The bridge between regular cost functions and omega-regular languages

Thomas Colcombet and Nathanaël Fijalkow

ICALP

July 15 2016

The bridge between regular cost functions and omega-regular languages

A quantitative extension of regular languages

Thomas Colcombet and Nathanaël Fijalkow

ICALP

July 15 2016

INSTITUT DE RECHERCHE EN INFORMATIQUE FONDAMENTALE

The bridge between regular cost functions and omega-regular languages

A quantitative extension of regular languages

Regular languages over words of length ω

Thomas Colcombet and Nathanaël Fijalkow

ICALP

July 15 2016

INSTITUT DE RECHERCHE EN INFORMATIQUE FONDAMENTALE

A way to transfer results and constructions

A quantitative extension of regular languages

Regular languages over words of length ω

The bridge between regular cost functions and omega-regular languages

Thomas Colcombet and Nathanaël Fijalkow

ICALP

July 15 2016

Regular cost functions

Regular cost functions

Regular languages:

toolbox for solving boolean problems over words and trees

Regular cost functions generalize it to:

toolbox for solving boundedness questions over words and trees

Regular cost functions

Regular languages:
toolbox for solving boolean problems over words and trees

Regular cost functions generalize it to:
toolbox for solving boundedness questions over words and trees

For instance, can you bound n such that:

$(L+\varepsilon)^n = L^* \quad ?$ (Finite power property **[Simon]**)

Regular cost functions

Regular languages:

toolbox for solving boolean problems over words and trees

Regular cost functions generalize it to:

toolbox for solving boundedness questions over words and trees

For instance, can you bound n such that:

$(L+\varepsilon)^n = L^*$? (Finite power property **[Simon]**)

Over all words u (infinite trees t), the fixpoint of ϕ(x,Z) is reached within at most n steps? (boundedness of fixpoint **[Blumensath&Otto&Weyer]**)

Regular cost functions

Regular languages:
  toolbox for solving boolean problems over words and trees
Regular cost functions generalize it to:
  toolbox for solving boundedness questions over words and trees

For instance, can you bound n such that:

$(L+\varepsilon)^n = L^* \quad ?$ (Finite power property **[Simon]**)

Over all words u (infinite trees t), the fixpoint of ϕ(x,Z) is reached within at most n steps? (boundedness of fixpoint **[Blumensath&Otto&Weyer]**)

For all words u (tree t), there exists a regular expression of star-height k of size n that accepts a subset of L that contains u (resp. t)?
(star-height problem **[Hashiguchi, Kirsten, C&Löding]**)

Regular cost functions

Regular languages:
toolbox for solving boolean problems over words and trees
Regular cost functions generalize it to:
toolbox for solving boundedness questions over words and trees

For instance, can you bound n such that:

$(L+\varepsilon)^n = L^* \quad ?$ (Finite power property **[Simon]**)

Over all words u (infinite trees t), the fixpoint of ϕ(x,Z) is reached within at most n steps? (boundedness of fixpoint **[Blumensath&Otto&Weyer]**)

For all words u (tree t), there exists a regular expression of star-height k of size n that accepts a subset of L that contains u (resp. t)? (star-height problem **[Hashiguchi, Kirsten, C&Löding]**)

For all infinite trees t, there exists a parity automaton of index [i,j], and size n that accepts a subset of L that contains t? (Mostowski **[CL,CKLvB]**)

Regular cost functions

Regular languages:
toolbox for solving boolean problems over words and trees

Regular cost functions generalize it to:
toolbox for solving boundedness questions over words and trees

For instance, can you bound n such that:

$(L+\varepsilon)^n = L^*$? (Finite power property **[Simon]**)

Over all words u (infinite trees t), the fixpoint of $\phi(x,Z)$ is reached within at most n steps? (boundedness of fixpoint **[Blumensath&Otto&Weyer]**)

For all words u (tree t), there exists a regular expression of star-height k of size n that accepts a subset of L that contains u (resp. t)? (star-height problem **[Hashiguchi, Kirsten, C&Löding]**)

For all infinite trees t, there exists a parity automaton of index [i,j], and size n that accepts a subset of L that contains t? (Mostowski **[CL,CKLvB]**)

The Church synthesis problem can be solved up to an error of n bits? (**[Rabinovitch&Velner]**)

Regular cost functions

Regular languages:
toolbox for solving boolean problems over words and trees
Regular cost functions generalize it to:
toolbox for solving boundedness questions over words and trees

For instance, can you bound n such that:

$(L+\varepsilon)^n = L^*$? (Finite power property **[Simon]**)

Over all words u (infinite trees t), the fixpoint of ϕ(x,Z) is reached within at most n steps? (boundedness of fixpoint **[Blumensath&Otto&Weyer]**)

For all words u (tree t), there exists a regular expression of star-height k of size n that accepts a subset of L that contains u (resp. t)? (star-height problem **[Hashiguchi, Kirsten, C&Löding]**)

For all infinite trees t, there exists a parity automaton of index [i,j], and size n that accepts a subset of L that contains t? (Mostowski **[CL,CKLvB]**)

The Church synthesis problem can be solved up to an error of n bits? (**[Rabinovitch&Velner]**)

For all these questions, we do not care about precise values.

Regular cost functions ideas

Regular cost functions ideas

A regular cost function $f : A^* \to \mathbb{N} \cup \{\infty\}$
can be seen as two (unusual)
languages:

Regular cost functions ideas

A regular cost function $f : A^* \to \mathbb{N} \cup \{\infty\}$
can be seen as two (unusual)
languages:

{ u : f(u) is small }

Regular cost functions ideas

A regular cost function $f : A^* \to \mathbb{N} \cup \{\infty\}$
can be seen as two (unusual)
languages:

{ u : f(u) is small } { u : f(u) is large }

Regular cost functions ideas

A regular cost function $f : A^* \to \mathbb{N} \cup \{\infty\}$
can be seen as two (unusual)
languages:

$$\{\, \mathbf{u} : \mathbf{f(u)} \textbf{ is small} \,\} \xrightleftharpoons{\text{complement}} \{\, \mathbf{u} : \mathbf{f(u)} \textbf{ is large} \,\}$$

Regular cost functions ideas

A regular cost function $f : A^* \to \mathbb{N} \cup \{\infty\}$ can be seen as two (unusual) languages:

it has a formal meaning in non-standard analysis

$\{\, \mathbf{u : f(u) \text{ is small}} \,\}^{\text{ext}}$ $\xrightleftharpoons{\text{complement}}$ $\{\, \mathbf{u : f(u) \text{ is large}} \,\}^{\text{ext}}$

Regular cost functions ideas

A regular cost function $f : A^* \to \mathbb{N} \cup \{\infty\}$ can be seen as two (unusual) languages:

it has a formal meaning in non-standard analysis

$$\{\, \mathbf{u} : \mathbf{f(u)} \text{ is small} \,\}^{\text{ext}} \quad \underset{\text{complement}}{\rightleftarrows} \quad \{\, \mathbf{u} : \mathbf{f(u)} \text{ is large} \,\}^{\text{ext}}$$

Functions that are large on the same inputs are **≈-equivalent**.

Regular cost functions ideas

A regular cost function $f : A^* \to \mathbb{N} \cup \{\infty\}$ can be seen as two (unusual) languages:

it has a formal meaning in non-standard analysis

$$\{\,\mathbf{u : f(u)\ is\ small}\,\}^{\text{ext}} \xrightleftharpoons[\text{complement}]{\text{complement}} \{\,\mathbf{u : f(u)\ is\ large}\,\}^{\text{ext}}$$

Functions that are large on the same inputs are **≈-equivalent**.

Example: f(u) = the length of longest block of consecutive a's

$$(a^{\texttt{small}}b)^*a^{\texttt{small}} \xrightleftharpoons[\text{complement}]{\text{complement}} (a^*b)^*a^{\texttt{large}}(ba^*)^*$$

Regular cost functions ideas

A regular cost function $f : A^* \to \mathbb{N} \cup \{\infty\}$ can be seen as two (unusual) languages:

it has a formal meaning in non-standard analysis

$$\{\, \mathbf{u} : \mathbf{f(u)} \text{ is small} \,\}^{\text{ext}} \quad \xrightleftharpoons{\text{complement}} \quad \{\, \mathbf{u} : \mathbf{f(u)} \text{ is large} \,\}^{\text{ext}}$$

Functions that are large on the same inputs are **≈-equivalent**.

Example: f(u) = the length of longest block of consecutive a's

$$(a^{\texttt{small}}b)^*a^{\texttt{small}} \quad \xrightleftharpoons{\text{complement}} \quad (a^*b)^*a^{\texttt{large}}(ba^*)^*$$

Main theorem of regular cost functions:

B-side		S-side
B-rational expressions		S-rational expressions
=		=
B-automata	complement	S-automata
=		=
cost MSO		cost MSO*
=		=
(stabilisation monoids,up-sets)		(stabilisation monoids,down-sets)

Difficulties of REG(cost functions)

Difficulties of REG(cost functions)

Manipulating bounds everywhere in the proofs is costly. ns-analysis

Difficulties of REG(cost functions)

Manipulating bounds everywhere in the proofs is costly. ns-analysis

Constructions are **complicated**, and in particular more complicated than for infinite words.

Difficulties of REG(cost functions)

Manipulating bounds everywhere in the proofs is costly. ns-analysis

Constructions are **complicated**, and in particular more complicated than for infinite words.

Automata **cannot be determinized**, while it is required for treating the case of trees.
One has to resort in history-deterministic automata which are more involved to handle.

Difficulties of REG(cost functions)

Manipulating bounds everywhere in the proofs is costly. ns-analysis

Constructions are **complicated**, and in particular more complicated than for infinite words.

Automata **cannot be determinized**, while it is required for treating the case of trees.
One has to resort in history-deterministic automata which are more involved to handle.

Similarities with REG(infinite objects)

Difficulties of REG(cost functions)

Manipulating bounds everywhere in the proofs is costly. ns-analysis

Constructions are **complicated**, and in particular more complicated than for infinite words.

Automata **cannot be determinized**, while it is required for treating the case of trees.
One has to resort in history-deterministic automata which are more involved to handle.

Similarities with REG(infinite objects)

$(ab)^n a = a(ba)^n \quad (n \text{ large})$ $\qquad (ab)^\omega = a(ba)^\omega$

Difficulties of REG(cost functions)

Manipulating bounds everywhere in the proofs is costly. ns-analysis

Constructions are **complicated**, and in particular more complicated than for infinite words.

Automata **cannot be determinized**, while it is required for treating the case of trees.
One has to resort in history-deterministic automata which are more involved to handle.

Similarities with REG(infinite objects)

$(ab)^n a = a(ba)^n \quad (n \text{ large})$ $\qquad (ab)^\omega = a(ba)^\omega$

High levels ideas are similar:

- use of games for dealing with trees
- use of the ideal decomposition of monoids

Difficulties of REG(cost functions)

Manipulating bounds everywhere in the proofs is costly. ns-analysis

Constructions are **complicated**, and in particular more complicated than for infinite words.

Automata **cannot be determinized**, while it is required for treating the case of trees.
One has to resort in history-deterministic automata which are more involved to handle.

Similarities with REG(infinite objects)

$(ab)^n a = a(ba)^n \quad (n \text{ large})$ $\qquad\qquad (ab)^\omega = a(ba)^\omega$

High levels ideas are similar:

- use of games for dealing with trees
- use of the ideal decomposition of monoids

The most efficient translations from B/S-automata to history-deterministic B/S-automata mimic the ideas of Safra's construction.

Difficulties of REG(cost functions)

Manipulating bounds everywhere in the proofs is costly. ns-analysis

Constructions are **complicated**, and in particular more complicated than for infinite words.

Automata **cannot be determinized**, while it is required for treating the case of trees.
One has to resort in history-deterministic automata which are more involved to handle.

Similarities with REG(infinite objects)

$$(ab)^n a = a(ba)^n \quad (n \text{ large}) \qquad\qquad (ab)^\omega = a(ba)^\omega$$

High levels ideas are similar:

- use of games for dealing with trees
- use of the ideal decomposition of monoids

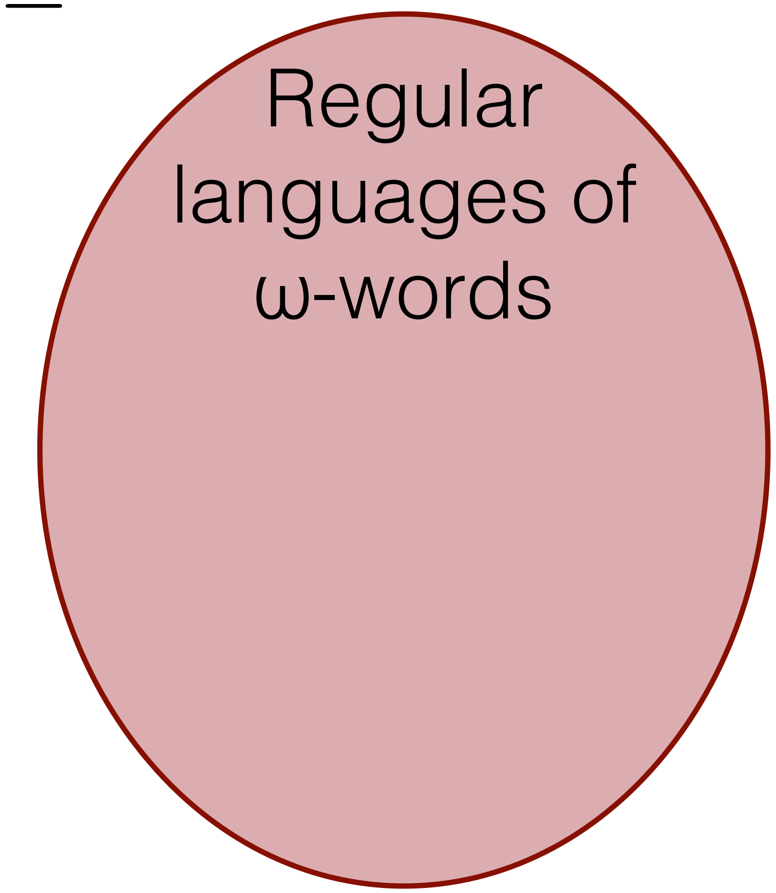
The most efficient translations from B/S-automata to history-deterministic B/S-automata mimic the ideas of Safra's construction.

This work makes formal some similarities, and use it to factorizing proofs.

The bridge

The bridge

$L \subseteq A^\omega$

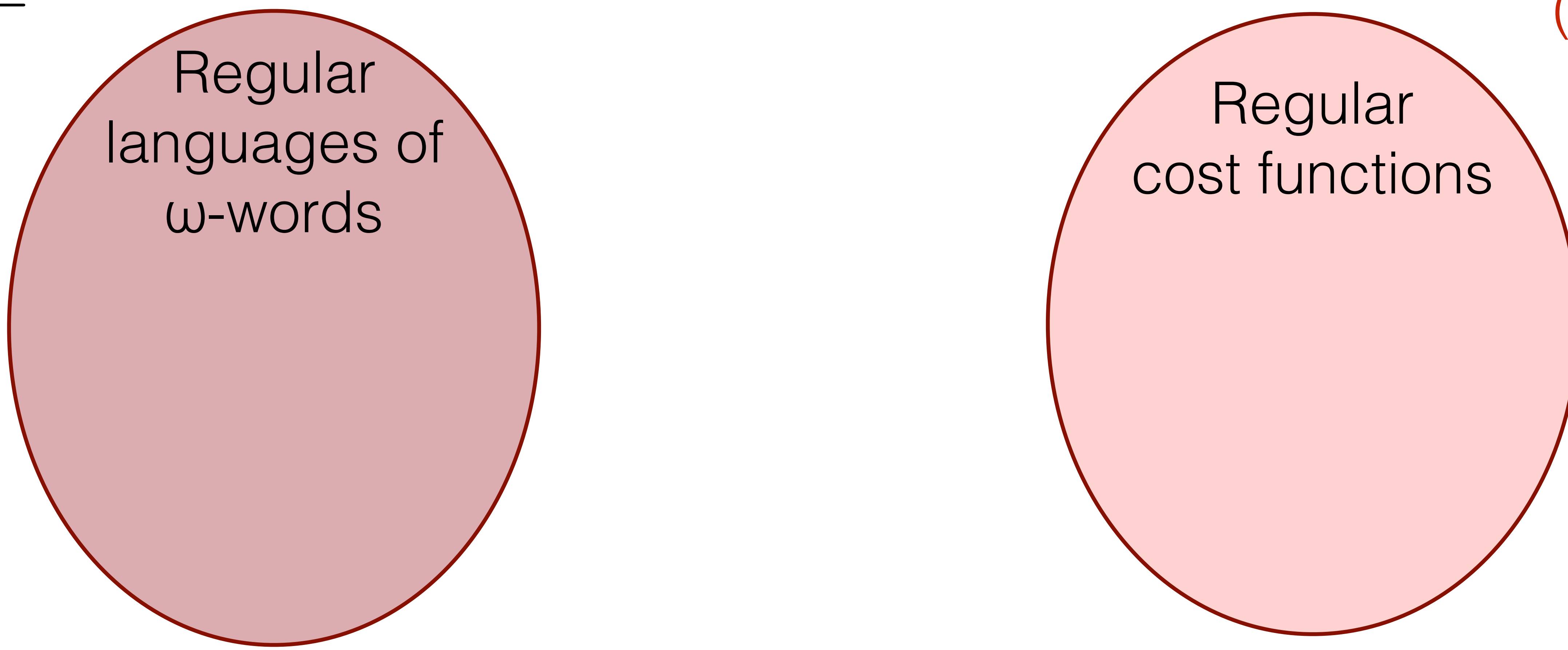


The bridge

$L \subseteq A^\omega$

$f : A^* \to \mathbb{N} \cup \{\infty\}$

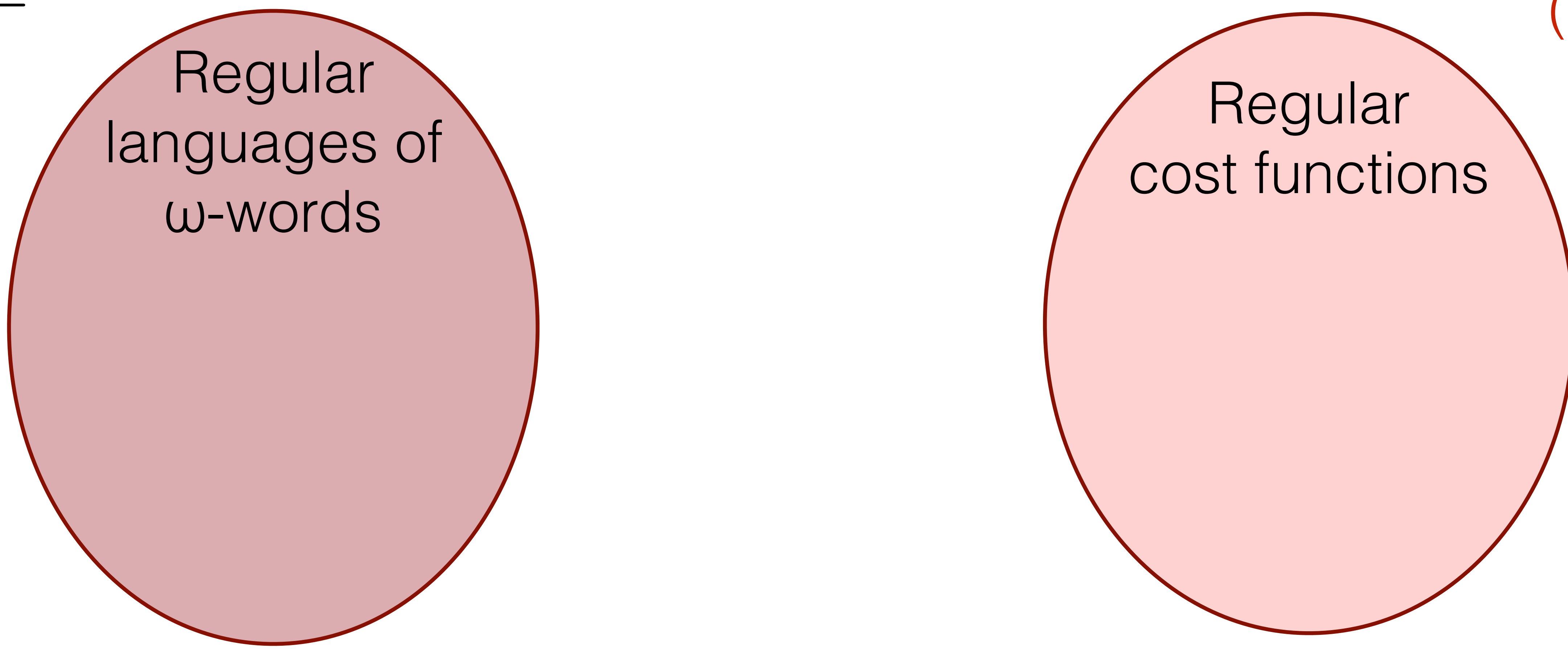
(up to $\approx$)



The bridge

$L \subseteq A^\omega$

$f : A^* \to \mathbb{N} \cup \{\infty\}$
(up to $\approx$)



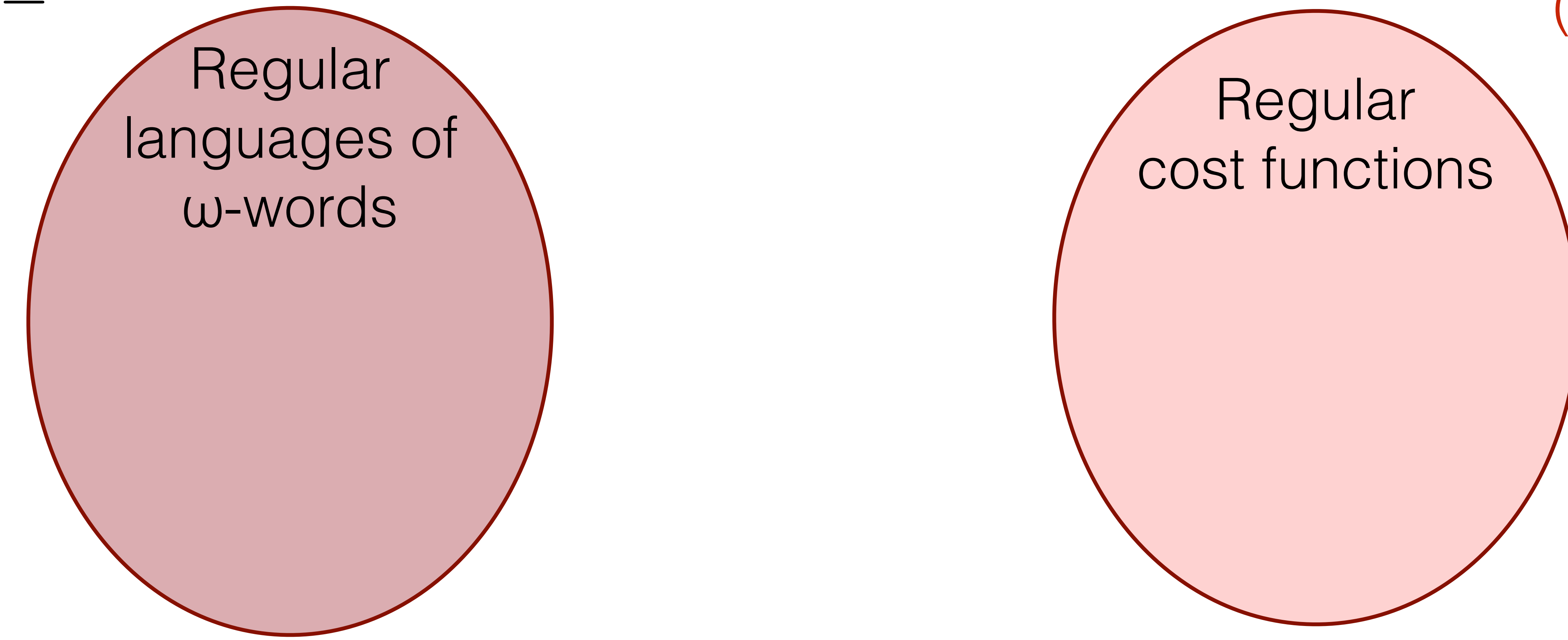
Definition: For $L \subseteq A^\omega$,

$$\begin{aligned} L^{\mathtt{ol}} : A^* &\to \mathbb{N} \cup \{\infty\} \\ u &\mapsto \sup\{n \ : \ u = v w_1 \dots w_n v' \\ &\qquad\qquad w_1, \dots, w_n \neq \varepsilon \\ &\qquad\qquad v\{w_1, \dots, w_n\}^\omega \subseteq L\} \end{aligned}$$

The bridge

$L \subseteq A^\omega$

$f : A^* \to \mathbb{N} \cup \{\infty\}$
(up to $\approx$)



Definition: For $L \subseteq A^\omega$,

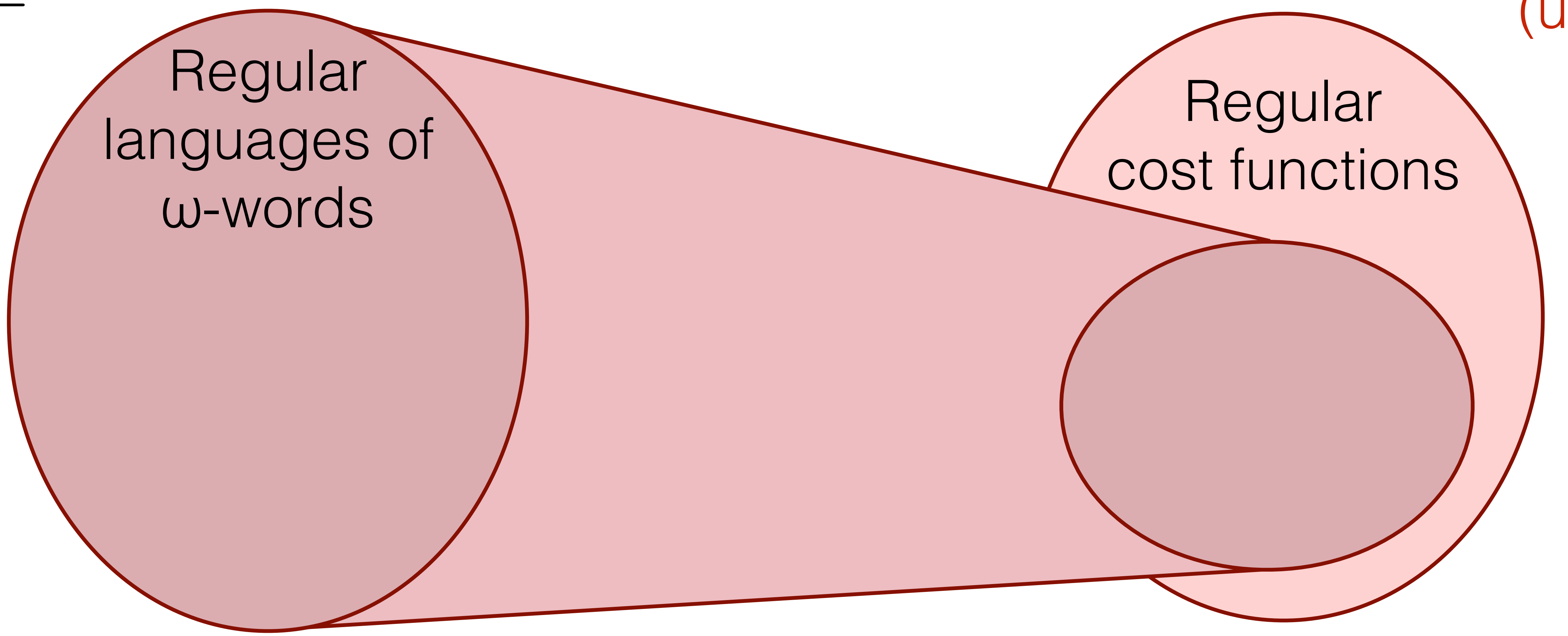
$$L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$
$$u \mapsto \sup\{n \ : \ u = v w_1 \dots w_n v' \quad w_1, \dots, w_n \neq \varepsilon \quad v\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

Lemma: For $L \subseteq A^\omega$ regular, $L^{\mathtt{ol}}$ is a regular cost function.

The bridge

$L \subseteq A^\omega$

$f : A^* \to \mathbb{N} \cup \{\infty\}$
(up to ≈)



Definition: For $L \subseteq A^\omega$,

$$
\begin{aligned}
L^{\mathtt{ol}} : A^* &\to \mathbb{N} \cup \{\infty\} \\
u &\mapsto \sup\{n \ : \ u = vw_1 \dots w_n v' \\
&\qquad\qquad w_1, \dots, w_n \neq \varepsilon \\
&\qquad\qquad v\{w_1, \dots, w_n\}^\omega \subseteq L\}
\end{aligned}
$$

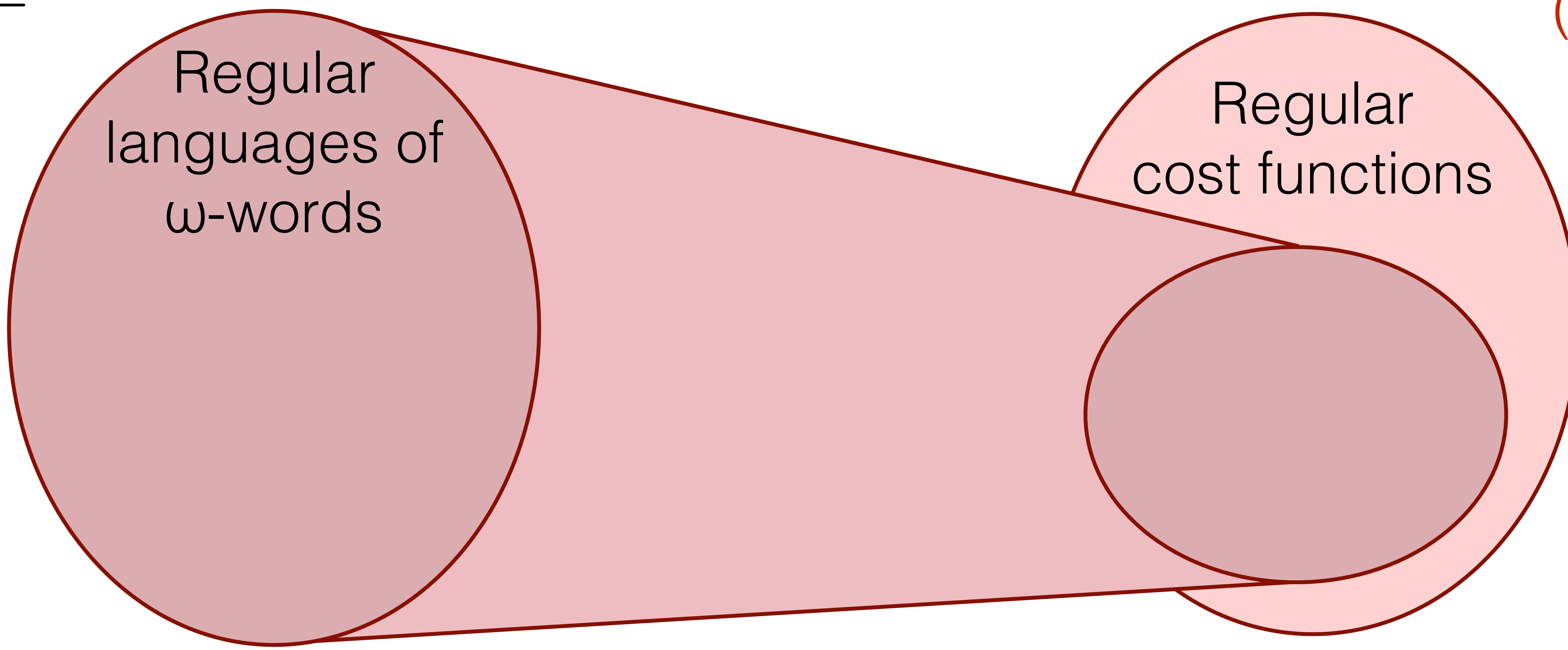
Lemma: For $L \subseteq A^\omega$ regular, $L^{\mathtt{ol}}$ is a regular cost function.

The bridge

$L \subseteq A^\omega$

$f : A^* \to \mathbb{N} \cup \{\infty\}$

(up to ≈)



Definition: For $L \subseteq A^\omega$,

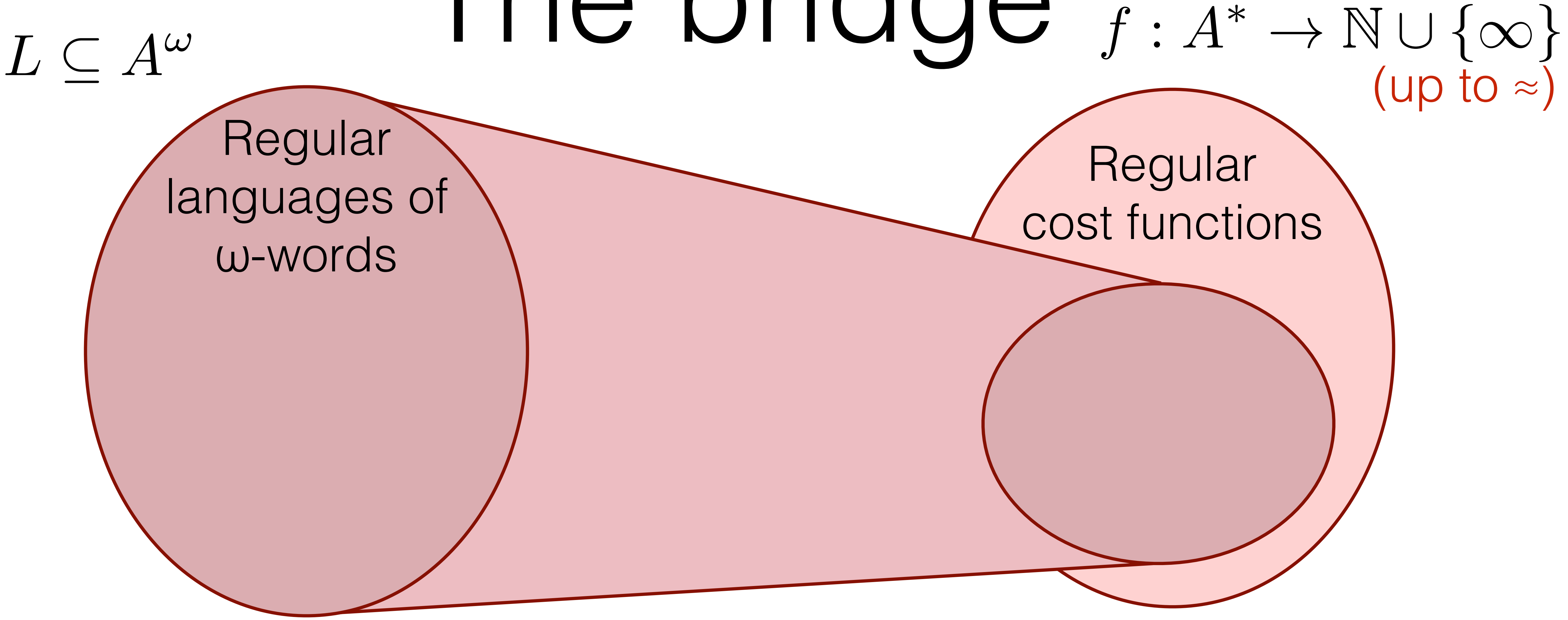
$$L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$
$$u \mapsto \sup\{n \ : \ u = vw_1 \dots w_n v' \quad w_1, \dots, w_n \neq \varepsilon \quad v\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

Lemma: For $L \subseteq A^\omega$ regular, $L^{\mathtt{ol}}$ is a regular cost function.

Lemma: $uv^\omega \in L$ iff

$$\lim_n L^{\mathtt{ol}}(uv^n) = \infty$$

The bridge



Definition: For $L \subseteq A^\omega$,

$$L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$

$$u \mapsto \sup\{n \ : \ u = vw_1 \dots w_n v' \quad w_1, \dots, w_n \neq \varepsilon \quad v\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

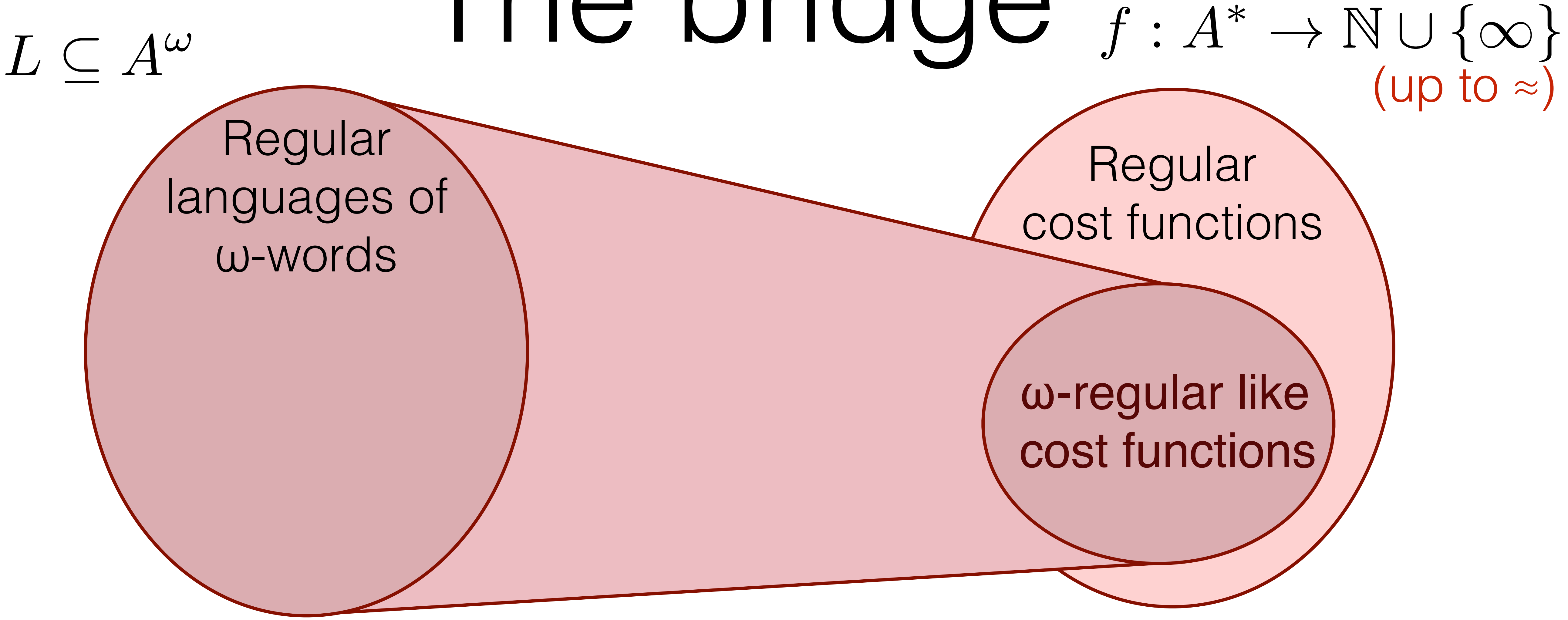
Lemma: For $L \subseteq A^\omega$ regular, $L^{\mathtt{ol}}$ is a regular cost function.

Lemma: $uv^\omega \in L$ iff

$$\lim_n L^{\mathtt{ol}}(uv^n) = \infty$$

Lemma: The $^{\mathtt{ol}}$ map is injective.

The bridge



Definition: For $L \subseteq A^\omega$,

$$L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$

$$u \mapsto \sup\{n \ : \ u = vw_1 \dots w_n v' \quad w_1, \dots, w_n \neq \varepsilon \quad v\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

Lemma: For $L \subseteq A^\omega$ regular, $L^{\mathtt{ol}}$ is a regular cost function.

Lemma: $uv^\omega \in L$ iff

$$\lim_n L^{\mathtt{ol}}(uv^n) = \infty$$

Lemma: The $^{\mathtt{ol}}$ map is injective.

First examples of the bridge

$L \subseteq A^\omega$

$$\begin{aligned} L^{\texttt{ol}} : A^* &\to \mathbb{N} \cup \{\infty\} \\ u &\mapsto \sup\{n \ : \ u = v w_1 \dots w_n v' \\ &\qquad\qquad w_1, \dots, w_n \neq \varepsilon \\ &\qquad\qquad u\{w_1, \dots, w_n\}^\omega \subseteq L\} \end{aligned}$$

First examples of the bridge

$L \subseteq A^\omega$ $\Rightarrow$

$$L^{\texttt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$
$$u \mapsto \sup\{n \ : \ u = vw_1 \dots w_n v'$$
$$w_1, \dots, w_n \neq \varepsilon$$
$$u\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

$$Buchi = (1^*2)^\omega$$

First examples of the bridge

$$L \subseteq A^\omega \quad \Rightarrow \quad \begin{aligned} L^{\mathtt{ol}} : A^* &\to \mathbb{N} \cup \{\infty\} \\ u &\mapsto \sup\{n \ : \ u = vw_1 \dots w_n v' \\ &\qquad w_1, \dots, w_n \neq \varepsilon \\ &\qquad u\{w_1, \dots, w_n\}^\omega \subseteq L\} \end{aligned}$$

$$Buchi = (1^*2)^\omega \qquad\qquad |\cdot|_2$$

First examples of the bridge

$L \subseteq A^\omega$ $\Rightarrow$

$$L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$
$$u \mapsto \sup\{n \ : \ u = vw_1 \dots w_n v'$$
$$w_1, \dots, w_n \neq \varepsilon$$
$$u\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

$Buchi = (1^*2)^\omega$ $\quad\quad$ $|\cdot|_2$

$coBuchi = (0+1)^*0^\omega$

First examples of the bridge

$$L \subseteq A^\omega$$

$$L^{\texttt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$

$$u \mapsto \sup\{n \;:\; u = vw_1 \dots w_n v' \quad w_1, \dots, w_n \neq \varepsilon \quad u\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

$$Buchi = (1^*2)^\omega \qquad |\cdot|_2$$

$$coBuchi = (0+1)^*0^\omega \qquad maxblock_0$$

First examples of the bridge

$$L \subseteq A^\omega$$

$$L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$

$$u \mapsto \sup\{n \;:\; u = vw_1 \dots w_n v' \quad w_1, \dots, w_n \neq \varepsilon \quad u\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

$$Buchi = (1^*2)^\omega \qquad |\cdot|_2$$

$$coBuchi = (0+1)^*0^\omega \qquad maxblock_0$$

$$parity_{i,j} = \{u \in [i,j]^\omega \mid \limsup_n u_n \text{ even}\}$$

First examples of the bridge

$L \subseteq A^\omega$ $\Rightarrow$ $$L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$ $$u \mapsto \sup\{n \;:\; u = vw_1 \ldots w_n v' \quad w_1, \ldots, w_n \neq \varepsilon \quad u\{w_1, \ldots, w_n\}^\omega \subseteq L\}$$

$Buchi = (1^*2)^\omega$	$\lvert \cdot \rvert_2$
$coBuchi = (0+1)^* 0^\omega$	$maxblock_0$
$parity_{i,j} = \{u \in [i,j]^\omega \mid \limsup_n u_n \text{ even}\}$	hierarchical B-condition

First examples of the bridge

$$L \subseteq A^\omega$$

$$L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$
$$u \mapsto \sup\{n \;:\; u = vw_1 \dots w_n v' \quad w_1, \dots, w_n \neq \varepsilon \quad u\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

$$Buchi = (1^*2)^\omega \qquad |\cdot|_2$$

$$coBuchi = (0+1)^*0^\omega \qquad maxblock_0$$

$$parity_{i,j} = \{u \in [i,j]^\omega \mid \limsup_n u_n \text{ even}\}$$

hierarchical B-condition
(up to ≈)

First examples of the bridge

$$L \subseteq A^\omega \quad \Longrightarrow \quad \begin{aligned} L^{\mathtt{ol}} : A^* &\to \mathbb{N} \cup \{\infty\} \\ u &\mapsto \sup\{n \;:\; u = vw_1 \ldots w_n v' \\ &\qquad w_1, \ldots, w_n \neq \varepsilon \\ &\qquad u\{w_1, \ldots, w_n\}^\omega \subseteq L\} \end{aligned}$$

$Buchi = (1^*2)^\omega$	$\lvert \cdot \rvert_2$
$coBuchi = (0+1)^*0^\omega$	$maxblock_0$
$parity_{i,j} = \{u \in [i,j]^\omega \mid \limsup_n u_n \text{ even}\}$	hierarchical B-condition (up to ≈)

Büchi automaton

First examples of the bridge

$$L \subseteq A^\omega \quad \Rightarrow \quad \begin{aligned} L^{\mathtt{ol}} : A^* &\to \mathbb{N} \cup \{\infty\} \\ u &\mapsto \sup\{n \ : \ u = vw_1 \dots w_n v' \\ &\qquad w_1, \dots, w_n \neq \varepsilon \\ &\qquad u\{w_1, \dots, w_n\}^\omega \subseteq L\} \end{aligned}$$

$Buchi = (1^*2)^\omega$	$\lvert \cdot \rvert_2$
$coBuchi = (0+1)^*0^\omega$	$maxblock_0$
$parity_{i,j} = \{u \in [i,j]^\omega \mid \limsup_n u_n \text{ even}\}$	hierarchical B-condition (up to ≈)
Büchi automaton	Same automaton seen as max-prefix-distance (up to ≈)

Büchi automaton case

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many *Büchi transitions*.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes for a finite u:

f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes for a finite u:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

A sequence of transitions starting in an initial state, forming a path reading the prefix of the input.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Goal: $L^{\mathtt{ol}} \approx f$

Seen as a **max-prefix-distance**, it computes for a finite u:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

A sequence of transitions starting in an initial state, forming a path reading the prefix of the input.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Goal: $L^{\mathtt{ol}} \approx f$

I.e. $L^{\mathtt{ol}}(u)$ large iff $f(u)$ large.

Seen as a **max-prefix-distance**, it computes for a finite u:

f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

A sequence of transitions starting in an initial state, forming a path reading the prefix of the input.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes for a finite u:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

A sequence of transitions starting in an initial state, forming a path reading the prefix of the input.

Goal: $L^{\mathtt{ol}} \approx f$

I.e. $L^{\mathtt{ol}}(u)$ large iff $f(u)$ large.

Assume $f(u)$ large.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes for a finite u:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

A sequence of transitions starting in an initial state, forming a path reading the prefix of the input.

Goal: $L^{\texttt{ol}} \approx f$

I.e. $L^{\texttt{ol}}(u)$ large iff $f(u)$ large.

Assume $f(u)$ large.

There is a prefix-run over u with a large number of Büchi transitions.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes for a finite u: f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

A sequence of transitions starting in an initial state, forming a path reading the prefix of the input.

Goal: $L^{\texttt{ol}} \approx f$

I.e. $L^{\texttt{ol}}(u)$ large iff $f(u)$ large.

Assume $f(u)$ large.

There is a prefix-run over u with a large number of Büchi transitions.

It can be decomposed into $\alpha\beta_1 \ldots \beta_n\gamma$ with n large and each of the β's start and end in the state state, and contain at least one Büchi transition.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes for a finite u:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

A sequence of transitions starting in an initial state, forming a path reading the prefix of the input.

Goal: $L^{\mathtt{ol}} \approx f$

I.e. $L^{\mathtt{ol}}(u)$ large iff $f(u)$ large.

Assume $f(u)$ large.

There is a prefix-run over u with a large number of Büchi transitions.

It can be decomposed into $\alpha\beta_1 \ldots \beta_n\gamma$ with n large and each of the β's start and end in the state state, and contain at least one Büchi transition.

Hence $\alpha\{\beta_1, \ldots, \beta_n\}^\omega$ is a set of accepting runs.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes for a finite u:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

A sequence of transitions starting in an initial state, forming a path reading the prefix of the input.

Goal: $L^{\mathtt{ol}} \approx f$

I.e. $L^{\mathtt{ol}}(u)$ large iff $f(u)$ large.

Assume $f(u)$ large.

There is a prefix-run over u with a large number of Büchi transitions.

It can be decomposed into $\alpha\beta_1 \dots \beta_n\gamma$ with n large and each of the β's start and end in the state state, and contain at least one Büchi transition.

Hence $\alpha\{\beta_1, \dots, \beta_n\}^\omega$ is a set of accepting runs.

Hence $L^{\mathtt{ol}}(u) \geq n$ is large.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

Goal: $L^{\mathtt{ol}} \approx f$

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a **max-prefix-distance**, it computes:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

Goal: $L^{\mathtt{ol}} \approx f$

Assume $L^{\mathtt{ol}}(u)$ is large.

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a max-prefix-distance, it computes:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

Goal: $L^{\mathtt{ol}} \approx f$

Assume $L^{\mathtt{ol}}(u)$ is large.

$$u \;=\; v\; w_1\; w_2\; \ldots\; w_n\; t$$

with n large,
and $v\{w_1,\ldots,w_n\}^\omega \subseteq L$

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a max-prefix-distance, it computes:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

Goal: $L^{\texttt{ol}} \approx f$

Assume $L^{\texttt{ol}}(u)$ is large.

$$u \;=\; v\; w_1\; w_2\; \ldots\; w_n\; t$$

with n large,
and $v\{w_1,\ldots,w_n\}^\omega \subseteq L$

By Ramsey these can be regrouped into:

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a max-prefix-distance, it computes:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

Goal: $L^{\mathtt{ol}} \approx f$

Assume $L^{\mathtt{ol}}(u)$ is large.

$$u \;=\; v\; w_1\; w_2\; \ldots\; w_n\; t$$

with n large,
and $v\{w_1, \ldots, w_n\}^\omega \subseteq L$

By Ramsey these can be regrouped into:

$$u \;=\; v'\; w'_1\; w'_2\; \ldots\; w'_{n'}\; t'$$

with n' large, all w''s corresponding to the same idempotent e in the transition semigroup of the automaton:

$$\mathcal{P}(Q \times \{1,2\} \times Q)$$

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a max-prefix-distance, it computes:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u.*

Goal: $L^{\mathtt{ol}} \approx f$

Assume $L^{\mathtt{ol}}(u)$ is large.

$$u \;=\; v\; w_1\; w_2\; \ldots\; w_n\; t$$

with n large,
and $v\{w_1,\ldots,w_n\}^\omega \subseteq L$

By Ramsey these can be regrouped into:

$$u \;=\; v'\; w'_1\; w'_2\; \ldots\; w'_{n'}\; t'$$

with n' large, all w''s corresponding to the same idempotent e in the transition semigroup of the automaton:

$$\mathcal{P}(Q\times\{1,2\}\times Q)$$

Since $v'\{w'_1,\ldots,w'_n\}^\omega \subseteq L$
this idempotent contains some \`reachable' (p,2,p).

Büchi automaton case

A **Büchi automaton** accepts the language L of ω-words such that there is a (infinite) run with infinitely many ***Büchi transitions***.

Seen as a max-prefix-distance, it computes:
f(u) = maximum number of Büchi transitions seen on a *prefix-run over u*.

Goal: $L^{\mathtt{ol}} \approx f$

Assume $L^{\mathtt{ol}}(u)$ is large.

$$u \;=\; v\; w_1\; w_2\; \ldots\; w_n\; t$$

with n large,
and $v\{w_1, \ldots, w_n\}^\omega \subseteq L$

By Ramsey these can be regrouped into:

$$u \;=\; v'\; w'_1\; w'_2\; \ldots\; w'_{n'}\; t'$$

with n' large, all w''s corresponding to the same idempotent e in the transition semigroup of the automaton:

$$\mathcal{P}(Q \times \{1,2\} \times Q)$$

Since $v'\{w'_1, \ldots, w'_n\}^\omega \subseteq L$ this idempotent contains some `reachable' (p,2,p).

This witnesses that $f(u)$ is large.

First examples of the bridge

$$L \subseteq A^\omega \quad \Rightarrow \quad \begin{aligned} L^{\mathtt{ol}} : A^* &\to \mathbb{N} \cup \{\infty\} \\ u &\mapsto \sup\{n \ : \ u = v w_1 \ldots w_n v' \\ & \qquad w_1, \ldots, w_n \neq \varepsilon \\ & \qquad u\{w_1, \ldots, w_n\}^\omega \subseteq L\} \end{aligned}$$

$Buchi = (1^* 2)^\omega$	$\lvert \cdot \rvert_2$
$coBuchi = (0+1)^* 0^\omega$	$maxblock_0$
$parity_{i,j} = \{u \in [i,j]^\omega \mid \limsup_n u_n \text{ even}\}$	hierarchical B-condition (up to ≈)
Büchi automaton	Same automaton seen as max-prefix-distance (up to ≈)

First examples of the bridge

$$L \subseteq A^\omega \quad \Rightarrow \quad L^{\mathtt{ol}} : A^* \to \mathbb{N} \cup \{\infty\}$$

$$u \mapsto \sup\{n \ : \ u = vw_1 \dots w_n v' \quad w_1, \dots, w_n \neq \varepsilon \quad u\{w_1, \dots, w_n\}^\omega \subseteq L\}$$

$Buchi = (1^*2)^\omega$ — $|\cdot|_2$

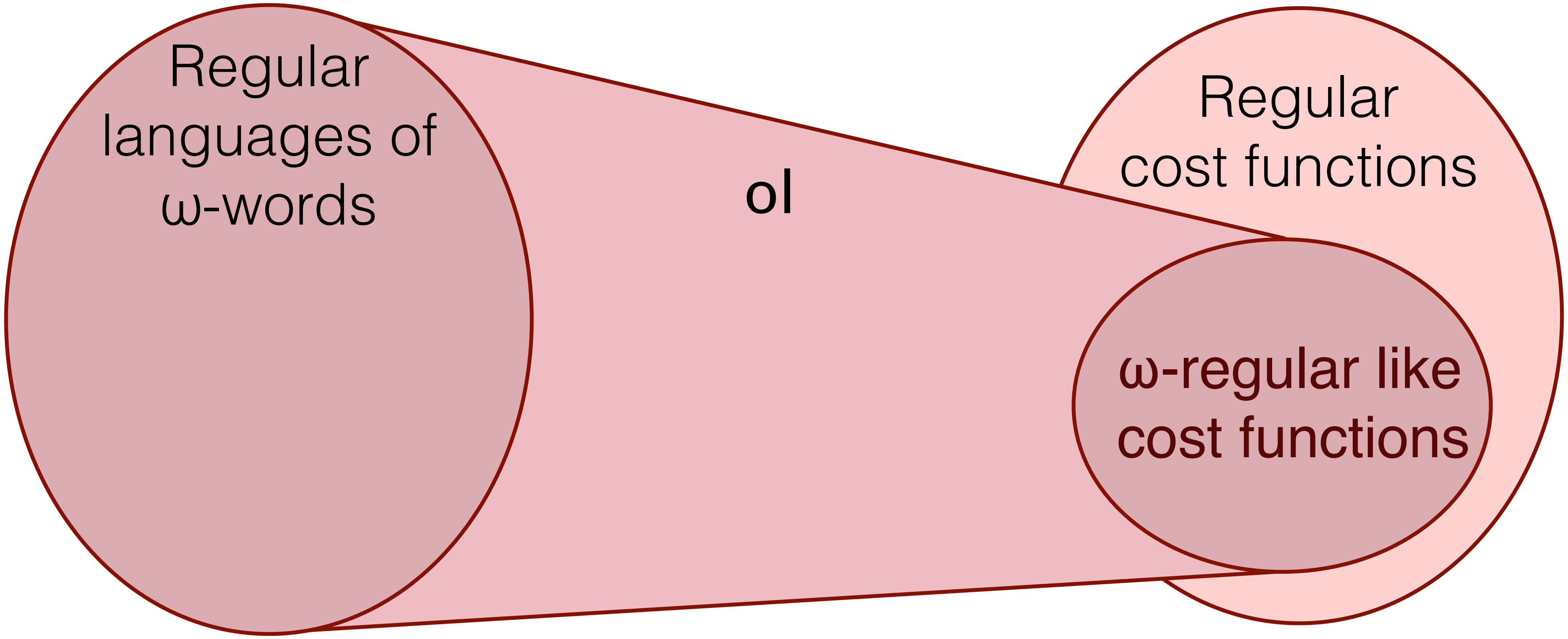
$coBuchi = (0+1)^*0^\omega$ — $maxblock_0$

$parity_{i,j} = \{u \in [i,j]^\omega \mid \limsup_n u_n \text{ even}\}$ — hierarchical B-condition (up to ≈)

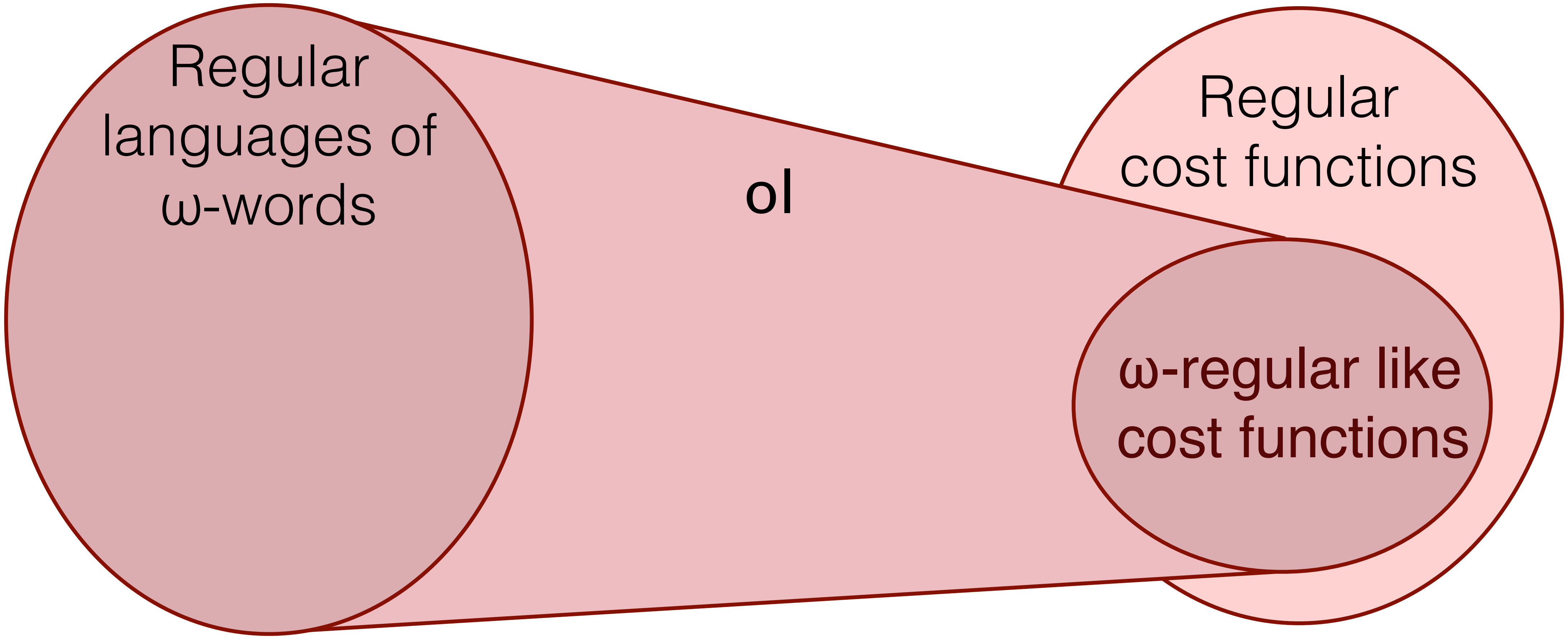
Büchi automaton — Same automaton seen as max-prefix-distance (up to ≈)

Rabin automaton — Same automaton seen as max-prefix-B automaton (up to ≈)

The bridge

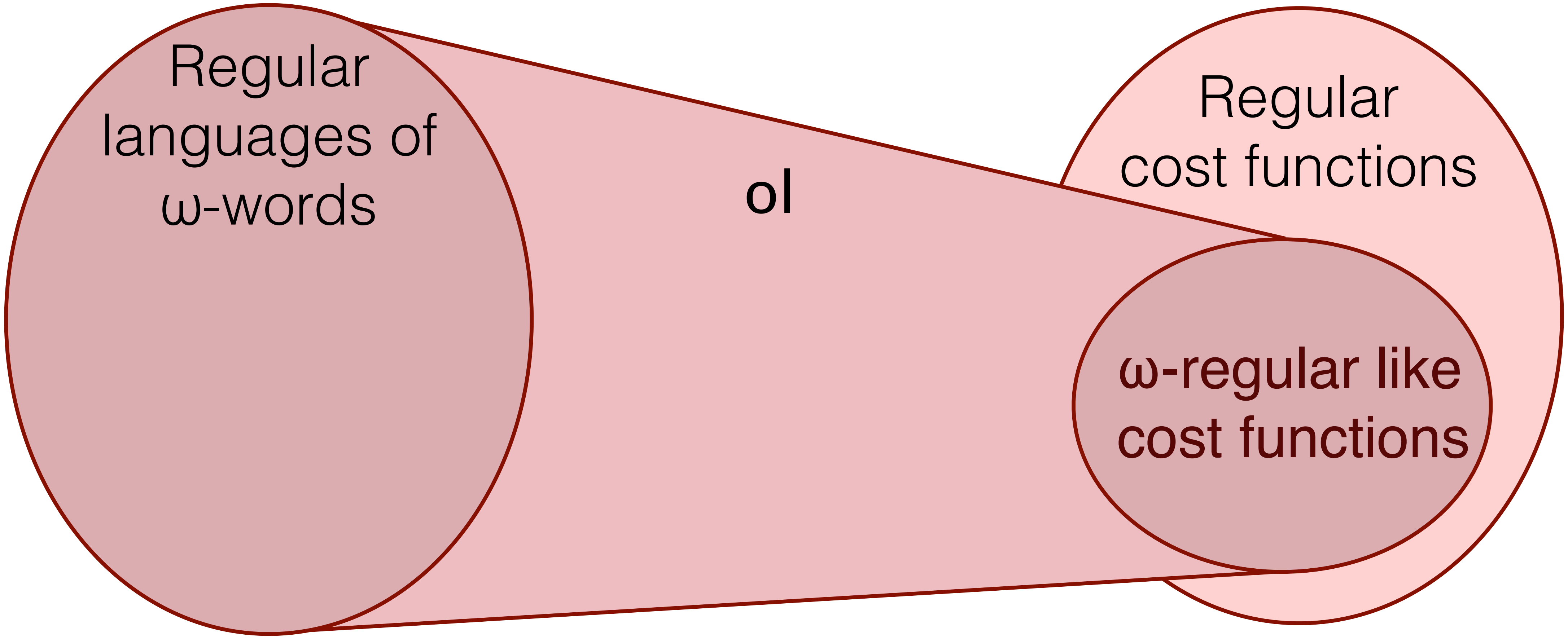

Regular
languages of
ω-words
ol
Regular
cost functions
ω-regular like
cost functions

The bridge



Consequence: For all ω-regular like cost function, there exists effectively a B-deterministic a that recognizes it (up to ≈).

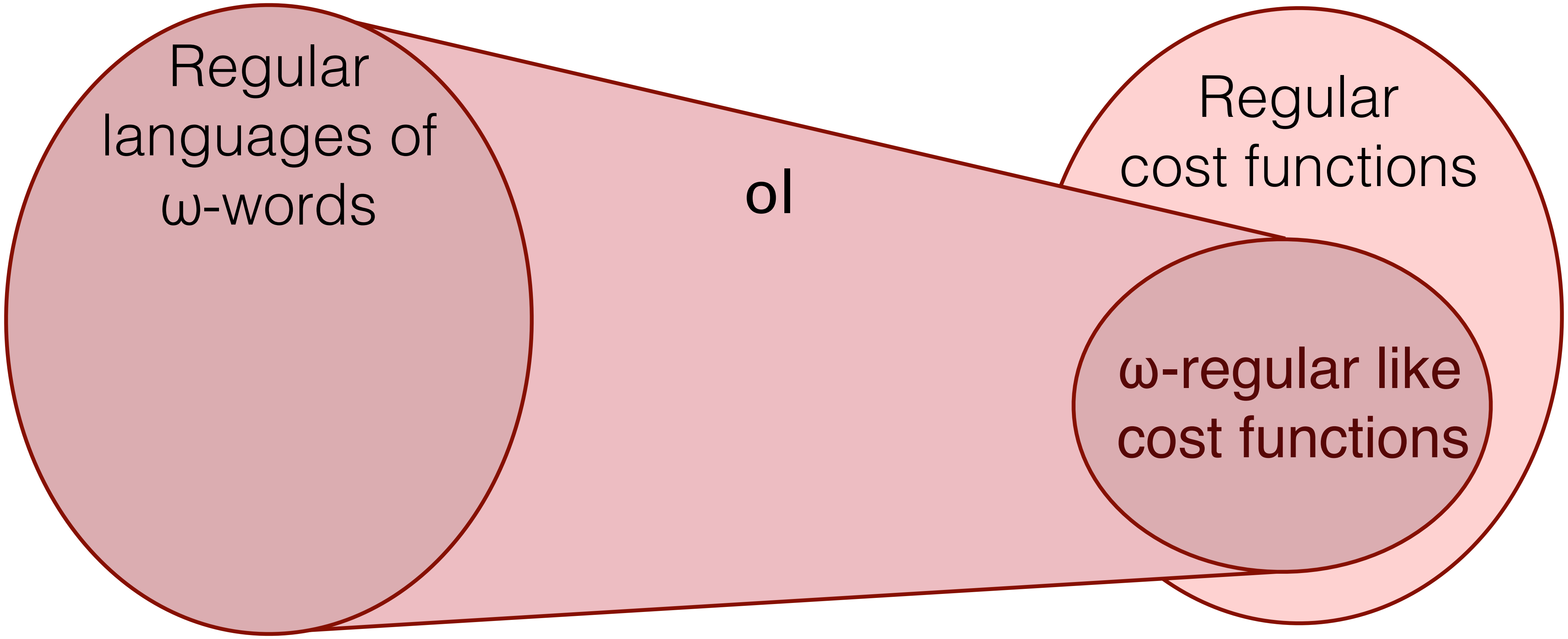
The bridge



Consequence: For all ω-regular like cost function, there exists effectively a B-deterministic a that recognizes it (up to ≈).

Proof: An ω-regular like cost function is of the form $L^{\mathtt{ol}}$ for some regular language of ω-words L .

The bridge

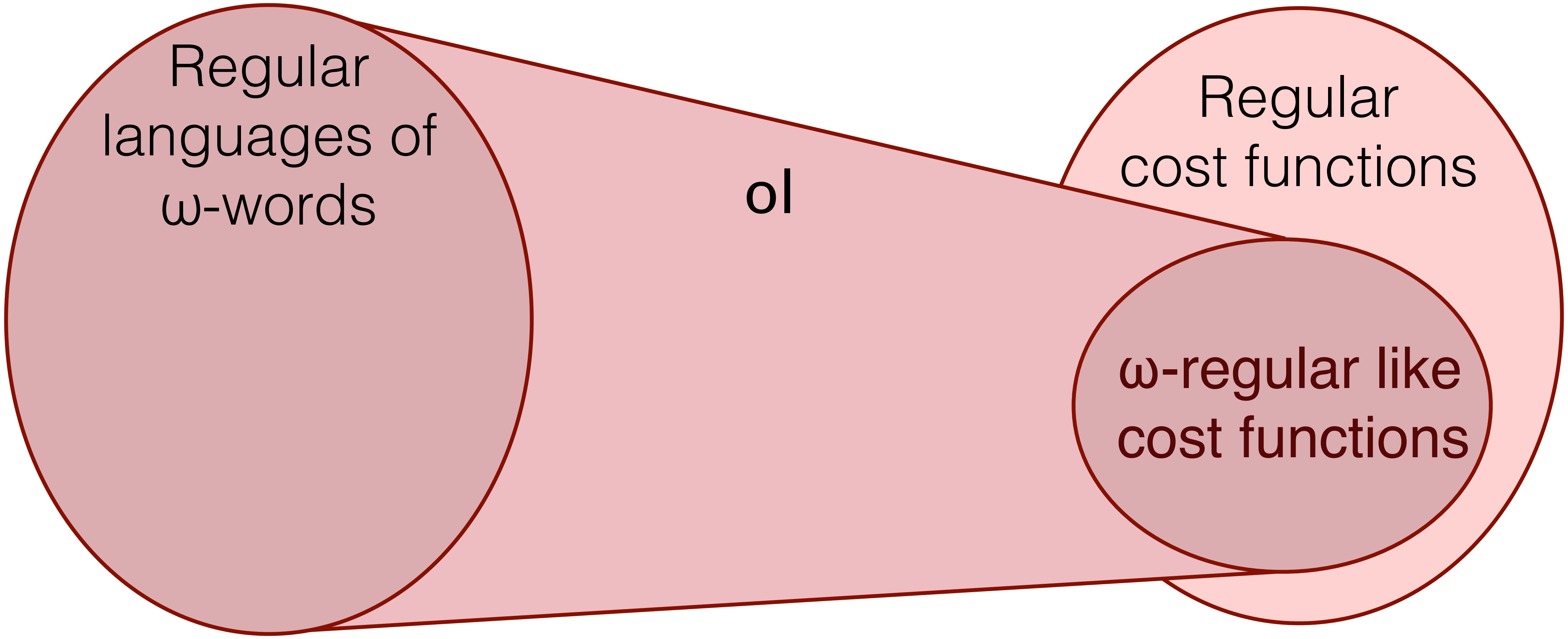


Consequence: For all ω-regular like cost function, there exists effectively a B-deterministic a that recognizes it (up to $\approx$).

Proof: An ω-regular like cost function is of the form $L^{\texttt{ol}}$ for some regular language of ω-words L .

There exists a deterministic Rabin automaton for it **[McNaughton/Safra]**.

The bridge



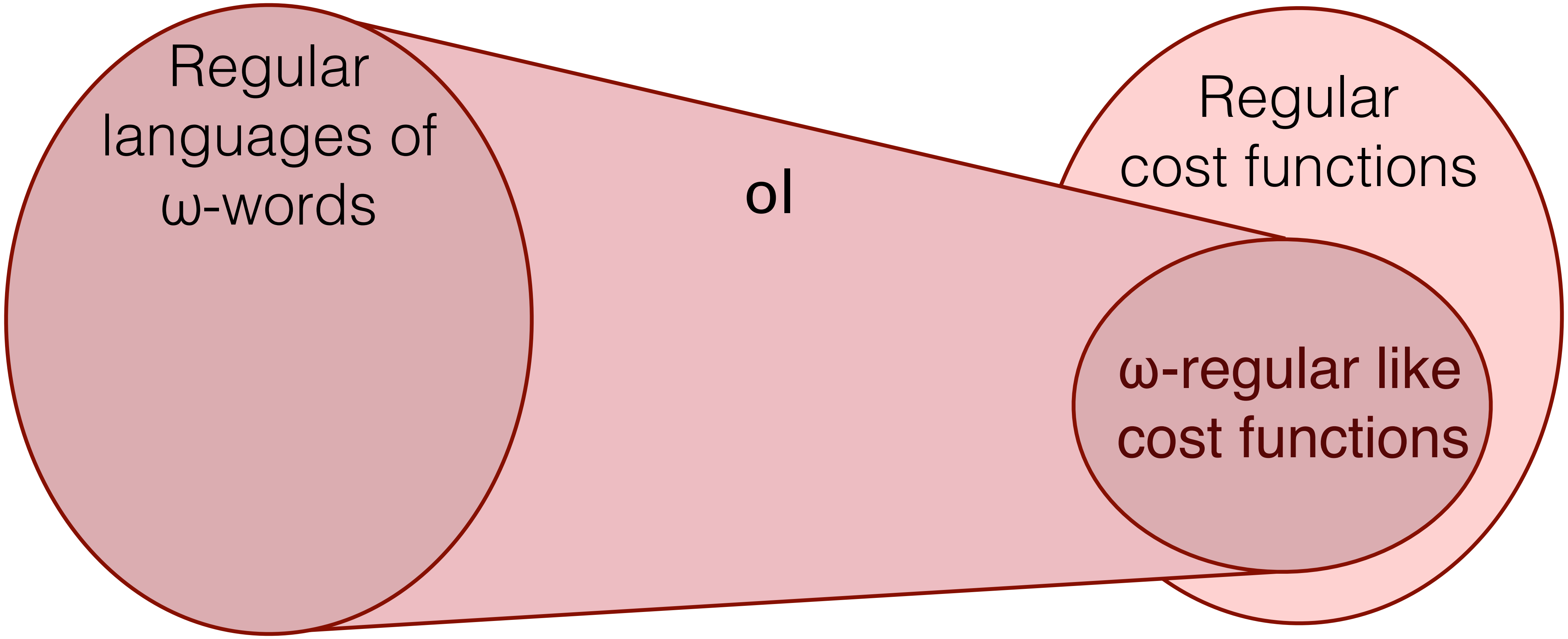
Consequence: For all ω-regular like cost function, there exists effectively a B-deterministic a that recognizes it (up to ≈).

Proof: An ω-regular like cost function is of the form $L^{\mathtt{ol}}$ for some regular language of ω-words L .

There exists a deterministic Rabin automaton for it **[McNaughton/Safra]**.

This is a max-prefix-B-automaton for $L^{\mathtt{ol}}$.

The bridge



Consequence: For all ω-regular like cost function, there exists effectively a B-deterministic a that recognizes it (up to ≈).

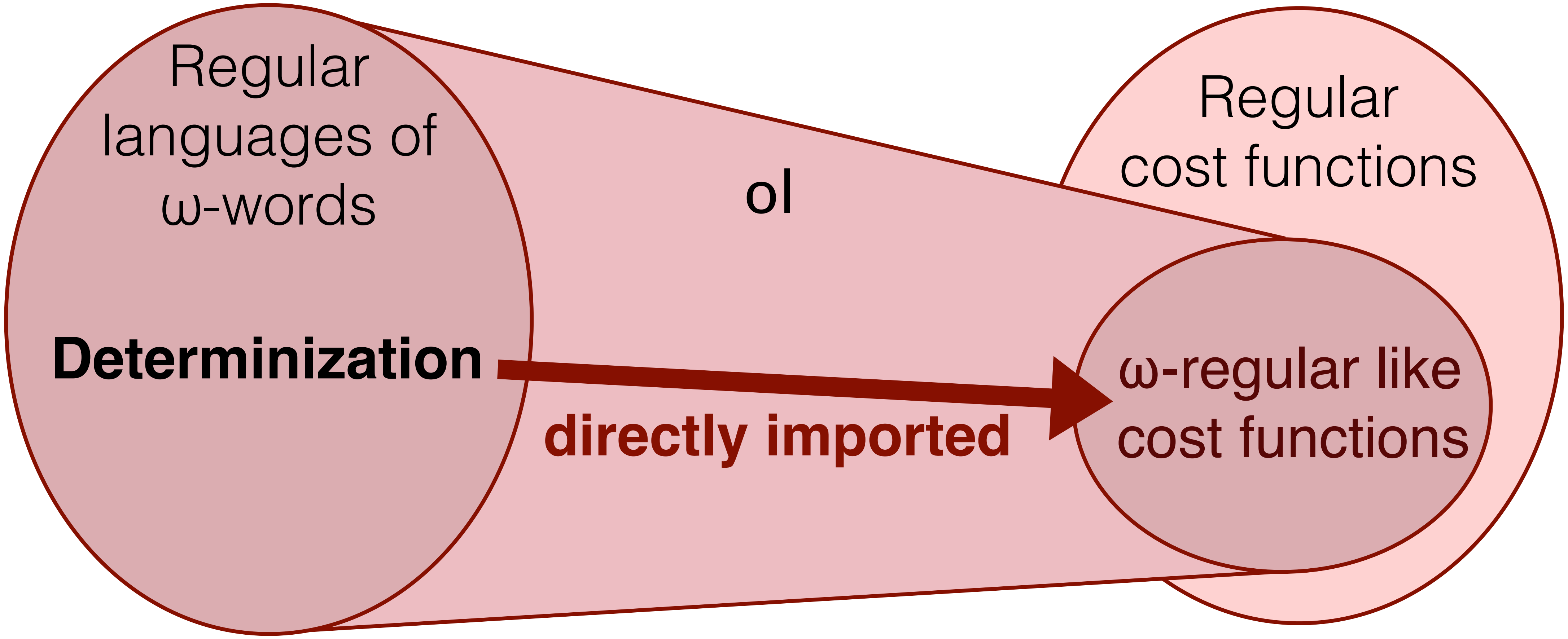
Proof: An ω-regular like cost function is of the form $L^{\mathtt{ol}}$ for some regular language of ω-words L .

There exists a deterministic Rabin automaton for it **[McNaughton/Safra]**.

This is a max-prefix-B-automaton for $L^{\mathtt{ol}}$.

Since it is deterministic and complete it is in fact a deterministic-B-automaton (all states set to final).

The bridge



Consequence: For all ω-regular like cost function, there exists effectively a B-deterministic a that recognizes it (up to ≈).

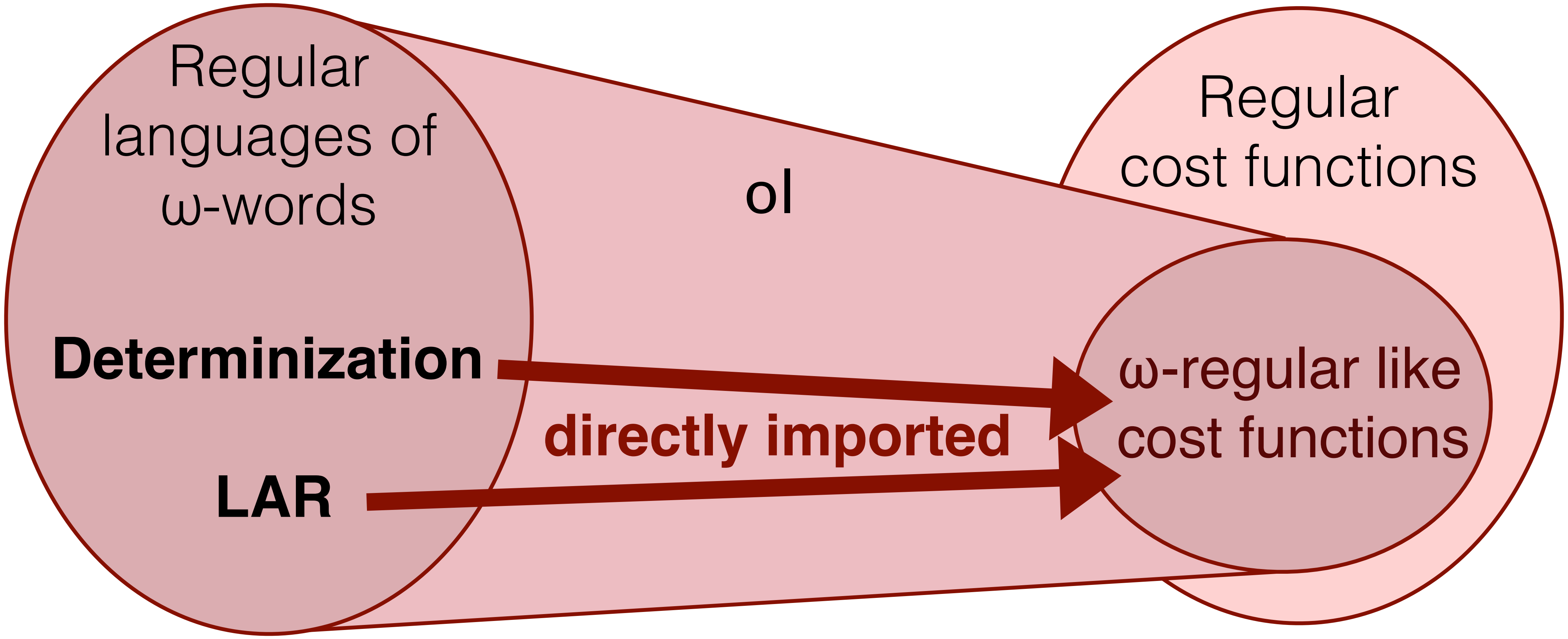
Proof: An ω-regular like cost function is of the form $L^{\mathtt{ol}}$ for some regular language of ω-words L .

There exists a deterministic Rabin automaton for it **[McNaughton/Safra]**.

This is a max-prefix-B-automaton for $L^{\mathtt{ol}}$.

Since it is deterministic and complete it is in fact a deterministic-B-automaton (all states set to final).

The bridge



Consequence: For all ω-regular like cost function, there exists effectively a B-deterministic a that recognizes it (up to ≈).

Proof: An ω-regular like cost function is of the form L^{ol} for some regular language of ω-words L .

There exists a deterministic Rabin automaton for it **[McNaughton/Safra]**.

This is a max-prefix-B-automaton for L^{ol} .

Since it is deterministic and complete it is in fact a deterministic-B-automaton (all states set to final).

Why do we care ?

Why do we care ?

I am interested in regular cost functions, not the special case of ω-regular like cost functions. So why do I care if this subclass inherits all the good properties and constructions of the regular languages of ω-words?

What is history-determinism?

(good-for-games automata **[Henzinger&Piterman]**)

What is history-determinism?

(good-for-games automata **[Henzinger&Piterman]**)

An history deterministic automaton is a non-deterministic automaton such that an almighty oracle can decide what is the best transition to take knowing the run so far.

What is history-determinism?

(good-for-games automata **[Henzinger&Piterman]**)

An history deterministic automaton is a non-deterministic automaton such that an almighty oracle can decide what is the best transition to take knowing the run so far.

Example one-counter-B: A counter can be incremented or reset.
Maximal value counts.
Infimum over all runs.

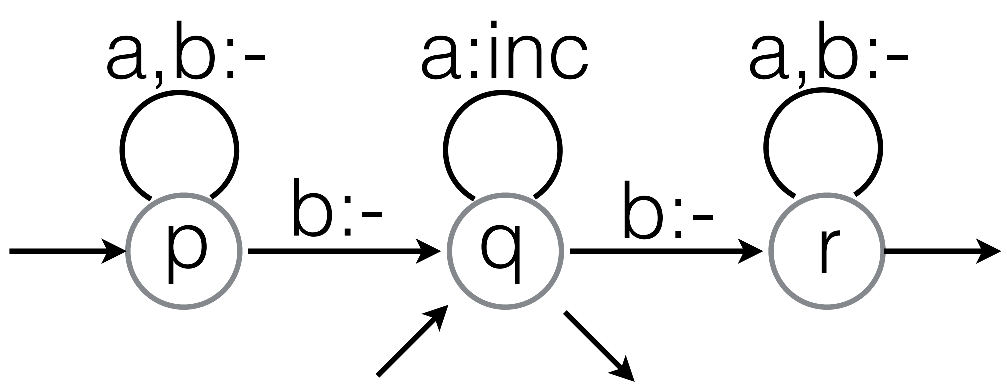
What is history-determinism?

(good-for-games automata **[Henzinger&Piterman]**)

An history deterministic automaton is a non-deterministic automaton such that an almighty oracle can decide what is the best transition to take knowing the run so far.

Example one-counter-B:

A counter can be incremented or reset.
Maximal value counts.
Infimum over all runs.

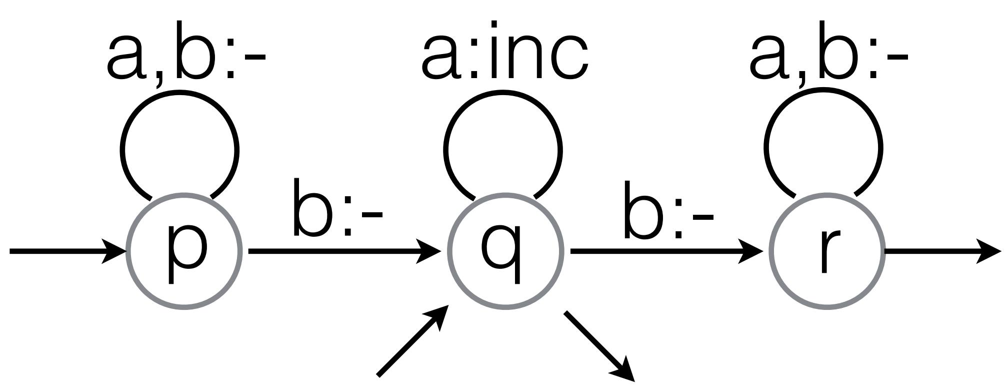


What is history-determinism?

(good-for-games automata **[Henzinger&Piterman]**)

An history deterministic automaton is a non-deterministic automaton such that an almighty oracle can decide what is the best transition to take knowing the run so far.

Example one-counter-B: A counter can be incremented or reset.
Maximal value counts.
Infimum over all runs.



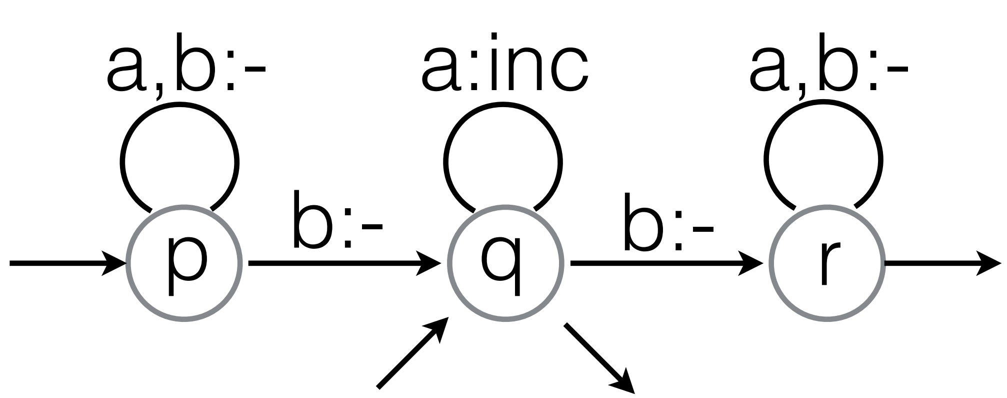
$$\mathtt{minblock}(a^{n_0}ba^{n_1}\dots ba^{n_k}) = \min(n_0,\dots,n_k)$$

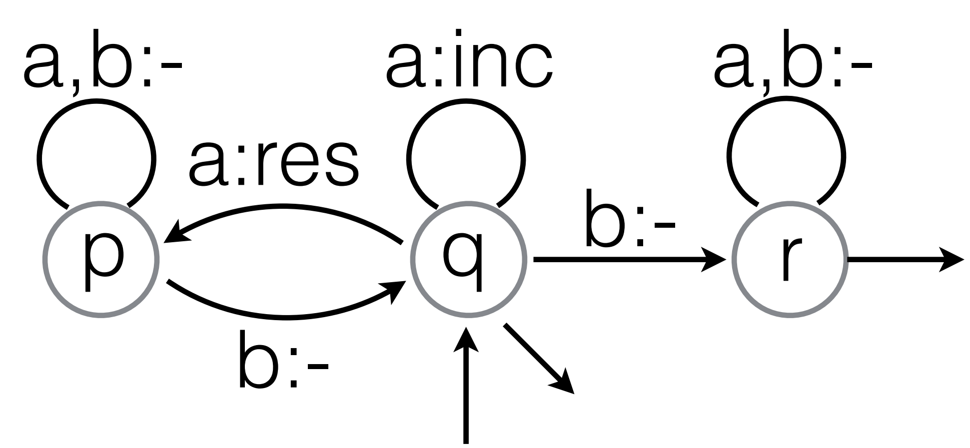
What is history-determinism?

(good-for-games automata **[Henzinger&Piterman]**)

An history deterministic automaton is a non-deterministic automaton such that an almighty oracle can decide what is the best transition to take knowing the run so far.

Example one-counter-B: A counter can be incremented or reset.
Maximal value counts.
Infimum over all runs.





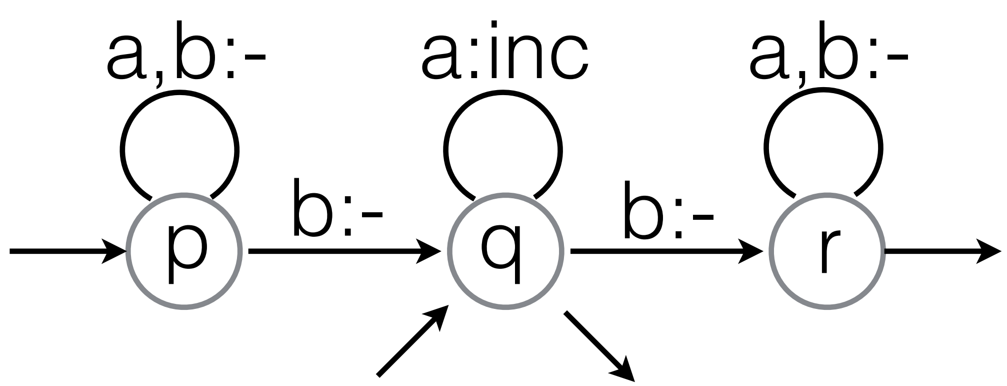
$$\texttt{minblock}(a^{n_0}ba^{n_1}\dots ba^{n_k}) = \min(n_0,\dots,n_k)$$

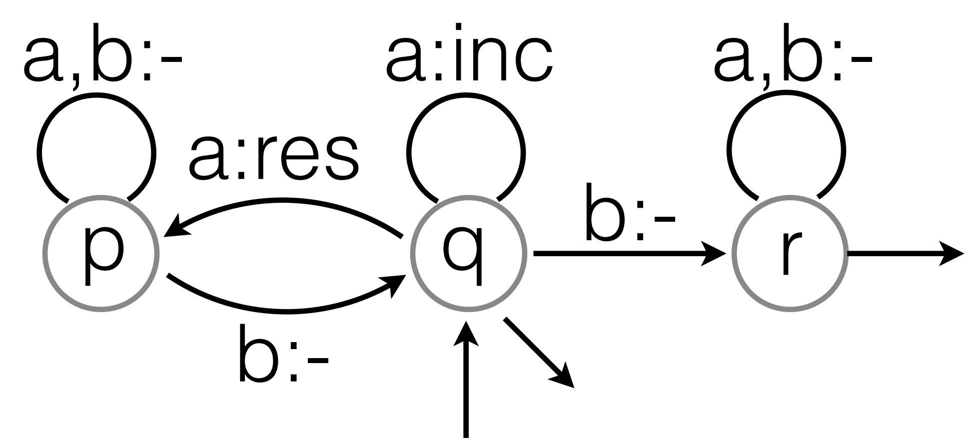
What is history-determinism?

(good-for-games automata **[Henzinger&Piterman]**)

An history deterministic automaton is a non-deterministic automaton such that an almighty oracle can decide what is the best transition to take knowing the run so far.

Example one-counter-B: A counter can be incremented or reset.
Maximal value counts.
Infimum over all runs.





$$\texttt{minblock}(a^{n_0}ba^{n_1}\dots ba^{n_k}) = \min(n_0,\dots,n_k)$$

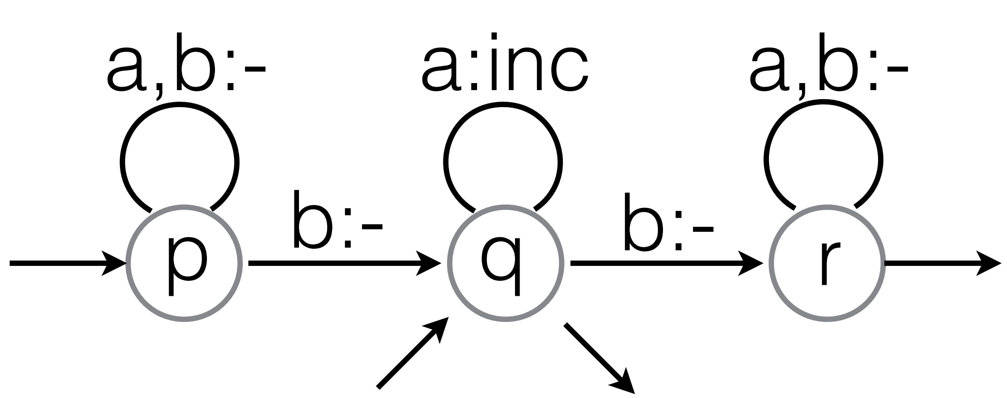
These automata are semantically deterministic, but not syntactically.

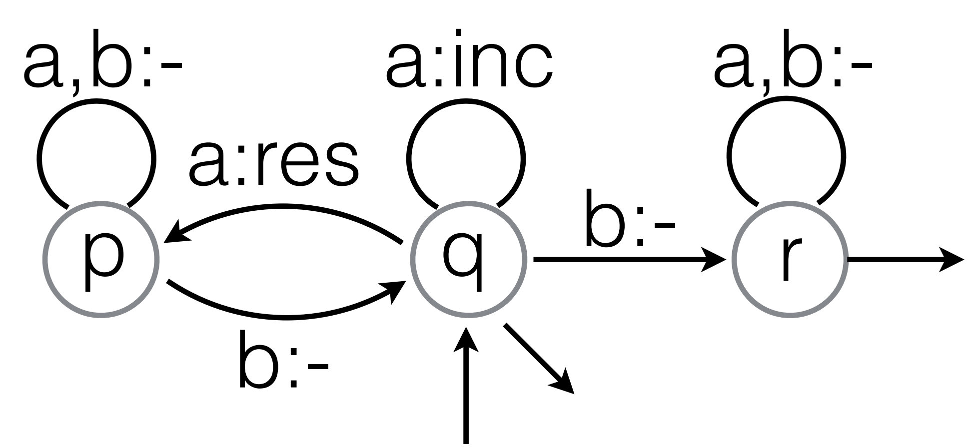
What is history-determinism?

(good-for-games automata **[Henzinger&Piterman]**)

An history deterministic automaton is a non-deterministic automaton such that an almighty oracle can decide what is the best transition to take knowing the run so far.

Example one-counter-B: A counter can be incremented or reset.
Maximal value counts.
Infimum over all runs.





$$\texttt{minblock}(a^{n_0}ba^{n_1}\dots ba^{n_k}) = \min(n_0,\dots,n_k)$$

These automata are semantically deterministic, but not syntactically.

These are as good as deterministic automata when run in a branching context (i.e. a tree or a game).

Why do we care ?

I am interested in regular cost functions, not the special case of ω-regular like cost functions. So why do I care if this subclass inherits all the good properties and constructions of the regular languages of ω-words?

Why do we care ?

I am interested in regular cost functions, not the special case of ω-regular like cost functions. So why do I care if this subclass inherits all the good properties and constructions of the regular languages of ω-words?

Because (inspired from **[Bojanczyk15]**) :

determinization of ω-regular like cost functions + positional determinacy of hierarchical B-games **[C&Löding06]** = history-determinization of regular cost functions **[C09,C11 unp]**

Conclusion

We have provided a bridge from regular languages of infinite words to a subset of regular cost functions over finite words.

This allows to transport constructions and results from the well studied and simpler theory of regular languages of infinite words to cost functions over infinite trees.

In particular a new, simple and optimal proof for transforming B-automata in historic-deterministic form can be derived (a central result for working on games and trees).

Conclusion

We have provided a bridge from regular languages of infinite words to a subset of regular cost functions over finite words.

This allows to transport constructions and results from the well studied and simpler theory of regular languages of infinite words to cost functions over infinite trees.

In particular a new, simple and optimal proof for transforming B-automata in historic-deterministic form can be derived (a central result for working on games and trees).

TODO:
Extend the approach to produce history deterministic S-automata.