Cost Functions and Value 1 problem in practice

Nathanaël Fijalkow, Denis Kuperberg

LIAFA/Institute of Informatics, Warsaw University, Poland
ONERA/IRIT, Toulouse

December 18th, 2014
An algorithmic back-end: Stabilization Monoids

**What?**
- An algebraic structure with two operations: a binary composition and a unary operator $\#$,
- Generalizes the transition monoid of a non-deterministic automaton to two weighted settings.

**Where? When?**
- First appeared in the Theory of Regular Cost Functions [Colcombet 2009],
- Later used for Probabilistic Automata [Fijalkow, Gimbert, Oualhadj 2012].
Regular Cost functions

\textbf{B-automata}

- Non-deterministic, and are enriched with \textit{counters} with operations: \(+1, r\)
- Introduced to generalize proofs on regular languages: finite substitution, star-height.
- Semantic is a function: \(A^* \rightarrow \mathbb{N} \cup \{\infty\}\),

\textbf{Questions on B-automata}

- \textbf{Equivalence}: bounded on same sets of words,
- \textbf{Boundedness}: equivalent to 0 (bounded).
Examples of $B$-Automata

$A_1$: number of $a$

$A_2$: smallest block of $a$

Not equivalent: $A_1$ is not bounded on $(ab)^*$ but $A_2$ is.
Problem: Given $L$, is there $n \in \mathbb{N}$ such that $L^\ast = (L + \epsilon)^n$?
Problem: Given $L$, is there $n \in \mathbb{N}$ such that $L^* = (L + \epsilon)^n$?

Solution:

1. Start with an automaton for $L$. 
Reductions from Finite Power to Boundedness

**Problem:** Given $L$, is there $n \in \mathbb{N}$ such that $L^* = (L + \varepsilon)^n$?

**Solution:**

1. Start with an automaton for $L$.
2. Add increment $\varepsilon$-transitions from final states to initial.
Reductions from Finite Power to Boundedness

**Problem:** Given $L$, is there $n \in \mathbb{N}$ such that $L^* = (L + \epsilon)^n$?

**Solution:**

1. Start with an automaton for $L$.
2. Add increment $\epsilon$-transitions from final states to initial.
3. Decide boundedness
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]
  Bound on 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, deterministic input
  Niwinski+Walukiewicz ’05, Büchi input K.+Vanden Boom ’11 CKLV ’13]
  Given \( i < j \), is there a parity automaton for \( L \) using ranks \( \{i, i + 1, \ldots, j\} \)?
Problem for probabilistic Automata

Probabilistic automata

- Transitions are distributions $\delta(q, a) : Q \rightarrow [0, 1]$,
- $P_A(w)$ is the probability that $A$ accepts $w$.

Value 1 problem: Given $A$, is there a sequence $w_n$ such that $P_A(w_n) \rightarrow 1$?
Problem for probabilistic Automata

Probabilistic automata

- Transitions are distributions $\delta(q, a) : Q \rightarrow [0, 1]$,
- $P_A(w)$ is the probability that $A$ accepts $w$.

**Value 1 problem:** Given $A$, is there a sequence $w_n$ such that $P_A(w_n) \rightarrow 1$?

Undecidable, but decidable in a restricted case: leaktight automata.

**Intuition:** leaks allow ”competitive” behaviours, outcome depends on fine tuning of transitions.
Matrix Representation

\[ \langle a \rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \langle b \rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \]

\[ I \cdot \langle u \rangle \cdot F = 1 \quad \text{if and only if} \quad u \text{ is accepted.} \]
Probabilistic automata

\[ \langle a \rangle = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \langle b \rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0.7 & 0 & 0.3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \]

\[ I \cdot \langle u \rangle \cdot F = P_A(u) \]
\[ B\text{-automata} \]

\[
\begin{align*}
\langle a \rangle &= \begin{pmatrix}
0 & \bot & \bot & \bot \\
\bot & 1 & r & \bot \\
\bot & \bot & 0 & \bot \\
\bot & \bot & \bot & 0
\end{pmatrix} \quad \quad \langle b \rangle &= \begin{pmatrix}
0 & \bot & \bot & \bot \\
\bot & 1 & \bot & \bot \\
\bot & \bot & r & \bot \\
\bot & \bot & \bot & 0
\end{pmatrix} \\
I \cdot \langle u \rangle \cdot F &= f(u)
\end{align*}
\]
Consider either the rational semiring \((\mathbb{Q}, +, \times)\) or the tropical semiring \((\mathbb{N} \cup \{\infty\}, \min, +)\):

- An automaton \(A\) is given by a matrix \(\langle a \rangle\) for each letter \(a \in A\),
- We would like to finitely represent \(\{\langle u \rangle \mid u \in A^*\}\).
Computing in Infinite Semirings

Consider either the rational semiring \((\mathbb{Q}, +, \times)\) or the tropical semiring \((\mathbb{N} \cup \{\infty\}, \min, +)\):

- An automaton \(A\) is given by a matrix \(\langle a \rangle\) for each letter \(a \in A\),
- We would like to finitely represent \(\{\langle u \rangle \mid u \in A^*\}\).

So we abstract away the precise values and consider two operators:

- a binary composition law: matrix multiplication,
- a stabilization unary operator \(\sharp\).
Consider either the rational semiring \((\mathbb{Q}, +, \times)\) or the tropical semiring \((\mathbb{N} \cup \{\infty\}, \min, +)\):

- An automaton \(A\) is given by a matrix \(\langle a \rangle\) for each letter \(a \in A\),
- We would like to finitely represent \(\{\langle u \rangle \mid u \in A^*\}\).

So we abstract away the precise values and consider two operators:

- a binary composition law: matrix multiplication,
- a stabilization unary operator \(\sharp\).

Intuitively, \(\langle u \rangle^\sharp\) represents \(\lim_n \langle u^n \rangle\).
Stabilization operations

Only defined on idempotents.

\textbf{B-automata:}
Replace $+1$ by $\omega$ in the matrix, meaning “unbounded”.

\textbf{Probabilisatic Automata:}
Model long runs $\rightarrow$ Keep only edges going to ergodic strongly connected components.

No reduction from one framework to the other on the automata level $\rightarrow$ versatility of stabilization monoids.
Example of computation

$A$: $a, b \xrightarrow{b} a : +1 \xrightarrow{b} a, b$

$\mathcal{A}^\#$: Not accepted

$ba^\#b$: Accepted, but not within some bound.
Formally, a stabilization monoid is $(M, \cdot, \# , \leq)$ such that:

- $(M, \cdot, \leq)$ is an ordered monoid (associativity, monotonicity of $\cdot$),
- $\#$ is a function from idempotents to idempotents,
- $x \leq x^#$,
- $x^# = xx^# = (x^#)^#$.

The order puts a constraint on accepting sets: it has to be upwards-closed.
Stabilization Monoid of an Automaton

Definition

The Stabilization Monoid of $A$ is the closure of $\{\langle a \rangle \mid a \in A\}$ under both operators.

The Stabilization Monoid of $A$ contains a lot of informations about $A$!
Using the Stabilization Monoid

**B-Automata**
- Decide whether a $B$-automaton is bounded,
- Decide whether two $B$-automata are equivalent.

**Probabilistic Automata**
- Decide whether a probabilistic automaton has (probably) value 1,
- Decide whether a probabilistic automaton is leaktight.
Algorithmics elements of ACME

Saturation
- Hashtable for elements obtained, and queue for new candidates.
- Worst-case complexity EXPTIME.
- Current improvements: keep short names for elements and rewriting rules.

Minimization
- needed for equivalence checking,
- general Myhill-Nerode Equivalence, needs a new operator $\omega#$.
- Union-find structure for partitions,
- Computation of abstract types,
- Polynomial in $|M|$ (exponential in $|A|$).

Currently migrating from OCaml to C++, with new optimizations.
Demo of ACME

[Demo of ACME]
The end.

Thank you for your attention!