Specification and Computation of Word Transductions

Emmanuel Filiot

Université libre de Bruxelles & FNRS

Workshop 'Trends in Transformations', 2018

Specification

Describe properties

Computation

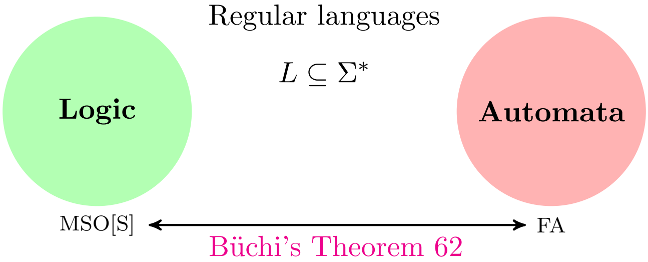
Decide those properties

Specification

Describe properties

Computation

Decide those properties



Regular languages

$L \subseteq \Sigma^*$

Logic

Automata

MSO[S]

FA

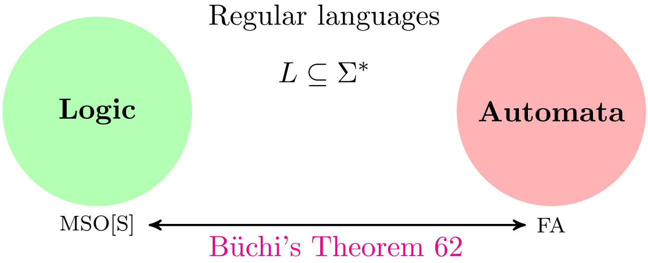
Büchi's Theorem 62

Specification

Describe properties

Computation

Decide those properties



Regular languages

$L \subseteq \Sigma^*$

Logic

Automata

MSO[S]

FA

Büchi's Theorem 62

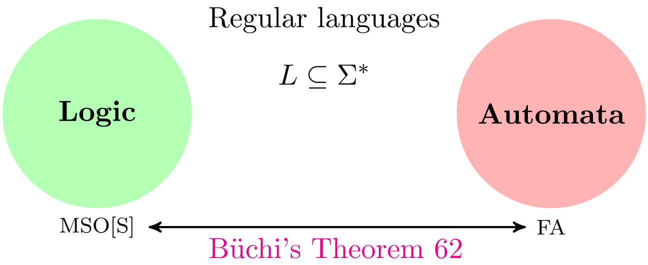
Many extensions: infinite words, finite and infinite trees, graphs, other logics ...

Specification

Describe properties

Computation

Decide those properties



Regular languages

$L \subseteq \Sigma^*$

Logic

Automata

MSO[S]

FA

Büchi's Theorem 62

Many extensions: infinite words, finite and infinite trees, graphs, other logics ...

Famous application: Model-checking $A \models \phi$?

Specification

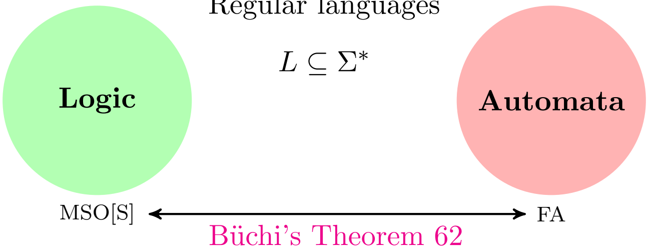
Describe properties

Computation

Decide those properties

Regular languages

$L \subseteq \Sigma^*$



Many extensions: infinite words, finite and infinite trees, graphs, other logics ...

Famous application: Model-checking $A \models \phi$?

$L(A) \cap L(A_{\neg\phi}) = \varnothing$

Objective of the talk

What about transductions ?

$$f : \Sigma^* \hookrightarrow \Sigma^*$$

Objective of the talk

What about transductions ?

$$f : \Sigma^* \hookrightarrow \Sigma^*$$

Append a # $abbab \mapsto abbab\#$

Objective of the talk

What about transductions ?

$$f : \Sigma^* \hookrightarrow \Sigma^*$$

Append a #	$abbab \mapsto abbab\#$
Delete all b	$abbab \mapsto aa$

Objective of the talk

What about transductions ?

$$f : \Sigma^* \hookrightarrow \Sigma^*$$

Append a #	$abbab \mapsto abbab\#$
Delete all b	$abbab \mapsto aa$
Squeeze all white space sequences ≥ 2	$fsttcs\textvisiblespace\textvisiblespace\textvisiblespace 18 \mapsto fsttcs\textvisiblespace 18$

Objective of the talk

What about transductions ?

$$f : \Sigma^* \hookrightarrow \Sigma^*$$

Append a #	$abbab \mapsto abbab\#$
Delete all b	$abbab \mapsto aa$
Squeeze all white space sequences ≥ 2	$fsttcs\textvisiblespace\textvisiblespace\textvisiblespace 18 \mapsto fsttcs\textvisiblespace 18$
Add a parity bit	$0100101 \mapsto \mathbf{1}100101$

Objective of the talk

What about transductions ?

$$f : \Sigma^* \hookrightarrow \Sigma^*$$

Append a #	$abbab \mapsto abbab\#$
Delete all b	$abbab \mapsto aa$
Squeeze all white space sequences ≥ 2	$fsttcs\text{␣␣␣}18 \mapsto fsttcs\text{␣}18$
Add a parity bit	$0100101 \mapsto \mathbf{1}100101$
Mirror the input word	$trends18 \mapsto 81sdnert$

Objective of the talk

What about transductions ?

$$f : \Sigma^* \hookrightarrow \Sigma^*$$

Append a #	$abbab \mapsto abbab\#$
Delete all b	$abbab \mapsto aa$
Squeeze all white space sequences ≥ 2	$fsttcs␣␣␣18 \mapsto fsttcs␣18$
Add a parity bit	$0100101 \mapsto \mathbf{1}100101$
Mirror the input word	$trends18 \mapsto 81sdnert$
Copy the input word	$krishna \mapsto krishnakrishna$

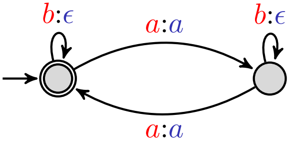
Outline

1. automata for transductions
2. logics for transductions
3. recent results

Automata models for transductions

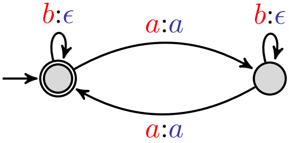
Automata for transductions: transducers

f_{del} :



Automata for transductions: transducers

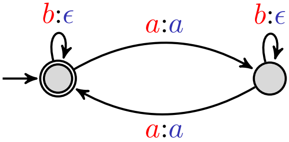
f_{del} :



$aabaa \mapsto aaaa$

Automata for transductions: transducers

f_{del} :

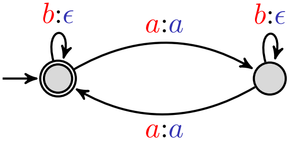


$$aabaa \mapsto aaaa$$

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

Automata for transductions: transducers

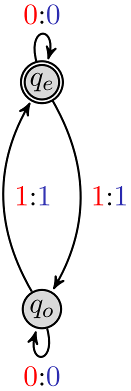
f_{del} :



$$\begin{array}{rcl} aabaa & \mapsto & aaaa \\ aaba & \mapsto & \text{undefined} \\ dom(f_{del}) & = & \text{'even number of } a\text{'} \end{array}$$

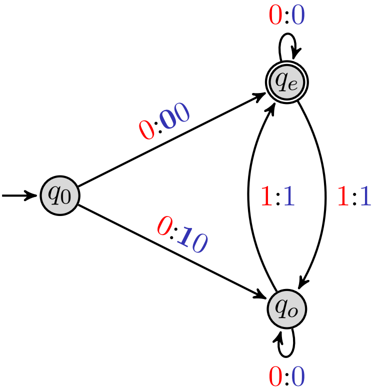
Parity bit

$01101 \mapsto \mathbf{1}01101, 01111 \mapsto \mathbf{0}01111$



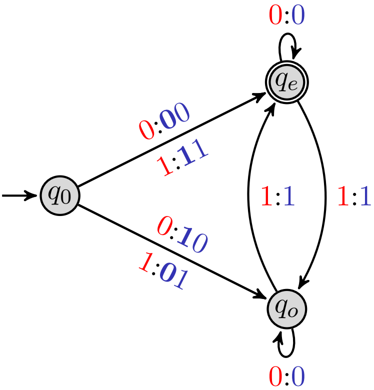
Parity bit

$01101 \mapsto \mathbf{1}01101, 01111 \mapsto \mathbf{0}01111$



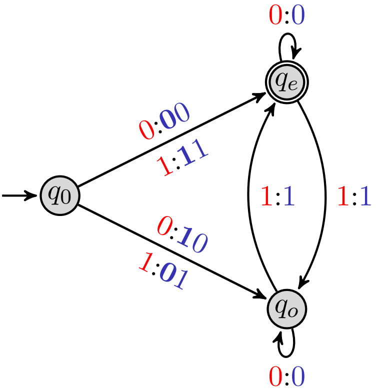
Parity bit

$01101 \mapsto \mathbf{1}01101, 01111 \mapsto \mathbf{0}01111$



Parity bit

$01101 \mapsto \mathbf{1}01101, 01111 \mapsto \mathbf{0}01111$



- input non-determinism needed here (aka non-sequential)
- PTIME decidable whether non-determinism is necessary

Choffrut, Sakarovitch, Carton,Beal,Prieur

Equivalence problem

Def Let $f, g : \Sigma^* \hookrightarrow \Sigma^*$ given by transducers T_f, T_g such that $dom(f) = dom(g)$. Decide whether $f = g$.

Equivalence problem

Def Let $f, g : \Sigma^* \hookrightarrow \Sigma^*$ given by transducers T_f, T_g such that $dom(f) = dom(g)$. Decide whether $f = g$.

Lem (Schützenberger) Inequivalence is witnessed by runs r_1, r_2 such that

(1) r_1, r_2 are over the same input
(2) r_1, r_2 produce different outputs
(3) r_1, r_2 have polynomial length

Equivalence problem

Def Let $f, g : \Sigma^* \hookrightarrow \Sigma^*$ given by transducers T_f, T_g such that $dom(f) = dom(g)$. Decide whether $f = g$.

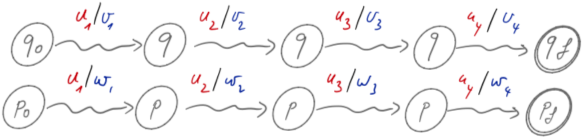
Lem (Schützenberger) Inequivalence is witnessed by runs r_1, r_2 such that

(1) r_1, r_2 are over the same input
(2) r_1, r_2 produce different outputs
(3) r_1, r_2 have polynomial length

Coro Equivalence is decidable in PSpace.

Equivalence problem

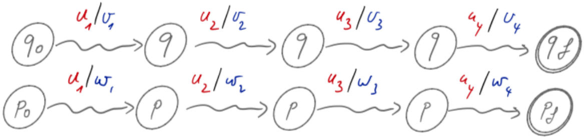
Lem Proof Assume $v = f(u) \neq g(u) = w$. If u is long enough, there exists a decomposition:



with $u = u_1u_2u_3u_4$ ($|u_2| > 0$,$|u_3| > 0$), $v = v_1v_2v_3v_4$, $w = w_1w_2w_3w_4$.

Equivalence problem

Lem Proof Assume $v = f(u) \neq g(u) = w$. If u is long enough, there exists a decomposition:



with $u = u_1 u_2 u_3 u_4$ ($|u_2| > 0$,$|u_3| > 0$), $v = v_1 v_2 v_3 v_4$, $w = w_1 w_2 w_3 w_4$. Show one of the following cases hold:

1. $f(u_1 u_4) = v_1 v_4 \neq g(u_1 u_4) = w_1 w_4$
2. $f(u_1 u_2 u_4) = v_1 v_2 v_4 \neq g(u_1 u_2 u_4) = w_1 w_2 w_4$
3. $f(u_1 u_3 u_4) = v_1 v_3 v_4 \neq g(u_1 u_3 u_4) = w_1 w_3 w_4$

Equivalence problem

Def Let $f, g : \Sigma^* \hookrightarrow \Sigma^*$ given by transducers T_f, T_g such that $dom(f) = dom(g)$. Decide whether $f = g$.

Lem (Schützenberger) Inequivalence is witnessed by runs r_1, r_2 such that

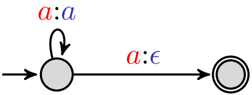
(1) r_1, r_2 are over the same input
(2) r_1, r_2 produce different outputs
(3) r_1, r_2 have polynomial length

Coro Equivalence is decidable in PSpace.

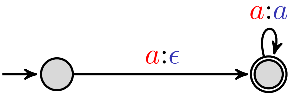
Thm (Gurari,Ibarra,83). Equivalence is decidable in PTime.

Transducer equivalence vs Automata equivalence

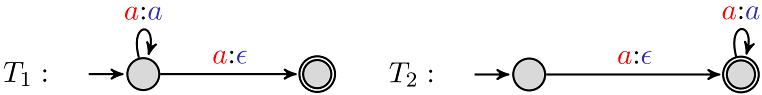
T_1 :



T_2 :



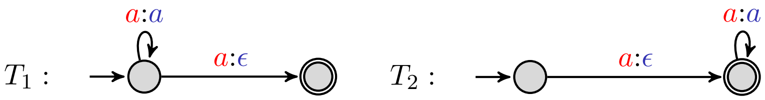
Transducer equivalence vs Automata equivalence



- Same transduction but different languages:

$$(a, \epsilon)(a, a)(a, a) \neq (a, a)(a, a)(a, \epsilon)$$

Transducer equivalence vs Automata equivalence



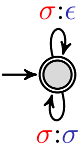
- Same transduction but different languages:

$$(a,\epsilon)(a,a)(a,a) \neq (a,a)(a,a)(a,\epsilon)$$

- Transducers are **asynchronous**
- Make most transducer problems conceptually difficult (and even computationally).

Non-determinism and relations

In general, transducers define binary *relations* in $\Sigma^* \times \Sigma^*$

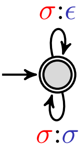


realizes $\{(u, v) \mid v \text{ is a subword of } u\}$

[1] $\exists K \forall u \ |T(u)| \leq K$

Non-determinism and relations

In general, transducers define binary *relations* in $\Sigma^* \times \Sigma^*$



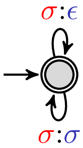
realizes $\{(u, v) \mid v \text{ is a subword of } u\}$

Thm. (Gurari, Ibarra 83) PTIME-decidable whether a given transducer defines a function.

[1] $\exists K \forall u \; |T(u)| \leq K$

Non-determinism and relations

In general, transducers define binary *relations* in $\Sigma^* \times \Sigma^*$



realizes $\{(u, v) \mid v \text{ is a subword of } u\}$

Thm. (Gurari, Ibarra 83) PTIME-decidable whether a given transducer defines a function.

Equivalence Problem

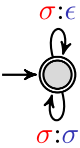
Let $R_1, R_2 \subseteq \Sigma^* \times \Gamma^*$ given by transducers. Decide if $R_1 = R_2$.

- Undecidable (Griffith 68), even if one alphabet is unary (Ibarra 78)

[1] $\exists K \forall u \; |T(u)| \leq K$

Non-determinism and relations

In general, transducers define binary *relations* in $\Sigma^* \times \Sigma^*$



realizes $\{(u, v) \mid v \text{ is a subword of } u\}$

Thm. (Gurari, Ibarra 83) PTIME-decidable whether a given transducer defines a function.

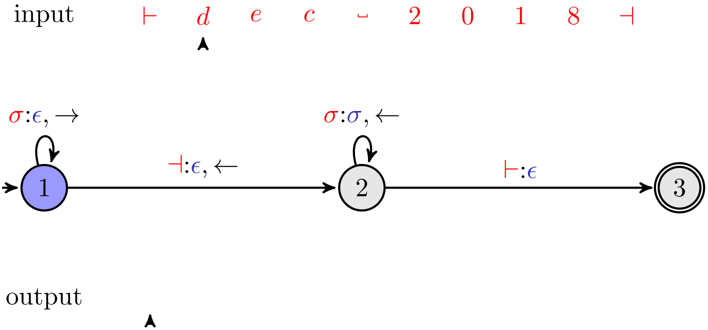
Equivalence Problem

Let $R_1, R_2 \subseteq \Sigma^* \times \Gamma^*$ given by transducers. Decide if $R_1 = R_2$.

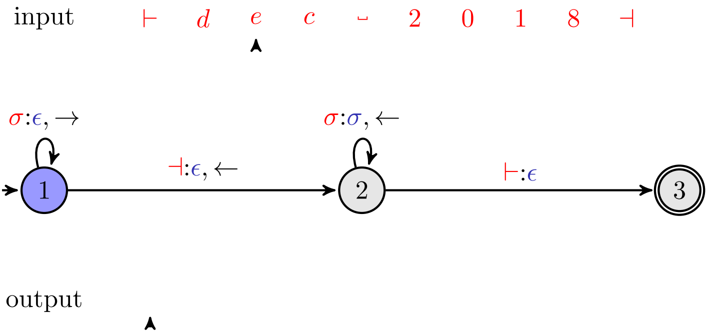
- Undecidable (Griffith 68), even if one alphabet is unary (Ibarra 78)
- Decidable for bounded-valued transducers (Culik Karhumäki 86) [1].

[1] $\exists K \forall u \; |T(u)| \leq K$

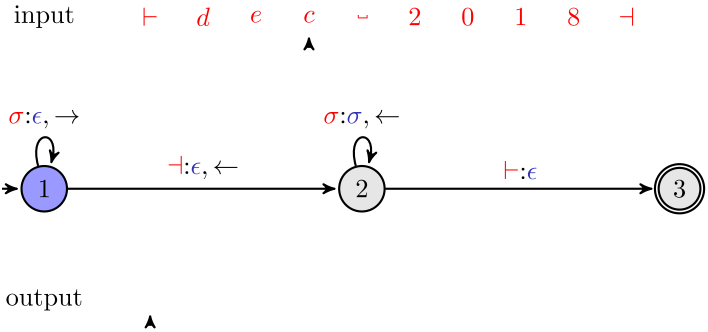
Two-way finite transducers (2FT)



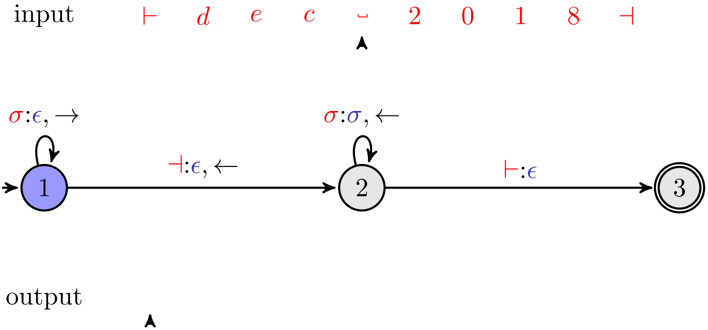
Two-way finite transducers (2FT)



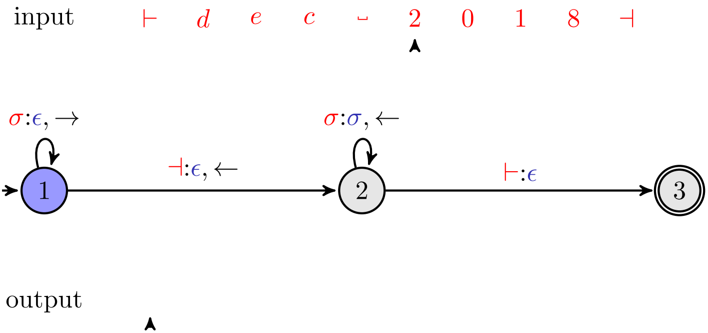
Two-way finite transducers (2FT)


input
$\vdash$ d e c ␣ 2 0 1 8 $\dashv$
σ:ϵ, $\rightarrow$
σ:σ, $\leftarrow$
$\dashv$:ϵ, $\leftarrow$
$\vdash$:ϵ
1
2
3
output

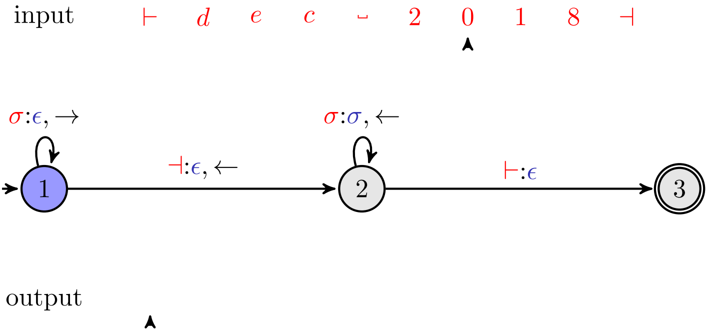
Two-way finite transducers (2FT)


input
$\vdash$ d e c ␣ 2 0 1 8 $\dashv$
σ:ϵ, $\rightarrow$
σ:σ, $\leftarrow$
$\dashv$:ϵ, $\leftarrow$
$\vdash$:ϵ
1
2
3
output

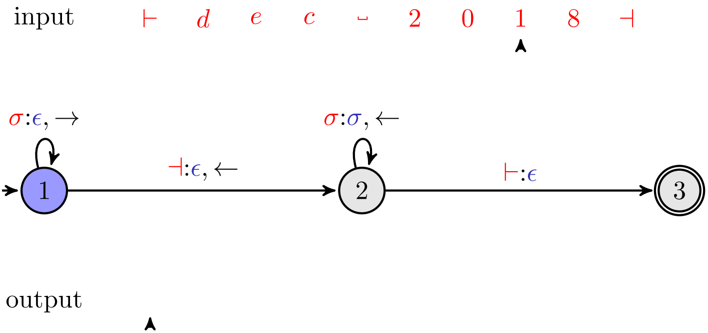
Two-way finite transducers (2FT)



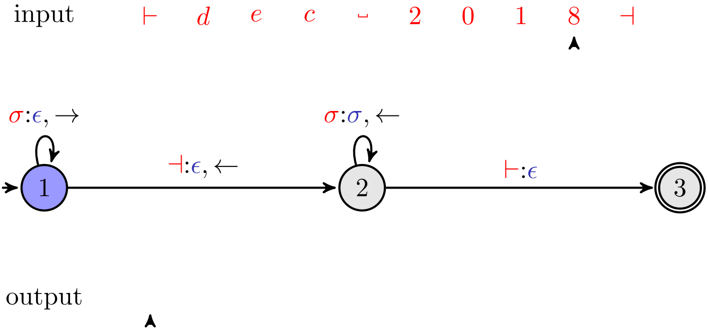
Two-way finite transducers (2FT)



Two-way finite transducers (2FT)


input
$\vdash$ d e c ␣ 2 0 1 8 $\dashv$
σ:ϵ, $\rightarrow$
σ:σ, $\leftarrow$
$\dashv$:ϵ, $\leftarrow$
$\vdash$:ϵ
1
2
3
output

Two-way finite transducers (2FT)



input $\vdash$ d e c ␣ 2 0 1 8 $\dashv$

σ:ϵ, $\rightarrow$

σ:σ, $\leftarrow$

$\dashv$:ϵ, $\leftarrow$

$\vdash$:ϵ

1

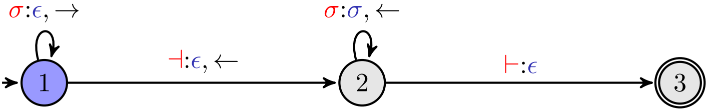
2

3

output

Two-way finite transducers (2FT)

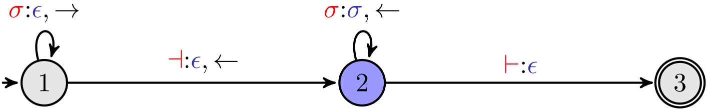
input $\vdash$ d e c ␣ 2 0 1 8 $\dashv$


σ:ϵ, $\rightarrow$
σ:σ, $\leftarrow$
$\dashv$:ϵ, $\leftarrow$
$\vdash$:ϵ
1
2
3

output

Two-way finite transducers (2FT)

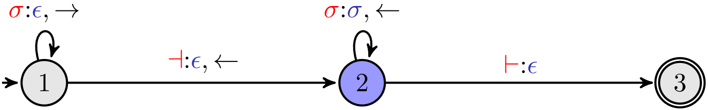
input $\vdash$ d e c ␣ 2 0 1 8 $\dashv$



output

Two-way finite transducers (2FT)

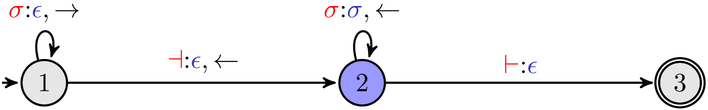
input $\vdash$ d e c ␣ 2 0 1 8 $\dashv$



output 8

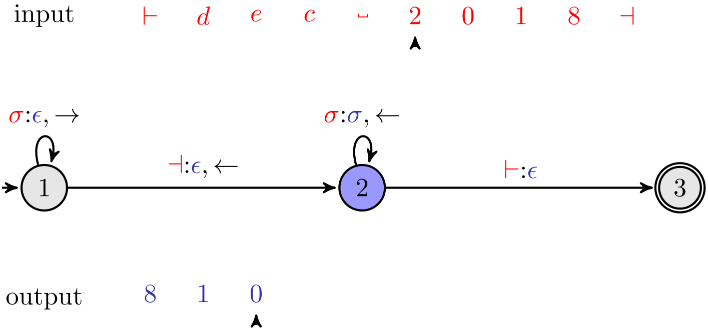
Two-way finite transducers (2FT)

input $\vdash$ d e c ␣ 2 0 1 8 $\dashv$

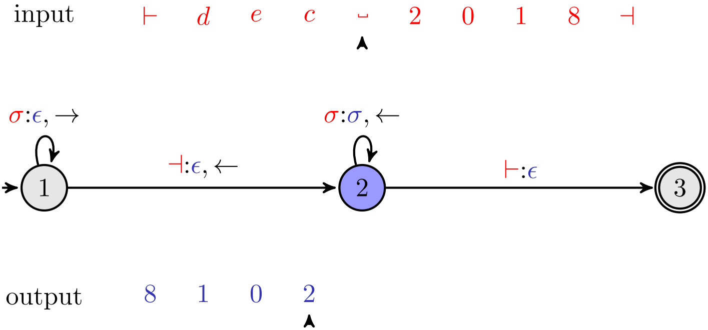


output 8 1

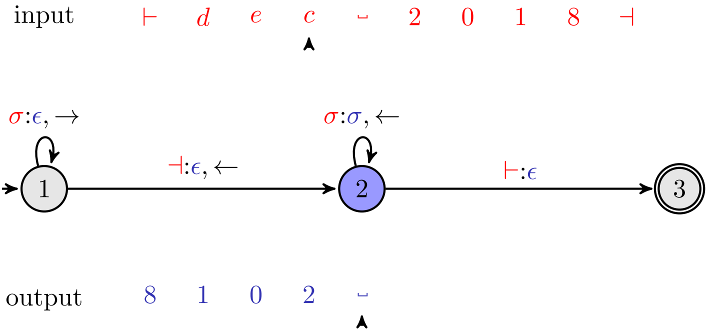
Two-way finite transducers (2FT)



Two-way finite transducers (2FT)

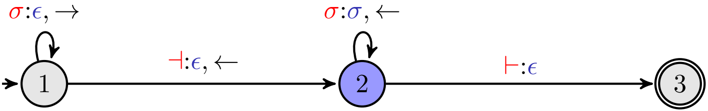


Two-way finite transducers (2FT)



Two-way finite transducers (2FT)

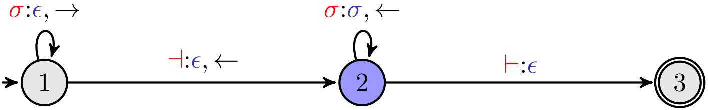
input $\vdash$ d e c ␣ 2 0 1 8 $\dashv$


σ:ϵ, $\rightarrow$
$\dashv$:ϵ, $\leftarrow$
σ:σ, $\leftarrow$
$\vdash$:ϵ
1
2
3

output 8 1 0 2 ␣ c

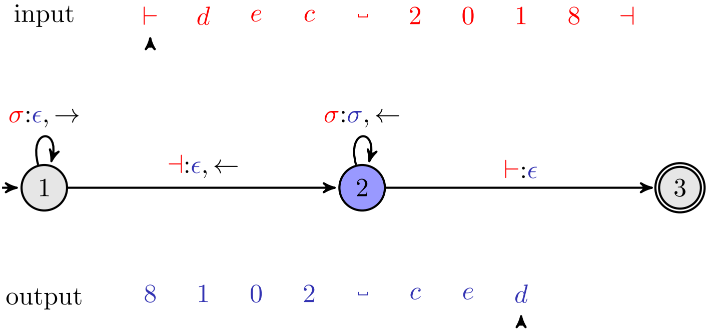
Two-way finite transducers (2FT)

input $\vdash$ d e c ␣ 2 0 1 8 $\dashv$



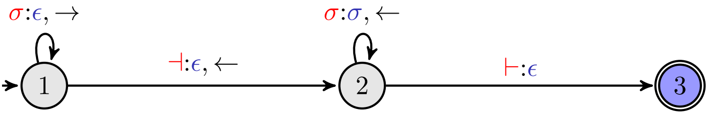
output 8 1 0 2 ␣ c e

Two-way finite transducers (2FT)



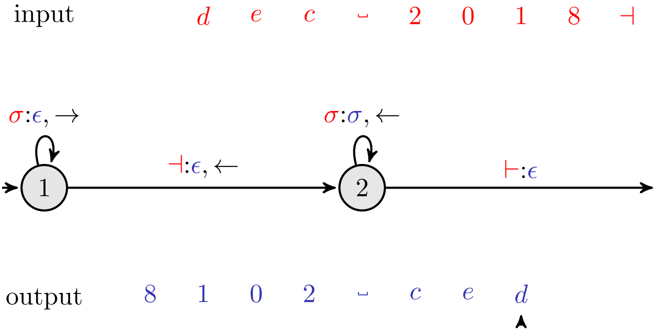
Two-way finite transducers (2FT)

input $\vdash$ d e c ␣ 2 0 1 8 $\dashv$



output 8 1 0 2 ␣ c e d

Two-way finite transducers (2FT)

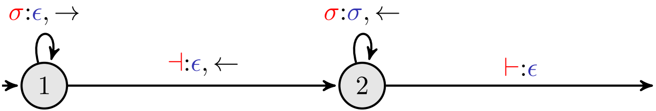

input
d e c ␣ 2 0 1 8 $\dashv$
σ:ϵ, $\rightarrow$
σ:σ, $\leftarrow$
$\dashv$:ϵ, $\leftarrow$
$\vdash$:ϵ
1
2
output
8 1 0 2 ␣ c e d

Other examples

$dec, 2018 \mapsto 2018, dec$

Two-way finite transducers (2FT)

input d e c ␣ 2 0 1 8 ⊣



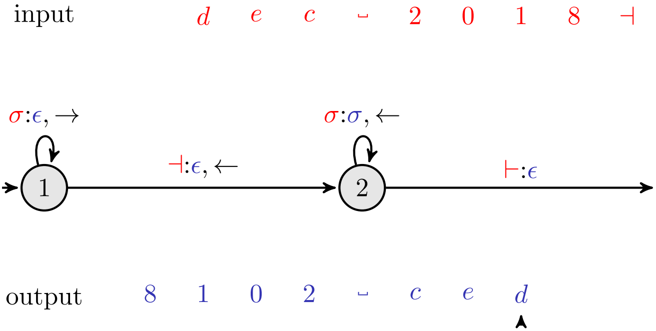
output 8 1 0 2 ␣ c e d

Other examples

$dec, 2018 \mapsto 2018, dec$

$u \mapsto uu$ (copy)

Two-way finite transducers (2FT)


input
d e c ␣ 2 0 1 8 $\dashv$
$\sigma{:}\epsilon, \rightarrow$
$\sigma{:}\sigma, \leftarrow$
$\dashv{:}\epsilon, \leftarrow$
$\vdash{:}\epsilon$
1
2
output
8 1 0 2 ␣ c e d

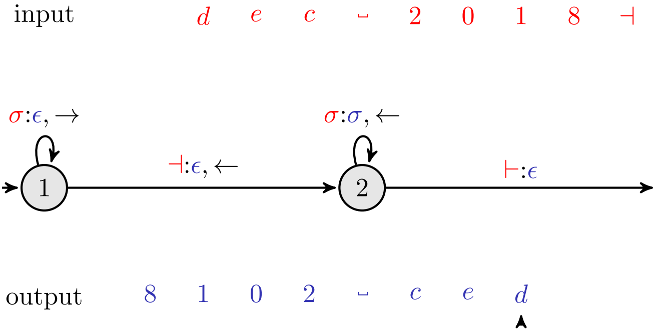
Other examples

$dec, 2018 \mapsto 2018, dec$

$u \mapsto uu$ (copy)

$u \mapsto f_1(u) f_2(u)$ (copy+trans)

Two-way finite transducers (2FT)



Other examples

$dec, 2018 \mapsto 2018, dec$

$u \mapsto uu$ (copy)

$u \mapsto f_1(u)f_2(u)$ (copy+trans)

$u_1\#u_2\#\ldots\#u_k \mapsto \overline{u_1}\#\ldots\#\overline{u_k}$ (local reverse)

Some important results on two-way transducers

Over (functional) transductions:

- equivalence is decidable in PSPACE

  (Gurari 82) (Culik, Karhumäki,87)

Some important results on two-way transducers

Over (functional) transductions:

- equivalence is decidable in PSPACE

  (Gurari 82) (Culik, Karhumäki,87)

- closed under composition

  (Chytil, Jakl, 77)

Some important results on two-way transducers

Over (functional) transductions:

- equivalence is decidable in PSPACE

  (Gurari 82) (Culik, Karhumäki,87)
- closed under composition

  (Chytil, Jakl, 77)
- equivalent to **reversible** two-way transducers

  (Dartois,Fournier,Jecker,Lhote,17)

Some important results on two-way transducers

Over (functional) transductions:

- equivalence is decidable in PSPACE

  (Gurari 82) (Culik, Karhumäki,87)

- closed under composition

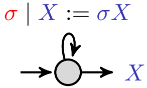
  (Chytil, Jakl, 77)

- equivalent to **reversible** two-way transducers

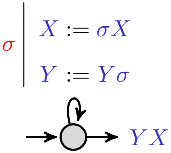
  (Dartois,Fournier,Jecker,Lhote,17)

- and to many other models ...

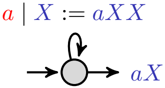
Transducers with registers



mirror



id.mirror



$exp{:}a^n \mapsto a^{2^n}$

- deterministic one-way
- equivalent to 2FT if linear updates

(Alur, Cerny, 10)

- decidable equivalence problem

(F., Reynier,17) (Benedikt et. al., 17)

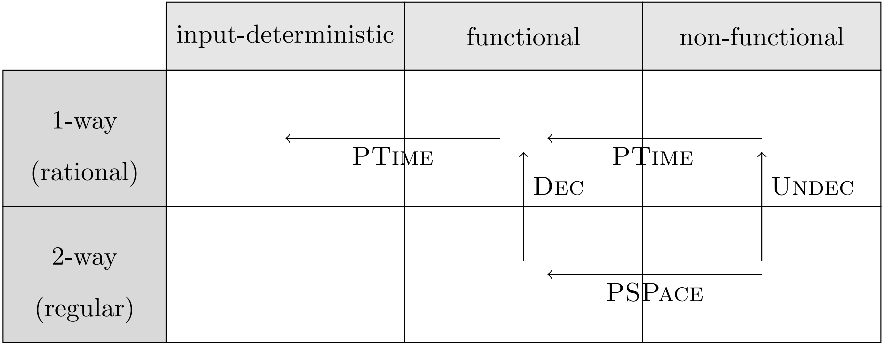
Summary – Expressiveness

	input-deterministic		functional		non-functional
1-way (rational)		<		<	
	∧		∧		∧
2-way (regular)		=		<	

Summary – Expressiveness

	input-deterministic	functional	non-functional
1-way (rational)		← PTime	← PTime
2-way (regular)			← PSpace

Summary – Expressiveness



	input-deterministic	functional	non-functional
1-way (rational)			
2-way (regular)			

PTime

PTime

Dec

Undec

PSPace

(F., Gauwin,Reynier,Servais,13)

(Baschenis, Gauwin,Muscholl,Puppis,17)

Summary – Equivalence problem

	input-deterministic	functional	non-functional
1-way (rational)	PTIME	PTIME	undec
2-way (regular)	PSPACE	PSPACE	undec

Logics for transductions

MSO on words

Over some finite alphabet Σ:

$$\varphi \ ::= \ \varphi \wedge \varphi \mid \neg\varphi \mid \exists x \varphi \mid \exists X \varphi \mid x \in X \mid \sigma(x) \mid S(x,y) \qquad \sigma \in \Sigma$$

Over finite words, (set) variables interpreted by (sets of) positions.

MSO on words

Over some finite alphabet Σ:

$$\varphi \ ::= \ \varphi \wedge \varphi \mid \neg\varphi \mid \exists x \varphi \mid \exists X \varphi \mid x \in X \mid \sigma(x) \mid S(x,y) \qquad \sigma \in \Sigma$$

Over finite words, (set) variables interpreted by (sets of) positions.

Some examples

- $\leq$: transitive closure of S
- first position is an a: $a(x) \wedge \forall y(x \leq y)$
- counting modulo: odd number of a, even length, ...

MSO on words

Over some finite alphabet Σ:

$$\varphi ::= \varphi \wedge \varphi \mid \neg\varphi \mid \exists x \varphi \mid \exists X \varphi \mid x \in X \mid \sigma(x) \mid S(x,y) \qquad \sigma \in \Sigma$$

Over finite words, (set) variables interpreted by (sets of) positions.

Some examples

- $\leq$: transitive closure of S
- first position is an a: $a(x) \wedge \forall y (x \leq y)$
- counting modulo: odd number of a, even length, ...

$L_\phi = \{w \in \Sigma^* \mid w \models \phi\}$

MSO on words

Over some finite alphabet Σ:

$$\varphi \;::=\; \varphi \wedge \varphi \mid \neg\varphi \mid \exists x \varphi \mid \exists X \varphi \mid x \in X \mid \sigma(x) \mid S(x,y) \qquad \sigma \in \Sigma$$

Over finite words, (set) variables interpreted by (sets of) positions.

Some examples

- $\leq$: transitive closure of S
- first position is an a: $a(x) \wedge \forall y (x \leq y)$
- counting modulo: odd number of a, even length, ...

$L_\phi = \{w \in \Sigma^* \mid w \models \phi\}$

Büchi-Elgot-Trakhenbrot

A language L is MSO-definable iff it is recognisable by some finite automaton.

Extension to transductions

Example (Delete all b)

- replace input label **by** a **if it is** a
- replace input label **by** ϵ **if it is** b

Extension to transductions

Example (Delete all b)

- replace input label **by** a **if it is** a

$$\phi_a(x) \equiv a(x)$$

- replace input label **by** ϵ **if it is** b

$$\phi_\epsilon(x) \equiv b(x)$$

Extension to transductions

Example (Append #)

- replace label σ of x by σ if x is **not** the last position

- replace label σ of x by $\sigma\#$ if x is the last position

Extension to transductions

Example (Append #)

- replace label σ of x by σ if x is **not** the last position

$$\phi_{\sigma}(x) \equiv \sigma(x) \wedge \exists y \; S(x,y)$$

- replace label σ of x by $\sigma\#$ if x is the last position

$$\phi_{\sigma\#}(x) \equiv \sigma(x) \wedge \forall y \; \neg S(x,y)$$

Extension to transductions

Example (Add a parity bit)

- replace label σ of x by 1σ if x is the first position and odd number of 1
- replace label σ of x by 0σ if x is the first position and even number of 1
- replace label σ of x by σ if x is not the first position

Extension to transductions

Example (Add a parity bit)

- replace label σ of x by 1σ if x is the first position and odd number of 1

$$\phi_{1\sigma}(x) \equiv \sigma(x) \wedge \forall y\ \neg S(y,x) \wedge \phi_{odd}$$

- replace label σ of x by 0σ if x is the first position and even number of 1

$$\phi_{0\sigma}(x) \equiv \sigma(x) \wedge \forall y\ \neg S(y,x) \wedge \phi_{even}$$

- replace label σ of x by σ if x is not the first position

$$\phi_{\sigma}(x) \equiv \sigma(x) \wedge \forall y\ \neg S(y,x)$$

Büchi Theorem for Rational Transductions

Def $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable if it can be "described" by a finite set of formulas $\phi_{v_1}(x), \ldots, \phi_{v_k}(x)$ ($v_1, \ldots, v_k \subseteq \Sigma^*$).

Büchi Theorem for Rational Transductions

Def $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable if it can be "described" by a finite set of formulas $\phi_{v_1}(x), \ldots, \phi_{v_k}(x)$ ($v_1, \ldots, v_k \subseteq \Sigma^*$).

Thm $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable iff it is realisable by a finite state transducer.

Büchi Theorem for Rational Transductions

Def $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable if it can be "described" by a finite set of formulas $\phi_{v_1}(x), \ldots, \phi_{v_k}(x)$ $(v_1, \ldots, v_k \subseteq \Sigma^*)$.

Thm $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable iff it is realisable by a finite state transducer.

Proof Idea Transducers → MSO

For all transitions $t = p \xrightarrow{\sigma:v} q$ define $\phi_v^t(x)$

- which expresses the **existence of an accepting run which at position** x **triggers transition** t.
- latter property is regular, so MSO-definable (Büchi's Theorem)

(the transducer must be unambiguous)

Büchi Theorem for Rational Transductions

Def $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable if it can be "described" by a finite set of formulas $\phi_{v_1}(x), \dots, \phi_{v_k}(x)$ ($v_1, \dots, v_k \subseteq \Sigma^*$).

Thm $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable iff it is realisable by a finite state transducer.

Proof Idea Transducers → MSO

For all transitions $t = p \xrightarrow{\sigma:v} q$ define $\phi^t_v(x)$

- which expresses the **existence of an accepting run which at position** x **triggers transition** t.
- latter property is regular, so MSO-definable (Büchi's Theorem)

(the transducer must be unambiguous)

Büchi Theorem for Rational Transductions

Def $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable if it can be "described" by a finite set of formulas $\phi_{v_1}(x), \ldots, \phi_{v_k}(x)$ $(v_1, \ldots, v_k \subseteq \Sigma^*)$.

Thm $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable iff it is realisable by a finite state transducer.

Proof Idea MSO $\rightarrow$ Transducers

- states must hold enough information to decide MSO satisfiability
- states are MSO-types (τ_1, τ_2) of (prefix,suffix)
- if formula $\phi_v(x)$ holds at (τ_1, τ_2), output v

Büchi Theorem for Rational Transductions

Def $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable if it can be “described” by a finite set of formulas $\phi_{v_1}(x), \ldots, \phi_{v_k}(x)$ $(v_1, \ldots, v_k \subseteq \Sigma^*)$.

Thm $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable iff it is realisable by a finite state transducer.

Proof Idea MSO → Transducers

- states must hold enough information to decide MSO satisfiability
- states are MSO-types (τ_1, τ_2) of (prefix,suffix)
- if formula $\phi_v(x)$ holds at (τ_1, τ_2), output v

Büchi Theorem for Rational Transductions

Def $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable if it can be "described" by a finite set of formulas $\phi_{v_1}(x), \ldots, \phi_{v_k}(x)$ $(v_1, \ldots, v_k \subseteq \Sigma^*)$.

Thm $f : \Sigma^* \hookrightarrow \Sigma^*$ is MSO-definable iff it is realisable by a finite state transducer.

Proof Idea MSO → Transducers

- states must hold enough information to decide MSO satisfiability
- states are MSO-types (τ_1, τ_2) of (prefix,suffix)
- if formula $\phi_v(x)$ holds at (τ_1, τ_2), output v

Büchi Theorem for Rational Transductions

What about mirror ? $csl2018 \mapsto 8102lsc$

Replace label of position x by σ if $last - x$ is labeled σ.

Not MSO-definable.

(Courcelle) MSO Transducers

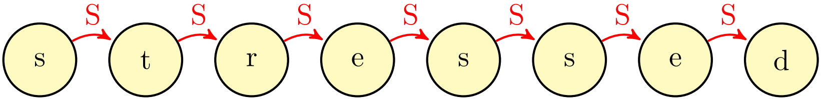
“interpreting the output structure in the input structure”

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

(Courcelle) MSO Transducers

"interpreting the output structure in the input structure"

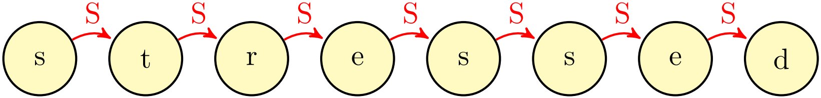
- output predicates defined by MSO formulas interpreted over the input structure



(Courcelle) MSO Transducers

"interpreting the output structure in the input structure"

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



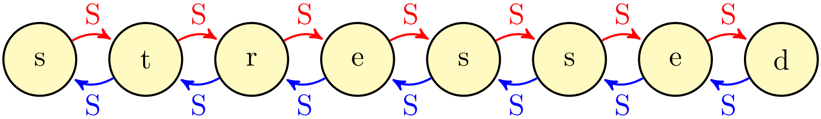
$$\phi_{S}(x,y) \equiv S(y,x)$$

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

(Courcelle) MSO Transducers

"interpreting the output structure in the input structure"

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



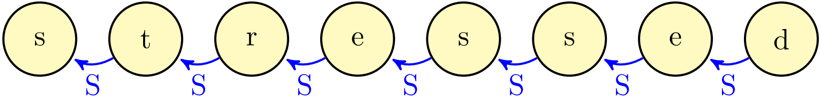
$$\phi_{S}(x,y) \equiv S(y,x)$$

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

(Courcelle) MSO Transducers

"interpreting the output structure in the input structure"

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



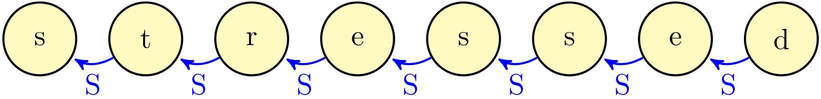
$$\phi_{S}(x,y) \equiv S(y,x)$$

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

(Courcelle) MSO Transducers

"interpreting the output structure in the input structure"

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

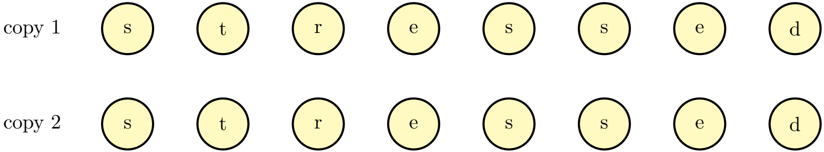


$$\phi_S(x,y) \equiv S(y,x)$$

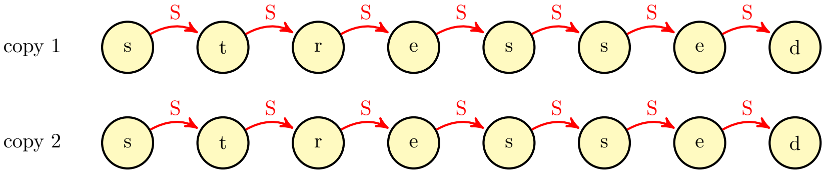
$$\phi_\sigma(x) \equiv \sigma(x)$$

- input structure can be copied a fixed number of times: $u \mapsto uu$, or $u \mapsto u.\mathrm{mirror}(u)$.

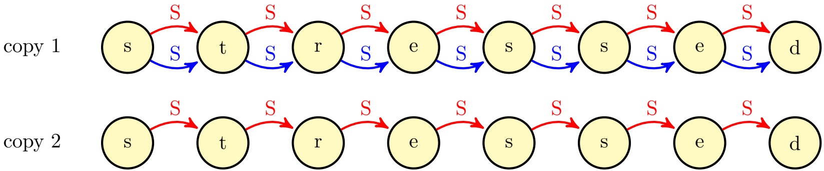
Other example : $u \mapsto u.\mathrm{mirror}(u)$



Other example : $u \mapsto u.\text{mirror}(u)$



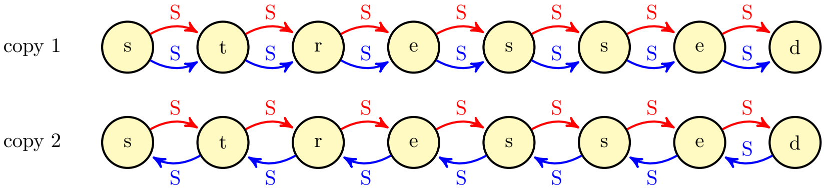
Other example : $u \mapsto u.\text{mirror}(u)$



Formulas

copy 1: $\phi^{\mathbf{1}}_{S}(x,y) \quad \equiv \quad S(x,y)$

Other example : $u \mapsto u.\mathrm{mirror}(u)$

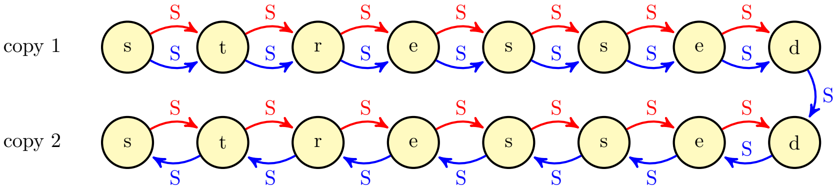


Formulas

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

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

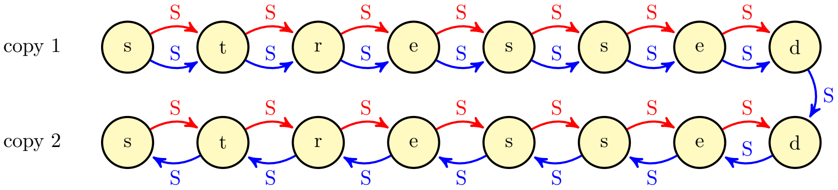
Other example : $u \mapsto u.\mathrm{mirror}(u)$



Formulas

$$\begin{aligned} \textbf{copy 1:} \quad & \phi_S^{\mathbf{1}}(x,y) \equiv S(x,y) \\ \textbf{copy 2:} \quad & \phi_S^{\mathbf{2}}(x,y) \equiv S(y,x) \\ \textbf{copy 1 to copy 2:} \quad & \phi_S^{\mathbf{1}\to\mathbf{2}}(x,y) \equiv x = y \wedge last(x) \end{aligned}$$

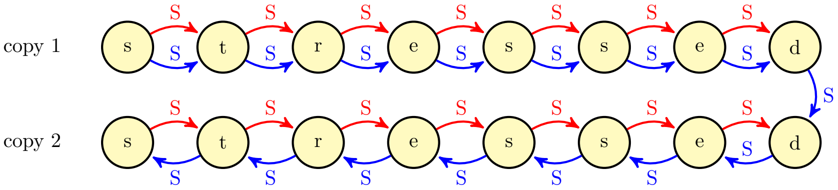
Other example : $u \mapsto u.\mathrm{mirror}(u)$



Formulas

$$\begin{aligned}
\textbf{copy 1:} \quad & \phi_S^{\mathbf{1}}(x,y) & \equiv \quad & S(x,y) \\
\textbf{copy 2:} \quad & \phi_S^{\mathbf{2}}(x,y) & \equiv \quad & S(y,x) \\
\textbf{copy 1 to copy 2:} \quad & \phi_S^{\mathbf{1}\to\mathbf{2}}(x,y) & \equiv \quad & x = y \wedge last(x) \\
\textbf{copy 2 to copy 1:} \quad & \phi_S^{\mathbf{2}\to\mathbf{1}}(x,y) & \equiv \quad & \perp
\end{aligned}$$

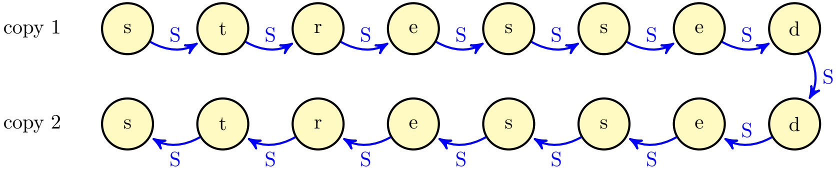
Other example : $u \mapsto u.\mathrm{mirror}(u)$



Formulas

$$\begin{aligned}
&\textbf{copy 1:} & \phi^{\mathbf{1}}_{S}(x,y) &\equiv S(x,y)\\
&\textbf{copy 2:} & \phi^{\mathbf{2}}_{S}(x,y) &\equiv S(y,x)\\
&\textbf{copy 1 to copy 2:} & \phi^{\mathbf{1}\to\mathbf{2}}_{S}(x,y) &\equiv x=y \wedge last(x)\\
&\textbf{copy 2 to copy 1:} & \phi^{\mathbf{2}\to\mathbf{1}}_{S}(x,y) &\equiv \perp\\
&\textbf{for all copies } i\textbf{:} & \phi^{\mathbf{i}}_{\sigma}(x) &\equiv \sigma(x)
\end{aligned}$$

Other example : $u \mapsto u.\text{mirror}(u)$



Formulas

$$
\begin{array}{rrcl}
\textbf{copy 1:} & \phi^{\mathbf{1}}_{S}(x,y) & \equiv & S(x,y) \\
\textbf{copy 2:} & \phi^{\mathbf{2}}_{S}(x,y) & \equiv & S(y,x) \\
\textbf{copy 1 to copy 2:} & \phi^{\mathbf{1}\to\mathbf{2}}_{S}(x,y) & \equiv & x = y \wedge last(x) \\
\textbf{copy 2 to copy 1:} & \phi^{\mathbf{2}\to\mathbf{1}}_{S}(x,y) & \equiv & \bot \\
\textbf{for all copies } i\textbf{:} & \phi^{\mathbf{i}}_{\sigma}(x) & \equiv & \sigma(x)
\end{array}
$$

Büchi Theorem for Regular Transductions

Let $f : \Sigma^* \hookrightarrow \Sigma^*$.

Theorem (Engelfriet, Hoogeboom, 01)

The following are equivalent:

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

Büchi Theorem for Regular Transductions

Let $f : \Sigma^* \hookrightarrow \Sigma^*$.

Theorem (Engelfriet, Hoogeboom, 01)

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, and they are closed under composition.

Büchi Theorem for Regular Transductions

Let $f : \Sigma^* \hookrightarrow \Sigma^*$.

Theorem (Engelfriet, Hoogeboom, 01)

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, and they are closed under composition.

Proof ideas: MSO-transducers are 2-way transducers with MSO jumps $\phi_S^{c \to c'}(x, y)$

- turn jumps into walks
- hold enough information to decide MSO-formulas locally: states = MSO-types

$f = \hat{f} \circ f_{types}$ (use composition closure of 2-way trans)

Some other (recent) results

Other specification languages

- FO-transducers
  - equivalent to aperiodic transducers with registers (F., Krishna, Trivedi, 14)
  - and to aperiodic 2-way transducers (Dartois, Jecker, Reynier, 16)

Other specification languages

- FO-transducers
  - equivalent to aperiodic transducers with registers (F., Krishna, Trivedi, 14)
  - and to aperiodic 2-way transducers (Dartois, Jecker, Reynier, 16)
- regular expressions for regular functions (called combinators)
  - iterated sum $f^*(u) = f(u_1)f(u_2)\dots f(u_n)$ for $u = u_1 \dots u_n$
  - chain sum $f^c(u) = f(u_1u_2)f(u_2u_3)\dots f(u_{n-1}u_n)$
  - introduced by Alur, Freilich, Raghothaman in 14
  - extended to infinite words by Dave, Gastin, Krishna in 18

Other specification languages

- FO-transducers
  - equivalent to aperiodic transducers with registers (F., Krishna, Trivedi, 14)
  - and to aperiodic 2-way transducers (Dartois, Jecker, Reynier, 16)
- regular expressions for regular functions (called combinators)
  - iterated sum $f^*(u) = f(u_1)f(u_2)\dots f(u_n)$ for $u = u_1 \dots u_n$
  - chain sum $f^c(u) = f(u_1u_2)f(u_2u_3)\dots f(u_{n-1}u_n)$
  - introduced by Alur, Freilich, Raghothaman in 14
  - extended to infinite words by Dave, Gastin, Krishna in 18
- new FO-equivalent expressions based on function composition $\circ$ by Bojanczyk, Daviaud and Krishna in 18

Other specification languages

- FO-transducers
  - equivalent to aperiodic transducers with registers (F., Krishna, Trivedi, 14)
  - and to aperiodic 2-way transducers (Dartois, Jecker, Reynier, 16)
- regular expressions for regular functions (called combinators)
  - iterated sum $f^*(u) = f(u_1)f(u_2)\dots f(u_n)$ for $u = u_1 \dots u_n$
  - chain sum $f^c(u) = f(u_1u_2)f(u_2u_3)\dots f(u_{n-1}u_n)$
  - introduced by Alur, Freilich, Raghothaman in 14
  - extended to infinite words by Dave, Gastin, Krishna in 18
- new FO-equivalent expressions based on function composition $\circ$ by Bojanczyk, Daviaud and Krishna in 18
- an expressive decidable logic tailored to (non-functional) transductions Dartois, F., Lhote, 18.

Definability Problems

Definition

$\mathcal{F}$: logical fragment of MSOT (e.g. FOT)

Input: T an MSOT

Output: Is $[\![T]\!]$ FO-definable ?

Definability Problems

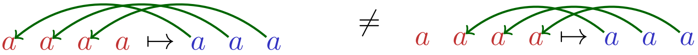
Definition

$\mathcal{F}$: logical fragment of MSOT (e.g. FOT)
Input: T an MSOT
Output: Is $[\![T]\!]$ FO-definable ?

Results

- Decidable for “rational” MSO (=rational functions)
  F., Gauwin, Lhote, 16
- Open for MSOT

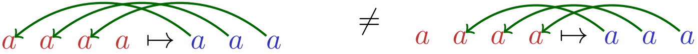
Origin semantics (Bojanczyk, 14)



$a\ a\ a\ a \mapsto a\ a\ a \quad \neq \quad a\ a\ a\ a \mapsto a\ a\ a$

Origin semantics $[\![T]\!]_o$ inherent to most transducer models T !

Origin semantics (Bojanczyk, 14)

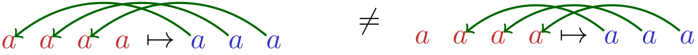


$a\ a\ a\ a \mapsto a\ a\ a \neq a\ a\ a\ a \mapsto a\ a\ a$

Origin semantics $[\![T]\!]_o$ inherent to most transducer models T !

- existence of a canonical transducer if *origin* is taken into account (Bojanczyk, ICALP'14)
- decidable FO-definability of MSOT *with* origin

Origin semantics (Bojanczyk, 14)

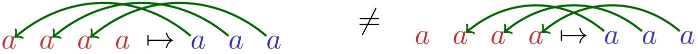


$a\ a\ a\ a \mapsto a\ a\ a \quad \neq \quad a\ a\ a\ a \mapsto a\ a\ a$

Origin semantics $[\![T]\!]_o$ inherent to most transducer models T !

- existence of a canonical transducer if *origin* is taken into account (Bojanczyk, ICALP'14)
- decidable FO-definability of MSOT *with* origin
- algorithmic problems modulo origin ($[\![T_1]\!]_o = [\![T_2]\!]_o$)
- extended to "similar" origins through resynchronisers (F., Maneth, Reynier, Talbot, 15) (F., Jecker, Löding, Winter,16), (Bose, Muscholl, Penell, Puppis, 18)

Origin semantics (Bojanczyk, 14)



Origin semantics $[\![T]\!]_o$ inherent to most transducer models T !

- existence of a canonical transducer if *origin* is taken into account (Bojanczyk, ICALP'14)
- decidable FO-definability of MSOT *with* origin
- algorithmic problems modulo origin ($[\![T_1]\!]_o = [\![T_2]\!]_o$)
- extended to "similar" origins through resynchronisers (F., Maneth, Reynier, Talbot, 15) (F., Jecker, Löding, Winter,16), (Bose, Muscholl, Penell, Puppis, 18)
- study of rational relation subclasses by control languages $REL(C)$, $C \subseteq \{\mathrm{in}, \mathrm{out}\}^*$ (Descotte, Figueira, Libkin, Puppis)

Some Other Results

- machine-independent characterisations

  Cadilhac,Krebs,Ludwig,Paperman, 15

Some Other Results

- machine-independent characterisations

  Cadilhac,Krebs,Ludwig,Paperman, 15
- uniformisation problems (Ismaël's Jecker and Sarah Winter's PhD thesis). E.g. given R rational, is there f sequential such that
  - $G_f \subseteq R$
  - $dom(f) = dom(R)$

Some Other Results

- machine-independent characterisations

  Cadilhac,Krebs,Ludwig,Paperman, 15
- uniformisation problems (Ismaël's Jecker and Sarah Winter's PhD thesis). E.g. given R rational, is there f sequential such that
  - $G_f \subseteq R$
  - $dom(f) = dom(R)$
- ressource analysis (streaming, number of registers, etc.)

  Baschenis, Daviaud, F., Gauwin, Muscholl, Puppis, Reynier, Talbot...

Some Other Results

- machine-independent characterisations

  Cadilhac,Krebs,Ludwig,Paperman, 15

- uniformisation problems (Ismaël's Jecker and Sarah Winter's PhD thesis). E.g. given R rational, is there f sequential such that
  - $G_f \subseteq R$
  - $dom(f) = dom(R)$

- ressource analysis (streaming, number of registers, etc.)

  Baschenis, Daviaud, F., Gauwin, Muscholl, Puppis, Reynier, Talbot...

- learning

  Boiret, Lemay, Niehren 12

Some Other Results

- machine-independent characterisations

  Cadilhac,Krebs,Ludwig,Paperman, 15

- uniformisation problems (Ismaël's Jecker and Sarah Winter's PhD thesis). E.g. given R rational, is there f sequential such that
  - $G_f \subseteq R$
  - $dom(f) = dom(R)$

- ressource analysis (streaming, number of registers, etc.)

  Baschenis, Daviaud, F., Gauwin, Muscholl, Puppis, Reynier, Talbot...

- learning

  Boiret, Lemay, Niehren 12

- data word transducers Léo Exibard's PhD thesis

Some Other Results

- machine-independent characterisations

  Cadilhac,Krebs,Ludwig,Paperman, 15
- uniformisation problems (Ismaël's Jecker and Sarah Winter's PhD thesis). E.g. given R rational, is there f sequential such that
  - $G_f \subseteq R$
  - $dom(f) = dom(R)$
- ressource analysis (streaming, number of registers, etc.)

  Baschenis, Daviaud, F., Gauwin, Muscholl, Puppis, Reynier, Talbot...
- learning

  Boiret, Lemay, Niehren 12
- data word transducers Léo Exibard's PhD thesis
- other structures: infinite strings, nested words, trees, graphs, data words ...

A Few Applications

- language and speech processing (M. Mohri)
- regular model-checking
- text analysis, document transformation
- reactive synthesis
- **Tools**: OpenFST, Vaucanson, DreX (Alur, d'Antoni, Raghothaman)
- line of works on symbolic transducers (d'Antoni, Veanes ...)

A Few Applications

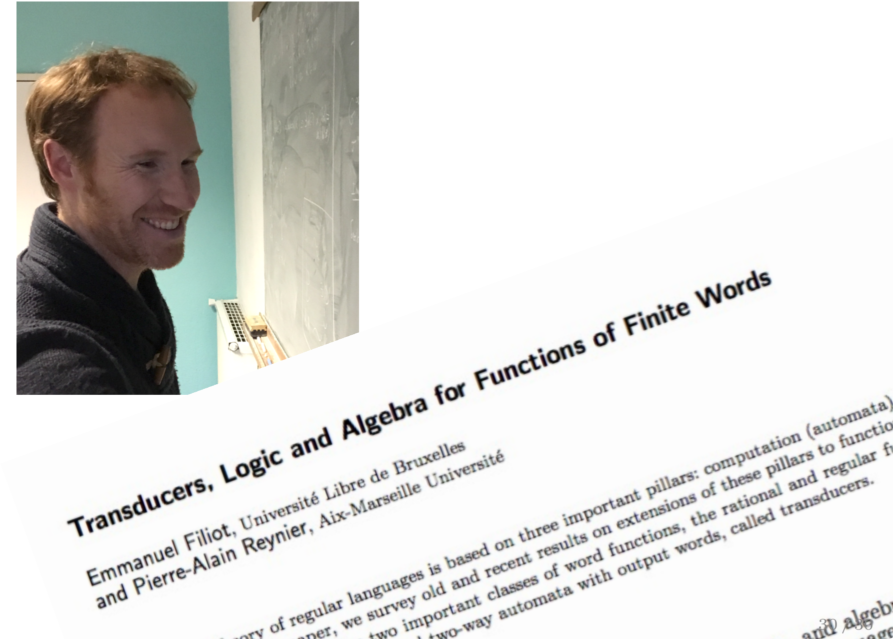
- language and speech processing (M. Mohri)
- regular model-checking
- text analysis, document transformation
- reactive synthesis
- **Tools**: OpenFST, Vaucanson, DreX (Alur, d'Antoni, Raghothaman)
- line of works on symbolic transducers (d'Antoni, Veanes ...)

But **so far**, no strong application of the "asynchronous" setting (non letter-to-letter)

People

From left-to-right top-to-bottom: Nathan Lhote, Ismaël Jecker, Luc Dartois, Olivier Gauwin, Anca Muscholl, Frédéric Servais, Ashutosh Trivedi, Léo Exibard, Jean-Marc Talbot, Krishna S., Nicolas Mazzocchi, Christof Löding, Sarah Winter.

SIGLOG News 9th



Transducers, Logic and Algebra for Functions of Finite Words

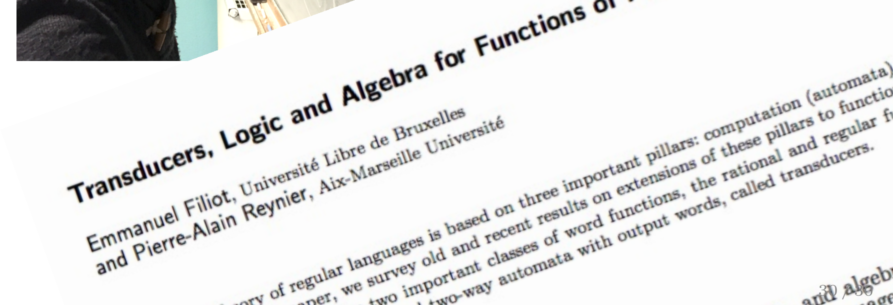
Emmanuel Filiot, Université Libre de Bruxelles
and Pierre-Alain Reynier, Aix-Marseille Université

ory of regular languages is based on three important pillars: computation (automata)
aper, we survey old and recent results on extensions of these pillars to functio
two important classes of word functions, the rational and regular f
two-way automata with output words, called transducers.

and algeb

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

ory of regular languages is based on three important pillars: computation (automata)
aper, we survey old and recent results on extensions of these pillars to functio
two important classes of word functions, the rational and regular f
l two-way automata with output words, called transducers.

and algeb