

Automata based verification over linearly ordered data domains

to be presented at STACS '11

Luc Segoufin

INRIA and ENS Cachan


Szymon Toruńczyk

University of Warsaw

Motivation

Motivation

Given a system which refers to a database and data values, verify properties of the system

Car Hire Home Page Return to Booking Help

Car Hire Offers Avis Locations Products & Services Join Avis Preferred Business Rentals Help and Contacts UK Fleet Home Delivery Newsletter

1 When and Where 2 Your Car Choice 3 Price and Extras 4 Checkout

Avis car hire quote and booking

Find pickup location by
Search for

City / town name
edinburgh

Return location

Same as pickup location
[Change](#)

Rental Start Date/Time

24 September 2010 09 00

Rental End Date/Time

28 September 2010 09 00

Number of days

4

Avis Worldwide Discount (AWD) No.

[View, modify or cancel a booking](#)

Sign in

[Create an account](#)

[Sign in](#) to your Online account/Avis Preferred

*Car hire delivered
to your front door*

*Click here to book
Home Delivery!*



UK from £17 per day

Car hire from £17 per day

Enjoy great discounts this Autumn. Get on the road and explore the wonderful places the UK has to offer. Book Now!



Europe from £15 per day

Car hire from £15 per day

Explore Europe this Autumn with great discounts which will take you further so you can see more. Book Now!



Car hire from £17 per day

Explore the world by taking advantage of our amazing Autumn Sale prices. Book Now!

Motivation

Given a system which refers to a database and data values, verify properties of the system

The screenshot shows the AVIS website's booking interface. At the top, the AVIS logo is on the left, and navigation links like 'Car Hire', 'Home Page', 'Return to Booking', and 'Help' are on the right. Below this is a secondary navigation bar with links such as 'Car Hire Offers', 'Avis Locations', 'Products & Services', 'Join Avis Preferred', 'Business Rentals', 'Help and Contacts', 'UK Fleet', 'Home Delivery', and 'Newsletter'. A progress bar indicates four steps: '1 When and Where' (highlighted in red), '2 Your Car Choice', '3 Price and Extras', and '4 Checkout'. On the right side, there are links for 'Sign in' and 'Create an account', with a note to 'Sign in to your Online account/Avis Preferred'.

The main booking form is titled 'Avis car hire quote and booking'. It includes the following fields and options:

- Find pickup location by Search for:** A text input field containing 'edinburgh' and a 'Go' button. A green callout box labeled 'input values' points to this field.
- Return location:** A radio button for 'Same as pickup location' (selected) and a 'Change' link.
- Rental Start Date/Time:** A date field with '24 September 2010' and time dropdowns for '09' and '00'.
- Rental End Date/Time:** A date field with '28 September 2010' and time dropdowns for '09' and '00'.
- Number of days:** A text input field containing '4'.
- Avis Worldwide Discount (AWD) No.:** An empty text input field.

At the bottom of the form is a red button labeled 'Get a quote' and a link 'View, modify or cancel a booking'. A green callout box labeled 'change state' points to the 'Get a quote' button.

Below the form, there are promotional banners for 'UK from £17 per day', 'Europe from £15 per day', and another 'UK from £17 per day' offer, each featuring a car image and descriptive text.


input values

*Car hire delivered
to your front door*

change state

Motivation

Given a system which refers to a database and data values, verify properties of the system



[Car Hire](#) [Home Page](#) [Return to Booking](#) [Help](#)

[Car Hire Offers](#) [Avis Locations](#) [Products & Services](#) [Join Avis Preferred](#) [Business Rentals](#) [Help and Contacts](#) [UK Fleet](#) [Home Delivery](#) [Newsletter](#)

[1 When and Where](#) **[2 Your Car Choice](#)** [3 Price and Extras](#) [4 Checkout](#)

You are viewing vehicles available at Edinburgh Airport

[Small](#) [Medium](#) [Large](#) [Select Series](#)

Price	Size	Number of Luggage
	Small : Economy (Example of this range : Peugeot 207)	
	Best price £155.65 per rental	Hide Info
	Available to book now	
Vehicle Features Air Bag - Driver Short Wheel Base	Radio/Cassette	Driver age requirements You must be at least 23 years old to hire this vehicle If you are under 25 years old a Young Driver Surcharge will apply Young Driver surcharge is £11/day + VAT; max £110
		Credit card requirements The number of Credit Cards required when you pick up this vehicle is: 1.
	Medium : Economy (Example of this range : Nissan Note 1.4)	
	Best price £181.37 per rental	More Info
	Available to book now	

Sign in

[Create an account](#)
[Sign in](#) to your Online account/Avis Preferred


Your Booking

- 1 WHEN AND WHERE** [Change](#)
Pickup : Edinburgh Airport 23/09/2010 19:00
Return : Edinburgh Airport 28/09/2010 19:00
Rental Days : 5
- 2 YOUR CAR CHOICE**
- 3 PRICE AND EXTRAS**
- 4 CHECKOUT**

[Empty Basket](#)

Motivation

Given a system which refers to a database and data values, verify properties of the system



[Car Hire](#) [Home Page](#) [Return to Booking](#) [Help](#)

[Car Hire Offers](#) [Avis Locations](#) [Products & Services](#) [Join Avis Preferred](#) [Business Rentals](#) [Help and Contacts](#) [UK Fleet](#) [Home Delivery](#) [Newsletter](#)

[1 When and Where](#) **[2 Your Car Choice](#)** [3 Price and Extras](#) [4 Checkout](#)

You are viewing vehicles available at Edinburgh Airport

[Small](#) [Medium](#) [Large](#) [Select Series](#)

Price	Size	Number of Luggage
	Small : Economy (Example of this range : Peugeot 207)	
	Best price <u>£155.65 per rental</u>	Hide Info
Vehicle Features Air Bag - Driver Short Wheel Base	Radio/Cassette	Driver age requirements You must be at least 23 years old to hire this vehicle. If you are under 25 years old a Young Driver Surcharge will apply. Young Driver surcharge is £11/day + VAT; max £110
		Credit card requirements The number of Credit Cards required when you pick up this vehicle is: 1.
	Medium : Economy (Example of this range : Nissan Note 1.4)	
	Best price <u>£181.37 per rental</u>	More Info

Sign in [Create an account](#)
[Sign in](#) to your Online account/Avis Preferred

Your Booking

1 WHEN AND WHERE [Change](#)
Pickup : Edinburgh Airport 23/09/2010 19:00
Return : Edinburgh Airport 28/09/2010 19:00
Rental Days : 5

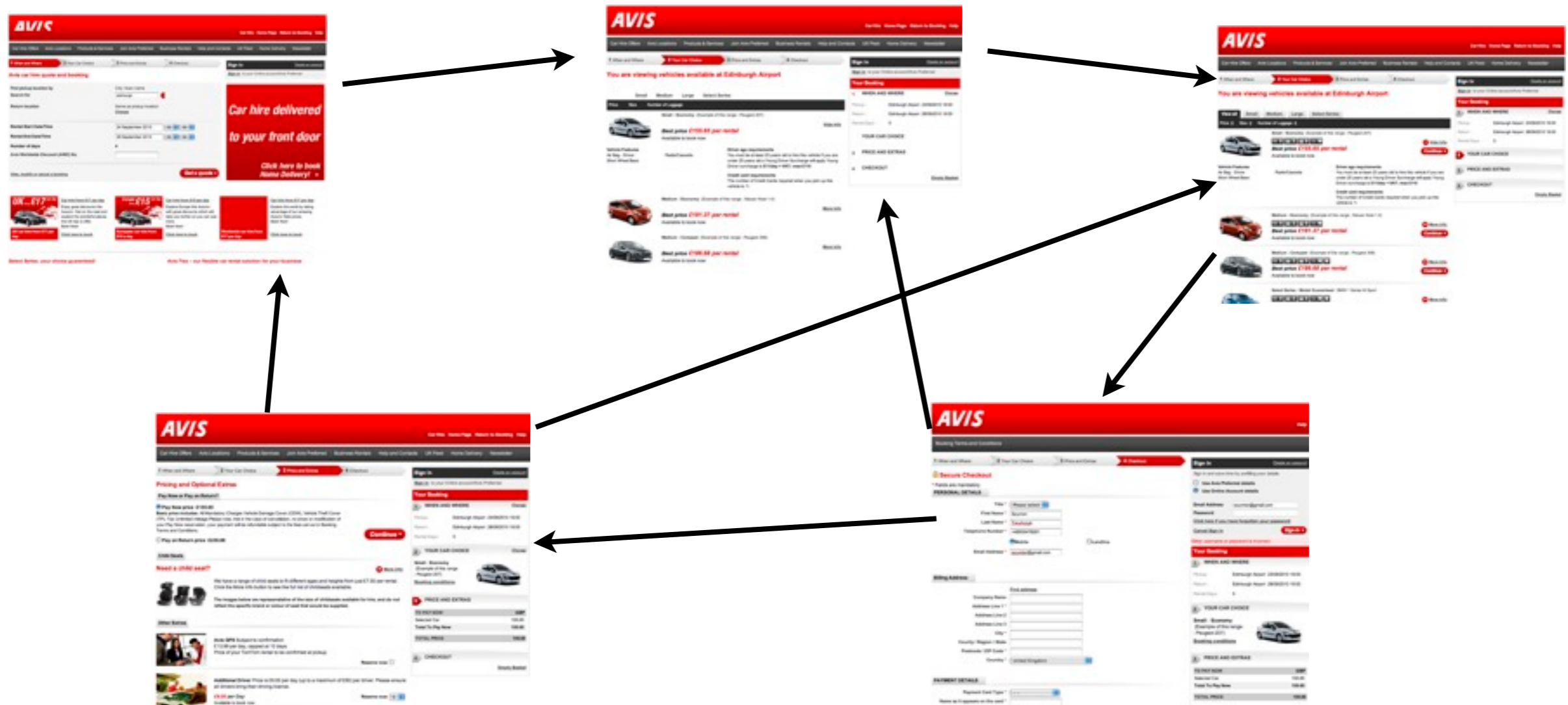
YOUR CAR CHOICE

3 PRICE AND EXTRAS

4 CHECKOUT [Empty Basket](#)

Motivation

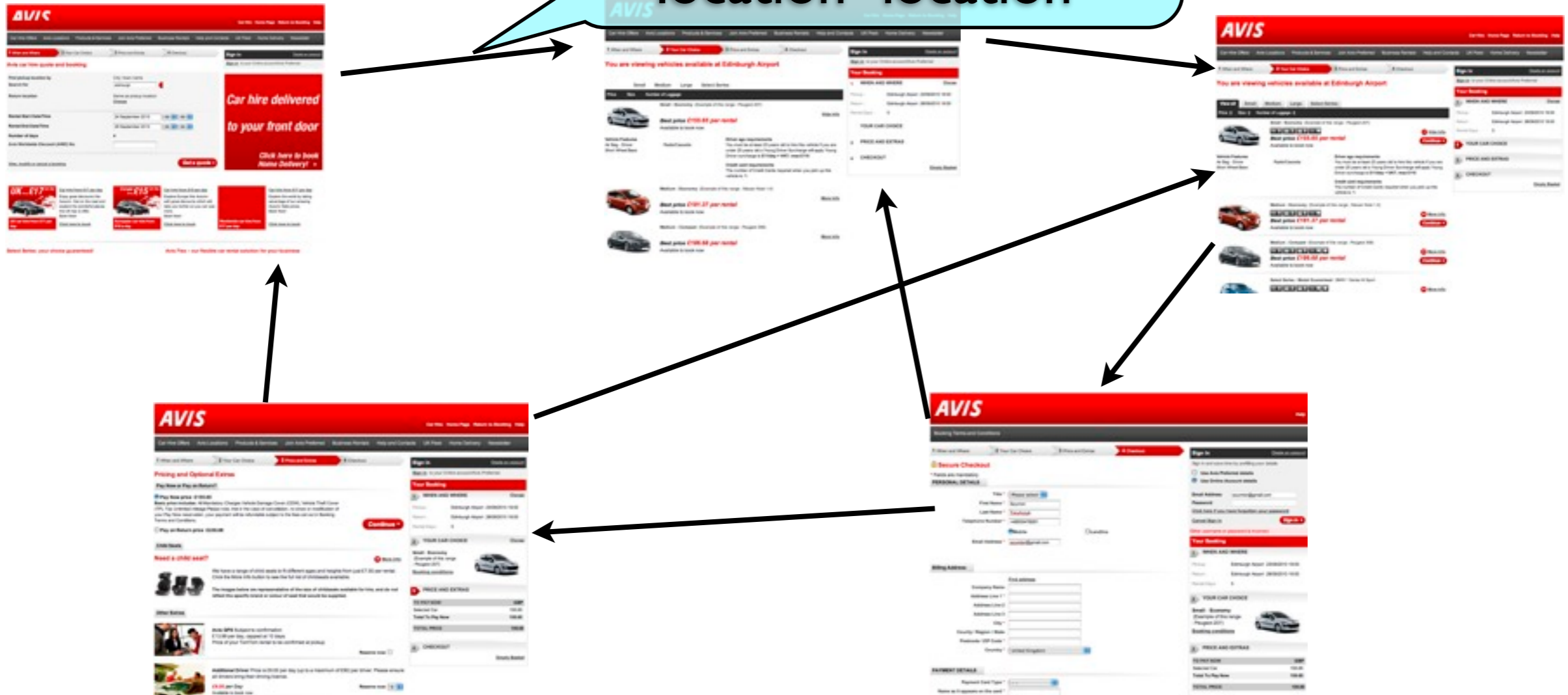
Given a system which refers to a database and data values, verify properties of the system



Motivation

Given a system which refers to a database and data values, verify properties of the system

`start_date=start_date'`
`end_date=end_date'`
`location=location'`

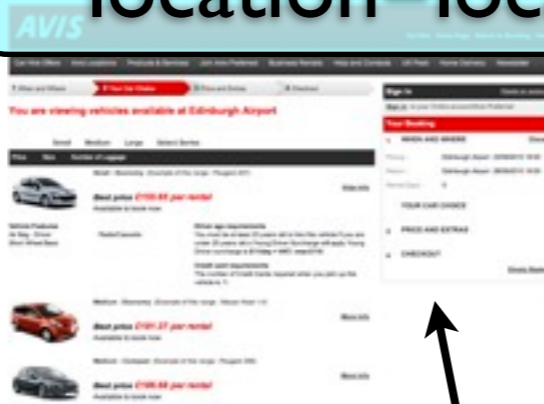


Motivation

Given a system which refers to a database and data values, verify properties of the system

$\text{start_date}=\text{start_date}'$
 $\text{end_date}=\text{end_date}'$
 $\text{location}=\text{location}'$

$\text{pricemin} \leq \text{price}$
 $\leq \text{pricemax}$



Motivation

Given a system which refers to a database and data values, verify properties of the system

$\text{start_date}=\text{start_date}'$
 $\text{end_date}=\text{end_date}'$
 $\text{location}=\text{location}'$

$\text{pricemin} \leq \text{price}$
 $\leq \text{pricemax}$

$\text{CAR}(\text{car_id}, \text{price})$



Motivation

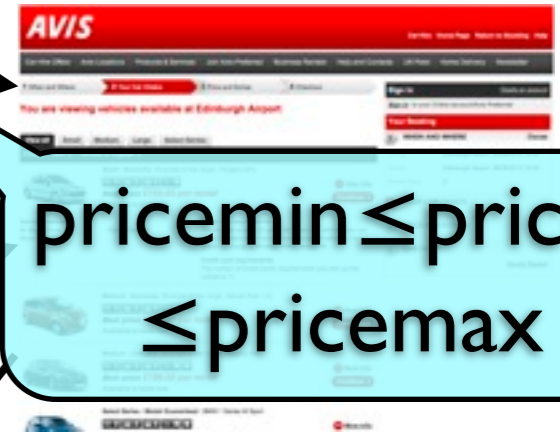
Given a system which refers to a database and data values, verify properties of the system

$\text{start_date}=\text{start_date}'$
 $\text{end_date}=\text{end_date}'$
 $\text{location}=\text{location}'$

$\text{pricemin} \leq \text{price}$
 $\leq \text{pricemax}$

$\text{CAR}(\text{car_id}, \text{price})$

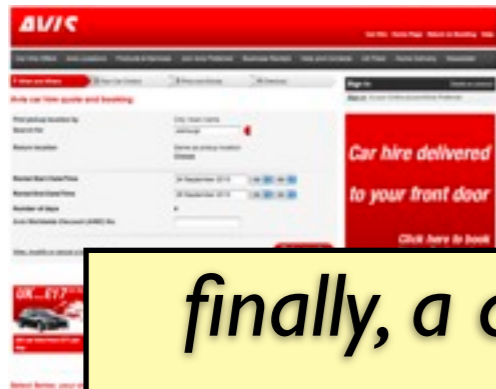
$\text{PAYMENT}(\text{user_id}, \text{car_id},$
 $\text{price}, \text{order_id})$



Motivation

Given a system which refers to a database and data values, verify properties of the system

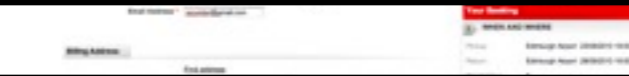
`start_date=start_date'`
`end_date=end_date'`
`location=location'`



`pricemin < price`

finally, a car is rented and

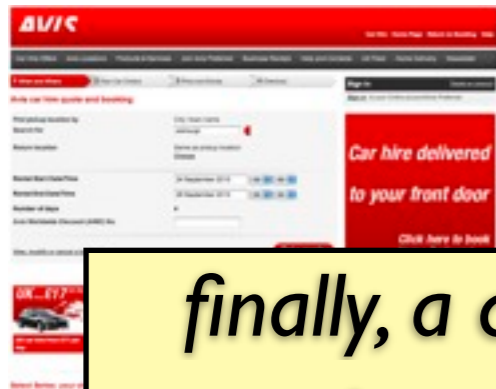
`PAYMENT(user_id, car_id,
price, order_id)`



Motivation

Given a system which refers to a database and data values, verify properties of the system

`start_date=start_date'`
`end_date=end_date'`
`location=location'`



`pricemin < price`

finally, a car is rented and

- no payment was recieved*

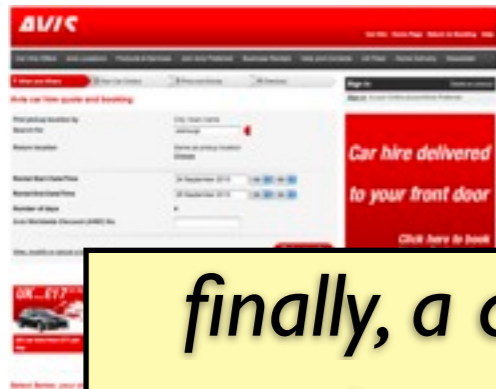


`PAYMENT(user_id, car_id, price, order_id)`

Motivation

Given a system which refers to a database and data values, verify properties of the system

`start_date=start_date'`
`end_date=end_date'`
`location=location'`



`pricemin < price`

finally, a car is rented and

- *no payment was recieved*
- *a payment was recieved but payed=0*

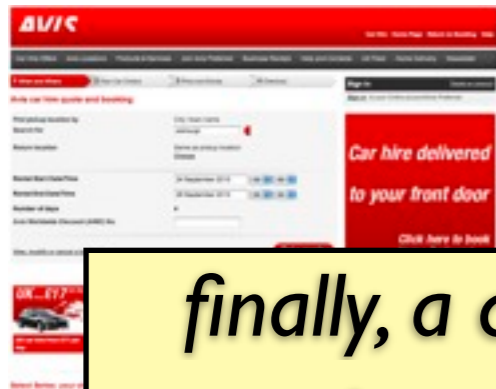


`PAYMENT(user_id,car_id,
price, order_id)`

Motivation

Given a system which refers to a database and data values, verify properties of the system

`start_date=start_date'`
`end_date=end_date'`
`location=location'`



`pricemin < price`

finally, a car is rented and

- *no payment was recieved*
- *a payment was recieved but payed=0*
- *there was a payment, but payed<100 & end_date>2050*

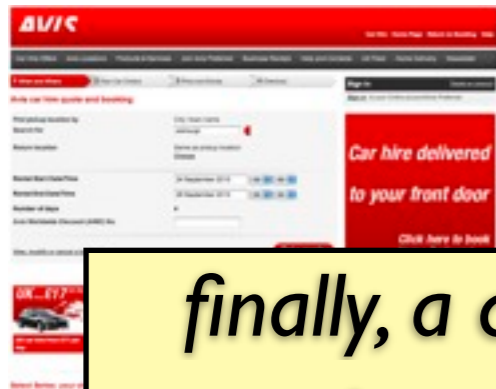


`PAYMENT(user_id,car_id,
price, order_id)`

Motivation

Given a system which refers to a database and data values, verify properties of the system

`start_date=start_date'`
`end_date=end_date'`
`location=location'`



`pricemin < price`

finally, a car is rented and

- *no payment was recieved*
- *a payment was recieved but payed=0*
- *there was a payment, but payed < 100 & end_date > 2050*
- *there was a payment, but payed < car_price*



`PAYMENT(user_id, car_id,
price, order_id)`

Outline

1. Define automata model
2. Analyze automata over dense orders
3. Analyze automata over discrete orders
4. Add database and infinite runs

Extended automaton

a definition that captures timed automata, or vector addition systems

D – fixed domain

Q – finite set of states

X – finite set of variables

$D^X = \{ v: X \rightarrow D \}$ – space of variable valuations

$Q \times D^X$ – space of configurations

restricted
form

$\delta \subseteq (Q \times D^X) \times (Q \times D^X)$ – set of allowed transitions

$I \subseteq (Q \times D^X)$ – set of initial configurations

$F \subseteq (Q \times D^X)$ – set of final configurations

Examples

- *Vector Addition System:* $D = \mathbb{N}$

$$(q, v) \xrightarrow[v' = v + w]{} (q', v')$$

- *Timed Automata:* $D = \mathbb{R}$

$$(q, v) \xrightarrow[c_2' = 0]{c_1 < 2} (q', v')$$

- *(Lossy) channel system:* $D = \{a+b\}^*$

$$(q, v) \xrightarrow[c_1' = c_1 \cdot b]{first_a(c_1)} (q', v')$$

Our setting

(without the database)

The domain:

$$\mathcal{D} = \langle D, <, P_1, P_2, P_3, \dots, P_l \rangle$$

linearly ordered set

unary predicates
(subsets of D)

Examples

$$\langle \mathbb{N}, <, 0, 100, P_{\text{even}}, P_{\text{prime}} \rangle$$

$$\langle \mathbb{Q}, <, 0, 100, P_{\text{integer}}, P_{<\pi} \rangle$$

$$\langle \{a+b\}^*, <_{\text{lex}}, P_{(ab)^*} \rangle$$

Transitions:

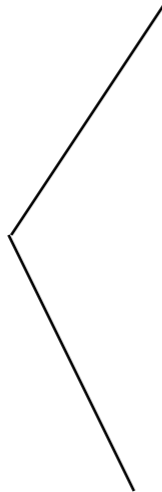
$$I, F \subseteq (Q \times D^X), \quad \delta \subseteq (Q \times D^X) \times (Q \times D^X)$$

are specified by quantifier free formulas over \mathcal{D}

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

\mathcal{A} 

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

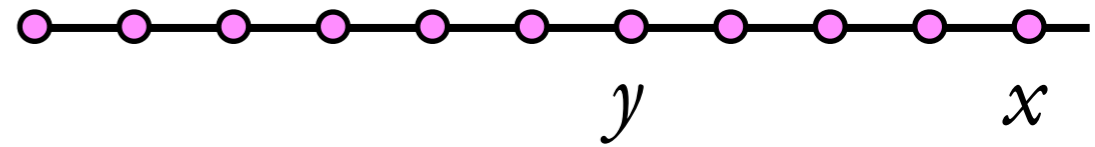
$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states



\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

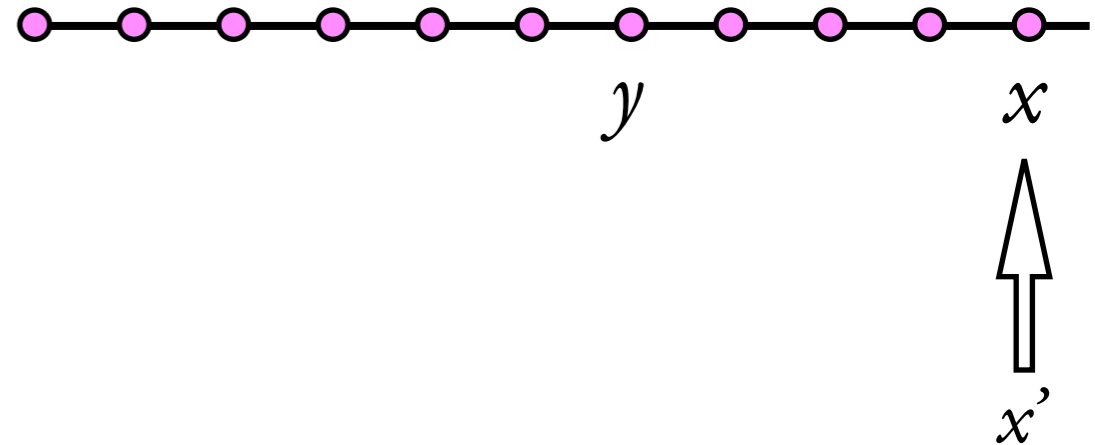
\mathcal{A} 

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

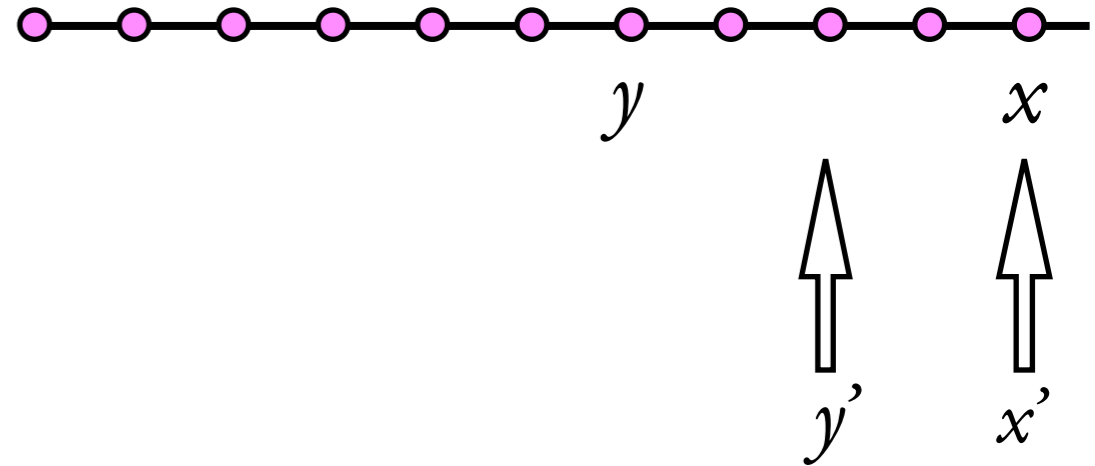
\mathcal{A} 

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

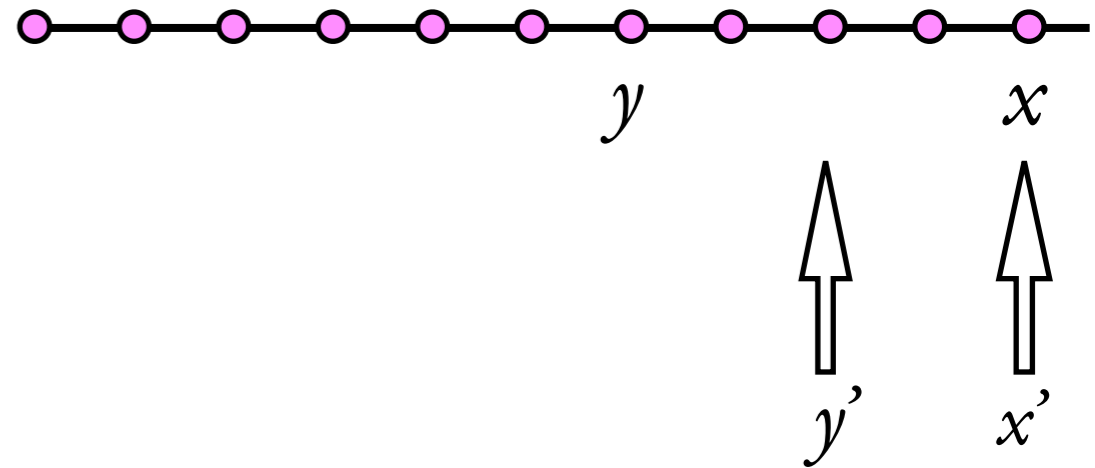
\mathcal{A} 

$$\tau_I: (x > 0) \wedge (y = 0)$$

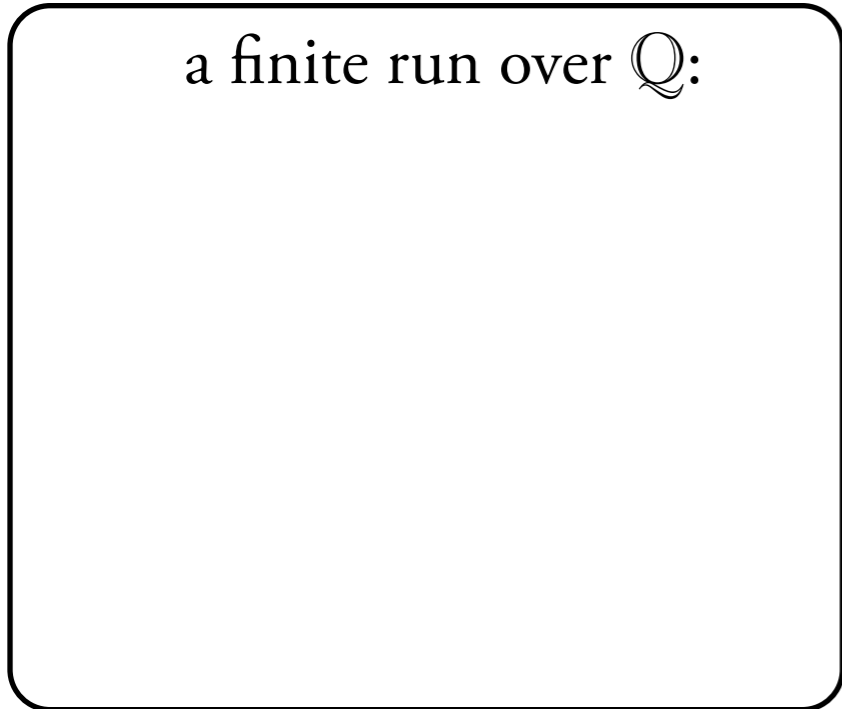
$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :



Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

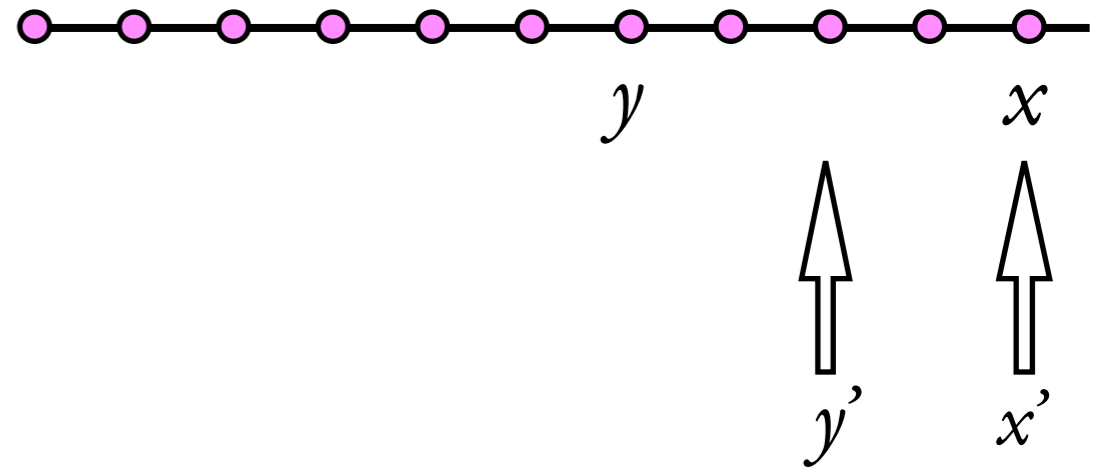
\mathcal{A} 

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

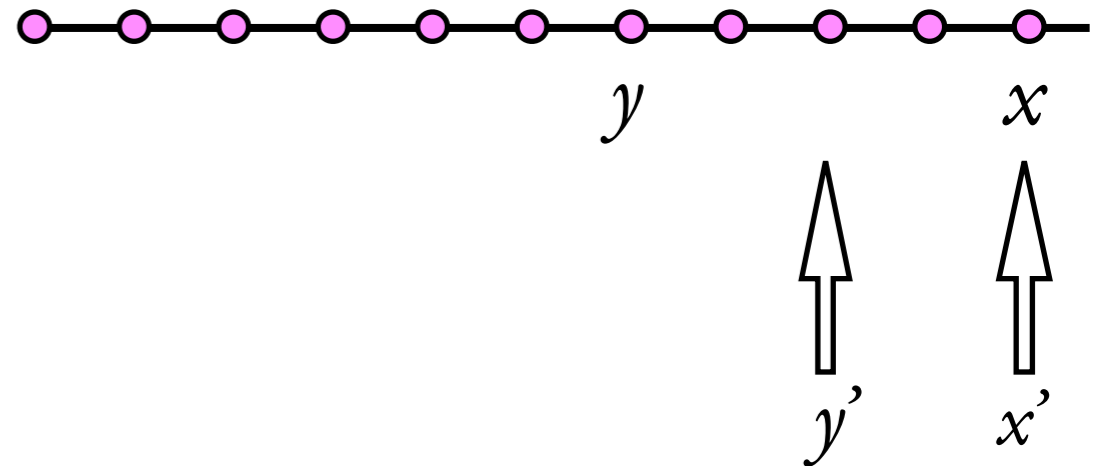
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x

y

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

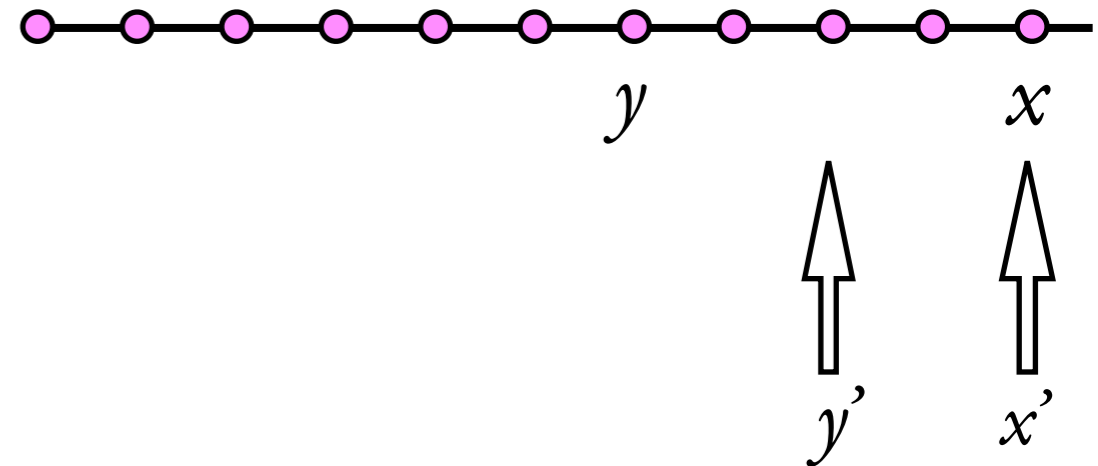
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x 5

y 0

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

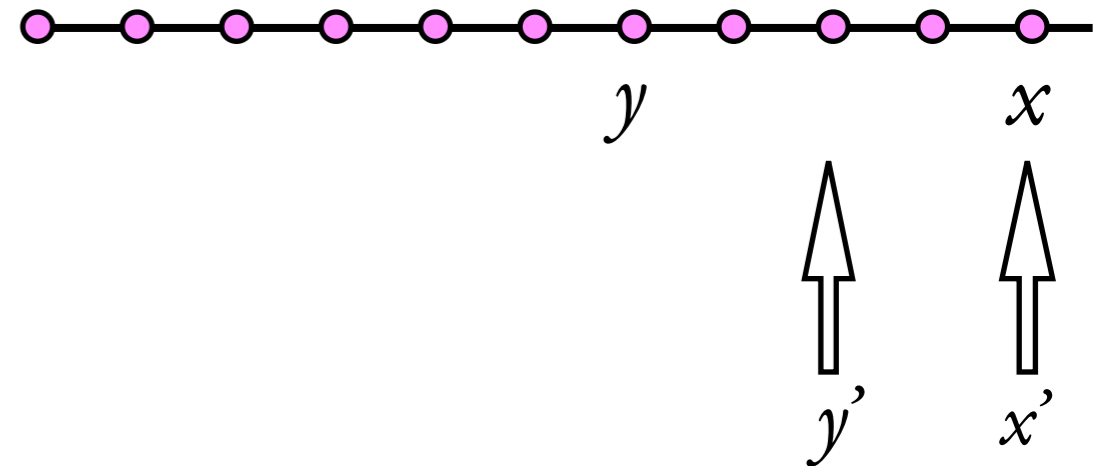
\mathcal{A} 

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x 5 5

y 0 1

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

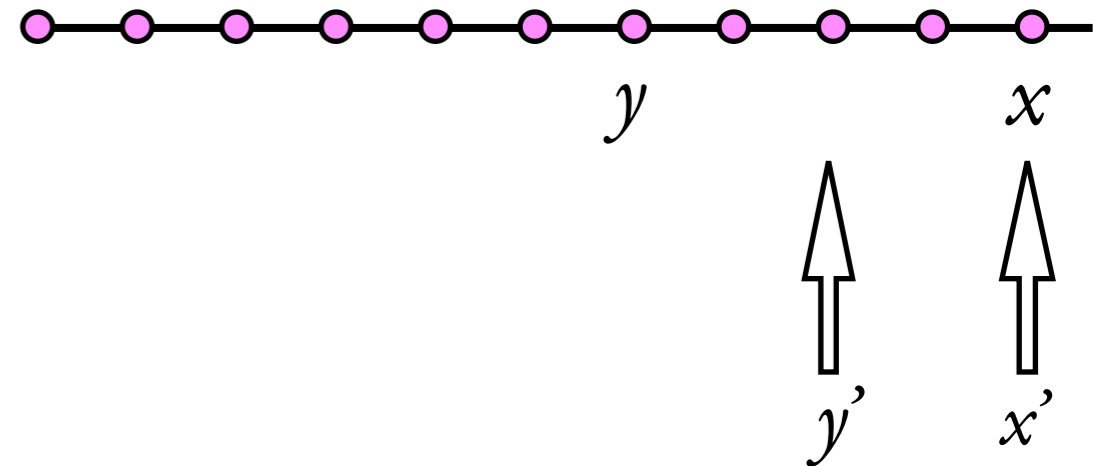
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x 5 5 5

y 0 1 2

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

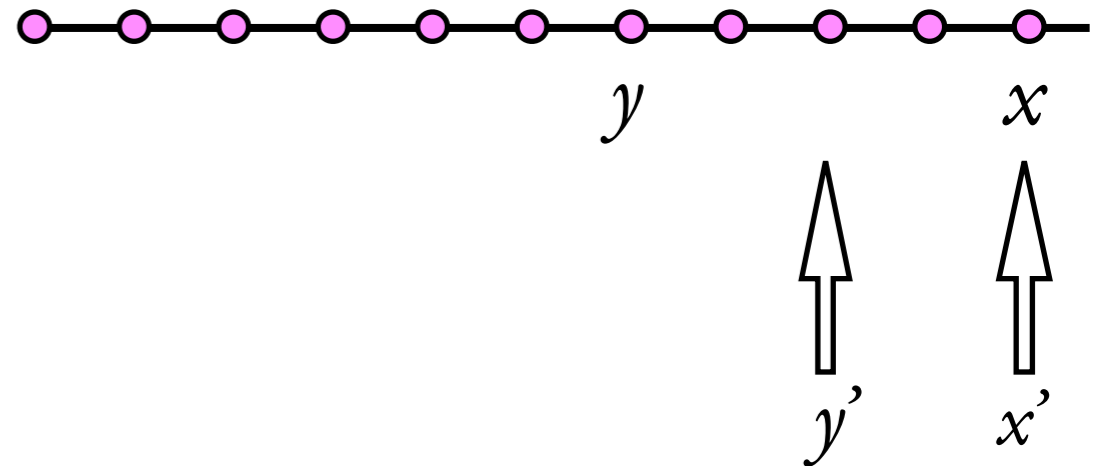
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x 5 5 5 5

y 0 1 2 4

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

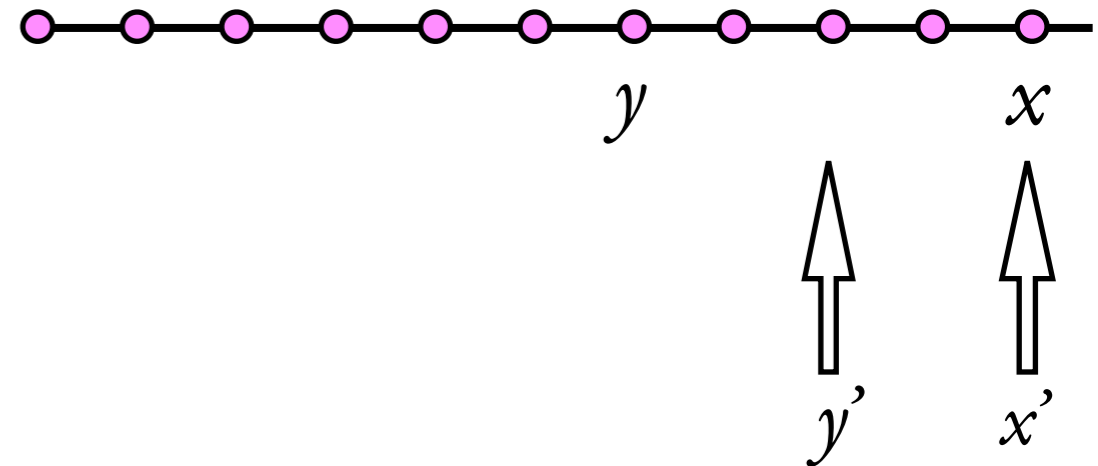
\mathcal{A} 

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x 5 5 5 5 5

y 0 1 2 4 0

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

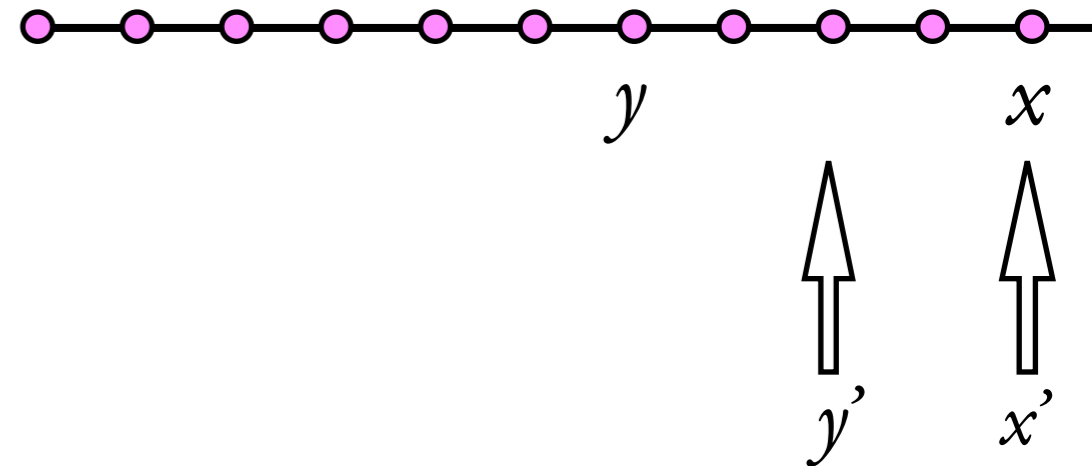
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x 5 5 5 5 5 5

y 0 1 2 4 0 3

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

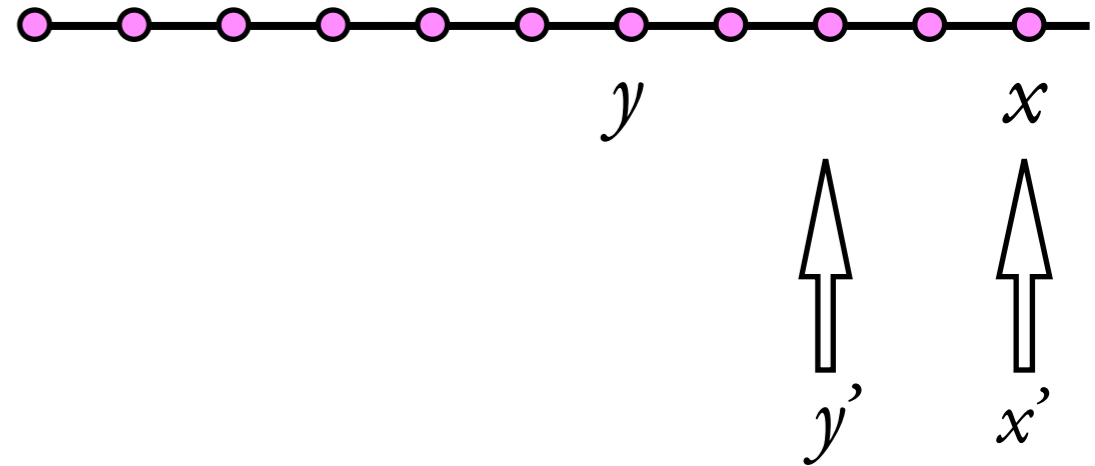
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x 5 5 5 5 5 5 5

y 0 1 2 4 0 3 0

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

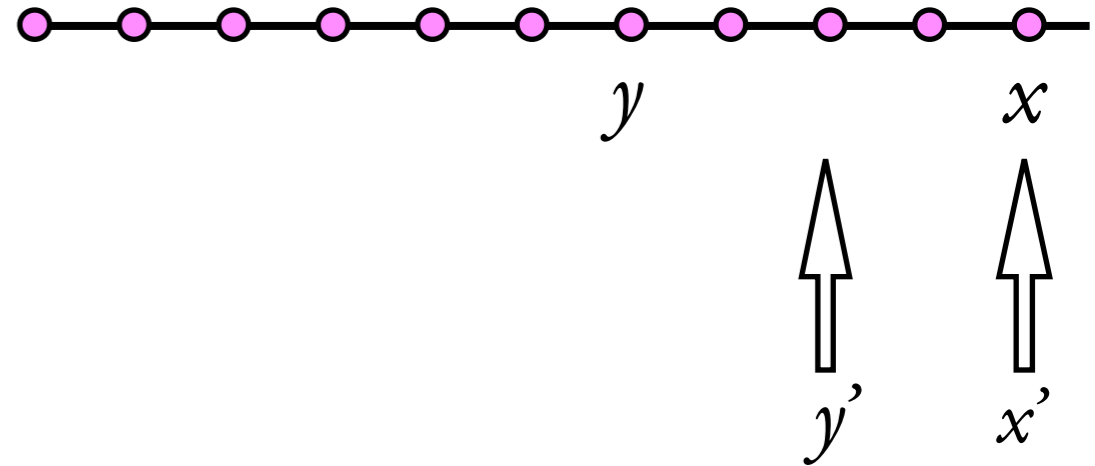
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

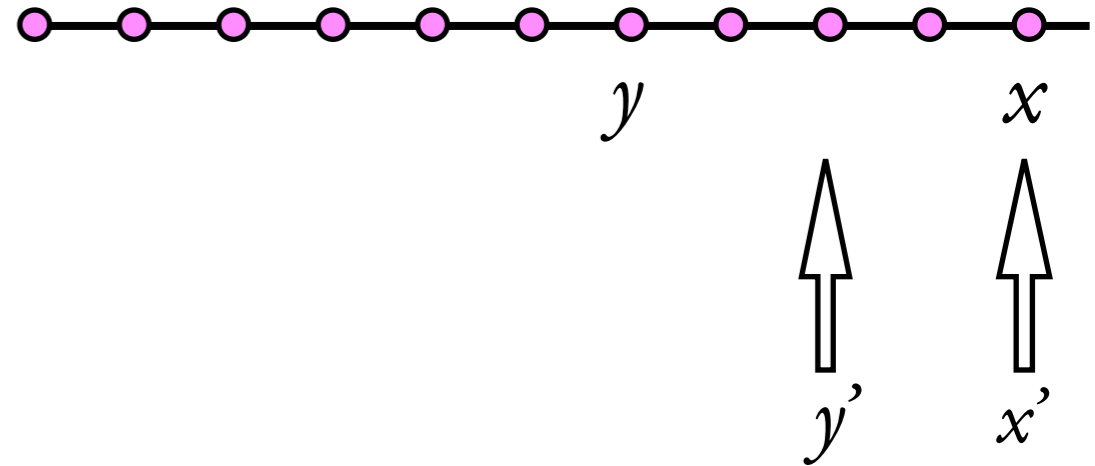
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

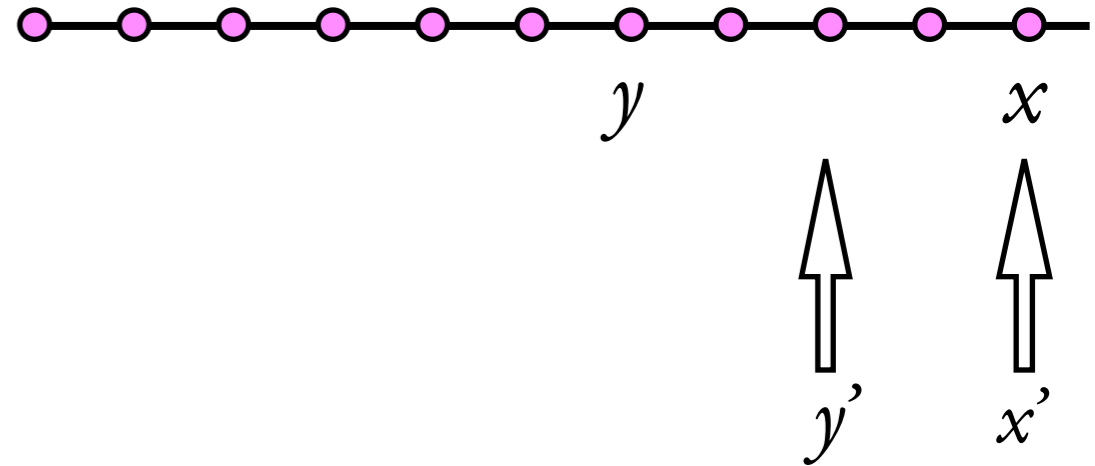
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a a$

x	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

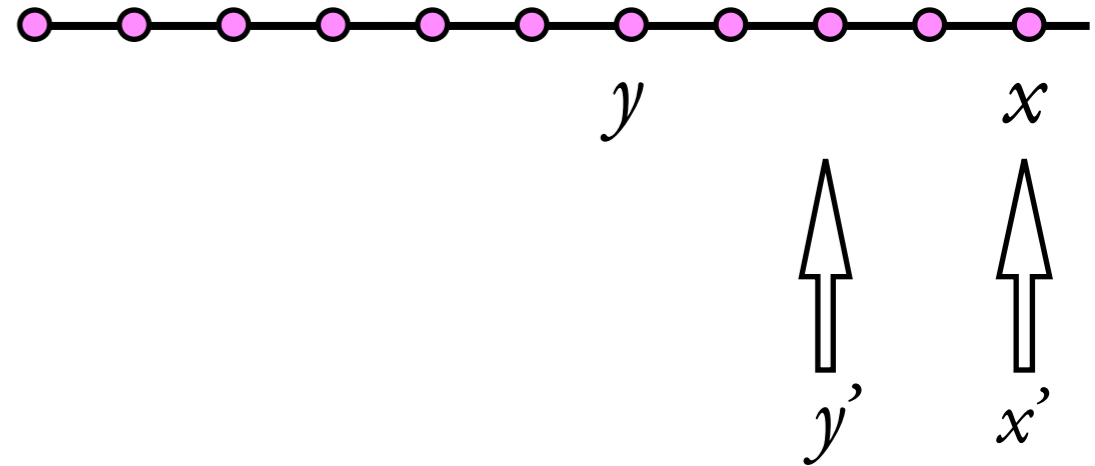
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

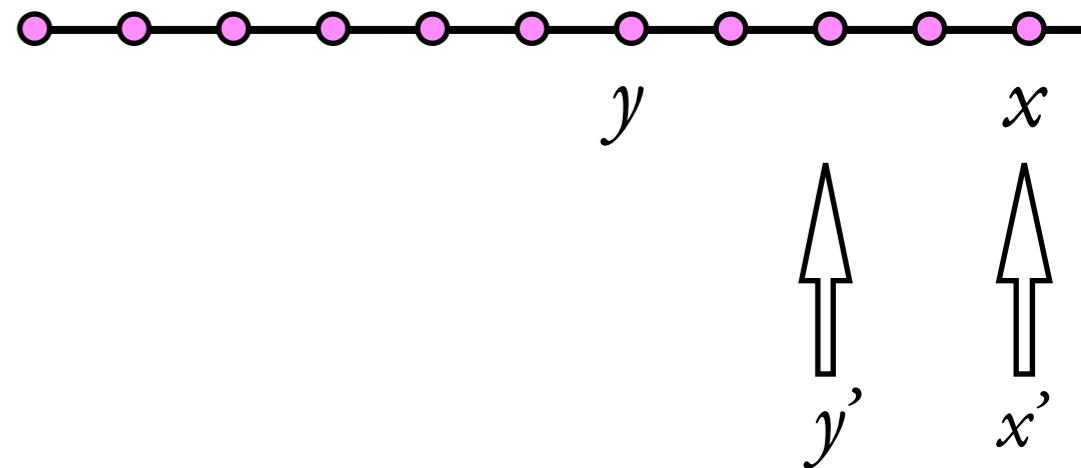
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	1

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

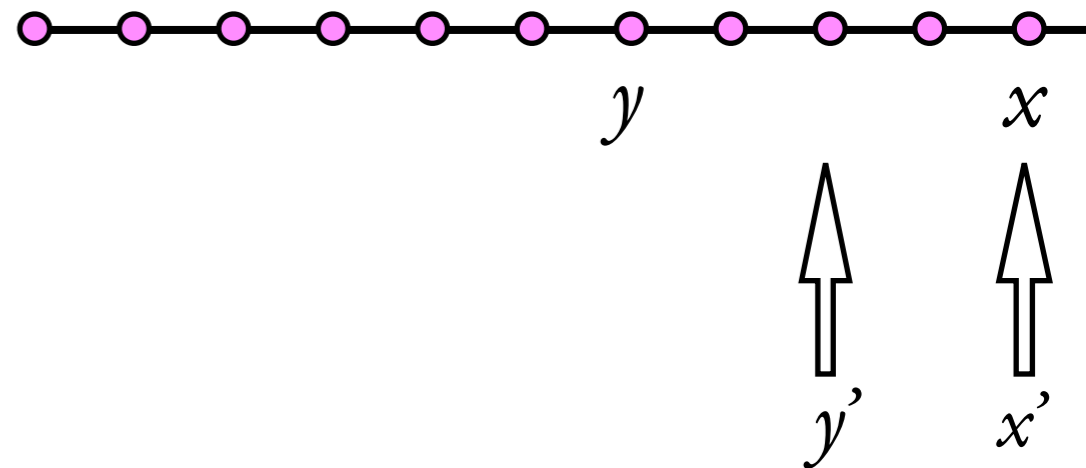
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	1

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states

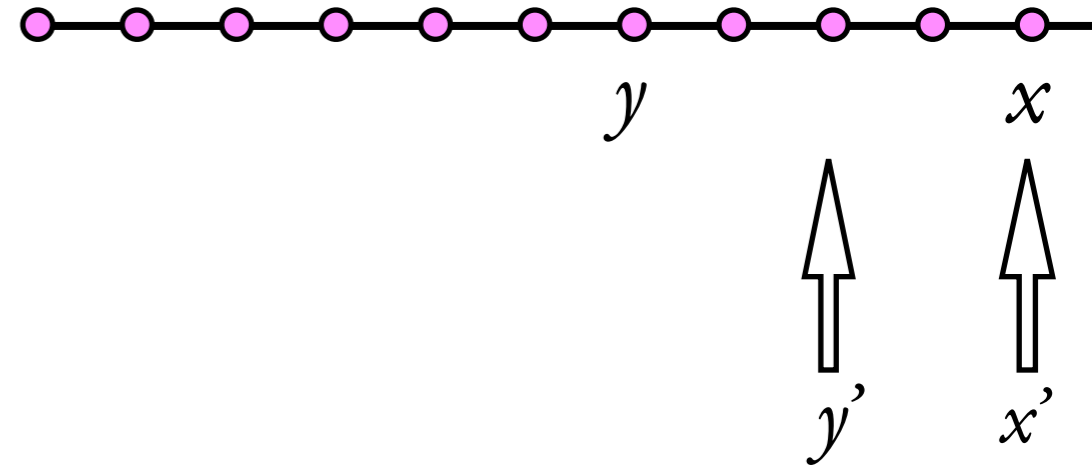
\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



a finite run over \mathbb{Q} :

$a a a b a b a a a a$

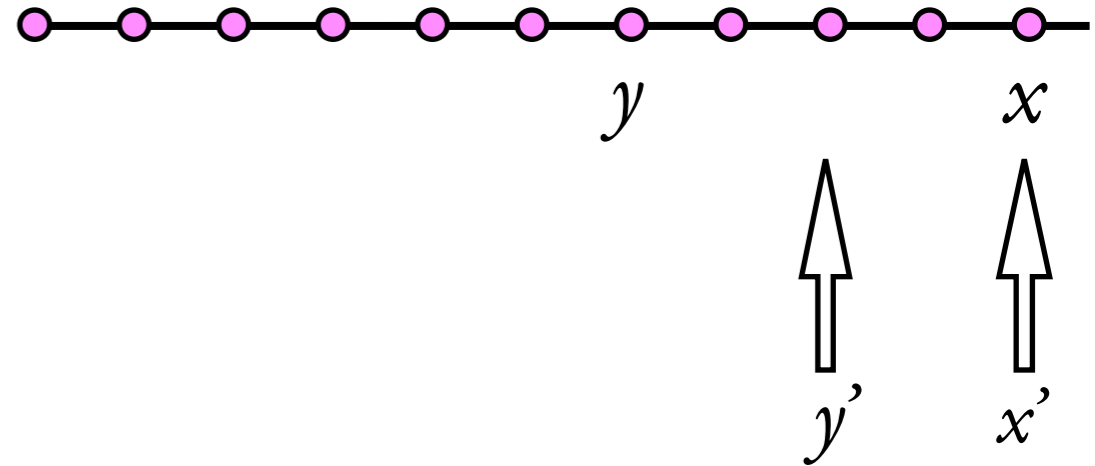
x	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	$\frac{1}{5}$	1

accepted language: $(a+b)^*a$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states



\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	$\frac{1}{5}$	1

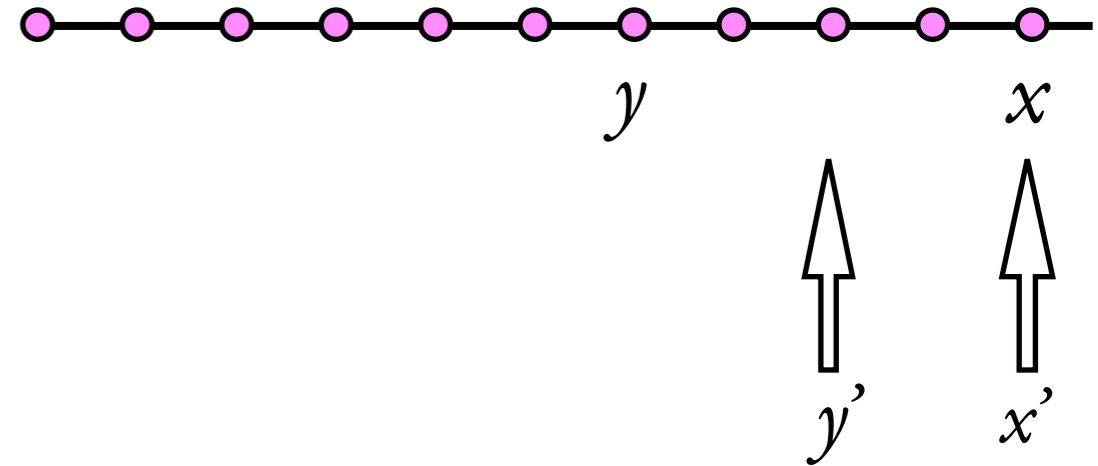
accepted language: $(a+b)^*a$

a finite run over \mathbb{N} :

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states



\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	$\frac{1}{5}$	1

accepted language: $(a+b)^*a$

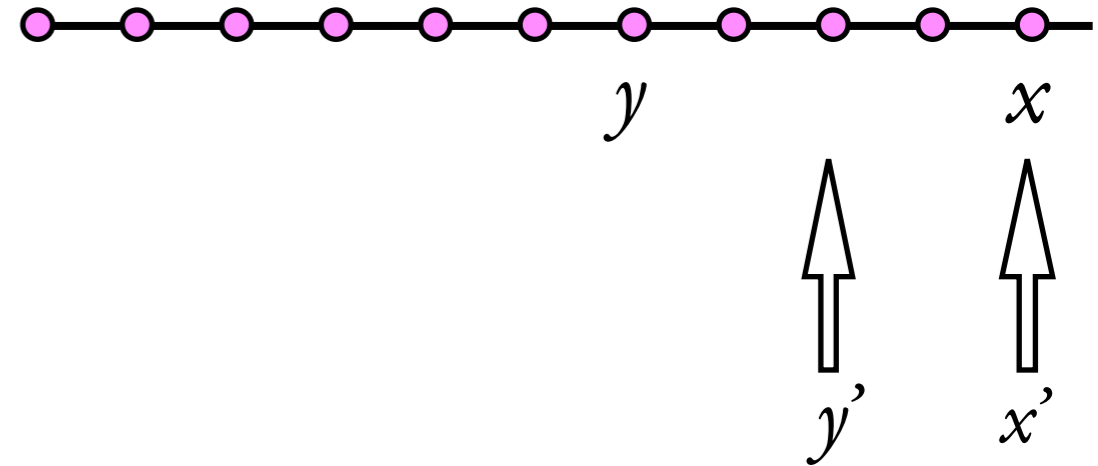
a finite run over \mathbb{N} :

$a a a b a b a a a a$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states



\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	1

accepted language: $(a+b)^*a$

a finite run over \mathbb{N} :

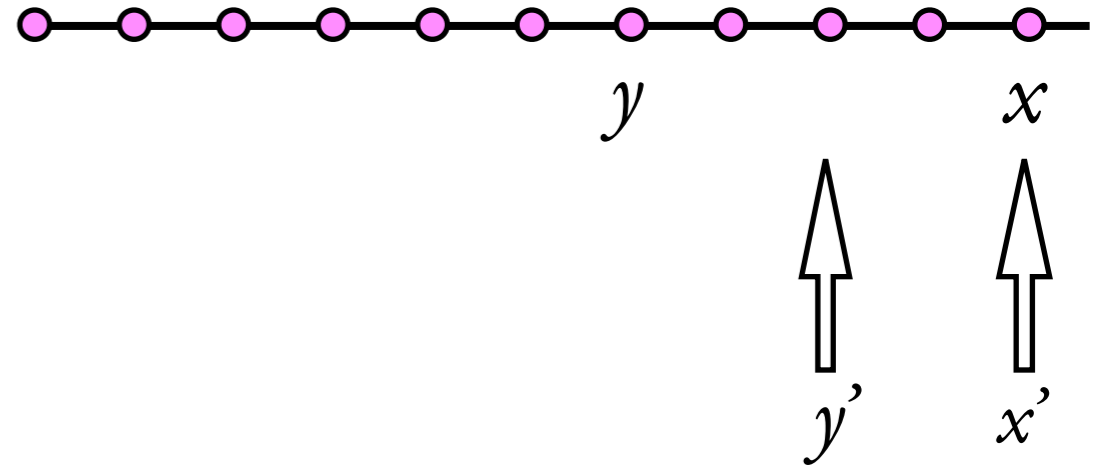
$a a a b a b a a a a$

x
 y

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states



\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	$\frac{1}{5}$	1	

accepted language: $(a+b)^*a$

a finite run over \mathbb{N} :

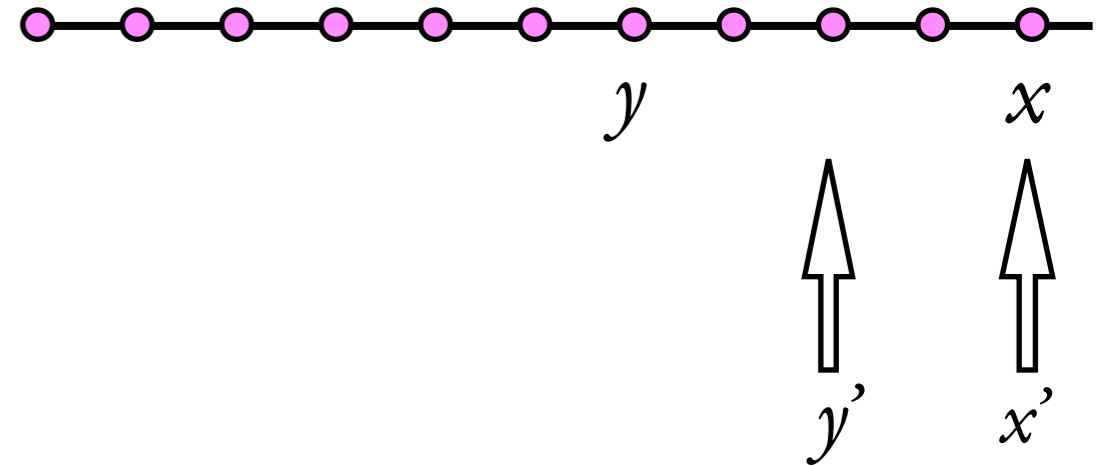
$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	1	2	3	4			

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states



\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

a finite run over \mathbb{Q} :

$a a a b a b a a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	$\frac{1}{5}$	1	

accepted language: $(a+b)^* a$

a finite run over \mathbb{N} :

$a a a b a b a a a a a$

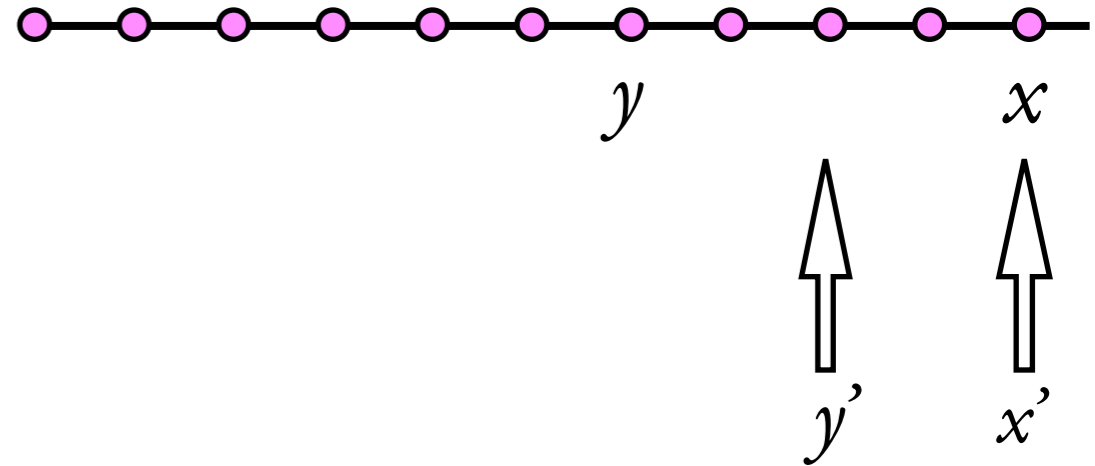
x	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	1	2	3	4		

accepted language: $(a^* b)^* a$

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states



\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

a finite run over \mathbb{Q} :

$aaababaaaa$

x	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	$\frac{1}{5}$	1		

accepted language: $(a+b)^*a$

a finite run over \mathbb{N} :

$aaababaaaa$

x	5	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	1	2	3	4			

accepted language: $(a^*b)^*a$

an infinite run over \mathbb{N} :

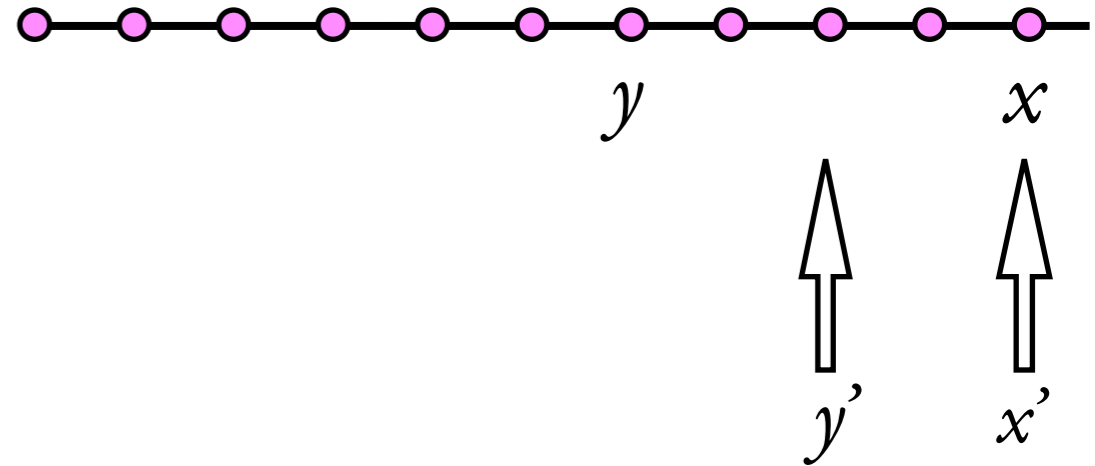
$aaababaaaa$

x	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	1	2	3	4				

Example D -automaton \mathcal{A}

x, y – variables of \mathcal{A}

no states



\mathcal{A}

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

a finite run over \mathbb{Q} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{4}{5}$	1	1

accepted language: $(a+b)^* a$

a finite run over \mathbb{N} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	1	2	3	4	4	4

accepted language: $(a^* b)^* a$

an infinite run over \mathbb{N} :

$a a a b a b a a a a$

x	5	5	5	5	5	5	5	5	5	5	5	5	5
y	0	1	2	4	0	3	0	1	2	3	4	4	4

accepted language: $(a^B b)^\omega$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$$\tau_I: (x=0)$$

$$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$$

$$\delta_b: (x' = 0)$$

$$\tau_F: (x=1)$$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$$\tau_I: (x=0)$$

$$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$$

$$\delta_b: (x'=0)$$

$$\tau_F: (x=1)$$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

} \exists

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$R = \{(0), (1), (2), (3), (4)\}$ $S = \{\}$ } \exists

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$

$R = \{(0), (1), (2), (3), (4)\} \quad S = \{\}$

} \exists

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$ } \exists
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{\}$

accepted language: $(a^*b)^*a$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$ } \exists
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{\}$

accepted language: $(a^*b)^*a$

an infinite run over \mathbb{N} :

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$ } \exists
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{\}$

accepted language: $(a^*b)^*a$

an infinite run over \mathbb{N} :

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

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1 \quad \left. \vphantom{x} \right\} \exists$
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^*b)^*a$

an infinite run over \mathbb{N} :

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

$R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$ } \exists
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^*b)^*a$

an infinite run over \mathbb{N} :

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

$x \quad 0 \quad 1 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1 \quad 4 \quad 0 \quad 1 \quad 3$
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

$\delta_b: (x'=0)$

$\tau_F: (x=1)$

EMPTINESS: is there a *finite* database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$ } \exists
 $R=\{(0),(1),(2),(3),(4)\} \quad S=\{\}$

accepted language: $(a^*b)^* a$

an infinite run over \mathbb{N} :

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

$x \quad 0 \quad 1 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1 \quad 4 \quad 0 \quad 1 \quad 3$
 $R=\{(0),(1),(2),(3),(4)\} \quad S=\{\}$

accepted language: $(a^B b)^\omega$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

Decide emptiness
of D -automata

EMPTINESS: is there a finite database M , a word w ,
and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$ } \exists
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^*b)^* a$

an infinite run over \mathbb{N} :

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

$x \quad 0 \quad 1 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1 \quad 4 \quad 0 \quad 1 \quad 3$
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^B b)^\omega$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

I. infinite, ordered data

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

Decide emptiness
of D -automata

$\tau_F: (x=)$

EMPTINESS: is there a finite database M , a word w , and an accepting run over w consistent with M ?

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$ } \exists
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^*b)^*a$

an infinite run over \mathbb{N} :

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

$x \quad 0 \quad 1 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1 \quad 4 \quad 0 \quad 1 \quad 3$
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^B b)^\omega$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

1. infinite, ordered data

$\delta_a: (x > x) \wedge R(x) \wedge \neg S(x, y)$

Decide emptiness

of D -automata

EMPTINESS: is there a finite database M , a word w , and an accepting run over w consistent with M ?

2. unbounded database

a finite run over \mathbb{N} :

$a a a b a b a$

$x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$ } \exists
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^*b)^*a$

an infinite run over \mathbb{N} :

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

$x \quad 0 \quad 1 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1 \quad 4 \quad 0 \quad 1 \quad 3$
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^B b)^\omega$

Adding database constraints

x, y, z, \dots – a finite set of variables

R, S, \dots – a finite set of relational symbols

$\tau_I: (x=0)$

1. infinite, ordered data

$\delta_a: (x' > x) \wedge R(x) \wedge \neg S(x, y')$

Decide emptiness
of D -automata

$\tau_F: (x=0)$

EMPTINESS: is there a finite database M , a word w ,
and an accepting run over w consistent with M ?

3. infinite runs

a finite run over \mathbb{N} :

$a a a b a b a$
 $x \quad 0 \quad 2 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1$
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$ } \exists

accepted language: $(a^*b)^*a$

2. unbounded database

an infinite run over \mathbb{N} :

$a a a b a b a a b a a \dots$
 $x \quad 0 \quad 1 \quad 3 \quad 4 \quad 0 \quad 3 \quad 0 \quad 1 \quad 4 \quad 0 \quad 1 \quad 3$
 $R = \{(0), (1), (2), (3), (4)\} \quad S = \{ \}$

accepted language: $(a^B b)^\omega$

Known results

deciding emptiness (for ω -runs)

	no database	with database
$\mathbb{N}, =$	Kaminski, Francez (94)	Deutsch, Sui, Vianu, Zhou (06) PSPACE
$\mathbb{Q}, <$	Čerans (94) PSPACE	Deutsch, Hull, Patrizi, Vianu (09) PSPACE
$\mathbb{N}, <$	Čerans (94) NONPRIMITIVE	?

Known results

deciding emptiness (for ω -runs)

	no database	with database
$\mathbb{N}, =$	Kaminski, Francez (94)	Deutsch, Sui, Vianu, Zhou (06) PSPACE
$\mathbb{Q}, <$	Čerans (94) PSPACE	Deutsch, Hull, Patrizi, Vianu (09) PSPACE
$\mathbb{N}, <$	Čerans (94) NONPRIMITIVE	?
$\mathcal{D}, <$	PSPACE	PSPACE

Known results

deciding emptiness (for ω -runs)

	no database	with database
$\mathbb{N}, =$	Kaminski, Francez (94)	Deutsch, Sui, Vianu, Zhou (06) PSPACE
$\mathbb{Q}, <$	Čerans (94) PSPACE	Deutsch, Hull, Patrizi, Vianu (09) PSPACE
$\mathbb{N}, <$	Čerans (94) NONPRIMITIVE	PSPACE
$\mathcal{D}, <$	PSPACE	PSPACE

Known results

deciding emptiness (for ω -runs)

	no database	with database
$\mathbb{N}, =$	Kaminski, Francez (94)	Deutsch, Sui, Vianu, Zhou (06) PSPACE
$\mathbb{Q}, <$	Čerans (94) PSPACE	Deutsch, Hull, Patrizi, Vianu (09) PSPACE
$\mathbb{N}, <$	PSPACE	PSPACE
$\mathcal{D}, <$	PSPACE	PSPACE

Outline

1. Define automata model
2. Analyze automata over dense orders
3. Analyze automata over discrete orders
4. Add database and infinite runs

Dense case

- $D = \mathbb{Q}$
- finite words
- no database

Dense case

- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

Dense case

- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

The region construction

$0 < y < 1 < x$, $0 < x < 1 = y$, $0 = 1 < y < x$, $0 < 1 = y < x$, $0 = y < 1 < x$, $0 = y < 1 = x$, ...

Dense case

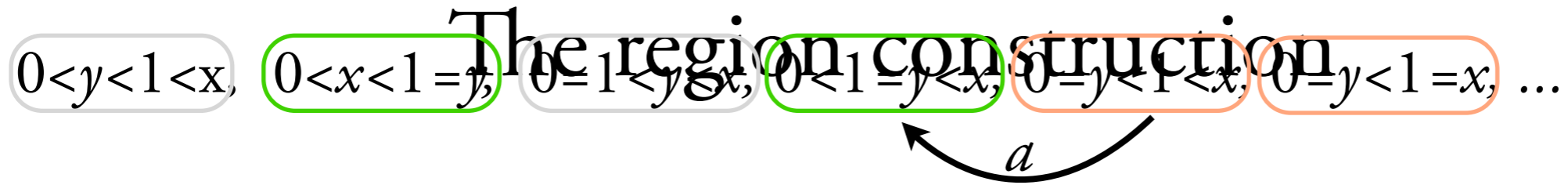
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



Dense case

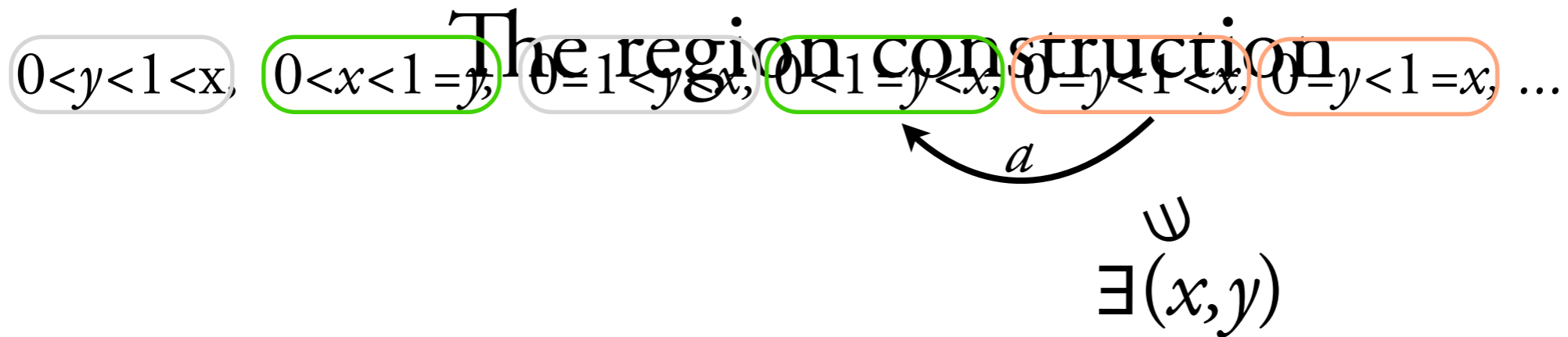
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



Dense case

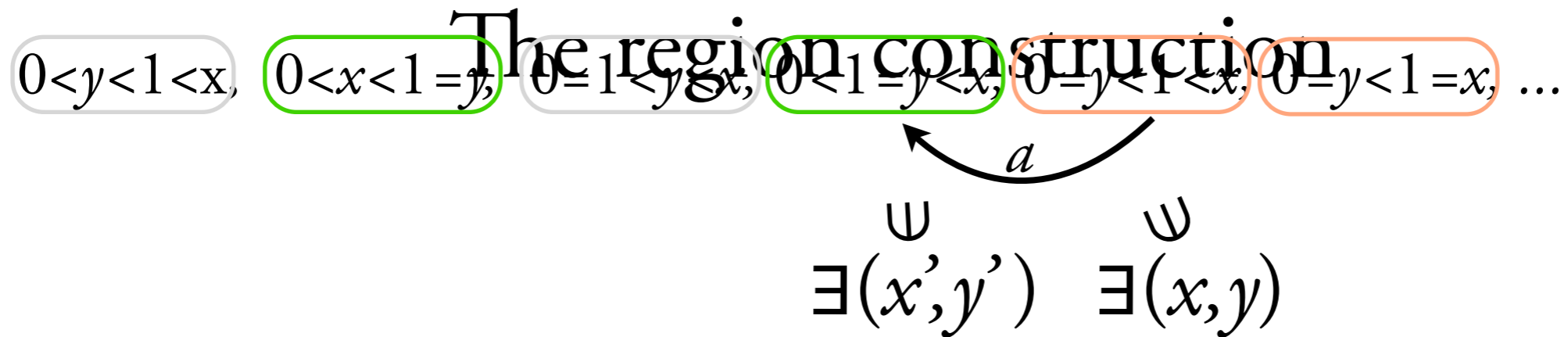
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



Dense case

- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

The region construction

$0 < y < 1 < x$, $0 < x < 1 = y$, $0 = 1 < y < x$, $0 < 1 = y < x$, $0 = y < 1 < x$, $0 = y < 1 = x$, ...

$$\begin{array}{ccc} & \curvearrowright a & \\ \Downarrow & & \Downarrow \\ \exists(x', y') & \exists(x, y): & \delta_a(x, y, x', y') \end{array}$$

Dense case

- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

The region construction

$0 < y < 1 < x$, $0 < x < 1 = y$, $0 = 1 < y < x$, $0 < 1 = y < x$, $0 = y < 1 < x$, $0 = y < 1 = x$, ...

$$\begin{array}{ccc}
 & \Downarrow & \Downarrow \\
 \exists(x', y') & \xrightarrow{a} & \exists(x, y): \delta_a(x, y, x', y') \\
 & & (5, 0)
 \end{array}$$

Dense case

- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$

The region construction

$0 < y < 1 < x$, $0 < x < 1 = y$, $0 = 1 < y < x$, $0 < 1 = y < x$, $0 = y < 1 < x$, $0 = y < 1 = x$, ...

$$\begin{array}{ccc}
 \Downarrow & & \Downarrow \\
 \exists(x', y') & \exists(x, y): & \delta_a(x, y, x', y') \\
 (5, 1) & & (5, 0)
 \end{array}$$

Dense case

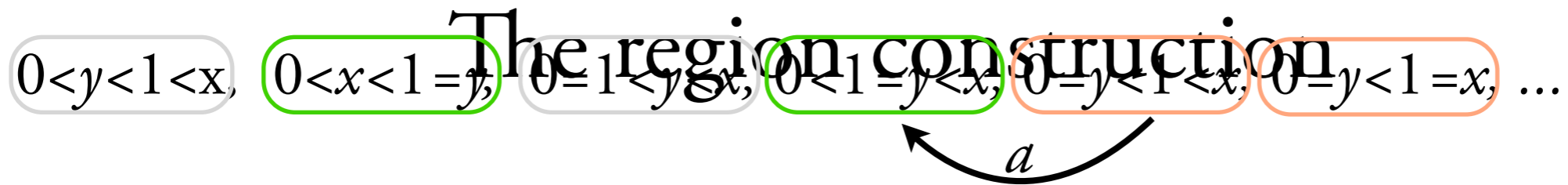
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



Dense case

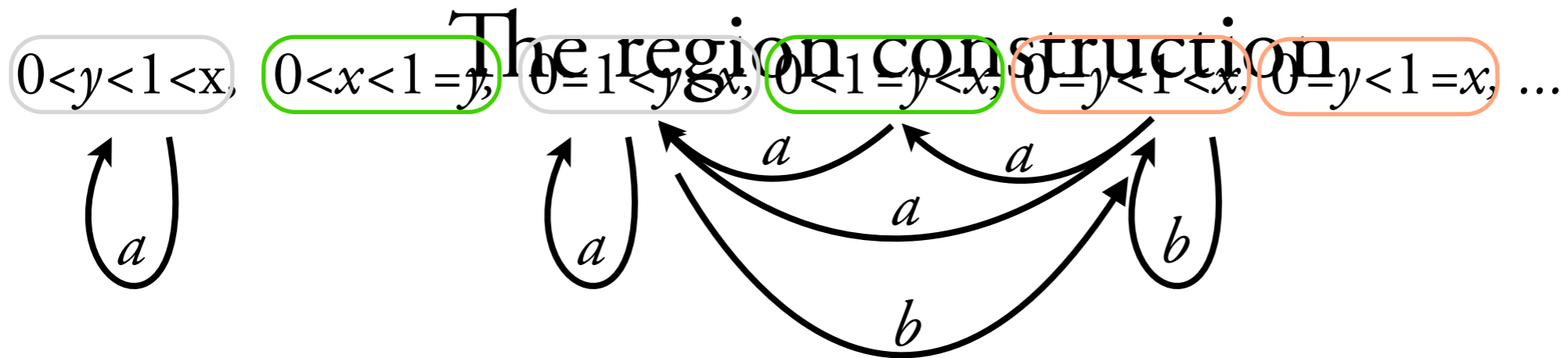
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



Dense case

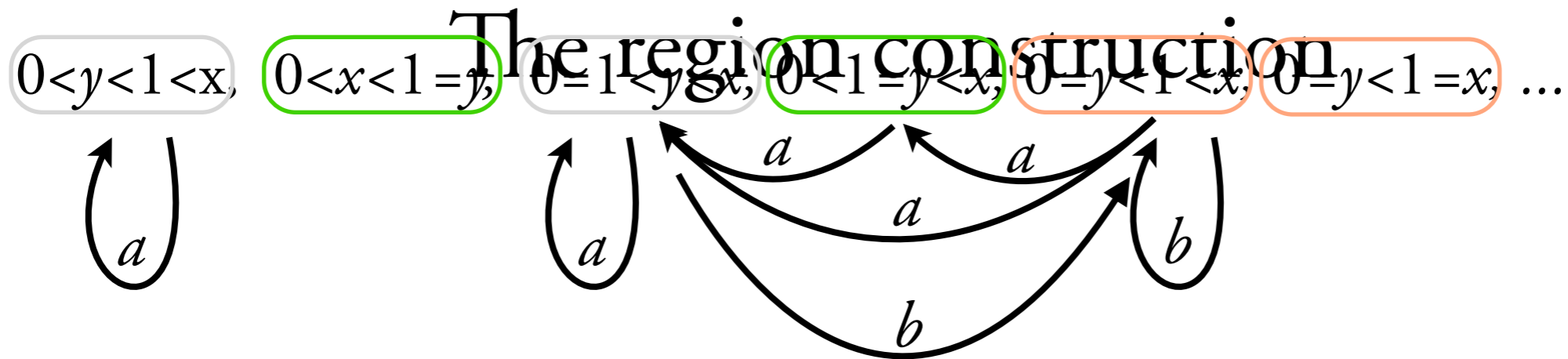
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



Dense case

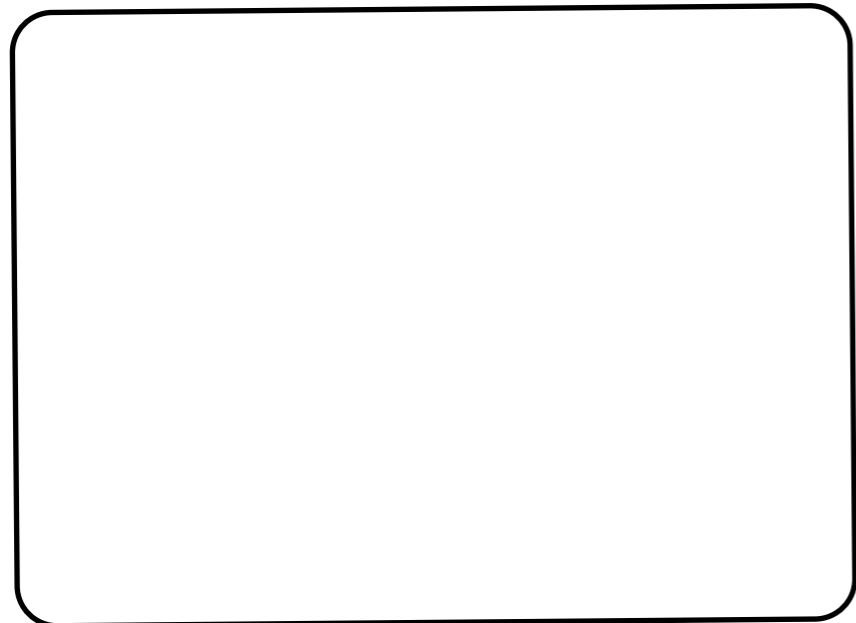
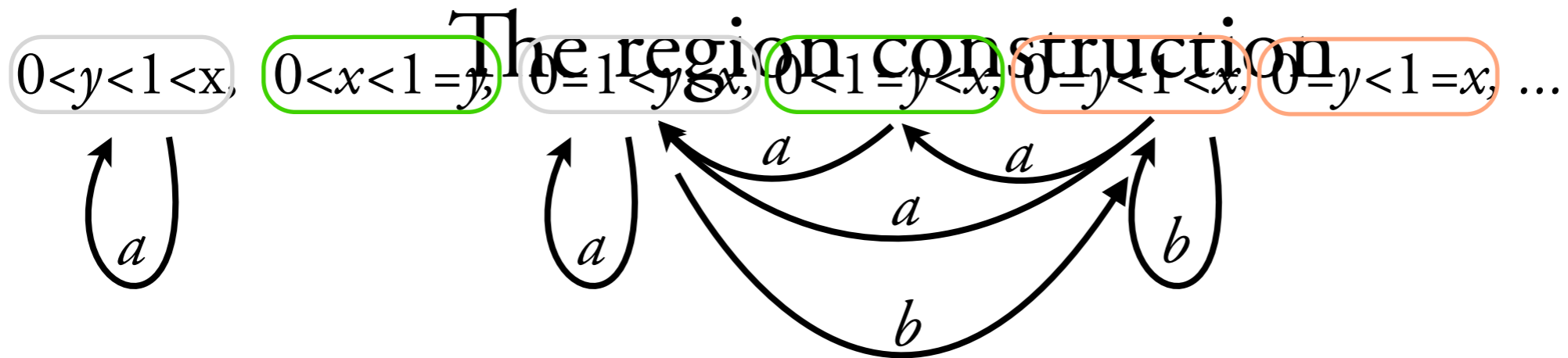
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation

Dense case

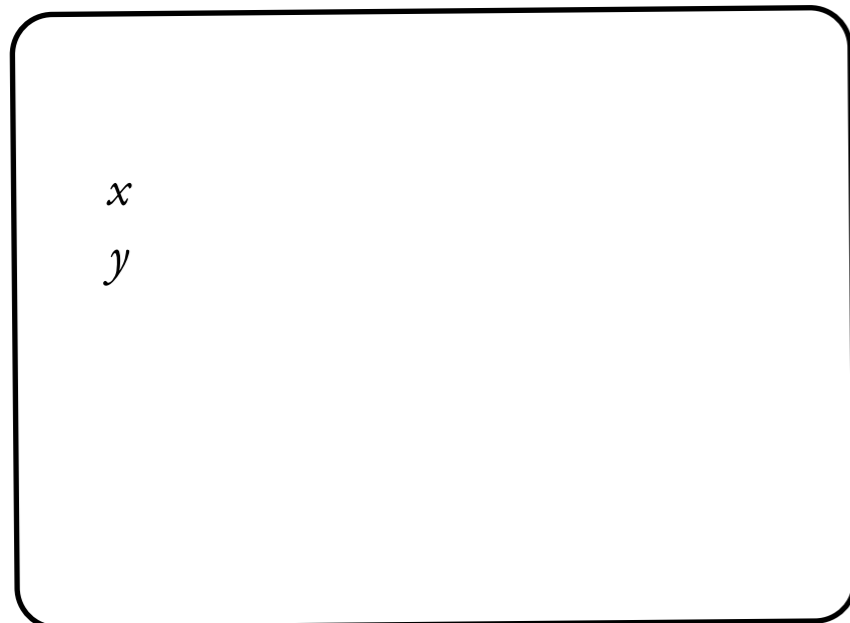
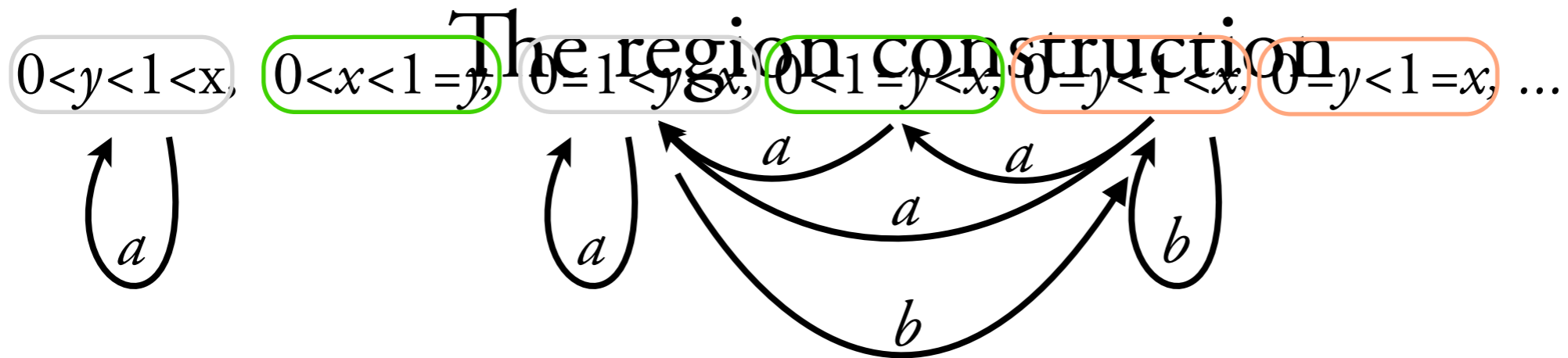
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation

Dense case

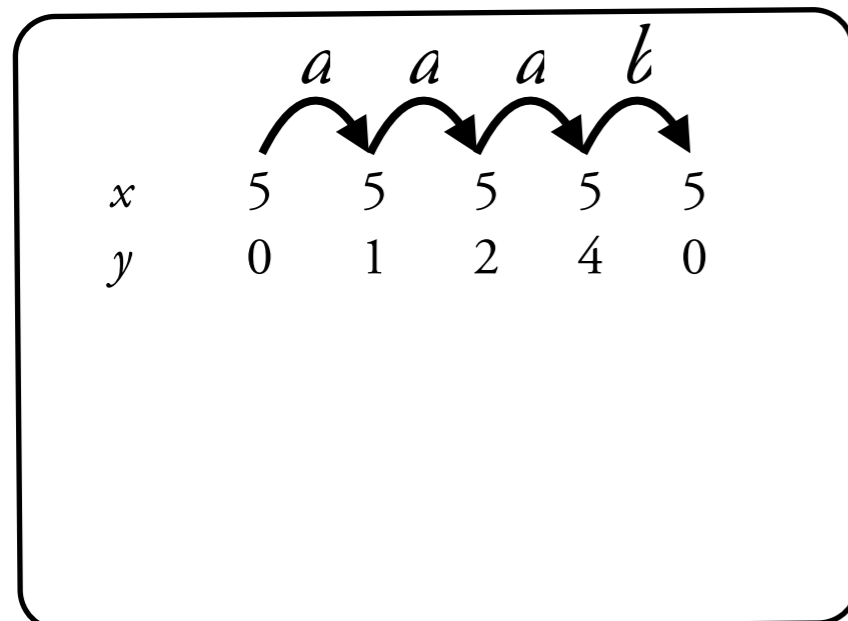
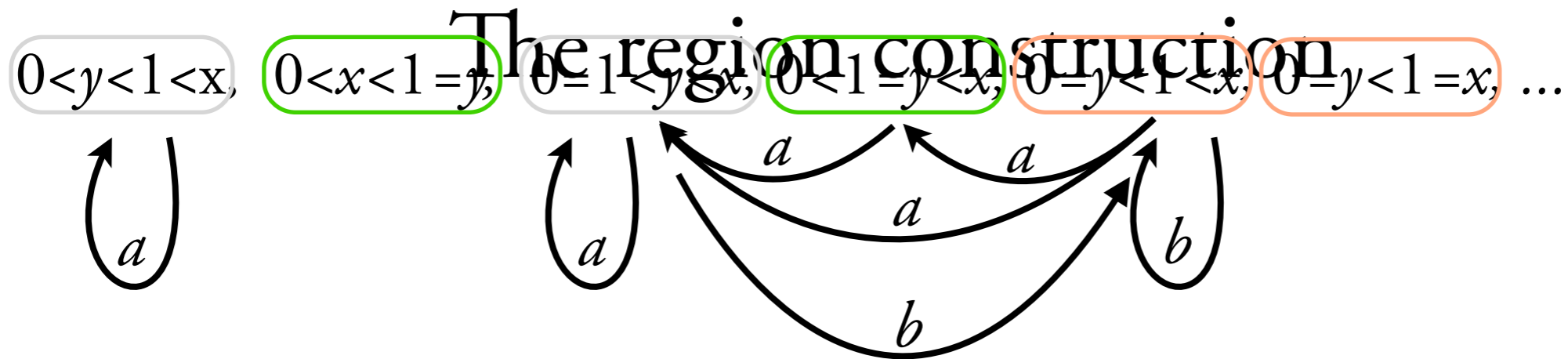
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation

Dense case

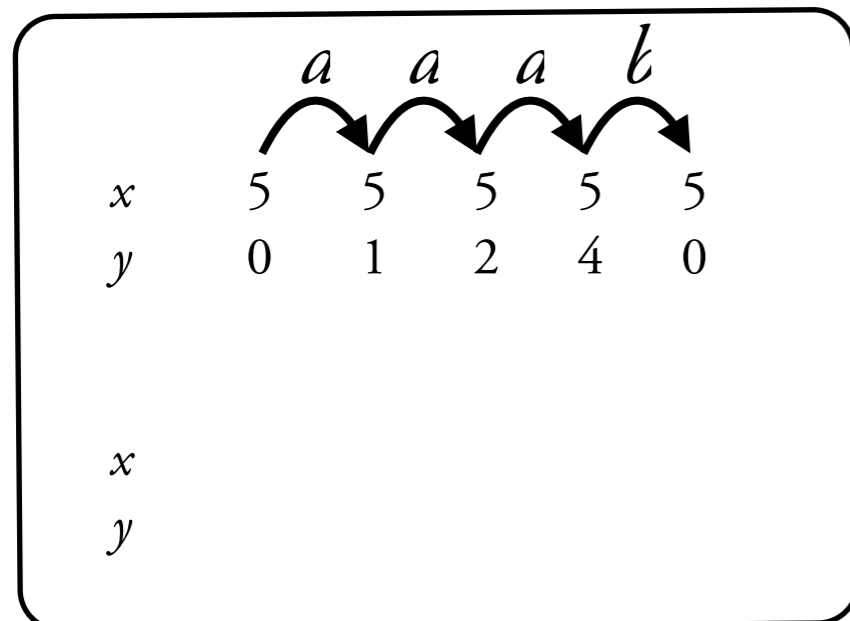
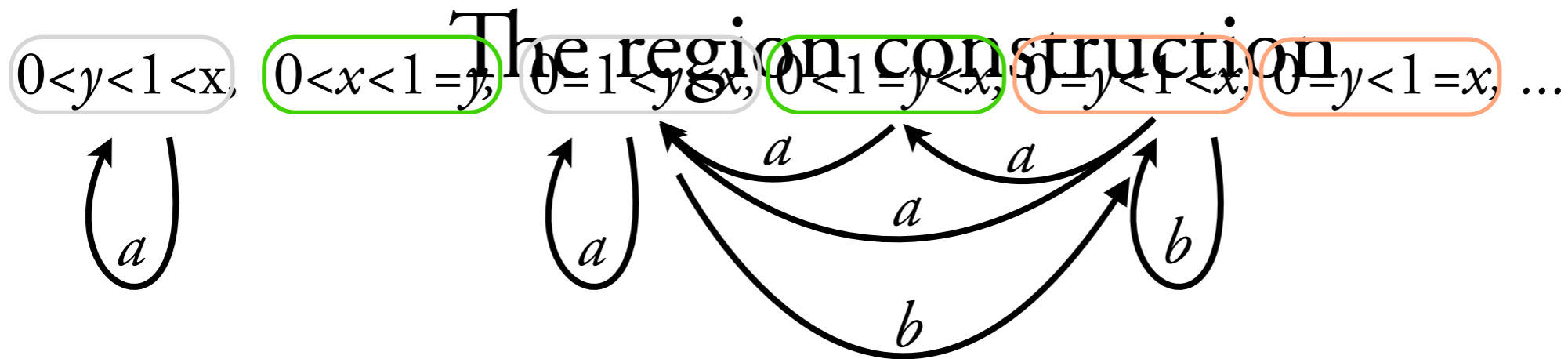
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation

Dense case

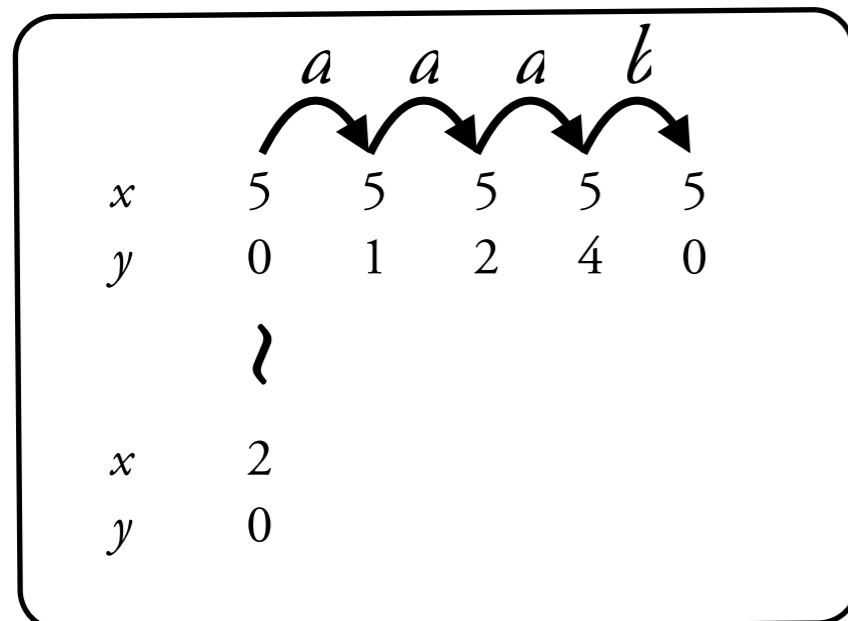
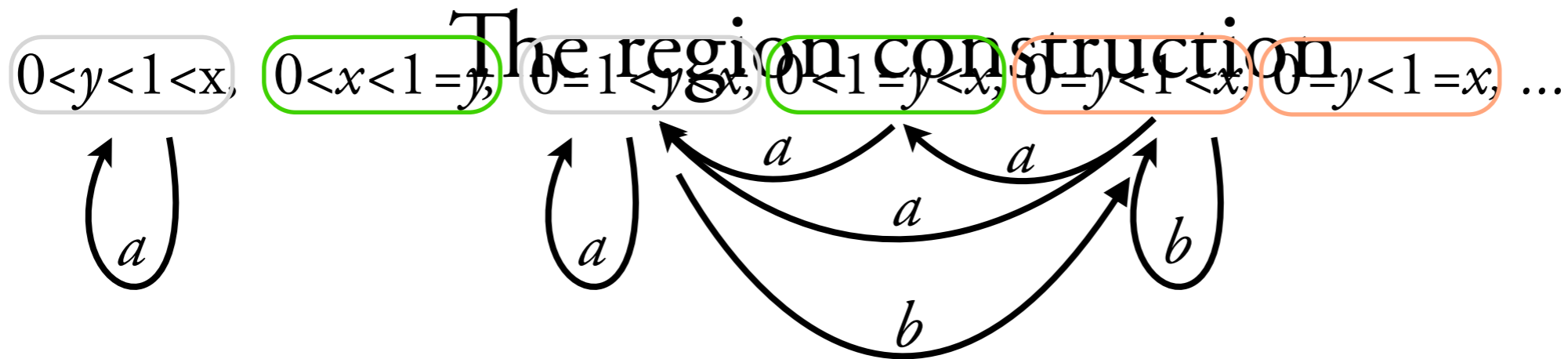
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation

Dense case

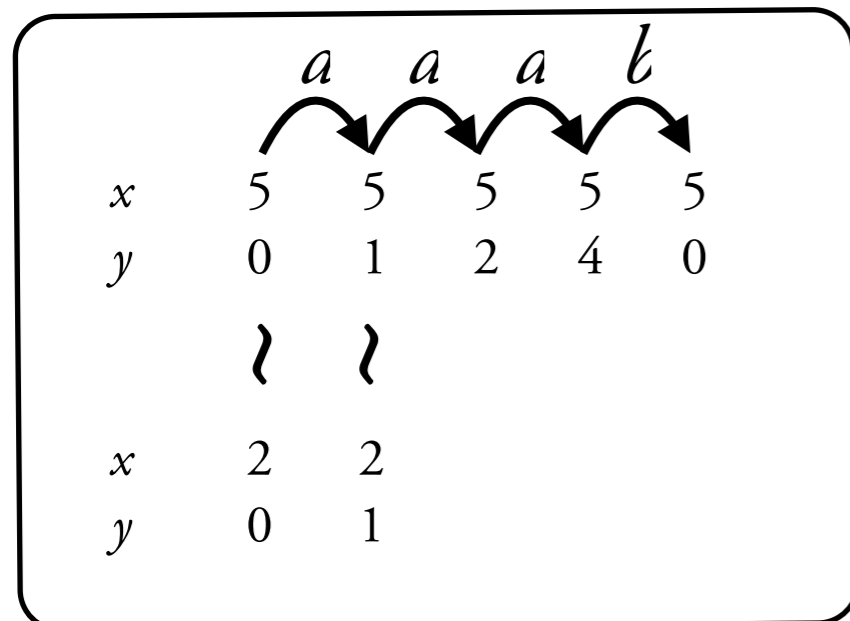
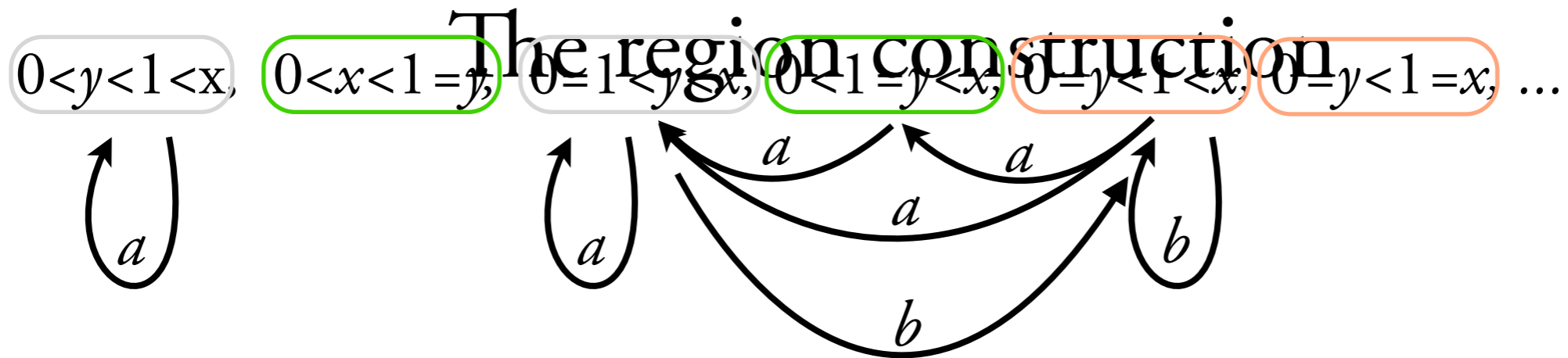
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation

Dense case

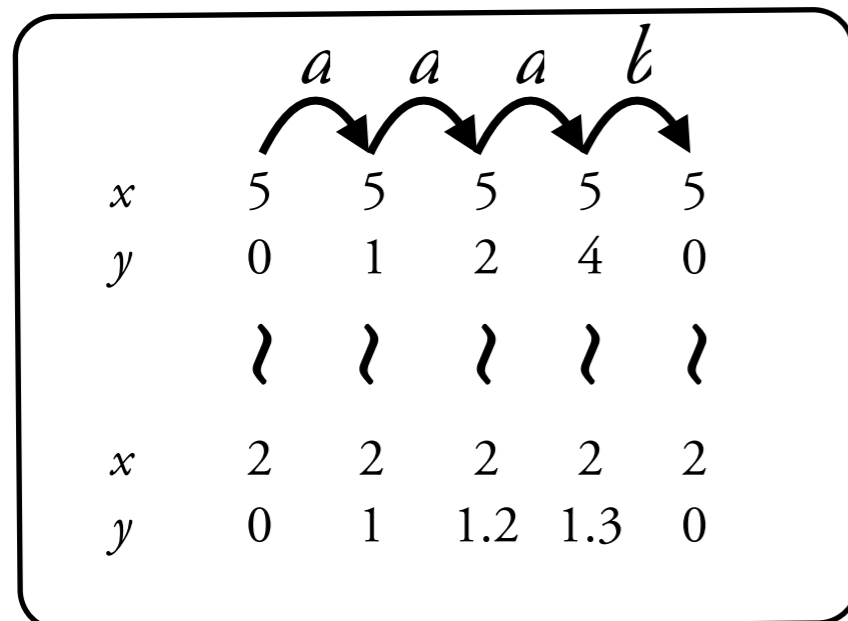
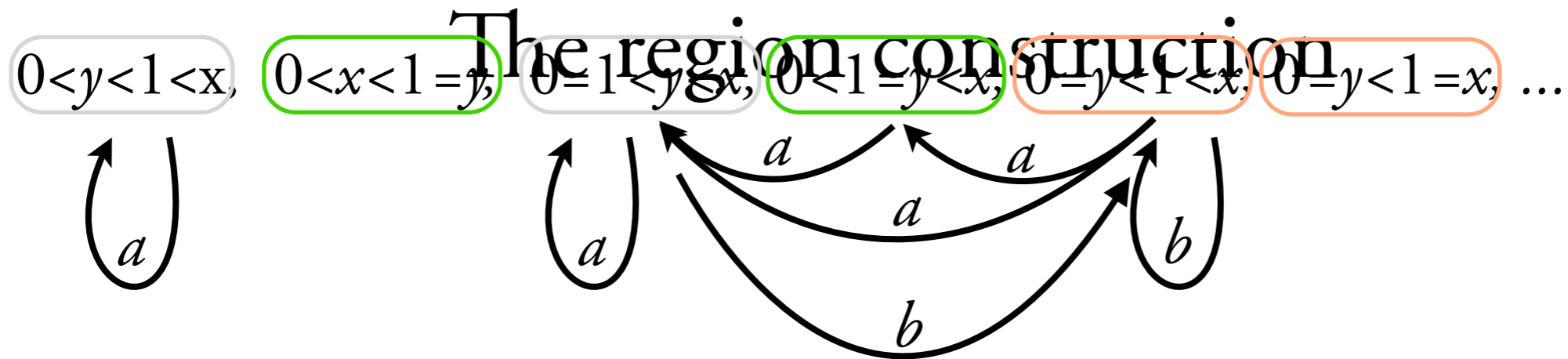
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation

Dense case

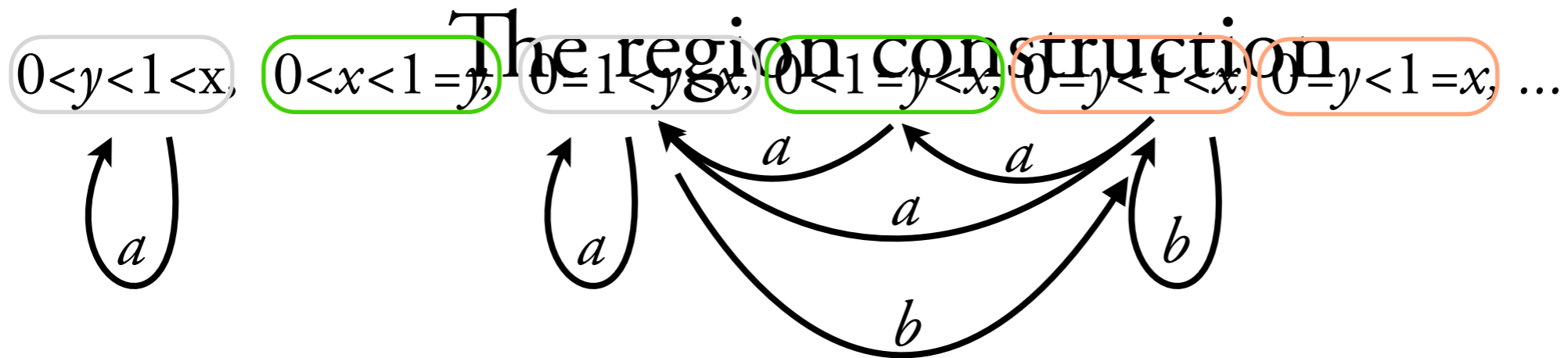
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



		<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>
<i>x</i>	5	5	5	5	5
<i>y</i>	0	1	2	4	0
	\setminus	\setminus	\setminus	\setminus	\setminus
<i>x</i>	2	2	2	2	2
<i>y</i>	0	1	1.2	1.3	0

bisimulation \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Dense case

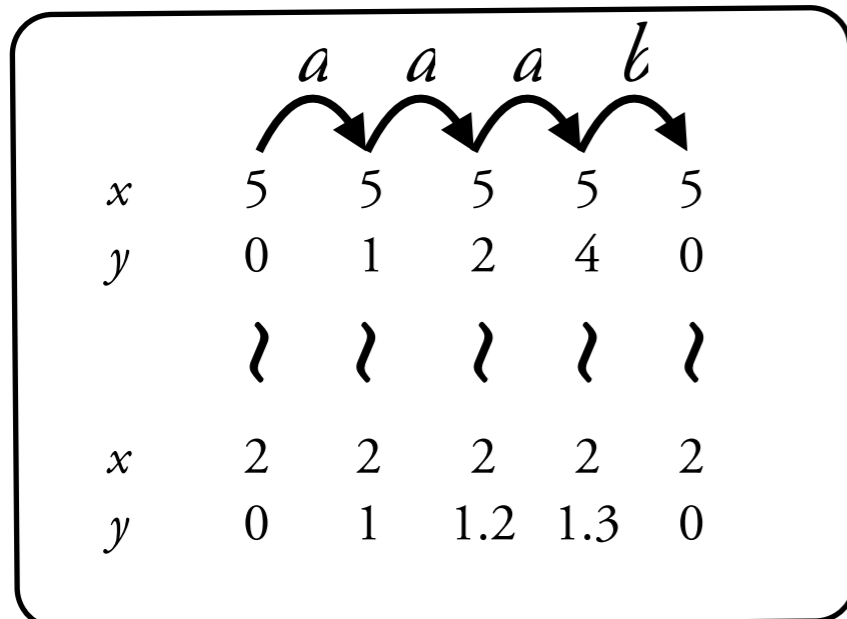
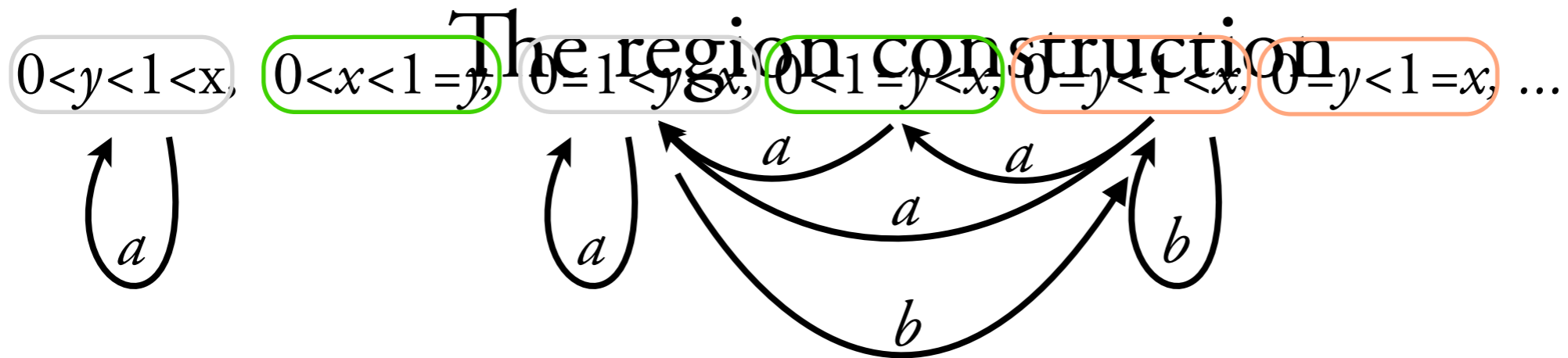
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

accepted language: $(a+b)^*a$

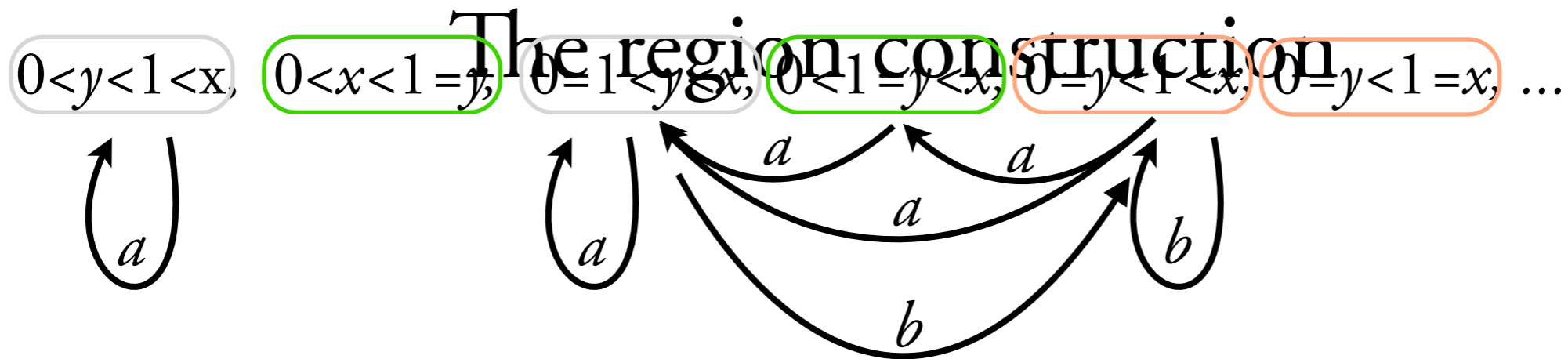
- $D = \mathbb{Q}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



		a	a	a	b
x	5	5	5	5	5
y	0	1	2	4	0
	\setminus	\setminus	\setminus	\setminus	\setminus
x	2	2	2	2	2
y	0	1	1.2	1.3	0

bisimulation \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

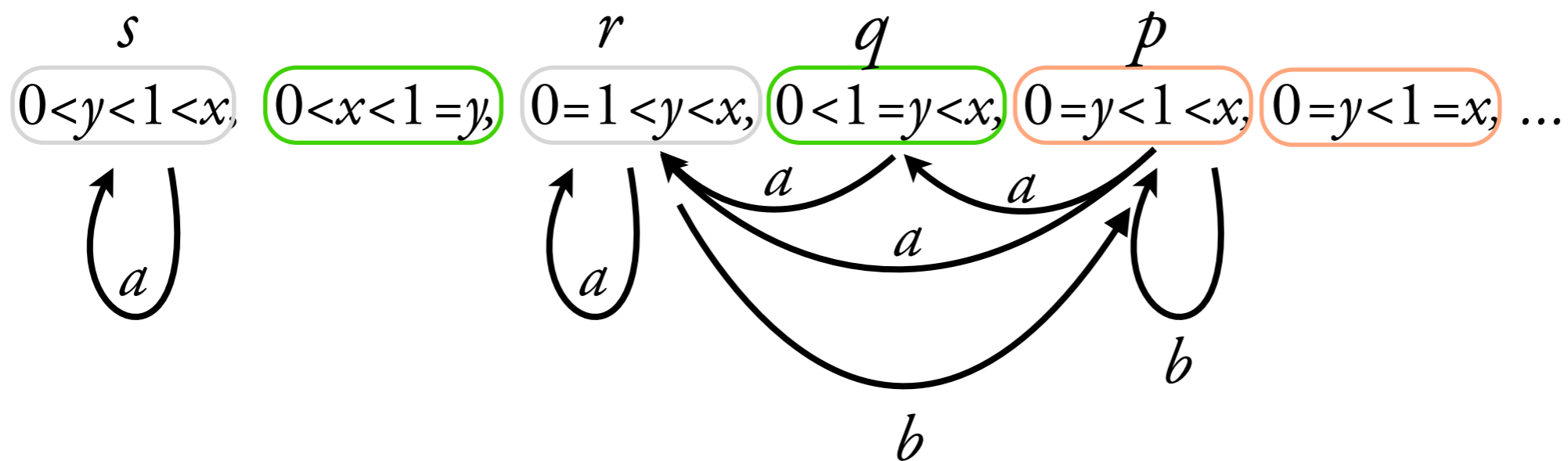
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

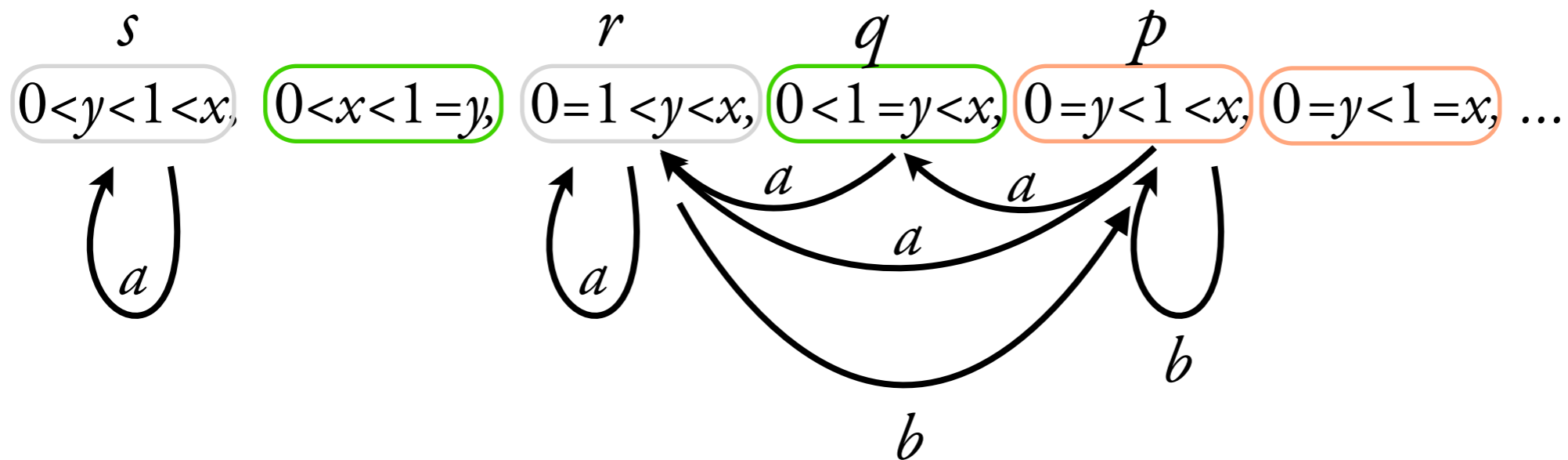
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



bisimulation \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

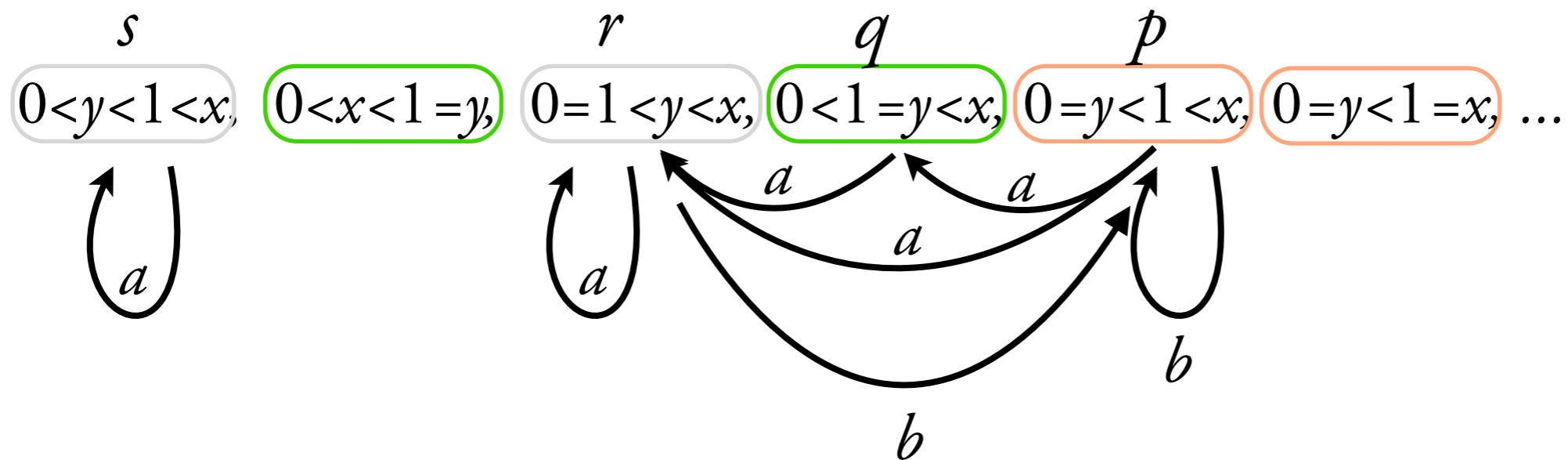
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



~~bisimulation~~ \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

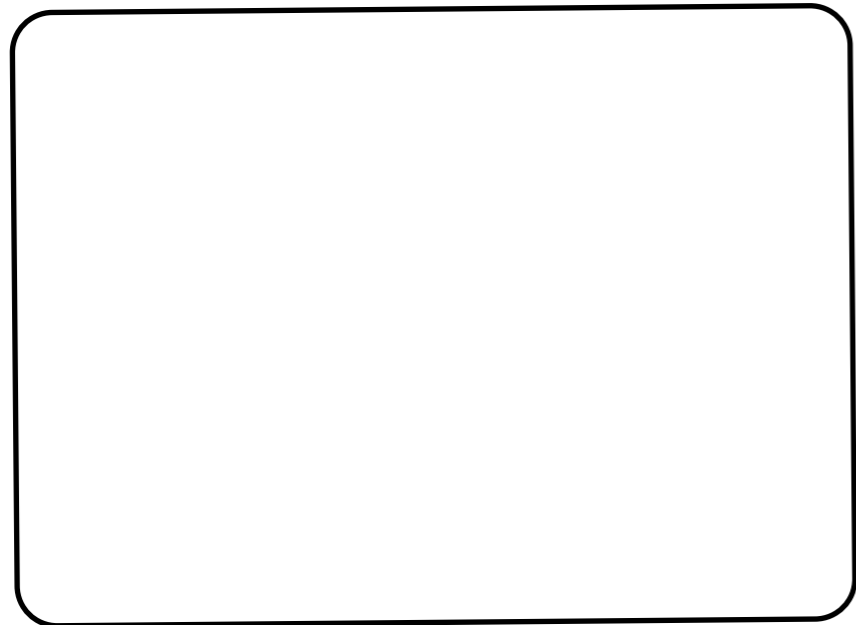
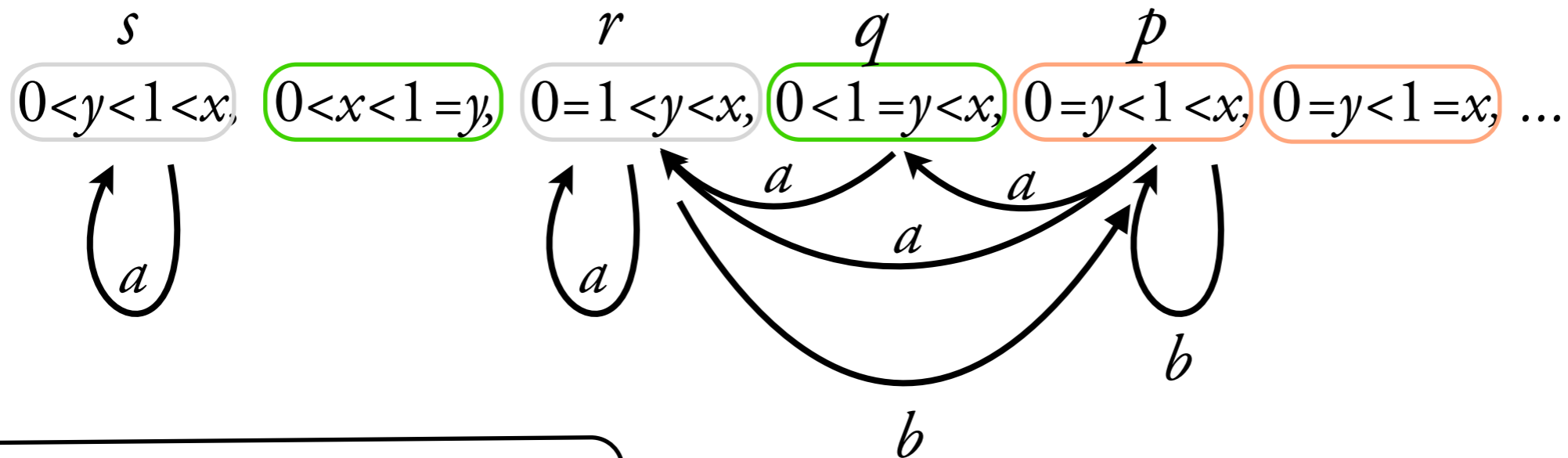
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



~~bisimulation~~ → runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

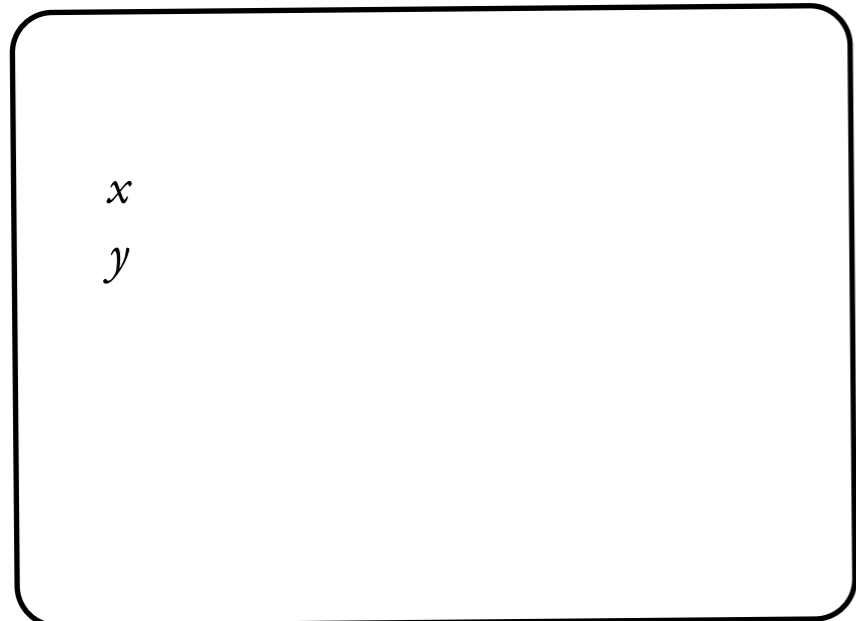
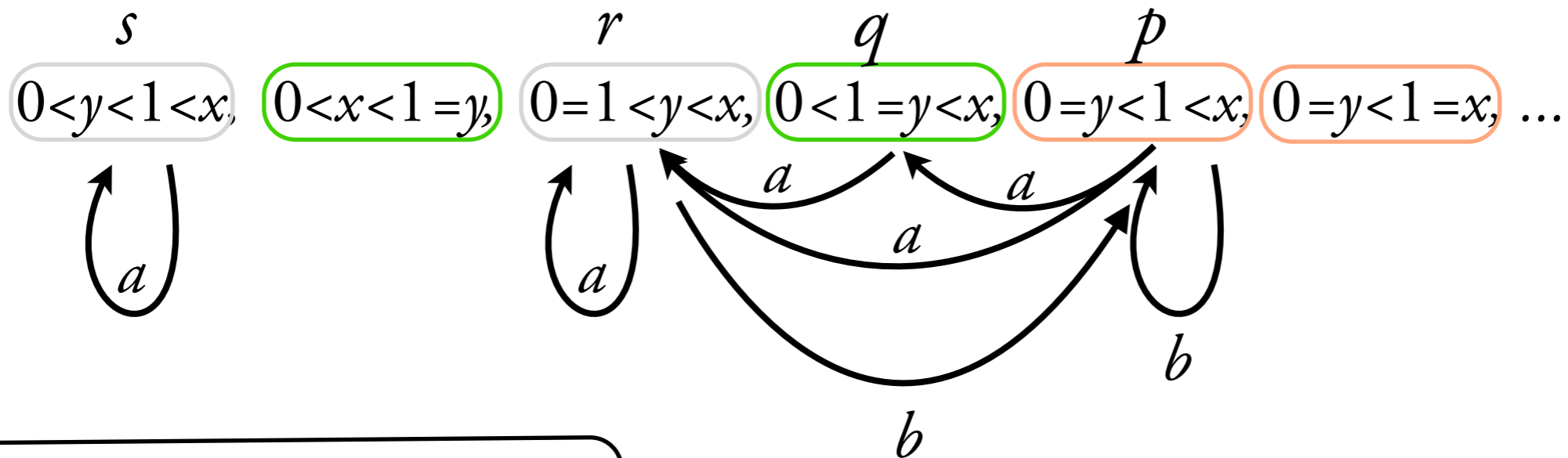
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



~~bisimulation~~ \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

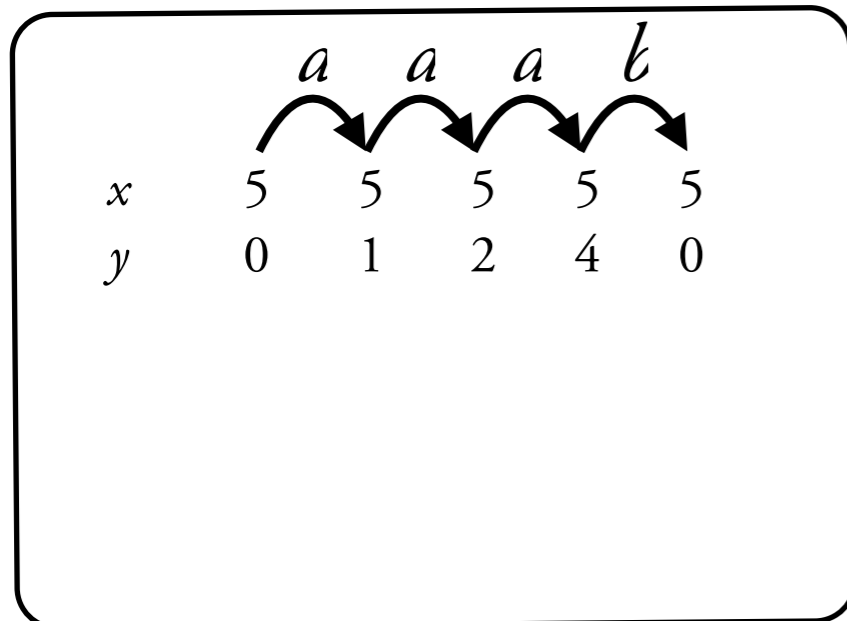
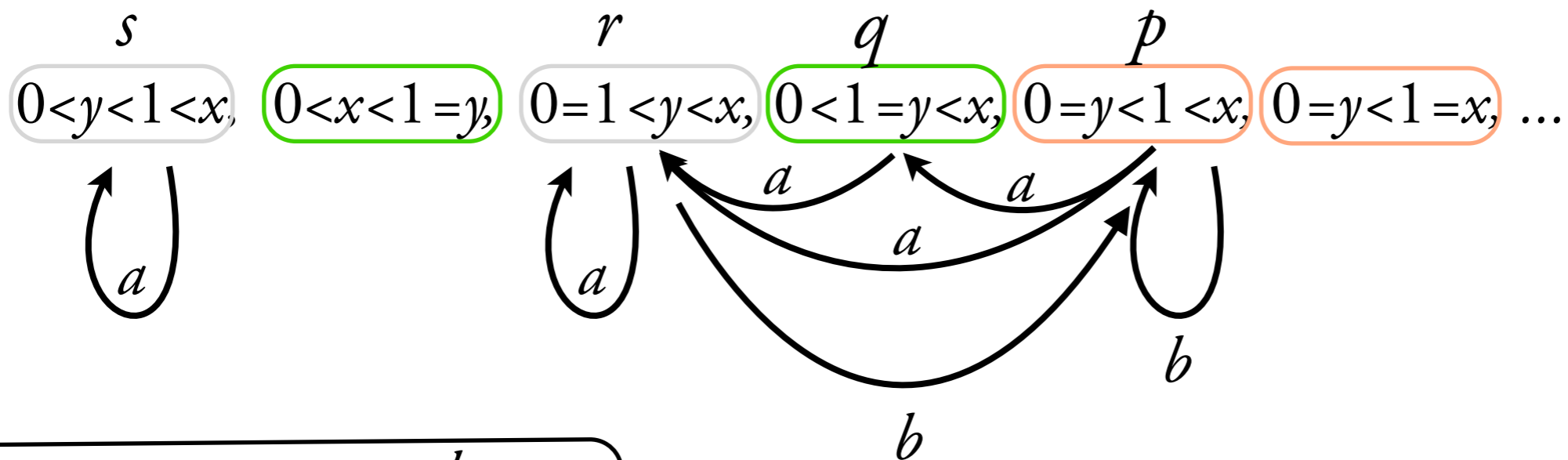
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



~~bisimulation~~ \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

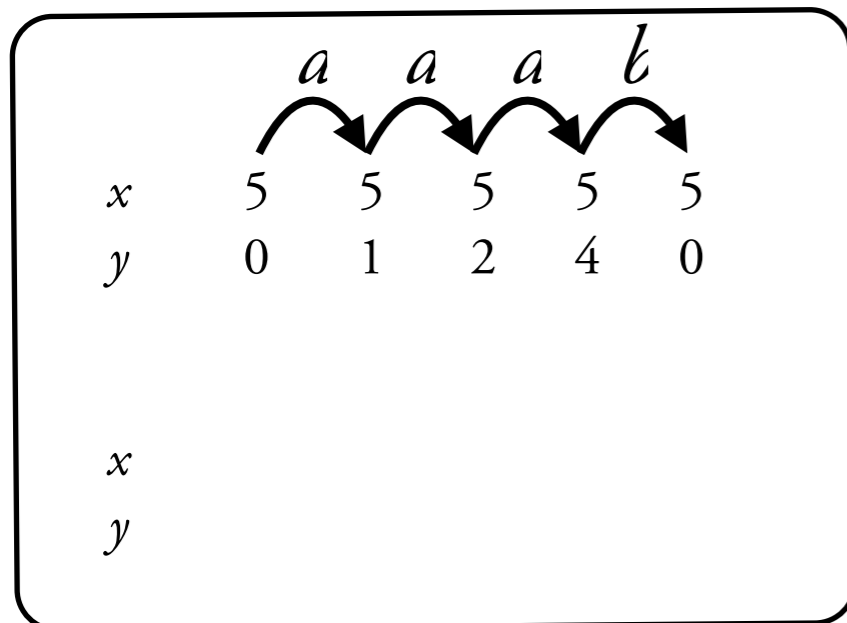
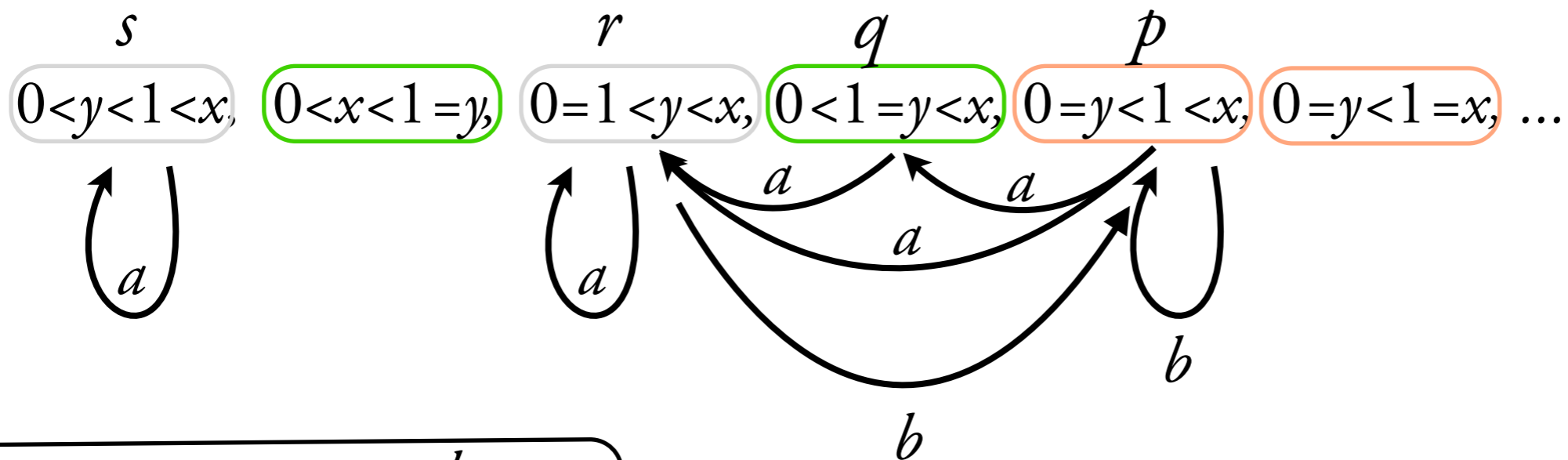
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



~~bisimulation~~ \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

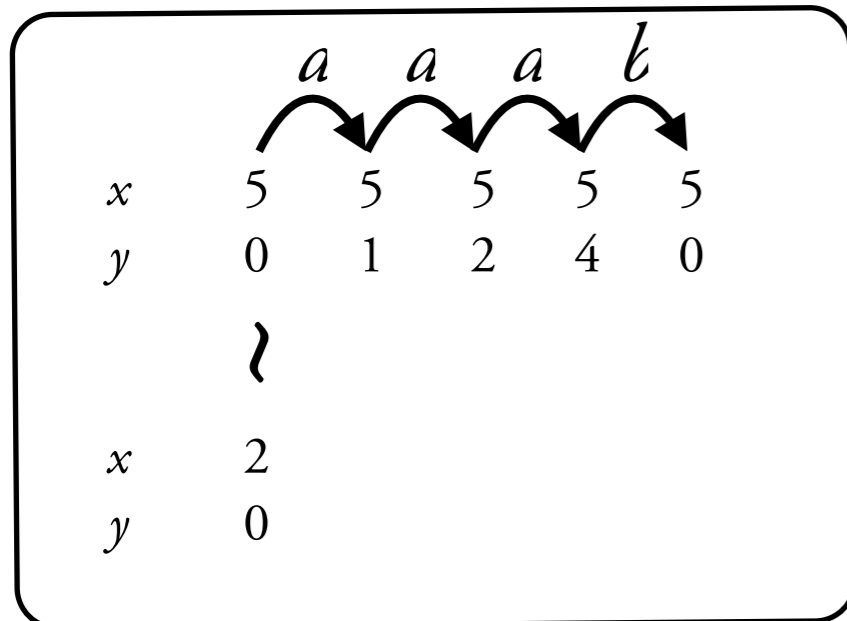
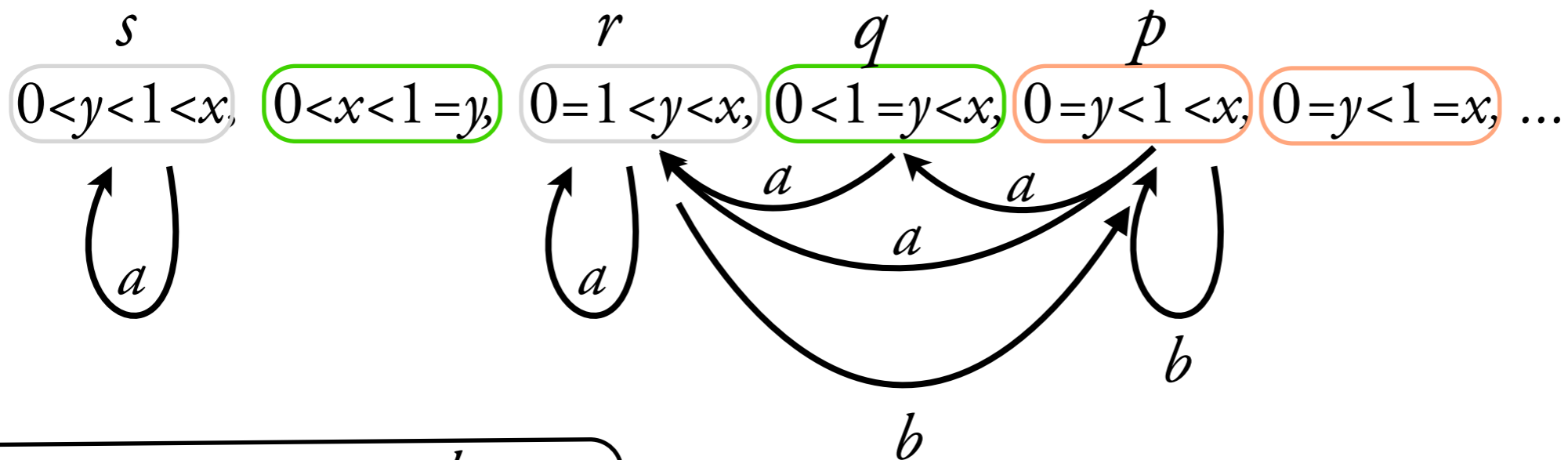
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



~~bisimulation~~ \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

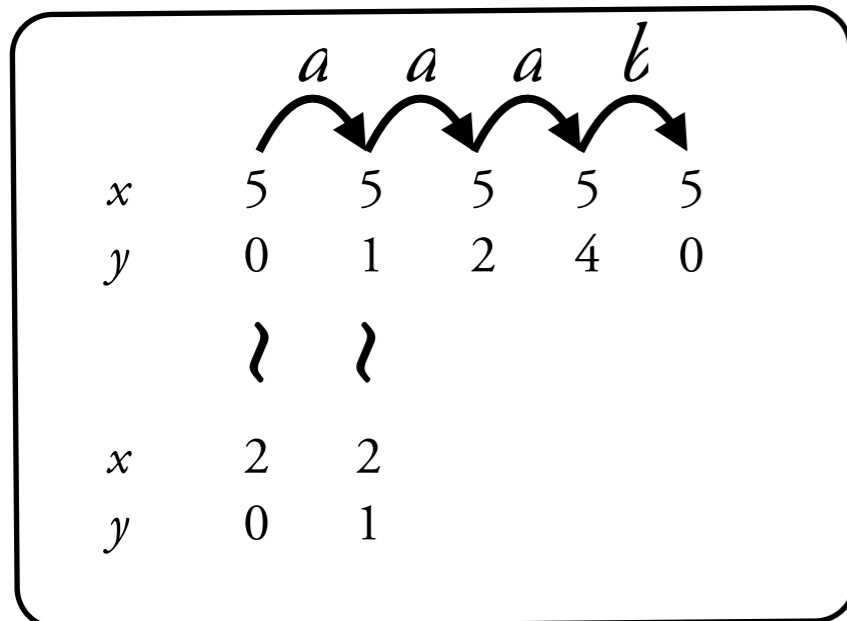
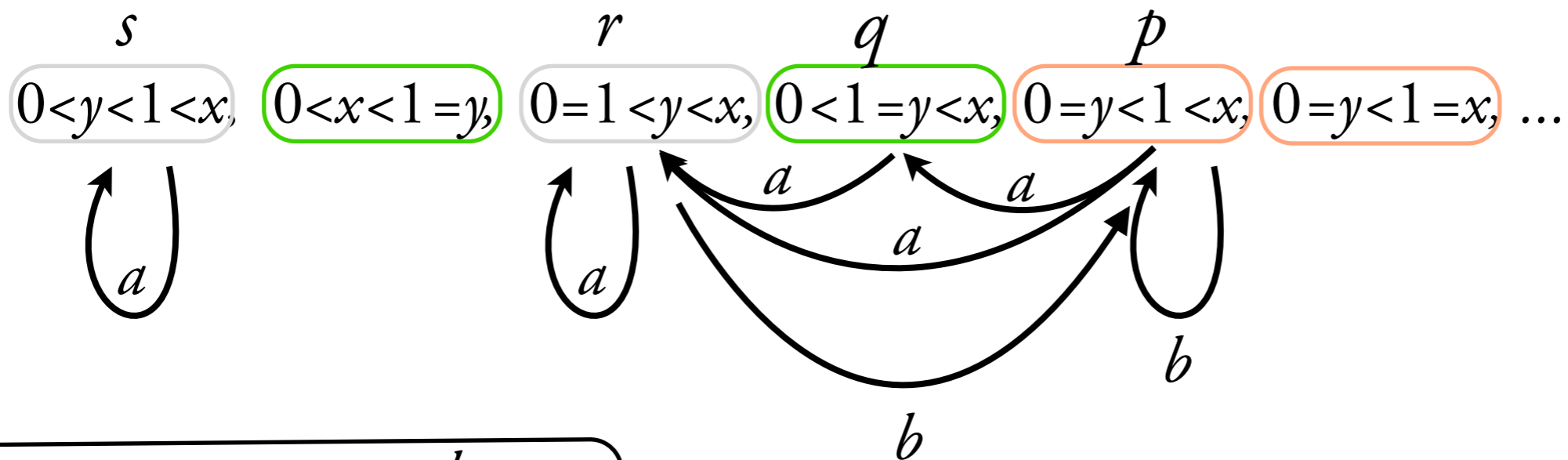
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



~~bisimulation~~ \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

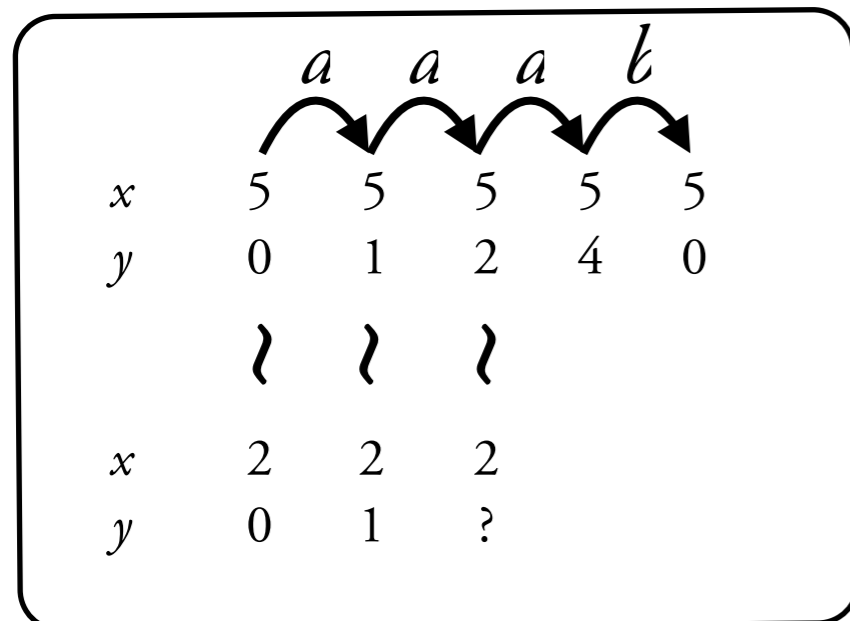
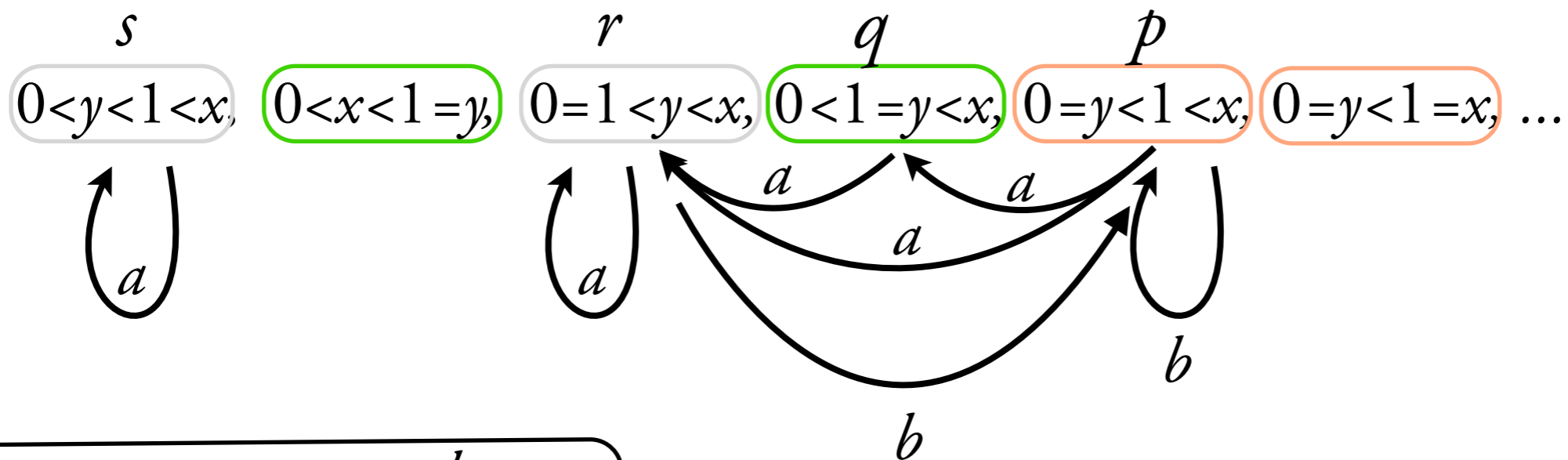
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



~~bisimulation~~ \rightarrow runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

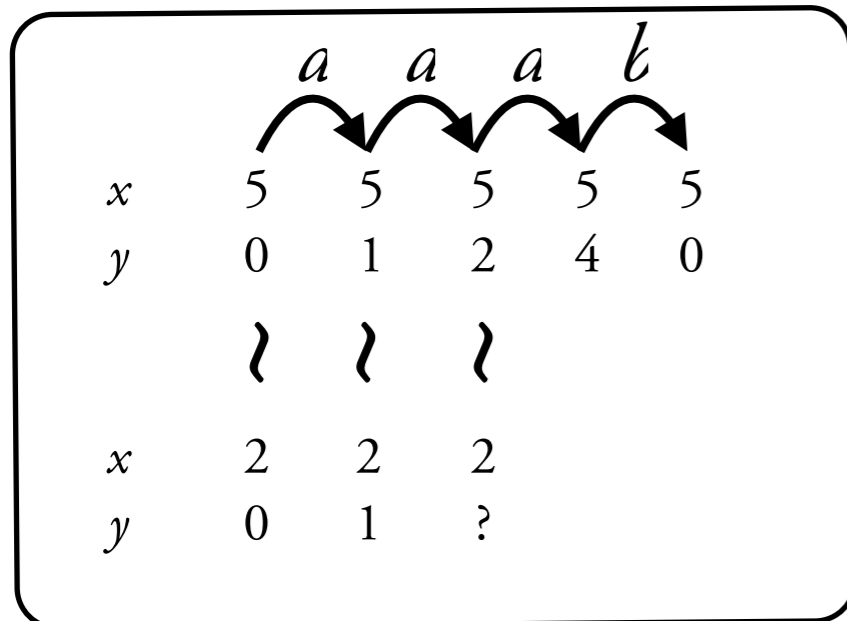
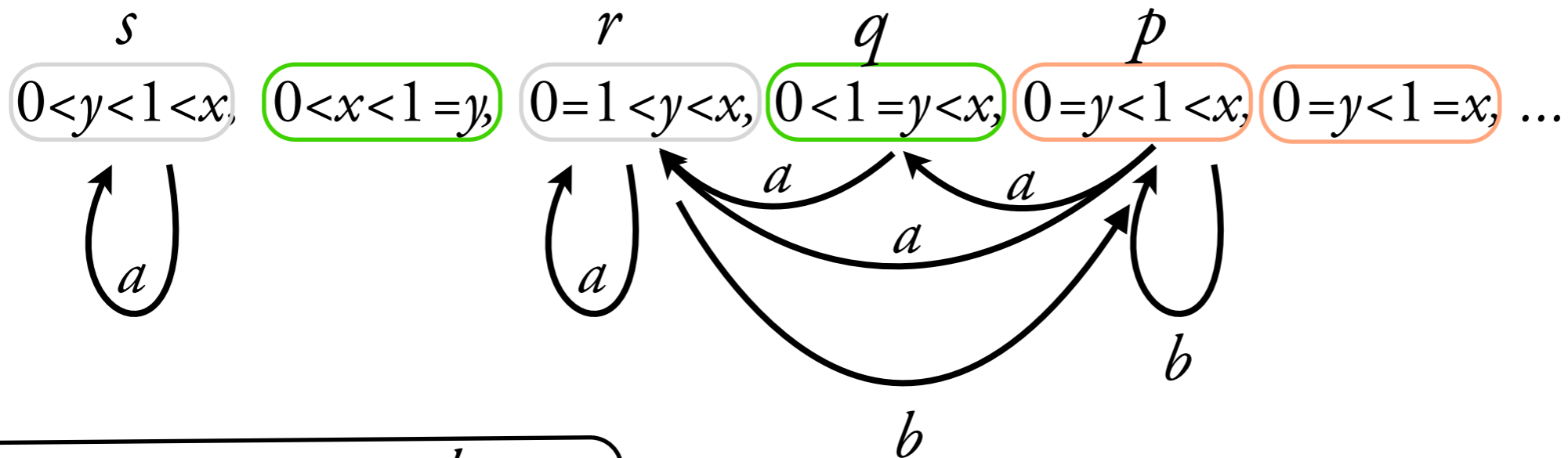
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 1)$$



but: in any cell and any n we can find configurations which are *sufficiently good* for all runs of length $\leq n$
 \rightarrow *finite* runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

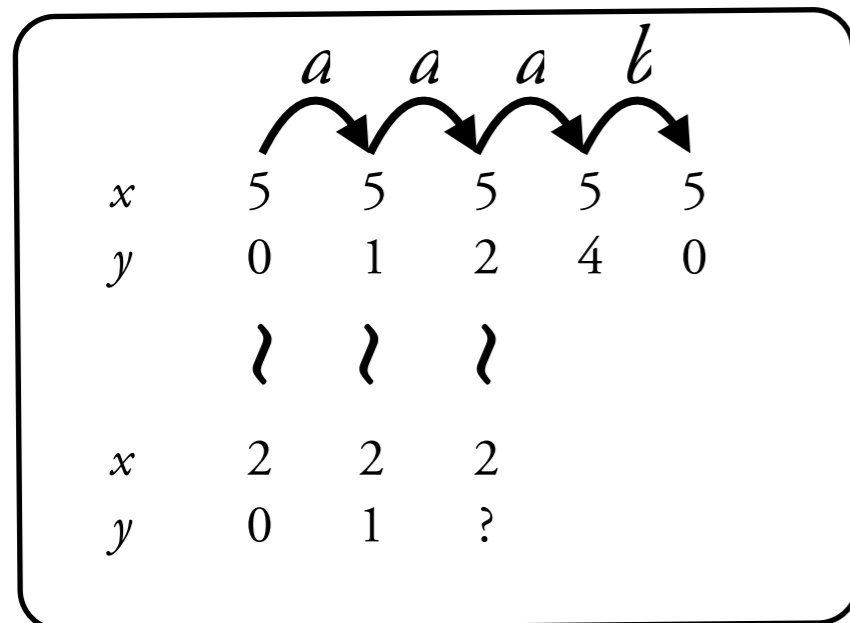
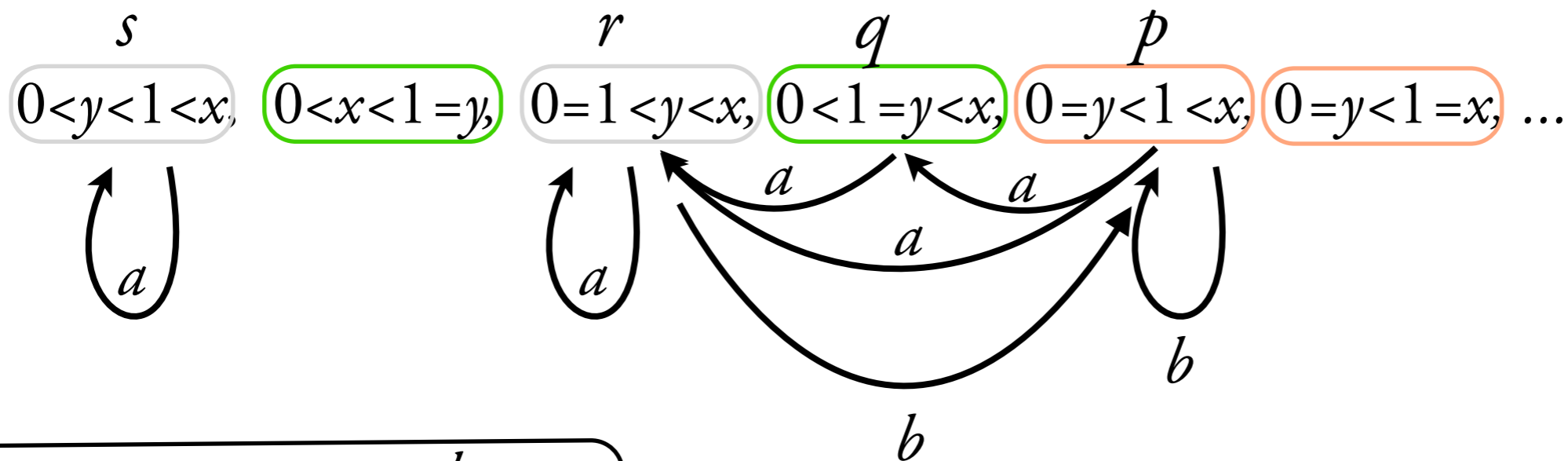
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 3)$$



but: in any cell and any n we can find configurations which are *sufficiently good* for all runs of length $\leq n$

\rightarrow *finite* runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

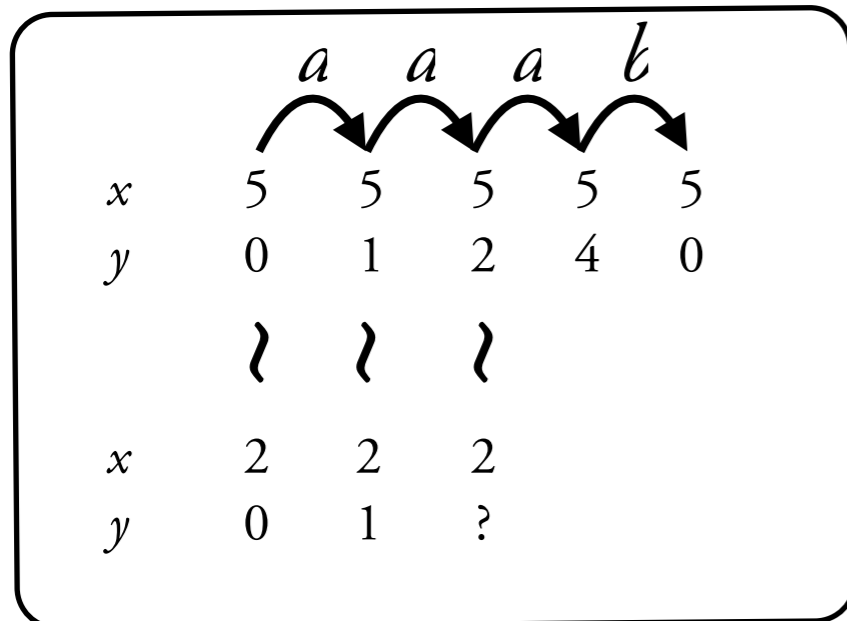
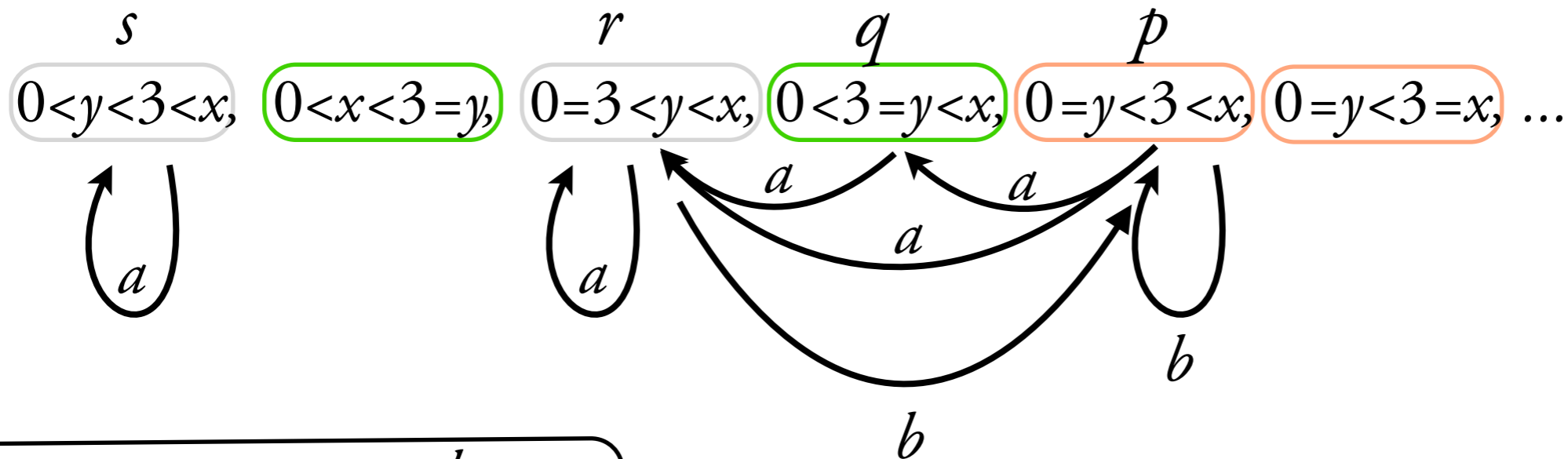
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 3)$$



but: in any cell and any n we can find configurations which are *sufficiently good* for all runs of length $\leq n$
 \rightarrow *finite* runs of the region automaton correspond to runs of \mathcal{A}

Discrete case

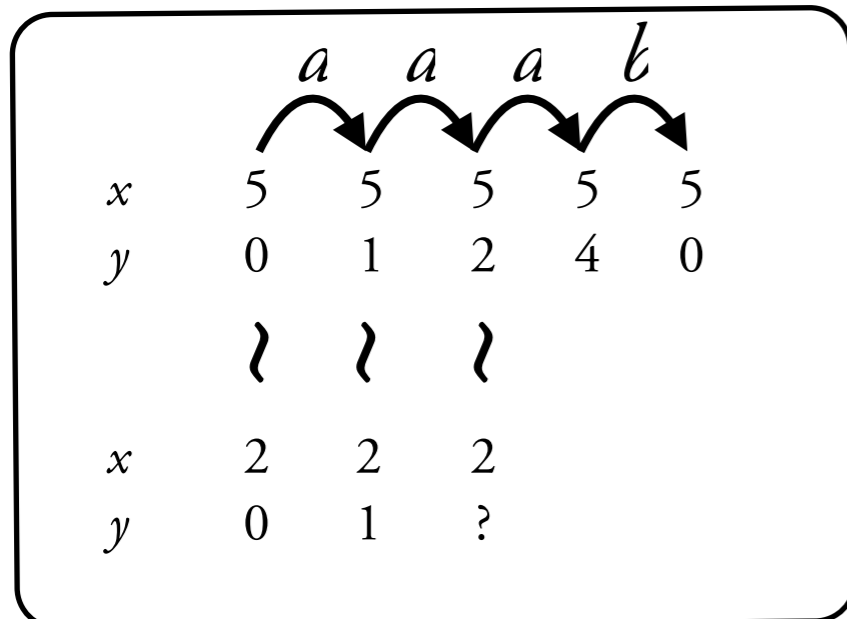
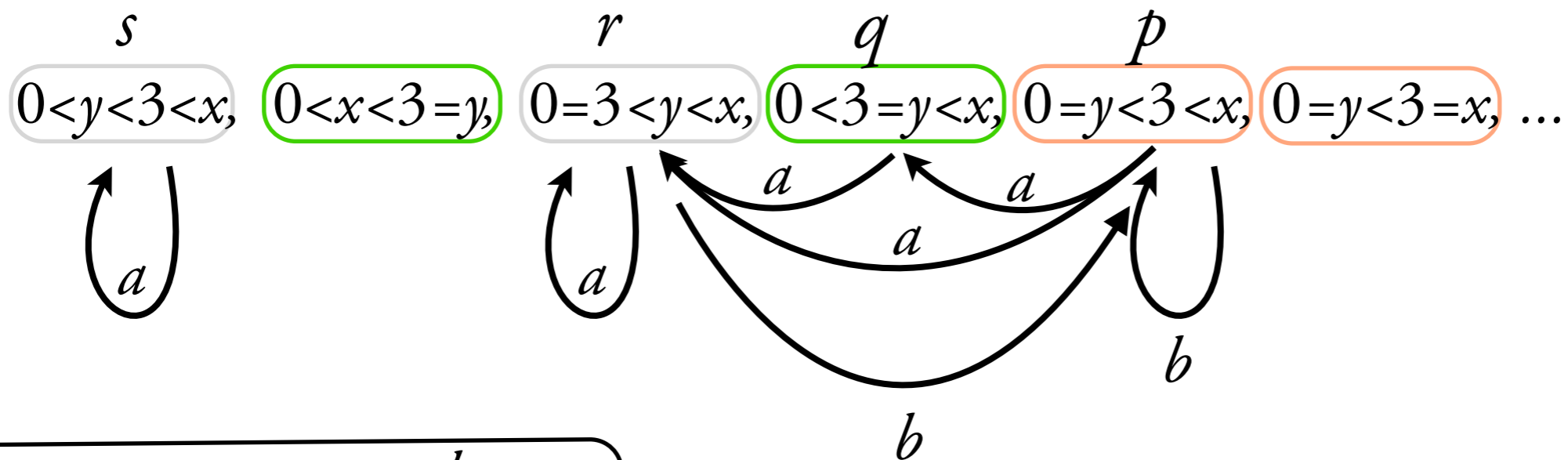
- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 3)$$



~~but: in any cell and any n we can find configurations which are *sufficiently good* for all runs of length $\leq n$
 → *finite runs* of the region automaton correspond to runs of \mathcal{A}~~

Discrete case

- $D = \mathbb{N}$
- finite words
- no database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 3)$$

Solution: consider 0,1,2,3 as special values (constants)

Discrete case

- $D = \mathbb{N}$
- finite words
- no database

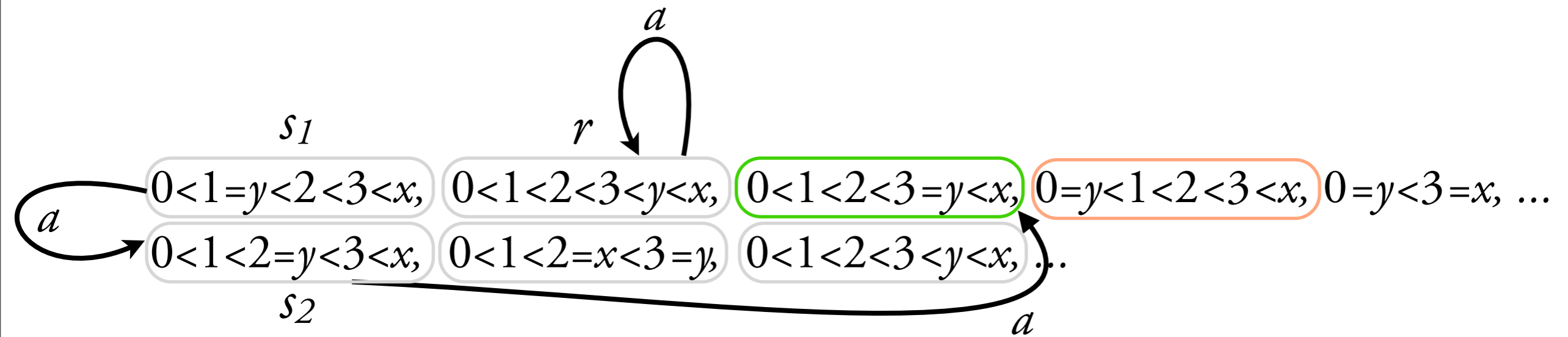
$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 3)$$

Solution: consider 0,1,2,3 as special values (constants)



Discrete case

- $D = \mathbb{N}$
- finite words
- no database

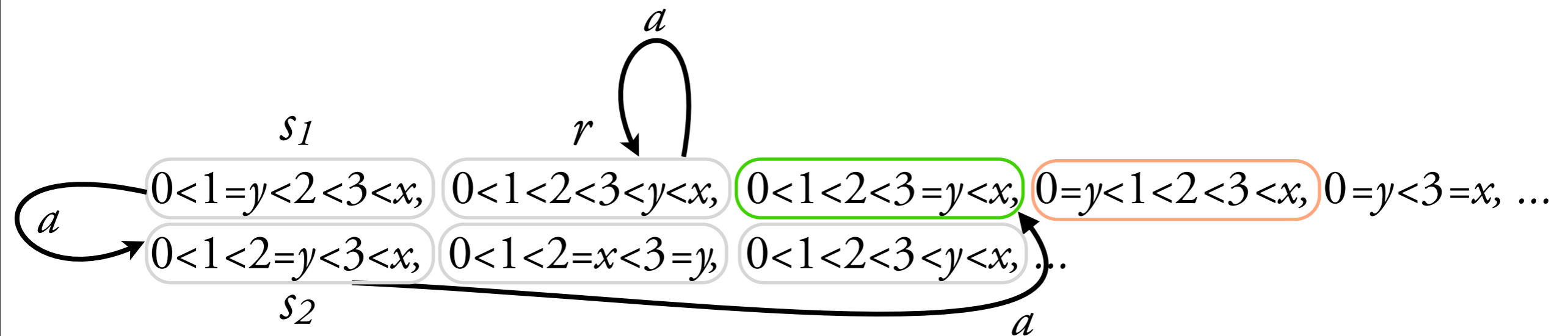
$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 3)$$

Solution: consider 0,1,2,3 as special values (constants)



Then the obtained cell automaton simulates \mathcal{A} faithfully

Discrete case

- $D = \mathbb{N}$
- finite words
- no database

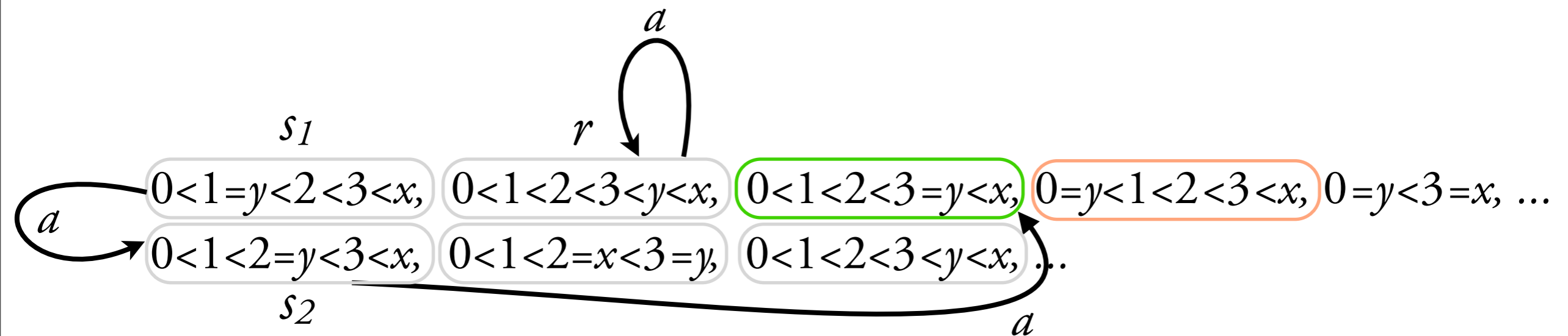
$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x)$$

$$\delta_b: (x' = x) \wedge (y' = 0)$$

$$\tau_F: (y = 3)$$

Solution: consider 0,1,2,3 as special values (constants)



Then the obtained cell automaton simulates \mathcal{A} faithfully

Theorem. For any linearly ordered structure there is a finite number of special values

This solves the case of finite runs, with no database.

This solves the case of finite runs, with no database.

Theorem. For all \mathcal{D} , \mathcal{D} -automata over finite words accept regular languages.

Theorem. Emptiness of \mathcal{D} -automata is decidable* in PSPACE

**For any reasonable linearly ordered structure - there should be an efficient algorithm which can compute the special values, given some initial constants*

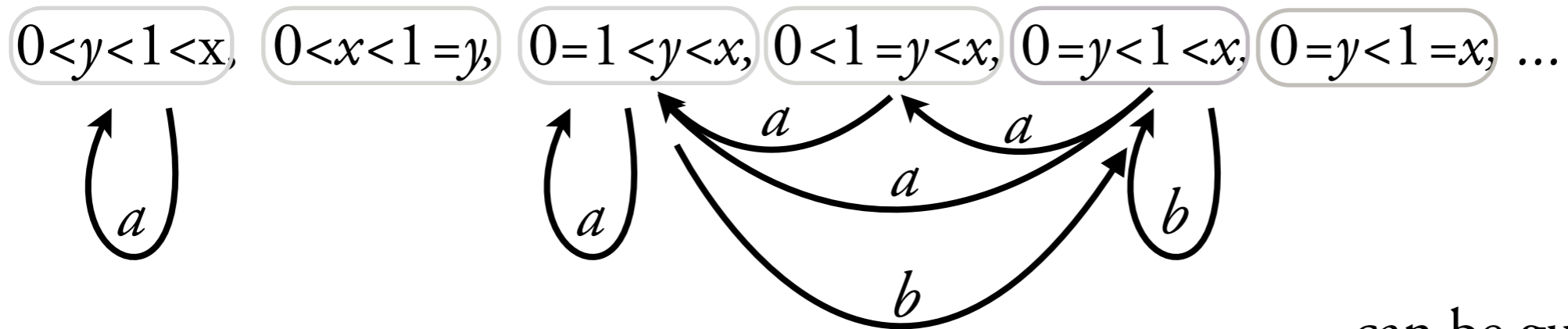
Adding a database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



can be guaranteed
to be new

is forced to be zero

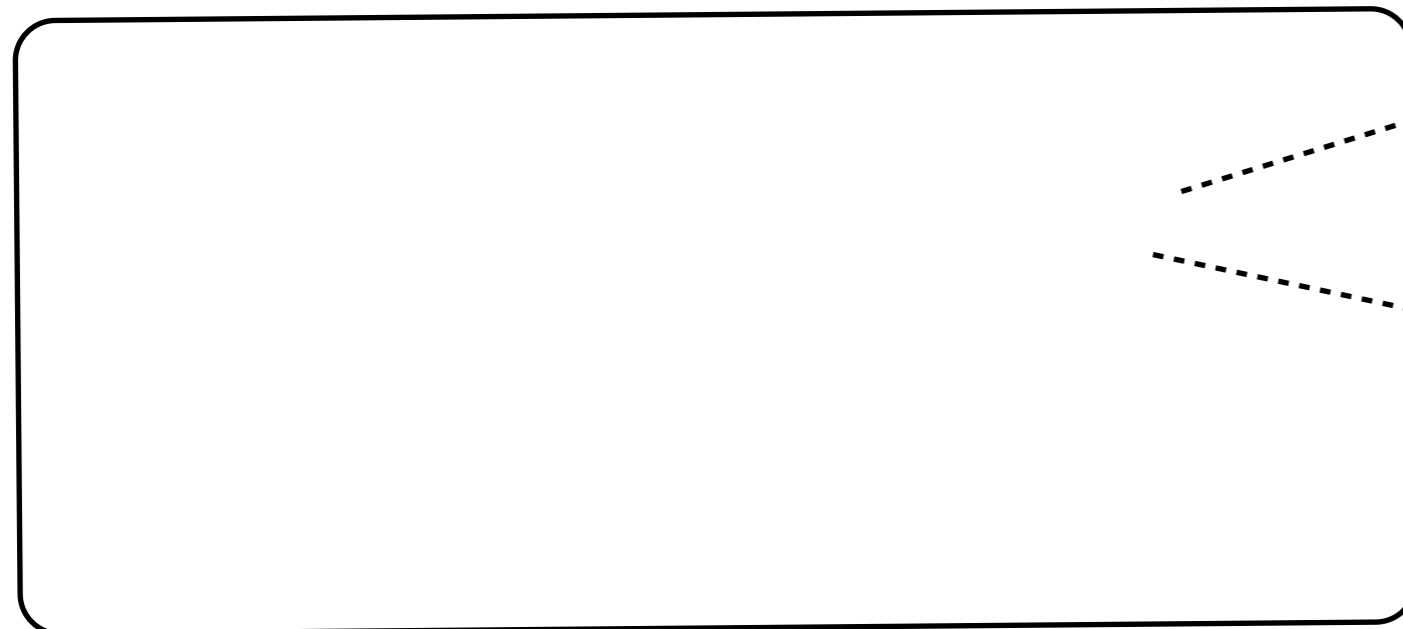
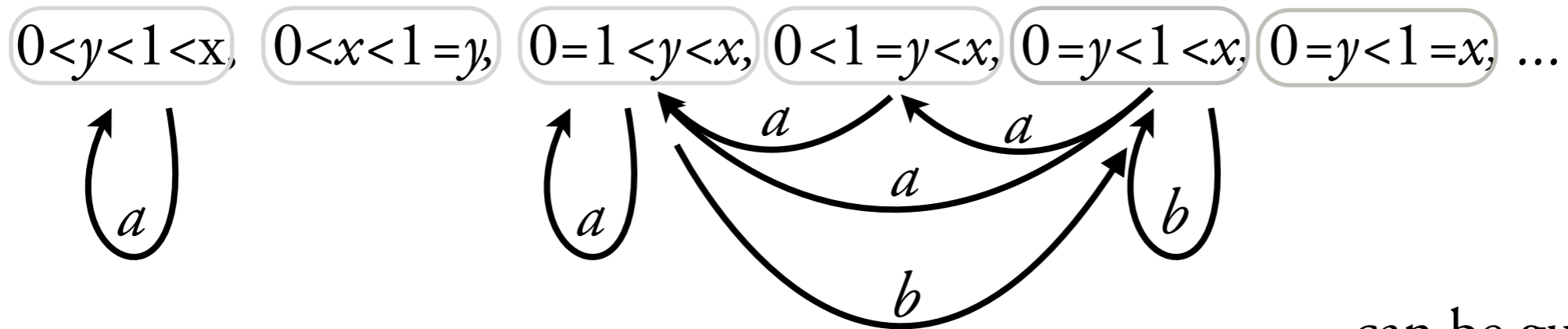
Adding a database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



can be guaranteed
to be new

is forced to be zero

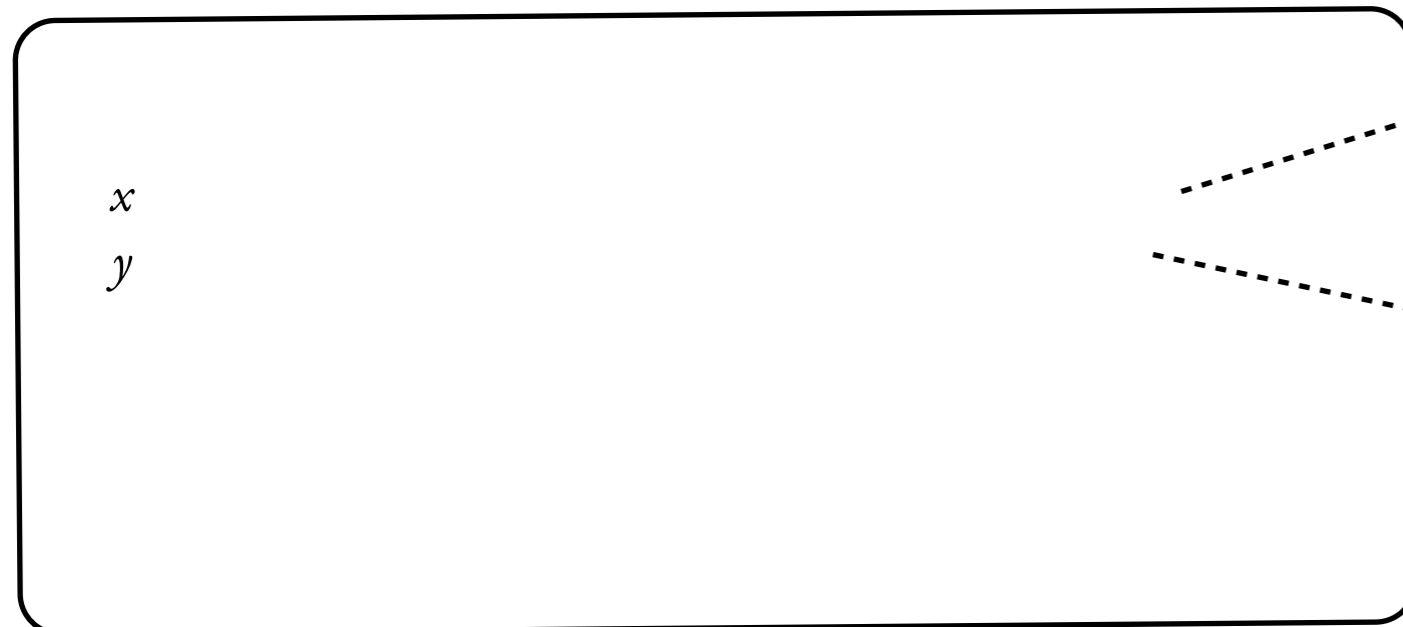
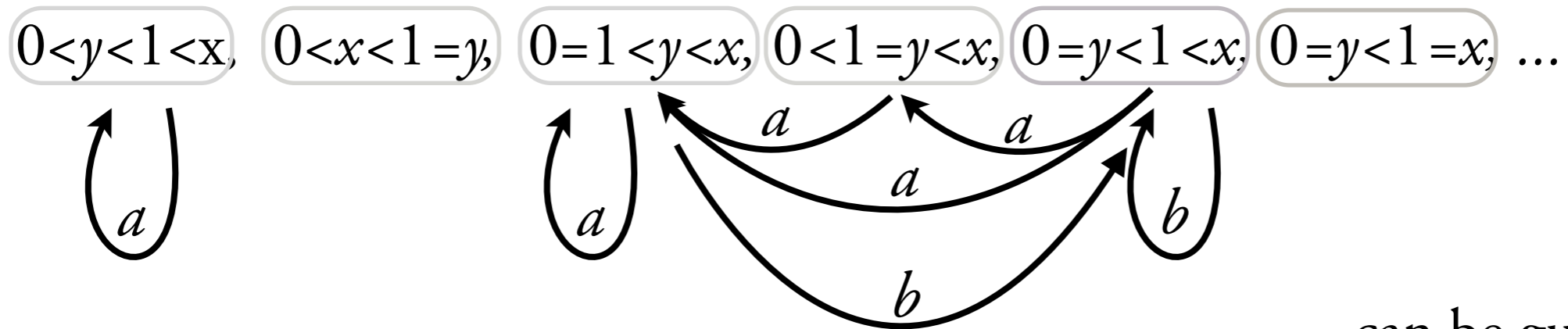
Adding a database

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



can be guaranteed to be new

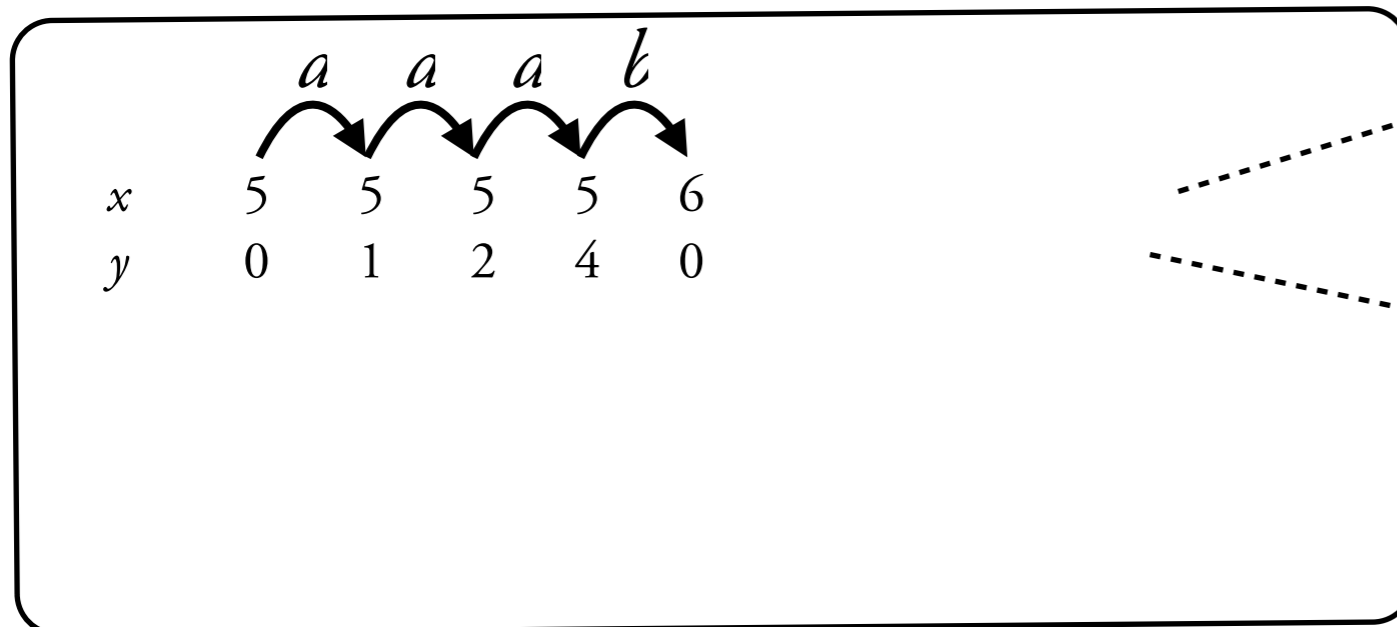
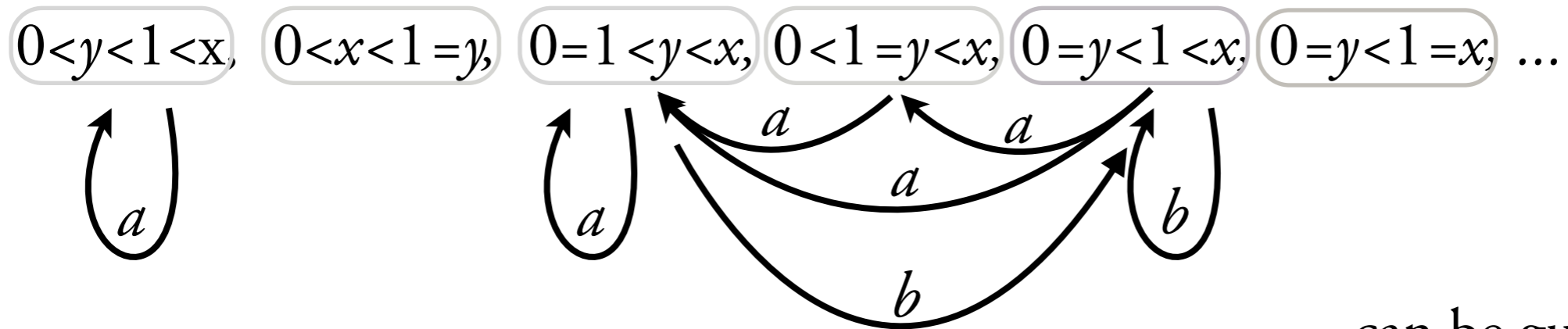
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



can be guaranteed to be new

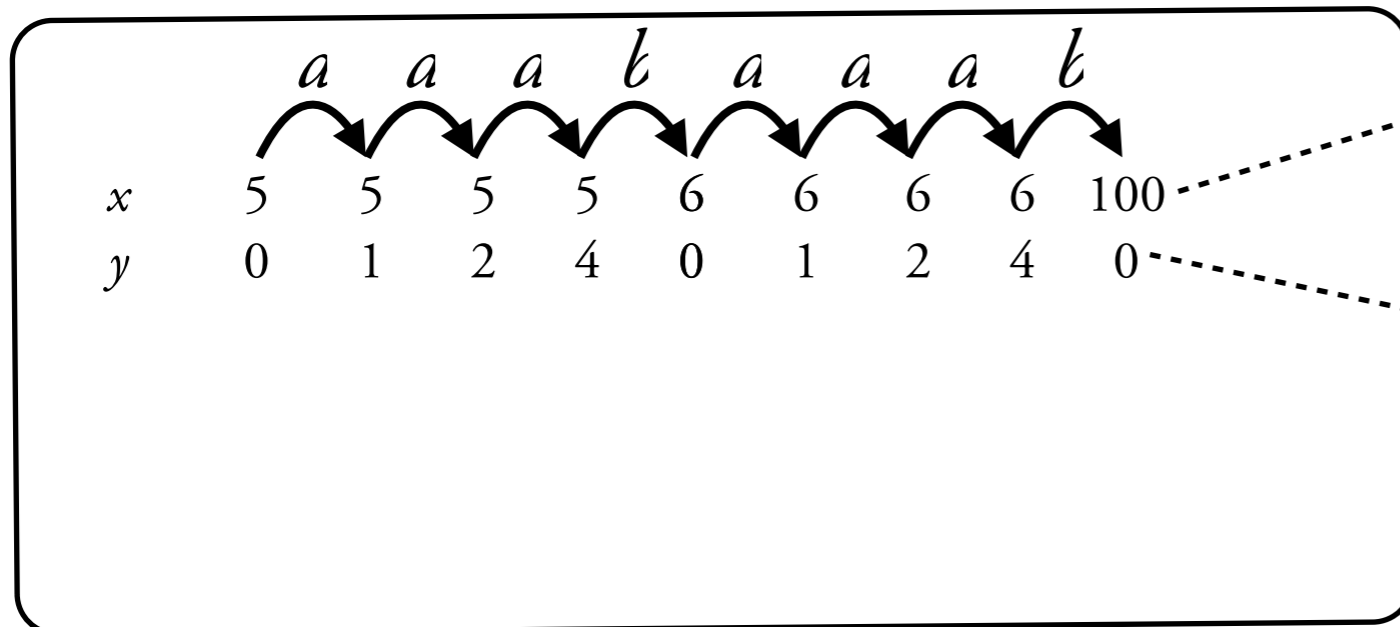
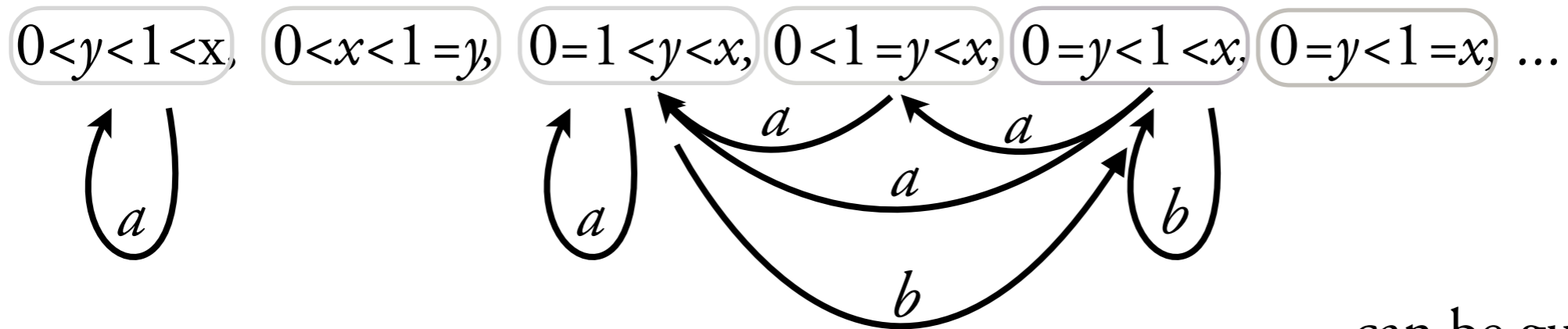
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



can be guaranteed to be new

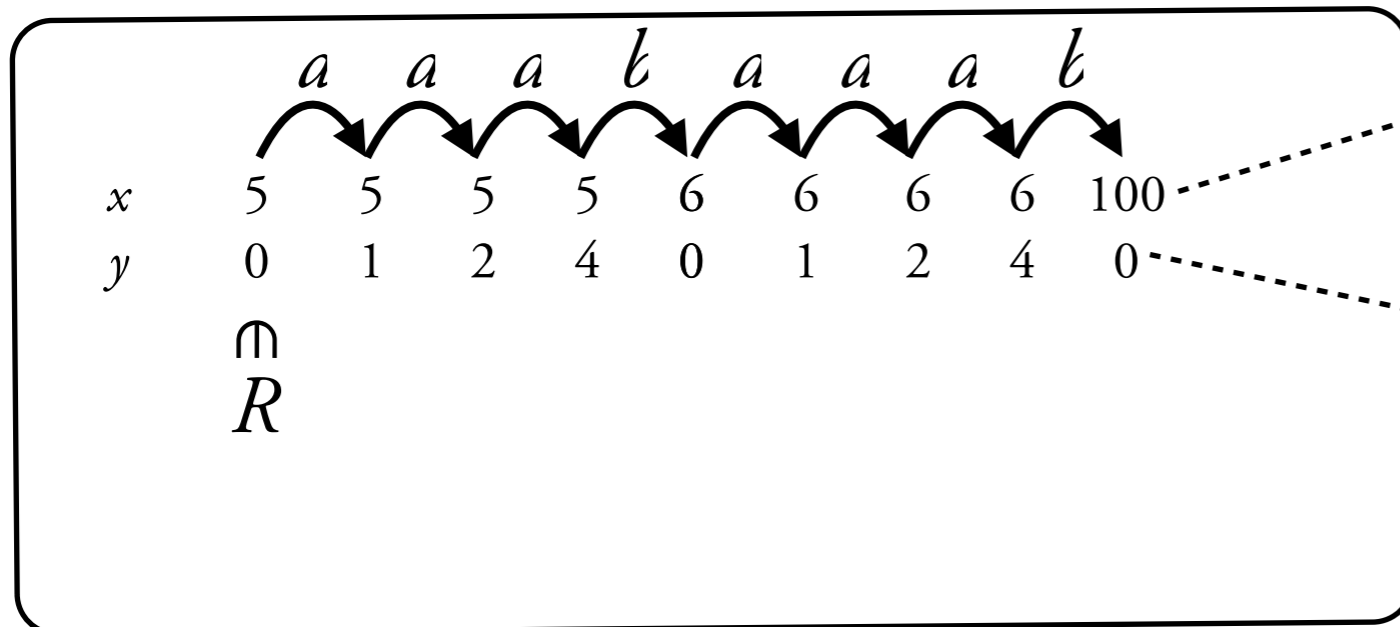
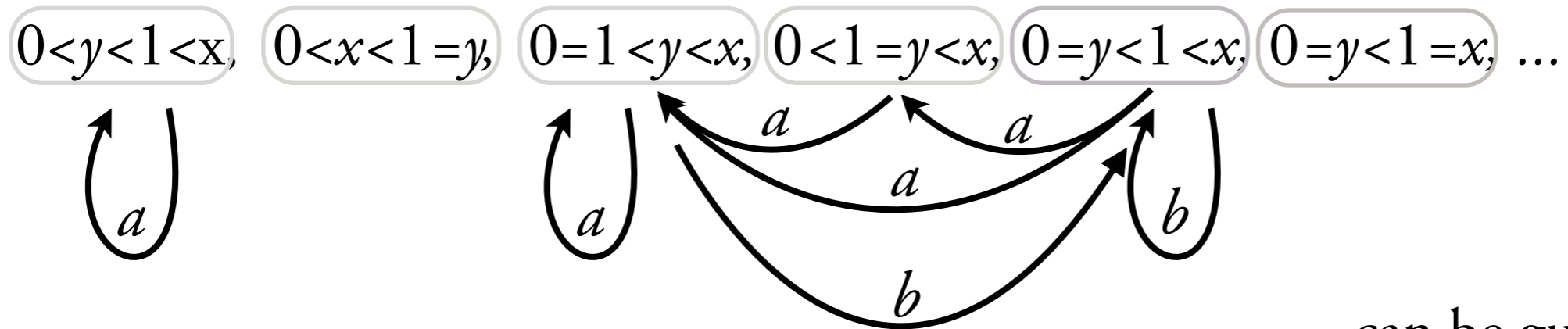
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



can be guaranteed to be new

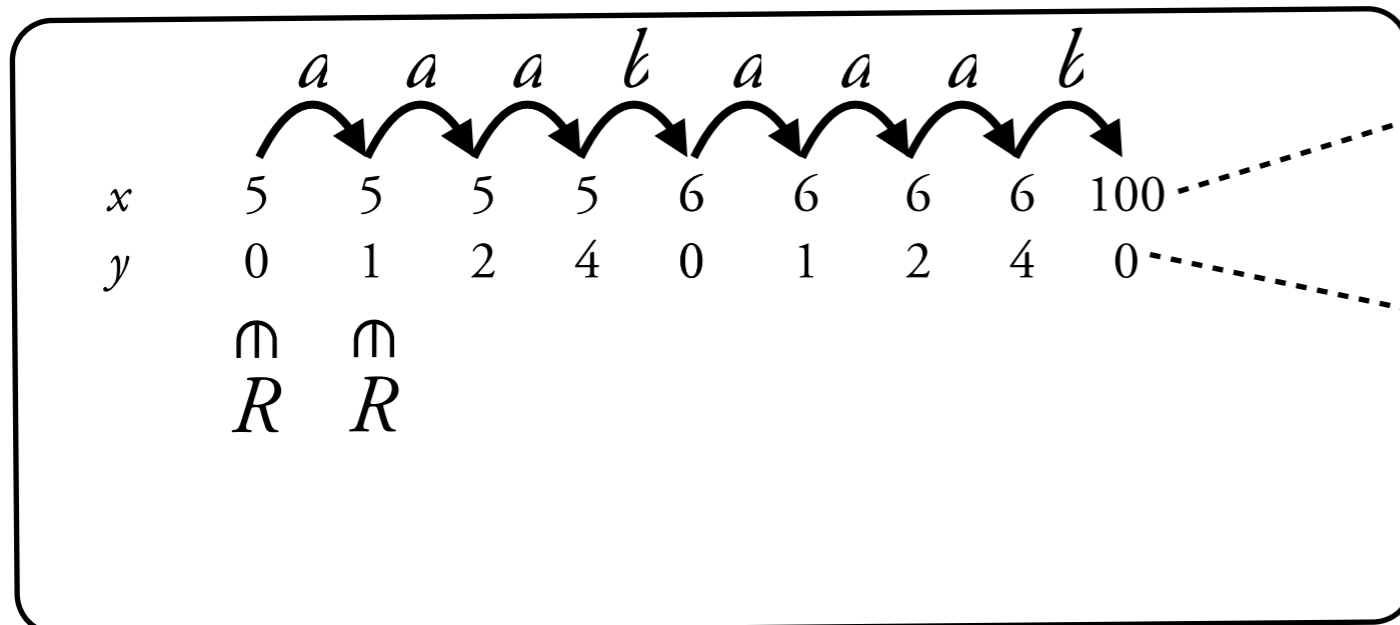
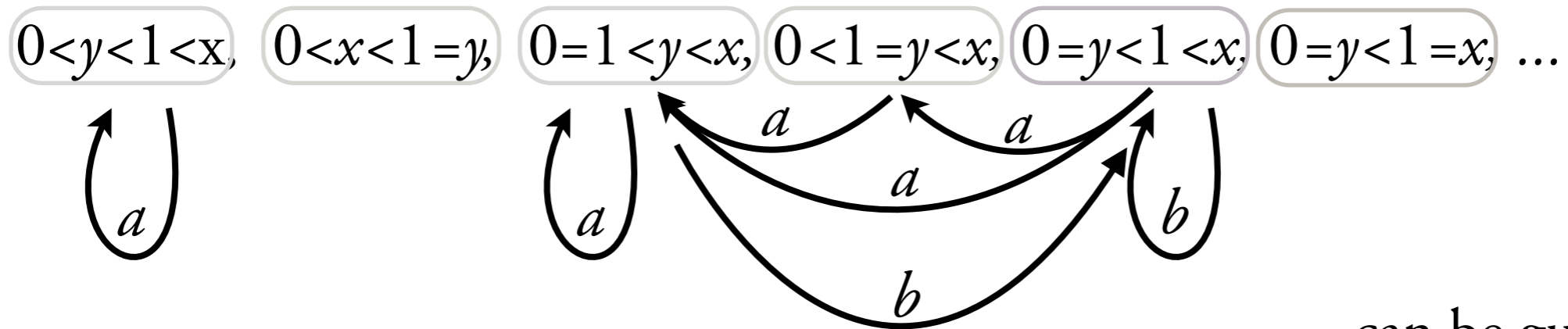
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



can be guaranteed to be new

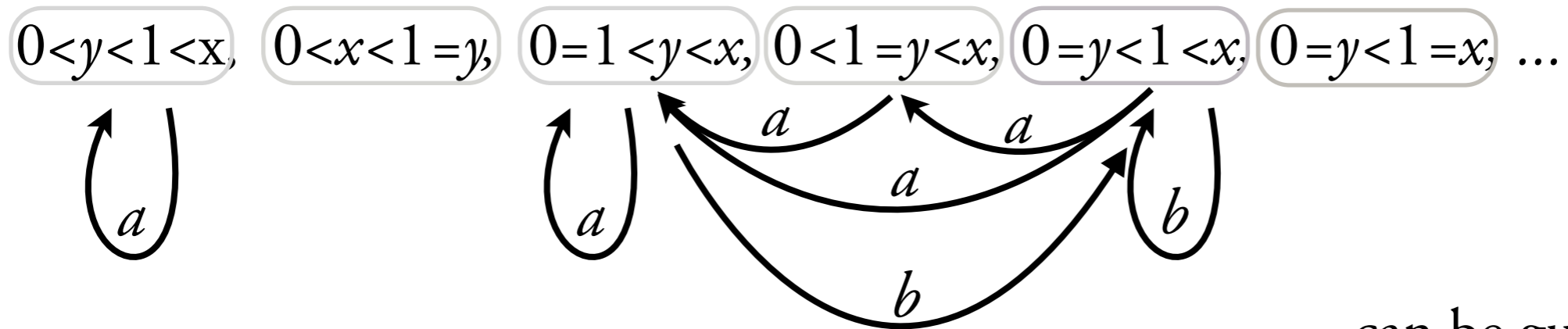
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	
<i>x</i>	5	5	5	5	6	6	6	6	100
<i>y</i>	0	1	2	4	0	1	2	4	0
	\cap	\cap	\cap						
	<i>R</i>	<i>R</i>	<i>R</i>						

can be guaranteed to be new

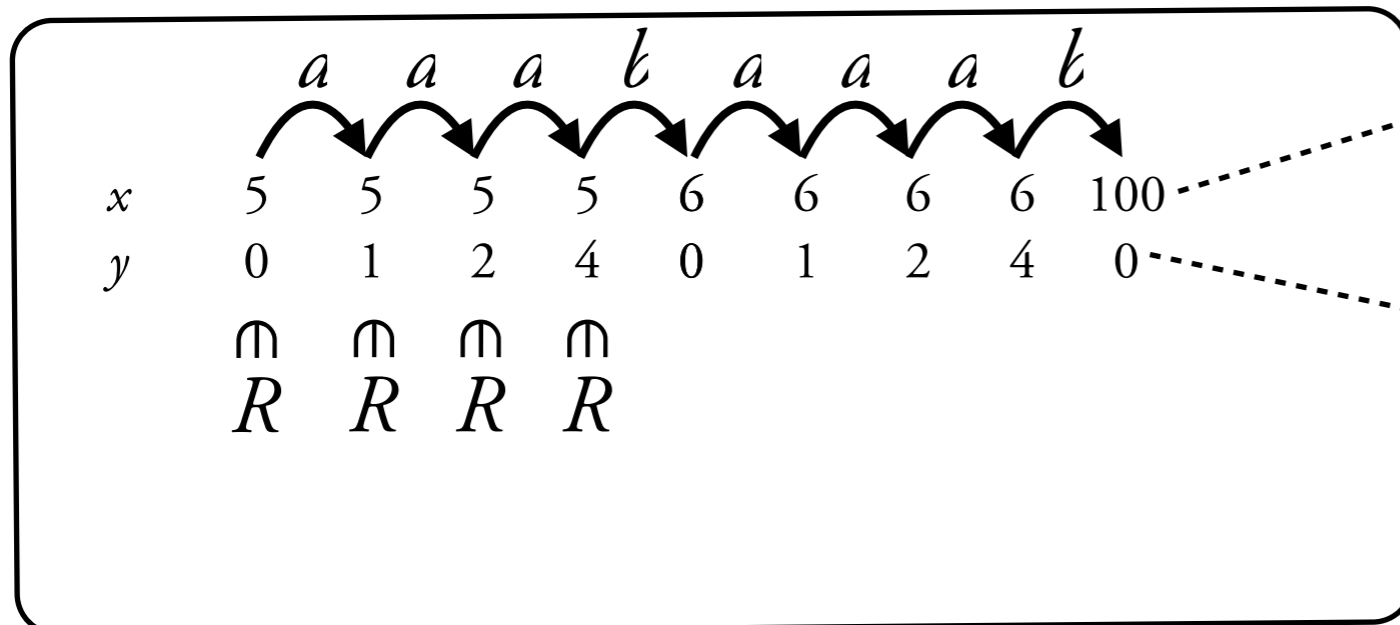
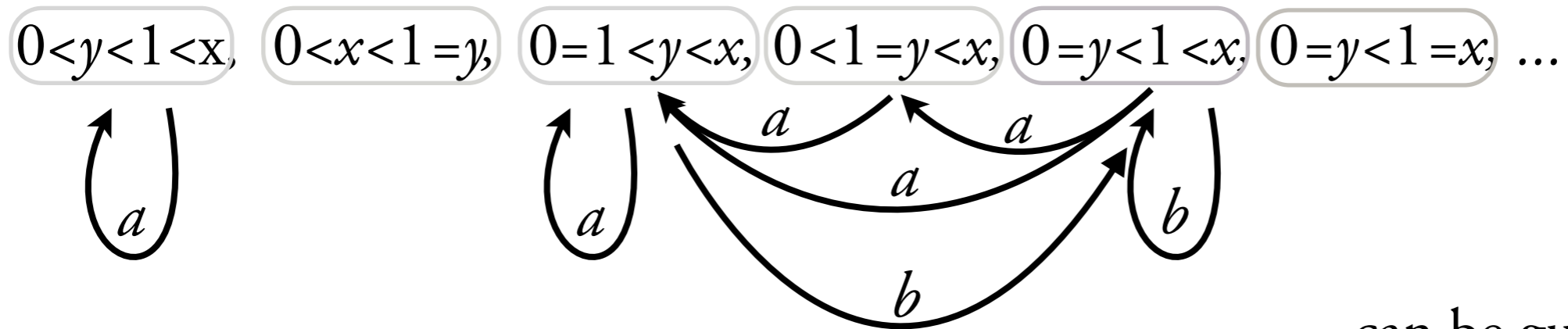
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



can be guaranteed to be new

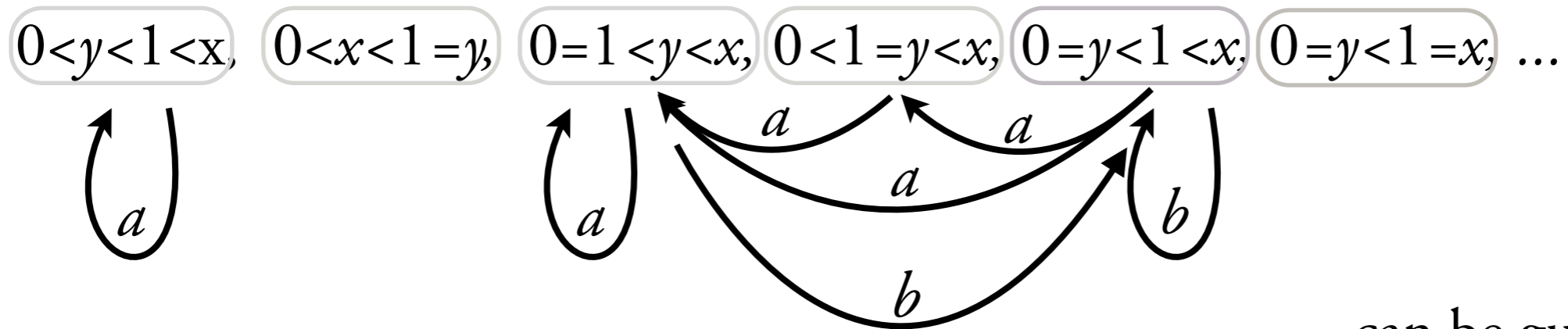
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



		<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	
<i>x</i>	5	5	5	5	6	6	6	6	100	
<i>y</i>	0	1	2	4	0	1	2	4	0	
	\cap	\cap	\cap	\cap	\cap	\cap	\cap	\cap	\cap	
	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	

can be guaranteed to be new

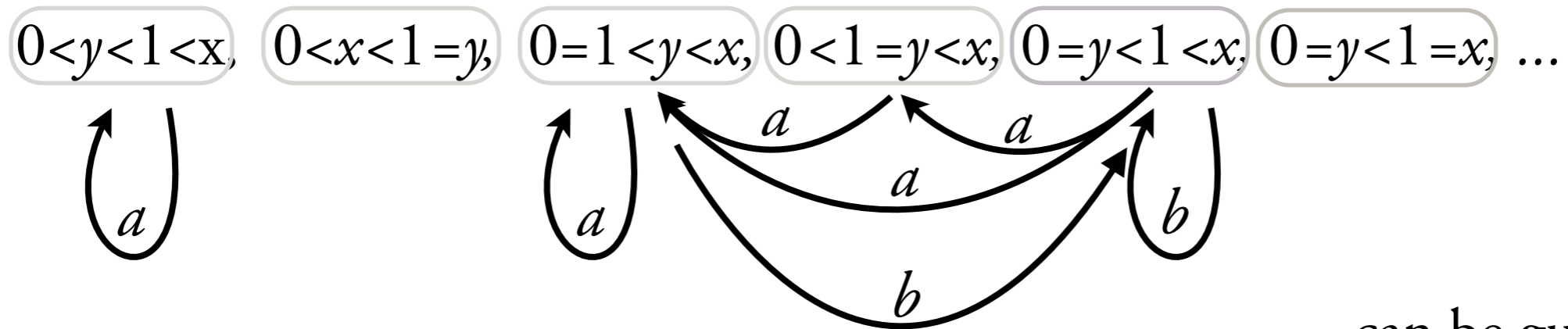
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	
<i>x</i>	5	5	5	5	6	6	6	6	100
<i>y</i>	0	1	2	4	0	1	2	4	0
	\cap	\cap	\cap	\cap	\cap				\cap
	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>				<i>R</i>

can be guaranteed to be new

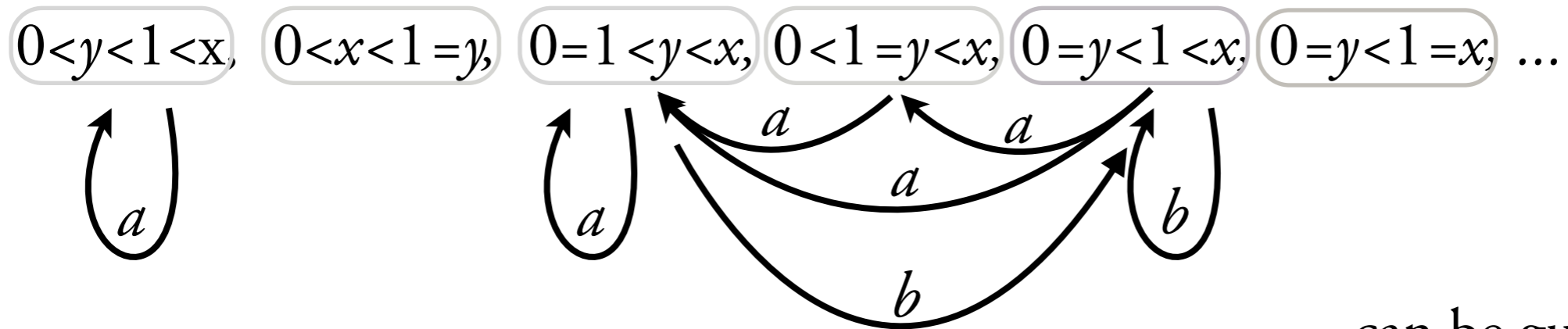
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



		<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	
<i>x</i>	5	5	5	5	6	6	6	6	100	
<i>y</i>	0	1	2	4	0	1	2	4	0	
	\cap	\cap	\cap	\cap	\cap	\cap			\cap	
	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>			<i>R</i>	

can be guaranteed to be new

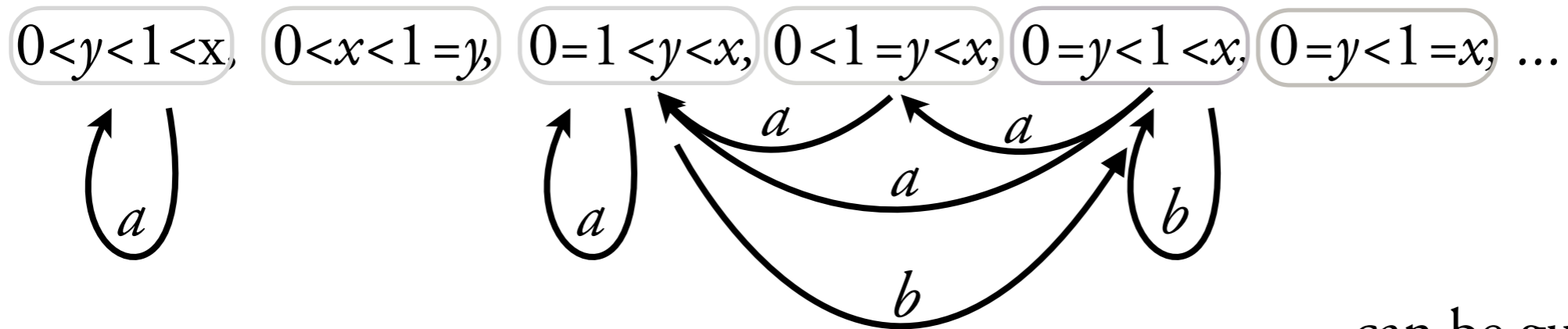
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



		<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	
<i>x</i>	5	5	5	5	6	6	6	6	100	
<i>y</i>	0	1	2	4	0	1	2	4	0	
	\cap	\cap	\cap	\cap	\cap	\cap	\cap		\cap	
	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>		<i>R</i>	

can be guaranteed to be new

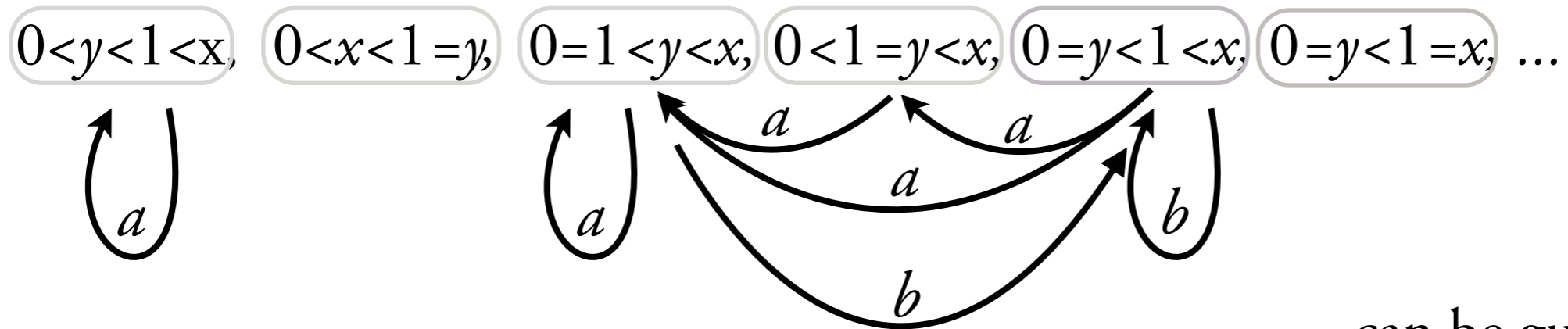
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



		<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	
<i>x</i>	5	5	5	5	6	6	6	6	100	
<i>y</i>	0	1	2	4	0	1	2	4	0	
	\cap	\cap	\cap	\cap	\cap	\cap	\cap	\cap	\cap	
	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	

can be guaranteed to be new

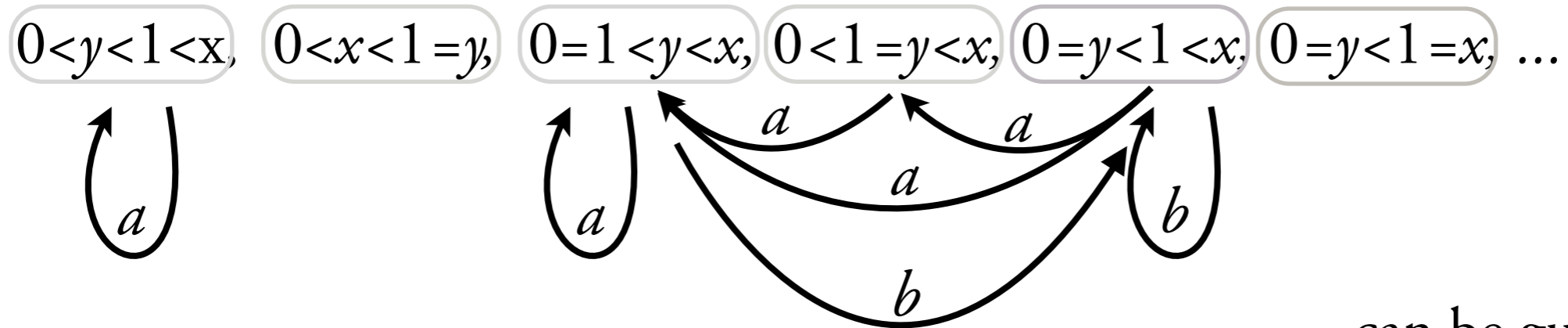
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	
<i>x</i>	5	5	5	5	6	6	6	6	100
<i>y</i>	0	1	2	4	0	1	2	4	0
	\cap	\cap	\cap	\cap	\cap	\cap	\cap	\cap	\cap
	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>

can be guaranteed to be new

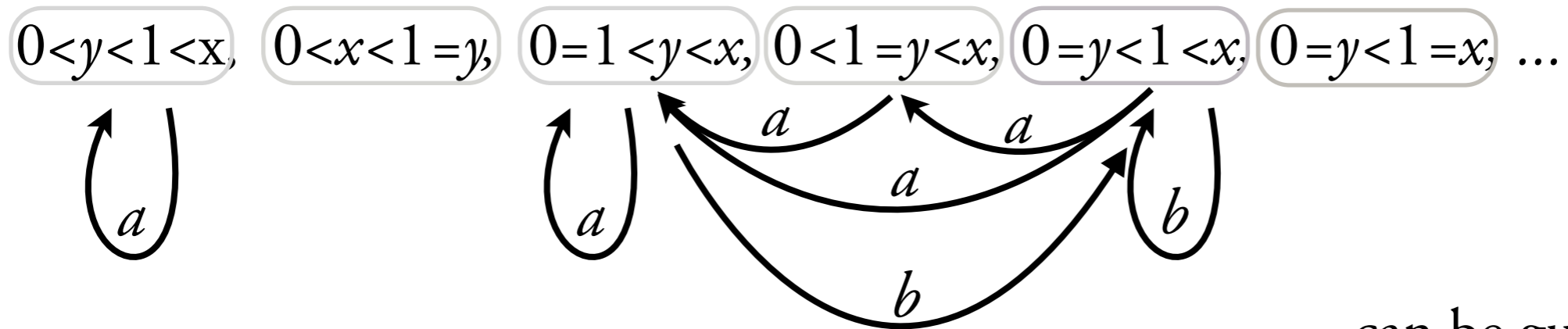
is forced to be zero

$$\tau_I: (x > 0) \wedge (y = 0)$$

$$\delta_a: (x' = x) \wedge (y < y' < x) \wedge R(x, y)$$

$$\delta_b: (y' = 0) \wedge \neg R(x, y)$$

$$\tau_F: (y = 1)$$



		<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>b</i>	
<i>x</i>	5	5	5	5	6	6	6	6	100	
<i>y</i>	0	1	2	4	0	1	2	4	0	
	\cap	\cap	\cap	\cap	\cap	\cap	\cap	\cap	\cap	
	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	<i>R</i>	

can be guaranteed to be new

is forced to be zero

En elaboration of these ideas solves the problem for automata with database, over infinite words:

En elaboration of these ideas solves the problem for automata with database, over infinite words:

Theorem. For all \mathcal{D} , \mathcal{D} -automata over infinite words accept ω B-regular languages.

En elaboration of these ideas solves the problem for automata with database, over infinite words:

Theorem. For all \mathcal{D} , \mathcal{D} -automata over infinite words accept ω B-regular languages.

Theorem. Emptiness of \mathcal{D} -automata is decidable* in PSPACE

*For any reasonable linearly ordered structure \mathcal{D}

En elaboration of these ideas solves the problem for automata with database, over infinite words:

Theorem. For all \mathcal{D} , \mathcal{D} -automata over infinite words accept ω B-regular languages.

Theorem. Emptiness of \mathcal{D} -automata is decidable* in PSPACE

*For any reasonable linearly ordered structure \mathcal{D}

Corollary. Deciding LTL+(data tests) properties of \mathcal{D} -automata is PSPACE-complete.

Thank you!