On nondeterministic two-way transducers

Bruno Guillon

Dipartimento di Informatica, Università degli Studi di Milano

NCMA

August 21, 2018

mainly, joint work with Christian Choffrut

Outline

1. Introduction
   - Word transductions
   - Automata and transducers
2. Algebraic descriptions of transduction classes
   - Rational operations
   - Hadamard operations
   - Mirror operation
3. Unary cases
   - Commutative outputs
   - Both alphabets unary
   - Only one alphabet unary

Relations in computer science

Relation $\equiv$ *set of tuples*

Omnipresent in computer science

- Graph structures
- Data bases
- Semantics of programs
- Rewriting systems
- ...

Relations in computer science

Relation $\equiv$ *set of tuples*

Omnipresent in computer science

- Graph structures
- Data bases
- Semantics of programs
- Rewriting systems
- ...

Transduction $\equiv$ *a binary relation*

in which an input and an output are implicitly understood

Word transductions

This talk:

- binary relations on words $R \subseteq \Sigma^* \times \Delta^*$
- computed by some kind of transducers

Word transductions

This talk:

- binary relations on words $R \subseteq \Sigma^* \times \Delta^*$
- computed by some kind of transducers

Equivalent formalisms:

- A function from words into languages: $f_R : \begin{array}{rcl} \Sigma^* & \to & 2^{\Delta^*} \\ u & \mapsto & \{v \mid (u, v) \in R\} \end{array}$

Word transductions

This talk:

- binary relations on words $\quad R \subseteq \Sigma^* \times \Delta^*$
- computed by some kind of transducers

Equivalent formalisms:

- A function from words into languages: $\quad f_R : \begin{array}{ccc} \Sigma^* & \to & 2^{\Delta^*} \\ u & \mapsto & \{v \mid (u,v) \in R\} \end{array}$
- A formal power series: $\quad \sigma = \sum_{u \in \Sigma^*} \langle \sigma, u \rangle u \quad$ with $\langle \sigma, u \rangle = f_R(u)$

Word transductions

This talk:

- binary relations on words $R \subseteq \Sigma^* \times \Delta^*$
- computed by some kind of transducers

Equivalent formalisms:

- A function from words into languages: $f_R : \begin{array}{ccc} \Sigma^* & \to & 2^{\Delta^*} \\ u & \mapsto & \{v \mid (u,v) \in R\} \end{array}$
- A formal power series: $\sigma = \sum_{u \in \Sigma^*} \langle \sigma, u \rangle u$ with $\langle \sigma, u \rangle = f_R(u)$
- computed by some kind of weighted automata over $\mathrm{RAT}(\Delta^*)$

[Lombardy's talk at NCMA'15 in Porto]

Word transductions

This talk:

- binary relations on words $R \subseteq \Sigma^* \times \Delta^*$
- computed by some kind of transducers

constant-memory nondeterministic devices

Equivalent formalisms:

- A function from words into languages: $f_R : \begin{array}{ccc} \Sigma^* & \to & 2^{\Delta^*} \\ u & \mapsto & \{v \mid (u,v) \in R\} \end{array}$
- A formal power series: $\sigma = \sum_{u \in \Sigma^*} \langle \sigma, u \rangle u$ with $\langle \sigma, u \rangle = f_R(u)$
- computed by some kind of weighted automata over $\mathrm{R{\scriptsize AT}}(\Delta^*)$

[Lombardy's talk at NCMA'15 in Porto]

Word transductions

This talk:

- binary relations on words $R \subseteq \Sigma^* \times \Delta^*$
- computed by some kind of transducers

constant-memory nondeterministic devices

Equivalent formalisms:

- A function from words into languages: $f_R : \begin{array}{ccc} \Sigma^* & \to & 2^{\Delta^*} \\ u & \mapsto & \{v \mid (u,v) \in R\} \end{array}$
- A formal power series: $\sigma = \sum_{u \in \Sigma^*} \langle \sigma, u \rangle u$ with $\langle \sigma, u \rangle = f_R(u)$
- computed by some kind of weighted automata over $\mathrm{RAT}(\Delta^*)$

[Lombardy's talk at NCMA'15 in Porto]

Which issues arise from nondeterminism?

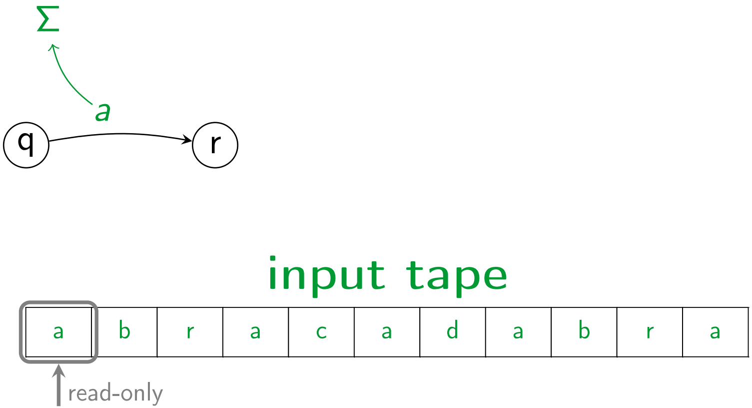
Which issues arise from nondeterminism?

How can we handle them

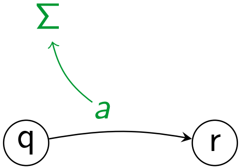
Which issues arise from nondeterminism?

How can we handle them, in some special cases?

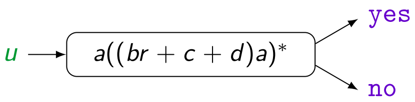
Automata with outputs: 1-way transducers

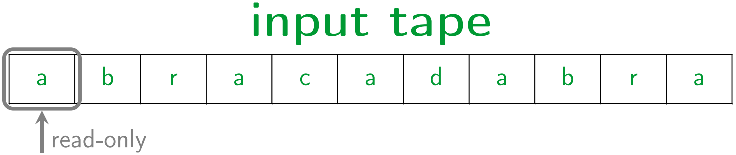


Automata with outputs: 1-way transducers

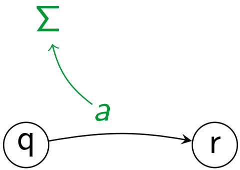


- Example:

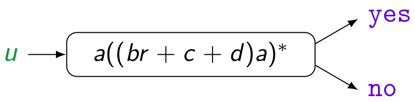




Automata with outputs: 1-way transducers



- Example:



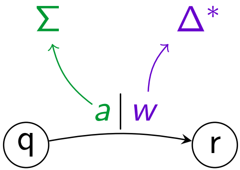
input tape

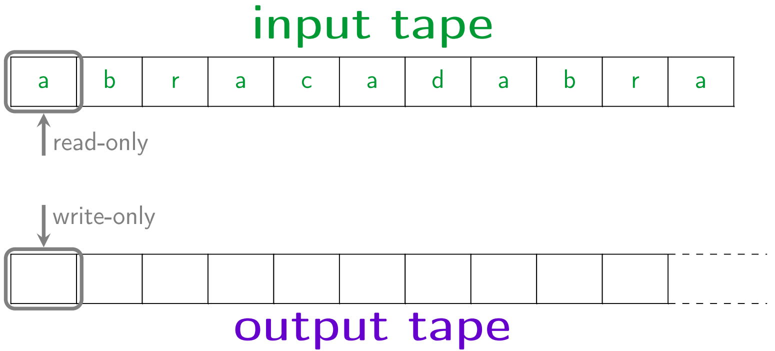
a	b	r	a	c	a	d	a	b	r	a

read-only

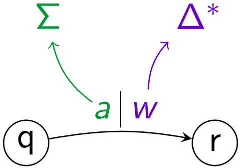
yes

Automata with outputs: 1-way transducers





Automata with outputs: 1-way transducers



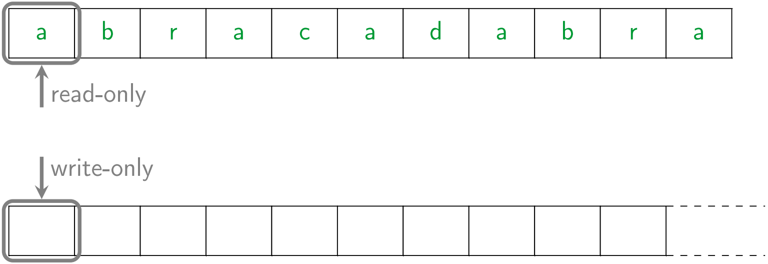
- Example:

$u \longrightarrow$

- replace a by b
- replace b by aa
- ignore other letters

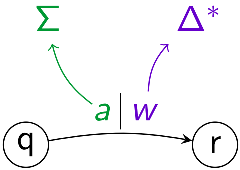
$\longrightarrow v \in \Delta^*$

input tape



output tape

Automata with outputs: 1-way transducers



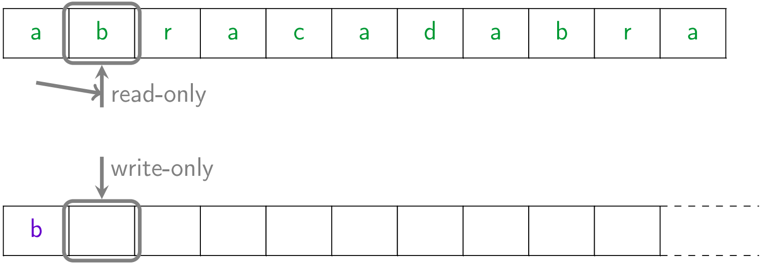
Example:

$u \longrightarrow$

- replace a by b
- replace b by aa
- ignore other letters

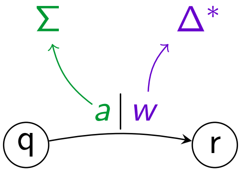
$\longrightarrow v \in \Delta^*$

input tape

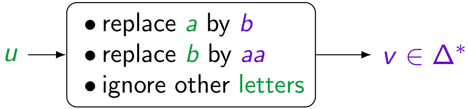


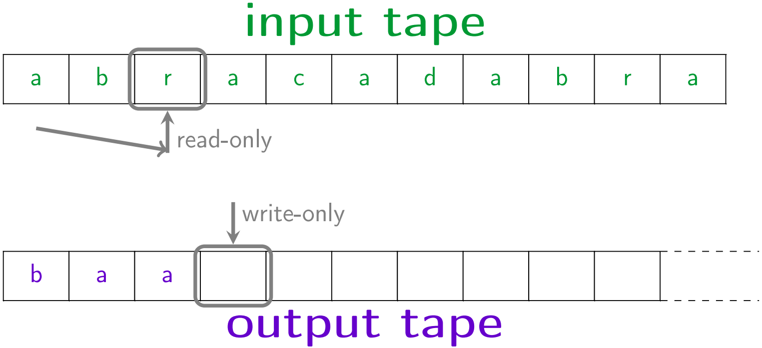
output tape

Automata with outputs: 1-way transducers

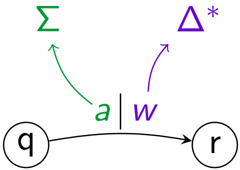


- Example:





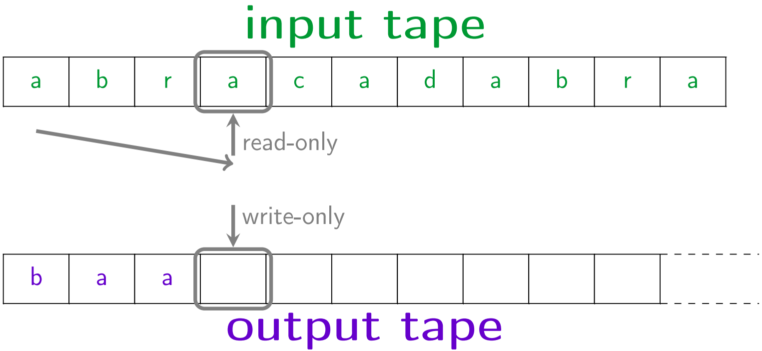
Automata with outputs: 1-way transducers



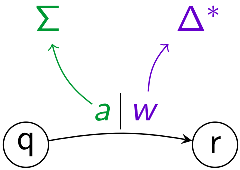
- Example:

$u \longrightarrow$
- replace a by b
- replace b by aa
- ignore other letters

$\longrightarrow v \in \Delta^*$



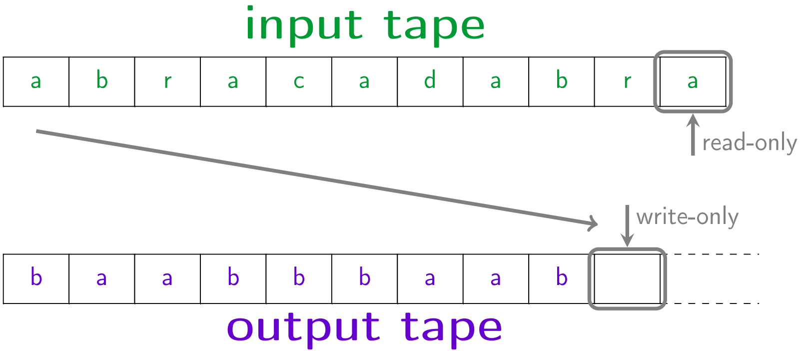
Automata with outputs: 1-way transducers



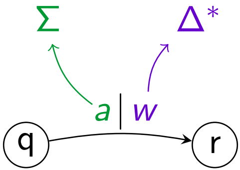
Example:

$u \longrightarrow$
- replace a by b
- replace b by aa
- ignore other letters

$\longrightarrow v \in \Delta^*$



Automata with outputs: 1-way transducers

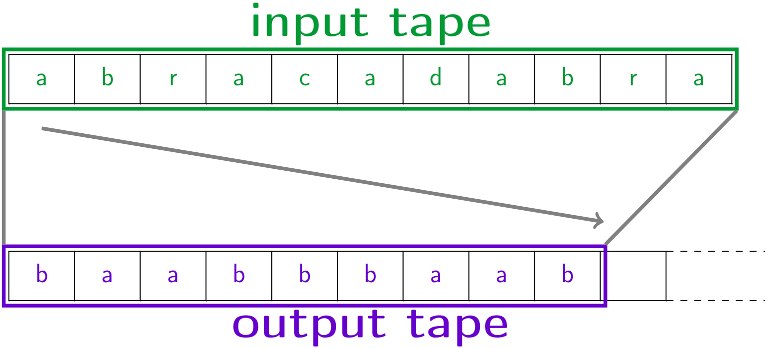


- Example:

$u \longrightarrow$

- replace a by b
- replace b by aa
- ignore other letters

$\longrightarrow v \in \Delta^*$



Examples

with $\Sigma = \Delta$ fixed

- $\textsc{Identity} : u \mapsto u$
- $\textsc{Erase} : u \mapsto \varepsilon$
- $\textsc{L-Rotate} : \sigma u \mapsto u\sigma$, for each $\sigma \in \Sigma = \Delta$
- $\textsc{R-Rotate} : u\sigma \mapsto \sigma u$, for each $\sigma \in \Sigma = \Delta$
- $\textsc{Subword} : \{(u, v) \mid v$ is a not-necessarily connected subword of $u\}$

Nondeterminism versus determinism

- functions $\subset$ relations

  e.g., no deterministic transducer realize subword

Nondeterminism versus determinism

- functions $\subset$ relations

  e.g., no deterministic transducer realize subword

- sequential functions $\subset$ rational functions

  e.g., no deterministic transducer realize right-rotate

Nondeterminism versus determinism

- functions $\subset$ relations

  e.g., no deterministic transducer realize subword

- sequential functions $\subset$ rational functions

  e.g., no deterministic transducer realize right-rotate

- [Griffith'68] equivalence, inclusion, intersection emptiness...
  are undecidable problems for nondeterministic transducers

Two-wayness

A transducer is defined by:

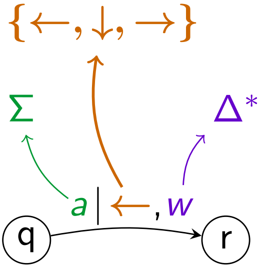
- an automaton with transition set δ
- a production function from δ to Δ^*

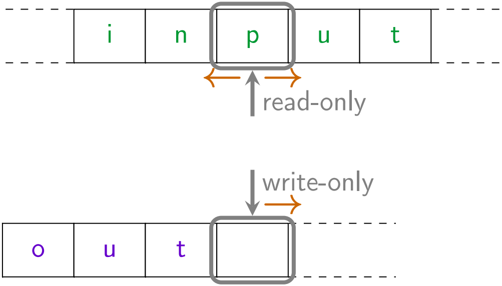
Two-wayness

A transducer is defined by:

- an automaton with transition set δ
- a production function from δ to Δ^*

Two-way transducers:



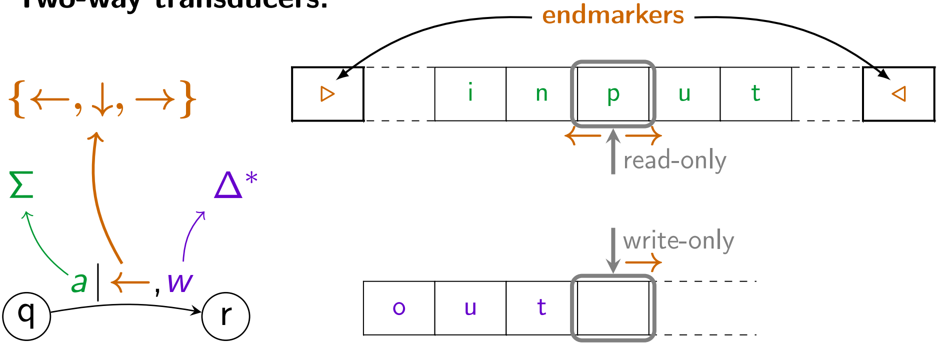


Two-wayness

A transducer is defined by:

- an automaton with transition set δ
- a production function from δ to Δ^*

Two-way transducers:



Examples

Two-wayness extends expressiveness of transducers...

- $\textsc{Square}$: $u \mapsto uu$
- $\textsc{Mirror}$: $u \mapsto \overline{u}$ ($\overline{u}$ denotes the reverse of u)
- $\textsc{Sort}$: $u \mapsto a^{|u|_a} b^{|u|_b} \dots z^{|u|_z}$
- $\textsc{Powers}$: $\left\{(u, u^k) \mid k \in \mathbb{N}\right\}$

"Regular" transductions

The class of **functions** realized by two-way transducers is **robust**

- closure under composition [Chytil&Jákl'77, Dartois *et al.*'17]
- decidable equivalence [Gurari'80]
- alternative characterizations:
  - reversible = deterministic = functional [Dartois *et al.*'17, Engelfriet&Hoogeboom'01]
  - MSO word transductions [Engelfriet&Hoogeboom'01]
  - copyless register automata [Alur&Černy'10]
  - "regular combinators" [Alur *et al.*'14, Baudru&Reynier'18, Dave *et al.*'18]

Expressiveness of transducers

	deterministic	functional	unrestricted
two-way	2DFT	$2f$NFT	2NFT
one-way	1DFT	$1f$NFT	1NFT

Expressiveness of transducers

	deterministic	functional		unrestricted
two-way	2DFT	$2f$NFT		2NFT
		functions	*relations*	
one-way	1DFT	$1f$NFT		1NFT

Expressiveness of transducers

	deterministic	functional		unrestricted
two-way	2DFT	$2f$NFT	$\subset$	2NFT
one-way	1DFT	$1f$NFT	$\subset$	1NFT

Expressiveness of transducers

	deterministic	functional		unrestricted
two-way	2DFT	$2f$NFT	$\subset$	2NFT
one-way	1DFT	$1f$NFT	$\subset$	1NFT

Expressiveness of transducers

	deterministic	functional		unrestricted
two-way	2DFT	2fNFT	$\subset$	2NFT
	$\cup$	$\cup$		$\cup$
one-way	1DFT	1fNFT	$\subset$	1NFT

Expressiveness of transducers

	deterministic	functional		unrestricted
two-way	2DFT	$2f\text{NFT}$	$\subset$	2NFT
	$\cup$	$\cup$		$\cup$
one-way	1DFT	$1f\text{NFT}$	$\subset$	1NFT
	sequential			

Expressiveness of transducers

	deterministic		functional		unrestricted
two-way	$2\textsc{dft}$		$2f\textsc{nft}$	$\subset$	$2\textsc{nft}$
	$\cup$		$\cup$		$\cup$
one-way	$1\textsc{dft}$	$\subset$	$1f\textsc{nft}$	$\subset$	$1\textsc{nft}$

Expressiveness of transducers

	deterministic		functional		unrestricted
two-way	$2\textsc{dft}$	$=$	$2f\textsc{nft}$	$\subset$	$2\textsc{nft}$
		regular			
	$\cup$		$\cup$		$\cup$
one-way	$1\textsc{dft}$	$\subset$	$1f\textsc{nft}$	$\subset$	$1\textsc{nft}$

What are the transductions realized by 2NFT?

What are the transductions realized by 2NFT?

2. Algebraic descriptions of transduction classes

Rational operations

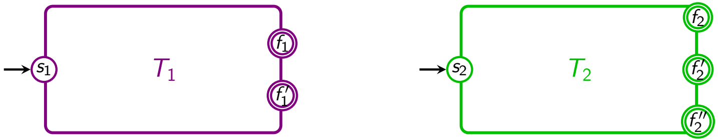
- set union $R_1 \cup R_2$
- componentwise concatenation
$$R_1 \cdot R_2 = \{(u_1 u_2, v_1 v_2) \mid (u_1, v_1) \in R_1 \text{ and } (u_2, v_2) \in R_2\}$$
- Kleene star $R^* = \{(u_1 \cdots u_k, v_1 \cdots v_k) \mid \forall i,\ (u_i, v_i) \in R\}$

Rational operations

- set union $R_1 \cup R_2$
- componentwise concatenation

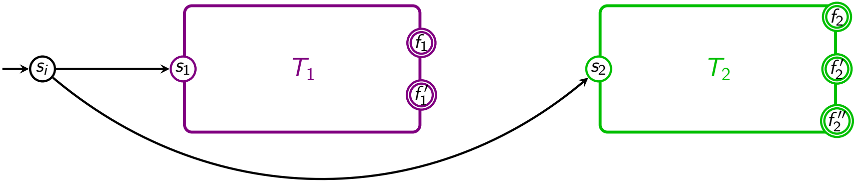
$$R_1 \cdot R_2 = \{(u_1 u_2, v_1 v_2) \mid (u_1, v_1) \in R_1 \text{ and } (u_2, v_2) \in R_2\}$$

- Kleene star $R^* = \{(u_1 \cdots u_k, v_1 \cdots v_k) \mid \forall i,\ (u_i, v_i) \in R\}$



Rational operations

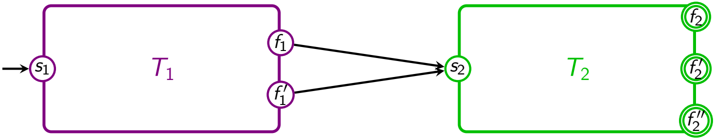
- set union $\quad R_1 \cup R_2$
- componentwise concatenation
$$R_1 \cdot R_2 = \{(u_1 u_2, v_1 v_2) \mid (u_1, v_1) \in R_1 \text{ and } (u_2, v_2) \in R_2\}$$
- Kleene star $\quad R^* = \{(u_1 \cdots u_k, v_1 \cdots v_k) \mid \forall i,\ (u_i, v_i) \in R\}$



- simulate T_1 or T_2

Rational operations

- set union $R_1 \cup R_2$
- componentwise concatenation
$$R_1 \cdot R_2 = \{(u_1 u_2, v_1 v_2) \mid (u_1, v_1) \in R_1 \text{ and } (u_2, v_2) \in R_2\}$$
- Kleene star $R^* = \{(u_1 \cdots u_k, v_1 \cdots v_k) \mid \forall i,\ (u_i, v_i) \in R\}$



- simulate T_1 on some prefix
- simulate T_2 on corresp. suffix

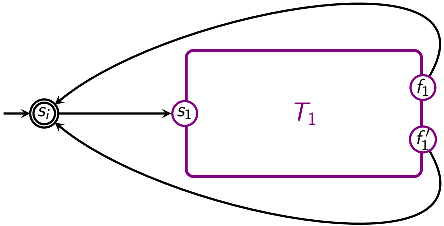
e.g., PREFIX = IDENTITY · ERASE

Rational operations

- set union $R_1 \cup R_2$
- componentwise concatenation

$$R_1 \cdot R_2 = \{(u_1 u_2, v_1 v_2) \mid (u_1, v_1) \in R_1 \text{ and } (u_2, v_2) \in R_2\}$$

- Kleene star $R^* = \{(u_1 \cdots u_k, v_1 \cdots v_k) \mid \forall i, (u_i, v_i) \in R\}$



- repeat
  simulate T_1 or accept

e.g., $\textsc{Subword} = (\textsc{Identity} \cup \textsc{Erase})^*$

Rational operations

- set union $R_1 \cup R_2$
- componentwise concatenation
$$R_1 \cdot R_2 = \{(u_1 u_2, v_1 v_2) \mid (u_1, v_1) \in R_1 \text{ and } (u_2, v_2) \in R_2\}$$
- Kleene star $R^* = \{(u_1 \cdots u_k, v_1 \cdots v_k) \mid \forall i, (u_i, v_i) \in R\}$

Definition (Rat) The class of *rational relations* is the smallest class

- including finite relations
- closed under rational operations

Rational operations

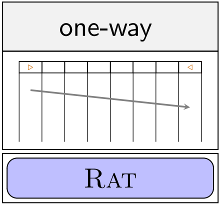
- set union $\qquad R_1 \cup R_2$
- componentwise concatenation
$$R_1 \cdot R_2 = \{(u_1 u_2, v_1 v_2) \mid (u_1, v_1) \in R_1 \text{ and } (u_2, v_2) \in R_2\}$$
- Kleene star $\qquad R^* = \{(u_1 \cdots u_k, v_1 \cdots v_k) \mid \forall i, (u_i, v_i) \in R\}$

Definition ($\textsc{Rat}$) The class of *rational relations* is the smallest class

- including finite relations
- closed under rational operations

Theorem: [Elgot & Mezei'65]

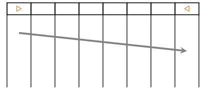
$$1\textsc{nft} = \text{rational}$$



Which operations capture behaviors of 2NFTs?

machine

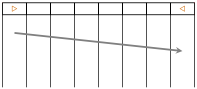
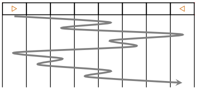
one-way



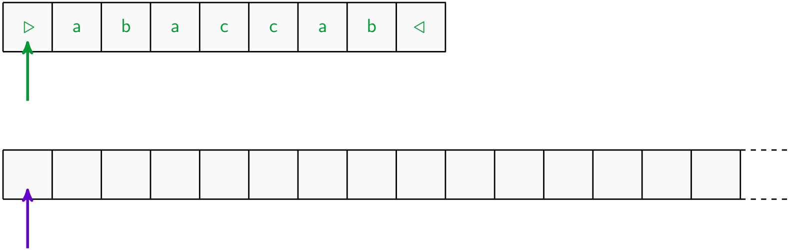
algebra

RAT

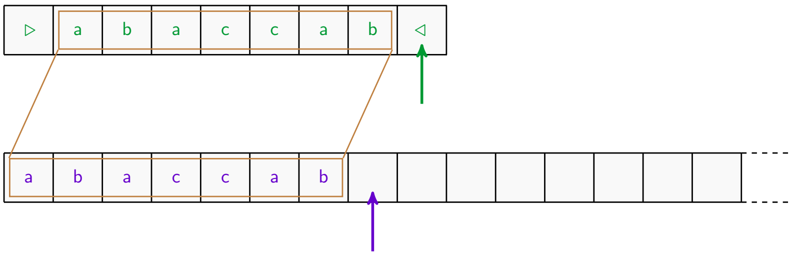
Which operations capture behaviors of 2NFTs?

machine	one-way	two-way
		
algebra	RAT	?

Abilities of two-way transducers

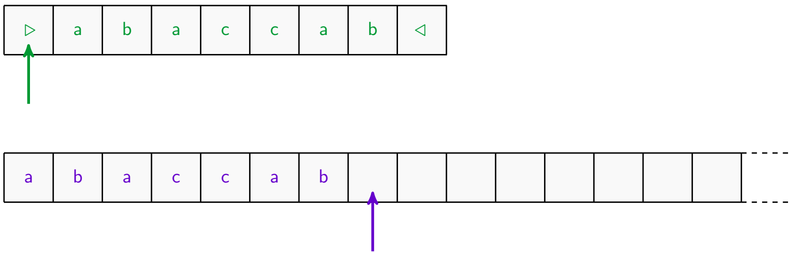


Abilities of two-way transducers



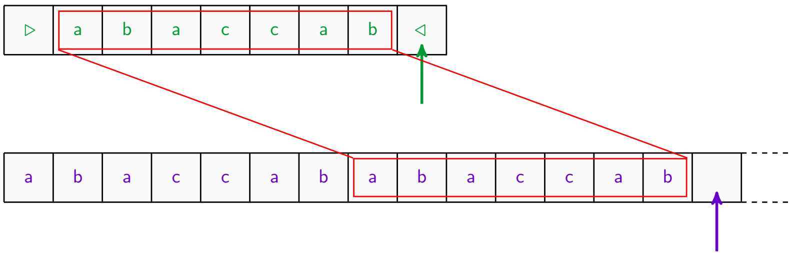
- copy the input word

Abilities of two-way transducers



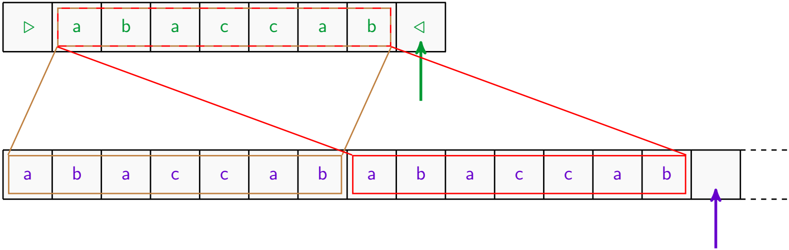
- copy the input word
- rewind the input tape

Abilities of two-way transducers



- copy the input word
- rewind the input tape
- append a copy of the input word

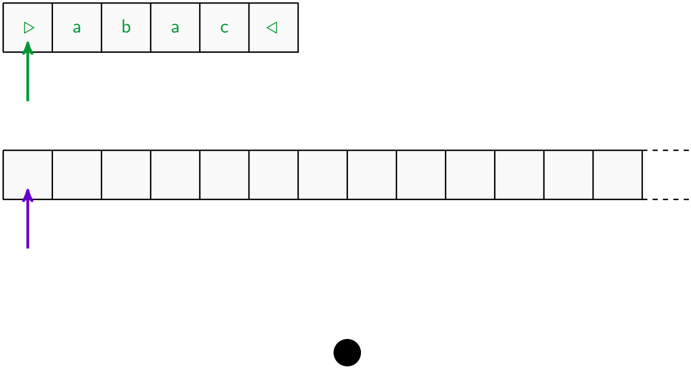
Abilities of two-way transducers



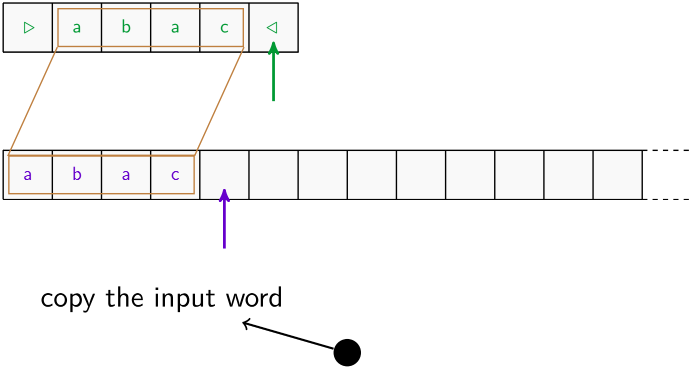
- copy the input word
- rewind the input tape
- append a copy of the input word

Abilities of two-way transducers

Abilities of two-way transducers

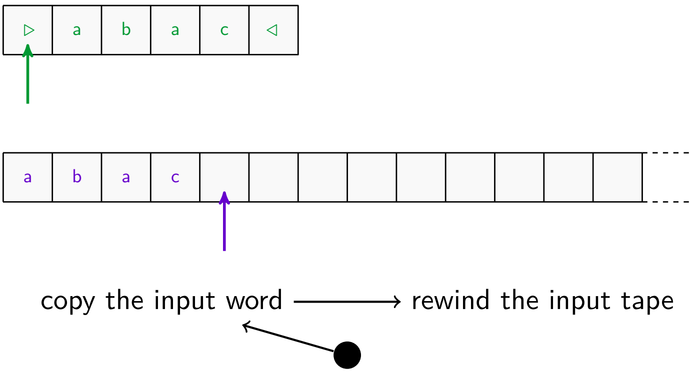


Abilities of two-way transducers

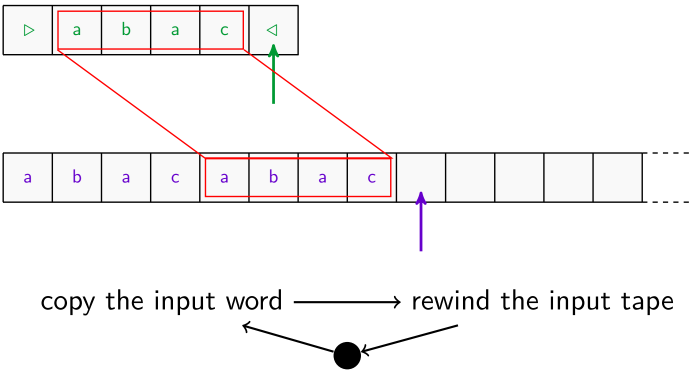


copy the input word

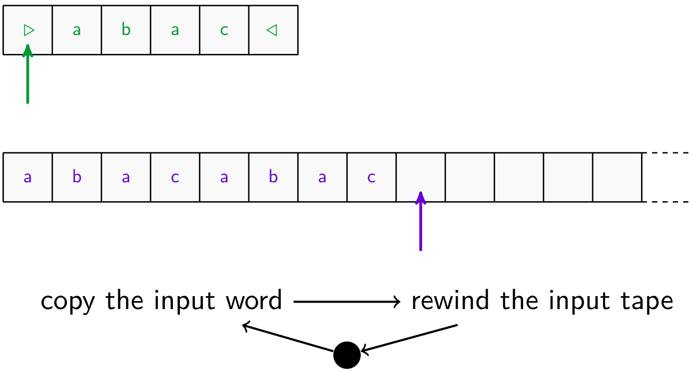
Abilities of two-way transducers



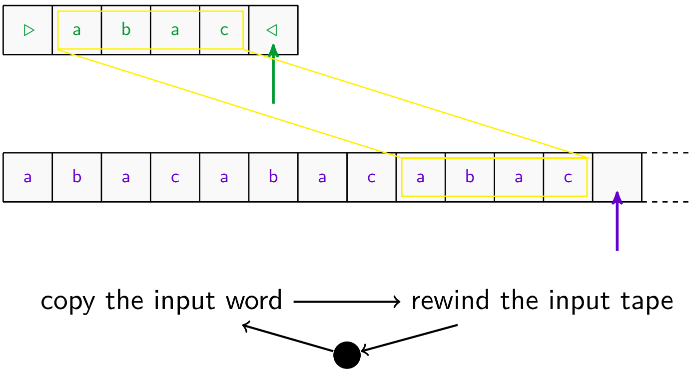
Abilities of two-way transducers



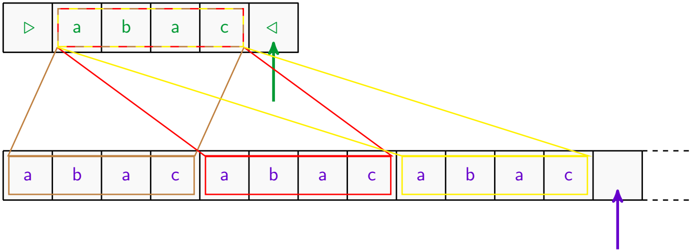
Abilities of two-way transducers



Abilities of two-way transducers



Abilities of two-way transducers



copy the input word ⟶ rewind the input tape

accept and halt with nondeterminism

Abilities of two-way transducers

$\textsc{Square}, \textsc{Powers} \notin \textsc{Rat}$

Hadamard operations

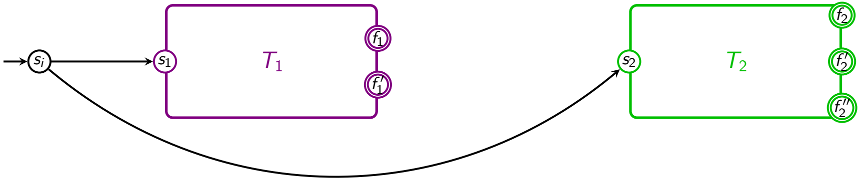
- set union $R_1 \cup R_2$
- Hadamard product
$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

Hadamard operations

- set union $R_1 \cup R_2$
- Hadamard product
$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

Hadamard operations

- set union $R_1 \cup R_2$
- Hadamard product
$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$



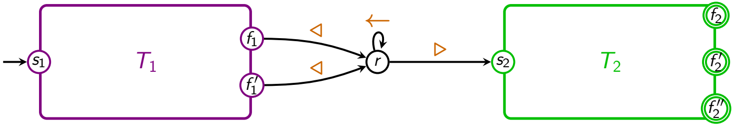
- simulate T_1 or T_2

Hadamard operations

- set union $R_1 \cup R_2$
- Hadamard product

$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$

- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$



- simulate T_1
- rewind the input tape
- simulate T_2

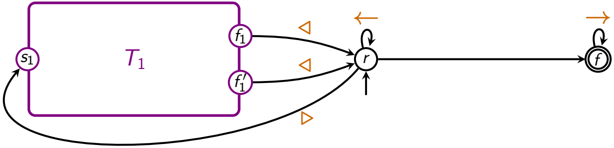
e.g., $\textsc{Square} = \textsc{Identity} \odot \textsc{Identity}$

Hadamard operations

- set union $R_1 \cup R_2$
- Hadamard product

$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$

- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i, (u, v_i) \in R\}$



- repeat
  - simulate T_1
  - rewind the input tape
- or accept nondeterministically

e.g., $\textsc{Powers} = \textsc{Identity}^{\circledast}$

Hadamard operations

- set union $R_1 \cup R_2$
- Hadamard product

$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$

- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i, (u, v_i) \in R\}$

Definition (Had) The class of *Hadamard relations* is the smallest class

- including rational relations
- closed under Hadamard operations

Hadamard operations

- set union $R_1 \cup R_2$
- Hadamard product
$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

Definition (Had) The class of *Hadamard relations* is the smallest class

- including rational relations
- closed under Hadamard operations

$$\textsc{Rat} \subset \textsc{Had} \qquad \textsc{2nft}$$

Hadamard operations

- set union $R_1 \cup R_2$
- Hadamard product
$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i, (u, v_i) \in R\}$

Definition ($\textsc{Had}$) The class of *Hadamard relations* is the smallest class

- including rational relations
- closed under Hadamard operations

$$\textsc{Rat} \subsetneq \textsc{Had} \subsetneq \textsc{2nft}$$

Hadamard operations

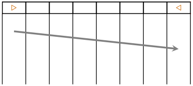
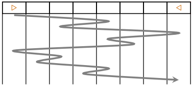
- set union $R_1 \cup R_2$
- Hadamard product

$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$

- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

Definition (HAD) The class of *Hadamard relations* is the smallest class

- including rational relations
- closed under Hadamard operations

machine	one-way		two-way
			
algebra	RAT	HAD	?

Hadamard operations

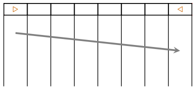
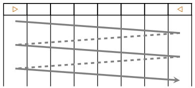
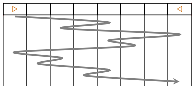
- set union $\quad R_1 \cup R_2$
- Hadamard product

$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$

- Hadamard star $\quad R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

Definition (Had) The class of *Hadamard relations* is the smallest class

- including rational relations
- closed under Hadamard operations

machine	one-way	rotating	two-way
			
algebra	Rat	Had	?

Hadamard operations

- set union $R_1 \cup R_2$
- Hadamard product

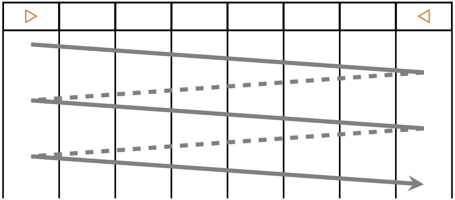
$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$

- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

Definition (Had) The class of *Hadamard relations* is the smallest class

- including rational relations
- closed under Hadamard operations

rotating

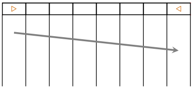
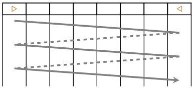
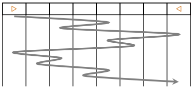


Hadamard operations

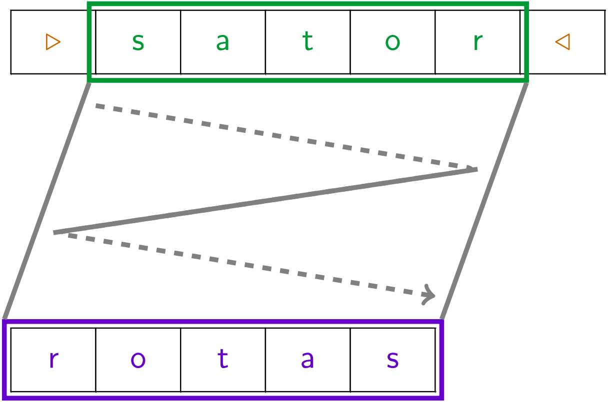
- set union $\qquad R_1 \cup R_2$
- Hadamard product
$$R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$$
- Hadamard star $\qquad R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

Definition (Had) The class of *Hadamard relations* is the smallest class

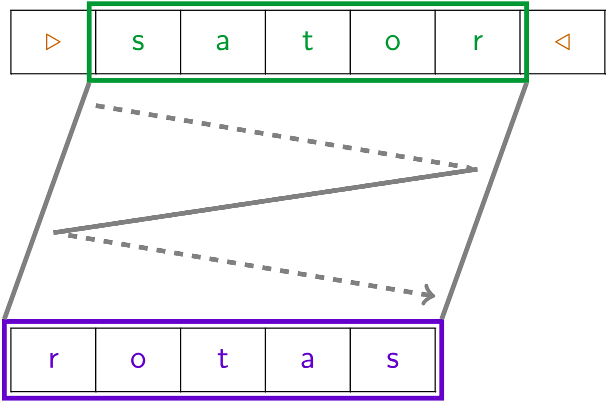
- including rational relations
- closed under Hadamard operations

machine	one-way	rotating	two-way
			
algebra	Rat	Had	?

Mirror



Mirror



$$\textsc{Mirror} \notin \textsc{Had}$$

Mirror operations

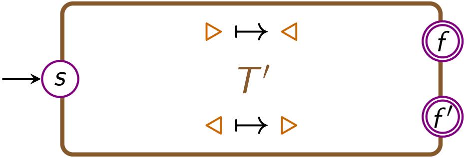
- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Mirror operations

- mirror $\qquad \overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

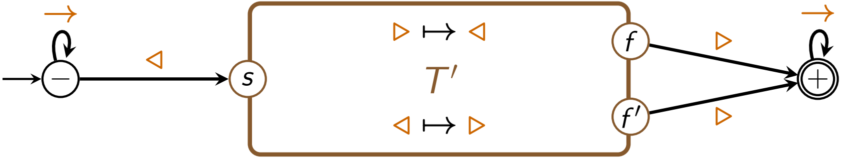
Mirror operations

- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$



Mirror operations

- mirror $\qquad \overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$



Mirror operations

- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Definition ($\textsc{MHad}$) class of *mirror-Hadamard* relations: smallest class

- including rational relations
- closed under Hadamard operations and mirror

Mirror operations

- set union $R_1 \cup R_2$
- Hadamard product $R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$
- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Definition (MHad) class of *mirror-Hadamard* relations: smallest class

- including rational relations
- closed under Hadamard operations and mirror

Mirror operations

- set union $R_1 \cup R_2$
- Hadamard product $R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Definition (MHad) class of *mirror-Hadamard* relations: smallest class

- including rational relations
- closed under Hadamard operations and mirror

Rat $\subset$ Had $\subset$ MHad 2nft

Mirror operations

- set union $R_1 \cup R_2$
- Hadamard product $R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Definition ($\textsc{MHad}$) class of *mirror-Hadamard* relations: smallest class

- including rational relations
- closed under Hadamard operations and mirror

$$\textsc{Rat} \subset \textsc{Had} \subset \textsc{MHad} \subset \textsc{2nft}$$

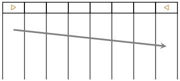
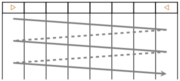
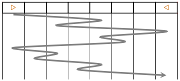
Mirror operations

- set union $R_1 \cup R_2$
- Hadamard product $R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Definition (MHad) class of *mirror-Hadamard* relations: smallest class

- including rational relations
- closed under Hadamard operations and mirror

machine	one-way	rotating		two-way
				
algebra	Rat	Had	MHad	?

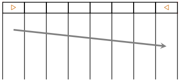
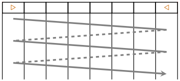
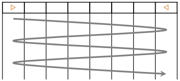
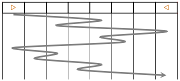
Mirror operations

- set union $R_1 \cup R_2$
- Hadamard product $R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Definition (MHad) class of *mirror-Hadamard* relations: smallest class

- including rational relations
- closed under Hadamard operations and mirror

machine	one-way	rotating	sweeping	two-way
				
algebra	Rat	Had	MHad	?

Mirror operations

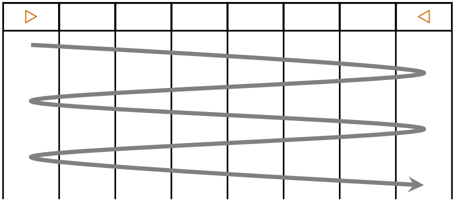
- set union $R_1 \cup R_2$
- Hadamard product $R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Definition (MHad) class of *mirror-Hadamard* relations: smallest class

- including rational relations
- closed under Hadamard operations and mirror

sweeping



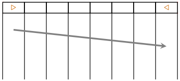
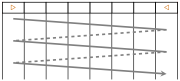
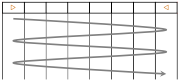
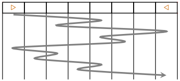
Mirror operations

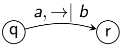
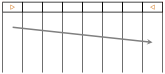
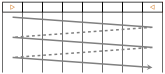
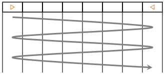
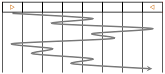
- set union $R_1 \cup R_2$
- Hadamard product $R_1 \odot R_2 = \{(u, v_1 v_2) \mid (u, v_1) \in R_1 \text{ and } (u, v_2) \in R_2\}$
- Hadamard star $R^{\circledast} = \{(u, v_1 \cdots v_k) \mid \forall i,\ (u, v_i) \in R\}$

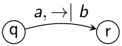
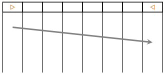
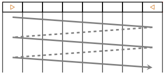
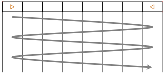
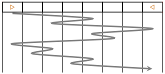
- mirror $\overline{R} = \{(\overline{u}, v) \mid (u, v) \in R\}$

Definition (MHAD) class of *mirror-Hadamard* relations: smallest class

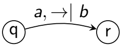
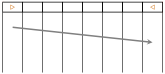
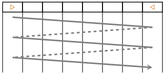
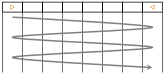
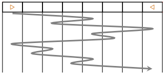
- including rational relations
- closed under Hadamard operations and mirror

machine	one-way	rotating	sweeping	two-way
				
algebra	RAT	HAD	MHAD	?

<table>
<tr><th>transducer</th><th>one-way</th><th>rotating</th><th>sweeping</th><th>two-way</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>general</td><td>Rat</td><td>Had</td><td>MHad</td><td>?</td></tr>
</table>

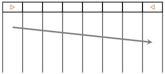
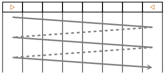
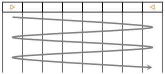
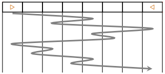
transducer	one-way	rotating	sweeping	two-way
				
general	RAT	HAD	MHAD	?

Study of particular cases

transducer	one-way	rotating	sweeping	two-way
				
general	RAT	HAD	MHAD	?

Study of particular cases

3. Unary cases

transducer	one-way	rotating	sweeping	two-way
$q \xrightarrow{a, \rightarrow \mid b} r$				
general	RAT	HAD	MHAD	?

Study of particular cases

3. Unary cases

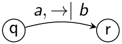
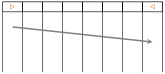
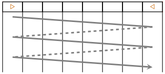
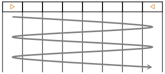
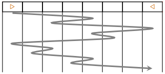
$\#\Sigma = 1$ and/or $\#\Delta = 1$

Contribution of mirror

Proposition: If $\#\Sigma = 1$ or $\#\Delta = 1$ then $\textsc{Had} = \textsc{MHad}$

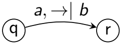
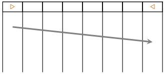
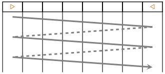
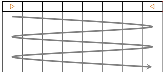
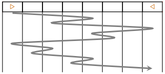
Contribution of mirror

Proposition: If $\#\Sigma = 1$ or $\#\Delta = 1$ then $\textsc{Had} = \textsc{MHad}$

<table>
<tr><th>transducer</th><th>one-way</th><th>rotating</th><th>sweeping</th><th>two-way</th></tr>
<tr><td>
</td><td></td><td></td><td></td><td></td></tr>
<tr><td>general</td><td rowspan="3">Rat</td><td rowspan="3">Had</td><td>MHad</td><td>?</td></tr>
<tr><td>input unary</td><td rowspan="2">Had</td><td>?</td></tr>
<tr><td>output unary</td><td>?</td></tr>
</table>

Contribution of mirror

Proposition: If $\#\Sigma = 1$ or $\#\Delta = 1$ then $\textsc{Had} = \textsc{MHad}$

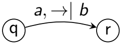
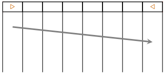
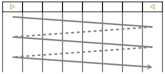
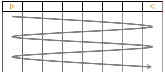
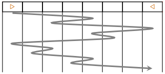
transducer	one-way	rotating	sweeping	two-way
				
general	$\textsc{Rat}$	$\textsc{Had}$	$\textsc{MHad}$	?
input unary	$\textsc{Rat}$	$\textsc{Had}$	$\textsc{Had}$	?
output unary	$\textsc{Rat}$	$\textsc{Had}$	$\textsc{Had}$	?

Theorem: from [Anselmo'90]

When $\#\Delta = 1$, loop-free $2\textsc{nft}$ realize rational transductions

Contribution of mirror

Proposition: If $\#\Sigma = 1$ or $\#\Delta = 1$ then $\textsc{Had} = \textsc{MHad}$

transducer	one-way	rotating	sweeping	two-way
				
general	RAT	HAD	MHAD	?
input unary	RAT	HAD	HAD	?
output unary	RAT	HAD	HAD	?

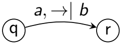
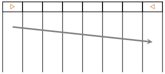
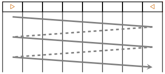
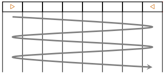
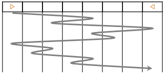
Theorem: from [Anselmo'90]

When $\#\Delta = 1$, loop-free $2\textsc{nft}$ realize rational transductions

- deterministic/unambiguous
- functional

Contribution of mirror

Proposition: If $\#\Sigma = 1$ or $\#\Delta = 1$ then $\textsc{Had} = \textsc{MHad}$

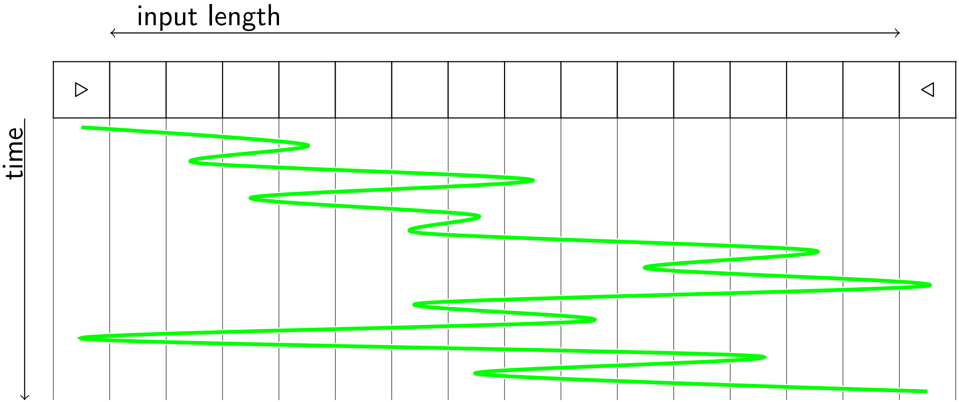
transducer	one-way	rotating	sweeping	two-way
				
general	RAT	HAD	MHAD	?
input unary	RAT	HAD	HAD	?
output unary	RAT	HAD	HAD	?
f output unary	RAT	RAT	RAT	RAT

Theorem: from [Anselmo'90]

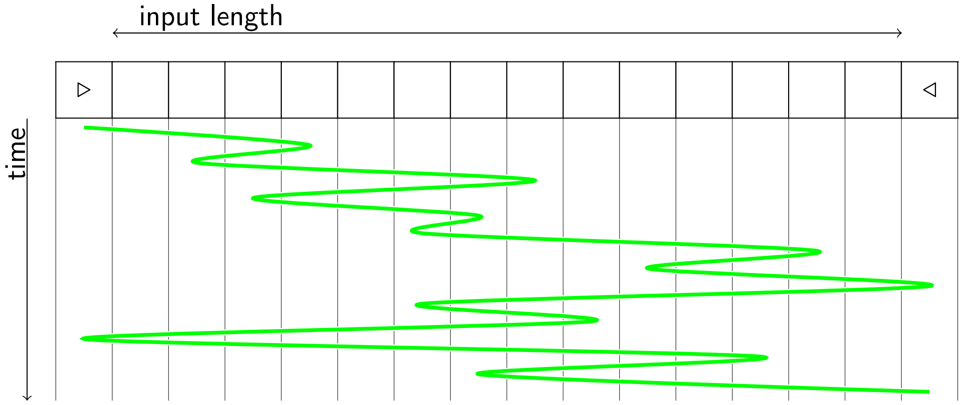
When $\#\Delta = 1$, loop-free 2NFT realize rational transductions

- deterministic/unambiguous
- functional

From 2-way to 1-way automata

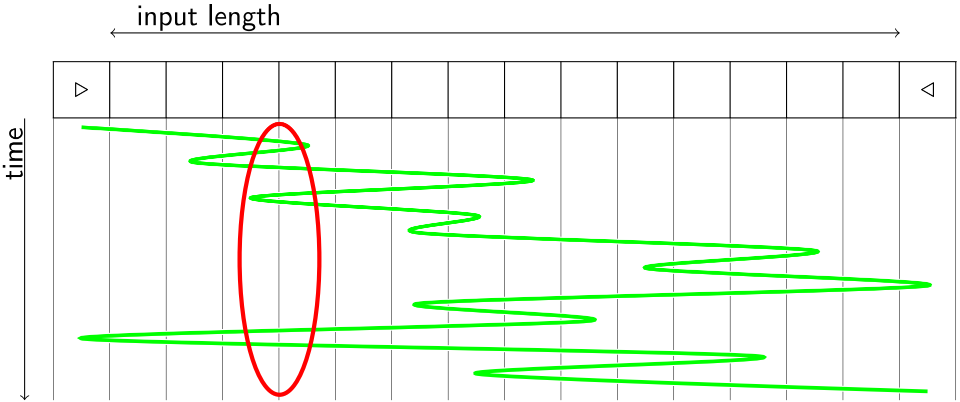


From 2-way to 1-way automata



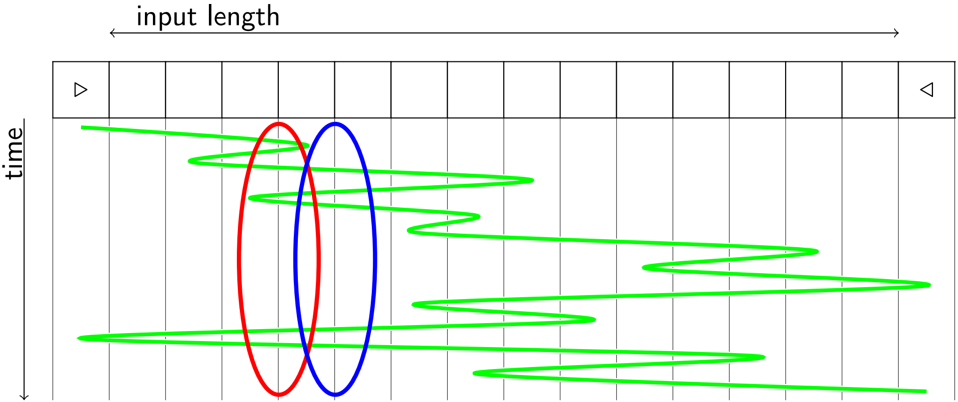
1. every accepted word admits a loop-free accepting computation

From 2-way to 1-way automata



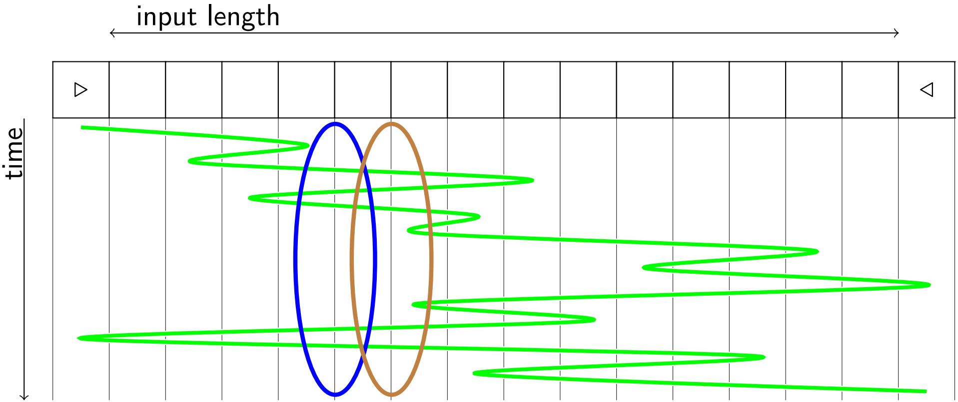
1. every accepted word admits a loop-free accepting computation

From 2-way to 1-way automata



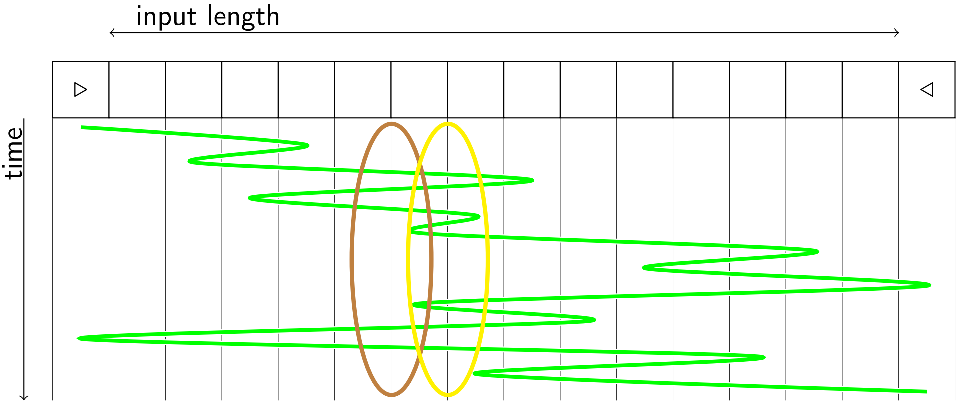
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



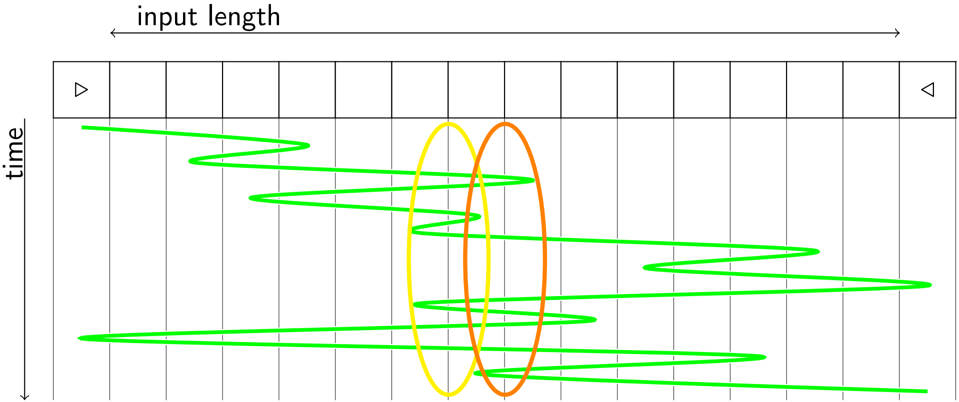
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



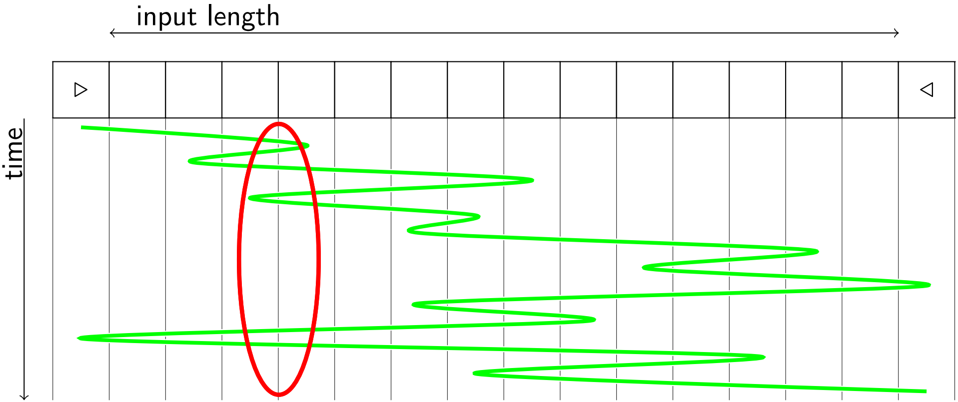
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



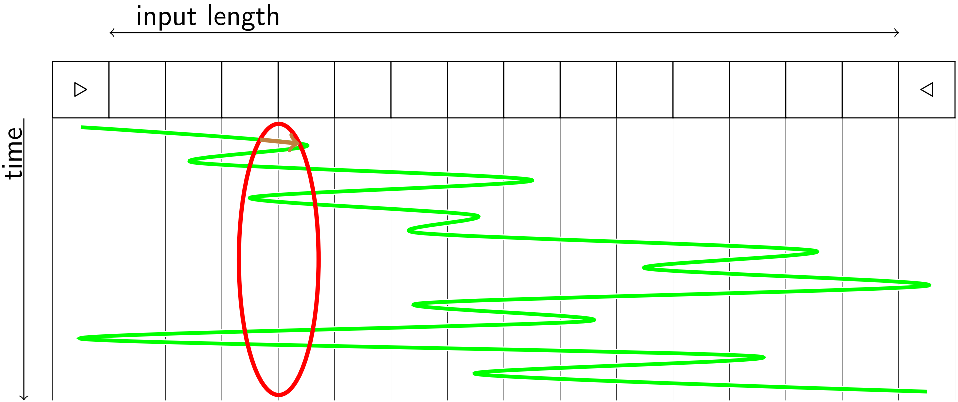
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



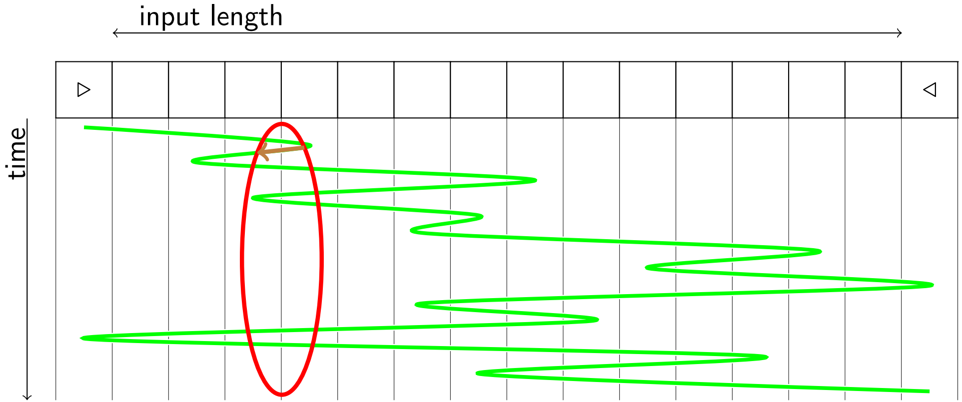
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



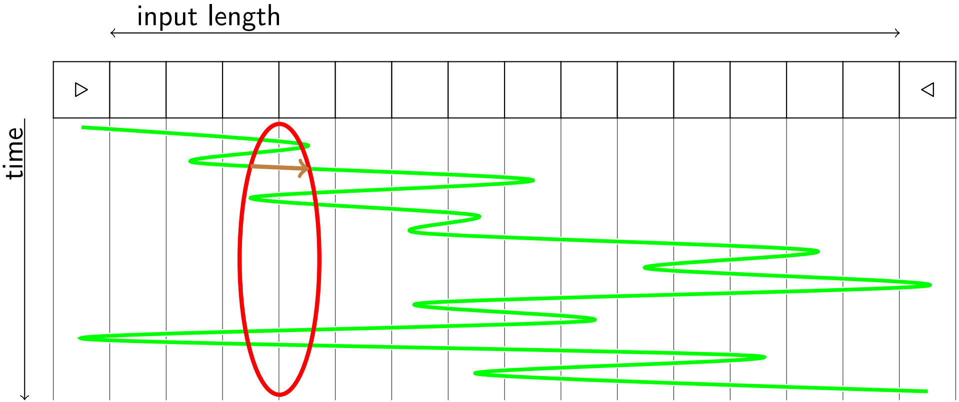
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



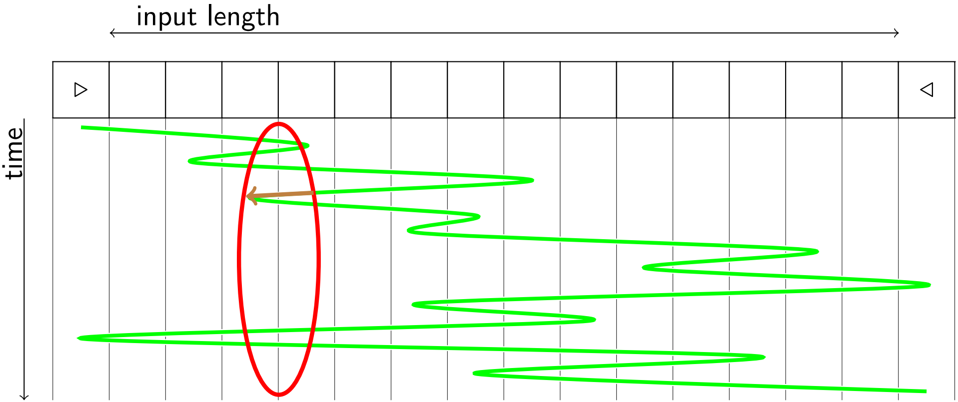
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



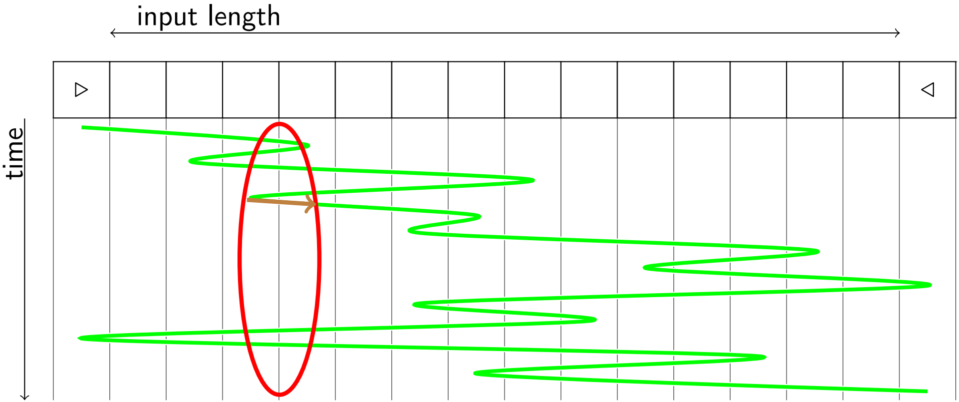
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



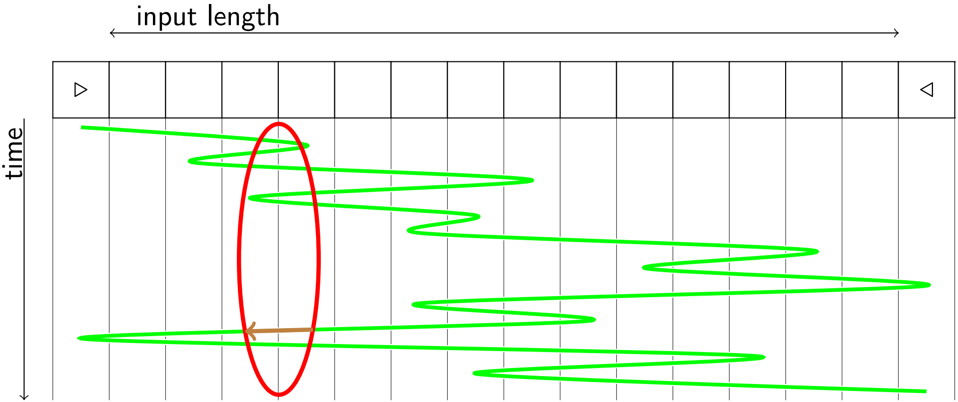
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



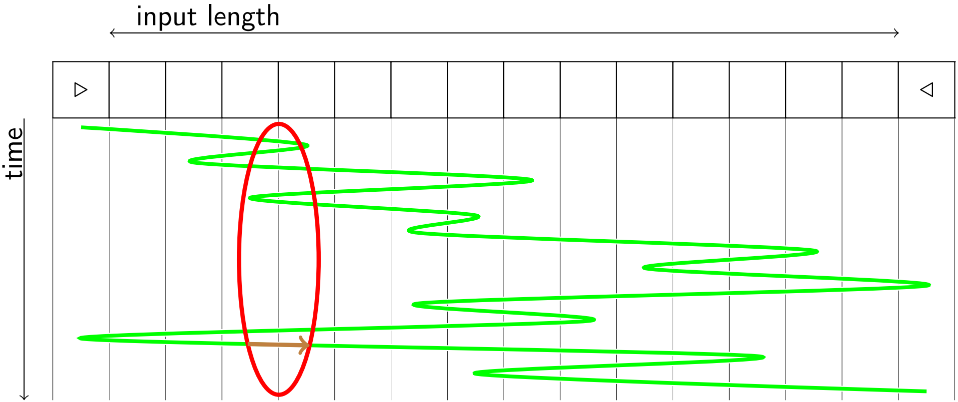
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



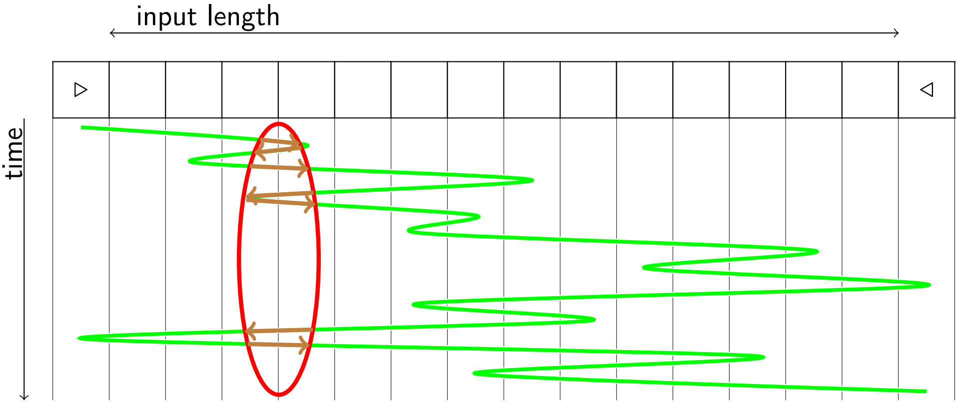
1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

From 2-way to 1-way automata



1. every accepted word admits a loop-free accepting computation
2. the successor relation of crossing sequences is locally testable

Both unary case $\Sigma = \Delta = \{a\}$

- Examples:
  - $\textsc{uIdentity} = \{(a^n, a^n) \mid n \in \mathbb{N}\}$ $\in \textsc{Rat}$
  - $\textsc{uSquare} = \{(a^n, a^{2n}) \mid n \in \mathbb{N}\} = \textsc{uIdentity} \odot \textsc{uIdentity}$ $\in \textsc{Rat}$
  - $\textsc{uPowers} = \left\{(a^n, a^{kn}) \mid k, n \in \mathbb{N}\right\} = \textsc{uIdentity}^{\circledast}$ $\notin \textsc{Rat}$

Both unary case $\Sigma = \Delta = \{a\}$

- Examples:
  - $\textsc{uIdentity} = \{(a^n, a^n) \mid n \in \mathbb{N}\}$ $\in \textsc{Rat}$
  - $\textsc{uSquare} = \{(a^n, a^{2n}) \mid n \in \mathbb{N}\} = \textsc{uIdentity} \odot \textsc{uIdentity}$ $\in \textsc{Rat}$
  - $\textsc{uPowers} = \left\{(a^n, a^{kn}) \mid k, n \in \mathbb{N}\right\} = \textsc{uIdentity}^{\circledast}$ $\notin \textsc{Rat}$

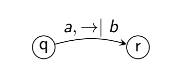
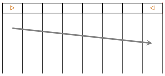
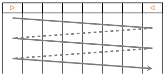
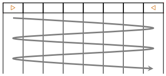
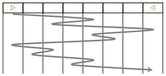
Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Both unary case $\Sigma = \Delta = \{a\}$

Examples:

- uIDENTITY $= \{(a^n, a^n) \mid n \in \mathbb{N}\}$ $\in$ RAT
- uSQUARE $= \{(a^n, a^{2n}) \mid n \in \mathbb{N}\} =$ uIDENTITY $\odot$ uIDENTITY $\in$ RAT
- uPOWERS $= \left\{(a^n, a^{kn}) \mid k, n \in \mathbb{N}\right\} =$ uIDENTITY$^{\circledast}$ $\notin$ RAT

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, 2NFT = HAD

transducer	one-way	rotating	sweeping	two-way
				
general	RAT	HAD	MHAD	?
input unary	RAT	HAD	HAD	?
output unary	RAT	HAD	HAD	?
both unary	RAT	HAD	HAD	HAD

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Proof: fix a unary transducer T realizing R

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

- each family of hits define a relation $R_{(q,s),(q',s')}$

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

- each family of hits define a relation $R_{(q,s),(q',s')}$

  e.g., $(p, \triangleright)$-to-$(q, \triangleleft)$, $(p, \triangleleft)$-to-$(q, \triangleleft)$

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\text{NFT} = \text{HAD}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

- each family of hits define a relation $R_{(q,s),(q',s')}$

  e.g., $(p, \triangleright)$-to-$(q, \triangleleft)$, $(p, \triangleleft)$-to-$(q, \triangleleft)$

- R is a combination of $R_{(q,s),(q',s')}$'s using Hadamard operations

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

- each family of hits define a relation $R_{(q,s),(q',s')}$

  e.g., $(p, \triangleright)$-to-$(q, \triangleleft)$, $(p, \triangleleft)$-to-$(q, \triangleleft)$

  each $R_{(q,s),(q',s')} \in \textsc{Rat}$

- R is a combination of $R_{(q,s),(q',s')}$'s using Hadamard operations

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

- each family of hits define a relation $R_{(q,s),(q',s')}$

  e.g., $(p, \triangleright)$-to-$(q, \triangleleft)$, $(p, \triangleleft)$-to-$(q, \triangleleft)$

  each $R_{(q,s),(q',s')} \in \textsc{Rat}$

- R is a combination of $R_{(q,s),(q',s')}$'s using Hadamard operations

  $\Longrightarrow \in \textsc{Had}$ $\square$

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\text{NFT} = \textsc{Had}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

- each family of hits define a relation $R_{(q,s),(q',s')}$

  e.g., $(p,\triangleright)$-to-$(q,\triangleleft)$, $(p,\triangleleft)$-to-$(q,\triangleleft)$

- simulate each hit family with a one-way transducer

  $\Longrightarrow$ each $R_{(q,s),(q',s')} \in \textsc{Rat}$

- R is a combination of $R_{(q,s),(q',s')}$'s using Hadamard operations

  $\Longrightarrow \in \textsc{Had}$ $\square$

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\textsc{nft} = \textsc{Had}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

- each family of hits define a relation $R_{(q,s),(q',s')}$

  e.g., $(p,\rhd)$-to-$(q,\lhd)$, $(p,\lhd)$-to-$(q,\lhd)$

- simulate each hit family with a one-way transducer

  $\Longrightarrow$ each $R_{(q,s),(q',s')} \in \textsc{Rat}$

- R is a combination of $R_{(q,s),(q',s')}$'s using Hadamard operations

  $\Longrightarrow \in \textsc{Had}$ $\square$

Proof sketch

Theorem: [Choffrut, G.'14] When $\#\Sigma = 1$ and $\#\Delta = 1$, $2\text{NFT} = \text{HAD}$

Proof: fix a unary transducer T realizing R

- decompose computations in *hits*

  hit $\equiv$ *computation path between 2 successive visits of endmarkers*

- each family of hits define a relation $R_{(q,s),(q',s')}$

  e.g., $(p, \triangleright)$-to-$(q, \triangleleft)$, $(p, \triangleleft)$-to-$(q, \triangleleft)$

- simulate each hit family with a one-way transducer

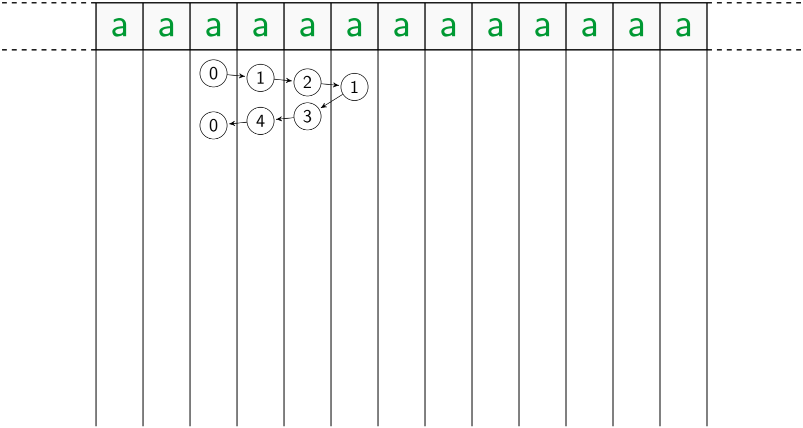
  $\Longrightarrow$ each $R_{(q,s),(q',s')} \in \text{RAT}$

  - commutative outputs
  - deal with *central loops*

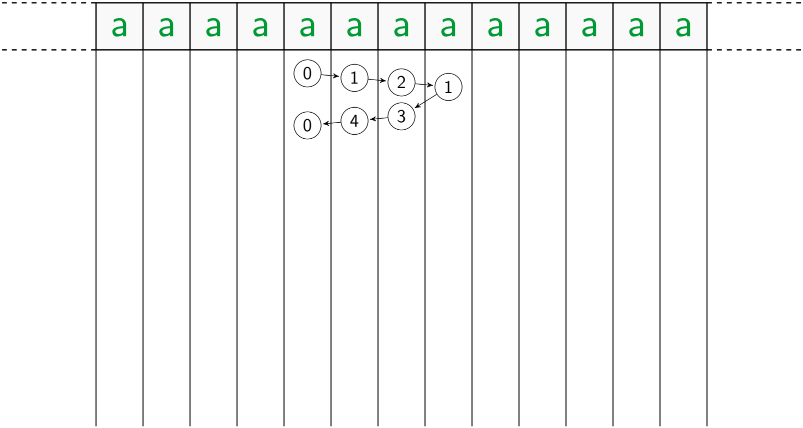
- R is a combination of $R_{(q,s),(q',s')}$'s using Hadamard operations

  $\Longrightarrow \in \text{HAD}$ $\square$

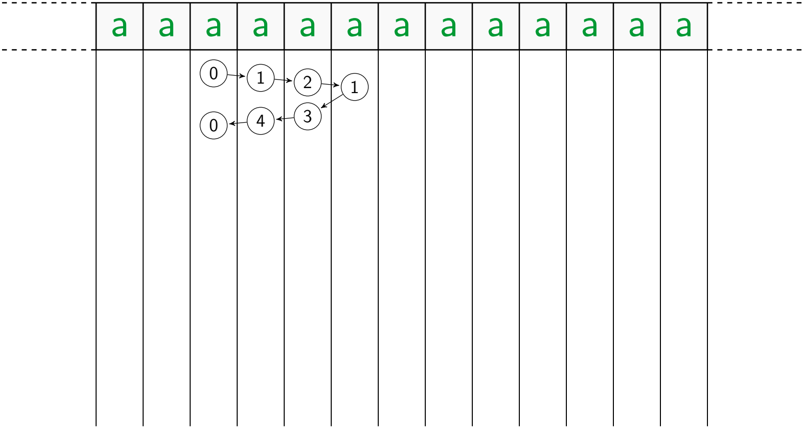
Unary central loops



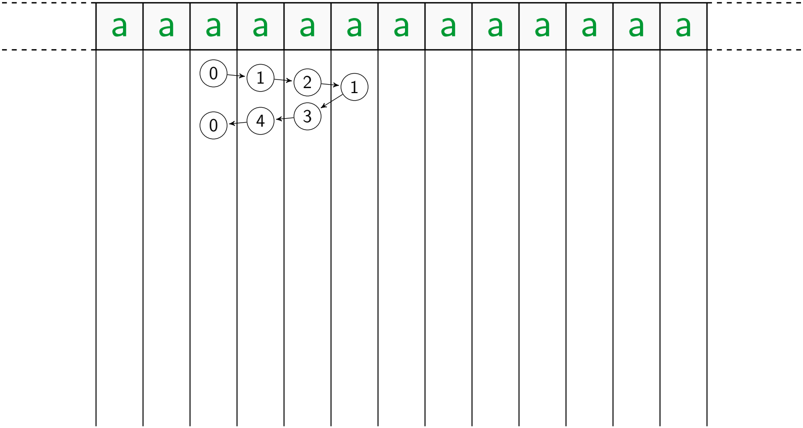
Unary central loops



Unary central loops

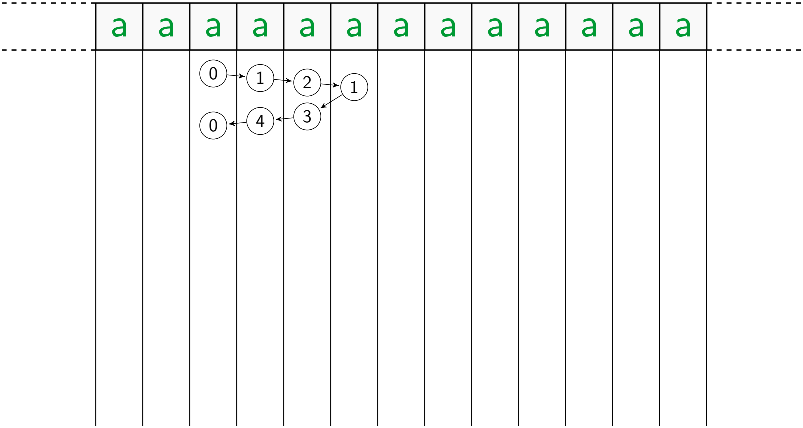


Unary central loops



- each language $L_q \subseteq a^*$ of outputs of all central loops around state q satisfies ${L_q}^* = L_q$

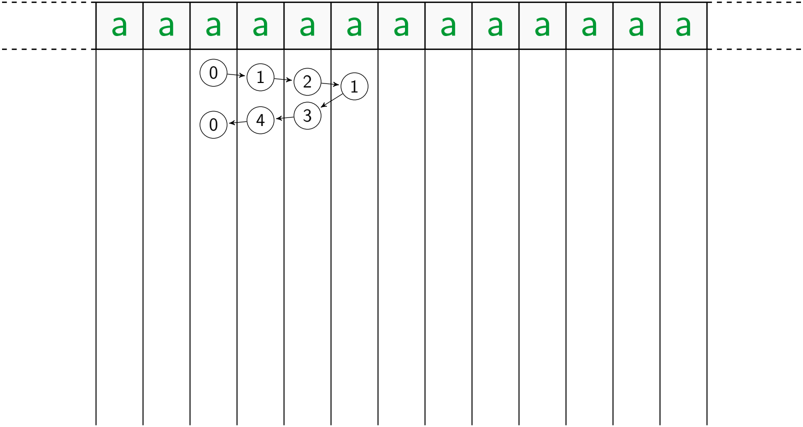
Unary central loops



- each language $L_q \subseteq a^*$ of outputs of all central loops around state q

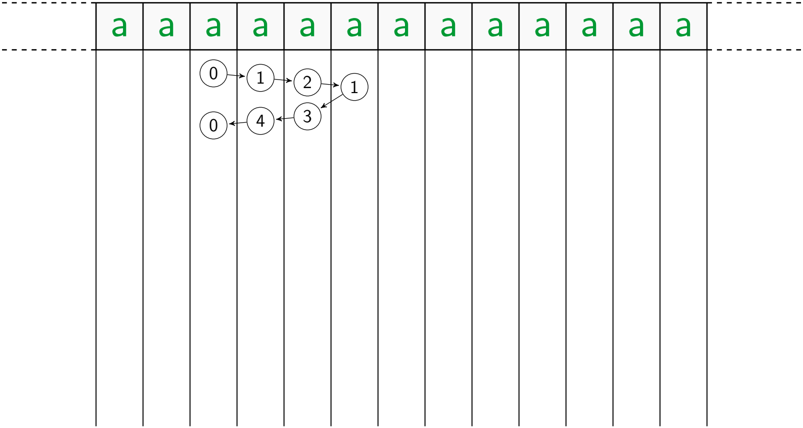
  satisfies ${L_q}^* = L_q \implies L_q \in \mathrm{RAT}$

Unary central loops



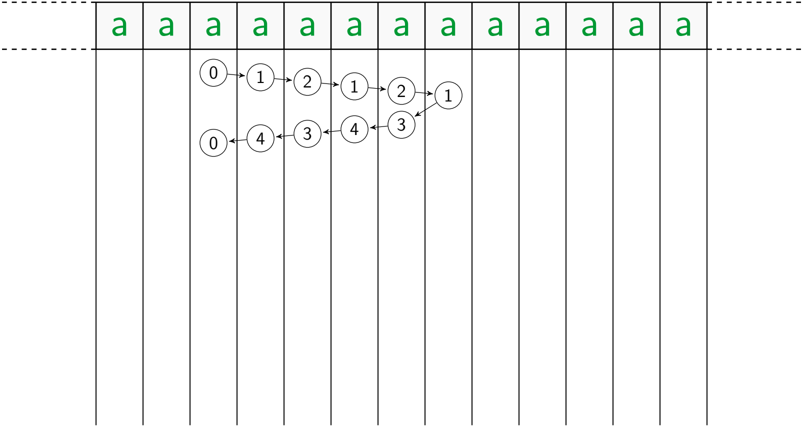
- each language $L_q \subseteq a^*$ of outputs of all central loops around state q satisfies ${L_q}^* = L_q \implies L_q \in \mathrm{RAT} \implies L$ is finitely generated

Unary central loops



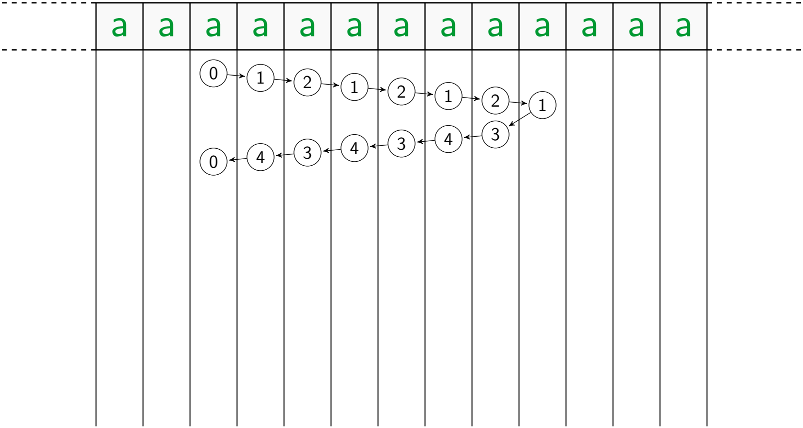
- each language $L_q \subseteq a^*$ of outputs of all central loops around state q satisfies ${L_q}^* = L_q \implies L_q \in \mathrm{RAT} \implies L$ is finitely generated
- $\exists N \in \mathbb{N}$ such that a window of size N is sufficient to generate each L_q

Unary central loops



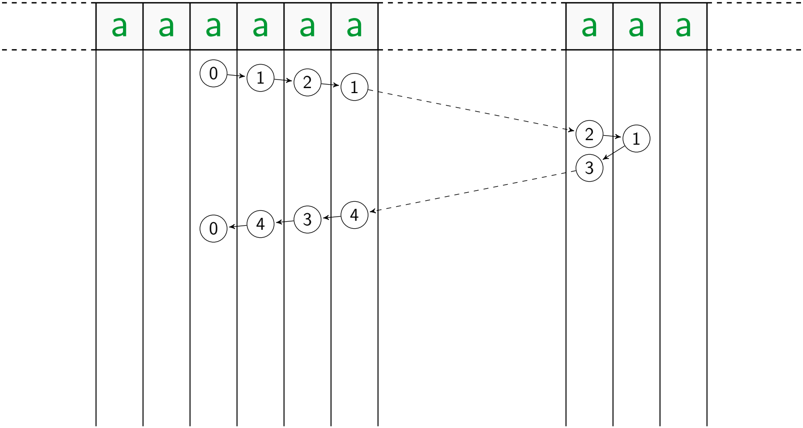
- each language $L_q \subseteq a^*$ of outputs of all central loops around state q satisfies ${L_q}^* = L_q \implies L_q \in \mathrm{RAT} \implies L$ is finitely generated
- $\exists N \in \mathbb{N}$ such that a window of size N is sufficient to generate each L_q

Unary central loops



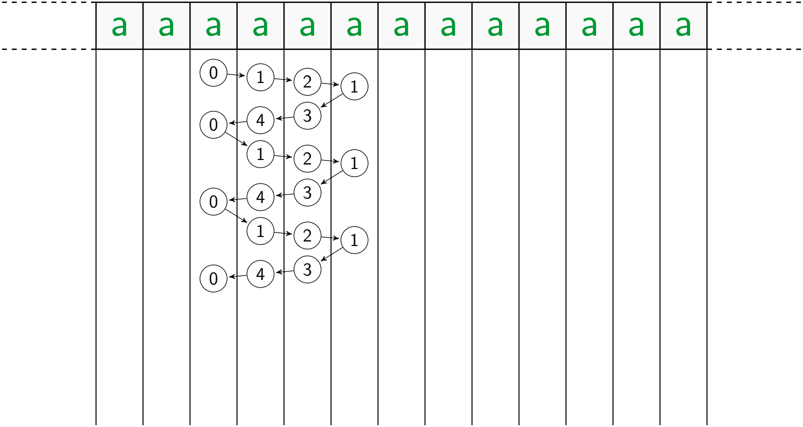
- each language $L_q \subseteq a^*$ of outputs of all central loops around state q satisfies ${L_q}^* = L_q \implies L_q \in \textsc{Rat} \implies L$ is finitely generated
- $\exists N \in \mathbb{N}$ such that a window of size N is sufficient to generate each L_q

Unary central loops



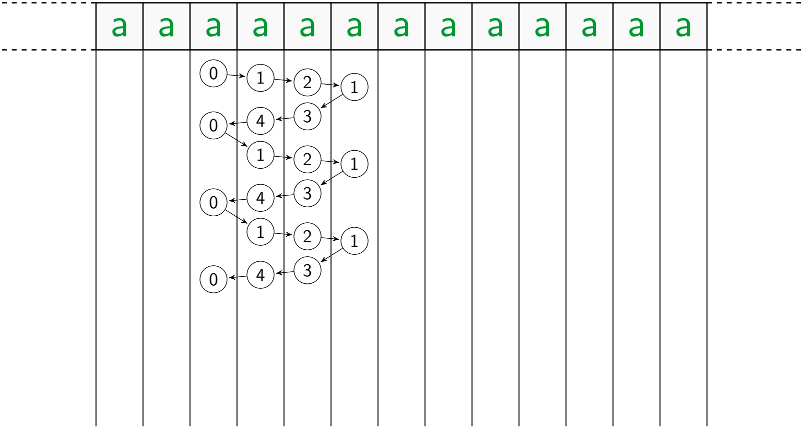
- each language $L_q \subseteq a^*$ of outputs of all central loops around state q satisfies ${L_q}^* = L_q \implies L_q \in \text{RAT} \implies L$ is finitely generated
- $\exists N \in \mathbb{N}$ such that a window of size N is sufficient to generate each L_q

Unary central loops



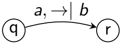
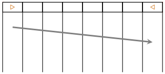
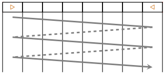
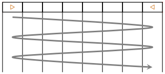
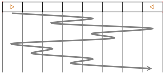
- each language $L_q \subseteq a^*$ of outputs of all central loops around state q satisfies ${L_q}^* = L_q \implies L_q \in \mathrm{RAT} \implies L$ is finitely generated
- $\exists N \in \mathbb{N}$ such that a window of size N is sufficient to generate each L_q

Unary central loops

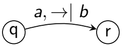
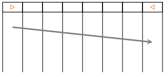
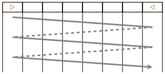
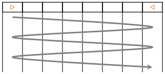
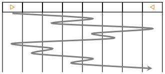


- each language $L_q \subseteq a^*$ of outputs of all central loops around state q
  satisfies ${L_q}^* = L_q \implies L_q \in \mathrm{RAT} \implies L$ is finitely generated
- $\exists N \in \mathbb{N}$ such that a window of size N is sufficient to generate each L_q
- **Open problem:** bound N?

Relax assumptions?

<table>
<tr><th>transducer</th><th>one-way</th><th>rotating</th><th>sweeping</th><th>two-way</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>general</td><td rowspan="4">RAT</td><td rowspan="4">HAD</td><td>MHAD</td><td>?</td></tr>
<tr><td>input unary</td><td rowspan="3"></td><td>?</td></tr>
<tr><td>output unary</td><td>?</td></tr>
<tr><td>both unary</td><td></td></tr>
</table>

Relax assumptions?

<table>
<tr><th>transducer</th><th>one-way</th><th>rotating</th><th>sweeping</th><th>two-way</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>general</td><td rowspan="4">Rat</td><td rowspan="4" colspan="3">Had</td></tr>
<tr><td>input unary</td></tr>
<tr><td>output unary</td></tr>
<tr><td>both unary</td></tr>
</table>

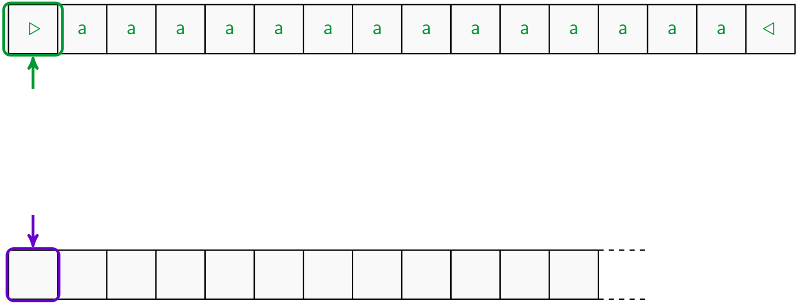
Theorem: [G.'16]

When $\#\Sigma > 1$, $\textsc{Had} \subset 2\textsc{nft}$

When $\#\Delta > 1$, $\textsc{Had} \subset 2\textsc{nft}$

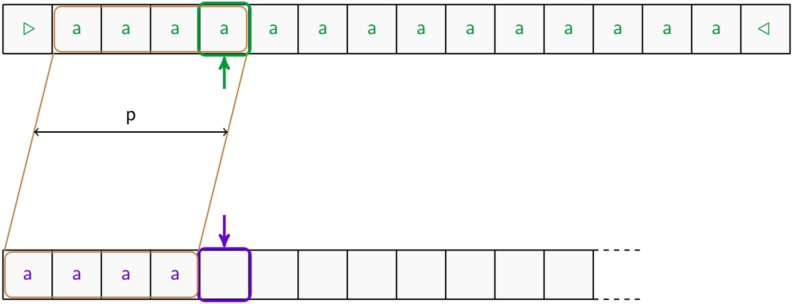
Input-unary witness

$$\textsc{RLPrefix} = \{(a^n, a^m b^m) \mid n, m \in \mathbb{N},\ m \leq n\}$$



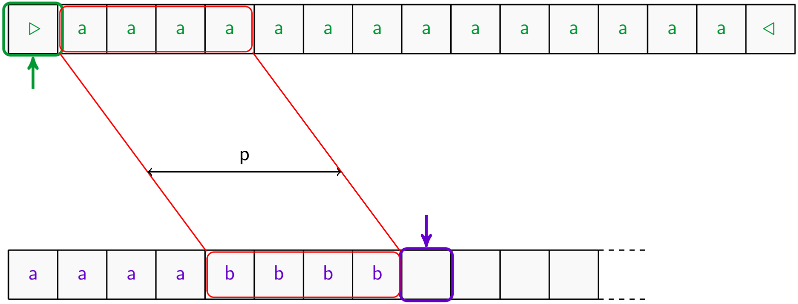
Input-unary witness

$$\textsc{RLPrefix} = \{(a^n, a^m b^m) \mid n, m \in \mathbb{N},\ m \leq n\}$$



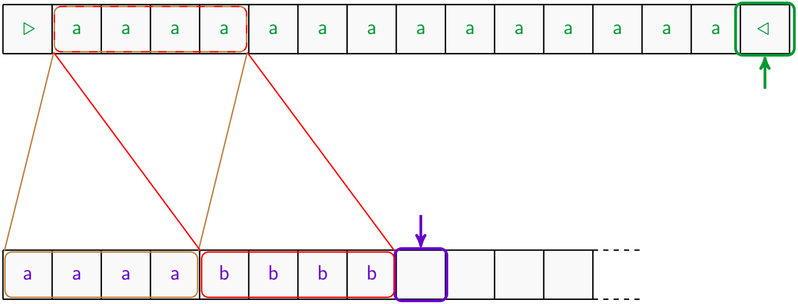
Input-unary witness

$$\textsc{RLPrefix} = \{(a^n, a^m b^m) \mid n, m \in \mathbb{N},\ m \leq n\}$$



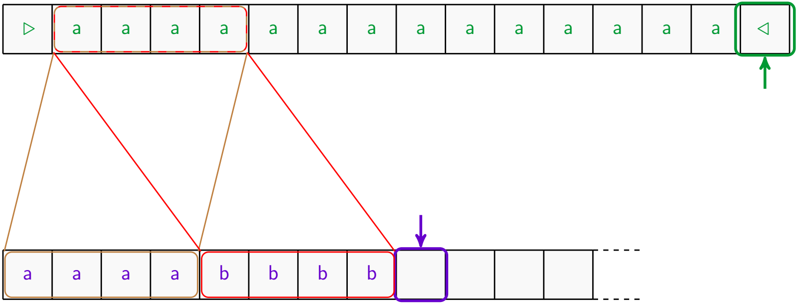
Input-unary witness

$$\text{RLP}\textsc{refix} = \{(a^n, a^m b^m) \mid n, m \in \mathbb{N},\ m \leq n\}$$



Input-unary witness

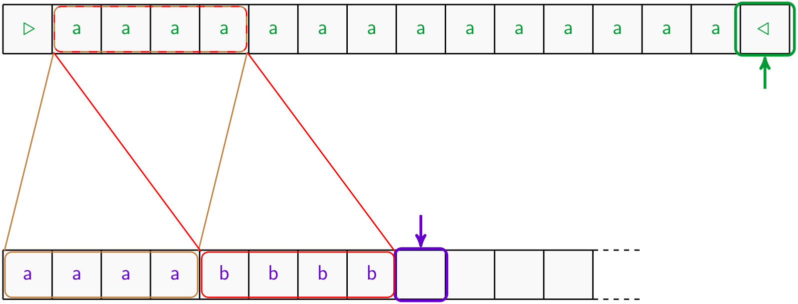
$$\text{RLPrefix} = \{(a^n, a^m b^m) \mid n, m \in \mathbb{N},\ m \leq n\}$$



$$\text{RLPrefix} \notin \text{Had}$$

Input-unary witness

$$\text{RLPrefix} = \{(a^n, a^m b^m) \mid n, m \in \mathbb{N},\ m \leq n\}$$

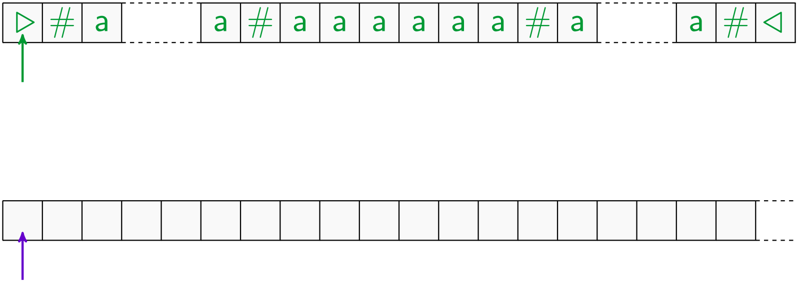


$$\text{RLPrefix} \notin \text{Had}$$

- Remark: Had is not closed under componentwise concatenation

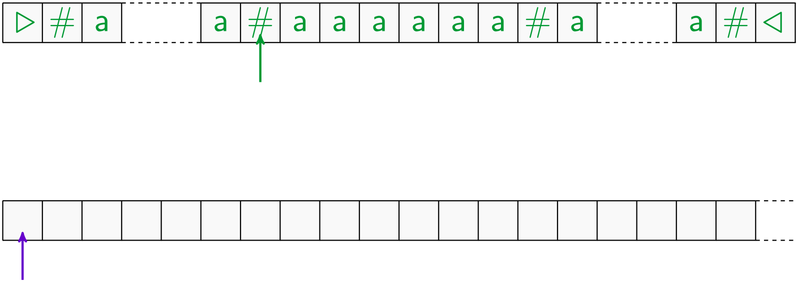
Output-unary witness

$$\textsc{Mult1Block} = \left\{ (u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u \right\}$$



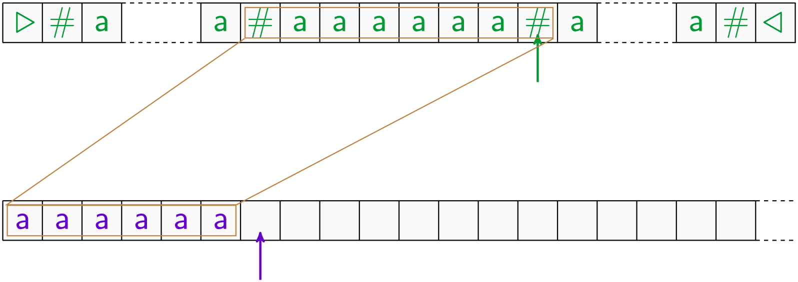
Output-unary witness

$$\textsc{Mult1Block} = \Big\{(u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u\Big\}$$



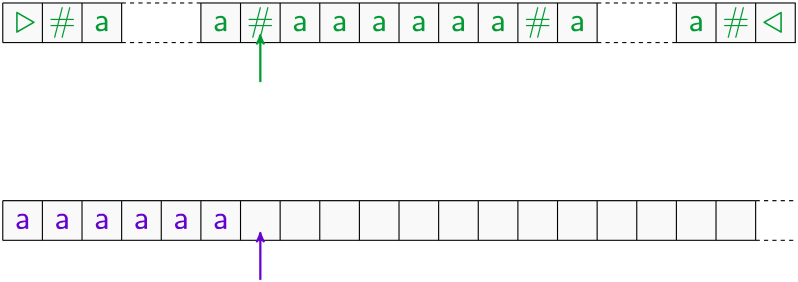
Output-unary witness

$$\textsc{Mult1Block} = \left\{ (u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u \right\}$$



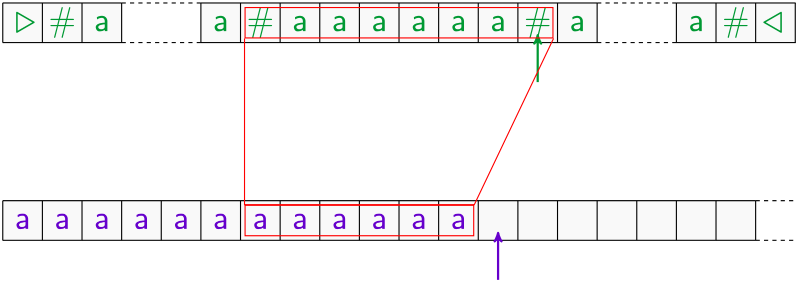
Output-unary witness

$$\textsc{Mult1Block} = \left\{ (u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u \right\}$$



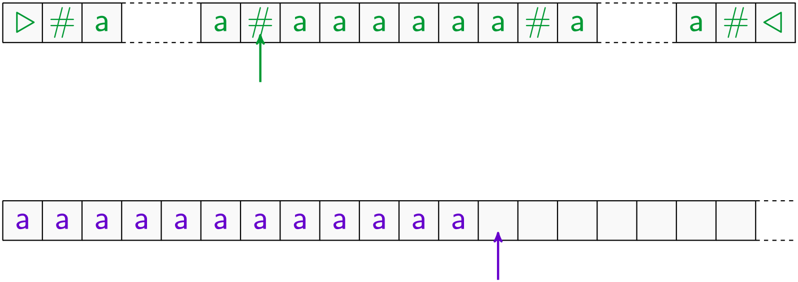
Output-unary witness

$$\textsc{Mult1Block} = \left\{(u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u\right\}$$



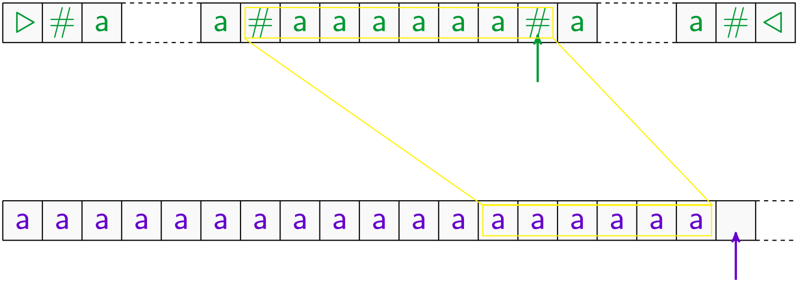
Output-unary witness

$$\textsc{Mult1Block} = \left\{(u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u\right\}$$



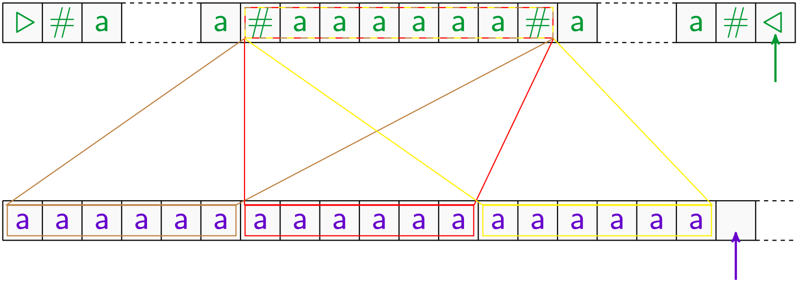
Output-unary witness

$$\textsc{Mult1Block} = \left\{ (u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u \right\}$$



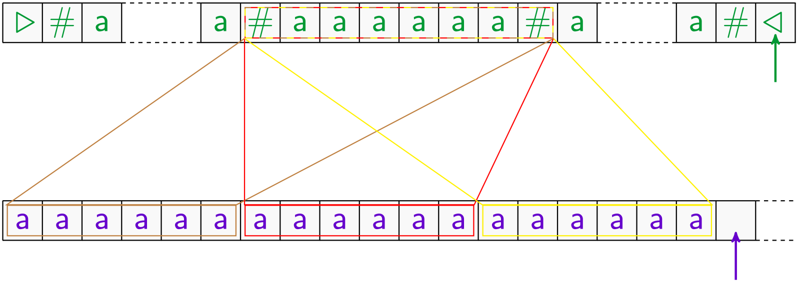
Output-unary witness

$$\textsc{Mult1Block} = \Big\{ (u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u \Big\}$$



Output-unary witness

$$\textsc{Mult1Block} = \Big\{ (u, a^{kn}) \mid u \in \{a, \sharp\}^*,\ k, n \in \mathbb{N},\ \text{and } \sharp a^n \sharp \text{ is a factor of } u \Big\}$$



$$\textsc{Mult1Block} \notin \textsc{Had}$$

- Remark: $\textsc{Had}$ is not closed under componentwise concatenation

Two-way versus componentwise concatenation

Proposition: 2NFT is not closed under componentwise concatenation and Kleene star

Two-way versus componentwise concatenation

Proposition: 2NFT is not closed under componentwise concatenation and Kleene star

Proposition: 2NFT is closed under **unambiguous** componentwise concatenation and **unambiguous** Kleene star

Two-way versus componentwise concatenation

Proposition: 2NFT is not closed under componentwise concatenation and Kleene star

Proposition: 2NFT is closed under **unambiguous** componentwise concatenation and **unambiguous** Kleene star

- Remark: these operations are used in the characterization of regular functions through regular combinators

[Alur *et al.*'14, Baudru&Reynier'18, Dave *et al.*'18]

Two-way versus componentwise concatenation

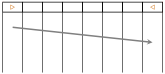
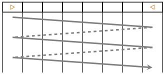
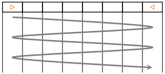
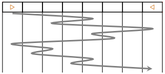
Proposition: 2NFT is not closed under
componentwise concatenation and Kleene star

Proposition: 2NFT is closed under
unambiguous componentwise concatenation
and **unambiguous** Kleene star

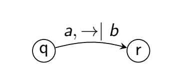
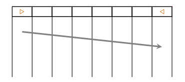
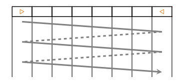
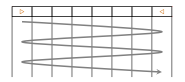
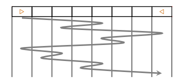
- Remark: these operations are used in the characterization of regular functions through regular combinators
  [Alur *et al.*'14, Baudru&Reynier'18, Dave *et al.*'18]

but they are still not sufficient for the non-functional case

Conclusion

transducer	one-way	rotating	sweeping	two-way
$q \xrightarrow{a, \rightarrow \mid b} r$				
general	RAT	HAD	MHAD	?
input unary	RAT	HAD	HAD	?
output unary	RAT	HAD	HAD	?
both unary	RAT	HAD	HAD	HAD

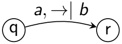
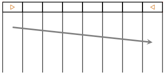
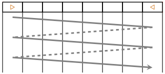
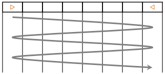
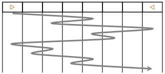
Conclusion

transducer	one-way	rotating	sweeping	two-way
				
general	RAT	HAD	MHAD	?
input unary	RAT	HAD	HAD	?
output unary	RAT	HAD	HAD	?
both unary	RAT	HAD	HAD	HAD

Abilities arising from nondeterminism:

- loop
  - output an unbounded word in one step *e.g.*, $\textsc{Erase}^{-1}$
  - loop over some portion of the input word *e.g.*, $\textsc{Powers}$

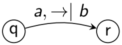
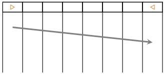
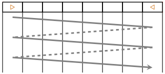
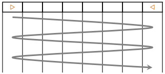
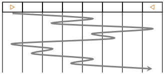
Conclusion

<table>
<tr><th>transducer</th><th>one-way</th><th>rotating</th><th>sweeping</th><th>two-way</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>general</td><td rowspan="4">RAT</td><td rowspan="4">HAD</td><td>MHAD</td><td>?</td></tr>
<tr><td>input unary</td><td rowspan="3"></td><td>?</td></tr>
<tr><td>output unary</td><td>?</td></tr>
<tr><td>both unary</td><td></td></tr>
</table>

Abilities arising from nondeterminism:

- loop
  - output an unbounded word in one step *e.g.*, ERASE^{-1}
  - loop over some portion of the input word *e.g.*, POWERS
- nondeterministically select some positions *e.g.*, RLPREFIX

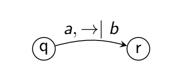
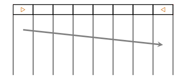
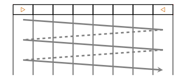
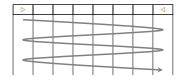
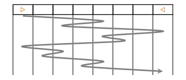
Conclusion

transducer	one-way	rotating	sweeping	two-way
				
general	Rat	Had	MHad	?
input unary	Rat	Had	Had	?
output unary	Rat	Had	Had	?
both unary	Rat	Had	Had	Had

Abilities arising from nondeterminism:

- loop
  - output an unbounded word in one step *e.g.*, Erase^{-1}
  - loop over some portion of the input word *e.g.*, Powers
- nondeterministically select some positions *e.g.*, RLPrefix

Conclusion

transducer	one-way	rotating	sweeping	two-way
				
general	RAT	HAD	MHAD	?
input unary	RAT	HAD	HAD	?
output unary	RAT	HAD	HAD	?
both unary	RAT	HAD	HAD	HAD

Abilities arising from nondeterminism:

- loop
  - output an unbounded word in one step *e.g.*, ERASE^{-1}
  - loop over some portion of the input word *e.g.*, POWERS
- nondeterministically select some positions *e.g.*, RLPREFIX

Thank you