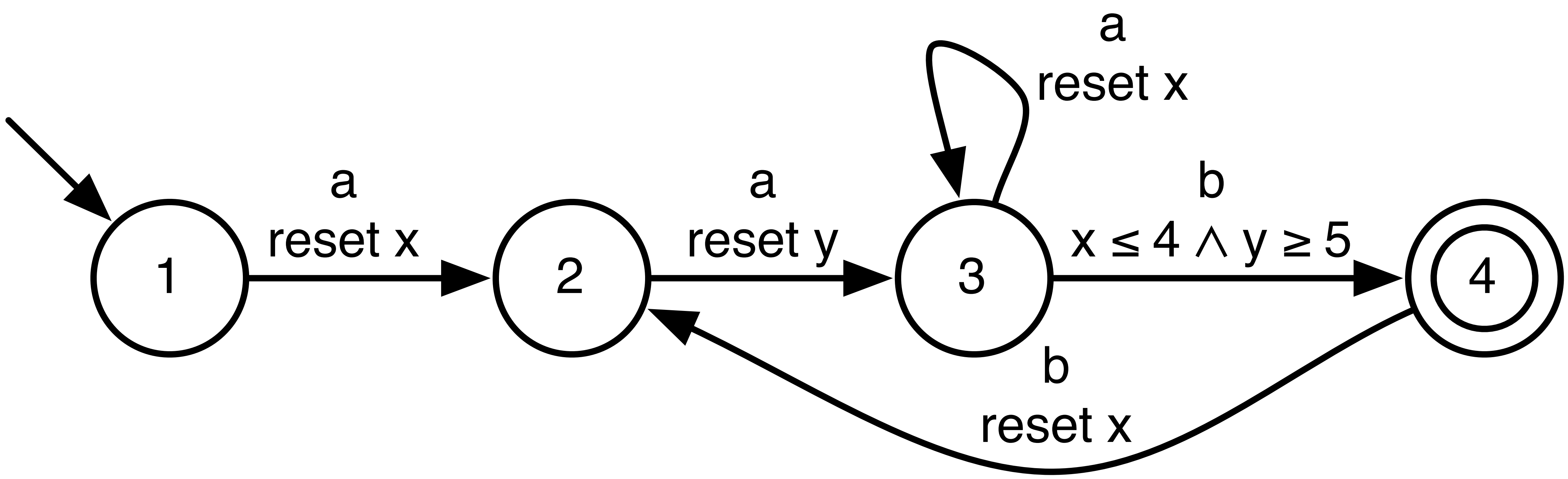
Polynomial distinguishability of timed automata

Sicco Verwer, Mathijs de Weerdt, Cees Witteveen

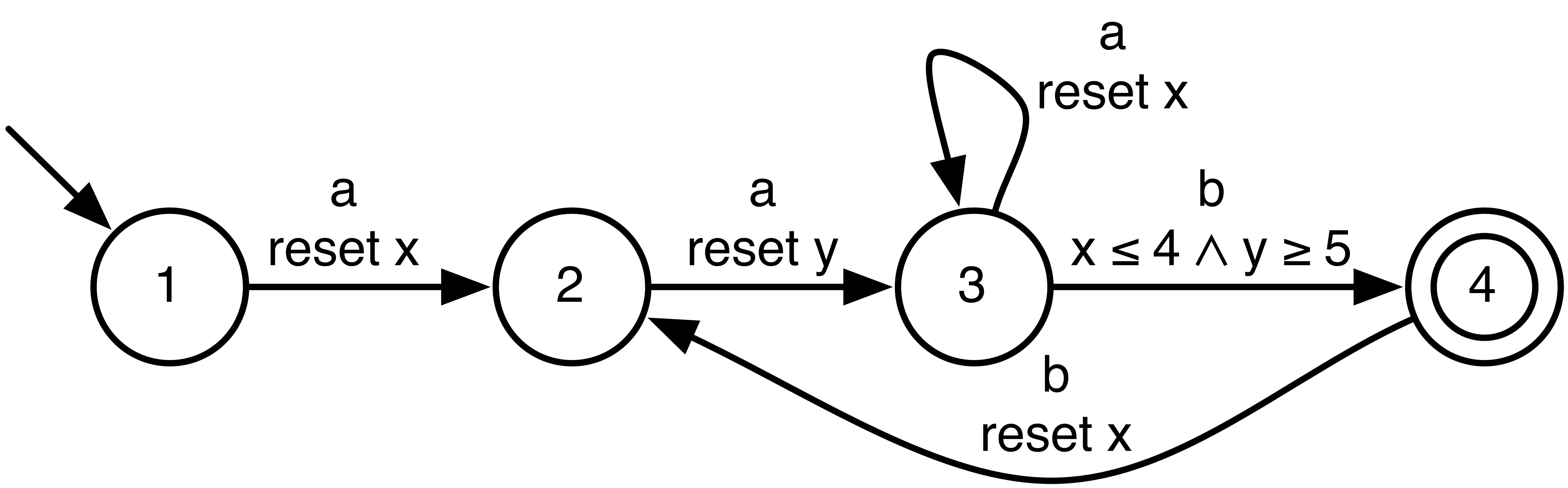
Overview

- Deterministic timed automata (DTAs)
- Polynomial distinguishability
- DTAs are not polynomially distinguishable
- Neither are DTAs with only two clocks (2-DTAs)
- But DTAs with a single clock (1-DTAs) are
- Conclusions and future work

DTAs



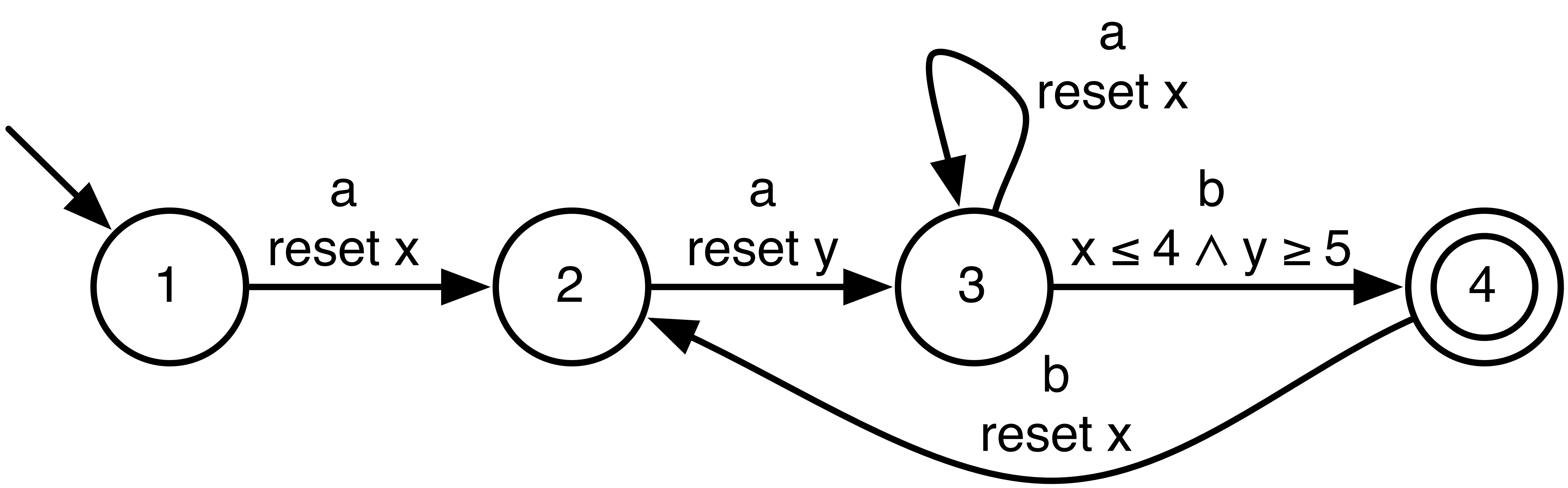
DTAs



accepts: (a, 1)(a, 2)(a, 3)(b, 4) rejects: (a, 1)(a, 2)(a, 1)(b, 2)

TU Delft

DTAs



rejects: (a, t)(a, t')(b, t'') for any t, t', t''
because x is reset before y in such a path

DTAs

- A deterministic timed automaton (DTA):
  - A deterministic finite state automaton (DFA)
  - A set of clocks X
  - A clock guard (constraint) g for every transition d
  - A set of clock resets R for every transition d
- Timed properties:
  - All clocks increase their values synchronously
  - A clock value can be reset to 0
  - A transition can fire if its clock guard is satisfied

Why learn DTAs?

- DTAs:
  - Use an explicit time representation (using numbers)
  - Are intuitive models for many real-time systems
  - Are used to model and verify reactive systems
- In practice it is often difficult to construct DTAs by hand, but data is easy to obtain:
  - We want to identify them from data

Why learn DTAs?

- Any timed system can also be represented using an implicit time representation, using DFAs or HMMs
  - Exponential blowup of the models and the data required for learning
  - Inefficient in the size of the timed data and the timed model
- We want to learn DTAs directly from timed data
  - Is it possible to do so efficiently?

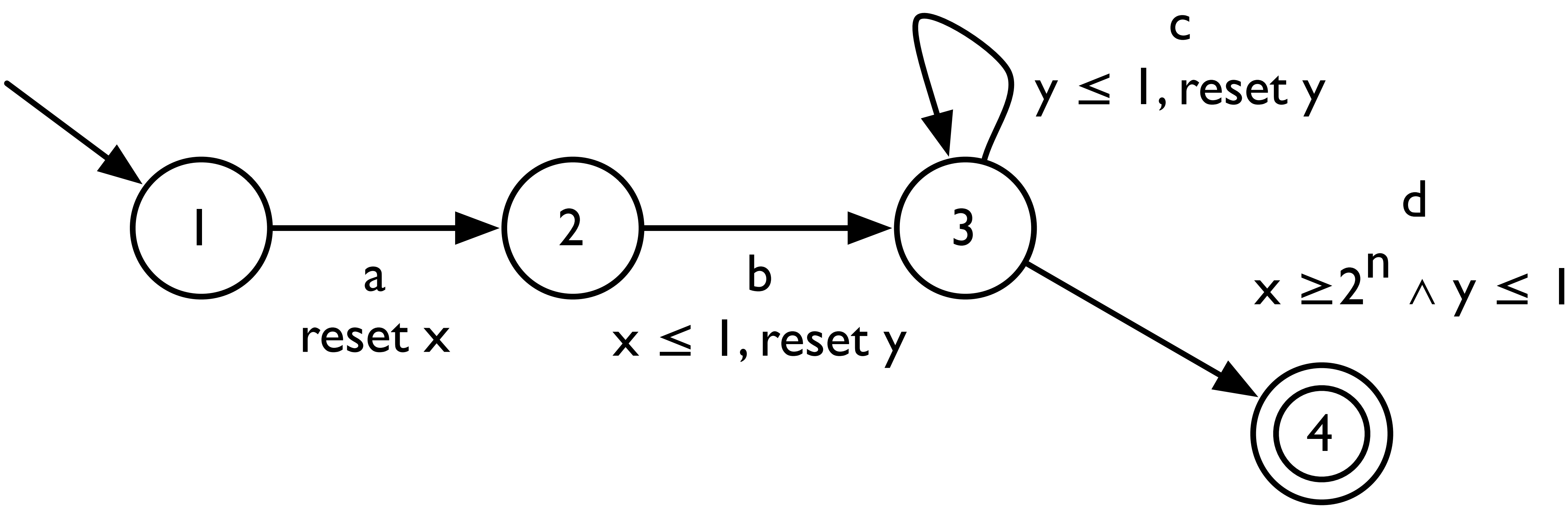
Polynomial Distinguishability

- A class of (timed) automata C is polynomially distinguishable if:
  - there exists a polynomial p() such that for any two (timed) automata A $\in$ C and A' $\in$ C, there exists a (timed) string s such that:
    - s $\in$ L(A) and s $\in$ L(A'), or vice versa, and
    - |s| is bounded by p(|A| + |A'|)
- If C is efficiently identifiable in the limit (from polynomial time and data), then C is polynomially distinguishable

Overview

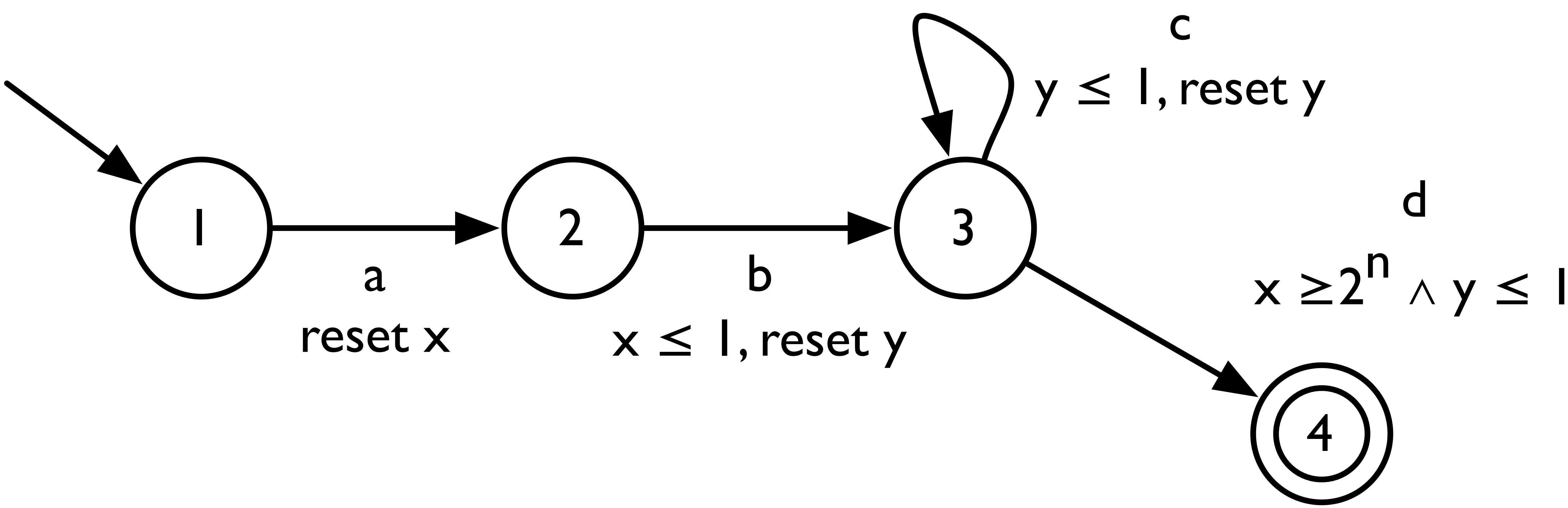
- Deterministic timed automata (DTAs)
- Polynomial distinguishability
- DTAs are not polynomially distinguishable
- Neither are DTAs with only two clocks (2-DTAs)
- But DTAs with a single clock (1-DTAs) are
- Conclusions and future work

DTAs are not pol. dist.



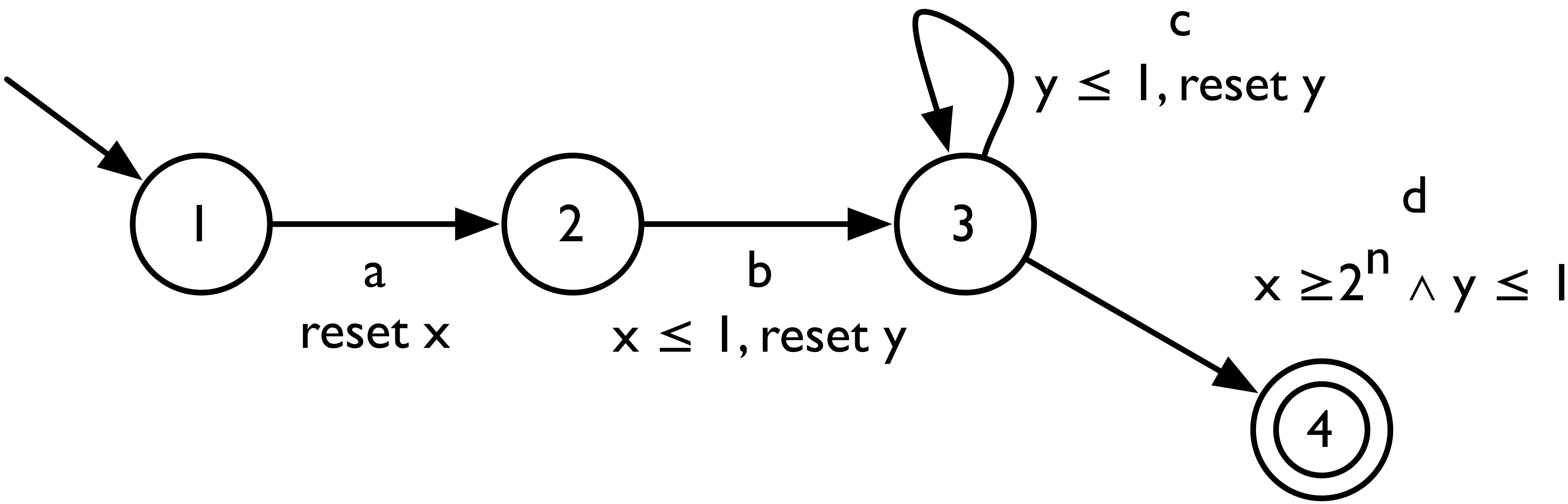
This DTA requires a timed string of exponential length in order to end in state 4

DTAs are not pol. dist.



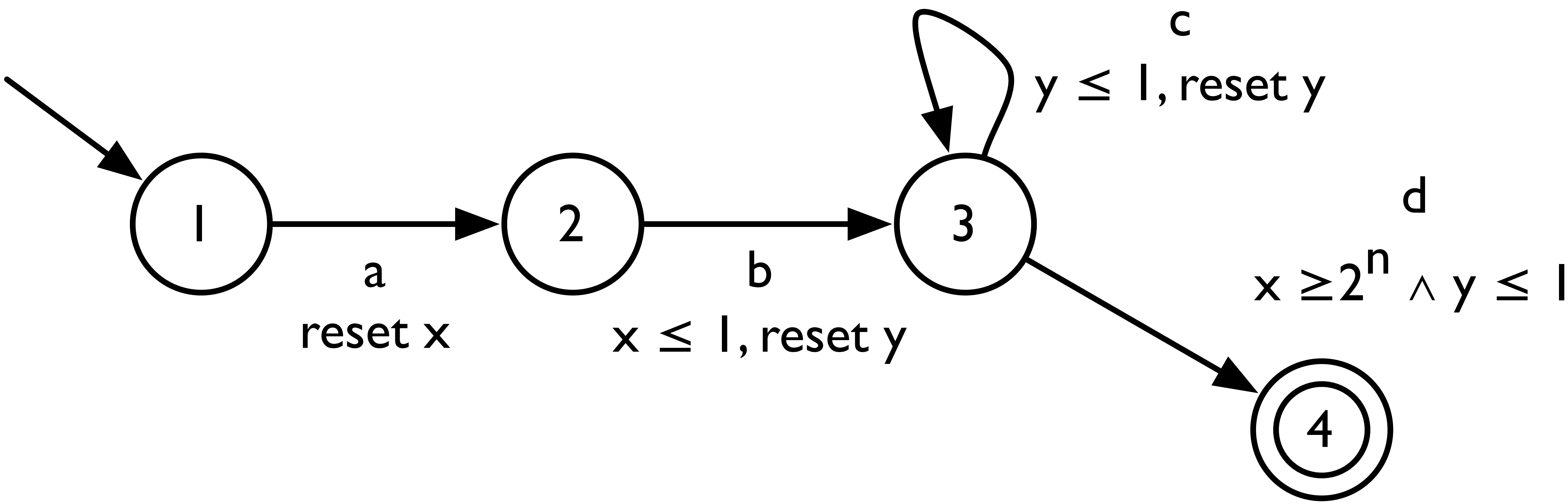
accepts only $(a, t)(b, t')(c, t_1)\ldots(c, t_m)(d, t'')$ where:
all $t', t'', t_1, \ldots, t_m \leq 1$ and $\sum t_1, \ldots, t_m \geq 2^n$,
hence, it has to hold that: $m \geq 2^n$

DTAs are not pol. dist.



We cannot polynomially bound the size of the shortest string that distinguishes these DTAs (for different n) from a DTA accepting the empty language

2-DTAs are not pol. dist.



These DTAs only require 2 clocks!

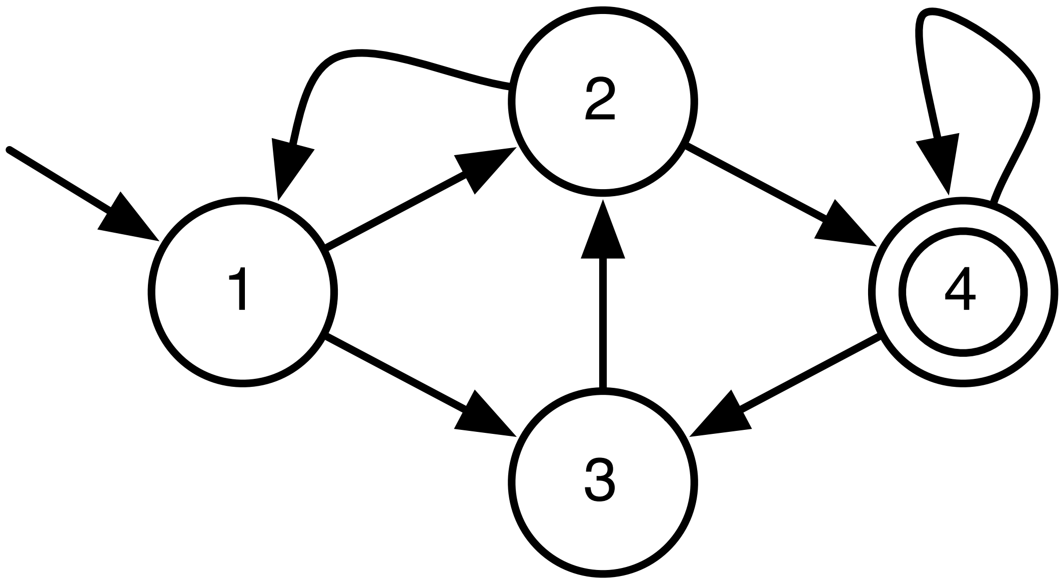
Overview

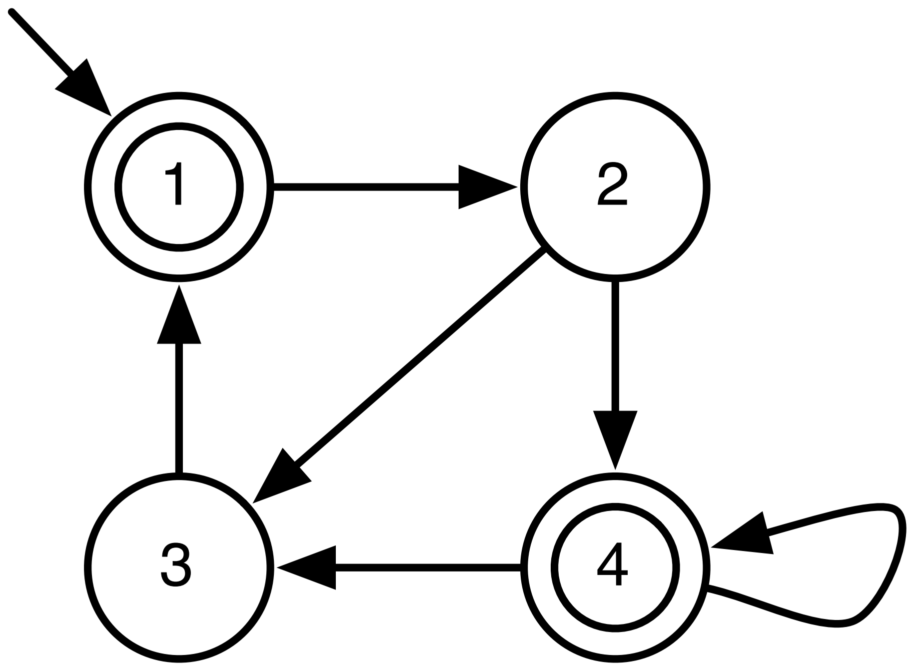
- Deterministic timed automata (DTAs)
- Polynomial distinguishability
- DTAs are not polynomially distinguishable
- Neither are DTAs with only two clocks (2-DTAs)
- But DTAs with a single clock (1-DTAs) are
- Conclusions and future work

1-DTAs

- An 1-DTA is a DTA with one clock x
- The DTAs we used to prove the non-polynomial distinguishability of DTAs require at least two clocks
- Are 1-DTAs polynomially distinguishable?

Distinguishing two 1-DTAs

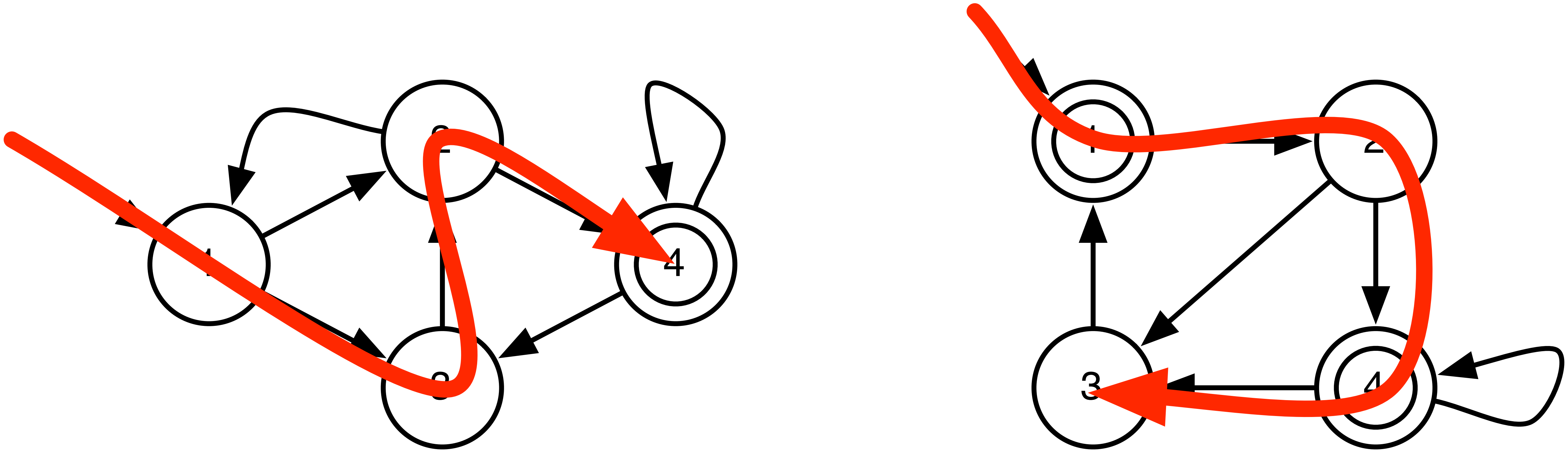




Does it hold that for any two DTAs, the size of a shortest timed string in the language of one and not in the language of the other is of polynomial length?

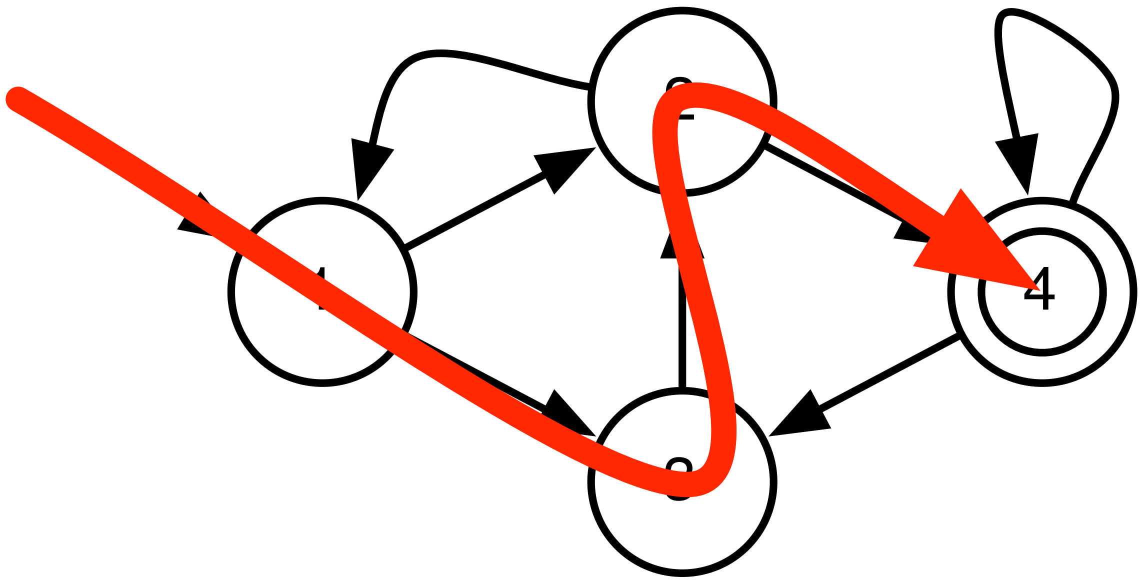
TU Delft

Distinguishing two 1-DTAs



Such a shortest timed string can follow different execution paths in the two 1-DTAs

Timed states

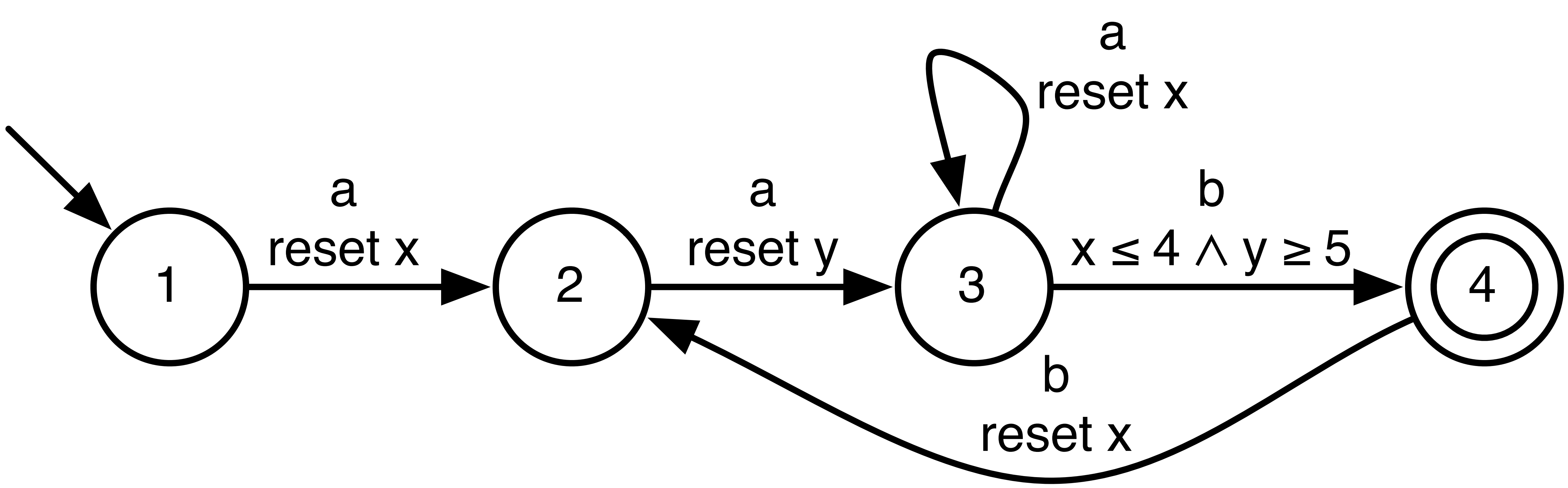


There has to exist a polynomial p() that bounds the length of a shortest timed string that ends in any reachable timed state (q,v)

Timed states

- A timed state (q, v) is a pair:
  - a state q from a TA
  - a valuation $v : X \Rightarrow \mathbb{N}$ maps clocks to time values
- A timed state (q, v) is reachable if there exists a timed string that ends in (q, v)

Timed states



(a, 1)(a, 2)(a, 3)(b, 4) ends in state 4 with
a valuation v such that v(x) = 4 and v(y) = 7

TU Delft

1-DTAs are pol. reachable

- Given a 1-DTA, let s be a shortest timed string that ends in some reachable timed state (q,v)
- It holds that:
  - a pair of prefixes s_i and s_j cannot end in the same timed state (q', v')
  - every s_i ends in (q',v') with v'(x) = 0 at most once
  - x is reset at most |Q| times in the path of s

1-DTAs are pol. reachable

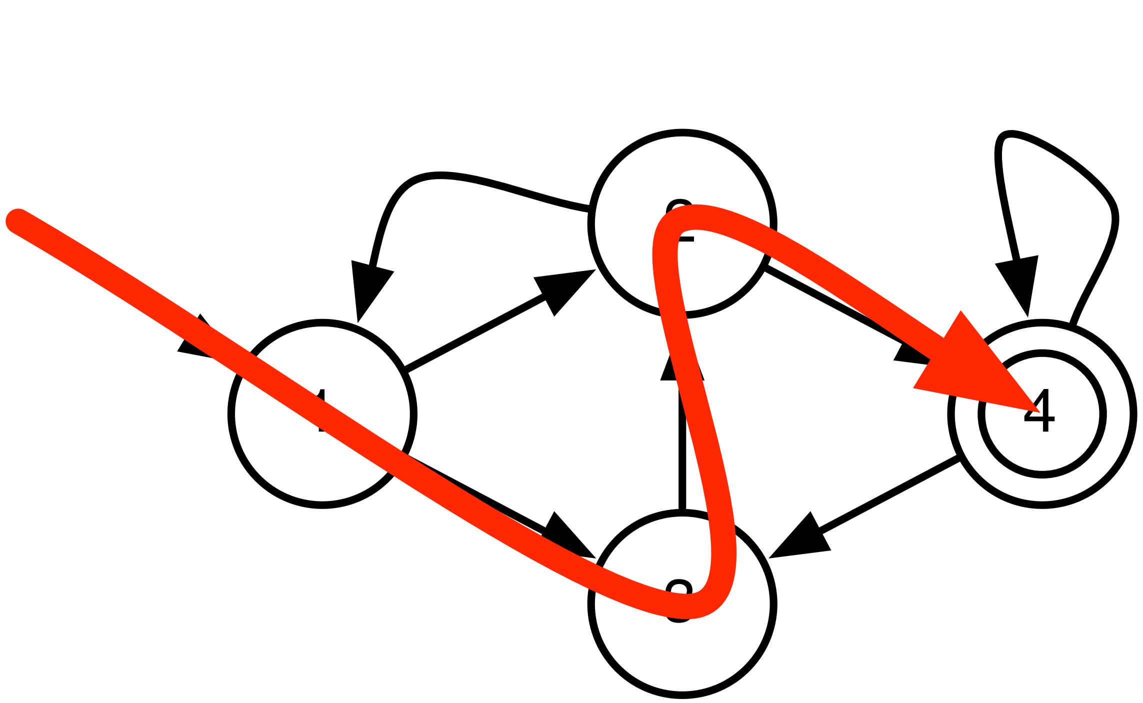
- When a timed string ends in (q', v'), then an 1-DTA can reach (q', v'') with $v''(x) \geq v'(x)$ by waiting some time in q'
- For a shortest string s that reaches (q, v):
  - if s_i ends in (q', v') and s_j ends in (q', v''), with $j > i$, then it has to hold that $v''(x) < v'(x)$
  - if s_i ends in (q', v') and s_j ends in (q', v''), then it has to hold that x is reset between index i and j
  - the amount of prefixes that end in (q', v') for any v' is bounded by the number of resets of x

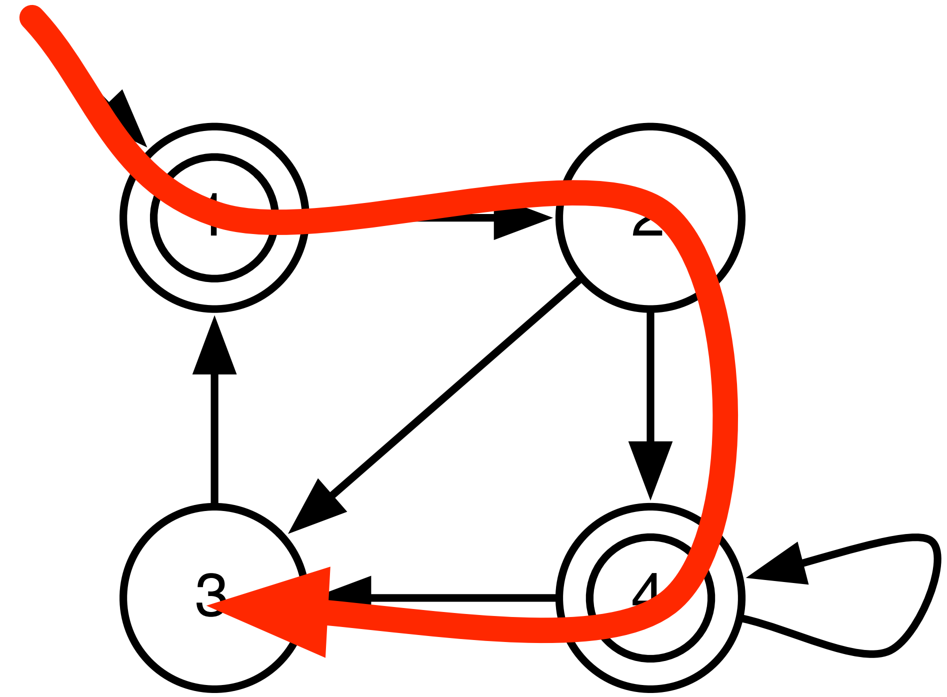
1-DTAs are pol. reachable

- A shortest string s that reaches (q, v) is of length bounded by:
  - $|Q| * $ the number of resets of x
  - $|Q| * |Q|$
  - a polynomial in the size of the 1-DTA
- Hence, 1-DTAs are polynomially reachable

TU Delft

1-DTAs are pol. dist.





A specific combination of timed states (q,v) in one and (q’,v’) in the other has to be reached

1-DTAs are pol. dist.

- Given two 1-DTAs, let s be a shortest timed string that reaches (q_1, v_1) in one and (q_2, v_2) in the other
- It holds that:
  - a pair of prefixes s_i and s_j cannot end in the same combination of timed state (q_1', v_1') and (q_2', v_2')
  - an 1-DTA can reach (q_n', v_n'') with $v_n''(x) \geq v_n'(x)$ by waiting some time in q_n'
  - x is reset between index i and j in one of the two 1-DTAs

TU Delft

1-DTAs are pol. dist.

- A shortest string s that reaches (q_1, v_1) and (q_2, v_2) is of length bounded by:
  - $|Q| * |Q'| *$ the number of resets of x
- In the paper, we use structural properties of 1-DTAs to polynomially bound the number of resets of x
- 1-DTAs are polynomially distinguishable

TU Delft

Overview

- Deterministic timed automata (DTAs)
- Polynomial distinguishability
- DTAs are not polynomially distinguishable
- Neither are DTAs with only two clocks (2-DTAs)
- But DTAs with a single clock (1-DTAs) are
- Conclusions and future work

Conclusions

- DTAs are an intuitive representation for real-time systems
- They are more compact (efficient) than DFA or HMM representations of the same systems
- Unfortunately, DTAs can in general not be identified efficiently since they are not polynomially distinguishable
- However, 1-DTAs are polynomially distinguishable

TU Delft

Future work

- Show that 1-DTAs are efficiently identifiable in the limit (soon to be submitted)
- Try to find multi-clock subclasses of DTAs that are polynomially distinguishable
- Determine whether a DTA identification algorithm could be used to identify 1-DTAs efficiently
- ...

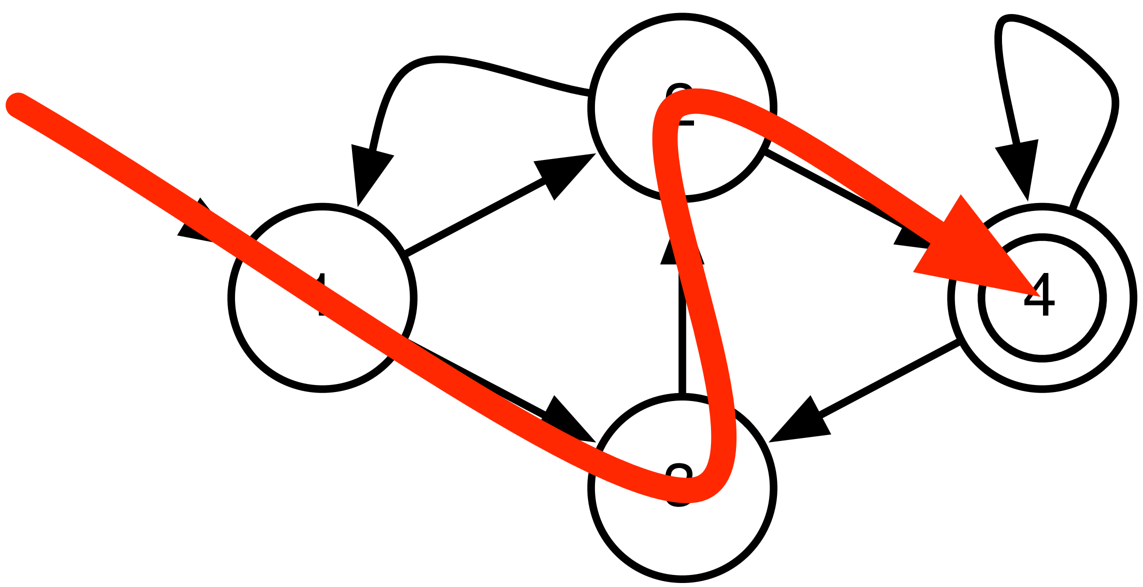
TU Delft

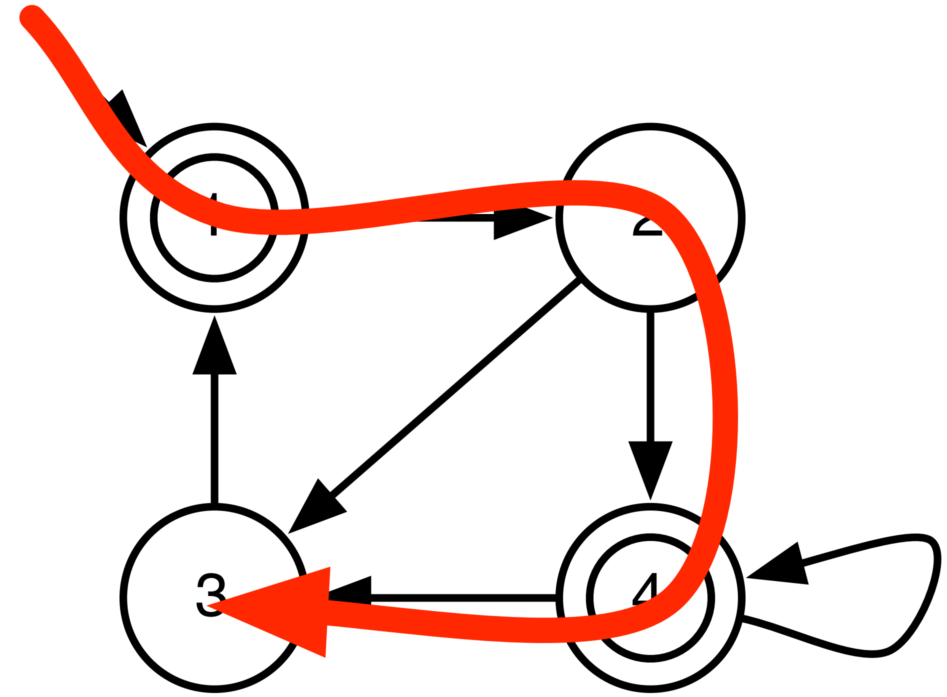
Questions

- ?

1-DTAs are pol. dist.

Suppose s visits in order

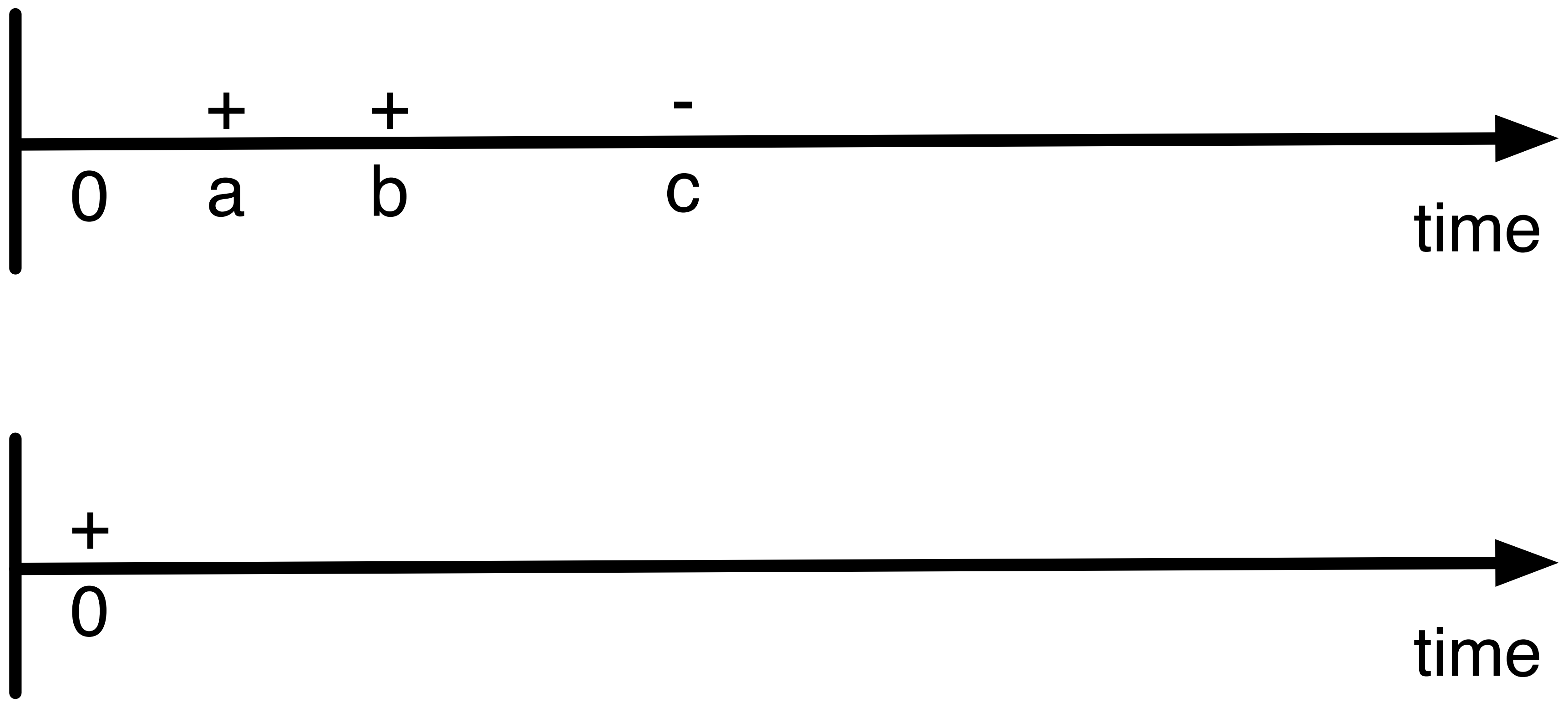




(q, v) with v(x) = 0
(q, v) with v(x) = 0
(q, v) with v(x) = 0

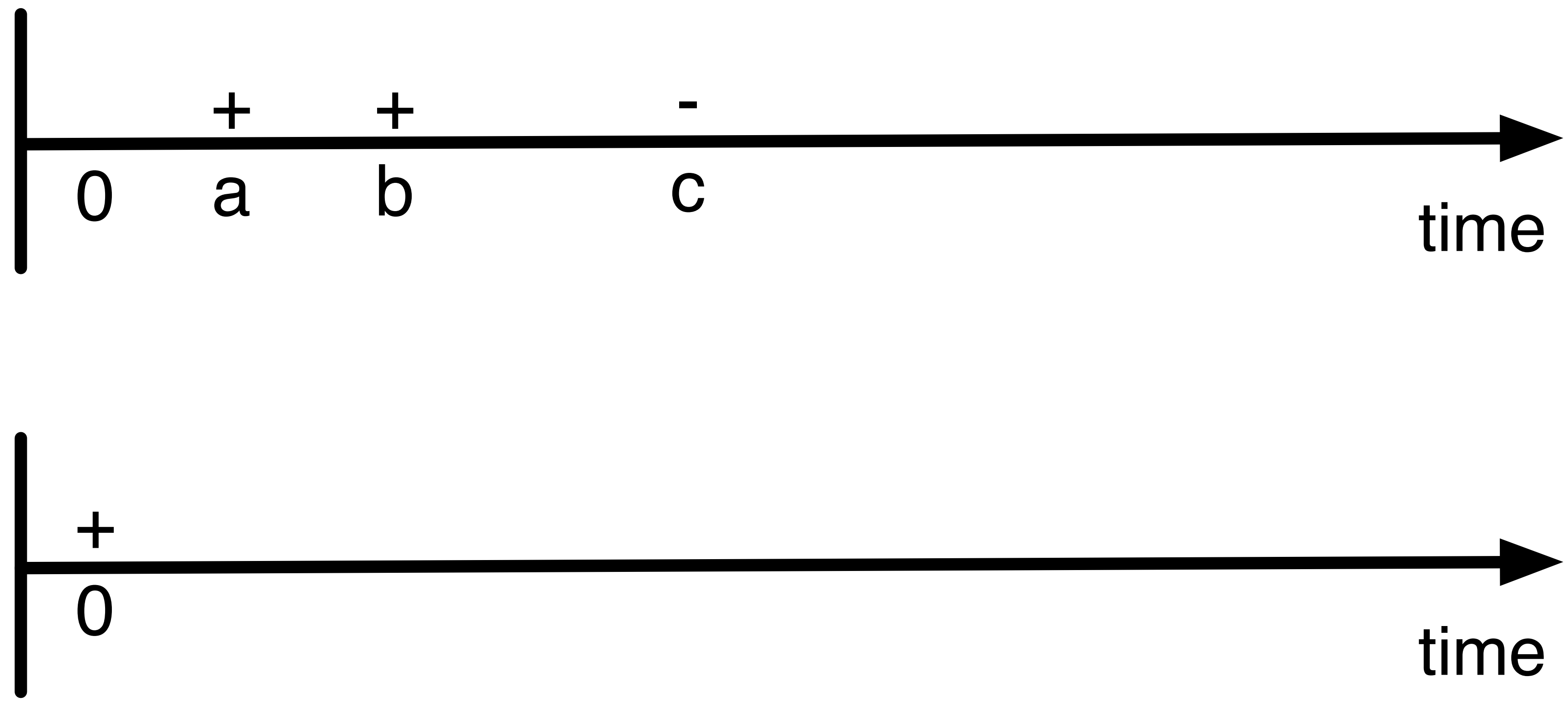
(q', v') with v'(x) = a
(q', v'') with v''(x) = b
(q', v''') with v'''(x) = c

1-DTAs are pol. dist.



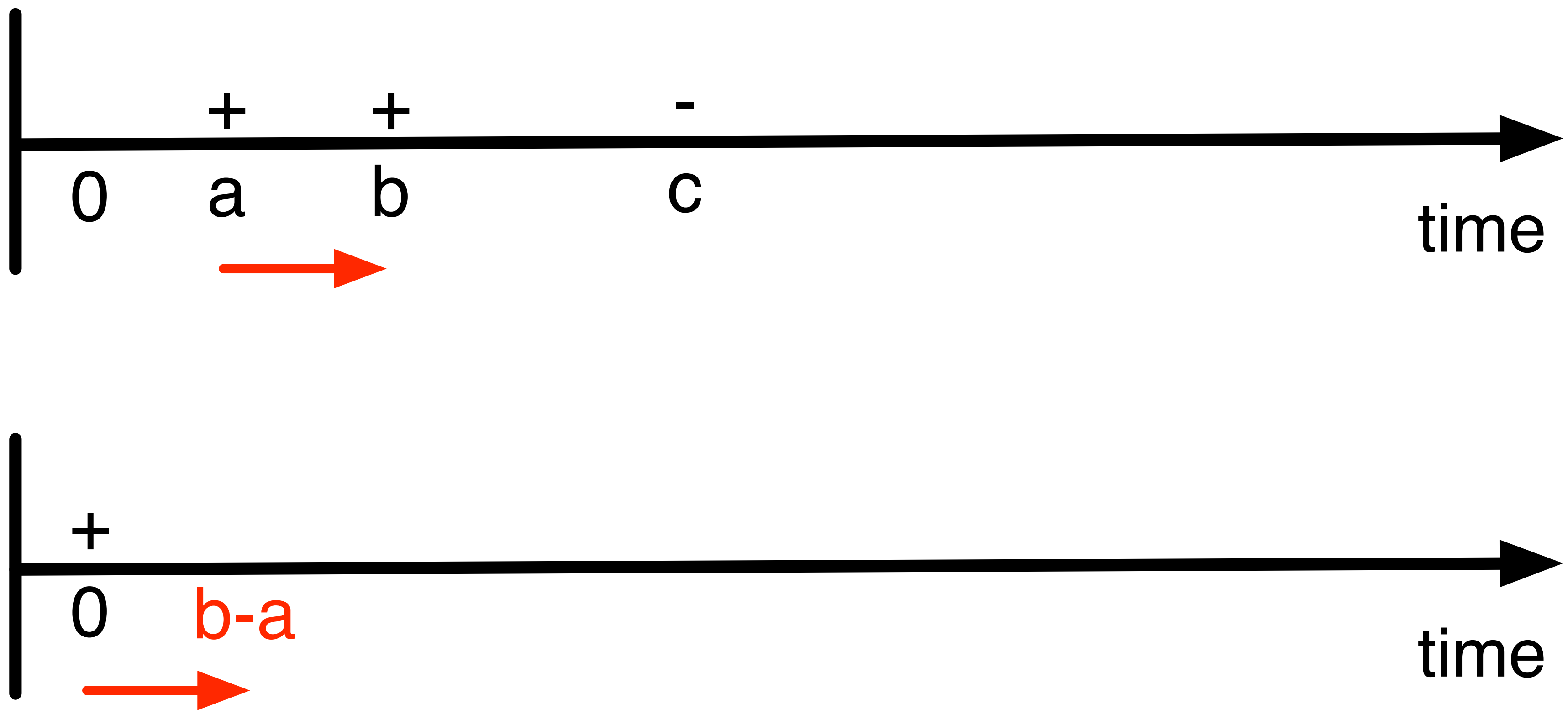
Whether following the last path leads to a final state is plotted along a time axis

1-DTAs are pol. dist.



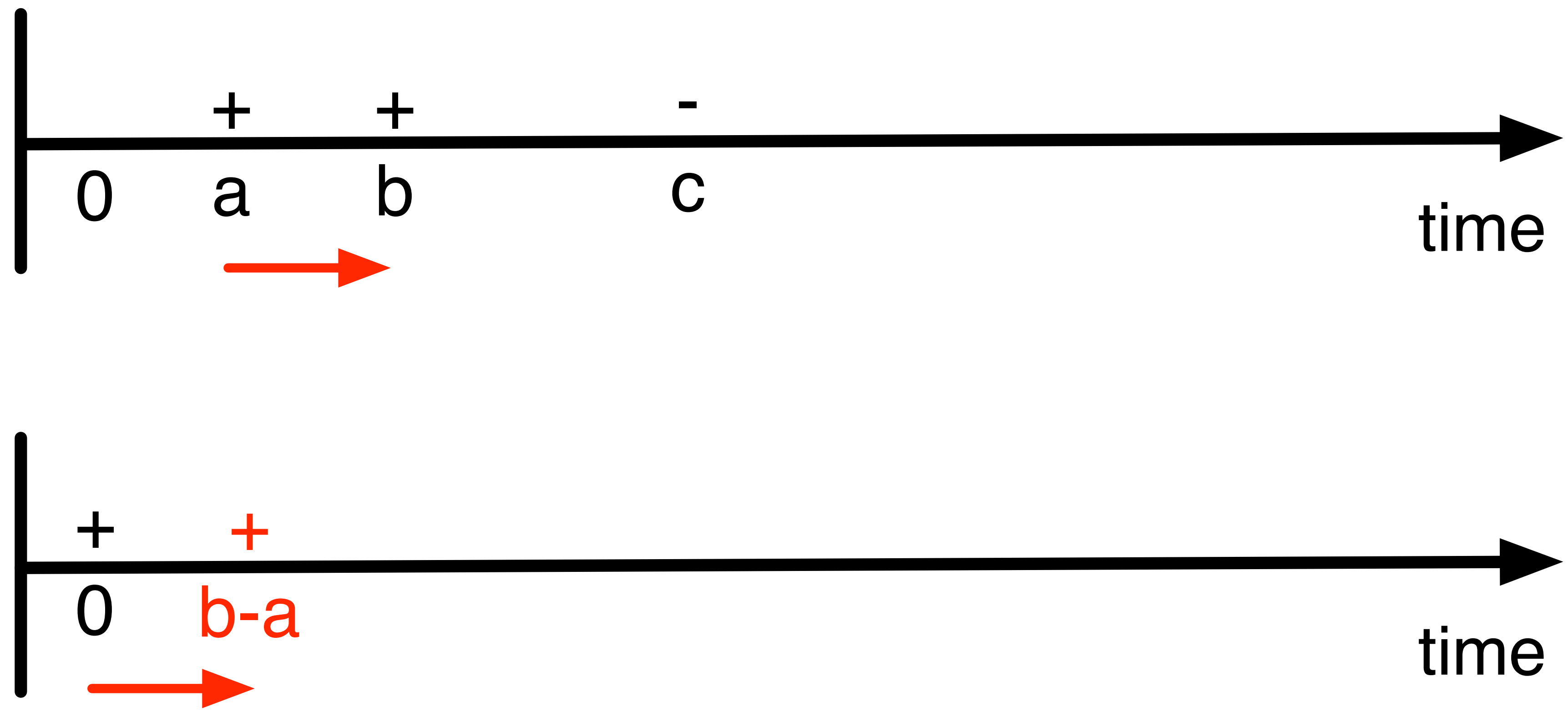
There exists no shorter distinguishing string
Following the final path earlier cannot distinguish one 1-DTA from the other

1-DTAs are pol. dist.



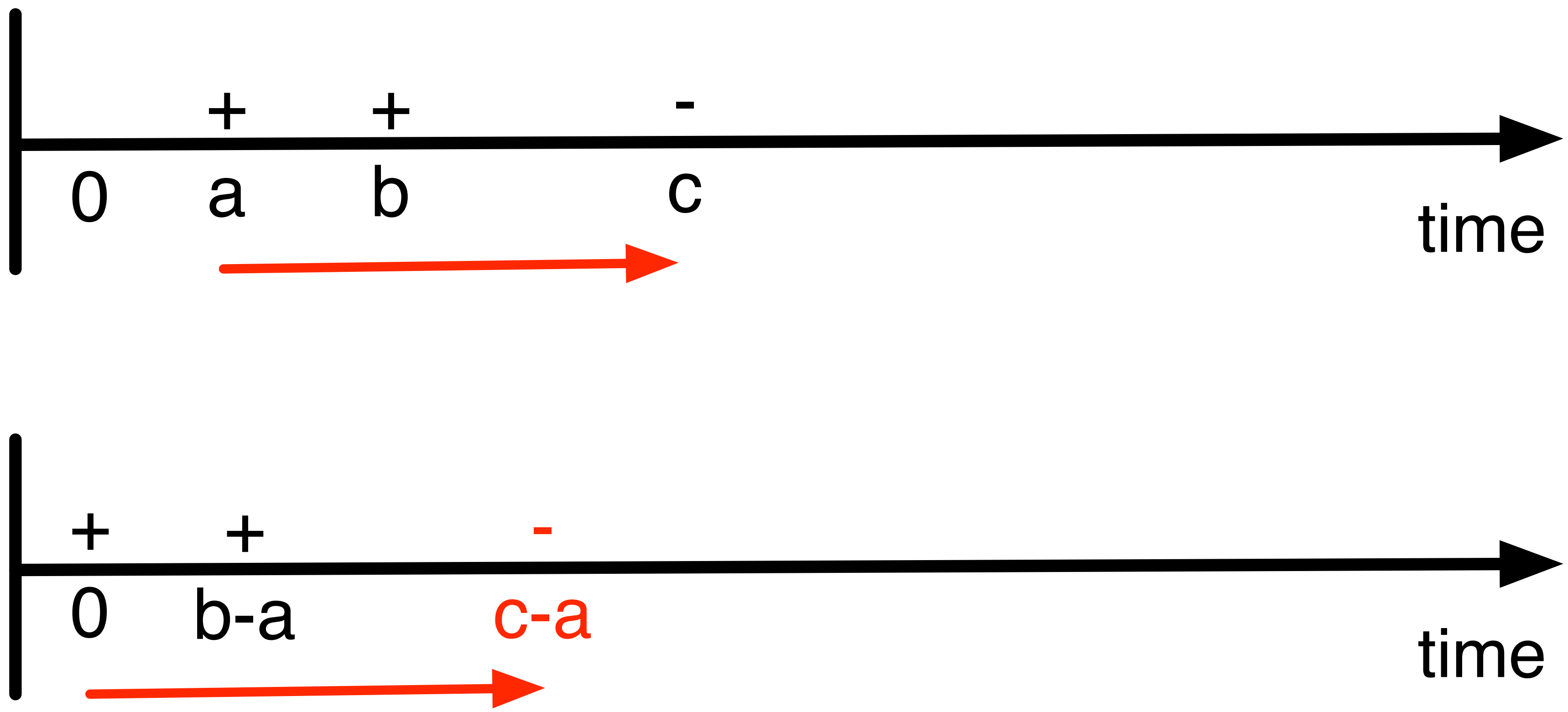
This also holds for first waiting and then following the final path

1-DTAs are pol. dist.



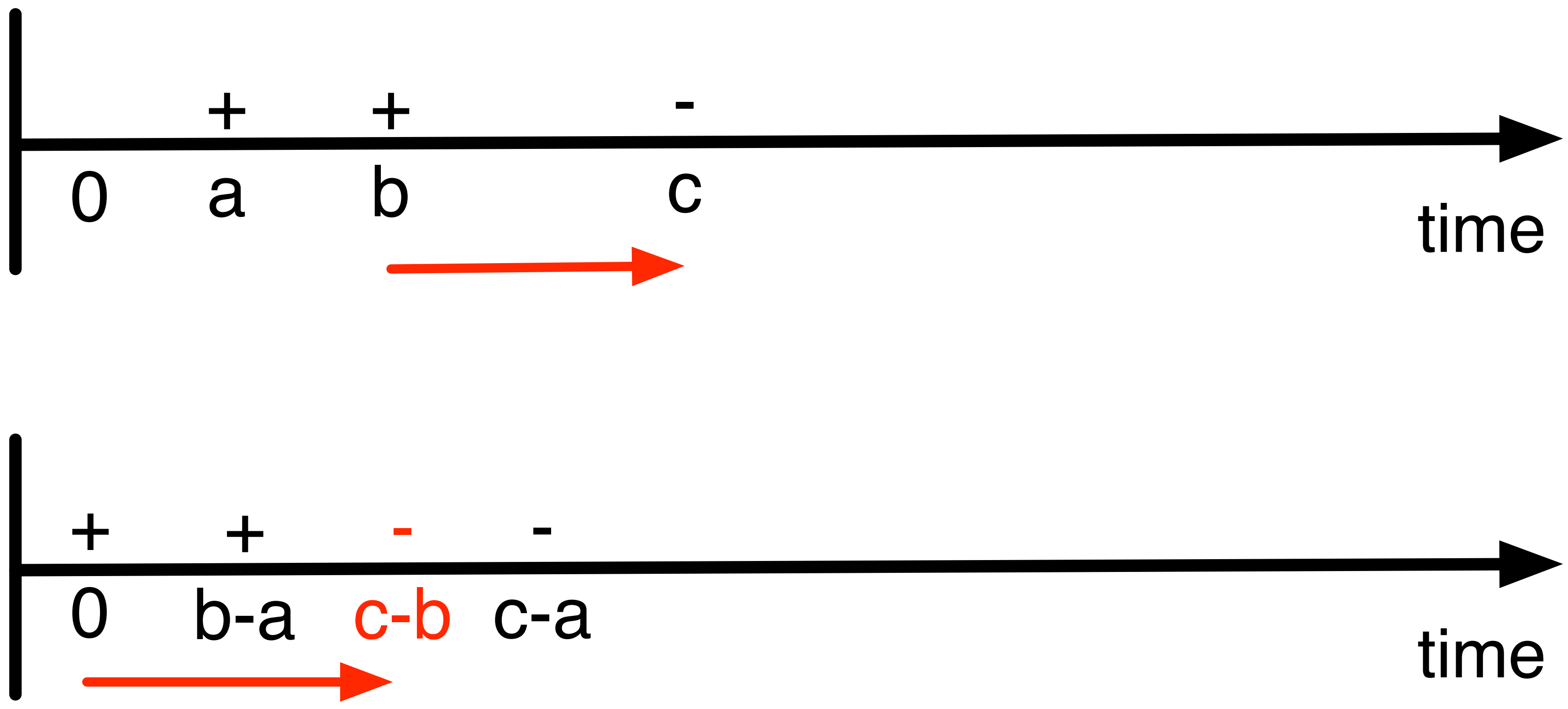
This also holds for first waiting and then following the final path

1-DTAs are pol. dist.



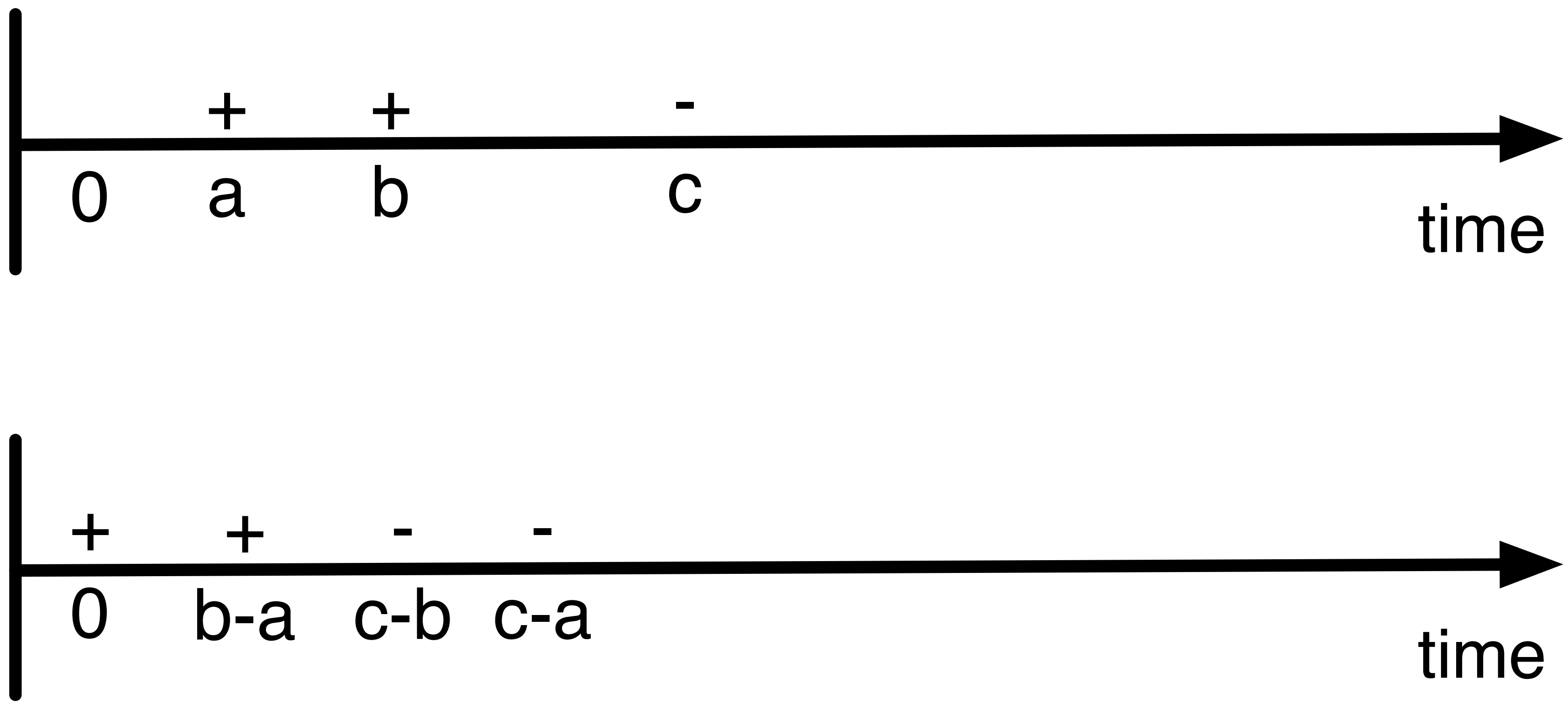
This also holds for first waiting and then following the final path

1-DTAs are pol. dist.



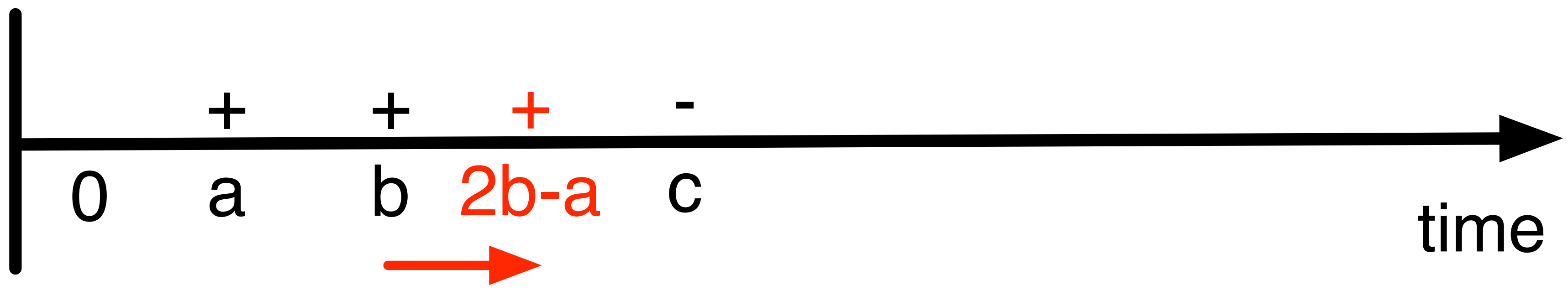
This also holds for first waiting and then following the final path

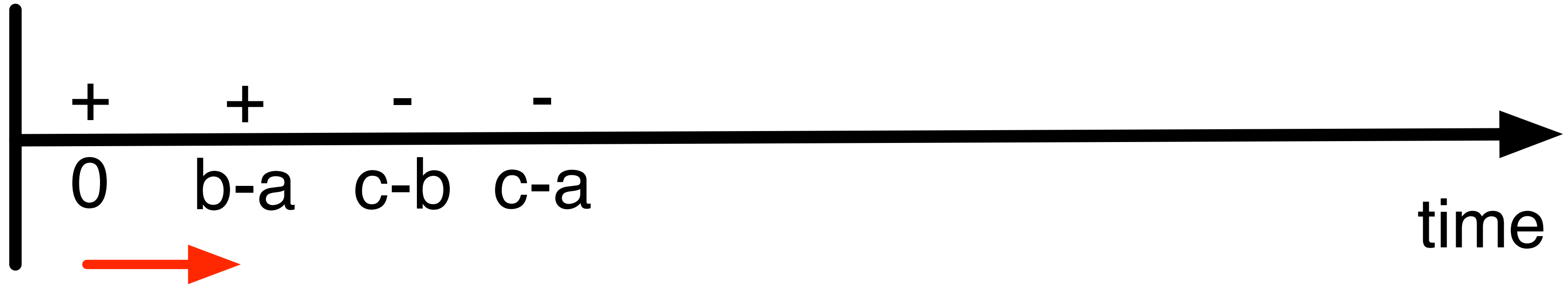
1-DTAs are pol. dist.



Waiting and following the final path leads to these values in the bottom 1-DTA

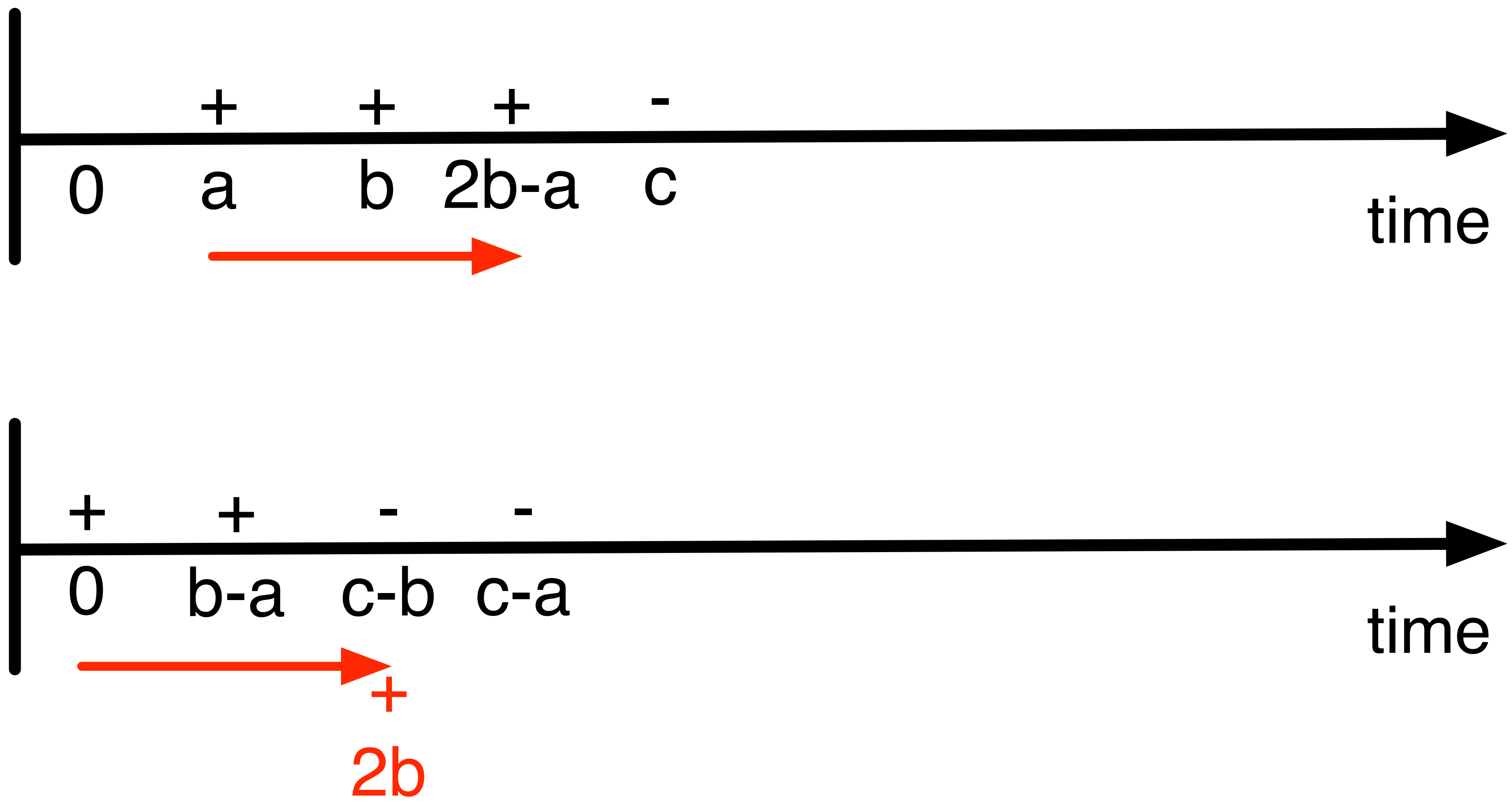
1-DTAs are pol. dist.





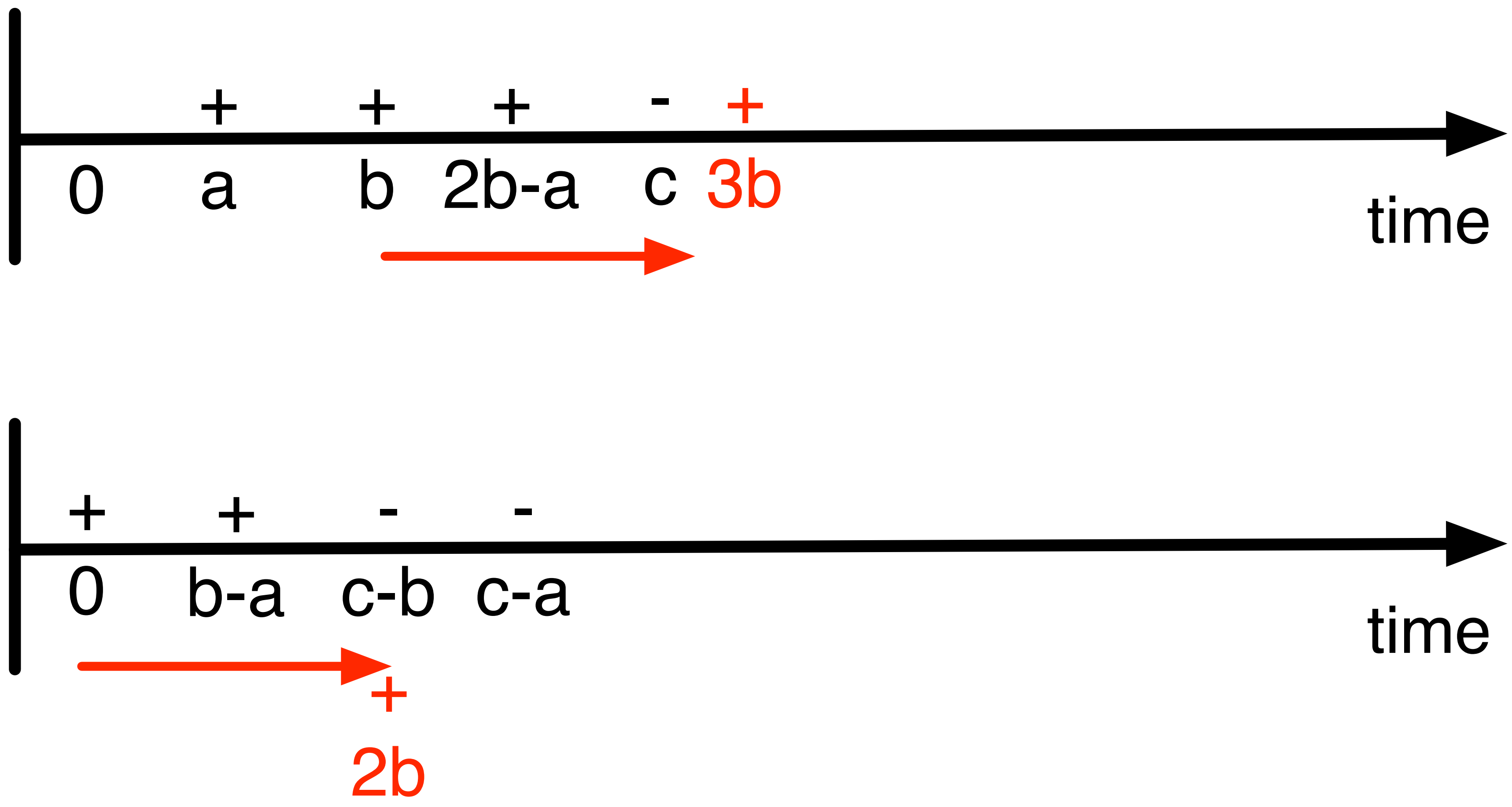
We can also wait in the top 1-DTA

1-DTAs are pol. dist.



This leads to new known values in the bottom 1-DTA

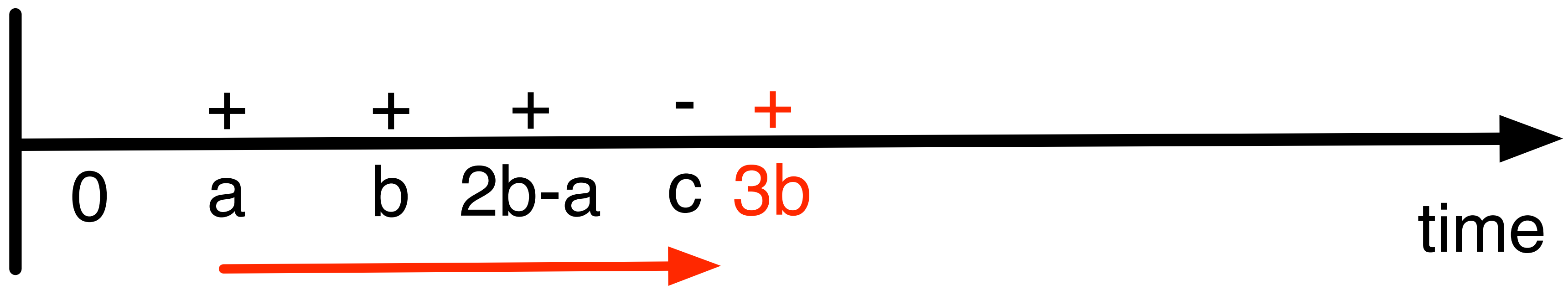
1-DTAs are pol. dist.

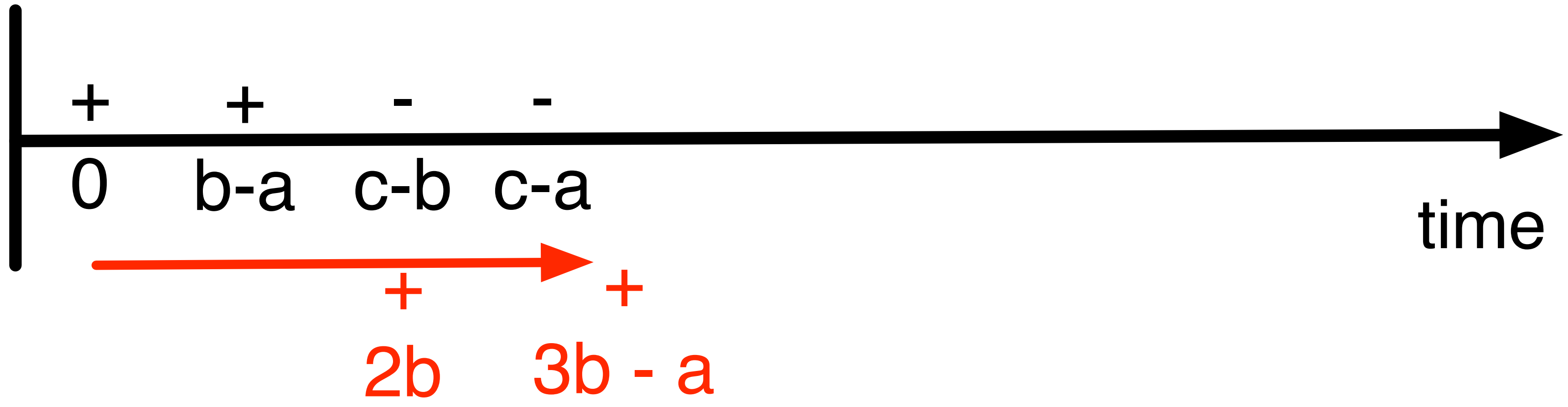


This can be continued infinitely

TU Delft

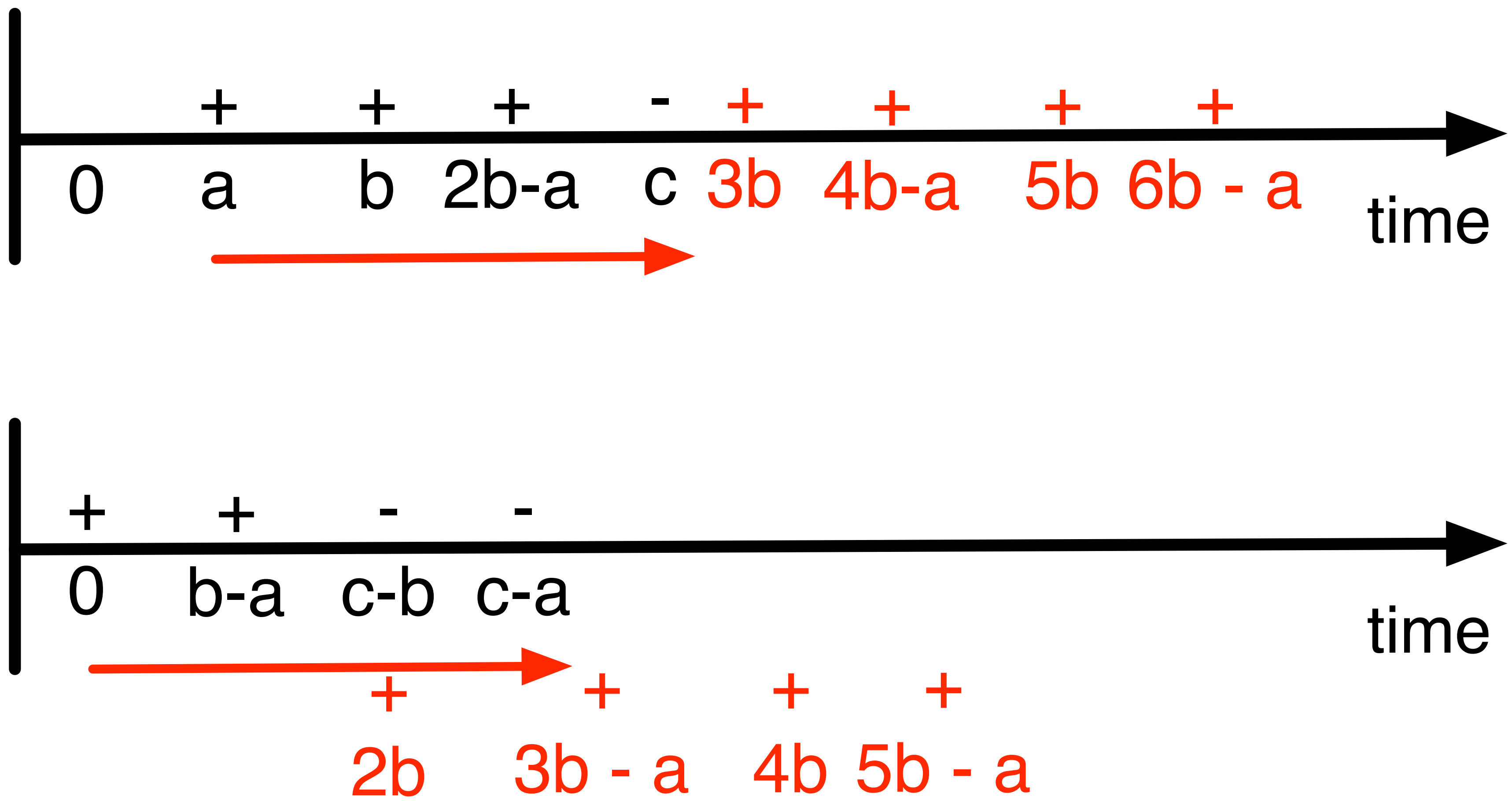
1-DTAs are pol. dist.





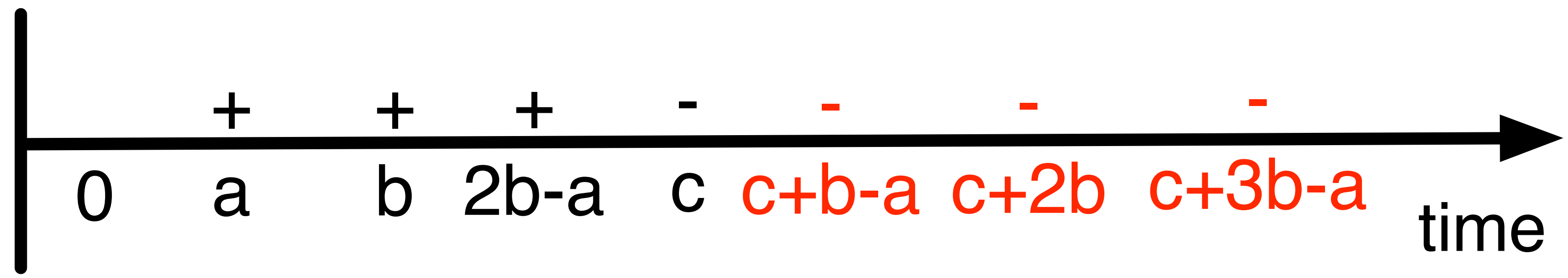
This can be continued infinitely

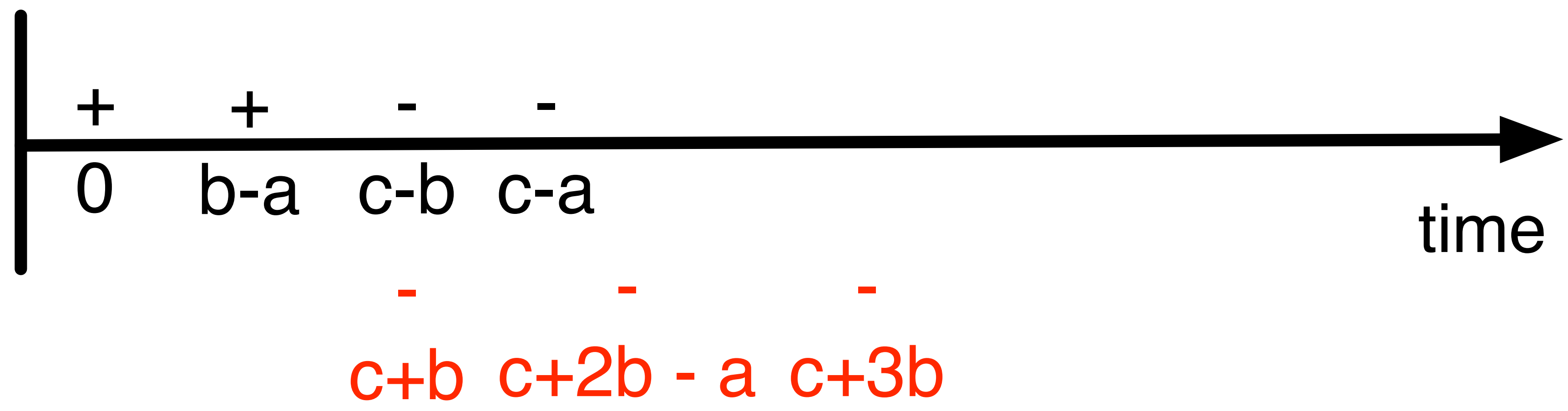
1-DTAs are pol. dist.



This can be continued infinitely

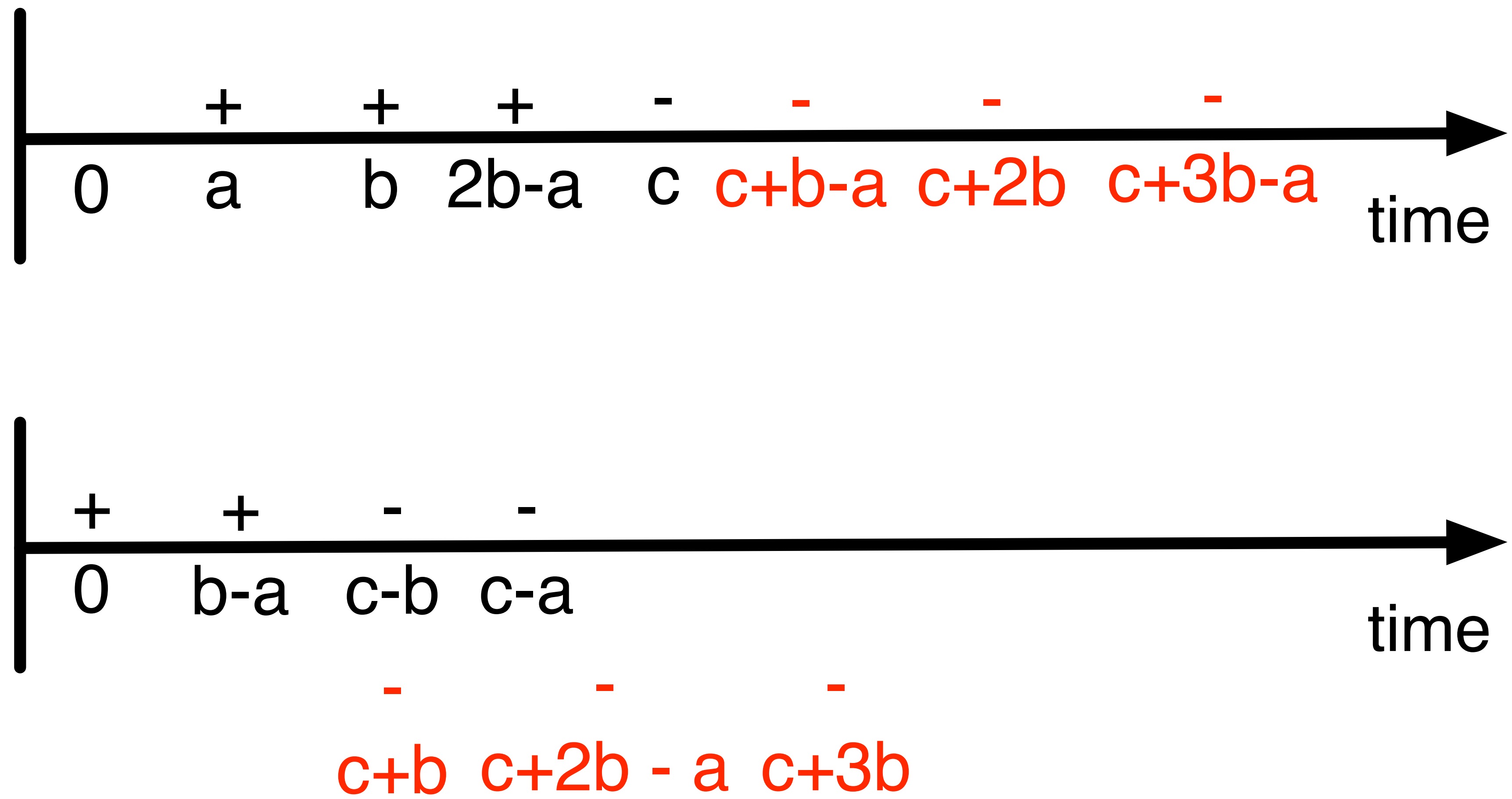
1-DTAs are pol. dist.





Also for negative values

1-DTAs are pol. dist.



Such an infinite change from positive to negative and vice versa cannot be modeled by an 1-DTA!