Learning in the limit and process mining

Sicco Verwer

2009

TU Delft

Overview

- What is learning in the limit?
- Some results on learning DFAs
- An algorithm for learning DFAs efficiently in the limit
- Why over-fitting is no issue
- How to compare two learned models
- Learning timed automata models (my thesis)

Learning in the limit

- Learning (or identification) in the limit views learning as an ongoing process:
  - A student get some data
  - The student uses this data to update its hypothesis
  - The student then gets new data, updates again, etc.
- Such a learning process is successful if at some point (in the limit) the student's hypothesis is correct and does not change anymore

Learning in the limit

- The data is assumed to be produced by some unknown process, but from a known class of processes (such as automata, petri-nets, etc.)
- The student's hypothesis is correct if the hypothesis (an automaton, petri-net, etc.) is language equivalent to the process that produced the data
  - also called convergence
- It is assumed that all data will at some point be presented to the student.

Learning by enumeration

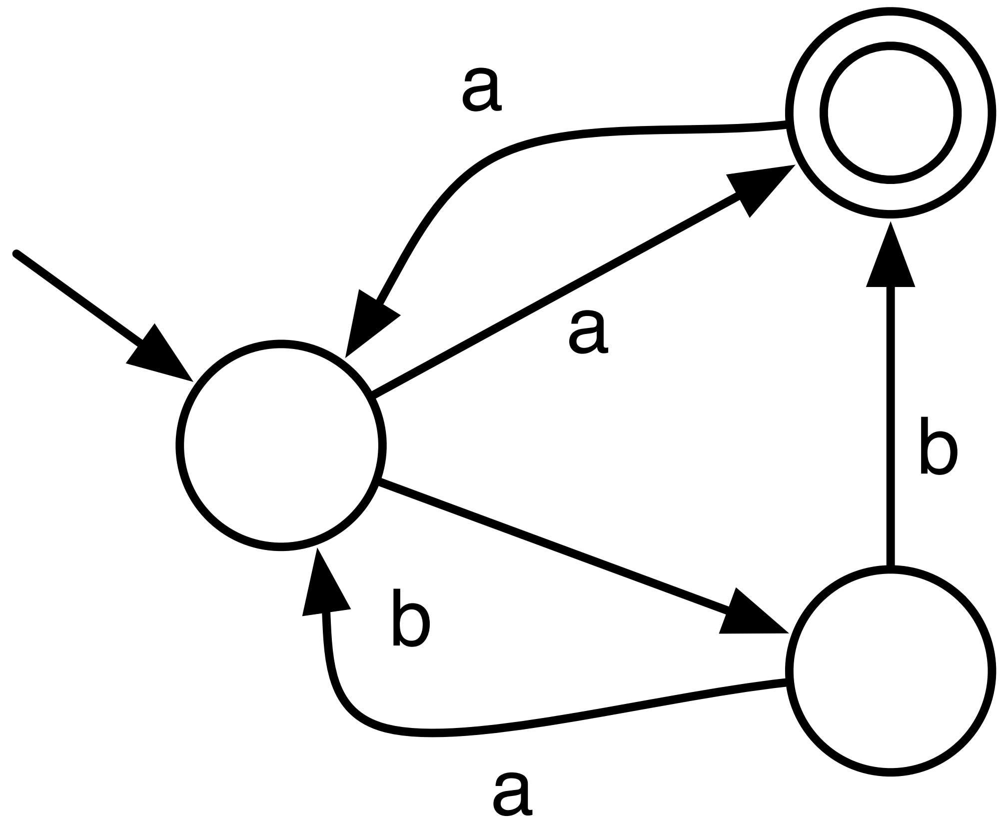
- A simple student for enumerable process classes:
  - Given an enumeration: $H_1, H_2, H_3, \ldots$
  - Assume H_1, until it is inconsistent with the data
  - Then assume the first H_n such that there is no H_m with $m < n$ that is consistent with the data
- This student can be used to learn many process classes in the limit, but not efficiently

TU Delft

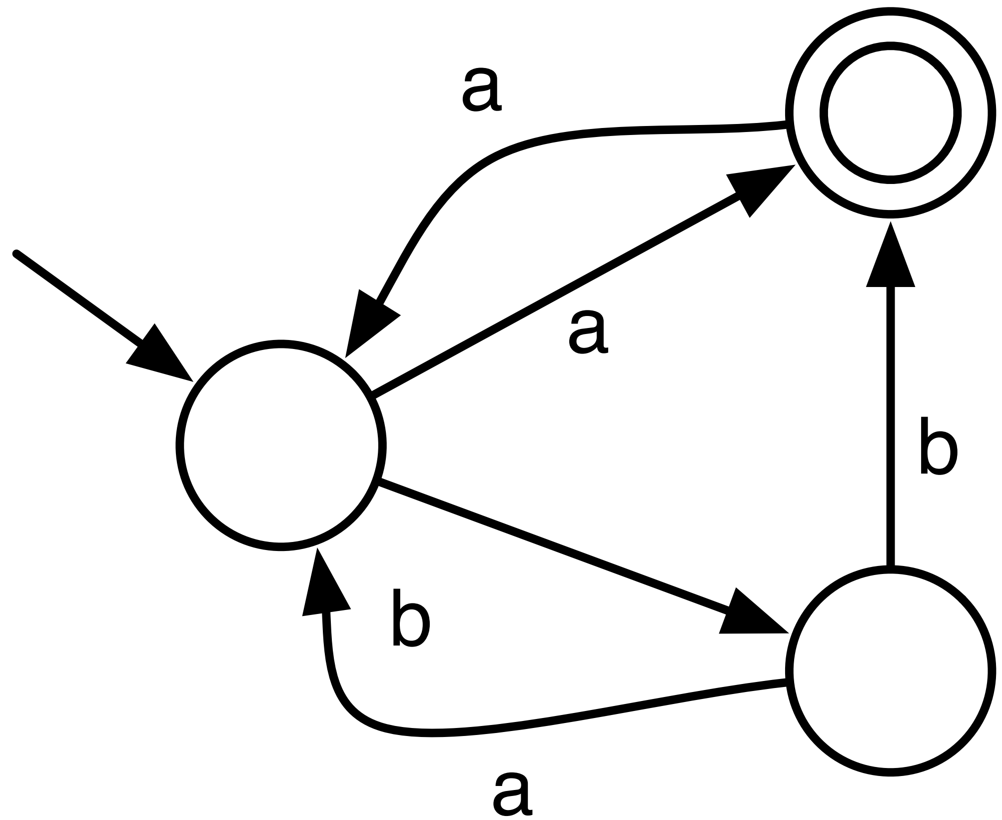
Labeled and unlabeled data

- There is an important difference between labeled and unlabeled data
  - Labeled: contains positive and negative examples
  - Unlabeled: contains only positive examples
- Many process classes are learnable in the limit from labeled, but not from unlabeled data
- For example, DFAs...

DFAs



DFAs



Positive strings: a, aabb, aaaa, aaaabb, bbaa, ...
Negative strings: aa, ab, aab, aaba, aabba, aabbb, ...

TU Delft

Learning DFAs

- The language *L(A)* of a DFA *A* is the set of all positive strings for *A*
- A student learns the class of DFAs in the limit if:
  - Assuming that the data is produced by some DFA *A*
  - The student converges in the limit on a hypothesis *H* such that *L(H)* = *L(A)*

Learning DFAs

- Learning by enumeration learns DFAs in the limit from labeled data
    - For example using Occam's razor (smallest DFA first)

Learning DFAs

- Learning by enumeration does not learn DFAs in the limit from unlabeled data:
  - We require a sequence of examples such that the student converges to the correct DFA
  - It is impossible to find such sequences for every DFA:
    - every such sequence is a finite DFA language
    - they are sublanguages of infinite DFA languages

Learning DFAs

- Input:
  - Labeled data
- Goal:
  - Find the smallest DFA that is consistent with the data
- NP-hard by reduction from Satisfiability

Learning DFAs

- DFAs cannot be learned from unlabeled data, and it is very difficult to learn them from labeled data
- This is slightly misleading:
  - In the limit more and more data becomes available, at some point the labeled data will no longer encode an NP-hard problem
  - Under statistical assumptions the unlabeled data can be used to simulate labels
- It has been shown that DFAs can be learned efficiently, even from unlabeled data!

Overview

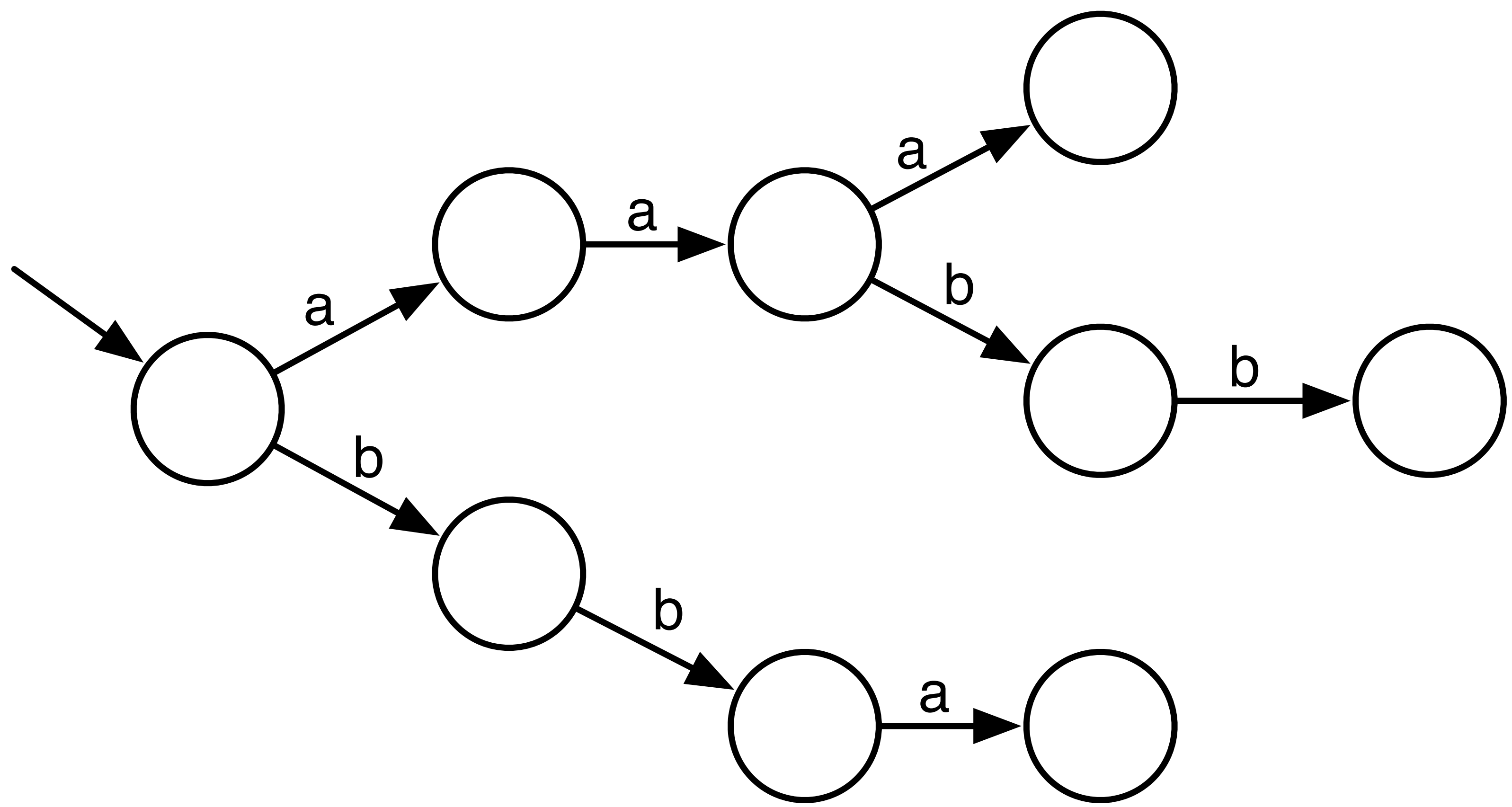
- What is learning in the limit?
- Some results on learning DFAs
- An algorithm for learning DFAs efficiently in the limit
- Why over-fitting is no issue
- How to compare two learned models
- Learning timed automata models (my thesis)

TU Delft

Efficient Identification

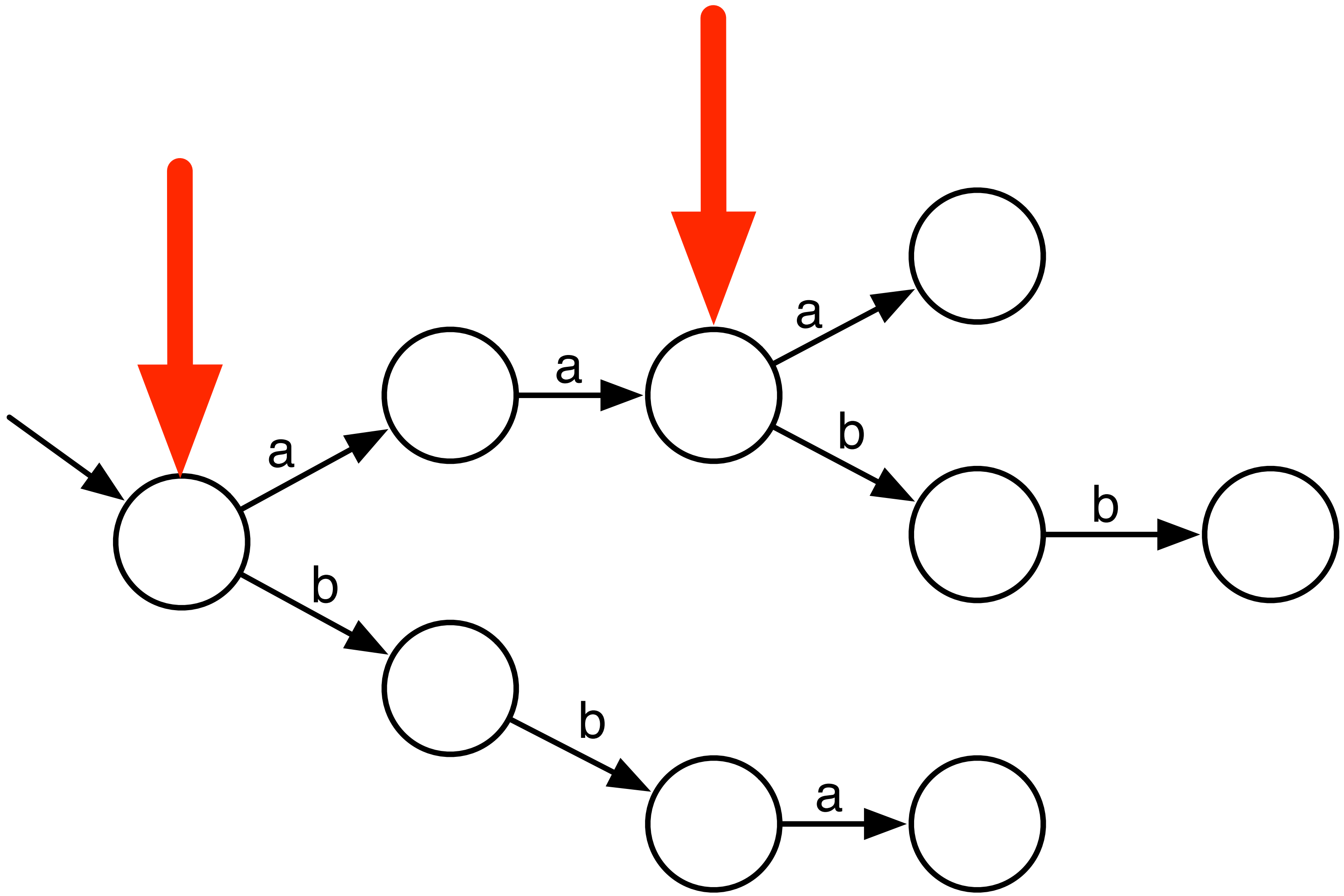
- A process class C is efficiently identifiable in the limit if:
  - there exists a polynomial time algorithm that can identify any language L from C
  - this algorithm is guaranteed to identify the correct language L_t when the input data contains a polynomial characteristic set:
    - a subset of size polynomial in the size of the smallest representation (automaton) A such that $L(A) = L_t$

Learning DFAs



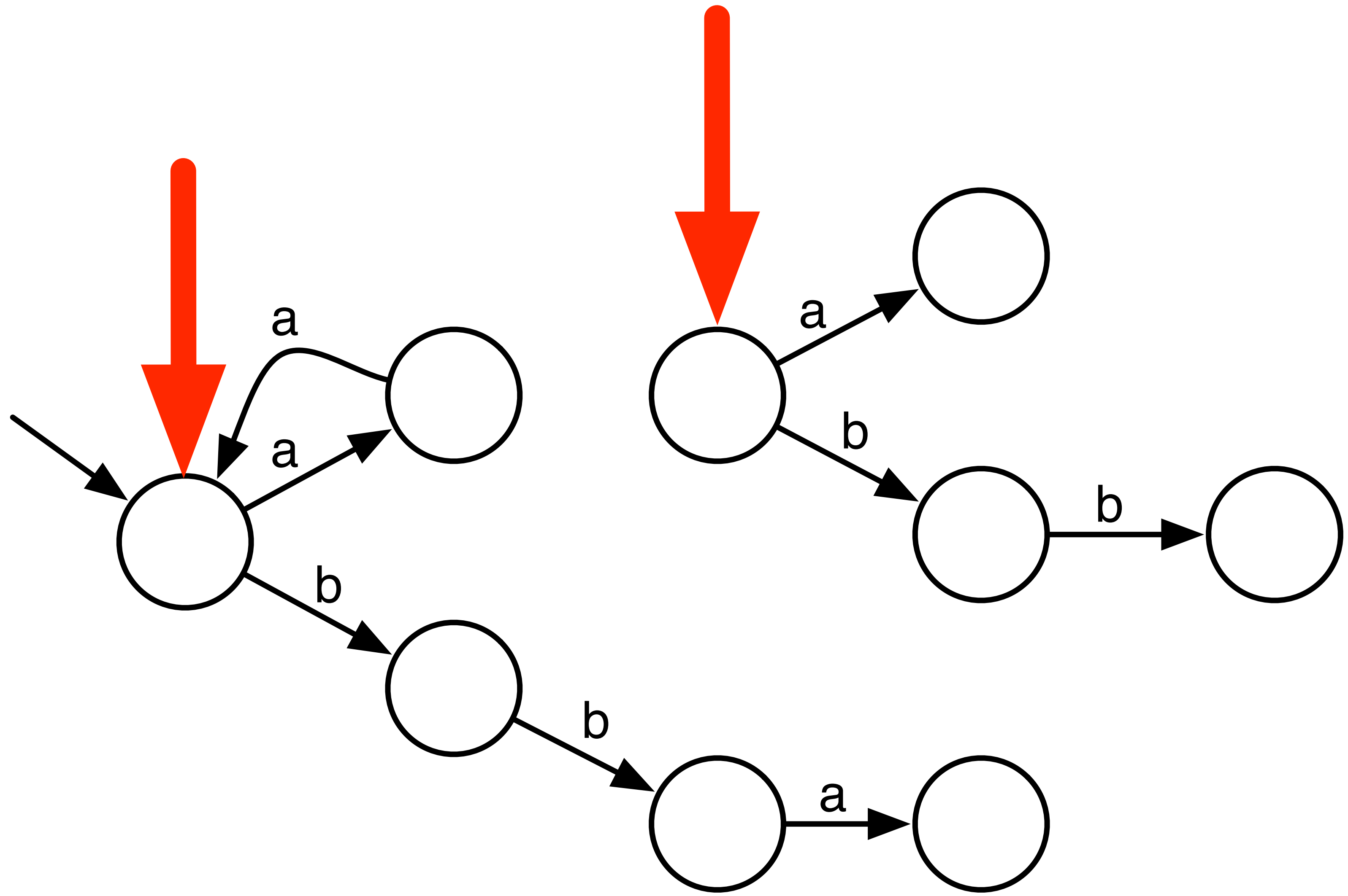
Some observations: a, aaa, aaa, aabb, b, bb, bb, bba
represented as a prefix tree

Learning DFAs



State merging:
select two nodes

Learning DFAs

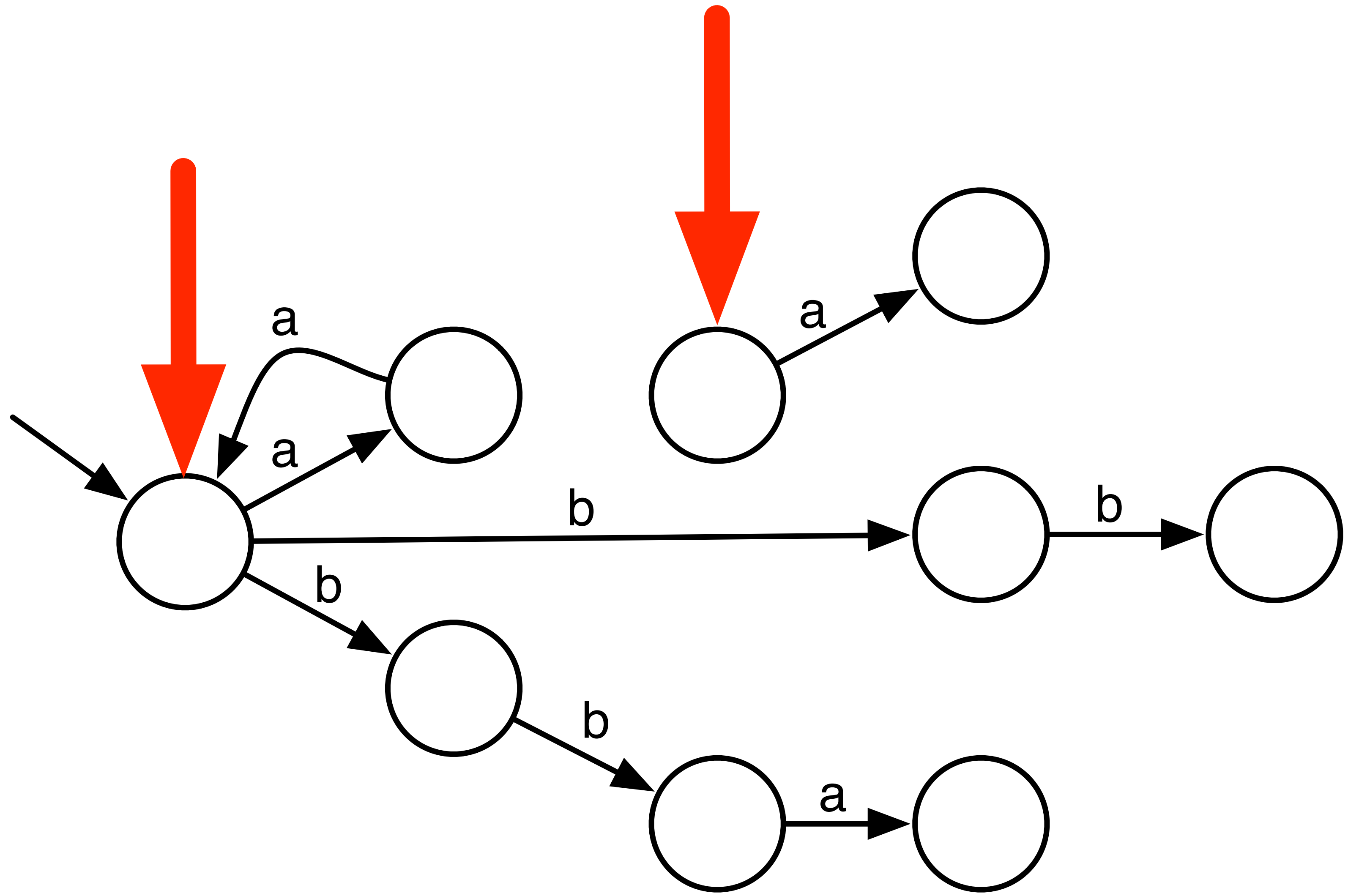


State merging:

move input transitions from one state to the other

TU Delft

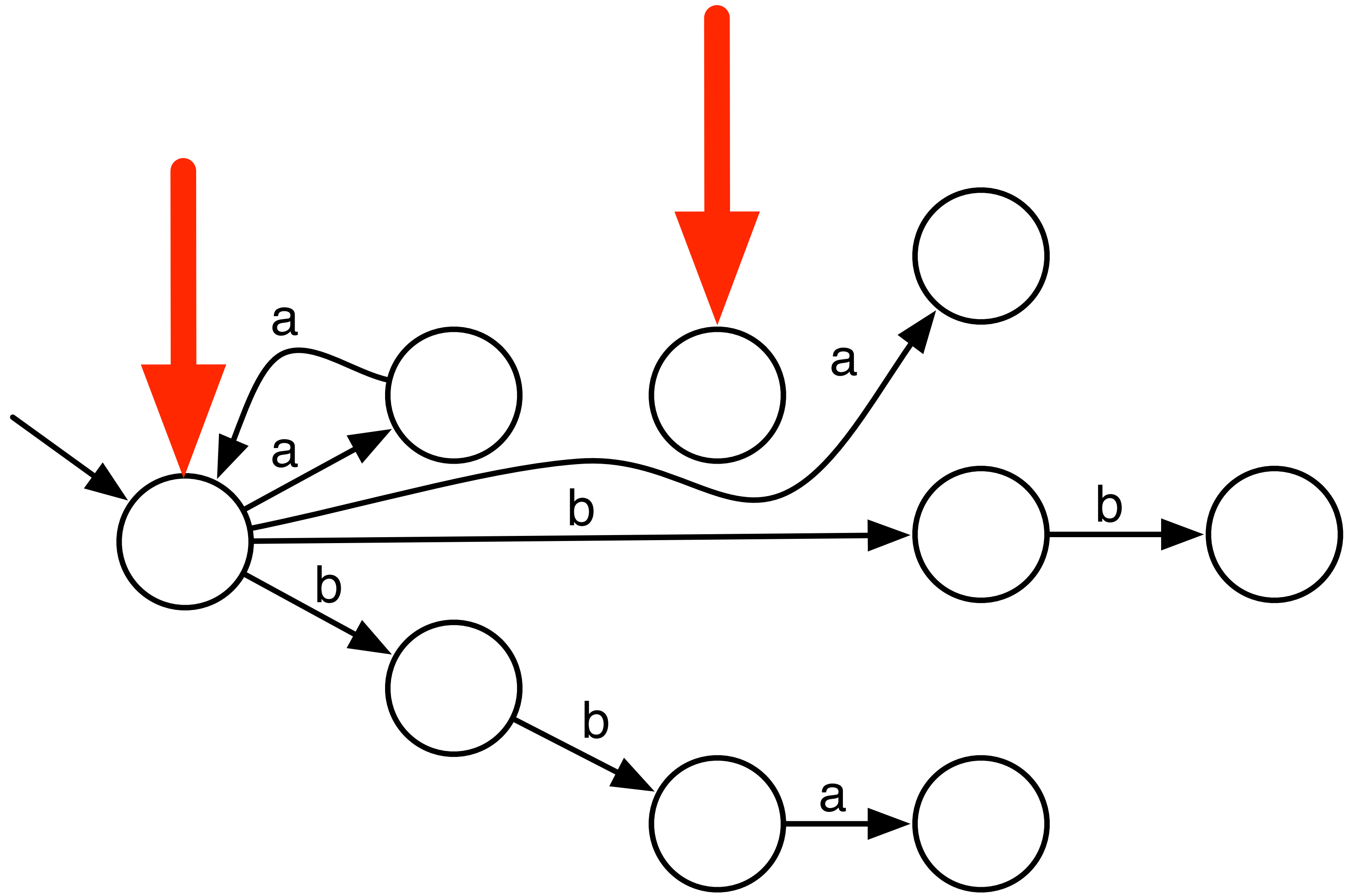
Learning DFAs



State merging:

move output transitions from one state to the other

Learning DFAs

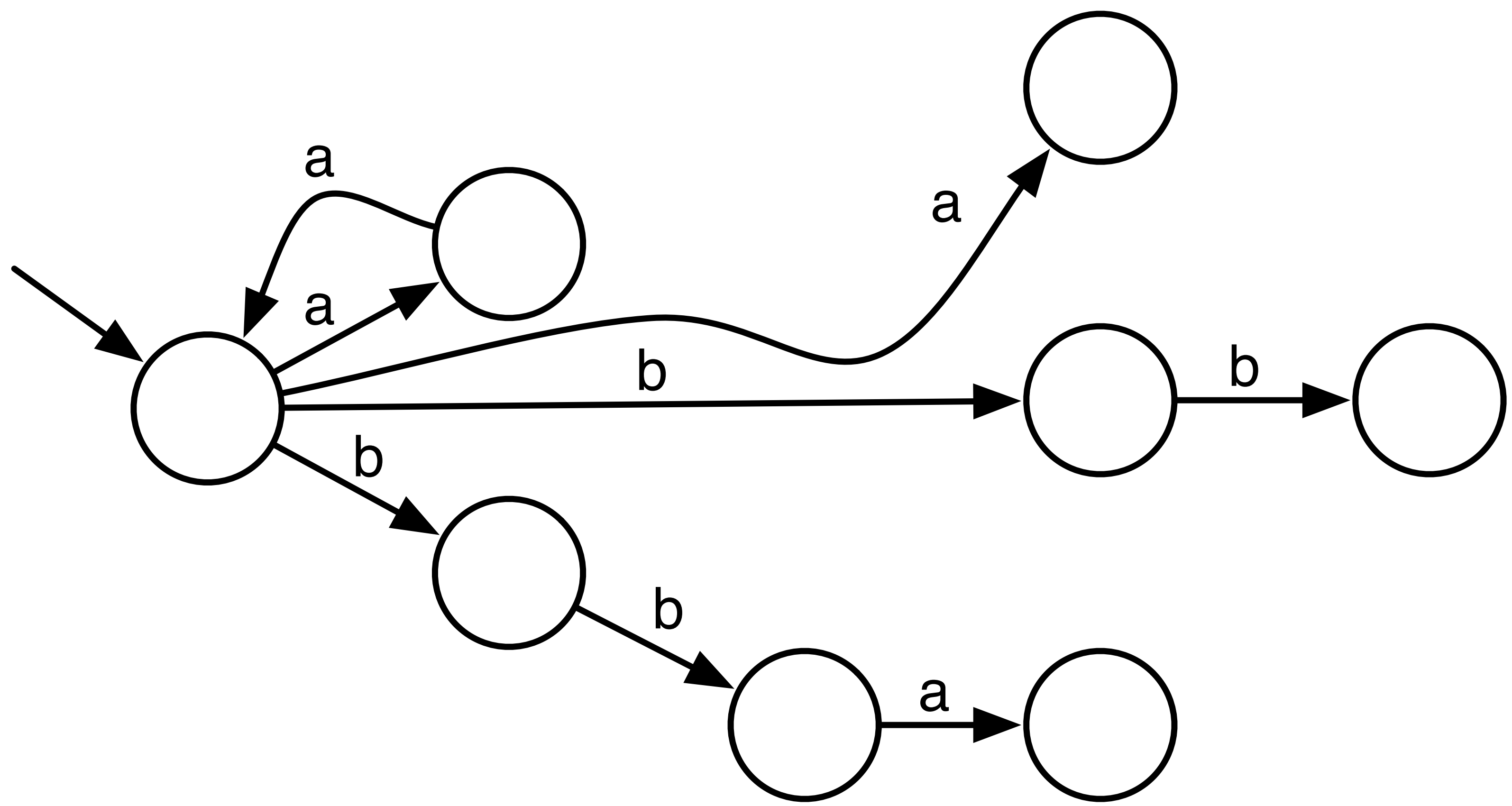


State merging:

move output transitions from one state to the other

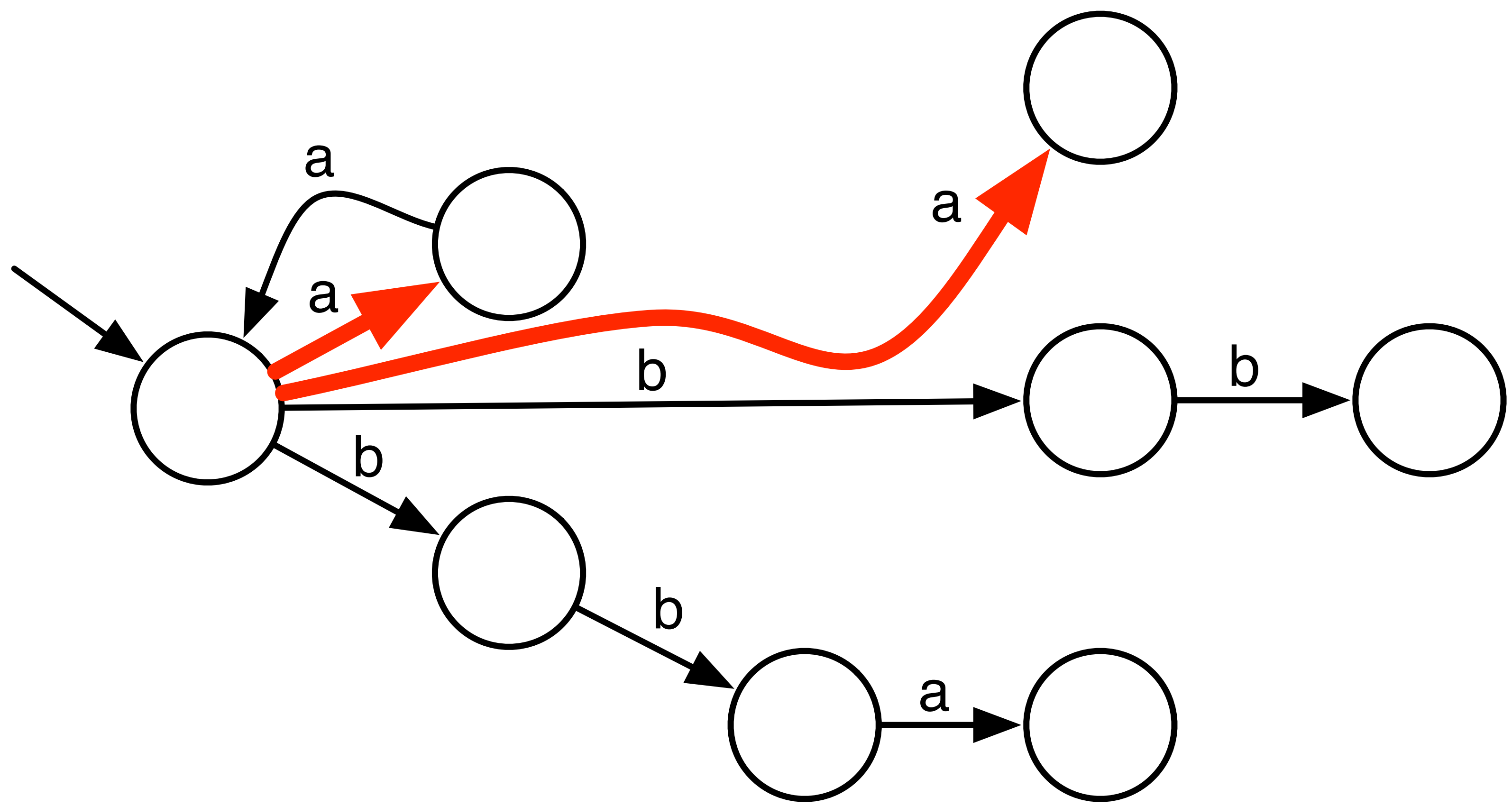
TU Delft

Learning DFAs



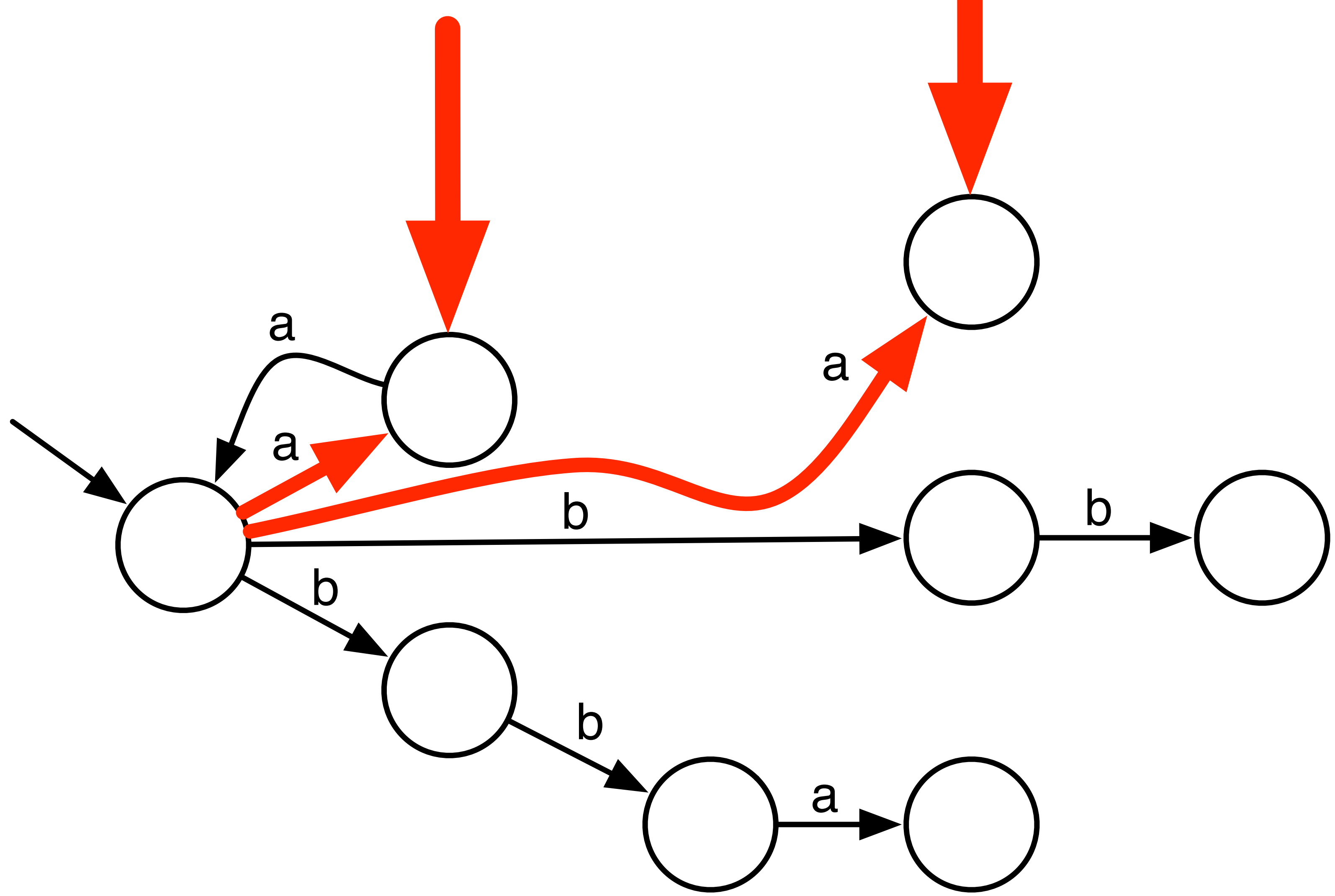
State merging:
delete the obsolete state

Learning DFAs



Determinization:
merge the targets of non-deterministic transitions

Learning DFAs

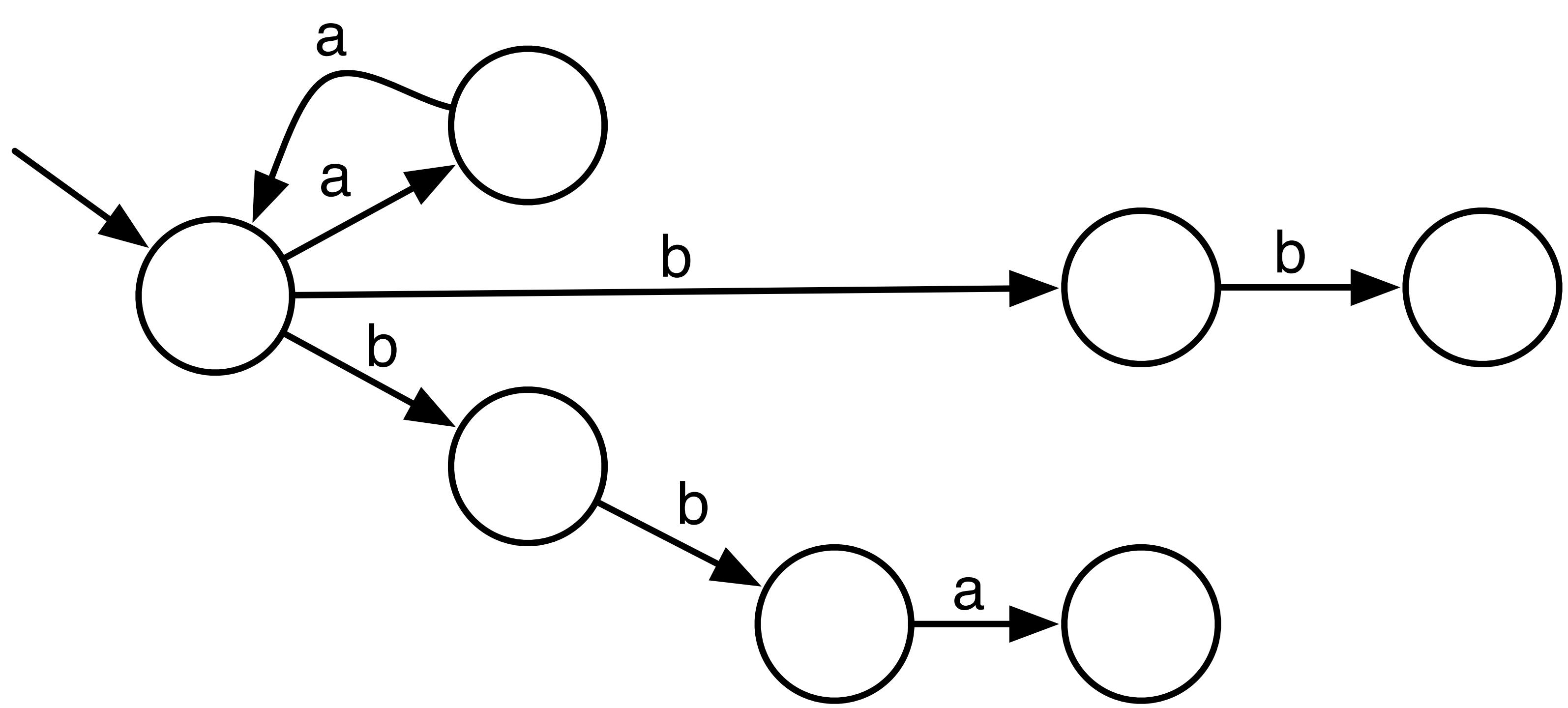


Determinization:

merge the targets of non-deterministic transitions

TU Delft

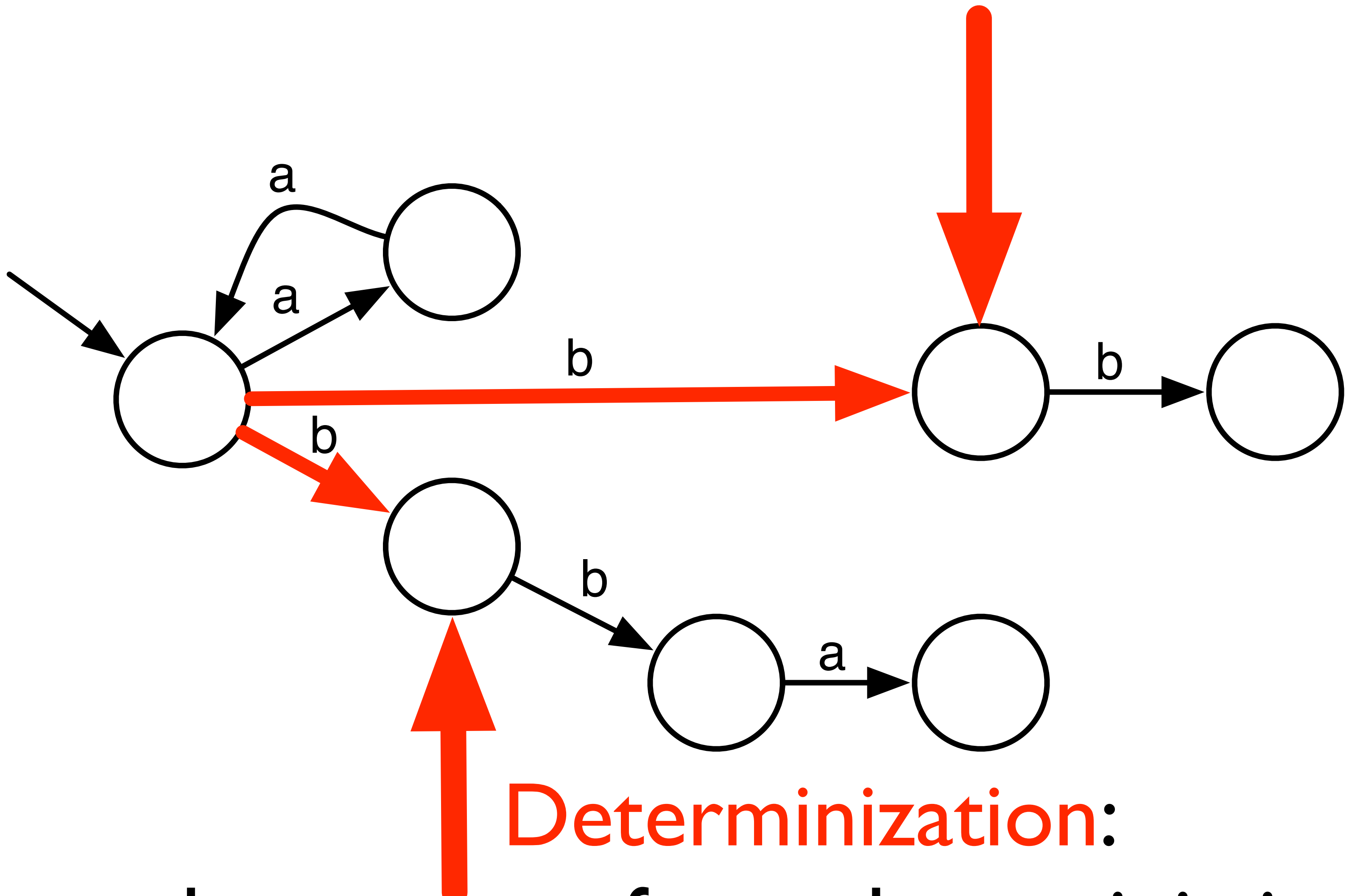
Learning DFAs



Determinization:

merge the targets of non-deterministic transitions

Learning DFAs

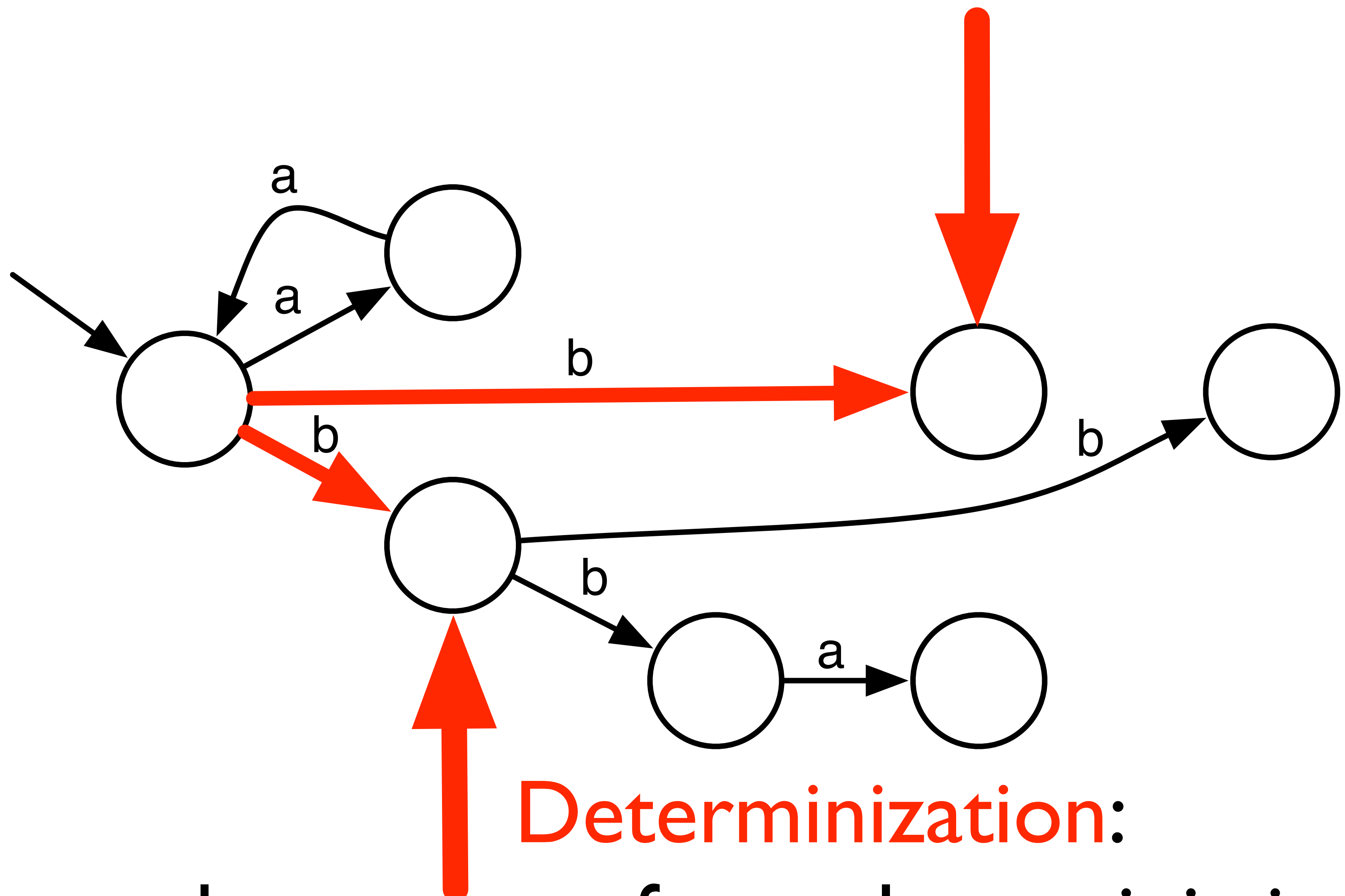


Determinization:

merge the targets of non-deterministic transitions

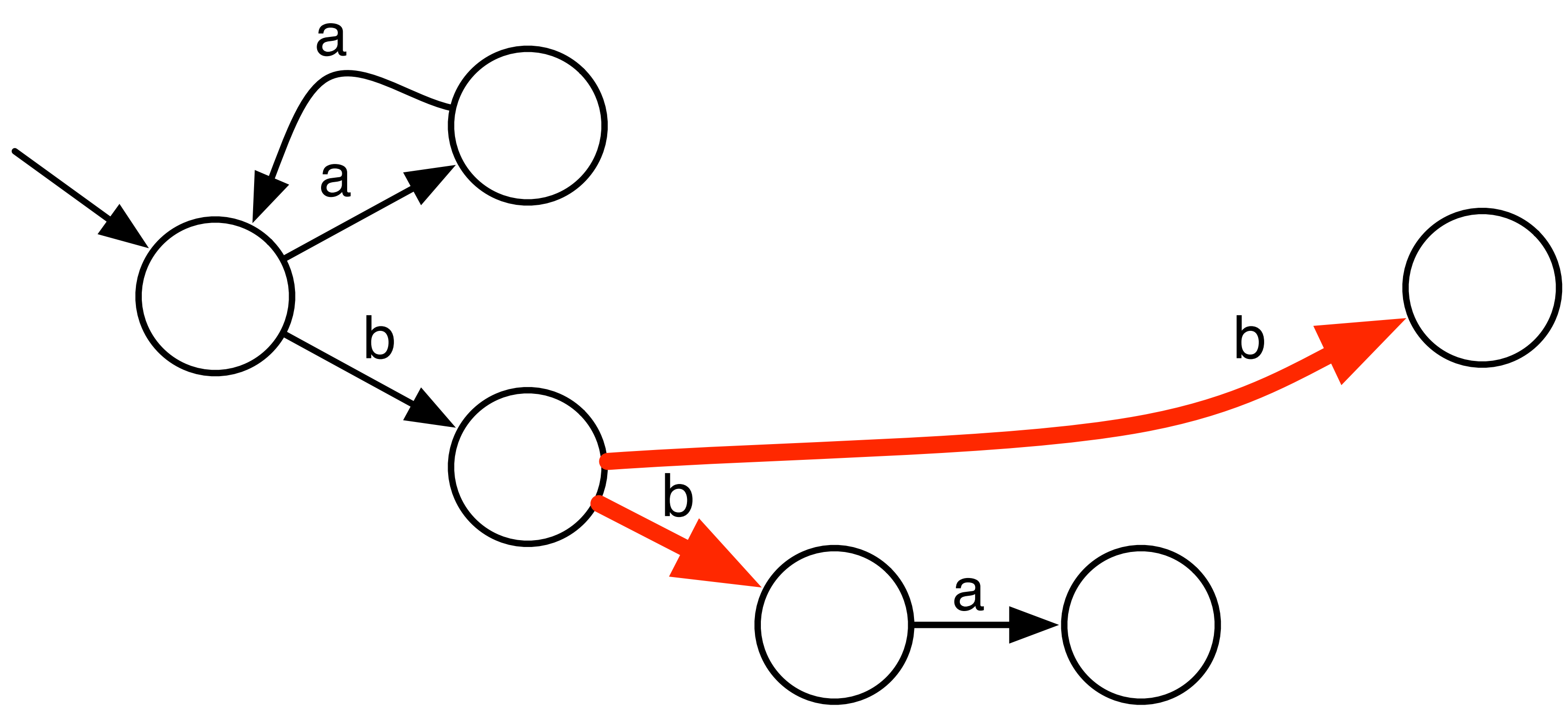
TU Delft

Learning DFAs



Determinization:
merge the targets of non-deterministic transitions

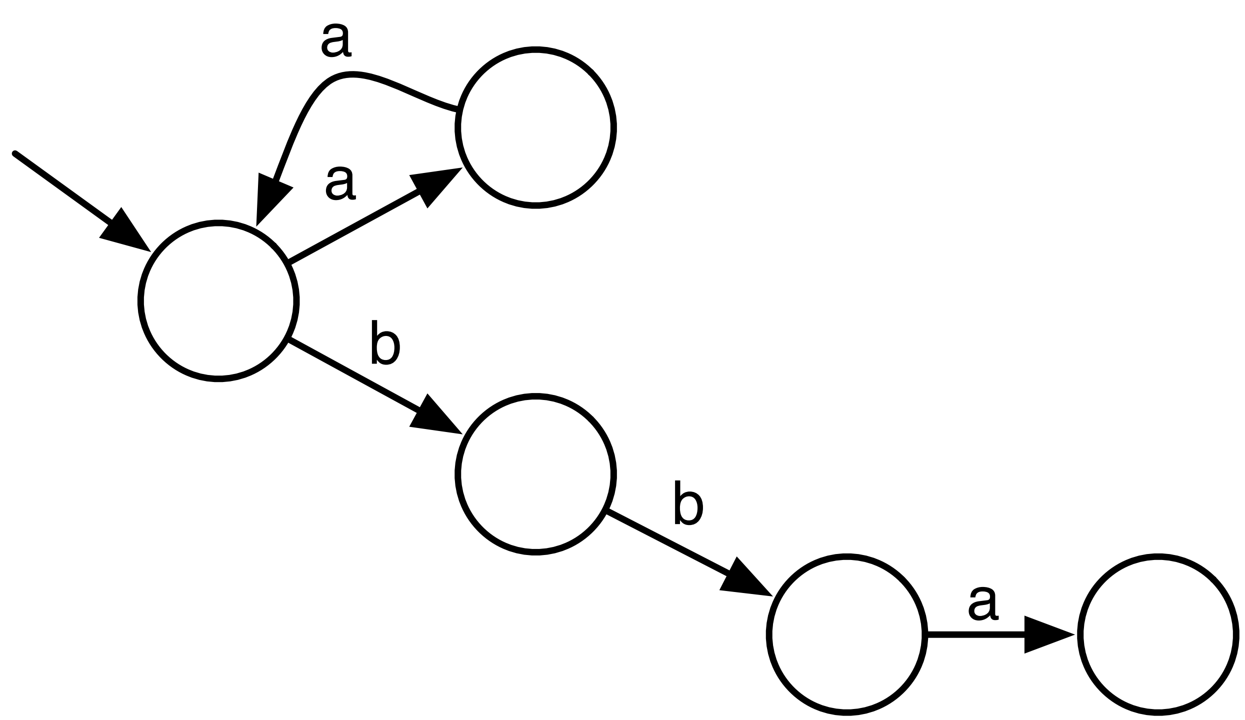
Learning DFAs



Determinization:

merge the targets of non-deterministic transitions

Learning DFAs

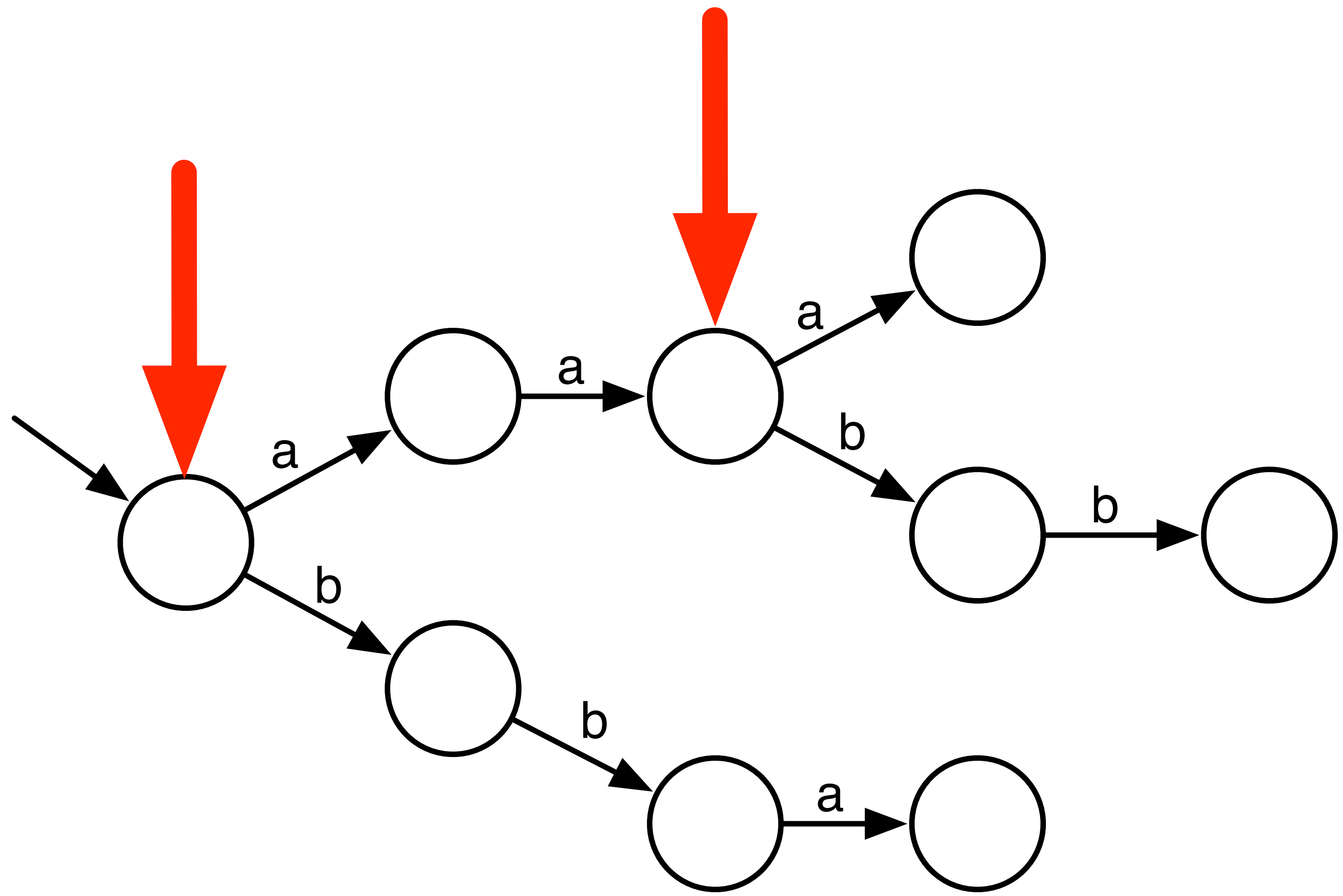


Select two new nodes to merge and iterate

Learning DFAs

- State merging:
  - Start from a tree
  - Try all possible merges, including determinization
  - Perform the one that scores best
  - Iterate
- Use a search procedure to find the smallest DFA:
  - Backtrack, beam search, best-first search, iterative deepening, ...

Learning DFAs



The score of a merge can be determined using labels or statistics

EDSM

- For every string *s* it is known whether s is an element of the language or not, it is positive or negative
- Evidence driven state merging (EDSM):
  - Initially, the states *q* of the prefix tree are labeled according to positive/negative strings that end in *q*
  - It is not possible to merge positively labeled states with negatively labeled states
  - Score = #positive merges + # negative merges

TU Delft

ALERGIA

- It is not known whether strings are in the language or not
- 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

Algorithms for learning DFA

- State merging using EDSM performs best for labeled data
- The idea in ALERGIA has been used in many other algorithms, also in approximating DFA distributions
- Under natural assumptions, both algorithms converge in the limit to the correct DFA
- For both algorithms it is possible to compute the amount of data necessary to converge with sufficient probability

TU Delft

Overview

- What is learning in the limit?
- Some results on learning DFAs
- An algorithm for learning DFAs efficiently in the limit
- Why over-fitting and under-fitting are no issue
- How to compare two learned models
- Learning timed automata models (my thesis)

TUDelft

Over- and under-fitting

- The state-merging algorithm does not over- or under-fit,
  - it converges efficiently to the correct DFA
- When it does not produce the correct DFA,
  - if this DFA is too small, there is too little data because there is a smaller consistent DFA
  - if this DFA is too big, there is too little data because finding the correct DFA is difficult

Comparing models

- When data is labeled:
  - use accuracy, precision, recall, or any well-known measure from machine learning

Comparing models

- When data is unlabeled, we learn using statistics, these can also be used to compare models:
  - Determine the likelihood of the data given the model
  - This model has to be probabilistic
  - Compute the Perplexity, Akaike Information Criterium, Minimum Description Length, or any model-selection criterium
- These measures are minimal if the model is equivalent to the model that generated the data

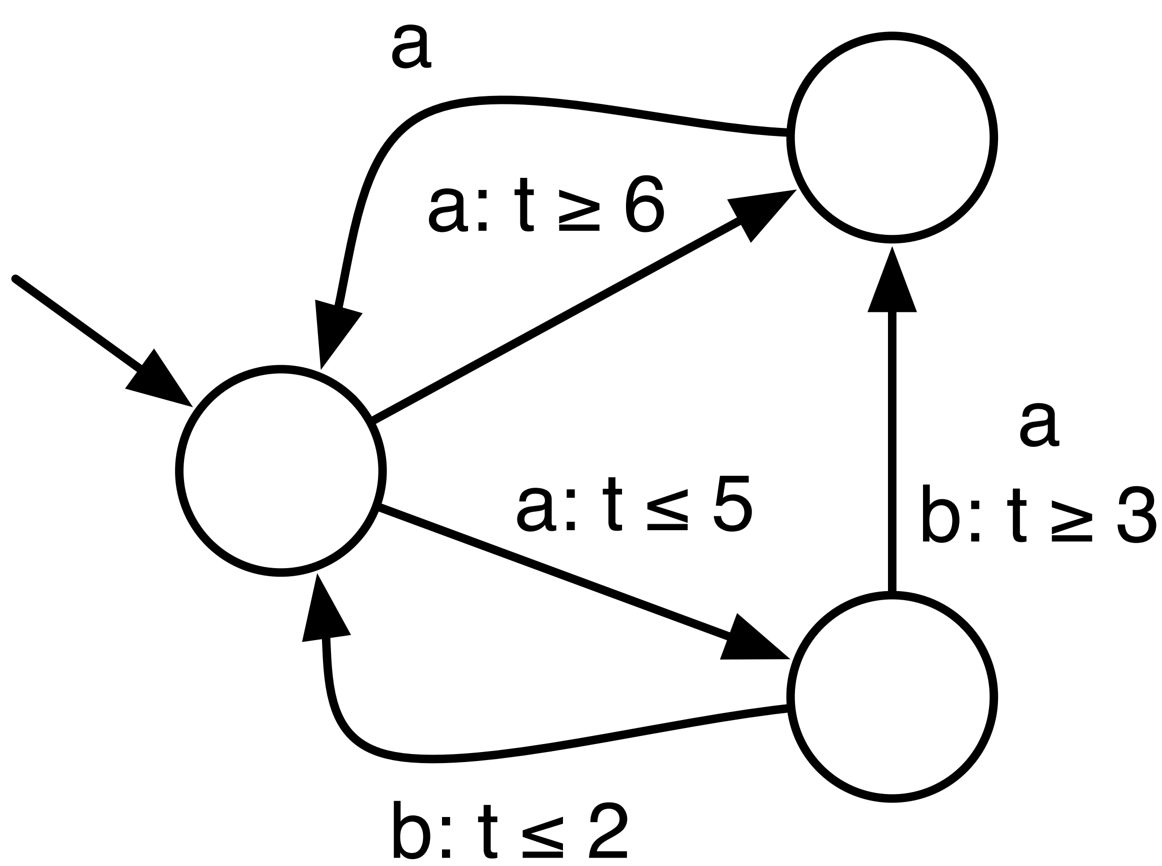
TU Delft

Overview

- What is learning in the limit?
- Some results on learning DFAs
- An algorithm for learning DFAs efficiently in the limit
- Why over-fitting is no issue
- How to compare two learned models
- Learning timed automata models (my thesis)

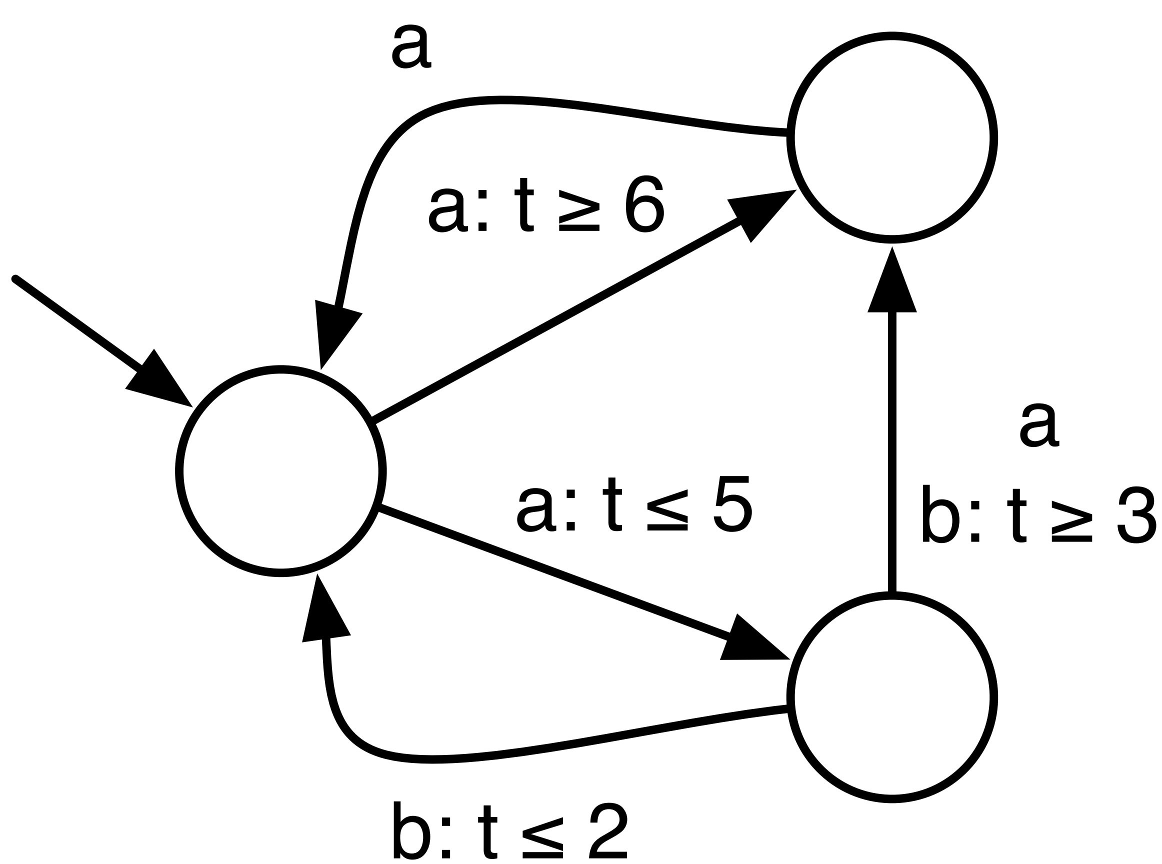
TU Delft

Real-time automata (DRTAs)



Transitions contain guards on time values

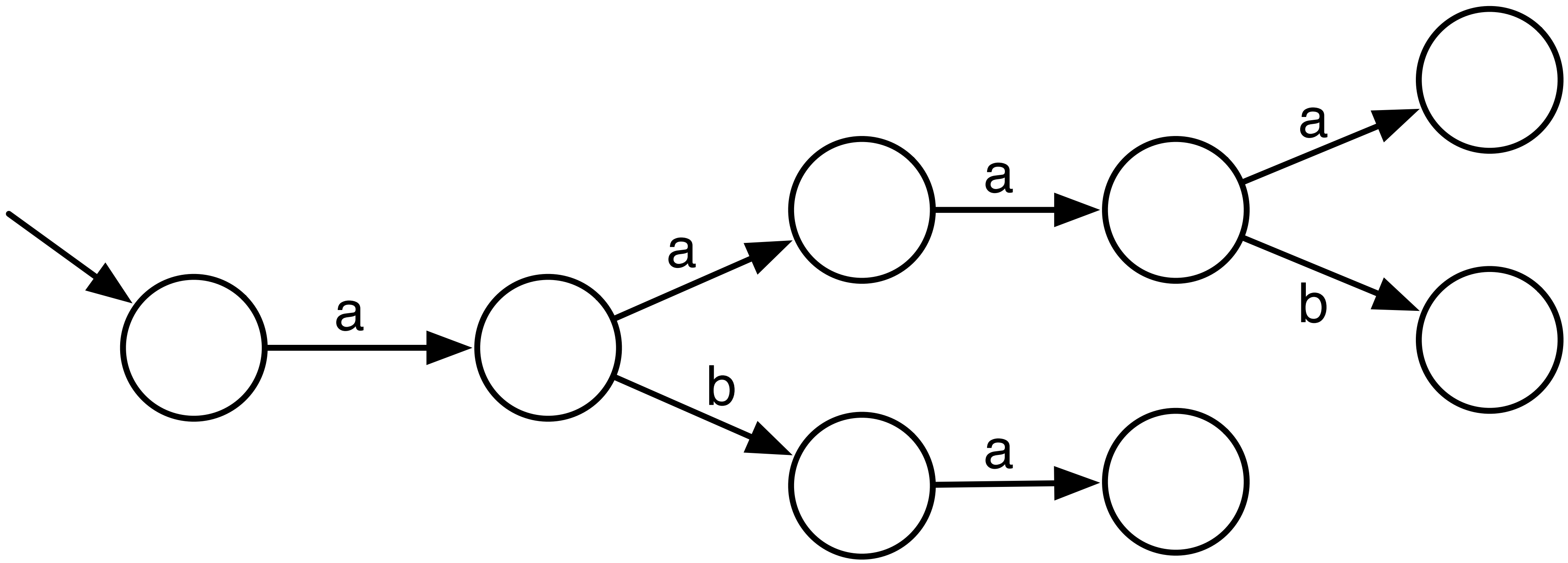
DRTAs and events



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, ∞)

Why learn DRTAs?

- DRTAs:
  - Use an explicit time representation (using numbers)
  - Are intuitive models for many real-time systems
  - Are used to model and verify reactive systems

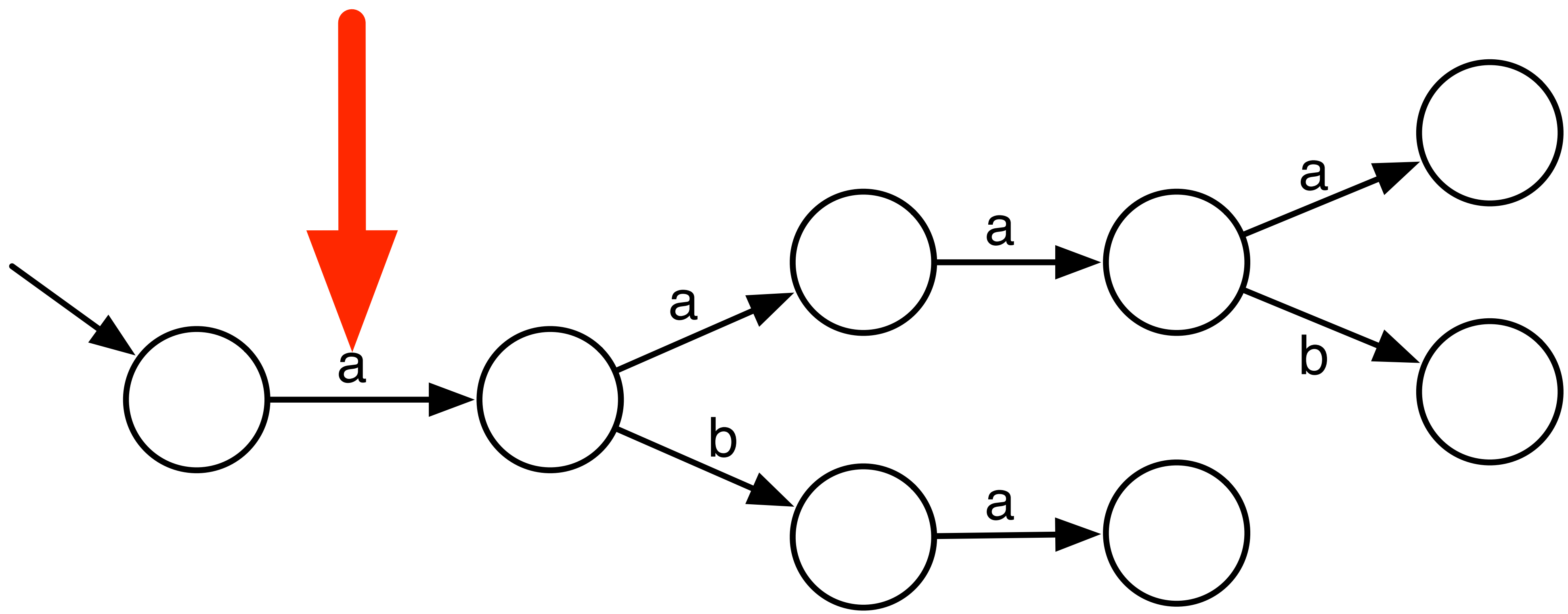
Why learn DRTAs?

- 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
- I have shown that it possible to learn some (but not all) DRTAs efficiently!
  - (see my homepage)

Applications

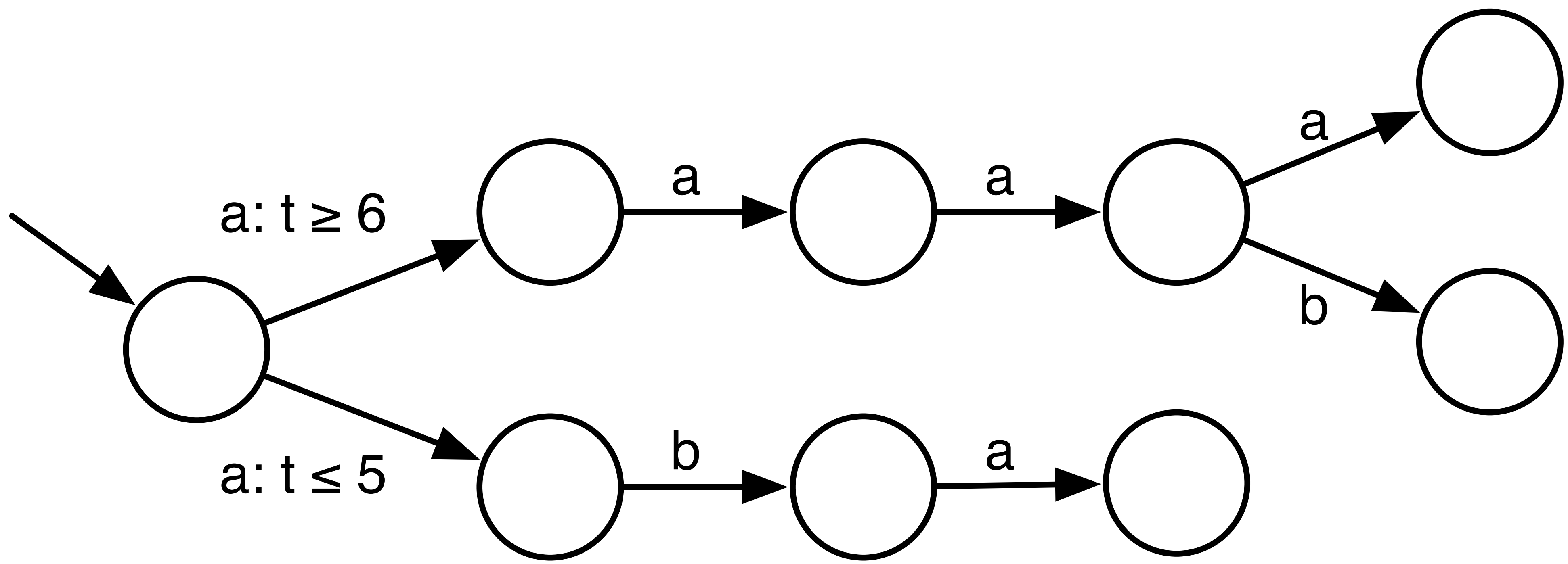
- Learning truck driver behavior
- Inferring models for ship movement
- Testing black-box real-time systems
- Identifying process models(?)
- ...
  - Anywhere where representing time explicitly results in a large reduction in model size

Learning DRTAs



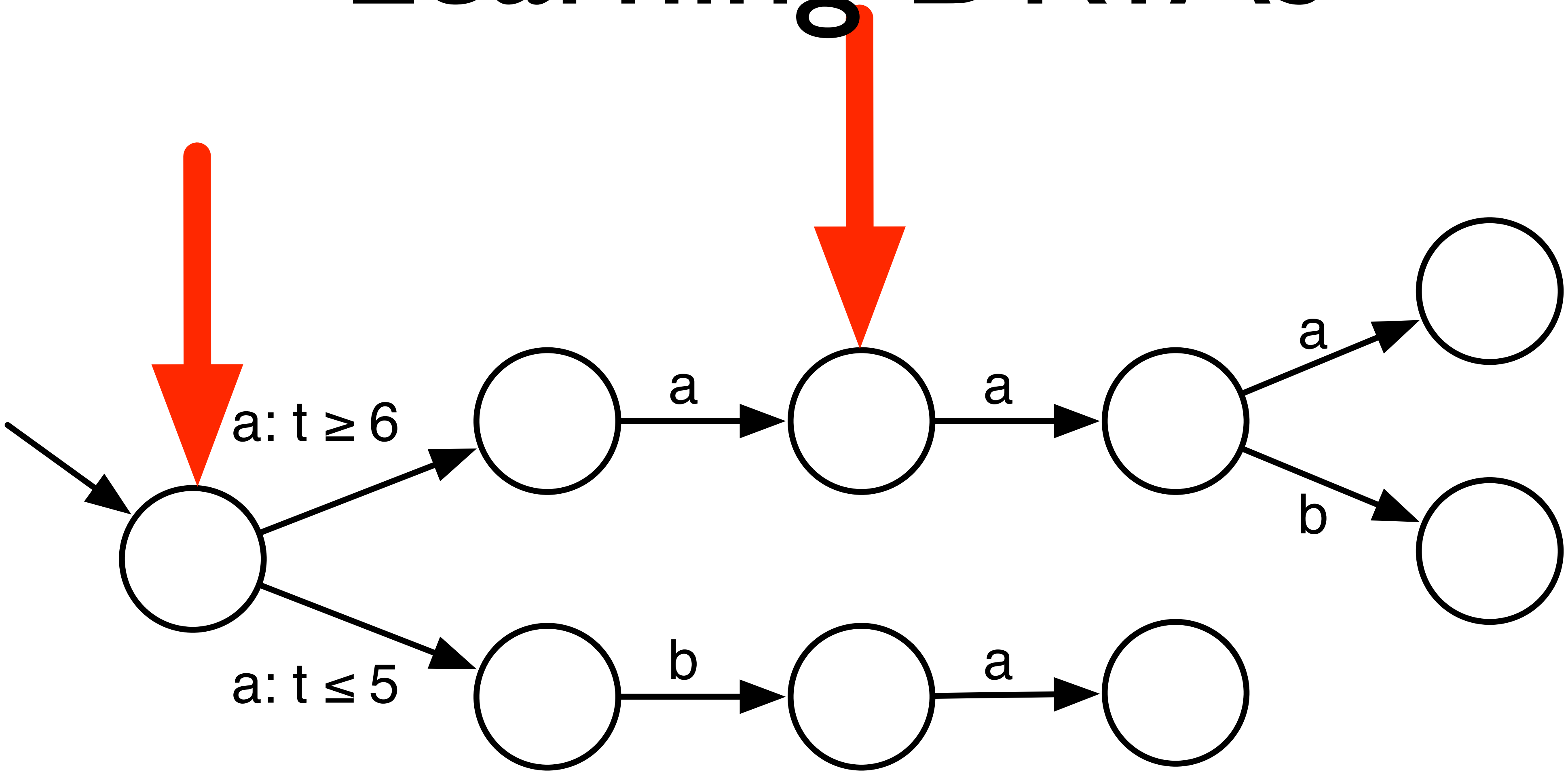
Transition splitting:
choose a transition

Learning DRTAs



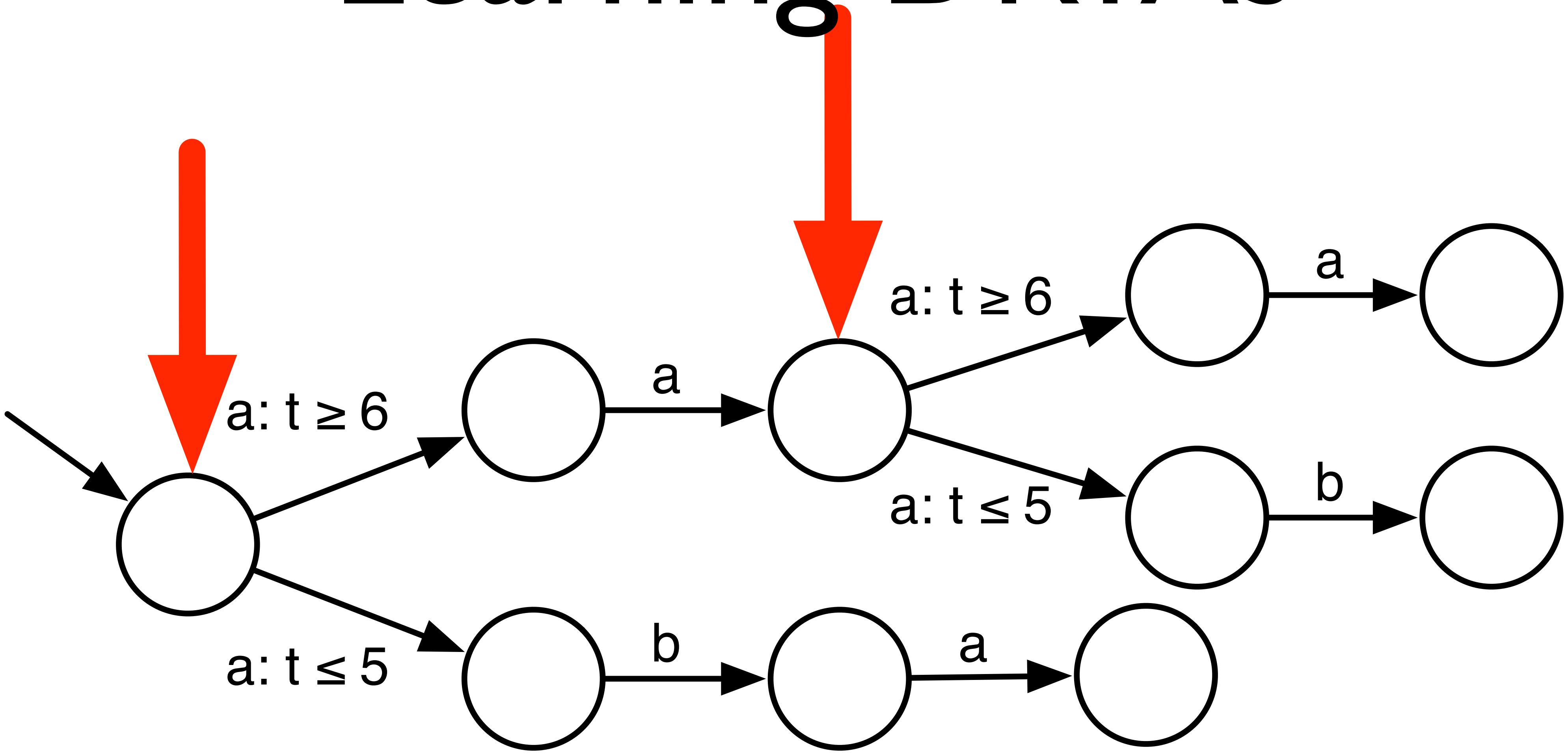
Split the transition and recalculate the subsequent part of the prefix tree

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

Learning DRTAs

- The algorithm converges efficiently to the correct DRTA
- The algorithm works both on labeled and unlabeled data
- In experiments on artificial data it was capable of learning DRTAs with 8 states, 16 guards, and an alphabet of size 4 from a data-set of 2000 examples with an average length of 20

Contact

- http://www.st.ewi.tudelft.nl/~sicco/
- s.e.verwer@tudelft.nl

- DFA learning references, see:
  - Colin de la Higuera, A bibliographical study of grammatical inference, Pattern Recognition, Volume 38, Issue 9, Pages 1332 1348
- DRTA learning references, see my homepage