Varieties of cost functions: results and open problems

L. Daviaud[1], D. Kuperberg[2], J.-É. Pin[3]

Partially funded by the ERC under the European Union's Horizon 2020 research and innovation programme (grant agreement No 670624) and by the DeLTA project (ANR-16-CE40-0007).

[1]University of Warwick
[2]ENS Lyon, CNRS
[3]IRIF, CNRS and University Paris Diderot

September 2018, Berlin

Cost functions (Colcombet)

Let $f, g : A^* \to \mathbb{N} \cup \{\infty\}$ be functions. Then $f \approx g$ if, for each subset E of A^*,

f restricted to E is bounded

if and only if g restricted to E is bounded

A cost function is an equivalence class for $\approx$.

Let $f(u) = |u|_a$, $g(u) = 2|u|_a$ and $h(u) = |u|$. Then $f \approx g$ and they represent the same cost function. However, $f \not\approx h$ since, on b^*, $f(b^n) = 0$ and $h(b^n) = n$.

Examples of cost functions

Let $A = \{a, b\}$. Then, for every fixed k,

$$u \to |u|_a \approx u \to |u|_a^k$$
$$\max\{|u|_a, |u|_b\} \approx |u| \approx k \max\{|u|_a, |u|_b\} \approx |u|^k.$$

The cost function $\mathrm{minblock}_a$ on the alphabet $\{a, b\}$ is defined by:

$$\mathrm{minblock}_a(a^{n_1} b a^{n_2} b \cdots b a^{n_k}) = \min(n_1, n_2, \ldots, n_k)$$

Nice property: if $f \approx f'$ and $g \approx g'$, then $\min(f, g) \approx \min(f', g')$, $\max(f, g) \approx \max(f', g')$.

Languages versus cost functions

The characteristic function of a language L on A^* is the function $\chi_L : A^* \to \mathbb{N} \cup \{\infty\}$ defined by

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

Observe that $\chi_L \approx \chi_{L'}$ iff $L = L'$.

Thus one can identify a language with the cost function defined by its characteristic function.

A summary of known results (2009)

Theorem (Colcombet, Bojańczyk, Kuperberg)

Everything available for regular languages (automata, syntactic monoids, rational expressions, logical characterizations) can be lifted to regular cost functions. The equivalences between these models are effective.

Theorem (Colcombet)

The equality of two regular cost functions is decidable.

Stabilisation monoids 1.0 (Colcombet '09)

A stabilisation monoid is a finite ordered monoid $(M, \leqslant)$ with a stabilisation operation $\sharp\colon E(M) \to E(M)$ satisfying:

(1) for all $s, t \in M$ such that $st \in E(M)$ and $ts \in E(M)$, $(st)^\sharp s = s(ts)^\sharp$,

(2) for all $e \in E(M)$, $(e^\sharp)^\sharp = e^\sharp$ and $e^\sharp \leqslant e$,

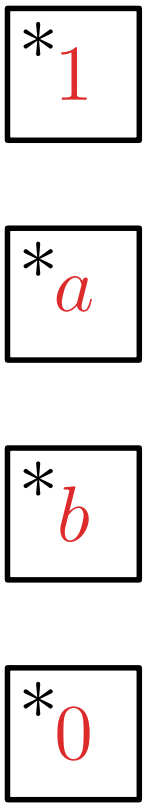
(3) for all $e, f \in E(M)$, $e \leqslant f$ implies $e^\sharp \leqslant f^\sharp$,

(4) $1^\sharp = 1$.

Consequences:

for all $e \in E(M)$, $(e^\sharp)^\sharp = e^\sharp e^\sharp = e^\sharp = e e^\sharp = e^\sharp e \leqslant e$,

A stabilisation monoid for $\mathrm{maxblock}_a$



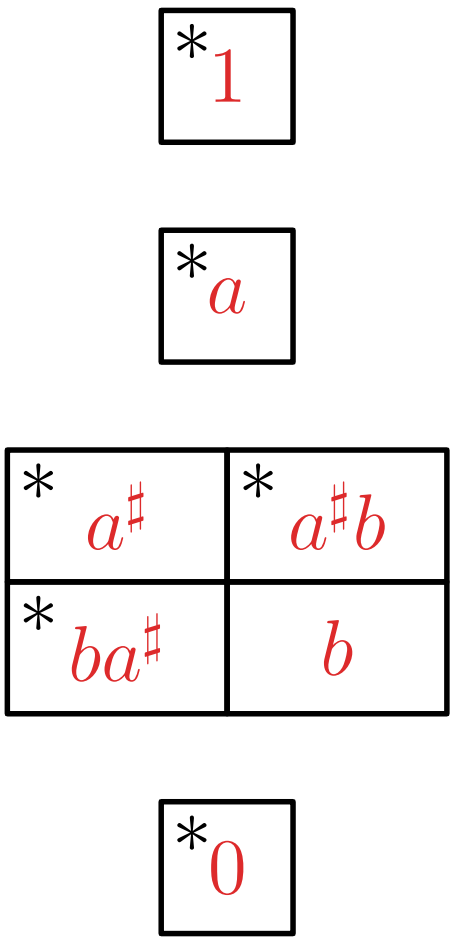
Relations

$$ab = ba = b$$

$$a^\sharp = 0$$
$$b^\sharp = b$$
$$0^\sharp = 0$$

$$0 < a < 1 < b$$

A stabilisation monoid for $\mathrm{minblock}_a$



Relations

$$ab = ba = b$$

$$(a^\sharp b)^\sharp = a^\sharp b$$
$$(ba^\sharp)^\sharp = ba^\sharp$$

$$a^\sharp \leqslant a$$
$$a^\sharp b \leqslant b$$
$$ba^\sharp \leqslant b \leqslant 0$$

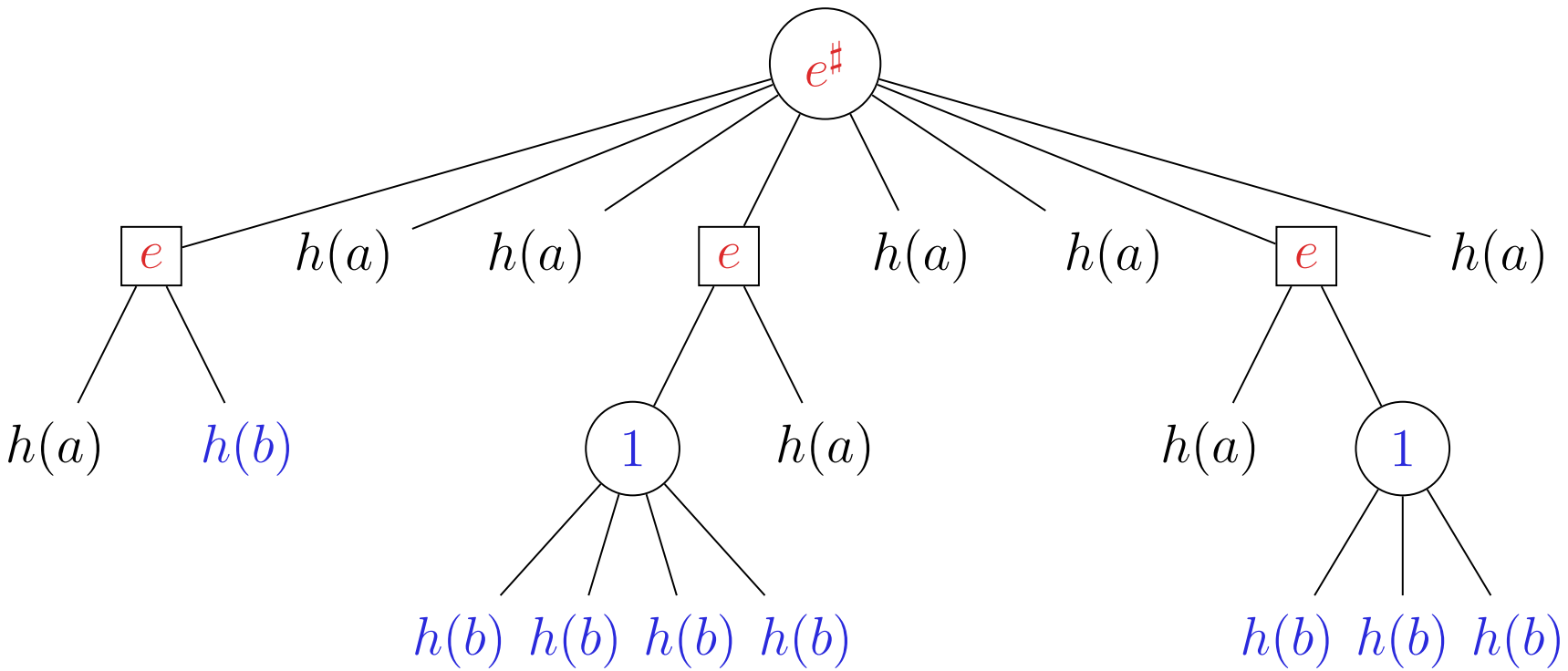
Factorisation trees

Let M be a stabilisation monoid and let $h : A \to M$ be a function. An h-factorisation tree of threshold n for a word $w = a_1 a_2 \cdots a_k$ is a finite tree labelled by the elements of M and such that:

- the tree has exactly k leaves, labelled by $h(a_1), \ldots, h(a_k)$, respectively,
- each binary node is labelled by the product of its left child's label by its right child's label,
- if a node has arity > 2, then all its children are labelled by the same idempotent e. If the arity of the node is $\leqslant n$, then the node is labelled by e, otherwise it is labelled by $e^\sharp$.

Example of factorisation tree

Let $M = \{1, e, 0\}$ with $0 < e < 1$, $0^\sharp = e^\sharp = 0$. Let $h(a) = e$ and $h(b) = 1$. Here is a factorisation tree of threshold 5 for the word $abaabbbbaaaabbba$:



A variant of Simon's factorisation theorem

A variant of Simon's factorisation theorem [Simon90, Colcombet09] guarantees the existence of trees of bounded height to evaluate input words.

More precisely, for each labelling function $h : A \to M$, there is a positive integer K $(= 3|M|)$ such that for all words w and for all integers $n \geqslant 3$, there is an h-factorisation tree of threshold n for w with height at most K.

Cost functions recognized by a stabilisation monoid

Let M be a finite stabilisation monoid, $h : A \to M$ a labelling map and let I be a downset of M.

The cost function recognised by (M, h, I) is the equivalence class of the function that maps a word u to the maximal threshold n such that there exists an h-factorisation tree for u of threshold n, height $\leqslant 3|M|$ and root in I.

Such an equivalence class is called a regular cost function.

Example

The previous tree of height 4 and threshold 5 has root labelled by $e^\sharp = 0$ because it is a witness that $|u|_a \geqslant 5$.

Conversely, such a factorisation tree of threshold n and height k would have its root labelled by 1 if $|u|_a = 0$, and labelled by e if $|u|_a \leqslant n^k$.

Because k is a fixed constant, these trees can be used to recognise the cost function $|u|_a$, since it is equivalent to $(|u|_a)^k$.

Problems towards an Eilenberg's theorem

Problem 1: No free stabilisation monoid.

Solution: replace stabilisation monoids by stabilisation algebras.

- Same behaviour for finite structures,
- Free stabilisation algebras are well defined.

Problem 2: What is the connection between regular cost functions and free stabilisation algebras?

Solution: boundedness is well-defined.

Stabilisation algebra

A stabilisation algebra is an ordered algebra M with signature $\langle 1, \leqslant, \cdot, \omega, \sharp \rangle$ satisfying the following axioms:

- all identities that are satisfied by all finite stabilisation monoids,
- a description of the behaviour of ω on idempotent elements: $x^2 = x$ implies $x^\omega = x$,
- the properties expressing that $\leqslant$ is compatible with the operations $\cdot, \omega, \sharp$.

Free stabilisation algebra

Let $T(A)$ be the free term algebra of signature $\{1, \omega, \sharp, \cdot\}$ over A.

Given two elements s and t in $T(A)$, we write $s \equiv t$ iff the identity $s = t$ holds in all finite stabilisation monoids. Let $F(A)$ be the set of $\equiv$-classes.

Theorem

The algebra $F(A)$ is the free stabilisation algebra on the alphabet A.

Deciding the theory of stabilisation algebras

Conjecture

The equational theory of stabilisation algebras is the same as the one of finite stabilisation algebras.

If the conjecture is true, one can decide whether an identity is true in all finite stabilisation monoids.

A related result

An epigroup is a semigroup in which every element has a power that belongs to a subgroup. Then $x^{\omega-1}$ is well-defined in an epigroup.

Theorem (Zhil'tsov 2000, Mikhailova 2016)

The equational theory of epigroups is the same as the one of finite epigroups.

Complete set of axioms for epigroups

Theorem (Zhil'tsov 2000, Mikhailova 2016)

Complete set of axioms:

$$
\begin{aligned}
(xy)z &= x(yz) \\
(xy)^{\omega-1}x &= x(xy)^{\omega-1} \\
(x^{\omega-1})^2 x &= x^{\omega-1} \\
x^2 x^{\omega-1} &= (x^{\omega-1})^{\omega-1} \\
(x^{\omega-1}x)^{\omega-1} &= x^{\omega-1}x \\
(x^p)^{\omega-1} &= (x^{\omega-1})^p \quad \textit{for each prime } p
\end{aligned}
$$

Languages versus regular cost functions

We see a language L as a function

$$\chi_L : A^* \to \{0, \infty\}$$

We now view a regular cost function f as a function

$$f : F(A) \to \{\text{bounded, unbounded}\}$$

It suffices to define

$$f : T(A) \to \{\text{bounded, unbounded}\}$$

in such a way that if $s \equiv t$, then $f(s) = f(t)$.

Boundedness

Let $t \in T(A)$. For each $k, n > 0$, let

$$t_{k,n} = t[\omega \mapsto k!, \sharp \mapsto k!n]$$
$$t_k = \{t_{k,n} \mid n > 0\}$$

Thus each $t_{k,n}$ is a word, but t_k is a language.

For instance, if $t = \left((a^\omega)b\right)^\sharp$, then

$$t_{k,n} = \left((a^{k!})b\right)^{k!n}$$
$$t_k = \left\{ \left((a^{k!})b\right)^{k!n} \mid n > 0 \right\}$$

A boundedness result

Proposition (Kuperberg 2011, Colcombet 2013)

Let f be a regular cost function recognised by a triple (M, h, I). For each $t \in T(A)$ and each $k \geqslant |M|$, the set $f(t_k)$ is bounded iff $h([t]) \in I$.

Let $t \in T(A)$. We say that $f(t)$ is bounded if $f(t_k)$ is bounded for almost all k.

Boundedness is preserved by $\equiv$ and thus a regular cost function defines a map

$$f : F(A) \to \{\text{bounded, unbounded}\}$$

Varieties of cost functions

A variety of regular cost functions associates with each alphabet A a set $\mathcal{V}(A)$ of regular cost functions from $F(A)$ to $\{\text{bounded}, \text{unbounded}\}$ satisfying:

- For each alphabet A, $\mathcal{V}(A)$ is closed under $\min$, $\max$ and under quotients.
- $\mathcal{V}$ is closed under inverses of morphisms.

Problems. How to define quotients? How to define morphisms?

Quotients of cost functions

Let $T(A)$ be the term algebra of signature $\{\cdot, \omega, \sharp, 1\}$ over the alphabet A. A context is an element of $T(A \cup \{x\})$, where $x \notin A$.

Given a context C and $t \in T(A)$, $C[t]$ is the term obtained by replacing the occurrences of x by t in C.

Let $f : F(A) \to \{\text{bounded}, \text{unbounded}\}$ be a regular cost function and let C be a context on $T(A)$. The context regular cost function $C^{-1}f$ is defined by setting $(C^{-1}f)(t) = f(C(t))$ for all $t \in F(A)$.

Inverses of morphisms

Let $f : F(A) \to \{\text{bounded, unbounded}\}$ be a cost function and let $\varphi : F(B) \to F(A)$ be a morphism. Define $\varphi^{-1}(f) : F(B) \to \{\text{bounded, unbounded}\}$ by

$$(\varphi^{-1}(f))(t) = f(\varphi(t))$$

for all $t \in F(B)$.

The variety theorem

A variety of finite stabilisation monoids is a class of finite stabilisation monoids closed under taking submonoids, quotients and finite products.

Theorem (Daviaud, Kuperberg, Pin 2016)

Varieties of regular cost functions are in one-to-one correspondence with varieties of finite stabilisation monoids.

Equational characterization

Theorem (Daviaud, Kuperberg, Pin 2016)

Every variety of cost functions is characterized by a set of profinite equations.

Link with positive varieties of languages

Any positive variety of languages is in particular a variety of cost functions.

If a positive variety of languages is defined by a set of identities E, then it is a variety of cost functions, defined by the set of identities $E \cup \{x^\sharp = x^\omega\}$.

Conversely, if a variety of regular cost functions is defined by a set of identities E, then the variety of cost functions defined by $E \cup \{x^\sharp = x^\omega\}$ can be identified with a positive variety of languages.

Examples of varieties

The variety of aperiodic cost functions is defined by the identity $x^\omega = x^{\omega+1}$. It contains recognisable upsets that are not languages, like $u \mapsto |u|_a$.

It coincides with the variety of Cost-FO-definable cost functions and with the variety of Cost-LTL-definable cost functions.

Temporal cost functions

These functions allow one to count the number of occurrences of consecutive events [Colcombet, Kuperberg, Lombardy 2010].

Temporal cost functions over A form a lattice of regular cost functions, defined by the equations

$$(xy^\sharp z)^\sharp = (xy^\sharp z)^\omega$$

for all $x, z \in F(A)$ and all $y \in F(A) - \{1\}$.

Commutative varieties

Proposition (DKP 16)

The smallest variety of cost functions containing the cost functions $|u|_a$ for each letter a, is defined by the equations $x \leqslant 1$, $x^2 = x$, $xy = yx$ and $x^\sharp y^\sharp = (xy)^\sharp$.

The cost function $\mathrm{maxblock}_a$ satisfies $x^2 = x$ and $xy = yx$ but not $x^\sharp y^\sharp = (xy)^\sharp$.

Commutative varieties (2)

For each $k \geqslant 0$ and each letter a, let

$$L_{a,k} = \{u \mid |u|_a = k\}$$

Proposition (DKP 16)

The smallest variety of cost functions containing the cost functions $|u|_a$ and $\chi_{L_{a,k}}$ is defined by the equations $x^\omega = x^{\omega+1}$, $xy = yx$ and $x^\sharp y^\sharp = (xy)^\sharp$.

The end

Cost regular expressions: B- and S-expressions

B-expression:

$$E = \emptyset \mid a \mid E + E \mid EE \mid E^* \mid E^{\leqslant N},$$

The k-unfolding E_k of E is obtained be replacing $E^{\leqslant N}$ by $1 + E + E^2 + \cdots E^k$. For instance, if $k = 3$, $a^{\leqslant N}$ becomes $1 + a + aa + aaa$.

A B-expression E defines a cost function $[\![E]\!]_B$

$$[\![E]\!]_B(u) = \inf\{k \in \mathbb{N} \mid u \in L(E_k)\}$$

Examples of cost regular expressions

Examples of B-expressions:

$[\![b^*(ab^*)^{\leqslant N}]\!]_B = u \to |u|_a$

$[\![(a^*b)^*a^{\leqslant N}(ba^*)^*]\!]_B = \mathrm{minblock}_a$

Dual form: S-expression:

$$E = \emptyset \mid a \mid E + E \mid EE \mid E^* \mid E^{>N},$$

$[\![b^*(ab^*)^{\geqslant N}]\!]_S = u \to |u|_a$

$[\![(a^*b)^*a^{\geqslant N}(ba^*)^*]\!]_S = \mathrm{maxblock}_a$

Cost automata

Automaton + a finite set of counters, with initial value 0. Counters can be reset to 0 (r), incremented by 1 (i), or checked (c).

Given a run p, let $C(p)$ be the set of checked values during this run.

Semantics of B-automata:

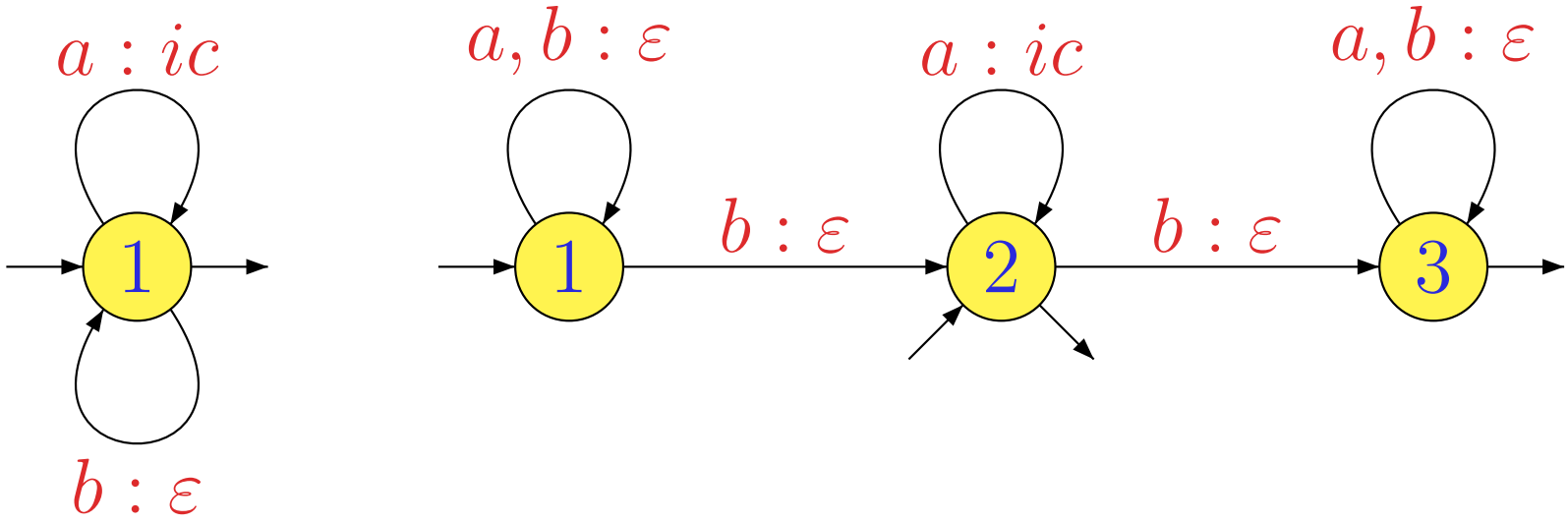
$$[\![\mathcal{A}]\!]_B(u) = \inf\{\sup C(p) \mid p \text{ is an accepting run over } u\}$$

Semantics of S-automata:

$$[\![\mathcal{A}]\!]_S(u) = \sup\{\inf C(p) \mid p \text{ is an accepting run over } u\}$$

Example of B-automaton

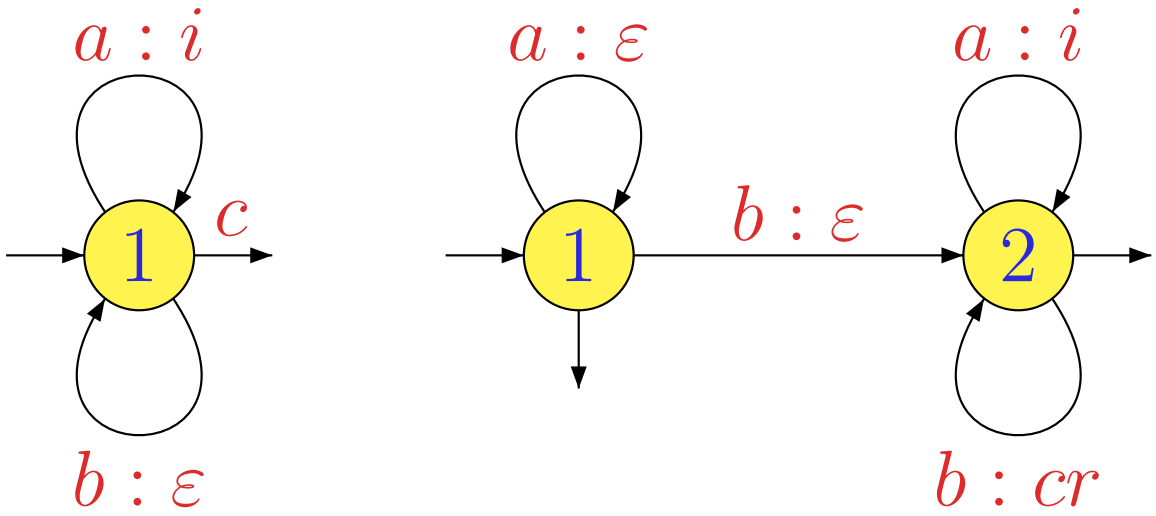
B-automata computing $|u|_a$ and $\mathrm{minblock}_a$.



$[\![\mathcal{A}]\!]_B(u) = \inf\{\sup C(p) \mid p \text{ is an accepting run over } u\}$

Examples of S-automata

S-automata computing $|u|_a$ and $\mathrm{minblock}_a$.



$[\![\mathcal{A}]\!]_S(u) = \sup\{\inf C(p) \mid p \text{ is an accepting run over } u\}$