mathscr{L}(\mathscr{A}^{\leftarrow p})$ is not a recursively enumerable language.*

Proof Let us denote $\ell=|w|$ and m (resp. m') the periodicity of $\mathscr{C}$ (resp. $\mathscr{C}'$). Moreover, let $a\in\Sigma$. We build a Markov chain $\mathscr{M}$ from $\mathscr{C}$ as follows: every transition $q\xrightarrow{a}q'$ has probability $1-p$ and for all $b\neq a$, every transition $q\xrightarrow{b}q'$ has probability $\frac{p}{|\Sigma|-1}$. We proceed similarly from $\mathscr{C}'$ to build $\mathscr{M}'$.

Let us denote α_n (resp. α_n') the probability in $\mathscr{M}$ (resp. $\mathscr{M}'$) that starting from q_1 (resp. q_1'), the current state at time n is q_1 (resp. q_1'). Since $\mathscr{M}$ and $\mathscr{M}'$ are irreducible, $\lim_{n\to\infty}\alpha_{mn}$ (resp. $\lim_{n\to\infty}\alpha'_{m'n}$) exists and is positive. Let us denote α (resp. α') this limit. There exists n_0 such that for all $n\geq n_0$, $\alpha_{mn}\geq\frac{\alpha}{2}$ and $\alpha'_{m'n}\geq\frac{\alpha'}{2}$.

Define $w_n = wa^{mm'(n+n_0)}$ for all $n \in \mathbb{N}$. The probability that w_n is accepted by $\mathscr{L}(\mathscr{A})^{\leftarrow p}$ is lower bounded by the probability that the prefix w is unchanged (thus reaching q_1) and that after $mm'(n+n_0)$ steps the current state in $\mathscr{M}$ is q_1. So a lower bound is: $\min(p,1-p)^\ell \frac{\alpha}{2}$.

The probability that w_n is rejected by $\mathscr{L}(\mathscr{A})^{\leftarrow p}$ is lower bounded by the probability that the prefix w is changed into w' (thus reaching q'_1) and that after $mm'(n+n_0)$ steps the current state in $\mathscr{M}'$ is q'_1. So a lower bound is: $\min(p,1-p)^\ell \frac{\alpha'}{2}$.

Let $W \subseteq W_n$. The probability that $L^{\leftarrow p} \cap W_n$ is equal to W is upper bounded by:

$$\left(1 - \min(p,1-p)^\ell \frac{\min(\alpha,\alpha')}{2}\right)^n$$

Thus $\rho_n \leq \left(1 - \min(p,1-p)^\ell \frac{\min(\alpha,\alpha')}{2}\right)^n$ and $\lim_{n\to\infty} \rho_n = 0$. ■

The DFA $\mathscr{A}$ of Figure 3 that represents the formula 'a Until b' of temporal logic LTL is equal-length-distinguishing. The corresponding pair of states consists of the accepting state and the leftmost one. Checking the hypotheses of this theorem can be done in quadratic time by first building a graph whose set of vertices is $Q \times Q$ and there is an edge $(q_1,q_2) \to (q'_1,q'_2)$ if there are some transitions $q_1 \xrightarrow{a_1} q'_1$ and $q_2 \xrightarrow{a_2} q'_2$ and then looking for a vertex (q_1,q_2) in some BSCC with $q_1 \in F$ and $q_2 \notin F$ reachable from (q_0,q_0).

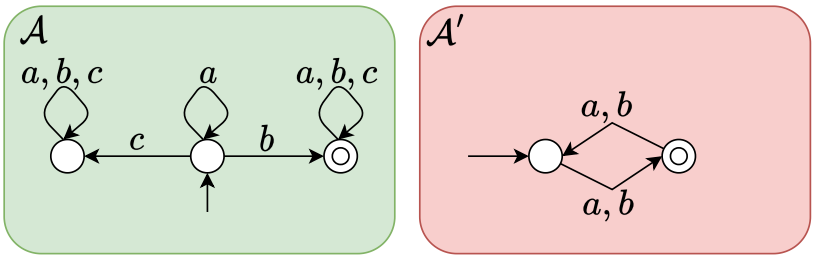

Figure 3: Two DFA

We have that the property of being equal-length-distinguishing is a sufficient condition for ensuring that almost surely $\mathscr{L}(\mathscr{A}^{\leftarrow p})$ is not a recursively enumerable language. So we want to investigate whether it is a necessary condition. The next proposition shows a particular case when this condition is necessary.

Proposition 1 *Let Σ be an alphabet with $|\Sigma| > 1$. Let $\mathscr{A} = (Q,F,\sigma,q_0)$ be a DFA that is not equal-length-distinguishing and such that every circuit of $\mathscr{A}$ belongs to a BSCC. Then, for every sampling L' of $\mathscr{L}(\mathscr{A}^{\leftarrow p})$, L' is regular.*

Proof Pick some $n_0 \in \mathbb{N}$ such that for all w with $|w| \geq n_0$ and $q_0 \xrightarrow{w} q$ implies that q belongs to some BSCC. Observe now that, since $\mathscr{A}$ is not equal-length-distinguishing, for words w,w' with $|w| = |w'| \geq n_0$, $w \in L$ iff $w' \in L$. Thus, for every sampling L' of $\mathscr{L}(\mathscr{A}^{\leftarrow p})$, $L' = (L' \cap \Sigma^{<n_0}) \cup (L \cap \Sigma^{\geq n_0})$ implying that L' is regular. ■

Observe that we establish the next proposition using a generalization of Lemma 1.

Proposition 2 *Let $\mathscr{A}$ be the DFA of Figure 4. Then, $\mathscr{A}$ is not equal-length-distinguishing. Moreover, almost surely $\mathscr{L}(\mathscr{A}^{\leftarrow \frac{2}{3}})$ is not recursively enumerable.*

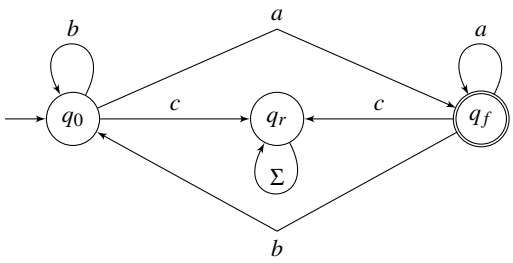

Figure 4: A DFA $\mathscr{A}$ with $\mathscr{L}(\mathscr{A}) = (a+b)^*a$

Proof There is a single BSCC with a single state $\{q_r\}$. So $\mathscr{A}$ is not equal-length-distinguishing. Let $w \neq \lambda$ be a word with $|w| = n$ and denote $\tilde{w}$ the random word obtained by the noisy perturbation. Observe that every letter of $\tilde{w}$ is uniformly distributed over Σ. So the probability that $\tilde{w}$ does not contain a c is $(\frac{2}{3})^n$ and the conditional probability that $\tilde{w}$ belongs to $\mathscr{L}(\mathscr{A}^{\leftarrow\frac{2}{3}})$ knowing that it does not contain a c is $\frac{1}{2}$.

Fix some $0 < \rho < 1$. The probability that for all words $w \in \Sigma^n$, $\tilde{w}$ contains a c is equal to $(1-(\frac{2}{3})^n)^{3^n} \leq e^{-2^n}$. Pick an increasing sequence $(n_k)_{k\in\mathbb{N}}$ such $\sum_{k\in\mathbb{N}} e^{-2^{n_k}} \leq 1-\rho$. Then with probability at least ρ, for all k, there is a word $w_k \in \Sigma^{n_k}$ such that $\tilde{w}_k$ does not contain a c. Letting ρ go to 1, almost surely there is an infinite number of words w such that $\tilde{w} \in (a+b)^+$.

Let us consider an arbitrary language L' and $(w_n)_{n\in\mathbb{N}}$ be an enumeration of Σ^+. Then almost surely there is an infinite number of w_n such that $\tilde{w}_n$ belong to $(a+b)^+$. Recall that for such a word, the probability that it belongs to $\mathscr{L}(\mathscr{A}^{\leftarrow\frac{2}{3}})$ is equal to $\frac{1}{2}$. Let W_n be the *random* set of the first n^{th} such words. Then for all n, $\mathbf{Pr}(L' = \mathscr{L}(\mathscr{A}^{\leftarrow\frac{2}{3}})) \leq \mathbf{Pr}(L' \cap W_n = \mathscr{L}(\mathscr{A}^{\leftarrow\frac{2}{3}}) \cap W_n) = 2^{-n}$.

Thus $\mathbf{Pr}(L' = \mathscr{L}(\mathscr{A}^{\leftarrow\frac{2}{3}})) = 0$ and $\mathbf{Pr}(\mathscr{L}(\mathscr{A}^{\leftarrow\frac{2}{3}}) \in \mathscr{F}) = \sum_{L'\in\mathscr{F}} \mathbf{Pr}(L' = \mathscr{L}(\mathscr{A}^{\leftarrow\frac{2}{3}})) = 0$ for $\mathscr{F}$ a countable family of languages. ■

To show the soundness of the structural criterion in Theorem 2 with experiments and comparisons, we have refined our experiments on DFA with noisy inputs partitioning the randomly generated DFA depending on whether they are equal-length-distinguishing.

We have chosen $|\Sigma| = 3$ since with greater size, it was difficult to generate DFAs that do not satisfy the hypotheses. Tables 6 and 7 summarize these experiments. The last rows of the tables (where the information gain is greater than one) confirm our conjecture.

Range	#	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}^{\leftarrow p}))$	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}_E))$	$d(\mathscr{L}(\mathscr{A}^{\leftarrow p}),\mathscr{L}(\mathscr{A}_E))$	gain
$[0.005, 0.025]$	85	0.01114	0.03604	0.04345	0.30902
$[0.002, 0.005]$	81	0.00338	0.00421	0.00747	0.80443
$[0.001, 0.002]$	25	0.00142	0.00035	0.00174	4.09784
$[0.0005, 0.001]$	16	0.00071	0.00006	0.00077	11.08439

Table 6: Experiments on equal-length-distinguishing DFA

Range	#	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}^{\leftarrow p}))$	$d(\mathscr{L}(\mathscr{A}),\mathscr{L}(\mathscr{A}_E))$	$d(\mathscr{L}(\mathscr{A}^{\leftarrow p}),\mathscr{L}(\mathscr{A}_E))$	gain
$[0.005, 0.025]$	36	0.01089	0.02598	0.03410	0.41905
$[0.002, 0.005]$	49	0.00308	0.00387	0.00646	0.79628
$[0.001, 0.002]$	35	0.00136	0.00057	0.00182	2.39863
$[0.0005, 0.001]$	36	0.00075	0.00063	0.00130	1.18583

Table 7: Experiments on non equal-length-distinguishing DFA

6 Conclusion

We have studied how the PAC-version of Angluin's algorithm behaves for devices which are obtained from a DFA by introducing noise. More precisely, we have investigated whether Angluin's algorithm reduces the noise producing a DFA closer to the original one than the noisy device. We have considered three kinds of noise belonging either to random noise or to structured noise. We have shown that, on average, Angluin's algorithm behaves well for random noise but not for structured noise. We have completed our study by establishing that almost surely the random noisy devices produce a non recursively enumerable language confirming the relevance of the structural criterion for robustness of Angluin's algorithm.

There are several directions for future work. First the algorithm could be tuned in a more precise way. In addition to stop when the maximal number of rounds is reached or the current automaton is declared equivalent, we could add early stopping when after some stage with distance decreasing the distance stabilizes. This would produce smaller DFA possibly closer to the original DFA. At longer term, Angluin's algorithm has no information about the original DFA. It would be interesting to introduce a priori knowledge and design more efficient algorithms. For instance, the algorithm could take as input the maximal size of the original DFA or a regular language that is a superset of the original language. In our setting the noise resulted in a noisy device which, once obtained, answers membership queries deterministically. A completely different form of noise would be that the answer to a query is randomly noisy meaning that for the same repeated query, different answers could occur.

References

[1] Wil M. P. van der Aalst (2012): *Process mining*. *CACM* 55(8), pp. 76–83, doi:10.1145/2240236.2240257.

[2] Dana Angluin (1987): *Learning Regular Sets from Queries and Counterexamples*. *Inf. Comput.* 75(2), pp. 87–106, doi:10.1016/0890-5401(87)90052-6.

[3] Dana Angluin & Philip D. Laird (1987): *Learning From Noisy Examples*. *Mach. Learn.* 2(4), pp. 343–370, doi:10.1023/A:1022873112823.

[4] Alan W. Biermann & Jerome A. Feldman (1972): *A survey of results in grammatical inference*. In S. Watanabe, editor: *Frontiers of Pattern Recognition*, Academic Press, New York, pp. 31–54, doi:10.1016/B978-0-12-737140-5.50007-5.

[5] Christos G. Cassandras & Stephane Lafortune (2010): *Introduction to Discrete Event Systems*. Springer Publishing Company, Incorporated, doi:10.1007/978-0-387-68612-7.

[6] E Mark Gold (1978): *Complexity of automaton identification from given data*. *Information and Control* 37(3), pp. 302 – 320, doi:10.1016/S0019-9958(78)90562-4.

[7] Wassily Hoeffding (1963): *Probability Inequalities for Sums of Bounded Random Variables*. *Journal of the American Statistical Association* 58(301), pp. 13–30, doi:10.2307/2282952.

[8] Michael J. Kearns (1998): *Efficient Noise-Tolerant Learning from Statistical Queries*. *J. ACM* 45(6), pp. 983–1006, doi:10.1145/293347.293351.

[9] Michael J. Kearns & Umesh V. Vazirani (1994): *An Introduction to Computational Learning Theory*. MIT Press, doi:10.7551/mitpress/3897.001.0001.

[10] J. R. Quinlan (1986): *The Effect of Noise on Concept Learning*. In: *Machine Learning, An Artificial Intelligence Approach Volume II*, chapter 6, Morgan Kaufmann, pp. 149–166.

[11] Ray J. Solomonoff (1964): *A Formal Theory of Inductive Inference*. *Inf. Control.* 7(1, 2), pp. 1–22, 224–254, doi:10.1016/S0019-9958(64)90223-2.

[12] Leslie G. Valiant (1984): *A Theory of the Learnable*. *Commun. ACM* 27(11), pp. 1134–1142, doi:10.1145/1968.1972.

[13] R. M. Wharton (1974): *Approximate language identification*. *Information and Control* 26(3), pp. 236 – 255, doi:10.1016/S0019-9958(74)91369-2.