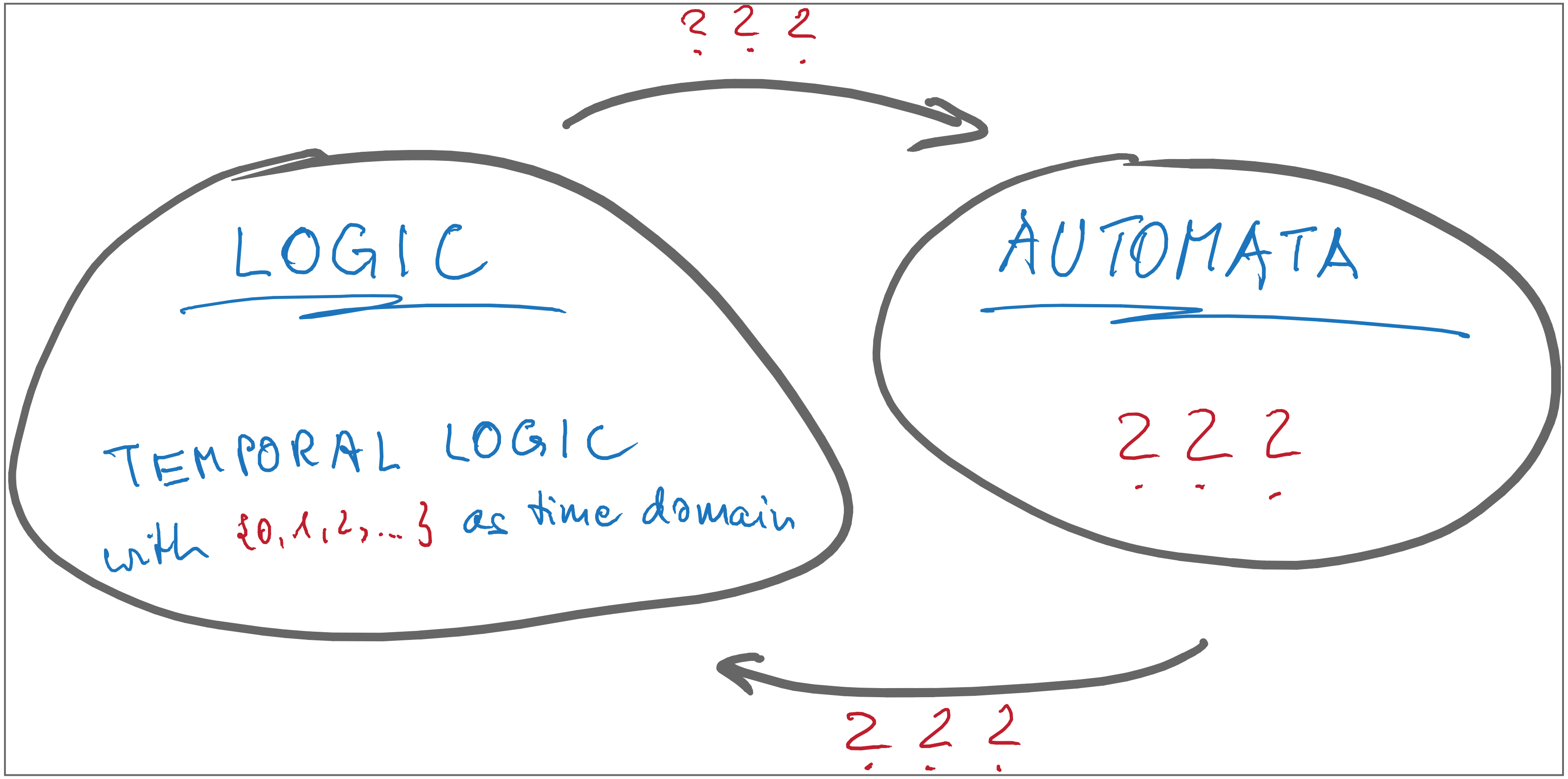
PAST, PRESENT, AND INFINITE FUTURE

ICALP 2016

THOMAS WILKE, KIEL UNIVERSITY

(My first HTML presentation—thanks to **Hakim** and Björn.)

INTERNATIONAL COLLOQUIUM ON AUTOMATA, LOGIC, AND PROGRAMMING

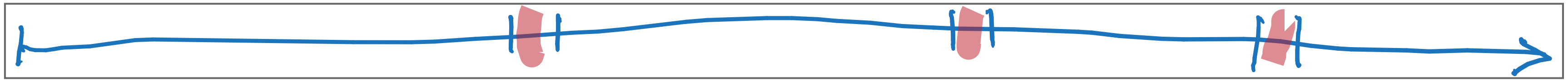


1

TEMPORAL LOGIC: FUTURE ONLY

EXAMPLE

“always in the future (sometime in the future p”)



QUESTION

What is the right automaton model for dealing with this?

TEMPORAL LOGIC: FUTURE AND PAST

EXAMPLE

“in the future (p and always in the past q)”

QUESTION

What is the right automaton model for dealing with this?

1

MORE BACKGROUND

FINITE AND INFINITE WORDS, IN GENERAL

TFAE: first-order logic, temporal logic (future only vs. future and past), counter-free automata, aperiodic semigroups, star-free expressions, ...

[Results by: McNaughton/Papert, Schützenberger, Kamp, Gabbay/Pnueli/Shelah/Stavi, Thomas, Diekert/Gastin, ...]

ISSUES ADDRESSED HERE, FOR INFINITE WORDS

- a good translation from automata to future temporal logic
- a good automaton model for future/past temporal logic

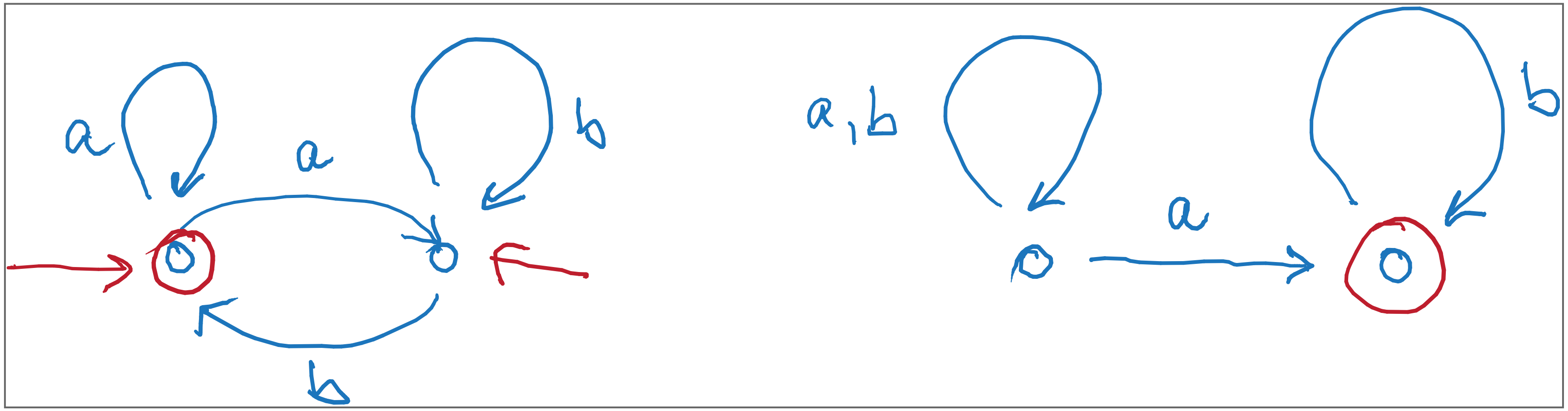
PART I: FUTURE ONLY

BACKWARD DETERMINISTIC ω-AUTOMATA

CARTON & MICHEL 2000

EXAMPLE

“infinitely often a”



1. For **every** word, there must be exactly **one** recurring run.
2. The transition function must be **backward deterministic**.

THE STANDARD TRANSLATION

1. Formulas in negation normal form.
2. A symbol = a set of propositional variables.
3. A state = a set of subformulas.
4. Reverse deterministic transition function:

$$\left.\begin{array}{r} p \\ \neg p \\ \text{next } \varphi \\ \varphi \text{ until } \psi \\ \end{array}\right\} \in a \langle \Phi \text{ iff } \left\{\begin{array}{l} p \in a \\ p \notin a \\ \varphi \in \Phi \\ \psi \in a \langle \Phi \text{ or} \\ \quad (\varphi \in a \langle \Phi \text{ and } \varphi \text{ until } \psi \in \Phi) \end{array}\right.$$

5. Appropriate recurrence condition.

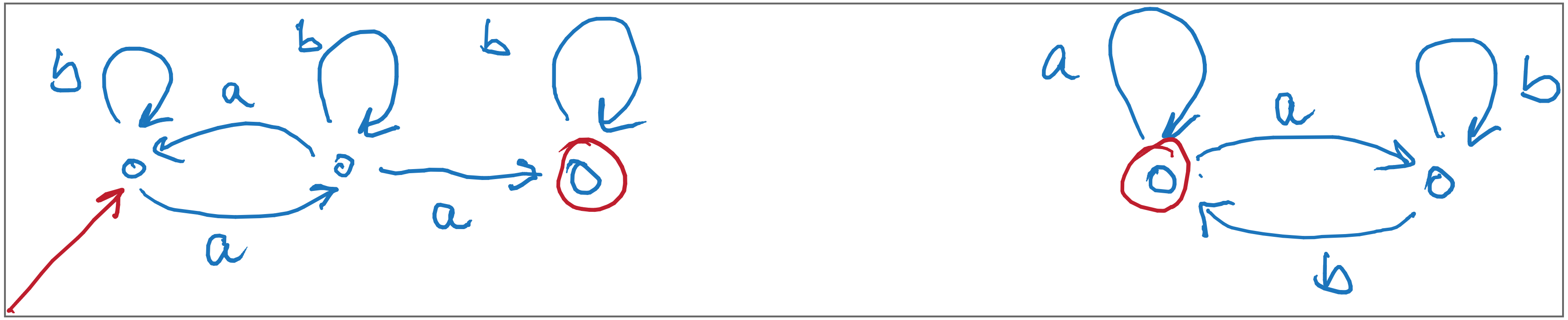
STRUCTURAL PROPERTY: COUNTER-FREE

DIEKERT & GASTIN 2008

There is no word which acts on a subset of the state space as a non-trivial permutation.

EXAMPLE

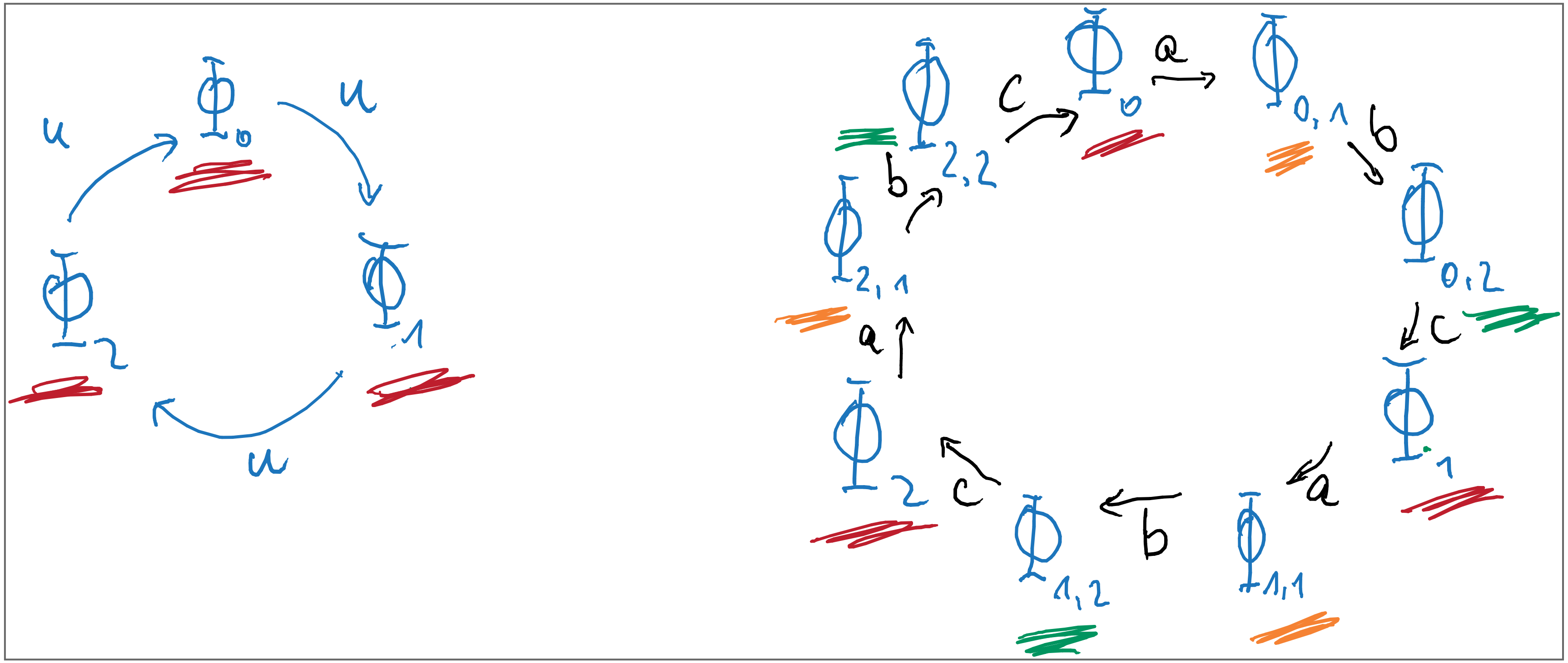
“a finite and even number of a's”



1

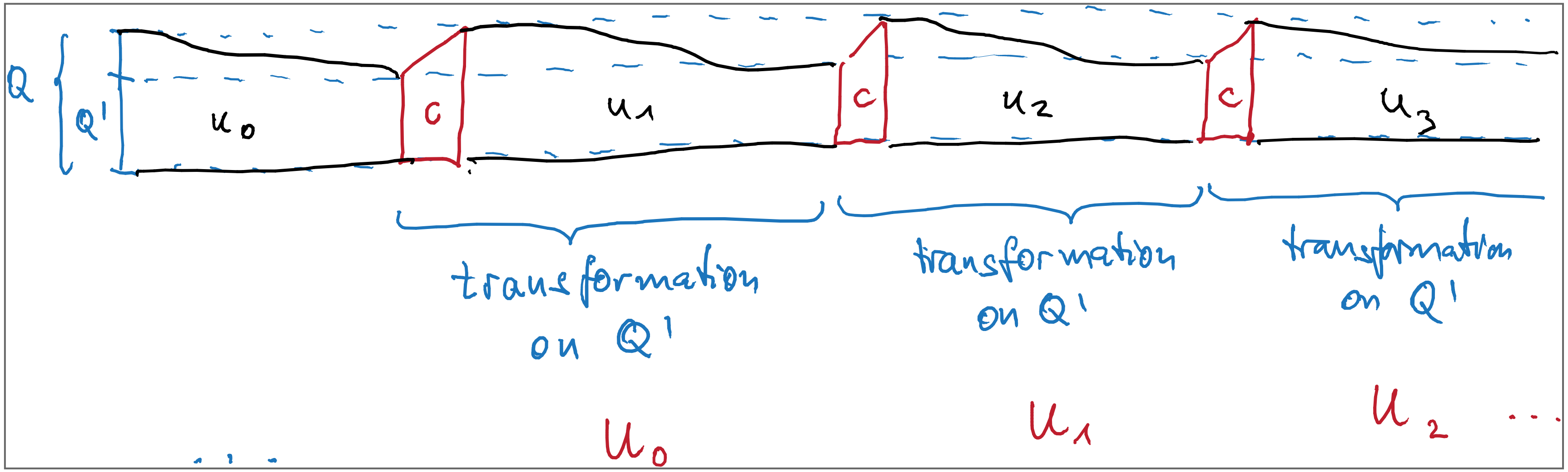
SKETCH OF PROOF

Assume u induces a permutation on $\{\Phi_0, \ldots, \Phi_{n-1}\}$. We show by induction: $\Phi_0 = \cdots = \Phi_{n-1}$. Most interesting: φ until $\psi \in \Phi_0$.



TRANSLATION FROM AUTOMATA TO LOGIC

Inductively, on the number of states and the number of symbols. Interesting case: there is some c s.th. $Q' = \mathrm{image}(c) \subsetneq Q$ and $c \in \inf(u)$.



Finite word case: formulas for words cu_i.

By induction: formulas for words over $\{U_0, U_1, \ldots\}$.

PART II: FUTURE AND PAST

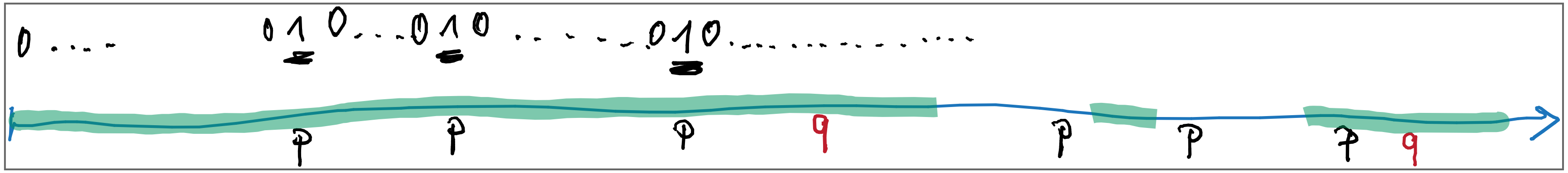
FORMULAS INDUCE FUNCTIONS

A future/past formula φ induces a function $f_\varphi\colon \Sigma^\omega \to \{0,1\}^\omega$,
where

$$f_\varphi(u)(i) = \begin{cases} 1 & \text{if } \varphi \text{ holds at } i \text{ in } u \\ 0 & \text{otherwise} \end{cases}$$

EXAMPLE

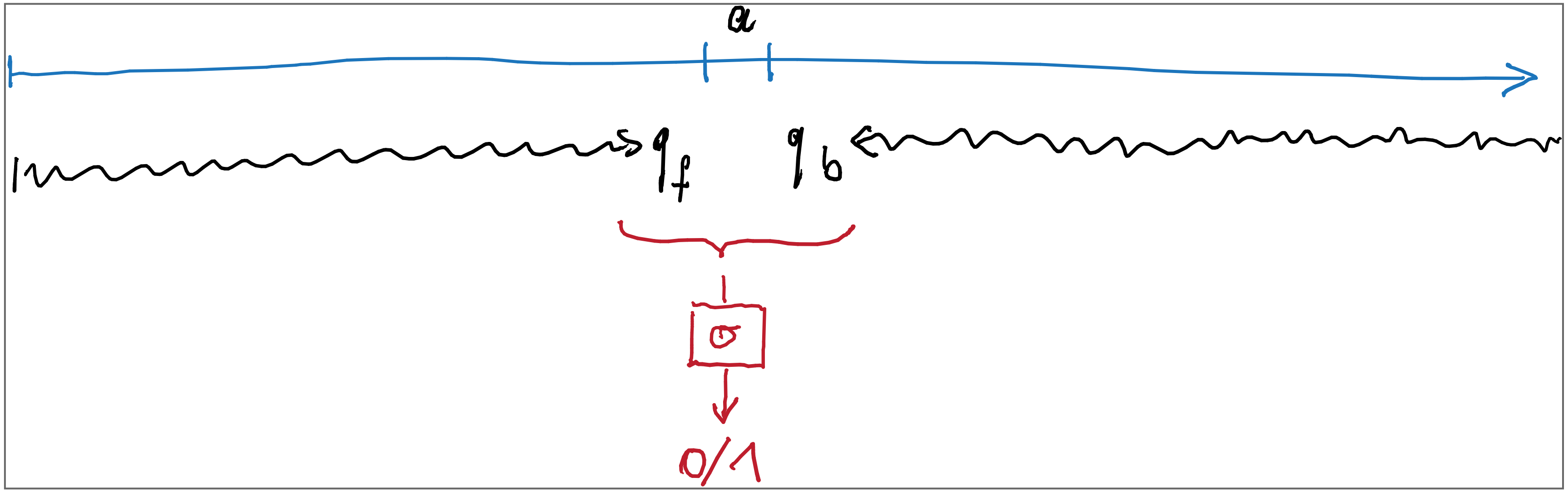
"p and in the future (q and always in the past r)"



(COUNTER-FREE) BIMACHINES

SCHÜTZENBERGER 1961

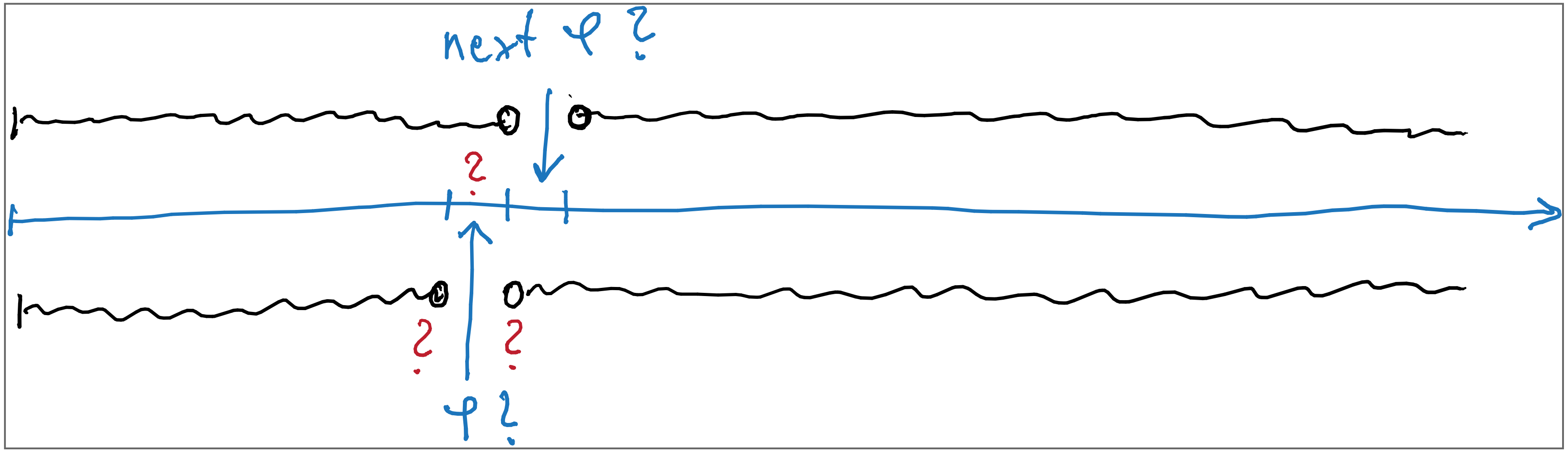
- a **forward deterministic** automaton on **finite** words
- a **backward deterministic** automaton on **infinite** words
- an **output function** $o\colon Q_f \times \Sigma \times Q_b \to \{0,1\}$



TRANSLATING FORMULAS INTO AUTOMATA

MORE COMPLEX THAN BEFORE

EXAMPLE “ NEXT φ ”



forward automaton lags behind: $Q_l \times (\Sigma \cup \epsilon)$

output function: $o'(\langle q_f, b\rangle, b, q_b) = o(q_f, b, a\langle q_b)$

1

TRANSLATING AUTOMATA INTO FORMULAS

Easy, because of prior results:

- formulas for the forward automaton (finite words),
- formulas for the backward automaton (Part I).

COROLLARY (GABBAY 1980, SYNTACTIC ARGUMENT)

Every future/past formula is equivalent to a boolean combination of

- pure past,
- present, and
- pure future formulas.

1

CONCLUSION

- Use **backward deterministic counter-free automata** for **future** temporal logic.
- Use **counter-free bimachines** for **future/past** temporal logic.

QUESTIONS

- What is the complexity of the various constructions?
- What happens when “counter-freeness” is replaced by some other interesting property?
- Is the automaton model useful for formal verification?