CAUSAL ANALYSIS OF RULE-BASED MODELS THROUGH

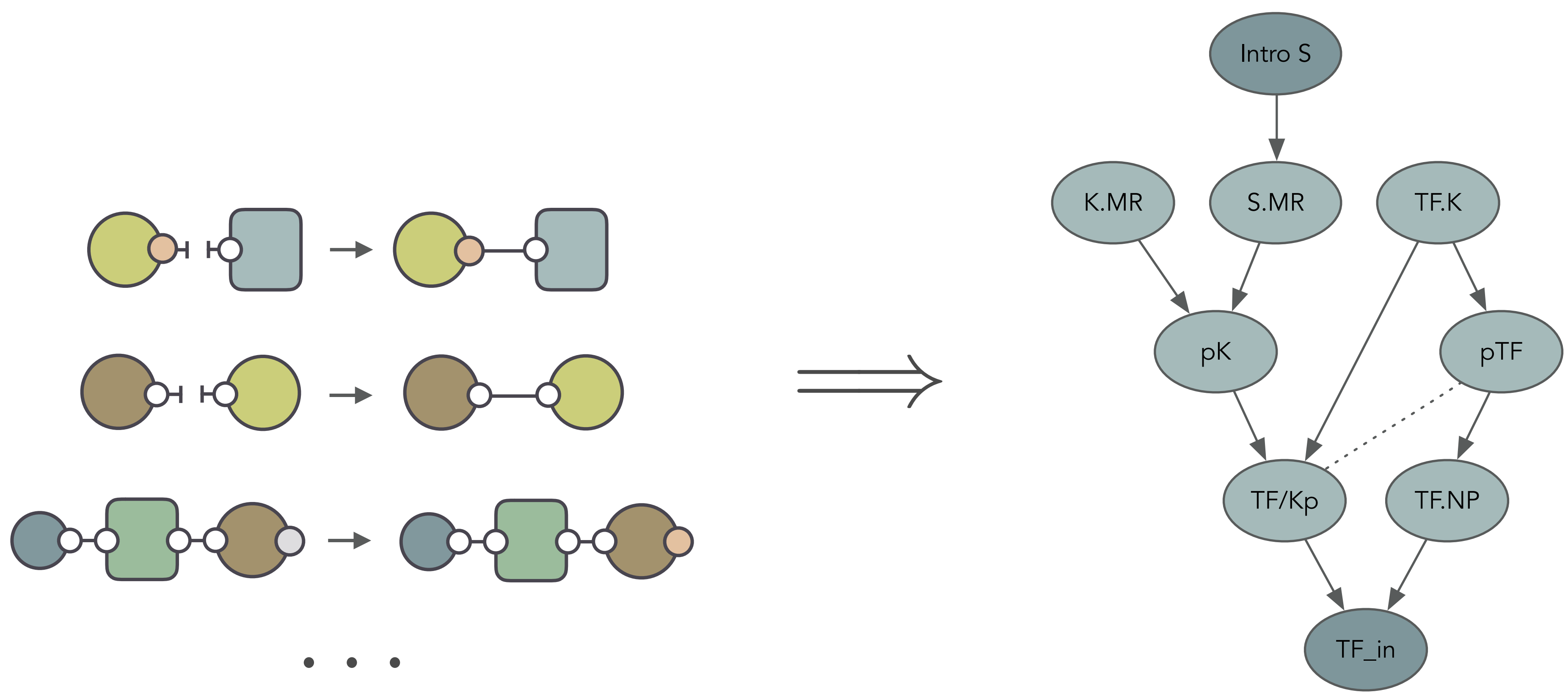
COUNTERFACTUAL REASONING

***Jonathan Laurent**, Jean Yang (Carnegie Mellon University),*

Walter Fontana (Harvard Medical School)

CAUSAL ANALYSIS

Some techniques have been developed to analyze the **causal structure** of rule-based models [Feret, Fontana and Krivine].

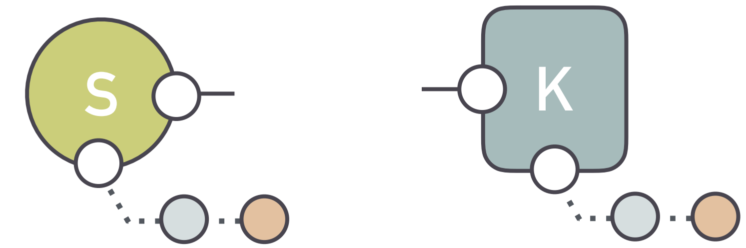


They take advantage of the structure of the rules to:

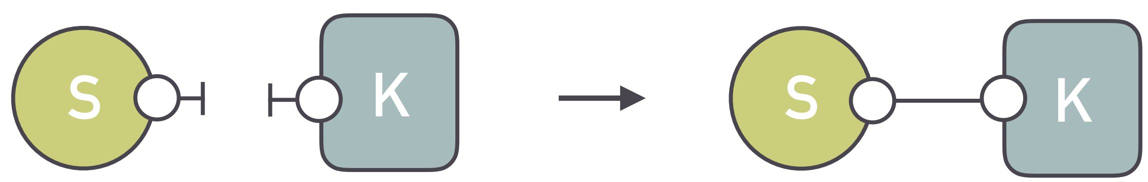
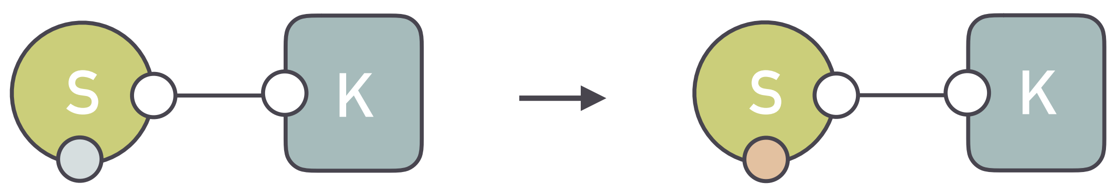
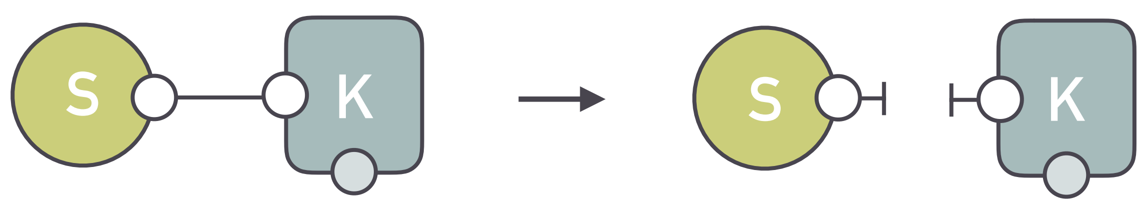
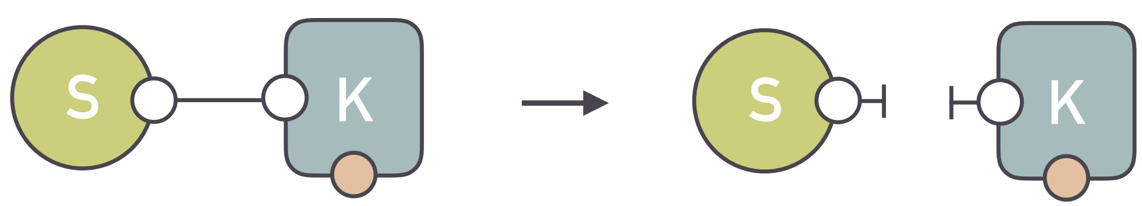
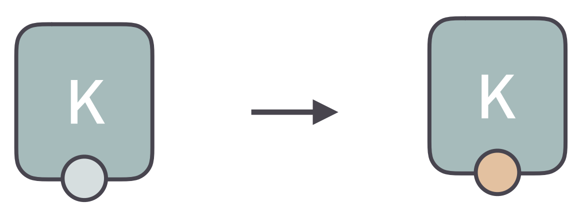
- slice simulation traces into minimal subsets of **necessary events**
- highlight **causal influences** between non-concurrent events

A MOTIVATING EXAMPLE

Here is a toy Kappa model that represents one step of a phosphorylation cascade:

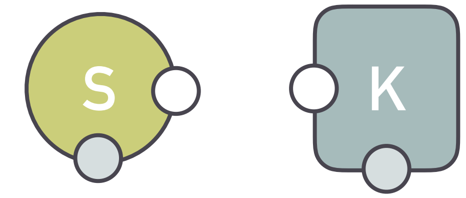


Substrate *Kinase*

b		@ *rate*
p		@ *rate*
u		@ *fast_rate*
*u**		@ *slow_rate*
pk		@ *rate*

A MOTIVATING EXAMPLE

Starting from the following initial mixture, how does rule ***p*** get triggered ?

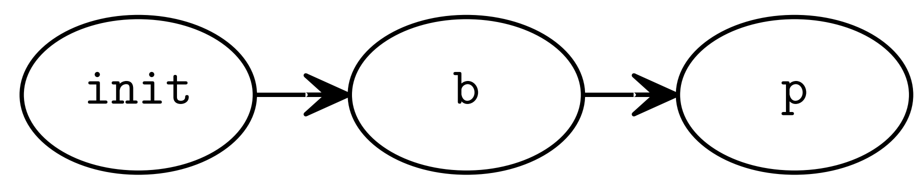


Initial mixture

Here is a stochastic simulation of the system:

init	*b*	*u*	*pk*	*b*	***p***	*u**	…

Existing causal analysis techniques would provide the following narrative:



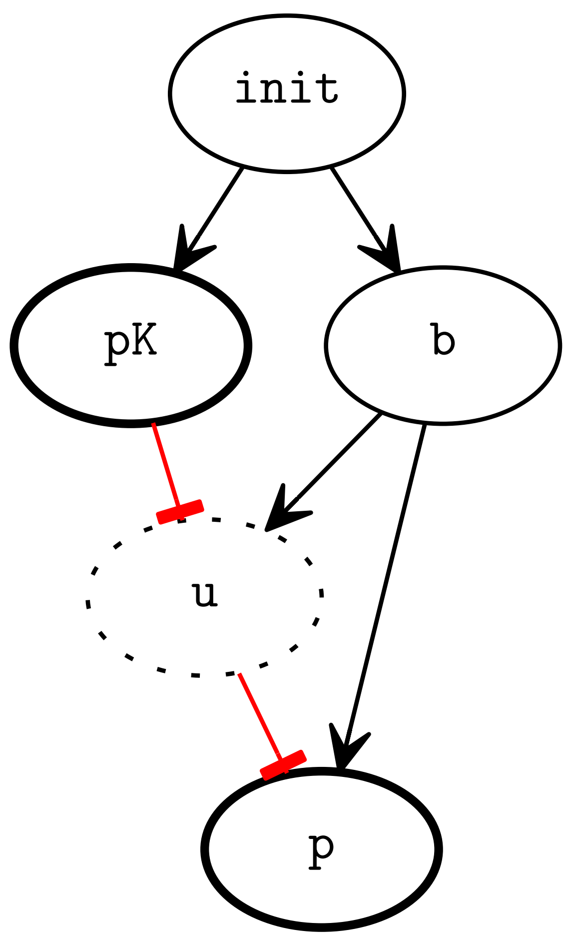
This seems wrong because it downplays the role of event ***pk***. Indeed:

> *Event p* ***would probably not*** *have happened* ***had*** *pk not happened, being prevented by an early unbinding event.*

Counterfactual

A MOTIVATING EXAMPLE

A better causal explanation for
pk would look like this:



Contributions

In this work, we make the following contributions:

- We propose a semantics for **counterfactual statements** in Kappa.
- We provide an algorithm to **evaluate** such statements efficiently.
- We show how inhibition arrows can be used to **explain** counterfactual experiments.

A MONTE CARLO SEMANTICS FOR KAPPA

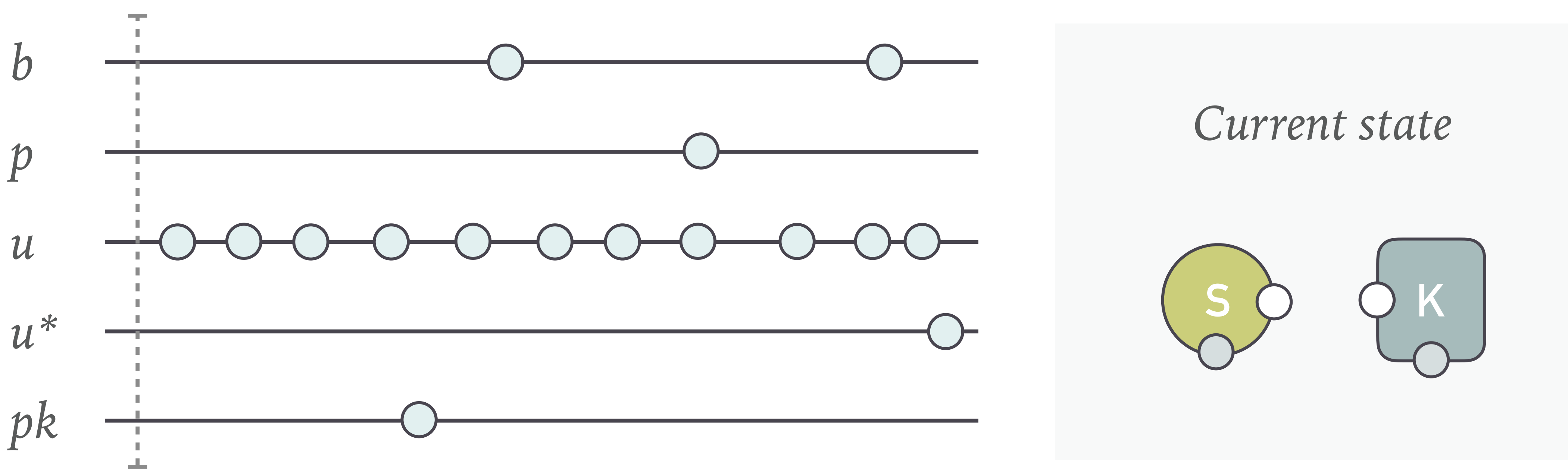
A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.

A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

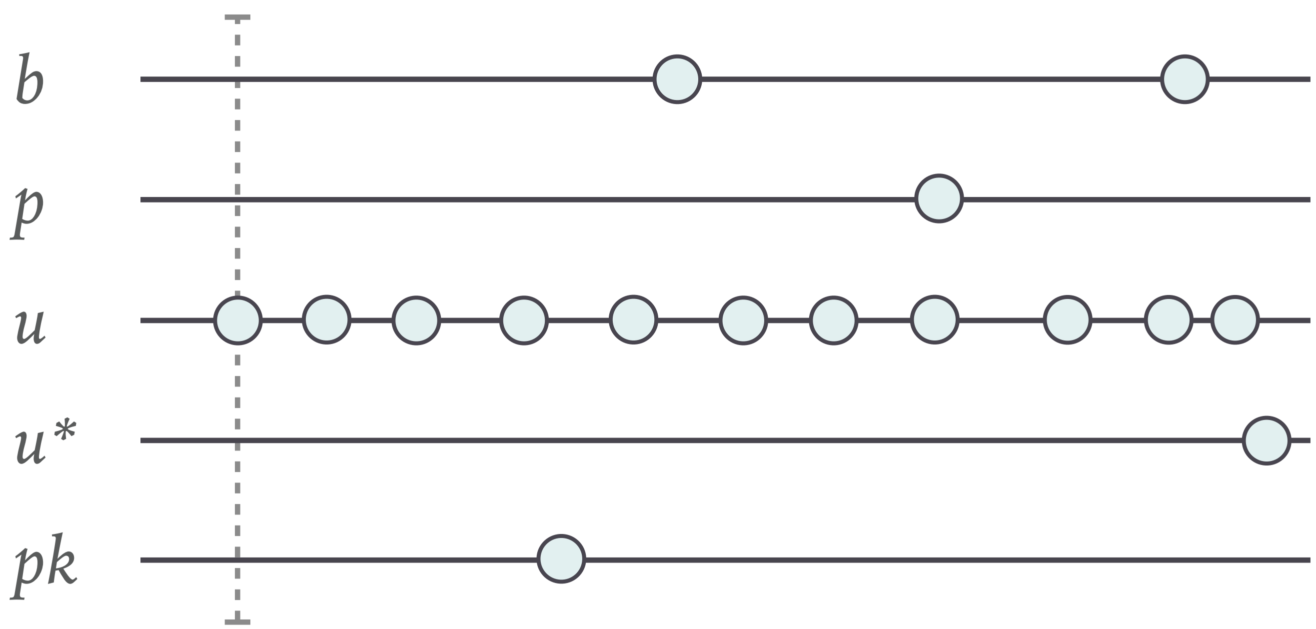
To every such potential event, we associate a **Poisson process**.

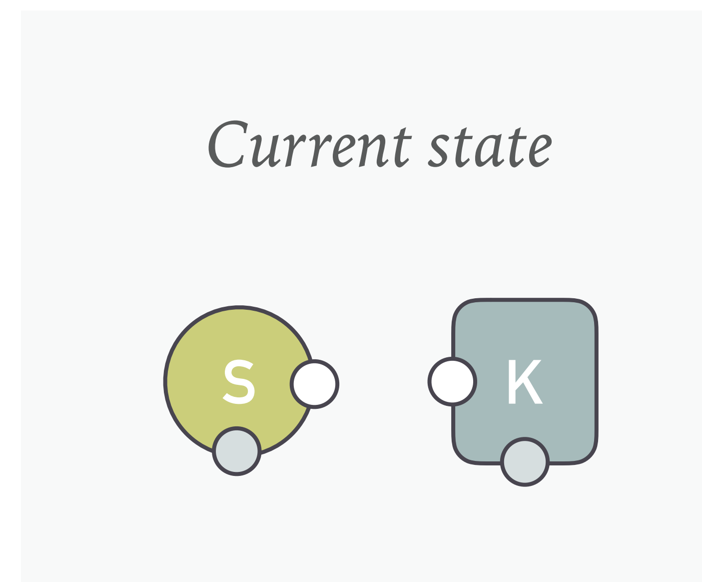


A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.

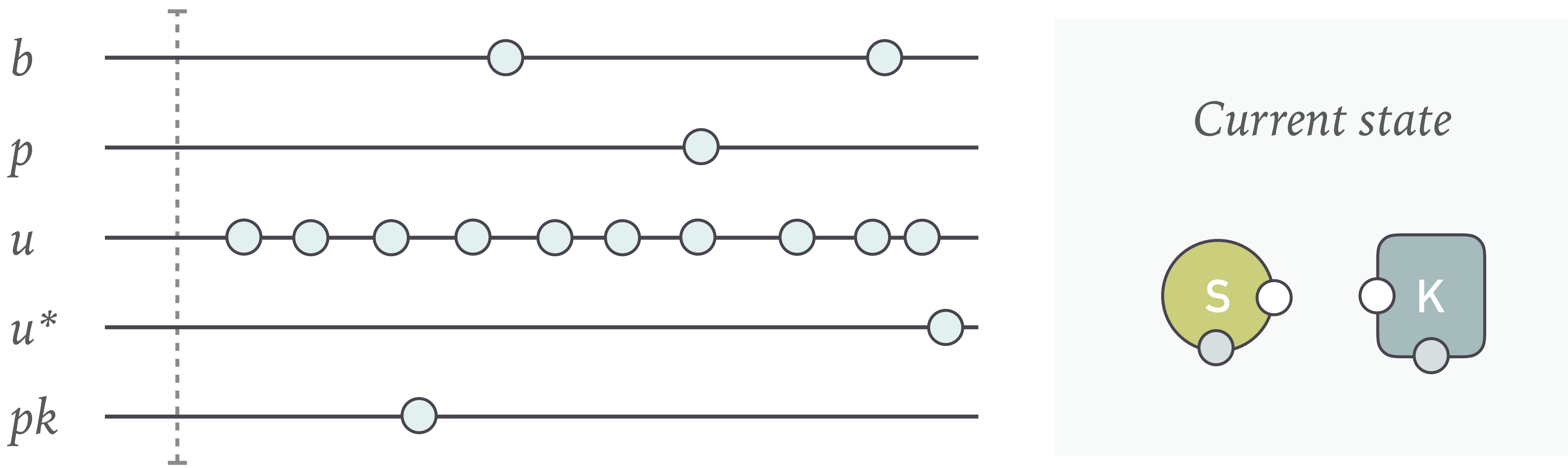




A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

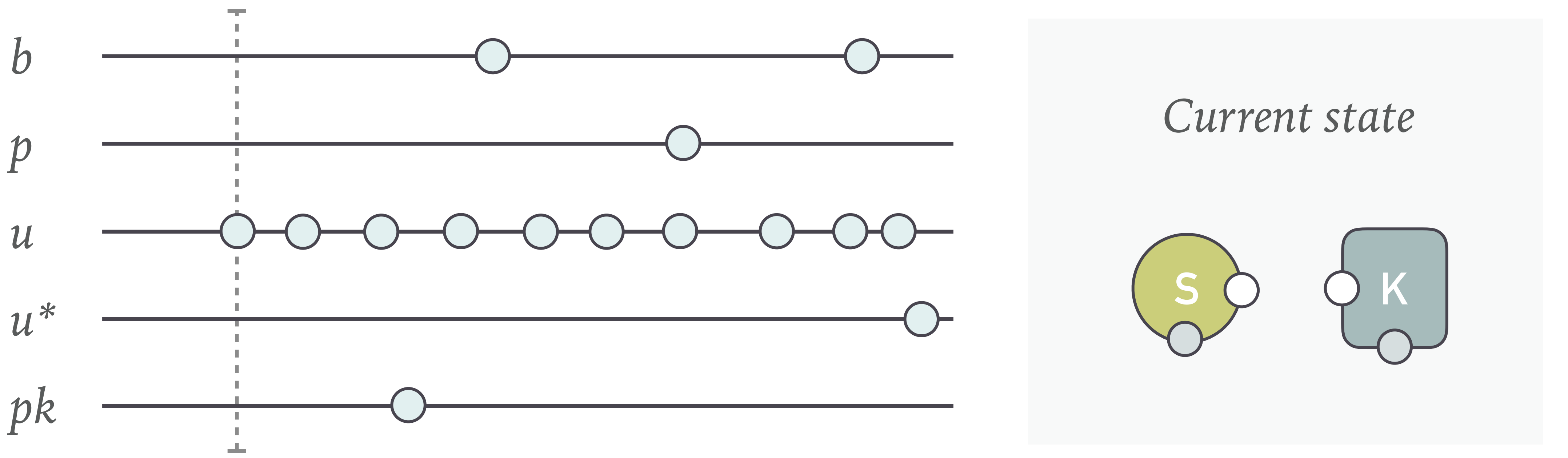
To every such potential event, we associate a **Poisson process**.



A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

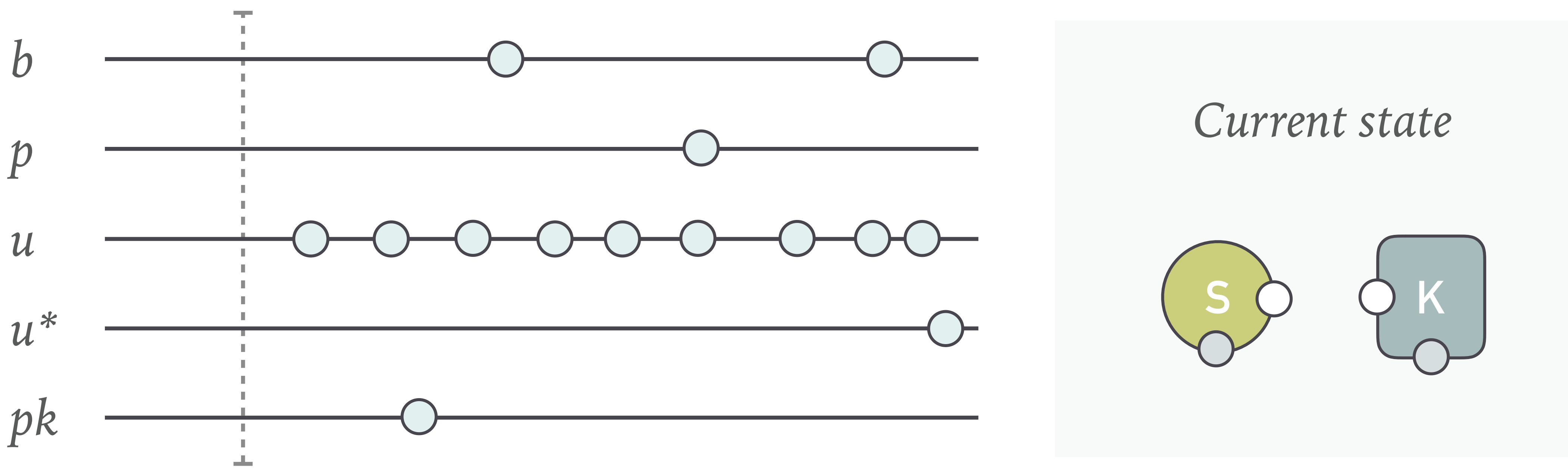
To every such potential event, we associate a **Poisson process**.



A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

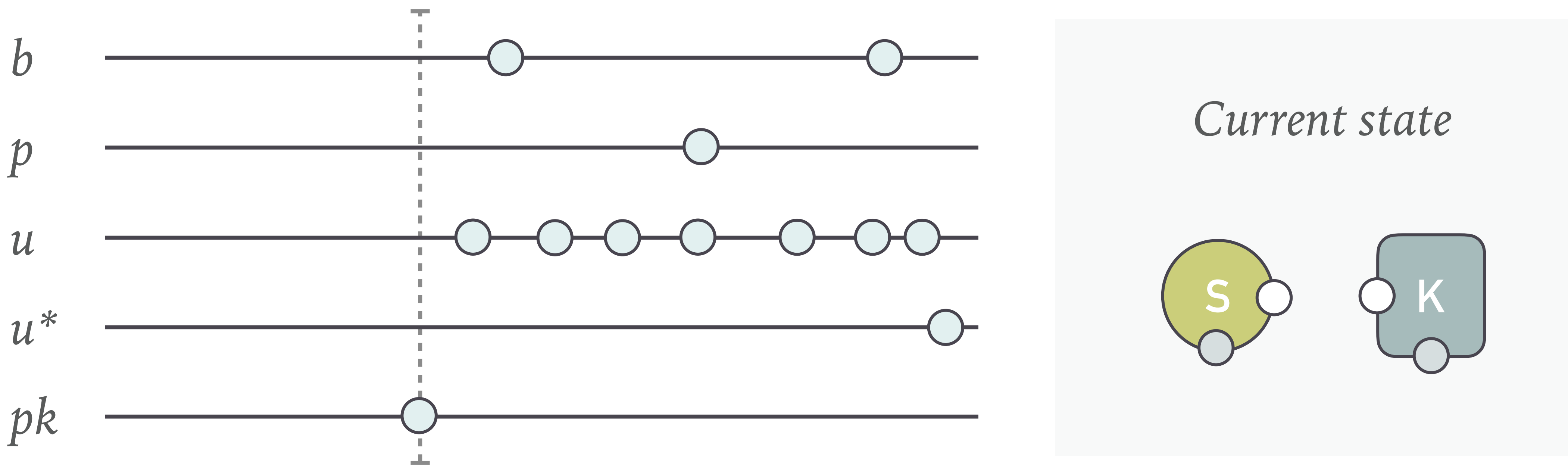
To every such potential event, we associate a **Poisson process**.



A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

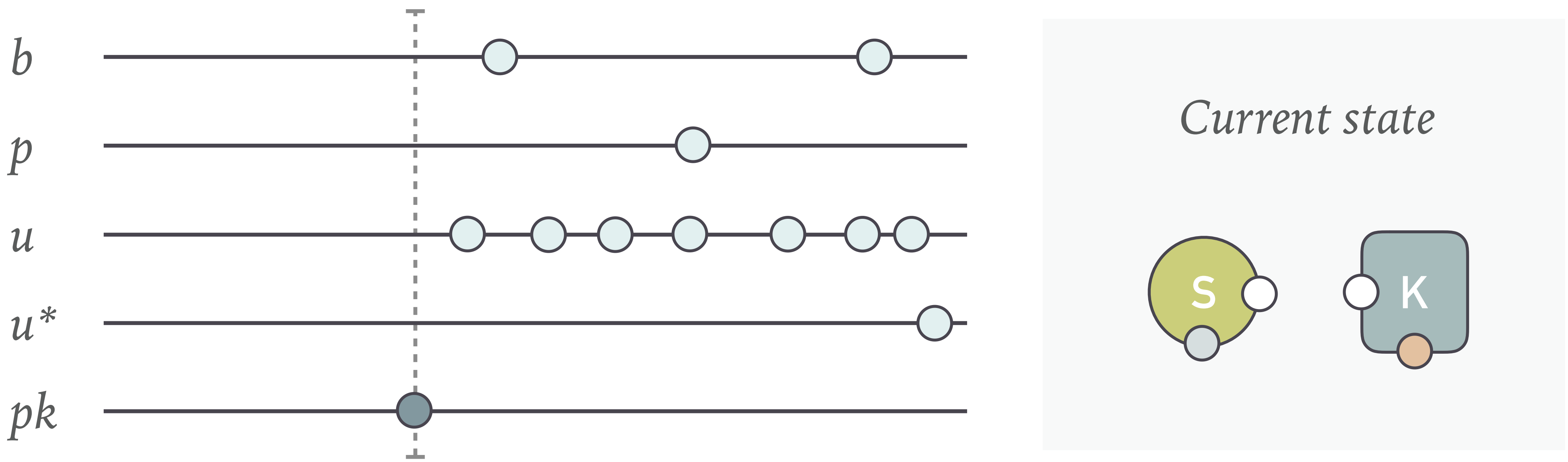
To every such potential event, we associate a **Poisson process**.



A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

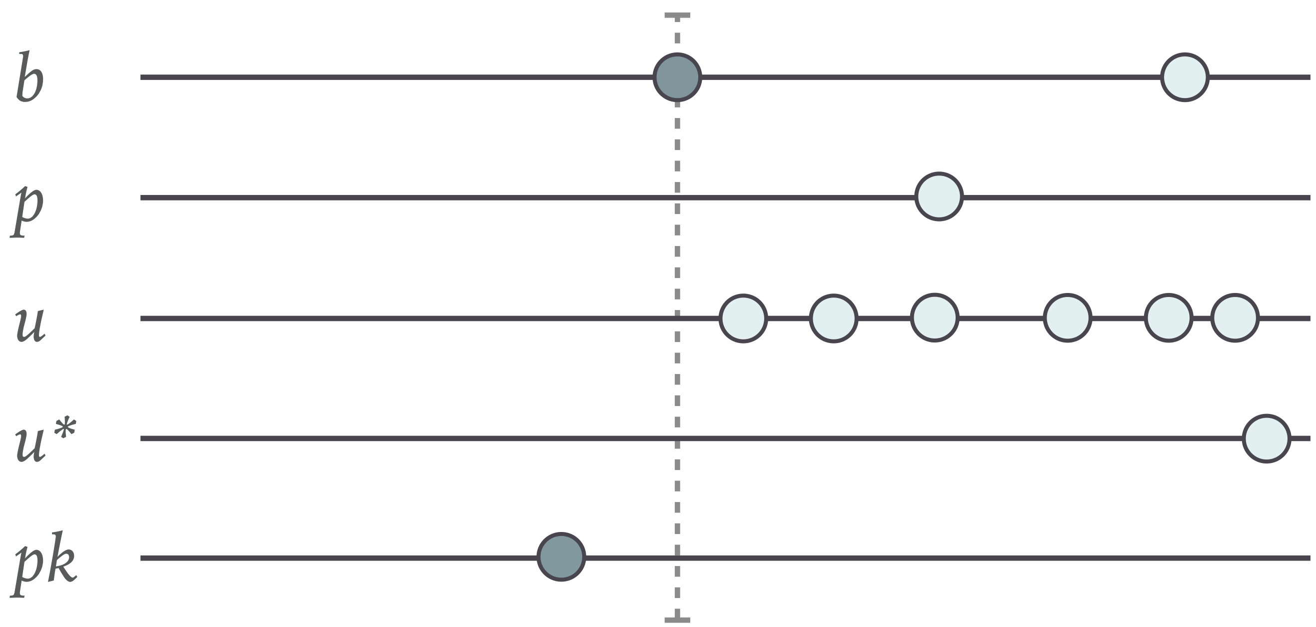
To every such potential event, we associate a **Poisson process**.

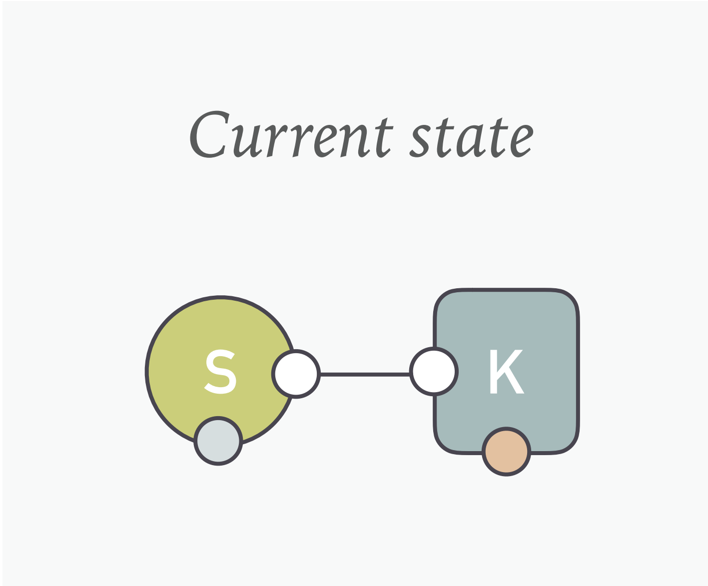


A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.

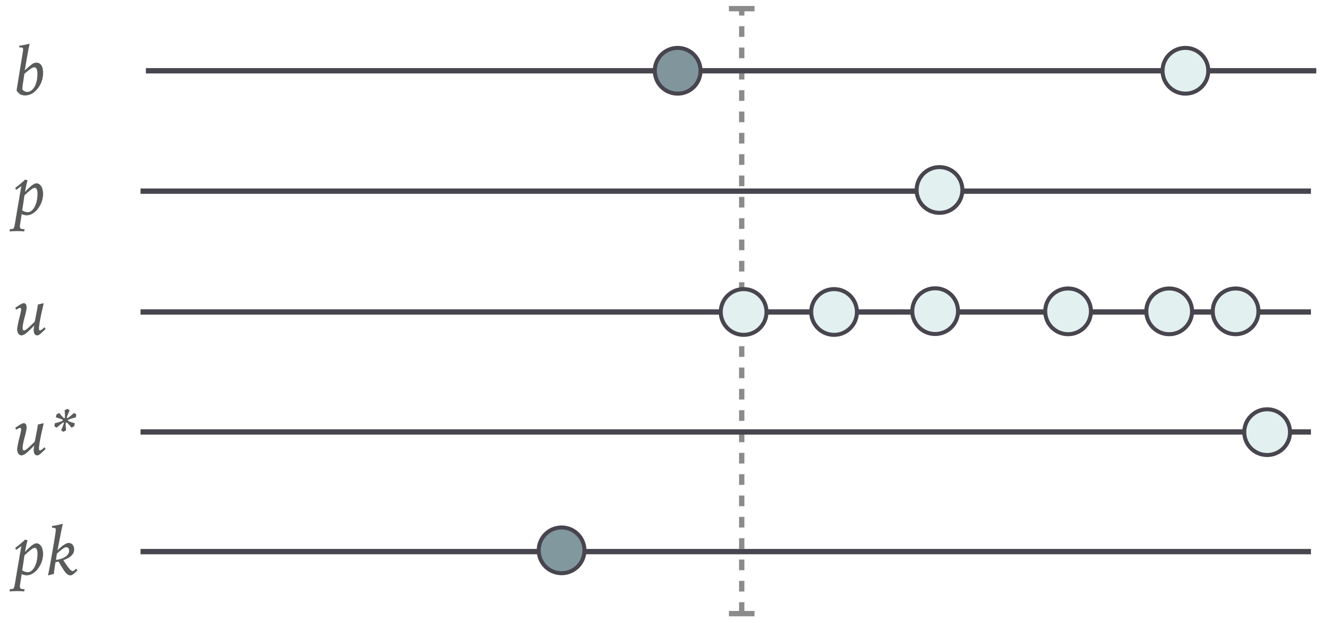


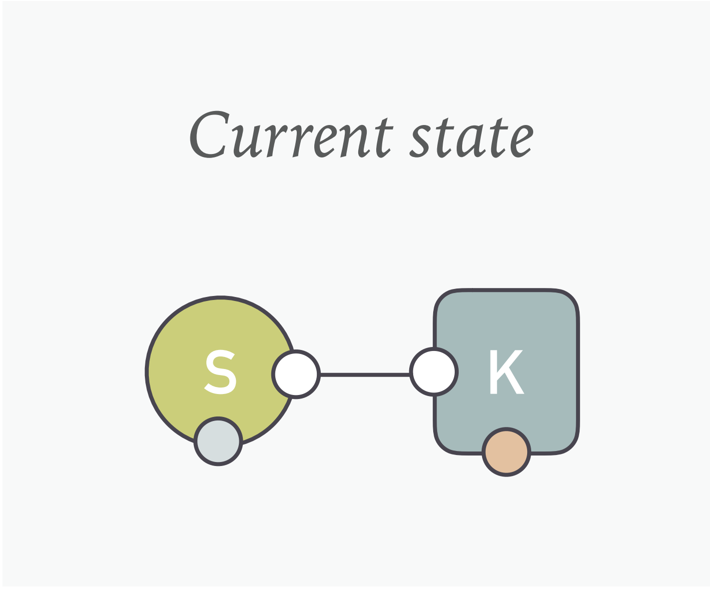


A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.

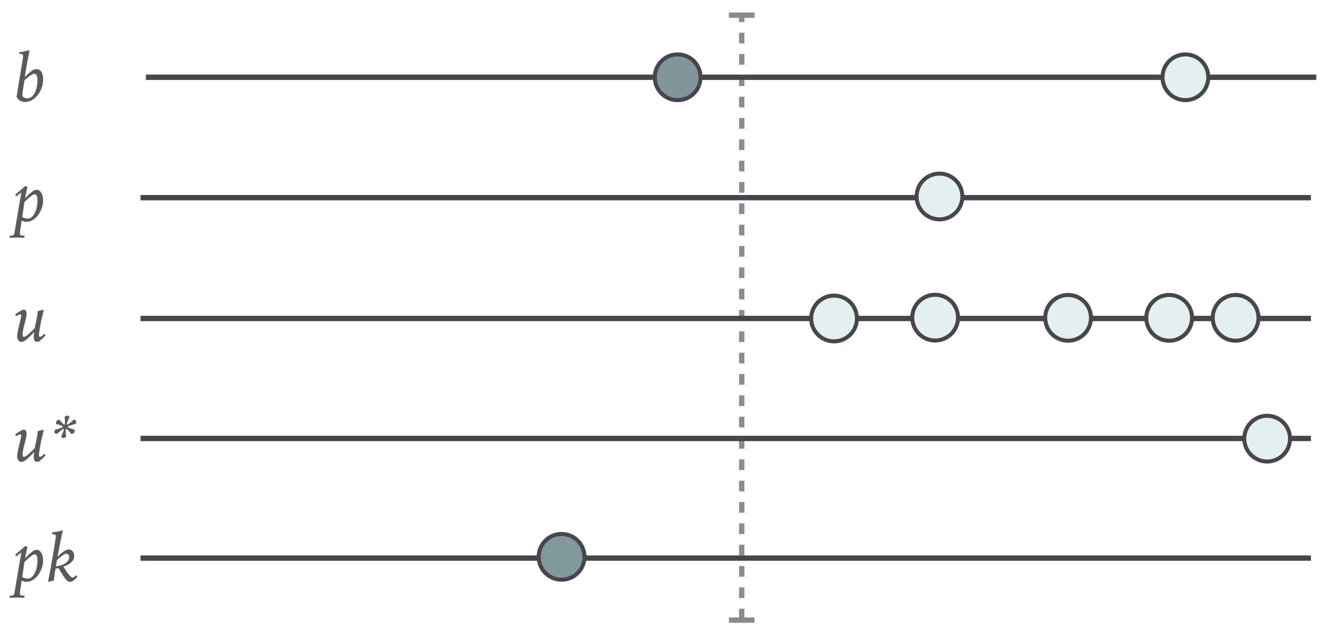


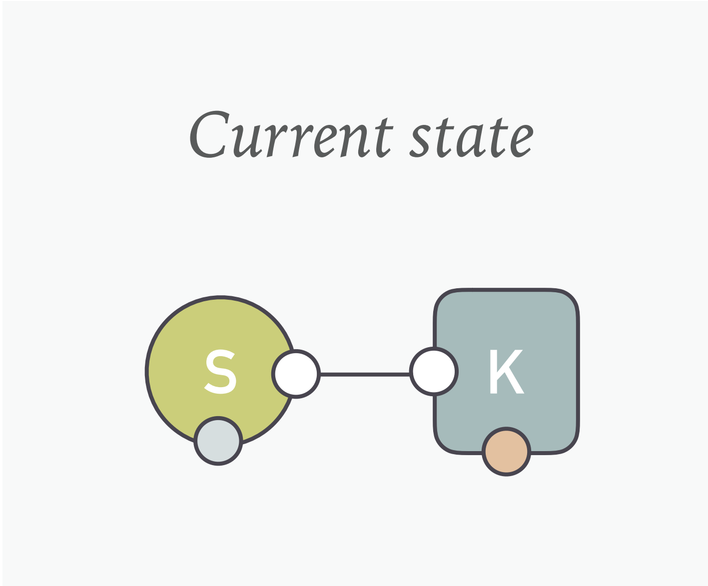


A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.

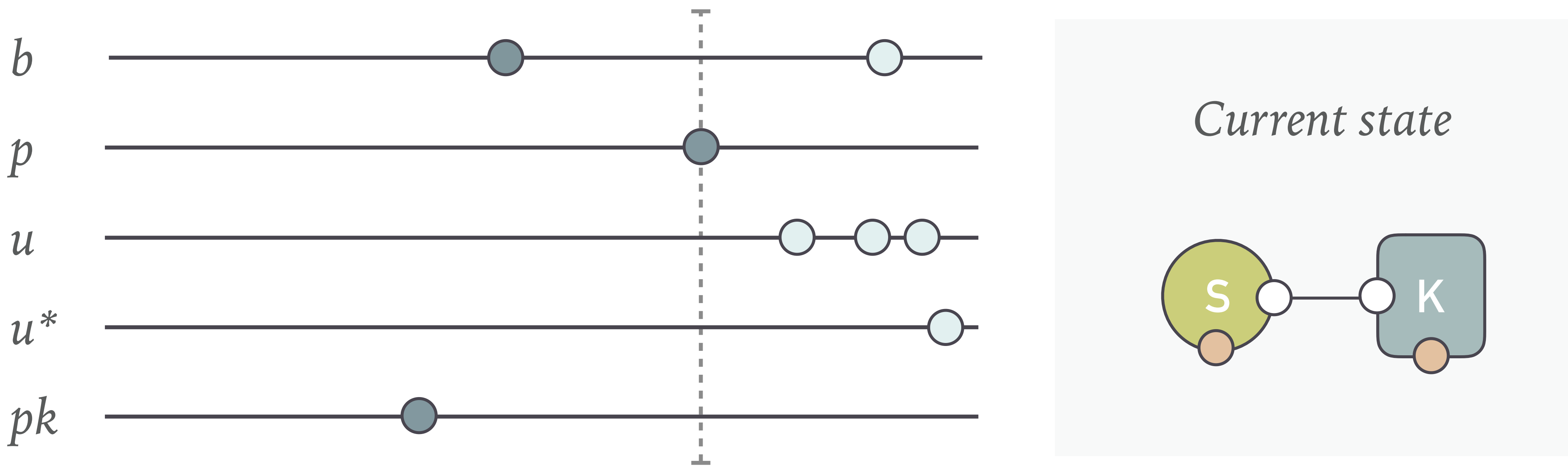




A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

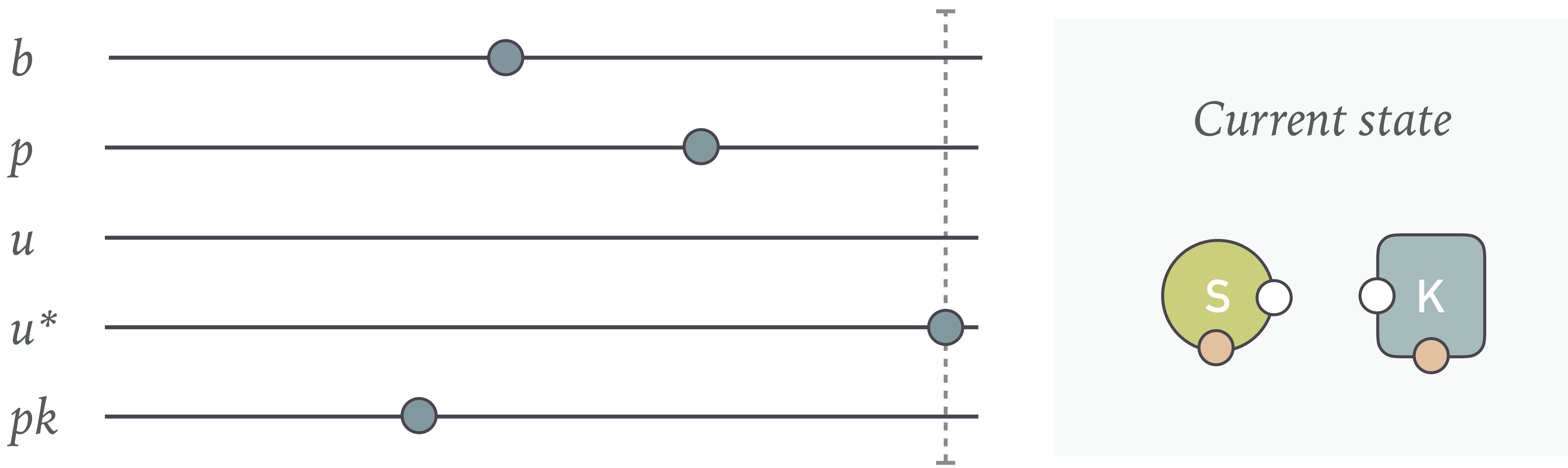
To every such potential event, we associate a **Poisson process**.



A MONTE CARLO SEMANTICS FOR KAPPA

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

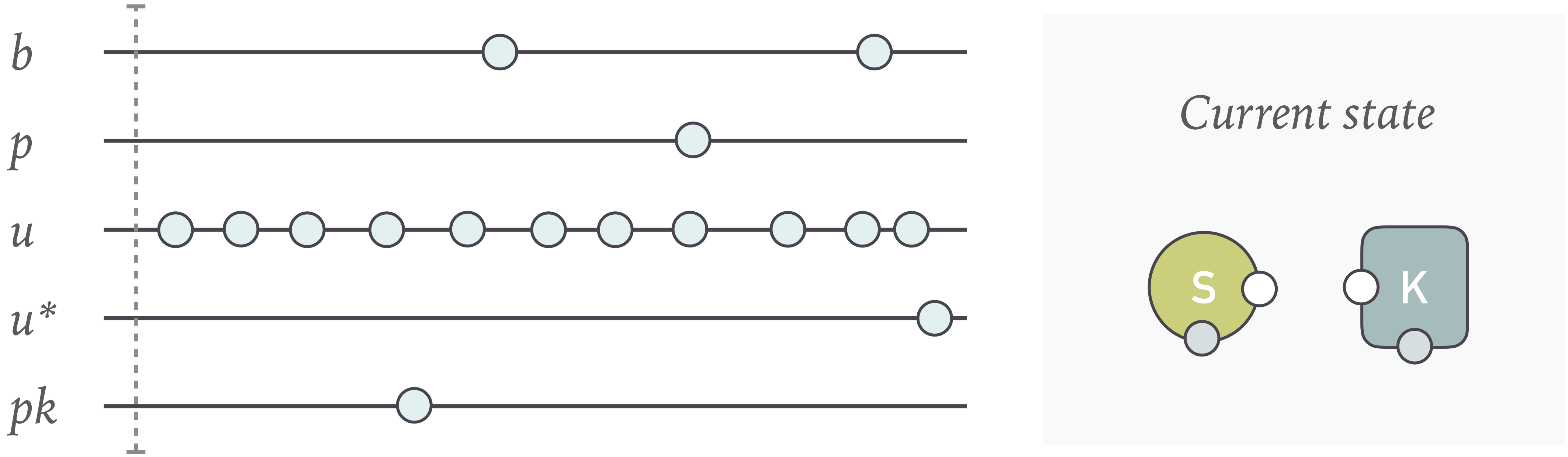
To every such potential event, we associate a **Poisson process**.



SIMULATING MODULO AN INTERVENTION

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.

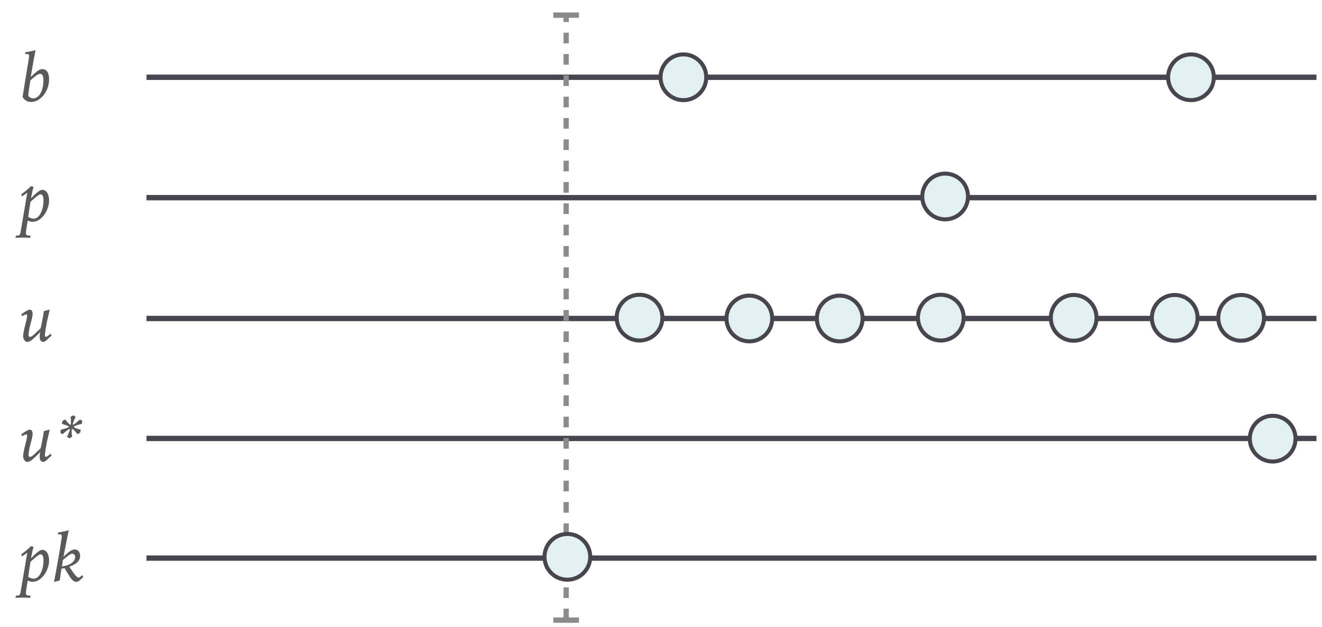


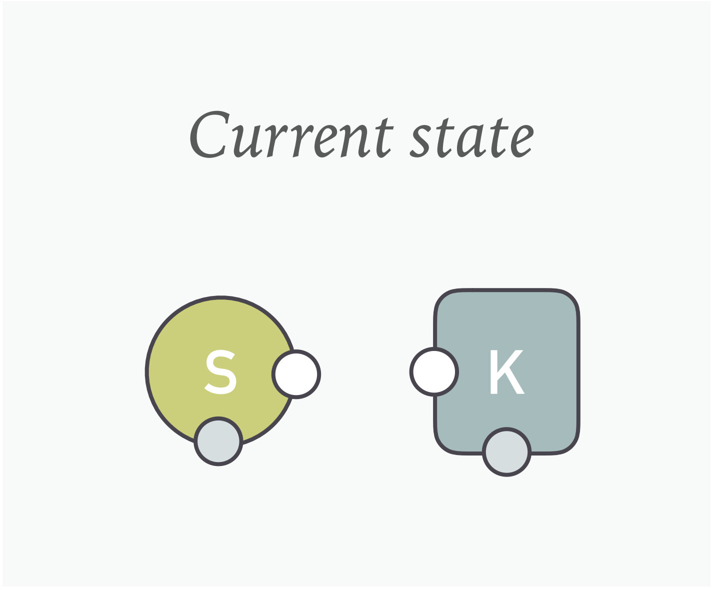
An **intervention** ι is defined as a predicate that specifies what events should be blocked. Let's simulate again, blocking the triggering of ***pk***.

SIMULATING MODULO AN INTERVENTION

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.



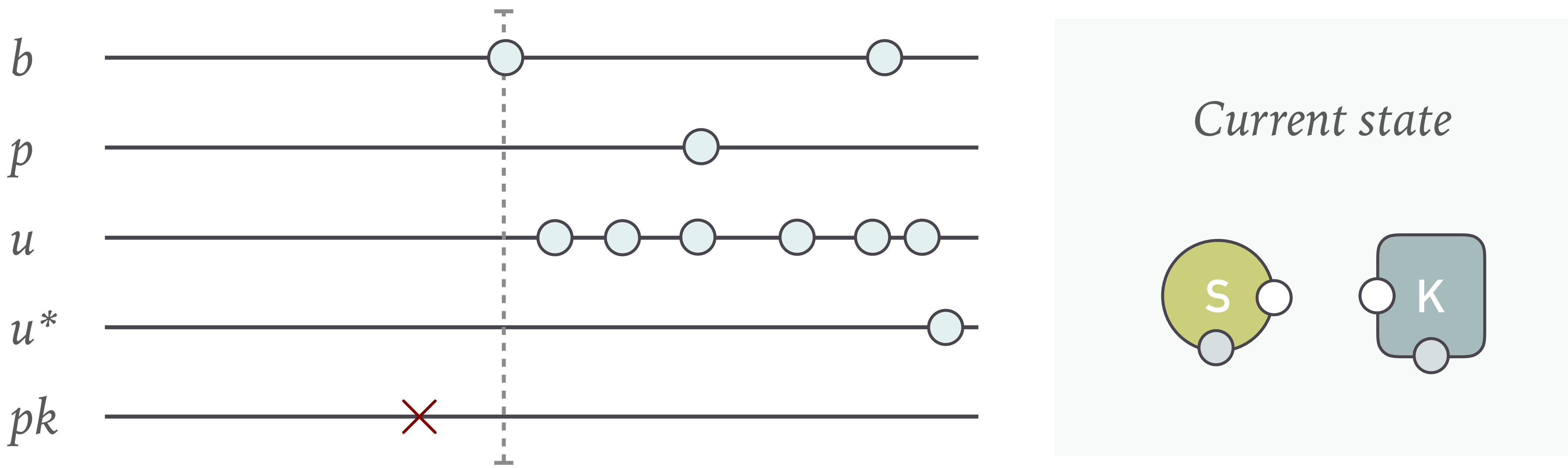


An **intervention** ι is defined as a predicate that specifies what events should be blocked. Let's simulate again, blocking the triggering of ***pk***.

SIMULATING MODULO AN INTERVENTION

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.

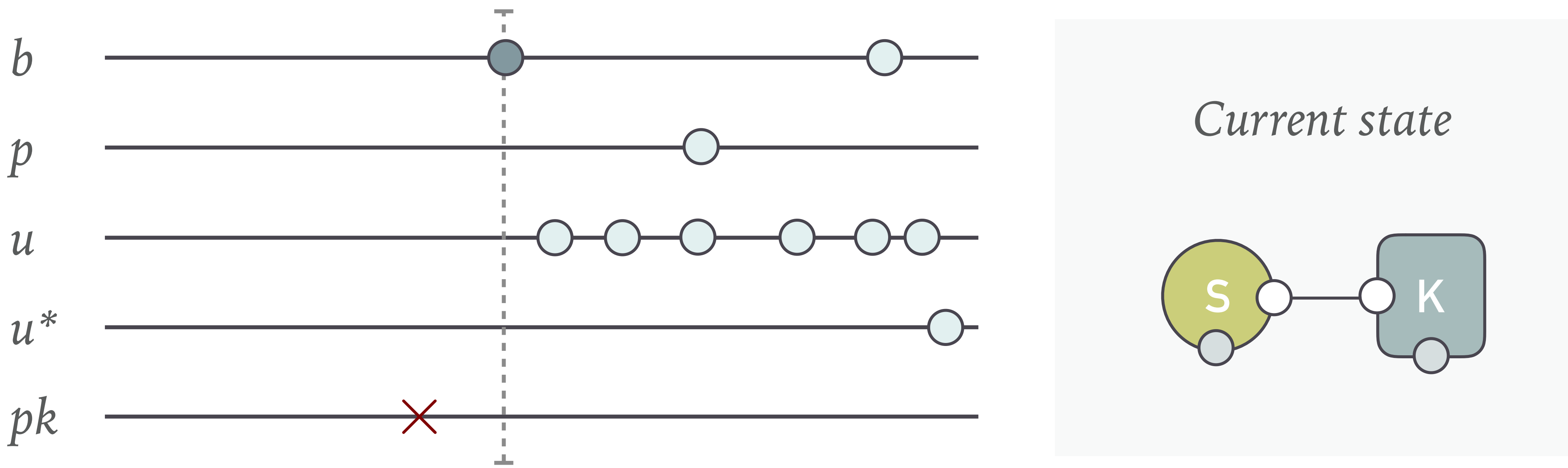


An **intervention** ι is defined as a predicate that specifies what events should be blocked. Let's simulate again, blocking the triggering of ***pk***.

SIMULATING MODULO AN INTERVENTION

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.

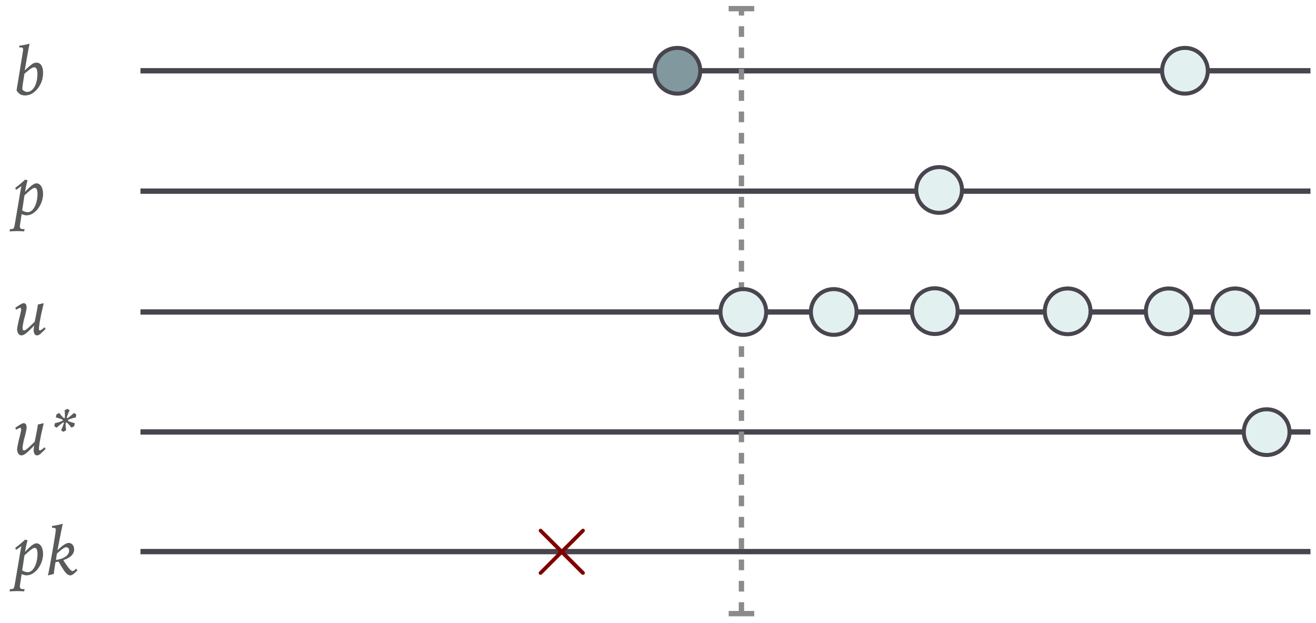


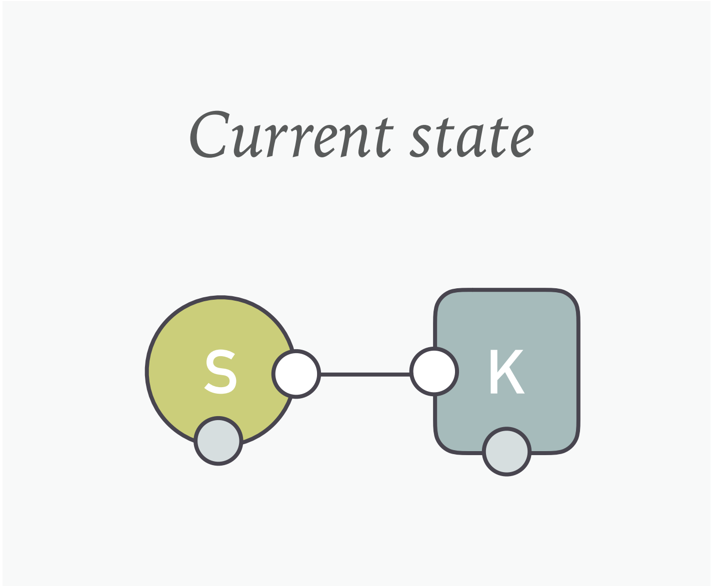
An **intervention** *ı* is defined as a predicate that specifies what events should be blocked. Let's simulate again, blocking the triggering of ***pk***.

SIMULATING MODULO AN INTERVENTION

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.



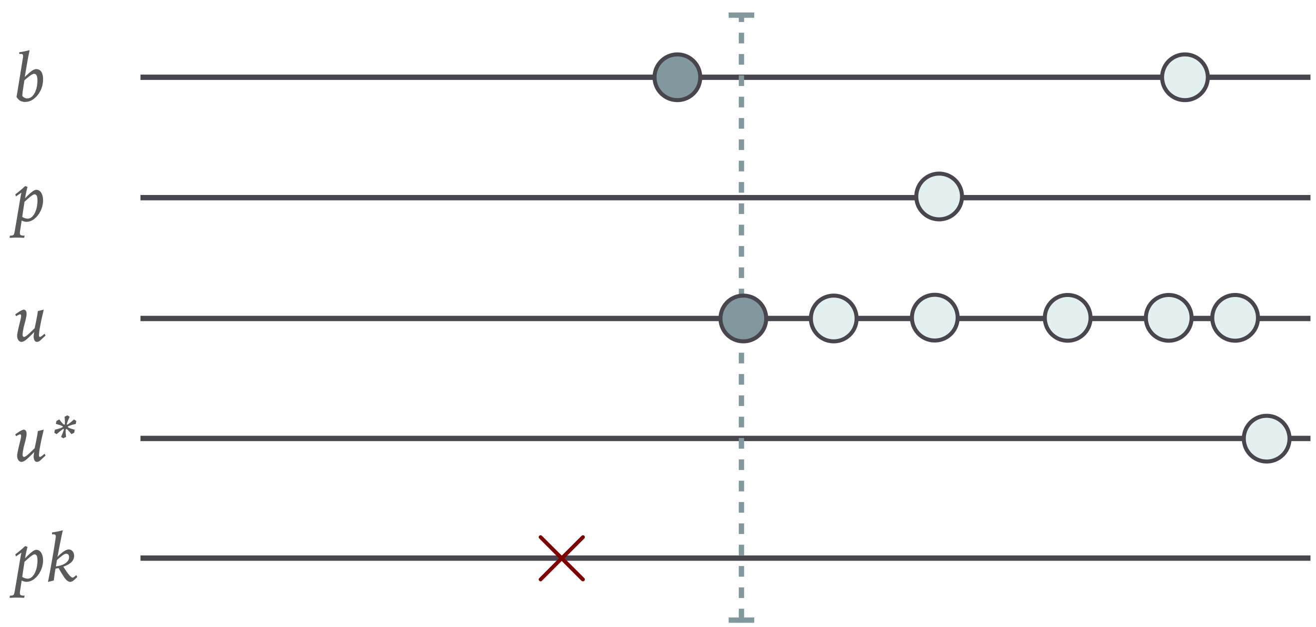


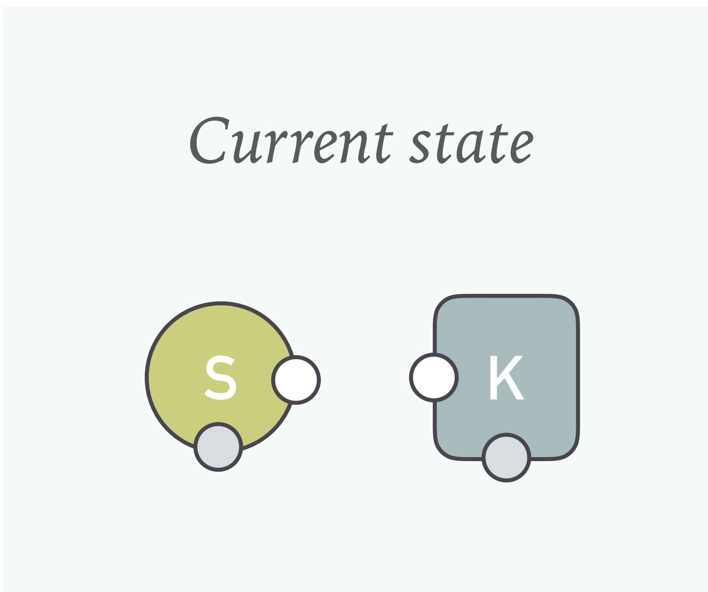
An **intervention** ι is defined as a predicate that specifies what events should be blocked. Let's simulate again, blocking the triggering of ***pk***.

SIMULATING MODULO AN INTERVENTION

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.



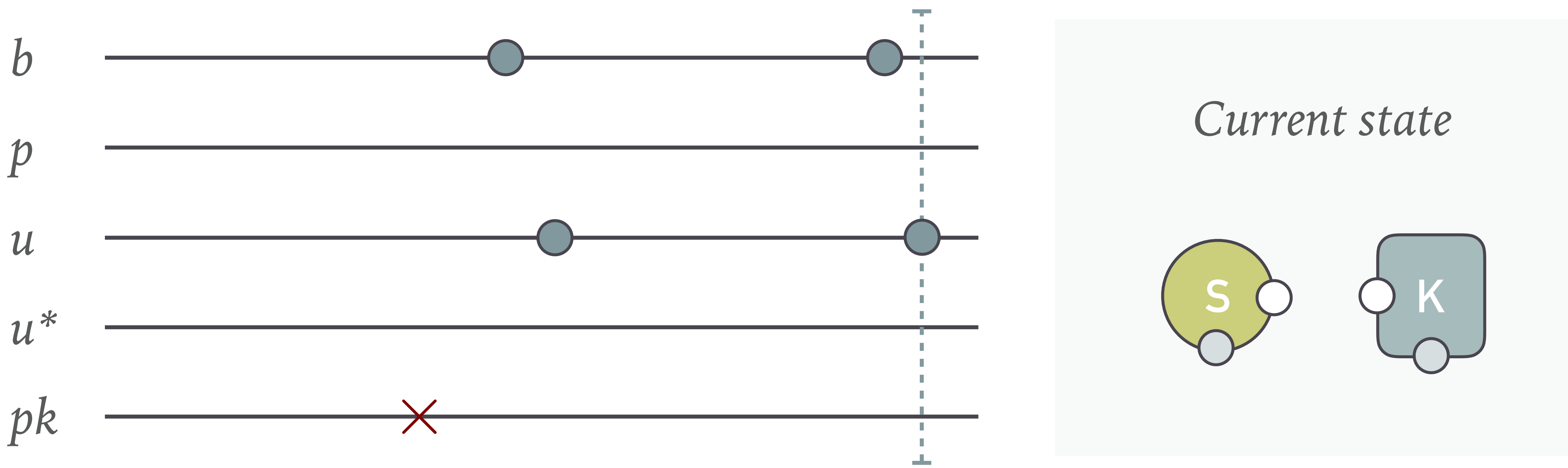


An **intervention** ι is defined as a predicate that specifies what events should be blocked. Let's simulate again, blocking the triggering of ***pk***.

SIMULATING MODULO AN INTERVENTION

A **potential event** is given by a rule *r* along with an injective mapping from the agents of *r* to global agents.

To every such potential event, we associate a **Poisson process**.



An **intervention** ι is defined as a predicate that specifies what events should be blocked. Let's simulate again, blocking the triggering of ***pk***.

COUNTERFACTUAL STATEMENTS

If we write:

T	Random variable corresponding to a simulation trace
$\hat{T}_\iota$	Simulation trace modulo intervention ι

The probability that a predicate Ψ would have been true on trace τ had intervention ι happened is defined as:

$$\mathbf{P}\left(\psi[\hat{T}_\iota] \mid T = \tau\right)$$

In order to estimate this quantity, we sample trajectories from $\hat{T}_\iota \mid \{T = \tau\}$ using a variation of the Gillespie algorithm: the counterfactual simulation algorithm — or **co-simulation algorithm**.

CO-SIMULATION ALGORITHM

Given a reference trace and an intervention ι, the **co-simulation** algorithm produces a random **counterfactual trace** that gives an account of what may have happened had ι occurred.

Ref.	**CF**
init	*init*
pk	×
b	*b*
·	*u*
p	·

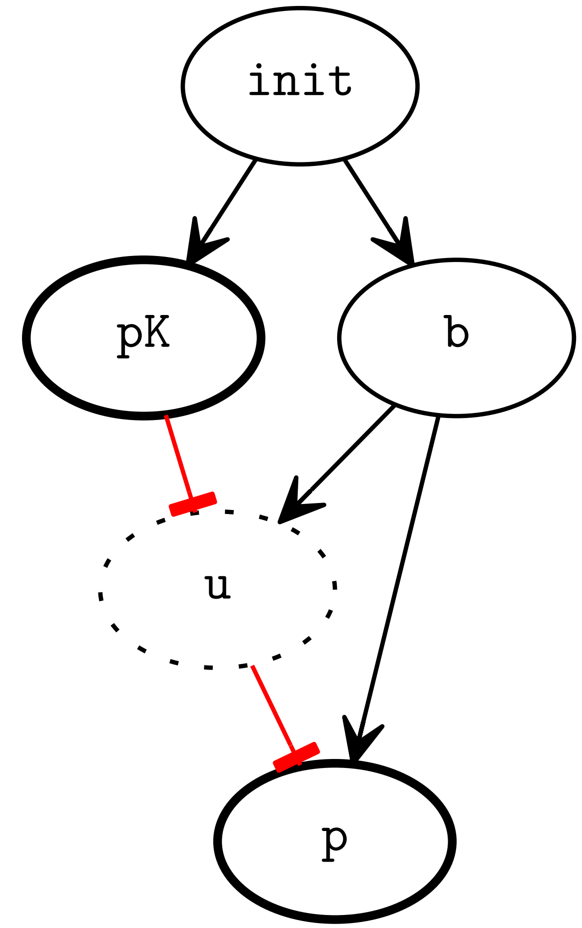
Example

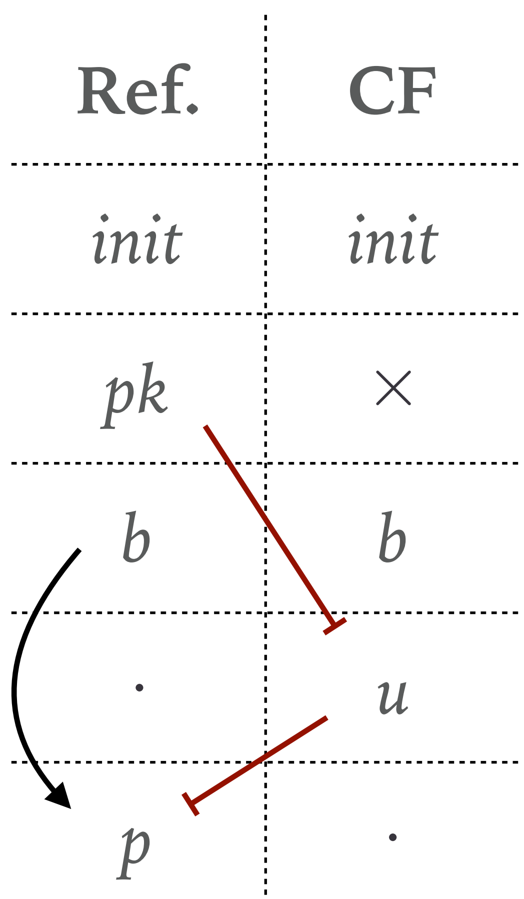
On the left, we show a run of the co-simulation algorithm, the intervention consisting in blocking rule ***pk***.

On performances: on average, co-simulating a trace is about 3 times slower than simulating it in the first place.

INHIBITION ARROWS

We can explain the differences between a reference trace and a corresponding counterfactual trace using **inhibition** arrows.





Theorem

Any event that is proper to the factual trace is connected by an event that is directly blocked by the intervention through a path containing an even number of inhibition arrows.

CONCLUSION AND PERSPECTIVES

The use of counterfactual reasoning enables us to produce **better causal explanations** by:

- being more sensitive to the **kinetic** aspects of a model
- providing a proper account of **inhibition** between molecular events

Current work

- What counterfactual experiments are worth trying ?
- How does counterfactual reasoning interact with trace slicing ?
  [Mickaël Laurent's internship]

Other applications for counterfactual reasoning ?

Our intuition is that counterfactual simulation could provide an interesting **experimental** tool, especially when studying highly stochastic models.

Special thanks to

Pierre Boutillier

Matt Fredrikson

Jérôme Feret

Jean Krivine

Iona Critescu

Algorithm 1 Resimulation loop.

{ t is the current time, M the current state mixture and M_0 the intermediate state of τ at time t }

$\alpha' \leftarrow \sum_r \lambda_r \cdot |\Delta_r(M, M_0)|$
draw $\delta \sim \textsc{Exp}(\alpha')$
$t_\mathsf{c} \leftarrow t + \delta$
$t_\mathsf{f} \leftarrow$ time of the next event in τ
$t' \leftarrow \min\{t_\mathsf{c}, t_\mathsf{f}\}$
if $t' = t_\mathsf{c}$ **then**
  draw a rule r with prob. $\propto \lambda_r \cdot |\Delta_r(M, M_0)|$
  draw a divergent embedding $\varphi \in \Delta_r(M, M_0)$
  $e \leftarrow (r, \varphi)$
else
  $e \leftarrow$ next event in τ
end if
if $\neg\mathsf{blocked}_\iota(t, e) \wedge e$ triggerable in M **then**
  update M by triggering event e
end if
$t \leftarrow t'$

MORE ON INHIBITION

Definition

An event *e* that happens at time *t* in the factual trace is said to **inhibit** an event *e'* that happens at time *t'* in the counterfactual trace if:

- $t < t'$
- there exists a site *s* such that *e* is the last event in the factual trace before time *t'* that modifies *s* from the value it is tested to by *e'* to a different value
- there are no events in the counterfactual trace modifying *s* in the time interval *(t, t')*