# Imre Simon Test-of-Time Award 2020

*Regular expressions into finite automata* by Anne Brüggemann-Klein

LATIN 1992, LNCS 583, 87—98, 1992.

## Context

**Document grammars** in SGML and XML are built from **deterministic** regular expressions.

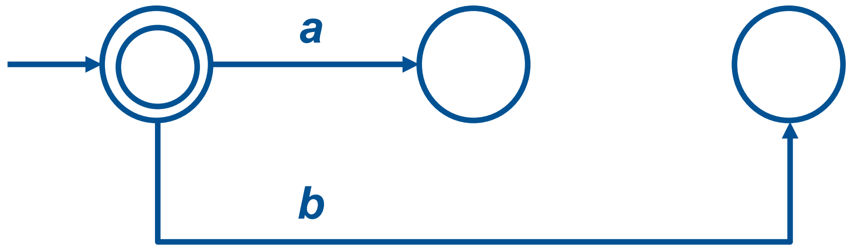
Determinism is defined **operationally**:

> **A [symbol] that occurs in the [word] of a [regular expression's language] must be able to satisfy only one occurrence of that symbol in the [expression] without looking ahead in the [word].**

Condition can be formalised with the help of the **position automaton** (Sakarovitch) that was originally discovered by Glushkov in 1961 – an NFA that naturally represents a regular expression.

Expression $\boldsymbol{E} = (a^* \, b^*)$

Position automaton $\mathbf{M}_E$



## Context

Document grammars in SGML and XML are built from deterministic regular expressions.

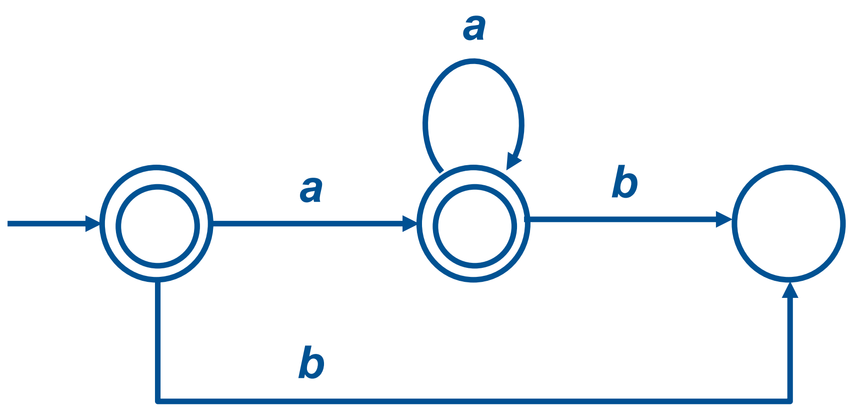
Determinism is defined operationally:

> **A [symbol] that occurs in the [word] of a [regular expression's language] must be able to satisfy only one occurrence of that symbol in the [expression] without looking ahead in the [word].**

Condition can be formalised with the help of position automata (Sakarovitch) that were originally discovered by Glushkov in 1961 – an NFA that naturally represents a regular expression.

Expression $\boldsymbol{E}$ = (a* b*)

Position automaton $\mathbf{M}_E$



## Context

Document grammars in SGML and XML are built from deterministic regular expressions.

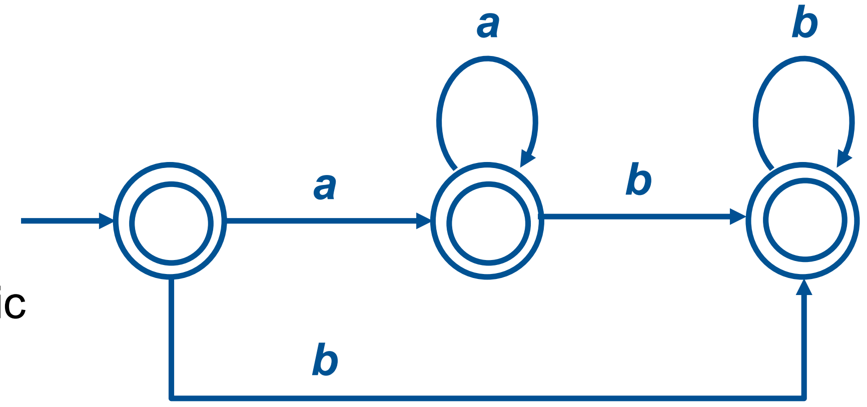
Determinism is defined operationally:

> **A [symbol] that occurs in the [word] of a [regular expression's language] must be able to satisfy only one occurrence of that symbol in the [expression] without looking ahead in the [word].**

Condition can be formalised with the help of position automata (Sakarovitch) that were originally discovered by Glushkov in 1961 – an NFA that naturally represents a regular expression.

Expression ***E*** = (a* b*)

Position automaton $\mathbf{M}_E$

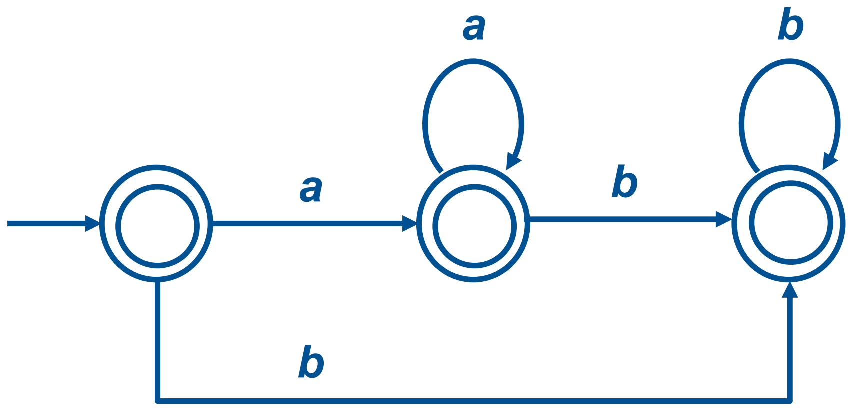


**Definition**
***E*** deterministic ⇔ $\mathbf{M}_E$ deterministic

## Construction time for position automaton

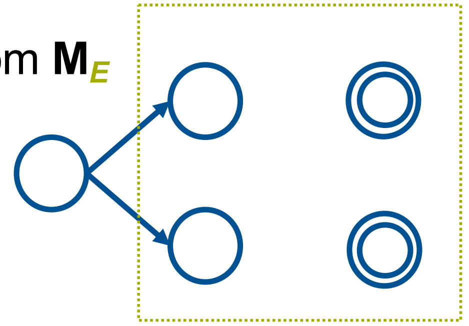
Expression ***E*** = (a* b*)

Position automaton $\mathbf{M}_E$

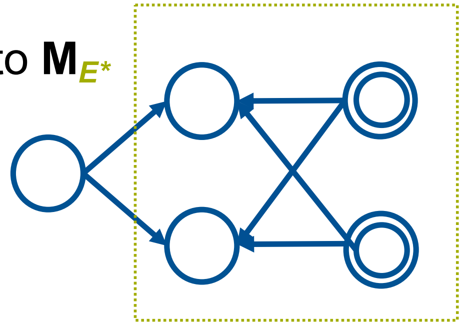


**Goal output-sensitive construction** (touch each transition once)

**Problem** going from $\mathbf{M}_E$



to $\mathbf{M}_{E^*}$

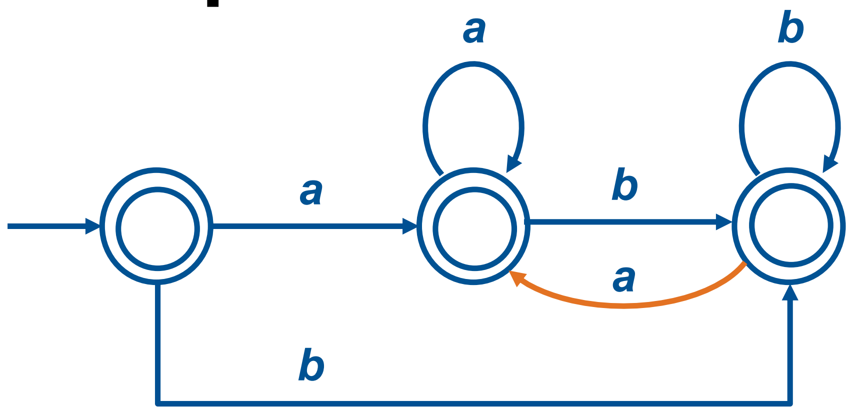


means to add **feedback transitions** that might be already present

# Construction time for position automaton

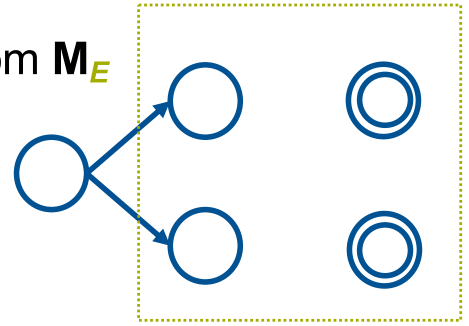
Expression $E^* = (a^* b^*)^*$

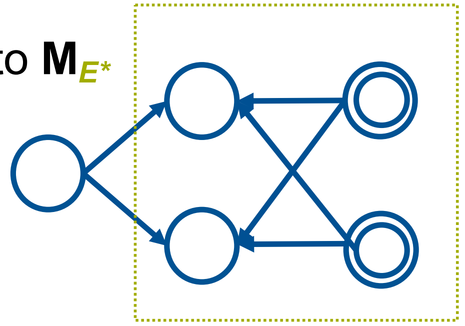
Position automaton $\mathbf{M}_{E^*}$



**Goal output-sensitive construction** (touch each transition once)

**Problem** going from $\mathbf{M}_E$ to $\mathbf{M}_{E^*}$ means to add **feedback transitions** that might be already present





**Solution**

Transform $E$ into **star normal form** $E^\bullet$ so that the position automata $\mathbf{M}_E$ and $\mathbf{M}_{E^\bullet}$ are identical and that all feedback transitions that are introduced during the constraction of $\mathbf{M}_{E^\bullet}$ are new. BTW, the star normal form of $(a^* b^*)^*$ is $(a+b)^*$

## Results

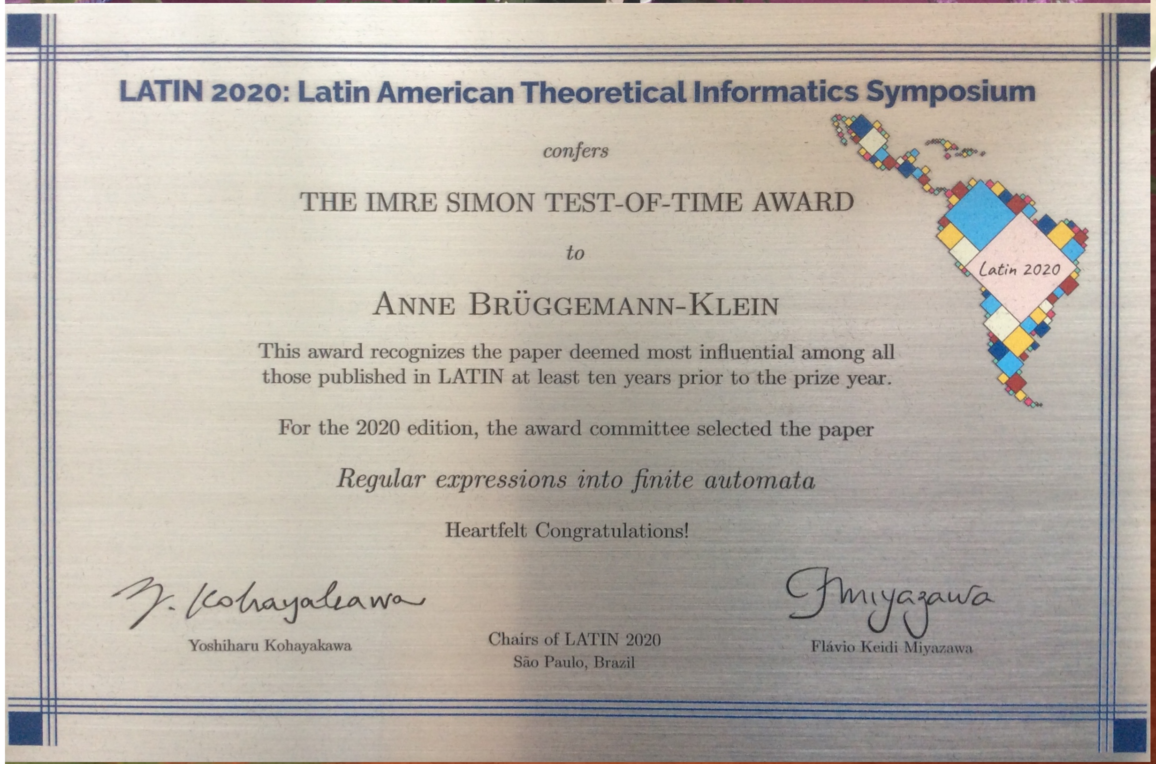
### Theorem

- The position automaton for any regular expression can be constructed in time that is **quadratic** in the size of the expression and proportional to the size of the automaton (**output-sensitive**).
- We can test in linear time if a regular expression is deterministic in the sense of SGML

### Further results (with Derick Wood)

- Characterize the regular languages that can be represented by deterministic expressions. For example, the language of (a+b)* a (a+b) is NOT one of them.
- Extend the work to the full set of operators that SGML expressions support, among them the shuffle operator &.

These results are cited as related work in the W3C recommendation for XML, the Extensible Markup Language, a sequel of SGML that was designed for use on the web. They provide a level of mathematical rigour in terms of concepts and reasoning to W3C standards that is not present in all W3C recommendations and that the document engineering community appreciates.



**LATIN 2020: Latin American Theoretical Informatics Symposium**

*confers*

THE IMRE SIMON TEST-OF-TIME AWARD

*to*

ANNE BRÜGGEMANN-KLEIN

This award recognizes the paper deemed most influential among all those published in LATIN at least ten years prior to the prize year.

For the 2020 edition, the award committee selected the paper

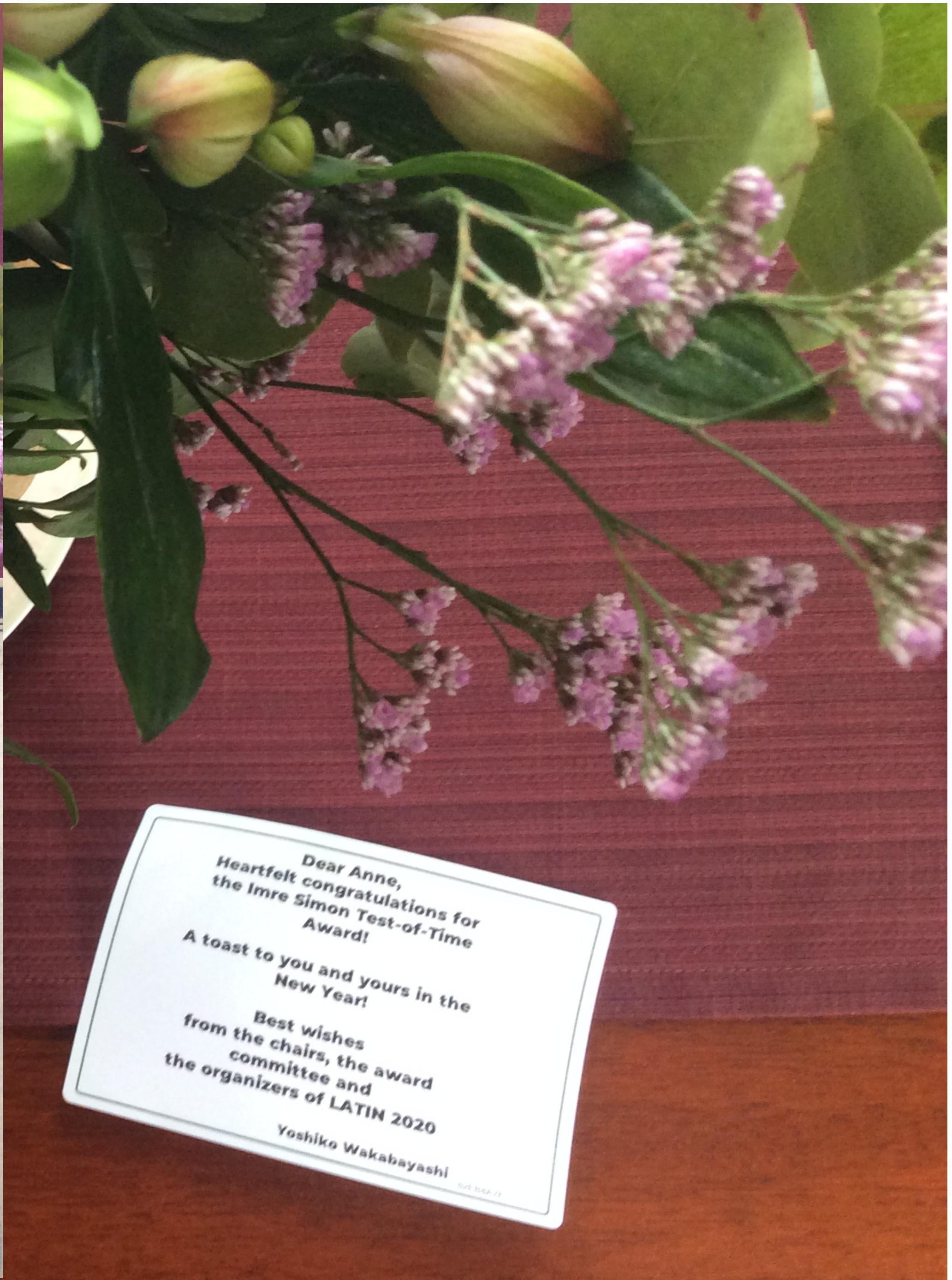
*Regular expressions into finite automata*

Heartfelt Congratulations!

Y. Kohayakawa
Yoshiharu Kohayakawa

Chairs of LATIN 2020
São Paulo, Brazil

F. Miyazawa
Flávio Keidi Miyazawa



Dear Anne,
Heartfelt congratulations for the Imre Simon Test-of-Time Award!

A toast to you and yours in the New Year!

Best wishes
from the chairs, the award committee and the organizers of LATIN 2020

Yoshiko Wakabayashi

# Thank you for this incredible honor!