Hierarchies of Abstract Machines

Jérémie Cabessa

University Joseph Fourier – Grenoble 1

26 March 2009

Outline

Outline

Outline

Outline

Outline

We assume that specific preferred firing sequences are tightly related to storage and processing of information in the brain.

We aim to study the computational capability of neural nets in terms of the complexity of preferred firing sequences involved in the information processing.

We assume that specific preferred firing sequences are tightly related to storage and processing of information in the brain.

We aim to study the computational capability of neural nets in terms of the complexity of preferred firing sequences involved in the information processing.

We adopt an automata theoretical approach (McCulloch and Pitts (1943), Kleene (1956), Minsky (1967)).

> *[Minsky (1967)] "It is evident that each neural network of the kind we have been considering is a finite state machine. [. . .] It is interesting and surprising that there is a converse to this. [. . .] Every finite state machine is equivalent to, and can be "simulated" by, some neural net."*

We have an equivalence between simple neural nets and finite state machines; But there is no classification of FSM's according to their computational capability.

We adopt an automata theoretical approach (McCulloch and Pitts (1943), Kleene (1956), Minsky (1967)).

> *[Minsky (1967)] "It is evident that each neural network of the kind we have been considering is a finite state machine. [. . .] It is interesting and surprising that there is a converse to this. [. . .] Every finite state machine is equivalent to, and can be "simulated" by, some neural net."*

We have an equivalence between simple neural nets and finite state machines; But there is no classification of FSM's according to their computational capability.

We adopt an automata theoretical approach (McCulloch and Pitts (1943), Kleene (1956), Minsky (1967)).

> *[Minsky (1967)] "It is evident that each neural network of the kind we have been considering is a finite state machine. [. . .] It is interesting and surprising that there is a converse to this. [. . .] Every finite state machine is equivalent to, and can be "simulated" by, some neural net."*

We have an equivalence between simple neural nets and finite state machines; But there is no classification of FSM's according to their computational capability.

But there exist relevant classifications of FSM's over infinite inputs.

Therefore we aim to find an equivalence between simple neural nets and specific FSM's over infinite inputs.

But there exist relevant classifications of FSM's over infinite inputs.

Therefore we aim to find an equivalence between simple neural nets and specific FSM's over infinite inputs.

Muller automata

- Muller automata are abstract machines working on infinite inputs.
- These kind of machines (over infinite words) can be classified in a very refined manner according to their computational capability.

Muller automata

- Muller automata are abstract machines working on infinite inputs.
- These kind of machines (over infinite words) can be classified in a very refined manner according to their computational capability.

Muller automata

- Muller automata are abstract machines working on infinite inputs.
- These kind of machines (over infinite words) can be classified in a very refined manner according to their computational capability.

Muller automaton (deterministic)

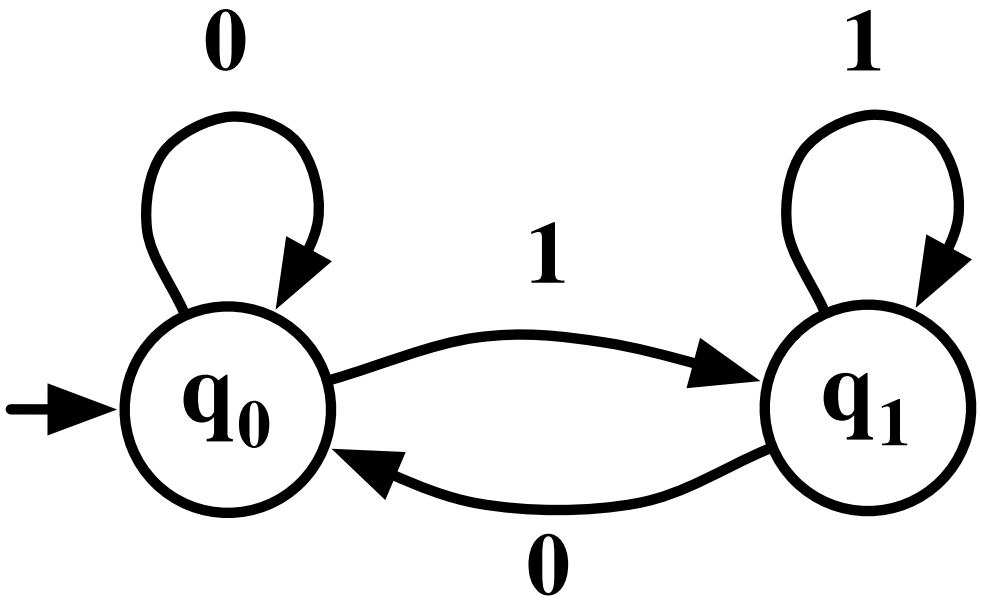


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Muller automaton (deterministic)

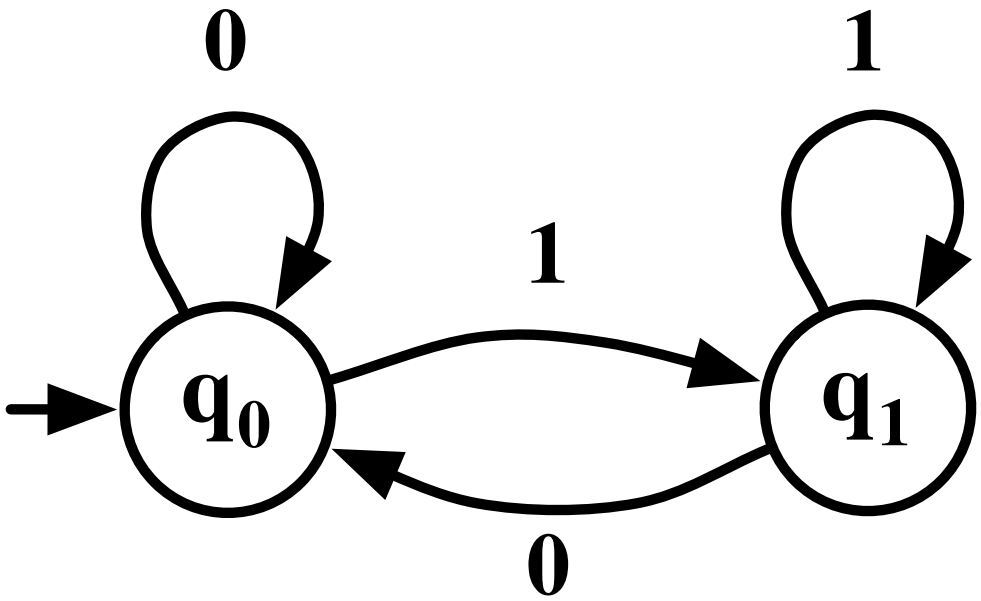


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

input 001111111111 . . . . . accepted

input 110000000000 . . . . . rejected

input 010101010101 . . . . . accepted

Muller automaton (deterministic)

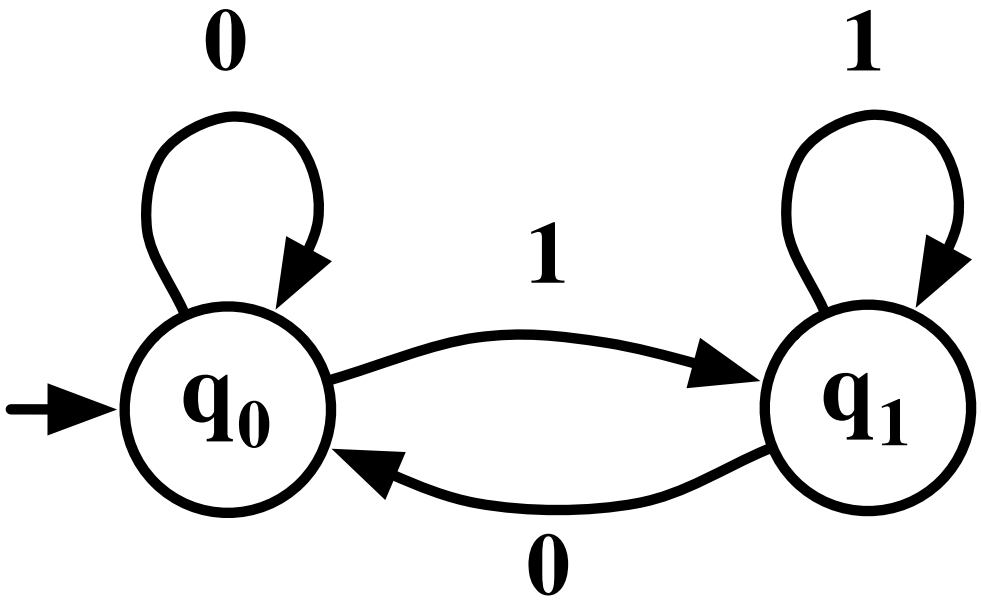


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

input 001111111111 . . . . . **accepted**

input 110000000000 . . . . . **rejected**

input 010101010101 . . . . . **accepted**

Muller automaton (deterministic)

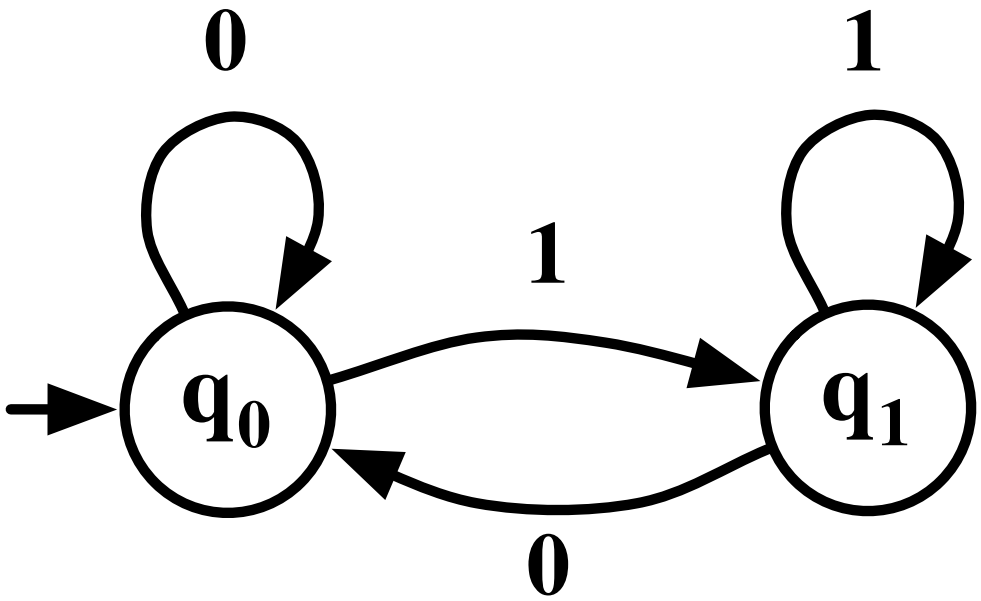


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

input	001111111111.....	**accepted**
input	110000000000.....	**rejected**
input	010101010101.....	**accepted**

Muller automaton (deterministic)

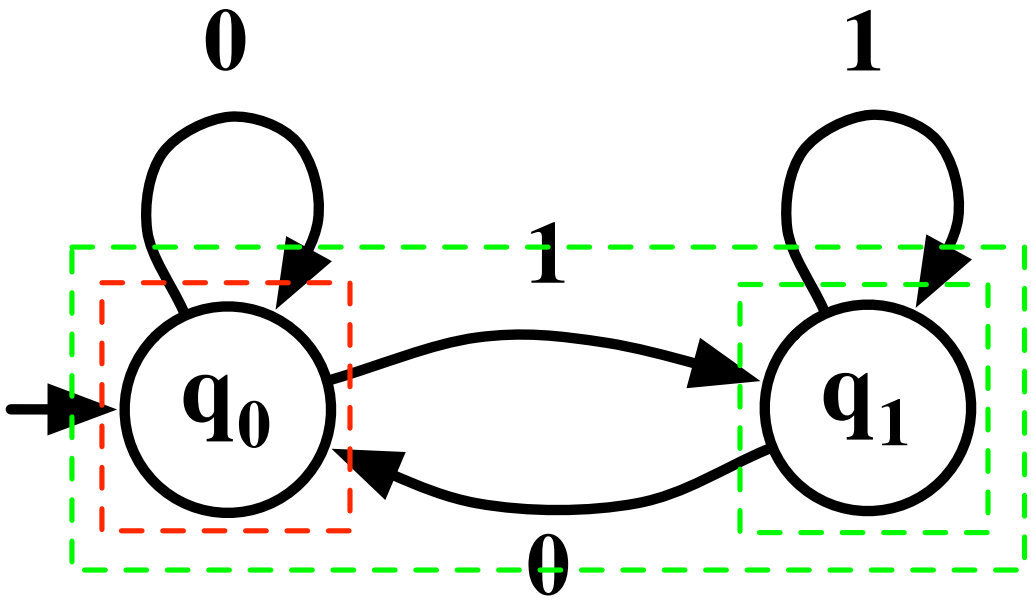


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

input	001111111111.....	**accepted**
input	110000000000.....	**rejected**
input	010101010101.....	**accepted**

Muller automaton (deterministic)

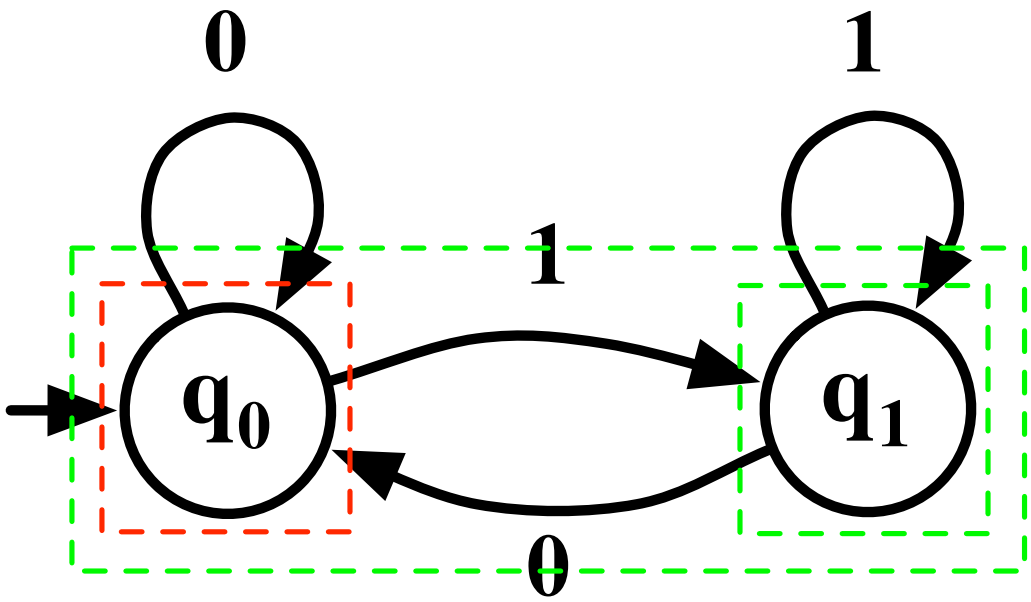


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

The accepted words are exactly those containing infinitely many 1's. This is the *language recognized by this automaton.*

Definition

A finite *Muller automaton* is a tuple $\mathcal{A} = (Q, A, \delta, i, \mathcal{T})$, where

- Q is a finite set of states,
- A is an alphabet,
- $\delta : Q \times A \longrightarrow Q$ is the transition function,
- $i \in Q$ is the initial state,
- $\mathcal{T} \subseteq \mathcal{P}(Q)$ is the table of $\mathcal{A}$.

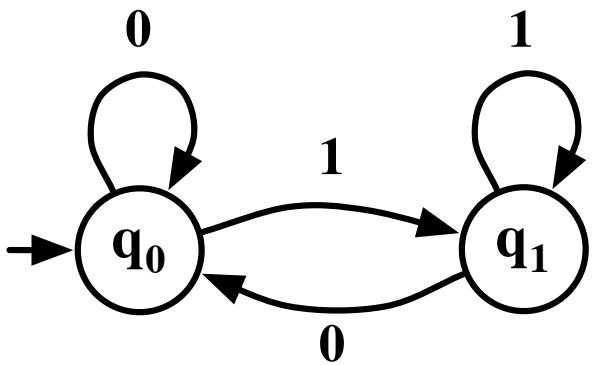


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Definition

A finite *Muller automaton* is a tuple $\mathcal{A} = (Q, A, \delta, i, \mathcal{T})$, where

- Q is a finite set of states,
- A is an alphabet,
- $\delta : Q \times A \longrightarrow Q$ is the transition function,
- $i \in Q$ is the initial state,
- $\mathcal{T} \subseteq \mathcal{P}(Q)$ is the table of $\mathcal{A}$.

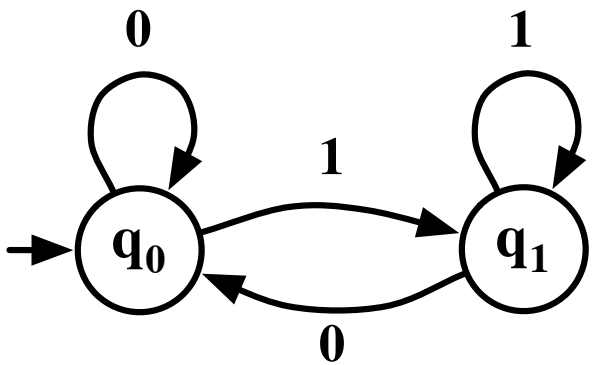


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Definition

A finite *Muller automaton* is a tuple $\mathcal{A} = (Q, A, \delta, i, \mathcal{T})$, where

- Q is a finite set of states,
- A is an alphabet,
- $\delta : Q \times A \longrightarrow Q$ is the transition function,
- $i \in Q$ is the initial state,
- $\mathcal{T} \subseteq \mathcal{P}(Q)$ is the table of $\mathcal{A}$.

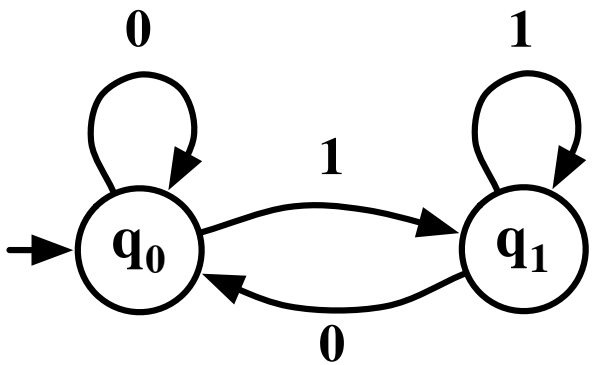


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Definition

A finite *Muller automaton* is a tuple $\mathcal{A} = (Q, A, \delta, i, \mathcal{T})$, where

- Q is a finite set of states,
- A is an alphabet,
- $\delta : Q \times A \longrightarrow Q$ is the transition function,
- $i \in Q$ is the initial state,
- $\mathcal{T} \subseteq \mathcal{P}(Q)$ is the table of $\mathcal{A}$.

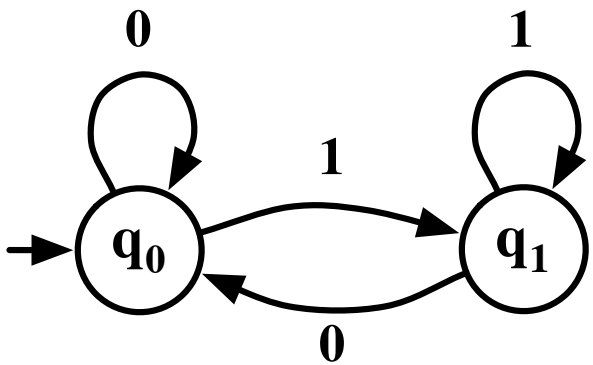


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Definition

A finite *Muller automaton* is a tuple $\mathcal{A} = (Q, A, \delta, i, \mathcal{T})$, where

- Q is a finite set of states,
- A is an alphabet,
- $\delta : Q \times A \longrightarrow Q$ is the transition function,
- $i \in Q$ is the initial state,
- $\mathcal{T} \subseteq \mathcal{P}(Q)$ is the table of $\mathcal{A}$.

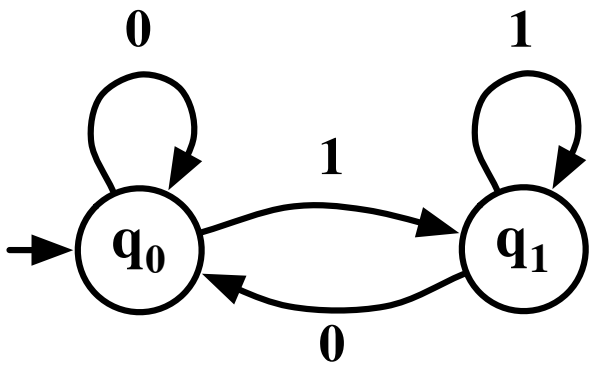


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Definition

A finite *Muller automaton* is a tuple $\mathcal{A} = (Q, A, \delta, i, \mathcal{T})$, where

- Q is a finite set of states,
- A is an alphabet,
- $\delta : Q \times A \longrightarrow Q$ is the transition function,
- $i \in Q$ is the initial state,
- $\mathcal{T} \subseteq \mathcal{P}(Q)$ is the table of $\mathcal{A}$.

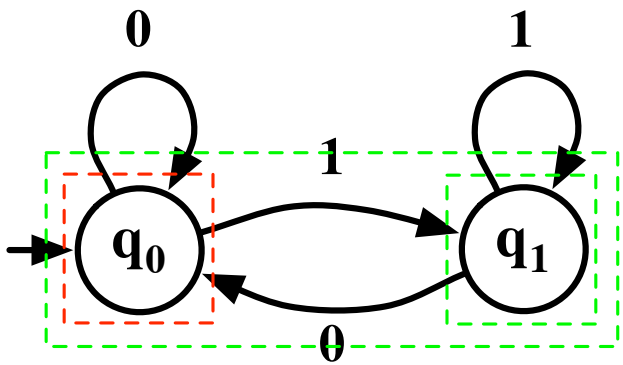


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

An infinite input w induces an infinite path in $\mathcal{A}$ ... An input w is *accepted* by $\mathcal{A}$ iff $States_\infty(w) \in \mathcal{T}$.

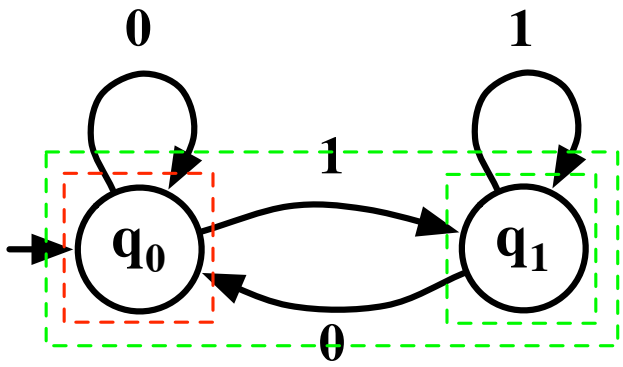


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Definition

The set $L(\mathcal{A})$ of all infinite words accepted by $\mathcal{A}$ is called the *language recognized by* $\mathcal{A}$.

An infinite input w induces an infinite path in $\mathcal{A}$ ... An input w is *accepted* by $\mathcal{A}$ iff $States_\infty(w) \in \mathcal{T}$.

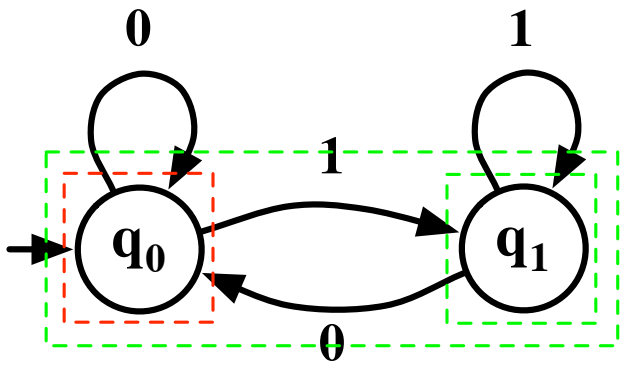


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Definition

The set $L(\mathcal{A})$ of all infinite words accepted by $\mathcal{A}$ is called the *language recognized by* $\mathcal{A}$.

An infinite input w induces an infinite path in $\mathcal{A}$ ... An input w is *accepted* by $\mathcal{A}$ iff $States_\infty(w) \in \mathcal{T}$.

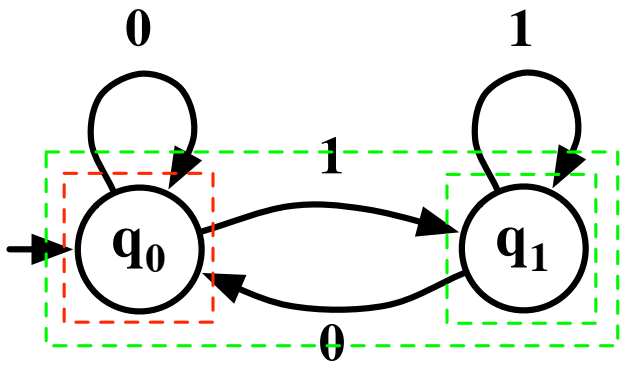


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

Definition

The set $L(\mathcal{A})$ of all infinite words accepted by $\mathcal{A}$ is called the *language recognized by* $\mathcal{A}$.

Example

Consider the following automaton $\mathcal{A}$. Then $L(\mathcal{A})$ is the set of words containing infinitely many 1's.

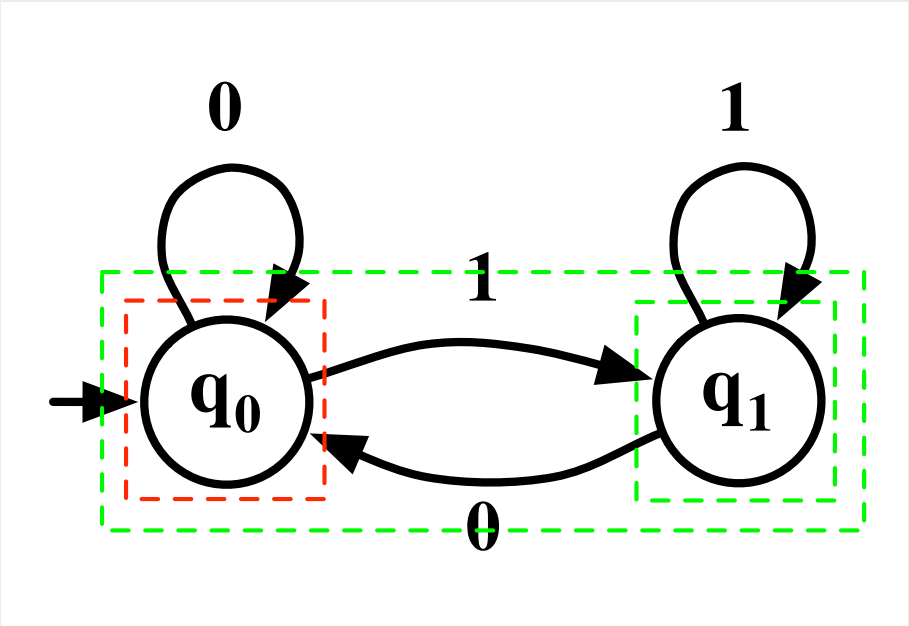
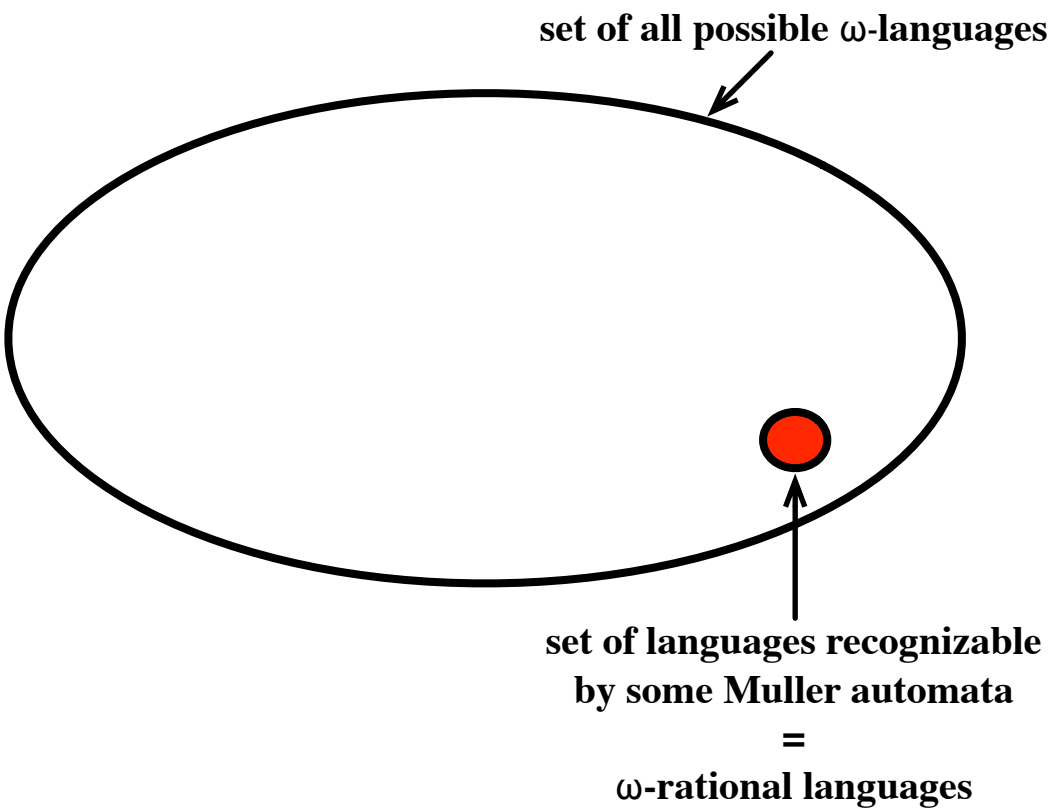


table $\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$

From now on, we will generally identify a Muller automaton $\mathcal{A}$ with its corresponding ω-language $L(\mathcal{A})$.

From now on, we will generally identify a Muller automaton $\mathcal{A}$ with its corresponding ω-language $L(\mathcal{A})$.



The Wagner hierarchy

- A very refined topological classification of ω-rational languages;
- or equivalently, a very refined classification of Muller automata.

The Wagner hierarchy

- A very refined topological classification of ω-rational languages;
- or equivalently, a very refined classification of Muller automata.

The Wagner hierarchy

- A very refined topological classification of ω-rational languages;
- or equivalently, a very refined classification of Muller automata.

Every classification comes from a partial order ($\leq$) between the objects that we consider. We will define a specific partial order $\leq_W$ between Muller automata.

Every classification comes from a partial order ($\leq$) between the objects that we consider. We will define a specific partial order $\leq_W$ between Muller automata.

Definition (alt. def. of the same reduction relation)

We say that $\mathcal{A} \leq_W \mathcal{B}$ iff there exists a continuous function f such that $L(\mathcal{A}) = f^{-1}(L(\mathcal{B}))$.

- We set $\mathcal{A} <_W \mathcal{B}$ if both $\mathcal{A} \leq_W \mathcal{B}$ and $\mathcal{B} \not\leq_W \mathcal{A}$.
- We set $\mathcal{A} \equiv_W \mathcal{B}$ if both $\mathcal{A} \leq_W \mathcal{B}$ and $\mathcal{B} \leq_W \mathcal{A}$.
- We set $\mathcal{A} \not\equiv_W \mathcal{B}$ if both $\mathcal{A} \not\leq_W \mathcal{B}$ and $\mathcal{B} \not\leq_W \mathcal{A}$.

Definition (alt. def. of the same reduction relation)

We say that $\mathcal{A} \leq_W \mathcal{B}$ iff there exists a continuous function f such that $L(\mathcal{A}) = f^{-1}(L(\mathcal{B}))$.

- We set $\mathcal{A} <_W \mathcal{B}$ if both $\mathcal{A} \leq_W \mathcal{B}$ and $\mathcal{B} \not\leq_W \mathcal{A}$.
- We set $\mathcal{A} \equiv_W \mathcal{B}$ if both $\mathcal{A} \leq_W \mathcal{B}$ and $\mathcal{B} \leq_W \mathcal{A}$.
- We set $\mathcal{A} \not\equiv_W \mathcal{B}$ if both $\mathcal{A} \not\leq_W \mathcal{B}$ and $\mathcal{B} \not\leq_W \mathcal{A}$.

Definition (alt. def. of the same reduction relation)

We say that $\mathcal{A} \leq_W \mathcal{B}$ iff there exists a continuous function f such that $L(\mathcal{A}) = f^{-1}(L(\mathcal{B}))$.

- We set $\mathcal{A} <_W \mathcal{B}$ if both $\mathcal{A} \leq_W \mathcal{B}$ and $\mathcal{B} \not\leq_W \mathcal{A}$.
- We set $\mathcal{A} \equiv_W \mathcal{B}$ if both $\mathcal{A} \leq_W \mathcal{B}$ and $\mathcal{B} \leq_W \mathcal{A}$.
- We set $\mathcal{A} \not\equiv_W \mathcal{B}$ if both $\mathcal{A} \not\leq_W \mathcal{B}$ and $\mathcal{B} \not\leq_W \mathcal{A}$.

Definition (alt. def. of the same reduction relation)

We say that $\mathcal{A} \leq_W \mathcal{B}$ iff there exists a continuous function f such that $L(\mathcal{A}) = f^{-1}(L(\mathcal{B}))$.

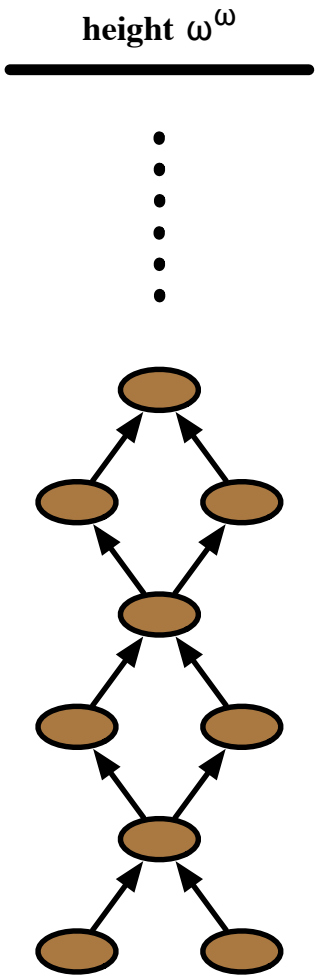
- We set $\mathcal{A} <_W \mathcal{B}$ if both $\mathcal{A} \leq_W \mathcal{B}$ and $\mathcal{B} \not\leq_W \mathcal{A}$.
- We set $\mathcal{A} \equiv_W \mathcal{B}$ if both $\mathcal{A} \leq_W \mathcal{B}$ and $\mathcal{B} \leq_W \mathcal{A}$.
- We set $\mathcal{A} \not\equiv_W \mathcal{B}$ if both $\mathcal{A} \not\leq_W \mathcal{B}$ and $\mathcal{B} \not\leq_W \mathcal{A}$.

Definition

The collection of all Muller automata (or ω-rational languages) ordered by the relation $\leq_W$ is called the *Wagner hierarchy*.

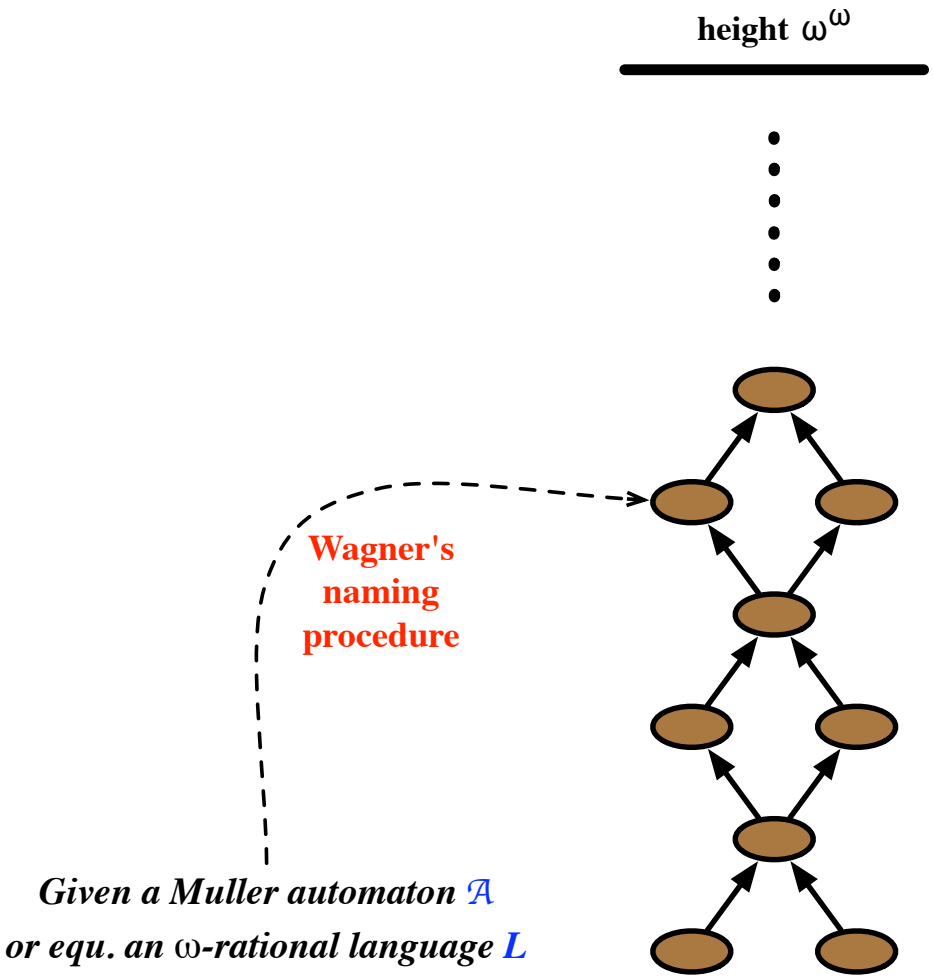
Theorem

The Wagner hierarchy has width 2 and height ω^ω.



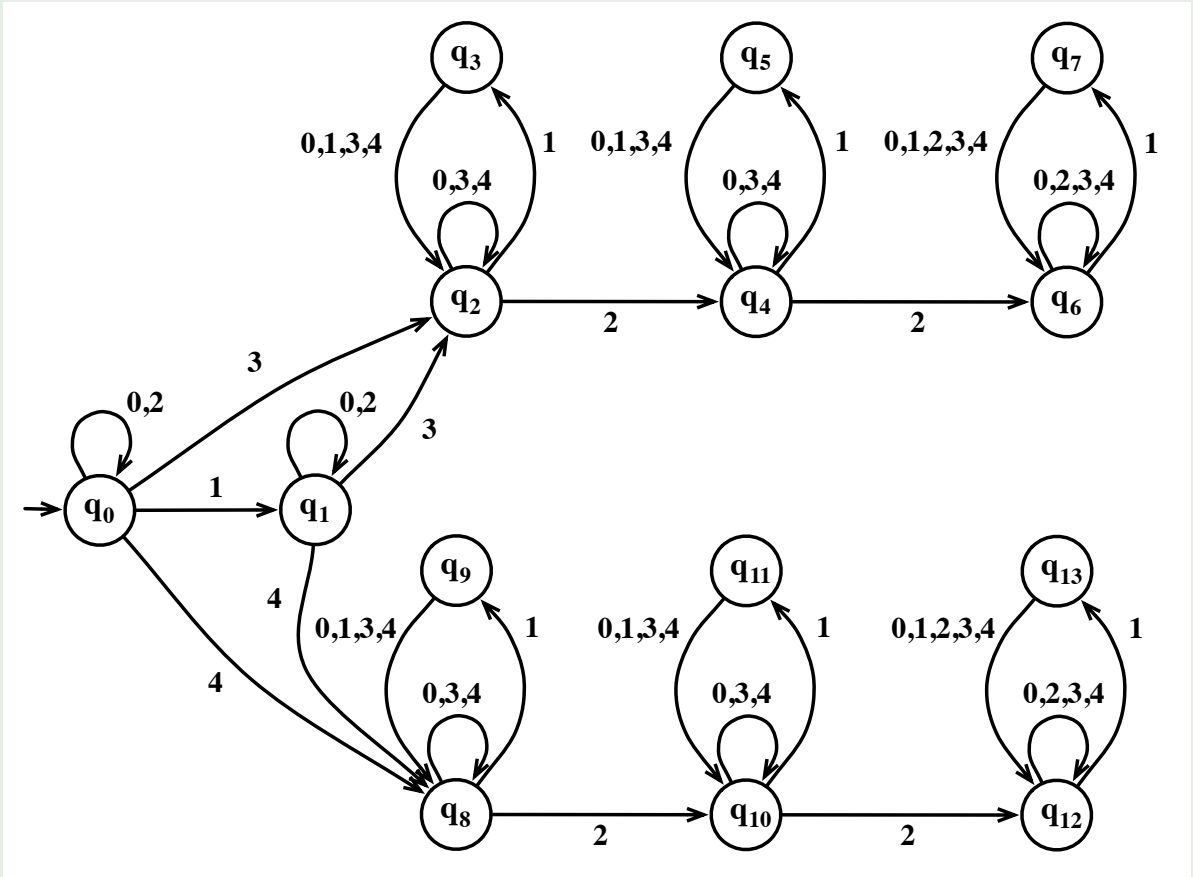
Theorem

The Wagner hierarchy is decidable.



Example

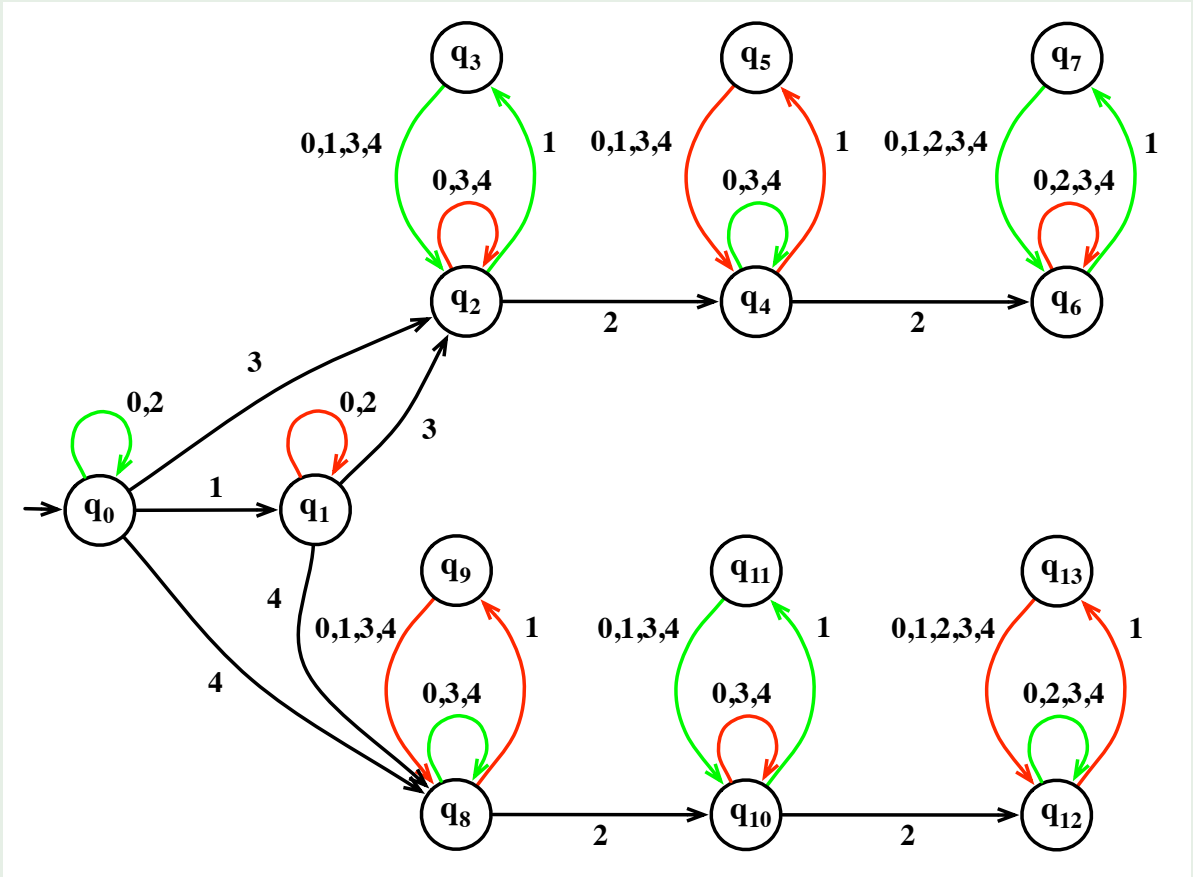
The following Muller automata has height $\omega \cdot 3 + 2$ in the Wagner hierarchy.



$$\mathcal{T} = \{\{q_0\}, \{q_2, q_3\}, \{q_4\}, \{q_6, q_7\}, \{q_8\}, \{q_{10}, q_{11}\}, \{q_{12}\}\}$$

Example

The following Muller automata has height $\omega \cdot 3 + 2$ in the Wagner hierarchy.



$$\mathcal{T} = \{\{q_0\}, \{q_2, q_3\}, \{q_4\}, \{q_6, q_7\}, \{q_8\}, \{q_{10}, q_{11}\}, \{q_{12}\}\}$$

Main kind abstract Machines reading infinite words:

- Deterministic Muller automata
- Deterministic Muller Counter automata
- Deterministic Muller Pushdown automata
- Deterministic Muller Turing machines

Main kind abstract Machines reading infinite words:

- Deterministic Muller automata
- Deterministic Muller Counter automata
- Deterministic Muller Pushdown automata
- Deterministic Muller Turing machines

Main kind abstract Machines reading infinite words:

- Deterministic Muller automata
- Deterministic Muller Counter automata
- Deterministic Muller Pushdown automata
- Deterministic Muller Turing machines

Main kind abstract Machines reading infinite words:

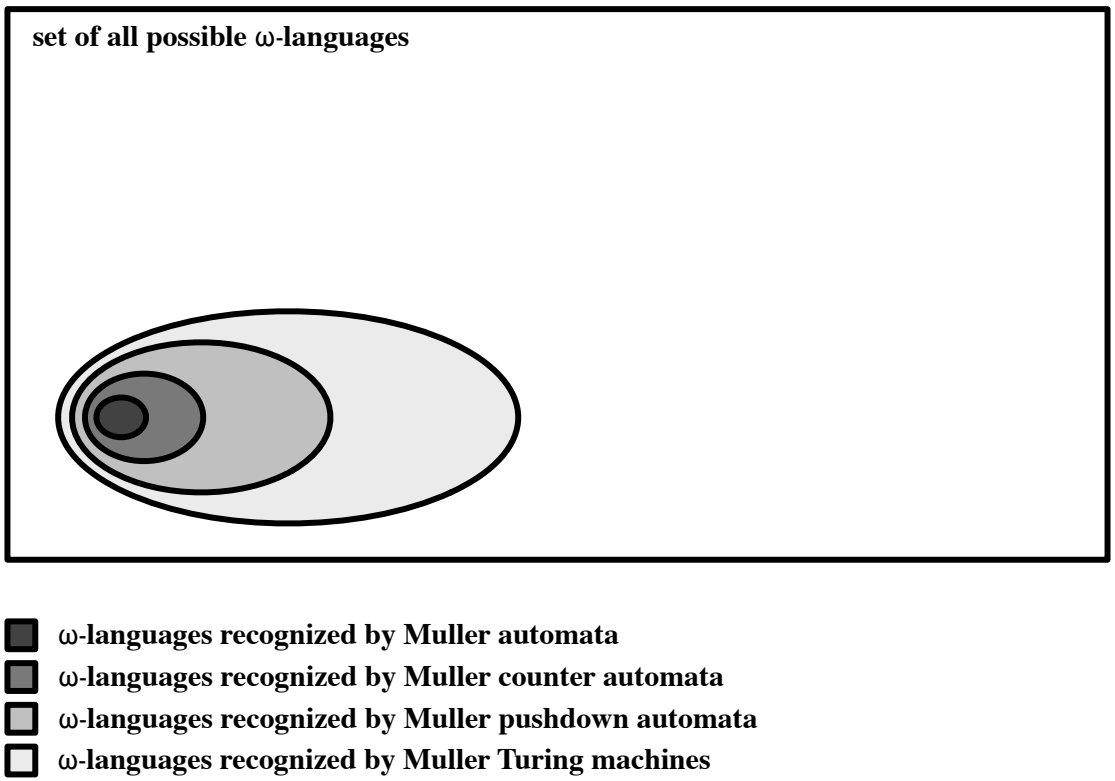
- Deterministic Muller automata
- Deterministic Muller Counter automata
- Deterministic Muller Pushdown automata
- Deterministic Muller Turing machines

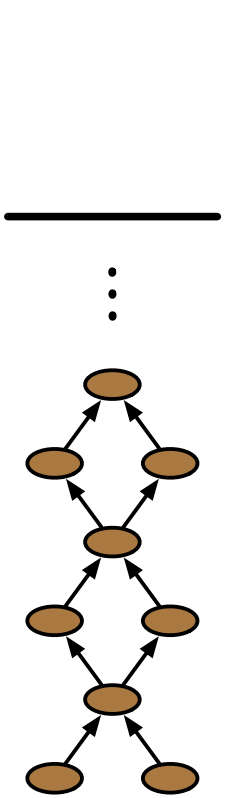
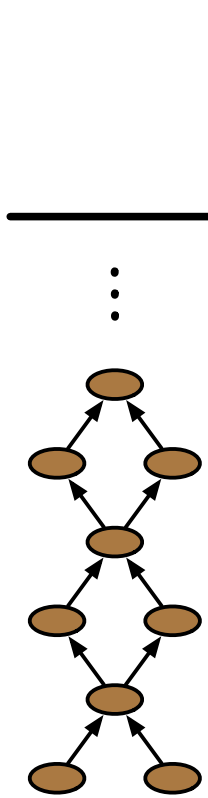
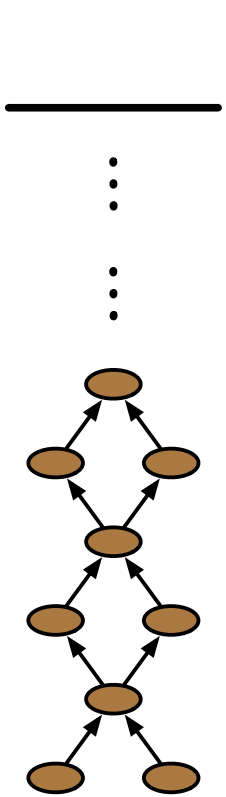
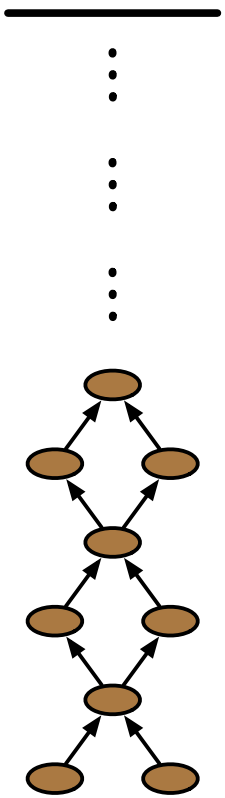
Main kind abstract Machines reading infinite words:

- Deterministic Muller automata
- Deterministic Muller Counter automata
- Deterministic Muller Pushdown automata
- Deterministic Muller Turing machines

Main kind abstract Machines reading infinite words:

- Deterministic Muller automata
- Deterministic Muller Counter automata
- Deterministic Muller Pushdown automata
- Deterministic Muller Turing machines



DetMA	DetMCA	DetMPDA	DetMTM
height ω^ω	height $(\omega^2)^\omega = \omega^\omega$	height $(\omega^\omega)^\omega$	height $(\omega_1^{CK})^\omega$
			
Decidability pb: solved	Decidability pb: solved	Decidability pb: unsolved	Decidability pb: unsolved

- The reduction relation $\leq_W$ leads to very refined transfinite hierarchies of abstract machines.
- We aim to study the computational complexity of neural nets in terms of infinite word reading automata.

- The reduction relation $\leq_W$ leads to very refined transfinite hierarchies of abstract machines.
- We aim to study the computational complexity of neural nets in terms of infinite word reading automata.