Infinite Computations

Lecture WS 2013/2014
Prof. Dr. W. Thomas
RWTH Aachen

Preliminary version
(Last update October 19, 2015)

Note

These lecture notes have not yet undergone a thorough revision. They may contain mistakes of any kind. Please report any bugs you find, comments, and proposals on what could be improved via e-mail to `lecture-notes@automata.rwth-aachen.de`

Contents

1 Introduction

Infinite Computations is our subject. More precisely: Automata on infinite words

1. Extension of classical finite automata theory to infinite input words (Part I - III)

2. Applications (Part II)

   (a) Decidability results in logic
   Illustration: Use $X, Y, \dots$ for sets of natural numbers, $x, y, \dots$ for natural numbers
   Check truth of the following sentence: $\forall X \exists Y (X \subseteq Y \wedge \forall x (x \in Y \rightarrow x + 1 \in Y))$. We obtain results that show how to check this automatically

   (b) Model-Checking

   Example 1. A MUX-Protocol

```
Process 1: Repeat
  00   non-critical section 1
  01   wait unless turn = 0
  10   critical section 1
  11   turn := 1
Process 2: Repeat
  00   non-critical section 2
  01   wait unless turn = 1
  10   critical section 2
  11   turn := 0
```

   State space consists of 32 bit vectors $(b_1, \dots, b_5)$ (P_1-instruction, P_2-instruction, value of turn) The protocol (P_1, P_2) “is” a nondeterministic finite automaton (without input letters), generating infinite runs. Possible correctness claims:

   When started in state $(0,0,0,0,0)$ the processes P_1, P_2 will

   i. never simultaneously visit their critical sections, i.e. never reach a state $(1,0,1,0,b)$

   ii. always from a waiting state of P_i eventually visit critical section i at a later time

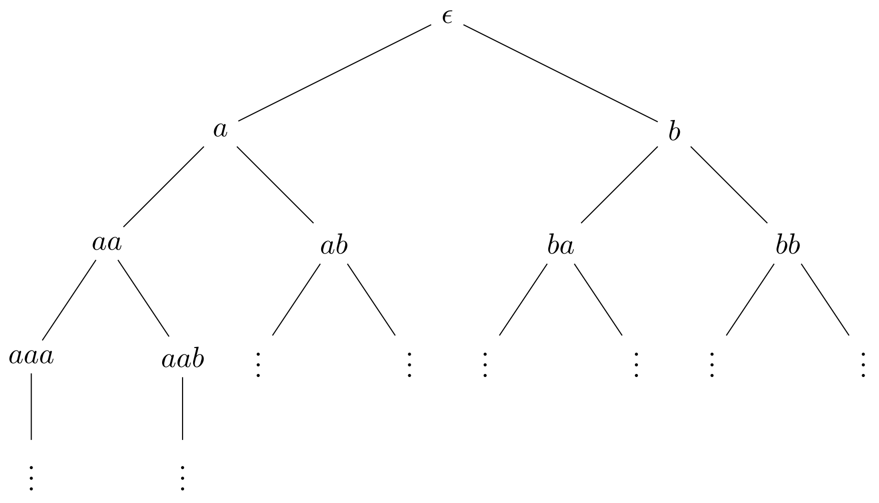
   (Property 1. is satisfied, Property 2. is not.)
   Model-Checking provides a method that does such checks automatically ⊠

   (c) Verification of hybrid systems (involving real numbers)
   Idea: Each $r \in \mathbb{R}$ is an infinite word $\pi = 3.14159\dots$, alphabet $\{0, \dots, 9, .\}$

2 Terminology

Σ	denotes a finite *alphabet*.
$\mathbb{B} = \{0,1\}$	is the Boolean alphabet.
$a, b, c, \ldots$	stand for *letters* of an alphabet.
Σ^*	is the set of finite *words* over Σ.
$u, v, w, \ldots$	stand for finite words.
ϵ	is the empty word.
$\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$	is the set of non-empty words over Σ.
$\alpha, \beta, \gamma, \ldots$	denote *ω-words* or *infinite words* where an ω-word over Σ is a sequence $\alpha = \alpha(0)\alpha(1)\ldots$ with $\alpha(i) \in \Sigma$ for all $i \in \mathbb{N}$. We write $\alpha[i,j]$ for $\alpha(i)\ldots\alpha(j)$ and $\alpha[i,j)$ for $\alpha(i)\ldots\alpha(j-1)$.
Σ^ω	is the set of infinite words over Σ.
$\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$	
$U, V, W, \ldots$	denote sets of finite words (*$*$-languages*) $\subseteq \Sigma^*$.
$K, L, M, \ldots$	denote sets of infinite words (*ω-languages*) $\subseteq \Sigma^\omega$.

It is useful to compare the two domains Σ^* and Σ^ω (for an alphabet with $|\Sigma| \geq 2$). Taking $\Sigma = \{a, b\}$, the set Σ^* is the set of *nodes* of the binary tree:



This set is *countable*: One can obtain a listing $w_0, w_1, w_2, \ldots$ of Σ^* by listing the levels of the tree from left to right. The elements of Σ^ω, however, can be represented by the paths through the binary tree. The set Σ^ω is *uncountable*, as shown by Cantor's diagonal argument: Assume, the elements of Σ^ω are listable as $\alpha_0, \alpha_1, \alpha_2, \ldots$ which leads to the diagramm

$$\begin{array}{llllll}
\alpha_0: & c_{00} & c_{01} & c_{02} & c_{03} & \cdots \\
\alpha_1: & c_{10} & c_{11} & c_{12} & c_{13} & \cdots \\
\alpha_2: & c_{20} & c_{21} & c_{22} & c_{23} & \cdots \\
 & & \cdots & & &
\end{array}$$

Using the map $\bar{}$ with $\bar{a} = b, \bar{b} = a$, we define the ω-word

$$\beta = \bar{c}_{00}\bar{c}_{11}\bar{c}_{22}\cdots$$

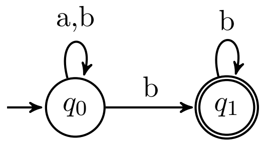
which (at position i) is different from α_i, for each i. So the sequence β does not occur in the list, contracting the assumption that all elements of Σ^ω occur in the list.

Part I
Regular ω-Languages

3 Büchi Automata

Recall: An *NFA* (nondeterministic finite automaton) has the format $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$, with $\Delta \subseteq Q \times \Sigma \times Q$

Example 2. $\mathfrak{A}_0$:



$L(\mathfrak{A}_0) = \{w \in \{a,b\}^* \mid w \text{ ends with b }\}$ ⊠

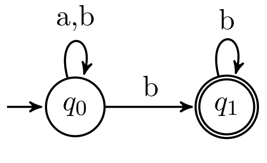
Definition 3. *A (nondeterministic)* Büchi automaton *has the format*

$$\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$$

with a finite set of states Q, input alphabet Σ, initial state $q_0 \in Q$, a transition relation $\Delta \subseteq Q \times \Sigma \times Q$, and a set of final states F. A Büchi automaton is equipped with an acceptance condition for infinite words.

- *A* run *of $\mathfrak{A}$ on $\alpha = \alpha(0)\alpha(1)\dots$ is a sequence of states $\rho = \rho(0)\rho(1)\cdots$ with $\rho(0) = q_0$ and $(\rho(i), \alpha(i), \rho(i+1)) \in \Delta$ for $i \geq 0$.*
- *Run ρ is* accepting *if $\rho(i) \in F$ for infinitly many i*
- *Automaton $\mathfrak{A}$ accepts α if exists an accepting run ρ of $\mathfrak{A}$ on α*
- *We write $L(\mathfrak{A}) = \{\alpha \in \Sigma^\omega \mid \mathfrak{A} \text{ accepts } \alpha\}$*

Example 4. $\mathfrak{A}_0$:



$$\begin{array}{lccccccccccccccccccccc}
\alpha = & & \text{a} & & \text{b} & & \text{a} & & \text{a} & & \text{b} & & \text{a} & & \text{b} & & \text{b} & & \text{b} & & \dots \\
\rho = & q_0 & & q_0 & & q_0 & & q_0 & & q_0 & & q_0 & & q_0 & & q_1 & & q_1 & & q_1 & & \dots
\end{array}$$

$$L(\mathfrak{A}_0) = \{\alpha \in \{a,b\}^\omega \mid \alpha = wbbb\dots \text{ for some } w \in \{a,b\}^*\}$$

⊠

For a Büchi automaton $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$, the acceptance condition is:
Automaton $\mathfrak{A}$ accepts $\alpha \Leftrightarrow$ Exists a run $\rho \in Q^\omega$ of $\mathfrak{A}$ on α $\underbrace{\text{s.t. for infinitely many } i : \rho(i) \in F}$

$$\begin{aligned}
&\Leftrightarrow \exists^\omega i : \rho(i) \in F \\
&\Leftrightarrow \exists^\omega i \exists q \in F : \rho(i) = q \\
&\Leftrightarrow \exists q \in F \exists^\omega i : \rho(i) = q \ (F \text{ finite!})
\end{aligned}$$

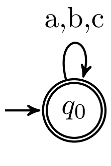
Definition 5. *A* deterministic Büchi automaton *has the format*

$$\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$$

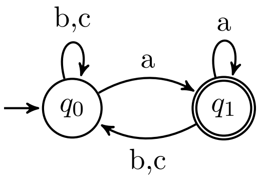
with a finite set of states Q, input alphabet Σ, initial state $q_0 \in Q$, a transition function $\delta : Q \times \Sigma \to Q$, and a set of final states F.

- *On each $\alpha \in \Sigma^\omega, \alpha = \alpha(0)\alpha(1)\ldots$, there is a unique run ρ of $\mathfrak{A}$ on α defined by $\rho(0) = q_0, \rho(i+1) = \delta(\rho(i), \alpha(i))$.*
- *Automaton $\mathfrak{A}$ accepts α*
  $:\Leftrightarrow$ for the unique run ρ of $\mathfrak{A}$ on α we have: $\exists^\omega i\colon \rho(i) \in F$
- *$L(\mathfrak{A}) = \{\alpha \in \Sigma^\omega \mid \mathfrak{A}$ accepts $\alpha\}$*

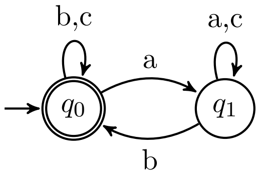
Example 6. Let $\Sigma = \{a, b, c\}$
$L_1 = \Sigma^\omega$ is accepted by $\mathfrak{A}_1$:



$L_2 = \{\alpha \in \Sigma^\omega \mid \underbrace{a \text{ occurs infinitely often}}_{\exists^\omega i\colon \alpha(i)=a}\}$ is accepted by $\mathfrak{A}_2$:



$L_3 = \{\alpha \in \Sigma^\omega \mid \underbrace{\text{in } \alpha \text{ after each } a \text{ sometimes later there is a } b}_{\forall i\left(\alpha(i)=a \to \exists j(i<j \wedge \alpha(j)=b)\right)}\}$ is accepted by $\mathfrak{A}_3$:



⊠

Theorem 7. *The ω-language*

$$L(\mathfrak{A}_0) = \{\alpha \in \{a, b\}^\omega \mid \textit{from some point in } \alpha \textit{ onwards, only letter } b \textit{ occurs}\}$$

can be recognized by a nondeterministic Büchi automaton, but not by a deterministic one.

Proof The NBA $\mathfrak{A}_0$ above shows the first claim.
Assume that $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$ recognizes L_0. Then the following holds:

- $\mathfrak{A}$ accepts b^ω, so in the accepting run ρ one can pick i_0 so that $\rho(i_0) \in F$
- $\mathfrak{A}$ accepts $b^{i_0}ab^\omega$, so in the accepting run ρ' pick i_1 such that for $\underbrace{b^{i_0}a\overbrace{bbb\dots bbb}^{i_1}}_{j}$ we have $\rho'(j) \in F$
- $\mathfrak{A}$ accepts $b^{i_0}ab^{i_1}ab^\omega$, so in the accepting run ρ'' there is later again a visit of F, say after $b^{i_0}ab^{i_1}a\underbrace{bbb\dots bbb}_{i_2}$.

Repeating this we obtain an ω-word $b^{i_0}ab^{i_1}ab^{i_2}ab^{i_3}\dots$ accepted by $\mathfrak{A}$ (final state visited before each a). Contradiction, since this word is outside L_0. $\square$

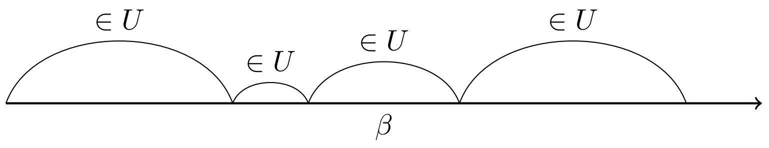
Corollary 8. *The class of ω-languages recognized by deterministic Büchi automata is not closed under complement.*

Next aim: Investigate in general which ω-languages are recognized by nondeterministic Büchi automata.

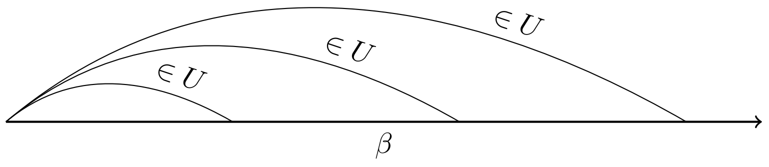
4 Regular ω-languages

Definition 9. ***(Language Operations)*** *Let $U \subseteq \Sigma^*$, $L \subseteq \Sigma^\omega$:*

- $U \cdot L := \{\beta \in \Sigma^\omega \mid \beta = u\alpha$ *with* $u \in U, \alpha \in L\}$
- $U^\omega := \{\beta \in \Sigma^\omega \mid \beta = u_0u_1u_2\dots$ *with* $u_i \in U\}$
  Illustration:



- $\lim U := \{\beta \in \Sigma^\omega \mid$ *there exist infinitely many* i *with* $\beta(0)\dots\beta(i) \in U\}$
  Illustration:

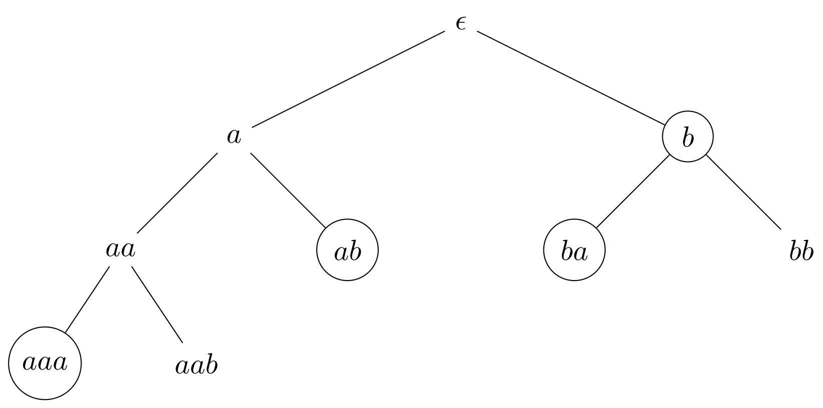


Example 10.
$U_1 = ab^*$
$U_1^\omega = \{\beta \mid \beta$ starts with a and has infinitely many $a\}$
$\lim U_1 = \{abbb\dots\}$

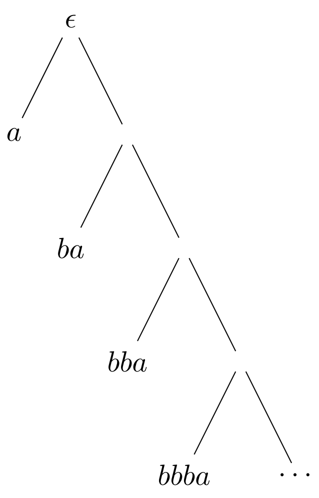
$U_2 = a^*b$
$U_2^\omega = \{\beta \mid \beta$ has infinitely many $b\}$
$\lim U_2 = \emptyset$

$U_3 = (a+b)^*b$
$U_3^\omega = U_2^\omega$
$\lim U_3 = U_2^\omega$ ⊠

As a further illustration, we use the representation of Σ^* as a tree. With $\Sigma = \{a, b\}$ and $U = \{b, ab, ba, aaa\}$ we get the following situation:



We see that $U \cdot \Sigma^\omega$ consists of all paths which miss the node aab.
To illustrate lim, consider $U_0 = \{a, ba, bba, bbba, \dots\}$



Each path α has either just one prefix in U_0 or none at all (if $\alpha = bbb\dots$). So $\lim(U_0) = \emptyset$.

Definition 11. *An ω-language is* regular *if it can be represented in the form*

$$L = \bigcup_{i=0}^{n} U_i \cdot V_i^\omega$$

with regular $U_1, V_1, \dots, U_n, V_n \subseteq \Sigma^*$. We introduce corresponding regular expressions of the form $r_1 s_1^\omega + \dots + r_n s_n^\omega$, with r_i defining U_i and s_i defining V_i (r_i, s_i are standard regular expressions).

Theorem 12.

a) L NBA-recognizable $\Leftrightarrow$ L regular

b) L DBA-recognizable $\Leftrightarrow$ $L = \lim U$ for some regular language $U \subseteq \Sigma^*$

Proof

b) $\Rightarrow$: Assume DBA $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$ recognizes L. Find a finite automaton for a suitable set U so that $L(\mathfrak{A}) = \lim U$. Use $\mathfrak{A}$ as the finite automaton, it defines a regular set U.
DBA $\mathfrak{A}$ accepts $\alpha \Leftrightarrow$ the unique run ρ of $\mathfrak{A}$ on α visits F infinitely often $\Leftrightarrow$ $\alpha \in \lim U$

$\Leftarrow$: Analogously

a) $\Rightarrow$: Assume NBA $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$ recognizes L.
NBA $\mathfrak{A}$ accepts $\alpha = u_0u_1u_2\dots \Leftrightarrow$ exists run ρ of $\mathfrak{A}$ on α and a final state such that we have

$$\mathfrak{A} : q_0 \xrightarrow{u_0} q \xrightarrow{u_1} q \xrightarrow{u_2} q \dots$$

where $\mathfrak{A} : p \xrightarrow{u} q$ indicates that there is a finite run of $\mathfrak{A}$ from p to q via u. Define for $p, q \in Q$:
$\mathfrak{A}_{pq} = (Q, \Sigma, p, \Delta, \{q\})$ and $W_{pq} := L(\mathfrak{A}_{pq})$. Each W_{pq} is regular. Then

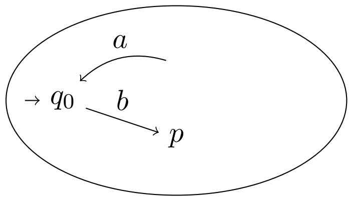
$$L = \bigcup_{q \in F} W_{q_0q} \cdot W_{qq}^{\omega}$$

This is a representation as desired.

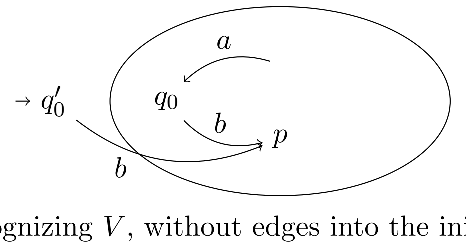
$\Leftarrow$: It suffices to show:

i. $V \subseteq \Sigma^*$ regular $\Rightarrow V^\omega$ Büchi recognizable.
ii. $U \subseteq \Sigma^*$ regular, $K \subseteq \Sigma^\omega$ Büchi recognizable $\Rightarrow U \cdot K$ Büchi recognizable.
iii. $K_1, K_2 \subseteq \Sigma^\omega$ Büchi recognizable $\Rightarrow K_1 \cup K_2$ Büchi recognizable.

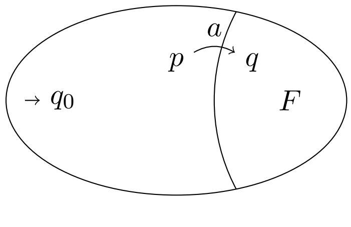
For i: Take nondeterministic finite automaton $\mathfrak{A}$ recognizing V. We construct a Büchi-automaton $\mathfrak{B}$ for V^ω. If appropiate modify $\mathfrak{A}$ such that the initial state has no incoming edges. This is done as follows: Introduce a new initial state q_0'. Pass from



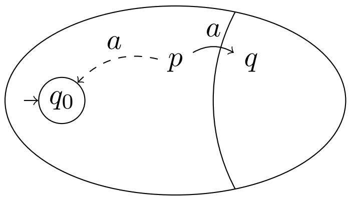
to



Now consider $\mathfrak{A}$ recognizing V, without edges into the initial state:



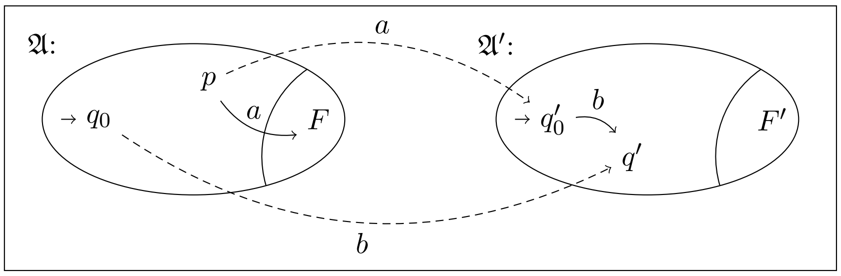
For each transition (p,a,q) add a transition (p,a,q_0) and declare $\{q_0\}$ as new set of final states:



For ii: Let $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$ be an NFA, which recognizes the language U, and let $\mathfrak{A}' = (Q', \Sigma, q_0', \Delta', F')$ be a Büchi automaton, which recognizes the language K. Now, construct a Büchi automaton $\mathfrak{B} = (Q \uplus Q', \Sigma, q_0, \Delta_{\mathfrak{B}}, F')$ for $U \cdot K$, where $\Delta_{\mathfrak{B}}$ contains, in addition to the transitions of Δ and Δ', the following:

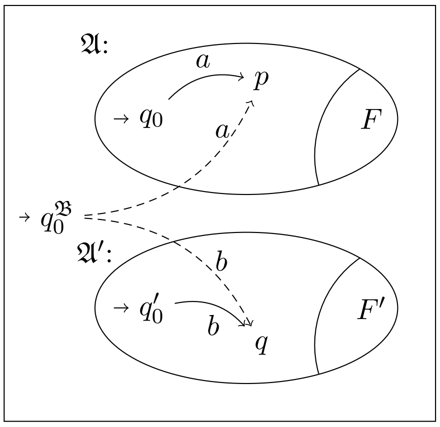
- for every transition (p,a,q) with $q \in F$ the transition (p,a,q_0')
- if $q_0 \in F$, for every transition $(q_0', b, q') \in \Delta'$ the transition (q_0, b, q').

$\mathfrak{B}$:



For iii: Merge the Büchi automata $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$ and $\mathfrak{A}' = (Q', \Sigma, q_0', \Delta', F')$ into a single automaton $\mathfrak{B} = (Q \uplus Q' \cup \{q_0^{\mathfrak{B}}\}, \Sigma, q_0^{\mathfrak{B}}, \Delta_{\mathfrak{B}}, F)$ with a new initial state $q_0^{\mathfrak{B}}$, where $\Delta_{\mathfrak{B}}$ contains all transitions of Δ, Δ', as well as $(q_0^{\mathfrak{B}}, a, p)$ for $(q_0, a, p) \in \Delta$ and $(q_0^{\mathfrak{B}}, b, q)$ for $(q_0', b, q) \in \Delta'$.

$\mathfrak{B}$:



□

Towards closure under intersection: Generalized Büchi automata

Definition 13. *A* generalized Büchi automaton *is of the form* $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F_1, \ldots, F_k)$ *with final state sets* $F_1, \ldots, F_k \subseteq Q$*. A run is successful if the automaton visits each of the sets* F_i *(i.e. the automaton enters a state in each* F_i*) infinitely often.*

Theorem 14. *The intersection* $L_1 \cap L_2$ *of two Büchi recognizable* ω*-languages* L_1, L_2 *is Büchi recognizable by a generalized Büchi automaton.*

Proof Assume L_i is recognized by the Büchi automaton $\mathfrak{A}_i = (Q_i, \Sigma, q_{i0}, \Delta_i, F_i)$ for $i = 1, 2$. Form the generalized automaton

$$(Q_1 \times Q_2, \Sigma, (q_{10}, q_{20}), \Delta, F_1 \times Q_2, Q_1 \times F_2),$$

clearly it recognizes $L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$. □

Theorem 15. *For each generalized Büchi automaton one can construct an equivalent (standard) Büchi automaton.*

Idea: Work with the state set $Q \times \{1, \ldots, k\}$. For $i = 1, \ldots, k$ a state (q, i) means "wait for visit to F_i". After visiting F_k go back to $i = 1$. Consequently, declare $F_k \times \{k\}$ as the set of final states.

Proof Given a generalized Büchi automaton $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F_1, \ldots, F_k)$, construct the Büchi automaton

$$\mathfrak{A}' = (Q \times \{1, \ldots, k\}, \Sigma, (q_0, 1), \Delta', F_k \times \{k\})$$

with the following transitions in Δ' (assuming that $(p, a, q) \in \Delta$):

- $((p, i), a, (q, i))$, if $i \leq k$ and $p \notin F_i$
- $((p, i), a, (q, i + 1))$, if $i < k$ and $p \in F_i$
- $((p, k), a, (q, 1))$, if $p \in F_k$

□

5 Nonemptiness Problem for Büchi Automata

Before finally turning our attention to the complementation of Büchi automata, we consider the nonemptiness problem for Büchi automata. As an observation from the proof of the decidability of the nonemptiness problem, we obtain a result corresponding to the following property of automata on finite words: If $L(\mathfrak{A}) \neq \emptyset$ for a finite automaton $\mathfrak{A}$ then $L(\mathfrak{A})$ contains a word w with $|w| \leq |Q|$.

Definition 16. *An ω-word over Σ is ultimately periodic if it has the form $\alpha = uvvv\ldots$ for some $u \in \Sigma^*, v \in \Sigma^+$.*

Remark. $\alpha \in \{0, \ldots, 9\}^\omega$ is ultimately periodic iff the real number with decimal expansion $0.\alpha$ is rational.

Theorem 17. *a) The nonemptiness problem "$L(\mathfrak{A}) \neq \emptyset$" for a Büchi automaton $\mathfrak{A}$ is decidable in polynomial time.*

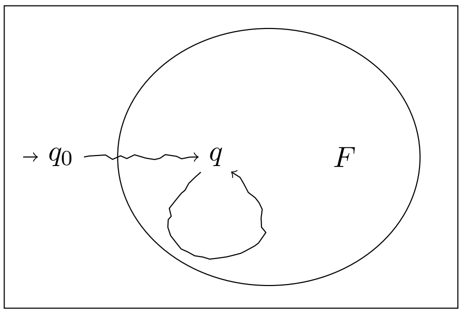
b) If $L(\mathfrak{A}) \neq \emptyset$, then it contains an ultimately periodic ω-word.

The proof is an easy consequence of the following remark.

Remark. Given $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$. Then
$L(\mathfrak{A}) \neq \emptyset \iff$ in the transition graph of $\mathfrak{A}$ there exists a path from q_0 to some $q \in F$, such that from q there is a nonempty path back to q.

We can test whether $\mathfrak{A}$ has this property by the following algorithm.

```
**for all** $q \in F$ **do**
  use DFS from $q_0$ to check whether $q$ is reachable from $q_0$
  use DFS from $q$ to check whether $q$ is reachable from $q$ via a nonempty path
**end for**
```



The time complexity is $O(|F| \cdot (|\Delta| + |Q|))$.

Two other fundamental problems are:

- Equivalence problem: Given Büchi automata $\mathfrak{A}, \mathfrak{B}$, does $L(\mathfrak{A}) = L(\mathfrak{B})$ hold?
- Inclusion problem: Given Büchi automata $\mathfrak{A}, \mathfrak{B}$, does $L(\mathfrak{A}) \subseteq L(\mathfrak{B})$ hold?

Obviously $L(\mathfrak{A}) \subseteq L(\mathfrak{B}) \Leftrightarrow L(\mathfrak{A}) \cap \overline{L(\mathfrak{B})} = \emptyset$. If we can show closure of the class of regular ω-languages under complement we get a solution by constructing a Büchi automaton $\mathfrak{C}$ with $L(\mathfrak{C}) = L(\mathfrak{A}) \cap \overline{L(\mathfrak{B})}$ and testing $\mathfrak{C}$ for emptiness.

6 Complementation of Büchi Automata

We now show that the complement of a regular language is again regular.

Theorem 18. *For each Büchi automaton $\mathfrak{A}$ over Σ one can construct a Büchi automaton $\mathfrak{B}$ such that $L(\mathfrak{B}) = \overline{L(\mathfrak{A})} = \Sigma^\omega \backslash L(\mathfrak{A})$.*

Idea. Recall that Büchi automata are equivalent to regular ω-languages, i.e. languages of the form $\bigcup_{i=1}^{n} U_i \cdot V_i^\omega$ for regular languages U_i, V_i. We will find a representation of $\overline{L(\mathfrak{A})}$ as $\bigcup_{i=1}^{n} U_i \cdot V_i^\omega$ with appropriate regular languages U_i, V_i. To construct the "building blocks" U_i, V_i, we introduce a congruence relation $\sim_{\mathfrak{A}}$ on finite words.

Definition 19. *Let $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$. For $p, q \in Q$ let*

- *$p \xrightarrow{w} q \Leftrightarrow$ there is a run from p to q via w in $\mathfrak{A}$.*
- *$p \overset{w}{\Rightarrow} q \Leftrightarrow$ there is a run from p to q via w in $\mathfrak{A}$ with a visit of a state in F.*

Define

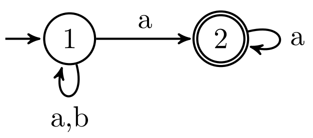
$$u \sim_{\mathfrak{A}} v \Leftrightarrow \quad \text{for all } p, q \in Q: \begin{array}{l} p \xrightarrow{u} q \Leftrightarrow p \xrightarrow{v} q \text{ and} \\ p \overset{u}{\Rightarrow} q \Leftrightarrow p \overset{v}{\Rightarrow} q. \end{array}$$

We observe that

- $\sim_{\mathfrak{A}}$ is an equivalence relation with only finitely many equivalence classes. ($\sim_{\mathfrak{A}}$ clearly is an equivalence relation, its index is finite since there are only finitely many states).
- $\sim_{\mathfrak{A}}$ is a congruence relation with respect to concatenation, i.e. $u \sim_{\mathfrak{A}} v, u' \sim_{\mathfrak{A}} v'$ implies $uu' \sim_{\mathfrak{A}} vv'$.
- Each equivalence class is regular.

(The proof of the last two items is given later.)
Given $\mathfrak{A}$, a method to determine the $\sim_{\mathfrak{A}}$-classes uses so called transition profiles. For each finite word u, list those (p, q) with $p \xrightarrow{u} q$ and $p \overset{u}{\Rightarrow} q$.
The equivalence class of a word u is described by its transition profile $\tau(u)$, i.e. the list of pairs (p, q) with $p \xrightarrow{u} q$ and the list of pairs (p', q') with $p' \overset{u}{\Rightarrow} q'$.

Example 20.



Transition profiles:
ε : $1 \rightarrow 1$, $2 \Rightarrow 2$
a: $1 \rightarrow 1$, $1 \Rightarrow 2$, $2 \Rightarrow 2$
b: $1 \rightarrow 1$
aa: same as a

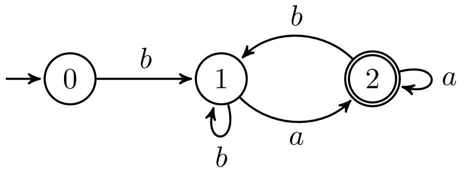
ab: same as b
ba: $1 \to 1$, $1 \Rightarrow 2$
bb: same as b

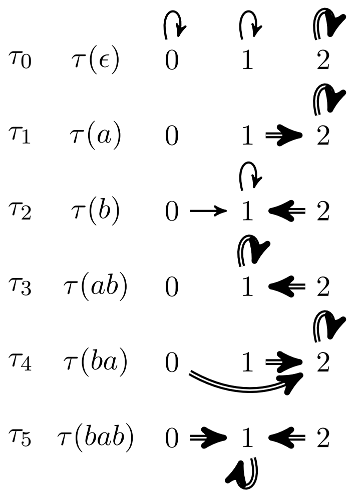
There are no other transition profiles of words over $\{a, b\}$.

⊠

We use a compressed notation, using arrow $\to$ and $\Rightarrow$ between states of Q. ($p \overset{u}{\Rightarrow} q$ implies $p \overset{u}{\to} q$ of course).

Example 21.



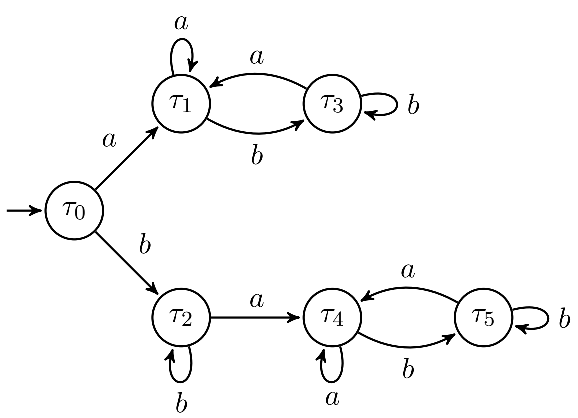

τ_0 $\tau(\epsilon)$ 0 1 2
τ_1 $\tau(a)$ 0 1 2
τ_2 $\tau(b)$ 0 1 2
τ_3 $\tau(ab)$ 0 1 2
τ_4 $\tau(ba)$ 0 1 2
τ_5 $\tau(bab)$ 0 1 2

Note that $\tau(aa) = \tau(a)$ and $\tau(bb) = \tau(b)$. ⊠

Remark. The profile $\tau(uv)$ can be computed from $\tau(u)$ and $\tau(v)$. We have $p \overset{uv}{\to} q$ if there exists r with $p \overset{u}{\to} r$ and $r \overset{v}{\to} q$. Similarly for $\Rightarrow$.
In other words: $u \sim_{\mathfrak{A}} u'$ and $v \sim_{\mathfrak{A}} v'$ implies $uv \sim_{\mathfrak{A}} u'v'$, *i.e.* $\sim_{\mathfrak{A}}$ *is a congruence with respect to concatenation.*
As consequence we can compute $\tau(ua)$ from $\tau(u)$ and $\tau(a)$. This motivates the following

Definition 22. *The deterministic automaton* TP($\mathfrak{A}$) *consists of all transition profiles of* $\mathfrak{A}$ *with an a-labelled edge from τ to τ' if $\tau = \tau(u)$ and $\tau' = \tau(ua)$ for some $u \in \Sigma^*$. The initial state is $\tau(\epsilon)$.*

Example 23. For the automaton $\mathfrak{A}$ of example 21, TP($\mathfrak{A}$) looks as follows.



⊠

Remark. The words of transition profile τ are accepted by $\mathrm{TP}(\mathfrak{A})$ with single final state τ. As an immediate consequence we obtain that each $\sim_{\mathfrak{A}}$-class is regular.

Example 24. For the automaton $\mathrm{TP}(\mathfrak{A})$ of the example above we have $\tau_1 = aa^*(bb^*aa^*)^*$. ⊠

Lemma 25. *Let U, V be $\sim_{\mathfrak{A}}$-classes and $U \cdot V^\omega \cap L(\mathfrak{A}) \neq \emptyset$. Then $U \cdot V^\omega \subseteq L(\mathfrak{A})$.*

This motivates to use the $\sim_{\mathfrak{A}}$-classes U, V with $U \cdot V^\omega \cap L(\mathfrak{A}) = \emptyset$ for a representation of $\overline{L(\mathfrak{A})}$.

Proof Suppose $\alpha \in U \cdot V^\omega \cap L(\mathfrak{A})$. $\mathfrak{A}$ has a run on α visiting a final state infinitely often. α has the form $\alpha = uv_0v_1v_2\dots$ with $u \in U$ and $v_i \in V$. The accepting run has the form $q_0 \xrightarrow{u} p_0 \xrightarrow{v_0} p_1 \xrightarrow{v_1} p_2 \dots$ where a final state is visited for infinitely many segment runs $p_i \xrightarrow{v_i} p_{i+1}$. We show that any $\beta \in U \cdot V^\omega$ is accepted by $\mathfrak{A}$. We know that β is of the form $u'v_0'v_1'v_2'\dots$ with $u' \in U, v_i' \in V$. Since $u \sim_{\mathfrak{A}} u', v_i \sim_{\mathfrak{A}} v_i'$ we obtain a run $q_0 \xrightarrow{u'} p_0 \xrightarrow{v_0'} p_1 \xrightarrow{v_1'} p_2 \dots$ passing a final state in v_i' for those indices i for which this was the case with v_i. Hence $\mathfrak{A}$ accepts β. □

Define $L' := \bigcup\{U \cdot V^\omega \mid U, V \text{ are } \sim_{\mathfrak{A}}\text{-classes and } U \cdot V^\omega \cap L(\mathfrak{A}) = \emptyset\}$. Note that L' is a regular ω-language disjoint from $L(\mathfrak{A})$. To prove that, in fact, $L' = \overline{L(\mathfrak{A})}$, we show in the next lemma that each infinite word is decomposable in an appropriate way. The proof is based on a combinatorial result due to Ramsey.

Lemma 26. *Each $\alpha \in \Sigma^\omega$ is decomposable as $\alpha = v_0v_1v_2\dots$ with $v_i \in V$ for some fixed $\sim_{\mathfrak{A}}$-class V for all $i > 0$.*

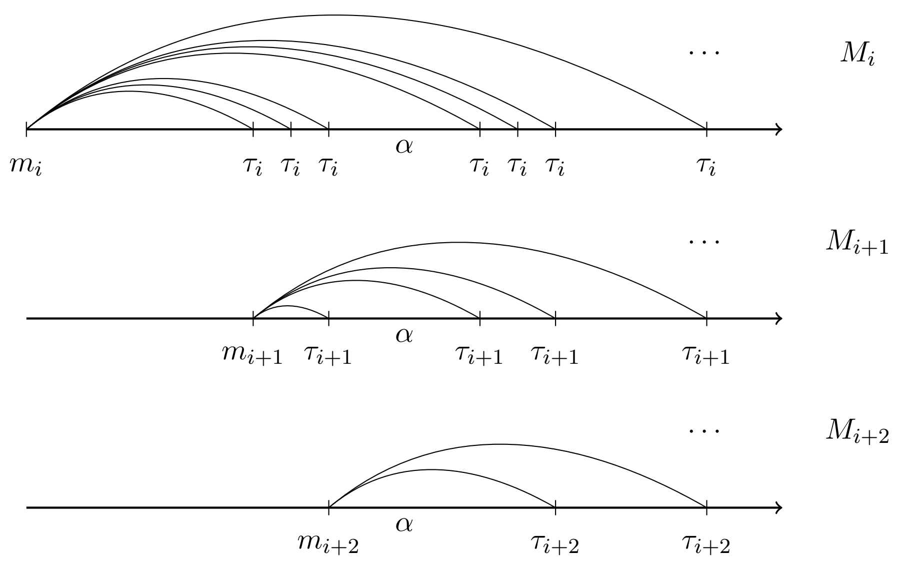
Then we let U be the $\sim_{\mathfrak{A}}$-class of v_0, so we get $\alpha \in U \cdot V^\omega$. If $\alpha \in \overline{L(\mathfrak{A})}$ then, by Lemma 25, $U \cdot V^\omega \cap L(\mathfrak{A}) = \emptyset$ and thus $\alpha \in L'$. Hence, indeed $L' = \overline{L(\mathfrak{A})}$.

Proof (of Lemma 26). Let $\alpha \in \Sigma^\omega$. Define sets $M_i \subseteq \mathbb{N}$ for $i = 0, 1, 2, \dots$.

- $M_0 = \mathbb{N}$.
- Given M_i define M_{i+1} as follows. Let $m_i := \min M_i$ and consider the transition profiles of $\alpha[m_i, n)$ for $n \in M_i, n > m_i$. Pick a profile τ_i that occurs infinitely often.

$$M_{i+1} := \{n \in M_i : n > m_i, \text{ and the profile of } \alpha[m_i, n) \text{ is } \tau_i\}.$$

Observe that M_{i+1} is an infinite subset of M_i.


M_i
$\dots$
m_i
τ_i τ_i τ_i
α
τ_i τ_i τ_i
τ_i
M_{i+1}
m_{i+1} τ_{i+1} τ_{i+1} τ_{i+1} τ_{i+1}
M_{i+2}
m_{i+2} τ_{i+2} τ_{i+2}

We obtain a sequence $0 = m_0 < m_1 < m_2 < \dots$. Observe that the profile of $\alpha[m_i, m_j)$ equals τ_i for $i < j$.
Among the profiles τ_i, some profile τ occurs infinitely often. Let $n_0 < n_1 < n_2 < \dots$ be a subsequence of $m_0 < m_1 < m_2 < \dots$ such that the profile of $\alpha[n_i, m_j)$ equals τ for $n_i < m_j$. This implies also that the profile of $\alpha[n_i, n_j)$ equals τ for $n_i < n_j$.
Now use the decomposition of α into the segments $\alpha = \alpha[0, n_0)\alpha[n_0, n_1)\alpha[n_1, n_2)\dots$. Let U be the $\sim_{\mathfrak{A}}$-class of $\alpha[0, n_0)$ and V the $\sim_{\mathfrak{A}}$-class of $\alpha[n_0, n_1)$. Then $\alpha \in U \cdot V^\omega$.

□

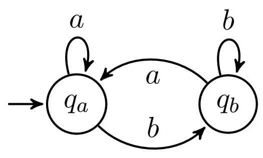
Remark on effectiveness. For an algorithmic construction of a Büchi automaton for $\overline{L(\mathfrak{A})}$ from $\mathfrak{A}$, one needs a test which checks for each pair U, V of $\sim_{\mathfrak{A}}$-classes, whether $U \cdot V^\omega \cap L(\mathfrak{A}) = \emptyset$. This test is clear from the intersection construction and the nonemptiness test.

Remark on the inclusion problem and equivalence problem. The effective complementation of Büchi automata helps in deciding $L(\mathfrak{A}) \subseteq L(\mathfrak{B})$ for any Büchi automata $\mathfrak{A}, \mathfrak{B}$ over Σ: We have $L(\mathfrak{A}) \subseteq L(\mathfrak{B})$ iff $L(\mathfrak{A}) \cap \overline{L(\mathfrak{B})} = \emptyset$, where from $\mathfrak{A}, \mathfrak{B}$ we can construct a Büchi automaton $\mathfrak{C}$ with $L(\mathfrak{C}) = L(\mathfrak{A}) \cap \overline{L(\mathfrak{B})}$. Thus the inclusion test reduces to the emptiness test for Büchi automata.

Part II
Determinization of Büchi Automaton

7 Deterministic ω-Automata

Recall: Deterministic Büchi-Automata cannot recognize $(a+b)^*b^\omega$.
Deterministic canditate automaton:



Need condition: "from some times onwards only q_b" this means: "q_a visited only finitely often"
Idea: Combine conditions of state visits: "infinitely often" and "finitely often"
For run $\rho \in Q^\omega$ define

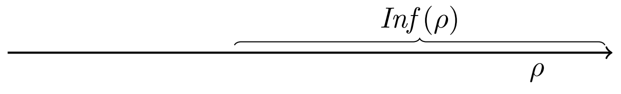
$$Inf = \{q \in Q \mid \exists^\omega i \rho(i) = q\}$$

$$P = Inf(\rho) \Leftrightarrow \bigwedge_{q \in P} \exists^\omega i\ \rho(i) = q \wedge \bigwedge_{q \in Q \setminus P} \neg\exists^\omega i\ \rho(i) = q$$

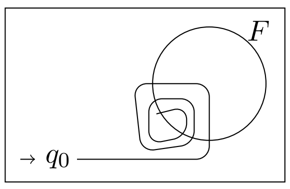
Each run ρ can be decomposed into

- a prefix with all occurences of states that are visited only finitely often
- a suffix in which only (and precisely) the states of $Inf(\rho)$ occur

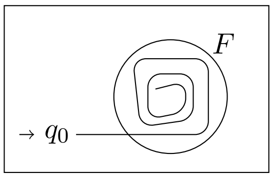
Illustration:



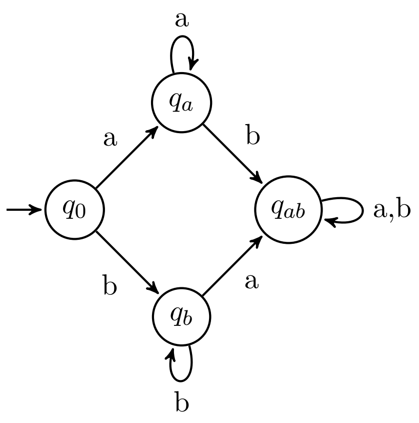
It is then clear that the subset $P = Inf(\rho)$ of Q gives a strongly connected subgraph of the transition graph of the automaton: For any $p, q \in P$ there is a nonempty path from p to q.
First attempt for definition of acceptance: Take automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$, and consider a run ρ accepting if $Inf(\rho) = F$.
The usual Büchi acceptance condition says $Inf(\rho) \cap F \neq \emptyset$:



We get a new acceptance condition by requiring $Inf(\rho) = F$:



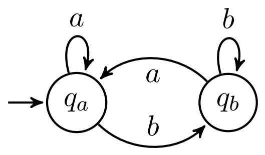
Example 27. $a^\omega + b^\omega$. We use the transition graph



The adequate condition is here: $Inf(\rho) = \{q_a\} \vee Inf(\rho) = \{q_b\}$. So we have to use two state sets. (In the exercises it is shown that any automaton recognizing $a^\omega + b^\omega$ needs more than one set for the acceptance component). ⊠

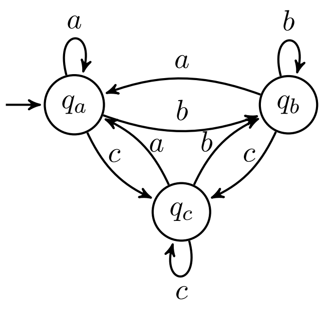
Definition 28. *A* Muller automaton *is of the form* $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ *where* $\mathcal{F} \subseteq \mathcal{P}(Q)$ *i.e.* $\mathcal{F} = \{F_1, \dots, F_k\}$ *with* $F_k \subseteq Q$ *using the following acceptance condition:*
$\mathfrak{A}$ *accepts* $\alpha \Leftrightarrow$ *for the unique run* ρ *of* $\mathfrak{A}$ *on* α *we have:* $Inf(\rho) \in \mathcal{F}$ *(i.e.* $Inf(\rho) = F_i$ *for some* $i \in \{1, \dots, k\}$*)*

Example 29. $(a+b)^* b^\omega$ is recognized by the Muller automaton



with $\mathcal{F} = \big\{\{q_b\}\big\}$ ⊠

Example 30. $\Sigma = \{a, b, c\}$
$L = \{\alpha \mid$ if a occurs infenitely often, so does $b\}$
A corresponding Muller automaton has states q_a, q_b, q_c which are reached by letter a, b, c respectively:



The acceptance requirement is $q_a \in \mathit{Inf}(\rho) \to q_b \in \mathit{Inf}(\rho)$.
So for $\mathcal{F}$ we collect all sets P with $q_a \in P \to q_b \in P : \mathcal{F} = \big\{\{q_b\}, \{q_c\}, \{q_a, q_b\}, \{q_b, q_c\}, \{q_a, q_b, q_c\}\big\}$.
$\boxtimes$

Now we connect deterministic Büchi automata and Muller automata.

Theorem 31. *An ω-language over Σ is Muller recognizable iff it is a Boolean combination (over Σ^ω) of deterministic Büchi recognizable ω-languages.*

Proof

$\Rightarrow$ Given $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$,
we have $\alpha \in L(\mathfrak{A}) \Leftrightarrow$ for the unique run ρ of $\mathfrak{A}$ on α:

$$\bigvee_{F \in \mathcal{F}} F = \mathit{Inf}(\rho)$$

We have

$$F = \mathit{Inf}(\rho) \Leftrightarrow \bigwedge_{q \in F} \underbrace{\exists^\omega i\ \rho(i) = q}_{(1)} \wedge \bigwedge_{q \in Q \setminus F} \neg \exists^\omega i\ \rho(i) = q.$$

Now observe that (1) says that the deterministic Büchi-automaton $\mathfrak{A}_q = (Q, \Sigma, q_0, \delta, \{q\})$ reaches his final state q.

So $L(\mathfrak{A}) = \bigcup_{F \in \mathcal{F}} \Big(\bigcap_{q \in F} L(\mathfrak{A}_q) \cap \bigcap_{q \in Q \setminus F} \overline{L(\mathfrak{A}_q)} \Big)$.
So $L(\mathfrak{A})$ is a Boolean combination of deterministic Büchi recognizable ω-languages.

$\Leftarrow$

1. Each deterministic Büchi recognizable ω-language is Muller recognizable. Given a Büchi-automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$ take the Muller-automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ with $\mathcal{F} = \{P \mid P \cap F \neq \emptyset\}$.
2. Assume L is recognizable by $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$. Then $\overline{L}$ is recognizable by $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{P}(Q) \setminus \mathcal{F})$
3. Given $\mathfrak{A}_1 = (Q_1, \Sigma, q_0^1, \delta_1, \mathcal{F}_1)$, $\mathfrak{A}_2 = (Q_2, \Sigma, q_0^2, \delta_2, \mathcal{F}_2)$ find a Muller-automaton for $L(\mathfrak{A}_1) \cap L(\mathfrak{A}_1)$. Take the product automaton $\mathfrak{B} = \big(Q_1 \times Q_2, \Sigma, (q_0^1, q_0^2), \underbrace{\delta}_{\text{as usual}}, \mathcal{F}\big)$ where $\mathcal{F}$ is defined as follows:

$$\underbrace{\big\{(p_1, q_1), \ldots, (p_k, q_k)\big\}}_{\subseteq Q_1 \times Q_2} \in \mathcal{F} :\Leftrightarrow \{p_1, \ldots, p_k\} \in \mathcal{F}_1 \text{ and } \{q_1, \ldots, q_k\} \in \mathcal{F}_2$$

$\square$

Rabin-, Streett- and Parity Automata

Definition 32. *A* Rabin automaton *is of the form* $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$ *where*

$$\Omega = \big((E_1, F_1), \ldots, (E_k, F_k)\big) \textit{ with } E_i, F_i \subseteq Q$$

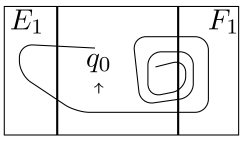
$\mathfrak{A}$ *accepts* $\alpha \Leftrightarrow$ *for the unique run* ρ *of* $\mathfrak{A}$ *on* α *we have:*

$$\bigvee_{i=1}^{k} \mathit{Inf}(\rho) \cap E_i = \emptyset \wedge \mathit{Inf}(\rho) \cap F_i \neq \emptyset$$

in other words:

$$\bigvee_{i=1}^{k} \neg \exists^\omega j\ \rho(j) \in E_i \wedge \exists^\omega j\ \rho(j) \in F_j$$

Illustration:



Remark. A deterministic Büchi automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$ is simulated by the Rabin automaton $\mathfrak{A}' = (Q, \Sigma, q_0, \delta, \Omega)$ with $\Omega = \big((\emptyset, F)\big)$.

Remark. A Rabin automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$ with $\Omega = \big((E_1, F_1), \ldots, (E_k, F_k)\big)$ is simulated by the Muller automaton $\mathfrak{A}' = (Q, \Sigma, q_0, \delta, \mathcal{F})$ with

$$P \in \mathcal{F} :\Leftrightarrow \exists i \in \{1, \ldots, k\} : \ P \cap E_i = \emptyset, P \cap F_i \neq \emptyset.$$

Definition 33. *A* Streett automaton *is of the format of Rabin automaton* $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$ *with the negation of the Rabin condition for acceptance:*

$$\begin{aligned} &\neg \bigvee_{i=1}^{k} \big(\mathit{Inf}(\rho) \cap E_i = \emptyset \wedge \mathit{Inf}(\rho) \cap F_i \neq \emptyset\big) \\ \equiv &\bigwedge_{i=1}^{k} \big(\mathit{Inf}(\rho) \cap E_i \neq \emptyset \vee \mathit{Inf}(\rho) \cap F_i = \emptyset\big) \\ \equiv &\bigwedge_{i=1}^{k} \big(\mathit{Inf}(\rho) \cap F_i \neq \emptyset \rightarrow \mathit{Inf}(\rho) \cap E_i \neq \emptyset\big) \end{aligned}$$

This condition is a "fairness property": If visits to F_i happen again and again, so do visits in E_i.

Definition 34. *A* parity automaton *is of the form* $\mathfrak{A} = (Q, \Sigma, q_0, \delta, c)$ *with a "coloring"*

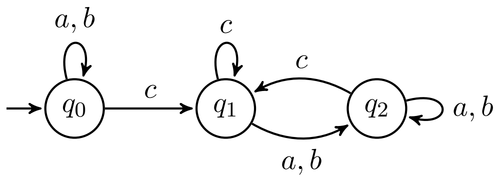
$$c : Q \rightarrow \{0, \ldots, k\}$$

$\mathfrak{A}$ *accepts* $\alpha \Leftrightarrow$ *for the unique run* ρ *of* $\mathfrak{A}$ *on* α*: the maximal color visited infinitely often in* ρ *is even.*
Formally this means: $\max\big(\mathit{Inf}(c(\rho))\big)$ *is even.*

Let us repeat the different variants of deterministic ω-automata introduced in the last lecture.

Name	Acceptance component (over Q)	Acceptance condition on ρ
Muller	$\mathcal{F} = \{F_1, \dots, F_k\}$, $F_i \subseteq Q$	$\mathit{Inf}(\rho) \in \mathcal{F}$
Rabin	$\Omega = \big((E_1, F_1), \dots, (E_k, F_k)\big)$, $E_i, F_i \subseteq Q$	for some i $\mathit{Inf}(\rho) \cap E_i = \emptyset$ and $\mathit{Inf}(\rho) \cap F_i \neq \emptyset$
Streett	$\Omega = \big((E_1, F_1), \dots, (E_k, F_k)\big)$, $E_i, F_i \subseteq Q$	Negation of Rabin, i.e. for all i $\mathit{Inf}(\rho) \cap F_i \neq \emptyset$ implies $\mathit{Inf}(\rho) \cap E_i \neq \emptyset$
Parity	$c : Q \to \{0, \dots, k\}$	$\mathit{max}(\mathit{Inf}(c(\rho)))$ even

Example 35. $\Sigma = \{a, b, c\}$, $L = \{\alpha \in \Sigma^\omega : c$ occurs in α, but only finitely often$\}$.



- Muller: $\mathcal{F} = \{\{q_2\}\}$.
- Rabin: $\Omega = ((E_1, F_1))$ with $E_1 = \{q_0, q_1\}, F_1 = \{q_2\}$.
- Parity: $c : Q \to \{0, 1\}$, $c(q_0) = c(q_1) = 1$, $c(q_2) = 0$.
- Streett: We use the following observation:
  "c occurs" means "infinitely often in $\{q_1, q_2\}$", $(\{q_1, q_2\}, \{q_0, q_1, q_2\})$, formally
  $$\mathit{Inf}(\rho) \cap \underbrace{\{q_0, q_1, q_2\}}_{F_1} \neq \emptyset \to \mathit{Inf}(\rho) \cap \underbrace{\{q_0, q_2\}}_{E_1} \neq \emptyset.$$
  $(\emptyset, \{q_1\})$ "c occurs only finitely often" means: Assuming $\mathit{Inf}(\rho) \cap \underbrace{\{q_1\}}_{F_2} \neq \emptyset$, we have a contradiction $(\mathit{Inf}(\rho) \cap \underbrace{\emptyset}_{E_2} \neq \emptyset)$.
  The desired Streett automaton for L has acceptance component $\Omega = ((E_1, F_1), (E_2, F_2))$.

⊠

Remark. Let $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$ be a Rabin automaton recognizing $L \subseteq \Sigma^\omega$. Then $\mathfrak{A}$ used as Streett automaton recognizes $\Sigma^\omega \setminus L$.

Remark. Obviously the Büchi acceptance condition is just a special parity condition. Assign to each state in F color 0 and each state in $Q\backslash F$ color 1, then the automaton as parity automaton accepts if and only if it accepts as a Büchi automaton.
As we will see in the next theorem it was no coincidence that we could provide automata with all the different acceptance conditions for L.

Theorem 36 (Main Theorem on deterministic ω-automata)**.** *For an ω-language L, the following are equivalent:*

1. *L is recognized by a nondeterministic Büchi automaton.*
2. *L is recognized by a deterministic Muller automaton.*
3. *L is recognized by a deterministic Rabin automaton.*
4. *L is recognized by a deterministic Streett automaton.*
5. *L is recognized by a deterministic parity automaton.*

We already know that 3) implies 2). We will state the missing connections as individual theorems and prove them over the next lectures.
We shall prove the following implications:

2) $\Rightarrow$ 1),

1) $\Rightarrow$ 3),

3) $\Leftrightarrow$ 4),

2) $\Rightarrow$ 5),

5) $\Rightarrow$ 3).

This is the fundamental theorem of McNaughton; it will be shown in the next section. Using 1) $\Rightarrow$ 3) we show in this section:

Theorem 37. *Each deterministic Muller automaton can be simulated by a nondeterministic Büchi automaton.*

Proof Consider $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$. We need to express $Inf(\rho) = F_1$ or $Inf(\rho) = F_2$ or ... or $Inf(\rho) = F_k$.
Idea: Given F_i and a point ("critical position") in a run from which onwards precisely the F_i states occur again and again, one can signal $Inf(\rho) = F_i$ by accumulating F_i states and reset to $\emptyset$ if F_i is reached. The nondeterministic Büchi automaton guesses both items (index i and the critical position in the run).
Implementation: Given Muller automaton $\mathfrak{A}$, use state set $Q' = Q \cup (Q \times \mathcal{P}(Q) \times \{1, \ldots, k\})$.
The Büchi automaton is $(Q', \Sigma, q_0, \Delta, F)$, where Δ contains the following transitions

- (p, a, q) and $(p, a, (q, \emptyset, j))$ for all $j \in \{1, \ldots, k\}$ if $\delta(p, a) = q$,
- $((p, P, j), a, (q, P \cup \{q\}, j))$ if $\delta(p, a) = q$ and $P \cup \{q\} \subsetneq F_j$ and
- $((p, P, j), a, (q, \emptyset, j))$ if $\delta(p, a) = q$ and $P \cup \{q\} = F_j$.

$F = \{(p, \emptyset, j) : p \in Q, j \in \{1, \dots, k\}\}$. □

Theorem 38. *A deterministic Rabin automaton can be simulated by a deterministic Streett automaton.*

Proof Given a Rabin automaton $\mathfrak{A}$, use it as a Streett automaton $\mathfrak{B}$, thus $\overline{L(\mathfrak{A})} = L(\mathfrak{B})$. For $\mathfrak{B}$ construct an equivalent Muller automaton $\mathfrak{M}$ by collecting in $\mathcal{F}$ the loops that satisfy the Streett condition. Then let $\mathfrak{R}'$ be a Rabin automaton equivalent to $\mathfrak{M}$, thus $L(\mathfrak{R}') = \overline{L(\mathfrak{A})}$. Using $\mathfrak{R}'$ as a Streett automaton yields a Streett automaton for $L(\mathfrak{A})$. □

Theorem 39. *Each parity automaton can be simulated by a Rabin automaton.*

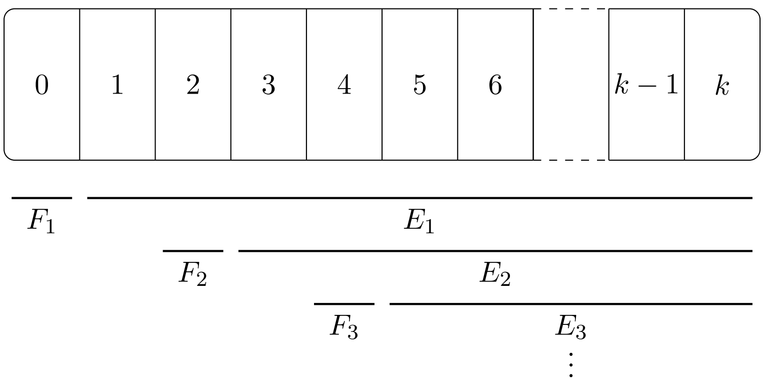
Proof Consider $\mathfrak{A} = (Q, \Sigma, q_0, \delta, c)$, $c : Q \to \{0, \dots, k\}$, w.l.o.g. k even. We need to express that the maximal color visited infinitely often is 0 or 2 or 4 or ... or k. To do this use the following Rabin pairs:

$$E_1 = \{q \in Q \mid c(q) \in \{1, \dots, k\}\},\ F_1 = \{q \in Q \mid c(q) = 0\}$$

$$E_2 = \{q \in Q \mid c(q) \in \{3, \dots, k\}\},\ F_2 = \{q \in Q \mid c(q) = 2\}$$

$$\vdots$$

$$E_{\frac{k}{2}+1} = \emptyset,\ F_{\frac{k}{2}+1} = \{q \in Q \mid c(q) = k\}$$



□

Theorem 40. *Each deterministic Muller automaton is simulated by a deterministic parity automaton.*

Proof Idea 1. Use a special data structure to remember visited states in the order of their last visits: "last visit record", short "LVR".

Example 41. For the Muller automaton $\mathfrak{A} = (\{1, 2, 3, 4\}, \Sigma, 1, \delta, \mathcal{F})$ with $\mathcal{F} = \{\{2, 4\}\}$ and for a run $\rho = 1, 3, 4, 2, 2, 4, 2, \dots$ we obtain the following LVR.

ρ	LVR	coloring
1	$\underline{1}234$	1
3	$31\underline{2}4$	5
4	$431\underline{2}$	7
2	$243\underline{1}$	7
2	$\underline{2}431$	1
4	$4\underline{2}31$	4
2	$2\underline{4}31$	4

In the first line we underline the first position, later we underline that position from which the respective currect state was taken. Assume, as indicated, $\mathit{Inf}(\rho) = \{2,4\}$. States $3,1$ will stay eventually untouched on the last two places of the LVR's. The maximum position occurring infinitely often underlined yields the cardinality of $\mathit{Inf}(\rho)$. The entries up to this position give $\mathit{Inf}(\rho)$. We call the underlined position "hit" and the state set up to the hit "hit set".
The state vector with hit is called LVR (for latest visit record) or LAR (for latest appearance record). ⊠

Idea 2. Introduce colors by taking double of hit if the hit set is accepting ($\in \mathcal{F}$) and the double of hit -1 if the hit set is non-accepting ($\notin \mathcal{F}$).

Given a Muller automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ with $Q = \{1, \dots, n\}$, construct the parity automaton as follows:

- State set: set of latest visit records over Q (LVR(Q)).
- Initial state: $q_0' = (\underline{1}, \dots, n)$.
- Transition function δ' given by
$$\delta'((i_1, \dots, \underline{i_l}, \dots, i_n), a) = (j_1, \dots, \underline{j_k}, \dots, j_n),$$
where $(j_1, \dots, j_n)$ is obtained from $(i_1, \dots, i_n)$ by shifting $\delta(i_1, a)$ to the front ($= j_1$), defining k by $i_k = j_1$.
- Coloring: $c : \text{LVR}(Q) \to \{1, \dots, 2n\}$
$$c(i_1, \dots, \underline{i_k}, \dots, i_n) = \begin{cases} 2k & \{i_1, \dots i_k\} \in \mathcal{F} \\ 2k-1 & \{i_1, \dots, i_k\} \notin \mathcal{F}. \end{cases}$$

Each run ρ of the Muller automaton induces a run ρ' of the constructed parity automaton. The LVR allows to describe $\mathit{Inf}(\rho)$ by the colors occurring in $\mathit{Inf}(\rho')$:

Lemma 42 (LVR Lemma)**.** *$\mathit{Inf}(\rho) = F$ with $|F| = m \Leftrightarrow$*

1. *In ρ' the hit is $> m$ only finitely often.*

2. *Afterwards the hit is $= m$ infinitely often and then the hit set $\{i_1, \ldots, i_m\}$ equals F.*

Proof

- $\Rightarrow$ Let $\mathit{Inf}(\rho) = F$ with $|F| = m$. In ρ the states of $Q \setminus F$ will stay at the end, thus hit $\leq m$. If hit $< m$ from some point onwards, i_m would be visited only finitely often and thus $|\mathit{Inf}(\rho)| \leq m - 1$, contradicting the assumption.

- $\Leftarrow$ Assume *1)* and *2)*. By *1)*, from some point onwards, the LVR entries $i_{m+1}, \ldots, i_n$ are not visited any more. Let $F = \{i_1, \ldots, i_m\}$. Then the states of $\mathit{Inf}(\rho)$ have to occur on positions $\leq m$, hence $\mathit{Inf}(\rho) \subseteq F$.

  Now we show $F \subseteq \mathit{Inf}(\rho)$. Assume $q \in F$ but $q \notin \mathit{Inf}(\rho)$. Then q is not visited from some point onwards, so q stays or moves to the right only. But by *1)* it stays finally at some position $\leq m$. This is not possible, because hit $= m$ again and again, so q would have to move again and again.

□

□

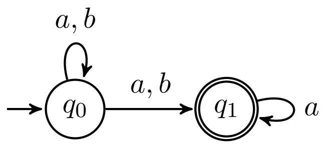
8 McNaughton's Theorem

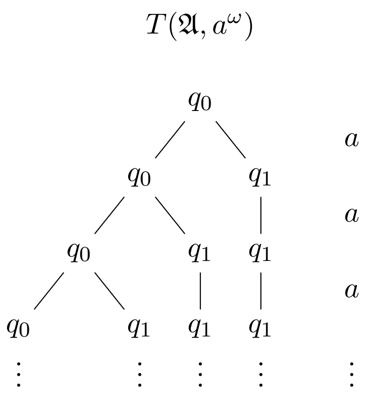
The topic of this section is to show the equivalence between nondeterministic Büchi automata and Muller automata. We have already shown the easy direction.

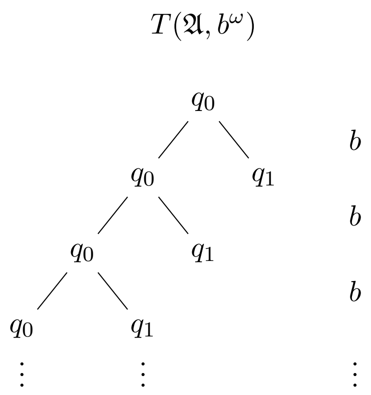
Theorem 43 (McNaughton's Theorem)**.** *Each nondeterministic Büchi automaton can be simulated by a deterministic Muller automaton.*

Proof (of McNaughton's Theorem) Idea. Use the run tree of the given Büchi automaton on a given ω-word. $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, F)$, $\alpha = \alpha(0)\alpha(1)\alpha(2)\ldots$.
We first try the simple approach of using the state sets of the run tree levels as new states and declare a set final if a state from F occurs in it. This corresponds to the classical powerset construction. Unfortunately this approach fails, as we show in the following example.

Example 44. Let $\mathfrak{A}$ be the following automaton.





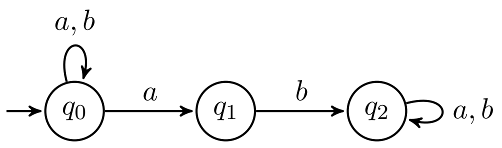


$\mathfrak{A}$ does not accept b^ω but the sequence of level state sets is $\{q_0\}, \{q_0, q_1\}, \{q_0, q_1\}, \ldots$ where each one, except the first is "final". ⊠

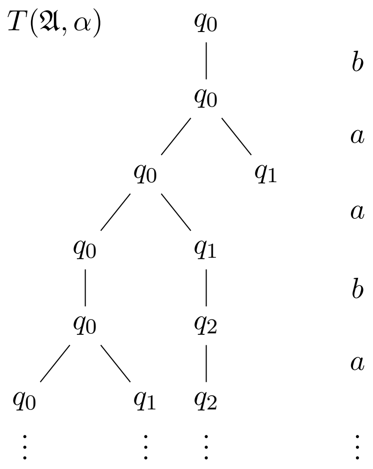
Definition 45. *Let $\mathfrak{A}$ be an automaton over Σ, α in Σ^ω. The run tree $T(\mathfrak{A}, \alpha)$ is defined inductively:*

- *the root is labelled q_0,*
- *each vertex on level i labelled p has, for each transition $(p, \alpha(i), q) \in \Delta$, a daughter labelled q.*

Example 46. Consider the following automaton



and $\alpha = baabab\ldots$.



⊠

Remark on $T(\mathfrak{A}, \alpha)$: $\mathfrak{A}$ accepts α if and only if $T(\mathfrak{A}, \alpha)$ has a path on which a final state occurs infinitely often.

This example shows that the powerset construction is too simple for the determinization of Büchi automata.
We refine the construction in several steps, starting from the run tree $T(\mathfrak{A}, \alpha)$:

1. Left-right tree $T_{LR}(\mathfrak{A}, \alpha)$.

2. Reduced left-right tree $R(\mathfrak{A}, \alpha)$.

3. Marked reduced left-right tree $M(\mathfrak{A}, \alpha)$.

We will then use the levels of $M(\mathfrak{A}, \alpha)$ as states for the Muller automaton. The proof follows Muller and Schupp (1991) in a variant given by Kähler and Wilke (2007).

Notation. Denote by $\Delta(P, a)$ the set $\{q : \exists p \in P \text{ with } (p, a, q) \in \Delta\}$.

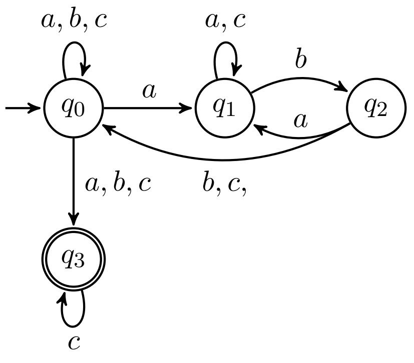
Definition 47. *The left-right tree $T_{LR}(\mathfrak{A}, \alpha)$ is defined inductively as follows:*

- *the root is $\{q_0\}$,*

- *if vertex v with label P is on level i, then v has two sons:*

  - *a left son labelled with $\Delta(P, \alpha(i)) \cap F$,*
  - *a right son labelled with $\Delta(P, \alpha(i)) \backslash F$.*

Example 48. Let $\mathfrak{A}$ be the following automaton



and let $\alpha = abcb\ldots$. Then $T_{LR}(\mathfrak{A}, \alpha)$ looks as follows

$$\begin{tikzcd}
 & & & & \{q_0\} \arrow[dlll] \arrow[drr] & & & & & & & \\
 & \{q_3\} \arrow[dl] \arrow[dr] & & & & & \{q_0, q_1\} \arrow[dl] \arrow[drr] & & & & & a \\
\emptyset & & \emptyset & & \{q_2, q_3\} \arrow[dl] \arrow[dr] & & & & \{q_0\} \arrow[dl] \arrow[dr] & & & b \\
\vdots & & \vdots & \{q_3\} & & \{q_0\} & & \{q_3\} & & \{q_0\} & & c \\
 & & & & & & \vdots & & & & \vdots &
\end{tikzcd}$$

$\boxtimes$

Remark. $\mathfrak{A}$ accepts α if and only if $T_{LR}(\mathfrak{A},\alpha)$ has an infinite path labelled with nonempty sets branching left infinitely often.

Proof

$\Rightarrow$ Each run ρ of $\mathfrak{A}$ on α determines a path π through $T_{LR}(\mathfrak{A},\alpha)$, ρ is accepting iff it branches left infinitely often. Obviously this means that the path π in $T_{LR}(\mathfrak{A},\alpha)$ branches left infinitely often.

$\Leftarrow$ Let π be a path in $T_{LR}(\mathfrak{A},\alpha)$ branching left infinitely often. Let the sequence of labels of π be $\{q_0\} = Q_0, Q_1, Q_2, \ldots$. Then $Q_0, Q_1, \ldots$ determines a subtree S of $T(\mathfrak{A},\alpha)$, where

- the root is labelled q_0 and
- given $p \in Q_i$ on level i, introduce a son labelled q for each $q \in Q_{i+1}$.

S is finitely branching and infinite. By König's Lemma, S has an infinite path with infinitely many occurrences of final states (because infinitely many of Q_i were left sons). So $\mathfrak{A}$ has an accepting run on α.

$\square$

Definition 49. *Define the reduced left-right tree $R(\mathfrak{A},\alpha)$ by starting with $T_{LR}(\mathfrak{A},\alpha)$, on each level keeping for each state only the leftmost occurrence and deleting the $\emptyset$-labelled vertices.*

Example 50. The reduced left-right tree for the automaton from the previous example looks as follows.

$$\begin{tikzcd}
 & & \{q_0\} \arrow[dll] \arrow[dr] & & & a \\
\{q_3\} & & & \{q_0, q_1\} \arrow[dl] \arrow[dr] & & b \\
 & & \{q_2, q_3\} \arrow[dl] \arrow[d] & & \{q_0\} & c \\
 & \{q_3\} & \{q_0\} & & & \\
 & & \vdots & & & \vdots
\end{tikzcd}$$

$\boxtimes$

Theorem 51. *Each (nondeterministic) Büchi automaton can be simulated by a deterministic Rabin automaton.*

Properties of $R(\mathfrak{A}, \alpha)$:

- On each level there are at most n vertices labeled by nonempty disjoint subsets of Q.
- The Update of the level information is captured by a finite function, so it is describable by the transition function of a finite automaton.

Lemma 52. $\mathfrak{A}$ *accepts* $\alpha \Leftrightarrow$ *in* $R(\mathfrak{A}, \alpha)$ *there exists a path branching left infinitely often.*

Proof

$\Leftarrow$ From an infinite path as given by the assumption we obtain a subtree of $T(\mathfrak{A}, \alpha)$. As for $T_{LR}(\mathfrak{A}, \alpha)$ we apply König's Lemma and get an infinite path of $T(\mathfrak{A}, \alpha)$ branching left infinitely often (i.e visiting infinitely often F).

$\Rightarrow$ In $R(\mathfrak{A}, \alpha)$ we use words over $\{L, R\}$ as names of vertices.
Assume in $T(\mathfrak{A}, \alpha)$ exists a path branching left infinitely often (i.e. successful run $\rho = \rho(0)\rho(1)\ldots$). Choose such a run ρ_1 visiting F as early as possible (after initial state), call this position i_1 and the state $q_1 = \rho_1(i_1)$.

$$\rho = \underbrace{\rho_1(0)\ldots}_{\notin F}\underbrace{\rho_1(i_1)}_{q_1 \in F}\rho_1'$$

Claim q_1 occurs in label of vertex $R^{i_1-1}L$ of $R(\mathfrak{A}, \alpha)$.
Otherwise q_1 occurs in a vertex $R^jL(L+R)^*$ with $0 < j < i_1 - 1$. Contradiction to the choice of i_1. Choose a run ρ_2 extending $\rho_1(0)\ldots\rho_1(i_1)$ and visiting a final state as early as possible after time i_1, call this position i_2 and the state $q_2 = \rho_2(i_2)$.

$$\rho_2 = \rho_1(0)\ldots\rho_1(i_1)\rho_2(i_1+1)\ldots\underbrace{\rho_2(i_2)}_{q_2 \in F}\rho_2'$$

Claim q_2 is in the label of $R^{i_1-1}LR^{i_2-1}L$. Proof similar as above.
Continuing in this way we obtain the desired path of $R(\mathfrak{A}, \alpha)$.

□

The levels of $R(\mathfrak{A}, \alpha)$ are only a good approximation of the states af the deterministic automaton we want to construct. We need more information in oder to be able to keep track of the paths of the tree. We introduce extra markings to levels of $R(\mathfrak{A}, \alpha)$ by tokens $t_1, \ldots, t_n$ with two extra pieces of information: birth number $b \in \{1, \ldots, n\}$ and color $c \in \{\mathit{green}, \mathit{yellow}, \mathit{red}\}$. Each vertex will have a label by a nonempty subset of Q and by a token with a birth number and a color. A sequence of such labels of a level of $R(\mathfrak{A}, \alpha)$ is called level data. The introduced extra markings will produce the tree $M(\mathfrak{A}, \alpha)$ from $R(\mathfrak{A}, \alpha)$. This construction of deterministic Rabin automata is due to Muller-Schupp and was improved by Kähler, Wilke:
Given Büchi automaton over state set Q; assume $|Q| = n$.

State space of Rabin automaton: Set of possible level data of marked reduced left-right run trees.

One state (level data) is given by a sequence of vertices, each of them marked with a non-empty subset of Q, with one of the n tokens (where each token has a color and a birth number).

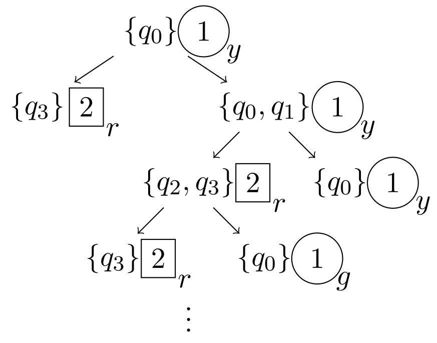
Initial state: Singleton of initial state of Büchi automaton, with token t_1 of birthnumber 1 and colored yellow

Update of level data given input letter:

1. Construct sequence of state sets as given by construction of reduced left-right run tree.
2. Move each token to right successor if that exists.
3. If right successor does not exist, move token to next-left cousin (of non-existing right successor). If there is no cousin on the left, remove the token and color it red (outside the tree).
4. On each vertex with multiple tokens, only keep the oldest one. Remove the younger ones and color them red (outside the tree).
5. Color those tokens on the tree that moved from a vertex to its right successor yellow, color the others green.
6. Supply red tokens to the places without token, keep the order of the birth numbers of the yellow and green tokens, and give the inserted red tokens higher birth numbers.

The resulting tree, given $\mathfrak{A}$ and α, is denoted $M(\mathfrak{A}, \alpha)$.

Example 53. (extending previous example)



⊠

Remark. Level data, initialisation and update procedure determine a finite state set Q', initial state q'_0 and a transition function δ of an automaton.

Lemma 54. *$\mathfrak{A}$ accepts $\alpha \Leftrightarrow$ in $M(\mathfrak{A}, \alpha)$ some token appears red only finitely often and green infinitely often.*

Remark. This induces the following acceptance component of a Rabin automaton:
$\Omega = \big((E_1, F_1), \ldots, (E_n, F_n)\big)$ with

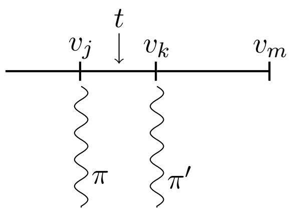
$$E_i = \text{ set of level data where } t_i \text{ occurs colored red}$$
$$F_i = \text{ set of level data where } t_i \text{ occurs colored green}$$

So the definition of the desired Rabin automaton is complete.

Proof of Lemma. It suffices to show: $R(\mathfrak{A}, \alpha)$ has a path branching left infinitely often $\Leftrightarrow$ in $M(\mathfrak{A}, \alpha)$ some token appears colored red only finitely often and colored green infinitely often.

$\Rightarrow$ Choose a level where all infinite paths are already separated. Call the vertices there $v_1, \ldots v_m$. Pick v_j such that path π passes through v_j, branching left infinitely often. Pick $k > j$ minimal such that an infinite path π' passes v_k (if no π' exists set $k := m+1$) Let t be the oldest token on $v_j, \ldots, v_{k-1}$.

Illustration:

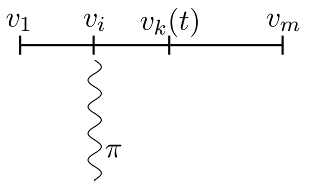


Claim 1 Token t never (later) gets color red:
Token t is never overwritten: Older tokens are absorbed by the path π' (or do not exist). Token t never drops out of the tree (by moving left) because it can move at most to π.

Claim 2 Token t gets color green infinitely often:
Assume t is not colored green from some level onwards. So t is colored yellow from some point onwards. By choice of k and π at some point t will sometimes be on the path π. But its color yellow contradicts to the choice of π.

$\Leftarrow$ Let t be a token that appears red only finitely often and green infinitely often. Choose a level as in "$\Rightarrow$": All infinite paths have separated and t is later never colored red. Assume t is on vertex v_k. Choose maximal $i \leq k$ such that an infinite path π passes v_i (such a path must exist, since otherwise, t would eventually be removed and colored red). The token t will always be on the rightmost descendant of $v_i, \ldots, v_k$. The token t will eventually arrive on the path π. If π branches right from some point onwards, t is yellow from that point onwards.
This leads to a contradiction, hence π must branch left again and again.

Illustration:



$\square$

9 Complexity of Determinization

We analyze the proof of Mc Naughton's Theorem to obtain an upper bound on the number of states of the resulting Rabin automaton. On the other hand we provide a lower bound which shows that the construction is optimal in an asymptotic sense.

Theorem 55 (Mc Naughton's Theorem)**.** *A nondeterministic Büchi automaton can be simulated by a deterministic Rabin automaton.*
Addendum: A Büchi automaton with n states can be simulated

a) by a Rabin automaton with $2^{O(n \cdot \log n)}$ states and

b) not by a Rabin automaton with $2^{O(n)}$ states.

Proof

a) We count the number of states that are needed in the Muller-Schupp-Kähler-Wilke construction. Given a Büchi automaton with state set Q, where $|Q| = n$, the states of the resulting Rabin automaton consist of tuples of disjoint state sets $\subseteq Q$ of length $\leq n$, together with tokens with birth number and color.

   Each tuple of (state sets, tokens, birth numbers, colors) can be described by a function:

   – $f_1 : Q \to \{0, \dots, n\}$

   $$f_1(q) = \begin{cases} 0 & \text{if } q \text{ does not occur in the tuple of state sets,} \\ i & q \text{ occurs at the } i\text{-th position of the tuple.} \end{cases}$$

   – $f_2 : \{1, \dots, n\} \to \{0, \dots, n\}$

   $$f_2(j) = \begin{cases} 0 & \text{if } t_j \text{ does not occur in the tuple,} \\ i & \text{if } t_j \text{ occurs at position } i \text{ in the tuple} \end{cases}$$

   – $f_3 : \{1, \dots, n\} \to \{0, \dots, n\}$

   $$f_3(j) = \begin{cases} 0 & \text{if } t_j \text{ does not occur in the tuple,} \\ i & \text{if the birth number of } t_j \text{ is } i \text{ in the tuple} \end{cases}$$

   – $f_4 : \{1, \dots, n\} \to \{\text{green}, \text{yellow}, \text{red}\}$

   $f_4(j) = \text{color of } t_j$.

   Thus to count the number of states, we can instead count the number of combinations of such functions and obtain the desired bound:

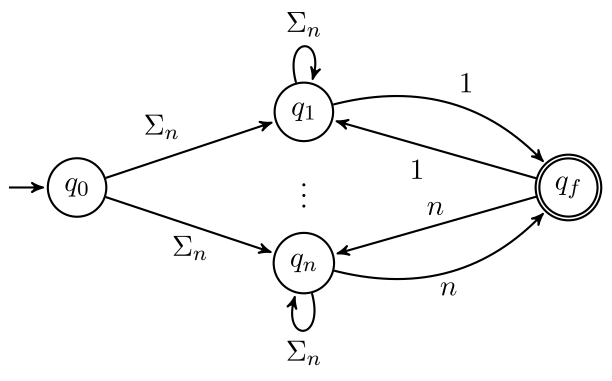
$$\begin{aligned} &\text{Number of states of the Rabin automaton} \\ \leq &(n+1)^n \cdot (n+1)^n \cdot (n+1)^n \cdot 3^n \\ \leq &(n+1)^{4n} \qquad (n \geq 2) \\ = &(2^{\lceil \log(n+1) \rceil})^{4n} \\ = &2^{O(n \cdot \log n)}. \end{aligned}$$

b) (Max Michel, Christoph Löding)
As preparation we note a fact about runs of Rabin automata.

Remark. *Let $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \Omega)$ be a Rabin automaton, $\Omega = ((E_1, F_1), \dots, (E_k, F_k))$ and let ρ_1, ρ_2 and ρ be runs with $Inf(\rho_1) \cup Inf(\rho_2) = Inf(\rho)$. If ρ_1 and ρ_2 are not successful, then ρ is not successful.*

Proof(of Remark). Assume towards a contradiction that ρ_1, ρ_2 are not successful, but ρ is successful. Then there exists an i with $\text{Inf}(\rho) \cap E_i = \emptyset$ and $\text{Inf}(\rho) \cap F_i \neq \emptyset$. Since $\text{Inf}(\rho_1) \cup \text{Inf}(\rho_2) = \text{Inf}(\rho_3)$ we have $Inf(\rho_1) \cap E_i = \emptyset$ and $Inf(\rho_2) \cap E_i = \emptyset$, and also $Inf(\rho_1) \cap F_i \neq \emptyset$ or $Inf(\rho_2) \cap F_i \neq \emptyset$. This means ρ_1 or ρ_2 is successful. Contradiction. □

Now consider the following Büchi automaton $\mathfrak{A}_n$ over alphabet $\Sigma_n = \{1, \dots, n, \#\}$



Let $L_n = L(\mathfrak{A}_n)$. $\mathfrak{A}_n$ has $n + 2$ states. We prove that any Rabin automaton for L_n has $n!$ states and thus there is no Rabin automaton for L_n with $2^{O(n)}$ states.

Remark on L_n. *$\alpha \in L_n \Leftrightarrow$ there are $i_1, \dots, i_k \in \{1, \dots, n\}$ such that the segments $i_1i_2, i_2i_3, \dots, i_{k-1}i_k, i_ki_1$ occur infinitely often in α. (We speak of a "cycle" in α)*

Proof

$\Leftarrow$ Define a successful run, assuming a cycle as above:
- $*$ go to q_{i_1}, stay there,
- $*$ with i_1i_2 jump via q_f to q_{i_2}, continue this way through the cycle up to q_{i_k} and then back to q_{i_1}, always via the final state q_f.

$\Rightarrow$ Let ρ be a successful run and a point where all segments ij of α occurring only finitely often have passed. Pick a q_i visited infinitely often. At some point the run must leave the loop on q_i via letter i and visit some q_{j_1} via q_f with letter j_1 because ρ is accepting. At some later point q_i is revisited.

$$q_i \xrightarrow{\quad i \quad} q_f \xrightarrow{\quad j_1 \quad} q_{j_1} \rightarrow \cdots \longrightarrow q_i$$

If $j_1 = i$, we are done. Otherwise consider the last occurrence of q_{j_1} before the next occurrence of q_i.

We have, for some $j_2 \neq j_1$

$$\begin{tikzcd} q_{j_1} \arrow[r, "j_1"] & q_f \arrow[r, "j_2"] & q_{j_2} \end{tikzcd}$$

Again if $j_2 = i$ we are done. Continuing this infinitely many steps is not possible, so after finitely many steps we get $q_{j_k} = q_i$ and obtain the desired cycle.

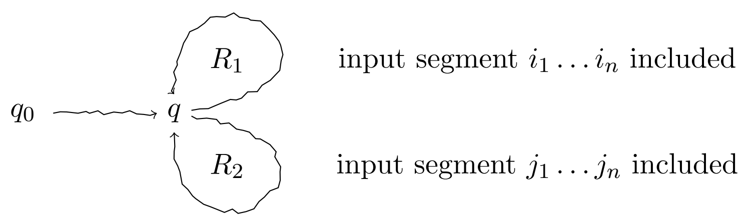
□

Now let $\mathfrak{R} = (Q, \Sigma, q_0, \delta, \Omega)$ be a Rabin automaton recognizing L_n. Consider two permuations of $1, \ldots, n$, say $i_1, \ldots, i_n$ and $j_1, \ldots, j_n$ and using these permutations define two ω-words $\beta_1 = (i_1 \ldots i_n\#)^\omega$ and $\beta_2 = (j_1 \ldots j_n\#)^\omega$. β_1, β_2 do not have cycles as defined above, so $\beta_1, \beta_2 \notin L_n$. Thus the runs ρ_1, ρ_2 of $\mathfrak{R}$ on β_1, β_2 are not successful. Let $R_1 = \mathit{Inf}(\rho_1), R_2 = \mathit{Inf}(\rho_2)$.

Claim: $R_1 \cap R_2 = \emptyset$.

Since there are $n!$ permutations, we obtain $n!$ disjoint cycles, so part (b) of the Theorem is proved.

Proof(of Claim). For a contradiction assume that $q \in R_1 \cap R_2$.


R_1
input segment $i_1 \ldots i_n$ included
q_0
q
R_2
input segment $j_1 \ldots j_n$ included

By looping through R_1, R_2 in alternation we get a new input word β including the segments $i_1 \ldots i_n, j_1 \ldots j_n$ infinitely often. The run ρ on β satisfies $\mathit{Inf}(\rho) = R_1 \cup R_2$ and thus by the remark above, ρ is not successful. We obtain a contradiction by finding that a (#-free) cycle occurs infinitely often in β which means $\beta \in L_n$. For this consider the smallest k such that $i_k \neq j_k$.

$$\begin{array}{ccccccccc}
 & & & & j_k & & & & \\
 & & & & \parallel & & & & \\
i_1 & \cdots & i_k & \cdots & i_r & & \cdots & i_n & \# \\
 & & \nparallel & & & & & & \\
j_1 & \cdots & j_k & & \cdots & j_s \cdots & & j_n & \# \\
 & & & & & \parallel & & & \\
 & & & & & i_k & & &
\end{array}$$

This leads to the cycle

$$\begin{array}{cccccccccc} i_k & i_{k+1} & \cdots & i_{r-1} & i_r & j_{k+1} & \cdots & j_{s-1} & j_s \\ &&&& \parallel &&&& \parallel \\ &&&& j_k &&&& i_k \end{array}$$

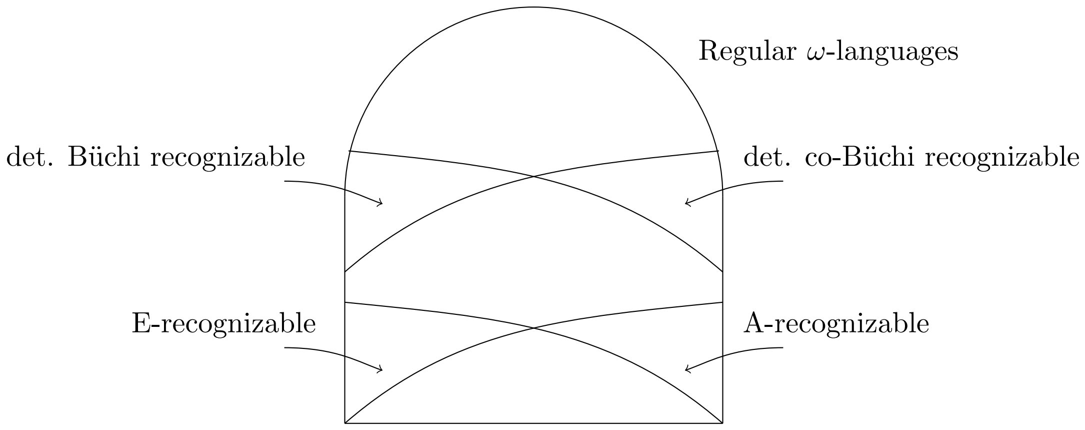
□

The same result holds for Muller automata but the proof is much more involved. The proof requires a different example automaton and different arguments. For the language L_n there is a Muller automaton with n^2 states (see the exercises).

Part III
Classification of the Regular ω-Languages

10 Four basic acceptance conditions

An overview:



One motivation for this part of the course is the study of nonterminating system runs. These are modelled by ω-words processed by automata. We consider the following properties of system runs α.

- Guarantee condition: Some prefix of α has a property P.
- Safety condition: All prefixes have a property P.
- Recurrence property: Again and again a prefix has property P.
- Persistence property: From some point onwards, all prefixes have property P.

We translate these conditions into automata acceptance conditions for deterministic automata.

Definition 56. *Given a deterministic finite automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$, call a run $\rho = \rho(0)\rho(1)\ldots$*

- *E-accepting if*
  $\exists i \quad \rho(i) \in F$.
- *A-accepting if*
  $\forall i \quad \rho(i) \in F$.
- *Büchi-accepting if*
  $\forall j \exists i \geq j \quad \rho(i) \in F$.
- *co-Büchi-accepting if*
  $\exists j \forall i \geq j \quad \rho(i) \in F$.

Call an automaton equipped with an E, A, Büchi, co-Büchi acceptance condition an E, A, Büchi, co-Büchi automaton, respectively. A language is called E-, A-, Büchi-, co-Büchi-recognizable, if it is recognized by an E-, A-, Büchi-, co-Büchi automaton, respectively.

Proposition.

1. *L is E-recognizable iff*
   $\overline{L}$ is A-recognizable.
2. *L is Büchi-recognizable iff*
   $\overline{L}$ is co-Büchi-recognizable.

Proof

1. Let $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$ be an E-automaton recognizing L.

$$\begin{aligned} \alpha \in \overline{L} &\Leftrightarrow \text{ $\mathfrak{A}$ does not E-accept α} \\ &\Leftrightarrow \text{ not: some prefix of α leads $\mathfrak{A}$ to a state in F} \\ &\Leftrightarrow \text{ each prefix of α leads $\mathfrak{A}$ to a non-final state} \\ &\Leftrightarrow \text{ $(Q, \Sigma, q_0, \delta, Q \backslash F)$ A-accepts α.} \end{aligned}$$

The other cases are treated completely analogously.

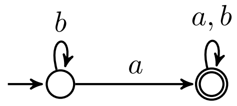
□

Proposition. *Let $\Sigma = \{a, b\}$. There are languages separating the acceptance models, as follows:*

1. *$\Sigma^* a \Sigma^\omega$ is E-recognizable, but not A-recognizable.*
2. *b^ω is A-recognizable, but not E-recognizable.*
3. *$(b^* a)^\omega$ is Büchi-recognizable, but not co-Büchi-recognizable.*
4. *$\Sigma^* b^\omega$ is co-Büchi-recognizable, but not Büchi-recognizable.*

Proof We prove *1)*, the other items follow from the complementation property presented above and from the already known fact that $\Sigma^* b^\omega$ is not Büchi-recognizable.

$\Sigma^* a \Sigma^\omega$ is recognized by the following automaton



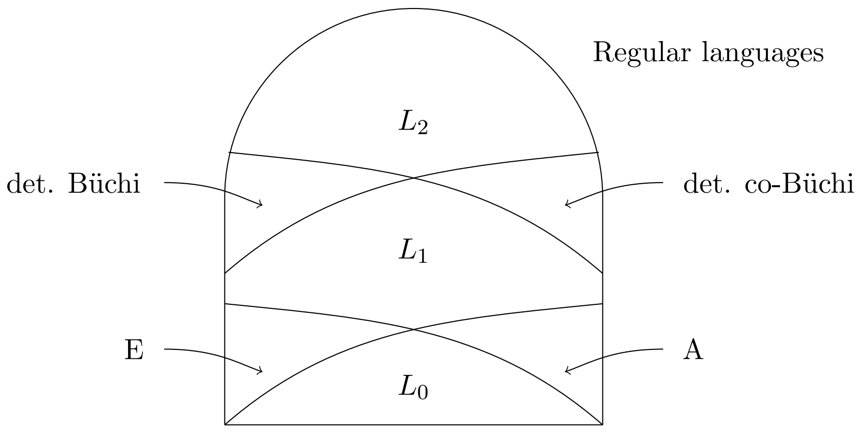
Assume $\Sigma^* a \Sigma^\omega$ is A recognized by an automaton $\mathfrak{A}$ with n states. $\mathfrak{A}$ accepts $b^n a b^\omega$, by visiting final states only. Since $\mathfrak{A}$ has only n states, there is a state repetition before the first visit of a. The (final!) states up to the point where a first repetition occurs will be the states of the run on b^ω. So $\mathfrak{A}$ A-accepts b^ω, contradiction. □

11 Landweber's Theorem

Establishing a Hierarchy
Proposition Let $\Sigma = \{a, b, c\}$ and define:

$$\begin{aligned}
L_0 &:= a\Sigma^\omega \\
L_1 &:= \underbrace{(\Sigma^* a \Sigma^\omega)}_{a \text{ occurs}} \cap \underbrace{\sim (\Sigma^* b \Sigma^\omega)}_{b \text{ does not occur}} \\
L_2 &:= \underbrace{(\Sigma^* a)^\omega}_{a \text{ occurs infinitely often}} \cap \underbrace{\sim (\Sigma^* b)^\omega}_{b \text{ occurs only finitely offten}}
\end{aligned}$$

Then L_0, L_1, L_2 are located as follows:



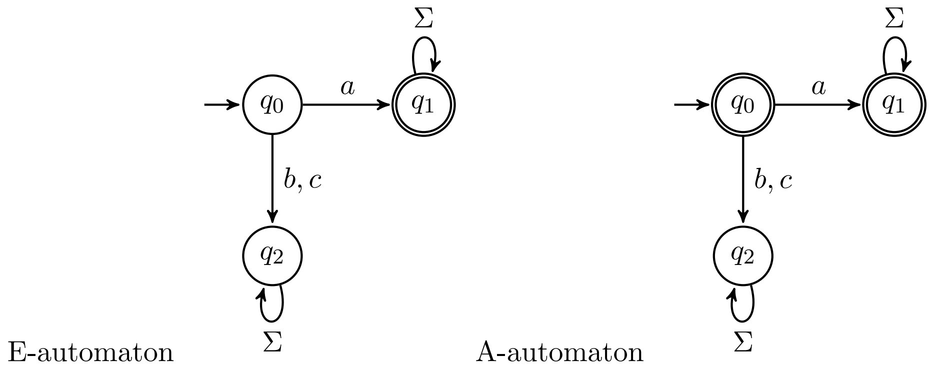
(a) L_0 is E-and A-recognizable

(b) L_1 is Büchi-and co-Büchi-recognizable, but neither E-recognizable nor A-recognizable

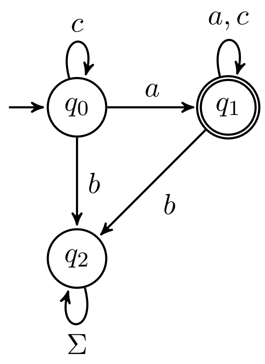
(c) L_2 is neither Büchi recognizable nor co-Büchi recognizable

Proof

(a)



(b) The depicted automaton, taken as a Büchi automaton or as a co-Büchi automaton, recognizes L_1:



Assume an E-automaton $\mathfrak{A}$ recognizes L_1 with n states. The automaton $\mathfrak{A}$ accepts the ω-word a^ω. It reaches a final state the first time, say after a^m with $m \leq n$. Hence the automaton $\mathfrak{A}$ also accepts the ω-word $a^m b^\omega$. This leads to a contradiction.

Assume an A-automaton $\mathfrak{A}'$ recognizes L_1 with n states. The automaton $\mathfrak{A}'$ accepts the ω-word $c^n a^\omega$, so it also accepts the ω-word c^ω. This leads to a contradiction.

(c) Assume a Büchi-automaton $\mathfrak{A}$ recognizes L_2. The automaton $\mathfrak{A}$ accepts the ω-word $(ac)^\omega$, therefore $\mathfrak{A}$ also accepts $(ac)^{n_1}b(ac)^\omega$,$(ac)^{n_1}b(ac)^{n_2}b(ac)^\omega$, etc.. Here n_1 is choosen that $\mathfrak{A}$ reaches a final state after $(ac)^{n_1}$, and n_2 so that $\mathfrak{A}$ reaches a final state after $(ac)^{n_1}b(ac)^{n_2}$, etc.. Repeating this argument there is a visit of a final state between any two b. Hence $\alpha = (ac)^{n_1}b(ac)^{n_2}b\ldots$ is accepted by $\mathfrak{A}$, but $\alpha \notin L_2$. This gives a contradiction.

Assume L_2 is recognized by a co-Büchi automaton $\mathfrak{B}$ then $\overline{L_2}$ is recognized by a Büchi automaton $\mathfrak{B}'$.

$\overline{L_2} = \{\alpha \in \Sigma^\omega \mid \alpha$ has infinitely many occurences of b or only finitely many occurences of $a\}$

The automaton $\mathfrak{B}'$ accepts c^ω, then it also accepts $c^{n_1}ac^\omega$. So choosing $n_1, n_2, \ldots$ as above, $\alpha = c^{n_1}ac^{n_2}ac^{n_3}a\ldots$ is accepted by $\mathfrak{B}$, but $\alpha \notin L_2$. This leads to a contradiction.

□

Language theoretical descriptions

Let $U, V \subseteq \Sigma^*$: We recall (a), (b) below and see that (c), (d) are shown with a simple transformation of an E-automaton into a classical finite automaton and conversely.

(a) L is regular	$\Leftrightarrow L = \bigcup_{i=1}^{n} U_i \cdot V_i^\omega$, with regular U_i, V_i
(b) L is deterministic Büchi-recognizable	$\Leftrightarrow L = \lim U$, with regular U
(c) L is E-recognizable	$\Leftrightarrow L = U \cdot \Sigma^\omega$, with regular U
(d) L is A- and E-recognizable	$\Leftrightarrow L = U \cdot \Sigma^\omega$, with finite U

Our next aim is the following: Given a Muller automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ decide wether $L(\mathfrak{A})$ is in fact Büchi-recognizable or even E-recognizable. We shall see that this decision can be based of simple properties of the accepting loops.

Definition 57. *Given an automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$. A* loop *in $\mathfrak{A}$ is a nonempty subset $S \subseteq Q$, such that $\delta(s, w) = s'$ and for all $s, s' \in S$ a word $w = a_1 a_2 \dots a_n \in \Sigma^+$ exists, such that for all $i \in \{1, \dots, n\}$: $\delta(s, a_1 a_2 \dots a_i) \in S$.*
Let S be a loop.

- *A loop S' with $S' \subseteq S$ is called a* subloop *of S.*
- *A loop S' with $S' \supseteq S$ is called a* superloop *of S.*

Given two loops S, S', then S' is reachable *from S, if there exists $q \in S$ and $q' \in S'$ such that there exists a word $w \in \Sigma^*$ with $\delta(q, w) = q'$.*

Remark. Given an infinite run ρ, so $\mathit{Inf}(\rho)$ is a loop. Given $\mathcal{F}$ define:
$\mathcal{F}_1 = \{S' \mid S' \text{ is reachable frome some loop in } \mathcal{F}\}$
$\mathcal{F}_2 = \{S' \mid S' \text{ is a superloop of } \mathcal{F}\}$

Theorem 58. ***Landweber***
Let $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ be a Muller automaton with reachable states only.

(a) $L(\mathfrak{A})$ is deterministic Büchi-recognizable $\Leftrightarrow$ $\mathcal{F} = \mathcal{F}_2$ ($\mathcal{F}$ is closed under superloops)

(b) $L(\mathfrak{A})$ is E-recognizable $\Leftrightarrow$ $\mathcal{F} = \mathcal{F}_1$ ($\mathcal{F}$ is closed under reachable loops)

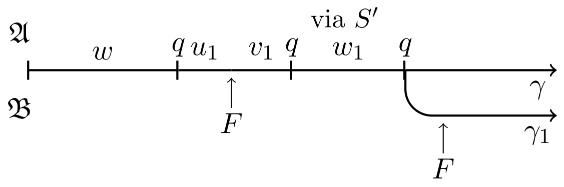
Proof

(a) $\Leftarrow$ Given a Muller automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$, where $\mathcal{F}$ is closed under superloops. Construct an equivalent Büchi-automaton $\mathfrak{A}'$. The automaton $\mathfrak{A}'$ has the state set $Q \times \mathcal{P}(Q)$ and the initial state $(q_0, \emptyset)$. The automaton $\mathfrak{A}'$ accumulates visited states in the second component until a loop of $\mathcal{F}$ is reached or exeeded, then the second component is reset to $\emptyset$. Declare $Q \times \emptyset$ as the set of final states. Then the following holds:

$\mathfrak{A}'$ accepts α
$\Leftrightarrow$ on input α, the run ρ of $\mathfrak{A}$ infinitely often passes trough loops $S' \supseteq S \in \mathcal{F}$ (thus $\mathit{Inf}(\rho) \supseteq S$)
$\Leftrightarrow$ for some $S \in \mathcal{F}$ and $S' \supseteq S$, precisely the states of S' are visited infinitely often
$\Leftrightarrow$ (since $\mathcal{F}$ is closed under superloops) for some $S' \in \mathcal{F}$, precisely the states of S' are visited infinitely often
$\Leftrightarrow$ $\mathfrak{A}$ accepts α

$\Rightarrow$ Given a Muller automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ and a Büchi automaton $\mathfrak{B} = (Q', \Sigma, q_0', \delta', F')$ equivalent to $\mathfrak{A}$. Let $S \in \mathcal{F}$ be a loop (reachable from q_0) and $S' \supseteq S$ be a superloop of S. Show $S' \in \mathcal{F}$.
Find an ω-word $\alpha \in L(\mathfrak{A})$ (therefore $\alpha \in L(\mathfrak{B})$) such that α causes $\mathfrak{A}$ to loop eventually through S'. Pick a state $q \in S$ and a word w such that $\delta(q_0, w) = q$ (recall that $\mathfrak{A}$ has only reachable states). Continue w by γ such that $\mathfrak{A}$ on $w\gamma$ loops through S again and again. In γ pick prefixes u_1, v_1, w_1 such that in $\mathfrak{B}$ the set of final states F is visited after reading wu_1, respective q is reached again after further reading v_1. Continue by w_1 which takes $\mathfrak{A}$ from q via S' to q.

Illustration



Continue by γ_1 which takes $\mathfrak{A}$ from q through loop S again and again. Pick segments u_2, v_2, w_2 of γ_1 as we did before for γ. Proceeding this way get the ω-word $\alpha = wu_1v_1w_1u_2v_2w_2\ldots$ such that after each u_i the automaton $\mathfrak{B}$ visits a final state. Hence $\mathfrak{B}$ accepts α and $\mathfrak{A}$ accepts α. By construction the infinity set of $\mathfrak{A}$ on α is S'. Hence $S' \in \mathcal{F}$.

(b) $\Leftarrow$ Assume $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ is a Muller automaton and $\mathcal{F}$ is closed under reachable loops. Define $\mathfrak{A}' = (Q, \Sigma, q_0, \delta, \bigcup \mathcal{F})$. Then the following holds:

$\mathfrak{A}'$ E-accepts α

$\Leftrightarrow$ $\mathfrak{A}$ reaches on α a state in $\bigcup \mathcal{F}$ (i.e. in some loop $S \in \mathcal{F}$)

$\Leftrightarrow$ $\mathfrak{A}$ reaches on α a loop $S' = \mathit{Inf}(\rho)$ from S

$\Leftrightarrow$ $S' = \mathit{Inf}(\rho_\alpha) \in \mathcal{F}$

$\Leftrightarrow$ $\mathfrak{A}$ accepts α

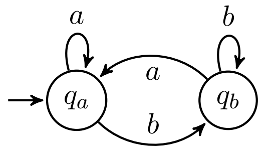
$\Rightarrow$ Let $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ be a Muller automaton. The automaton $\mathfrak{A}$ has w.l.o.g. only reachable states. Assume $L(\mathfrak{A})$ is also recognized by the E-automaton $\mathfrak{B} = (Q', \Sigma, q_0', \delta', F)$.

Let $q \in S \in \mathcal{F}$. Show that the loop S' reachable from S belongs to $\mathcal{F}$ (hence $S' \in \mathcal{F}$ and we are done).

Pick a word u such that $\delta(q_0, u) = q$. Continue u by an ω-word γ causing $\mathfrak{A}$ to loop through S. The ω-word $u\gamma$ is accepted by $\mathfrak{A}$, hence $\mathfrak{B}$ also accepts $u\gamma$, say after the prefix uv. Extend uv by a word w to reach the state q in $\mathfrak{A}$ (within the loop S). Extend uvw by an ω-word γ' leading $\mathfrak{A}$ to S' and going trough S' again and again. The automaton $\mathfrak{A}$ assumes the infinity set S' on $uvw\gamma'$. The automaton $\mathfrak{B}$ accepts $uvw\gamma'$, so $\mathfrak{A}$ accepts. Hence $S' \in \mathcal{F}$.

□

Example 59. Let $L = (a+b)^*a^\omega$.
The following Muller automaton with $\mathcal{F} = \{\{q_a\}\}$ recognizes L:



The acceptance component $\mathcal{F}$ is not closed under superloops (consider $\{q_a, q_b\}$). So by Landweber's Theorem L is not Büchi-recognizable. ⊠

Corollary 60. *Let $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ be a Muller automaton. The language $L(\mathfrak{A})$ is co-Büchi-recognizable $\Leftrightarrow$ $\mathcal{F}$ is closed under subloops.*

Proof (with (a) of Landweber's Theorem)
Assume the Muller automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ is given. Recall $L(\mathfrak{A})$ is co-Büchi-recognizable $\Leftrightarrow$ $\overline{L(\mathfrak{A})}$ Büchi-recognizable.
For $\overline{L(\mathfrak{A})}$ take the automaton $\mathfrak{A}$ with $\overline{\mathcal{F}} = \mathcal{P}(Q) \setminus \mathcal{F}$ and call it $\overline{\mathfrak{A}}$. Apply (a) of Landweber's Theorem to $\overline{\mathfrak{A}}$. It holds if S is a loop, then for $S \subseteq S'$ we have $S \in \overline{\mathcal{F}} \Rightarrow S' \in \overline{\mathcal{F}}$, in other words $S \notin \mathcal{F} \Rightarrow S' \notin \mathcal{F}$, equivalently $S' \in \mathcal{F} \Rightarrow S \in \mathcal{F}$. Hence $\mathcal{F}$ is closed under subloops. $\square$

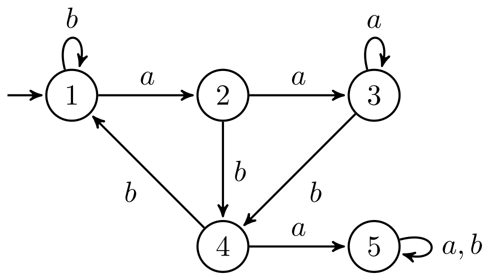
12 The Theorem of Staiger and Wagner

We analyze the property of regular languages to be deterministically Büchi and deterministically co-Büchi recognizable, a property called *obligation property.* To do so, we introduce a new automaton model, the model of Staiger and Wagner and prove as a main theorem that this model captures exactly those languages which are deterministically Büchi and deterministically co-Büchi recognizable.

Denote by $\mathrm{Occ}(\rho)$ the set of all states occuring in a run ρ: $\mathrm{Occ}(\rho) = \{q \in Q : q \text{ occurs in } \rho\}$.

Definition 61. *A Staiger-Wagner automaton (weak Muller automaton) is of the same form $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$ as a Muller automaton with the acceptance condition that a run ρ is accepting if and only if $Occ(\rho) \in \mathcal{F}$.*

Example 62. Let $L = \{\alpha \in \{a,b\}^\omega : aa \text{ never occurs or } aba \text{ occurs}\}$. Consider the following automaton



If aba occurs, then state 5 is reached. If aba does not occur, then the occurrence of aa leads to state 3. Thus the desired acceptance condition is given by

$$\mathcal{F} = \{\{1\}, \{1,2,4\}, \{1,2,4,5\}, \{1,2,3,4,5\}\}.$$

⊠

Remark.

- L is E-recognizable $\Rightarrow$ L is Staiger-Wagner recognizable.
- L is A-recognizable $\Rightarrow$ L is Staiger-Wagner recognizable.

Proof Consider a given automaton $(Q, \Sigma, q_0, \delta, F)$

- as an E-automaton: let $\mathcal{F} := \{P \subseteq Q : F \cap P \neq \emptyset\}$.
- as an A-automaton: let $\mathcal{F} := \{P \subseteq Q : P \subseteq F\}$.

□

Theorem 63. *$L \subseteq \Sigma^\omega$ is a Boolean combination of E-recognizable ω-languages if and only if L is Staiger-Wagner-recognizable.*

Proof

$\Rightarrow$ By induction on the construction of Boolean combinations.

1. If L is E-recognizable then L is Staiger-Wagner-recognizable by the above remark.
2. If L is Staiger-Wagner-recognizable by automaton $(Q, \Sigma, \delta, q_0, \mathcal{F})$, then $\overline{L}$ is recognized by $(Q, \Sigma, \delta, q_0, \mathcal{P}(Q) \backslash \mathcal{F})$.
3. If L_1, L_2 are Staiger-Wagner recognizable, say by $(P, \Sigma, \delta_1, p_0, \mathcal{F}_1)$ and $(Q, \Sigma, \delta_2, q_0, \mathcal{F}_2)$. Then $L_1 \cup L_2$ is Staiger-Wagner-recognizable by the product automaton with acceptance condition $\{(p_1, q_1), \ldots, (p_k, q_k)\} \in \mathcal{F} \Leftrightarrow \{p_1, \ldots, p_k\} \in \mathcal{F}_1$ or $\{q_1, \ldots, q_k\} \in \mathcal{F}_2$.

$\Leftarrow$ Consider a Staiger-Wagner-automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$. $\mathfrak{A}$ accepts with run ρ if and only if

$$\bigvee_{F \in \mathcal{F}} (\bigwedge_{q \in F} \underbrace{\exists i \quad \rho(i) = q}_{\substack{E\text{-aut. } (Q, \ldots, \{q\}) \\ \text{accepts}}} \wedge \bigwedge_{q \in Q \backslash F} \underbrace{\neg \exists i \quad \rho(i) = q}_{\substack{E\text{-aut. } (Q, \ldots, \{q\}) \\ \text{does not accept}}} .$$

Denote $L_q := L(Q, \ldots, \{q\})$. Then

$$\mathfrak{A} \text{ accepts } \alpha \Leftrightarrow \alpha \in \bigcup_{F \in \mathcal{F}} (\bigcap_{q \in F} L_q \cap \bigcap_{q \in Q \backslash F} \overline{L}_q).$$

□

Theorem 64 (Staiger, Wagner)**.** *L is Büchi-recognizable and co-Büchi-recognizable if and only if L is Staiger-Wagner-recognizable.*

Proof

$\Leftarrow$ Consider a Staiger-Wagner-automaton $(Q, \Sigma, q_0, \delta, \mathcal{F})$. Construct an automaton with state set $Q \times \mathcal{P}(Q)$ where in the second ("memory") component the visited states are accumulated. Declare as final state set $F := \{(q, P) : P \in \mathcal{F}\}$.

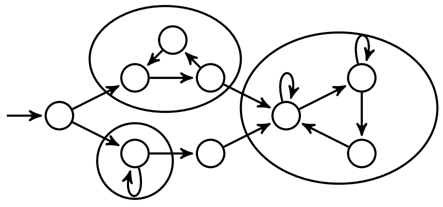
As a Büchi automaton, this automaton accepts α
$\Leftrightarrow$ for some $P \in \mathcal{F}$, the set of visited states is P infinitely often on α
$\Leftrightarrow$ $(Q, \Sigma, q_0, \delta, \mathcal{F})$ on α has an occurence set in $\mathcal{F}$.
$\Leftrightarrow$ $(Q, \Sigma, q_0, \delta, \mathcal{F})$ accepts α.

As a co-Büchi automaton, this automaton accepts α
$\Leftrightarrow$ for some $P \in \mathcal{F}$, only states (q, P) are visited from some point onwards
$\Leftrightarrow$ the occurence set of $(Q, \Sigma, q_0, \delta, \mathcal{F})$ on α is $P \in \mathcal{F}$.

$\Rightarrow$ Assume L is Büchi-recognizable and co-Büchi-recognizable. Consider a Muller automaton for L. Decompose the transition graph into its strongly connected components.

Recall: A strongly connected component (SCC) is a maximal strongly connected subgraph.

Example 65. Strongly connected components of a graph.



⊠

The SCCs are arranged as a partial order. For an SCC S let $S_+ := \{q \in Q \backslash S : q$ reachable from S in one step$\}$. By Landweber's Theorem, all loops in an SCC are accepting or all are not accepting. Call an SCC "good", if all loops are accepting. Now define the accepting condition in a way that $\mathfrak{A}$ accepts if and only if some good SCC S is reached but not the corresponding set S_+. This gives a Boolean combination of E-recognizable languages and hence a Staiger-Wagner automaton.

□

Weak Parity Automata

Just as the modification of Muller automata lead to Staiger-Wagner automata, also called weak Muller automata, a corresponding modification of parity automata leads to weak parity automata. As we will see, the equivalence of Muller and parity automata carries over to weak Muller and weak parity automata.

Definition 66. *A weak parity automaton is of the form $\mathfrak{A} = (Q, \Sigma, q_0, \delta, c)$ as a parity automaton, with the convention that a run ρ is accepting if the highest color visited is even, i.e. $max(Occ(c(\rho)))$ is even.*

Theorem 67. *L is Staiger-Wagner-recognizable if and only if L is recognized by a weak parity automaton.*

Proof

- $\Leftarrow$ Use a given weak parity automaton as Staiger-Wagner automaton, where the acceptance condition is given by a Boolean combination of conditions "color d is visited". (This is E-recognized by the same automaton with final states of color d.)

- $\Rightarrow$ A first possibility is to color the SCCs of the corresponding Muller automaton in an appropriate way. We give a direct proof without SCCs.

  Given a Staiger-Wagner automaton $\mathfrak{A} = (Q, \Sigma, q_0, \delta, \mathcal{F})$, introduce $\mathfrak{B}$ over $Q \times \mathcal{P}(Q)$, where in the first component $\mathfrak{A}$ is simulated and in the second component the visited states are accumulated. As coloring choose

$$c((q, P)) = \begin{cases} 2 \cdot |P| & P \in \mathcal{F} \\ 2 \cdot |P| - 1 & P \notin \mathcal{F}. \end{cases}$$

  Then the highest color visited in a run of $\mathfrak{B}$ is even if and only if the original run of $\mathfrak{A}$ accepts.

□

Part IV
Applications

We will address three areas of application of the theory of ω-automata

1. Theories of Logic
2. Model-Checking
3. Handling Sets of Reals

13 Theories of Logic

As a first application of automata theory on infinite words, we investigate theories of logic. Historically, Büchi invented his automaton model when he was researching on monadic second-order theories of arithmetic structures.
We consider structures like $(\mathbb{N}, +, \cdot, 0, 1)$ or the field of real numbers $(\mathbb{R}, +, \cdot, 0, 1)$. Recall that first-order predicate logic (FO) uses variables $x, y, z, \ldots$, ranging over elements of a structure. Terms are built from constant and variable symbols and inductively from function symbols. In our case terms are for example $0, 1, x, y, \ldots$ and $x + 1, x + 1 + 0, \ldots$.
Formulae are built by comparing terms via $=$ and inductively with the symbols $\neg, \wedge, \vee, \rightarrow, \exists, \forall$. A formula without free variables is called a sentence.
The first-order theory of a structure $\mathfrak{S}$, FO-Th($\mathfrak{S}$), is the set of all FO-sentences which are true in $\mathfrak{S}$.
The first-order theory of $\mathfrak{N} = (\mathbb{N}, +, \cdot, 0, 1)$ is the set of FO-sentences which are true in $\mathfrak{N}$. We denote it by $FO-Th(\mathbb{N}, +, \cdot, 0, 1)$.

Example 68. The sentence

$$\forall y \exists x_1, \ldots, x_4 : x_1 \cdot x_1 + x_2 \cdot x_2 + x_3 \cdot x_3 + x_4 \cdot x_4 = y$$

is true in $\mathfrak{N}$ (Lagrange's Theorem, saying that each natural number is a sum of four squares); the sentence

$$\forall y \exists x : x + x = y$$

is obviously false in $\mathfrak{N}$. ⊠

Gödel (1931) $FO-Th(\mathbb{N})$ is undecidable (not even enumerable).

Today's Proof: Reduction from undecidability of halting problem for 2-counter machines (2-register machines)

Instructions of 2-counter machines:

- l Incr(x)
- l Decr(x)
- l if $x = 0$ goto l' else goto l"

- Final instruction: s stop

Configuration of a 2-counter machine: (l, n_1, n_2).

Halting problem for 2-counter machine M: *Given M, does M reach from configuration $(1,0,0)$ the configuration $(s,0,0)$?*
Idea: Write down a FO-sentence φ_M (with $+, \cdot, 0, 1$) expressing that
$\exists$ *sequence* $(l_0, m_0, n_0), (l_1, m_1, n_1), \dots, \underbrace{(l_r, m_r, n_r)}_{(s,0,0)}$ where $(l_i, m_i, n_i) \vdash_M (l_{i+1}, m_{i+1}, n_{i+1})$ for $i < r$.
Problem: Express a quantifier $\exists(x_0, \dots, x_r)$ by $\exists y$.

- First attempt: Code $(x_0, \dots, x_r)$ by $p_0^{x_0} p_1^{x_1}, \dots, p_r^{x_r}$ where p_i denotes the ith prime number. But exponentation is not avaible with $+, \cdot$.
- Second attempt: Gödels β-function (more refined coding of sequences by numbers using only $+, \cdot$). Explanation in detail in *Ebb. Fl. Th., Chapter X, Section 6.*

The following theories are decidable:

- $FO\!-\!Th(\mathbb{N}, +, 0)$ "Presburger arithmetic"
- $FO\!-\!Th(\mathbb{N}, \cdot, 1)$ "Skolem arithmetic"

Gödels Remark: If we use relation quantifiers $(\exists R, \forall R)$ but only the structure $\mathfrak{N} = (\mathbb{N}, +1, 0)$ then we get again undecidability.

Proof To show that $SO\!-\!Th(\mathbb{N}, +1, 0)$ (the "second order theory of $\mathfrak{N}$") is undecidable, we define $+, \cdot$ in $\mathfrak{N}$:

$$\begin{aligned} x + y = z \Leftrightarrow\ & \text{each relation which contains } (0, y) \text{ and is closed under } +1 \text{ in both} \\ & \text{components, contains } (x, z) \\ \Leftrightarrow\ & \forall R(R(0, y) \land \forall s, t(R(s, t) \to R(s+1, t+1)) \to R(x, z) \end{aligned}$$

Multiplication: analogous □

Tarski's Question: What happens if we are only allowed set quantifiers?
The "monadic second order theory $MSO\!-\!Th(\mathbb{N}, +1, 0)$", also called $S1S$, is the set of true sentences in $\mathfrak{N} = (\mathbb{N}, +1, 0)$. We present the positive solution of this problem (by Büchi 1960). This was the original motivation to introduce automata on infinite words.

S1S and Büchi Automata

Syntax of S1S

Definition 69. *(Syntax of S1S) S1S-formulas are defined over* variables*:*

- first-order *variables $s, t, \dots, x, y, \dots$ (ranging over natural numbers, i.e. positions in ω-words),*

- second-order *variables $X, X_1, X_2, Y, Y_1, \dots$ (ranging over sets of natural numbers).*

Terms *are*

- *the constant* 0 *and first-order variables,*
- *for any term τ also its successor $\tau + 1$ (sometimes written as τ').*

Examples of terms are: $t, t+1, (t+1)+1, 0, 0+1, (0+1)+1$. We can now define three classes of S1S-formulas:

- Atomic formulas*: $X(\tau)$, $\sigma = \tau$ for terms σ, τ. Note that the atomic formula $X(\tau)$ is also denoted by $\tau \in X$.*
- First-order formulas *($\mathsf{S1S}_1$-formulas) are built up from atomic formulas using Boolean connectives and quantifiers $\exists, \forall$ over first-order variables.*
- S1S-formulas *are built up from atomic formulas using Boolean connectives and quantifiers $\exists, \forall$ over first-order variables and second-order variables.*

Example 70.

$$\varphi(X_1) := \exists X_2 \underbrace{(\forall x X_1(x) \rightarrow X_2(x))}_{\psi(X_1,X_2)}$$

Here $\psi(X_1, X_2)$ means "$X_1 \subseteq X_2$" and $\varphi(X_1)$ means "there exists a superset of X_1" ⊠

Semantics of S1S

We need a mathematical structure over which S1S-formulas can be interpreted.
Main Idea: Bit-ω-sequences define sets of natural numbers and conversely.

Definition 71. *(Semantics of S1S) We will*

- *use $\mathbb{N}$ as the universe for the first-order variables,*
- *use $2^{\mathbb{N}}$ (the powerset of $\mathbb{N}$) as the universe for the second-order variables,*
- *apply the standard semantics for Boolean connectives and quantifiers.*

We write $(\mathbb{N}, 0, +1, P_1, \dots, P_n) \models \varphi(X_1, \dots, X_n)$ if φ is true in this semantics, with $P_1 \subseteq \mathbb{N}, \dots, P_n \subseteq \mathbb{N}$ as interpretations of $X_1, \dots, X_n$.

We note that $\overline{P}$ can be coded by the ω-word $\alpha = \alpha(\overline{P}) \in (\{0,1\}^n)^\omega$ defined by

$$i \in P_k \iff (\alpha(i))_k = 1.$$

Then we simply write: $\alpha \models \varphi(X_1, \dots, X_n)$.

Example 72.

$$\varphi_1(X) = \forall x \exists y (y = x + 1 \wedge X(y))$$

Let $\alpha = 10101010\dots$, it codes the set $\{0, 2, 4, 6, \dots\}$ (set of even numbers). The formula $\varphi_1(X)$ says in α: "each successor is even". So $\alpha \not\models \varphi(x)$. ⊠

Example 73. Let M_1 = set of even numbers and M_2 = set of prime numbers. We obtain the ω-word

$$\alpha = \begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}$$

⊠

Definition 74. *(S1S-definable languages) An ω-language $L \subseteq (\{0,1\}^n)^\omega$ is* S1S-definable *if for some S1S-formula $\varphi(X_1, \dots, X_n)$ we have*

$$L = \{\alpha \in (\{0,1\}^n)^\omega \mid \alpha \models \varphi(X_1, \dots, X_n)\}.$$

Some syntactic sugar: We can define the relation $<$ over $\mathfrak{N}$ in *S1S*:

$$x < y \Leftrightarrow \forall X\big(X(x+1) \land \forall z\big(X(z) \to X(z+1)\big) \to X(y)\big)$$

From Büchi automata to S1S

Theorem 75. *(Büchi) An ω-language $L \subseteq (\{0,1\}^n)^\omega$ is Büchi recognizable iff L is S1S-definable.*

Example 76. Let $\Sigma = \{0,1\}^2$ and

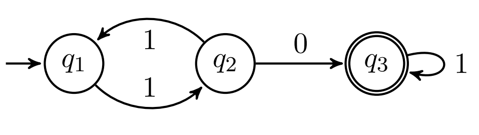
$$\varphi(X_1, X_2) := \exists x\Big(X_1(x) \land \exists y\big(x < y \land X_1(y)\big) \land \forall z\big(x < z \land z < y \to \neg X_2(z)\big)\Big)$$

then

$$\begin{aligned} L(\varphi) &= \Sigma^* \cdot \left(\begin{pmatrix}1\\0\end{pmatrix} + \begin{pmatrix}1\\1\end{pmatrix}\right) \cdot \left(\begin{pmatrix}0\\0\end{pmatrix} + \begin{pmatrix}1\\0\end{pmatrix}\right) \cdot \left(\begin{pmatrix}1\\0\end{pmatrix} + \begin{pmatrix}1\\1\end{pmatrix}\right) \cdot \Sigma^\omega \\ &= \Sigma^* \begin{pmatrix}1\\*\end{pmatrix} \begin{pmatrix}*\\0\end{pmatrix}^* \begin{pmatrix}1\\*\end{pmatrix} \Sigma^\omega \end{aligned}$$

⊠

Example 77. (Transformation of a Büchi automaton to a S1S-formula)
Let $\Sigma = \{0,1\}^2$



Input		1	1	1	0	1	1	1
Run	Y_1	1$*$	0	1$*$				
	Y_2	0	1$*$	0	$*$			
	Y_3	0	0	0		$*$	$*$	$*$

The stars mark the state at the given point of the input word. Naturally the automaton can only be in one state for each point of time. Therefore there is just one 1 in every column of the run. We use an existential quantification on sets Y_1, Y_2, Y_3 to express the existance of a run:

$$\begin{aligned}\varphi(X_1) \quad = \quad & \exists Y_1 Y_2 Y_3 \ (\text{Partition}(Y_1, \dots, Y_3) \wedge Y_1(0) \wedge \\ & \forall t((Y_1(t) \wedge X_1(t) \wedge Y_2(t')) \vee (Y_2(t) \wedge X_1(t) \wedge Y_1(t')) \\ & \vee (Y_2(t) \wedge \neg X_1(t) \wedge Y_3(t')) \vee (Y_3(t) \wedge X_1(t) \wedge Y_3(t'))) \\ & \wedge \forall s \exists t (s < t \wedge Y_3(t))).\end{aligned}$$

Partition is an expression for the above mentioned unambiguous description of the automaton state. Since there is just one 1 in every Y-bitvector, Y_1, Y_2, Y_3 have to form a partition of $\mathbb{N}$. $Y_1(0)$ states that the automaton starts in q_1. The following subformulas in the scope of the first $\forall$-quantifier represent the transition relation. The last subformula demands that the automaton enters the final state infinitely often. ⊠

Proof of Theorem 75 ($\Rightarrow$) In order to be able to translate an Büchi automata with m states, some formulas, which are needed, have to be prepared:
Preparation 1: $\text{Partition}(Y_1, \dots, Y_m) := \forall t \left(\bigvee_{i=1}^m Y_i(t)\right) \quad \wedge \quad \forall t\big(\neg \bigvee_{i \neq j} (Y_i(t) \wedge Y_j(t))\big)$.
Preparation 2: For $a \in \{0,1\}^n$, say $a = (b_1, \dots, b_n)$, we write $X_a(t)$ as an abbreviation for

$$(b_1)X_1(t) \wedge (b_2)X_2(t) \wedge \dots \wedge (b_n)X_n(t)$$

where $(b_i) = \neg$ if $b_i = 0$, and b_i is empty if $b_i = 1$. For instance,for $a = (1,0,1)$, we have $X_a(t) = X_1(t) \wedge \neg X_2(t) \wedge X_3(t)$.
Now we can translate any Büchi automaton to an equivalent S1S-formula: Given the Büchi automaton $\mathcal{A} = (Q, \{0,1\}^n, 1, \Delta, F)$ with $Q = \{1, \dots, m\}$, define

$$\begin{aligned}\varphi(X_1, \dots, X_n) \quad = \quad & \exists Y_1 \dots Y_m \Big(\text{Partition}(Y_1, \dots, Y_m) \wedge Y_1(0) \\ & \wedge \forall t \Big(\bigvee_{(i,a,j) \in \Delta} (Y_i(t) \wedge X_a(t) \wedge Y_j(t')) \Big) \\ & \wedge \forall s \exists t \big(s < t \wedge \bigvee_{i \in F} Y_i(t)\big)\Big).\end{aligned}$$

Obviously this is just a generalization of Example 77. The first line gives the partitioning of $\mathbb{N}$ and the start state 1. Line 2 describes all transitions of $\mathcal{A}$ and line 3 the acceptance condition. □

We conclude: A Büchi recognizable ω-language is existential second-order definable (within S1S).

From S1S to Büchi Automata

As we have seen in the last lecture, every Büchi recognizable ω-language is definable in monadic second-order logic. We now show that the converse also holds and thus obtain that monadic second-order logic over infinite words captures exactly the regular ω-languages.

Theorem 78. *If $L \subseteq (\{0,1\}^n)^\omega$ is S1S-definable, then L is Büchi recognizable.*

Proof We use induction over S1S-formulas $\varphi(X_1, \dots, X_n)$ for each n.

Preparation: To carry out the proof without the need to distinguish too many cases, we eliminate the use of first-order variables and quantifiers by introducing an equivalent logic $\mathrm{S1S}_0$. The idea is to rather write $\{y\}$ than y and modify the formula accordingly.

We introduce new atomic formulas:

- $\mathrm{Sing}(X)$, meaning X is a singleton,
- $X \subseteq Y$, with the obvious meaning,
- $\mathrm{Succ}(X, Y)$, meaning $X = \{x\}, Y = \{y\}$ with $x + 1 = y$.

The set of all $\mathrm{S1S}_0$-formulas is then obtained by closing the formulas under $\neg, \vee$ and $\exists$.

Lemma 79. *An S1S-formula $\varphi(X_1, \dots, X_n)$ can be rewritten as an $S1S_0$ formula.*

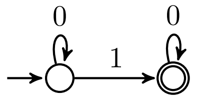
The proof is done by an easy induction. We just give an example providing the necessary idea: first-order quantifiers are replaced by second-order quantifiers and the quantified variables are relativized to singleton sets.

$$\forall x \exists y (x + 1 = y \wedge Z(y)) \mapsto \forall X(\mathrm{Sing}(X) \to \exists Y(\mathrm{Sing}(Y) \wedge \mathrm{Succ}(X, Y) \wedge Y \subseteq Z)).$$

Obviously each $\mathrm{S1S}_0$-formula can be rewritten as an S1S-formula, thus S1S and $\mathrm{S1S}_0$ are equivalent in expressive power.
We continue with the proof of the theorem, providing a Büchi automaton for each $\mathrm{S1S}_0$-formula via induction.

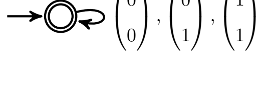
Induction start

- $\mathrm{Sing}(X_1)$:

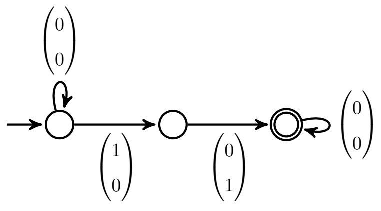
  

  Note that we are actually dealing with the formula $\varphi(X_1, \dots, X_n) = \mathrm{Sing}(X_i)$ and thus transitions of the automaton have officially to be labelled with letters from $\{0,1\}^n$. The interesting part happens in the i-th component, which we pictured above.

- $X_1 \subseteq X_2$:

- Succ(X_1, X_2):

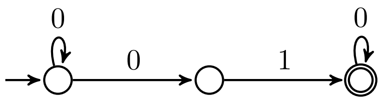


Induction step

- $\vee$: Given Büchi automata $\mathfrak{A}_1, \mathfrak{A}_2$ for $\varphi_1(X_1, \ldots, X_n), \varphi_2(X_1, \ldots, X_n)$. For $\varphi_1 \vee \varphi_2$ take the union automaton of $\mathfrak{A}_1$ and $\mathfrak{A}_2$.
- $\neg$: Given Büchi automaton $\mathfrak{A}$ for $\varphi(X_1, \ldots, X_n)$ use the complement automaton for $\neg\varphi$.
- $\exists$: Given Büchi automaton $\mathfrak{A}$ for $\varphi(X_1, \ldots, X_{n+1})$ over $\{0,1\}^{n+1}$. Find a Büchi automaton $\mathfrak{B}$ for $\psi(X_1, \ldots, X_n) = \exists X_{n+1}\varphi(X_1, \ldots, X_{n+1})$. Intuitiveley, on $\alpha \in (\{0,1\}^n)^\omega$, $\mathfrak{B}$ guesses the $(n+1)$-rst component and on α with the extra component works like $\mathfrak{A}$. The implementation of this idea is really simple, $\mathfrak{B}$ has the same transitions as $\mathfrak{A}$ but in the transitions the $(n+1)$-rst component is deleted.

$\square$

Example 80. The formula $\phi(X_2) = \exists X_1 \text{Succ}(X_1, X_2)$ is equivalent to the following Büchi automaton:



$\boxtimes$

Corollary. *S1S, the monadic second-order theory of* $(\mathbb{N}, +1, 0)$ *is decidable.*

Proof Use Büchi's Theorem to construct for $\varphi(X_1, \ldots, X_n)$ a Büchi automaton $\mathfrak{A}_\varphi$ for the case $n = 0$. This automaton has unlabelled transitions and it holds that $(\mathbb{N}, +1, 0) \models \varphi \Leftrightarrow \mathfrak{A}_\varphi$ has a successful run. The right-hand side is easily decidable using e.g. depth-first search over $\mathfrak{A}_\varphi$. $\square$

Remark on the complexity of deciding S1S. A result of Meyer and Stockmeyer (1974) shows that S1S is non-elementary, i.e. there is no algorithm deciding the theory in time

$$\left.2^{\cdot^{\cdot^{\cdot^{2^{|\varphi|}}}}}\right\}k$$ for any fixed k.

In his paper from 1962, Büchi asked three questions related to his work.

1. Consider the structure of the infinite binary tree rather than $(\mathbb{N}, +1, 0)$, which is in Büchi's formulation the structure $(\mathbb{N}\backslash\{0\}, 2x, 2x+1, 1)$. Is the monadic second-order theory of this structure, S2S, decidable? Rabin answered this question positively in 1969.

2. Consider ordinals rather than $(\mathbb{N}, +1, 0)$. Büchi showed the decidability of the monadic second-order theory of each countable ordinal.
3. Use an S1S-formula $\varphi(X_1, X_2)$ as specification of a function $f : \{0,1\}^\omega \to \{0,1\}^\omega$ with the requirement that $\varphi(X, f(X))$ for all X. The existence of such a function can be reformulated as the existence of a winning strategy of the second player in a two-player game. The problem was solved by Büchi and Landweber in 1969 and started the research on regular games.

We consider three further possibilities of extending decidability of S1S.

1. The two-dimensional infinite grid is the structure $G_2 = (\mathbb{N} \times \mathbb{N}, s_1, s_2, (0,0))$, with $s_1(i,j) = (i+1, j)$ and $s_2(i,j) = (i, j+1)$. The monadic second-order theory of G_2, $M\text{-}Th(G_2)$ is undecidable.
2. The monadic second-order theory of $(\mathbb{N}, +1, 2x, 0)$ is undecidable.
3. The monadic second-order theory of $(\mathbb{N}, +1, 0, P)$ where $P \subseteq \mathbb{N}$ is decidable if

   - P is the set of factorial numbers,
   - P is the set of powers of 2.

   It is open whether it is decidable if P is the set of all prime numbers.

We give a hint on the third item. Recall that Büchi's Theorem yields for $\varphi_1(X_1)$ an automaton $\mathfrak{A}_\varphi$ on $\{0,1\}$. Now $(\mathbb{N}, +1, 0) \models \varphi(P) \Leftrightarrow \mathfrak{A}_\varphi$ accepts α_P. Thus the monadic second-order theory of $(\mathbb{N}, +1, 0, P)$ is decidable if and only if the following decision problem Acc_P is decidable:

Given a Büchi automaton $\mathfrak{A}$, does $\mathfrak{A}$ accept α_P?

We give two examples for P, where this problem is quite easy to decide.

Example 81. Let $\alpha_P := \underbrace{10110}_{prefix}\,\underbrace{100}_{period}\,100100\ldots$ be an ultimately periodic word. It is easy to decide whether any given Büchi automaton accepts this α_P. ⊠

Example 82. Let P be the set of all factorial numbers, $P = \{1, 2, 6, 24, 120, 720, \ldots\}$, i.e. $\alpha_P = 011000100\ldots$. One can use a pumping argument to show that a Büchi automaton accepts this α_P, iff it accepts a modified, ultimately periodic ω-word α'. The latter condition is decidable as seen above. ⊠

14 Some undecidability results

Recall that the monadic second-order theory of $(\mathbb{N}, +1, 0)$, also called S1S, is decidable. We showed that for certain predicates $P \subseteq \mathbb{N}$, again M-Th$(\mathbb{N}, +1, 0, P)$ is decidable. As an example we considered $P = \{n! : n \geq 0\}$. The method to show this uses the fact that MSO-formulas and Büchi automata can be used interchangably. We now consider the expansion of $(\mathbb{N}, +1, 0)$ with a unary function $f : \mathbb{N} \to \mathbb{N}$ and show that for certain functions M-Th$(\mathbb{N}, +1, 0, f)$ is undecidable.

Theorem 83. *Let $d : \mathbb{N} \to \mathbb{N}$ be the 'double function', defined as $d(x) = 2x$. M-Th$(\mathbb{N}, +1, 0, d)$ is undecidable.*

Proof By Gödel's remark (see section above) the second-order theory of $(\mathbb{N}, +1, 0)$ (with relation quantifiers) is undecidable. We are able to imitate relation quantifiers by set quantifiers in $(\mathbb{N}, +1, 0, d)$.

We show how to imitate binary relation quantifiers, relations of higher arity are treated analogously. Let R be a binary relation, $R = \{(r_1, s_1), (r_2, s_2), \ldots\}$. Code R by a set $P_R = \{k_1, l_1, k_2, l_2, \ldots\}$, where $k_1 < l_1 < k_2 < l_2, \ldots$. Each l_i is a pointer to r_i, each k_i is a pointer to s_i. Then $R(r, s)$ iff there are k, l in P_R pointing to r, s respectively, such that k occurs at an odd position in the ordering of the elements of P_R and l on the next position.

Definition 84. *For each $n \geq 0$ define the set of pointers $S(n) := \{(2n + 1) \cdot 2^i : i \geq 0\}$.*

For example 4 has the pointers $9, 18, 36, 72, \ldots$. Any number > 0 points to a unique n and there are infinitely many pointers to each n. Furthermore,

$$p \text{ points to } n \Leftrightarrow \forall X(2n + 1 \in X \wedge \forall x(x \in X \to d(x) \in X) \to p \in X).$$

Given R, we can now define P_R inductively. Let $R = \{(r_1, s_1), (r_2, s_2), \ldots\}$. Let

$$\begin{aligned} k_1 &= r_1 \\ l_1 &= \text{smallest pointer } > k_1 \text{ to } s_1 \\ k_2 &= \text{smallest pointer } > l_1 \text{to } r_2 \\ &\vdots \end{aligned}$$

Exercise: Show that the predicate Odd(X, x), expressing that x occurs at an odd position in the ordering of X is definable in M-Th$(\mathbb{N}, +1, 0)$. Show that the predicate Next(X, x, y), expressing that $x, y \in X$ and in the ordering of X, y comes next after x is definable in M-Th$(\mathbb{N}, +1, 0)$.

Now we express a relation quantifier $\exists R\varphi(R)$ by $\exists X\varphi*$ where we replace Rxy by

$$\exists x'\exists y'(\text{Odd}(X, x') \wedge \text{Next}(X, x', y') \wedge \text{"}x' \text{ points to } x\text{"} \wedge \text{"}y' \text{ points to } y\text{"}).$$

□

We now consider the infinite grid. Recall that $G_2 = (\mathbb{N} \times \mathbb{N}, s_1, s_2, (0,0))$.

Theorem 85. *The monadic second-order theory of the infinite grid is undecidable.*

Proof by reduction of the halting problem for Turing machines. For any Turing machine M construct a sentence φ_M of the monadic second-order language of G_2 such that M halts when started on the empty tape iff $G_2 \models \varphi_M$.

Without loss of generality we assume that M works on a left-bounded tape. A configuration of M is a word $uq\beta$, where u is a finite word over the tape alphabet, q is a state of M and β is an infinite word over the tape alphabet. A halting computation of M can be coded by a finite sequence of configuration words $C_0, C_1, \ldots C_m$. Let $a_0, \ldots, a_n$ be the tape symbols (a_0 is the blank) and let $q_0, \ldots, q_m$ be the states of M with halting state q_m. Assume that after a visit of q_m the reached configuration is repeated. We can arrange the configurations row by row in a right-infinite rectangular array:

$$\begin{array}{ccccc}
\enclose{circle}{q_0} & a_0 & a_0 & a_0 & a_0 \cdots \\
a_1 & \enclose{circle}{q_1} & a_0 & a_0 & a_0 \cdots \\
\enclose{circle}{q_2} & a_1 & a_2 & a_0 & a_0 \cdots \\
a_3 & \enclose{circle}{q_3} & a_2 & a_0 & a_0 \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots
\end{array}$$

The sentence φ_M will express over G_2 the existence of such a sequence of configurations. We use set variables $X_0, \ldots, X_n, Y_0, \ldots, Y_m$. X_i collects the grid positions where a_i occurs, Y_i collects the grid positions where state q_i occurs.

$$\begin{aligned}
\varphi_M := &\exists X_0 \ldots X_n Y_0 \ldots Y_m (\text{Partition}(X_0, \ldots, Y_k) \\
&\quad \text{"the first row is the initial configuration"} \\
&\quad \text{"a successor row is the successor configuration of the preceding one"} \\
&\quad \text{"at some position the halting state is reached"})
\end{aligned}$$

A closer analysis shows that this formula is monadic second-order. To express that a successor row is the successor configuration of the preceding one, note that at each position in a configuration only the three positions around it can change in the successor configuration. A move to the right from state q to q' and printing b' over b corresponds to the following change:

$$\begin{tikzcd}
a & q \arrow[dr] & b & c \\
a & b' & q' & c
\end{tikzcd}$$

□

15 Model-Checking

The model-checking problem

Problem: Given a state based (non-terminating) program P and a specfication φ, does P satisfy φ (*short:* $P \models \varphi$) ?

Classical Logic: Given a structure $\mathfrak{N}_{succ} = (\mathbb{N}, +1)$ and a MSO-sentence φ, is φ true in $\mathfrak{N}_{succ}$ (*short:* $\mathfrak{N}_{succ} \models \varphi$) ?

Remark.

- A structure is fixed in mathematics, but varies in computer science.
- Positive solutions in computer science only with efficient algorithms.

Plan:

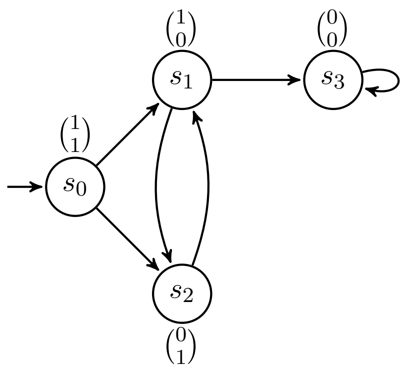
1. Structures for representation of programs: Kripke structures
2. Specification language: Linear-time temporal logic (*LTL*)
3. Adaption of automata theoretic methods for *MCP* (Model-Checking Problem): Transformation from *LTL* to Büchi automata

Definition 86. *A* Kripke structure $\mathcal{M} = (S, R, \lambda)$ *over* n *state properties* $p_1, \dots, p_n$ *is defined as follows:*

- *finite state space* S
- *transition relation* $R \subseteq S \times S$
- *labeling function* $\lambda : S \to \mathbb{B}^n$ *(where* $\lambda(s) = (b_1, \dots, b_n)$ *means in state* s *property* p_i *holds iff* $b_i = 1$*)*

A pointed Kripke structure has a designated start state s *and is denoted as* $(\mathcal{M}, s)$*. A path (execution sequence) trough* $(\mathcal{M}, s)$ *is a state sequence* $s_0, s_1, s_2, \dots$ *with* $s = s_0$ *and* $(s_i, s_{i+1}) \in R$*. A label sequence is the sequence of* λ*-values of an execution sequence.*

Example 87. (over p_1, p_2)



Path:	s_0	s_2	s_1	s_2	s_1	s_3	s_3	s_3	$\dots$
Label Sequence:	$\binom{1}{1}$	$\binom{0}{1}$	$\binom{1}{0}$	$\binom{0}{1}$	$\binom{1}{0}$	$\binom{0}{0}$	$\binom{0}{0}$	$\binom{0}{0}$	$\dots$

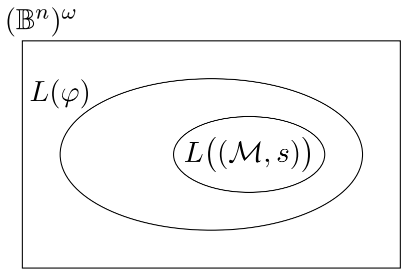
$\boxtimes$

Model-Checking Problem reformulated: We use a logic $\mathcal{L}$ which can specify properties of ω-sequences over $\mathbb{B}^n$.

Given $(\mathcal{M}, s)$ and $\varphi \in \mathcal{L}$, does every label sequence of $(\mathcal{M}, s)$ satisfy φ?
We may write this as an inclusion problem: "$L((\mathcal{M}, s)) \subseteq L(\varphi)$"?

Approach for a solution:

1. Transform $(\mathcal{M}, s)$ into a Büchi automaton $\mathcal{A}_{(\mathcal{M},s)}$, accepting the label sequence of $(\mathcal{M}, s)$.
2. Transform $\neg\varphi$ into a Büchi automaton $\mathcal{A}_{\neg\varphi}$.
3. Build the intersection automaton $\mathcal{B}$ (= "$\mathcal{A}_{(\mathcal{M},s)} \otimes \mathcal{A}_{\neg\varphi}$").
4. Check $\mathcal{B}$ for non-emptiness (if $L(\mathcal{B}) \neq \emptyset$, then $\alpha \in L(\mathcal{B})$ describes an error scenario).
   Illustration:



$$L\big((\mathcal{M}, s)\big) \subseteq L(\varphi) \Leftrightarrow L\big((\mathcal{M}, s)\big) \cap L(\neg\varphi) = \emptyset$$

Remark. With MSO-logic item 2 is not practical.

Linear-time temporal logic, LTL

Definition 88. Linear-time temporal logic *(Pnueli 1977) over state properties $p_1, \dots, p_n$ is built up using boolean operators $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ and the temporal operators:*

- **X** *"next" (unary), written* $\mathbf{X}\varphi$
- **F** *"eventually" (unary), written* $\mathbf{F}\varphi$
- **G** *"always" (unary), written* $\mathbf{G}\varphi$
- **U** *"until" (binary), written* $\psi\mathbf{U}\varphi$

Example 89. Formulas over p_1, p_2:

(a) $\mathbf{F}p_1$

(b) $p_1 \wedge \mathbf{X}\neg p_1 \wedge \mathbf{XX}\neg p_1 \wedge \mathbf{G}(p_1 \leftrightarrow \mathbf{XXX}p_1)$

(c) $\mathbf{GF}p_1$

(d) $\mathbf{G}(p_1 \rightarrow \mathbf{XF}p_2)$

(e) $\mathbf{G}(p_1 \rightarrow \mathbf{X}(p_2\mathbf{U}p_1)$

These formulas are interpreted in ω-sequences α over $\mathbb{B}^n$:

(a) $\mathbf{F}p_1$ means: exists t with $(\alpha(t))_1 = 1$

(b) $p_1 \wedge \mathbf{X}\neg p_1 \wedge \mathbf{XX}\neg p_1 \wedge \mathbf{G}(p_1 \leftrightarrow \mathbf{XXX}p_1)$ means: $(\alpha)_1 = 100100100\dots$

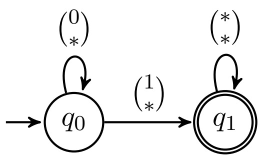
(c) $\mathbf{GF}p_1$ means: $(\alpha)_1$ has infinitely many 1

(d) $\mathbf{G}(p_1 \rightarrow \mathbf{XF}p_2)$ means: after each 1 in $(\alpha)_1$ there is an 1 in $(\alpha)_2$

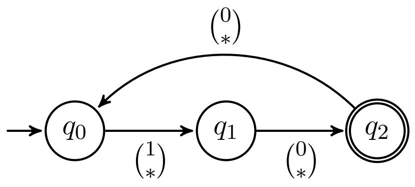
(e) $\mathbf{G}(p_1 \rightarrow \mathbf{X}(p_2\mathbf{U}p_1)$ means:
$\forall t\big((\alpha(t))_1 = 1 \rightarrow \exists s > t(\alpha(s))_1 \wedge \forall z(t < z < s \rightarrow (\alpha(z))_2 = 1)\big)$

Büchi automata:

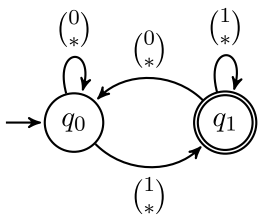
(a)



(b)



(c)



$\boxtimes$

Notation If $\alpha = \alpha(0)\alpha(1)\ldots \in (\mathbb{B}^n)^\omega$, then

1. α^s stands for $\alpha(s)\alpha(s+1)\ldots$, so $\alpha = \alpha^0$.
2. $(\alpha(s))_i$ is the i-th component of $\alpha(s)$.

Definition 90. *(Semantics of LTL)*
Define the satisfaction relation $\alpha^s \models \varphi$ inductively over the construction of φ as follows:

- $\alpha^s \models p_i$ *iff* $(\alpha(s))_i = 1$.
- $\alpha^s \models \neg\varphi$ *iff not* $\alpha^s \models \varphi$.
- *similarly for* $\vee, \wedge, \rightarrow$.
- $\alpha^s \models \mathbf{X}\varphi$ *iff* $\alpha^{s+1} \models \varphi$.
- $\alpha^s \models \mathbf{F}\varphi$ *iff for some* $t \geq s$: $\alpha^t \models \varphi$.
- $\alpha^s \models \mathbf{G}\varphi$ *iff for all* $t \geq s$: $\alpha^t \models \varphi$.
- $\alpha^s \models \varphi\mathbf{U}\psi$ *iff for some* $t \geq s$, $\alpha^t \models \psi$ *and for all* $k = s, \ldots t-1$: $\alpha^k \models \varphi$.

Example 91. $\alpha \models \mathrm{XX}(p_2 \rightarrow \mathrm{F}p_1)$

iff $\alpha^2 \models p_2 \rightarrow \mathrm{F}p_1$

iff if $(\alpha(2))_2 = 1$ then $\alpha^2 \models \mathrm{F}p_1$

iff if $(\alpha(2))_2 = 1$ then $\exists j \geq 2$: $(\alpha(j))_1 = 1$

iff "if second component of $\alpha(2)$ is 1, then the first component of some $\alpha(j)$ with $j \geq 2$ is 1".

For example, this is true in: $\alpha = \binom{1}{0}\binom{0}{0}\binom{1}{1}\binom{0}{1}\binom{1}{0}\binom{0}{1}\cdots$ ⊠

From Kripke structures to Büchi automata

Remark. Given a pointed Kripke structure $(\mathcal{M}, s)$ with $\mathcal{M} = (S, R, \lambda)$, $\lambda : S \rightarrow \mathbb{B}^n$, construct a Büchi automaton $\mathcal{A}_{\mathcal{M},s} = (S, \mathbb{B}^n, s, \Delta, S)$ with

$$(s, (b_1 \ldots b_n), s') \in \Delta \text{ iff } (s, s') \in R \text{ and } \lambda(s) = (b_1 \ldots b_n).$$

So a transition gets the label of the source state.

From LTL to Büchi automata

Our aim: Transform formulae φ into a Büchi automata $\mathfrak{A}$ such that $L(\varphi) = L(\mathfrak{A})$.

Preparation: To simplify the inductive structure of formulas, we only consider the temporal operators X and U. Eliminate F and G by the rules:

$\mathrm{F}\varphi$	is equivalent to	$\mathtt{tt}\mathrm{U}\varphi$ with $\mathtt{tt} \equiv p_1 \vee \neg p_1$.
$\mathrm{G}\varphi$	is equivalent to	$\neg\mathrm{F}\neg\varphi$

Recall:

$$\alpha \models \varphi\mathbf{U}\psi \Leftrightarrow \exists t(\alpha^t \models \psi \wedge \forall s < t\ \alpha^s \models \varphi$$

So we have the following: $\varphi\mathbf{U}\psi \equiv \psi \vee (\varphi \wedge \mathbf{X}(\varphi\mathbf{U}\psi))$
We evaluate LTL-formulae in a given sequence α by increasing complexity of subformulas.

Example 92. Let $\alpha \in (\mathbb{B}^n)^\omega$, $\varphi_1, \ldots, \varphi_n$ the list of subformulas of φ. Illustration for $\varphi = p_1 \vee \mathrm{X}(\neg p_2 \mathrm{U} p_1)$:

p_1	0	0	1	0	1	0	0	1	0	0	0	...
p_2	0	0	0	1	0	0	0	1	0	0	0	...
$\neg p_2$	1	1	1	0	1	1	1	0	1	1	1	...
$\neg p_2 \mathrm{U} p_1$	1	1	1	0	1	1	1	1	0	0	0	...
$\mathrm{X}(\neg p_2 \mathrm{U} p_1)$	1	1	0	1	1	1	1	0	0	0	0	...
$p_1 \vee \mathrm{X}(\neg p_2 \mathrm{U} p_1)$	1	1	1	1	1	1	1	1	0	0	0	...

⊠

Definition 93. *Given φ over $p_1, \ldots, p_n$ let $\varphi_1, \ldots, \varphi_{n+m}$ be the subformulas of φ listed in increasing complexity, so $\varphi_i = p_i$, for $i = 1, \ldots, n$. The* φ-expansion *of $\alpha \in (\mathbb{B}^n)^\omega$ is the sequence $\beta \in (\mathbb{B}^{n+m})^\omega$ defined as follows:*

$$(\beta)_1 = (\alpha)_1, \ldots, (\alpha)_n = (\beta)_n$$

$$\textit{for } i = n+1, \ldots, n+m : (\beta(t))_i = 1 \Leftrightarrow \alpha^t \models \varphi_i$$

Compatibility conditions that hold in a φ-expansion β of α:

$$\begin{array}{llll}
\varphi_j = \neg\varphi_{j_1} & \Rightarrow & (\beta(t))_j = 1 \text{ iff } & (\beta(t))_{j_1} = 0 \\
\varphi_j = \varphi_{j_1} \wedge \varphi_{j_2} & \Rightarrow & (\beta(t))_j = 1 \text{ iff } & (\beta(t))_{j_1} = 1 \text{ and } (\beta(t))_{j_2} = 1 \\
\varphi_j = \varphi_{j_1} \vee \varphi_{j_2} & \Rightarrow & (\beta(t))_j = 1 \text{ iff } & (\beta(t))_{j_1} = 1 \text{ or } (\beta(t))_{j_2} = 1 \\
\varphi_j = \mathrm{X}\varphi_{j_1} & \Rightarrow & (\beta(t))_j = 1 \text{ iff } & (\beta(t+1))_{j_1} = 1 \\
\varphi_j = \varphi_{j_1}\mathrm{U}\varphi_{j_2} & \Rightarrow & (\beta(t))_j = 1 \text{ iff } & (\beta(t))_{j_2} = 1 \text{ or } \big[(\beta(t))_{j_1} = 1 \\
 & & & \text{and } (\beta(t+1))_j) = 1\big]
\end{array}$$

To ensure the satisfaction of a subformula $\varphi_j = \varphi_{j_1} \mathrm{U} \varphi_{j_2}$ we have to add the condition

$$(*) \text{ for infinitely many } t(\beta(t))_j = 0 \text{ or } (\beta(t))_{j_2} = 1.$$

The first conditions are local (controllable by comparing successive column vectors of β). The last condition $(*)$ is non-local.

Theorem 94. *The compatibility conditions of φ-expansion β are satisfied in the φ-expansion of α, and they fix β uniquely.*

Recall the compatibility conditions introduced in the last lecture for an ω-word β over $\mathbb{B}^{n+m}$, associated with a word $\alpha \in \mathbb{B}^n$ and a formula φ with m subformulas. It remains to be proved that given α and assuming that the compatibility conditions are satisfied for β, then β is the φ-expansion of α.

Proof by induction on the indices of subformulas.

- Components $1, \ldots, n$ are fixed, thus α agrees with β on the first n components.
- Clearly, the jth component of β is fixed if φ_j is a negated formula, a conjunction, a disjunction or an X-formula.
- Let $\varphi_j = \varphi_{j_1} U \varphi_{j_2}$. $(\beta(t))_{j_2} = 1$ implies $(\beta(t))_j = 1$ and thus it follows via backward induction from the local compatibility conditions that if $(\beta(t))_{j_2} = 1$ then $(\beta(s))_j$ is fixed for $s \leq t$. We now distinguish two cases.

  Case 1. $(\beta(t))_{j_2} = 1$ for infinitely many t. Then β is fixed for all times t by the argument above.

  Case 2. For some t_0 $(\beta(t))_{j_2} = 0$ for $t \geq t_0$. Then also $(\beta(t))_j = 0$ for $t \geq t_0$, since otherwise $(\beta(t_1))_j = 1$ for some $t_1 \geq t_0$. This contradicts $(*)$; recall $(*)$: for infinitely many t $(\beta(t))_j = 0$ or $(\beta(t))_{j_2} = 1$.

$\square$

We now construct a generalized Büchi automaton for φ over $p_1, \ldots, p_n$ with subformulas $\underbrace{\varphi_1, \ldots, \varphi_n}_{=p_1,\ldots,p_n}, \varphi_{n+1}, \ldots, \varphi_{n+m}$.

Let $\mathfrak{A} = (Q, \mathbb{B}^n, q_0, \Delta, F_1, \ldots, F_k)$, where $Q = \mathbb{B}^{n+m} \cup \{q_0\}$ and k is the number of U-subformulas of φ. We have the following transitions in Δ (writing $\bar{b}$ for vectors in $\mathbb{B}^n$, $\bar{c}$ for vectors in $\mathbb{B}^m$:

$q_0 \xrightarrow{\bar{b}} \begin{pmatrix} \bar{b} \\ \bar{c} \end{pmatrix}$, with last component 1. This checks the truth of $\varphi = \varphi_{n+m}$.

$\begin{pmatrix} \bar{b} \\ \bar{c} \end{pmatrix} \xrightarrow{\bar{b}'} \begin{pmatrix} \bar{b}' \\ \bar{c}' \end{pmatrix}$ such that all local compatibility conditions are satisfied. We have nondeterminism in U-formula components.

For $\varphi_j = \varphi_{j_1} U \varphi_{j_2}$ introduce a generalized accepting component $F_j = \{ \begin{pmatrix} \bar{b} \\ \bar{c} \end{pmatrix}$: j-th component $= 0$ or j_2-th component $= 1\}$.

By construction, the run of $\mathfrak{A}$ on $\alpha \in (\mathbb{B}^n)^\omega$ is uniquely fixed if accepting and this run presents the φ-expansion of α. So $\mathfrak{A}$ accepts α iff $\alpha \models \varphi$.

Remark. If the answer to '$(\mathcal{M}, s) \models \varphi$?' is negative, an error scenario can be extracted from the nonemptiness of the intersection automaton $\mathfrak{A}_{\mathcal{M},s} \times \mathfrak{A}_{\neg\varphi} = \mathfrak{B}$.

The complexity of model-checking

Theorem 95. *The model-checking problem 'Given $(\mathcal{M}, s), \varphi$, does $(\mathcal{M}, s) \models \varphi$ hold?' is PSPACE complete.*

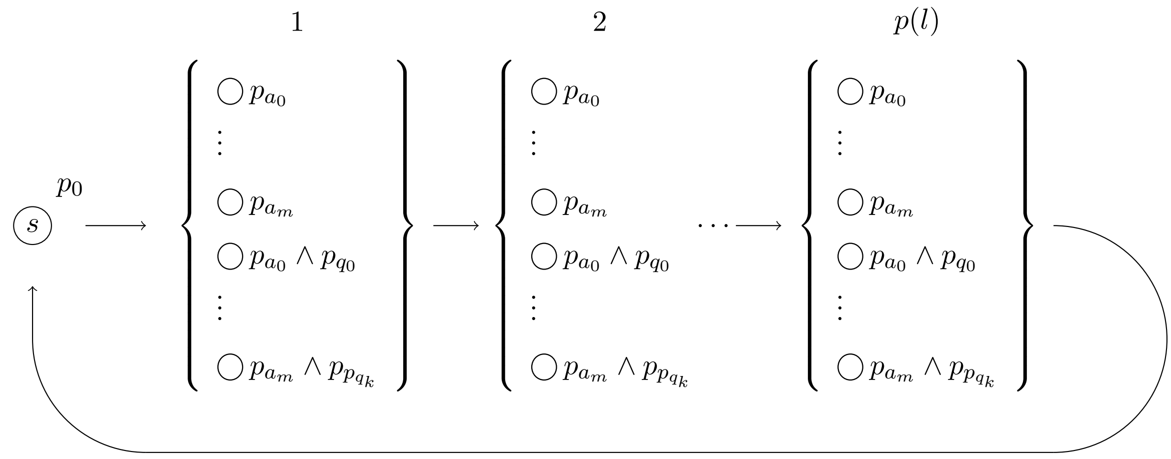
Proof Membership in PSPACE. Use space of size $2 \cdot (n + m)$ for guessing a path in $\mathfrak{A}_{\neg\varphi}$. The space of size $2(m + n)$ is used for noting two successive vertices of the guessed path.

PSPACE-completeness. The following problem is PSPACE-complete:

Given a polynomially space bounded TM M with a polynomial space bound p and an input word w, does M not accept w?

We need a PTIME transformation $(M, p, w) \mapsto ((\mathcal{M}, s), \varphi)$, such that M does not accept w iff $(\mathcal{M}, s) \models \varphi$. We assume that M works on a left-bounded tape and if it reaches an accepting configuration stays in this configuration. Let $M = (Q, \Gamma, q_0, \delta, F)$, $Q = \{q_0, \ldots, q_k\}$, $\Gamma = \{a_0 = \Box, a_1, \ldots, a_m\}$. A configuration is a word from $\Gamma^*(Q \times \Gamma)\Gamma^*$.

$(\mathcal{M}, s)$ is a Kripke structure over $p_0, p_{a_0}, \ldots, p_{a_m}, p_{q_0}, \ldots, p_{q_k}$. For M and w of length l, $\mathcal{M}$ is the following structure:



We define φ, expressing that the assumed path is a non-accepting computation of M on w. Then $(\mathcal{M}, s) \models \phi$ iff all paths are non-accepting, i.e. M does not accept w. Let $\varphi_Q := \bigvee_{q \in Q} p_q$.

$$\varphi = \neg(\varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4)$$

φ_1 describes that the path is a sequence of configurations:

$$\varphi_1 = G(p_0 \to X(\neg\varphi_Q U(\varphi_Q \wedge X(\neg\varphi_Q U p_0))))$$

φ_2 describes the initial configuration. For $w = w_1 \dots w_l$ this is $(q_0 w_1) w_2 \dots w_l \square \dots \square$.

$$\varphi_2 = X(p_{w_1} \wedge p_{q_0} \wedge X(p_{w_2} \wedge X(\dots X(p_{a_0} \wedge X(p_0)) \dots)))$$

φ_3 describes a visit of an F state.

$$\varphi_3 = F \bigvee_{q \in F} p_q$$

φ_4 enforces that successive path segments between p_0-visits represent successive M-configurations.

$$\begin{aligned}\varphi_4 = G \quad [\quad & \bigwedge_{a \in \Gamma} (p_a \wedge \neg \varphi_Q \to X^{p(l)+1} p_a) \wedge \\ & \bigwedge_{\delta(q,a)=(b,R,q')} (p_a \wedge p_q \to X^{p(l)+1}(p_b \wedge X p_{q'})) \wedge \\ & \text{(analogous for left moves and no move)} \\ & \bigwedge_{q \in F} (p_q \to X^{p(l)+1} p_q)]\end{aligned}$$

$\square$

16 Linear Arithmetic

Recall:

1. FO-theory of $(\mathbb{N}, +, \cdot, 0, 1)$ is undecidable
2. MSO-theory of $(\mathbb{N}, +1, 0)$ is decidable

Now consider the Presburger arithmetic: FO-theory of $(\mathbb{N}, +, 0)$. Main results:

1. FO-theory of $(\mathbb{N}, +, 0)$ is decidable
2. FO-theory of $(\mathbb{R}, \mathbb{Z}, +, <, 0)$ is decidable (applications in hybrid systems).

Definition 96. *Presburger arithmetic.*
Syntax:

- *variables:* $x, y, \ldots$
- *constant:* 0
- *terms:* $x, 0, x + y, y + 0, x + y + y, \ldots$
- *atomic formulas:* $t_1 = t_2$*, with terms* t_1, t_2
- *boolean connectives:* $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
- *quantifiers:* $\exists, \forall$

Semantics: standard interpretation over $(\mathbb{N}, +, 0)$

Some in $(\mathbb{N}, +, 0)$ definable relations:

- $x \leq y$ is definable by: $\varphi_{\leq}(x, y) = \exists z(x + z = y)$
- $x < y$ is definable by: $\varphi_{<}(x, y) = \exists z(\neg z = 0 \wedge x + z = y)$
- $x = 1$ is definable by: $\varphi_1(x) = 0 < x \wedge \neg\exists z(0 < z \wedge z < x)$
  (analogously: $x = 2, 3, 4, \ldots$ $x = k$ is definable by a formula $\varphi_k(x)$)

The linear equation $3x + 5y + 10 = s$ is definable by

$$\exists z(\varphi_{10} \wedge (x + x + x) + (y + y + y + y + y) + z = s)$$

Theorem 97. *Presburger arithmetic (the set of FO-sentences which are true in* $(\mathbb{N}, +, 0)$*) is decidable.*

Approach: Show that each relation $R \subseteq \mathbb{N}^n$ which is definable in Presburger arithmetic is definable by a finite automaton. Then apply decidability of emtptiness for automaton definable languages.

Define for $k \in \mathbb{N}$ the word $\hat{k}$:= reverse binary notation of k.

Example 98. Reverse binary notation for some numbers in $\mathbb{N}$.

$$\begin{aligned}\hat{3} &= 11\\ \hat{4} &= 001\\ \hat{0} &= 0\end{aligned}$$

$\boxtimes$

For a word tupel $(w_1, \ldots, w_n) \in (\{0,1\}^*)^n$ with $w_i = w_i(0), \ldots, w_i(l_i)$ define

$$[w_1, \ldots, w_n] = \begin{pmatrix} w_1(0) \\ \vdots \\ w_n(0) \end{pmatrix} \begin{pmatrix} w_1(1) \\ \vdots \\ w_n(1) \end{pmatrix} \cdots \begin{pmatrix} w_1(m) \\ \vdots \\ w_n(m) \end{pmatrix}$$

with $m = \max\{l_i \mid i = 1, \ldots, n\}$ where $w_i(j) = \$$ if $j > l_j$.

Example 99. Let $(w_1, w_2, w_3) = (11, 0, 0101)$ then

$$[w_1, w_2, w_3] = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ \$ \\ 1 \end{pmatrix} \begin{pmatrix} \$ \\ \$ \\ 0 \end{pmatrix} \begin{pmatrix} \$ \\ \$ \\ 1 \end{pmatrix}$$

$\boxtimes$

To each relation $R \subseteq \mathbb{N}^n$ associate the language

$$[R] = \{[\hat{k_1}, \ldots, \hat{k_n}] \mid (k_1, \ldots, k_n) \in R\}$$

Example 100. (continued) Let $R = \{(k, l, m) \mid 2k + l = m\}$. An example word in $[R]$:

$$[(\hat{3}, \hat{1}, \hat{7})] = (11, 1, 111) = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ \$ \\ 1 \end{pmatrix} \begin{pmatrix} \$ \\ \$ \\ 1 \end{pmatrix}$$

$\boxtimes$

For a formula $\varphi(x_1, \ldots, x_n)$ let $R_\varphi = \{(k_1, \ldots, k_n) \mid (\mathbb{N}, +, 0) \models \varphi(k_1, \ldots, k_n)\}$

Lemma 101. *For each Presburger formula φ the language $[R_\varphi]$ is regular, and one can construct from φ a finite automaton for it.*

Proof (by induction). Atomic formulas are of the form $x + y = z$ (and possibly zeros for x,y or z).

Example 102. To get this simple form of atomic formulas, observe that we can write each term in the desired format:

$$x + x + x = y \rightsquigarrow \exists z(x + x = z \wedge z + x = y)$$

$$x_1 + x_2 = y_1 + y_2 \rightsquigarrow \exists z_1(x_1 + x_2 = z_1 \wedge y_1 + y_2 = z_1)$$

$\boxtimes$

Basis:
Check of "$x + y = z$" on a word $[(\hat{k_1}, \hat{k_2}, \hat{k_3})]$.
A finite automaton uses two states to store the carry (0 or 1) and the finite table of correct summations. It has also to check for each component that after the first occurence of \$ only \$ occur.

Induction Step: We only have to treat the connectives $\vee, \neg, \exists$.

$\wedge$: Clear, using the union of regular languages

$\neg$: Clear by complementation relative to the words where \$ appears correctly (regular set)

$\exists$: Given a relation $R \in \mathbb{N}^{n+1}$, defined by $\varphi(x_1, \dots, x_{n+1})$, and the corresponding automaton $\mathfrak{A}_\varphi$ over $(\{0, 1, \$\}^*)^{n+1}$. Find an automaton for $S = \{(k_1, \dots, k_n) \mid \exists k(k_1, \dots, k_n, k) \in R\}$. For this proceed as for monadic logic (i.e. delete $(n+1)$-st component in the labels of the transitions) and declare each state final from where a final state can be reached only by completely \$-marked transitions.

$\square$

Theorem 97 follows by applying the lemma for the case $n = 0$:
Check wether the constructed automaton with unlabelled transitions has an accepting run.

Remark.

1. Finite automata are stronger than Presburger formulas. The relation "is a power of 2" is definable via binary notation, but not definable in Presburger arithmetic.
   Let $Pow_2 = \{k \mid k \textit{ is power of 2}\}$. It follows that the FO-theory of $(\mathbb{N}, +, 0, Pow_2)$ is decidable.

2. A Relation R is definable in Presburger arithmetic iff it is semilinear.

3. A structure $\underline{S} = (S, R_1, \dots, R_k)$ with $R_i \subseteq S^{n_i}$ is called automatic if for some auxiliary alphabet Γ, where $S \subseteq \Gamma^*$ is regular, the S-elements are coded by words over Γ and the relations R_i (coded over $\Gamma \cup \{\$\}$ as above) are all recognizable by finite automata. Then we have the following:

$$\underline{S} \text{ automatic } \Rightarrow \text{ FO-Th}(\underline{S}) \text{ is decidable}$$

We give a sketch about the significance of FO-Th($\mathbb{R}, \mathbb{Z}, <, +, 0$) and that this theory is decidable.

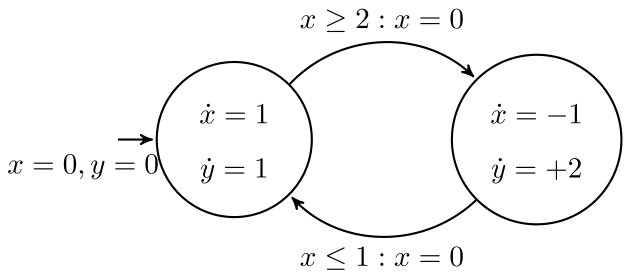
Example 103. Assume $\varphi(x, y)$ defines the function $f : \mathbb{R} \to \mathbb{R}$. In the language of FO-Th($\mathbb{R}, \mathbb{Z}, <, +, 0$) we can express that f is continuous at x:

$$\forall \epsilon > 0 \exists \delta > 0 \forall x' (|x - x'| < \delta \to |f(x) - f(x')| < \epsilon)$$

⊠

An application domain is given by hybrid systems. They are composed of discretely and continuously working components.

Example 104. The depicted hybrid system has two discrete states and two value functions $x(t)$ and $y(t)$. Transistions are executed subject to constraints and reset variables. (Timed automata use only functions $x(t)$ with $\dot{x} = 1$)



⊠

The analysis of reachability of "configurations" requires the study of sets of configurations. A configuration for systems with variables $x_1, \ldots, x_n$, is given by a discrete state q, and values $r_1, \ldots, r_n$ where $r_i \in \mathbb{R}$.
For applications in the analysis of hybrid systems, we need an efficient way to transform linear constraints and check their satisfiability.

Method: We work with formulas $\varphi(x_1, \ldots, x_n)$ and transform/check them by the use of automata.

Idea: Code a real number $r \geq 0$ by binary expansion of the form $w * \alpha$ (w codes the integer part, α codes the fraction part) where $w \in \{0,1\}^*$, $\alpha \in \{0,1\}^\omega$.

Problem: The number 3.5 has two representations:

$$11 * 100\ldots$$
$$11 * 011\ldots$$

Technical Remark: We need to consider only positive numbers for coefficients and constants. For this, use transformations as in this example:

$$x - 3z \leq -7 \rightsquigarrow x + 7 \leq 3z$$

Now one can repeat the idea of Presburger arithmetic analysis, however using ω-automata. A first approach works by the use of nondeterministic Büchi automata.

Lemma 105. *Let $R \subseteq \mathbb{R}^n$.*

$$R \text{ is FO-definable } \Rightarrow [R] \text{Büchi automata recognizable}$$

Problem: This procedure is not efficient by repeated use of projection and negation.

Solution:

1. Weak automata suffice (variant of Staiger-Wagner automata) (Boigelot, Jodogne, Wolper)
2. Deterministic weak automata can be minimized efficiently (Löding)

One can use weak Büchi automata where each SCC consists only of final or non-final states.

Part V
Outlook

17 Cantor Space and the Borel Hierarchy

We discuss the Classification of ω-languages in set theory.

G. Cantor considered arbitrary sets of real numbers.
Cardinalities: $|\mathbb{N}| = |\mathbb{Q}| < |\mathbb{R}|$
Continuum hypothesis: There is no set $M \subseteq \mathbb{R}$ such that

$$|\mathbb{N}| < |M| < |\mathbb{R}|$$

Axiomatic system of set theory: ZFC (Zermelo-Fraenkel set theory, with axiom of choice)

Cantor classified sets $M \subseteq \mathbb{R}$ according to "complexity" and showed that for "simple" sets M the following holds:

$$|M| = |\mathbb{N}| \text{ or } |M| = |\mathbb{R}|$$

Simple sets are the "open sets". Here we discuss the space $\{0,1\}^\omega$ rather than $\mathbb{R}$.

Definition 106. Open sets.

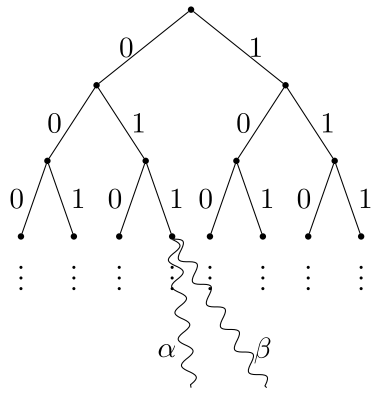
- *Over $\mathbb{R}$:*

$$M \subseteq \mathbb{R} \textit{ is open } \Leftrightarrow M \textit{ is a union of } \epsilon\textit{-neighbourhoods } (x-\epsilon, x+\epsilon)$$

- *Over $\{0,1\}^\omega$ a metric is introduced:*

$$d(\alpha,\beta) = \begin{cases} 0 & \alpha = \beta \\ \frac{1}{2^i} & \textit{for the smallest } i \textit{ with } \alpha(i) \neq \beta(i) \end{cases}$$

Illustration:



Remark. For $\alpha \neq \beta$ we have:

$$d(\alpha,\beta) \geq \frac{1}{2^i} \Leftrightarrow \text{ exists } j \leq i : \alpha(j) \neq \beta(j)$$
$$\text{So we have } d(\alpha,\beta) < \frac{1}{2^i} \Leftrightarrow \text{ for all } j \leq i : \alpha(j) = \beta(j)$$

Hence the $\frac{1}{2^i}$-neighbourhood of α is the set $N_{\frac{1}{2^i}}(\alpha) = \{\beta \mid \beta(0),\ldots,\beta(i) = \alpha(0),\ldots,\alpha(i)\}$

Definition 107. *$L \subseteq \{0,1\}^\omega$ is open $\Leftrightarrow L = W \cdot \{0,1\}^\omega$ for some $W \subseteq \{0,1\}^*$*

A *basic open set* is of the form $\{w\} \cdot \Sigma^\omega$. A set $L \subseteq \Sigma^\omega$ is *closed* iff $\Sigma^\omega \backslash L$ is open. For example $\{0^\omega\}$ is a closed set.

Remark. L is closed iff there exists $V \subseteq \Sigma^*$ such that $\alpha \in L \Leftrightarrow$ each finite α-prefix is in V.

Proof Consider L closed, i.e. $\overline{L}$ open. $\overline{L} = W \cdot \Sigma^\omega$ for some set $W \in \Sigma^*$. W.l.o.g. W is closed under suffixes (i.e. if $w \in W$ and w is a prefix of w' then $w' \in W$). Define $V := \Sigma^* \backslash W$. Then $\alpha \in L \Leftrightarrow$ each prefix of α is in V.
Conversely let $V \in \Sigma^*$ such that $\alpha \in L$ iff each finite α-prefix is in V. Take $W := \Sigma^* \backslash V$, then the open set $W \cdot \Sigma^\omega$ is the complement of L. □

Remark (Path Lemma). Let L be closed and α an ω-word. If each α-prefix can be extended to some $\beta \in L$ then $\alpha \in L$.

Proof Consider V such that $\alpha \in L \Leftrightarrow$ all α-prefixes are in V. V is closed under prefixes. Thus all α-prefixes are in V, which (by the remark) implies $\alpha \in L$. □

Definition 108. *L is clopen iff L is open and closed.*

Remark. L is clopen iff $L = W \cdot \Sigma^\omega$ with finite W.

Proof Let $L = W \cdot \Sigma^\omega$ with finite W. Assume w.l.o.g. that for all $w, w' \in W$ w is not a prefix of w' and vice versa. We need to find V such that $\overline{L} = V \cdot \Sigma^\omega$. Let $l := \max\{|w| : w \in W\}$. A suitable V is $\{v \in \Sigma^* : v \text{ has no } W\text{-word as prefix}, |v| = l\}$.
Conversely let L be clopen. Take V, W such that $L = W \cdot \Sigma^\omega$, $\overline{L} = V \cdot \Sigma^\omega$. Each path has a prefix in W or in V. Let $U := \{u : u \text{ has neither a } W\text{- nor a } V\text{-prefix}\}$. We show that U is finite. Then minimal words in $W \cup V$ give finite sets. Assume U is infinite, then U gives an infinite finitely branching tree. by König's Lemma there is an infinite path, a contradiction to the fact that each path has to meet V or W. □

The use of this classification in computer science comes from the following intuition. Open ω-languages capture guarantee properties, closed ω-languages capture safety properties and clopen ω-languages capture what is called bounded properties. They form the lowest stages of a hierarchy, which we introduce shortly.

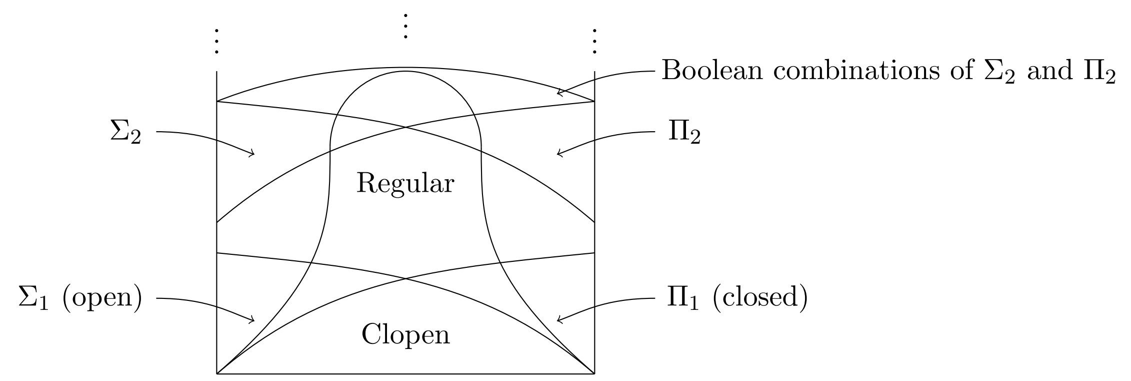
Remark. Open sets are closed under union and intersection. For example we have $W_1\Sigma^\omega \cup W_2 \cdot \Sigma^\omega = (W_1 \cup W_2) \cdot \Sigma^\omega$.

Open sets are closed under countable unions, namely, if $L_i = W_i \cdot \Sigma^\omega$ then $\bigcup L_i = (\bigcup W_i) \cdot \Sigma^\omega$. But one can show that open sets are not closed under countable intersections.

Definition 109 (Borel Hierarchy). *We introduce classes Σ_i, Π_i of ω-languages for $i \geq 1$.*

- Σ_1 *is the class of open ω-languages,* Π_1 *is the class of closed ω-languages.*
- Σ_{i+1} *is the class of countable unions of ω-languages in* Π_i.
- Π_{i+1} *is the class of countable intersections of ω-languages in* Σ_i.

Illustration for Π_2: $L \in \Pi_2 \Leftrightarrow L$ is representable as a union $\bigcup_{i\geq 0} W_i \cdot \Sigma^\omega$.



The sequence $\Sigma_1, \Sigma_2, \ldots$ gives an infinite hierarchy; its structure is indicated in the figure.

Remark. $L \in \Pi_2 \Leftrightarrow L = \lim W$ for some $W \in \Sigma^*$.

Special case: If we assume W regular we get the deterministic Büchi recognizable ω-languages. Boolean combinations give the Muller recognizable sets. As a consequence, we obtain that the regular ω-languages are all included in the boolean closure of Σ_2 and Π_2 (and as one can show included in $\Sigma_3 \cap \Pi_3$).

18 Perspective: Infinite Games

We give an outlook to next semester's course on infinite games, where the theory of ω-automata finds another application. We consider infinite two-player games where two players build up ω-words by picking letters in alternation. If player 1 picks a_i in his i-th move and player 2 picks b_i in his i-th move then the play $a_0 b_0 a_1 b_1 \ldots$ is produced. By this, ω-words are built up bit by bit. A winning condition is captured by an ω-language L: player 2 wins a play α iff $\alpha \in L$. The games are thus classified by the ω-languages defining their winning conditions (Borel games, regular games, ...). The task is to check whether player 2 has a winning strategy (which guarantees that any play compatible with this strategy will be in L) and if so, to construct such a strategy. A fundamental theorem of Martin shows that from each position in a Borel game, one of the players has a winning strategy. This may require infinite memory though. Büchi and Landweber showed that for regular games (given by regular ω-languages as winning conditions) one can efficiently determine which player wins and construct automaton winning strategies for the winner.

Example 110. Player 1 chooses letters A, B, C, D, player 2 chooses numbers $1, 2, 3, 4$. Player 2 wins a play if the number of infinitely often chosen letters is the highest number chosen infinitely often. A winning strategy for player 2 uses the well known LVR. Player 2 chooses the number by the index of the underlined letter:

Player 1	LVR	Response player 2
A	$\underline{A}BCD$	1
C	$CA\underline{B}D$	3
B	$BC\underline{A}D$	3
C	$C\underline{B}AD$	2
A	$AC\underline{B}D$	3
D	$DAC\underline{B}$	4
C	$CD\underline{A}B$	4
C	$\underline{C}DAB$	1
D	$D\underline{C}AB$	2
C	$C\underline{D}AB$	2

⊠