

The theory of regular cost functions, from finite words to infinite trees.

Denis Kuperberg

PhD under supervision of Thomas Colcombet

LIAFA, Université Paris Diderot, France

Séminaire du LSV, Cachan
19-03-2012

Introduction

- **Church-Turing 1936:**

Which problems can be answered by an algorithm ?

It has yield the notion of decidability.

Introduction

- **Church-Turing 1936:**
Which problems can be answered by an algorithm ?
It has yield the notion of decidability.
- Some natural problems are **undecidable**.

Introduction

- **Church-Turing 1936:**
Which problems can be answered by an algorithm ?
It has yield the notion of decidability.
- Some natural problems are **undecidable**.
- For some problems, decidability is open.

Introduction

- **Church-Turing 1936:**
Which problems can be answered by an algorithm ?
It has yield the notion of decidability.
- Some natural problems are **undecidable**.
- For some problems, decidability is open.
- **Finite Automata:** Formalism with a lot of **decidable** properties.

Introduction

- **Church-Turing 1936:**
Which problems can be answered by an algorithm ?
It has yield the notion of decidability.
- Some natural problems are **undecidable**.
- For some problems, decidability is open.
- **Finite Automata:** Formalism with a lot of **decidable** properties.
- **Automata theory:**
Toolbox to decide many problems arising naturally.
Verification of systems can be done automatically.
Theoretical and practical advantages.

Introduction

- **Church-Turing 1936:**

Which problems can be answered by an algorithm ?
It has yield the notion of decidability.

- Some natural problems are **undecidable**.

- For some problems, decidability is open.

- **Finite Automata:** Formalism with a lot of **decidable** properties.

- **Automata theory:**

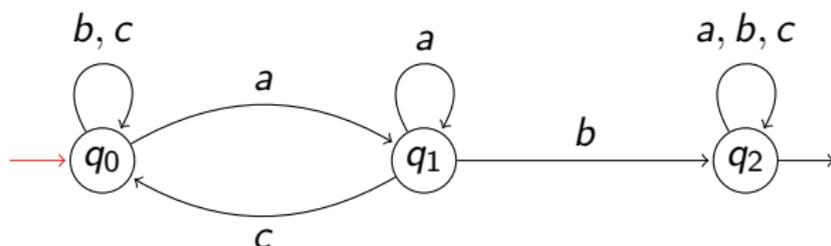
Toolbox to decide many problems arising naturally.
Verification of systems can be done automatically.
Theoretical and practical advantages.

- **Problem:**

Decidability is still open for some automata-related problems.

- 1 Automata theory
- 2 Regular Cost Functions
- 3 Contributions of the thesis
- 4 Zoom: Aperiodic Cost Functions

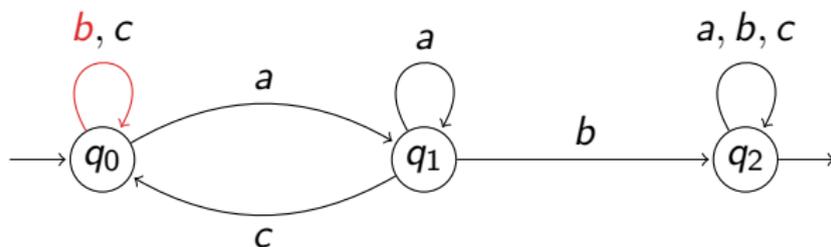
Finite Automaton



A word $u \in \mathbb{A}^*$ is **accepted** by \mathcal{A} if there is an accepting path labeled by u :

Example : Accepting path for the word $babbc$.

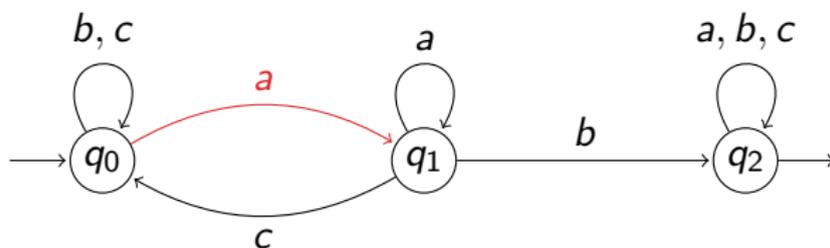
Finite Automaton



A word $u \in \mathbb{A}^*$ is **accepted** by \mathcal{A} if there is an accepting path labeled by u :

Example : Accepting path for the word bab .

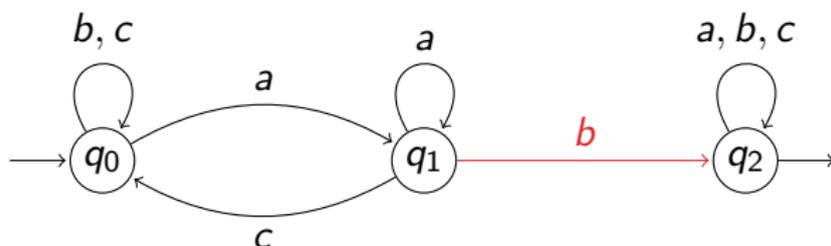
Finite Automaton



A word $u \in \mathbb{A}^*$ is **accepted** by \mathcal{A} if there is an accepting path labeled by u :

Example : Accepting path for the word abc .

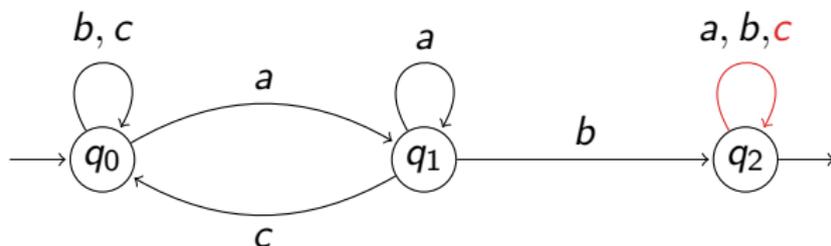
Finite Automaton



A word $u \in \mathbb{A}^*$ is **accepted** by \mathcal{A} if there is an accepting path labeled by u :

Example : Accepting path for the word $ba**b**bc$.

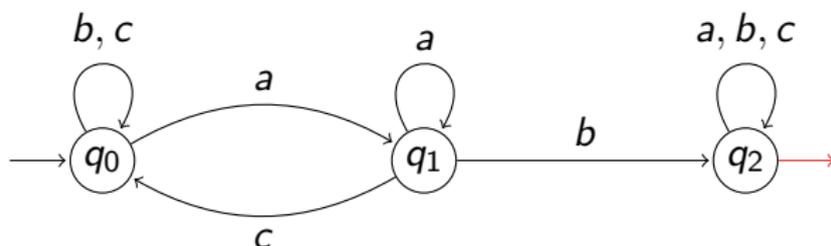
Finite Automaton



A word $u \in \mathbb{A}^*$ is **accepted** by \mathcal{A} if there is an accepting path labeled by u :

Example : Accepting path for the word $bab\mathbf{c}$.

Finite Automaton

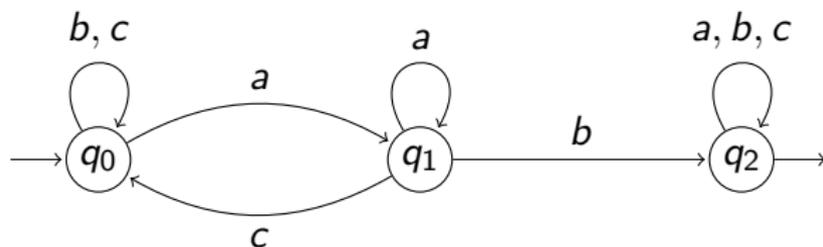


A word $u \in \mathbb{A}^*$ is **accepted** by \mathcal{A} if there is an accepting path labeled by u :

Example : Accepting path for the word $babbc$.

The **language recognized** by \mathcal{A} is the set $L \subseteq \mathbb{A}^*$ of words accepted by \mathcal{A} .

Descriptions of a language

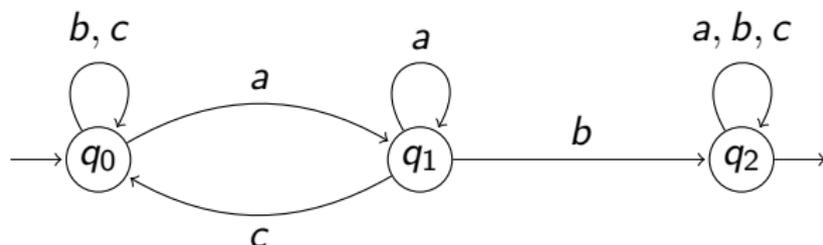


Language recognized : $L_{ab} = \{\text{words containing } ab\}$.

Other ways than automata to specify L_{ab} :

- **Regular expression** : $\mathbb{A}^*ab\mathbb{A}^*$,

Descriptions of a language

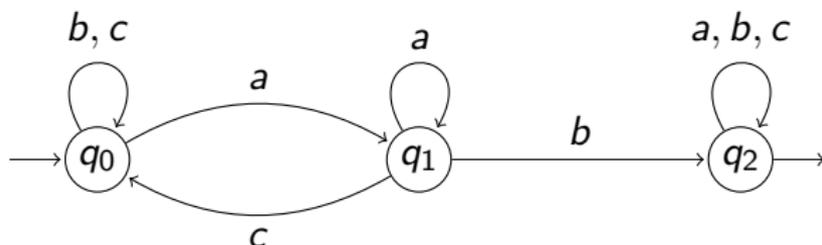


Language recognized : $L_{ab} = \{\text{words containing } ab\}$.

Other ways than automata to specify L_{ab} :

- **Regular expression** : $\mathbb{A}^* ab \mathbb{A}^*$,
- **Logical sentence (MSO)** : $\exists x \exists y a(x) \wedge b(y) \wedge (y = Sx)$.

Descriptions of a language



Language recognized : $L_{ab} = \{\text{words containing } ab\}$.

Other ways than automata to specify L_{ab} :

- **Regular expression** : $\mathbb{A}^* ab \mathbb{A}^*$,
- **Logical sentence (MSO)** : $\exists x \exists y a(x) \wedge b(y) \wedge (y = Sx)$.
- **Finite monoid** : $M = \{1, a, b, c, ba, 0\}$, $P = \{0\}$
 $ab = 0$, $aa = ca = a$, $bb = bc = b$, $cc = ac = cb = c$

Regular Languages

All these formalisms are effectively equivalent.

$a^n b^n$

Regular Languages

Expressions

MSO

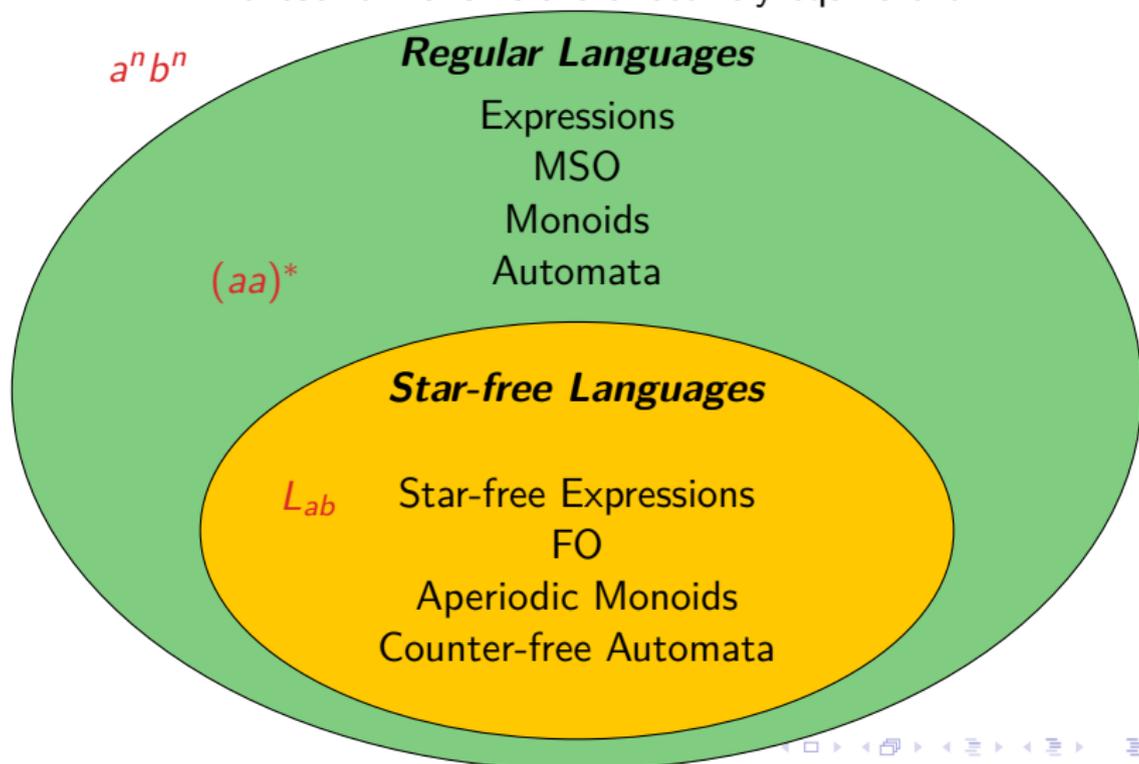
Monoids

Automata

$(aa)^*$

Regular Languages

All these formalisms are effectively equivalent.



Historical motivation

Given a class of languages \mathcal{C} , is there an algorithm which given an automaton for L , decides whether $L \in \mathcal{C}$?

Theorem (Schützenberger 1965)

It is decidable whether a regular language is star-free, thanks to the equivalence with aperiodic monoids.

Historical motivation

Given a class of languages \mathcal{C} , is there an algorithm which given an automaton for L , decides whether $L \in \mathcal{C}$?

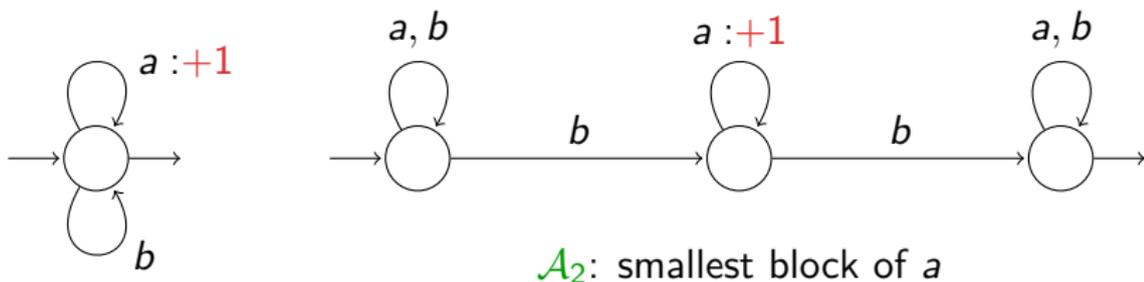
Theorem (Schützenberger 1965)

It is decidable whether a regular language is star-free, thanks to the equivalence with aperiodic monoids.

Finite Power Problem: Given L , is there n such that
$$(L + \varepsilon)^n = L^* ?$$

There is no known algebraic characterization,
other technics are needed to show decidability.

Distance Automata



Unbounded: There are words with arbitrarily large value.

Deciding **Boundedness** for distance automata \Rightarrow solving finite power problem.

Theorem (Hashiguchi 82, Kirsten 05)

Boundedness is decidable for distance automata.

Problems solved using counters

- **Finite Power** (finite words) [Simon '78, Hashiguchi '79]

Is there n such that $(L + \varepsilon)^n = L^*$?

- **Fixed Point Iteration** (finite words)

[Blumensath+Otto+Weyer '09]

Can we bound the number of fixpoint iterations in a MSO formula ?

- **Star-Height** (finite words/trees)

[Hashiguchi '88, Kirsten '05, Colcombet+Löding '08]

Given n , is there an expression for L , with at most n nesting of Kleene stars?

- **Parity Rank** (infinite trees)

[reduction in Colcombet+Löding '08, decidability open, deterministic input Niwinski+Walukiewicz '05]

Given $i < j$, is there a parity automaton for L using ranks $\{i, i + 1, \dots, j\}$?

- 1 Automata theory
- 2 Regular Cost Functions**
- 3 Contributions of the thesis
- 4 Zoom: Aperiodic Cost Functions

Theory of Regular Cost Functions

Aim: General framework for previous constructions.

- Generalize from languages $L : \mathbb{A}^* \rightarrow \{0, 1\}$
to functions $f : \mathbb{A}^* \rightarrow \mathbb{N} \cup \{\infty\}$
- Accordingly generalize automata, logics, semigroups, in order to obtain a **theory of regular cost functions**, which behaves as well as possible.
- Obtain decidability results thanks to this new theory.

Cost automata over words

Nondeterministic finite-state automaton \mathcal{A}

+ **finite set of counters**

(initialized to 0, values range over \mathbb{N})

+ **counter operations on transitions**

(increment I, reset R, check C, no change ε)

Semantics: $\llbracket \mathcal{A} \rrbracket : \Sigma^* \rightarrow \mathbb{N} \cup \{\infty\}$

Cost automata over words

Nondeterministic finite-state automaton \mathcal{A}

+ **finite set of counters**

(initialized to 0, values range over \mathbb{N})

+ **counter operations on transitions**

(increment **I**, reset **R**, check **C**, no change ε)

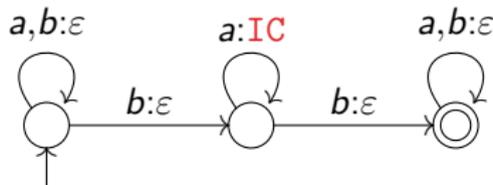
Semantics: $\llbracket \mathcal{A} \rrbracket : \Sigma^* \rightarrow \mathbb{N} \cup \{\infty\}$

$val_B(\rho) := \max$ checked counter value during run ρ

$\llbracket \mathcal{A} \rrbracket_B(u) := \min\{val_B(\rho) : \rho \text{ is an accepting run of } \mathcal{A} \text{ on } u\}$

Example

$\llbracket \mathcal{A} \rrbracket_B(u) = \min$ length of block of a 's surrounded by b 's in u



Cost automata over words

Nondeterministic finite-state automaton \mathcal{A}

+ **finite set of counters**

(initialized to 0, values range over \mathbb{N})

+ **counter operations on transitions**

(increment **I**, reset **R**, check **C**, no change ε)

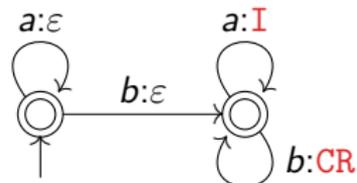
Semantics: $\llbracket \mathcal{A} \rrbracket : \Sigma^* \rightarrow \mathbb{N} \cup \{\infty\}$

$val_S(\rho) := \min$ checked counter value during run ρ

$\llbracket \mathcal{A} \rrbracket_S(u) := \max\{val_S(\rho) : \rho \text{ is an accepting run of } \mathcal{A} \text{ on } u\}$

Example

$\llbracket \mathcal{A} \rrbracket_S(u) = \min$ length of block of a 's surrounded by b 's in u



Boundedness relation

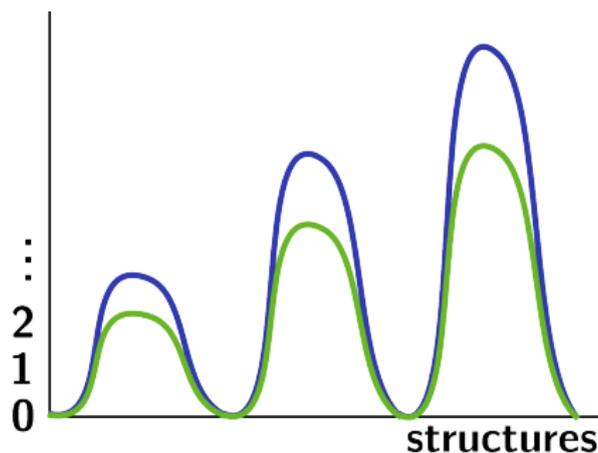
" $[[\mathcal{A}]] = [[\mathcal{B}]]$ ": undecidable [Krob '94]

Boundedness relation

" $[[\mathcal{A}]] = [[\mathcal{B}]]$ ": undecidable [Krob '94]

" $[[\mathcal{A}]] \approx [[\mathcal{B}]]$ ": decidable on words

[Colcombet '09, following Bojányczyk+Colcombet '06]
for all subsets U , $[[\mathcal{A}]](U)$ bounded iff $[[\mathcal{B}]](U)$ bounded



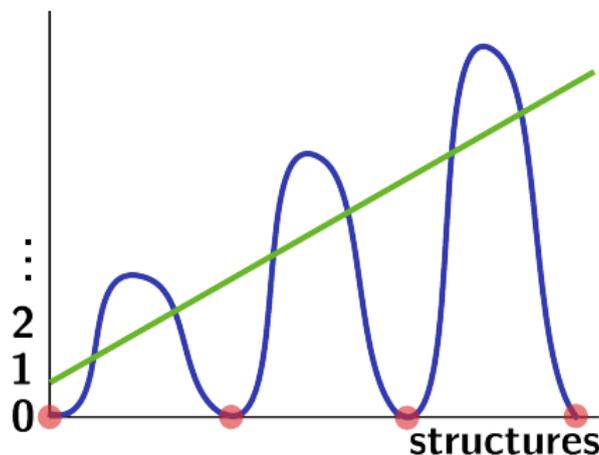
$$[[\mathcal{A}]] \approx [[\mathcal{B}]]$$

Boundedness relation

" $\llbracket \mathcal{A} \rrbracket = \llbracket \mathcal{B} \rrbracket$ ": undecidable [Krob '94]

" $\llbracket \mathcal{A} \rrbracket \approx \llbracket \mathcal{B} \rrbracket$ ": decidable on words

[Colcombet '09, following Bojányczyk+Colcombet '06]
for all subsets U , $\llbracket \mathcal{A} \rrbracket(U)$ bounded iff $\llbracket \mathcal{B} \rrbracket(U)$ bounded



$\llbracket \mathcal{A} \rrbracket \neq \llbracket \mathcal{B} \rrbracket$

Therefore we always identify two functions if they are bounded on the same sets.

Example

For any function f , we have $f \approx 2f \approx \exp(f)$.

But $(u \mapsto |u|_a) \not\approx (u \mapsto |u|_b)$, as witnessed by the set a^* .

Therefore we always identify two functions if they are bounded on the same sets.

Example

For any function f , we have $f \approx 2f \approx \exp(f)$.

But $(u \mapsto |u|_a) \not\approx (u \mapsto |u|_b)$, as witnessed by the set a^* .

Theorem (Colcombet '09, following Hashiguchi, Leung, Simon, Kirsten, Bojańczyk+Colcombet)

Cost automata \Leftrightarrow Cost logics \Leftrightarrow Stabilisation monoids.

For some suitable models of Cost Logics and Stabilisation Monoids, extending the classical ones.

Boundedness decidable.

All these equivalences are only valid up to \approx .

It provides a toolbox to decide boundedness problems.

Languages as cost functions

A language L is represented by its characteristic function

$$\chi_L(u) = \begin{cases} 0 & \text{if } u \in L \\ \infty & \text{if } u \notin L \end{cases}$$

If \mathcal{A} is a classical automaton for L , then $\llbracket \mathcal{A} \rrbracket_B = \chi_L$ and $\llbracket \mathcal{A} \rrbracket_S = \chi_{\bar{L}}$. Switching between B and S is the generalization of **language complementation**.

Cost function theory strictly extends language theory.

All theorems on cost functions are in particular true for languages.

Goal of the thesis: Studying cost function theory, and generalise known theorems from languages to cost functions.

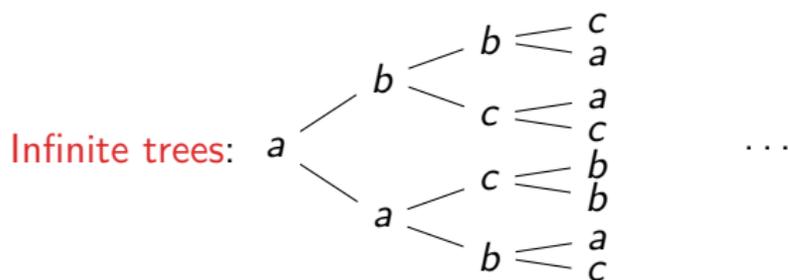
- 1 Automata theory
- 2 Regular Cost Functions
- 3 Contributions of the thesis**
- 4 Zoom: Aperiodic Cost Functions

Contributions of the thesis

Input structures:

Finite words: $accba$

Infinite words: $abaabaccbaba\dots$



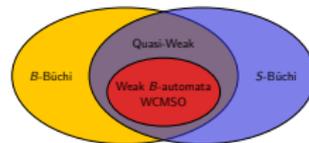
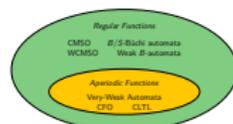
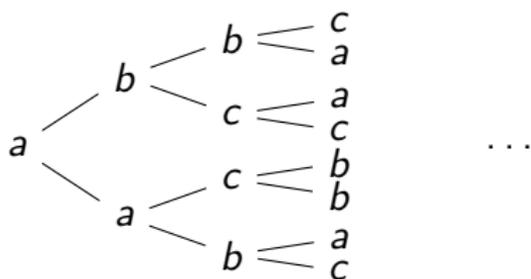
Contributions of the thesis

Input structures:

Finite words: *accba*

Infinite words: *abaabaccbaba...*

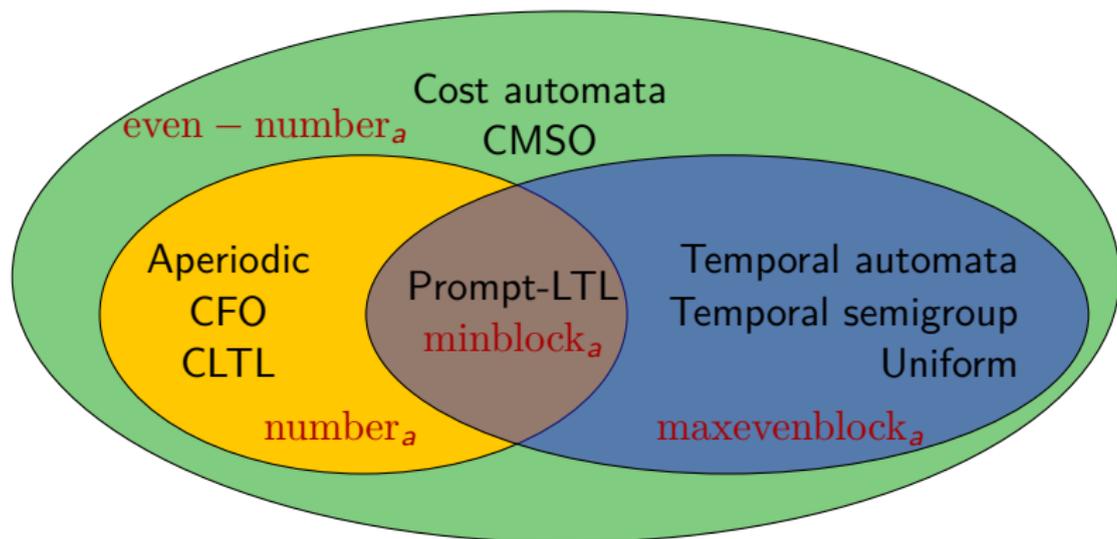
Infinite trees:



Different kinds of results:

- Generalisation of language notions and theorems,
- Study of classes specific to cost functions,
- Reduction of classical decision problems to boundedness problems.

Cost Functions on finite words



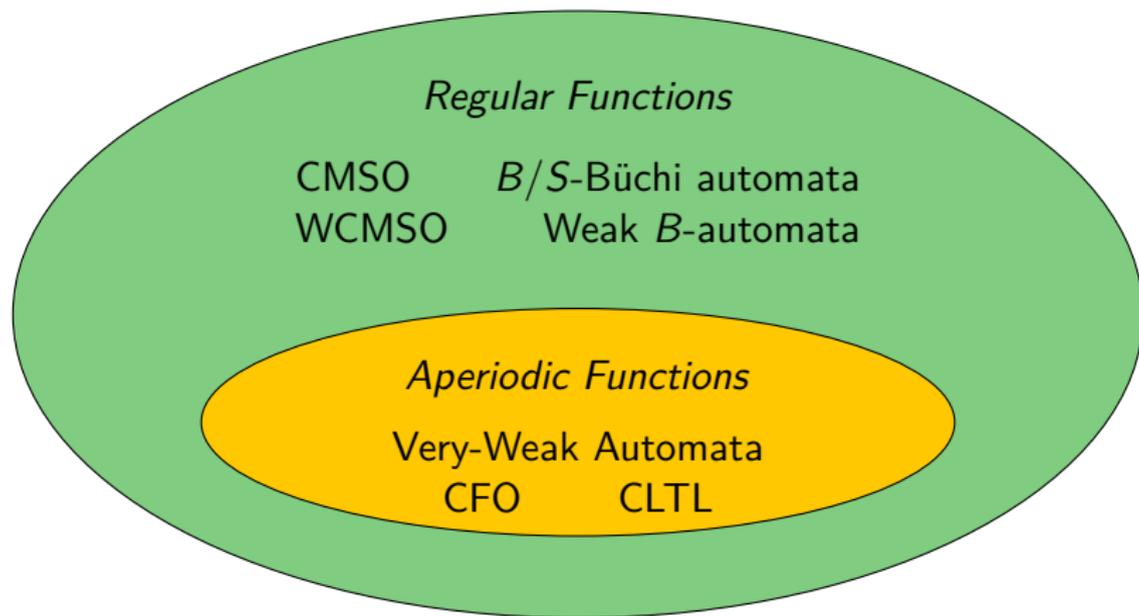
Decidability of membership and effectiveness of translations

[Colcombet+K.+Lombardy ICALP '10, K. STACS '11].

Generalization of Myhill-Nerode Equivalence [K. STACS '11].

Boundedness of CLTL is PSPACE-complete [Submitted to LMCS].

Cost Functions on infinite words



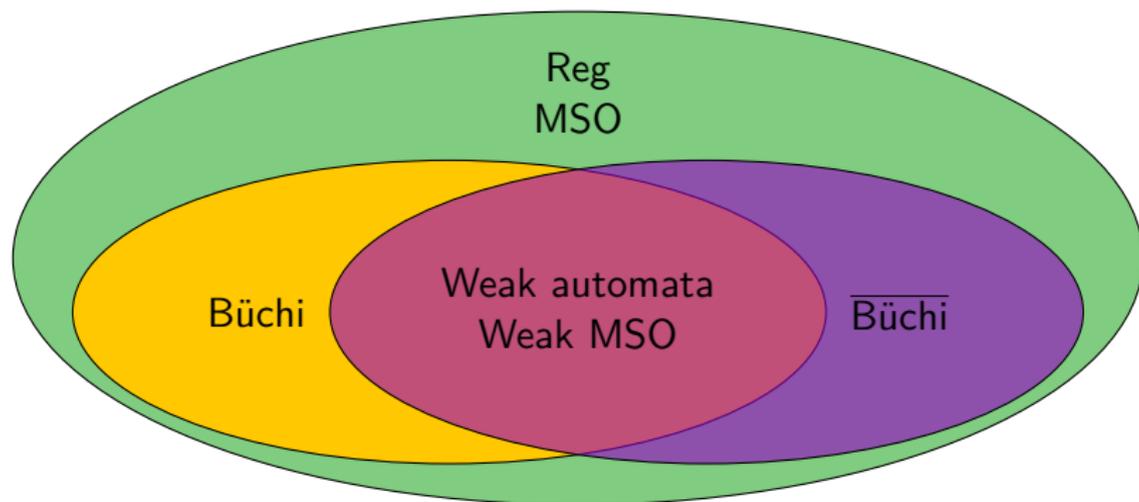
Decidability of membership and effectiveness of translations
[K.+Vanden Boom, ICALP '12].

Languages on infinite trees

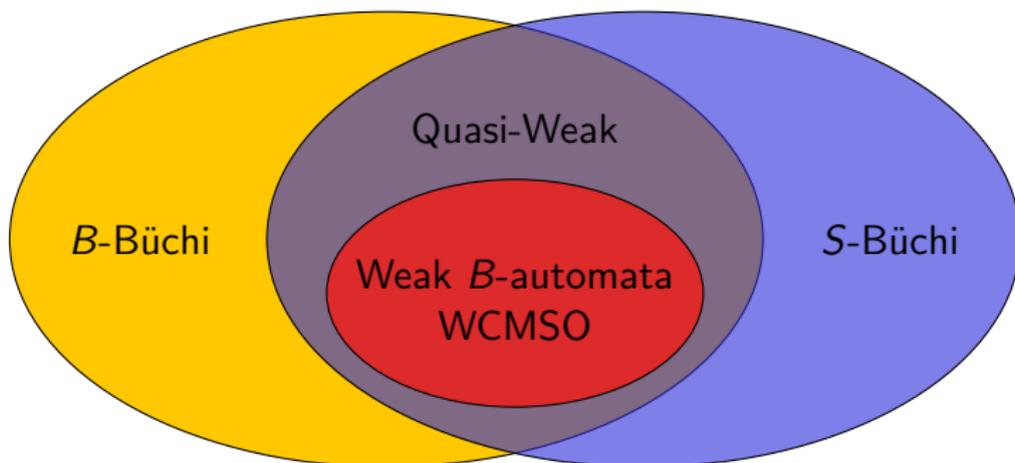
Theorem (Rabin 1970, Kupferman + Vardi 1999)

L recognizable by an alternating weak automaton \Leftrightarrow

L recognizable by WMSO \Leftrightarrow there are Büchi automata \mathcal{U} and $\overline{\mathcal{U}'}$ such that $L = L(\mathcal{U}) = \overline{L(\overline{\mathcal{U}'})}$.



Cost functions on infinite trees



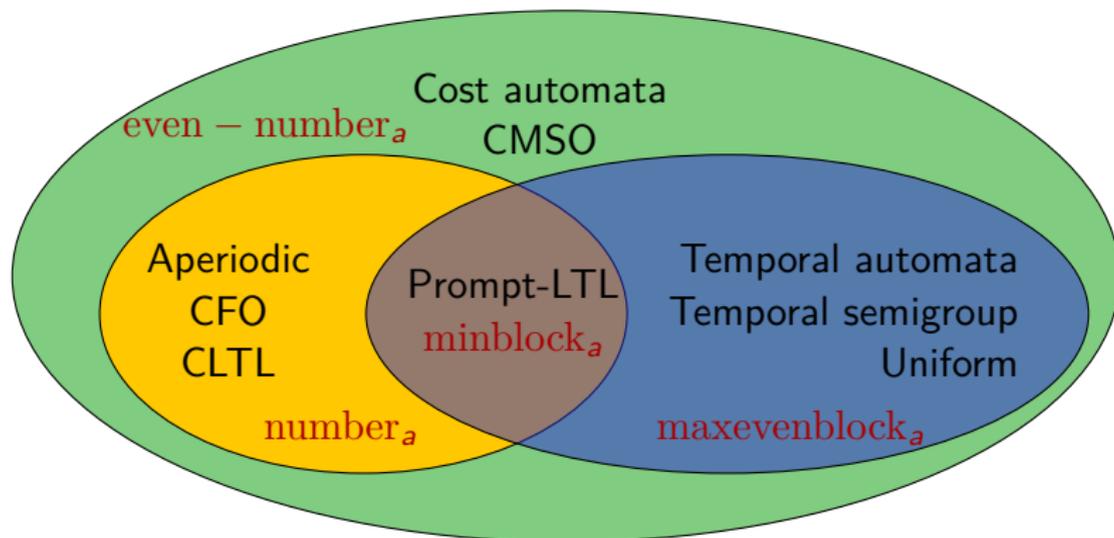
Decidability of boundedness for Quasi-Weak automata. [K.+Vanden Boom, FSTTCS '11].

If \mathcal{A} is a Büchi automaton, it is decidable whether $L(\mathcal{A})$ is weak [submitted to LICS '13].

Logic for the Quasi-Weak class [Submitted to ICALP '13].

- 1 Automata theory
- 2 Regular Cost Functions
- 3 Contributions of the thesis
- 4 Zoom: Aperiodic Cost Functions**

Cost Functions on finite words



Logics on Finite Words

- **First-Order Logic (FO)**: we quantify over positions in the word.

$$\varphi := a(x) \mid x \leq y \mid \neg\varphi \mid \varphi \vee \psi \mid \exists x\varphi$$

Logics on Finite Words

- **First-Order Logic (FO)**: we quantify over positions in the word.

$$\varphi := a(x) \mid x \leq y \mid \neg\varphi \mid \varphi \vee \psi \mid \exists x\varphi$$

- **MSO**: FO with quantification on sets, noted X, Y .

Logics on Finite Words

- **First-Order Logic (FO)**: we quantify over positions in the word.

$$\varphi := a(x) \mid x \leq y \mid \neg\varphi \mid \varphi \vee \psi \mid \exists x\varphi$$

- **MSO**: FO with quantification on sets, noted X, Y .
- **Linear Temporal Logic (LTL)** over \mathbb{A}^* :

$$\varphi := a \mid \Omega \mid \neg\varphi \mid \varphi \vee \psi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U}\psi$$

$$\varphi \mathbf{U}\psi: \quad \begin{array}{cccccccccccc} \varphi & \psi \\ a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} \end{array}$$

Future operators **G** (Always) and **F** (Eventually).

Example: To describe L_{ab} , we can write $\mathbf{F}(a \wedge \mathbf{X}b)$.

Generalisation: cost LTL

- **CLTL** over \mathbb{A}^* :

$$\varphi := a \mid \Omega \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U}\psi \mid \varphi \mathbf{U}^{\leq N}\psi$$

Negations pushed to the leaves.

Generalisation: cost LTL

- **CLTL** over \mathbb{A}^* :

$$\varphi := a \mid \Omega \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U}\psi \mid \varphi \mathbf{U}^{\leq N}\psi$$

Negations pushed to the leaves.

- $\varphi \mathbf{U}^{\leq N}\psi$ means that ψ is true in the future, and φ is false at most N times in the mean time.

$$\varphi \mathbf{U}^{\leq N}\psi: \quad \begin{array}{cccccccccccc} \varphi & \varphi & \times & \varphi & \varphi & \times & \varphi & \varphi & \psi & & & \\ a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} \end{array}$$

Generalisation: cost LTL

- **CLTL** over \mathbb{A}^* :

$$\varphi := a \mid \Omega \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U}\psi \mid \varphi \mathbf{U}^{\leq N}\psi$$

Negations pushed to the leaves.

- $\varphi \mathbf{U}^{\leq N}\psi$ means that ψ is true in the future, and φ is false at most N times in the mean time.

$$\varphi \mathbf{U}^{\leq N}\psi: \quad \begin{array}{cccccccccccc} \varphi & \varphi & \times & \varphi & \varphi & \times & \varphi & \varphi & \psi \\ a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} \end{array}$$

- “Error variable” N is unique, shared by all occurrences of $\mathbf{U}^{\leq N}$.

Generalisation: cost LTL

- **CLTL** over \mathbb{A}^* :

$$\varphi := a \mid \Omega \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U}\psi \mid \varphi \mathbf{U}^{\leq N}\psi$$

Negations pushed to the leaves.

- $\varphi \mathbf{U}^{\leq N}\psi$ means that ψ is true in the future, and φ is false at most N times in the mean time.

$$\varphi \mathbf{U}^{\leq N}\psi: \quad \begin{array}{cccccccccccc} \varphi & \varphi & \times & \varphi & \varphi & \times & \varphi & \varphi & \psi \\ a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} \end{array}$$

- “Error variable” N is unique, shared by all occurrences of $\mathbf{U}^{\leq N}$.
- $\mathbf{G}^{\leq N}\varphi$: φ is false at most N times in the future ($\varphi \mathbf{U}^{\leq N}\Omega$).

Generalisation : Cost FO and Cost MSO

- **CFO** over \mathbb{A}^* :

$$\varphi := a(x) \mid x = y \mid x < y \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \exists x \varphi \mid \forall x \varphi \mid \forall^{\leq N} x \varphi$$

Negations pushed to the leaves.

- As before, N unique free variable.
- $\forall^{\leq N} x \varphi(x)$ means φ is false on at most N positions.
- **CMSO** extends CFO by allowing quantification over sets.

Semantics of Cost Logics

From formula to cost function:

Formula $\varphi \longrightarrow$ cost function $\llbracket \varphi \rrbracket : \mathbb{A}^* \rightarrow \mathbb{N} \cup \{\infty\}$, defined by

$$\llbracket \varphi \rrbracket(u) = \inf\{n \in \mathbb{N} : \varphi \text{ is true over } u \text{ with } n \text{ as error value}\}$$

Example with the alphabet $\{a, b\}$

- $\text{number}_a = \llbracket \mathbf{G}^{\leq N} b \rrbracket = \llbracket \forall^{\leq N} x \ b(x) \rrbracket$.

Semantics of Cost Logics

From formula to cost function:

Formula $\varphi \longrightarrow$ cost function $\llbracket \varphi \rrbracket : \mathbb{A}^* \rightarrow \mathbb{N} \cup \{\infty\}$, defined by

$$\llbracket \varphi \rrbracket(u) = \inf\{n \in \mathbb{N} : \varphi \text{ is true over } u \text{ with } n \text{ as error value}\}$$

Example with the alphabet $\{a, b\}$

- $\text{number}_a = \llbracket \mathbf{G}^{\leq N} b \rrbracket = \llbracket \forall^{\leq N} x \ b(x) \rrbracket$.
- $\text{maxblock}_a = \llbracket \mathbf{G}(\perp \mathbf{U}^{\leq N}(b \vee \Omega)) \rrbracket$
 $= \llbracket \forall X \ \text{block}_a(X) \Rightarrow (\forall^{\leq N} x \ x \notin X) \rrbracket$.

Semantics of Cost Logics

From formula to cost function:

Formula $\varphi \longrightarrow$ cost function $\llbracket \varphi \rrbracket : \mathbb{A}^* \rightarrow \mathbb{N} \cup \{\infty\}$, defined by

$$\llbracket \varphi \rrbracket(u) = \inf\{n \in \mathbb{N} : \varphi \text{ is true over } u \text{ with } n \text{ as error value}\}$$

Example with the alphabet $\{a, b\}$

- $\text{number}_a = \llbracket \mathbf{G}^{\leq N} b \rrbracket = \llbracket \forall^{\leq N} x \ b(x) \rrbracket$.
- $\text{maxblock}_a = \llbracket \mathbf{G}(\perp \mathbf{U}^{\leq N}(b \vee \Omega)) \rrbracket$
 $= \llbracket \forall X \ \text{block}_a(X) \Rightarrow (\forall^{\leq N} x \ x \notin X) \rrbracket$.
- If φ is a classical formula for L , then $\llbracket \varphi \rrbracket = \chi_L$.

Stabilisation monoids

- **Aim:** Generalise monoids to a quantitative setting.

Stabilisation monoids

- **Aim:** Generalise monoids to a quantitative setting.
- Stabilisation $\#$ means “repeat many times” the element.

Stabilisation monoids

- **Aim:** Generalise monoids to a quantitative setting.
- Stabilisation \sharp means “repeat many times” the element.
- if we “count” a , then $a^\sharp \neq a$, otherwise $a^\sharp = a$.

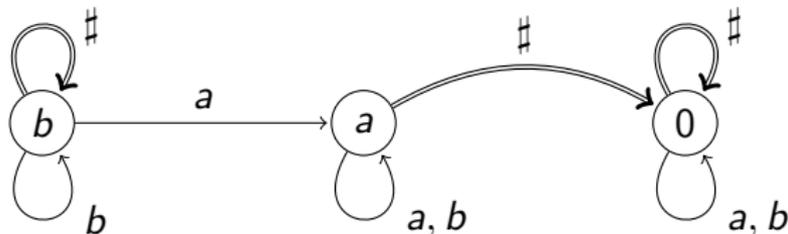
Stabilisation monoids

- **Aim:** Generalise monoids to a quantitative setting.
- Stabilisation \sharp means “repeat many times” the element.
- if we “count” a , then $a^\sharp \neq a$, otherwise $a^\sharp = a$.

Example: Stabilisation Monoid for number a

$M = \{b, a, 0\}$, $P = \{a, b\}$,

b : “no a ”, a : “a little number of a ”, 0 : “a lot of a ”.



Cayley graph

Aperiodic Monoids

Definition: A [stabilisation] monoid M is **aperiodic** if for all $x \in M$ there is $n \in \mathbb{N}$ such that $x^n = x^{n+1}$.

Aperiodic Monoids

Definition: A [stabilisation] monoid M is **aperiodic** if for all $x \in M$ there is $n \in \mathbb{N}$ such that $x^n = x^{n+1}$.

Theorem (McNaughton-Papert, Schützenberger, Kamp)

Aperiodic Monoids \Leftrightarrow *FO* \Leftrightarrow *LTL* \Leftrightarrow *Star-free Expressions*.

We want to generalise this theorem to cost functions.

The problems are:

- No complementation \Rightarrow No Star-free expressions.
- Deterministic automata are strictly weaker.
- Heavy formalisms (semantics of stabilisation monoids).
- New quantitative behaviours.
- Original proofs already hard.

Aperiodic cost functions

Theorem (K. STACS 2011)

Aperiodic stabilisation monoid \Leftrightarrow CLTL \Leftrightarrow CFO.

Proof Ideas:

- Generalisation of Myhill-Nerode \Rightarrow Syntactic object.
- Induction on $(|M|, |\mathbb{A}|)$.
- Extend functions to sequences of words.
- Use bounded approximations.
- Extend CLTL with Past operators, show Separability.

