Automata over infinite objects

Karoliina Lehtinen
CNRS, Aix-Marseille Université, LIS

EPIT 2025

Part I:
Finite state automata over infinite words

Words

- Finite alphabet Σ, e.g.
  $\{0,1\}, \{a,b,c,\ldots,y,z\}, \{0,1,2,3,4,5\}, \{\texttt{req, wait,grant}\}$

Words

- Finite alphabet Σ, e.g.
  $\{0,1\}, \{a,b,c,\ldots,y,z\}, \{0,1,2,3,4,5\}, \{\texttt{req, wait,grant}\}$
- $hello$

Words

- Finite alphabet Σ, e.g.
  $\{0,1\}, \{a,b,c,\ldots,y,z\}, \{0,1,2,3,4,5\}, \{\texttt{req, wait,grant}\}$
- $hello$
- 00100

Words

- Finite alphabet Σ, e.g. $\{0,1\}, \{a,b,c,\ldots,y,z\}, \{0,1,2,3,4,5\}, \{\texttt{req, wait,grant}\}$
- $hello$
- 00100
- $010101...$ a.k.a. $(01)^\omega$

Words

- Finite alphabet Σ, e.g.
  $\{0,1\}, \{a,b,c,\ldots,y,z\}, \{0,1,2,3,4,5\}, \{\texttt{req, wait,grant}\}$
- $hello$
- 00100
- $010101...$ a.k.a. $(01)^\omega$
- $31415926535897932384...$

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0, 1\}$ in which 1 occurs: $(0+1)^*1(0+1)^*$

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0, 1\}$ in which 1 occurs: $(0+1)^*1(0+1)^*$
- $\{a^n b^n \mid n \in \mathbb{N}\}$

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0,1\}$ in which 1 occurs: $(0+1)^*1(0+1)^*$
- $\{a^n b^n \mid n \in \mathbb{N}\}$
- Every `request` is eventually followed by a `grant`

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0, 1\}$ in which 1 occurs: $(0+1)^*1(0+1)^*$
- $\{a^n b^n \mid n \in \mathbb{N}\}$
- Every `request` is eventually followed by a `grant`

Languages of infinite words

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0, 1\}$ in which 1 occurs: $(0 + 1)^*1(0 + 1)^*$
- $\{a^n b^n \mid n \in \mathbb{N}\}$
- Every `request` is eventually followed by a `grant`

Languages of infinite words

- Liveness: `progress` occurs infinitely often

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0, 1\}$ in which 1 occurs: $(0 + 1)^*1(0 + 1)^*$
- $\{a^n b^n \mid n \in \mathbb{N}\}$
- Every `request` is eventually followed by a `grant`

Languages of infinite words

- Liveness: `progress` occurs infinitely often
- Eventually `error` does not occur

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0, 1\}$ in which 1 occurs: $(0+1)^*1(0+1)^*$
- $\{a^n b^n \mid n \in \mathbb{N}\}$
- Every `request` is eventually followed by a `grant`

Languages of infinite words

- Liveness: `progress` occurs infinitely often
- Eventually `error` does not occur
- Every `request` is eventually followed by a `grant`

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0,1\}$ in which 1 occurs: $(0+1)^*1(0+1)^*$
- $\{a^n b^n \mid n \in \mathbb{N}\}$
- Every `request` is eventually followed by a `grant`

Languages of infinite words

- Liveness: `progress` occurs infinitely often
- Eventually `error` does not occur
- Every `request` is eventually followed by a `grant`
- The word is well-bracketed

Properties of finite and infinite words

Languages (i.e. sets) of finite words:

- Words over $\{0, 1\}$ in which 1 occurs: $(0 + 1)^*1(0 + 1)^*$
- $\{a^n b^n \mid n \in \mathbb{N}\}$
- Every `request` is eventually followed by a `grant`

Languages of infinite words

- Liveness: `progress` occurs infinitely often
- Eventually `error` does not occur
- Every `request` is eventually followed by a `grant`
- The word is well-bracketed
- Words over $\{+1, -1\}$ where the energy value reaches 0 infinitely often.

(ω-)regular languages

Languages (i.e. sets) of finite words:

- Words over $\{0,1\}$ in which 1 occurs: $(0+1)^*1(0+1)^*$
- $\cancel{\{a^n b^n \mid n \in \mathbb{N}\}}$
- Every **request** is eventually followed by a **grant**

Languages of infinite words

- "Liveness": **progress** occurs infinitely often
- Eventually **error** does not occur
- Every **request** is eventually followed by a **grant**
- ~~The word is well-bracketed~~
- ~~Words over $\{+1,-1\}$ where the energy value reaches 0 infinitely often.~~

(ω-)regular languages

Languages (i.e. sets) of finite words:

- Words over $\{0,1\}$ in which 1 occurs: $(0+1)^*1(0+1)^*$
- $\cancel{\{a^n b^n \mid n \in \mathbb{N}\}}$
- Every `request` is eventually followed by a **grant**

Languages of infinite words

- "Liveness": `progress` occurs infinitely often
- Eventually `error` does not occur
- Every `request` is eventually followed by a **grant**
- ~~The word is well-bracketed~~
- ~~Words over $\{+1,-1\}$ where the energy value reaches 0 infinitely often.~~

Finitely bounded behaviour.

Representing (ω-) regular languages

Over finite words:

- Regular expressions
- Logics: e.g. FO, MSO
- Automata: DFA, NFA
- Monoids
- ...

Over infinite words:

Representing (ω-) regular languages

Over finite words:

- Regular expressions
- Logics: e.g. FO, MSO
- Automata: DFA, NFA
- Monoids
- ...

Over infinite words:

- ω-regular expressions

Representing (ω-) regular languages

Over finite words:

- Regular expressions
- Logics: e.g. FO, MSO
- Automata: DFA, NFA
- Monoids
- ...

Over infinite words:

- ω-regular expressions
- Logics: e.g. LTL, linear-time μ-calculus

Representing (ω-) regular languages

Over finite words:

- Regular expressions
- Logics: e.g. FO, MSO
- Automata: DFA, NFA
- Monoids
- ...

Over infinite words:

- ω-regular expressions
- Logics: e.g. LTL, linear-time μ-calculus
- Automata: ω-regular automata

Representing (ω-) regular languages

Over finite words:

- Regular expressions
- Logics: e.g. FO, MSO
- Automata: DFA, NFA
- Monoids
- ...

Over infinite words:

- ω-regular expressions
- Logics: e.g. LTL, linear-time μ-calculus
- Automata: ω-regular automata
- Monoids
- ...

Representing (ω-) regular languages

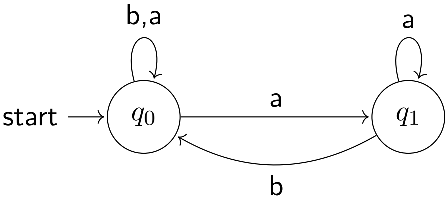
Over finite words:

- Regular expressions
- Logics: e.g. FO, MSO
- Automata: DFA, NFA
- Monoids
- ...

Over infinite words:

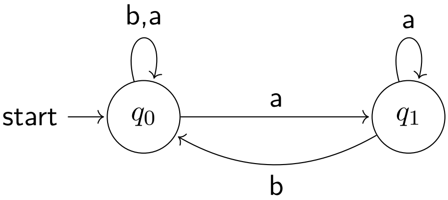
- ω-regular expressions
- Logics: e.g. LTL, linear-time μ-calculus
- **Automata: ω-regular automata**
- Monoids
- ...

Finite automata over infinite words



- Finite state space Q
- Transitions Δ
- Deterministic or nondeterministic
- Infinite runs

Finite automata over infinite words



- Finite state space Q
- Transitions Δ
- Deterministic or nondeterministic
- Infinite runs

What is an accepting run?

What is an accepting run?

When is a run $t_0 t_1 \cdots = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \ldots$ accepting?

What is an accepting run?

When is a run $t_0 t_1 \cdots = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \ldots$ accepting?

- **Safety**: All infinite runs are accepting.

What is an accepting run?

When is a run $t_0 t_1 \cdots = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \ldots$ accepting?

- **Safety**: All infinite runs are accepting.
- **Reachability** with target $\alpha \subseteq \Delta$: $\exists n. t_n \in \alpha$

What is an accepting run?

When is a run $t_0 t_1 \cdots = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \ldots$ accepting?

- **Safety**: All infinite runs are accepting.
- **Reachability** with target $\alpha \subseteq \Delta$: $\exists n. t_n \in \alpha$
- **Büchi**: accepting transitions $\alpha \subseteq \Delta$ occur infinitely often
  $\forall n. \exists i > n. t_i \in \alpha$

What is an accepting run?

When is a run $t_0 t_1 \cdots = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \ldots$ accepting?

- **Safety**: All infinite runs are accepting.
- **Reachability** with target $\alpha \subseteq \Delta$: $\exists n. t_n \in \alpha$
- **Büchi**: accepting transitions $\alpha \subseteq \Delta$ occur infinitely often
  $\forall n. \exists i > n. t_i \in \alpha$
- **coBüchi**: eventually only accepting transitions occur
  $\exists n. \forall i > n. t_i \in \alpha$

What is an accepting run?

When is a run $t_0 t_1 \cdots = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2$... accepting?

- **Safety**: All infinite runs are accepting.
- **Reachability** with target $\alpha \subseteq \Delta$: $\exists n. t_n \in \alpha$
- **Büchi**: accepting transitions $\alpha \subseteq \Delta$ occur infinitely often
  $\forall n. \exists i > n. t_i \in \alpha$
- **coBüchi**: eventually only accepting transitions occur
  $\exists n. \forall i > n. t_i \in \alpha$
- **Parity**: $\alpha : \Delta \to \{0, 1, \ldots, h\}$.
  Accept if the **highest priority** that occurs infinitely often is ***even***
  $\exists 2p \in [0, h]. \forall n. \exists i > n. \alpha(q_i) = 2p \wedge \exists n. \forall i > n. \alpha(q_i) \leq 2p$

What is an accepting run?

When is a run $t_0 t_1 \cdots = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \ldots$ accepting?

- **Safety**: All infinite runs are accepting.
- **Reachability** with target $\alpha \subseteq \Delta$: $\exists n. t_n \in \alpha$
- **Büchi**: accepting transitions $\alpha \subseteq \Delta$ occur infinitely often
  $\forall n. \exists i > n. t_i \in \alpha$
- **coBüchi**: eventually only accepting transitions occur
  $\exists n. \forall i > n. t_i \in \alpha$
- **Parity**: $\alpha : \Delta \to \{0, 1, \ldots, h\}$.
  Accept if the **highest priority** that occurs infinitely often is ***even***
  $\exists 2p \in [0, h]. \forall n. \exists i > n. \alpha(q_i) = 2p \wedge \exists n. \forall i > n. \alpha(q_i) \leq 2p$
- **Müller**: Boolean combination of transitions occurring infinitely/finitely often.

Büchi and coBüchi as simple parity

- Büchi $=$ Parity $\{1, 2\}$
- coBüchi $=$ Parity $\{0, 1\}$

Exercise: automata for these languages

- Liveness: **`progress`** occurs infinitely often
- Eventually **`error`** does not occur
- {`req`,`grant`,`wait`}: Every **`request`** is eventually followed by a **`grant`**
- {`req`,`grant`,`wait`}: If **`request`** occurs infinitely often, then **`grant`** occurs infinitely often

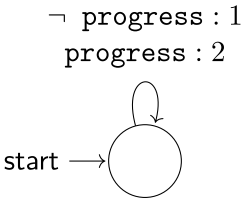


Figure: **progress** occurs infinitely often

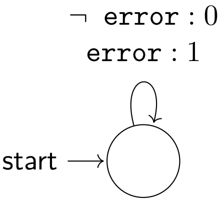
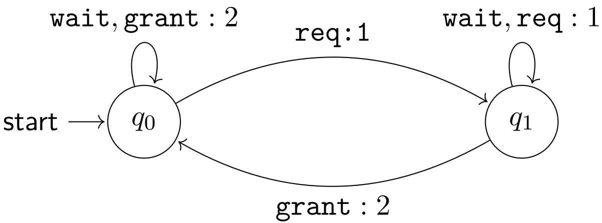
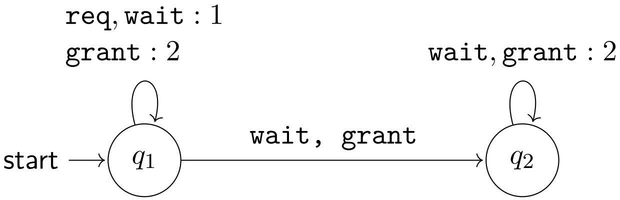


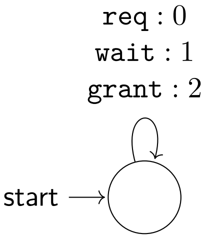
Figure: Eventually **error** does not occur

Every request is eventually followed by a grant



If infinitely many `req` then infinitely many `grants`

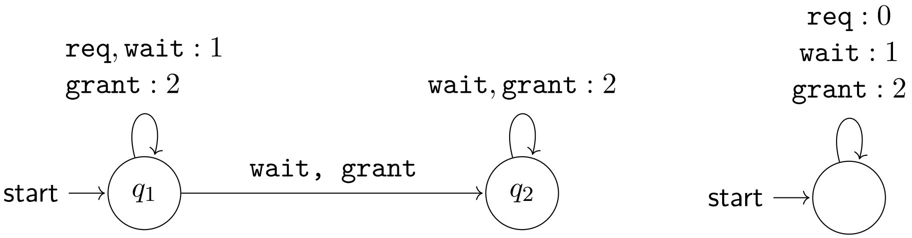




Trade-offs

- determinism vs. nondeterminism
- acceptance condition

Parity vs Büchi



Parity vs Büchi

Theorem

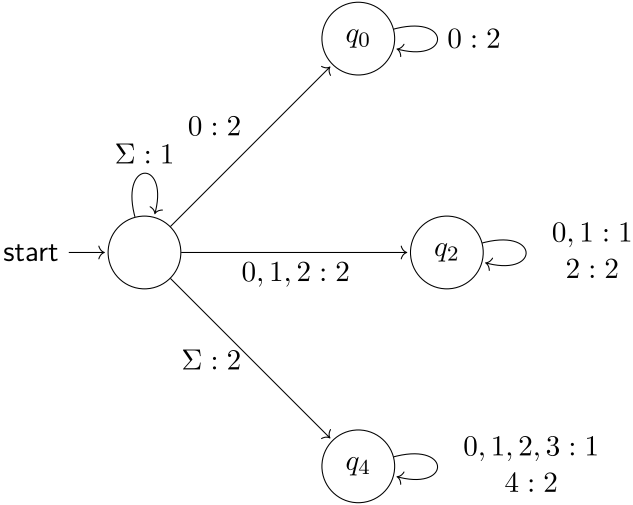
Every parity automaton can be turned into a nondeterministic Büchi automaton.

Exercise

Nondeterministic Büchi automaton for $\texttt{Parity}_{\{0,1,2,3,4,5\}}$

- Alphabet $\{0, 1, 2, 3, 4, 5\}$
- Accept if the max that occurs infinitely often is even.

Nondeterministic Büchi automaton for $\texttt{Parity}_{0,1,2,3,4}$



Parity vs Büchi

Theorem

Every parity automaton can be turned into a nondeterministic Büchi automaton.

Proof Idea

Parity vs Büchi

Theorem

Every parity automaton can be turned into a nondeterministic Büchi automaton.

Proof Idea

Given A with priorities $0, 1, 2, 3, 4$, the composition of A with the nondeterministic Büchi automaton for $\texttt{Parity}_{\{0,1,2,3,4\}}$ is a nondeterministic Büchi automaton for A.

Parity vs. Büchi (cont.)

Parity vs. Büchi (cont.)

Theorem

The language $\texttt{Parity}_{0,\ldots,n}$ is not recognised by a deterministic parity automaton with priorities $\{1, \ldots, n+1\}$

Parity vs. Büchi (cont.)

Theorem

The language $\texttt{Parity}_{0,\ldots,n}$ is not recognised by a deterministic parity automaton with priorities $\{1, \ldots, n+1\}$

Proof Idea

- Consider a deterministic automaton with priorities $\{1, \ldots, n+1\}$ over alphabet $\{0, \ldots n\}$.
- $a_0 a_1 \ldots$ where if the run over $a_0 \ldots a_i$ has as last priority p, then $a_{i+1} = p - 1$
- This word is accepted if and only if it is not in $\texttt{Parity}_{0\ldots n}$, so the automatons does not recognise this language.

Trade-offs

- For deterministic automata, the Parity-Index Hierarchy is infinite.
- For nondeterministic automata, it collapses to Büchi.

Recap

- Extend DFA/NFA to infinite words
- Accepting runs?

Recap

- Extend DFA/NFA to infinite words
- Accepting runs?
- Büchi: something good (2) happens infinitely often

Recap

- Extend DFA/NFA to infinite words
- Accepting runs?
- Büchi: something good (2) happens infinitely often
- coBüchi: bad things (1) only happen finitely often (then only 0)

Recap

- Extend DFA/NFA to infinite words
- Accepting runs?
- Büchi: something good (2) happens infinitely often
- coBüchi: bad things (1) only happen finitely often (then only 0)
- Parity: The highest priority that occurs infinitely often is even

Recap

- Extend DFA/NFA to infinite words
- Accepting runs?
- Büchi: something good (2) happens infinitely often
- coBüchi: bad things (1) only happen finitely often (then only 0)
- Parity: The highest priority that occurs infinitely often is even
- DPW $=$ NBW

Recap

- Extend DFA/NFA to infinite words
- Accepting runs?
- Büchi: something good (2) happens infinitely often
- coBüchi: bad things (1) only happen finitely often (then only 0)
- Parity: The highest priority that occurs infinitely often is even
- DPW $=$ NBW
- Deterministic parity hierarchy is infinite

Recap

- Extend DFA/NFA to infinite words
- Accepting runs?
- Büchi: something good (2) happens infinitely often
- coBüchi: bad things (1) only happen finitely often (then only 0)
- Parity: The highest priority that occurs infinitely often is even
- DPW $=$ NBW
- Deterministic parity hierarchy is infinite
- Non-determinism $=$ disjunctions... what about conjunctions?

Alternating automata

Alternating automata

Intuition:

- nondeterminism = disjunctions
- universal choices = conjunctions

Alternating automata

Intuition:

- nondeterminism = disjunctions
- universal choices = conjunctions

Transitions:

Alternating automata

Intuition:

- nondeterminism = disjunctions
- universal choices = conjunctions

Transitions:

- $q \xrightarrow{a} \{q_1, q_2\}, q \xrightarrow{a} \{q_3\}$

Alternating automata

Intuition:

- nondeterminism = disjunctions
- universal choices = conjunctions

Transitions:

- $q \xrightarrow{a} \{q_1, q_2\}, q \xrightarrow{a} \{q_3\}$
- or $q \xrightarrow{a} (q_1 \wedge q_2) \vee q_3$

Alternating automata

Intuition:

- nondeterminism $=$ disjunctions
- universal choices $=$ conjunctions

Transitions:

- $q \xrightarrow{a} \{q_1, q_2\}, q \xrightarrow{a} \{q_3\}$
- or $q \xrightarrow{a} (q_1 \wedge q_2) \vee q_3$
- or disjunctive states, conjunctive states, ε-transitions.

Alternating automata

Intuition:

- nondeterminism = disjunctions
- universal choices = conjunctions

Transitions:

- $q \xrightarrow{a} \{q_1, q_2\}, q \xrightarrow{a} \{q_3\}$
- or $q \xrightarrow{a} (q_1 \wedge q_2) \vee q_3$
- or disjunctive states, conjunctive states, ε-transitions.

With priorities:

- $q \xrightarrow{a:1} (q_1 \wedge q_2) \vee q_3$

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Game-vision:

- $A(w)$ is a parity game:
  - Positions $\Sigma^\omega \times \mathbb{BF}(\mathbb{Q})$

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Game-vision:

- $A(w)$ is a parity game:
  - Positions $\Sigma^\omega \times \mathbb{BF}(\mathbb{Q})$
  - Moves $(aw, q) \xrightarrow{p} (w, b)$ where $q \xrightarrow{a:p} b$

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Game-vision:

- $A(w)$ is a parity game:
  - Positions $\Sigma^\omega \times \mathbb{BF}(\mathbb{Q})$
  - Moves $(aw, q) \xrightarrow{p} (w, b)$ where $q \xrightarrow{a:p} b$
  - and $(w, b_0 \wedge b1) \rightarrow (w, b_i), i \in \{0, 1\}$
  - and $(w, b_0 \vee b1) \rightarrow (w, b_i), i \in \{0, 1\}$

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Game-vision:

- $A(w)$ is a parity game:
  - Positions $\Sigma^\omega \times \mathbb{BF}(\mathbb{Q})$
  - Moves $(aw, q) \xrightarrow{p} (w, b)$ where $q \xrightarrow{a:p} b$
  - and $(w, b_0 \wedge b1) \to (w, b_i), i \in \{0, 1\}$
  - and $(w, b_0 \vee b1) \to (w, b_i), i \in \{0, 1\}$
  - Verifier resolves disjuntions
  - Falsifier resolves conjunctions

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Game-vision:

- $A(w)$ is a parity game:
  - Positions $\Sigma^\omega \times \mathbb{BF}(\mathbb{Q})$
  - Moves $(aw, q) \xrightarrow{p} (w, b)$ where $q \xrightarrow{a:p} b$
  - and $(w, b_0 \wedge b1) \to (w, b_i), i \in \{0, 1\}$
  - and $(w, b_0 \vee b1) \to (w, b_i), i \in \{0, 1\}$
  - Verifier resolves disjuntions
  - Falsifier resolves conjunctions
  - Results in an infinite path

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Game-vision:

- $A(w)$ is a parity game:
  - Positions $\Sigma^\omega \times \mathbb{BF}(\mathbb{Q})$
  - Moves $(aw, q) \xrightarrow{p} (w, b)$ where $q \xrightarrow{a:p} b$
  - and $(w, b_0 \wedge b1) \to (w, b_i), i \in \{0, 1\}$
  - and $(w, b_0 \vee b1) \to (w, b_i), i \in \{0, 1\}$
  - Verifier resolves disjuntions
  - Falsifier resolves conjunctions
  - Results in an infinite path
  - Verifier wins if the path is `Parity` accepting

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Game-vision:

- $A(w)$ is a parity game:
  - Positions $\Sigma^\omega \times \mathbb{BF}(\mathbb{Q})$
  - Moves $(aw, q) \xrightarrow{p} (w, b)$ where $q \xrightarrow{a:p} b$
  - and $(w, b_0 \wedge b1) \rightarrow (w, b_i), i \in \{0, 1\}$
  - and $(w, b_0 \vee b1) \rightarrow (w, b_i), i \in \{0, 1\}$
  - Verifier resolves disjuntions
  - Falsifier resolves conjunctions
  - Results in an infinite path
  - Verifier wins if the path is `Parity` accepting
- $w \in L(A)$ iff Verifier wins $A(w)$

Semantics

Run-vision:

- A run is an infinite tree
- Accepts if every branch is `Parity` accepting

Game-vision:

- $A(w)$ is a parity game:
    - Positions $\Sigma^\omega \times \mathbb{BF}(\mathbb{Q})$
    - Moves $(aw, q) \xrightarrow{p} (w, b)$ where $q \xrightarrow{a:p} b$
    - and $(w, b_0 \wedge b1) \to (w, b_i), i \in \{0, 1\}$
    - and $(w, b_0 \vee b1) \to (w, b_i), i \in \{0, 1\}$
    - Verifier resolves disjuntions
    - Falsifier resolves conjunctions
    - Results in an infinite path
    - Verifier wins if the path is `Parity` accepting
- $w \in L(A)$ iff Verifier wins $A(w)$
- Verifier strategy $=$ run tree

Alternating automaton

Exercise

Alternating coBüchi automaton for
$\{$`wait, req, grant`$\}$: Every grant is eventually followed by a request.

Trade-offs

The following are equally expressive:

- Alternating weak automata (SCCs are accepting or rejecting)
- Nondeterministic Büchi automata
- Deterministic parity

Logic and automata

Some formulas:

- $\mu X.a \vee \Diamond X$ (eventually a)
- $\nu X.a \wedge \Diamond X$ (always a)
- $\nu X.(\Diamond X \wedge \mu Y.a \vee \Diamond Y)$ (Always eventually a)
- $\mu X.\nu Y.(a \wedge \Diamond Y) \vee (\neg a \wedge \Diamond X)$ (Eventually always a)

Part II.
Finite state automata over infinite trees

Infinite trees

- Labelled *unordered* trees
- Unfolded rooted graphs
- $(V, E, \ell : V \to \Sigma)$ where (V, E) is a tree

Examples

- a occurs finitely often on all paths.
- A subtree without any a is reachable from everywhere.
- b is reachable whenever a occurs.
- Player 1 wins the a-reachability game (alternating reachability).

Alternating parity tree automata

- $\Diamond q$: accept from q in *some* successor

Alternating parity tree automata

- $\Diamond q$: accept from q in *some* successor
- $\Box q$: accept from q in *all* successors

Alternating parity tree automata

- $\Diamond q$: accept from q in *some* successor
- $\Box q$: accept from q in *all* successors
- Transitions: DNF formula over $\{\Diamond q, \Box q \mid q \in Q\}$
  $q \xrightarrow{a} (\Diamond q_1 \wedge \Diamond q_2) \vee \Box q_3$

Example

- $q_0 \xrightarrow{\Sigma:0} \Box q_0 \wedge \Diamond q_1$
- $q_1 \xrightarrow{a:1} \Diamond q_1$
- $q_1 \xrightarrow{b:0} \Box \top$
- $q_\top \xrightarrow{\Sigma:0} \Box \top$

Example

- $q_0 \xrightarrow{\Sigma:0} \Box q_0 \wedge \Diamond q_1$
- $q_1 \xrightarrow{a:1} \Diamond q_1$
- $q_1 \xrightarrow{b:0} \Box \top$
- $q_\top \xrightarrow{\Sigma:0} \Box \top$

$$\nu X_0.\Box X_0 \wedge \Diamond \mu X_1.(a \wedge \Diamond X_1) \vee b$$

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

- Positions: $V \times \mathbb{BF}(\{\Diamond q, \Box q \mid q \in Q\})$ (positive boolean formulas)

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

- Positions: $V \times \mathbb{BF}(\{\Diamond q, \Box q \mid q \in Q\})$ (positive boolean formulas)
- $(v, q) \xrightarrow{p} (v, \phi)$ where $q \xrightarrow{l(v):p} \phi$

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

- Positions: $V \times \mathbb{BF}(\{\Diamond q, \Box q \mid q \in Q\})$ (positive boolean formulas)
- $(v, q) \xrightarrow{p} (v, \phi)$ where $q \xrightarrow{l(v):p} \phi$
- $(v, b_0 \vee b_1) \to (v, b_i)$ for $i \in \{0, 1\}$

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

- Positions: $V \times \mathbb{BF}(\{\Diamond q, \Box q \mid q \in Q\})$ (positive boolean formulas)
- $(v, q) \xrightarrow{p} (v, \phi)$ where $q \xrightarrow{l(v):p} \phi$
- $(v, b_0 \vee b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, b_0 \wedge b_1) \to (v, b_i)$ for $i \in \{0, 1\}$

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

- Positions: $V \times \mathbb{BF}(\{\Diamond q, \Box q \mid q \in Q\})$ (positive boolean formulas)
- $(v, q) \xrightarrow{p} (v, \phi)$ where $q \xrightarrow{l(v):p} \phi$
- $(v, b_0 \vee b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, b_0 \wedge b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, \Diamond q) \to (v', q)$ for all $(v, v') \in E$

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

- Positions: $V \times \mathbb{BF}(\{\Diamond q, \Box q \mid q \in Q\})$ (positive boolean formulas)
- $(v, q) \xrightarrow{p} (v, \phi)$ where $q \xrightarrow{l(v):p} \phi$
- $(v, b_0 \vee b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, b_0 \wedge b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, \Diamond q) \to (v', q)$ for all $(v, v') \in E$
- $(v, \Box q) \to (v', q)$ for all $(v, v') \in E$

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

- Positions: $V \times \mathbb{BF}(\{\Diamond q, \Box q \mid q \in Q\})$ (positive boolean formulas)
- $(v, q) \xrightarrow{p} (v, \phi)$ where $q \xrightarrow{l(v):p} \phi$
- $(v, b_0 \vee b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, b_0 \wedge b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, \Diamond q) \to (v', q)$ for all $(v, v') \in E$
- $(v, \Box q) \to (v', q)$ for all $(v, v') \in E$
- $\vee, \Diamond$-positions belong to Verifier

Semantics via parity games

$t = (V, E, l : V \to \Sigma) \in L(A)$ if Verifier wins the following parity game:

- Positions: $V \times \mathbb{BF}(\{\Diamond q, \Box q \mid q \in Q\})$ (positive boolean formulas)
- $(v, q) \xrightarrow{p} (v, \phi)$ where $q \xrightarrow{l(v):p} \phi$
- $(v, b_0 \vee b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, b_0 \wedge b_1) \to (v, b_i)$ for $i \in \{0, 1\}$
- $(v, \Diamond q) \to (v', q)$ for all $(v, v') \in E$
- $(v, \Box q) \to (v', q)$ for all $(v, v') \in E$
- $\vee, \Diamond$-positions belong to Verifier
- $\wedge, \Box$-positions belong to Falsifier

Example

- $q_0 \xrightarrow{\Sigma:0} \Box q_0 \wedge \Diamond q_1$
- $q_1 \xrightarrow{a:1} \Diamond q_1$
- $q_1 \xrightarrow{b:0} q_\top$
- $q_\top \xrightarrow{\Sigma:0} \top$

Example

- $q_0 \xrightarrow{\Sigma:0} \Box q_0 \wedge \Diamond q_1$
- $q_1 \xrightarrow{a:1} \Diamond q_1$
- $q_1 \xrightarrow{b:0} q_\top$
- $q_\top \xrightarrow{\Sigma:0} \top$

$$\nu X_0.\Box X_0 \wedge \Diamond \mu X_1.(a \wedge \Diamond X_1) \vee b$$

Examples

- a occurs finitely often on all paths.
- A subtree without any a is reachable from everywhere.
- b is reachable whenever a occurs.
- Player 1 wins the a-reachability game (alternating reachability).

Automata and the modal μ-calculus

Intuition:

- Formula $\longleftrightarrow$ Automaton
- subformulas $\longleftrightarrow$ states
- $\Diamond, \Box \longleftrightarrow$ transitions
- $\mu, \nu \longleftrightarrow$ priorities
- nested fixpoint semantics $\longleftrightarrow$ parity condition

Intuitive equivalence, but...

Many annoying technicalities of varying importance:

- Automata are often defined on *ordered binary trees* (v.s. arbitrary/infinite branching)

Intuitive equivalence, but...

Many annoying technicalities of varying importance:

- Automata are often defined on *ordered binary trees* (v.s. arbitrary/infinite branching)
- Nondeterminism is neater on ordered binary trees

Intuitive equivalence, but...

Many annoying technicalities of varying importance:

- Automata are often defined on *ordered binary trees* (v.s. arbitrary/infinite branching)
- Nondeterminism is neater on ordered binary trees
- Labels on nodes vs. edges, $< a >, [a]$ vs. $\Diamond, \Box$

Intuitive equivalence, but...

Many annoying technicalities of varying importance:

- Automata are often defined on *ordered binary trees* (v.s. arbitrary/infinite branching)
- Nondeterminism is neater on ordered binary trees
- Labels on nodes vs. edges, $< a >, [a]$ vs. $\Diamond, \Box$
- Size of a formula/automaton?

Intuitive equivalence, but...

Many annoying technicalities of varying importance:

- Automata are often defined on *ordered binary trees* (v.s. arbitrary/infinite branching)
- Nondeterminism is neater on ordered binary trees
- Labels on nodes vs. edges, $< a >, [a]$ vs. $\Diamond, \Box$
- Size of a formula/automaton?
- ...

Parity index hierarchy

Recall, over words, these are equally expressive:

- alternating weak
- nondeterministic Büchi
- deterministic parity

Parity index hierarchy

Recall, over words, these are equally expressive:

- alternating weak
- nondeterministic Büchi
- deterministic parity

Over trees:

- Automata can not always be determinised

Parity index hierarchy

Recall, over words, these are equally expressive:

- alternating weak
- nondeterministic Büchi
- deterministic parity

Over trees:

- Automata can not always be determinised
- Alternations can be removed

Parity index hierarchy

Recall, over words, these are equally expressive:

- alternating weak
- nondeterministic Büchi
- deterministic parity

Over trees:

- Automata can not always be determinised
- Alternations can be removed
- Parity-index hierarchy is infinite for deterministic, nondeterministic and alternating automata.

A classic for a reason

- Highly expressive
  subsumes CTL, bisimulation invariant MSO...
- Highly decidable

Some classic open problems on tree automata

- Complexity of model-checking ($t \in L(A)$?):
  NP$\cap$ co-NP but no known PTime algorithm...
  ecently quasipolynomial [2017, Calude et el.]

Some classic open problems on tree automata

- Complexity of model-checking ($t \in L(A)?$):
  $\mathrm{NP} \cap$ $\textsc{co}\text{-}\mathrm{NP}$ but no known $\mathrm{P}\textsc{Time}$ algorithm...
  Recently quasipolynomial [2017, Calude et el.]

Some classic open problems on tree automata

- Complexity of model-checking ($t \in L(A)?$):
  $\mathrm{NP} \cap \textsc{co}\text{-}\mathrm{NP}$ but no known $\mathrm{P}\textsc{Time}$ algorithm...
  Recently quasipolynomial [2017, Calude et el.]
- Deciding the parity index:
  Given a tree language, what is the least number of priorities needed to express it with an alternating or nondeterministic automaton?

A strong foundation for extended models

Especially in the case of words:

- Infinite alphabets

A strong foundation for extended models

Especially in the case of words:

- Infinite alphabets
- Infinite memory (counters, stacks...)

A strong foundation for extended models

Especially in the case of words:

- Infinite alphabets
- Infinite memory (counters, stacks...)
- Quantitative models

A strong foundation for extended models

Especially in the case of words:

- Infinite alphabets
- Infinite memory (counters, stacks...)
- Quantitative models
- ...

A strong foundation for extended models

Especially in the case of words:

- Infinite alphabets
- Infinite memory (counters, stacks...)
- Quantitative models
- ...

And for cyclic proofs?

Reading

- Automata, logics, and infinite games: a guide to current research (2003) *E Grädel, W Thomas, T Wilke*
- ω-automata (2014) *T. Wilke*
  `https://arxiv.org/abs/1609.03062`
- ω-regular automata (2021) *S. Schewe, T. Wilke*
  Automata on infinite trees *Christof Löding*
  in Handbook of Automata Theory
- Automata on Infinite Objects *Wolfgang THOMAS*
  Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics 1990