Efficient Inclusion Testing for Simple Classes of Unambiguous ω-Automata

Dimitri Isaak[a], Christof Löding[a]

[a] *Informatik 7, RWTH Aachen, Germany*

Abstract

We show that language inclusion for languages of infinite words defined by non-deterministic automata can be tested in polynomial time if the automata are unambiguous and have simple acceptance conditions, namely safety or reachability conditions. An automaton with safety condition accepts an infinite word if there is a run that never visits a forbidden state, and an automaton with reachability condition accepts an infinite word if there is a run that visits an accepting state at least once.

Keywords: formal languages, automata theory, ω-automata, unambiguous automata, inclusion

1. Introduction

Testing language inclusion for nondeterministic finite automata is a difficult operation from a complexity theoretic point of view. Even checking whether a nondeterministic automaton accepts all finite words is a PSPACE-complete problem (see Section 10.6 of [1]). For deterministic automata the problem is tractable because deterministic automata can be complemented and thus the inclusion problem can be reduced to the emptiness of the intersection of two automata. In [9] it has been shown that inclusion testing is possible in polynomial time if the automata are unambiguous, that is, for each input there is at most one accepting run. This result has been lifted to unambiguous automata on finite trees in [8], and can, for example, be used to derive efficient inclusion tests for certain classes of automata on unranked trees [6].

Concerning ω-automata (automata on infinite words), it is known that unambiguous Büchi automata capture the same class of ω-languages as unrestricted nondeterministic Büchi automata [2], while deterministic Büchi automata are easily seen to be less expressive. A construction directly transforming nondeterministic Büchi automata into unambiguous ones (without going through deterministic automaton models) has recently been presented in [5].

While the construction of unambiguous Büchi automata has been studied, their algorithmic properties have not yet been analyzed. In particular, the complexity of the inclusion problem for unambiguous Büchi automata is open. In

[3] it is shown that inclusion testing is tractable for the class of strongly unambiguous Büchi automata. A Büchi automaton is called strongly unambiguous if it remains unambiguous even if the set of all states is declared initial. These automata are expressively complete, as shown in [4] (see also the chapter on prophetic automata in [7]).

In this paper we consider the standard notion of unambiguous Büchi automata. To obtain classes of automata with tractable inclusion problem, we look at subclasses of Büchi automata with simpler acceptance conditions. A safety automaton (sometimes called looping automaton) accepts if there is an infinite run on the input word, while a reachability automaton accepts an infinite input if there is an infinite run in which an accepting state is visited. These automata can be used to capture simple classes of properties, namely safety and guarantee properties that are frequently used in verification. We show that the inclusion problems for these two classes of unambiguous automata can be solved efficiently, basically by reducing them to inclusion problems for unambiguous automata on finite words.

The remainder of the paper is structured as follows. In Section 2 we introduce basic terminology and the models of automata considered in this work. In Sections 3 and 4 we show how to efficiently test inclusion for unambiguous safety and reachability automata, respectively, and in Section 5 we conclude.

The authors thank the anonymous reviewers for their comments.

2. Automata on Finite and Infinite Words

In this section we present basic definitions and results concerning finite automata. We assume that the reader is familiar with finite automata on finite words and only briefly fix our notation.

For an alphabet Σ we denote as usual the set of finite words over Σ by Σ^* and the set of infinite words by Σ^ω. For an infinite word $\alpha \in \Sigma^\omega$ we denote the jth letter by $\alpha(j)$, i.e., $\alpha = \alpha(0)\alpha(1)\cdots$. For a nonempty finite word u we write u^ω for its infinite iteration $uuu\cdots$.

Nondeterministic finite automata (NFA) are of the form $\mathcal{A} = (Q, \Sigma, q_{\text{in}}, \Delta, F)$, where Q is a finite set of states, Σ is the input alphabet, $q_{\text{in}} \in Q$ is the initial state, $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation, and $F \subseteq Q$ is the set of accepting states. An automaton is called *complete*, if for each combination of state $q \in Q$ and letter $a \in \Sigma$ there exists a transition of the form $(q, a, q') \in \Delta$. For $q, q' \in Q$ and $u \in \Sigma^*$ we write $\mathcal{A} : q \xrightarrow{u} q'$ if there is a path in $\mathcal{A}$ from q to q' that is labeled by u. The language of finite words accepted by $\mathcal{A}$ is $L_*(\mathcal{A}) = \{w \in \Sigma^* \mid \mathcal{A} : q_{\text{in}} \xrightarrow{w} q \in F\}$. We use the notation $L_*(\mathcal{A})$ to distinguish the language of finite words clearly from the language of infinite words accepted by $\mathcal{A}$ (as defined below).

We consider ω-automata $\mathcal{A} = (Q, \Sigma, q_{\text{in}}, \Delta, F)$ that are of the same form as NFAs. A *run* of $\mathcal{A}$ on an infinite word α is an infinite sequence $q_0 q_1 \ldots$ of states such that $q_0 = q_{\text{in}}$ and for each i the transition $(q_i, \alpha(i), q_{i+1})$ is in Δ. The run is accepting if it satisfies the *acceptance condition*, which depends on the type of the automaton. The standard acceptance condition is the Büchi condition. If $\mathcal{A}$ is considered as *Büchi automaton*, then a run is accepting if it infinitely often visits an accepting state from F. We are mainly concerned with simpler conditions, namely safety and reachability: A run of a *safety automaton* is accepting if it does not contain a state from $Q \setminus F$. A run of a *reachability automaton* is accepting if it contains a state from F.

From the above definition it is easy to see that a safety automaton can be viewed as a Büchi automaton in which all states are accepting except for possibly one rejecting sink state. In this case, a run visits F infinitely often if, and only if, it never visits the rejecting sink. Since we are working with nondeterministic automata, this rejecting sink state is not necessary and can be omitted.

Note that while each safety automaton can easily be made complete by adding a non-accepting sink, incomplete reachability automata are more expressive than complete ones, because a complete reachability automaton has no means to reject an input after a run has reached an accepting state, while an incomplete automaton can still reject if the run cannot be extended. Consider, for example, the language $(a+b)^*a^\omega$ consisting of all words with finitely many b. An incomplete reachability automaton can accept this language by looping on the non-accepting initial state on a and b, and allowing a nondeterministic transition on a to an accepting state that loops on a but has no transition for b. It is easy to see that this language cannot be accepted by a complete reachability automaton. In this paper we only consider complete reachability automata. A complete reachability automaton can be viewed as a Büchi automaton in which all states are rejecting except for one accepting sink state (looping on every input letter). Then a run visits F infinitely often if, and only if, it visits the accepting sink state. We usually denote this unique accepting state as q_f.

In the following, we always assume that safety and reachability automata are given in this normalized version and thus we can simply view them as special classes of Büchi automata. Using this convention, we can write $L_\omega(\mathcal{A})$ for the language of all infinite words that are accepted by $\mathcal{A}$ using the Büchi condition.

We are interested in the complexity of the inclusion problem for ω-automata, that is, the problem of deciding for two given automata $\mathcal{A}$ and $\mathcal{A}'$ whether $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$. A special instance of the inclusion problem is the universality problem, that is, the problem of deciding whether a given automaton accepts all words. It is well known that for NFAs these problems are PSPACE-hard (see Section 10.6 of [1]), and it is straightforward to lift this hardness result to Büchi automata, and even safety and reachability automata. Therefore, we consider a subclass of nondeterministic automata, called unambiguous.

A nondeterministic automaton $\mathcal{A}$ is called *unambiguous* if for each input there is at most one accepting run of $\mathcal{A}$ on this input. The class of unambiguous NFAs is interesting because they admit efficient algorithms but can be exponentially more succinct than deterministic automata.

Theorem 1 ([9]). *For unambiguous NFAs the inclusion problem can be solved in polynomial time.*

The complexity of the inclusion problem (or even the universality problem)

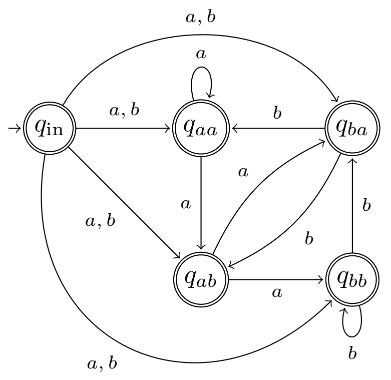

Figure 1: An unambiguous safety automaton accepting all infinite words (the rejecting sink state is omitted).

for unambiguous Büchi automata is open. The aim of this paper is to show that the inclusion problem for unambiguous safety and reachability automata can be reduced in polynomial time to the same problem for unambiguous NFAs.

3. Inclusion Testing for Safety Automata

It is not difficult to verify that for safety automata $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$ holds if, and only if, $L_*(\mathcal{A}) \subseteq L_*(\mathcal{A}')$. This observation is based on the fact that an infinite word α is accepted by a safety automaton if, and only if, each of its finite prefixes is accepted by the corresponding NFA.

However, an automaton $\mathcal{A}$ can be unambiguous when viewed as an ω-automaton but ambiguous when viewed as NFA. This is illustrated in Figure 1. The depicted safety automaton accepts all infinite words, and for each infinite word there is exactly one accepting run (the automaton always guesses the next two letters of the input). Viewed as NFA the automaton is not unambiguous. For this reason, we cannot simply reduce the inclusion test for unambiguous safety automata to the inclusion test for unambiguous NFAs by viewing the safety automata as NFAs.

We use the following generic construction to turn an unambiguous Büchi automaton into an unambiguous NFA (the construction works for arbitrary Büchi automata but we only use it in the context of safety and reachability automata). For a Büchi automaton $\mathcal{A} = (Q, \Sigma, q_{\text{in}}, \Delta, F)$ and $\alpha \in \Sigma^\omega$ define

$$F_\alpha = \{q \in Q \mid \mathcal{A} \text{ accepts } \alpha \text{ with } q \text{ as initial state}\}$$

and $\mathcal{A}_\alpha = (Q, \Sigma, q_{\text{in}}, \Delta, F_\alpha)$. For example, setting $\alpha = a^\omega$ we obtain $F_\alpha = \{q_{\text{in}}, q_{aa}\}$ for the automaton in Figure 1 because these are the only states from which a^ω is accepted.

Lemma 2. *If $\mathcal{A}$ is an unambiguous Büchi automaton, then $\mathcal{A}_\alpha$ is an unambiguous NFA for each infinite word α.*

Proof. Assume a word $w \in \Sigma^*$ is accepted by two different runs in $\mathcal{A}_\alpha$. Then these two runs end in a final state of $\mathcal{A}_\alpha$ and thus can be extended to two different accepting runs of $\mathcal{A}$ on $w\alpha$, contradicting the unambiguity of $\mathcal{A}$. □

Another important property of this construction is that it preserves language inclusion in the following sense.

Lemma 3. *Let $\mathcal{A}$ and $\mathcal{A}'$ be two Büchi automata. Then $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$ if, and only if, $L_*(\mathcal{A}_\alpha) \subseteq L_*(\mathcal{A}'_\alpha)$ for each $\alpha \in \Sigma^\omega$.*

Proof. If $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$, then there is $\alpha \in L_\omega(\mathcal{A}) \setminus L_\omega(\mathcal{A}')$. We obtain that $\varepsilon \in L_*(\mathcal{A}_\alpha)$ but $\varepsilon \notin L_*(\mathcal{A}'_\alpha)$. Now assume that $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$. If $w \in L_*(\mathcal{A}_\alpha)$, then one can reach from the initial state of $\mathcal{A}$ via w a state from which α is accepted. Therefore $w\alpha \in L_\omega(\mathcal{A})$ and therefore $w\alpha \in L_\omega(\mathcal{A}')$, which implies that w is in $L_*(\mathcal{A}'_\alpha)$. □

Our aim is to reduce the inclusion test $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$ for unambiguous safety automata $\mathcal{A}$ and $\mathcal{A}'$ to (several) inclusion tests $L_*(\mathcal{A}_\alpha) \subseteq L_*(\mathcal{A}'_\alpha)$ for unambiguous NFAs. If all these inclusion tests succeed we can conclude that the inclusion also holds for the safety automata.

We start with a remark that is an easy consequence of König's Lemma.

Remark 1. If a word $\alpha \in \Sigma^\omega$ is not accepted by a safety automaton $\mathcal{A}$, then there is a finite prefix w of α such that $w\Sigma^\omega \cap L_\omega(\mathcal{A}) = \emptyset$.

Using this observation we can derive the existence of simple witnesses if the inclusion test fails.

Lemma 4. *Let $\mathcal{A}$ and $\mathcal{A}'$ be safety automata. For all states q of $\mathcal{A}$ that are on a cycle let u_q be an arbitrary word with $\mathcal{A} : q \xrightarrow{u_q} q$. Then $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$ if, and only if, there is a state q of $\mathcal{A}$ and a finite word $w \in \Sigma^*$ such that $\mathcal{A} : q_{\text{in}} \xrightarrow{w} q$ and $wu_q^\omega \notin L_\omega(\mathcal{A}')$.*

Proof. Clearly, if $wu_q^\omega \notin L_\omega(\mathcal{A}')$ for a w with $\mathcal{A} : q_{\text{in}} \xrightarrow{w} q$, then $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$ because $wu_q^\omega \in L_\omega(\mathcal{A})$.

For the other direction, assume that $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$. Then there is $\alpha \in L_\omega(\mathcal{A})$ that is not in $L_\omega(\mathcal{A}')$. By Remark 1 there is a prefix w of α such that $w\Sigma^\omega \cap L_\omega(\mathcal{A}') = \emptyset$. Let q be such that $\mathcal{A} : q_{\text{in}} \xrightarrow{w} q$ is the initial segment of an accepting run of $\mathcal{A}$ on α. If q is on a cycle, then $wu_q^\omega \in L_\omega(\mathcal{A}) \setminus L_\omega(\mathcal{A}')$. If q is not on a cycle, then a state p on a cycle is reachable from q via some word v since the accepting run of $\mathcal{A}$ on α is going through q, and we obtain $wvu_p^\omega \in L_\omega(\mathcal{A}) \setminus L_\omega(\mathcal{A}')$. □

Theorem 5. *The inclusion problem for unambiguous safety automata can be solved in polynomial time.*

Proof. Let $\mathcal{A}$ and $\mathcal{A}'$ be unambiguous safety automata. According to Lemma 4, the following algorithm is sufficient:

- Compute for each state q of $\mathcal{A}$ on a cycle a word $u_q \in \Sigma^*$ such that $\mathcal{A} : q \xrightarrow{u_q} q$.
- Test if $L_*(\mathcal{A}_{u_q^\omega}) \subseteq L_*(\mathcal{A}'_{u_q^\omega})$ for all these u_q.
- If one of the tests fails, then output $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$, otherwise output $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$.

If one of the tests fails, then $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$ according to Lemma 3. Vice versa, if $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$, let wu_q^ω with $\mathcal{A} : q_{\text{in}} \xrightarrow{w} q$ be a witness according to Lemma 4. Then w is in $L_*(\mathcal{A}_{u_q^\omega})$ but not in $L_*(\mathcal{A}'_{u_q^\omega})$.

Concerning the complexity of the algorithm, the automata $\mathcal{A}_{u_q^\omega}$ and $\mathcal{A}'_{u_q^\omega}$ are unambiguous NFAs (Lemma 2) and therefore the inclusion test can be done in polynomial time according to Theorem 1. The words u_q can be computed in polynomial time by a simple depth-first traversal starting from q. Finally, computing the set of accepting states for the automata $\mathcal{A}_{u_q^\omega}$ and $\mathcal{A}'_{u_q^\omega}$ can be done in polynomial time because testing whether a word u_q^ω is accepted from some state of an automaton can be done in polynomial time. □

A consequence of the proof of Theorem 5 is a simple reduction for the universality test of unambiguous safety automata.

Corollary 6. *Let $\mathcal{A}$ be an unambiguous safety automaton and let $a \in \Sigma$ be some letter from the input alphabet. Then $L_\omega(\mathcal{A}) = \Sigma^\omega$ if, and only if, $L_*(\mathcal{A}_{a^\omega}) = \Sigma^*$.*

Proof. The automaton $\mathcal{A}$ is universal (accepts all inputs) if, and only if, the inclusion $L_\omega(\mathcal{B}) \subseteq L_\omega(\mathcal{A})$ holds for the trivial automaton $\mathcal{B}$ consisting of only one state q looping on all input letters. So the only word u_q that has to be considered in the inclusion test from Theorem 5 can be chosen $u_q = a$ for some input letter a. □

4. Inclusion Testing for Reachability Automata

In contrast to safety automata, unambiguous reachability automata considered as NFAs are also unambiguous (recall that we assume that in a reachability automaton the only accepting state is a sink state looping on all input letters). But now we have to face the opposite problem that $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$ does not imply $L_*(\mathcal{A}) \subseteq L_*(\mathcal{A}')$. For example, $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$ for the automata depicted in Figure 2, since each word starting with a is accepted by $\mathcal{A}'$. But $a \in L_*(\mathcal{A})$ and $a \notin L_*(\mathcal{A}')$. The problem would be solved if we would add state 3 or state 5 to the set of final states of $\mathcal{A}'$ when considering it as NFA.

However, if the goal is to extend the set of final states of $\mathcal{A}'$ such that the resulting inclusion test for NFAs gives the correct answer for the inclusion of the ω-languages, then we run into the following problem. Adding 3 to the set

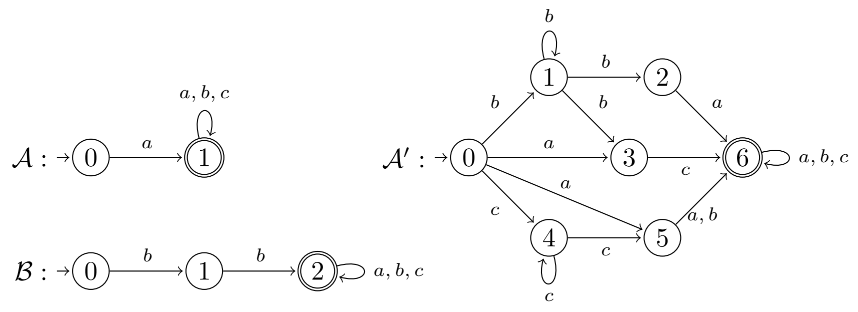

Figure 2: Unambiguous reachability automata with $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$ and $L_\omega(\mathcal{B}) \not\subseteq L_\omega(\mathcal{A}')$.

of final states of $\mathcal{A}'$ results in an NFA that accepts all words starting with bb, but $\mathcal{A}'$ does not accept all ω-words starting with bb (b^ω is not accepted). Thus, adding 3 to the set of final states of $\mathcal{A}'$ and then applying the inclusion test for NFAs results in a wrong answer for $\mathcal{B}$ because $L_\omega(\mathcal{B}) \not\subseteq L_\omega(\mathcal{A}')$. The same problem appears when adding 5 to the set of final states of $\mathcal{A}'$ for the words starting with cc.

This shows that we cannot simply extend the set of final states of $\mathcal{A}'$ and look at the result of the inclusion test for the NFAs. But we show below that this kind of reduction works if we first do a different test on the two automata (that would then fail for $\mathcal{B}$ and $\mathcal{A}'$ and thus show that $L_\omega(\mathcal{B}) \not\subseteq L_\omega(\mathcal{A}')$).

The problem in the above example stems from the fact that reading the word bb in $\mathcal{B}$ leads to the accepting sink and in $\mathcal{A}'$ it can lead to a rejecting state that allows a loop (state 1). Let us start with a general observation on words looping on rejecting states. In the following we assume that all states of the automata are live in the sense that from every state the accepting sink is reachable.

Lemma 7. *Let $\mathcal{A} = (Q, \Sigma, q_{\text{in}}, \Delta, \{q_{\text{f}}\})$ be an unambiguous reachability automaton, $u, v \in \Sigma^*$, and $q \in Q$ with $q \neq q_{\text{f}}$ be such that $\mathcal{A} : q_{\text{in}} \xrightarrow{u} q \xrightarrow{v} q$. Then $uv^\omega \notin L_\omega(\mathcal{A})$.*

Proof. If uv^ω is accepted by $\mathcal{A}$, then there is a finite prefix uv^n with $\mathcal{A} : q_{\text{in}} \xrightarrow{uv^n} q_{\text{f}}$. Then $uv^n\beta \in L_\omega(\mathcal{A}')$ for all infinite words β. On the other hand, the state q is live and thus there is a word $x \in \Sigma^*$ with $\mathcal{A} : q \xrightarrow{x} q_{\text{f}}$. This yields at least two different accepting runs for all words of the form $uv^n x\beta$ for some infinite word β, namely one run that reaches q_{f} already after reading uv^n, and another run that reaches q_{f} only after reading $uv^n x$. This contradicts the unambiguity of $\mathcal{A}$. □

Lemma 7 motivates the definition of the set

$$\mathsf{Loop}(\mathcal{A}) = \{q \in Q \mid \exists u, v \in \Sigma^*, q' \in Q \setminus \{q_{\text{f}}\} \text{ with } \mathcal{A} : q \xrightarrow{u} q' \xrightarrow{v} q'\}$$

of those states of $\mathcal{A}$ from which a rejecting loop is reachable (and since we assume that all states are live, this loop cannot be on a rejecting sink state). For example, $\mathsf{Loop}(\mathcal{A}') = \{0, 1, 4\}$ for $\mathcal{A}'$ as shown in Figure 2.

Now we can state a first criterion showing that the inclusion of two ω-languages does not hold.

Lemma 8. *Let $\mathcal{A} = (Q, \Sigma, q_{\text{in}}, \Delta, \{q_{\text{f}}\})$ and $\mathcal{A}' = (Q', \Sigma, q'_{\text{in}}, \Delta', \{q'_{\text{f}}\})$ be unambiguous reachability automata. If there is $u \in \Sigma^*$ and $q' \in \mathsf{Loop}(\mathcal{A}')$ such that $\mathcal{A} : q_{\text{in}} \xrightarrow{u} q_{\text{f}}$ and $\mathcal{A}' : q'_{\text{in}} \xrightarrow{u} q'$, then $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$.*

Proof. From the definition of $\mathsf{Loop}(\mathcal{A}')$ we conclude that there are $v, w \in \Sigma^*$ and a state $q'' \in Q'$ such that $\mathcal{A} : q' \xrightarrow{v} q'' \xrightarrow{w} q''$ and Lemma 7 implies that $uvw^\omega \notin L_\omega(\mathcal{A}')$. Since $\mathcal{A} : q_{\text{in}} \xrightarrow{u} q_{\text{f}}$, we obtain that $uvw^\omega \in L_\omega(\mathcal{A})$. □

If the situation of Lemma 8 is given, then we say that $\mathcal{A}'$ has a rejecting loop witness for $\mathcal{A}$. For example, for the automata depicted in Figure 2, $\mathcal{A}'$ has a rejecting loop witness for $\mathcal{B}$: bb leads to the accepting sink of $\mathcal{B}$ and to state 1 in $\mathcal{A}'$ which is in $\mathsf{Loop}(\mathcal{A}')$.

Now we are ready to state the necessary and sufficient criterion for the inclusion test of unambiguous reachability automata. We use the notation $\mathcal{A}_\alpha$ for $\alpha = a^\omega$, as defined in Section 3.

Lemma 9. *Let $\mathcal{A}$ and $\mathcal{A}'$ be unambiguous reachability automata over input alphabet Σ such that $\mathcal{A}'$ has no rejecting loop witness for $\mathcal{A}$. Then $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$ if, and only if, $L_*(\mathcal{A}) \subseteq L_*(\mathcal{A}'_{a^\omega})$, where $a \in \Sigma$ is an arbitrary input letter.*

Proof. Let $\mathcal{A} = (Q, \Sigma, q_{\text{in}}, \Delta, \{q_{\text{f}}\})$ and $\mathcal{A}' = (Q', \Sigma, q'_{\text{in}}, \Delta', \{q'_{\text{f}}\})$ be unambiguous reachability automata such that $\mathcal{A}'$ has no rejecting loop witness for $\mathcal{A}$, and let $a \in \Sigma$.

Assume that $L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$ and let $u \in L_*(\mathcal{A})$. We have to show that $u \in L_*(\mathcal{A}'_{a^\omega})$. Since $u \in L_*(\mathcal{A})$ we know that $\mathcal{A} : q_{\text{in}} \xrightarrow{u} q_{\text{f}}$ and thus $ua^\omega \in L_\omega(\mathcal{A}) \subseteq L_\omega(\mathcal{A}')$. Therefore, there is an accepting run of $\mathcal{A}'$ on ua^ω and hence $u \in L_*(\mathcal{A}'_{a^\omega})$.

For the other implication assume that $L_*(\mathcal{A}) \subseteq L_*(\mathcal{A}'_{a^\omega})$ and let $\alpha \in L_\omega(\mathcal{A})$. We have to show that $\alpha \in L_\omega(\mathcal{A}')$. Let u be a prefix of α with $\mathcal{A} : q_{\text{in}} \xrightarrow{u} q_{\text{f}}$. Since $\mathcal{A}'$ has no rejecting loop witness for $\mathcal{A}$, all runs of $\mathcal{A}'$ on u must lead to a state outside $\mathsf{Loop}(\mathcal{A}')$. Let $v \in \Sigma^*$ such that uv is a prefix of α. Then $uv \in L_*(\mathcal{A}) \subseteq L_*(\mathcal{A}'_{a^\omega})$. Consider the accepting run of $\mathcal{A}'_{a^\omega}$ on uv. As explained above, after having processed u, the run is in a state q' outside $\mathsf{Loop}(\mathcal{A}')$. If we choose v such that the length of v is at least $|Q'| + 1$, then the accepting run of $\mathcal{A}'_{a^\omega}$ on uv must have reached q'_{f} because no rejecting loop is reachable from q'. Thus, $\alpha \in L_\omega(\mathcal{A}')$. □

The conditions from Lemma 8 and Lemma 9 can be turned into an algorithm for inclusion.

Theorem 10. *The inclusion problem for unambiguous reachability automata can be solved in polynomial time.*

Proof. Let $\mathcal{A} = (Q, \Sigma, q_{\text{in}}, \Delta, \{q_{\text{f}}\})$ and $\mathcal{A}' = (Q', \Sigma, q'_{\text{in}}, \Delta', \{q'_{\text{f}}\})$ be unambiguous reachability automata. We can decide in polynomial time whether $\mathcal{A}'$ has a rejecting loop witness for $\mathcal{A}$ by checking emptiness of the intersection $L_*(\mathcal{A}) \cap L_*(\mathcal{A}'_{\mathsf{Loop}})$, where $\mathcal{A}'_{\mathsf{Loop}} = (Q', \Sigma, q'_{\text{in}}, \Delta', \mathsf{Loop}(\mathcal{A}'))$. This test is linear in the product of $\mathcal{A}$ and $\mathcal{A}'_{\mathsf{Loop}}$. The set $\mathsf{Loop}(\mathcal{A}')$ can be computed by standard graph algorithms in time linear in the size of $\mathcal{A}'$.

If $\mathcal{A}'$ has a rejecting loop witness for $\mathcal{A}$, then we know that $L_\omega(\mathcal{A}) \not\subseteq L_\omega(\mathcal{A}')$. Otherwise, we have to test $L_*(\mathcal{A}) \subseteq L_*(\mathcal{A}'_{a^\omega})$, which can be done in polynomial time according to Theorem 1 since $\mathcal{A}'_{a^\omega}$ is unambiguous (Lemma 2). $\square$

5. Conclusion

We have shown how to use the efficient inclusion testing for unambiguous NFAs to solve the inclusion problem for simple classes of unambiguous Büchi automata, namely safety and (complete) reachability automata. The complexity of the inclusion problem for the full class of unambiguous Büchi automata remains open. It seems that the methods presented in this work are not suitable for solving this question. An intermediate class that could be interesting to look at is the class of unambiguous co-Büchi automata. A run of co-Büchi automaton is accepting if it visits non-accepting states only finitely often. Unambiguous co-Büchi automata can be transformed into unambiguous reachability automata (but with undefined transitions in contrast to complete reachability automata considered in this work). So this class basically combines the features of safety and reachability automata.

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, New York, 1974.

[2] A. Arnold. Rational ω-languages are non-ambiguous. *Theor. Comput. Sci.*, 26:221–223, 1983.

[3] N. Bousquet and C. Löding. Equivalence and inclusion problem for strongly unambiguous Büchi automata. In *Proceedings of LATA 2010*, volume 6031 of *LNCS*, pages 118–129. Springer, 2010.

[4] O. Carton and M. Michel. Unambiguous Büchi automata. *Theor. Comput. Sci.*, 297(1–3):37–81, 2003.

[5] D. Kähler and T. Wilke. Complementation, disambiguation, and determinization of Büchi automata unified. In *Proceedings of ICALP 2008*, volume 5125 of *LNCS*, pages 724–735. Springer, 2008.

[6] W. Martens and J. Niehren. On the minimization of xml schemas and tree automata for unranked trees. *J. Comput. Syst. Sci.*, 73(4):550–583, 2007.

[7] D. Perrin and J.-É. Pin. *Infinite words*, volume 141 of *Pure and Applied Mathematics.* Elsevier, 2004.

[8] H. Seidl. Deciding equivalence of finite tree automata. *SIAM J. Comput.*, 19(3):424–437, 1990.

[9] R. E. Stearns and H. B. Hunt III. On the equivalence and containment problems for unambiguous regular expressions, regular grammars and finite automata. *SIAM J. Comput.*, 14(3):598–611, 1985.