Languages, automata and computations II
Tutorial 1

Winter semester 2023/2024

During the lecture it was shown Angluin's DFA learning algorithm. Recall that in Angluin's learning framework, Learner can ask two kind of queries: Membership queries ("is word $w \in \Sigma^*$ in the language?" answer: yes / no) and equivalence queries ("is the language recognised by a given DFA A?" answer: yes / no + counter-example word witnessing inequivalence). Note that the size n of the minimal automaton to be learned is not known by Learner in advance, however it is fixed by Teacher before the learning algorithm can start (otherwise Teacher can cheat and always answers "no" continually increasing n). In this tutorial we explore what happens when we question one or more of the aspects of this framework.

In the sequel, we fix a binary alphabet $\Sigma = \{a, b\}$. Recall that a (total) DFA is a tuple $A = (\Sigma, Q, q_I, F, \delta)$, where Q is a finite set of states, q_I is the initial state, $F \subseteq Q$ is the set of final states, and $\delta : Q \times \Sigma \to Q$ is the transition function. The *size* of a DFA A is the number of its states. The *complexity* of a learning algorithm is a function $f : \mathbb{N} \to \mathbb{N}$ s.t. every automaton of size $n \in \mathbb{N}$ can be learned with at most $f(n)$ queries.

Exercise 1 (Learning from left quotient queries)**.** Consider a learning framework where Learner can ask queries of the form $\varepsilon \in u^{-1}L$ and $u^{-1}L = v^{-1}L$. Is there a polynomial learning algorithm for DFA?

Solution: Yes. Learner can gradually discover the minimal DFA. Initially there is state ε, which is initial. From a known state $u \in \Sigma^*$ and a letter $a \in \Sigma$, she checks using quotient equivalence queries whether or not there is a known state v s.t. $(ua)^{-1}L = v^{-1}L$. If yes, then there is a transition from u to v labelled a. If not, then ua is added to the set of known states together with a transition from u labelled a. This suffices to discover the transition structure of the automaton. In order to discover accepting states, we use the other type of query. For every known state $u \in \Sigma^*$, it is declared accepting iff $\varepsilon \in u^{-1}L$. □

Exercise 2 (Learning from inclusion queries)**.** Consider a learning framework where Learner can ask language inclusion queries of the form $L(A) \subseteq L$ and $L \subseteq L(A)$. Is there a polynomial learning algorithm for DFA?

Solution: Yes, we reduce to Angluin's algorithm for DFAs. The idea is that inclusion queries can simulate both membership queries and equivalence queries. When Learner wants to ask a membership query $w \in L$, we instead ask an inclusion query $\{w\} \subseteq L$. When Learner wants to ask an equivalence query

$L = L(A)$, we instead ask two inclusion queries $L(A) \subseteq L$ and $L \subseteq L(A)$, returning counter-examples as appropriate. □

Exercise 3 (146)**.** Consider a modified learning framework where Learner can only ask equivalence queries, to which Teacher only answers yes / no (i.e., without inequivalence counter-examples).

1. Design a learning algorithm of exponential complexity.

2. Show that one cannot do better. Does knowing the size n of Teacher's automaton help Learner?

Solution: We observe that there are at most $n \cdot 2^n \cdot n^{2 \cdot n} \in 2^{O(n \log n)}$ DFAs of size n, and thus $2^{O(n \log n)}$ DFA of size of size $\leq n$. Regarding the first point, Learner starts enumerating all DFAs of size $1, 2, 3, \ldots$ and asks for equivalence until it finds the right one. If the automaton to be learned has size n, then this algorithm terminates after $2^{O(n \log n)}$ queries.

We argue that one cannot do better than this. Suppose that Learner uses an algorithm that in order to learn a DFA of (unknown, but fixed) size n asks equivalence queries for DFAs $A_1, \ldots, A_k$ where k is strictly less than the number $f(n)$ of *minimal* DFA of size n, and then (incorrectly) returns A_k. Teacher answers all queries negatively, which is consistent since she claims that the automaton to be learned was one not enumerated by Learner. It remains to argue that $f(n)$ is exponential in n. For instance already over a unary alphabet consider all languages $L \subseteq \{a\}^{\leq n}$ containing words of length at most n. There are 2^{n+1} such languages and their minimal recognisers have size $\leq n$, therefore $f(n) \geq 2^{n+1}$.

This construction shows that knowing n does not help Learner. □

The modification of the exercise above where Teacher additionally needs to provide inequivalence counter-examples yields the same outcome, but now the lower bound is harder to design. It was given as an open problem in [1], later solved in [2].

Exercise 4. (147) Consider a modified learning framework where Learner can only ask membership queries and a single equivalence query at the end.

1. Show that there is no learning algorithm.

2. What happens if Learner knows in advance the size n of the automaton to be learned?

3. In the previous situation, is there a polynomial learning algorithm?

Solution: There is no algorithm which works for every size n of DFA to be learned. By contradiction, suppose such an algorithm exists. Let Teacher's automaton A be the automaton with $n = 1$ states recognising the empty language and run the algorithm. Teacher always answers "no" to membership queries $w_1, \ldots, w_k$ until Learner correctly proposes B. Now let Teacher's automaton be any automaton B rejecting $w_1, \ldots, w_k$ with nonempty language $L(B) \neq \emptyset$. Since Learner's strategy is deterministic and membership queries are answered the same way as before, also in this case Learner will propose A, which however will be incorrect.

The situation is different if Learner knows that the automaton to be learned has size n. Recall that two inequivalent DFAs of size n differ already at a word of length at most n^2 (complementation + product construction). Thus Learner enumerates all words of length up to n^2 and knowing the answer to each of these 2^{n^2} membership queries they can correctly infer the automaton to be learned. It is not clear whether this bound can be improved, for instance deciding nonemptiness of the intersection of two DFA languages in sub-quadratic time $O(n^{2-\epsilon})$ for some $\epsilon > 0$ is a long-standing open problem.

There is no polynomial learning algorithm. By way of contradiction, let $p(x) \in \mathbb{N}[x]$ and consider membership queries $w_1, \ldots, w_{p(n)}$. Teacher always answers "no". Suppose Learner guesses automaton A. If there is a word w of length $\leq n$ not accepted by A and different from the w_i's, then Teacher chooses the characteristic automaton B_w. Otherwise, A accepts all words w's of length $\leq n$ different from the w_i's. Since $p(n)$ is a polynomial and there are exponentially many words of length $\leq n$, A accepts at least two such words. In this case, Teacher chooses B_w for any word w of length $\leq n$ different from the w_i's (which exists by the same argument). In both cases, $L(A) \neq L(B_w)$. □

Exercise 5 (148)**.** Consider a modified learning framework where Learner can ask membership and equivalence queries, however Teacher does not provide any counter-example to the equivalence queries.

1. Is there a polynomial time learning algorithm in this case? Does knowing the size n of Teacher's automaton help Learner?

2. What happens if Teacher provides only negative counter-examples (i.e., counter-examples only of the kind $w \in L(A) \setminus L$)? And only positive ones $(L \setminus L(A))$?

Solution: There is no such algorithm. Teacher keeps track of a set $\{B_w \mid w \in \Sigma^n\}$ of potential DFAs, where B_w has size n and recognises $L(B_w) = \{w\}$. Whenever Learner asks a membership query $w \in \Sigma^*$, Teacher answers negatively and removes B_w from the set if it is there. Whenever Learner asks an equivalence query A, Teacher answers negatively and removes A from the set if it is there. Since Teacher's initial set has exponential size, and at each round zero or one DFAs are removed from the set, it is clear that after a polynomial number of rounds the set will be non-empty and Teacher can claim that the automaton to be learned was any such survivor. Knowing the size n of Teacher's automaton does not help Learner.

Negative counter-examples don't help, since Teacher can always return any word w from $L(A)$ and if this word has length n, remove the corresponding B_w from the set. Also positive counter-examples don't help, but in this case Teacher keeps track of automata B_w's recognising $\Sigma^* \setminus \{w\}$ for words w of length n: Membership queries are always answered positively, and equivalence queries negatively, always providing a positive counter-example not in $L(A)$. □

Exercise 6 (Learning from positive and negative examples)**.** Show that the following problem is NP-complete: In input we are given two disjoint finite languages $L, M \subseteq_{\text{fin}} \Sigma^*$ and a number $n \in \mathbb{N}$, and we need to decide whether there exists a DFA of size at most n recognising all words from L and rejecting all words from M. *[Hint: Reduce from SAT instances where every clause contains either only positive or only negative literals (known to be NP-complete).]*

References

[1] Dana Angluin. Learning regular sets from queries and counterexamples. *Inf. Comput.*, 75(2):87–106, November 1987.

[2] Dana Angluin. Negative results for equivalence queries. *Mach. Learn.*, 5(2):121–150, jul 1990.