Regular temporal cost functions

Denis Kuperberg[1]
Joint work with Thomas Colcombet[1] and Sylvain Lombardy[2]

Liafa/CNRS/Université Paris 7, Denis Diderot, France

Ligm - Université Paris-Est Marne-la-Vallée, France

ICALP 2010

Introduction

- Cost function : counting extension of languages

Introduction

- Cost function : counting extension of languages
- Temporal class : measuring time

Introduction

- Cost function : counting extension of languages
- Temporal class : measuring time
- Simplify constructions and lower complexity of algorithms for this class

Introduction

- Cost function : counting extension of languages
- Temporal class : measuring time
- Simplify constructions and lower complexity of algorithms for this class
- Algebraic characterization of temporal cost functions

Outline

Cost automata

Aim : To represent functions $\mathbb{A}^* \longrightarrow \mathbb{N} \cup \{\infty\}$ with automata.

Cost automaton :

- nondeterministic finite-state
- finite set of counters, ranging over $\mathbb{N}$, initial value 0
- each transition perform actions on each counter

Atomic actions : increment (i), reset (r), check (c).

Semantics of cost automata

Two semantics :

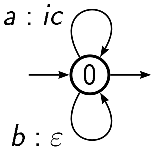
$$[\![\mathcal{A}]\!]_B(u) = \inf\{n/ \text{ there is a run with maximal check value } n\},$$

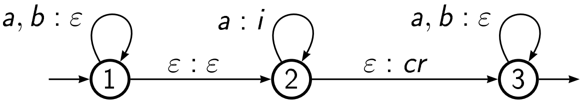
and $$[\![\mathcal{A}]\!]_S(u) = \sup\{n/ \text{ there is a run with minimal check value } n\} \ .$$

with $\inf \emptyset = \infty$ and $\sup \emptyset = 0$.

Example

$[\![\mathcal{A}]\!]_B = |\cdot|_a$ *and* $[\![\mathcal{A}']\!]_S = \mathrm{maxblock}_a : u \mapsto \max\{n/a^n \textit{ factor of } u\}$





More on cost automata

Remark : A standard automaton $\mathcal{A}$ computing L can be viewed as a cost automaton without any counter.
Then $[\![\mathcal{A}]\!]_B = \chi_L$ and $[\![\mathcal{A}]\!]_S = \chi_{\mathbb{A}^*\setminus L}$ with
$$\chi_L(u) = \begin{cases} 0 & \text{if } u \in L \\ \infty & \text{if } u \notin L \end{cases}$$

Theorem ([Krob 94])

The equivalence of two cost automata is undecidable.

Solution : Loosing some precision on the counting, but keeping information about bounds.

Cost functions

If $f, g : \mathbb{A}^* \to \mathbb{N} \cup \{\infty\}$, then

$$f \approx g \text{ if } \forall X \subseteq \mathbb{A}^*, f|_X \text{ bounded } \Leftrightarrow g|_X \text{ bounded.}$$

iff $\exists\, \alpha : \mathbb{N} \to \mathbb{N}$ such as $f \leq \alpha \circ g$ and $g \leq \alpha \circ f$ (with $\alpha(\infty) = \infty$)

Cost function : equivalence class for $\approx$ relation.

Example

For $\mathbb{A} = \{a, b, c\}$,

$\max(|\cdot|_a, |\cdot|_b) \approx |\cdot|_a + |\cdot|_b$ *but* $|\cdot|_a \not\approx \mathrm{maxblock}_a$

Known results on cost automata

Extension of the notion of language via $\chi_L : L \neq L' \implies \chi_L \not\approx \chi_{L'}$.

Theorem (Colcombet 09)

B- and S-automata have the same expressive power (modulo $\approx$), and the translations are effective.

Theorem (Colcombet 09)

It is decidable whether two cost automata compute the same cost function (modulo $\approx$).

Automata-computable cost functions are called regular.

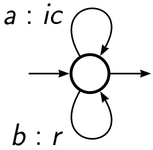
Temporal automata

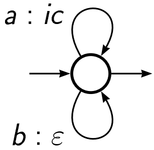
Intuitive idea : Measuring the time, counting consecutive events.

Temporal automata : Only actions : $\{ic, r\}$ for B-automata (Kirsten and Bala's *desert automata*) and $\{i, cr\}$ for S-automata.

Example

For $\mathbb{A} = \{a, b\}$, $\mathrm{maxblock}_a$ *is temporal, but* $|\cdot|_a$ *is not.*





Temporal automata

Intuitive idea : Measuring the time, counting consecutive events.

Temporal automata : Only actions : $\{ic, r\}$ for B-automata (Kirsten and Bala's *desert automata*) and $\{i, cr\}$ for S-automata.

Example

For $\mathbb{A} = \{a, b\}$, $\mathrm{maxblock}_a$ *is temporal, but* $|\cdot|_a$ *is not.*

Theorem

For a cost function, it is equivalent (modulo $\approx$*) to be recognized by*

- *temporal* B*-automaton*
- *temporal* B*-automaton with* 1 *counter*
- *temporal* S*-automaton*
- *temporal* S*-automaton with* 1 *counter*

We say then that the cost function is temporal.

A ticking clock to measure time

Problem : Two different formalisms, inefficient constructions in the general case

Idea : Using an automaton reading simultaneously a word and a "clock" to measure the time

Clock on u : Word on $\{w, t\}$ (for *"wait"* and *"tick"*) of length $|u|$.

Example of $\mathrm{maxblock}_a$

Call a clock good for u if each block of a's in u contains at most one *tick* of c.

Principle Minimal maxperiod of all good clocks for u and maximal minperiod of all bad clocks for u are both equivalent (modulo $\approx$) to $\mathrm{maxblock}_a(u)$.

t t t t $\longleftrightarrow$ t t t t

b a b b a a a a a a a a b a a b

t $\longleftrightarrow$ t

Representing temporal cost functions with regular languages

Definition

L regular on alphabet $\mathbb{A} \times \{w, t\}$ is a clock-language for f iff

$$\begin{aligned} \langle\langle L \rangle\rangle_B &: u \mapsto \inf\{maxperiod(c) \ : \ \langle u, c\rangle \in L\} \\ \langle\langle L \rangle\rangle_S &: u \mapsto \sup\{minperiod(c) \ : \ \langle u, c\rangle \notin L\} \end{aligned}$$

are both equivalent to f modulo $\approx$.

Theorem

A cost function f is temporal iff there exists a clock-language for f.

This give us easily closure of temporal class under min, max, projections,...

Algebraic charcaterization of regular cost functions

Reminder : regular language $\Leftrightarrow$ finite semigroup (Myhill)

Stabilization semigroup : $\mathbf{S} = \langle S, \cdot, \leq, \sharp \rangle$, ordered semigroup with a $\sharp$-operator : stabilization over idempotents ($e = e \cdot e$).

$e^\sharp$ means "e repeated a lot of times".

Theorem (Colcombet 09)

Stabilization semigroups recognize exactly the set of regular cost functions, and translations to or from cost automata are effective.

Temporal semigroups

Reminder : Star-free language $\Leftrightarrow$ group-trivial semigroup (Schützenberger)

Theorem

A cost function is temporal iff it is recognizable by a temporal stabilization semigroup (effective structural condition).

Temporal semigroups

Reminder : Star-free language $\Leftrightarrow$ group-trivial semigroup (Schützenberger)

Theorem

A cost function is temporal iff it is recognizable by a temporal stabilization semigroup (effective structural condition).

Theorem

Let f be a regular cost function,

- *There exists a (quotient-wise) minimal stabilization semigroup* $\mathbf{S}$ *recognizing f*
- $\mathbf{S}$ *is computable effectively*
- *f is temporal iff* $\mathbf{S}$ *is temporal*

Temporal semigroups

Reminder : Star-free language $\Leftrightarrow$ group-trivial semigroup (Schützenberger)

Theorem

A cost function is temporal iff it is recognizable by a temporal stabilization semigroup (effective structural condition).

Theorem

Let f be a regular cost function,

- *There exists a (quotient-wise) minimal stabilization semigroup* $\mathbf{S}$ *recognizing f*
- $\mathbf{S}$ *is computable effectively*
- *f is temporal iff* $\mathbf{S}$ *is temporal*

Corollary

It is decidable whether a regular cost function is temporal.

Conclusion

Summary :

- Temporal class defined via cost automata
- Simplifications of constructions in this class, via clock-languages
- Characterization by stabilization semigroups
- Decidability of the temporal class

Further work :

- Extension to infinite words, trees
- Other classes of regular cost functions