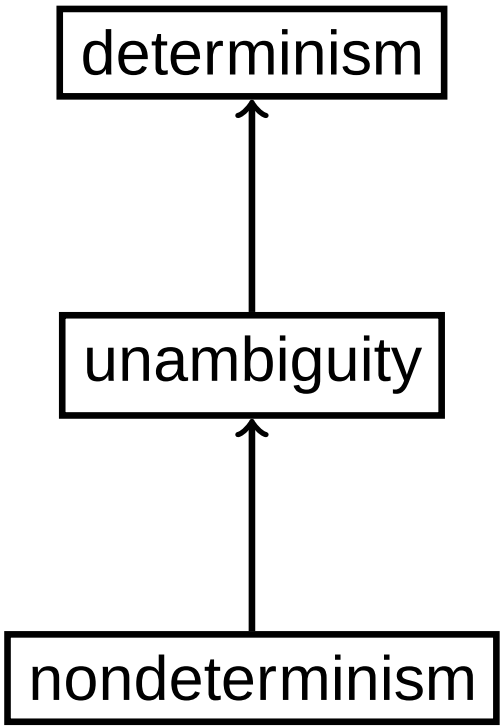
Unambiguous Finite Automata

Christof Löding
Department of Computer Science
RWTH Aachen University, Germany

17th International Conference on
Developments in Language Theory
Paris-Est, June 20, 2013

Unambiguous Automata

Definition: A nondeterministic automaton is called unambiguous if for each word there is at most one accepting run from an initial state.



big automata

inclusion, equivalence, universality are easy

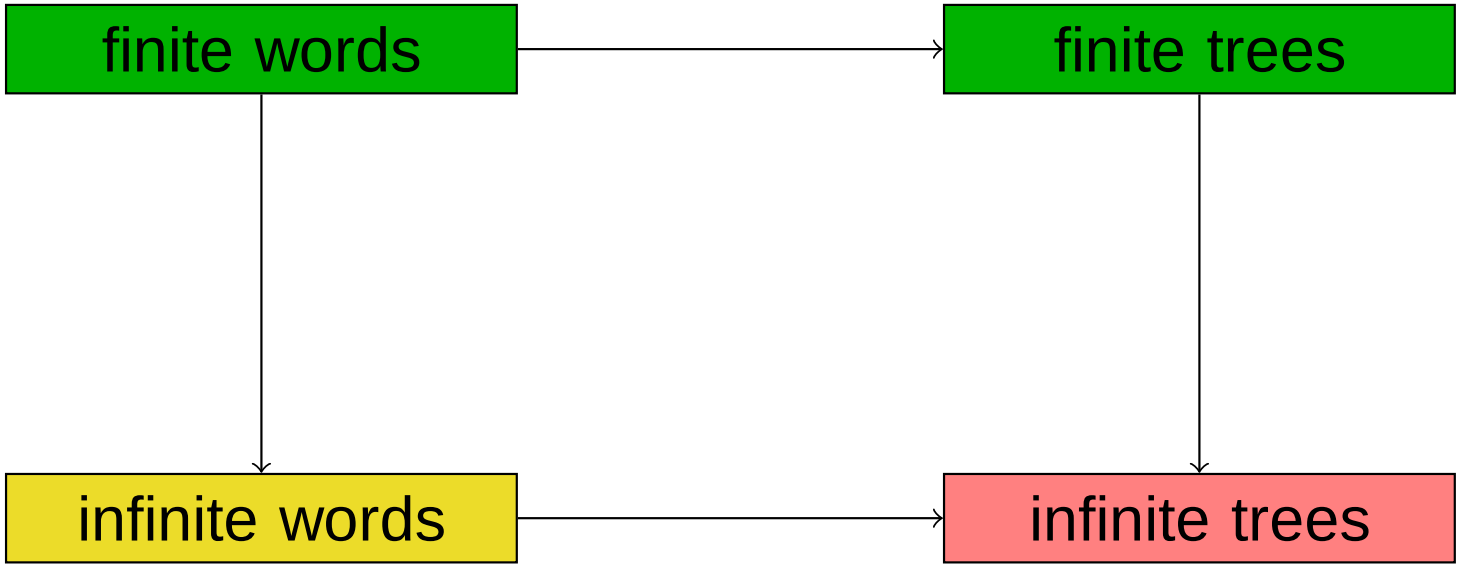
small automata

inclusion, equivalence, universality are hard

In this talk

Unambiguous automata on finite words and trees have already been investigated a lot.

This talk: Present some results and open problems concerning unambiguous automata on infinite words and trees.



Outline

1. Finite Words (and Trees)

2. Infinite Words
   - Strongly unambiguous Büchi automata
   - Safety and reachability automata

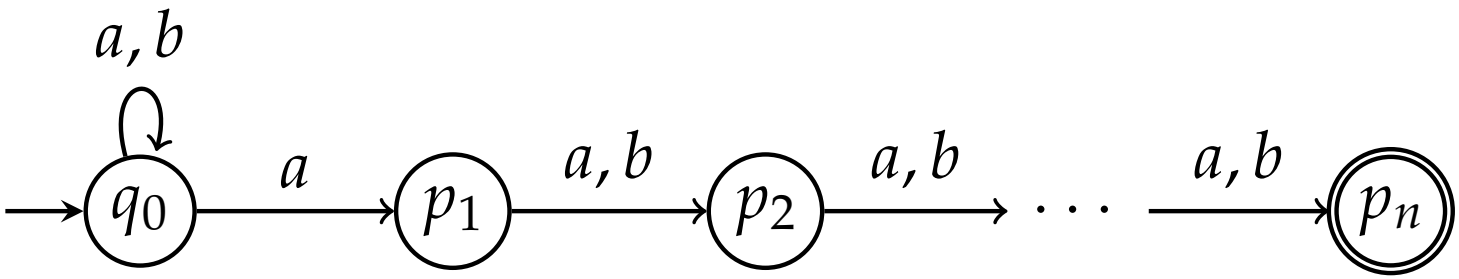
3. Infinite Trees

4. Conclusion

Outline

Unambiguous automata on finite words

- On finite words, deterministic automata are as expressive as nondeterministic ones. Therefore each regular language can be accepted by an unambiguous automaton.
- Unambiguous automata can be exponentially more succinct than deterministic ones.



Decision Problems

Theorem (Meyer/Stockmeyer'72)

The problems of universality, equivalence, and inclusion for NFAs are PSPACE-hard.

Theorem (Stearns/Hunt'85)

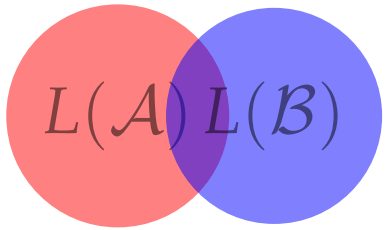
The problems of universality, equivalence, and inclusion for unambiguous finite automata are in PTIME.

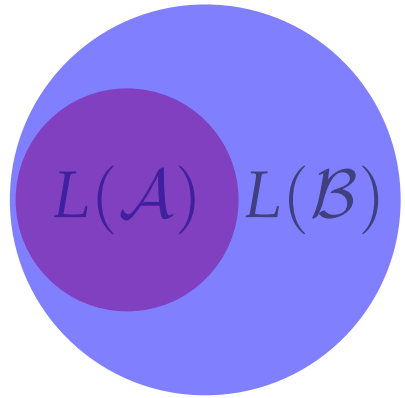
(Can also be derived from earlier results of Schützenberger)

Counting Accepting Runs

Idea for the PTIME inclusion algorithm:

- Problem: test if $L(\mathcal{A}) \subseteq L(\mathcal{B})$ for given automata $\mathcal{A}$ and $\mathcal{B}$
- Consider $\mathcal{A} \cap \mathcal{B}$ (product construction), which is again unambiguous:
  - $L(\mathcal{A} \cap \mathcal{B}) \subseteq L(\mathcal{A})$
  - $L(\mathcal{A}) \subseteq L(\mathcal{B}) \Leftrightarrow L(\mathcal{A} \cap \mathcal{B}) = L(\mathcal{A})$





Counting Accepting Runs

Idea for the PTIME inclusion algorithm:

- Problem: test if $L(\mathcal{A}) \subseteq L(\mathcal{B})$ for given automata $\mathcal{A}$ and $\mathcal{B}$
- Consider $\mathcal{A} \cap \mathcal{B}$ (product construction), which is again unambiguous:
  - $L(\mathcal{A} \cap \mathcal{B}) \subseteq L(\mathcal{A})$
  - $L(\mathcal{A}) \subseteq L(\mathcal{B}) \Leftrightarrow L(\mathcal{A} \cap \mathcal{B}) = L(\mathcal{A})$
- If for each length n, the number of accepting runs of length n is the same in $\mathcal{A} \cap \mathcal{B}$ and $\mathcal{A}$, then $L(\mathcal{A} \cap \mathcal{B}) = L(\mathcal{A})$:
  - computing the number of accepting runs of increasing length is easy (dynamic programming)
  - if the equality holds for runs up to length $|\mathcal{A}| \cdot |\mathcal{B}|$, then it holds for all lengths

Finite Trees

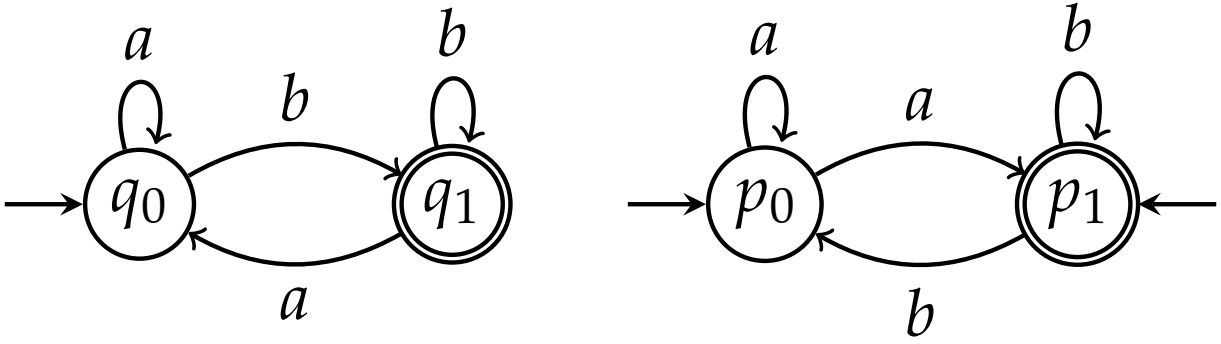
The results can be extended to finite trees.

Theorem (Seidl'90)

The problems of universality, equivalence, and inclusion for unambiguous finite automata on ranked trees are in PTIME.

Outline

Büchi Automata



Büchi automaton:

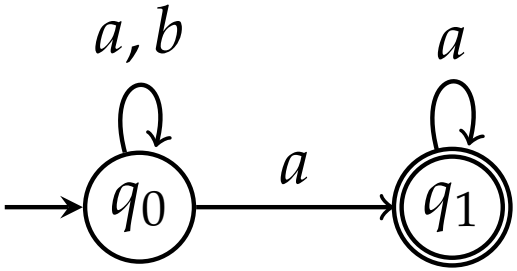
- same syntax as nondeterministic finite automata (NFAs)
- accepts all infinite words that admit a run visiting infinitely often a final state

Most decision problems for Büchi automata are at least as hard as for NFAs

Unambiguous Büchi Automata Always Exist

Deterministic Büchi automata are weaker than nondeterministic ones.

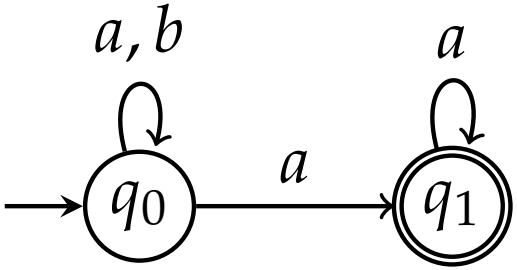
A nondeterministic Büchi automaton for “finitely many b”:



Unambiguous Büchi Automata Always Exist

Deterministic Büchi automata are weaker than nondeterministic ones.

A nondeterministic Büchi automaton for “finitely many b”:



Theorem (Arnold’83)

For each Büchi automaton there is an equivalent unambiguous Büchi automaton.

What about algorithmic properties of unambiguous Büchi automata?

Methods from Finite Case do not Generalize

- Runs in a Büchi automaton are infinite, counting runs up to a certain length does not work.
- But: Regular languages of infinite words can be characterized using finite words: ultimately periodic words.
- Can this be used to lift the methods from the finite case?

Ultimately Periodic Words

Definition: An infinite word is called ultimately periodic if it is of the form $uv^\omega = uvvvvv\cdots$ for finite words u, v.

Examples:

- $aabaabaabaabaab\cdots = a(aba)^\omega = aa(baa)^\omega$
- $aba\underbrace{bb}_{2}a\underbrace{bbb}_{3}a\underbrace{bbbb}_{4}a\underbrace{bbbbb}_{5}a\cdots$ is not ultimately periodic

Ultimately Periodic Words

Definition: An infinite word is called ultimately periodic if it is of the form $uv^\omega = uvvvvv\cdots$ for finite words u, v.

Examples:

- $aabaabaabaabaab\cdots = a(aba)^\omega = aa(baa)^\omega$
- $aba\underbrace{bb}_{2}a\underbrace{bbb}_{3}a\underbrace{bbbb}_{4}a\underbrace{bbbbb}_{5}a\cdots$ is not ultimately periodic

Theorem (Büchi'62)

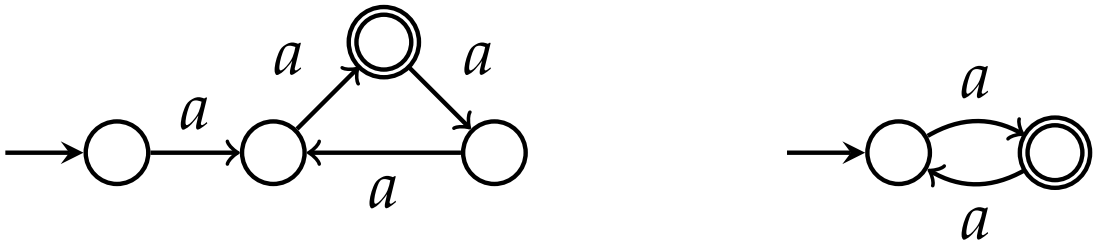
Two Büchi automata are equivalent if, and only if, they accept the same ultimately periodic words. For inclusion the corresponding result holds.

Can we count runs for ultimately periodic words?

Unambiguous Büchi Automata

Problems when trying to lift the counting method:

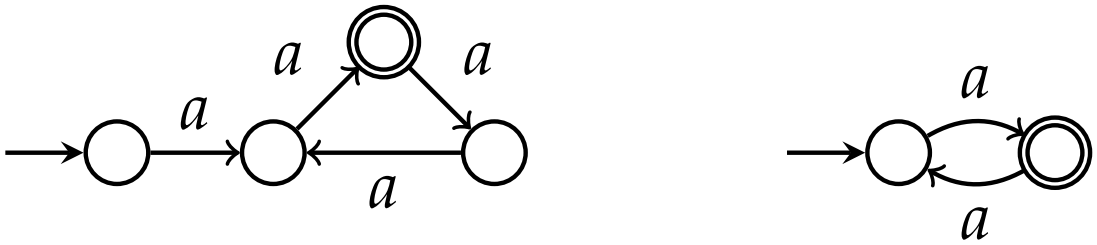
Decompositions of ultimately periodic words are not unique:



Unambiguous Büchi Automata

Problems when trying to lift the counting method:

Decompositions of ultimately periodic words are not unique:



Working with ultimately periodic words is difficult, even for deterministic automata:

Proposition (Bousquet/L.2010)

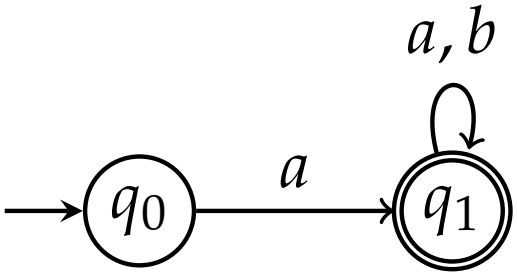
Deciding whether a given deterministic Büchi automaton accepts an ultimately periodic word with period of length n for given n is NP-hard.

Outline

Strong Unambiguity

Definition: A Büchi automaton is called strongly unambiguous if for each word there is at most one state from which it is accepted.

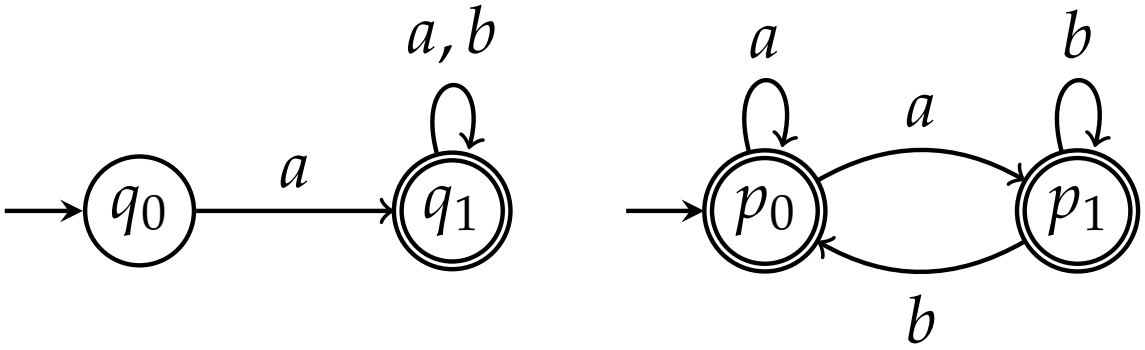
Remark 1: Determinism does not imply strong unambiguity:



Strong Unambiguity

Definition: A Büchi automaton is called strongly unambiguous if for each word there is at most one state from which it is accepted.

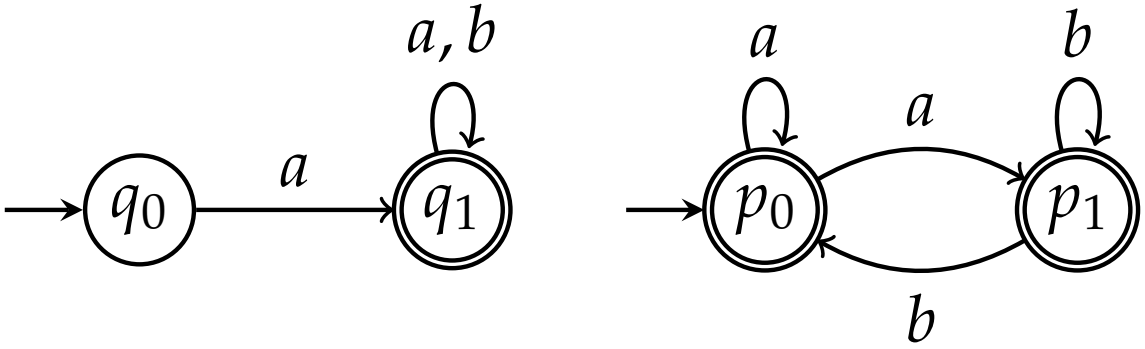
Remark 1: Determinism does not imply strong unambiguity:



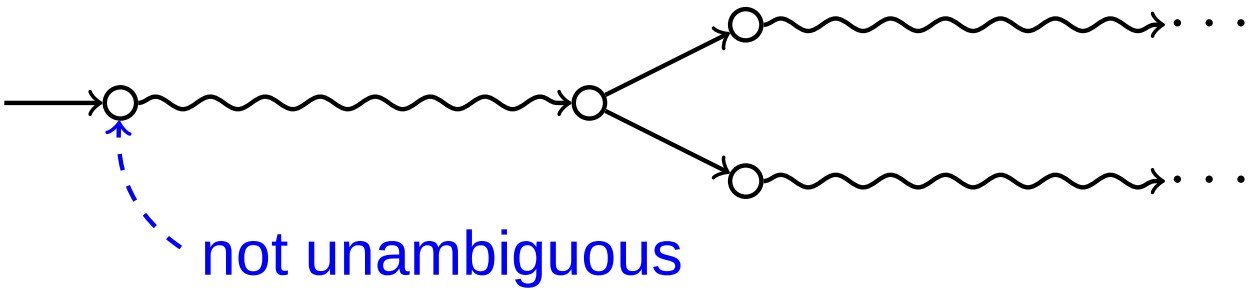
Strong Unambiguity

Definition: A Büchi automaton is called strongly unambiguous if for each word there is at most one state from which it is accepted.

Remark 1: Determinism does not imply strong unambiguity:



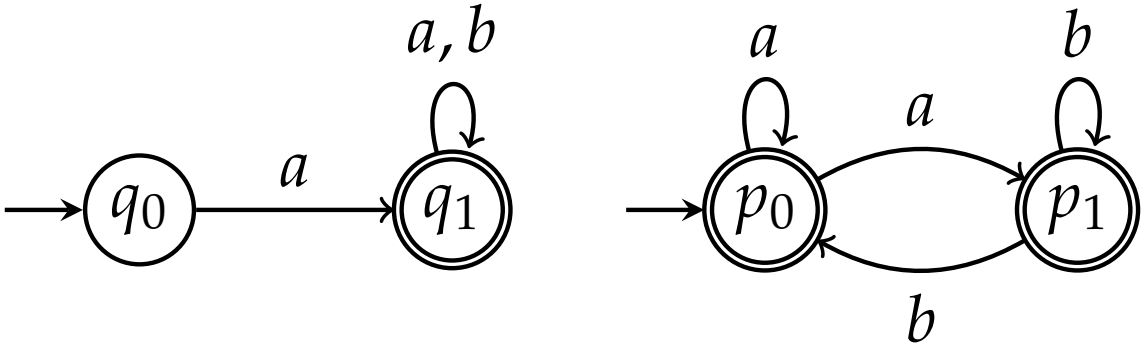
Remark 2: Strong unambiguity implies unambiguity.



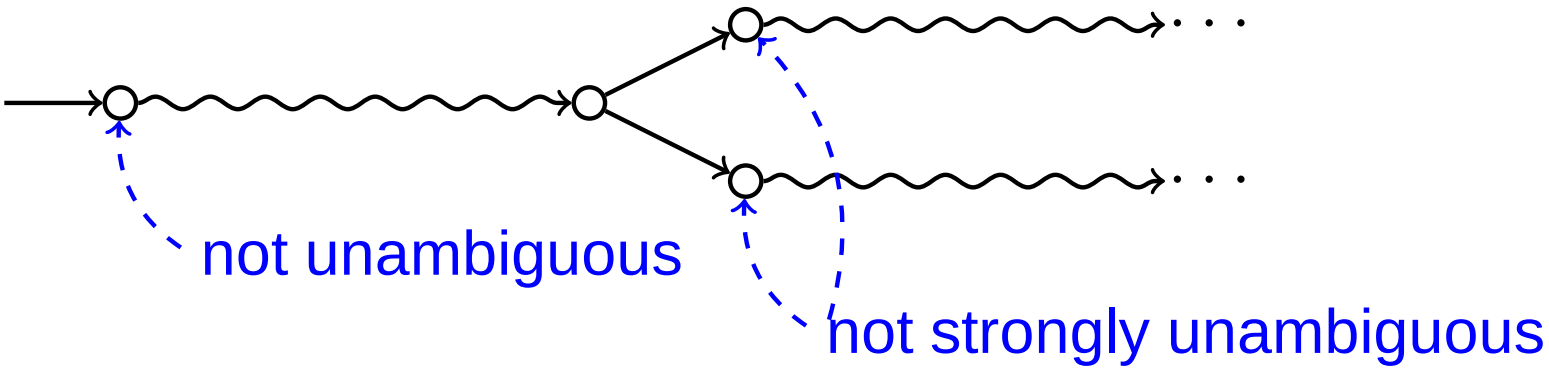
Strong Unambiguity

Definition: A Büchi automaton is called strongly unambiguous if for each word there is at most one state from which it is accepted.

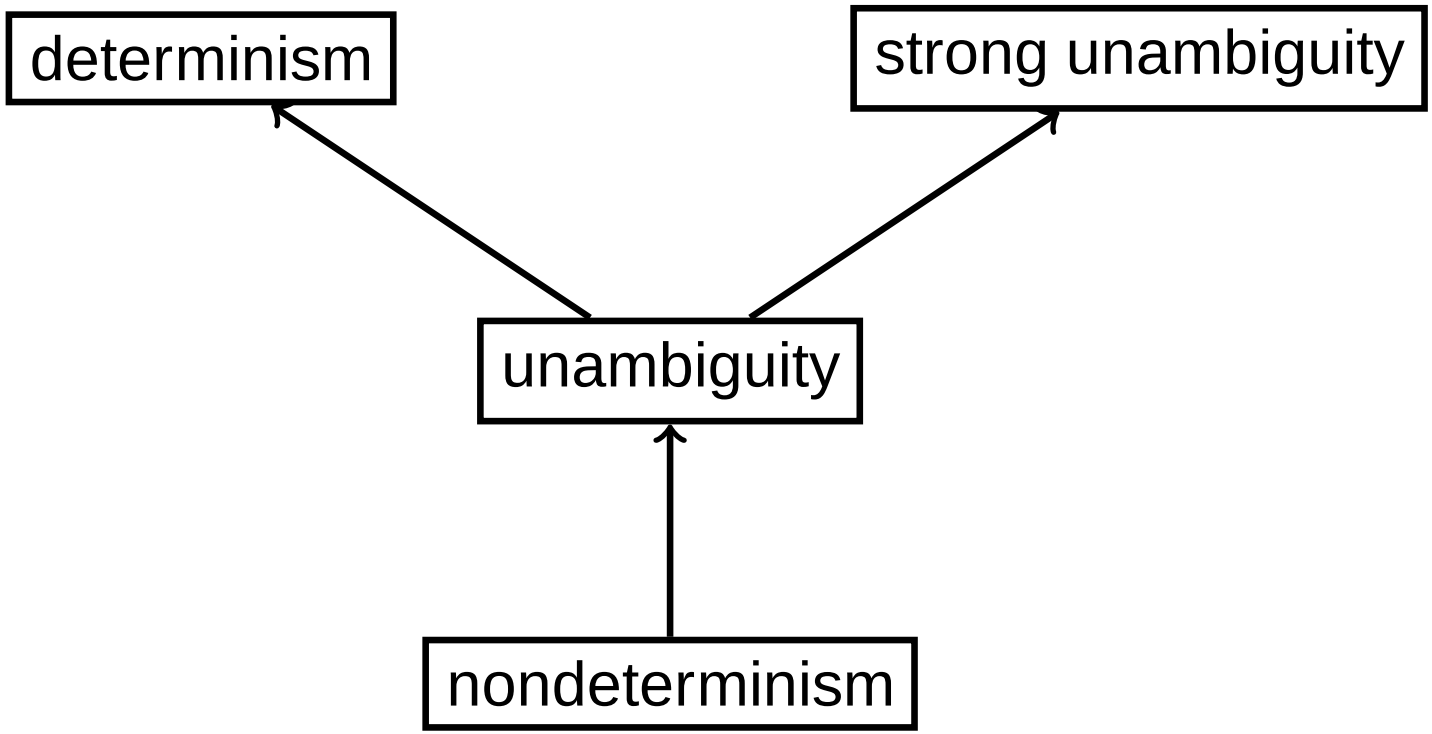
Remark 1: Determinism does not imply strong unambiguity:



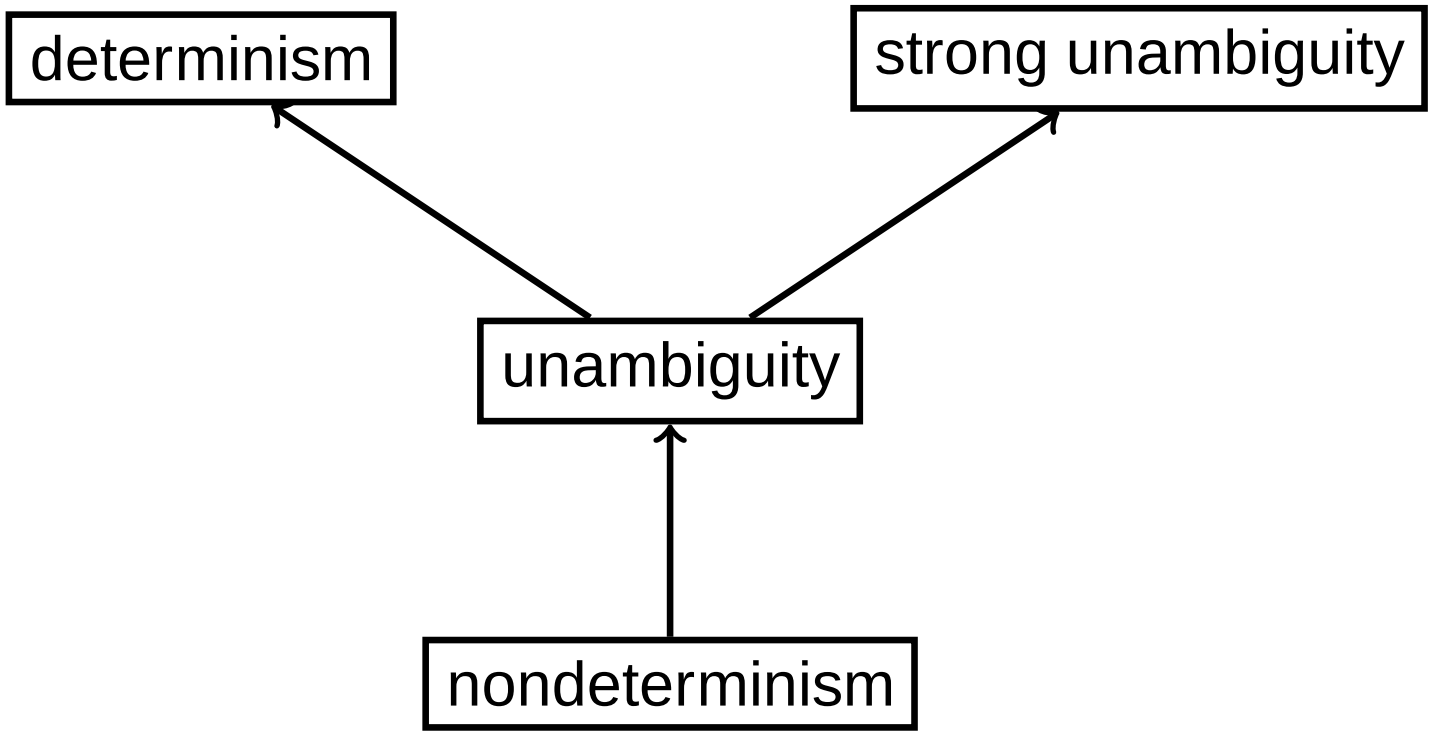
Remark 2: Strong unambiguity implies unambiguity.



Overview


determinism
strong unambiguity
unambiguity
nondeterminism

Overview



Theorem (Carton, Michel 2003)

For every nondeterministic Büchi automaton there is an equivalent strongly unambiguous Büchi automaton.

Reduction to finite words

Theorem (Bousquet/L. 2010)

There is a polynomial time reduction from the equivalence (resp. inclusion) problem for strongly unambiguous Büchi automata to the equivalence (resp. inclusion) problem for unambiguous automata on finite words.

Reduction to finite words

Theorem (Bousquet/L. 2010)

There is a polynomial time reduction from the equivalence (resp. inclusion) problem for strongly unambiguous Büchi automata to the equivalence (resp. inclusion) problem for unambiguous automata on finite words.

Proof steps: for a strongly unambiguous Büchi automaton $\mathcal{A}$

- ultimately periodic words uv^ω are accepted with short loops:

Reduction to finite words

Theorem (Bousquet/L. 2010)

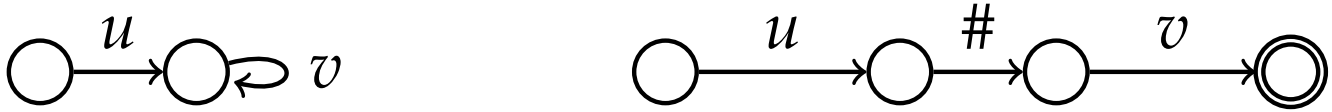
There is a polynomial time reduction from the equivalence (resp. inclusion) problem for strongly unambiguous Büchi automata to the equivalence (resp. inclusion) problem for unambiguous automata on finite words.

Proof steps: for a strongly unambiguous Büchi automaton $\mathcal{A}$

- ultimately periodic words uv^ω are accepted with short loops:

- construct polynomial size unambiguous NFA $\mathcal{A}'$ such that

uv^ω is accepted by $\mathcal{A}$ iff $u\#v$ is accepted by $\mathcal{A}'$



Reduction to finite words

Theorem (Bousquet/L. 2010)

There is a polynomial time reduction from the equivalence (resp. inclusion) problem for strongly unambiguous Büchi automata to the equivalence (resp. inclusion) problem for unambiguous automata on finite words.

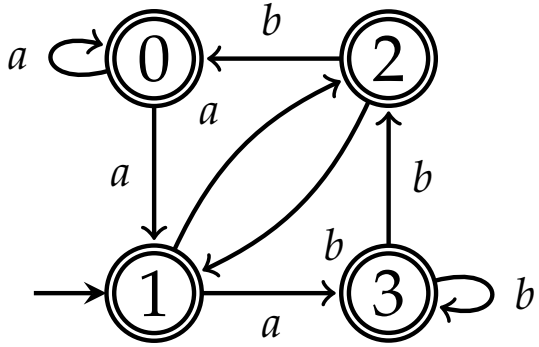
Corollary

The equivalence and inclusion problems for strongly unambiguous Büchi automata are in PTIME.

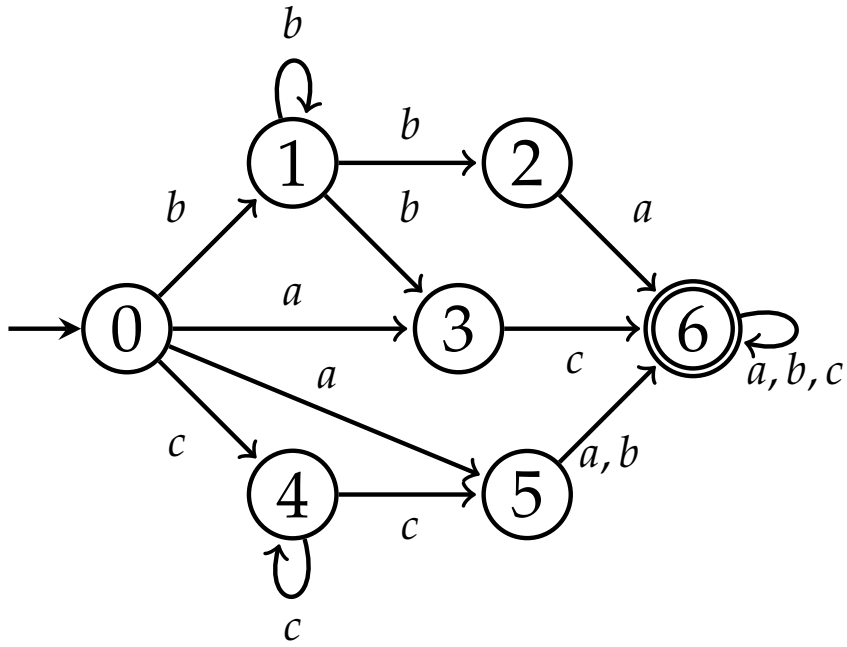
Outline

Safety and reachability automata

Safety automata: Büchi automata in which all states are accepting, except a rejecting sink state (usually omitted):



Reachability automata: Büchi automata in which all states are rejecting, except one accepting sink state:



Result on safety an reachability automata

Theorem (Isaak/L. 2012)

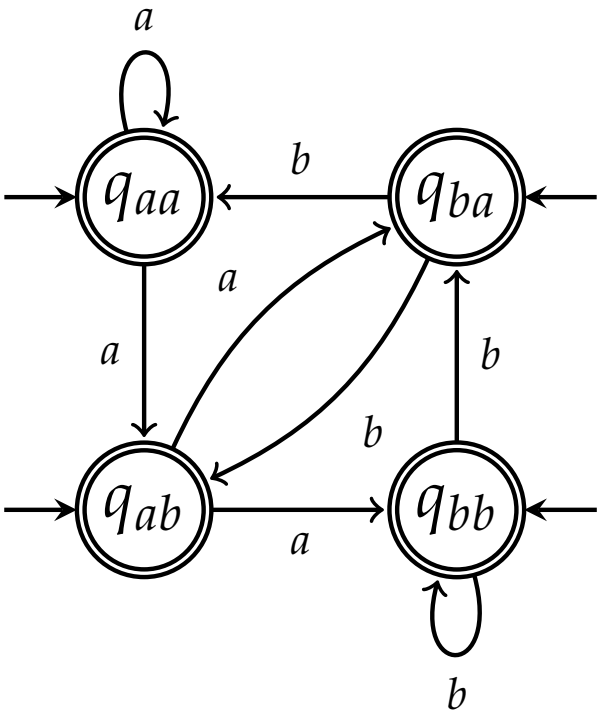
For unambiguous safety and reachability automata the problems of inclusion, equivalence, and universality can be solved in polynomial time.

A simple case: universality of safety automata

Let $\mathcal{A}$ be an unambiguous safety automaton.

Observation: $\mathcal{A}$ accepts all infinite words iff $\mathcal{A}$ considered as NFA accepts all finite words.

But: $\mathcal{A}$ considered as NFA needs not to be unambiguous.

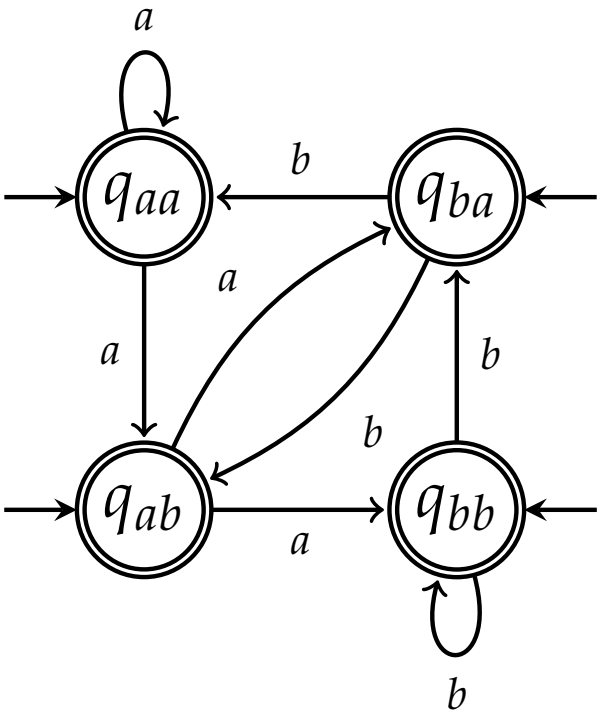


A simple case: universality of safety automata

Let $\mathcal{A}$ be an unambiguous safety automaton.

Observation: $\mathcal{A}$ accepts all infinite words iff $\mathcal{A}$ considered as NFA accepts all finite words.

But: $\mathcal{A}$ considered as NFA needs not to be unambiguous.



Solution:

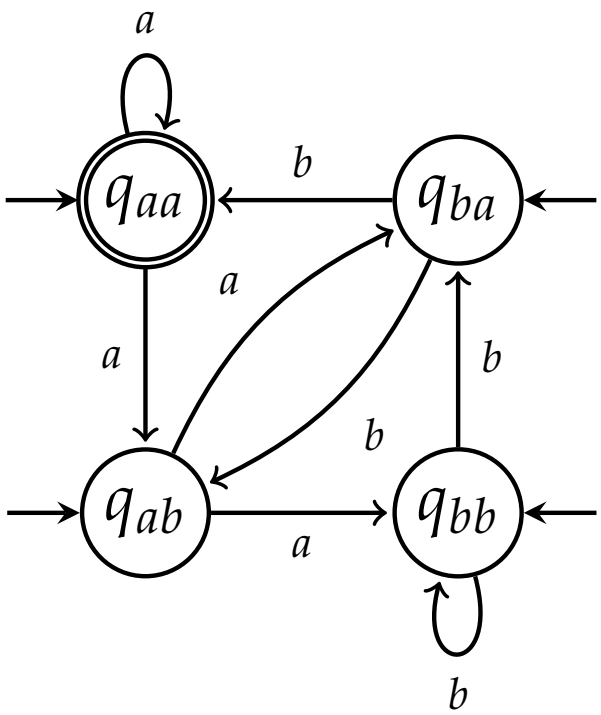
- Pick some infinite word, say a^ω.
- Declare those states final, from which a^ω is accepted.
- The resulting NFA is unambiguous and accepts all finite words iff $\mathcal{A}$ accepts all infinite words.

A simple case: universality of safety automata

Let $\mathcal{A}$ be an unambiguous safety automaton.

Observation: $\mathcal{A}$ accepts all infinite words iff $\mathcal{A}$ considered as NFA accepts all finite words.

But: $\mathcal{A}$ considered as NFA needs not to be unambiguous.



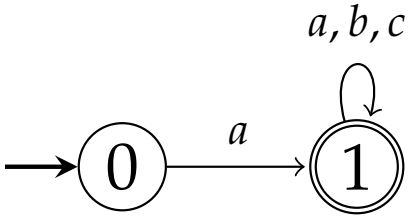
Solution:

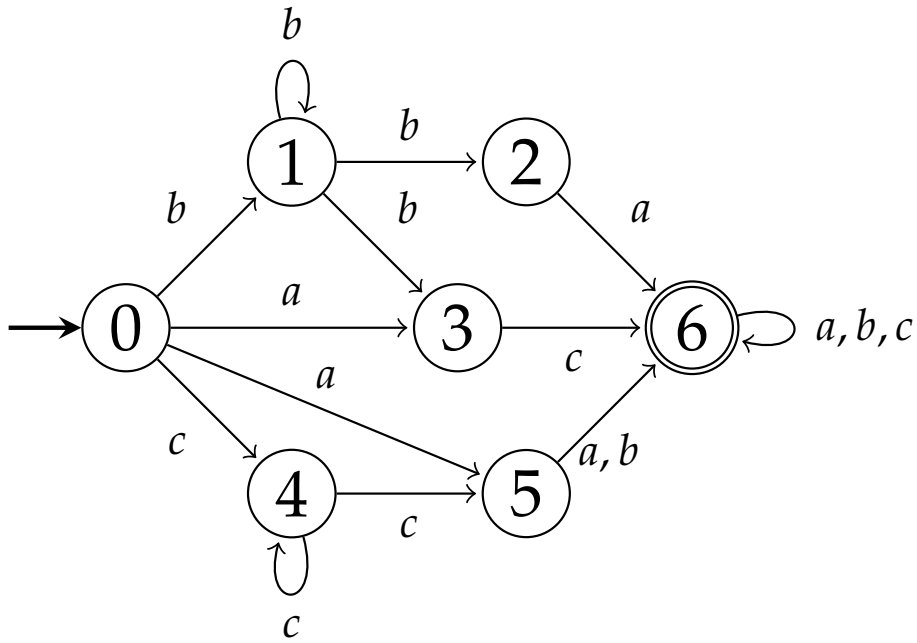
- Pick some infinite word, say a^ω.
- Declare those states final, from which a^ω is accepted.
- The resulting NFA is unambiguous and accepts all finite words iff $\mathcal{A}$ accepts all infinite words.

Reachability automata

For reachability automata $\mathcal{A}$ the situation is dual:

- $\mathcal{A}$ considered as NFA is unambiguous if $\mathcal{A}$ is unambiguous.
- However, language inclusion or equivalence is not preserved when considering the automata as NFAs.

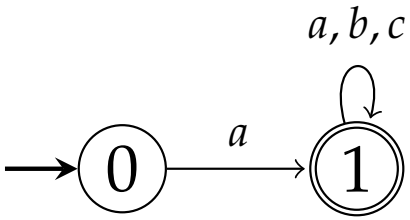


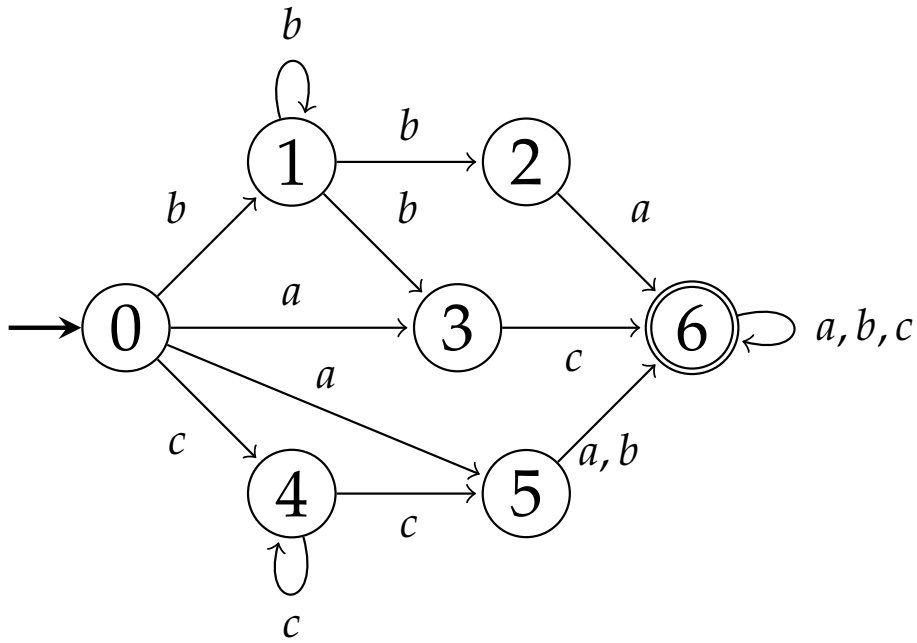


Reachability automata

For reachability automata $\mathcal{A}$ the situation is dual:

- $\mathcal{A}$ considered as NFA is unambiguous if $\mathcal{A}$ is unambiguous.
- However, language inclusion or equivalence is not preserved when considering the automata as NFAs.





As for safety automata, a reduction to the finite word case can be achieved (basically) by recomputing the set of final states.

The general case

Question: What is the complexity of inclusion, equivalence, universality for the full class of unambiguous Büchi automata?

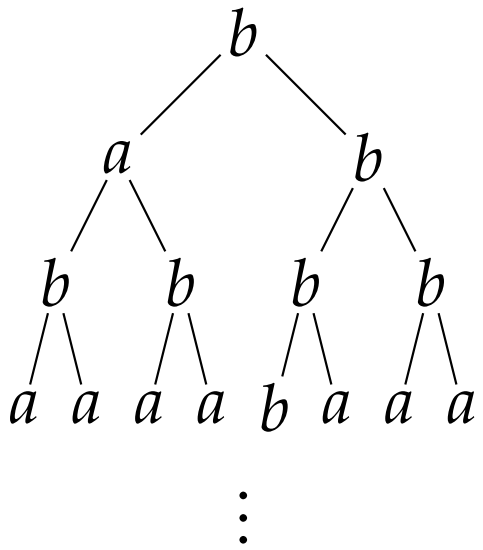
The methods developed so far do not seem to generalize.

Outline

Automata on infinite trees

- Automata $\mathcal{A} = (Q, \Sigma, q_0, \Delta, Acc)$ on infinite binary trees
- Transitions of the form (q, a, q', q'')

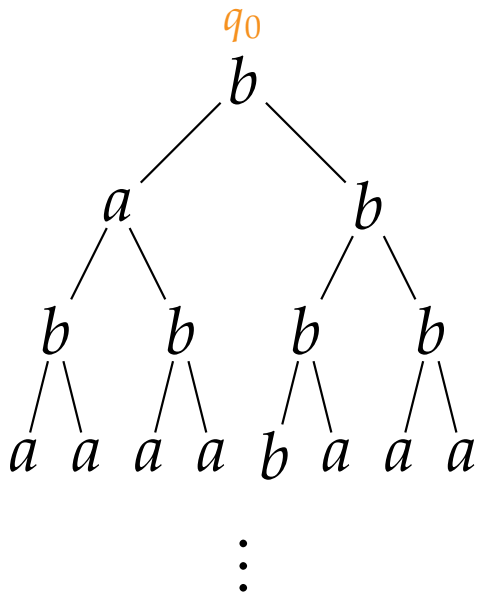
Run of $\mathcal{A}$:



Automata on infinite trees

- Automata $\mathcal{A} = (Q, \Sigma, q_0, \Delta, Acc)$ on infinite binary trees
- Transitions of the form (q, a, q', q'')

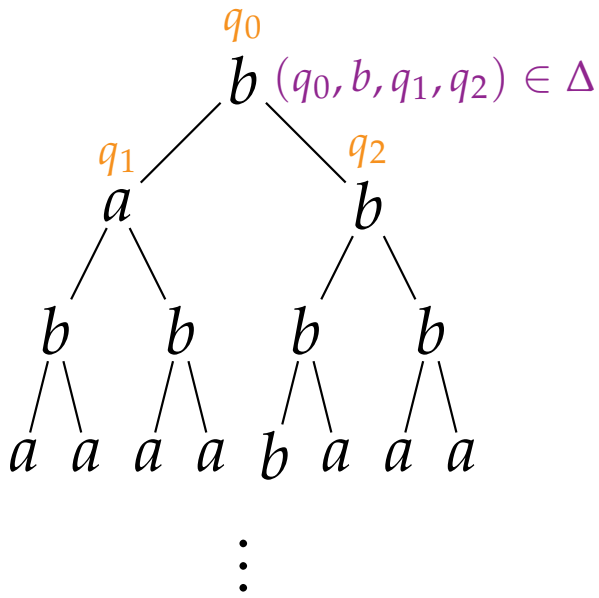
Run of $\mathcal{A}$:



Automata on infinite trees

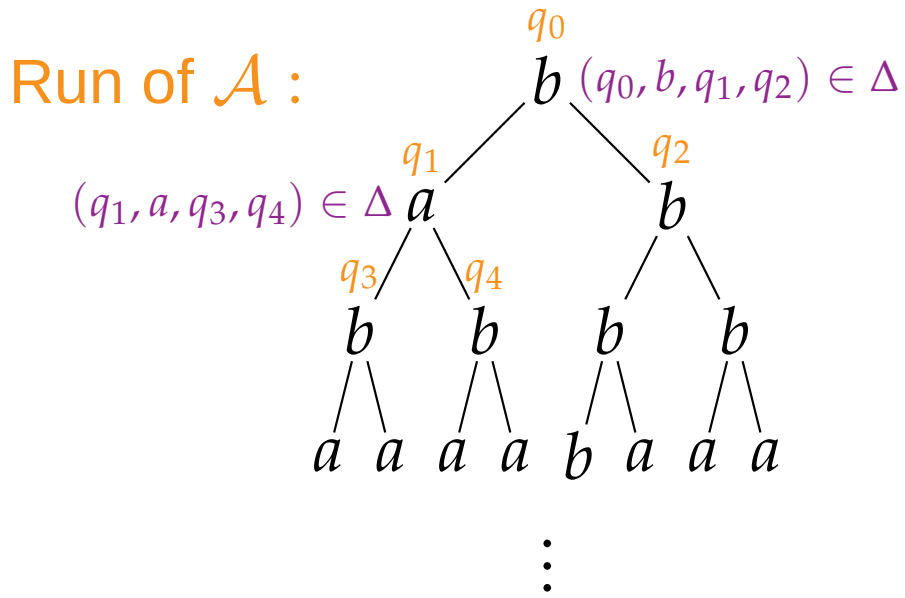
- Automata $\mathcal{A} = (Q, \Sigma, q_0, \Delta, Acc)$ on infinite binary trees
- Transitions of the form (q, a, q', q'')

Run of $\mathcal{A}$:



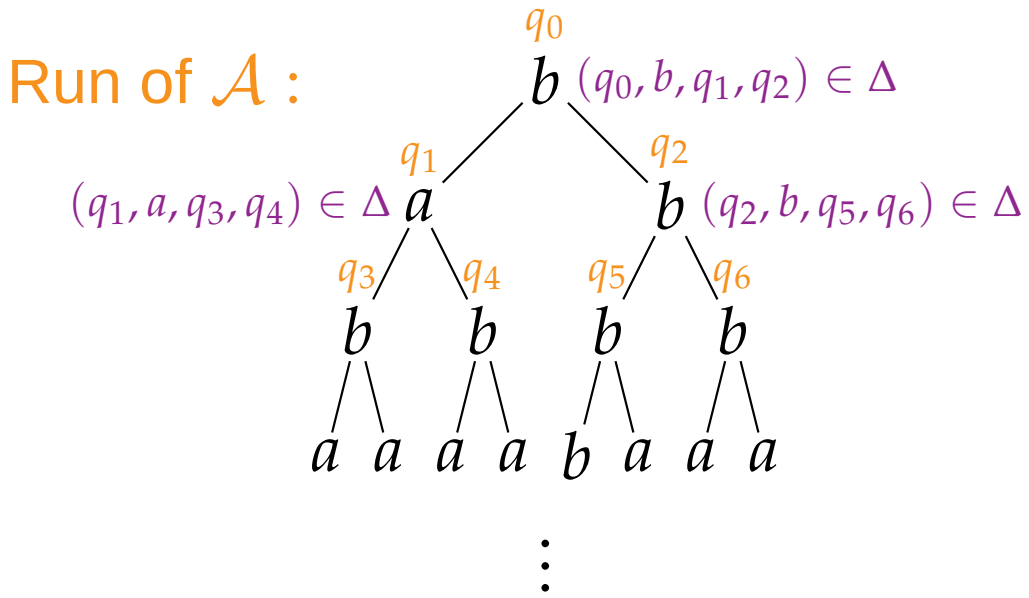
Automata on infinite trees

- Automata $\mathcal{A} = (Q, \Sigma, q_0, \Delta, Acc)$ on infinite binary trees
- Transitions of the form (q, a, q', q'')



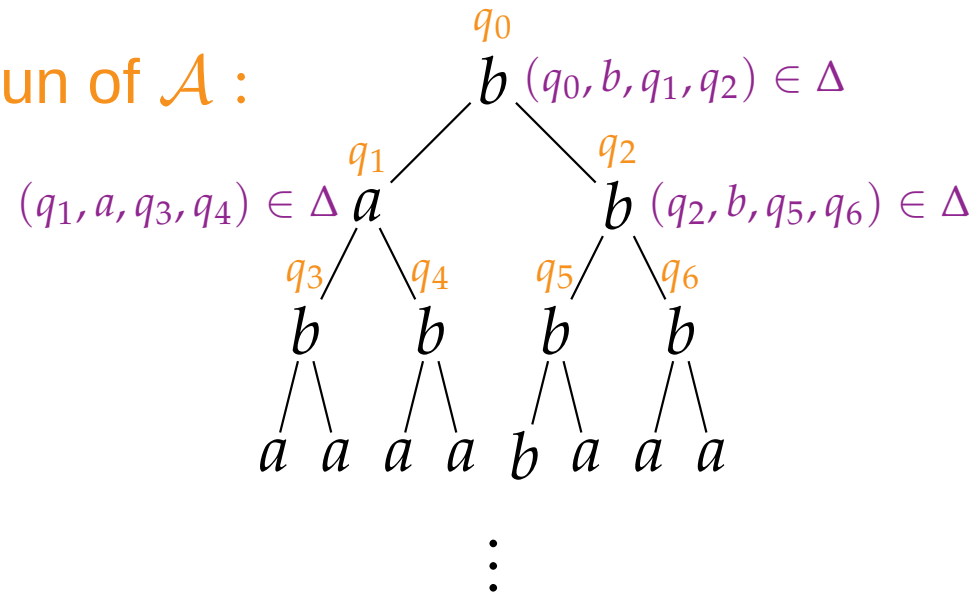
Automata on infinite trees

- Automata $\mathcal{A} = (Q, \Sigma, q_0, \Delta, Acc)$ on infinite binary trees
- Transitions of the form (q, a, q', q'')



Automata on infinite trees

- Automata $\mathcal{A} = (Q, \Sigma, q_0, \Delta, Acc)$ on infinite binary trees
- Transitions of the form (q, a, q', q'')



- $Acc \subseteq Q^\omega$ (conditions as for automata on infinite words can be used)
- Run accepting if state sequence on each path is in Acc.

Example

The set T_a of all trees t over $\{a, b\}$ such that t contains at least one node labeled a:

- States: q_a that nondeterministically searches for an a, q for the rest of the tree
- Initial state: q_a
- Transitions: (q_a, b, q_a, q) (q_a, b, q, q_a) (q_a, a, q, q)
- Acceptance: on each path finally only q.

Nondeterminism is required for this language.

Ambiguous languages

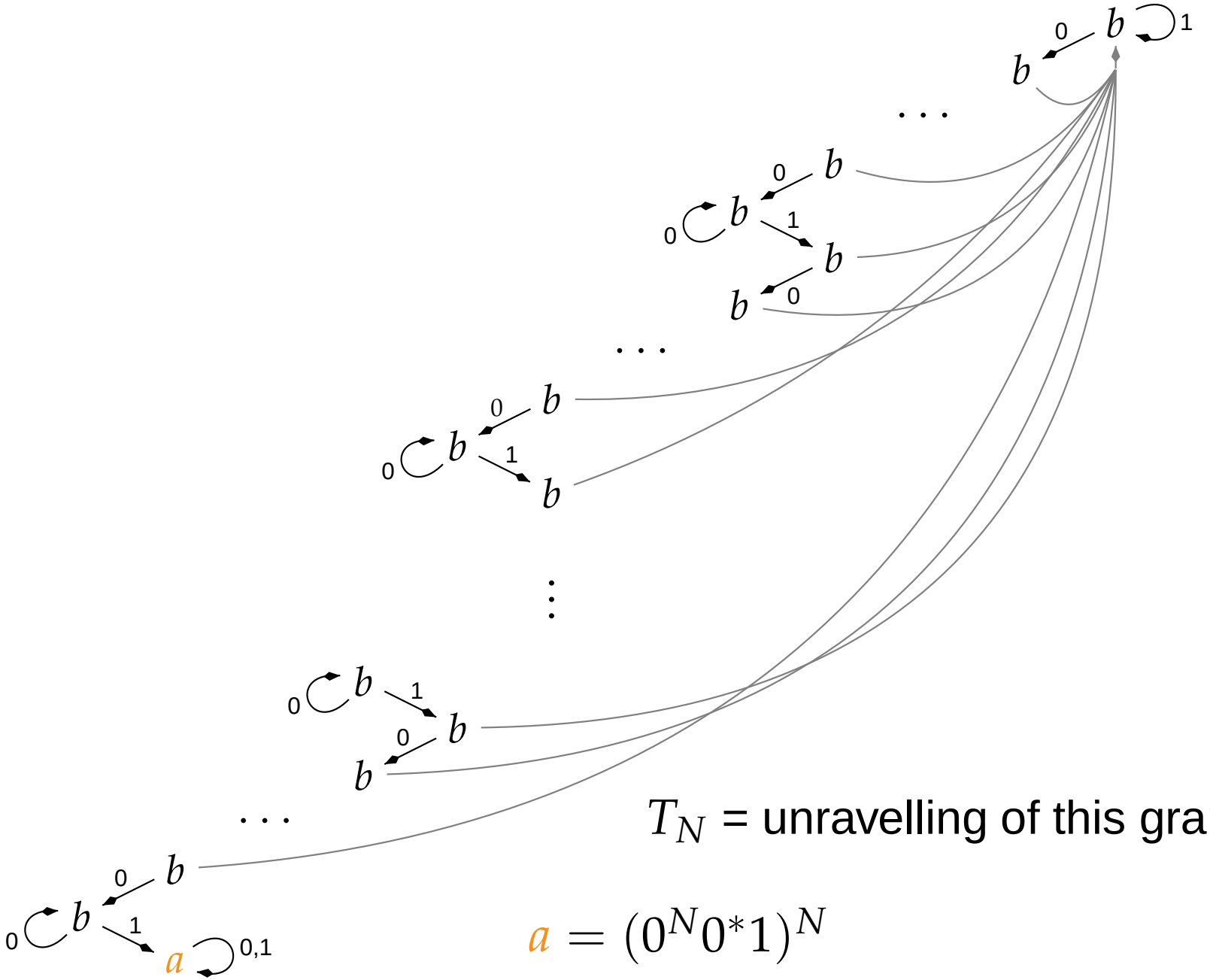
Theorem (Carayol/L./Niwinski/Walukiewicz 2010)

There are languages of infinite trees that can be accepted by a nondeterministic automaton but not by an unambiguous automaton. The language T_a is such a language.

Proof idea:

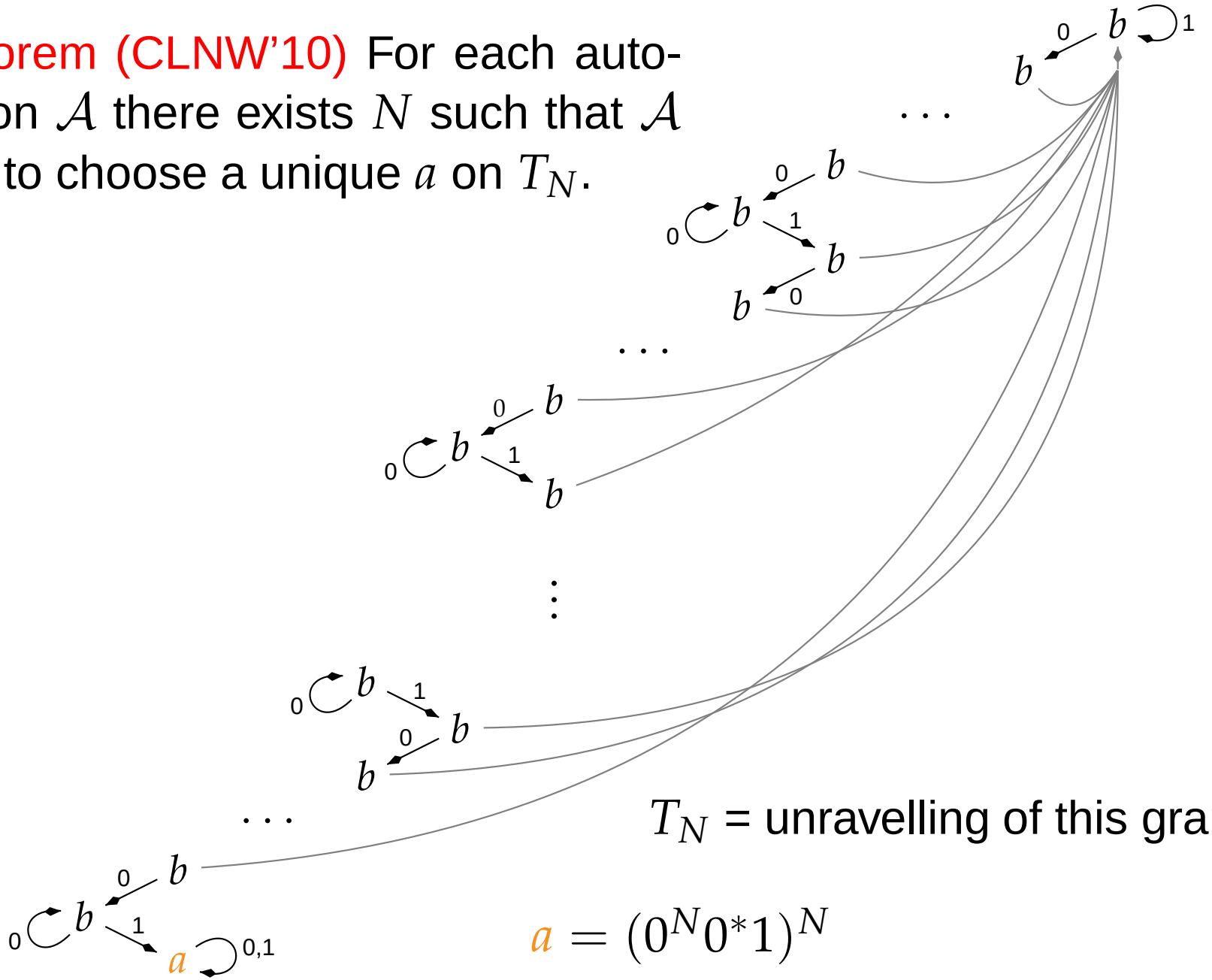
- An automaton for T_a can be modified, such that it marks in each accepting run exactly one a (by a special state, say).
- An unambiguous automaton could thus be turned into an automaton choosing for each tree in T_a exactly one a.
- Such a “choice automaton” cannot exist (Gurevich/Shelah’83, Carayol/L./Niwinski/Walukiewicz 2010)

The counterexamples



The counterexamples

Theorem (CLNW'10) For each automaton $\mathcal{A}$ there exists N such that $\mathcal{A}$ fails to choose a unique a on T_N.



T_N = unravelling of this graph

$a = (0^N 0^* 1)^N$

Deciding unambiguity

The result on inherently ambiguous languages of infinite trees raises a new question:

Is it decidable whether a given regular language of infinite trees can be accepted by an unambiguous automaton?

Deciding unambiguity

The result on inherently ambiguous languages of infinite trees raises a new question:

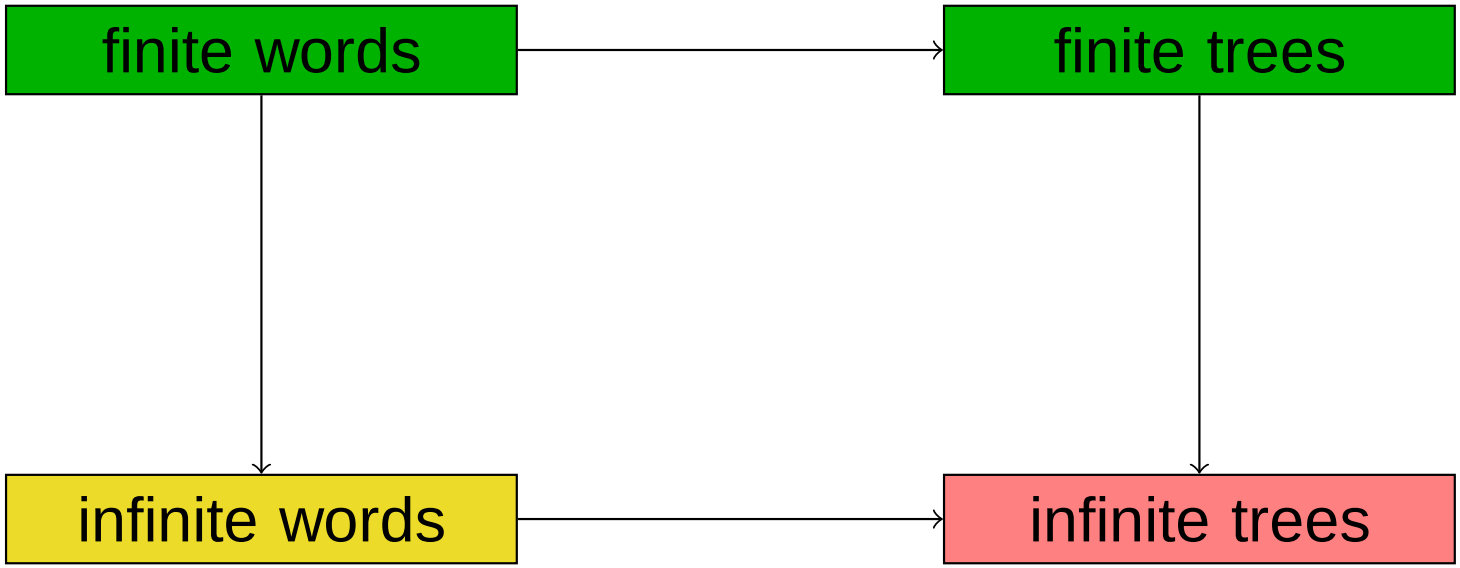
Is it decidable whether a given regular language of infinite trees can be accepted by an unambiguous automaton?

Recent partial results (Bilkowski/Skrzypczak'13):

- It is decidable for a deterministic tree automaton whether its complement is unambiguous.
- Bi-unambiguity (unambiguity of a language and its complement) is decidable if a certain conjecture on definability of choice functions on infinite trees is true.

Outline

Conclusion



Two central open questions:

- What is the complexity of universality, equivalence, inclusion testing for unambiguous Büchi automata?
- Is it decidable for a given language of infinite trees whether there is an unambiguous automaton for this language?