Statistics for state merging

Sicco Verwer

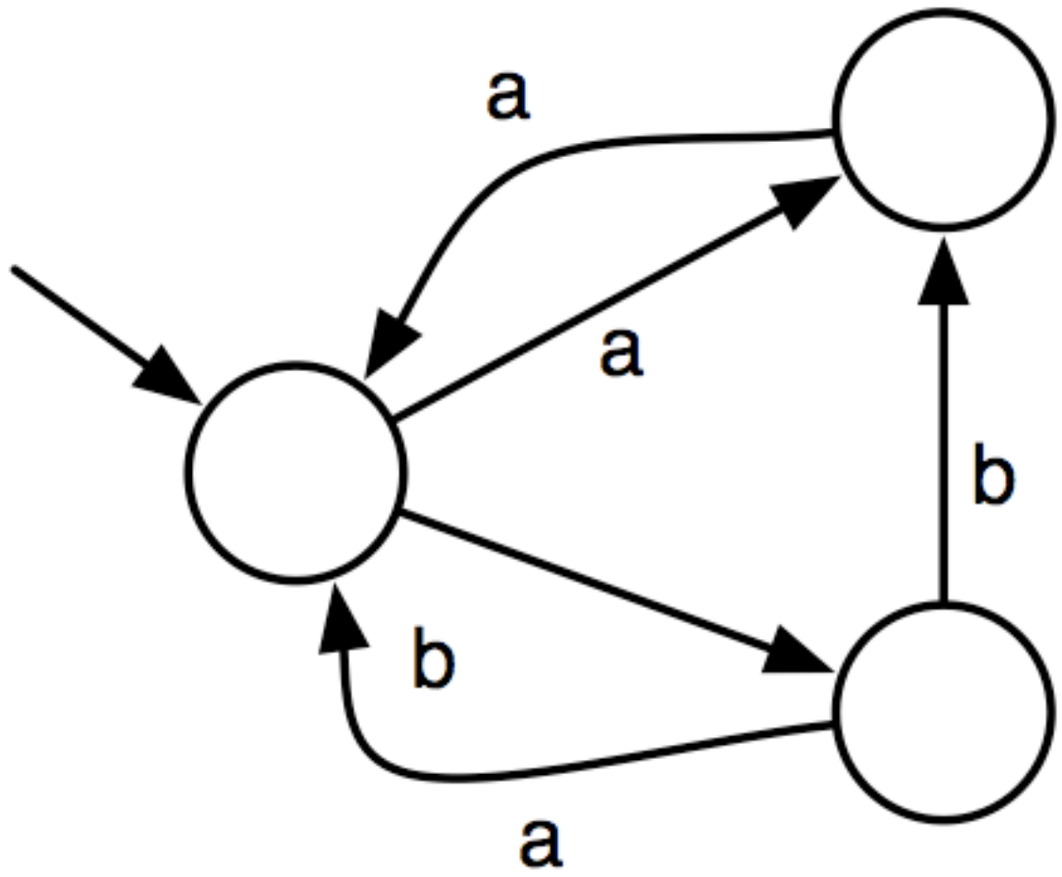
2009

Technische Universiteit
Eindhoven
University of Technology

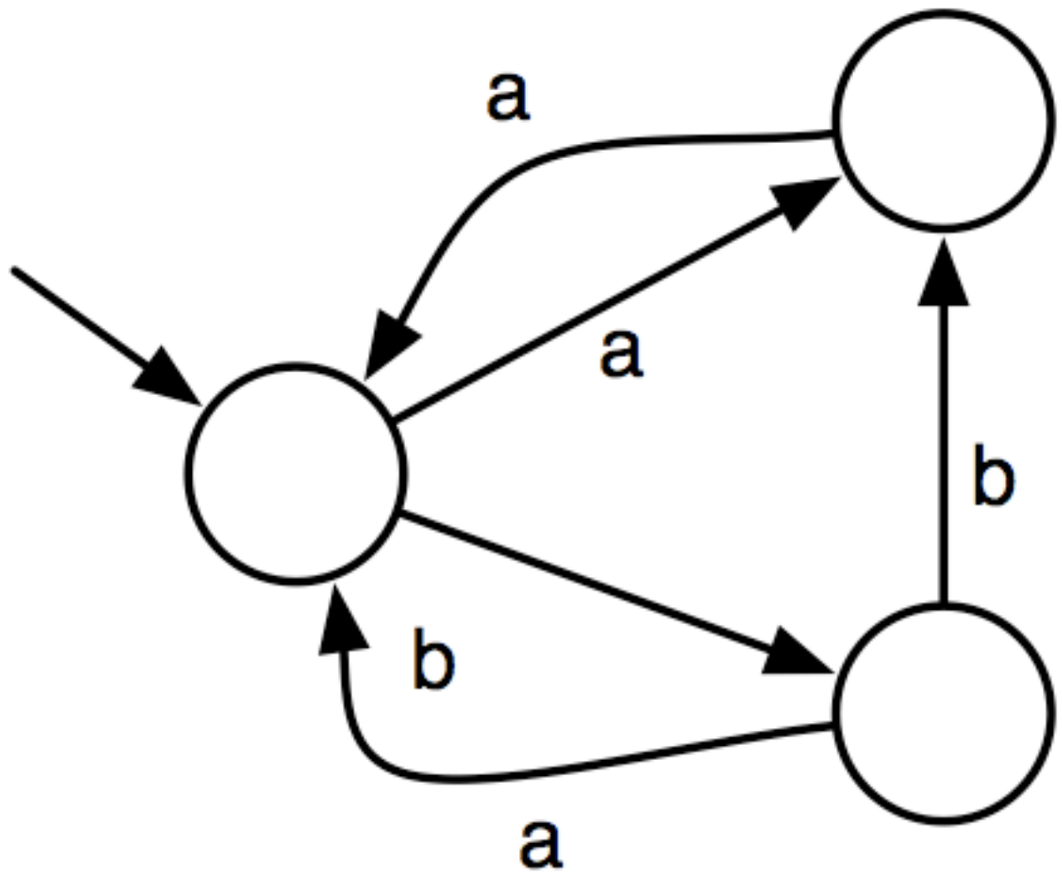
Overview

- **Automata**
- **State merging**
- **Real-time automata**
- **Learning real-time automata using statistics**
- **Results**

Automata



Automata and events



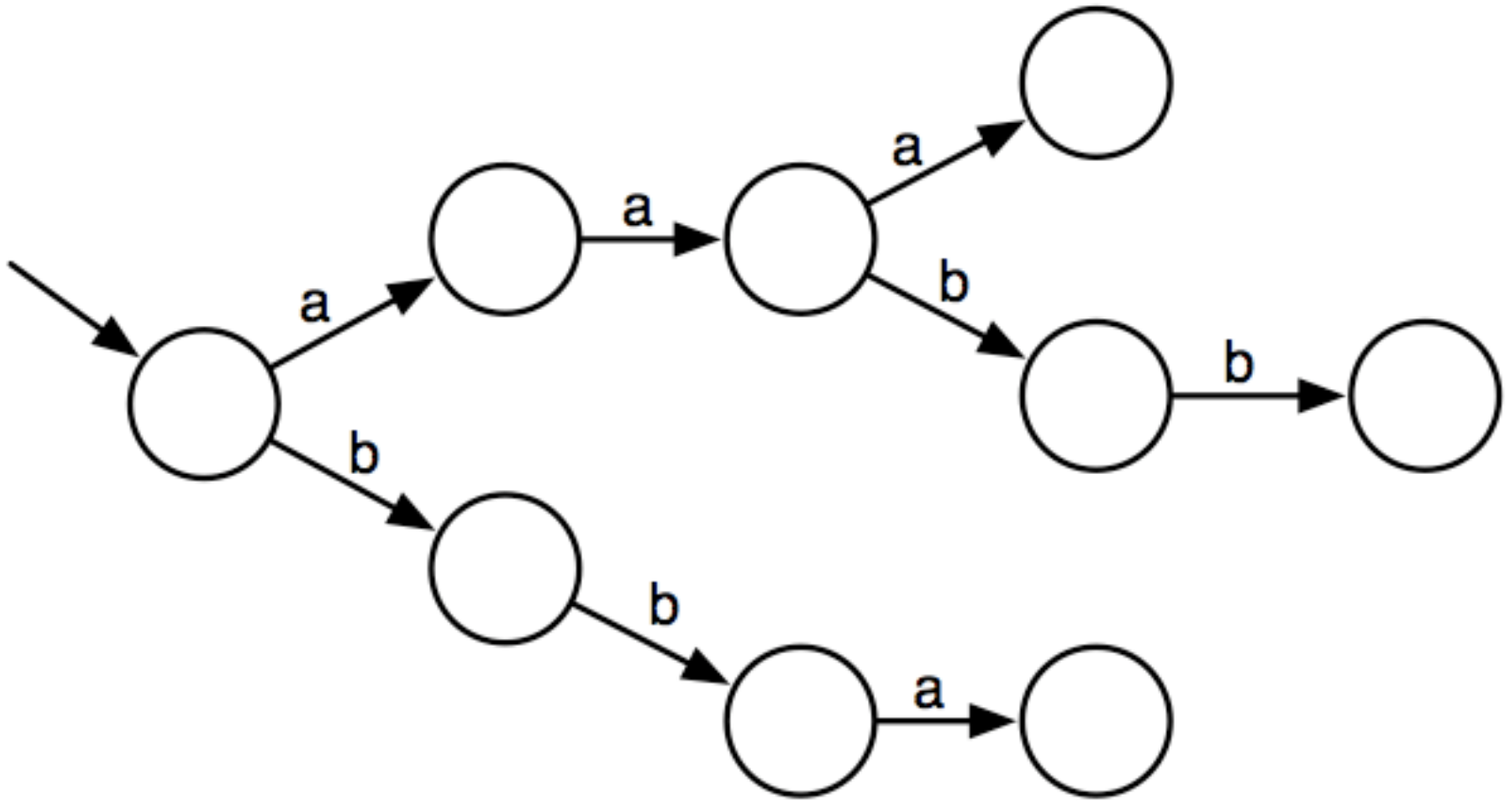
Produces strings: aa, aabb, aaaab, babba, bbab

Properties of DFAs

- Transitions are labeled (instead of states)
- States are Markovian
- Model exactly the regular languages

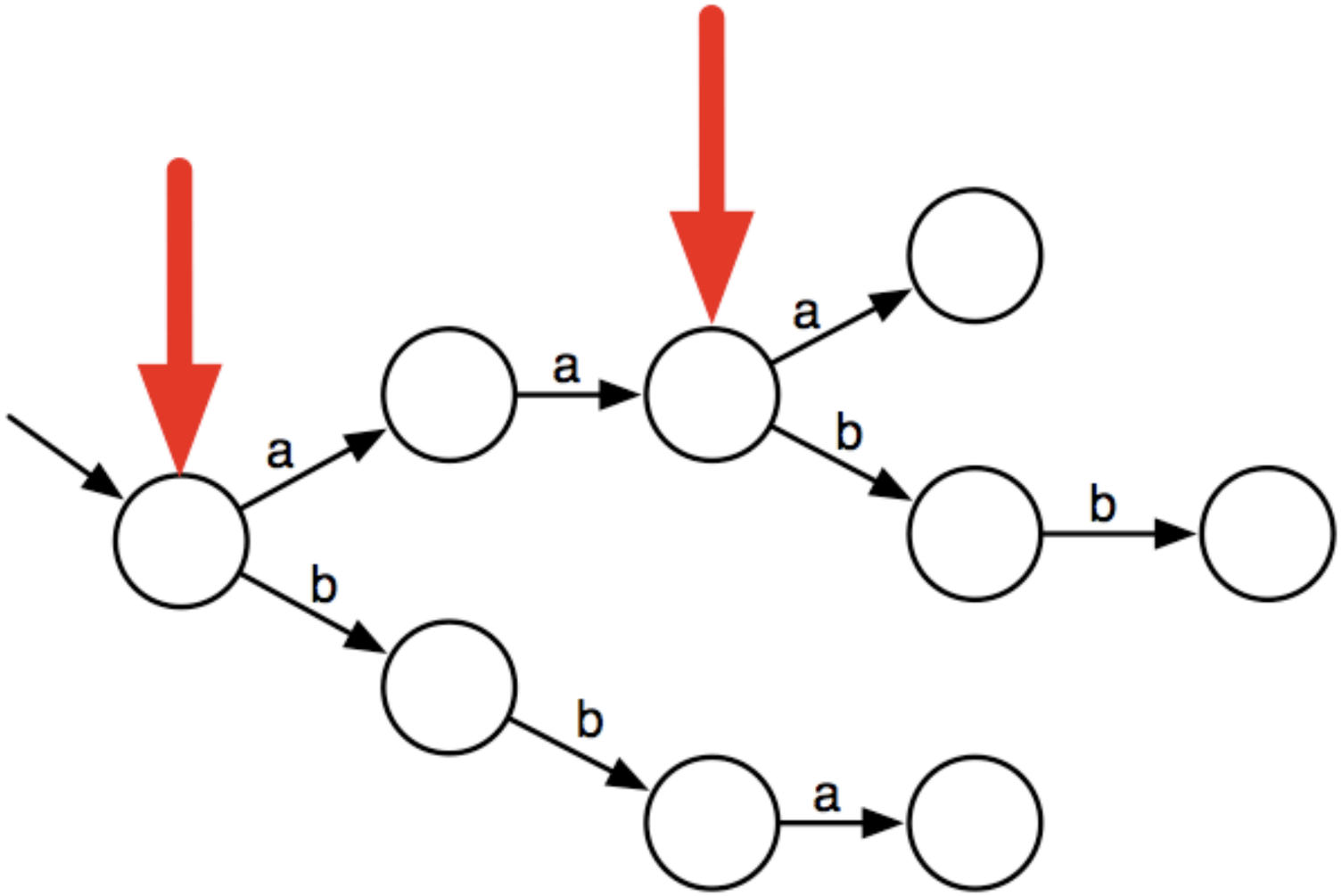
- Learning:
  - NP-hard to find smallest consistent DFA
  - Efficiently learnable in the limit

Learning DFAs



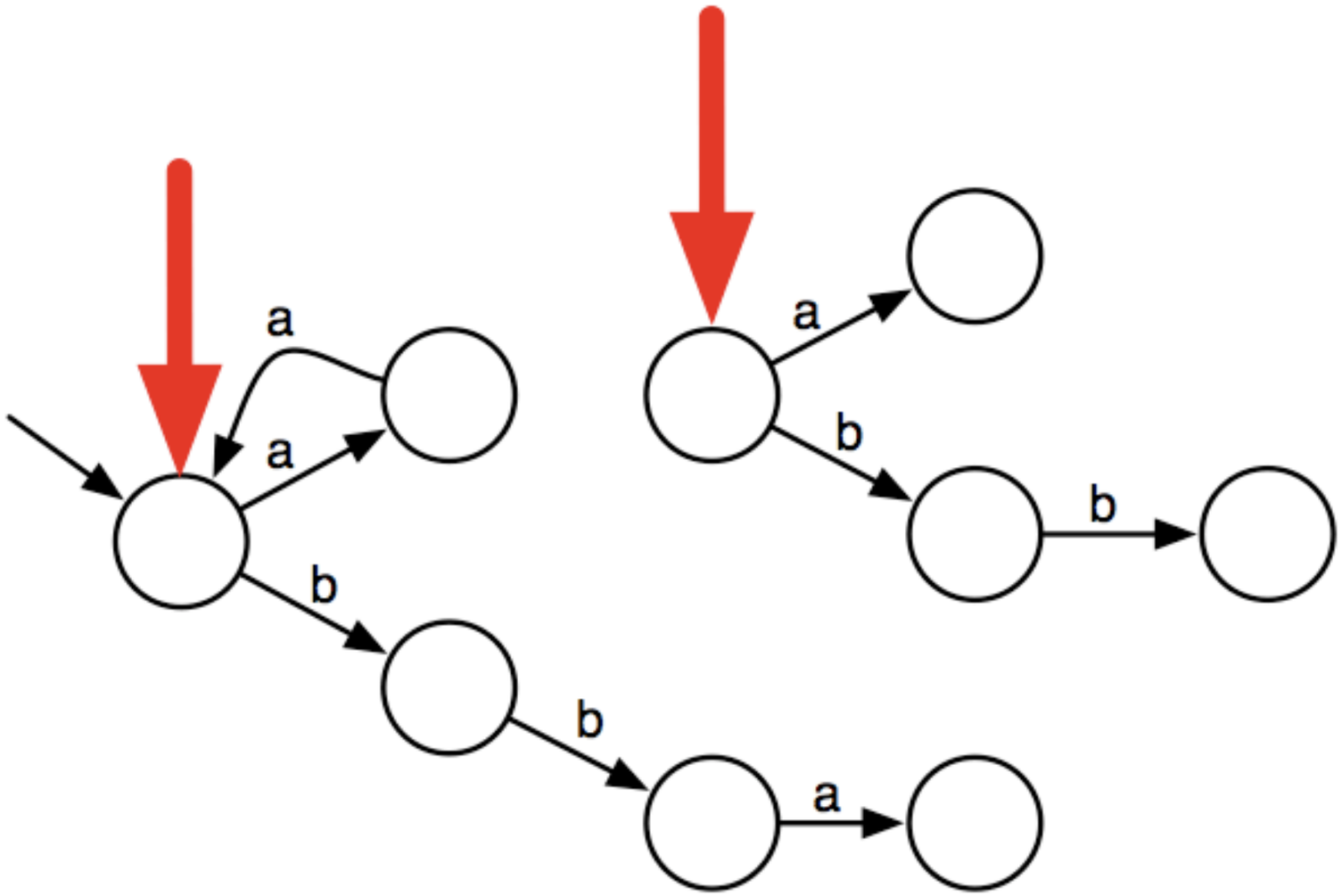
Observations: a, aaa, aaa, aabb, b, bb, bb, bba
represented as a prefix tree

Learning DFAs



State merging:
select two nodes

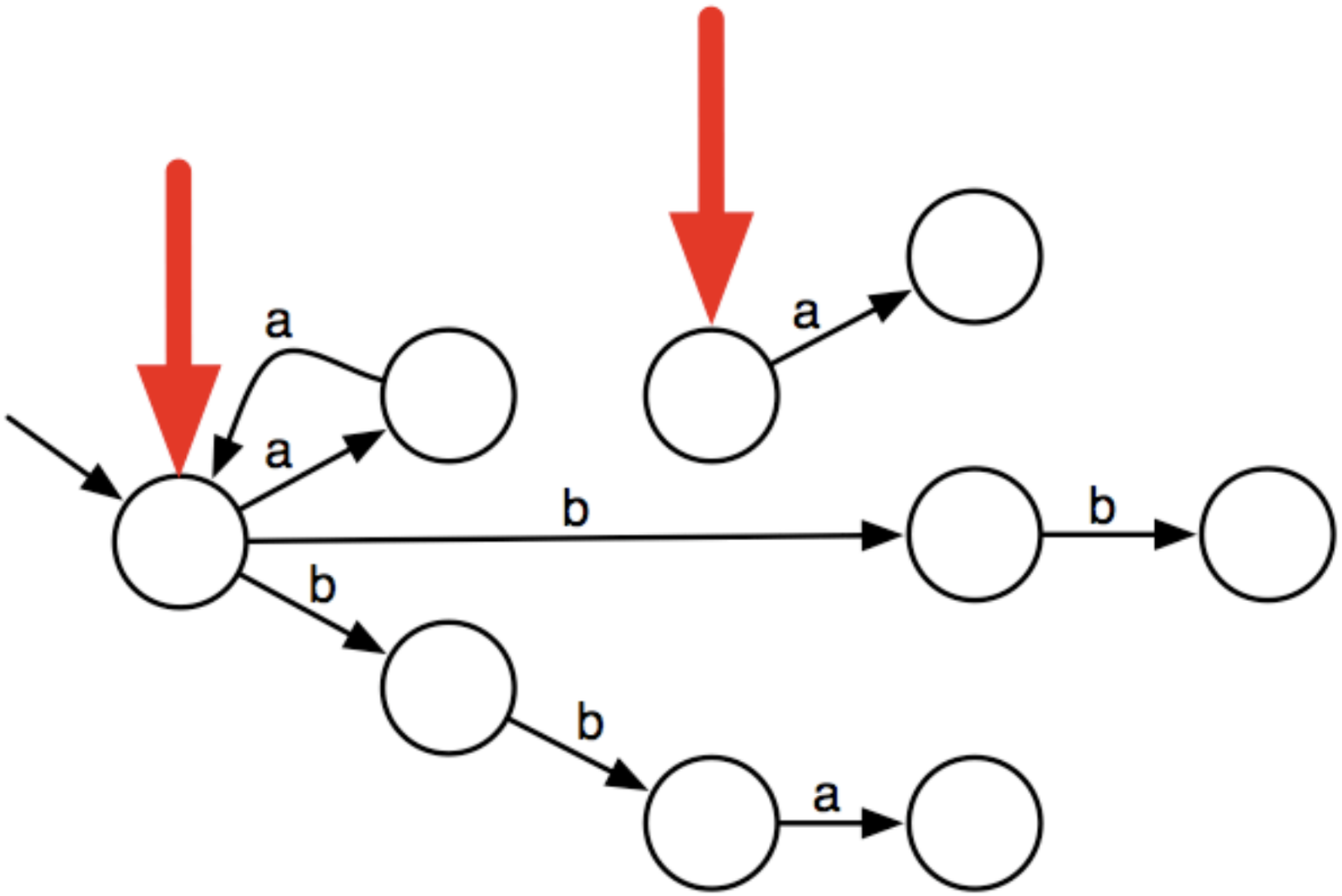
Learning DFAs



State merging:

move input transitions from one state to the other

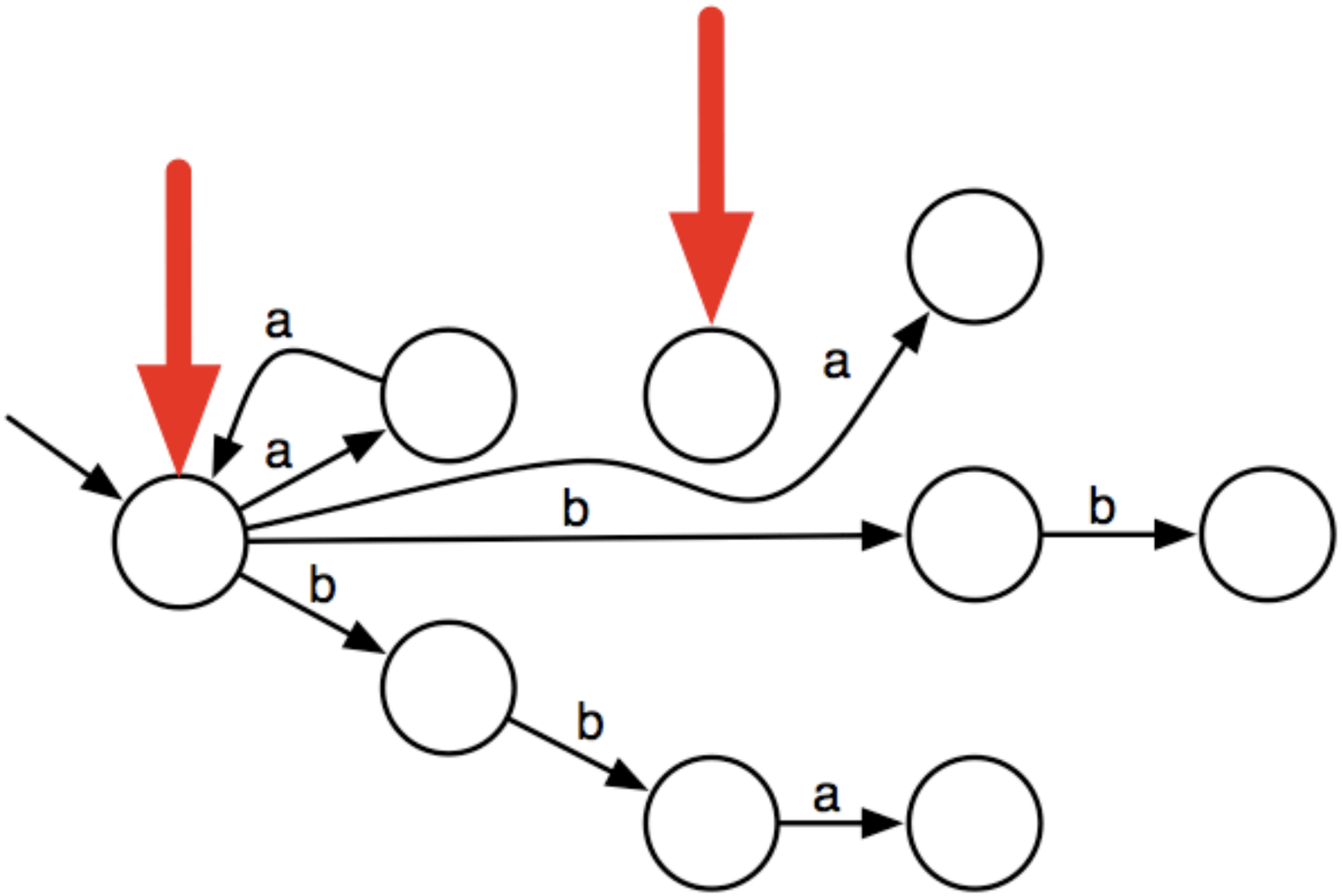
Learning DFAs



State merging:

move output transitions from one state to the other

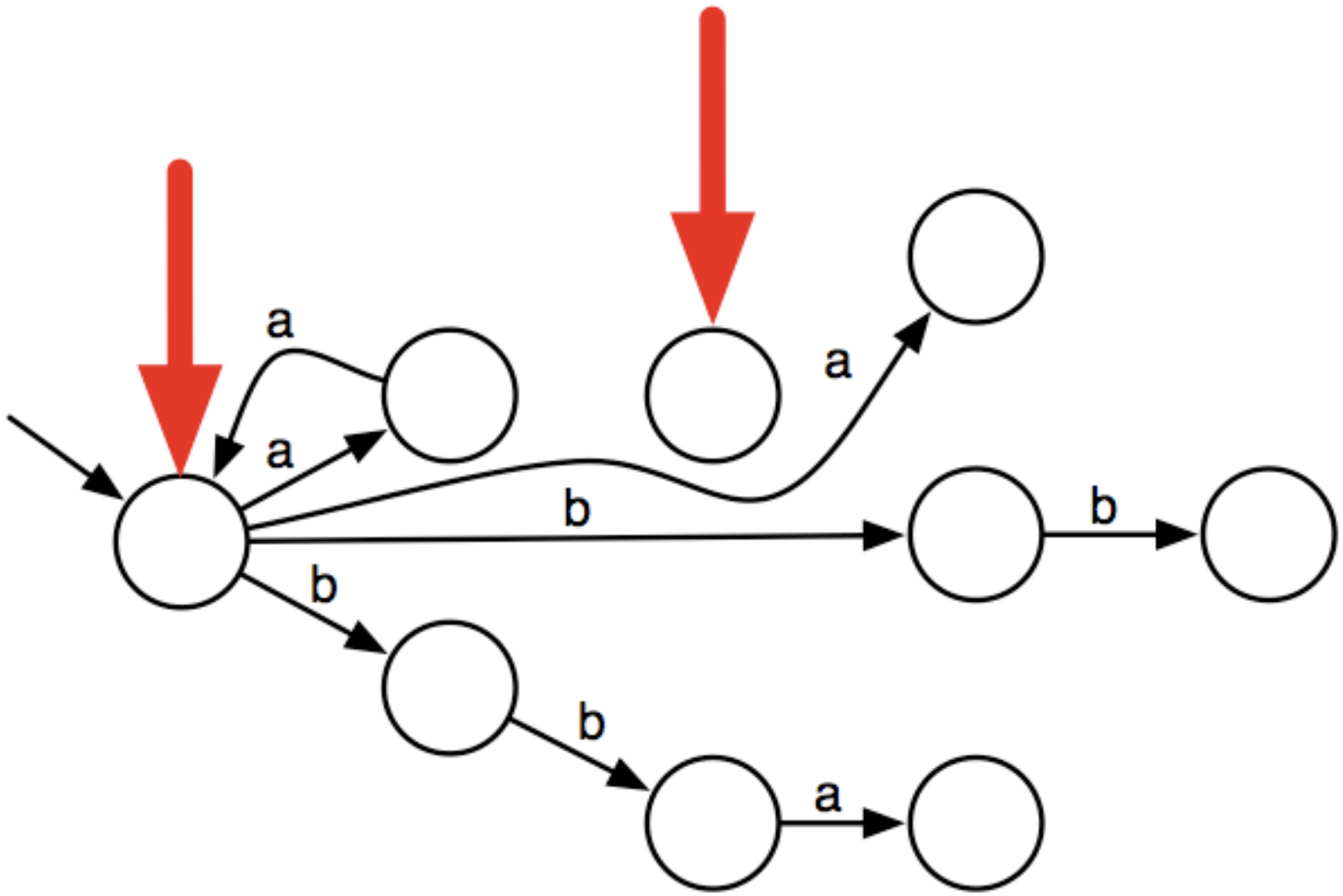
Learning DFAs



State merging:

move output transitions from one state to the other

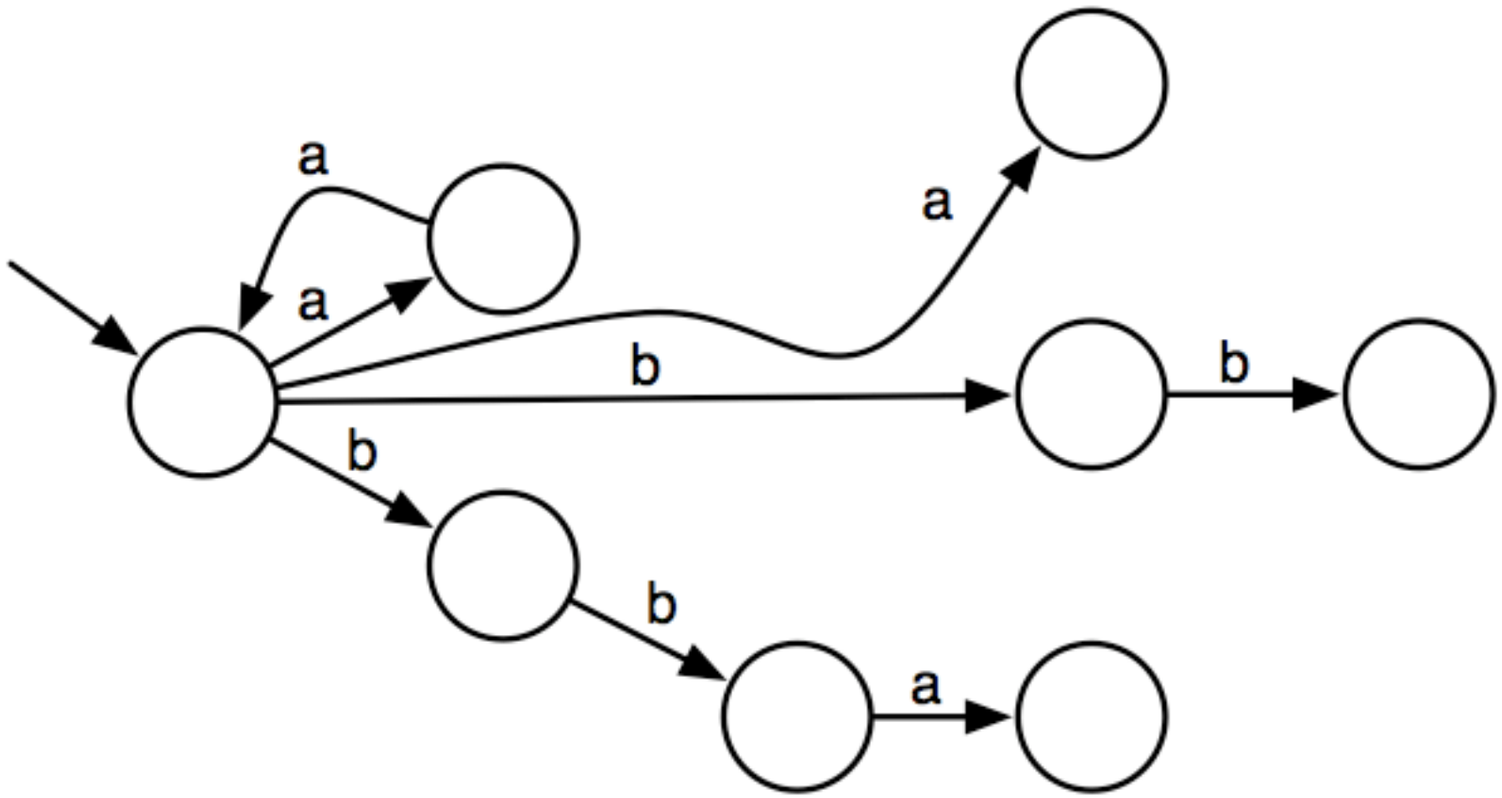
Learning DFAs



State merging:

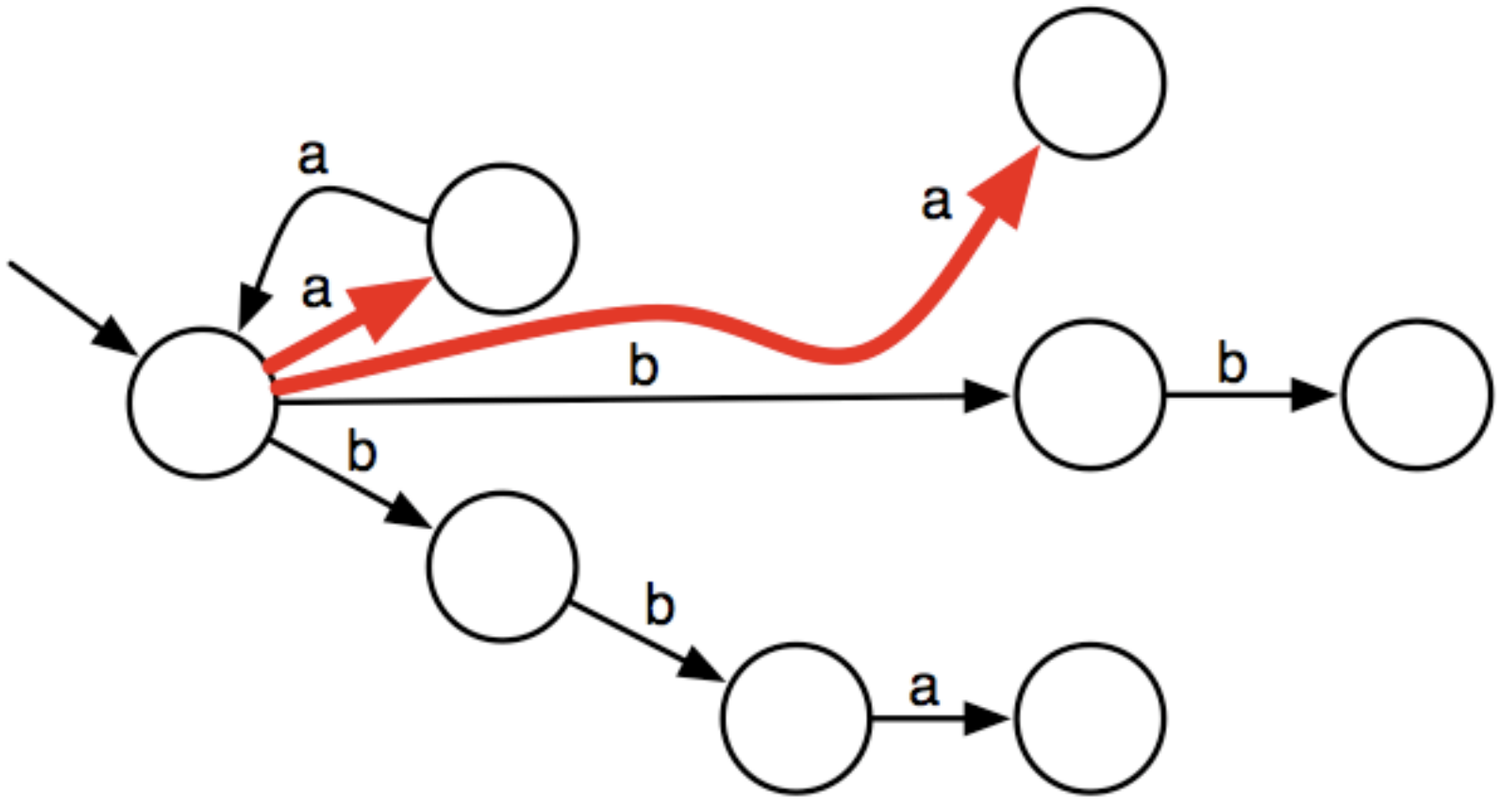
move output transitions from one state to the other

Learning DFAs



State merging:
delete the obsolete state

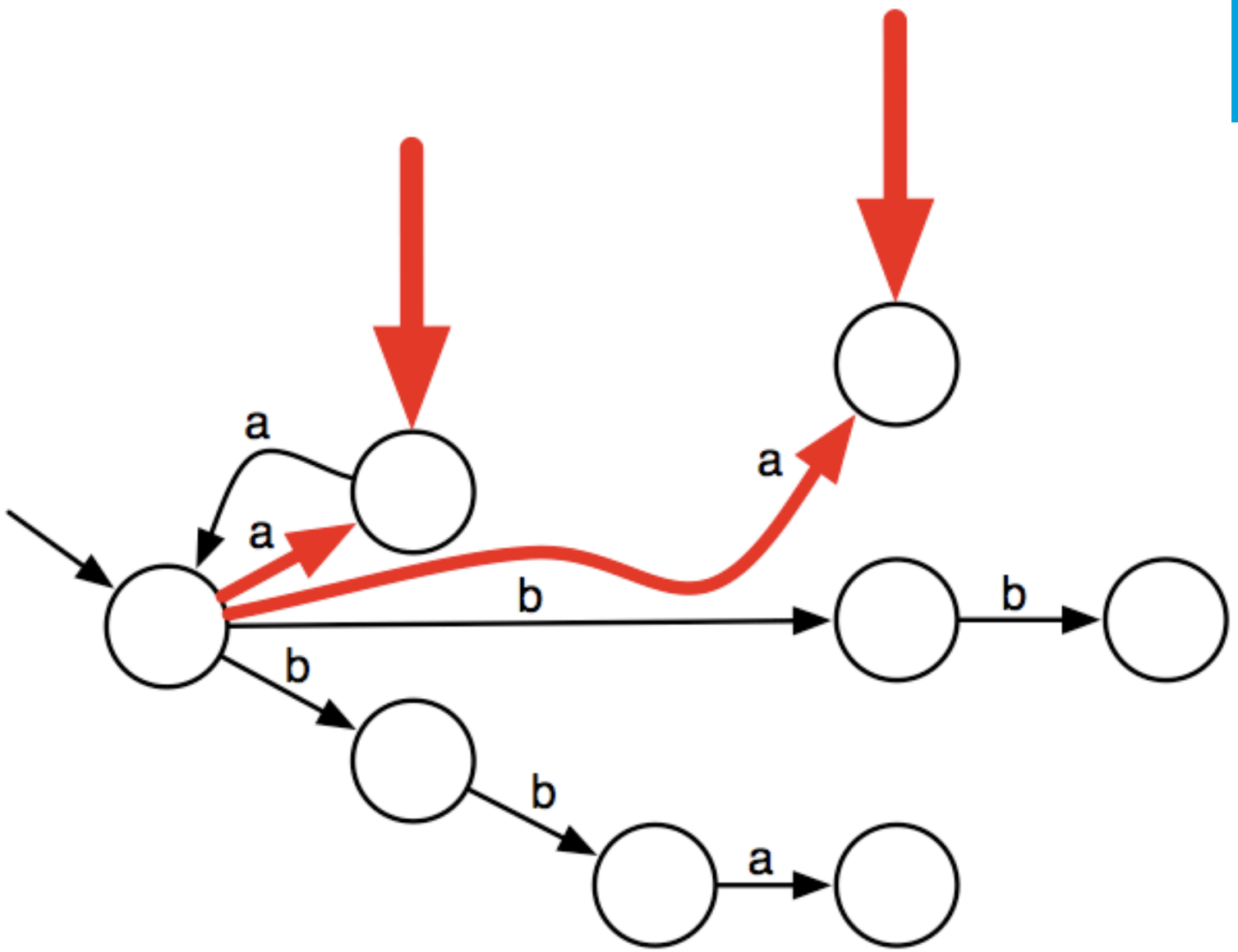
Learning DFAs



Determinization:

merge the targets of non-deterministic transitions

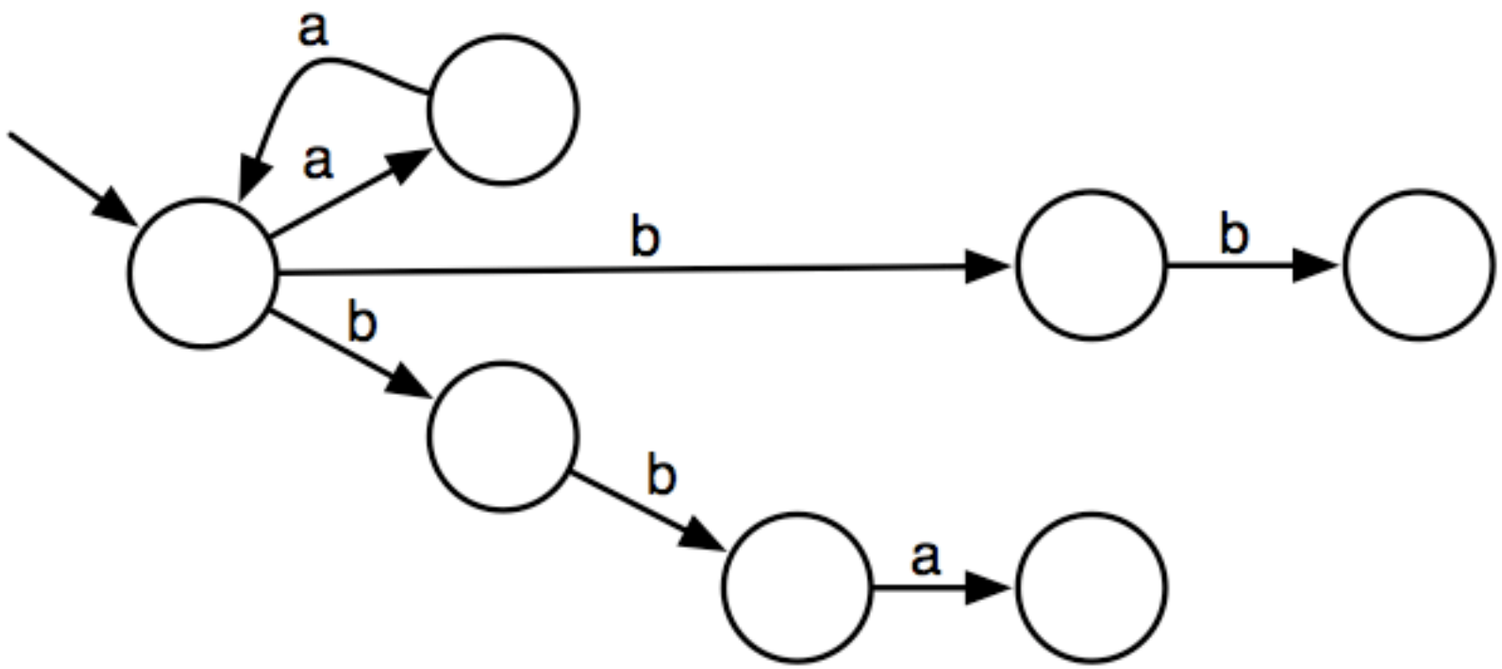
Learning DFAs



Determinization:

merge the targets of non-deterministic transitions

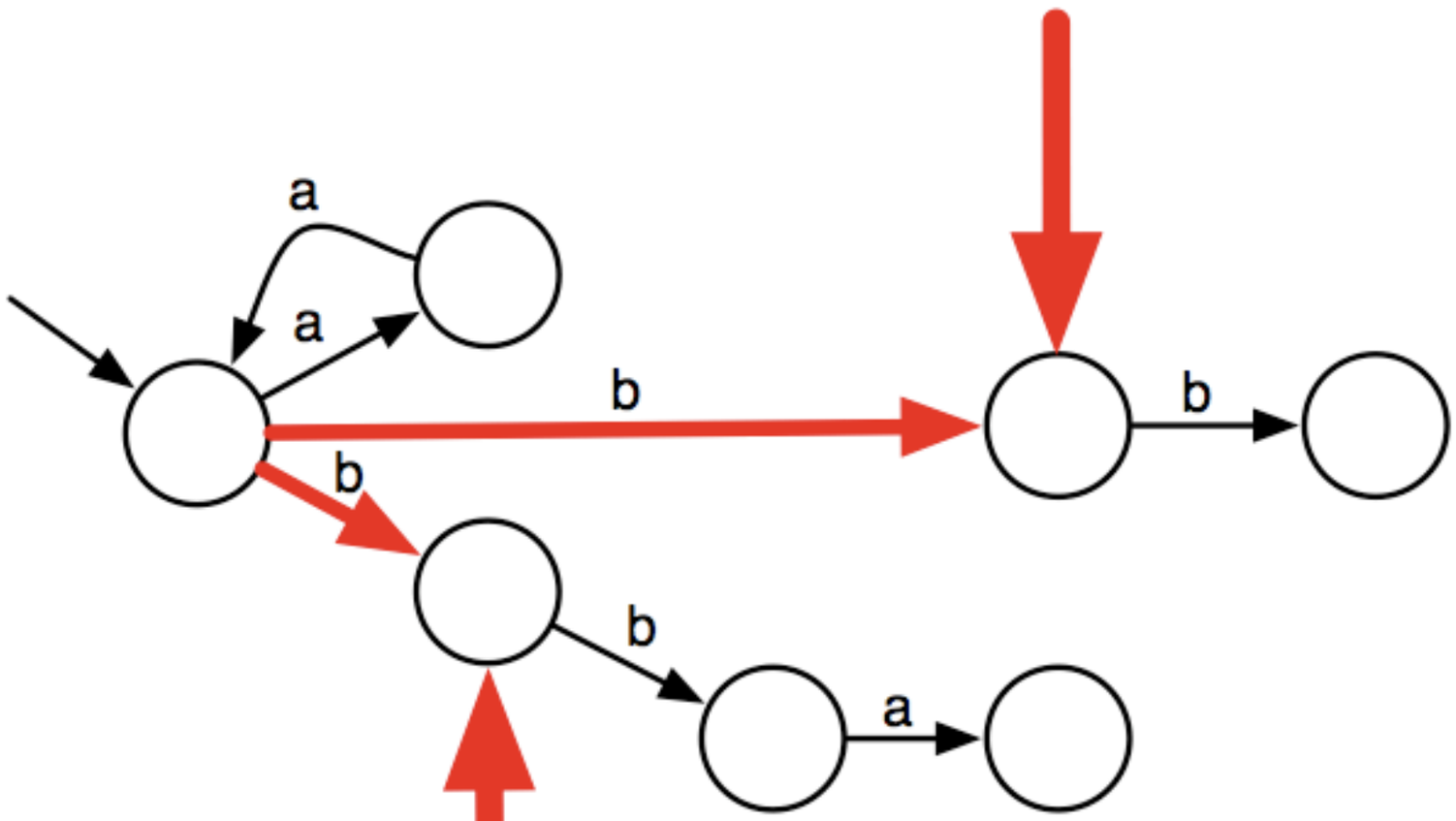
Learning DFAs



Determinization:

merge the targets of non-deterministic transitions

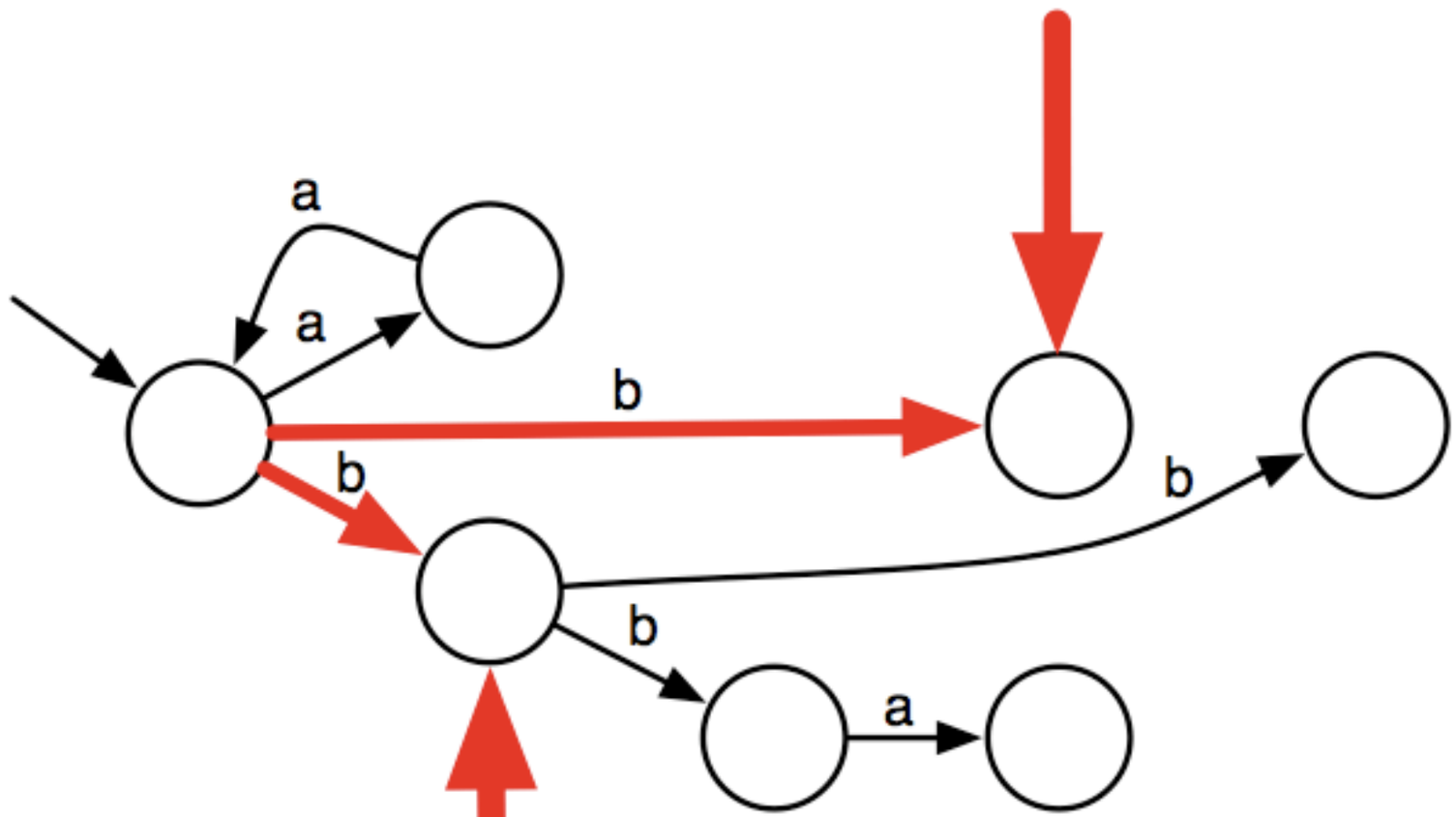
Learning DFAs



Determinization:

merge the targets of non-deterministic transitions

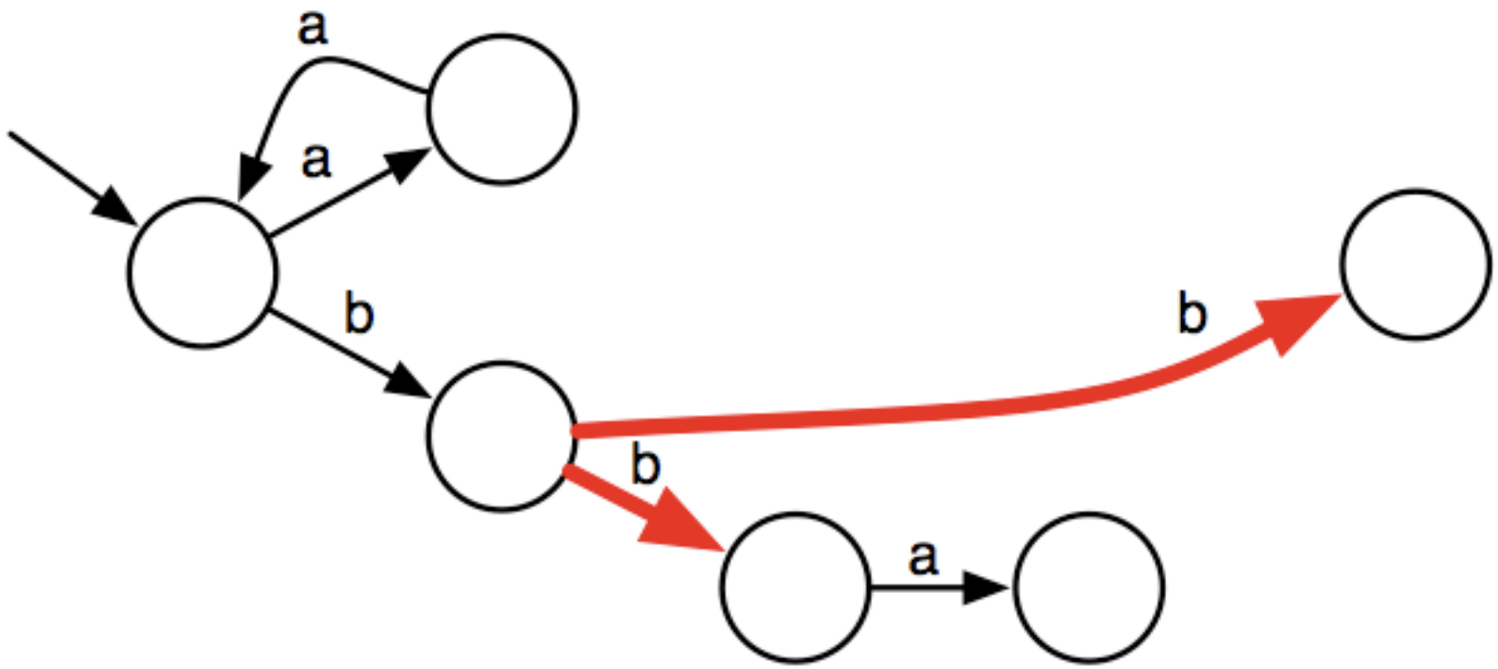
Learning DFAs



Determinization:

merge the targets of non-deterministic transitions

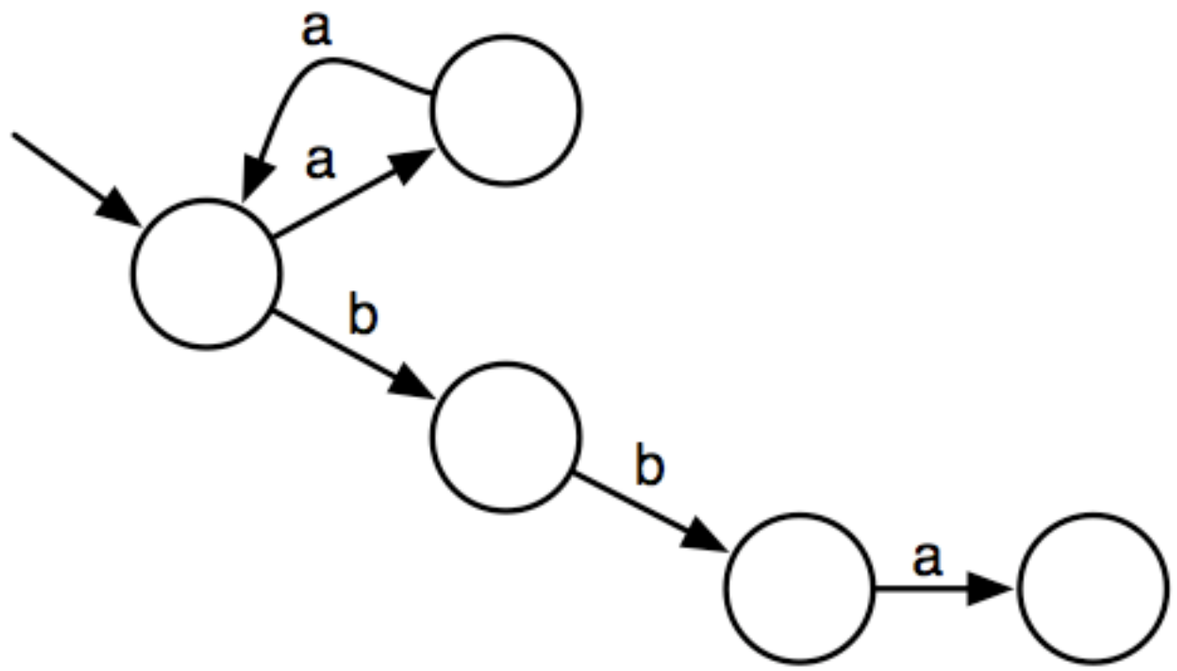
Learning DFAs



Determinization:

merge the targets of non-deterministic transitions

Learning DFAs



Select two new nodes to merge and iterate

Learning DFAs

- **State merging:**
  - **Start from a tree**
  - **Try all possible merges, including determinization**
  - **Perform the one that scores best**
  - **Iterate**

ALERGIA

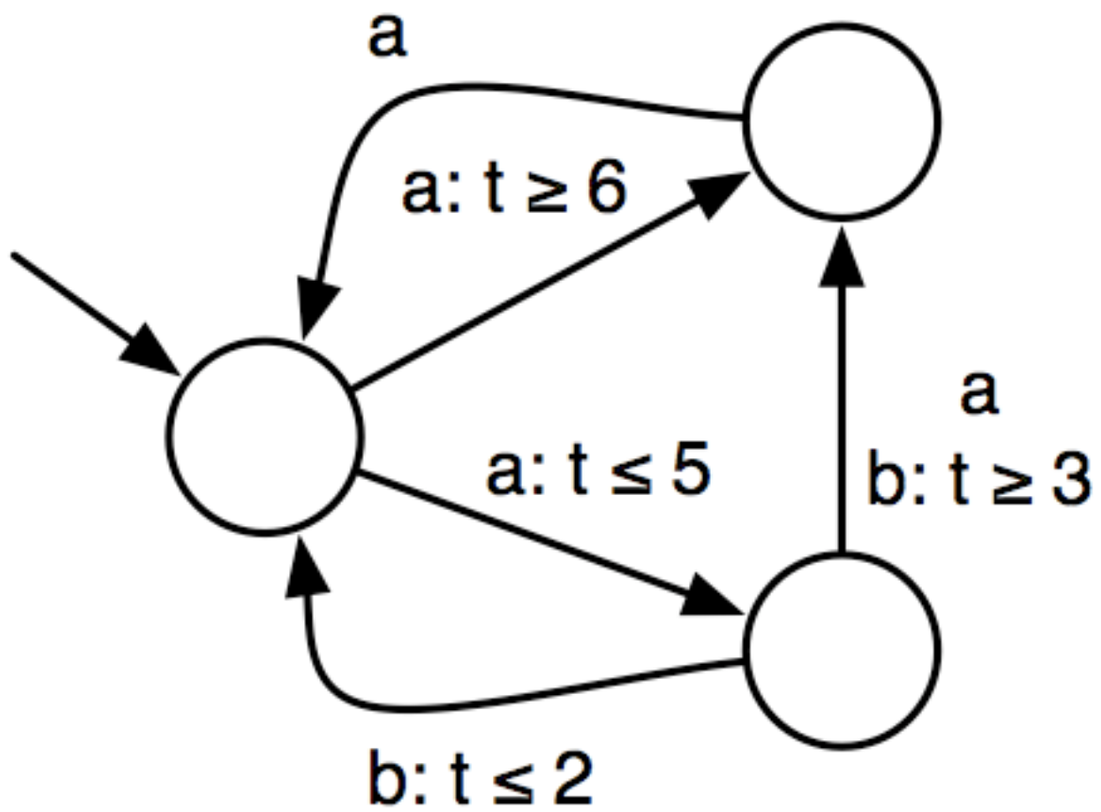
- Use a **norm** or **statistic**, like L_∞ or chi squared
- Define a **bound** ***b***, for the **similarity** between states
- It is not possible to merge states for which the norm or statistical dissimilarity is **greater than** ***b***
- **Score** = value of norm or statistical difference

- The statistics are computed for **futures** of states
- The **amount of data** necessary to converge with sufficient probability can be computed

Overview

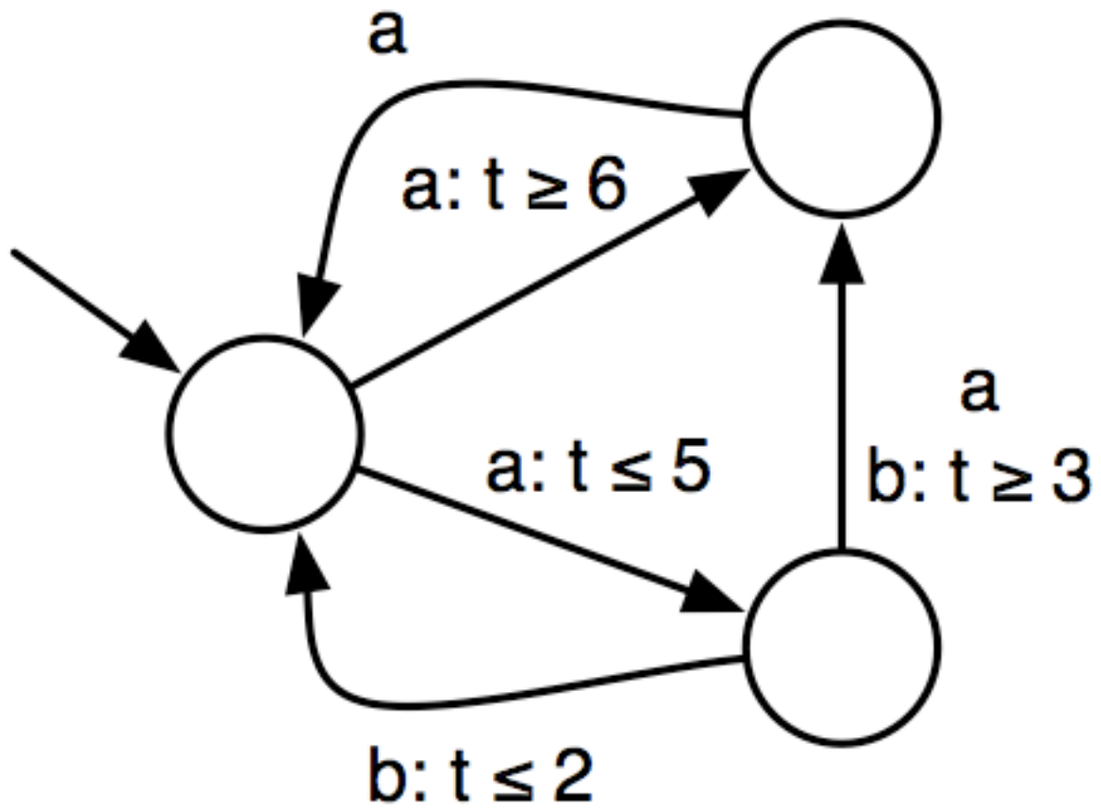
- Automata
- State merging
- Real-time automata
- Learning real-time automata using statistics
- Results

Real-time automata (DRTAs)



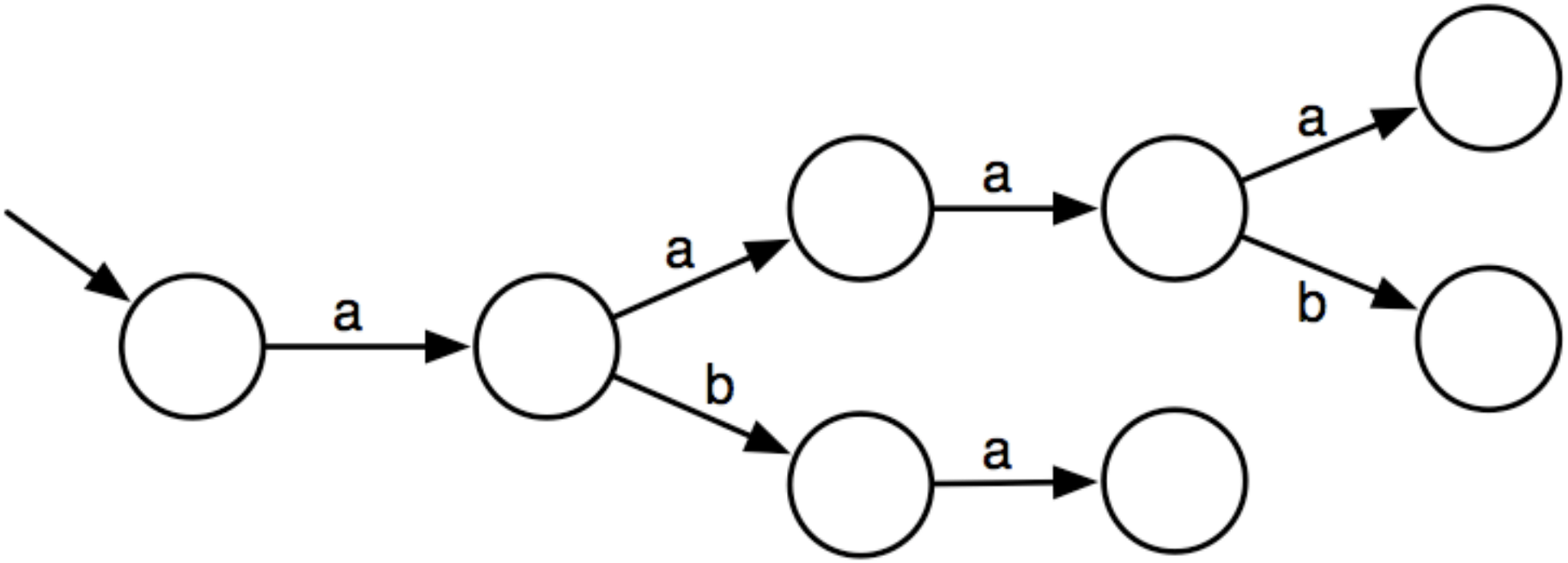
Transitions contain guards on time values

DRTAs and events



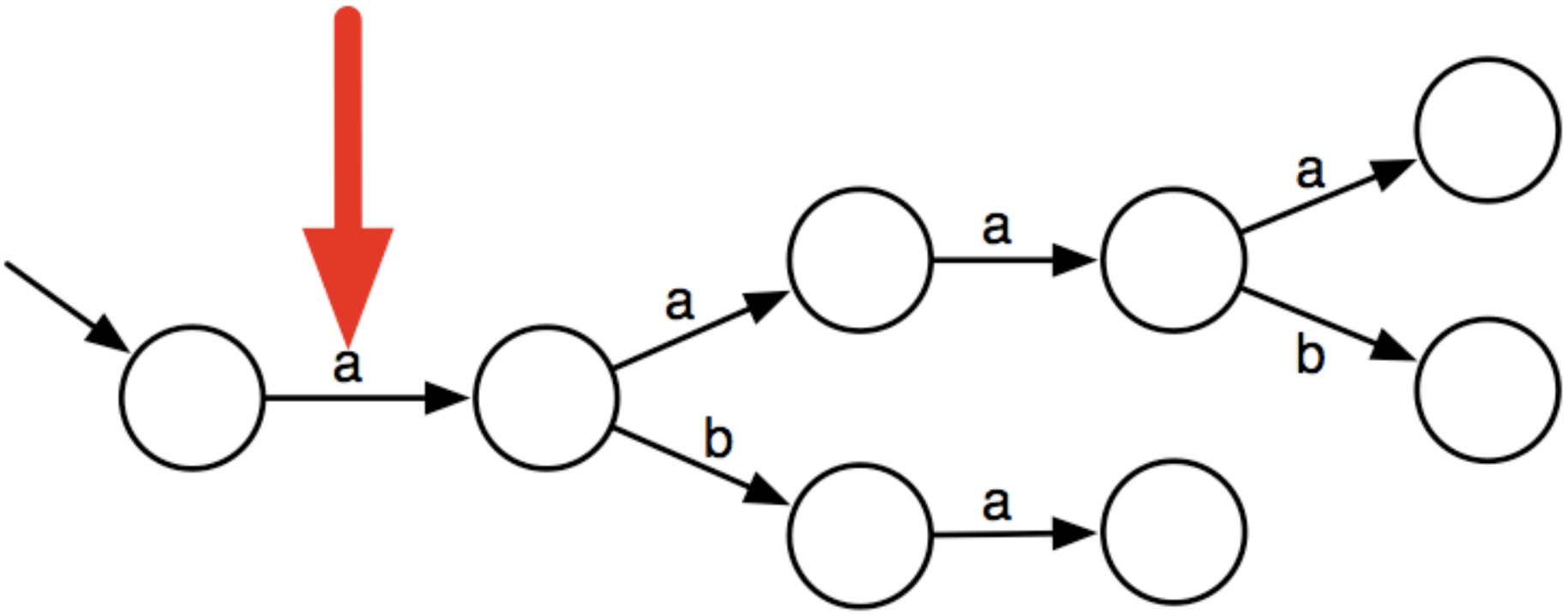
Produces timed strings:
(a,6)(a,2)(a,3); (a,2)(b,1)(a,1)(b,8)

Learning DRTAs



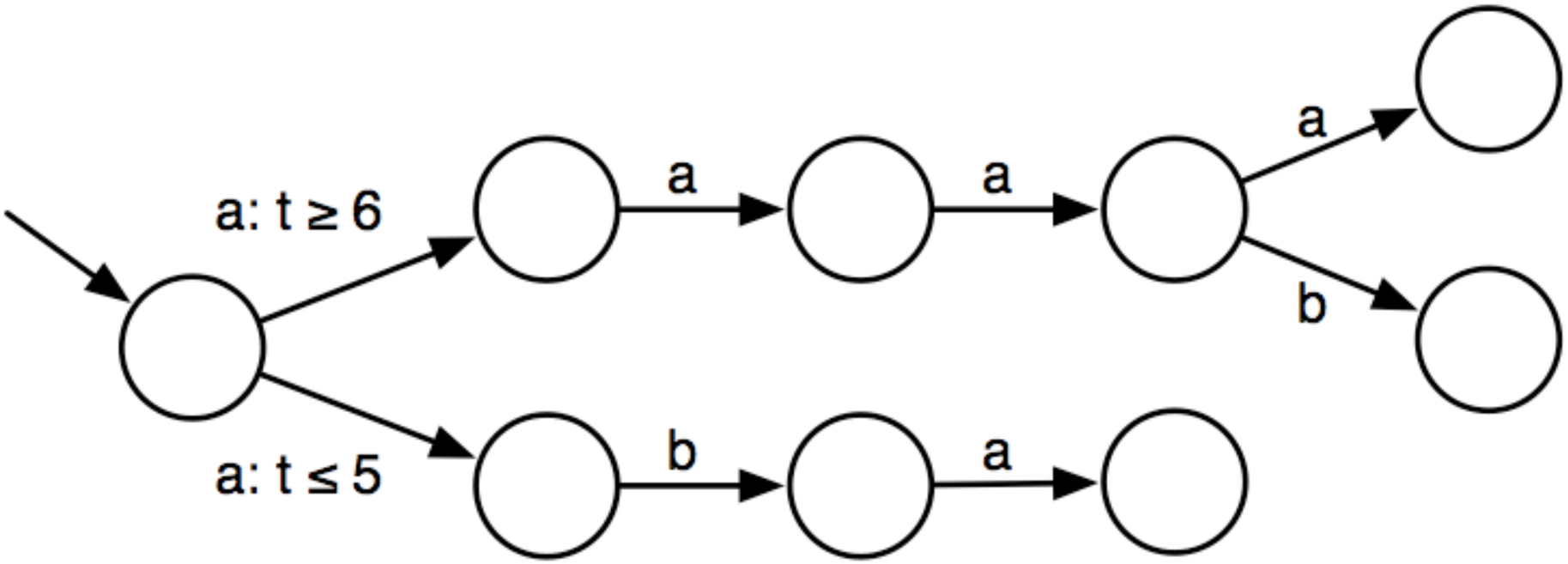
A prefix tree
all guards are set to true, [0, ∞)

Learning DRTAs



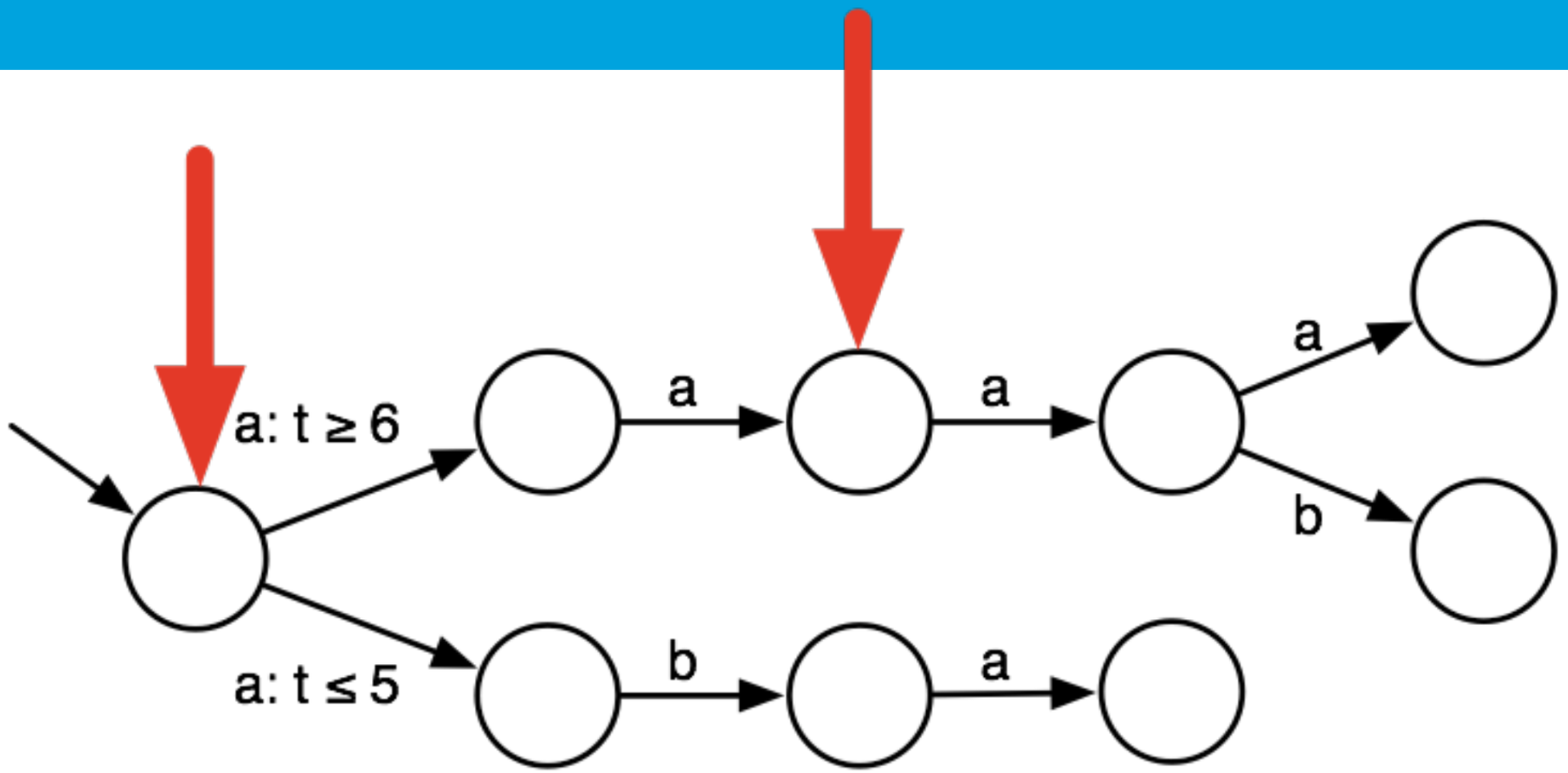
Transition splitting:
choose a transition

Learning DRTAs



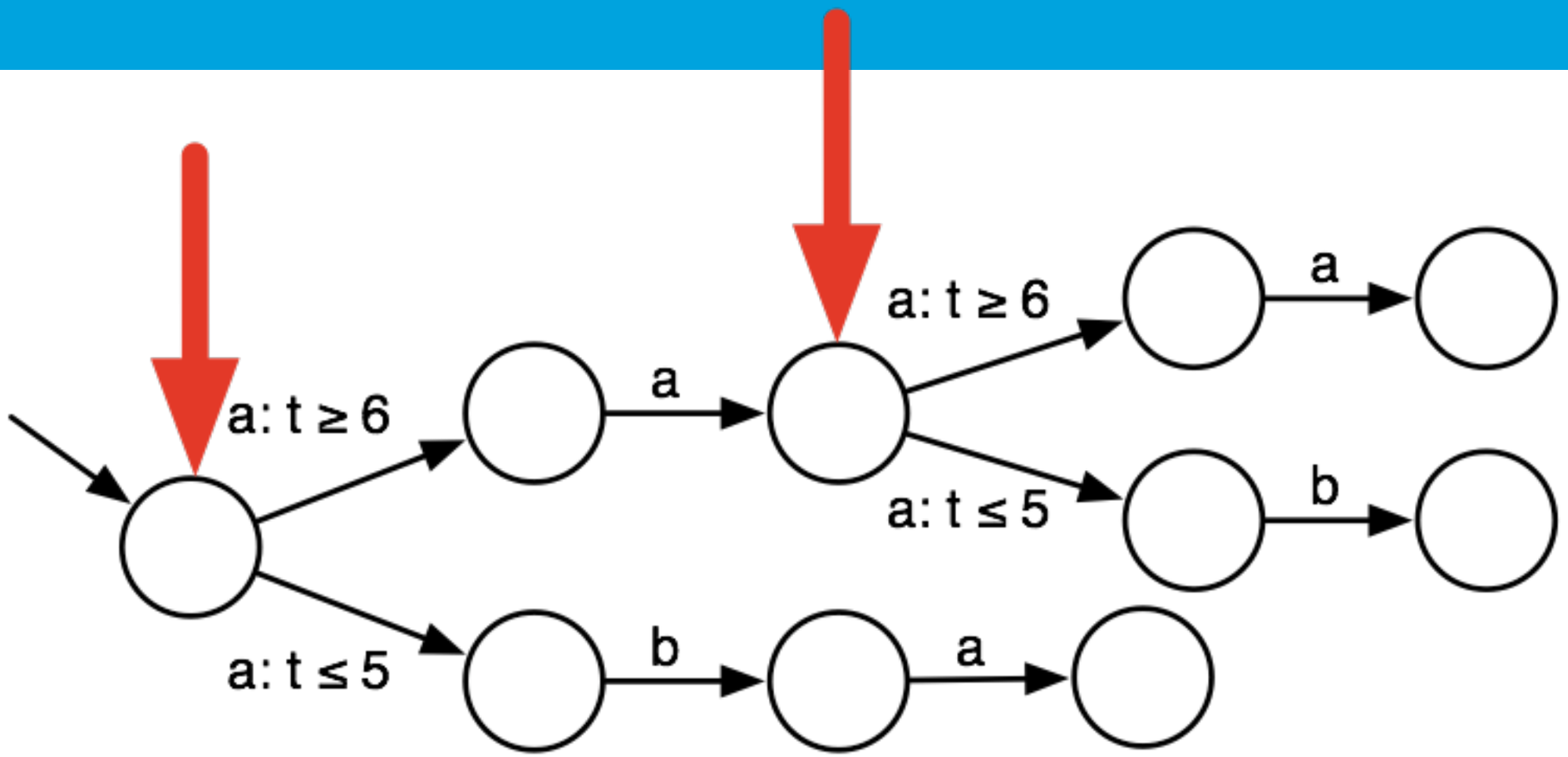
Split the transition and recalculate the subsequent part of the prefix tree

Learning DRTAs



Later, when merging states

Learning DRTAs



First split the transitions such that the guards match

Learning DRTAs

- State merging and transition splitting:
  - Start from a tree
  - Try all possible merges and splits
  - If one scores good:
    - Perform the one that scores best
  - Else
    - break
  - Iterate

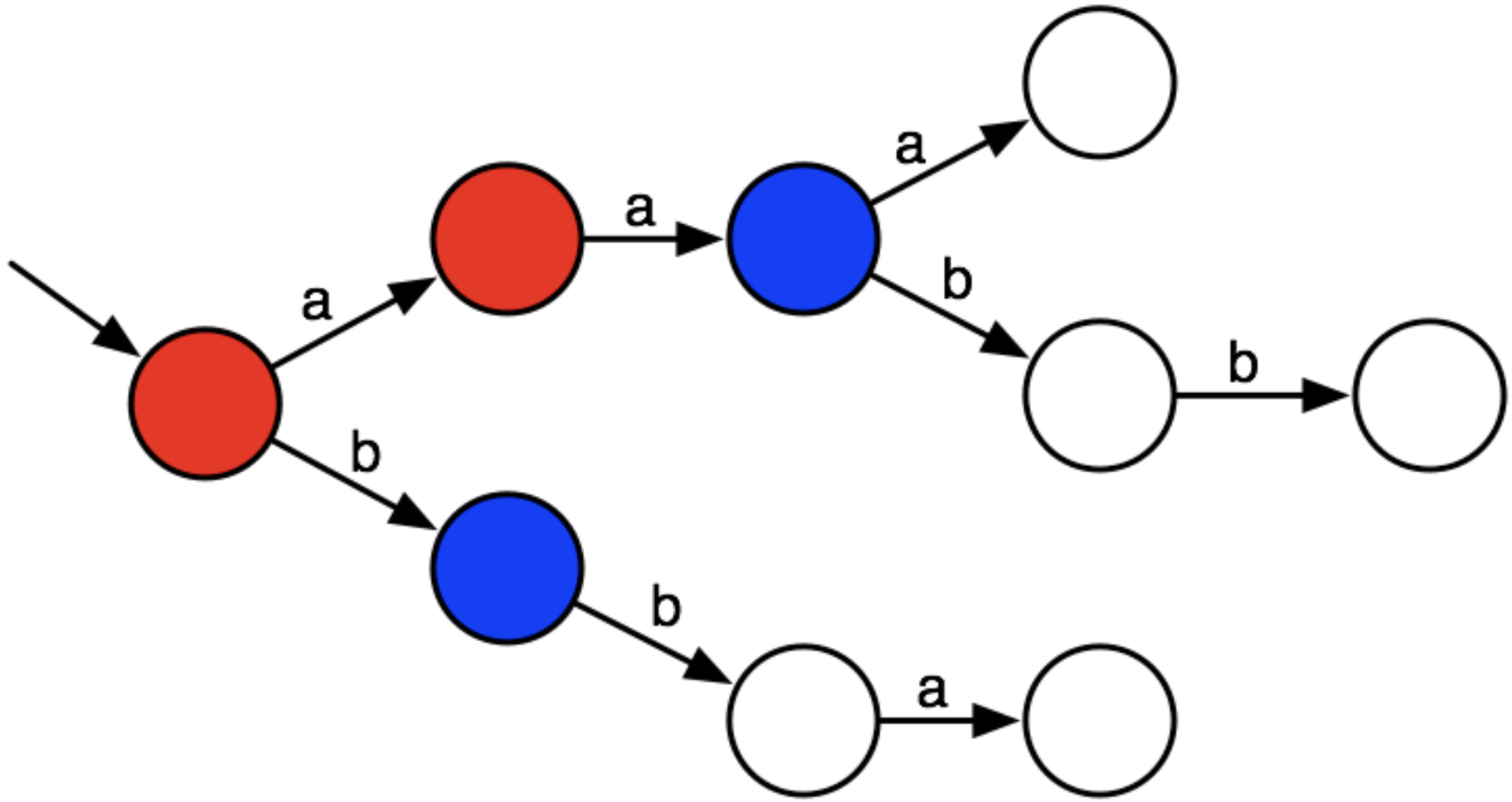
Overview

- Automata
- State merging
- Real-time automata
- Learning real-time automata using statistics
- Results

Learning DRTAs

- **Input: one sets S_+ of timed strings**
- **Use the state merging and transition splitting algorithm, but**
  - **Maintain a core of inferred states using the red-blue framework**
  - **Only perform splits/merges of the transition/state that is visited by the most examples from S_+**

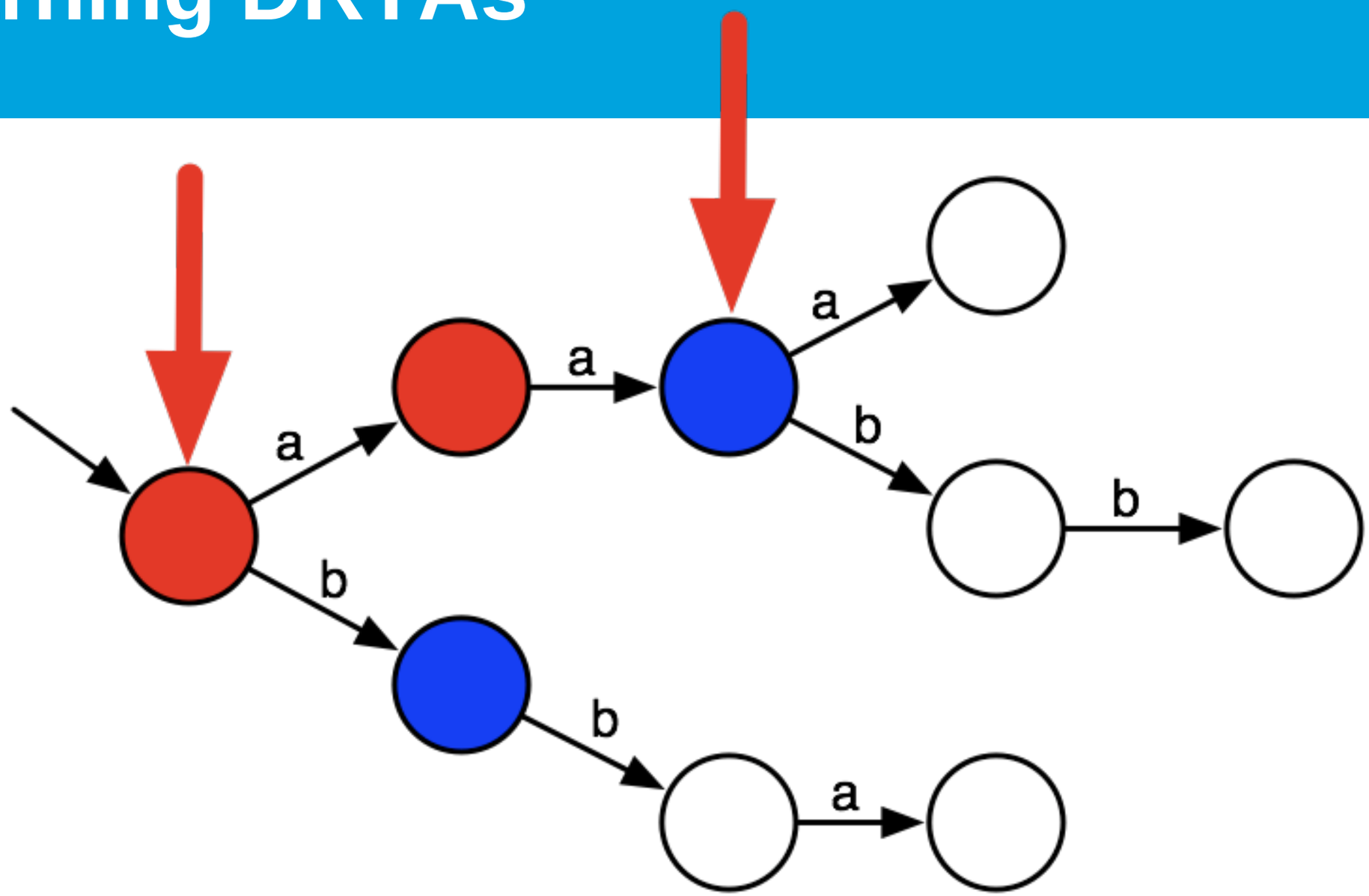
Learning DRTAs



Red = inferred

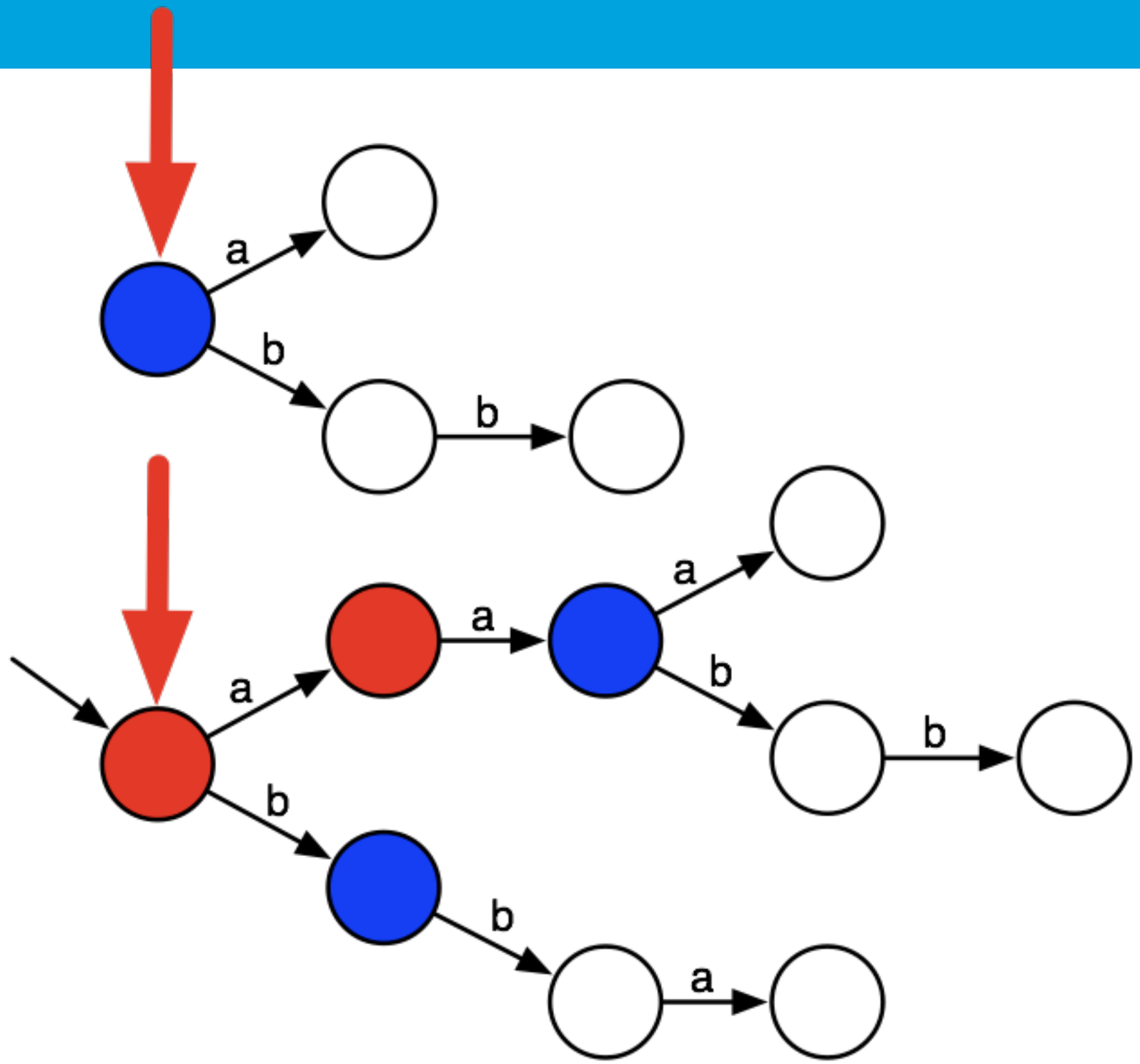
Blue = merge candidate

Learning DRTAs



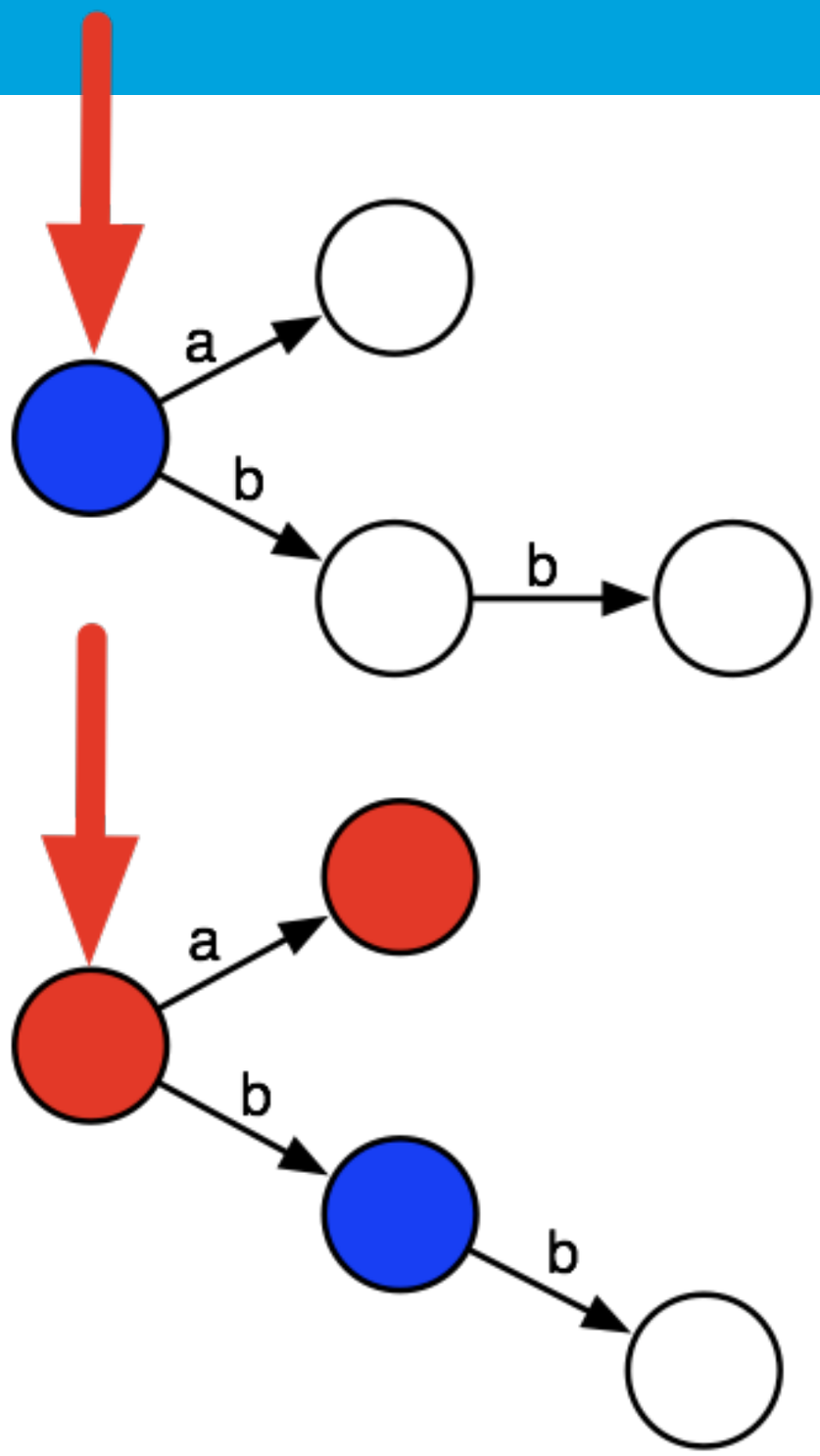
States should be merged if their future behavior is similar

Learning DRTAs



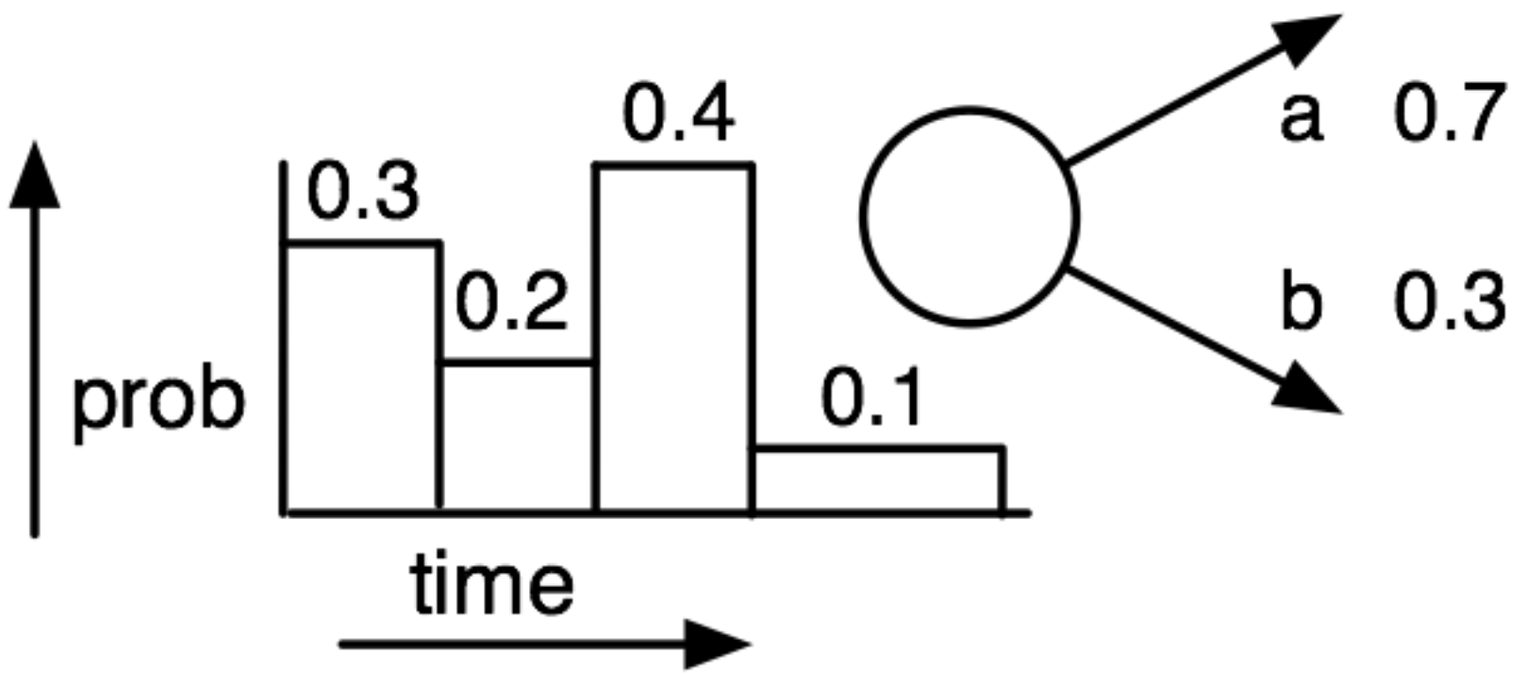
This behavior is determined by the trees rooted by the two states

Learning DRTAs



We are only interested in the overlapping behavior, i.e., the behavior for which we have data

DRTA Distributions

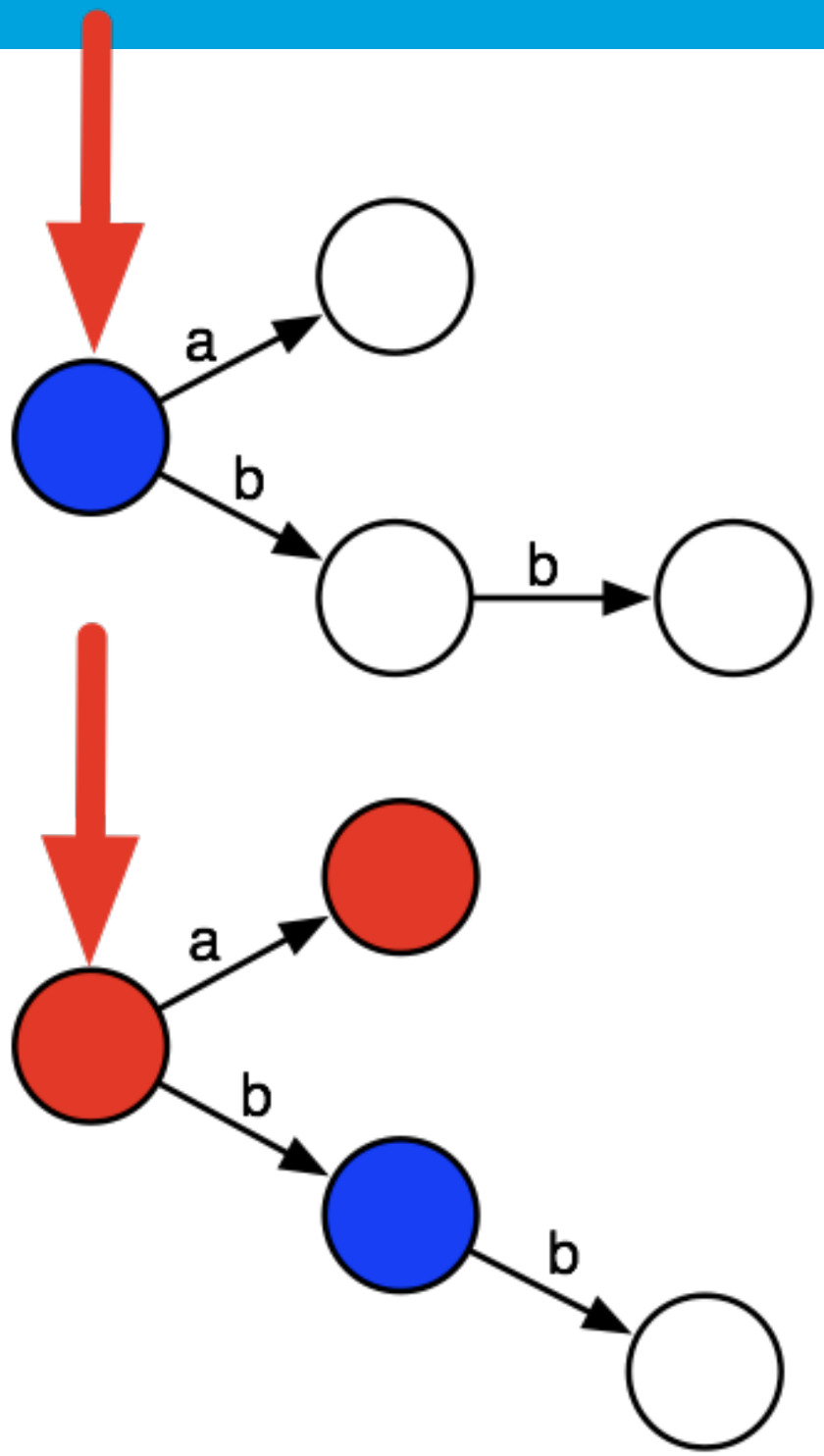


By counting the occurrences of symbols, and bins of time points (forming a histogram) in S_+
Every state defines a distribution over timed symbols

Three statistics

- **Two states and their future states define a distribution over timed strings**
- **Two states are similar if:**
  1. **Chi squared, for S_+ their distributions are not statistically dissimilar**
  2. **Likelihood ratio, merging them results in a significantly smaller model with respect to the likelihood of S_+**
  3. **Akaike information criterium (AIC), merging them results in a smaller AIC (model selection criterium) for S_+**

Chi squared

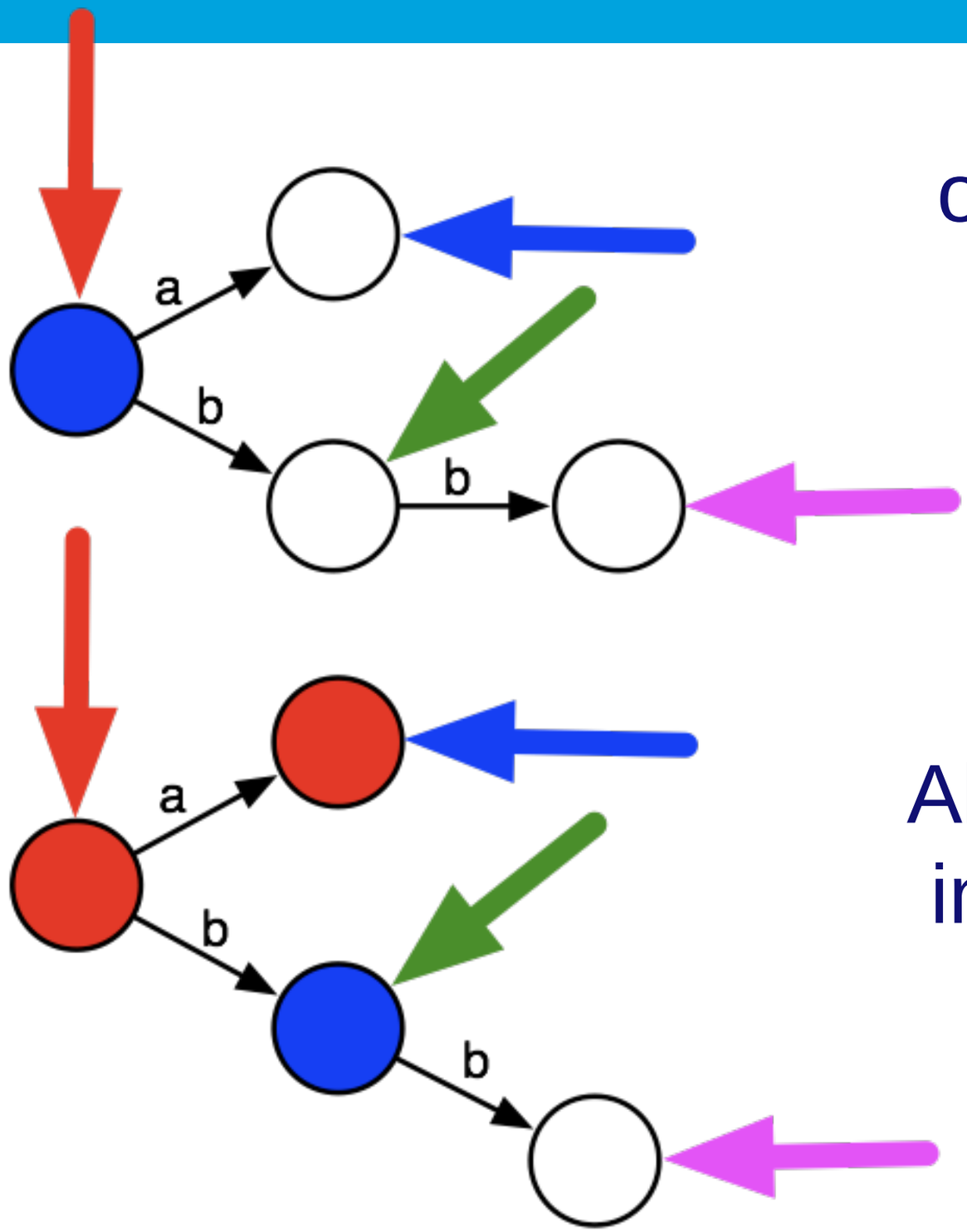


In the two states:

Compute a Chi-squared test for independence of the distributions over time values and symbols

This results in two p-values

Chi squared

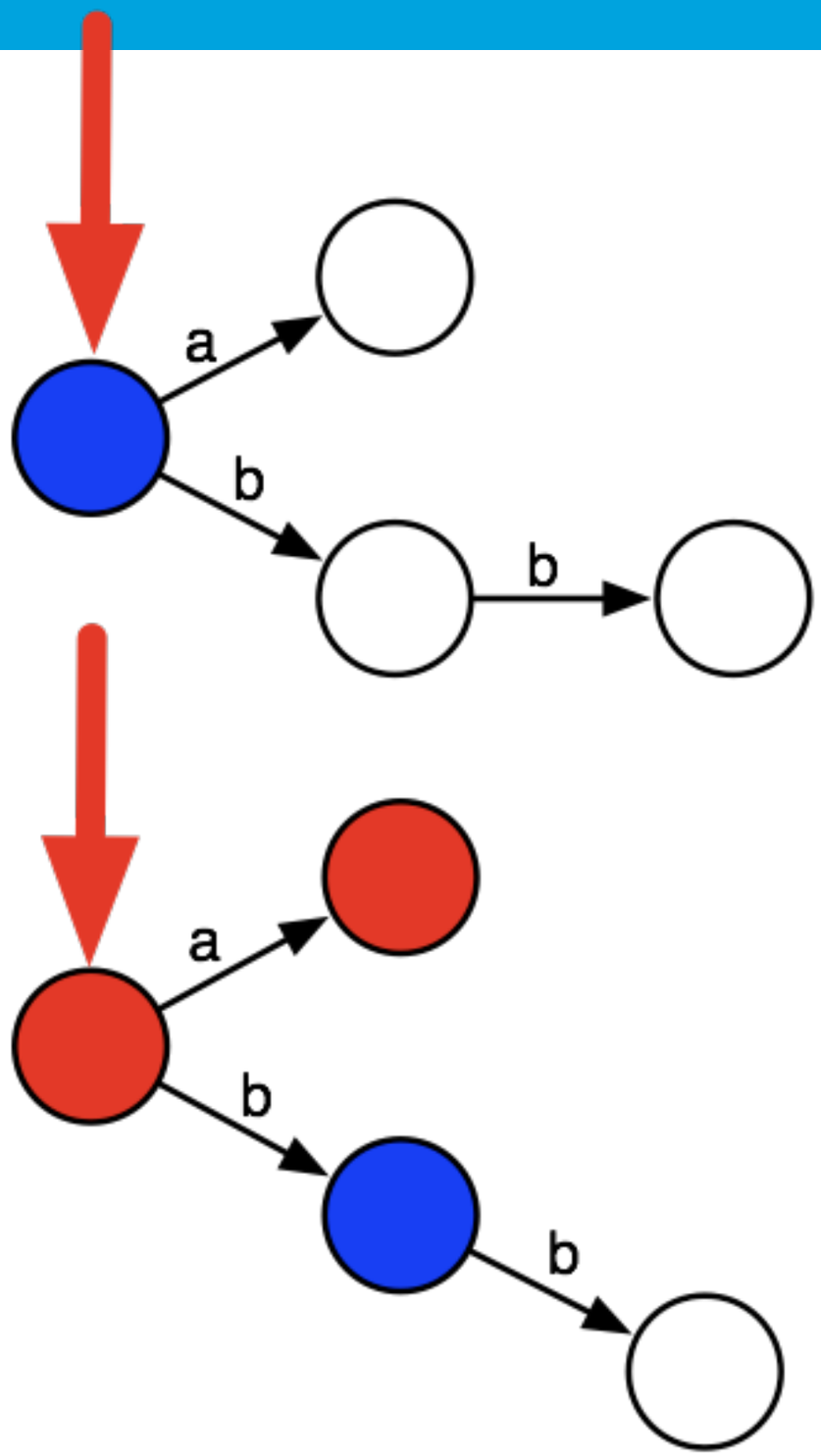


Do the same for every other overlapping future states

The result is many p-values

All of these are combined into a single test using a consensus test

Likelihood ratio

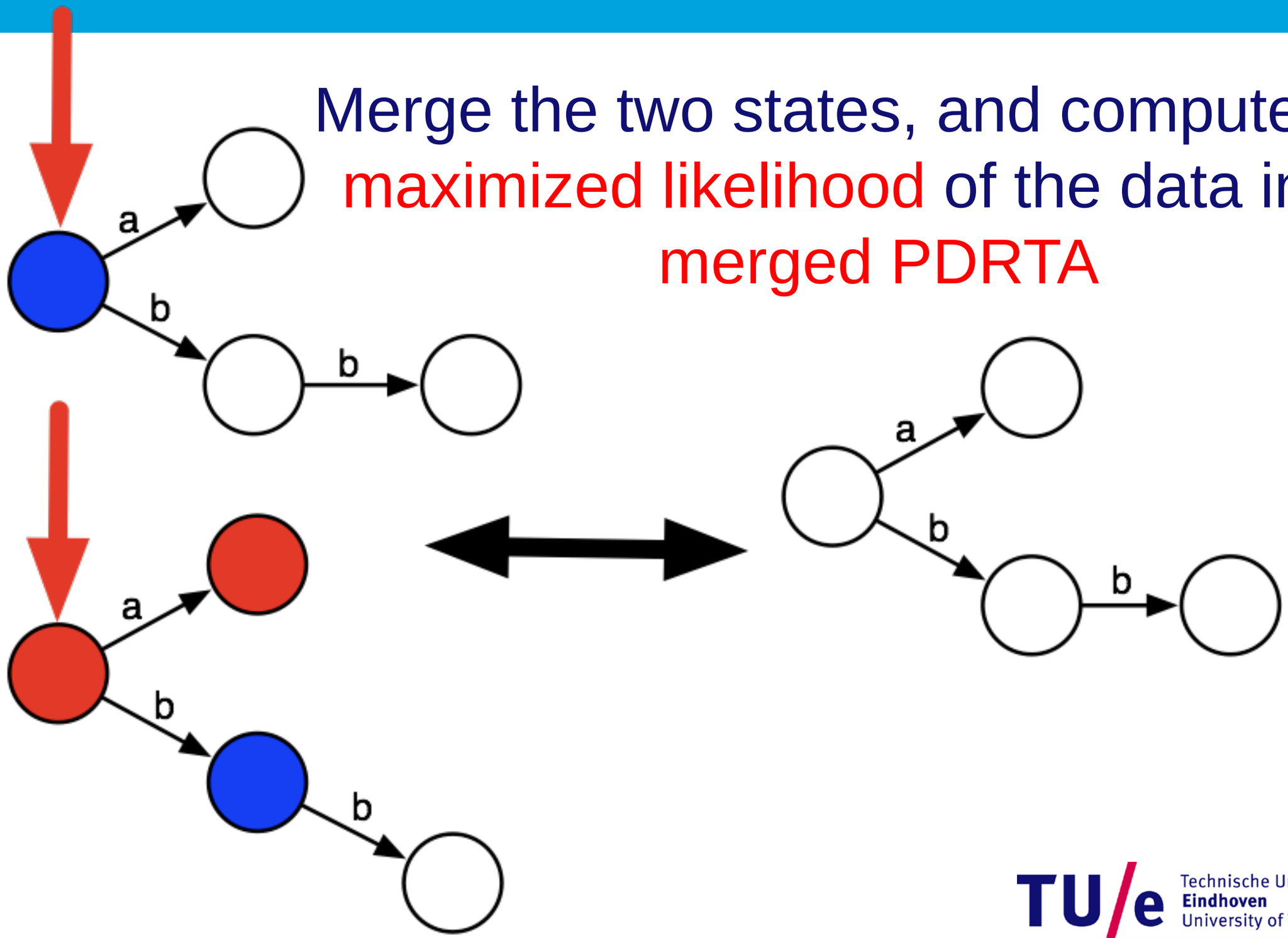


In the two states and the future states:

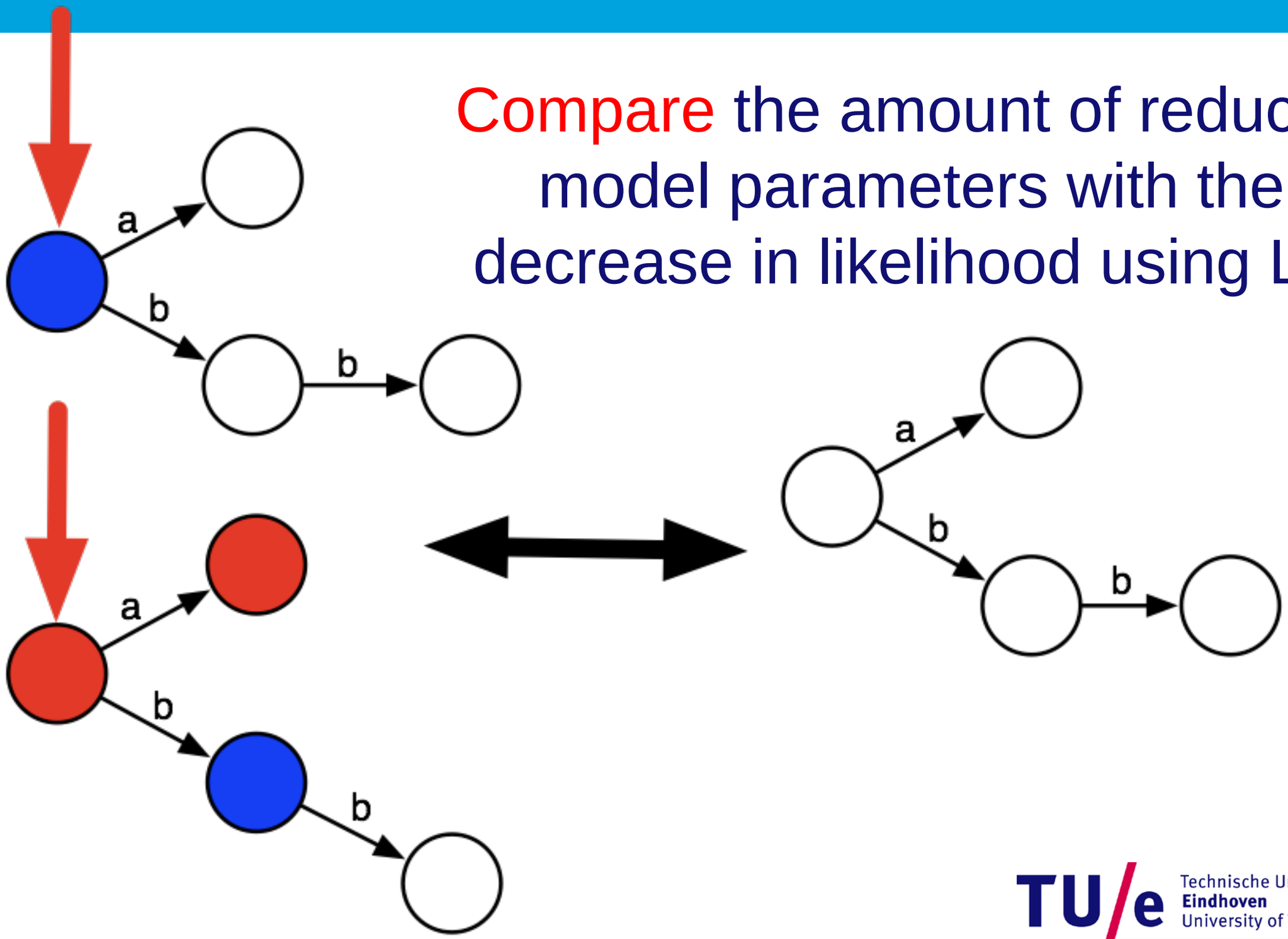
Determine the maximized likelihood of the data

Likelihood ratio

Merge the two states, and computed the maximized likelihood of the data in the merged PDRTA

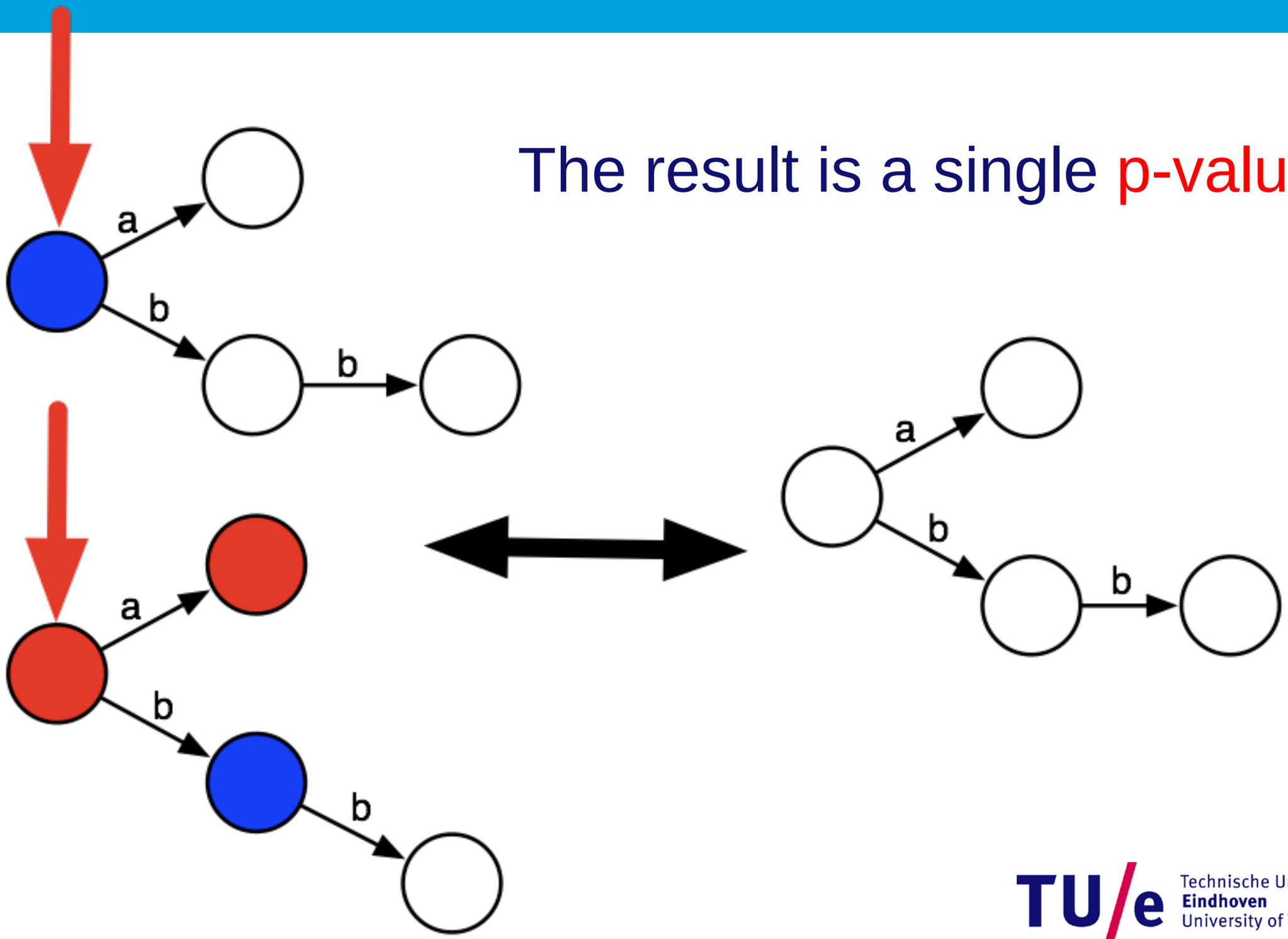


Likelihood ratio



Compare the amount of reduced model parameters with the decrease in likelihood using LR

Likelihood ratio



The result is a single p-value

Using p-values

- Try all possible merges and splits of the transition to a blue state, compute their similarity p-values
- If the smallest split p-value is less than 0.05:
  - perform the split with the smallest p-value
- Else if the largest merge p-value is greater then 0.05:
  - perform the merge with the largest p-value
- Else color the blue state red

Using the AIC

- Try all possible merges and splits of the transition to a blue state, compute their AIC values
- If the AIC of a DRTA resulting from a merge or split is less than the AIC of the current DRTA:
  - Perform this merge or split
- Else
  - Color the blue state red

Computing statistics efficiently

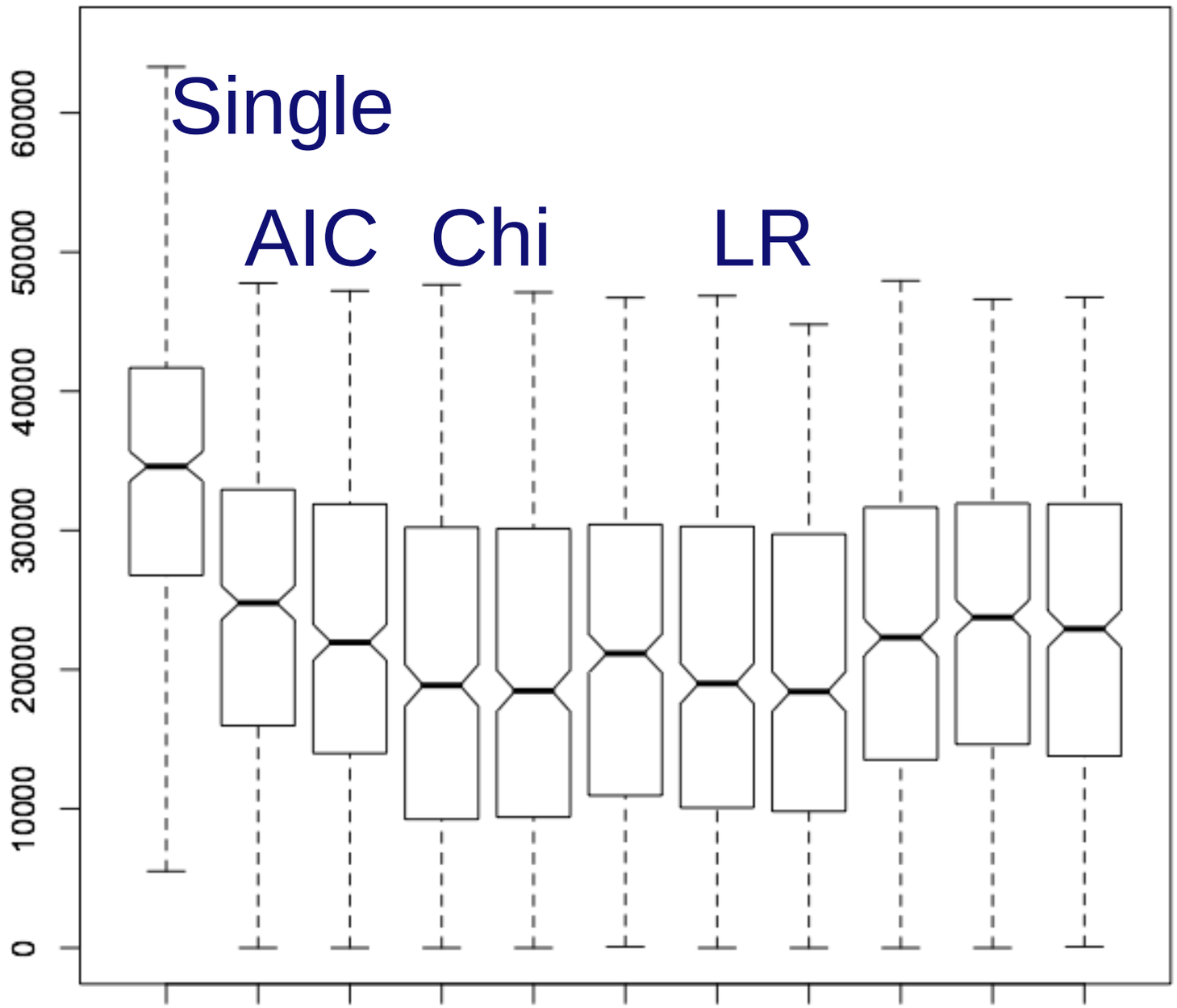
- **A transition can be split in many different ways:**
  - **If there are 10000 possible time values, then there are 10000 possible splits**
- **The statistics are computed based on occurrence counts in suffix trees**
- **Recomputing these counts for all splits is done efficently by performing updates:**
  - **Two consecutive split values effect only a small number of timed strings**
- **Computing the statistics is done very efficiently in standard libraries**

Results 1000 strings

AIC result

-

AIC optimal

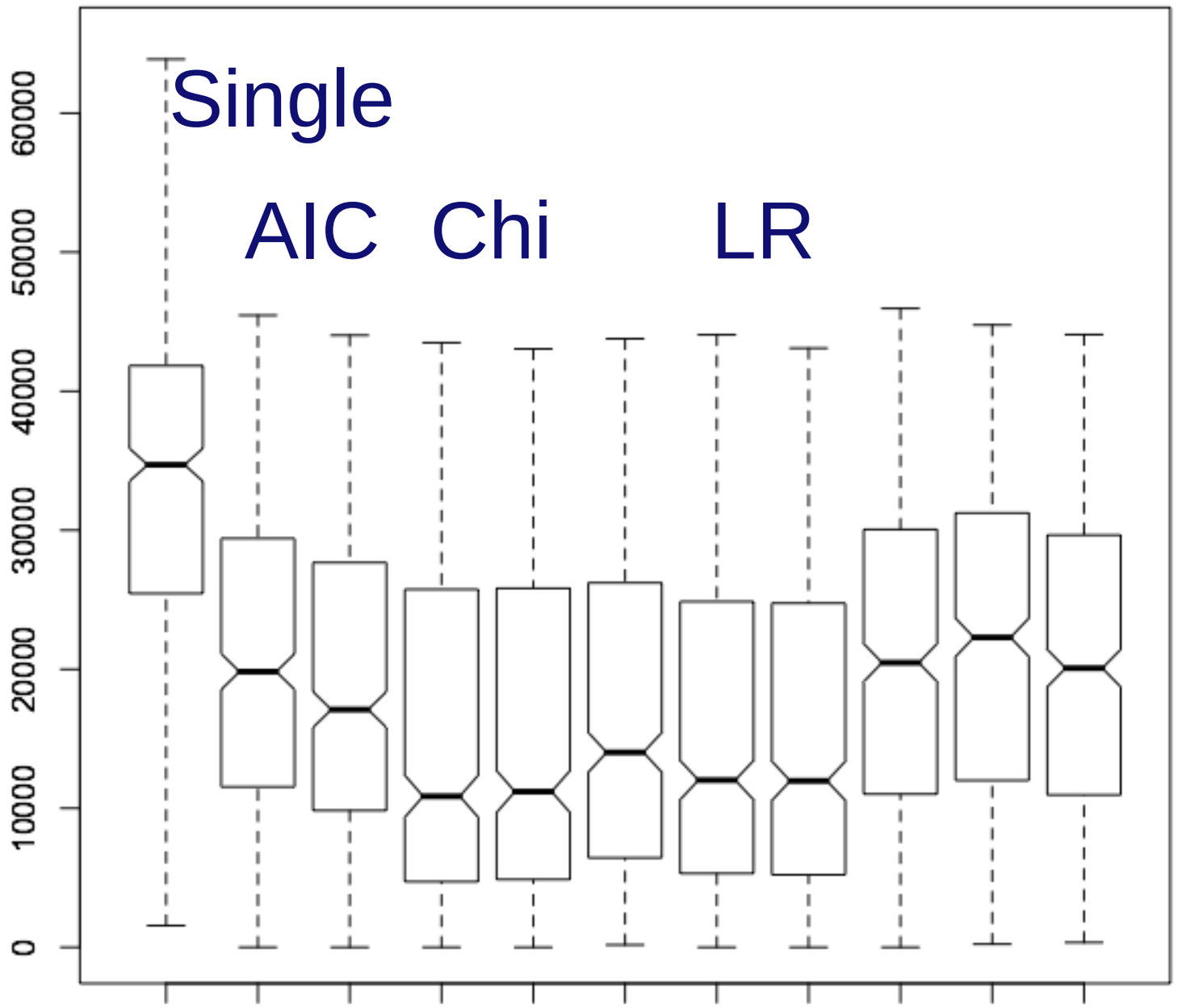


Results 2000 strings

AIC result

-

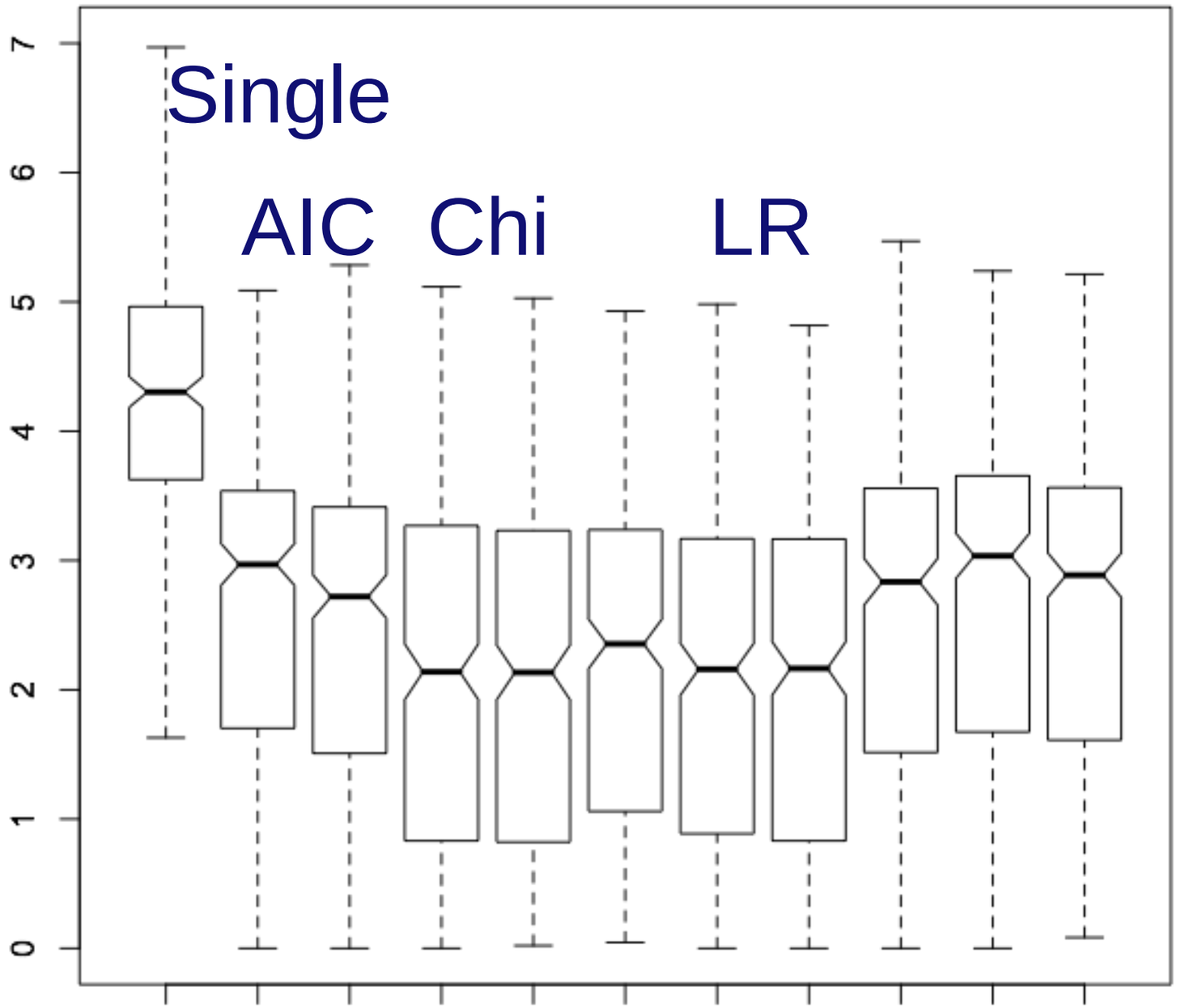
AIC optimal



Results

Alphabet size = 4

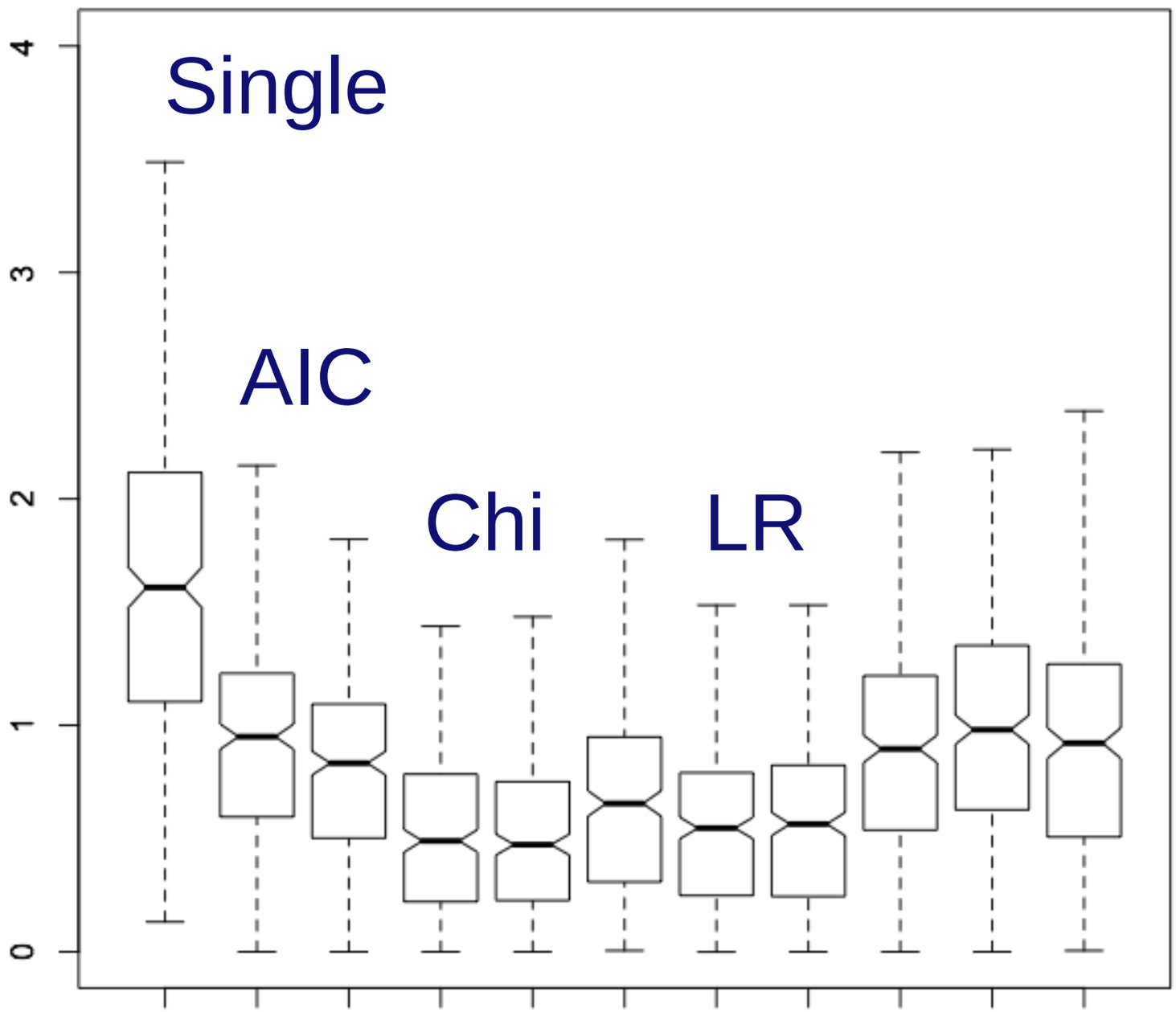
Perplexity result - Perplexity optimal



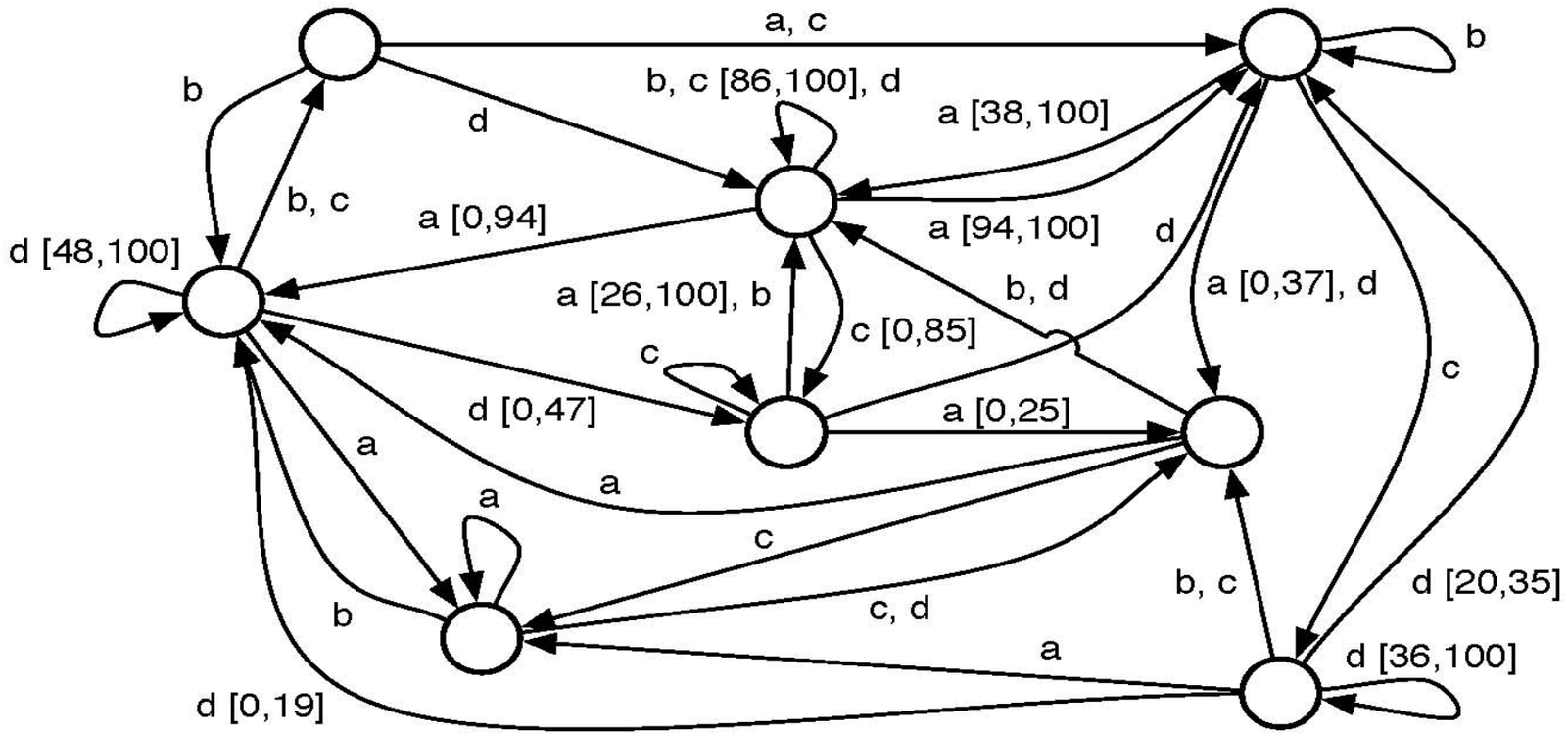
Results

Alphabet size = 2

Perplexity result - Perplexity optimal



Originally generated random DRTA



Identified using RTI+ with the likelihood ratio test