SPECIFICATION AND SYNTHESIS OF FINITE WORD TRANSDUCTIONS

EMMANUEL FILIOT — UNIVERSITE LIBRE DE BRUXELLES

JOINT WITH

LUC DARTOIS — ULB

NATHAN LHOTE — ULB / LABRI

FUNCTIONAL TRANSDUCTIONS

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

* Erase all b's $a\,b\,a\,a\,b\,a \mapsto a\,a\,a\,a$

FUNCTIONAL TRANSDUCTIONS

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

* Erase all b's $\quad abaaba \mapsto aaaa$
* Count the length $\quad abaaba \mapsto a^6$

FUNCTIONAL TRANSDUCTIONS

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

* Erase all b's $\quad a\,b\,a\,a\,b\,a \longmapsto a\,a\,a\,a$
* Count the length $\quad a\,b\,a\,a\,b\,a \longmapsto a^6$
* Mirror $\quad a\,b\,a\,a\,b \longmapsto b\,a\,a\,b\,a$

FUNCTIONAL TRANSDUCTIONS

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

* Erase all b's $\quad a\,b\,a\,a\,b\,a \longmapsto a\,a\,a\,a$
* Count the length $\quad a\,b\,a\,a\,b\,a \longmapsto a^6$
* Mirror $\quad a\,b\,a\,a\,b \longmapsto b\,a\,a\,b\,a$
* copy $\quad u \longmapsto uu$

FUNCTIONAL TRANSDUCTIONS

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

* Erase all b's $\quad a\,b\,a\,a\,b\,a \longmapsto a\,a\,a\,a$
* Count the length $\quad a\,b\,a\,a\,b\,a \longmapsto a^6$
* Mirror $\quad a\,b\,a\,a\,b \longmapsto b\,a\,a\,b\,a$
* copy $\quad u \longmapsto uu$
* Pass a token to the right $\quad N^n T N^m \longmapsto N^{n+1} T N^{m-1}$ (regular model-checking)

FUNCTIONAL TRANSDUCTIONS

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

* Erase all b's $\quad a\,b\,a\,a\,b\,a \mapsto a\,a\,a\,a$
* Count the length $\quad a\,b\,a\,a\,b\,a \mapsto a^6$
* Mirror $\quad a\,b\,a\,a\,b \mapsto b\,a\,a\,b\,a$
* copy $\quad u \mapsto uu$
* Pass a token to the right $\quad N^n T N^m \mapsto N^{n+1} T N^{m-1}$ (regular model-checking)
* Reactive systems : grant every request

$r \quad g \quad \neg r \quad \neg g \quad r \quad g \quad \neg r \quad \neg g \quad \neg r \quad \neg g$

$r \quad \neg r \quad r \quad \neg r \quad \neg r \mapsto g \quad \neg g \quad g \quad \neg g \quad \neg g$

TRANSDUCTIONS

BINARY RELATIONS OF FINITE WORDS

$$R \subseteq \Sigma^* \times \Sigma^*$$

* subword relation

$$\{ (oxford, ford), (oxford, \varepsilon), (oxford, xor), \dots \}$$

TRANSDUCTIONS

BINARY RELATIONS OF FINITE WORDS

$$R \subseteq \Sigma^* \times \Sigma^*$$

* subword relation

$$\{ (oxford, ford), (oxford, \varepsilon), (oxford, xor), \ldots \}$$

* shuffle

$$\{ (oxford, rdfoxo), (oxford, oxfdro), \ldots \}$$

TRANSDUCTIONS

BINARY RELATIONS OF FINITE WORDS

$$R \subseteq \Sigma^* \times \Sigma^*$$

* subword relation

$$\{ (oxford, ford), (oxford, \varepsilon), (oxford, xor), \dots \}$$

* shuffle

$$\{ (oxford, rdfoxo), (oxford, oxfdro), \dots \}$$

NOTATION

$$dom(R) = \{ u \mid \exists (u,v) \in R \}$$

OBJECTIVES

MODEL-CHECKING :

IMPLEMENTATION $\models$ SPECIFICATION ?

$\uparrow$ A <u>functional</u> transduction $f : \Sigma^* \hookrightarrow \Sigma^*$

$\uparrow$ A transduction $R \subseteq \Sigma^* \times \Sigma^*$

$f \models R$ if $\forall u \in dom(f)$, $(u, f(u)) \in R$

OBJECTIVES

MODEL-CHECKING :

IMPLEMENTATION $\models$ SPECIFICATION ?

A functional transduction

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

A transduction

$R \subseteq \Sigma^* \times \Sigma^*$

$f \models R$ if $\forall u \in dom(f),\ (u, f(u)) \in R$

SYNTHESIS :

? $\models$ SPECIFICATION

$\exists f \cdot f \models R \wedge dom(f) = dom(R)$?

EXAMPLES OF SPECIFICATIONS

- True : $\Sigma^* \times \Sigma^*$

EXAMPLES OF SPECIFICATIONS

- True : $\Sigma^* \times \Sigma^*$

- There exists an 'a' in the output $\Sigma^* \times \Sigma^* a \Sigma^*$

EXAMPLES OF SPECIFICATIONS

- True : $\Sigma^* \times \Sigma^*$
- There exists an 'a' in the output $\Sigma^* \times \Sigma^*_a \Sigma^*$
- The lenght is preserved $\{(u,v) : |u| = |v|\}$

EXAMPLES OF SPECIFICATIONS

- True : $\Sigma^*_{\times}\,\Sigma^*$
- There exists an 'a' in the output $\Sigma^*_{\times}\,\Sigma^*_{a}\,\Sigma^*$
- The lenght is preserved $\{(u,v) : |u| = |v|\}$
- $G(r \to F g)$ $\{(\sigma_1 \ldots \sigma_k, \beta_1 \ldots \beta_k) \mid \forall i\ \sigma_i = r \to \exists j \geq i\ \beta_j = g\}$

EXAMPLES OF SPECIFICATIONS

- True : $\Sigma^* \times \Sigma^*$

- There exists an 'a' in the output $\Sigma^* \times \Sigma^* a \Sigma^*$

- The lenght is preserved $\{ (u, v) : |u| = |v| \}$

- $G(r \to F g)$ $\{ (\sigma_1 \dots \sigma_k , \beta_1 \dots \beta_k) \mid \forall i \; \sigma_i = r \to \exists j \geq i \; \beta_j = g \}$

- Every task occurrence is scheduled exactly once

  Σ = finite set of tasks

$$R = \{ (t_1 \dots t_n , t_{\pi(1)} \dots t_{\pi(n)}) : \pi \text{ permutation} \}$$

OUTLINE

WHAT KIND OF MACHINES TO USE AS IMPLEMENTATIONS ?

WHAT LANGUAGE TO EXPRESS SPECIFICATIONS ?

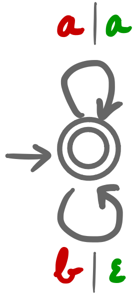
AUTOMATA FOR TRANSDUCTIONS

(TRANSDUCERS)

RATIONAL TRANSDUCTIONS

extend finite automata with outputs

* erase all b's



* move the token to the right

$$N^n T N^m \mapsto N^{n+1} T N^{m-1}$$

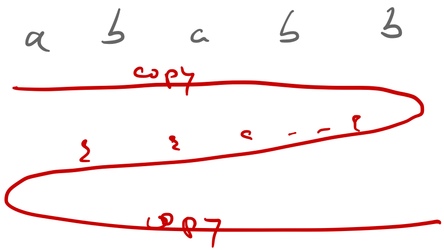
REGULAR TRANSDUCTIONS

extend 2-way automata with outputs

MANY MODELS :
- 2-way deterministic transducers
- Courcelle's MSO transducers
- streaming string transducers [Alur, Cerny, 10]
- regular combinators [Alur, Freilich, Raghothaman, 14]
- 2-way reversible transducers [Dartois, Fournier, Jecker, Lhote, 17]
- ...

TYPICAL EXAMPLES mirror , copy

TWO-WAY DETERMINISTIC TRANSDUCERS (2DFT)



REGULAR TRANSDUCTIONS

COURCELLE'S MSO TRANSDUCERS

WORD SIGNATURE $\{ S, a(x), b(x), \ldots \}$

MAIN IDEA : define output predicates by MSO formulas over input

EXAMPLE :

$$a \xrightarrow{S} b \xrightarrow{S} a \xrightarrow{S} a \xrightarrow{S} a$$

REGULAR TRANSDUCTIONS

COURCELLE'S MSO TRANSDUCERS WORD SIGNATURE $\{S, a(x), b(x), \dots\}$

MAIN IDEA : define output predicates by MSO formulas over input

EXAMPLE :

$$a \xrightarrow{S} b \xrightarrow{S} a \xrightarrow{S} a \xrightarrow{S} a$$

THEOREM [Engelfriet, Hoogeboom, 99] MSO TRANSDUCERS = 2DFT

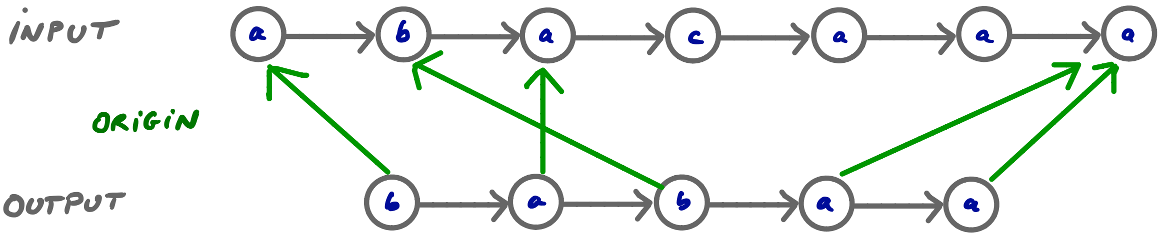
PROPERTIES

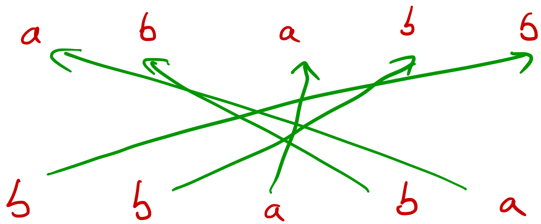
* Decidable equivalence problem (Pspace-c for 2DFT)
* Closed under composition

A LOGIC TO EXPRESS PROPERTIES OF TRANSDUCTIONS

TOWARDS A LOGIC FOR TRANSDUCTIONS

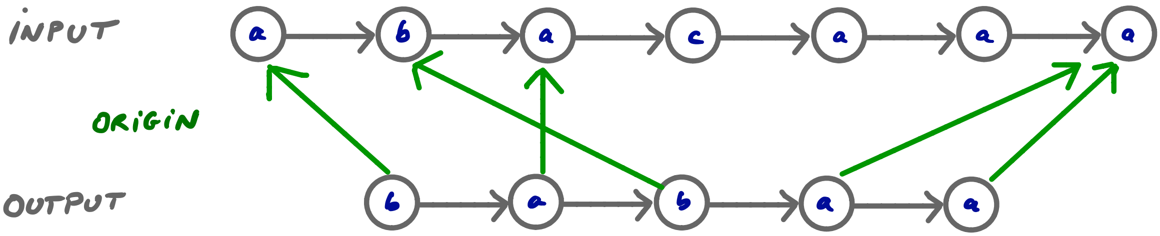
IDEA : SEE TRANSDUCTIONS AS SINGLE STRUCTURES WITH ORIGIN
(called *origin graphs*)





TOWARDS A LOGIC FOR TRANSDUCTIONS

IDEA : SEE TRANSDUCTIONS AS SINGLE STRUCTURES WITH ORIGIN
(called *origin graphs*)

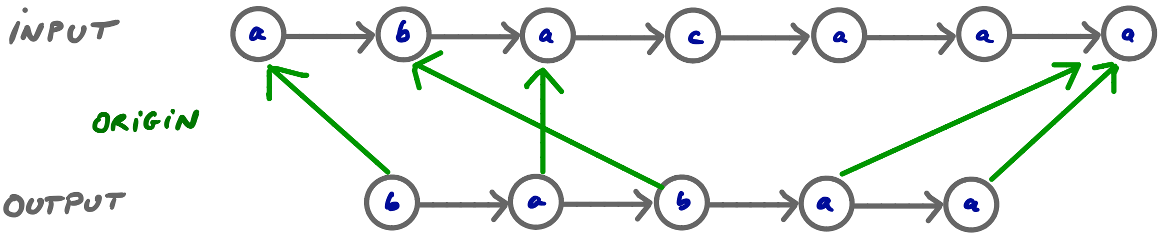


$$\mathrm{MSO}[\leq_{in}, \leq_{out}, o]$$

input order — output order — origin function

TOWARDS A LOGIC FOR TRANSDUCTIONS

IDEA : SEE TRANSDUCTIONS AS SINGLE STRUCTURES WITH ORIGIN (called origin graphs)



$$\mathrm{MSO}[\leq_{in}, \leq_{out}, o]$$

input order — output order — origin function

Characterizations of classes of origin graphs [Bojańczyk, Daviaud, Guillon, Penelle '17]

EXAMPLES

- True : T

EXAMPLES

$\exists^{out} x \qquad \exists x \,.\, x \leq_{out} x$

- True : T

- Output contains 'a' : $\exists^{out} x.\ a(x)$

EXAMPLES

- True : $\top$
- Output contains 'a' : $\exists^{out} x.\ a(x)$
- Lenght is preserved : "σ is bijective"

injectivity $\quad \forall^{out}_x \forall^{out}_y \ x \neq y \rightarrow \sigma(x) \neq \sigma(y)$

surjectivity $\quad$ always surjective by definition of origin-graphs

EXAMPLES

- True : $\top$

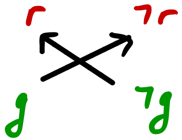
- Output contains 'a' : $\exists^{out} x.\ a(x)$

- Lenght is preserved : "σ is bijective"

- $G(r \to F g)$: $\forall^{out} x.\ r(\sigma(x)) \longrightarrow \exists^{out} y \geq_{out} x,\ g(y)$

EXAMPLES

- True : $\top$
- Output contains 'a' : $\exists^{out} x .\; a(x)$
- Lenght is preserved : "σ is bijective"
- $G(r \rightarrow F g)$: $\forall^{out} x .\; r(\sigma(x)) \longrightarrow \exists^{out} y \geq_{out} x ,\; g(y)$ $\not\equiv$



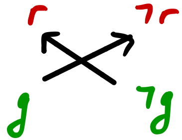
EXAMPLES

- True : $\top$

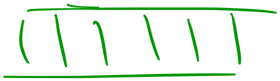
- Output contains 'a' : $\exists^{out} x.\ a(x)$

- Lenght is preserved : "σ is bijective"

- $G(r \to F g)$: $\forall^{out}_x.\ r(\sigma(x)) \to \exists^{out}_y \geq_{out} x,\ g(y)$ $\not\equiv$



ADD "σ is bijective" and "order-preserving"



$$\forall^{out}_x \forall^{out}_y\ x \leq_{out} y \to \sigma(x) \leq_{in} \sigma(y)$$

EXAMPLES

- True : $\top$

- Output contains 'a' : $\exists^{out} x.\ a(x)$

- Lenght is preserved : "θ is bijective"

- $G(r \to F g)$

- Every task is scheduled exactly once (shuffle)

  "θ is bijective" and "θ is label-preserving"

$$\forall^{\cancel{in}\ out} x.\ \bigwedge_{\sigma \in \Sigma} \sigma(x) \to \sigma(\theta(x))$$

EXAMPLES

- True : $\top$
- Output contains 'a' : $\exists^{out} x.\ a(x)$
- lenght is preserved : "σ is bijective"
- $G(r \to F g)$
- Every task is scheduled exactly once (shuffle)
- identity "σ is bijective" and "σ is order-preserving"
  and "σ is label-preserving"

EXAMPLES

- True : $\top$
- Output contains 'a' : $\exists^{out} x.\ a(x)$
- lenght is preserved : "o is bijective"
- $G(r \to Fg)$
- Every task is scheduled exactly once (shuffle)
- identity "o is bijective" and "o is order-preserving"
  and "o is label-preserving"
- $u \mapsto uu$?

MODEL-CHECKING

$T \models \varphi$ is decidable for $T \in$ 2DFT and $\varphi \in \mathrm{MSO}[\leq_{in}, \leq_{out}, \theta]$

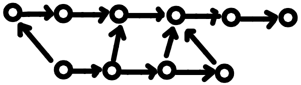
Bojańczyk, Daviaud, Guillon, Penelle '17

MODEL-CHECKING

$T \models \varphi$ is decidable for $T \in$ 2DFT and $\varphi \in MSO[\leq_{in}, \leq_{out}, \theta]$

Bojańczyk, Daviaud, Guillon, Penelle '17

WORD SET OF ORIGIN GRAPHS

$u \in \Sigma^* \xrightarrow{\alpha} \Big\{$  $\sigma \begin{smallmatrix} u \\ v \end{smallmatrix} \;\Big|\; (u, v, \sigma) \in [\![T]\!] \Big\}$

① $\alpha \in$ Courcelle's word-to-graph transductions

② Backward Translation Theorem*: $\alpha^{-1}([\![\neg\varphi]\!]) \in REG(\Sigma)$

③ Check $\alpha^{-1}([\![\neg\varphi]\!]) = \emptyset$

* $T^{-1}(MSO) \in MSO$

SATISFIABILITY

Input : $\varphi \in MSO[\leq_{in}, \leq_{out}, \sigma]$ **Output** : $\exists G \cdot G \models \varphi$?

RESULTS

- Undecidable (even for $FO_2[\leq_{out}, S_{out}, \leq_{in}, \sigma]$)
- Decidable for $\mathcal{L}_T := FO_2[\leq_{out}, \sigma, MSO_{bin}[\leq_{in}]]$

Example : $\forall^{out}_x \exists^{out}_y \; P(\sigma(x), \sigma(y)) \to \forall^{out}_x \; y \geq_{out} x \to Q(\sigma(y), \sigma(x))$

$P, Q \in MSO[\leq_{in}]$

SATISFIABILITY *Input*: $\varphi \in MSO[\leq_{in}, \leq_{out}, \sigma]$ *Output*: $\exists G \cdot G \models \varphi$?

RESULTS

- Undecidable (even for $FO_2[\leq_{out}, S_{out}, \leq_{in}, \sigma]$)
- Decidable for $\mathcal{L}_T := FO_2[\leq_{out}, \sigma, MSO_{bin}[\leq_{in}]]$

Example : $\forall^{out}_x \exists^{out}_y \; P(\sigma(x), \sigma(y)) \rightarrow \forall^{out}_x \; y \geq_{out} x \rightarrow Q(\sigma(y), \sigma(x))$

$P, Q \in MSO[\leq_{in}]$

$\varphi_1, \varphi_2 \qquad \neg(\varphi_1 \Leftrightarrow \varphi_2)$

CONSEQUENCES

- Decidable equivalence problem for $\mathcal{L}_T$-transductions
- Decidable satisfiability for $\exists \mathcal{L}_T$ $(\exists X_1 \cdots \exists X_n \cdot \mathcal{L}_T)$

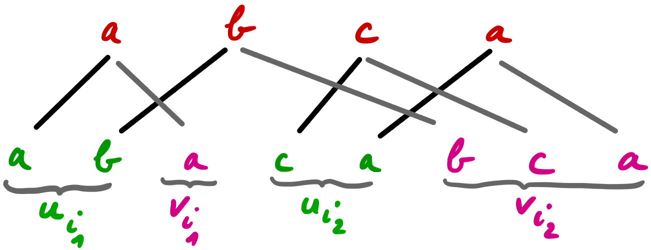
$\exists X_1 \cdots \exists X_n \cdot \phi(X_1, \cdots, X_n)$

UNDECIDABILITY OF $MSO[\leq_{in}, \leq_{out}, o]$ OVER ORIGIN GRAPHS

PCP instance $(u_1, v_1), \dots, (u_n, v_n)$

Does $u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}$ for some $i_1, \dots, i_k$?

ENCODING A SOLUTION AS AN ORIGIN GRAPH

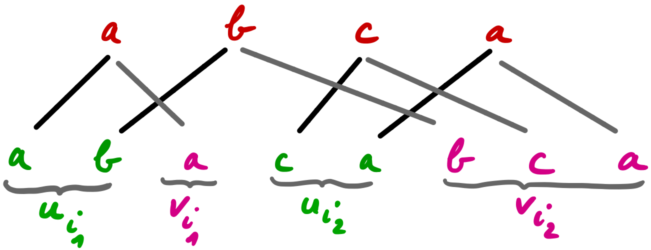


UNDECIDABILITY OF $MSO[\leq_{in}, \leq_{out}, o]$ OVER ORIGIN GRAPHS

PCP instance $(u_1, v_1), \dots, (u_n, v_n)$

Does $u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}$ for some $i_1, \dots, i_k$?

ENCODING A SOLUTION AS AN ORIGIN GRAPH



1 CHECK THAT THE GREEN AND PINK PROJECTIONS EQUAL THE INPUT

Express that the origin graph restricted to green (resp. pink) outputs has bijective, order-preserving and label-preserving origin

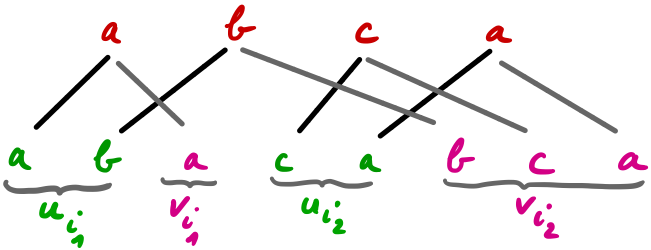
$\longrightarrow$ guard the quantifications : $\forall^{out}_x \phi \rightsquigarrow \forall^{out}_x \, green(x) \to \phi$

UNDECIDABILITY OF MSO[$\leq_{in}, \leq_{out}, o$] OVER ORIGIN GRAPHS

PCP instance $(u_1, v_1), \dots, (u_n, v_n)$

Does $u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}$ for some $i_1, \dots, i_k$?

ENCODING A SOLUTION AS AN ORIGIN GRAPH



② CHECK INDICES CONSISTENCY

$$\forall X,Y \,.\, \left[\text{maxgreen}(X) \wedge \text{maxpink}(Y) \wedge \text{succ}(X,Y)\right] \rightarrow \left[\bigvee_{i=1}^{n} u_i(X) \wedge v_i(Y)\right]$$

$\text{maxgreen}(X) \equiv$ "X is maximal block of consecutive green symbols"

$\text{succ}(X,Y) \equiv$ "X and Y are consecutive blocks"

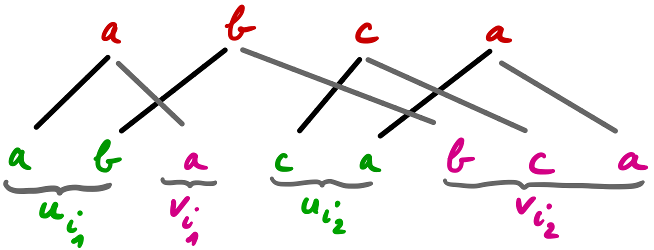
$u(X) \equiv$ "the positions of X, ordered by $\leq_{out}$, form the word u"

UNDECIDABILITY OF $MSO[\leq_{in}, \leq_{out}, \sigma]$ OVER ORIGIN GRAPHS

PCP instance $(u_1, v_1), \dots, (u_n, v_n)$

Does $u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}$ for some $i_1, \dots, i_k$?

ENCODING A SOLUTION AS AN ORIGIN GRAPH



② CHECK INDICES CONSISTENCY

$$\forall X, Y \cdot \left[\mathrm{max\,green}(X) \wedge \mathrm{max\,pink}(Y) \wedge \mathrm{succ}(X,Y) \right] \rightarrow \left[\bigvee_{i=1}^{n} u_i(X) \wedge v_i(Y) \right]$$

⇝ can be done with only two first-order variables

(identify consecutive green/pink blocks by their first position)

$\mathcal{L}_T := FO_2[\leq_{out}, \theta, MSO_{bin}[\leq_{in}]]$

EXPRESSIVENESS OF $\mathcal{L}_T$

- ALL PREVIOUS EXAMPLES (SHUFFLE INCLUDED)
- WHAT ABOUT $u \mapsto uu$?

$\mathcal{L}_T := FO_2[\leq_{out}, \sigma, MSO_{bin}[\leq_{in}]]$

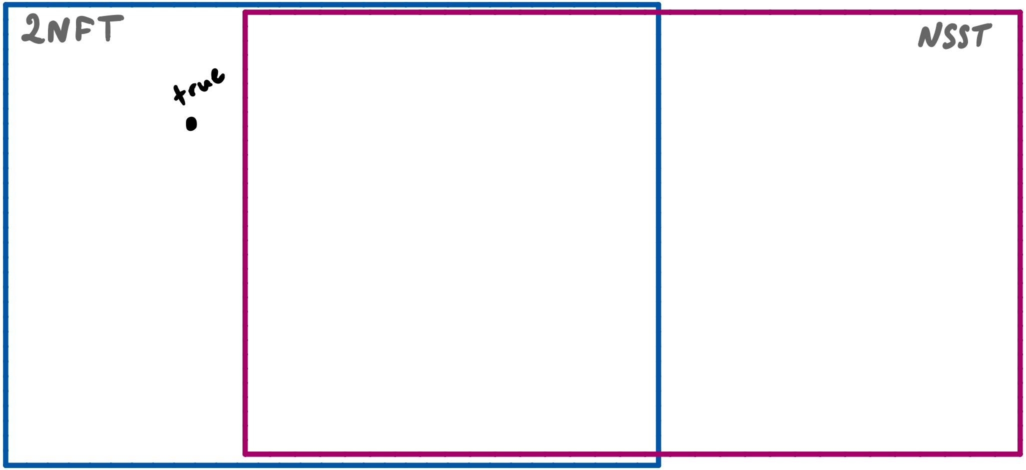
EXPRESSIVENESS OF $\mathcal{L}_T$

- ALL PREVIOUS EXAMPLES (SHUFFLE INCLUDED)
- $\mathcal{L}_T$ > REG

$\mathcal{L}_T := FO_2[\leq_{out}, \sigma, MSO_{bin}[\leq_{in}]]$

EXPRESSIVENESS OF $\mathcal{L}_T$

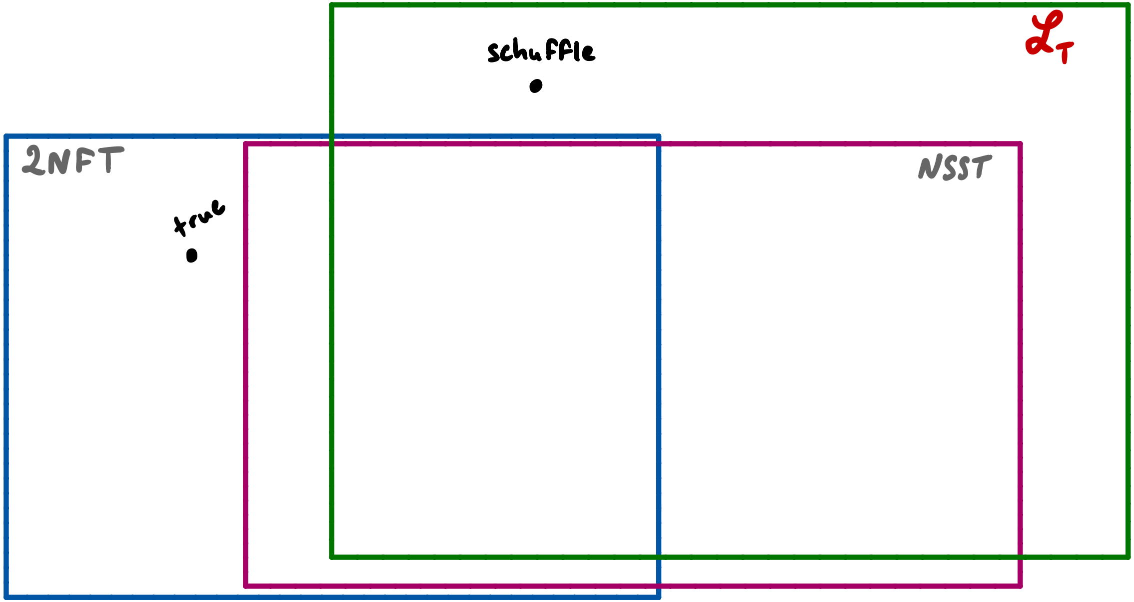
- ALL PREVIOUS EXAMPLES (SHUFFLE INCLUDED)
- $\mathcal{L}_T$ > REG



EXPRESSIVENESS OF $\mathcal{L}_T$

$\mathcal{L}_T := FO_2[\leq_{out}, \sigma, MSO_{bin}[\leq_{in}]]$

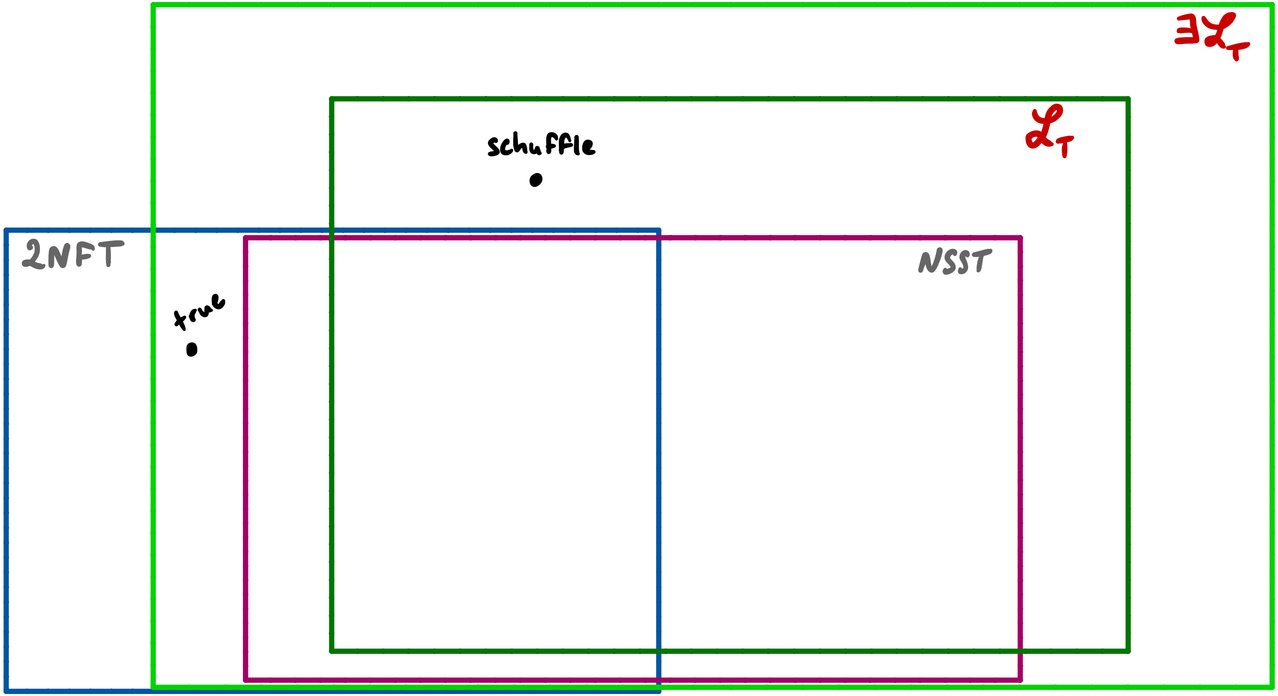
- ALL PREVIOUS EXAMPLES (SHUFFLE INCLUDED)
- $\mathcal{L}_T > REG$



EXPRESSIVENESS OF $\mathcal{L}_T$

$\mathcal{L}_T := FO_2[\leq_{out}, \sigma, MSO_{bin}[\leq_{in}]]$

- ALL PREVIOUS EXAMPLES (SHUFFLE INCLUDED)
- $\mathcal{L}_T > REG$



SYNTHESIS

THEOREM

$$\forall \varphi \in \exists \mathcal{L}_T \cdot \exists f : \Sigma^* \to \Sigma^* \cdot \begin{cases} 1. & f \in REG \\ 2. & \mathrm{dom}\, f = \mathrm{dom}\, \varphi \\ 3. & f \models \varphi \end{cases}$$

SYNTHESIS

THEOREM

$$\forall \varphi \in \exists \mathcal{L}_T \,.\, \exists f : \Sigma^* \to \Sigma^* \,.\, \begin{cases} 1. & f \in REG \\ 2. & \operatorname{dom} f = \operatorname{dom} \varphi \\ 3. & f \models \varphi \end{cases}$$

CONSEQUENCES

- New characterization of REG : $REG = \mathcal{L}_T \cap \text{functions}$
- Decidable functionality problem for $\mathcal{L}_T$

1. $\varphi : \mathcal{L}_T \xrightarrow{\text{synthesis}} \psi : \text{2DFT} \xrightarrow{\text{expressiveness}} \varphi' : \mathcal{L}_T \cap \text{functions}$
2. Test $\varphi \equiv \varphi'$

DATA WORDS (OVER Σ, D)

DEFINITION

$$(\sigma_1, d_1) \dots (\sigma_n, d_n) \in (\Sigma \times D)^*$$

infinite set of ordered data

finite alphabet

DATA WORDS (OVER $\Sigma, \mathbb{D}$)

DEFINITION

$(\sigma_1, d_1) \dots (\sigma_n, d_n) \in (\Sigma \times \mathbb{D})^*$

infinite set of ordered data

finite alphabet

DATA WORD AS FINITE STRUCTURE

$\underbrace{\text{finite word structure}}$ + $\underbrace{\text{data comparison}}$

$\{1, \dots, n\}$: positions

$\leq$: linear_order

$\preccurlyeq$: preorder on positions

DATA WORDS (OVER Σ, D)

DEFINITION

$$(\sigma_1, d_1) \ldots (\sigma_n, d_n) \in (\Sigma \times D)^*$$

infinite set of <u>ordered</u> data

Finite alphabet

DATA WORD AS FINITE STRUCTURE

$\underbrace{\text{finite word structure}}$ + $\underbrace{\text{data comparison}}$

$\{1, \ldots, n\}$: positions

$\lesssim$: preorder on positions

$\leqslant$: linear order

LOGICS FOR DATA WORDS

$FO_2[\leqslant, S_p, \sim_d]$: decidable

$FO_2[\leqslant, S_d, \lesssim]$: decidable

$FO_2[\leqslant, S_p, \lesssim]$: undecidable [Schwentick, Zeume, '10]

[Bojańczyk Muscholl Schwentick Segoufin David '06]

TYPED DATA WORDS (over Σ, D, T)

DEFINITION

$(w, \tau : D \rightarrow T)$

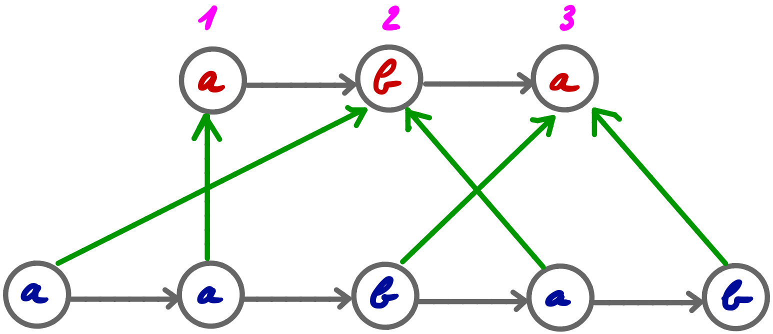
data word

finite set of types

closed if $data(w) = \downarrow data(w)$

$$\begin{pmatrix} d_1 \\ t_1 \\ \sigma_1 \end{pmatrix} \begin{pmatrix} d_2 \\ t_2 \\ \sigma_2 \end{pmatrix} \begin{pmatrix} d_3 \\ t_3 \\ \sigma_3 \end{pmatrix} \begin{pmatrix} d_4 \\ t_4 \\ \sigma_4 \end{pmatrix} \begin{pmatrix} d_5 \\ t_5 \\ \sigma_5 \end{pmatrix} \begin{pmatrix} d_6 \\ t_6 \\ \sigma_6 \end{pmatrix} \begin{matrix} \cdots \\ \cdots \\ \cdots \end{matrix} \begin{pmatrix} d_n \\ t_n \\ \sigma_n \end{pmatrix}$$

TYPED DATA WORDS vs TRANSDUCTIONS



data = origin !

$$\begin{pmatrix} 2 \\ b \\ a \end{pmatrix} \quad \begin{pmatrix} 1 \\ a \\ a \end{pmatrix} \quad \begin{pmatrix} 3 \\ a \\ b \end{pmatrix} \quad \begin{pmatrix} 2 \\ b \\ a \end{pmatrix} \quad \begin{pmatrix} 3 \\ a \\ b \end{pmatrix}$$

MODELS

NON-ERASING TRANSDUCTIONS WITH ORIGIN OVER Σ, Γ	$\longleftrightarrow$	CLOSED TYPED DATA WORDS OVER $\Sigma, \mathbb{N}, \Gamma$

MODELS

NON-ERASING TRANSDUCTIONS WITH ORIGIN OVER Σ, Γ $\longleftrightarrow$ CLOSED TYPED DATA WORDS OVER $\Sigma, \mathbb{N}, \Gamma$

LOGICS

$FO_2[\leq_{out}, \sigma, MSO_{bin}[\leq_{in}]]$ $\longleftrightarrow$ $FO_2[\leq, MSO_{bin}[\preccurlyeq]]$

TRANSDUCTIONS

DATA WORDS

MODELS

NON-ERASING TRANSDUCTIONS WITH ORIGIN OVER Σ, Γ $\Longleftrightarrow$ CLOSED TYPED DATA WORDS OVER $\Sigma, \mathbb{N}, \Gamma$

LOGICS

$FO_2[\leq_{out}, \sigma, MSO_{bin}[\leq_{in}]]$ $\Longleftrightarrow$ $FO_2[\leq, MSO_{bin}[\preceq]]$

TRANSDUCTIONS | DATA WORDS

CONSEQUENCE

- $FO_2[\leq, MSO_{bin}[\preceq]]$ is decidable on typed data words over $\mathbb{N}$ (closedness does not matter)
- $EMSO_2[\leq, MSO_{bin}[\preceq]]$ is decidable too.

$FO_2[\leq_\ell, \preceq_p]$: [Schwentick, Zeume]

SUMMARY

- new logic tailored to express non-functional transductions
- decidable satisfiability / model-checking / equivalence / functionality
- regular synthesis
- new characterization of REG
- origin-sensitive ! otherwise undecidable

$$\varphi, \varphi' \in \mathcal{L}_T \qquad \pi_o([\![\varphi]\!]) = \pi_o([\![\varphi']\!])$$

SUMMARY

- new logic tailored to express non-functional transductions
- decidable satisfiability / model-checking / equivalence / functionality
- regular synthesis
- new characterization of REG
- origin-sensitive ! otherwise undecidable

FUTURE WORK

- synthesis of "memory-efficient" machines from $\mathcal{L}_T$
- extensions to trees & data word transductions

SUMMARY

- new logic tailored to express non-functional transductions
- decidable satisfiability / model-checking / equivalence / functionality
- regular synthesis
- new characterization of REG
- origin-sensitive ! otherwise undecidable

FUTURE WORK

- synthesis of "memory-efficient" machines from $\mathcal{L}_T$
- extensions to trees & data word transductions

CHALLENGE

- MSOT = 2DFT = 2RFT = SST = $\mathcal{L}_T \cap$ functions = ...
- canonical model ?