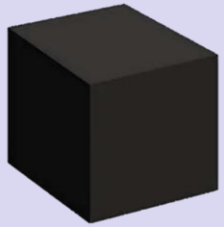
Learning Broadcast Protocols (with LeoParDS)

Dana Fisman [1], Noa Izsak [1], Swen Jacobs [2]

[1] Computer Science Department, Ben-Gurion University,
[2] CISPA Helmholtz Center for Information Security

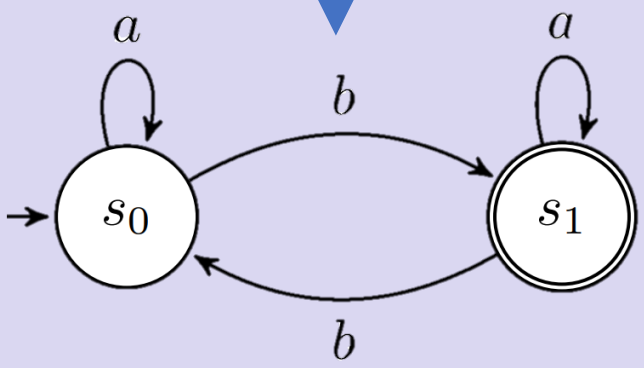
Israel Verification Day - IVD'24

Automata Learning

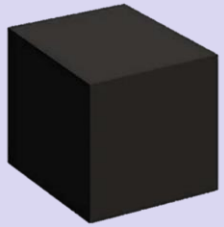


+	-
aba	*aa*
ba	*bba*
abbb	*abb*

Learner

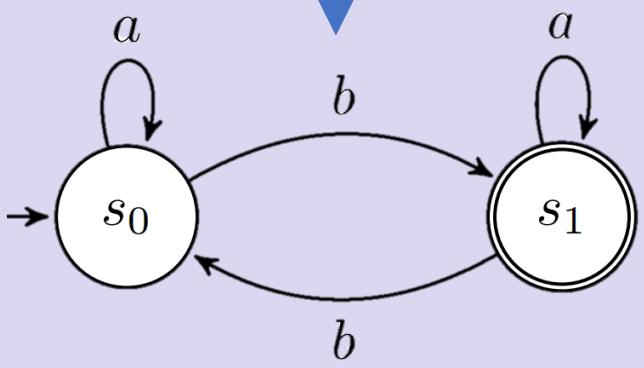


Automata Learning



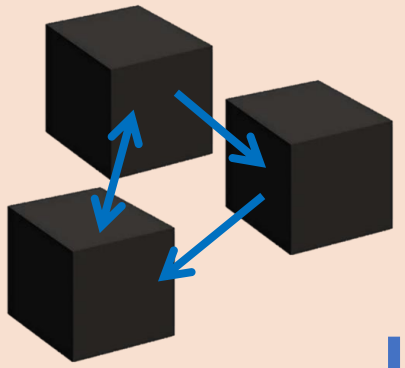
+	-
aba	aa
ba	bba
abbb	abb

Learner



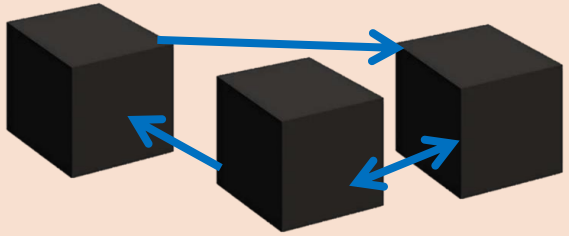
Learning Concurrent systems

Fixed size
(state of the art)

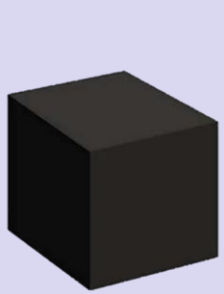


+	-
aba	aa
ba	bba
abbb	abb

Learner

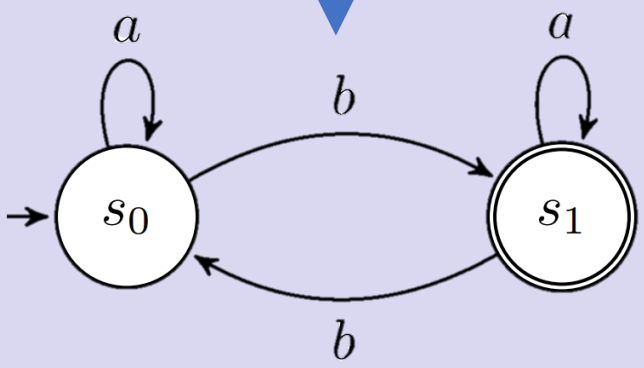


Automata Learning



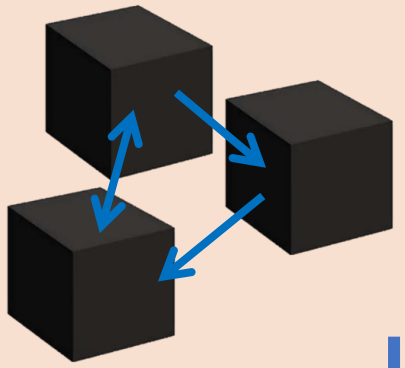
+	-
aba	aa
ba	bba
abbb	abb

Learner



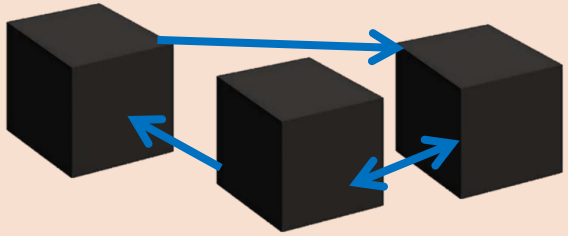
Learning Concurrent systems

Fixed size
(state of the art)

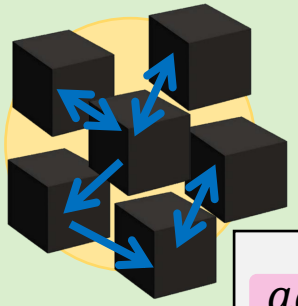


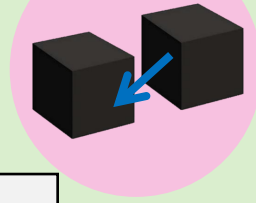
+	-
aba	aa
ba	bba
abbb	abb

Learner

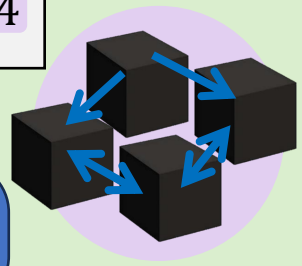


Arbitrary size
Learning Broadcast Protocols, AAAI24

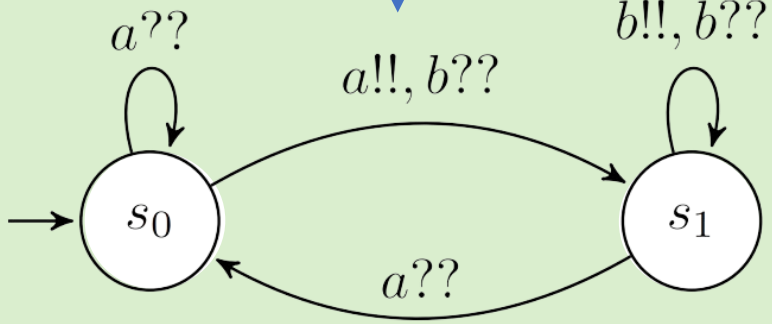




+	-
aab, 2	aa, 1
ab, 1	bba, 2
abbb, 6	abba, 4



Learner



Parametrized protocols

Parametrized protocols

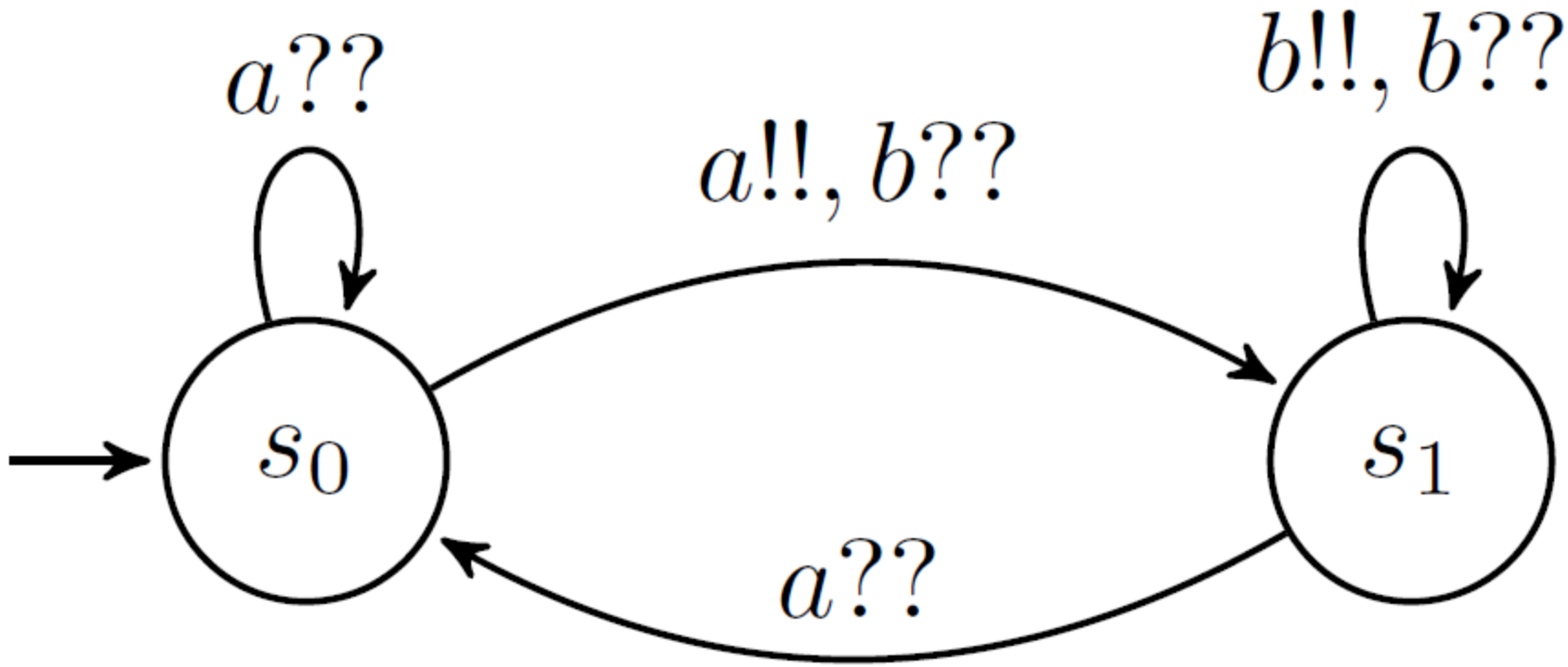
Given a protocol P, $n \in \mathbb{N}$.

A parallel running of protocol P for n processes: $P^n = \underbrace{P \parallel P \parallel \cdots \parallel P}_{n}$

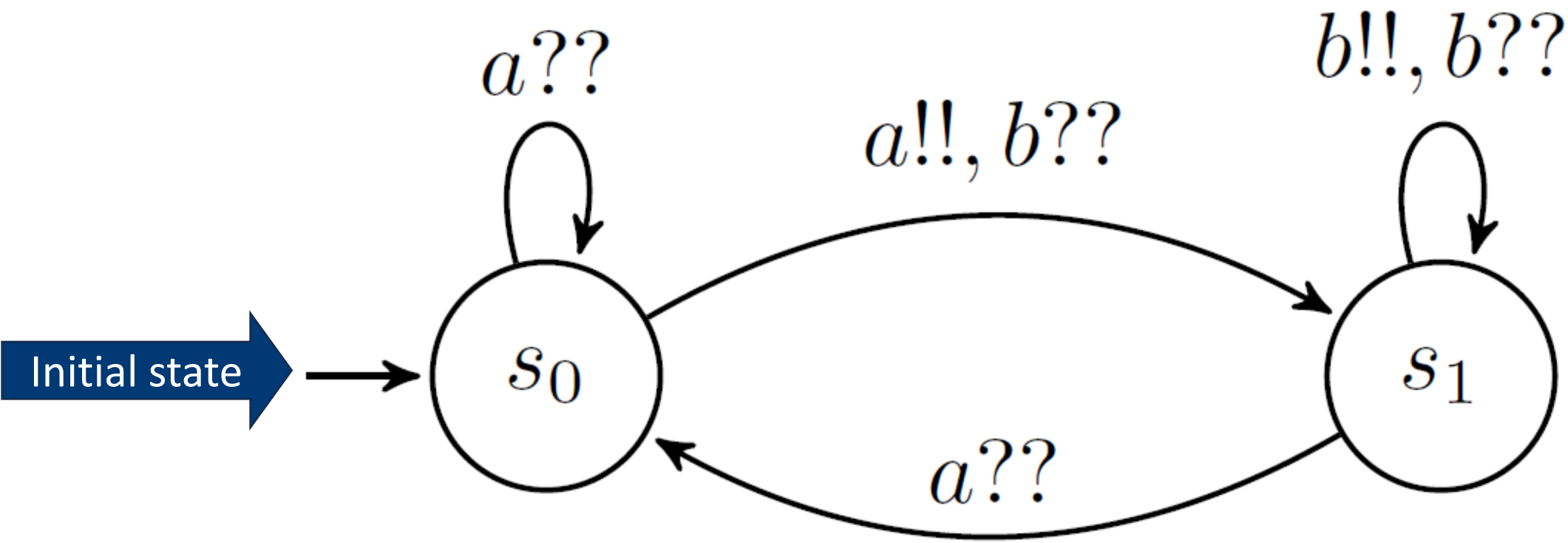
The "language" of P^n: $\mathcal{L}(P^n) = \underbrace{P \parallel P \parallel \cdots \parallel P}_{n}$

The "language" of a protocol P: $\mathcal{L}(P) = \bigcup_{n \in \mathbb{N}} \mathcal{L}(P^n)$

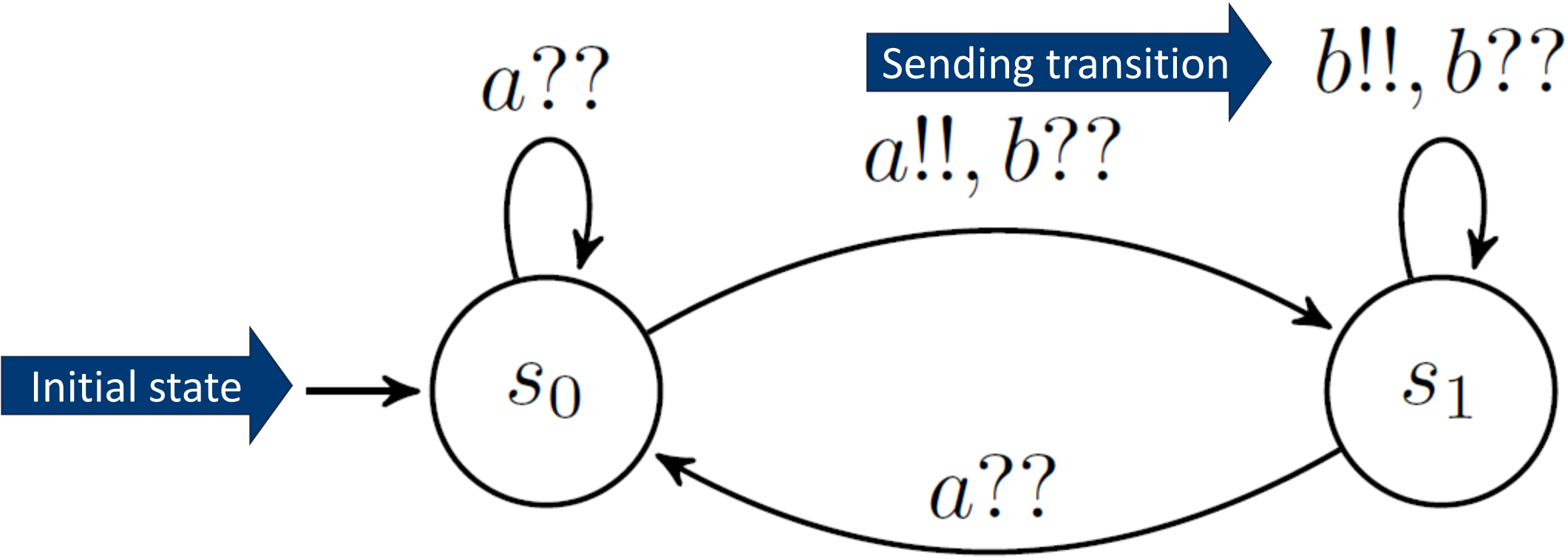
A Broadcast Protocol (BP)



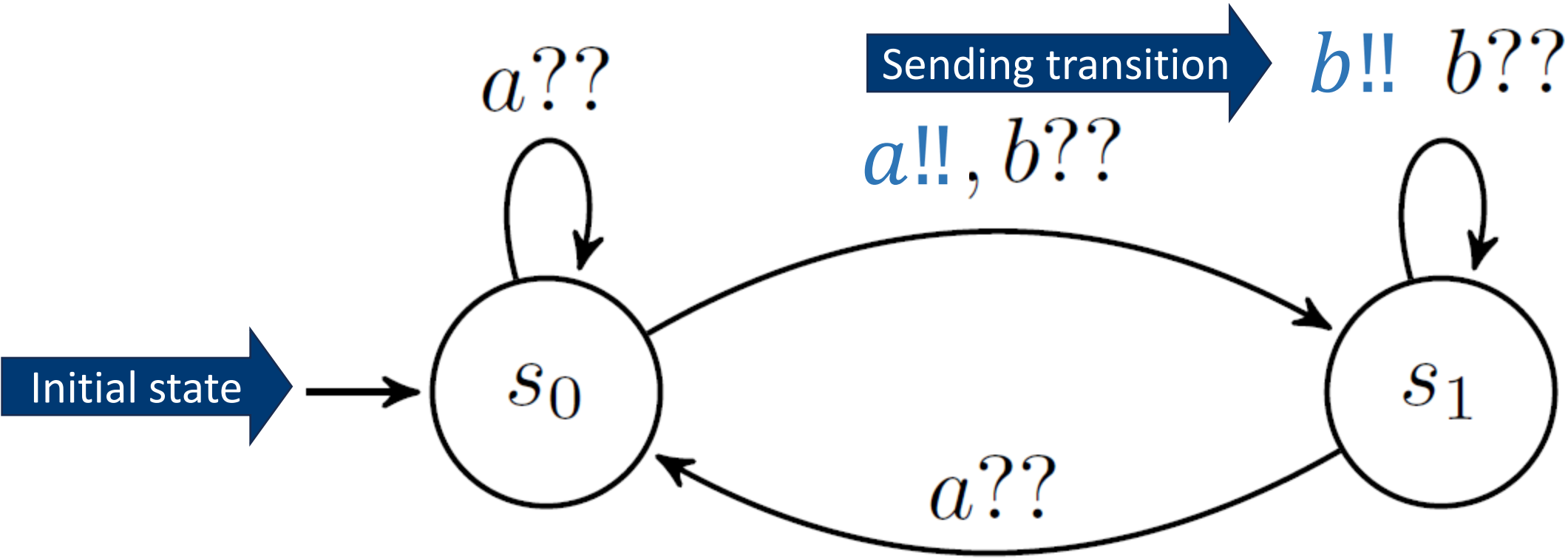
A Broadcast Protocol (BP)



A Broadcast Protocol (BP)

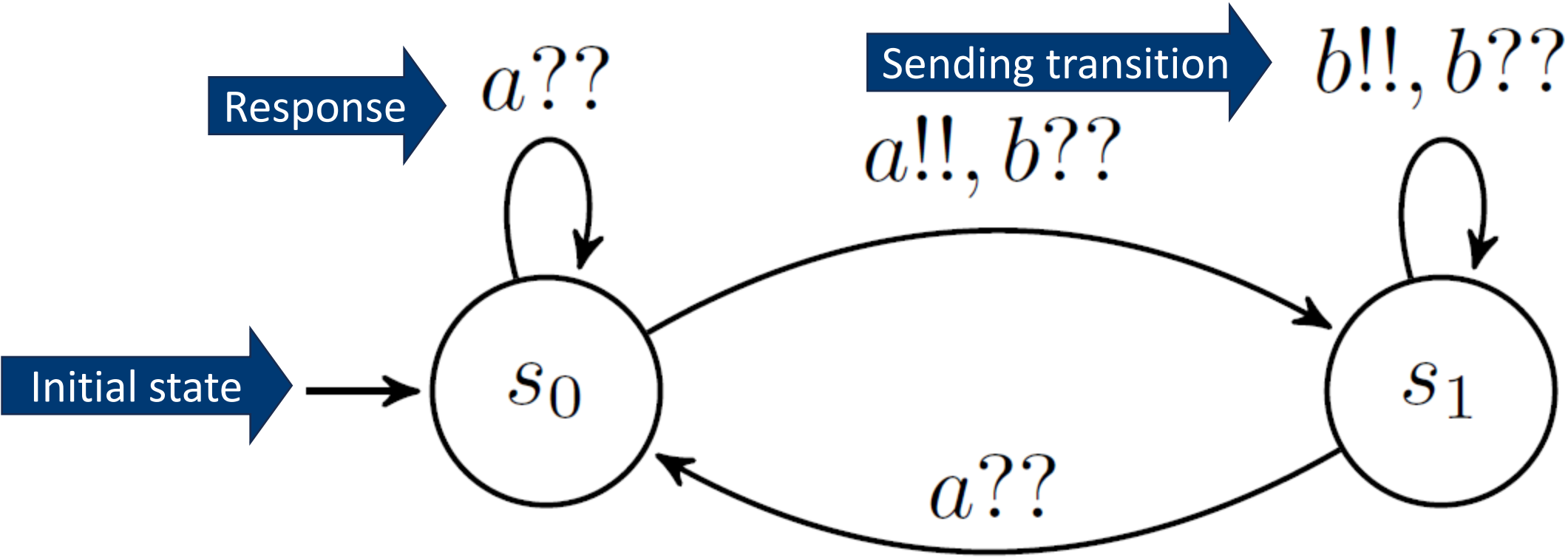


A Broadcast Protocol (BP)

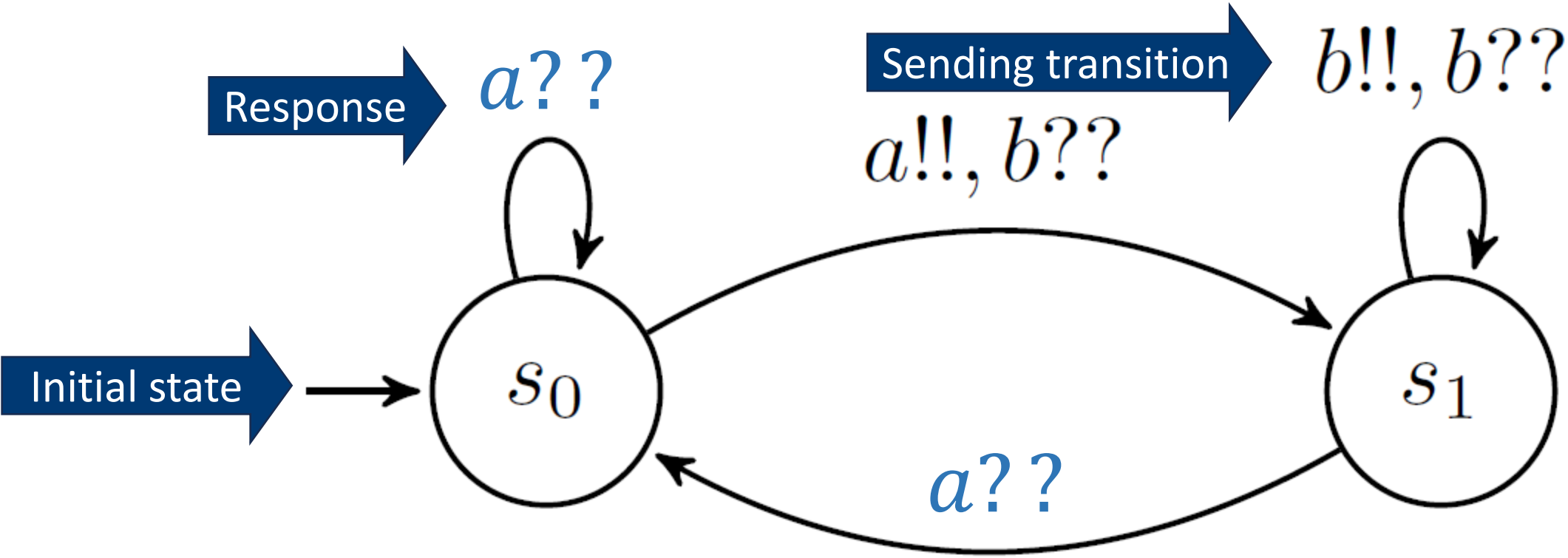


[1999] On the Verification of Broadcast Protocols

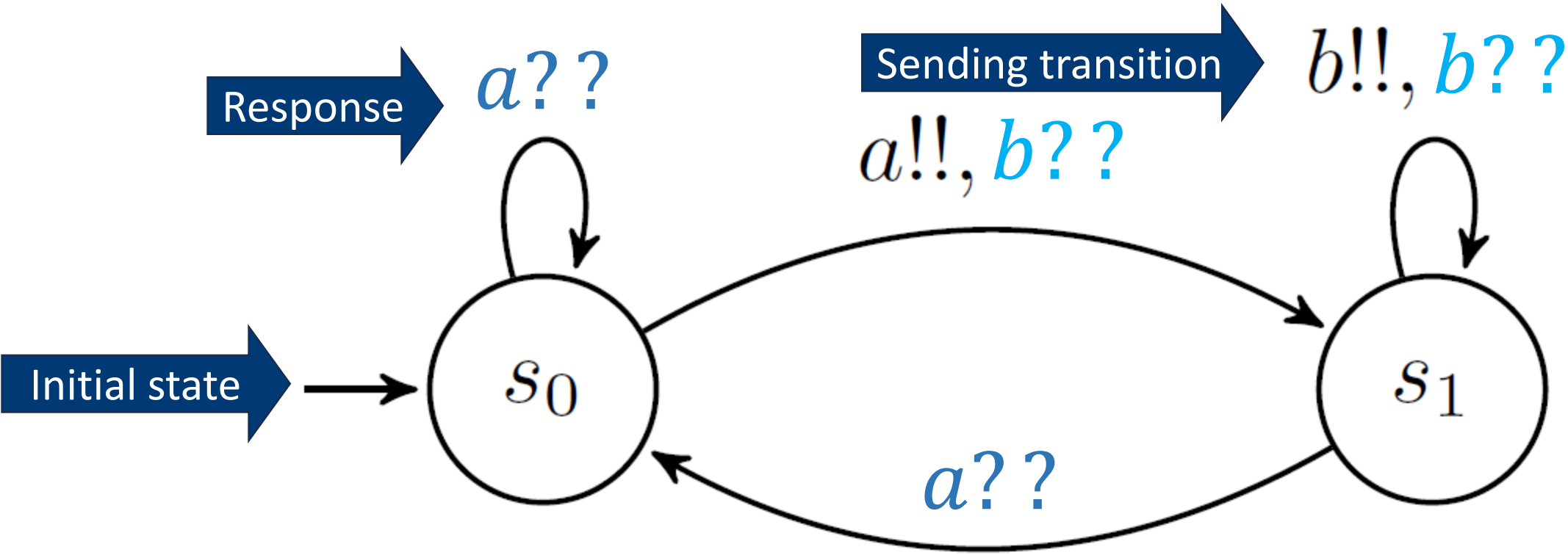
A Broadcast Protocol (BP)



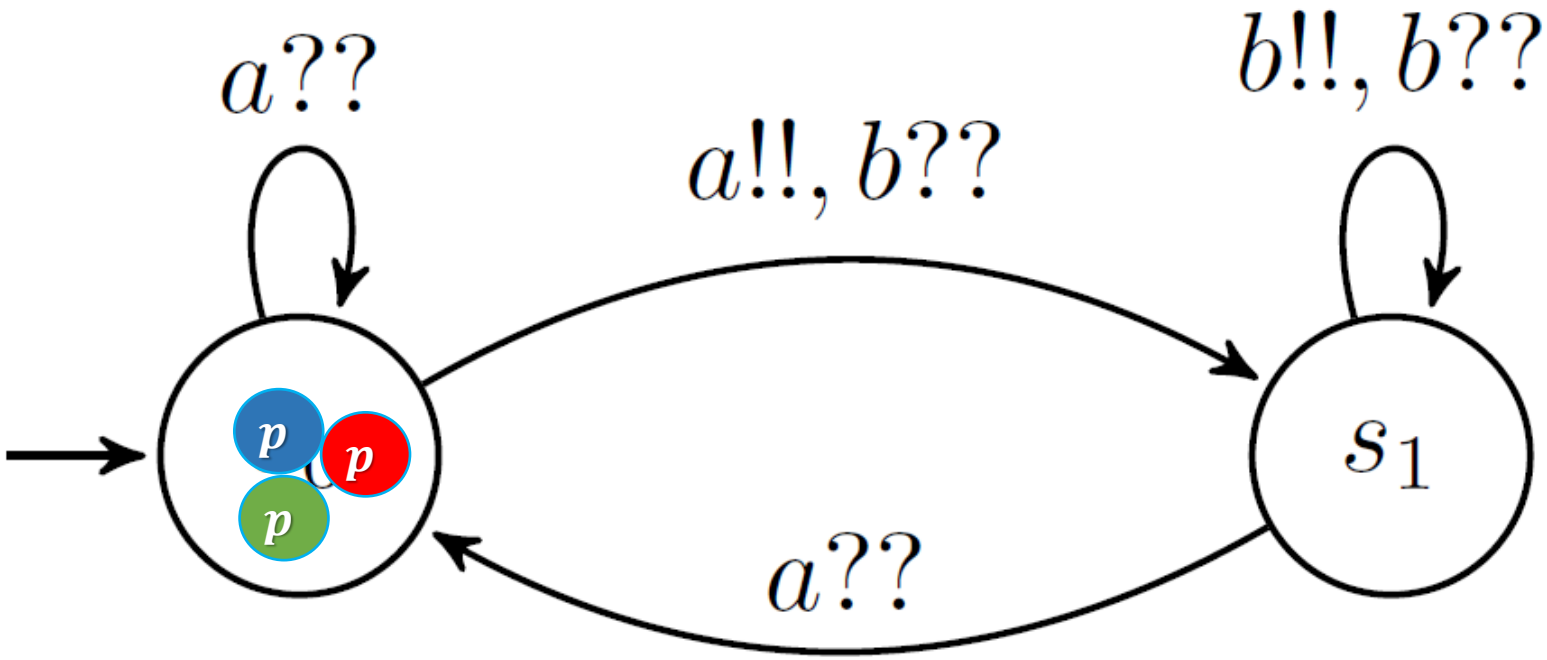
A Broadcast Protocol (BP)



A Broadcast Protocol (BP)



A simple BP and its execution



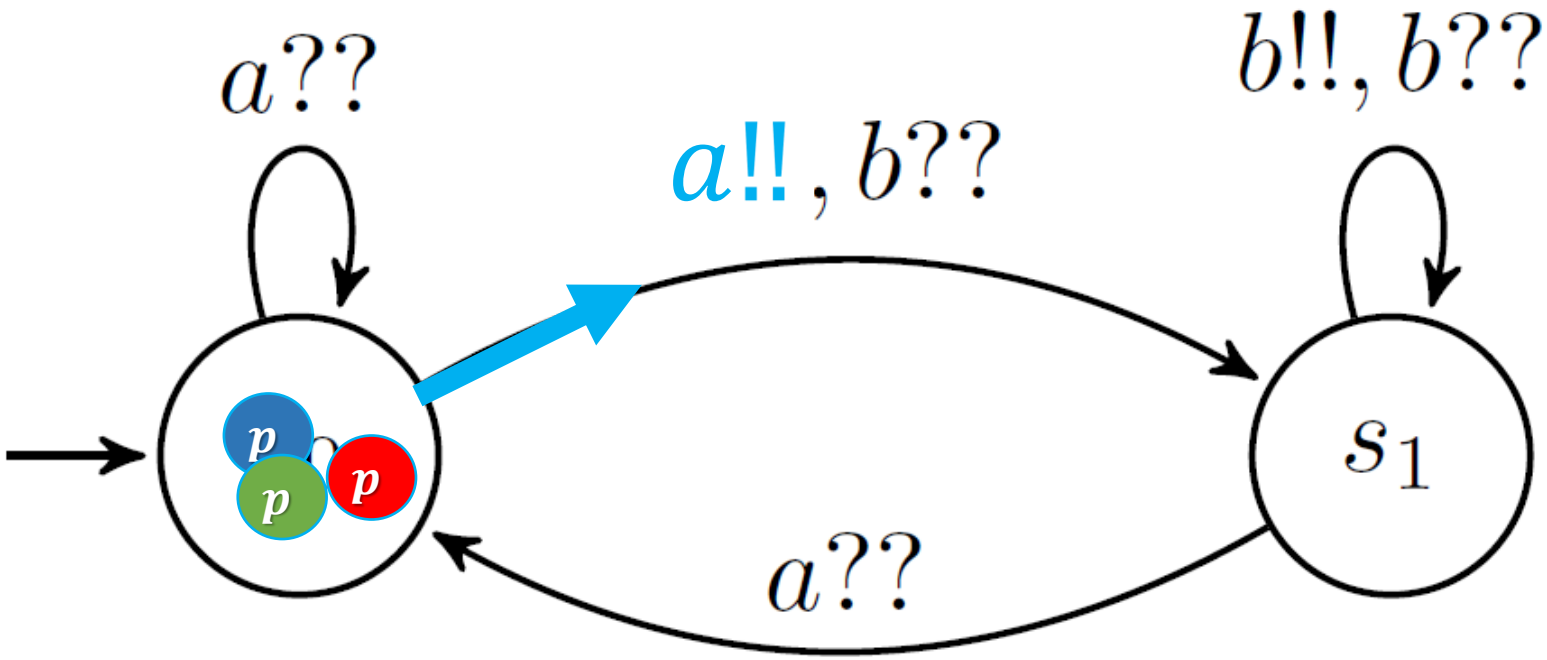
$$\begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

state vector

A simple BP and its execution

$$\begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

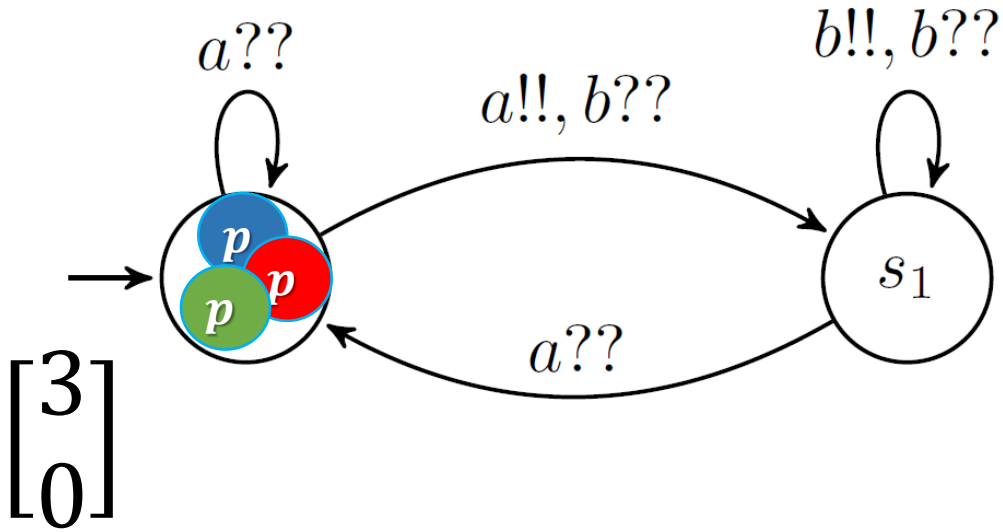
state vector



one process, the sender, takes a $\sigma!!$ transitions

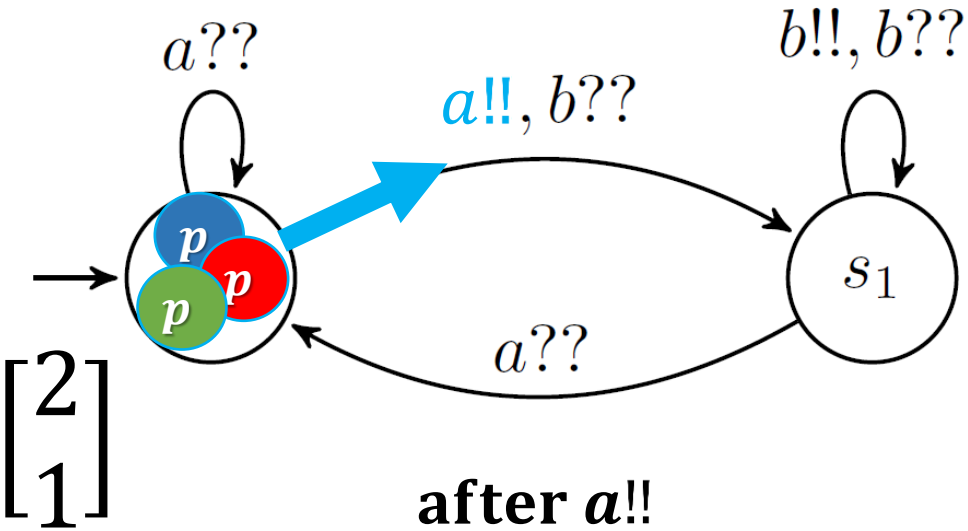
all the other processes, the receivers, respond by following the $\sigma??$ transitions

A simple BP and its execution



$$\begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

state vector



$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

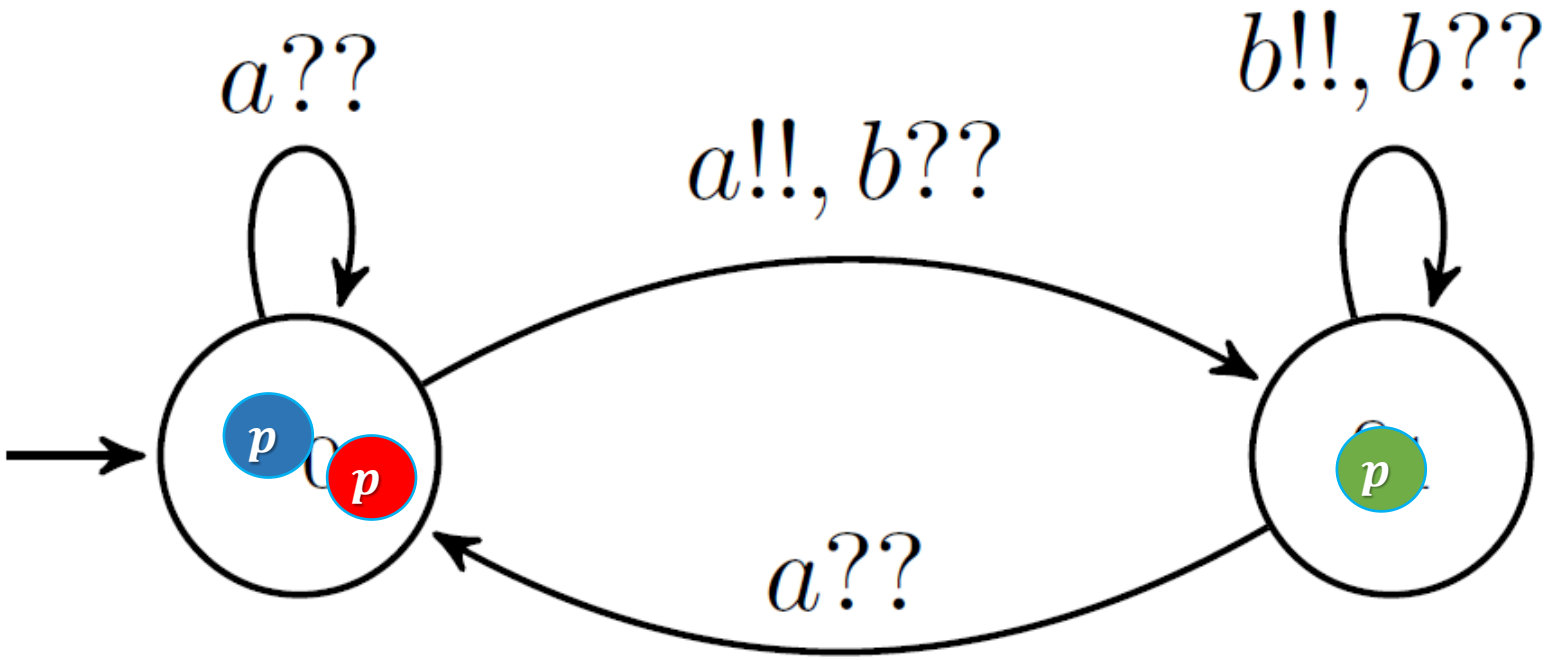
after *a*!!

state vector

one process, the sender, takes a $\sigma!!$ transitions

all the other processes, the receivers, respond by following the $\sigma??$ transitions

A simple BP and its execution



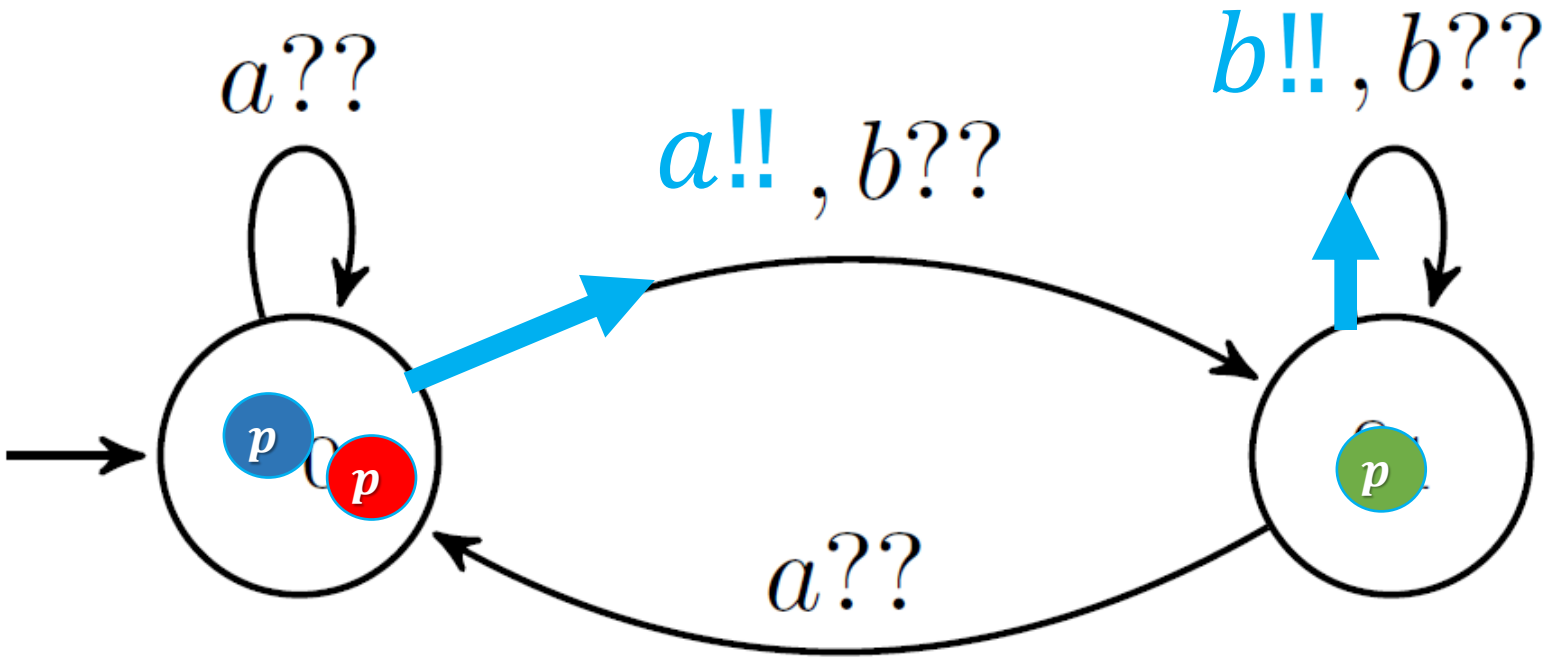
$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

state vector

one process, the sender, takes a $\sigma!!$ transitions

all the other processes, the receivers, respond by following the $\sigma??$ transitions

A simple BP and its execution



$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

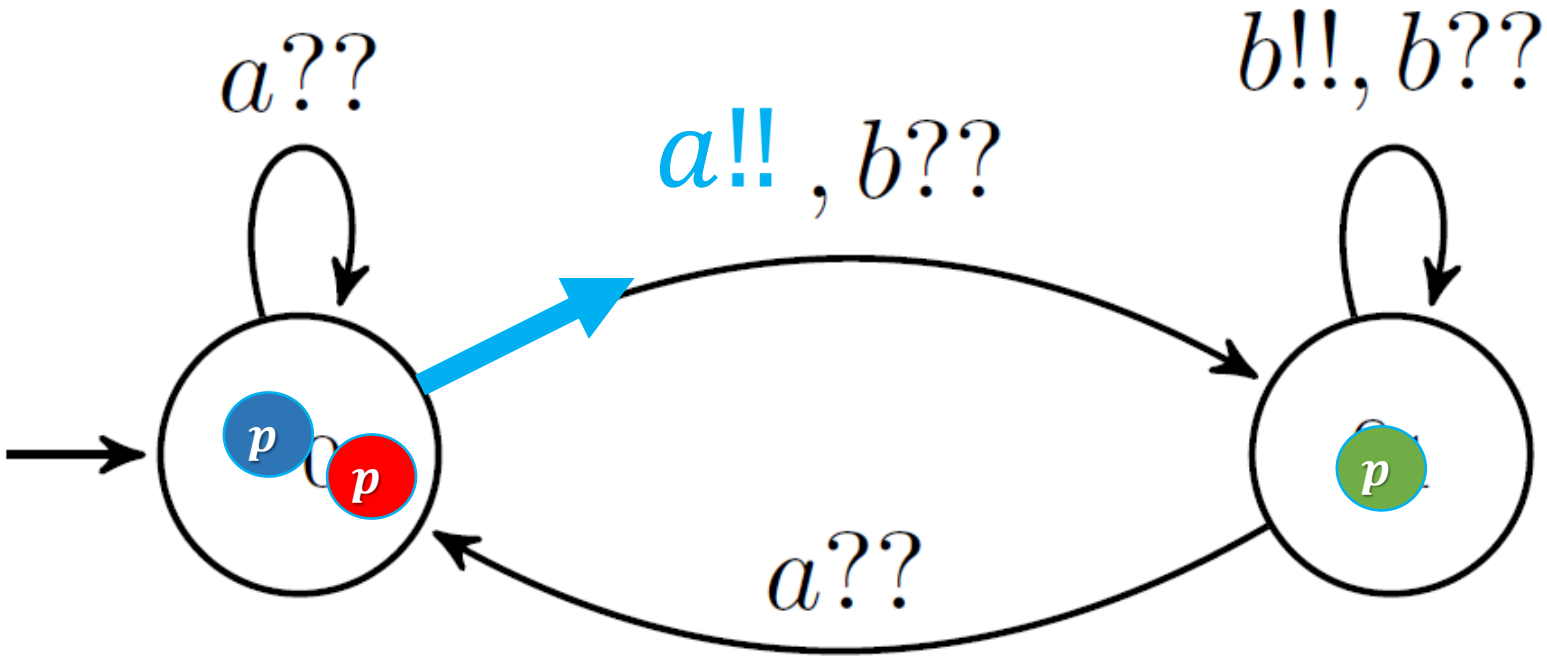
state vector

one process, the sender, takes a $\sigma!!$ transitions

all the other processes, the receivers, respond by following the $\sigma??$ transitions

A simple BP and its execution

action $\boldsymbol{a}$ is being broadcasted



$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

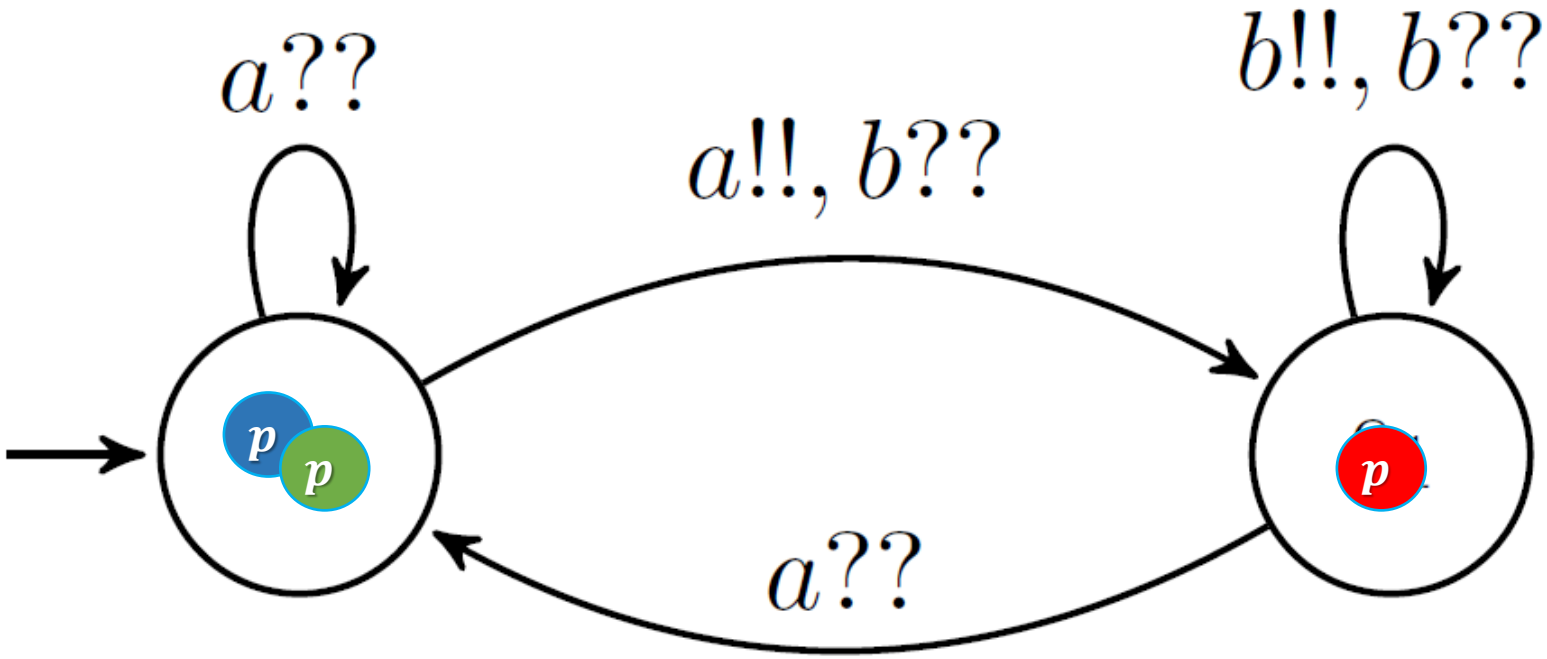
$state\ vector$

one process, the sender, takes a $\sigma!!$ transitions

all the other processes, the receivers, respond by following the $\sigma??$ transitions

A simple BP and its execution

action $\boldsymbol{a}$ is being broadcasted



$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

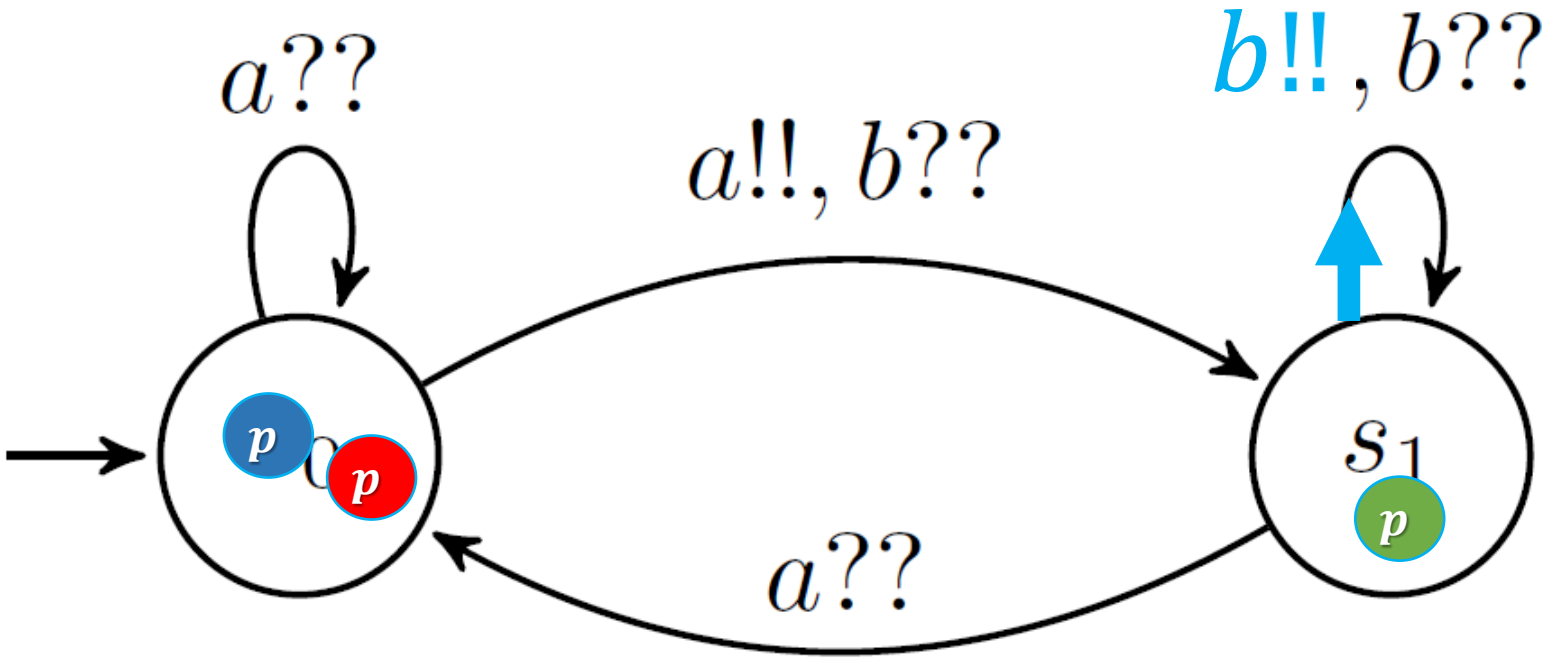
$state\ vector$

one process, the sender, takes a $\sigma!!$ transitions

all the other processes, the receivers, respond by following the $\sigma??$ transitions

A simple BP and its execution

action $\boldsymbol{b}$ is being broadcasted



$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

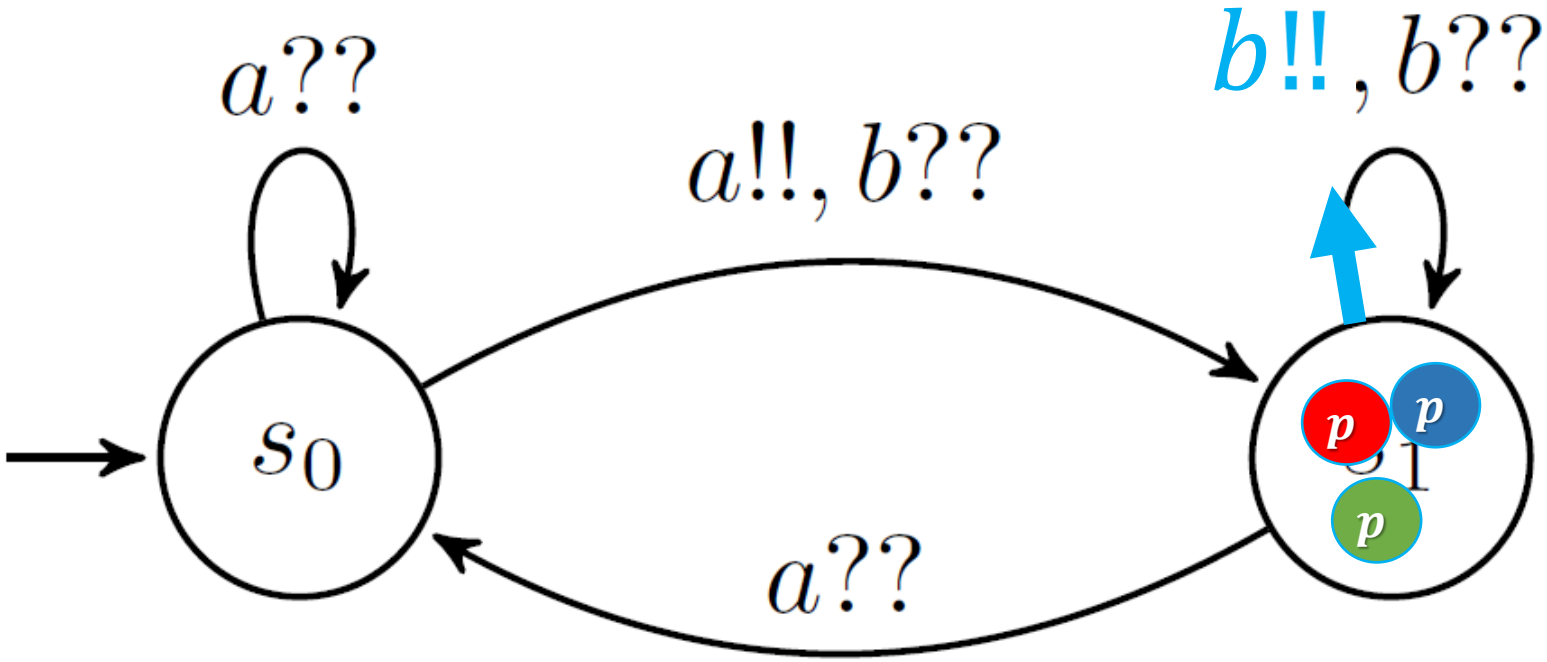
$state\ vector$

one process, the sender, takes a $\sigma!!$ transitions

all the other processes, the receivers, respond by following the $\sigma??$ transitions

A simple BP and its execution

action $\boldsymbol{b}$ is being broadcasted



$$\begin{bmatrix} 0 \\ 3 \end{bmatrix}$$

state vector

one process, the sender, takes a $\sigma!!$ transitions

all the other processes, the receivers, respond by following the $\sigma??$ transitions

Note that $\mathcal{L}(P^1) \subseteq \mathcal{L}(P^2) \subseteq \mathcal{L}(P^3) \subseteq \cdots$

Are these inclusions strict, or does there exist an n s.t. adding more processes does not change the language?

Cutoff

$$\text{If } \exists n \in \mathbb{N} \text{ s.t. } \forall m > n \;\; \mathcal{L}(P^n) = \mathcal{L}(P^m)$$

If such an n exists, then the system has a **cutoff**, n.
Otherwise, we say there is no cutoff.

Cutoff

$$\exists n \in \mathbb{N} \text{ s.t. } \forall m > n \;\; \mathcal{L}(P^n) = \mathcal{L}(P^m)$$

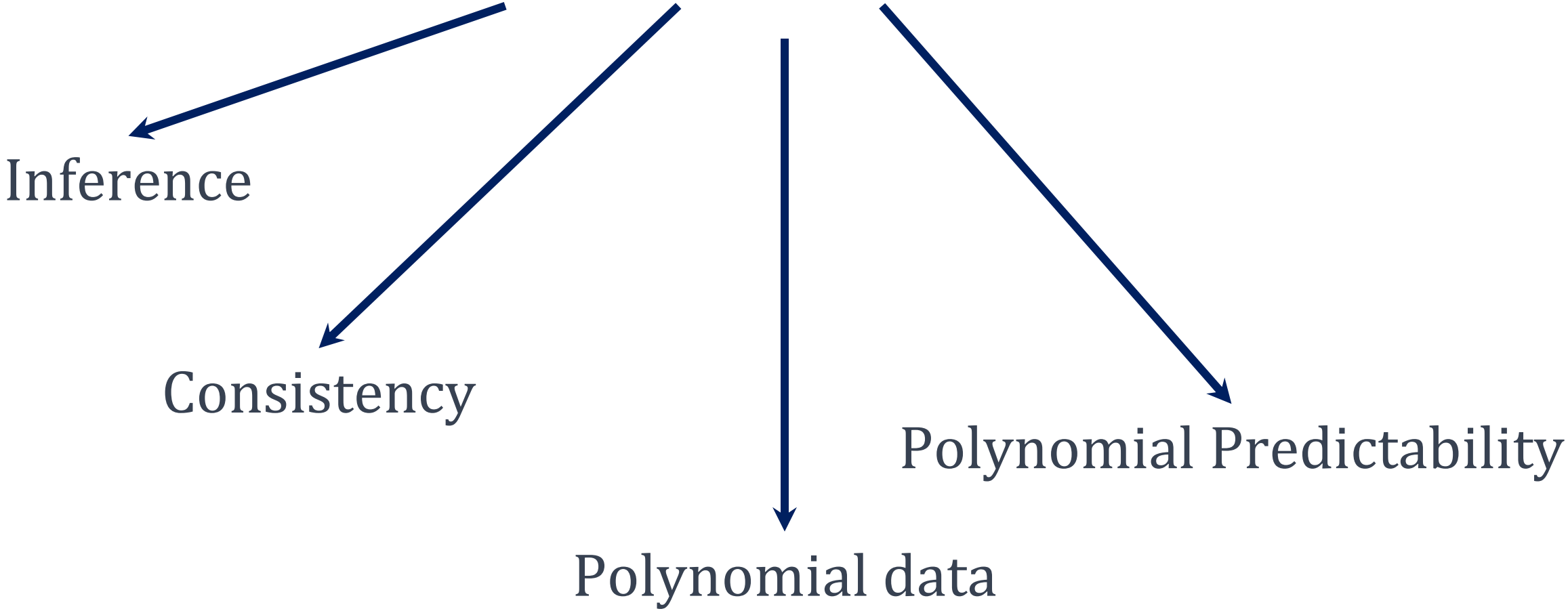
Fine BPs

A BP that:

1. Has no **hidden** states
2. A cutoff **exists**

Learning paradigms

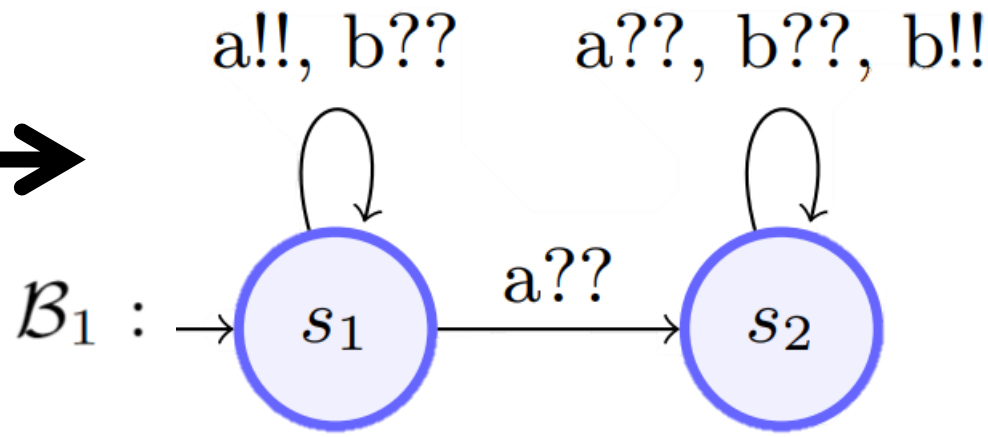
Learning paradigms



Protocols Inference

+	-
$aab, 2$	$ab, 1$
$aaa, 1$	$bba, 2$
$abbb, 6$	$b, 1$
...	...

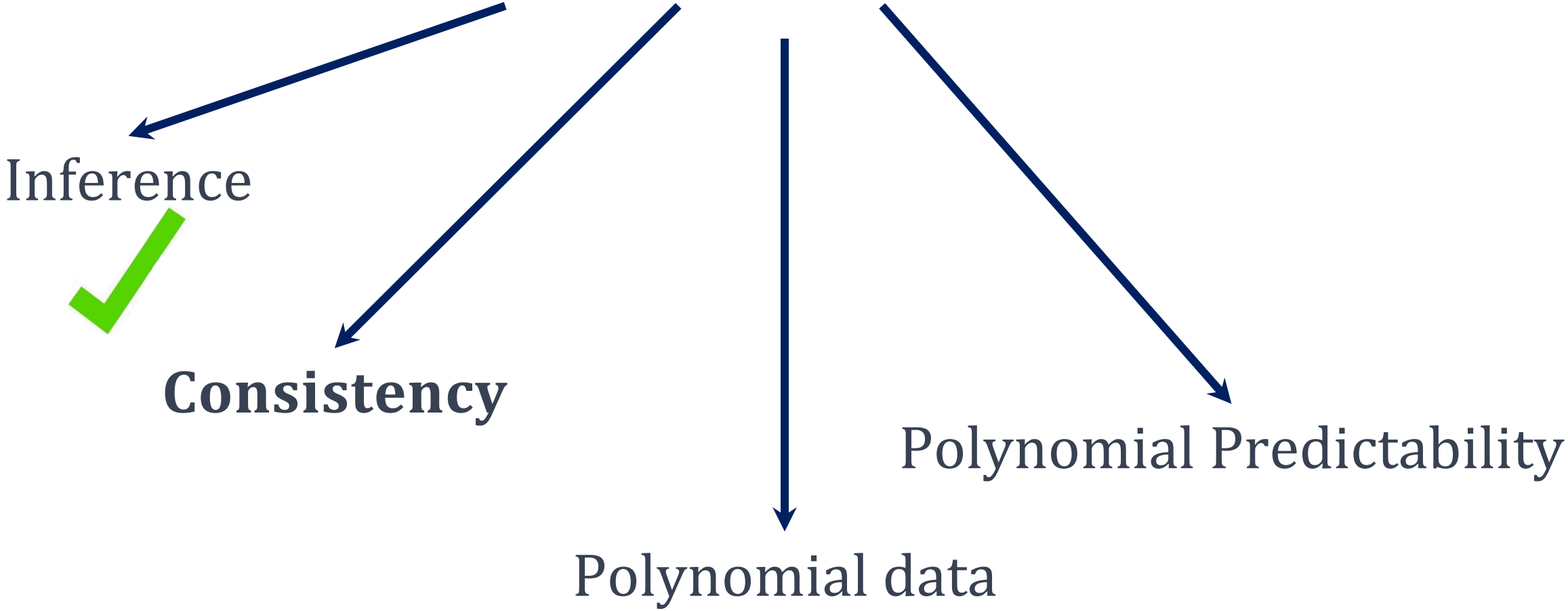
Consistent sample



Inference

We provide an inference algorithm for BPs,
given a sample of words that are consistent with a BP,
infers a correct BP.

Learning paradigms



Consistency

Let C be the class of fine BPs,

Given sample $\mathcal{S}$ and $k \in \mathbb{N}$, determine whether there

exists a BP $B \in C$ consistent with $\mathcal{S}$ with at most k states.

Consistency

We show that consistency is NP-hard for the class of fine BPs.

Consistency

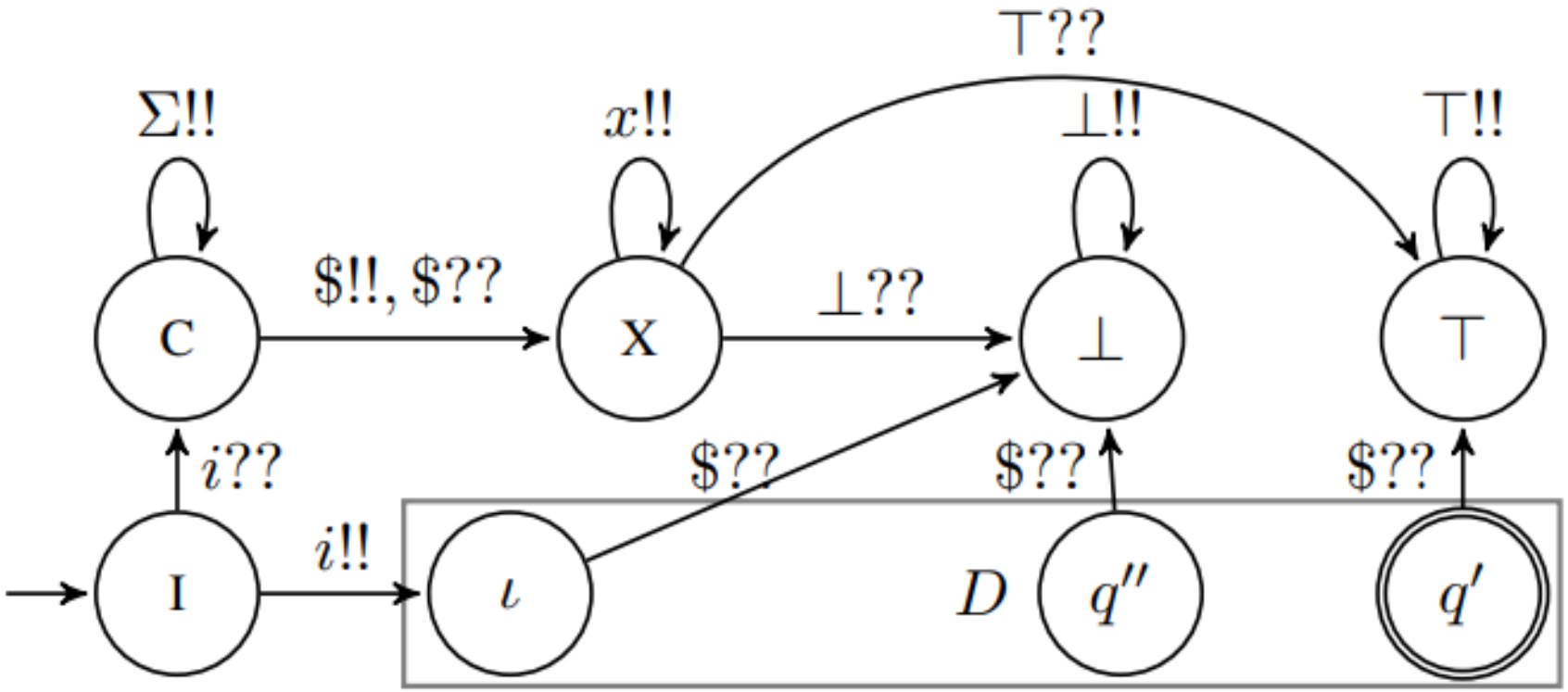
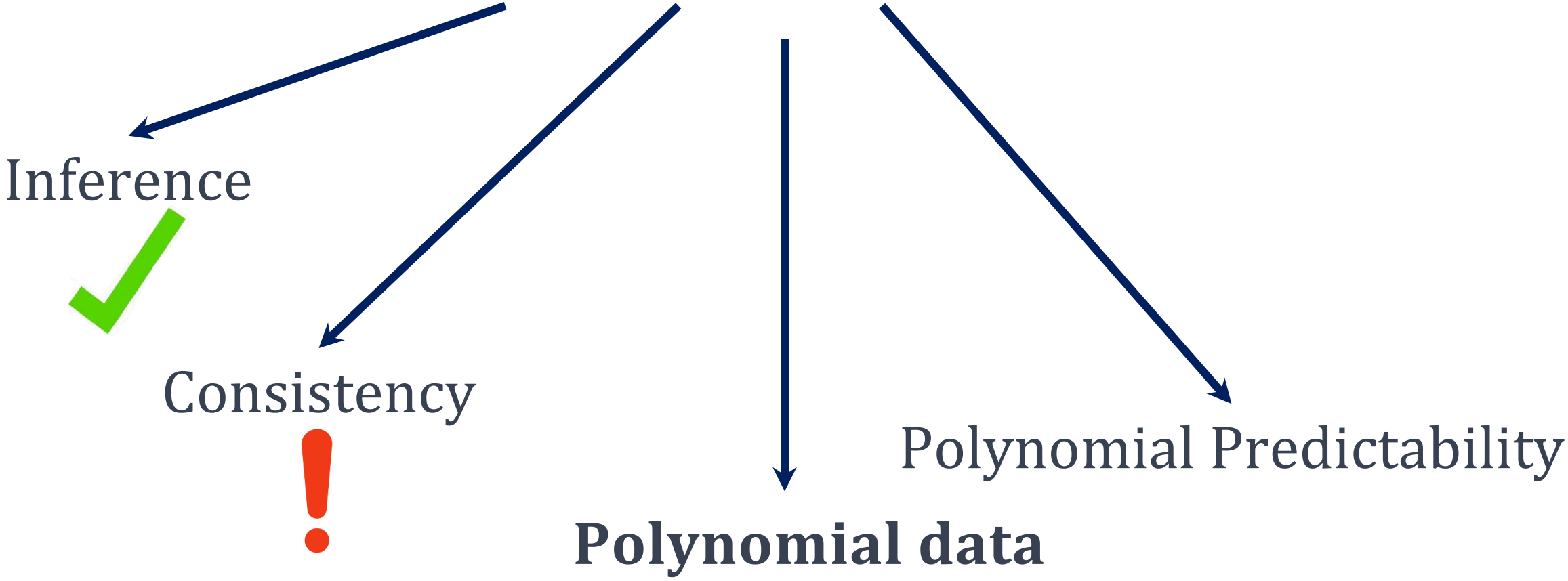


Figure 3: Reduction of DFA-consistency to BP-consistency.

Learning paradigms



Polynomial data

Is there an inference-algorithm $\mathcal{A}$ s.t. for all BP $B \in C$,

one can associate a polynomial-sized sample $\mathcal{S}_B$

so that $\mathcal{A}$ correctly infers $\mathcal{L}(B)$ from any sample

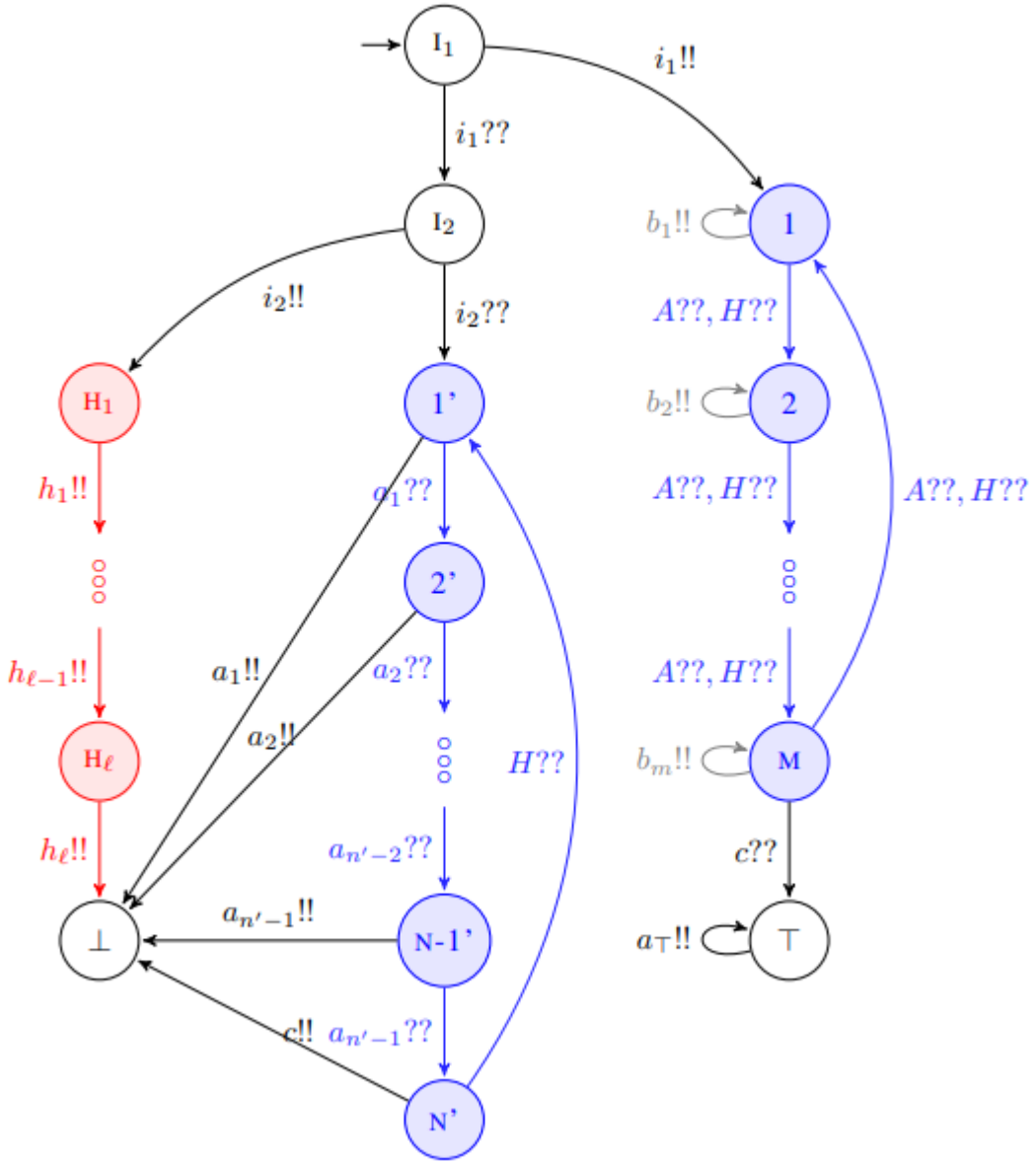
subsuming $\mathcal{S}_B$.

Recall: We mark the class of fine BPs as C.

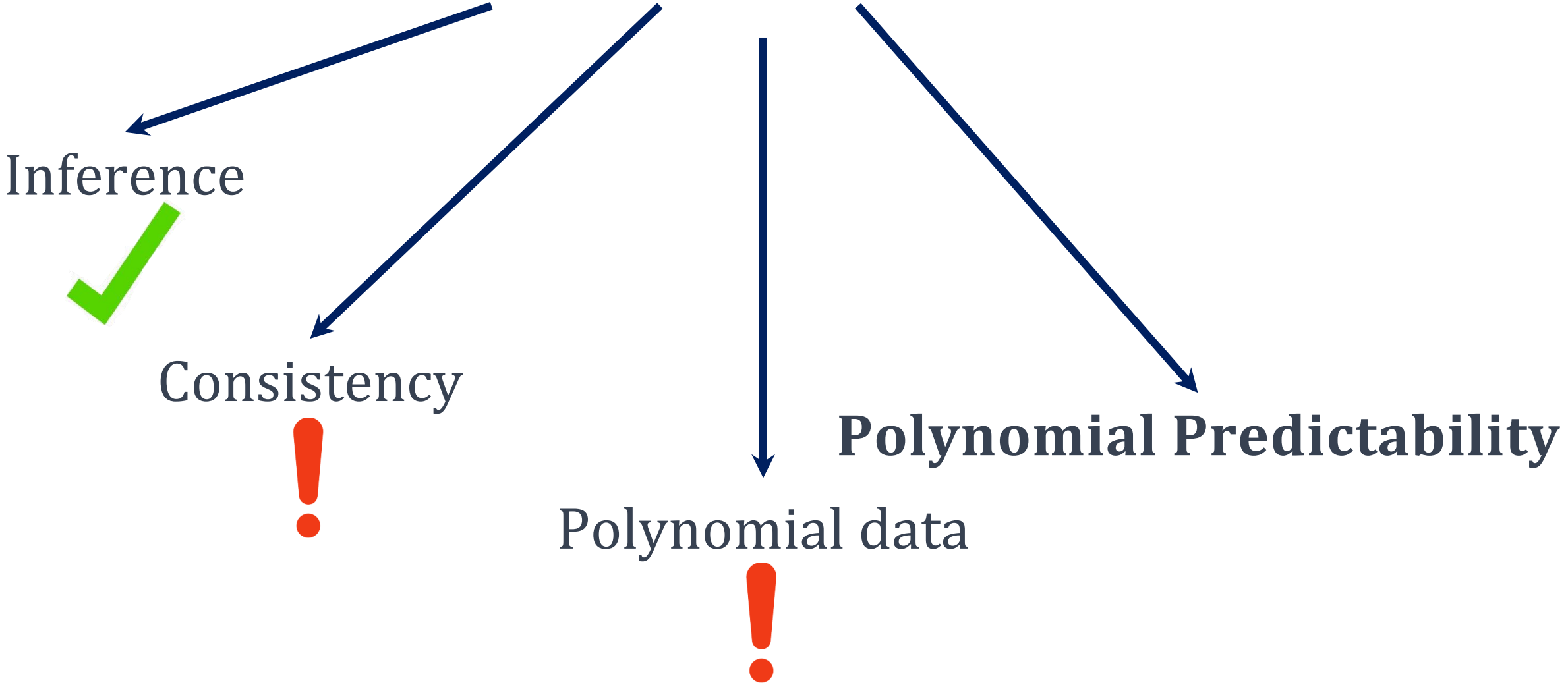
Polynomial data

We show that there exist fine BPs
for which there is no characteristic set of polynomial size.

Polynomial data



Learning paradigms



Polynomial Predictability

Can a learner correctly classify an unknown word with high probability after asking polynomially many membership queries.

Polynomial Predictability

We show that under plausible cryptography assumptions, fine BPs (thus BPs in general) are not polynomially predictable.

Polynomial Predictability

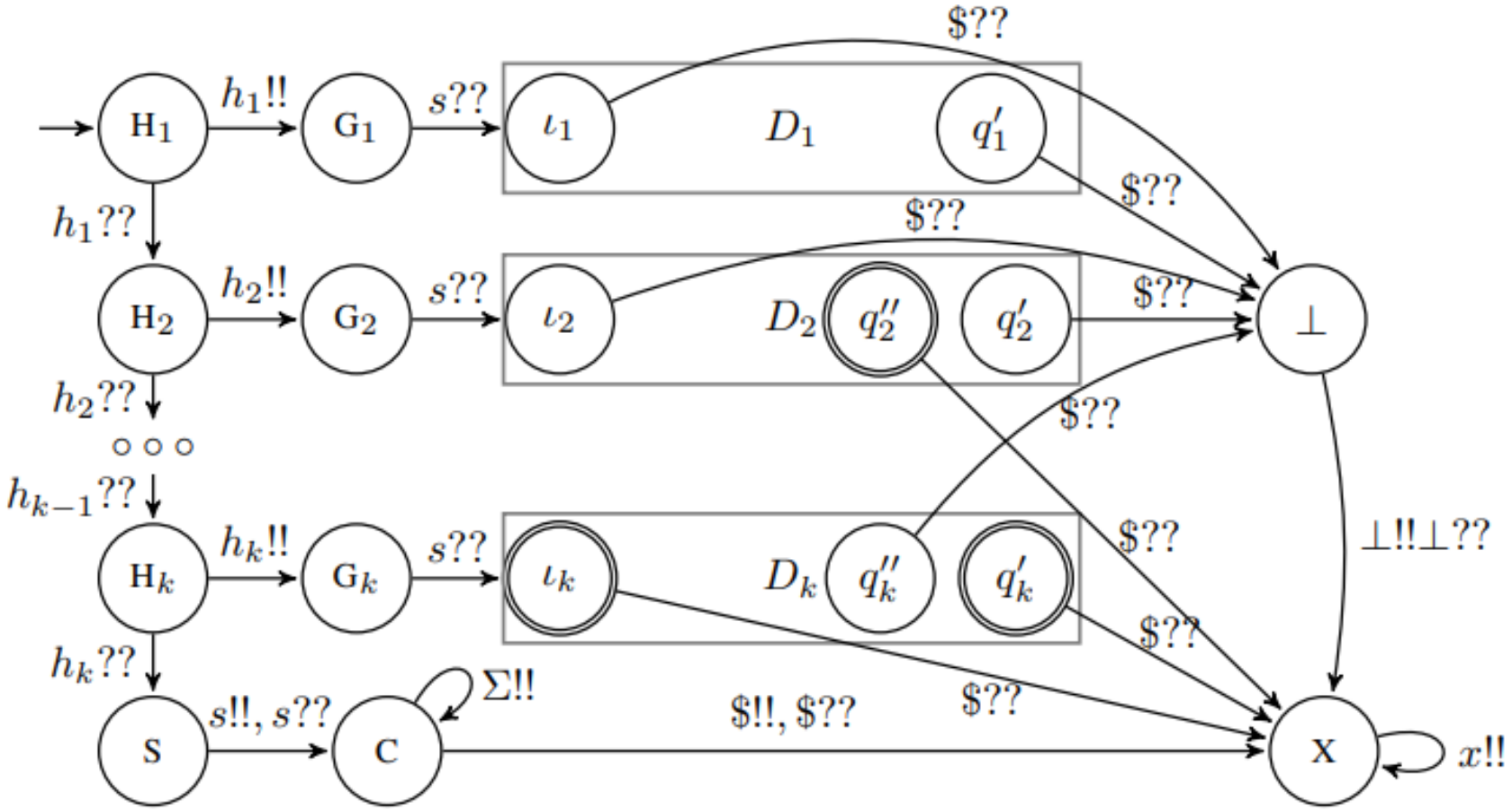
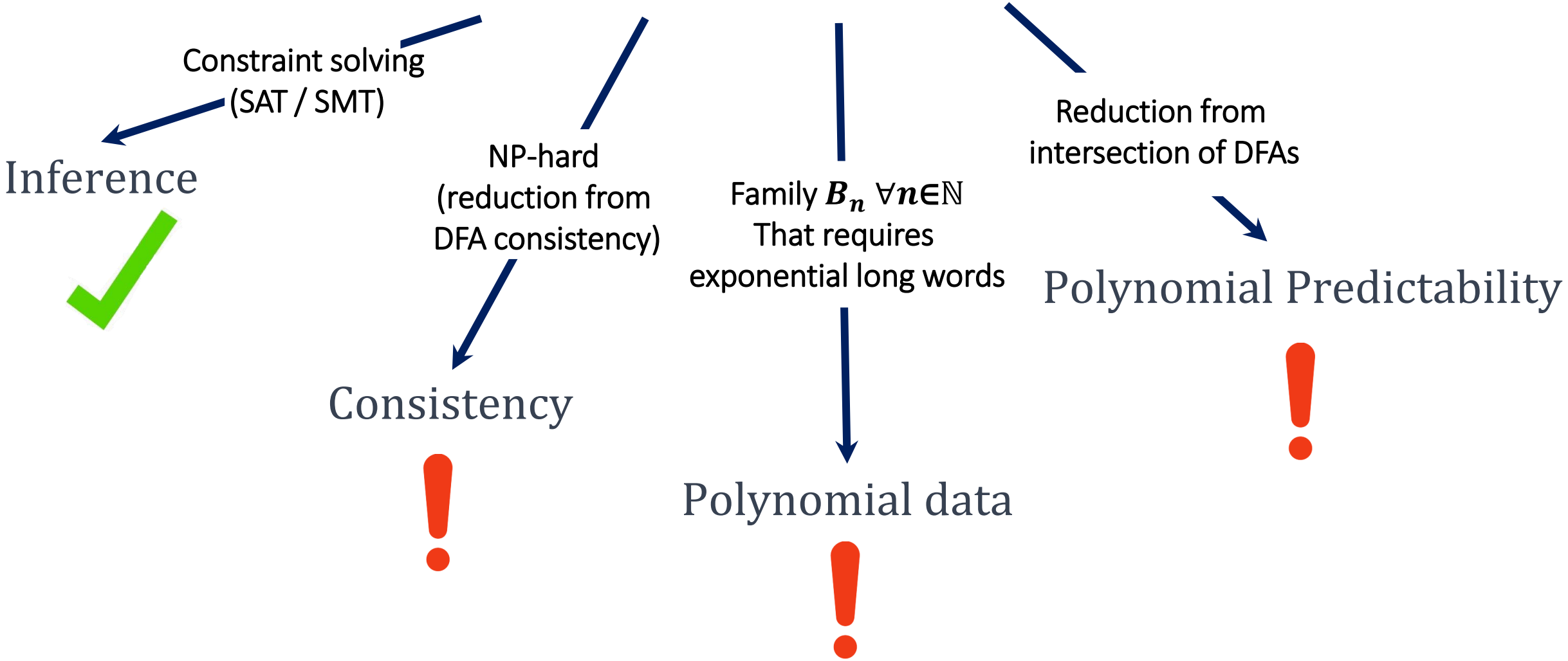


Figure 4: A BP simulating intersection of k DFAs.

Learning paradigms



Learning Broadcast Protocols, AAAI 2024

Learning Broadcast Protocols with LeoParDS, ATVA 2024

Learning paradigms

Constraint solving (SAT / SMT)

Inference

LeoParDS

NP-hard (reduction from DFA consistency)

Consistency

!

Family B_n $\forall n \in \mathbb{N}$ That requires exponential long words

Polynomial data

!

Reduction from intersection of DFAs

Polynomial Predictability

!

Learning Broadcast Protocols, AAAI 2024

Learning Broadcast Protocols with LeoParDS, ATVA 2024

Questions?

Constraint solving
(SAT / SMT)

Inference

NP-hard
(reduction from
DFA consistency)

Consistency

Family B_n $\forall n \in \mathbb{N}$
That requires
exponential long words

Polynomial data

Reduction from
intersection of DFAs

Polynomial Predictability

Learning Broadcast Protocols, AAAI 2024
Learning Broadcast Protocols with LeoParDS, ATVA 2024