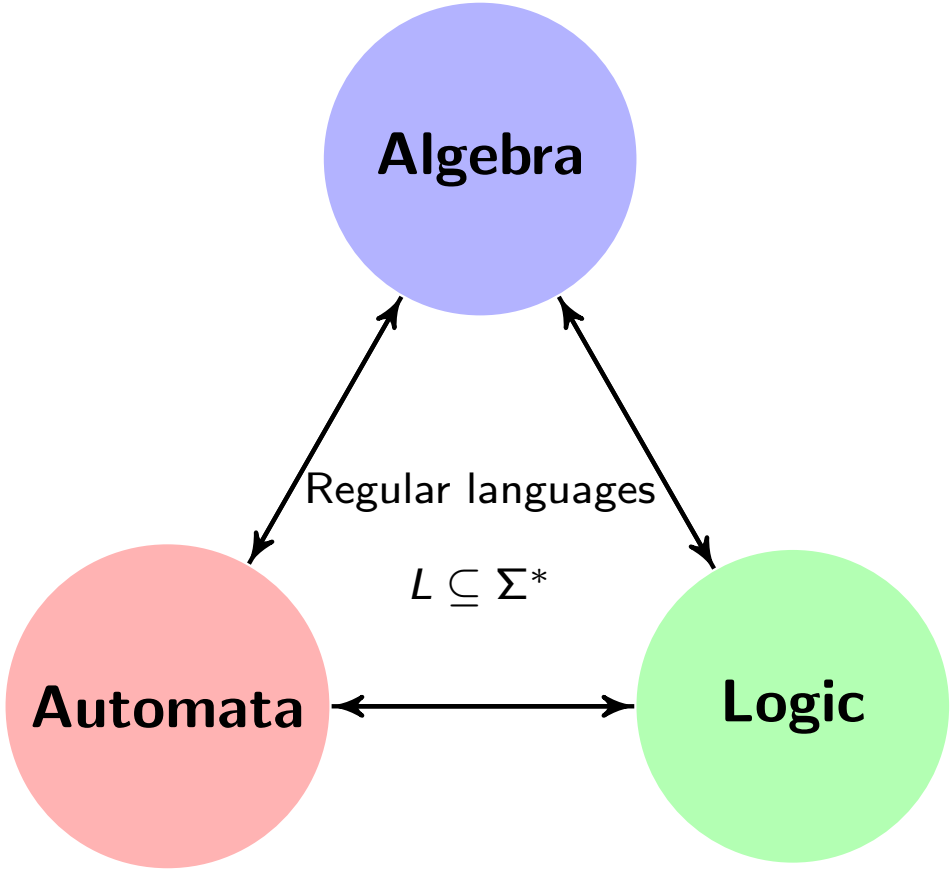
Automata, Logic and Algebra for (Finite) Word Transductions

Emmanuel Filiot

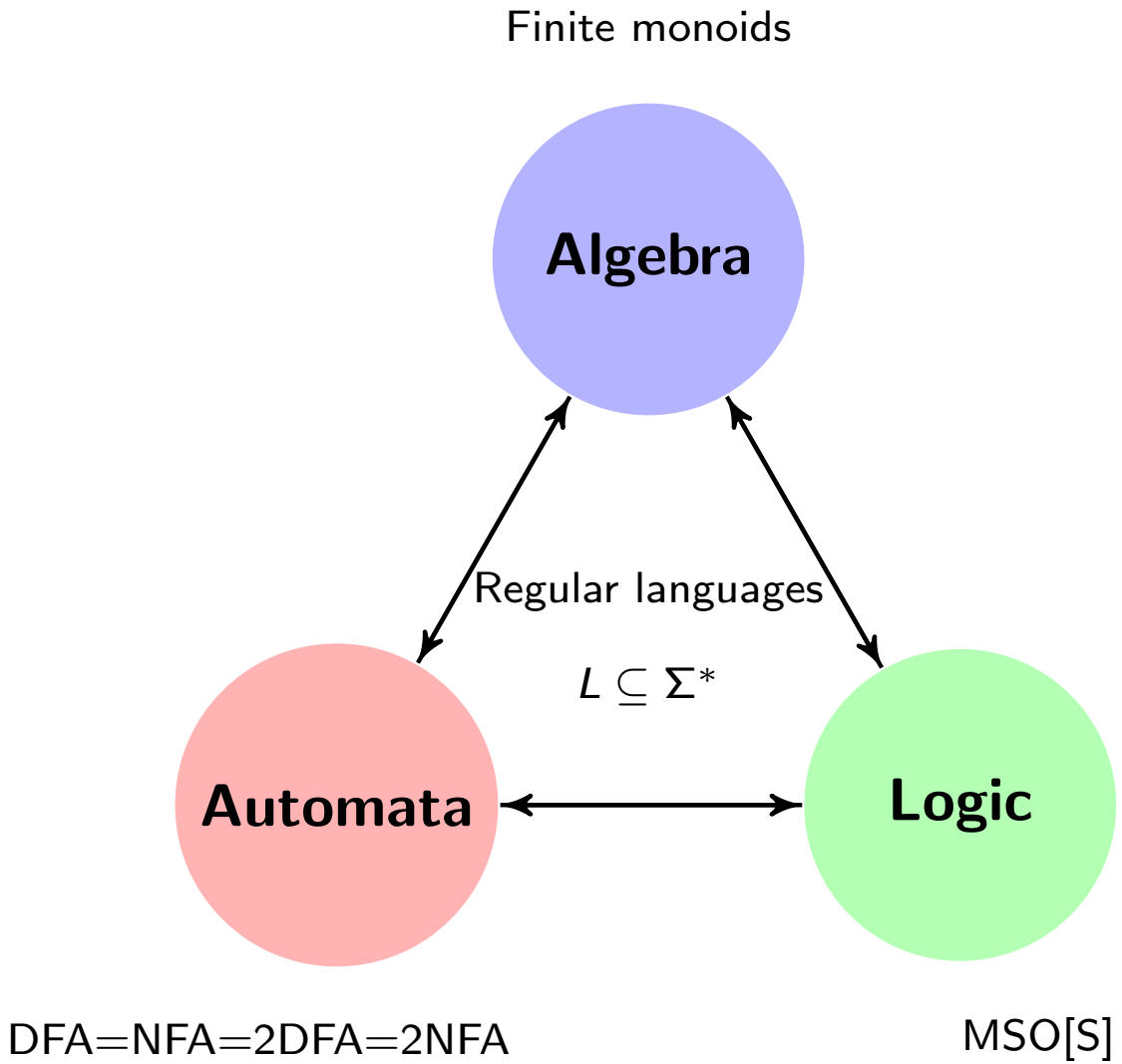
Université libre de Bruxelles & FNRS

Highlights of Logic, Automata and Games 2016

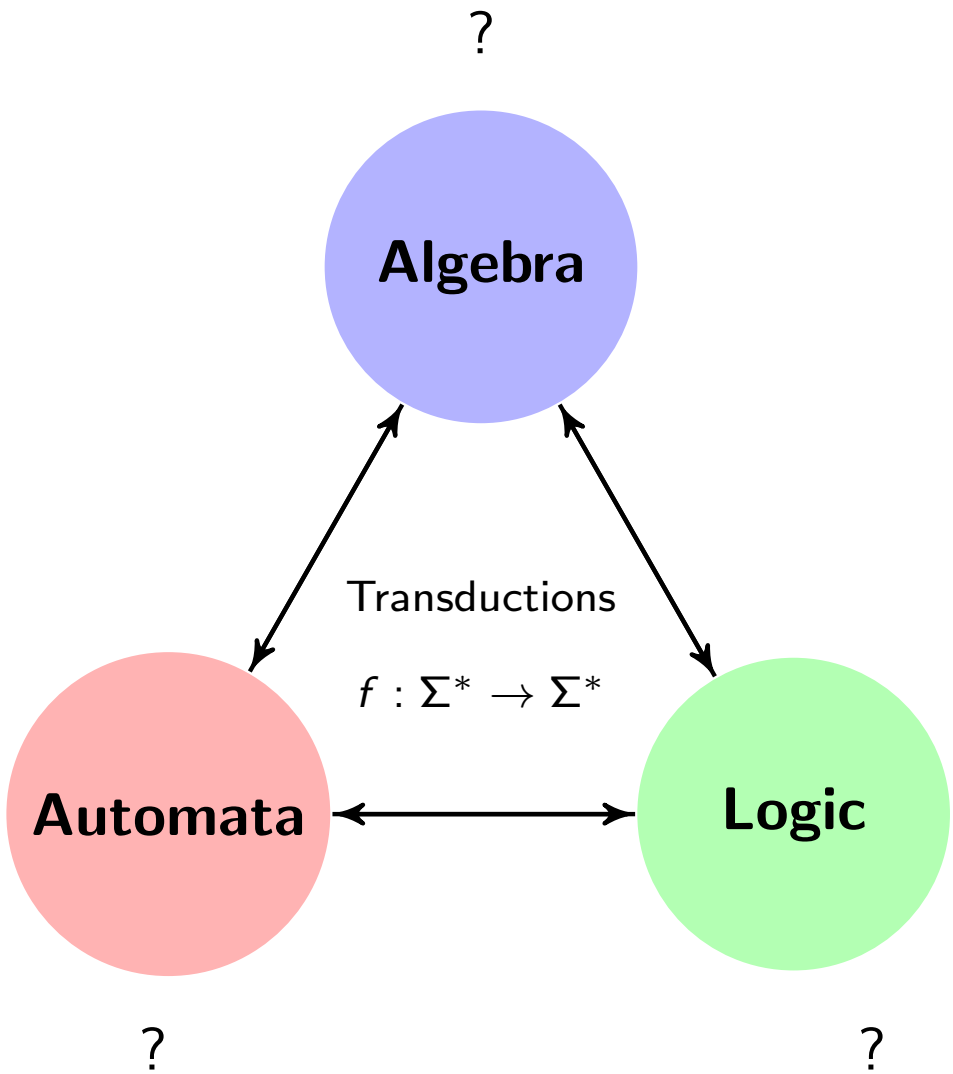
Trinity for Regular Languages



Trinity for Regular Languages

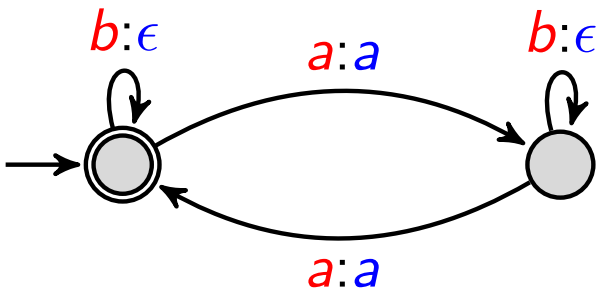


Objective of the talk



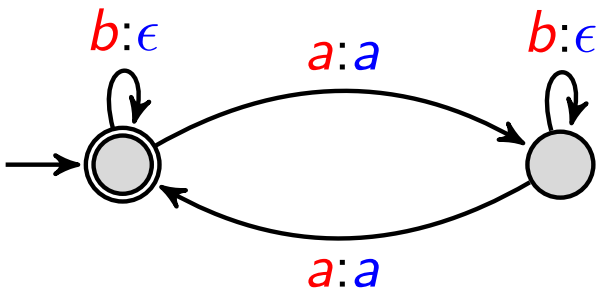
Automata for transductions: transducers

f_{del} :



Automata for transductions: transducers

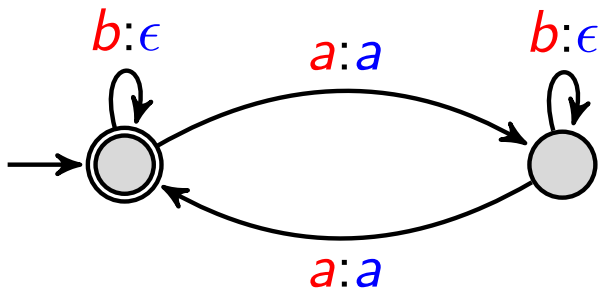
f_{del} :



$$aabaa \mapsto aaaa$$

Automata for transductions: transducers

f_{del} :

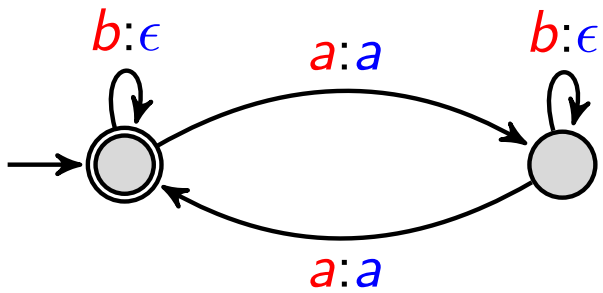


$$aabaa \mapsto aaaa$$

$$aaba \mapsto \text{undefined}$$

Automata for transductions: transducers

f_{del} :



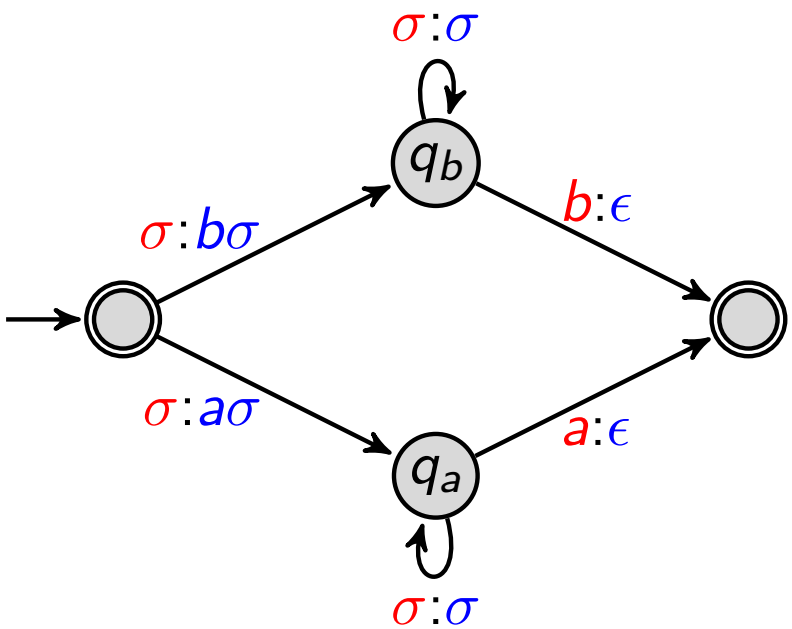
$$aabaa \mapsto aaaa$$

$$aaba \mapsto \text{undefined}$$

$$dom(f_{del}) = \text{'even number of } a\text{'}$$

Automata for transductions: transducers

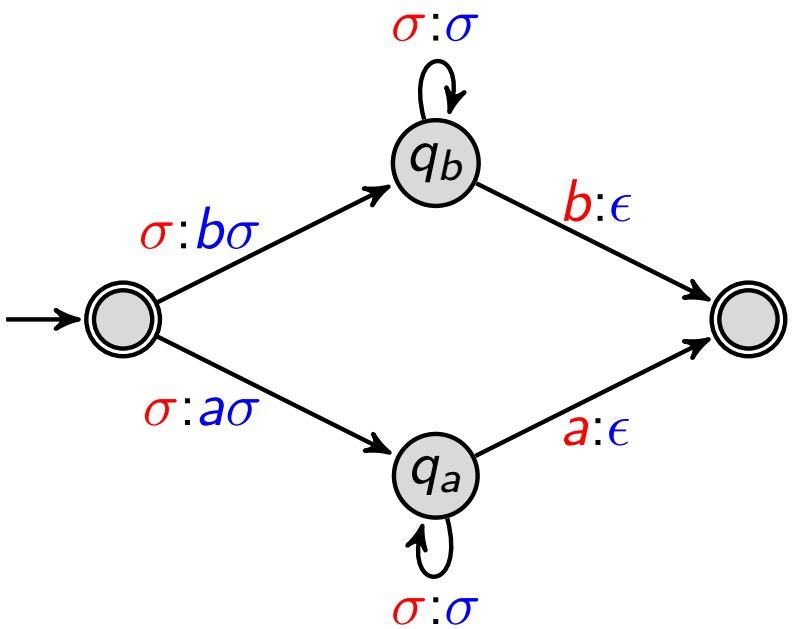
f_{sw} :



for all $\sigma \in \Sigma$

Automata for transductions: transducers

f_{sw} :

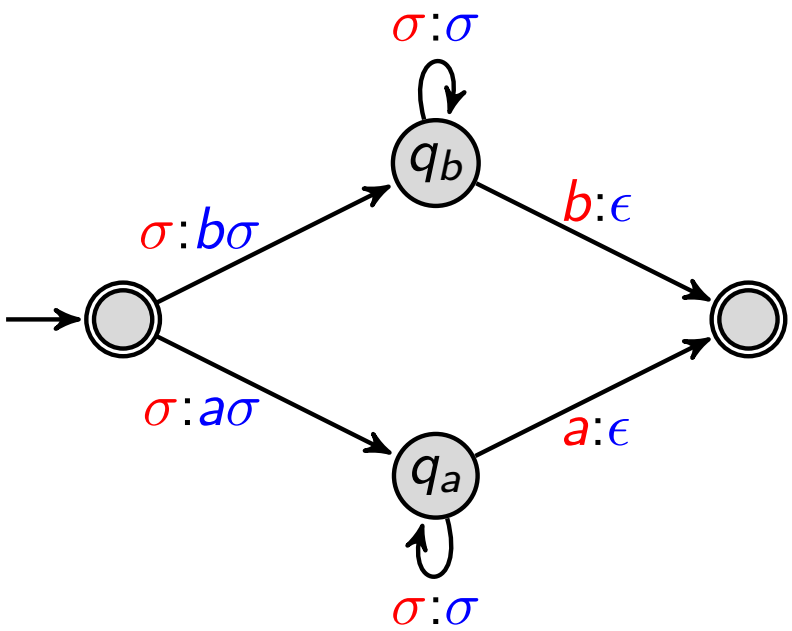


for all $\sigma \in \Sigma$

$$babaa \mapsto ababa$$

Automata for transductions: transducers

f_{sw} :



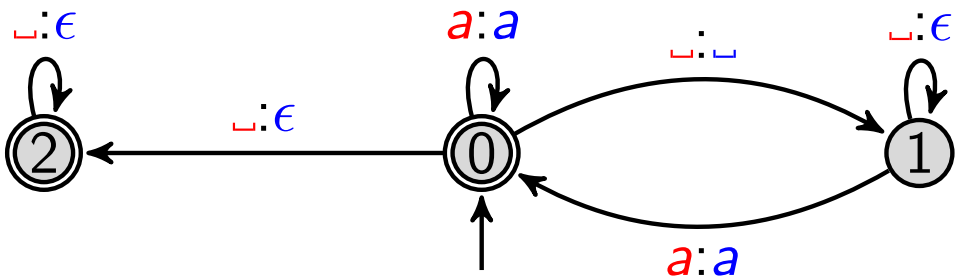
for all $\sigma \in \Sigma$

$$babaa \mapsto ababa$$

$$u\sigma \mapsto \sigma u \qquad |u| \geq 1$$

input-determinism (aka sequential) $<$ non-determinism

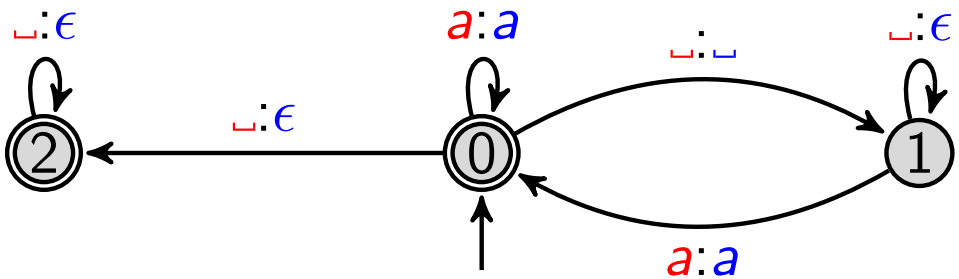
Automata for transductions: transducers



␣ = white space

Automata for transductions: transducers

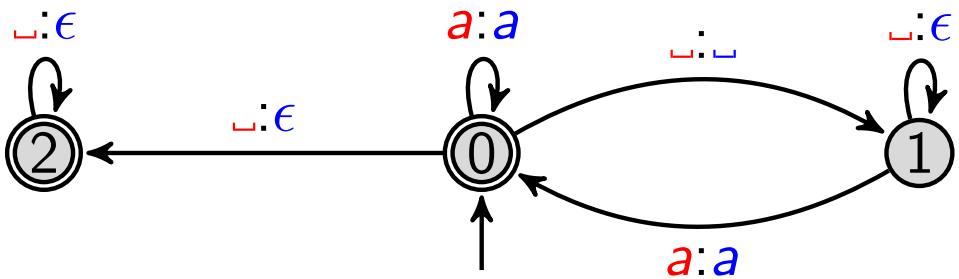
␣ = white space



␣␣aa␣␣␣a␣␣ ↦ ␣aa␣a

Automata for transductions: transducers

␣ = white space

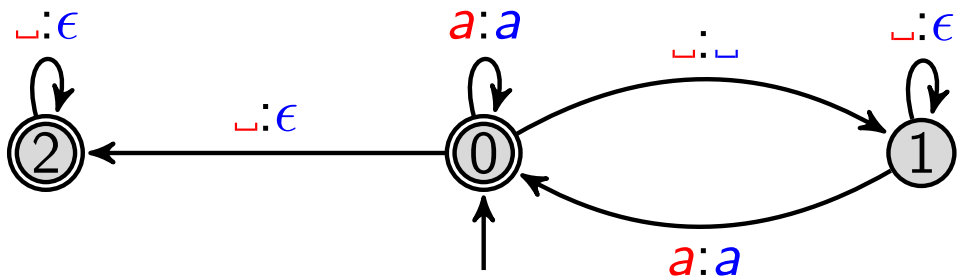


␣␣aa␣␣␣a␣␣ ↦ ␣aa␣a

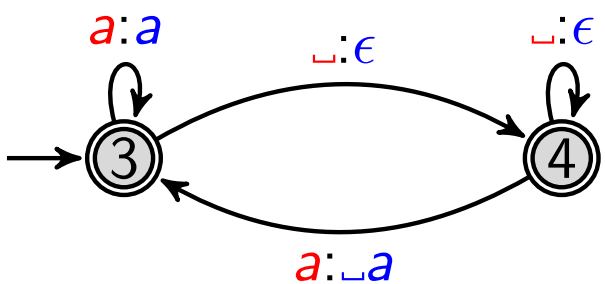
Is non-determinism needed ?

Automata for transductions: transducers

␣ = white space

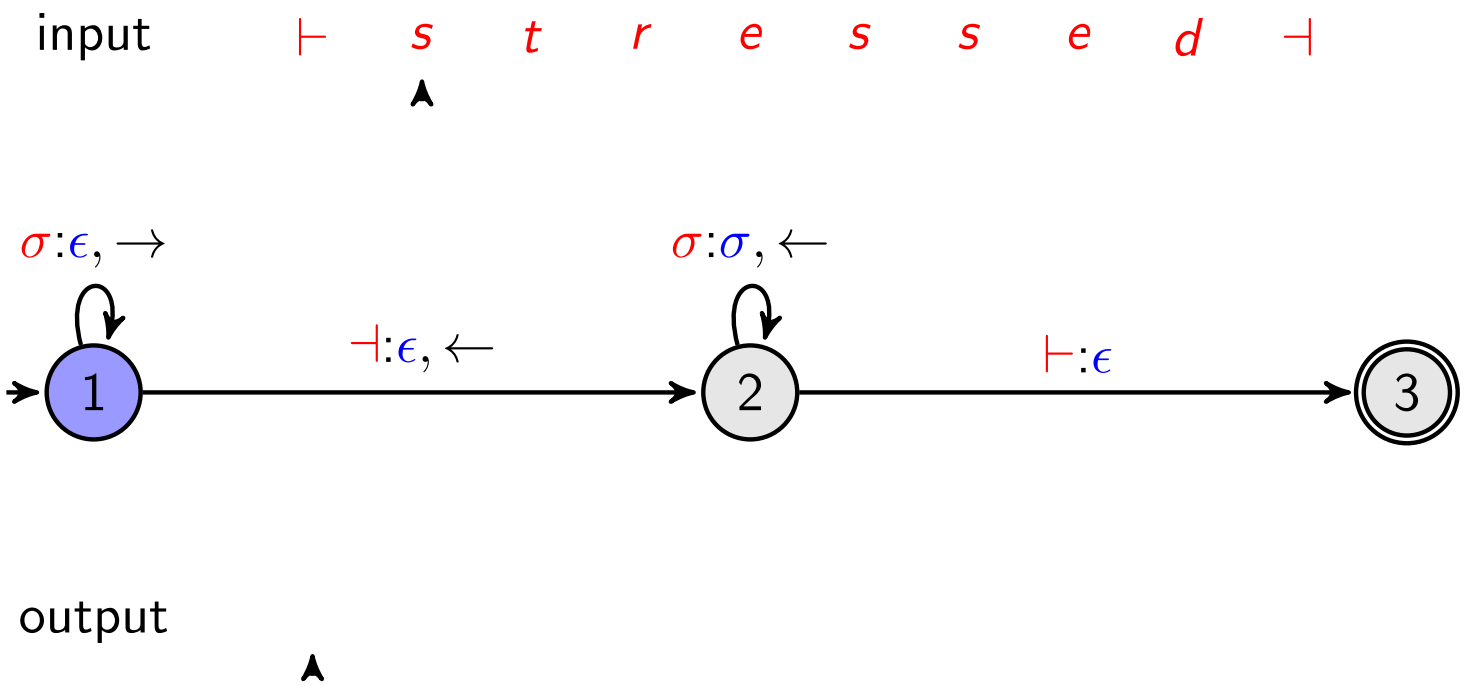


␣␣aa␣␣␣a␣␣ ↦ ␣aa␣a

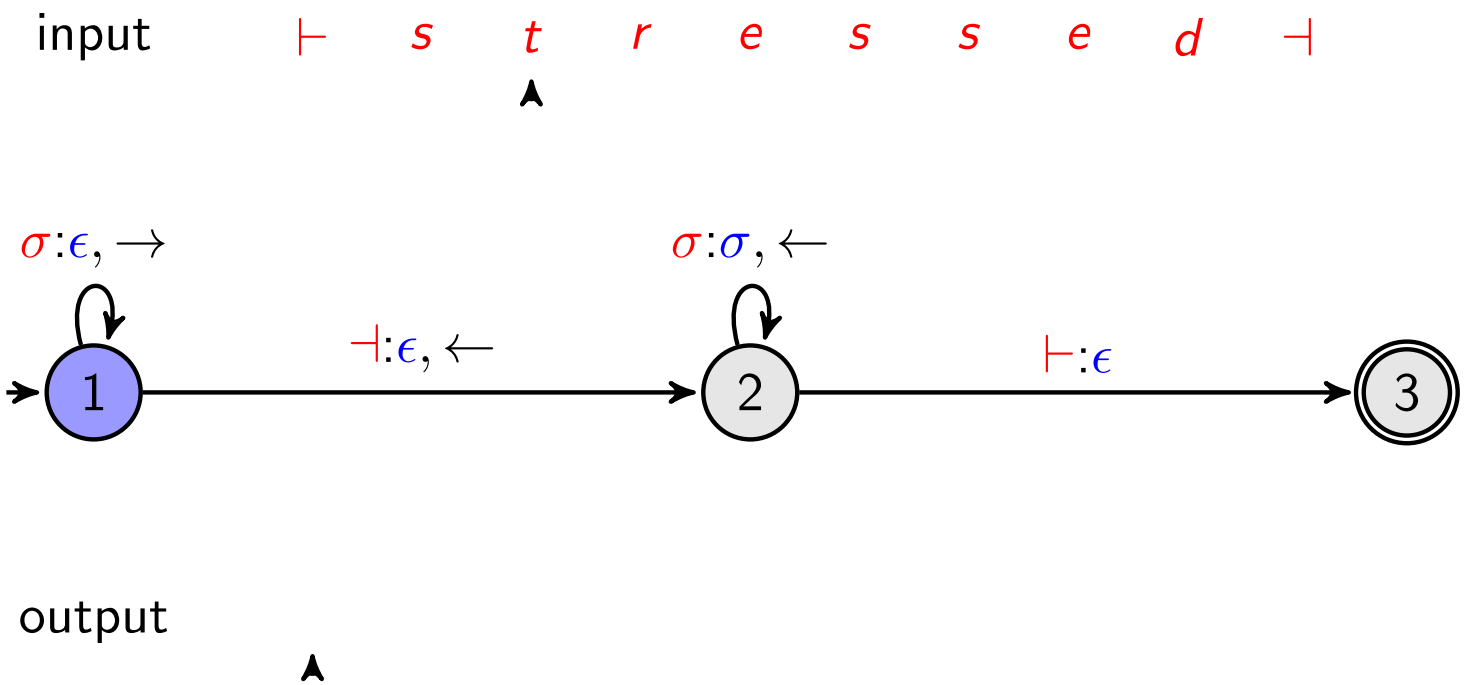


Is non-determinism needed ? No.

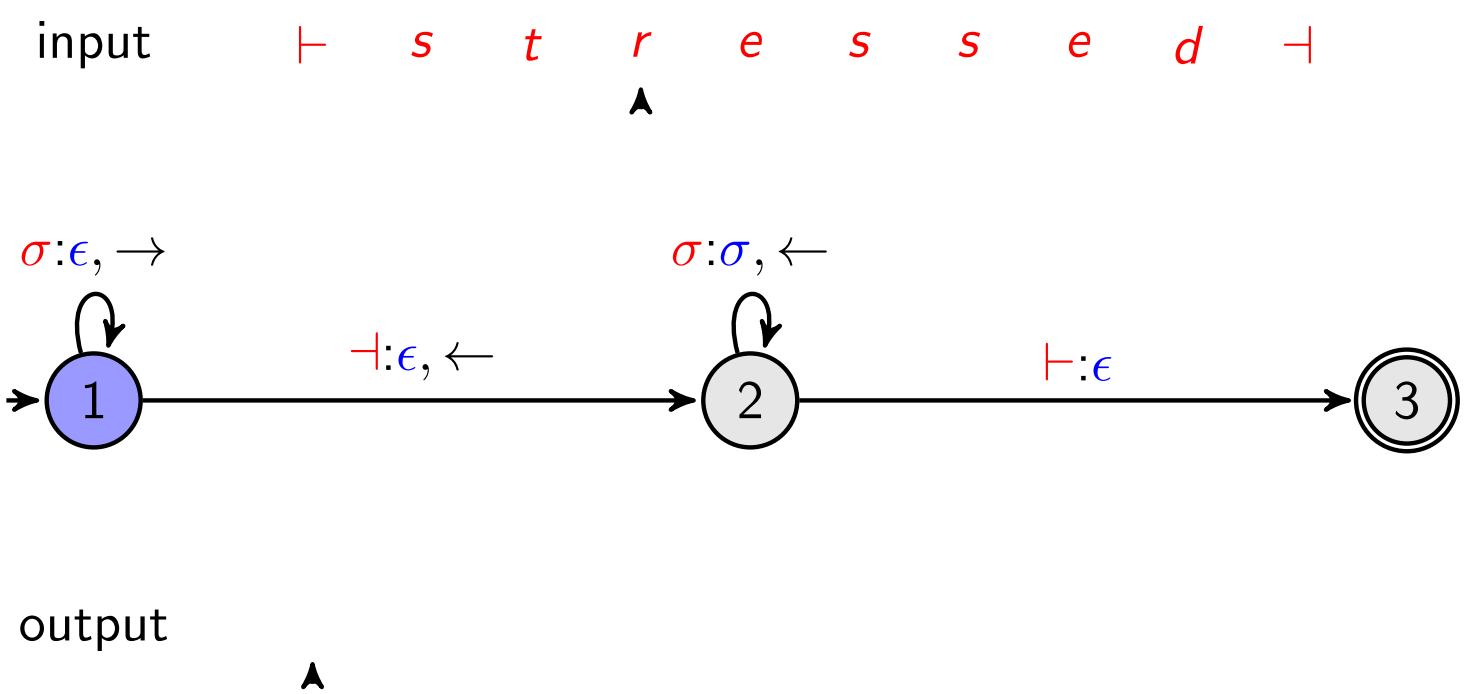
Automata for transductions: two-way transducers



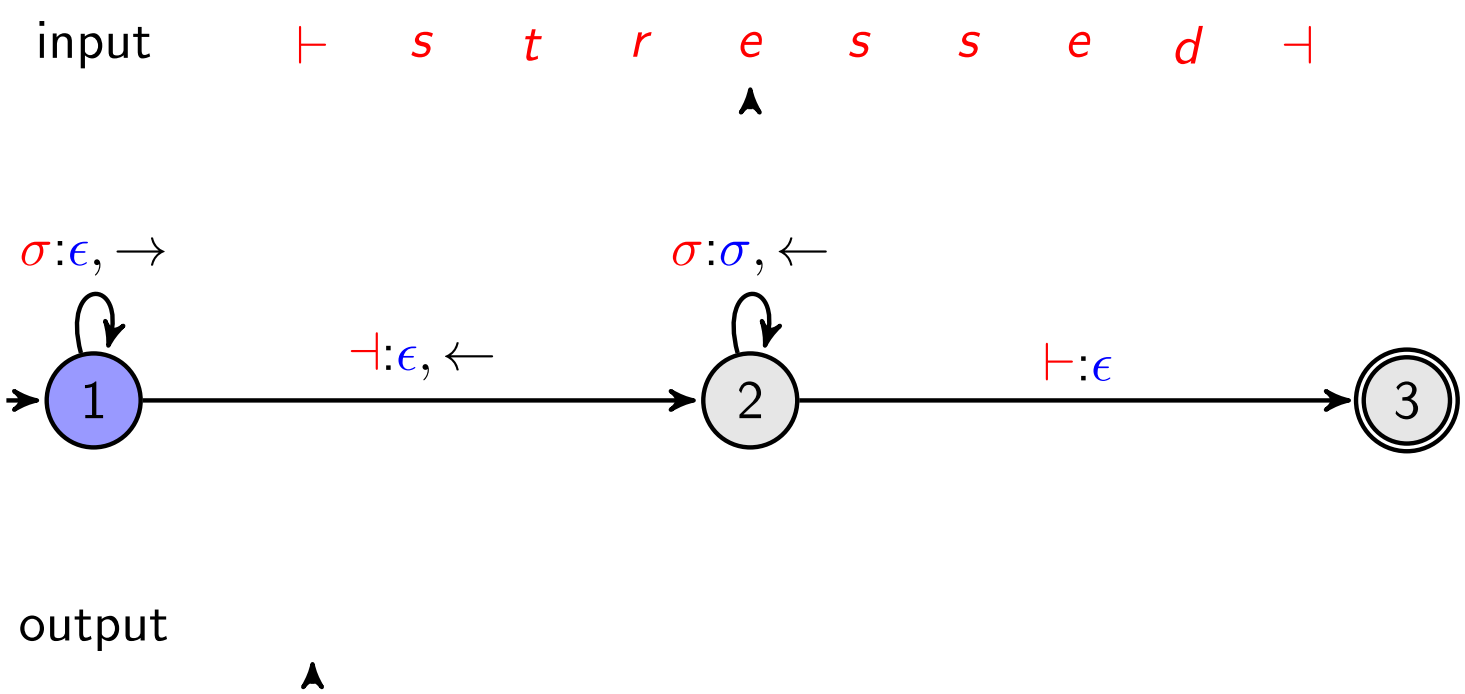
Automata for transductions: two-way transducers



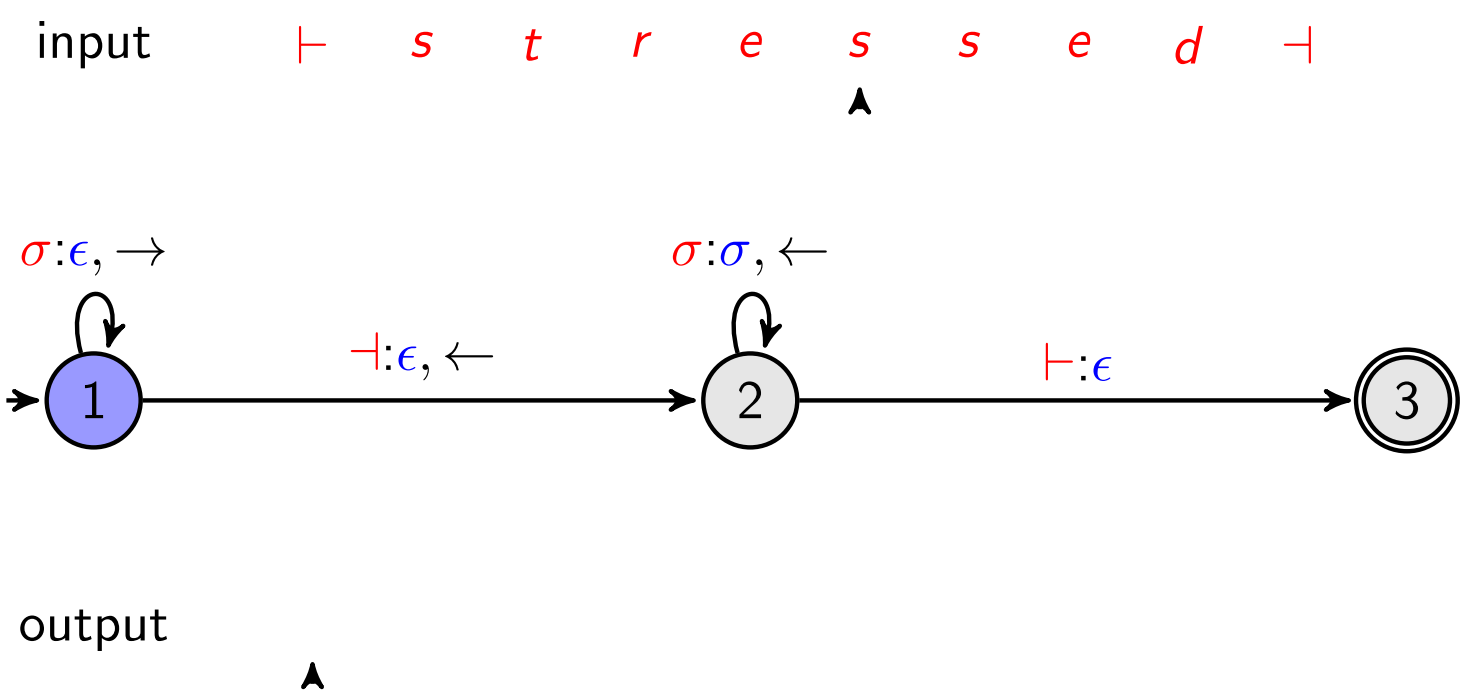
Automata for transductions: two-way transducers



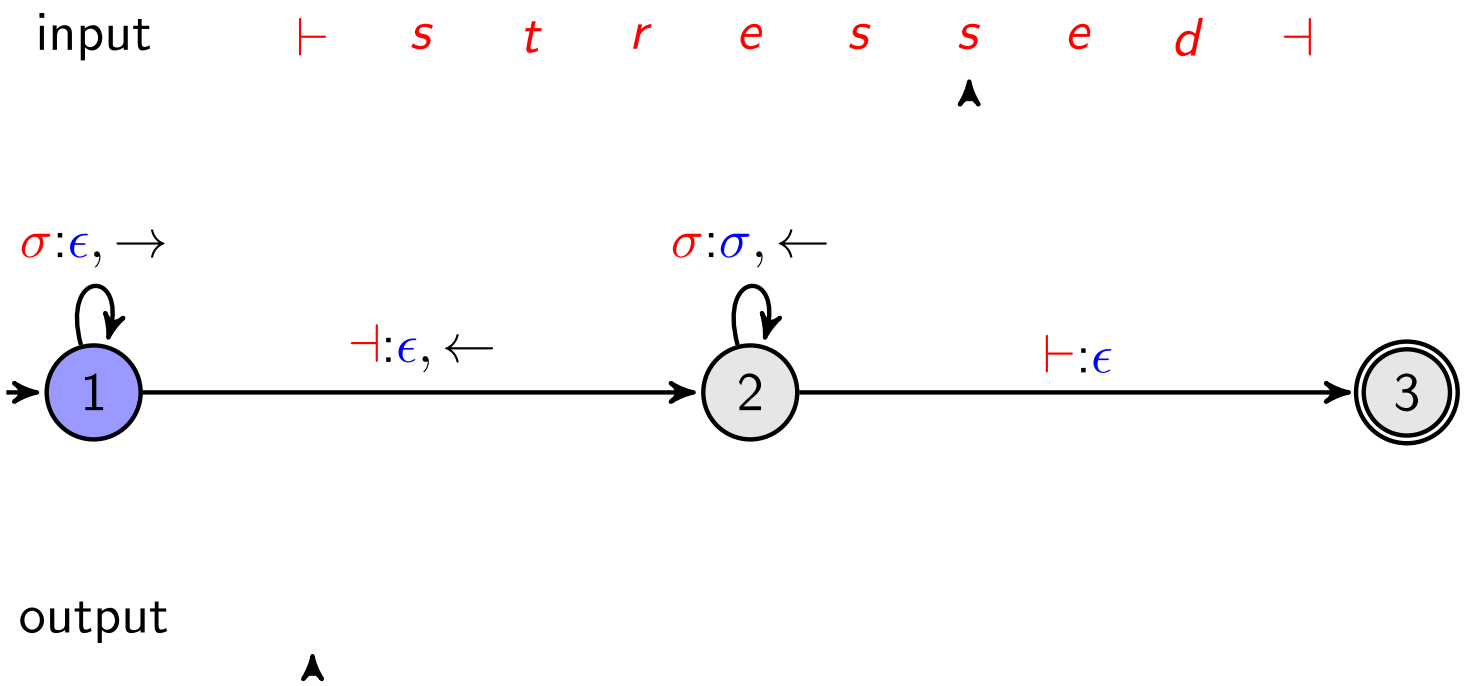
Automata for transductions: two-way transducers



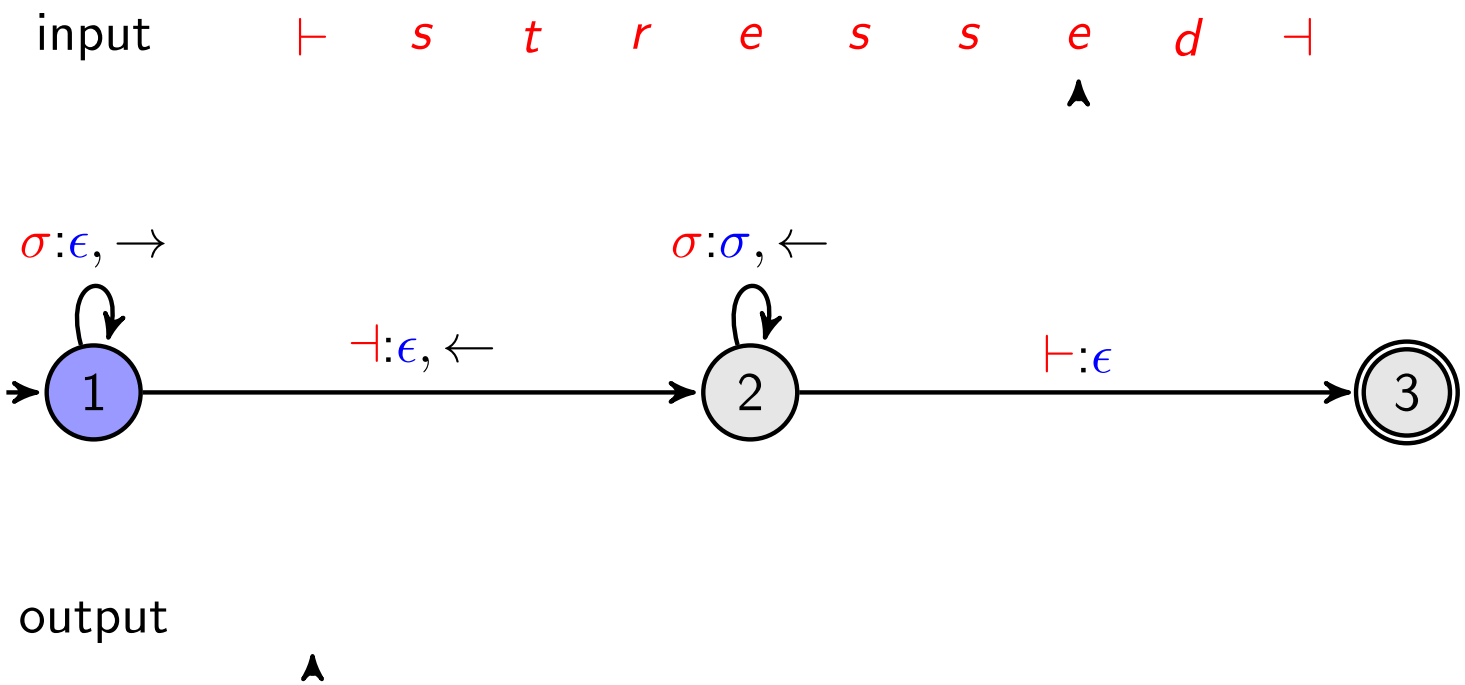
Automata for transductions: two-way transducers



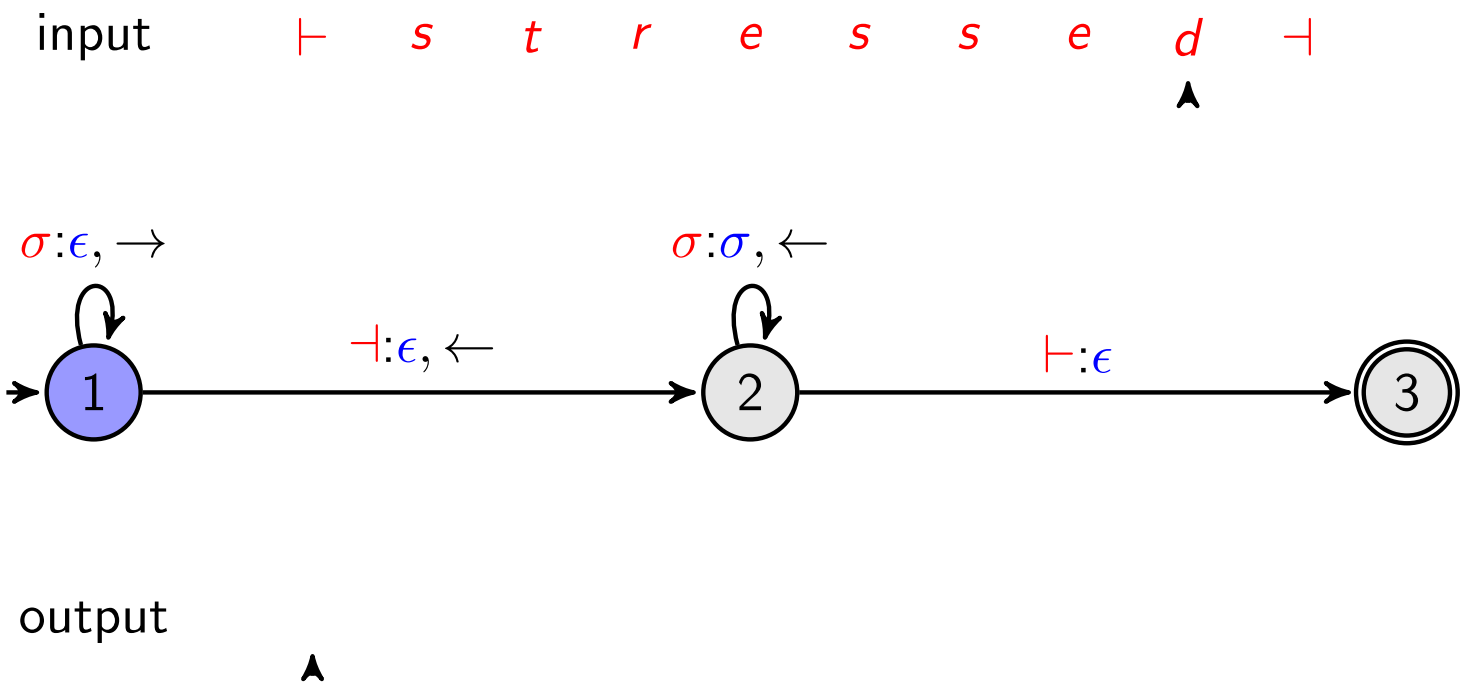
Automata for transductions: two-way transducers



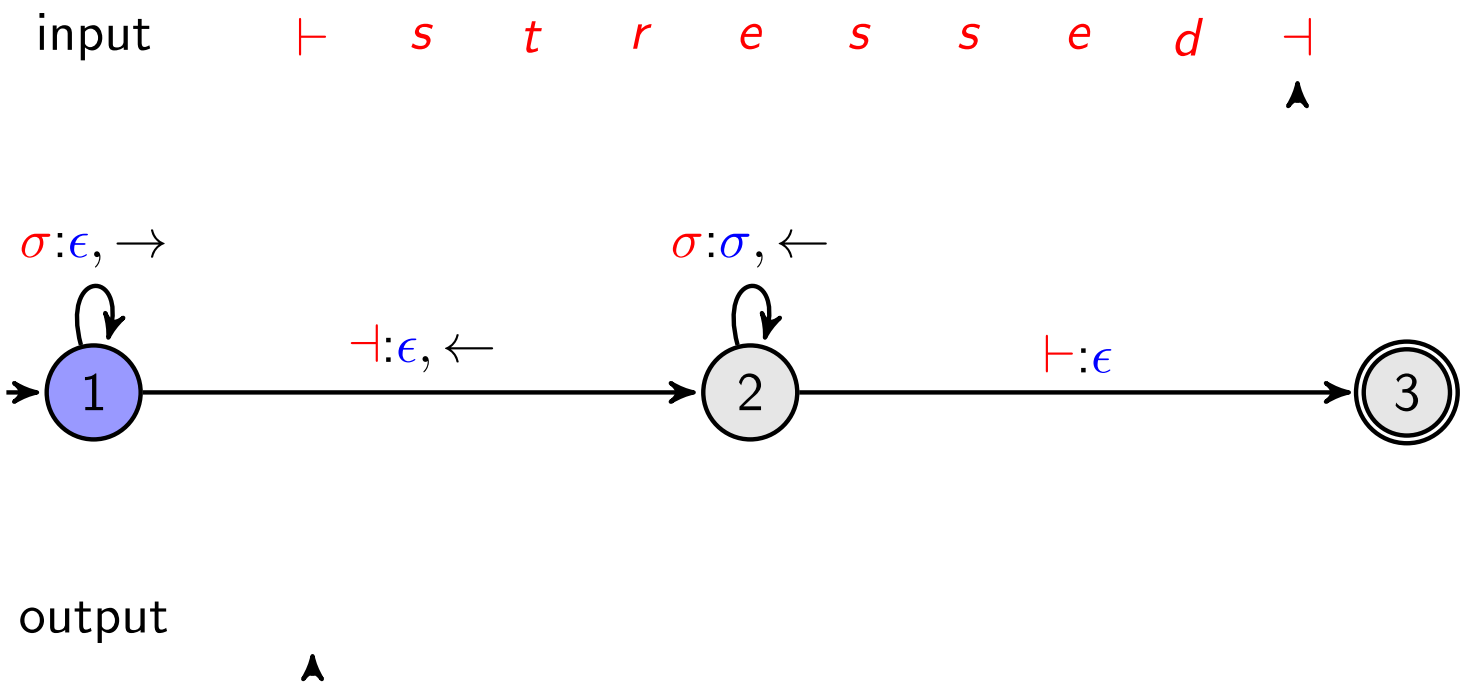
Automata for transductions: two-way transducers



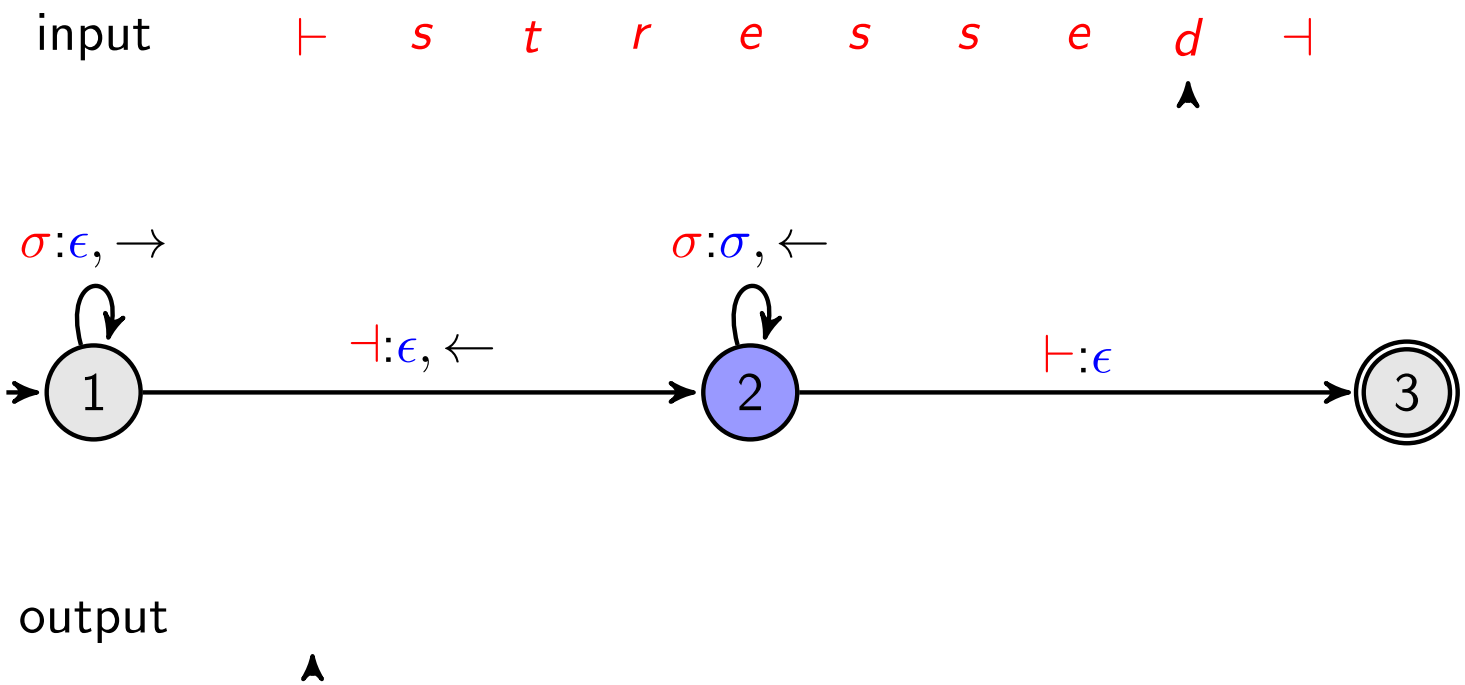
Automata for transductions: two-way transducers



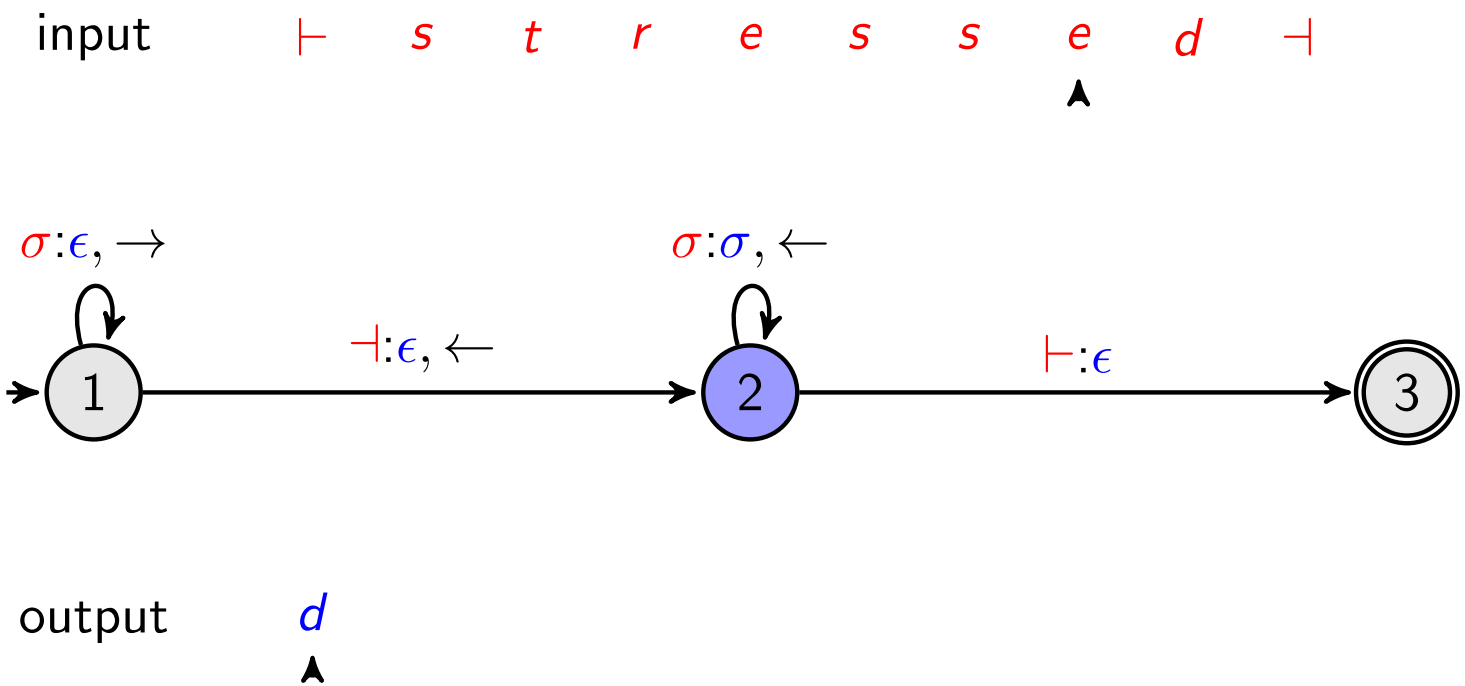
Automata for transductions: two-way transducers



Automata for transductions: two-way transducers

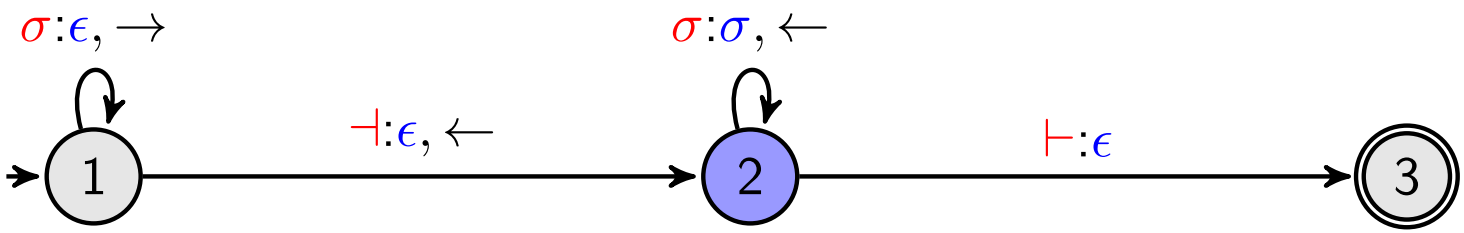


Automata for transductions: two-way transducers



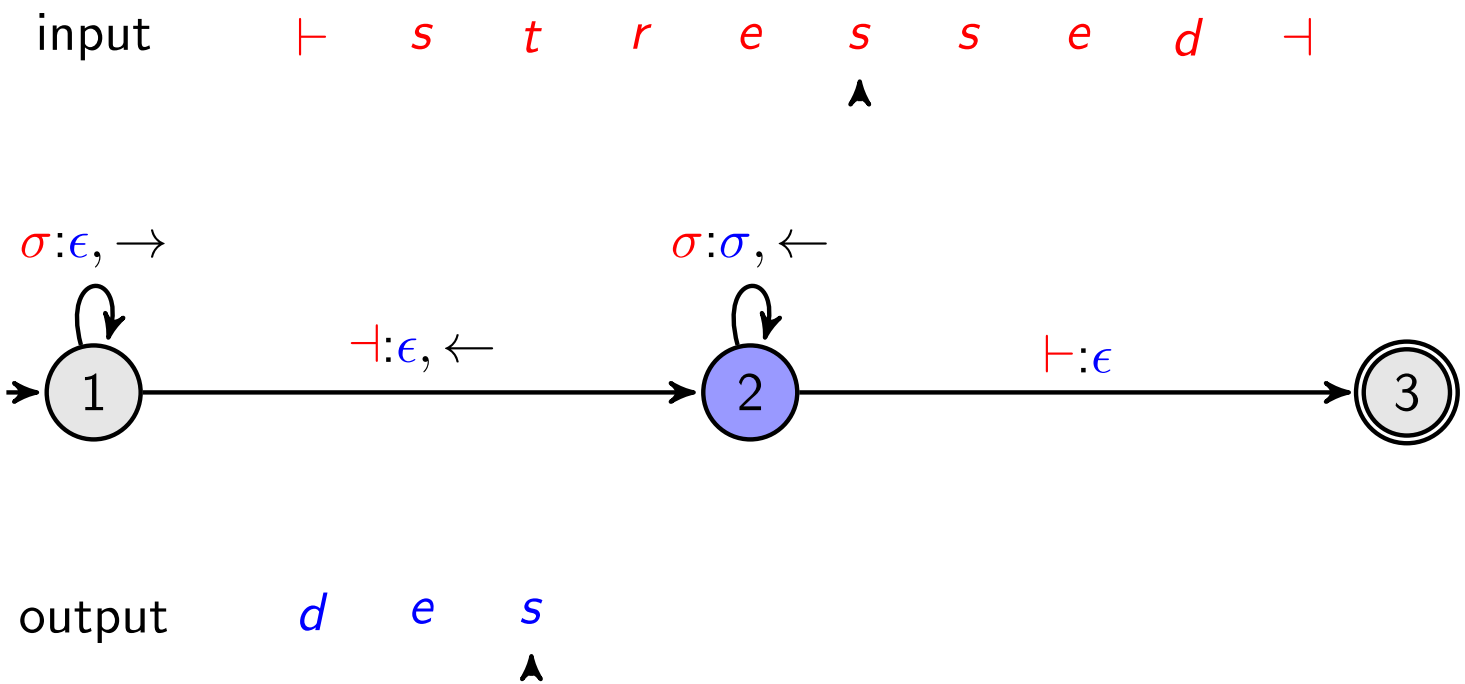
Automata for transductions: two-way transducers

input $\vdash$ s t r e s s e d $\dashv$

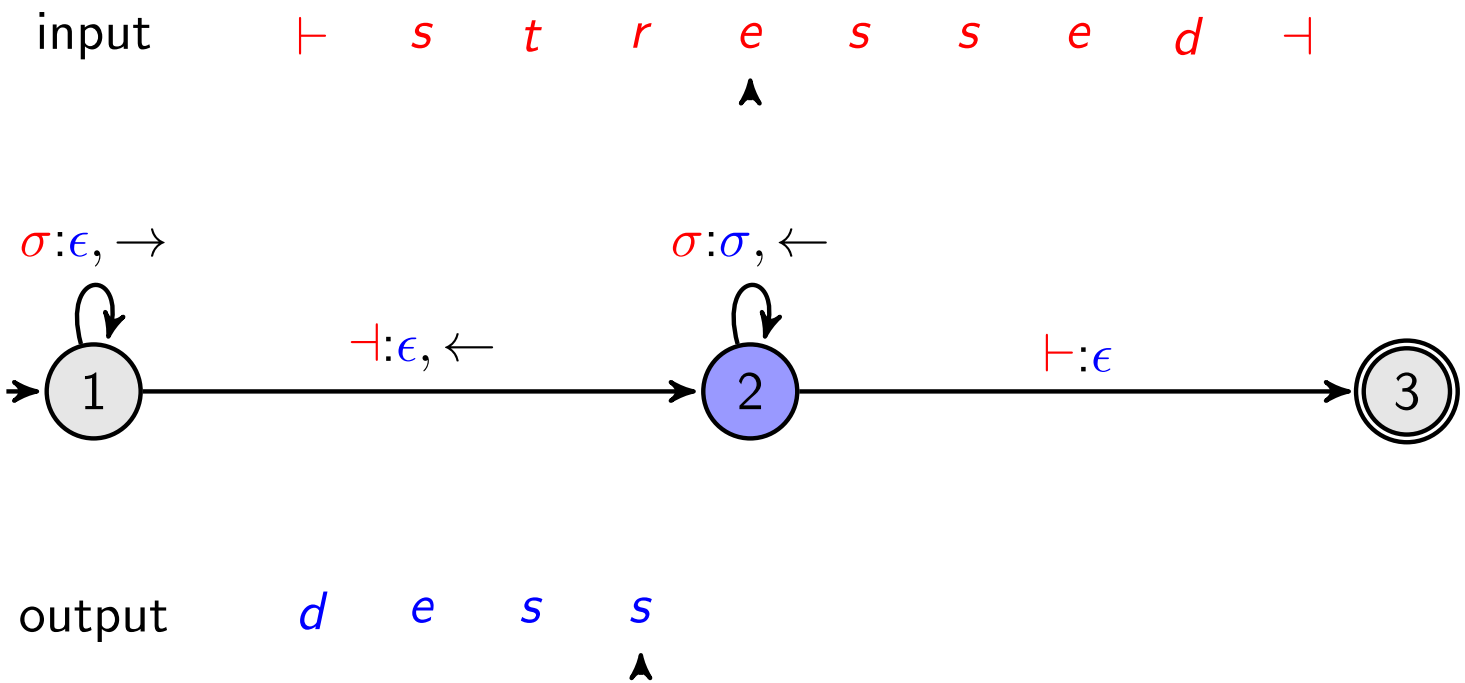


output d e

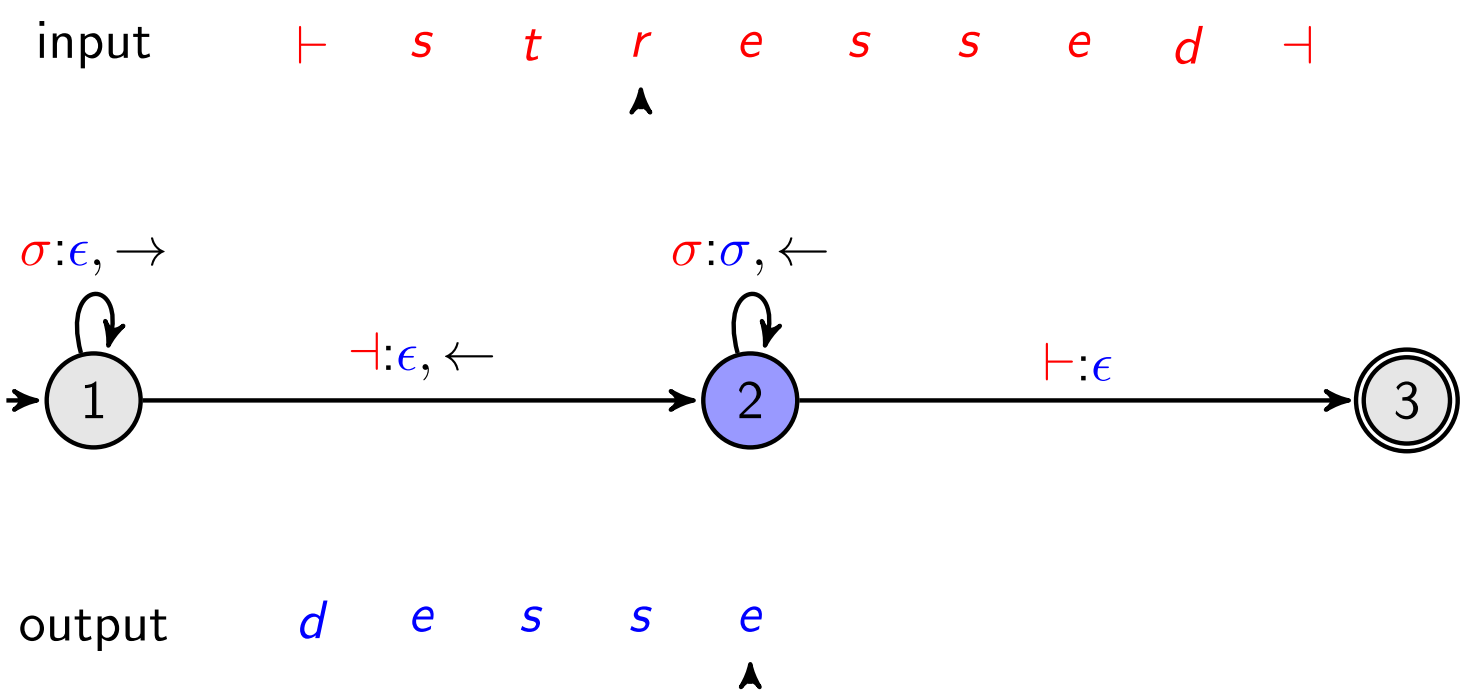
Automata for transductions: two-way transducers



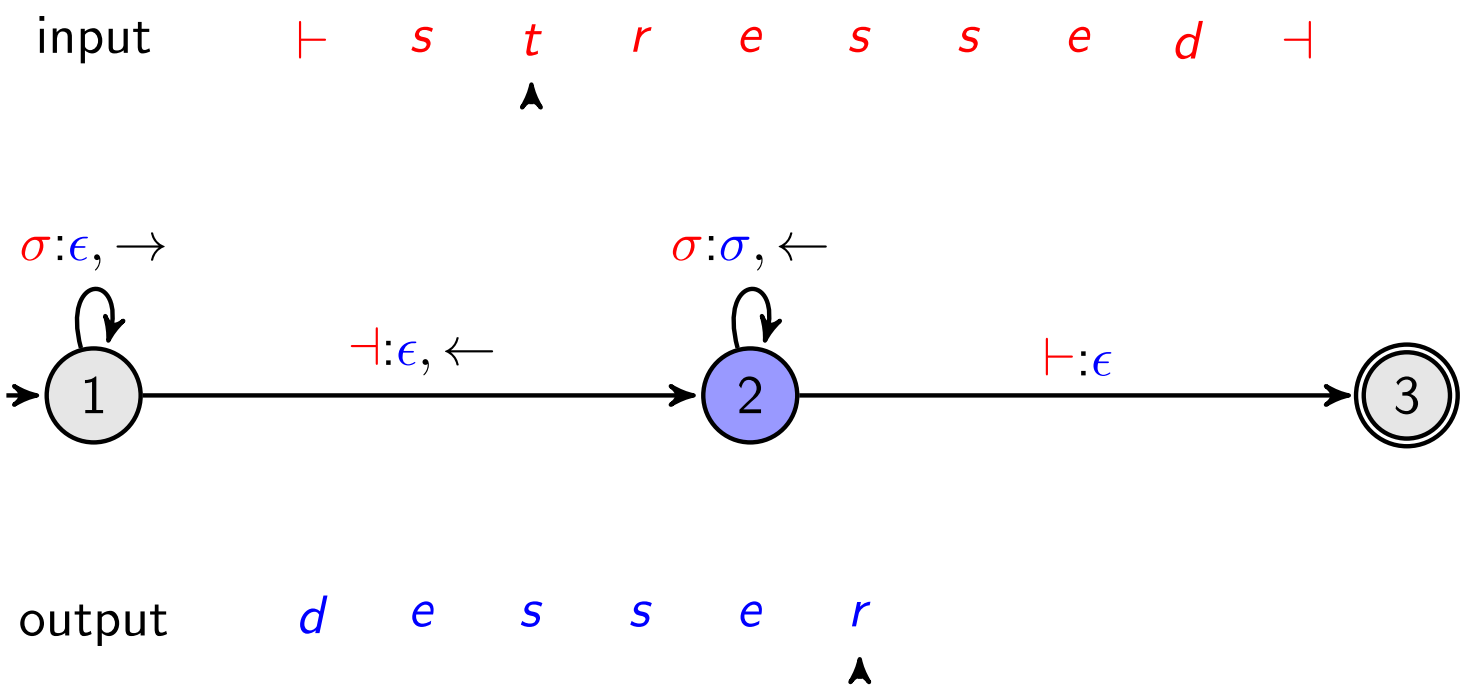
Automata for transductions: two-way transducers



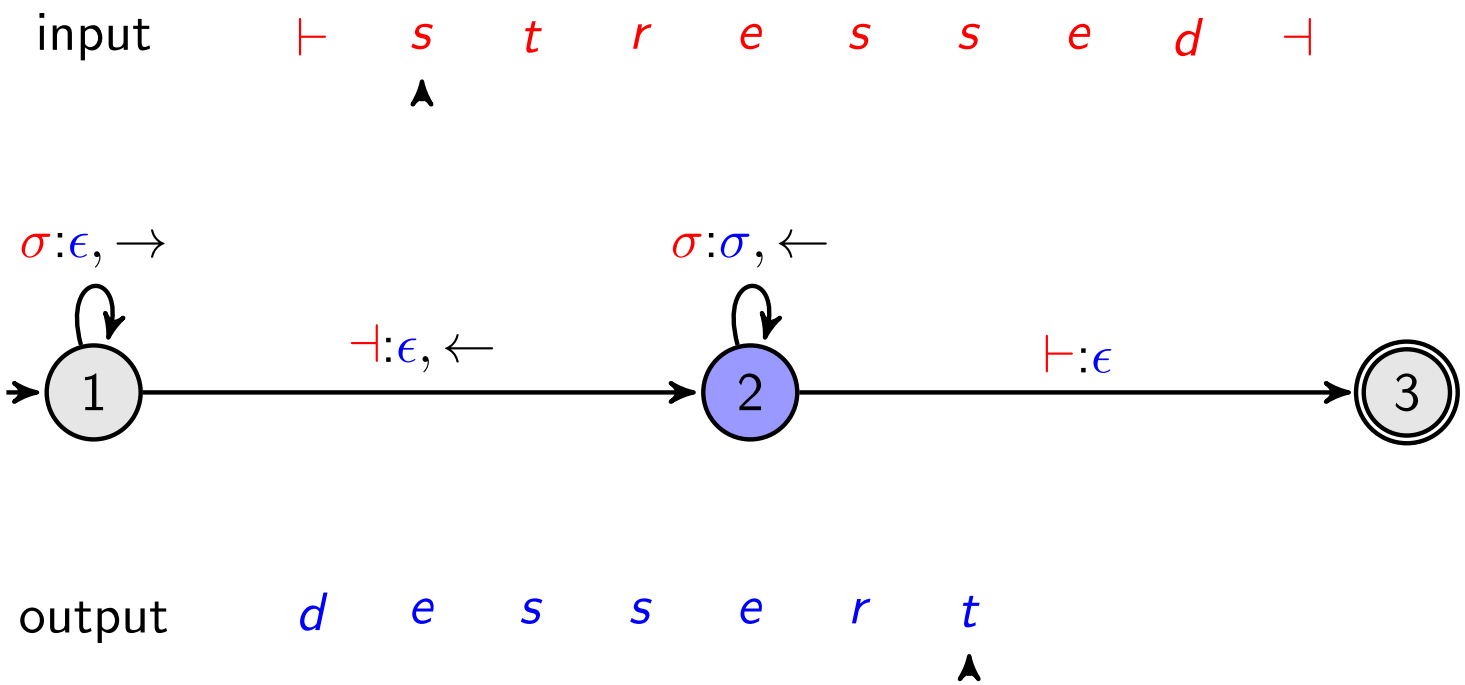
Automata for transductions: two-way transducers



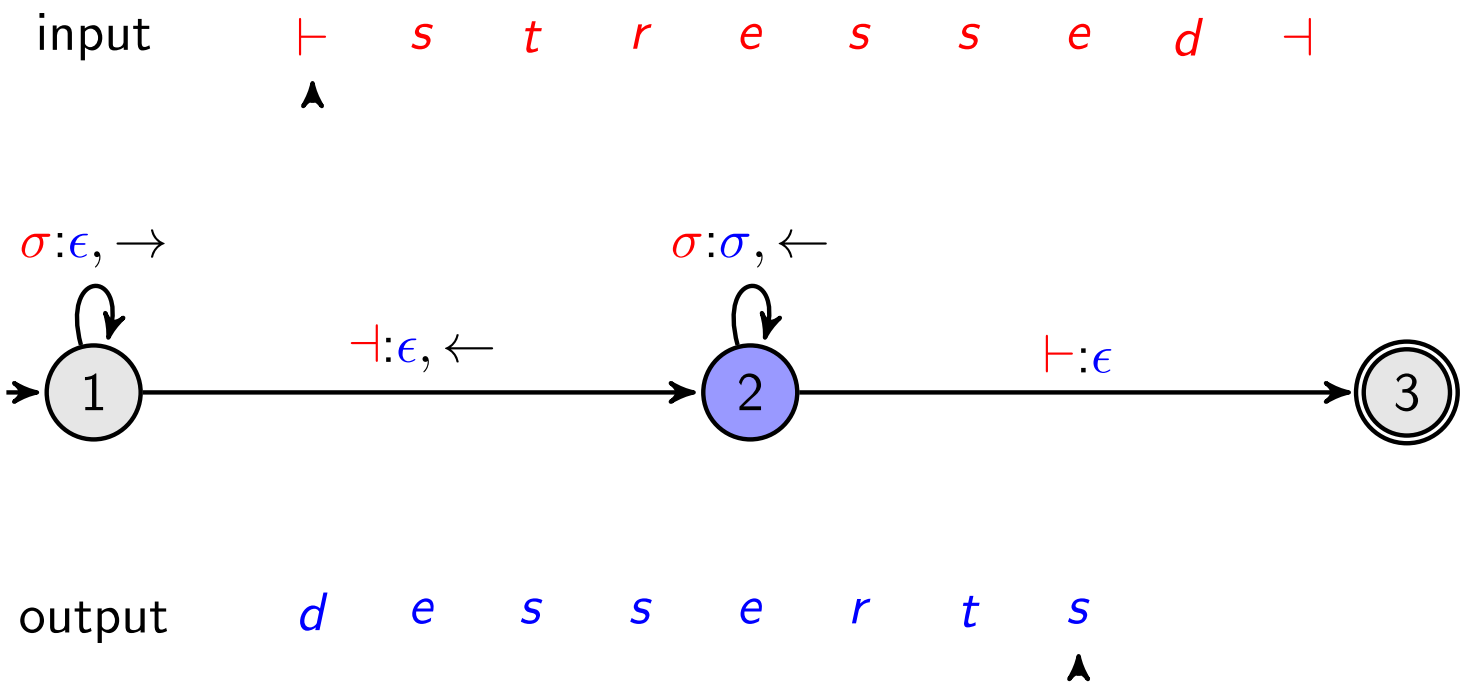
Automata for transductions: two-way transducers



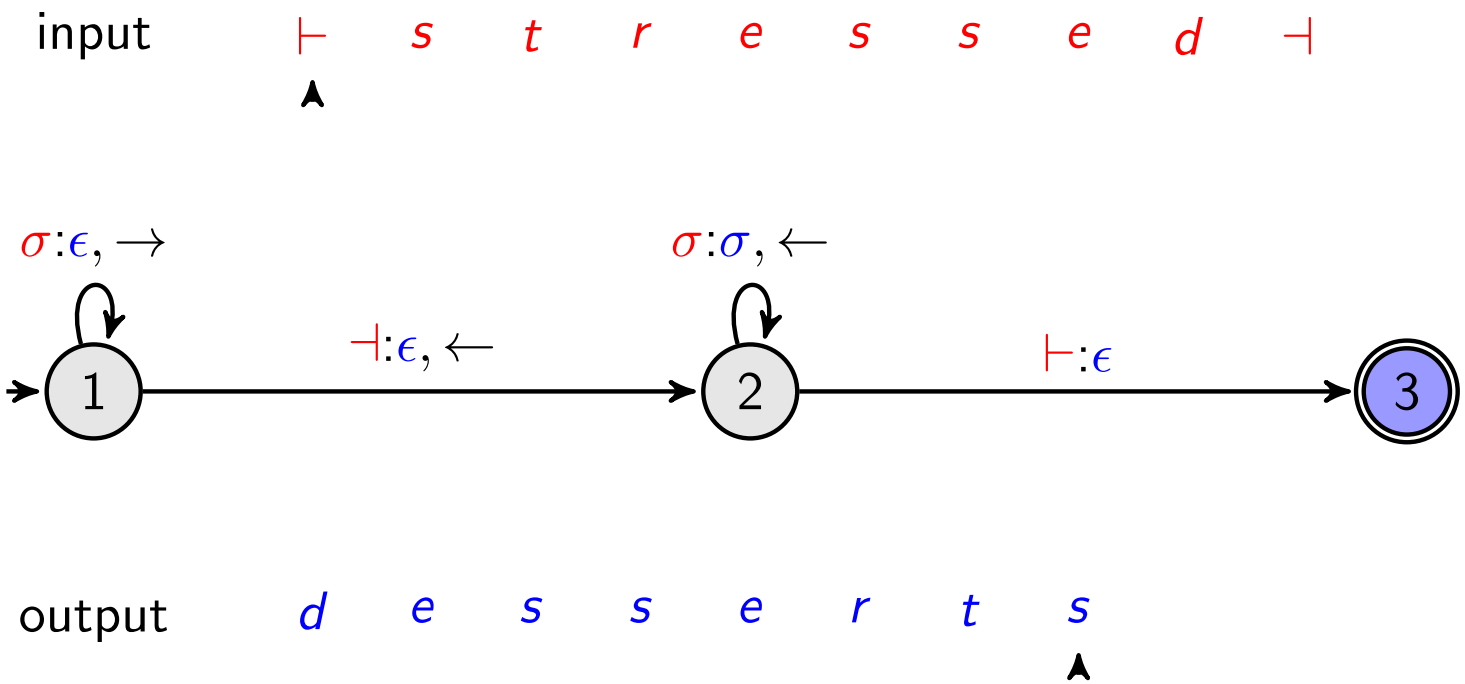
Automata for transductions: two-way transducers



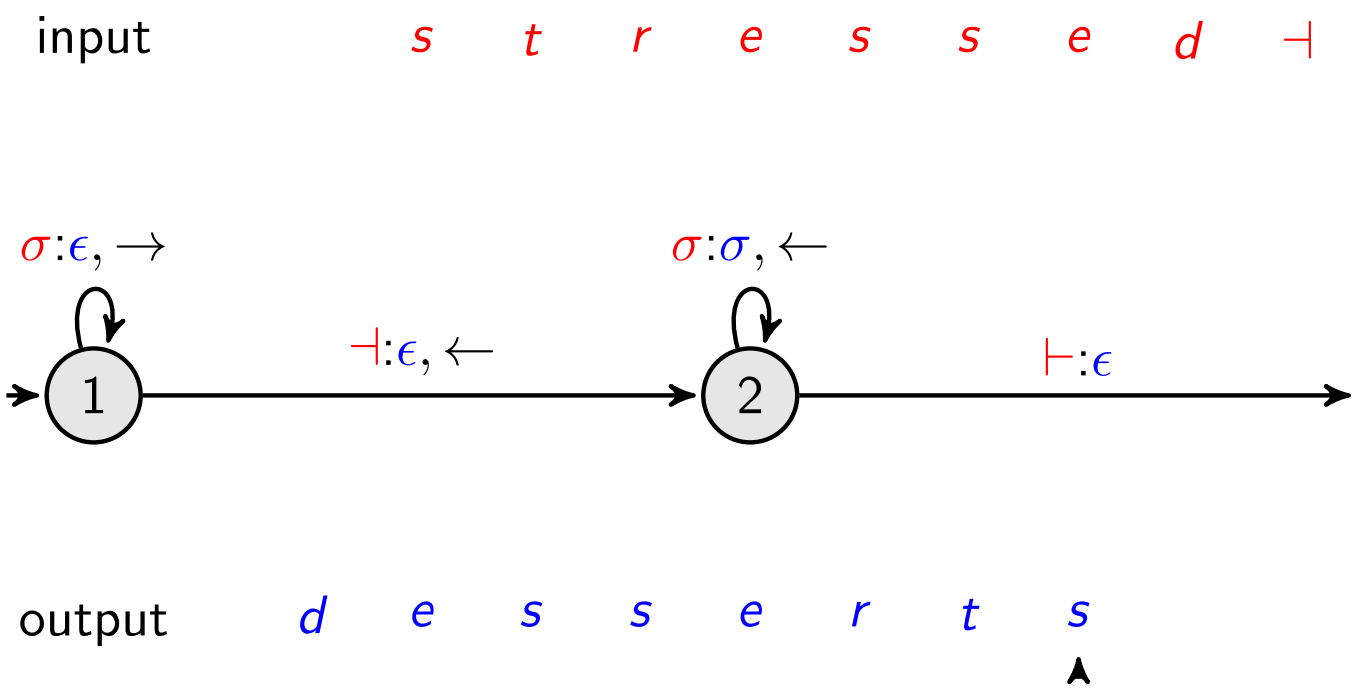
Automata for transductions: two-way transducers



Automata for transductions: two-way transducers



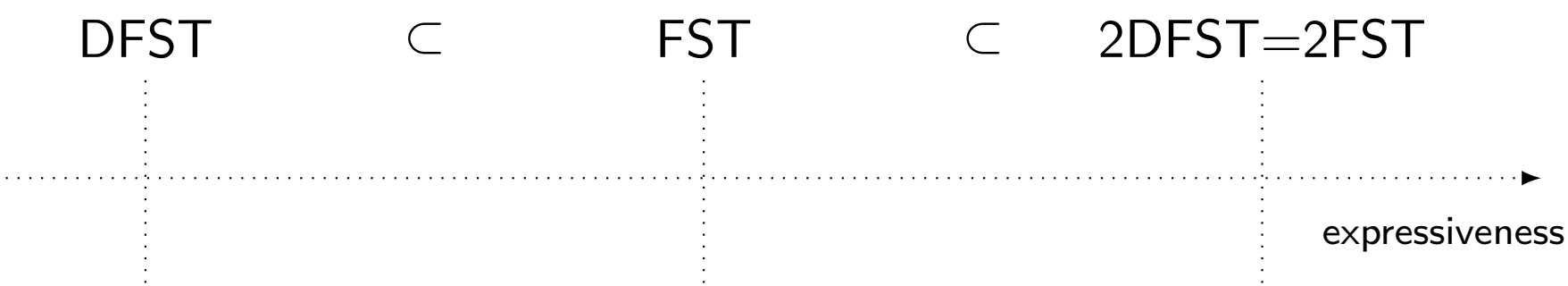
Automata for transductions: two-way transducers



one-way $<$ two-way

- ☺ decidable equivalence problem (Culik, Karhumaki, 87).
- ☺ closed under composition $\circ$ (Chytil, Jakl, 77)

Landscape of Transducer Classes

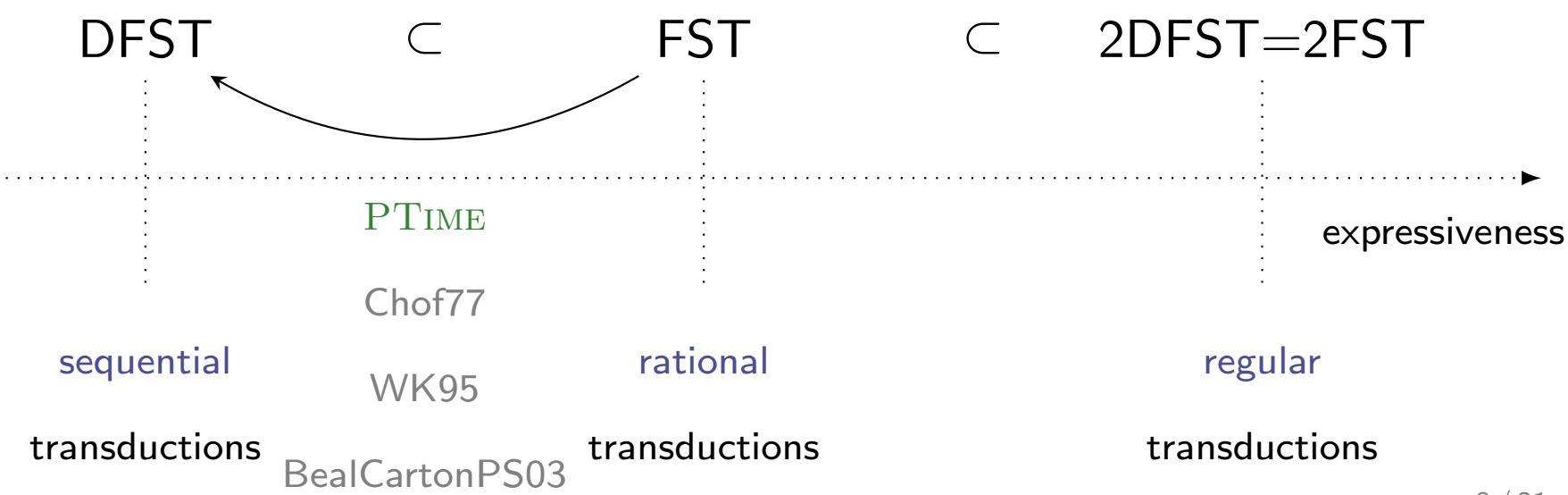


sequential transductions

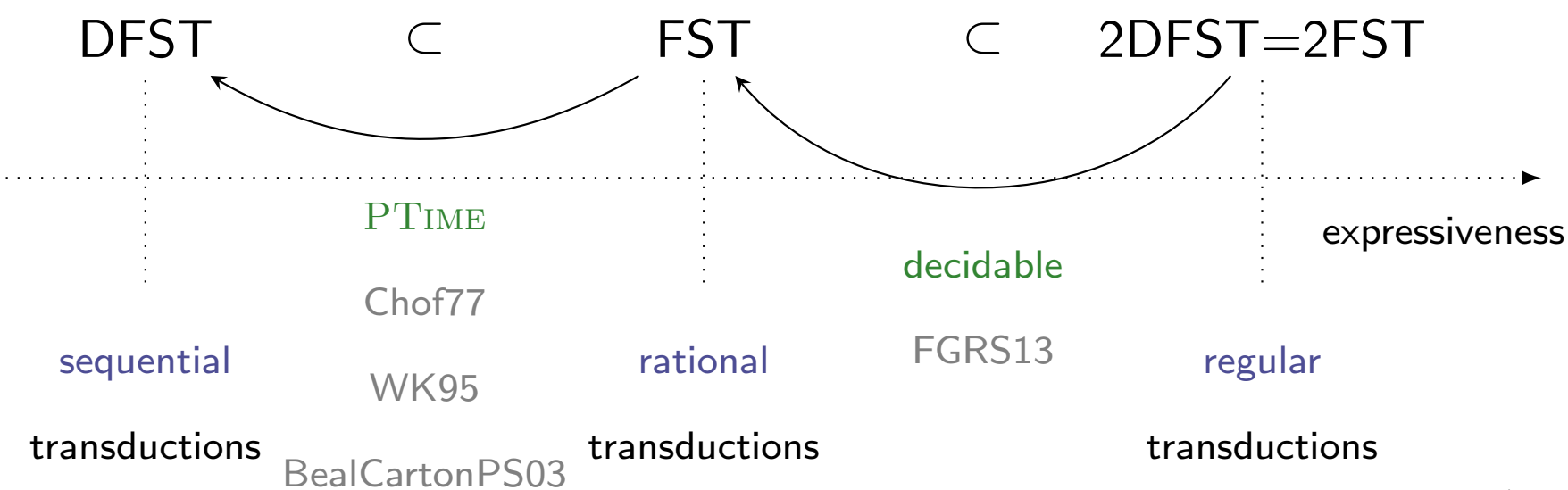
rational transductions

regular transductions

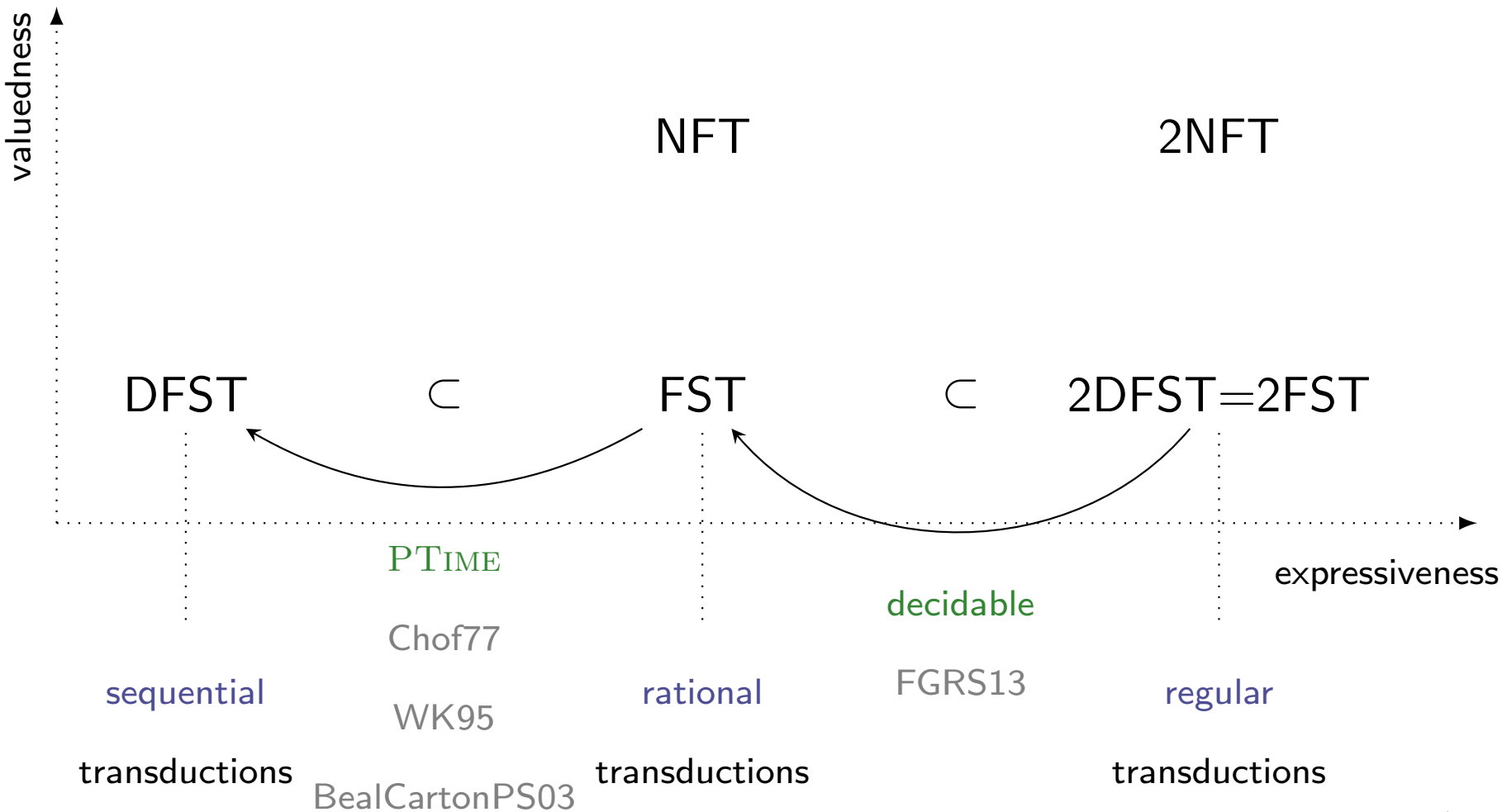
Landscape of Transducer Classes



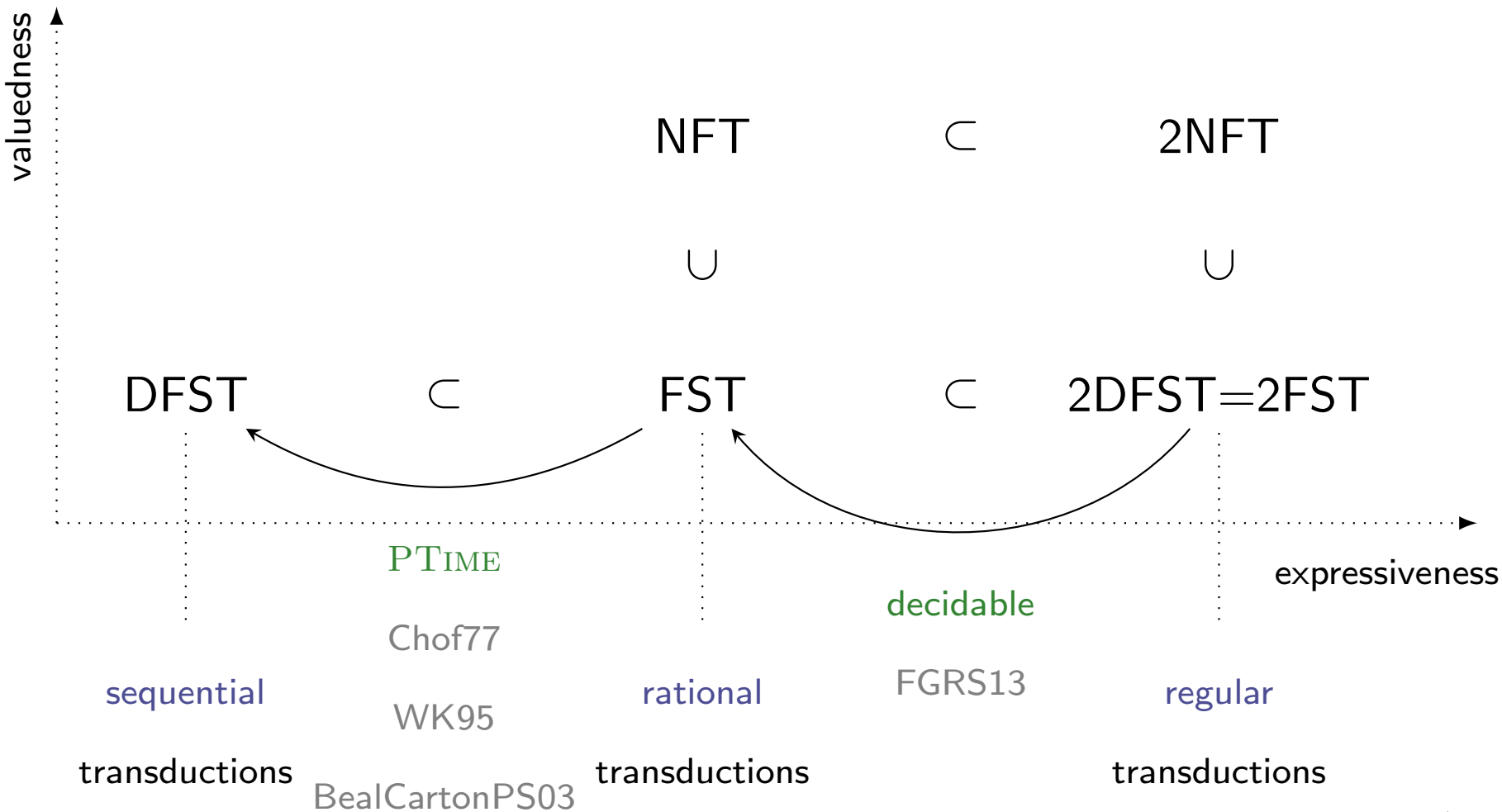
Landscape of Transducer Classes



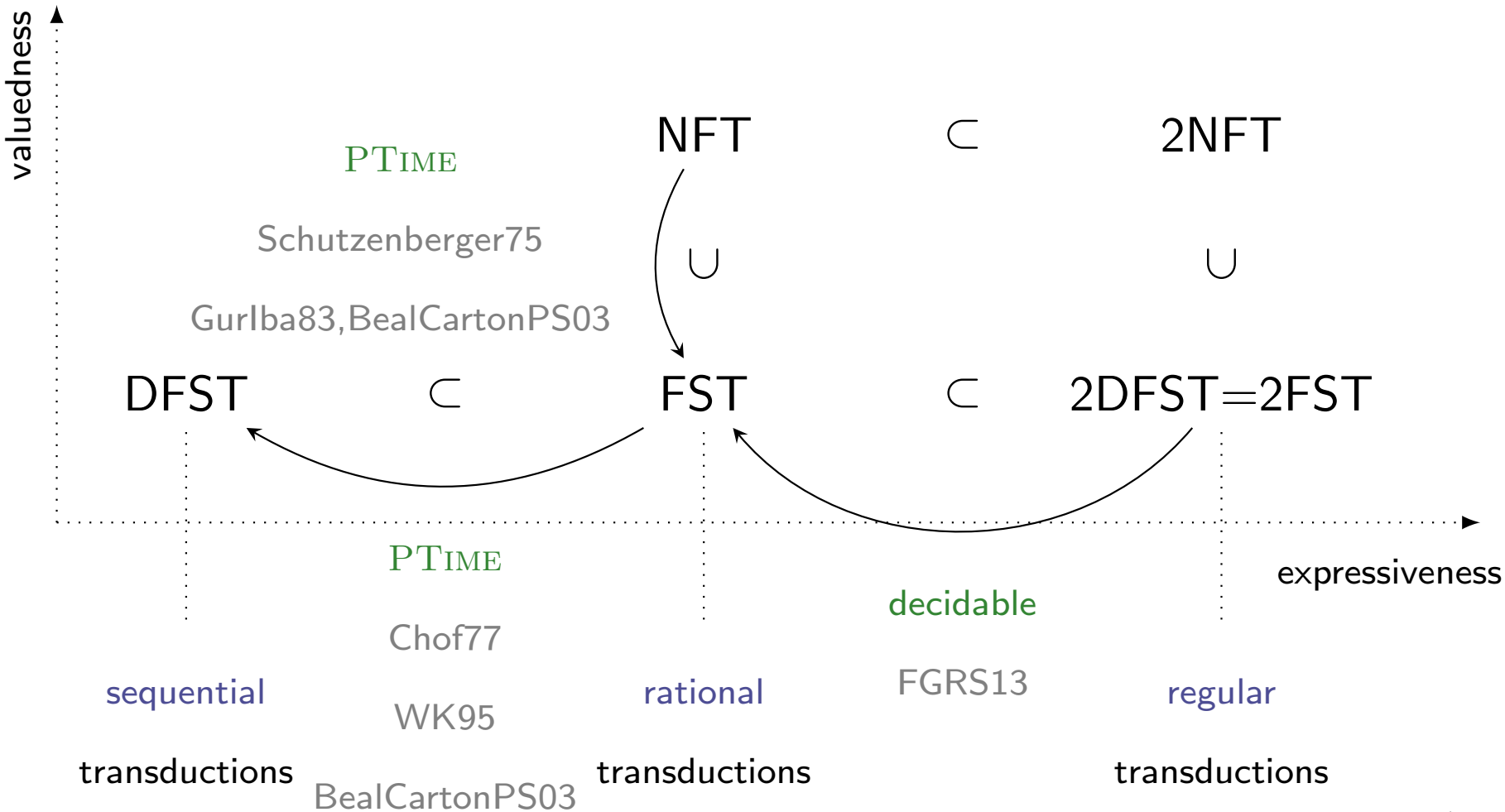
Landscape of Transducer Classes



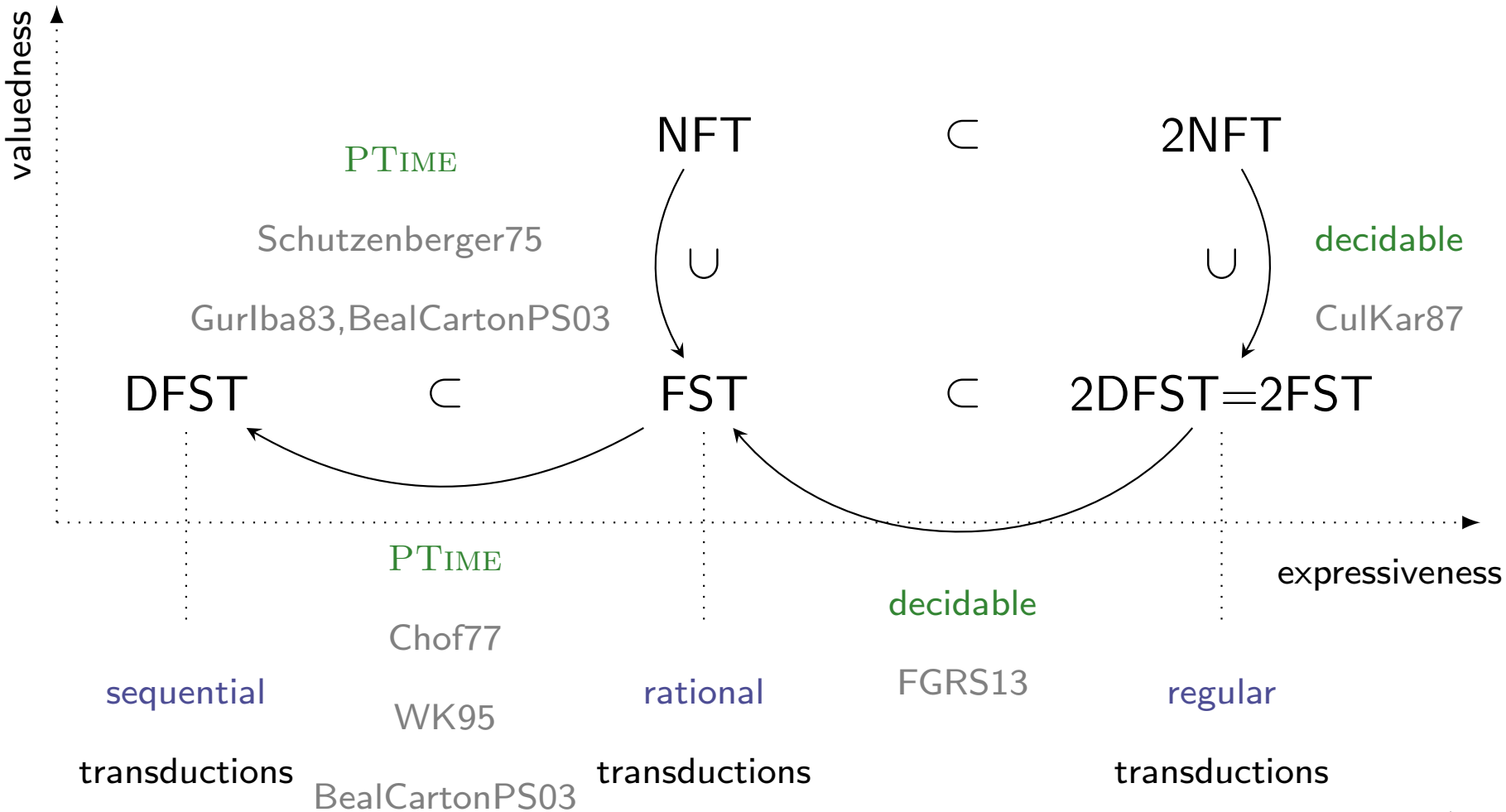
Landscape of Transducer Classes



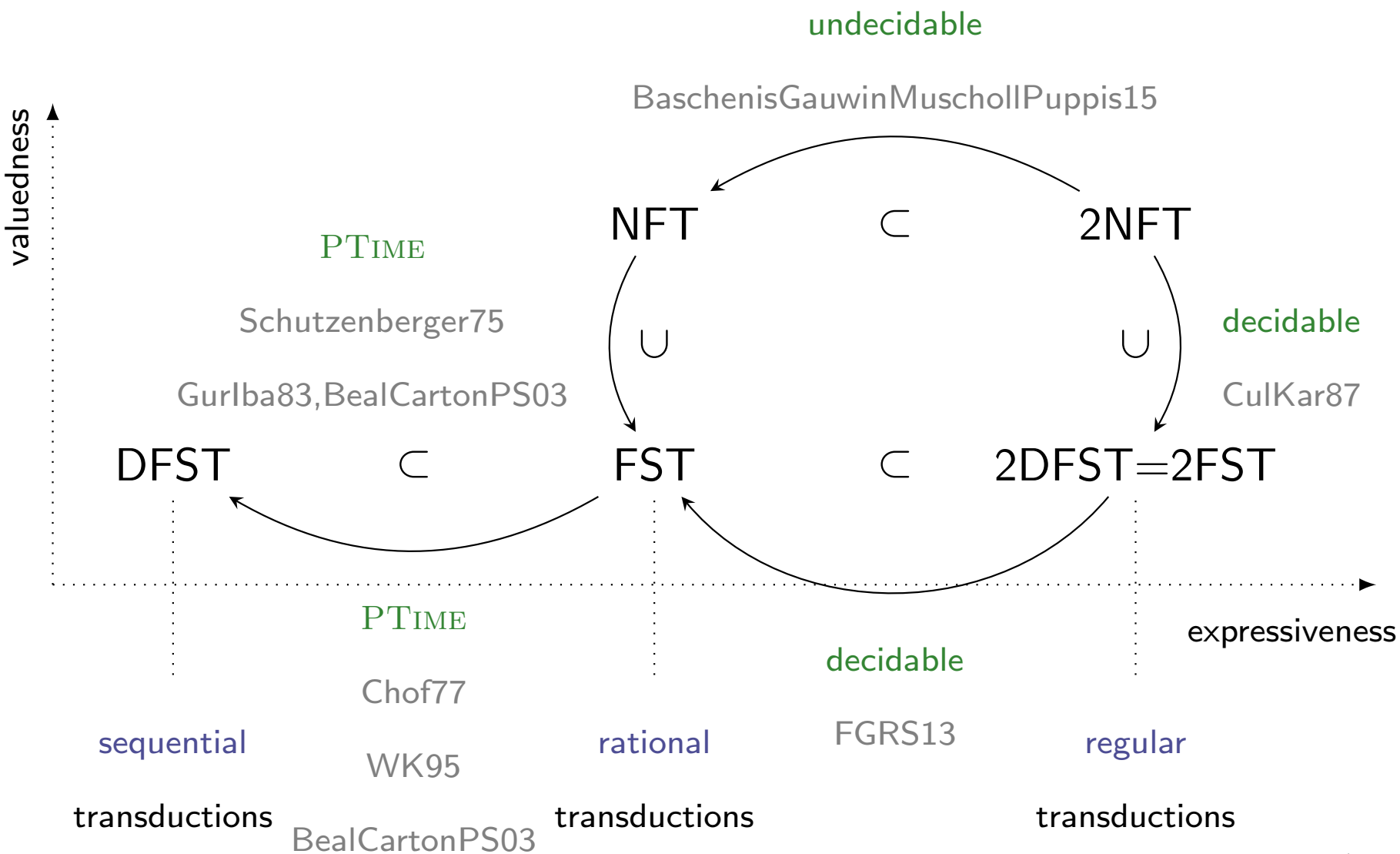
Landscape of Transducer Classes



Landscape of Transducer Classes



Landscape of Transducer Classes



(Courcelle) MSO Transformations

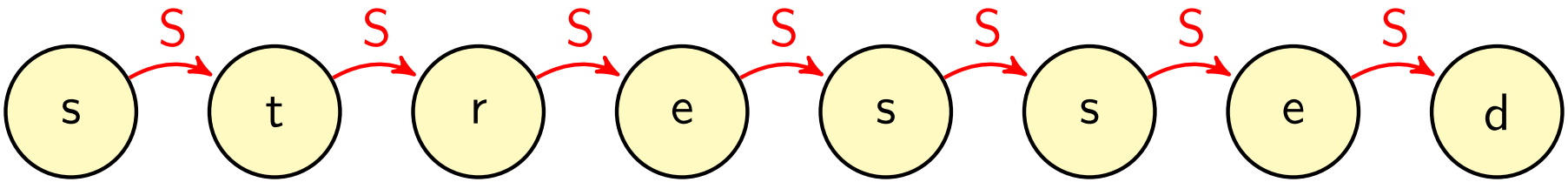
"interpreting the output structure in the input structure"

- output predicates defined by MSO[S] formulas interpreted over the input structure

(Courcelle) MSO Transformations

"interpreting the output structure in the input structure"

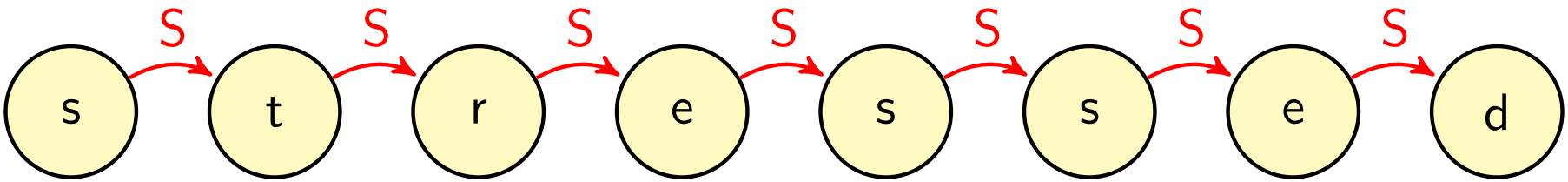
- output predicates defined by MSO[S] formulas interpreted over the input structure



(Courcelle) MSO Transformations

"interpreting the output structure in the input structure"

- output predicates defined by MSO[S] formulas interpreted over the input structure



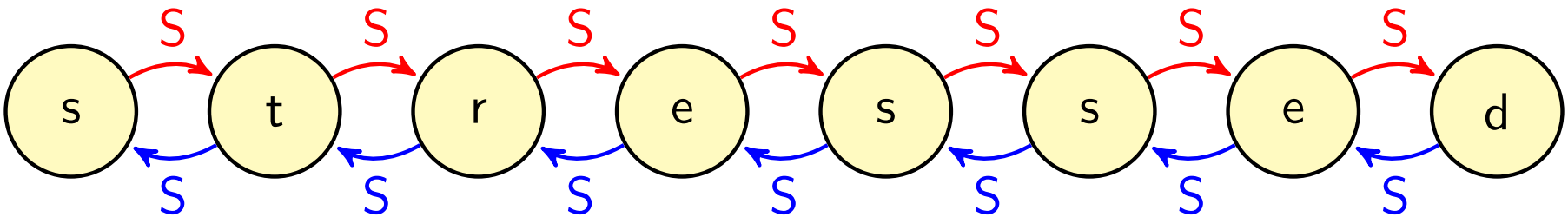
$$\phi_S(x,y) \equiv S(y,x)$$

$$\phi_\sigma(x) \equiv \sigma(x)$$

(Courcelle) MSO Transformations

"interpreting the output structure in the input structure"

- output predicates defined by MSO[S] formulas interpreted over the input structure



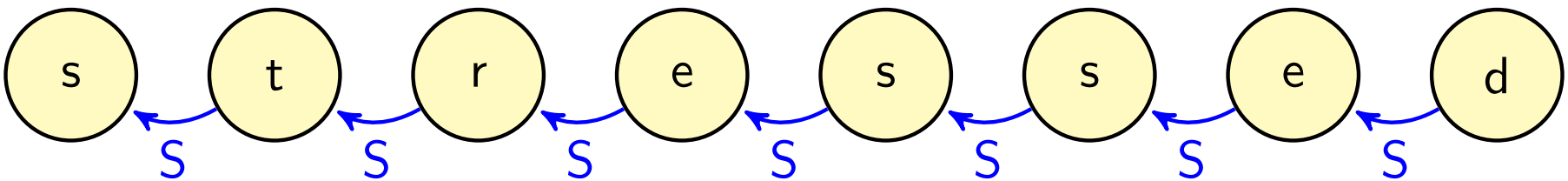
$$\phi_S(x,y) \equiv S(y,x)$$

$$\phi_\sigma(x) \equiv \sigma(x)$$

(Courcelle) MSO Transformations

"interpreting the output structure in the input structure"

- output predicates defined by MSO[S] formulas interpreted over the input structure



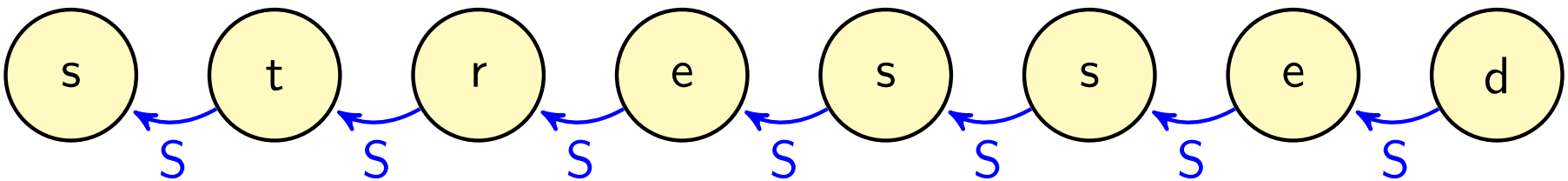
$$\phi_S(x,y) \equiv S(y,x)$$

$$\phi_\sigma(x) \equiv \sigma(x)$$

(Courcelle) MSO Transformations

"interpreting the output structure in the input structure"

- output predicates defined by MSO[S] formulas interpreted over the input structure



$$\phi_S(x,y) \equiv S(y,x)$$

$$\phi_\sigma(x) \equiv \sigma(x)$$

(Courcelle) MSO Transformations

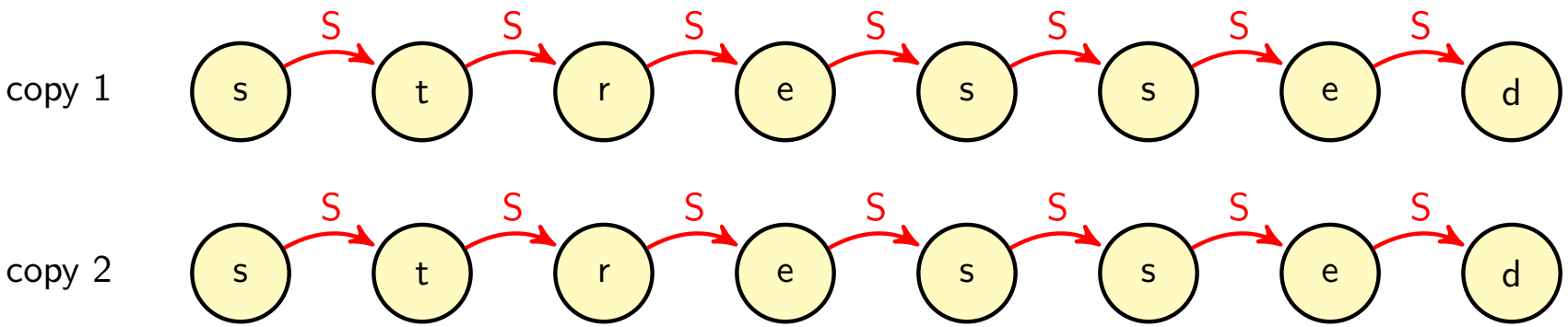
- input structure can be copied a fixed number of times:

$$w \mapsto w.\mathrm{mirror}(w)$$

(Courcelle) MSO Transformations

- input structure can be copied a fixed number of times:

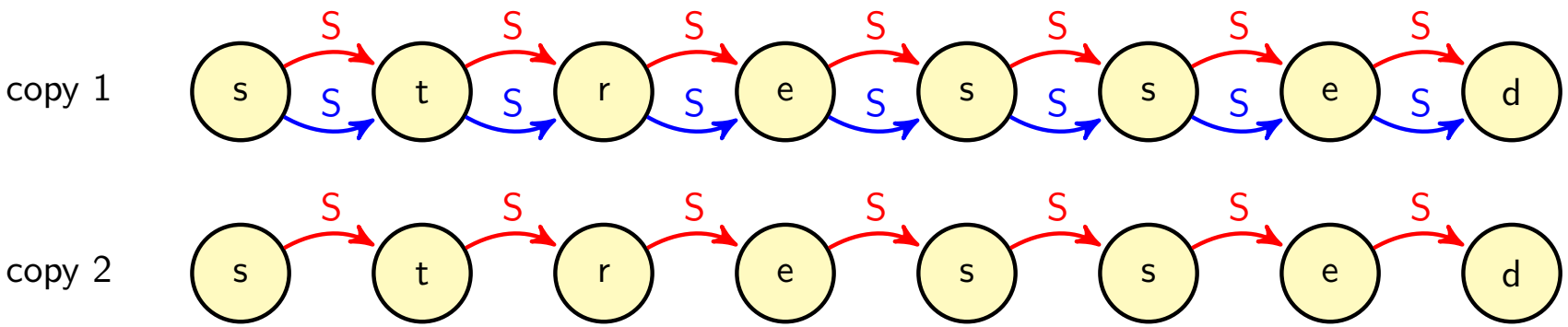
$$w \mapsto w.\mathrm{mirror}(w)$$



(Courcelle) MSO Transformations

- input structure can be copied a fixed number of times:

$$w \mapsto w.\mathrm{mirror}(w)$$



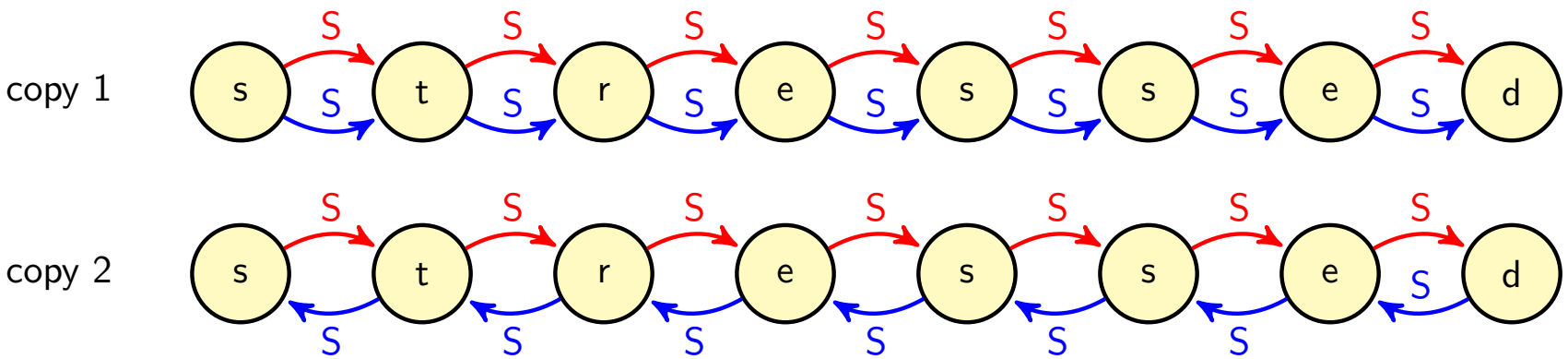
Formulas

copy 1: $\phi^1_S(x,y) \equiv S(x,y)$

(Courcelle) MSO Transformations

- input structure can be copied a fixed number of times:

$$w \mapsto w.\mathrm{mirror}(w)$$



Formulas

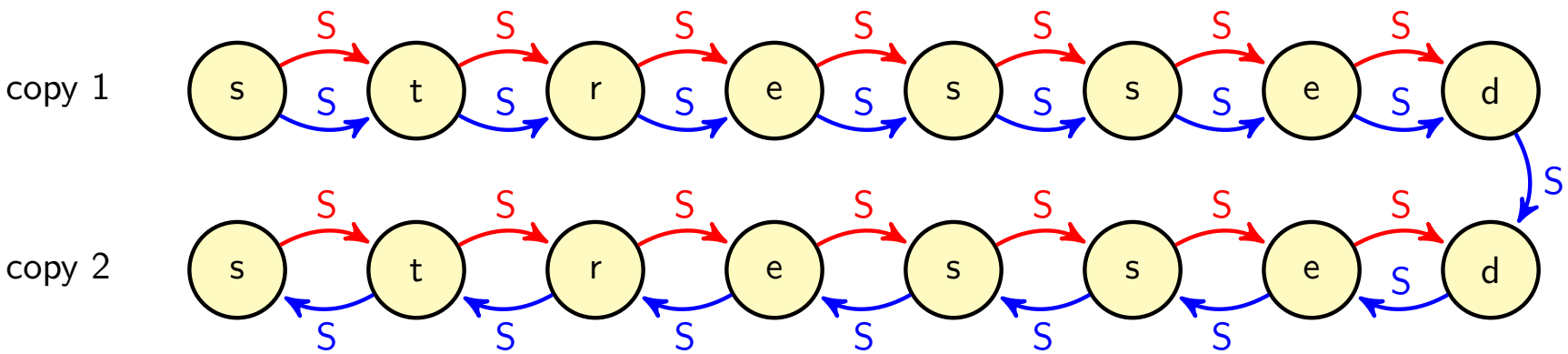
copy 1: $\phi^1_S(x,y) \equiv S(x,y)$

copy 2: $\phi^2_S(x,y) \equiv S(y,x)$

(Courcelle) MSO Transformations

- input structure can be copied a fixed number of times:

$$w \mapsto w.\mathrm{mirror}(w)$$



Formulas

$$\textbf{copy 1:} \quad \phi_S^1(x,y) \equiv S(x,y)$$

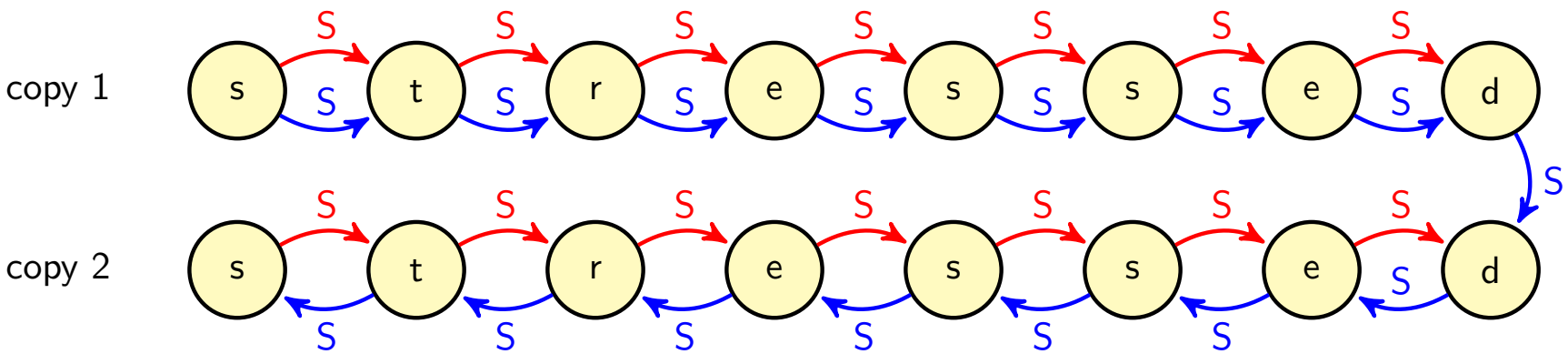
$$\textbf{copy 2:} \quad \phi_S^2(x,y) \equiv S(y,x)$$

$$\textbf{copy 1 to copy 2:} \quad \phi_S^{1\to 2}(x,y) \equiv x = y \wedge \mathit{last}(x)$$

(Courcelle) MSO Transformations

- input structure can be copied a fixed number of times:

$$w \mapsto w.\mathrm{mirror}(w)$$



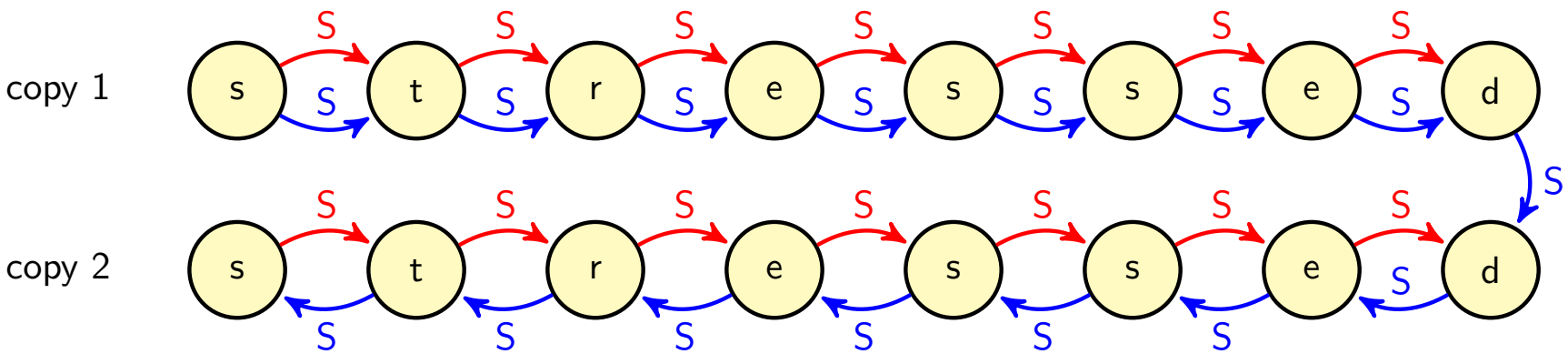
Formulas

copy 1:	$\phi_S^1(x,y)$	$\equiv$	$S(x,y)$
copy 2:	$\phi_S^2(x,y)$	$\equiv$	$S(y,x)$
copy 1 to copy 2:	$\phi_S^{1\to 2}(x,y)$	$\equiv$	$x = y \wedge \mathit{last}(x)$
copy 2 to copy 1:	$\phi_S^{2\to 1}(x,y)$	$\equiv$	$\bot$

(Courcelle) MSO Transformations

- input structure can be copied a fixed number of times:

$$w \mapsto w.\mathrm{mirror}(w)$$



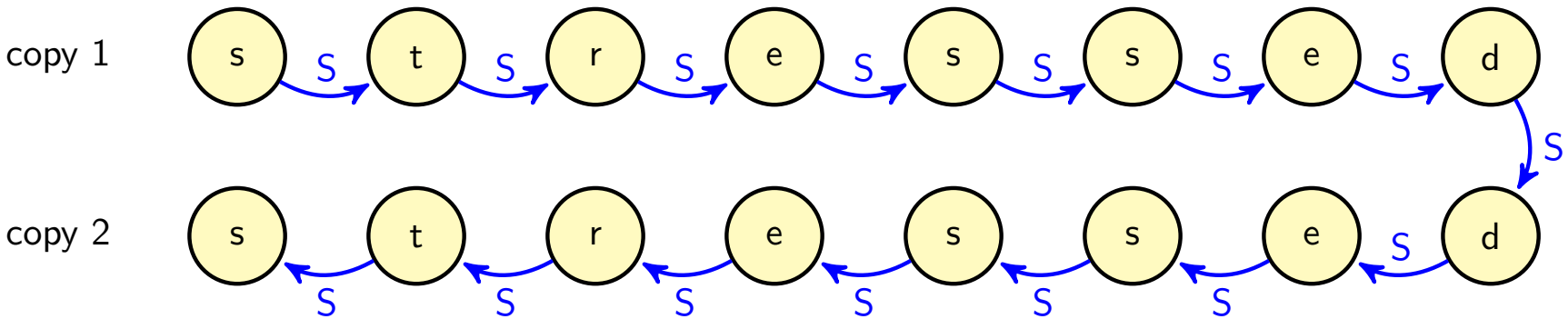
Formulas

copy 1:	$\phi_S^1(x,y)$	$\equiv$	$S(x,y)$
copy 2:	$\phi_S^2(x,y)$	$\equiv$	$S(y,x)$
copy 1 to copy 2:	$\phi_S^{1\to 2}(x,y)$	$\equiv$	$x = y \wedge \mathit{last}(x)$
copy 2 to copy 1:	$\phi_S^{2\to 1}(x,y)$	$\equiv$	$\bot$
for all copies i:	$\phi_\sigma^{\mathsf{i}}(x)$	$\equiv$	$\sigma(x)$

(Courcelle) MSO Transformations

- input structure can be copied a fixed number of times:

$$w \mapsto w.\mathrm{mirror}(w)$$



Formulas

copy 1:	$\phi_S^1(x,y)$	$\equiv$	$S(x,y)$
copy 2:	$\phi_S^2(x,y)$	$\equiv$	$S(y,x)$
copy 1 to copy 2:	$\phi_S^{1\to 2}(x,y)$	$\equiv$	$x = y \wedge \mathit{last}(x)$
copy 2 to copy 1:	$\phi_S^{2\to 1}(x,y)$	$\equiv$	$\bot$
for all copies i:	$\phi_\sigma^{\mathbf{i}}(x)$	$\equiv$	$\sigma(x)$

Büchi Theorems for Word Transductions

Theorem (Engelfriet, Hoogeboom, 01)

Let $f : \Sigma^* \to \Sigma^*$. The following are equivalent:

1. *f is definable by a deterministic two-way transducer*
2. *f is MSO-definable.*

Büchi Theorems for Word Transductions

Theorem (Engelfriet, Hoogeboom, 01)

Let $f : \Sigma^* \to \Sigma^*$. The following are equivalent:

1. *f is definable by a deterministic two-way transducer*
2. *f is MSO-definable.*

Consequence Equivalence is decidable for MSO-transducers.

Büchi Theorems for Word Transductions

Theorem (Engelfriet, Hoogeboom, 01)

Let $f : \Sigma^* \to \Sigma^*$. The following are equivalent:

1. *f is definable by a deterministic two-way transducer*
2. *f is MSO-definable.*

Consequence Equivalence is decidable for MSO-transducers.

Theorem (Bojanczyk 14, F. 15)

Let $f : \Sigma^* \to \Sigma^*$. The following are equivalent:

1. *f is definable by a (one-way) transducer*
2. *f is definable in order-preserving MSO*

Order-preserving MSO: $\phi_S^{i,j}(x, y) \models x \preceq y$.

Myhill-Nerode congruence for $L \subseteq \Sigma^*$

- $u \sim_L v$ if: for all $w \in \Sigma^*$, $uw \in L$ iff $vw \in L$
- u and v have the same "effect" on continuations w
- **Myhill-Nerode's Thm**: L is regular iff $\Sigma^*/_{\sim_L}$ is finite
- canonical (and minimal) deterministic automaton for L, $\Sigma^*/_{\sim_L}$ as set of states

Myhill-Nerode congruence for $L \subseteq \Sigma^*$

- $u \sim_L v$ if: for all $w \in \Sigma^*$, $uw \in L$ iff $vw \in L$
- u and v have the same "effect" on continuations w
- **Myhill-Nerode's Thm**: L is regular iff $\Sigma^*/_{\sim_L}$ is finite
- canonical (and minimal) deterministic automaton for L, $\Sigma^*/_{\sim_L}$ as set of states

Goal

Extend Myhill-Nerode's theorem to classes of transductions

Sequential transductions (Choffrut)

Refinement of the MN congruence.

Two ideas

1. produce asap: $F(u) = LCP\{f(uw) \mid uw \in dom(f)\}$

Sequential transductions (Choffrut)

Refinement of the MN congruence.

Two ideas

1. produce asap: $F(u) = LCP\{f(uw) \mid uw \in dom(f)\}$
2. $u \sim_f v$ if
   1. $u \sim_{dom(f)} v$
   2. $F(u)^{-1} f(uw) = F(v)^{-1} f(vw)$ $\quad \forall w \in u^{-1} dom(f)$

"u and v have the same effect on continuations w w.r.t. domain membership and produced outputs"

Sequential transductions (Choffrut)

Refinement of the MN congruence.

Two ideas

1. produce asap: $F(u) = LCP\{f(uw) \mid uw \in dom(f)\}$
2. $u \sim_f v$ if
   1. $u \sim_{dom(f)} v$
   2. $F(u)^{-1} f(uw) = F(v)^{-1} f(vw)$ $\quad \forall w \in u^{-1} dom(f)$

"u and v have the same effect on continuations w w.r.t. domain membership and produced outputs"

Theorem (Choffrut)

f is sequential iff $\sim_f$ has finite index

$\sim_f$ is a right congruence $\rightsquigarrow$ canonical and minimal transducer !

$$\text{Transitions: } [u] \xrightarrow{\sigma | F(u)^{-1} F(u\sigma)} [u\sigma]$$

Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$a\ b\ b\ a\ a\ a\ a\ b\ b\ b\ b\ a\ b$

Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$a\ b\ b\ a\ a\ a\ a\ b\ b\ b\ b\ a\ \overset{\epsilon}{b}$

Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$a\ b\ b\ a\ a\ a\ a\ b\ b\ b\ b\ \overset{b}{a}\ \overset{\epsilon}{b}$

Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$$a\ b\ b\ a\ a\ a\ a\ b\ b\ b\ \overset{b}{b}\ \overset{b}{a}\ \overset{\epsilon}{b}$$

Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$$\begin{array}{cccccccccccc} & & & & & & & b & b & b & \epsilon & \\ a & b & b & a & a & a & a & b & b & b & a & b \end{array}$$

Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$$\begin{array}{cccccccccccccc} & & & & & & & b & b & b & b & \epsilon & \\ a & b & b & a & a & a & a & b & b & b & b & a & b \end{array}$$

Rational transductions are almost sequential

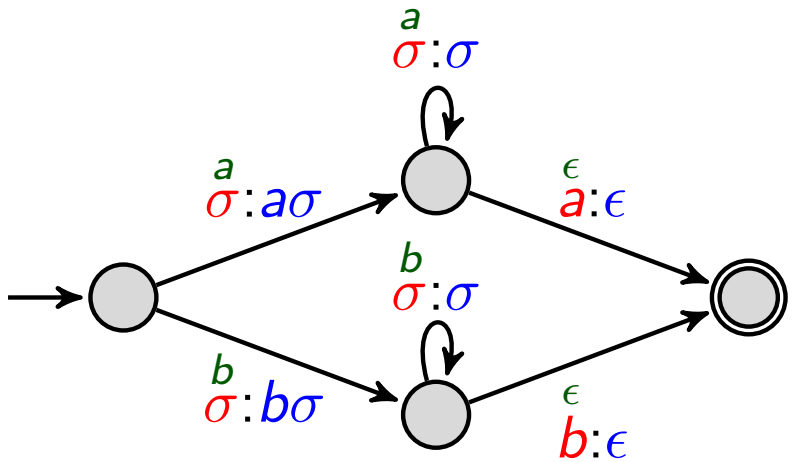
- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$$\begin{array}{cccccccccccc} b & b & b & b & b & b & b & b & b & b & b & \epsilon \\ a & b & b & a & a & a & a & b & b & b & b & a & b \end{array}$$

Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

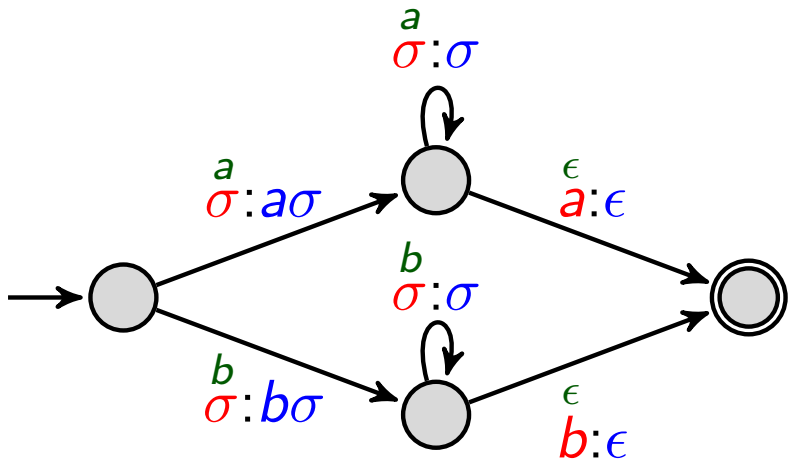
$$\begin{array}{cccccccccccc} b & b & b & b & b & b & b & b & b & b & b & \epsilon \\ a & b & b & a & a & a & a & b & b & b & b & a & b \end{array}$$



Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$$\begin{matrix} b & b & b & b & b & b & b & b & b & b & b & \epsilon \\ a & b & b & a & a & a & a & b & b & b & b & a & b \end{matrix}$$

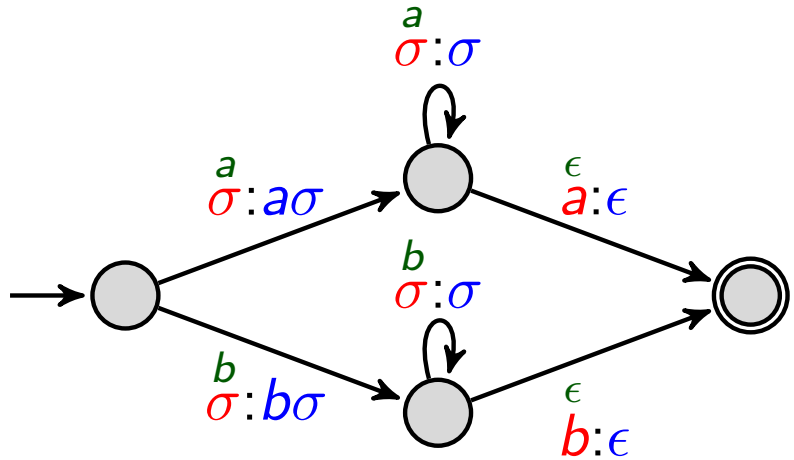


- look-ahead information: $\mathcal{L} : \Sigma^* \to \mathcal{I}$
- $f[\mathcal{L}]$: f with input words extended with look-ahead information

Rational transductions are almost sequential

- $f_{sw} : u\sigma \mapsto \sigma u$ is not sequential
- but sequential modulo *look-ahead information* $\mathcal{I} = \{a, b, \epsilon\}$.

$$\begin{matrix} b & b & b & b & b & b & b & b & b & b & b & \epsilon \\ a & b & b & a & a & a & a & b & b & b & b & a & b \end{matrix}$$



- look-ahead information: $\mathcal{L} : \Sigma^* \to \mathcal{I}$
- $f[\mathcal{L}]$: f with input words extended with look-ahead information

Theorem (Elgot, Mezei, 65)

f is rational iff $f[\mathcal{L}]$ is sequential, for some finite look-ahead information $\mathcal{L}$ computable by a right sequential transducer.

Original statement: $RAT = SEQ \circ RightSEQ$.

A canonical look-ahead

Identify suffixes with a 'bounded' effect on prefixes.

Equivalent suffixes

- $u \equiv_f v$ if $\{d(f(wu), f(wv)) \mid wu, wv \in dom(f)\}$ is finite[a].

A canonical look-ahead

Identify suffixes with a 'bounded' effect on prefixes.

Equivalent suffixes

- $u \equiv_f v$ if $\{d(f(wu), f(wv)) \mid wu, wv \in dom(f)\}$ is finite[a].
- $\equiv_f$ is a left congruence

A canonical look-ahead

Identify suffixes with a 'bounded' effect on prefixes.

Equivalent suffixes

- $u \equiv_f v$ if $\{d(f(wu), f(wv)) \mid wu, wv \in dom(f)\}$ is finite[a].
- $\equiv_f$ is a left congruence
- look-ahead: return the class of the suffix.

[a] $d(\alpha, \beta) = |\alpha| + |\beta| - 2|lcp(\alpha, \beta)|$.

Ex: For f_{sw}, $\Sigma^*/_{\equiv} = \{\{ua \mid u \in \Sigma^*\}, \{ub \mid u \in \Sigma^*\}, \{\epsilon\}\}$.

A canonical look-ahead

Identify suffixes with a 'bounded' effect on prefixes.

Equivalent suffixes

- $u \equiv_f v$ if $\{d(f(wu), f(wv)) \mid wu, wv \in dom(f)\}$ is finite[a].
- $\equiv_f$ is a left congruence
- look-ahead: return the class of the suffix.

[a] $d(\alpha, \beta) = |\alpha| + |\beta| - 2|lcp(\alpha, \beta)|$.

Ex: For f_{sw}, $\Sigma^*/_{\equiv} = \{\{ua \mid u \in \Sigma^*\}, \{ub \mid u \in \Sigma^*\}, \{\epsilon\}\}$.

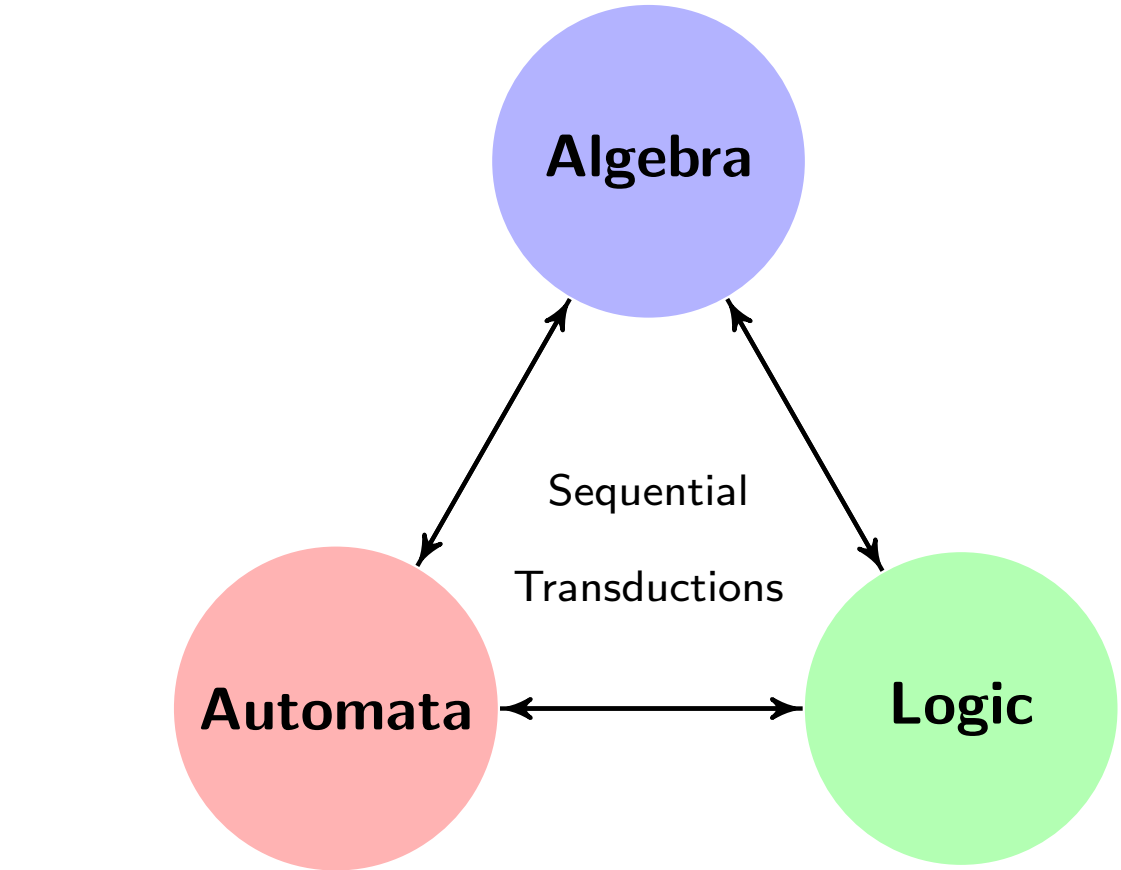
Theorem (Reutenauer, Schützenberger, 91)

The following are equivalent:

- *f is rational*
- *$\equiv_f$ has finite index and $f[\equiv_f]$ is sequential*
- *$\equiv_f$ and $\sim_{f[\equiv_f]}$ have finite index.*

Summary: sequential transductions

finiteness of $\sim_f$ (Choffrut)

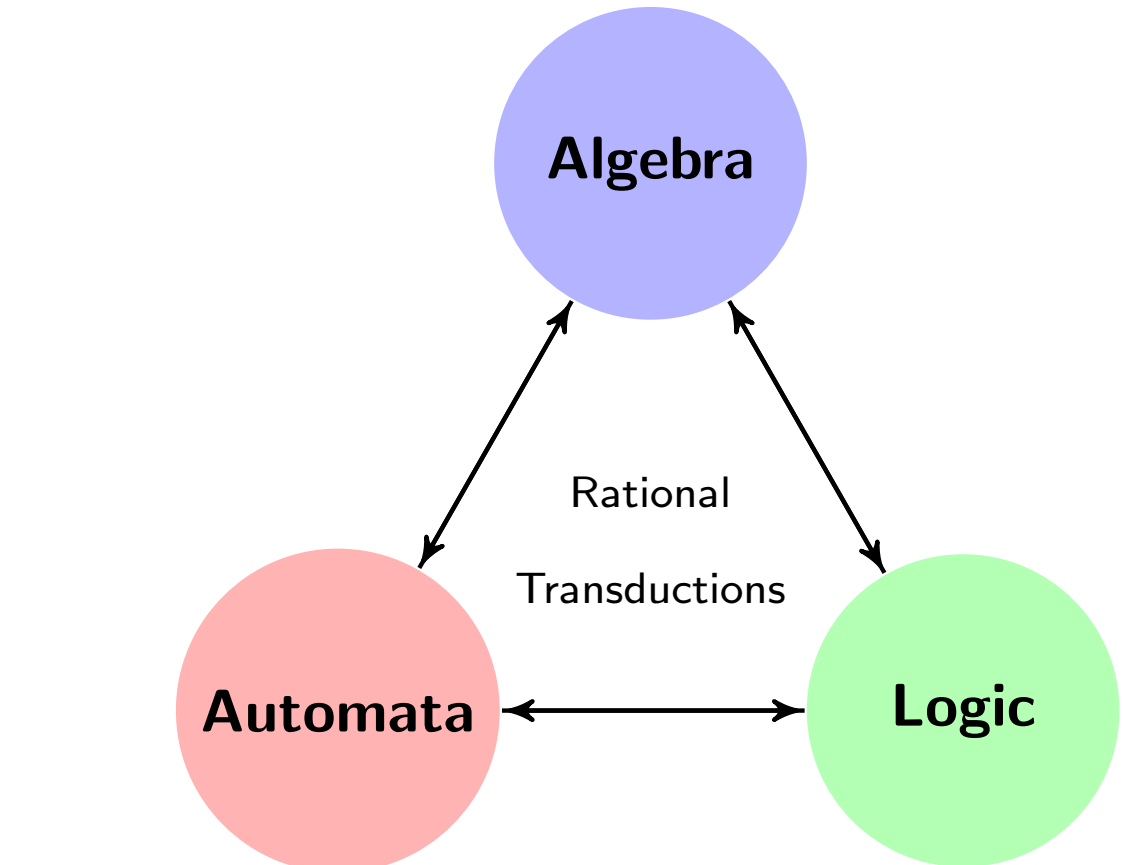


input-deterministic
one-way transducers

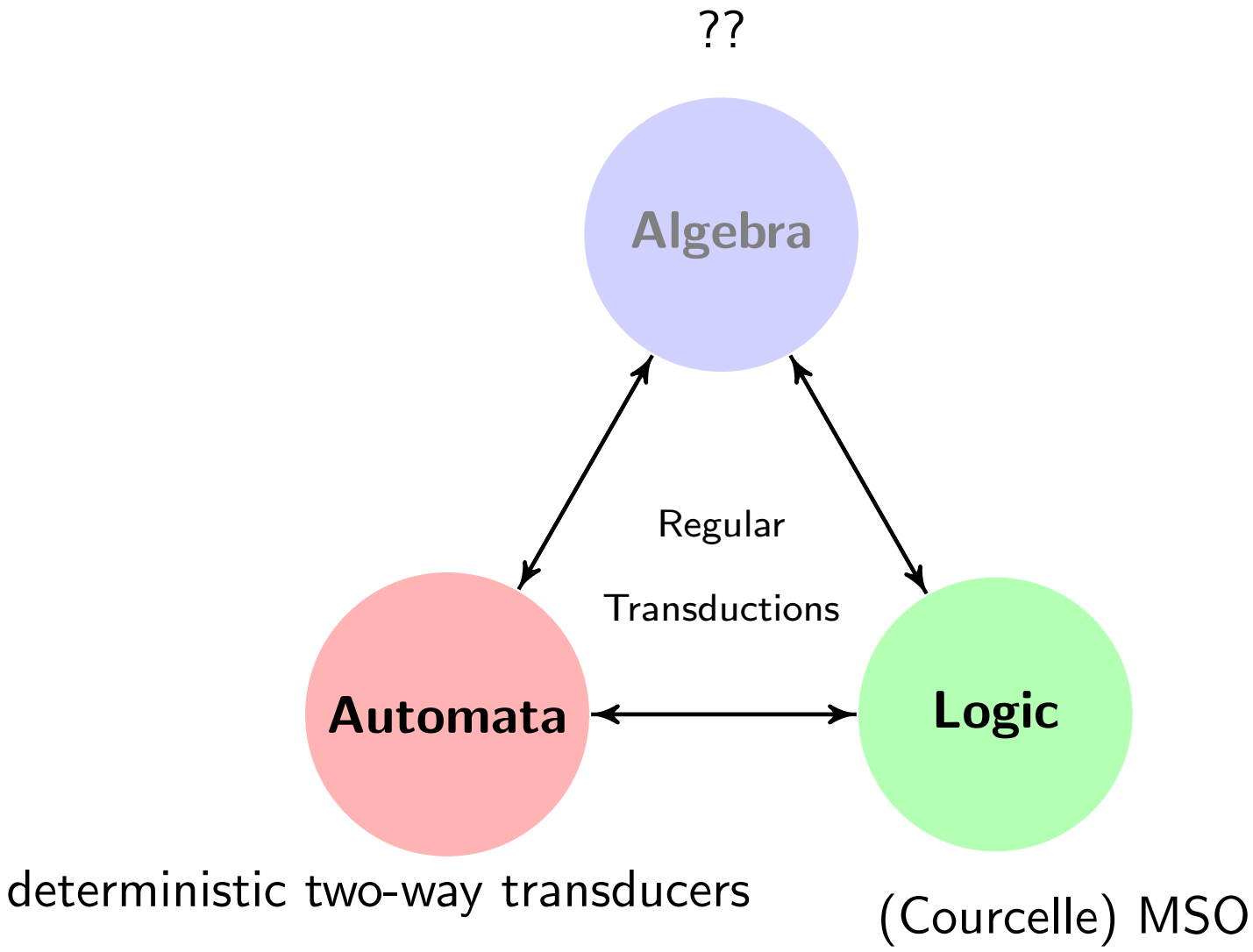
prefix-independent MSO, ?

Summary: rational transductions

finiteness of $\equiv_f$ and $\sim_{f[\equiv_f]}$ (Reutenauer, Schützenberger)



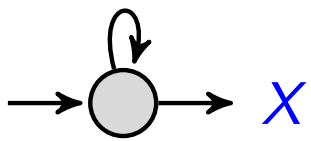
Summary: regular transductions



Transducers with Registers FST_{re} (Alur, Cerny, 10)

- deterministic one-way
- register updates:
  $X := YZ$, $X := u$

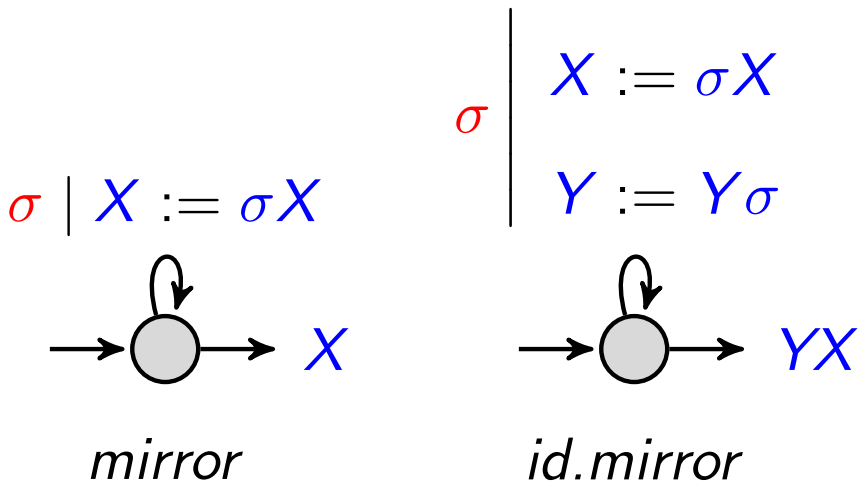
$\sigma \mid X := \sigma X$



mirror

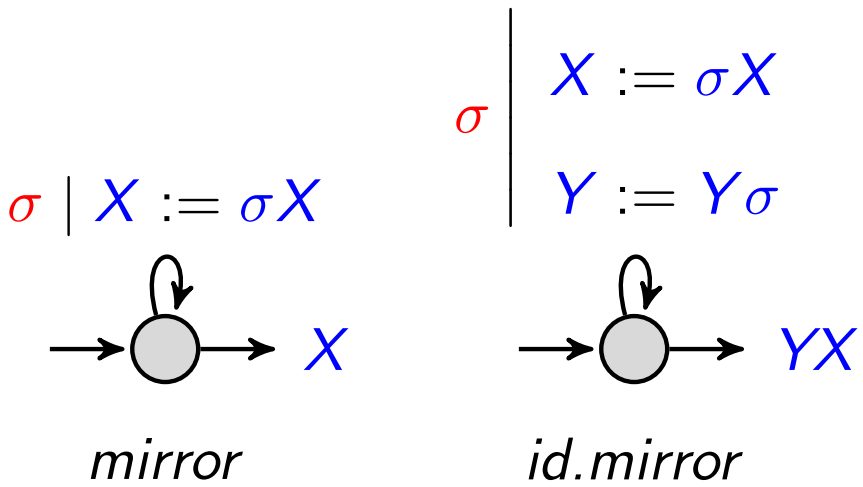
Transducers with Registers FST_{re} (Alur, Cerny, 10)

- deterministic one-way
- register updates:
  $X := YZ$, $X := u$



Transducers with Registers FST_{re} (Alur, Cerny, 10)

- deterministic one-way
- register updates: $X := YZ$, $X := u$



Results

- MSO-equivalent if copyless updates (Alur, Cerny, 10)
- decidable equivalence problem (F., Reynier) $\sim$ HDT0L

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

$\mathcal{S}$	repr	$\mathcal{T}$	status	reference
SEQ	.	AC^0	decidable	Cadilhac,Krebs,Ludwig,Paperman,15

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

$\mathcal{S}$	repr	$\mathcal{T}$	status	reference
SEQ	.	AC^0	decidable	Cadilhac,Krebs,Ludwig,Paperman,15
RAT	bimachine	opFO=**Ap**-FST	PSpace-c	Lhote, F, Gauwin, submitted

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

$\mathcal{S}$	repr	$\mathcal{T}$	status	reference
SEQ	.	AC^0	decidable	Cadilhac,Krebs,Ludwig,Paperman,15
RAT	bimachine	opFO=**Ap**-FST	PSpace-c	Lhote, F, Gauwin, submitted
RAT	.	**V**-FST	decidable	Lhote, F, Gauwin,16

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

$\mathcal{S}$	repr	$\mathcal{T}$	status	reference
SEQ	.	AC^0	decidable	Cadilhac,Krebs,Ludwig,Paperman,15
RAT	bimachine	opFO=**Ap**-FST	PSpace-c	Lhote, F, Gauwin, submitted
RAT	.	**V**-FST	decidable	Lhote, F, Gauwin,16
REG	2FST	RAT	decidable	F, Gauwin, Reynier, Servais,13

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

$\mathcal{S}$	repr	$\mathcal{T}$	status	reference
SEQ	.	AC^0	decidable	Cadilhac,Krebs,Ludwig,Paperman,15
RAT	bimachine	opFO=**Ap**-FST	PSpace-c	Lhote, F, Gauwin, submitted
RAT	.	**V**-FST	decidable	Lhote, F, Gauwin,16
REG	2FST	RAT	decidable	F, Gauwin, Reynier, Servais,13
SWEEP	2FST	k-SWEEP	ExpSpace	Baschenis,Gauwin,Muscholl,Puppis,16

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

$\mathcal{S}$	repr	$\mathcal{T}$	status	reference
SEQ	.	AC^0	decidable	Cadilhac,Krebs,Ludwig,Paperman,15
RAT	bimachine	opFO=**Ap**-FST	PSpace-c	Lhote, F, Gauwin, submitted
RAT	.	**V**-FST	decidable	Lhote, F, Gauwin,16
REG	2FST	RAT	decidable	F, Gauwin, Reynier, Servais,13
SWEEP	2FST	k-SWEEP	ExpSpace	Baschenis,Gauwin,Muscholl,Puppis,16
REG	2FST	FO	?	

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

$\mathcal{S}$	repr	$\mathcal{T}$	status	reference
SEQ	.	AC^0	decidable	Cadilhac,Krebs,Ludwig,Paperman,15
RAT	bimachine	opFO=**Ap**-FST	PSpace-c	Lhote, F, Gauwin, submitted
RAT	.	**V**-FST	decidable	Lhote, F, Gauwin,16
REG	2FST	RAT	decidable	F, Gauwin, Reynier, Servais,13
SWEEP	2FST	k-SWEEP	ExpSpace	Baschenis,Gauwin,Muscholl,Puppis,16
REG	2FST	FO	?	
FO	.	**Ap**-*2FST*	*iff*	*Carton, Dartois,15*

Recent definability problems for functions

Given a transduction $f \in \mathcal{S}$, does $f \in \mathcal{T}$?

$\mathcal{S}$	repr	$\mathcal{T}$	status	reference
SEQ	.	AC^0	decidable	Cadilhac,Krebs,Ludwig,Paperman,15
RAT	bimachine	opFO=**Ap**-FST	PSpace-c	Lhote, F, Gauwin, submitted
RAT	.	**V**-FST	decidable	Lhote, F, Gauwin,16
REG	2FST	RAT	decidable	F, Gauwin, Reynier, Servais,13
SWEEP	2FST	k-SWEEP	ExpSpace	Baschenis,Gauwin,Muscholl,Puppis,16
REG	2FST	FO	?	
FO	.	**Ap**-*2FST*	*iff*	*Carton, Dartois,15*
				Dartois, Jecker, Reynier, 16
FO	.	**Ap**-FST_{re}	*iff*	*F, Trivedi, Krishna, 14*

Origin Information

Asynchronocity

- make reasoning about transducers difficult
- equivalence problem is undecidable for NFT

Origin Information

Asynchronocity

- make reasoning about transducers difficult
- equivalence problem is undecidable for NFT

Origin semantics (Bojanczyk, 14)

Reintroduce synchronicity: $\Sigma^* \to (\Sigma^* \times (\mathbb{N} \to \mathbb{N}))$

Origin Information

Asynchronocity

- make reasoning about transducers difficult
- equivalence problem is undecidable for NFT

Origin semantics (Bojanczyk, 14)

Reintroduce synchronicity: $\Sigma^* \to (\Sigma^* \times (\mathbb{N} \to \mathbb{N}))$

Results

- machine-independent characterization of regular transductions with origin and first-order definability (Bojanczyk, 14)
- relaxation of origin semantics & decision problems (F., Jecker, Löding, Winter, 16)

Other works

- regular combinators (Alur, Freilich, Raghothaman, 14)
- variants of two-way transducers (Guillon, Choffrut, 14,15,16), (Carton, 12) (McKenzie, Schwentick, Thérien, Vollmer, 06)
- extensions with non-determinism

SIGLOG News 9th

Transducers, Logic and Algebra for Functions of Finite Words

Emmanuel Filiot, Université Libre de Bruxelles
and Pierre-Alain Reynier, Aix-Marseille Université

The robust theory of regular languages is based on three important pillars: computation (automata)
and algebra. In this paper, we survey old and recent results on extensions of these pillars to functio
words to words. We consider two important classes of word functions, the rational and regular f
respectively defined by one-way and two-way automata with output words, called transducers.

ween computation, mathematical logic, and algeb
f finite words. The class of regular language
ite automata, to the class of languag
ssor [Büchi 1960; Elgot 196
onoid, a canonical
While a
the a

SIGLOG News 9th

Thank You.

Transducers, Logic and Algebra for Functions of Finite Words

Emmanuel Filiot, Université Libre de Bruxelles
and Pierre-Alain Reynier, Aix-Marseille Université

The robust theory of regular languages is based on three important pillars: computation (automata)
and algebra. In this paper, we survey old and recent results on extensions of these pillars to functio
words to words. We consider two important classes of word functions, the rational and regular f
respectively defined by one-way and two-way automata with output words, called transducers.