The theory of regular cost functions.

Denis Kuperberg
PhD under supervision of Thomas Colcombet

Hebrew University of Jerusalem

ERC Workshop on Quantitative Formal Methods
Jerusalem, 10-05-2013
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 yielded 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
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: \( A^* ab A^* \),
- Logical sentence (MSO): \( \exists x \ \exists y \ a(x) \land b(y) \land (y = Sx) \).
Descriptions of a language

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

Other ways than automata to specify \( L_{ab} \):

- Regular expression: \( A^* ab A^* \),
- Logical sentence (MSO): \( \exists x \ \exists y \ a(x) \land b(y) \land (y = Sx) \).
- Finite monoid: \( M = \{ 1, a, b, c, ba, 0 \}, \ P = \{ 0 \} \)
  \( ab = 0, \ aa = ca = a, \ bb = bc = b, \ cc = ac = cb = c \)
All these formalisms are effectively equivalent.
All these formalisms are effectively equivalent.
Historical motivation

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

**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 $C$, is there an algorithm which given an automaton for $L$, decides whether $L \in C$?

**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**

- **$A_1$: number of $a$**

- **$A_2$: smallest block of $a$**

**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 \( \{ i, i+1, \ldots, j \} \)?
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 $\varepsilon$)

Semantics: $[\mathcal{A}] : \Sigma^* \rightarrow \mathbb{N} \cup \{\infty\}$
Cost automata over words

Nondeterministic finite-state automaton $A$
+ finite set of counters
  (initialized to 0, values range over $\mathbb{N}$)
+ counter operations on transitions
  (increment $I$, reset $R$, check $C$, no change $\varepsilon$)

Semantics: $[A] : \Sigma^* \rightarrow \mathbb{N} \cup \{\infty\}$

$\text{val}_B(\rho) := \max$ checked counter value during run $\rho$

$[A]_B(u) := \min \{\text{val}_B(\rho) : \rho \text{ is an accepting run of } A \text{ on } u\}$

Example

$[A]_B(u) = \min$ length of block of $a$’s surrounded by $b$’s in $u$

\begin{tikzpicture}
  \node[state] (q0) at (0,0) {$a,b: \varepsilon$};
  \node[state] (q1) at (2,0) {$a: \text{IC}$};
  \node[state] (q2) at (4,0) {$a,b: \varepsilon$};
  \node[state] (q3) at (0,-2) {$b: \varepsilon$};
  \node[state] (q4) at (2,-2) {$b: \varepsilon$};
  \node[state] (q5) at (4,-2) {$b: \varepsilon$};

  \draw (q0) edge [loop above] node {$a,b: \varepsilon$} (q0);
  \draw (q0) edge [loop below] node {$b: \varepsilon$} (q0);
  \draw (q0) edge node {$a, b: \varepsilon$} (q1);
  \draw (q1) edge [loop above] node {$a: \text{IC}$} (q1);
  \draw (q1) edge node [above] {$a, b: \varepsilon$} (q2);
  \draw (q2) edge [loop above] node {$a,b: \varepsilon$} (q2);
  \draw (q2) edge node [below] {$b: \varepsilon$} (q3);
  \draw (q3) edge [loop above] node {$b: \varepsilon$} (q3);
  \draw (q3) edge node [below] {$b: \varepsilon$} (q4);
  \draw (q4) edge [loop above] node {$b: \varepsilon$} (q4);
  \draw (q4) edge node [below] {$b: \varepsilon$} (q5);
  \draw (q5) edge [loop above] node {$b: \varepsilon$} (q5);
\end{tikzpicture}
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 $\varepsilon$)

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

$\text{val}_S(\rho) := \min \text{ checked counter value during run } \rho$

$[\mathcal{A}]_S(u) := \max\{\text{val}_S(\rho) : \rho \text{ is an accepting run of } \mathcal{A} \text{ on } u\}$

Example

$[\mathcal{A}]_S(u) = \min \text{ length of block of } a\text{'s surrounded by } b\text{'s in } u$

\[
\begin{array}{c}
\mathcal{A} = \begin{array}{c}
\text{a:}\varepsilon \\
\text{b:}\varepsilon \\
\text{a:}I \\
\text{b:}CR
\end{array}
\end{array}
\]
Boundedness relation

“$[A]_B = [B]_B$”: undecidable [Krob ’94]
Boundedness relation

“$[A]_B = [B]_B$”: undecidable [Krob ’94]

“$[A]_B \approx [B]_B$”: decidable on words
[Colcombet ’09, following Bojánctyk+Colcombet ’06]
for all subsets $U$, $[A](U)$ bounded iff $[B](U)$ bounded
Boundedness relation

“$[A]_B = [B]_B$”: undecidable [Krob ’94]

“$[A]_B \approx [B]_B$”: decidable on words
[Colcombet ’09, following Bojánczyk+Colcombet ’06]
for all subsets $U$, $[A](U)$ bounded iff $[B](U)$ bounded

\[ [A] \not\approx [B] \]
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 $\iff$ Cost logics $\iff$ 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 $A$ is a classical automaton for $L$, then $[,A]_B = \chi_L$ and $[,A]_S = \chi_L$. 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...

Infinite trees: a

b

b

b

Infinite trees: a

b

b

b

a

a

a

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.
Contributions of the thesis

Input structures:

Finite words: $accba$

Infinite words: $abaabaccbaba \ldots$

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

Regular Functions

- CMSO
- WCMSO

Aperiodic Functions

- Very-Weak Automata
- CFO
- CLTL

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 $⇔$
$L$ recognizable by WMSO $⇔$ there are Büchi automata $U$ and $U'$ such that $L = L(U) = L(U')$. 
Cost functions on infinite trees

Decidability of boundedness for Quasi-Weak automata.\cite{K.+Vanden Boom, FSTTCS '11}.
If $\mathcal{A}$ is a Büchi automaton, it is decidable whether $L(\mathcal{A})$ is weak \cite{submitted to CSL '13}.
Logic for the Quasi-Weak class.
1 Automata theory

2 Regular Cost Functions

3 Contributions of the thesis

4 Zoom: Aperiodic Cost Functions
Cost Functions on finite words

- Cost automata
- CMSO
- Aperiodic
- CFO
- CLTL
- Temporal automata
- Temporal semigroup
- Uniform

even \( - \) number\(_a\)

Prompt-LTL

minblock\(_a\)

number\(_a\)

maxevenblock\(_a\)
Logics on Finite Words

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

\[ \varphi ::= a(x) \mid x \leq y \mid \neg \varphi \mid \varphi \lor \psi \mid \exists x \varphi \]
Logics on Finite Words

- **First-Order Logic (FO):** we quantify over positions in the word.
  \[ \varphi ::= a(x) \mid x \leq y \mid \neg \varphi \mid \varphi \lor \psi \mid \exists x \varphi \]

- **MSO:** FO with quantification on sets, noted \( X, Y \).
Logics on Finite Words

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

  \[ \varphi := a(x) \mid x \leq y \mid \neg \varphi \mid \varphi \lor \psi \mid \exists x \varphi \]

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

- **Linear Temporal Logic (LTL) over \( \mathbb{A}^* \):**

  \[ \varphi := a \mid \Omega \mid \neg \varphi \mid \varphi \lor \psi \mid X \varphi \mid \varphi U \psi \]

  \[ \varphi U \psi : \quad a_0 a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_9 a_{10} \]

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

**Example:** To describe \( L_{ab} \), we can write \( F(a \land Xb) \).
Generalisation: cost LTL

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

  $$\varphi ::= a \mid \Omega \mid \varphi \land \psi \mid \varphi \lor \psi \mid X\varphi \mid \varphi U\psi \mid \varphi U^{\leq N}\psi$$

  Negations pushed to the leaves.
Generalisation: cost LTL

- **CLTL** over $A^*$:

$$\varphi := a \mid \Omega \mid \varphi \land \psi \mid \varphi \lor \psi \mid X\varphi \mid \varphi U\psi \mid \varphi U^\leq N \psi$$

Negations pushed to the leaves.

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

$$\varphi U^\leq N \psi: \quad \varphi \varphi \times \varphi \varphi \times \varphi \varphi \psi$$
$$\quad a_0 a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_9 a_{10}$$
Generalisation: cost LTL

- **CLTL** over $A^*$:

  $$\phi ::= a \mid \Omega \mid \phi \land \psi \mid \phi \lor \psi \mid X\phi \mid \phi U\psi \mid \phi U^{\leq N} \psi$$

  Negations pushed to the leaves.

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

  $$\phi U^{\leq N} \psi: \quad \phi \quad \times \quad \phi \quad \times \quad \phi \quad \times \quad \phi \quad \times \psi \quad a_0 \quad a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad a_6 \quad a_7 \quad a_8 \quad a_9 a_{10}$$

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

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

$$\varphi := a \mid \Omega \mid \varphi \land \psi \mid \varphi \lor \psi \mid X\varphi \mid \varphi U\psi \mid \varphi U\leq N\psi$$

Negations pushed to the leaves.

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

$$\varphi U\leq N\psi: \quad \varphi \varphi \times \varphi \varphi \times \varphi \varphi \psi$$

$$a_0 a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_9 a_{10}$$

- “Error variable” $N$ is unique, shared by all occurrences of $U\leq N$.
- $G\leq N\varphi$: $\varphi$ is false at most $N$ times in the future ($\varphi U\leq N\Omega$).
Generalisation: Cost FO and Cost MSO

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

  $$\varphi ::= a(x) \mid x = y \mid x < y \mid \varphi \land \varphi \mid \varphi \lor \varphi \mid \exists x \varphi \mid \forall x \varphi \mid \forall \leq N x \varphi$$

  Negations pushed to the leaves.

- As before, $N$ unique free variable.

- $\forall \leq N x \varphi(x)$ means $\varphi$ is false on at most $N$ positions.

- **CMSO** extends CFO by allowing quantification over sets.
From formula to cost function:
Formula $\varphi \rightarrow$ 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 $\{a, b\}$

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

From formula to cost function:
Formula \( \varphi \) \( \rightarrow \) cost function \([\varphi] : \mathbb{A}^* \rightarrow \mathbb{N} \cup \{\infty\}\), defined by

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

Example with the alphabet \{a, b\}

- number\(_a\) = \([G \leq N b] = [\forall \leq N x \ b(x)]\).

- maxblock\(_a\) = \([G (\bot \ U \leq N (b \lor \Omega))] = [\forall X \ \text{block}\(_a\)(X) \Rightarrow (\forall \leq N x \ x \notin X)]\).
Semantics of Cost Logics

From formula to cost function:
Formula $\varphi \rightarrow$ cost function $[[\varphi]] : \mathbb{A}^* \rightarrow \mathbb{N} \cup \{\infty\}$, defined by

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

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

- **number\(_a\) = $[[G^{\leq N}b]] = [[[\forall^{\leq N}x \ b(x)]]].$**
- **maxblock\(_a\) = $[[G(\bot U^{\leq N}(b \vee \Omega))]$**
  $$= [[[\forall X \ block\(_a\)(X) \Rightarrow (\forall^{\leq N}x \ x \notin X)]]].$$
- If $\varphi$ is a classical formula for $L$, then $[[\varphi]] = \chi_L.$
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 $\#$ means “repeat many times” the element.
- if we “count” $a$, then $a^\# \neq a$, otherwise $a^\# = a$. 
Stabilisation monoids

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

**Example:** Stabilisation Monoid for number $a$

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

$b$: “no $a$”, $a$: “a little number of $a$”, $\bot$: “a lot of $a$”.

Cayley graph
**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)**

\[ \text{Aperiodic stabilisation monoid} \iff \text{CLTL} \iff \text{CFO.} \]

**Proof Ideas:**

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