Origin-equivalence of two-way word transducers is in PSPACE

Sougata Bose

LaBRI, Université de Bordeaux

Delta Meeting 2018

Joint work with Anca Muscholl, Gabriele Puppis and Vincent Penelle

Transformations of finite words

- Input Alphabet Σ, Output Alphabet Γ
- Define a relation $R \subseteq \Sigma^* \times \Gamma^*$

Transformations of finite words

- Delete all occurrences of the letter "A"

$$\text{AHMEDABAD} \longrightarrow \text{HMEDBD}$$

Transformations of finite words

- Subword Relation

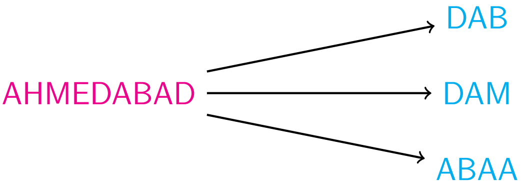
AHMEDABAD → HMEDBD

AHMEDABAD → ADD

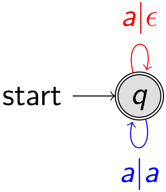
AHMEDABAD → HEAD

Transformations of finite words

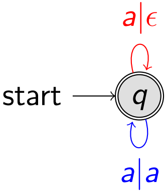
- Reverse Subword Relation



Transducers: For the Subword Relation

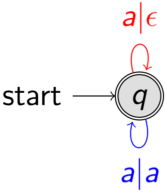


Transducers: For the Subword Relation

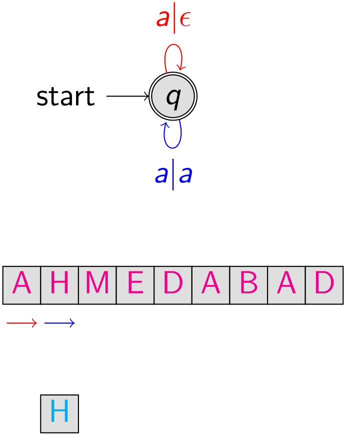


A	H	M	E	D	A	B	A	D

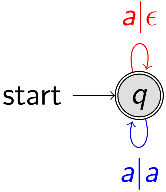
Transducers: For the Subword Relation

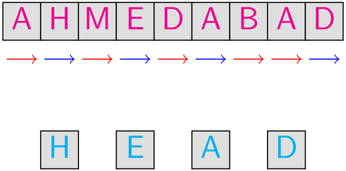


Transducers: For the Subword Relation

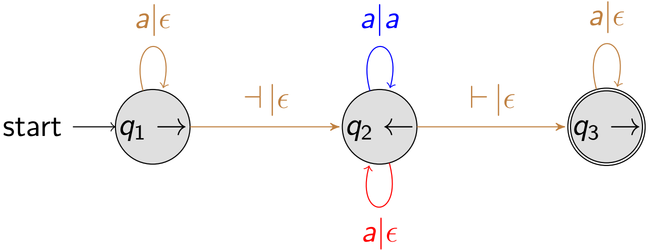


Transducers: For the Subword Relation

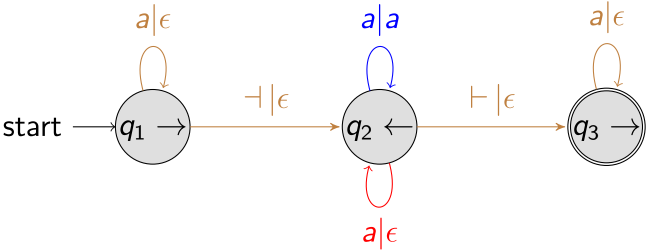




Two-Way Transducers: The Reverse Subword Relation

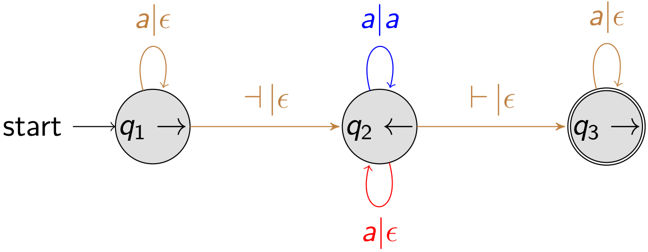


Two-Way Transducers: The Reverse Subword Relation

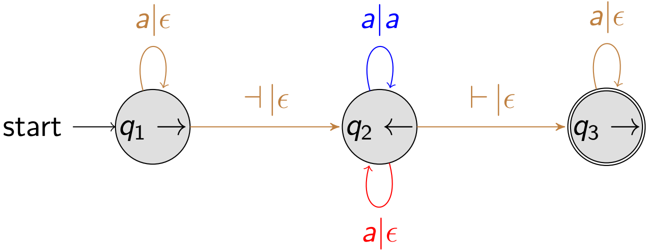


⊢	A	H	M	E	D	A	B	A	D	⊣

Two-Way Transducers: The Reverse Subword Relation



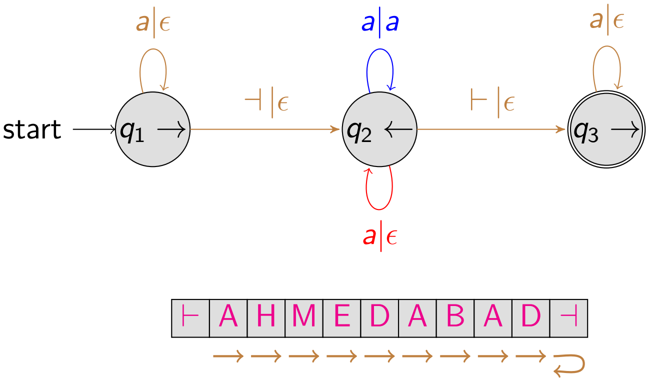
Two-Way Transducers: The Reverse Subword Relation



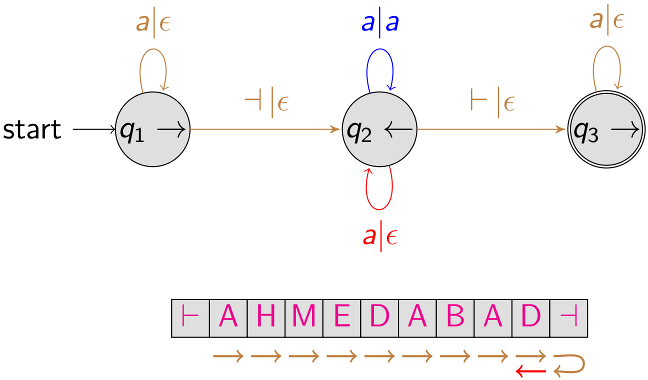
$\vdash$	A	H	M	E	D	A	B	A	D	$\dashv$

$\rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow$

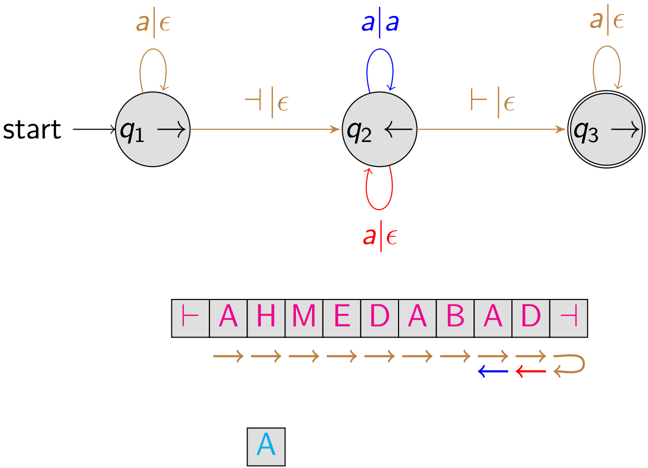
Two-Way Transducers: The Reverse Subword Relation



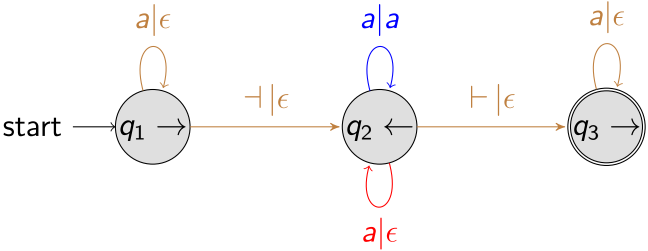
Two-Way Transducers: The Reverse Subword Relation

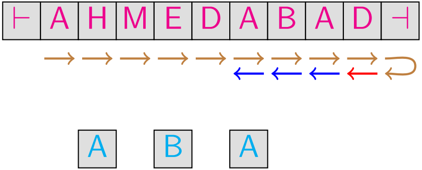


Two-Way Transducers: The Reverse Subword Relation

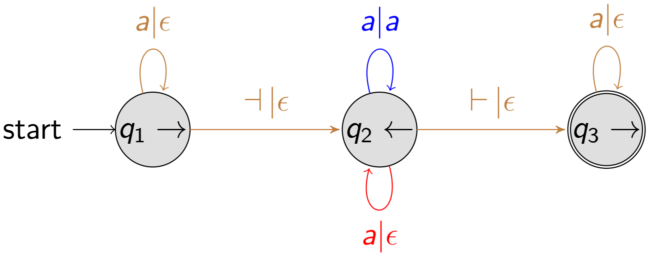


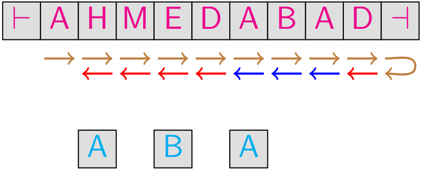
Two-Way Transducers: The Reverse Subword Relation



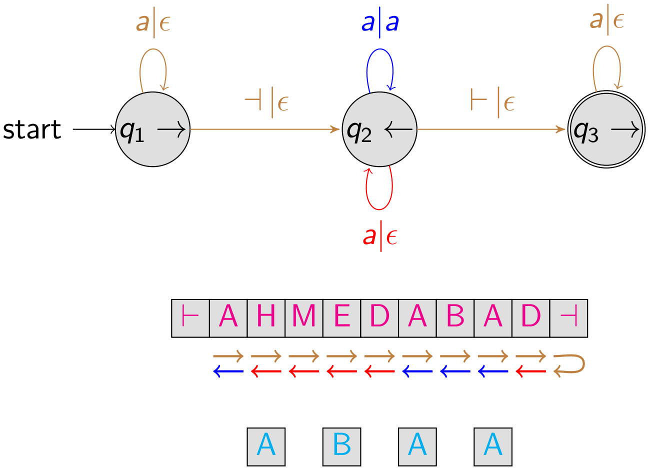


Two-Way Transducers: The Reverse Subword Relation

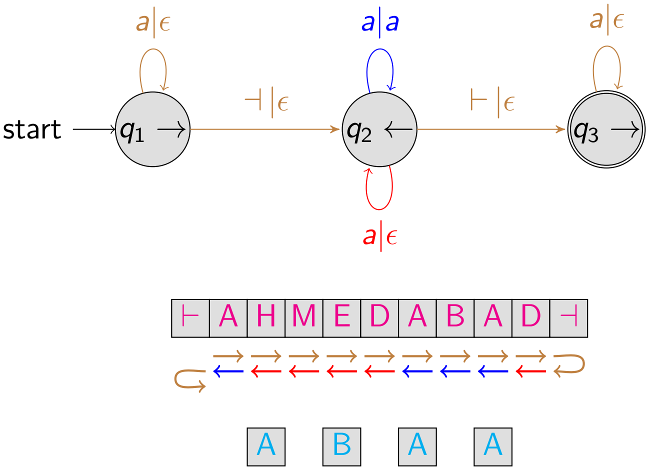




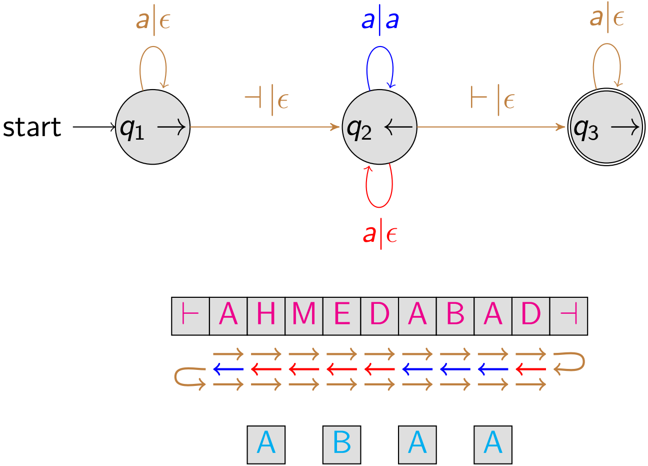
Two-Way Transducers: The Reverse Subword Relation



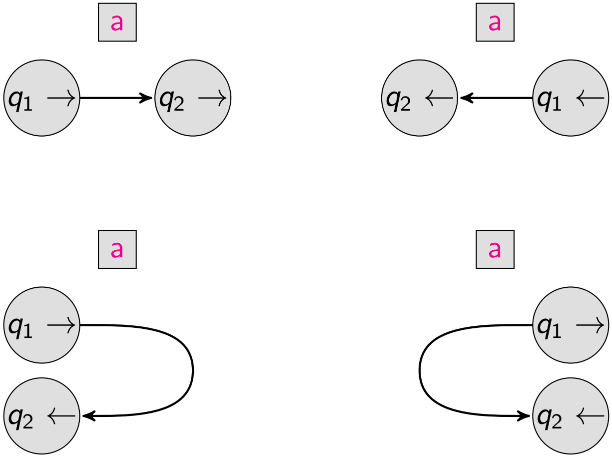
Two-Way Transducers: The Reverse Subword Relation



Two-Way Transducers: The Reverse Subword Relation



Shape of a run



Shape of a run

- Shape of a run is the sequence of shape of transition taken in the run.

$$\hookrightarrow \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \begin{matrix}\rightarrow\\ \leftarrow\\ \rightarrow\end{matrix} \hookleftarrow$$

Equivalence Problem

The problem

Given two transducers T_1 and T_2, check if they compute the same relation.

Equivalence Problem

The problem

Given two transducers T_1 and T_2, check if they compute the same relation.

Functional Case

Equivalence Problem is decidable in PSPACE for 2-way functional transducers. [Culik, Karhumäki, '87]

Equivalence Problem

The problem

Given two transducers T_1 and T_2, check if they compute the same relation.

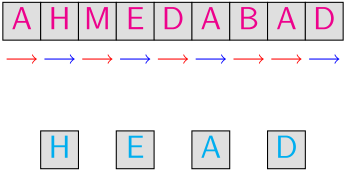
Functional Case

Equivalence Problem is decidable in PSPACE for 2-way functional transducers. [Culik, Karhumäki, '87]

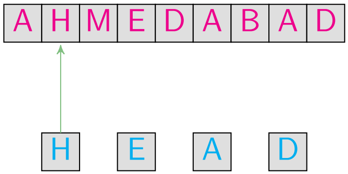
Relational Case

Equivalence Problem is undecidable even for 1-way transducers. [Griffiths '68]

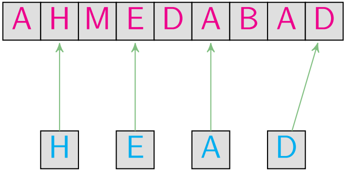
Origin Semantics [Bojańczyk, '14]



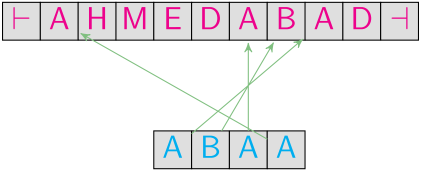
Origin Semantics [Bojańczyk, '14]



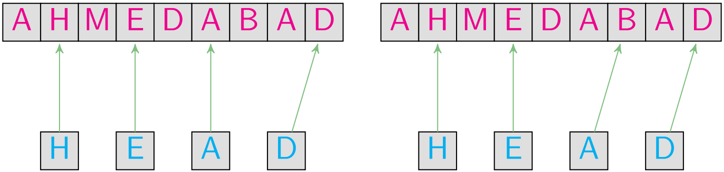
Origin Semantics [Bojańczyk, '14]



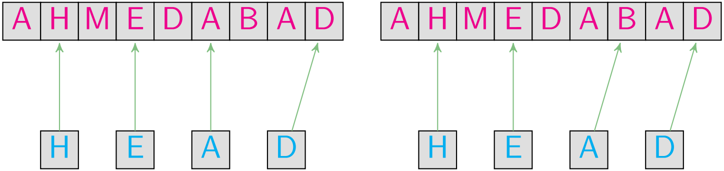
Origin Semantics [Bojańczyk, '14]



Equivalence under Origin Semantics



Equivalence under Origin Semantics



Not Equivalent

Equivalence under Origin Semantics

State of the Art

- 1-way Transducers [Filiot et al '16]
- Streaming String Transducers [Bojańczyk et al, '17]
- Top-down Tree Transducers [Filiot, et al '18]

Equivalence under Origin Semantics

State of the Art

- 1-way Transducers [Filiot et al '16]
- Streaming String Transducers [Bojańczyk et al, '17]
- Top-down Tree Transducers [Filiot, et al '18]

Input is processed in one direction

Equivalence under Origin Semantics

State of the Art

- 1-way Transducers [Filiot et al '16]
- Streaming String Transducers [Bojańczyk et al, '17]
- Top-down Tree Transducers [Filiot, et al '18]

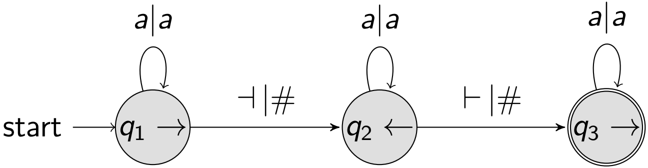
Input is processed in one direction

Theorem

Origin-equivalence is decidable in PSPACE for non deterministic 2-way transducers

Subcase : Busy Transducers

All transitions produce non-empty output.



$w \longrightarrow w\#w^r\#w$

Equivalence of Busy transducers

Characterization for containment

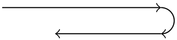
For every run of T_1, there exists a run of T_2 with the same shape and same output.

Equivalence of Busy transducers

Characterization for containment

For every run of T_1, there exists a run of T_2 with the same shape and same output.

- Guess a run of T_1

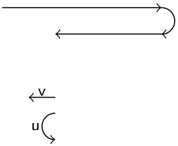


Equivalence of Busy transducers

Characterization for containment

For every run of T_1, there exists a run of T_2 with the same shape and same output.

- Guess a run of T_1

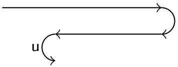


Equivalence of Busy transducers

Characterization for containment

For every run of T_1, there exists a run of T_2 with the same shape and same output.

- Guess a run of T_1

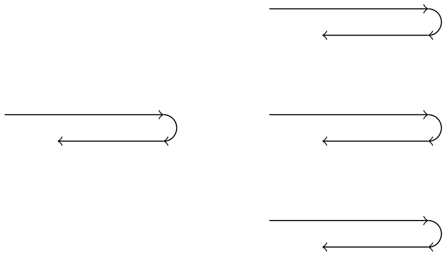


Equivalence of Busy transducers

Characterization for containment

For every run of T_1, there exists a run of T_2 with the same shape and same output.

- Guess a run of T_1 and multiple matching runs of T_2.

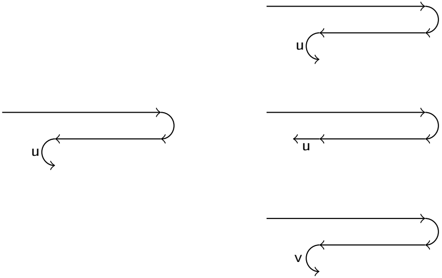


Equivalence of Busy transducers

Characterization for containment

For every run of T_1, there exists a run of T_2 with the same shape and same output.

- Guess a run of T_1 and multiple matching runs of T_2.

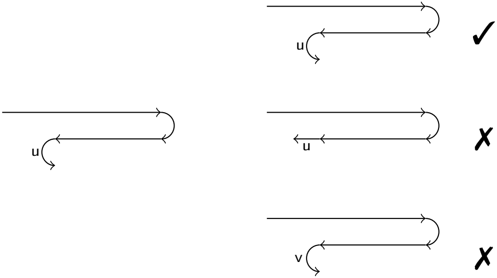


Equivalence of Busy transducers

Characterization for containment

For every run of T_1, there exists a run of T_2 with the same shape and same output.

- Guess a run of T_1 and multiple matching runs of T_2.



Equivalence of Busy transducers

Characterization for containment

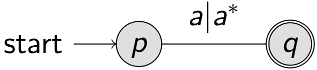
For every run of T_1, there exists a run of T_2 with the same shape and same output.

Can be checked in PSPACE by similar techniques for two-way automata. [Vardi '89]

Same as equivalence for NFA!

Busy Transducer with regular outputs

- Transitions are of the form (q, a, L, q').
- Transition can output any word from the language L.

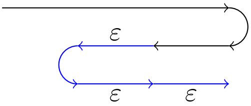


Reduce any arbitrary transducer to Busy transducer with regular output

Origin-Containment of arbitrary transducers

Key Idea

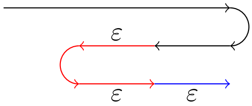
Reduce to the busy case



Origin-Containment of arbitrary transducers

Key Idea

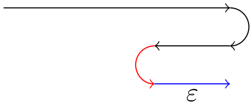
Reduce to the busy case



Origin-Containment of arbitrary transducers

Key Idea

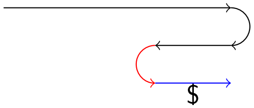
Reduce to the busy case



Origin-Containment of arbitrary transducers

Key Idea

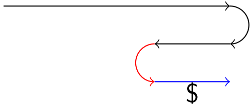
Reduce to the busy case



Origin-Containment of arbitrary transducers

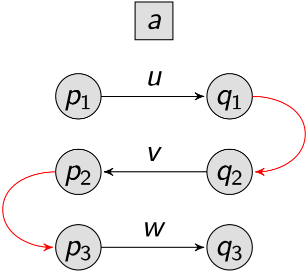
Key Idea

Reduce to the busy case

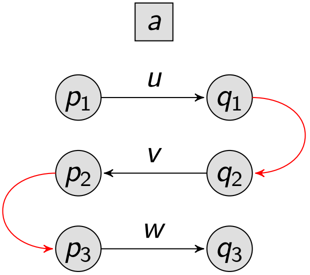


- Remove Lazy U turns
- Output special symbol $ on straight lazy paths

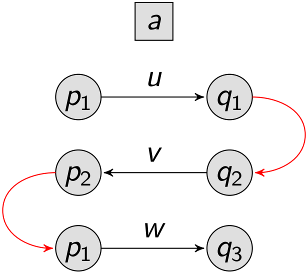
Removing lazy U-turns



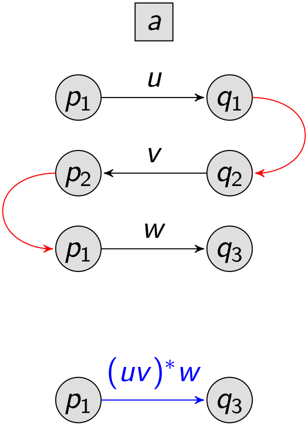
Removing lazy U-turns



Removing lazy U-turns

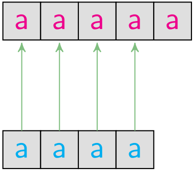


Removing lazy U-turns

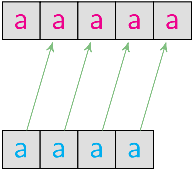


Identifying similar origins

- Origin equivalence is stronger than classical equivalence



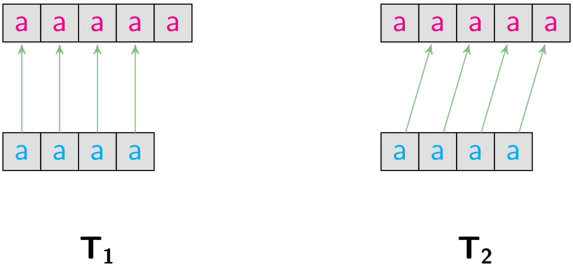
$\mathbf{T_1}$



$\mathbf{T_2}$

Identifying similar origins

- Origin equivalence is stronger than classical equivalence

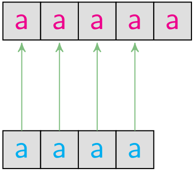


Goal

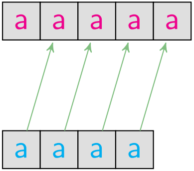
Relax Containment relation under origin semantics

Identifying similar origins

- Origin equivalence is stronger than classical equivalence



$\mathbf{T}_1$

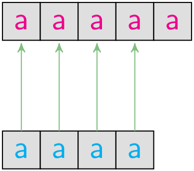


$\mathbf{T}_2$

Resynchronizers

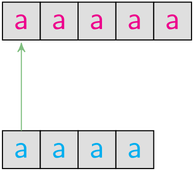
Introduced by Filiot et al '17 for 1-way case

Resynchronizers



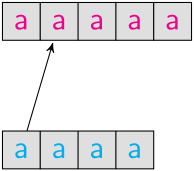
- $\gamma(y, z) : z = y + 1$
- MSO formula on the input

Resynchronizers



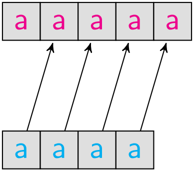
- $\gamma(y, z) : z = y + 1$
- MSO formula on the input
- Outputs with origin y can get origin z.
- $\gamma(1, 2)$

Resynchronizers



- $\gamma(y, z) : z = y + 1$
- MSO formula on the input
- Outputs with origin y can get origin z.

Resynchronizers



- $\gamma(y,z) : z = y + 1$
- MSO formula on the input
- Outputs with origin y can get origin z.

Containment modulo Resynchronizer

$\mathbf{T}_1$ $\qquad$ $\mathbf{T}_2$

Containment modulo Resynchronizer

$\mathbf{T_1}$	$\mathbf{T_2}$
$\mathbf{G_1}$	$\mathbf{G_2}$

Containment modulo Resynchronizer

$$\begin{array}{ccc} \mathbf{T_1} & & \mathbf{T_2} \\ \mathbf{G_1} & \mathbf{R}_\gamma(\mathbf{G_2}) \longleftarrow & \mathbf{G_2} \end{array}$$

Containment modulo Resynchronizer

$$\begin{array}{ccccc} \mathbf{T_1} & & & & \mathbf{T_2} \\ & & & & \\ \mathbf{G_1} & \subseteq & \mathbf{R}_\gamma(\mathbf{G_2}) & \longleftarrow & \mathbf{G_2} \end{array}$$

Containment modulo Resynchronizer

$$\begin{array}{ccccc} \mathbf{T_1} & & \subseteq_R & & \mathbf{T_2} \\ \mathbf{G_1} & \subseteq & \mathbf{R}_\gamma(\mathbf{G_2}) & \longleftarrow & \mathbf{G_2} \end{array}$$

Containment modulo Resynchronizer

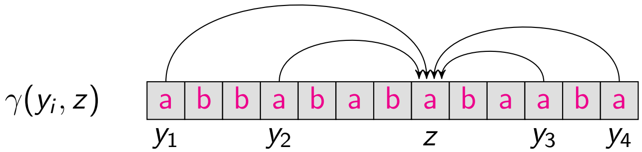
$$\begin{array}{ccccc} \mathbf{T_1} & & \subseteq_R & & \mathbf{T_2} \\ \mathbf{G_1} & \subseteq & \mathbf{R}_\gamma(\mathbf{G_2}) & \longleftarrow & \mathbf{G_2} \end{array}$$

- $\gamma(y, z) = \text{true}$
- This reduces to classical containment

Restrict the formula γ

k-bounded Restriction

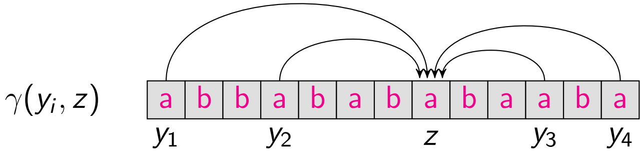
- Outputs with origin y get origin z.



For a fixed z, there are at most k positions $y_1, y_2, \ldots, y_k$ such that $\gamma(y_i, z)$ is true.

k-bounded Restriction

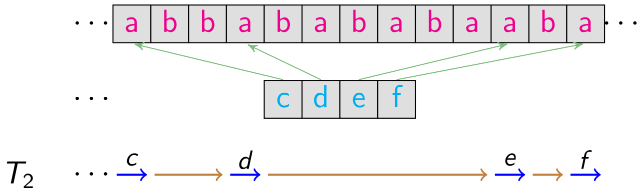
- Outputs with origin y get origin z.



k-boundedness is decidable

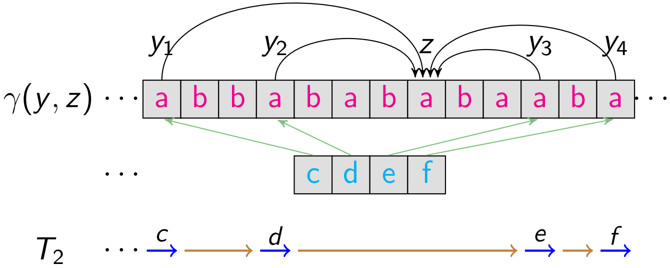
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



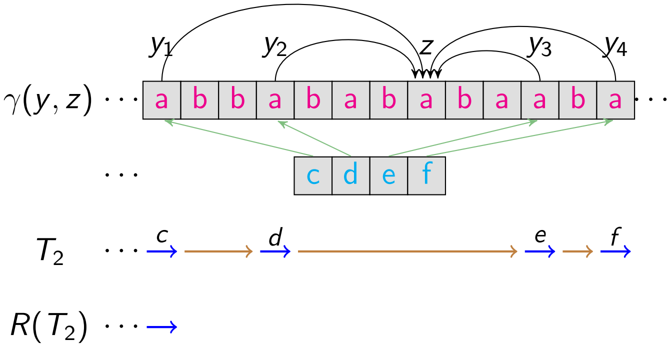
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



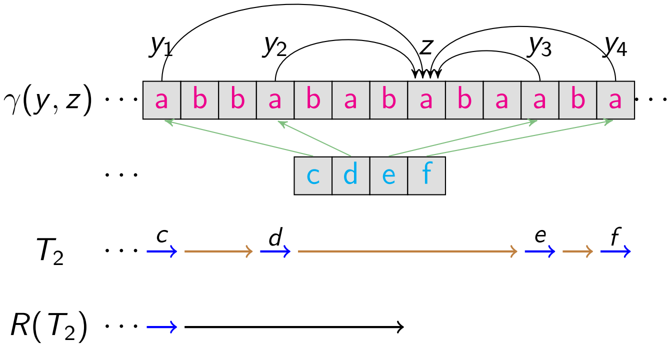
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



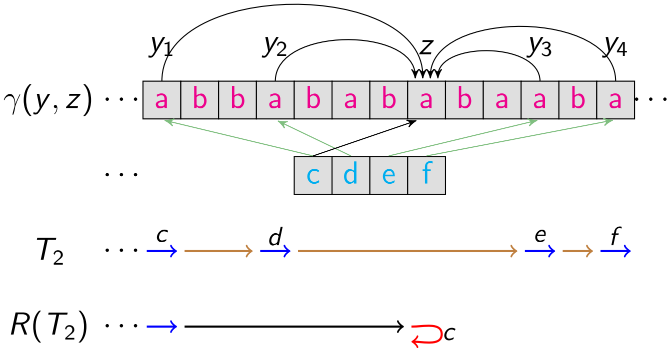
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



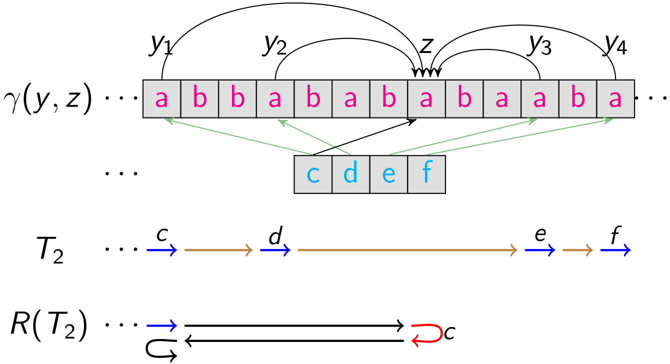
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



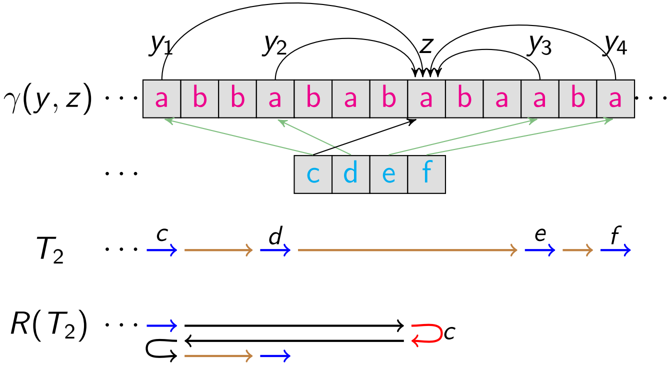
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



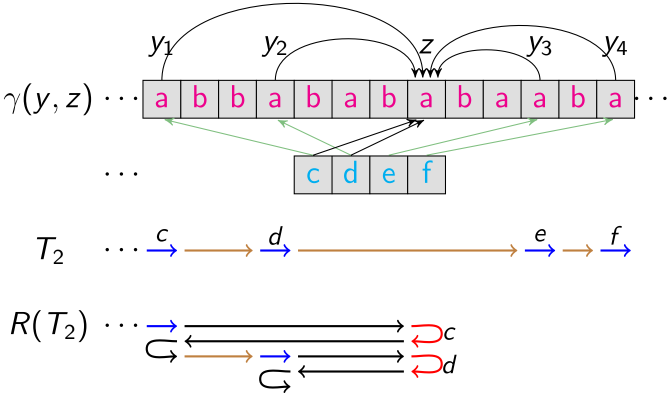
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



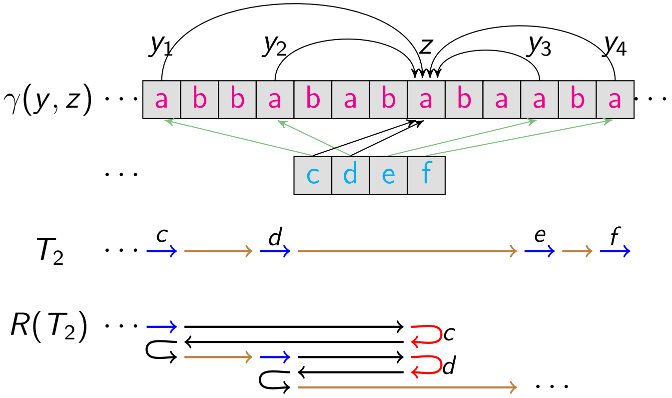
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



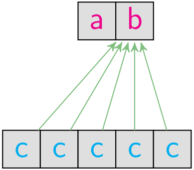
Equivalence modulo Resynchronizer

Equivalence modulo Resynchronizer is decidable.



Resynchronizers

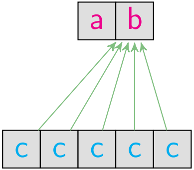
Split the block of c's in an ordered manner



Resynchronizers

Split the block of c's in an ordered manner

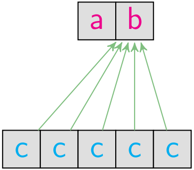
- $\gamma(y, z)$: $z = y$



Resynchronizers

Split the block of c's in an ordered manner

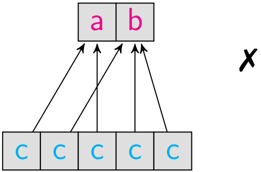
- $\gamma(y, z)$: $z = y \ \vee z = y - 1$



Resynchronizers

Split the block of c's in an ordered manner

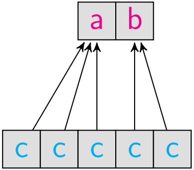
- $\gamma(y, z)$: $z = y \ \vee z = y - 1$



Resynchronizers

Split the block of c's in an ordered manner

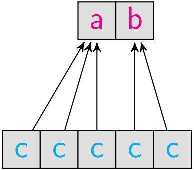
- $\gamma(y, z)$: $z = y \ \vee z = y - 1$
- $\delta(y, y')$: $y = y' \wedge y = y' - 1$



Resynchronizers

Split the block of c's in an ordered manner

- α β
- $\gamma(y, z)$: $z = y \;\vee z = y - 1$
- $\delta(y, y')$: $y = y' \wedge y = y' - 1$



Conclusion

Summary

- Origin Equivalence for 2-way transducers
- Resynchronizers for 2-way transducers

Conclusion

Summary

- Origin Equivalence for 2-way transducers
- Resynchronizers for 2-way transducers

Future Works

- Capture one way resynchronizers
- Composition of resynchronizers

Thank You!
Questions?