Computing temporal data from temporal data

Nicole Bidoit and François Hantry

LRI-Université Paris XI, Orsay

nicole.bidoit AT lri.fr , françois.hantry AT voila.fr

Motivation and Goal

"Time is ubiquitous" [Chomicki, Toman 98]

Ordered information is ubiquitous

- medical applications
- animation
- multimedia applications
- web applications
- videos, scenaries
- transport
- ...
- verification of complex systems
- bioinformatics
- simulation
- digital libraries
- personal information management
- sensor applications

Temporal database: temporal data management and querying

State of the art

- **Formal temporal query languages**

  ► Temporal database = finite sequence of database states

  languages based on linear TL [Prior 57], and extensions ETL [Wolper 83], μ-TL [Vardi88] :

  T-WHILE, T-FIXPOINT, RNTL ...

  [Chomicki, Toman 98], [Abiteboul,Herr,Van den Busche 99], [Bidoit, De Amo 99], [Bidoit, Objois 05]

  ► temporal database = time-stamped database

  Time-stamped calculus (TS-FO) and extensions TS-WHILE, TS-FIXPOINT ...,

  [Abiteboul,Herr,Van den Busche 99],[Bidoit, Objois 05]

Temporal database: temporal data management and querying

State of the art

- **Formal temporal query languages**

  ▶ Temporal database = finite sequence of database states

  languages based on linear TL [Prior 57], and extensions ETL [Wolper 83], μ-TL [Vardi88] :

  T-WHILE, T-FIXPOINT, RNTL ...

  [Chomicki, Toman 98], [Abiteboul,Herr,Van den Busche 99], [Bidoit, De Amo 99], [Bidoit, Objois 05]

  ▶ temporal database = time-stamped database

  Time-stamped calculus (TS-FO) and extensions TS-WHILE, TS-FIXPOINT ...,

  [Abiteboul,Herr,Van den Busche 99],[Bidoit, Objois 05]

- **Extensions of SQL:** TSQL2, ChronoLog, ChronoSQL, ATSQL2, ...

  [M. Bölhen, Ch. Jensen, R. Snodgrass, A. Steiner, & all]

  ▶ snapshot queries — data at some time point ?

  ▶ history extractions — evolution of data ?

  ▶ bitemporal data — valide time; transaction time

Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation

- **temporal queries : TDB $\longrightarrow$ BD**

  "temporal" to "static"

Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

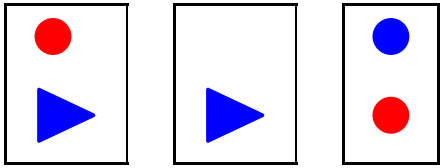
Limitation

- **temporal queries : TDB $\longrightarrow$ BD**

  "temporal" to "static"

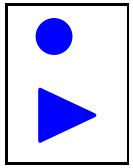
- **Example : elements whose color appears everywhere.**

Temporal input



$\longmapsto$

Static output



Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Goal

- **temporal query :** **TDB** $\longrightarrow$ **TDB**

  ”temporal” to ”temporal”

  ”sequence” to ”sequence”

Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Goal

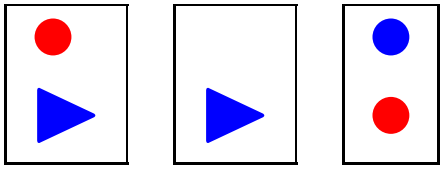
- **temporal query : TDB $\longrightarrow$ TDB**

  ”temporal” to ”temporal”

  ”sequence” to ”sequence”

- **Example : color based spliting.**

Temporal input



Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Goal

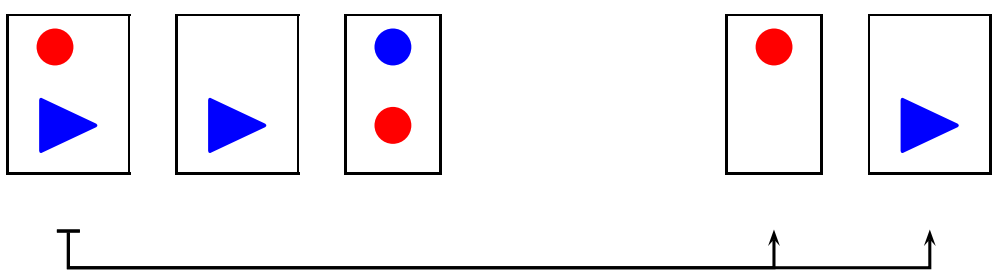
- **temporal query : TDB $\longrightarrow$ TDB**

  ”temporal” to ”temporal”

  ”sequence” to ”sequence”

- **Example : color based spliting.**

Temporal input



Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Goal

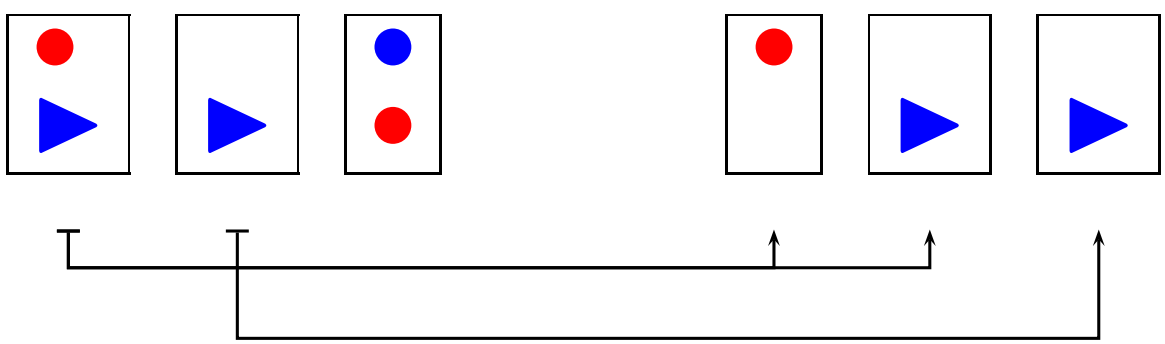
- **temporal query : TDB $\longrightarrow$ TDB**

  ”temporal” to ”temporal”

  ”sequence” to ”sequence”

- **Example : color based spliting.**

Temporal input



Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

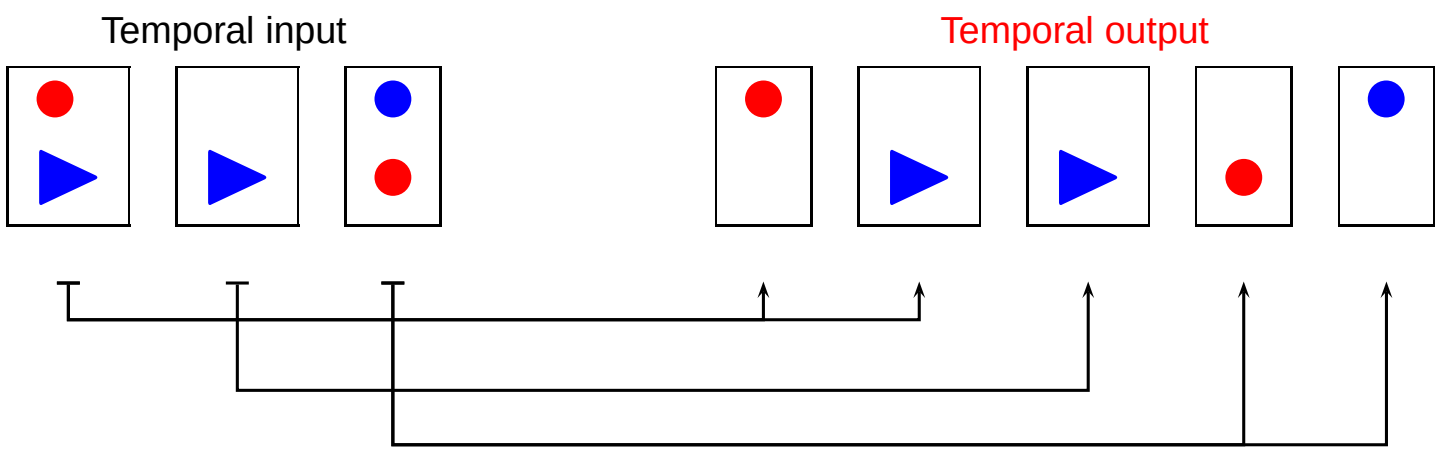
Goal

- **temporal query : TDB $\longrightarrow$ TDB**

  ”temporal” to ”temporal”

  ”sequence” to ”sequence”

- **Example : color based spliting.**



Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Goal

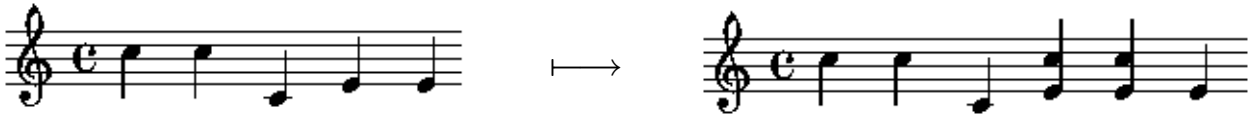
- **temporal query :** **TDB** $\longrightarrow$ **TDB**

  ”temporal” to ”temporal”

  ”sequence” to ”sequence”

- **Example : Canon** **Musical score $\approx$ temporal instance**

  (replay the scores v times, with time offset o. (for the example: $v = 2, o = 3$)



Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Goal

- **temporal query :** **TDB** $\longrightarrow$ **TDB**

  ”temporal” to ”temporal”

  ”sequence” to ”sequence”

- **Example : Coalescing**: collapse consecutive and identical states into a single one.

$$I_1, I_1, I_1, I_2, I_2, I_3, I_4, I_4, I_4, I_5 \quad \longmapsto \quad I_1, I_2, I_3, I_4, I_5$$

Temporal database: temporal data management and querying

State of the Art	Formal temporal query languages, Extensions of SQL
Limitation	temporal queries = ”temporal” to ”static” mappings
Goal	**temporal query = ”temporal” to ”temporal” mappings**

Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Goal **temporal query = ”temporal” to ”temporal” mappings**

$\rightsquigarrow$ **What are ”temporal” to ”temporal” queries**

Relational Temporal Machines

joint work with François Hantry (TIME 2007)

Temporal database: temporal data management and querying

State of the Art Formal temporal query languages, Extensions of SQL

Limitation temporal queries = ”temporal” to ”static” mappings

Goal **temporal query = ”temporal” to ”temporal” mappings**

⇝ **What are ”temporal” to ”temporal” queries**

Relational Temporal Machines

joint work with François Hantry (TIME 2007)

⇝ **What languages for defining ”temporal” to ”temporal” queries / computations**

SQTL: a Preliminary Proposal for a Temporal-to-temporal Query Language

joint work with Matthieu Objois (TIME 2007, Thèse)

Outline of the talk

- **Relational Temporal Machines (RTM)**
  - ▶ Abstract model capturing general temporal queries
    - ↪ Turing Machine [Turing36],[Turing37]
    - ↪ Relational Machine, Loosely coupled Generic Machine [Abiteboul,Vianu 95]

- **Toward a "normal form" of Relational Temporal Machines**

  Which components are really useful ?
  - ▶ Extended one tape RTM
  - ▶ One tape RTM

- RTM **complexity classes and temporal languages**

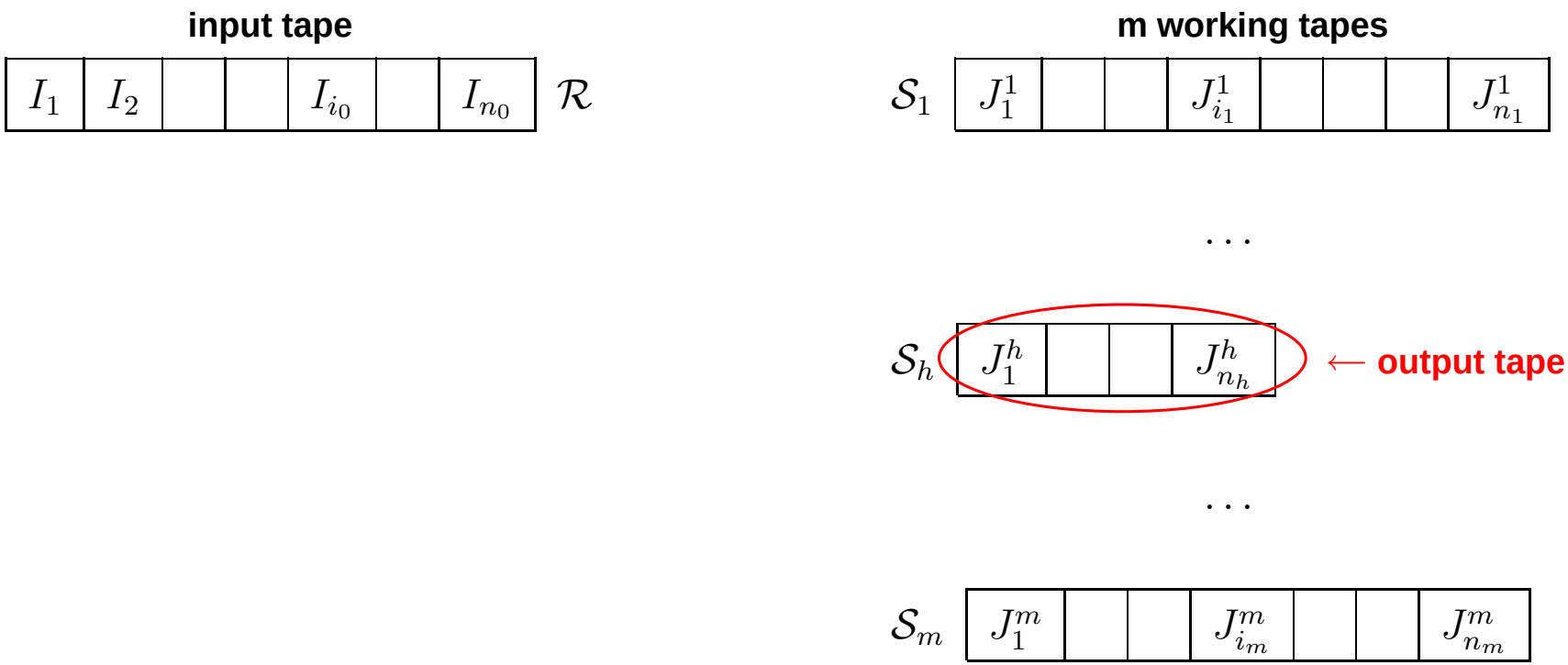
  Space complexity = size of auxiliary tapes
  - ▶ T-WHILE and TS-WHILE versus linear and polynomial RTM
  - ▶ Complete languages for constant RTM

- **Discussion and Future Work**

Outline of the talk

- **Relational Temporal Machines (RTM)**
  - ▶ Abstract model capturing general temporal queries
    - ↪ Turing Machine [Turing36],[Turing37]
    - ↪ Relational Machine, Loosely coupled Generic Machine [Abiteboul,Vianu 95]

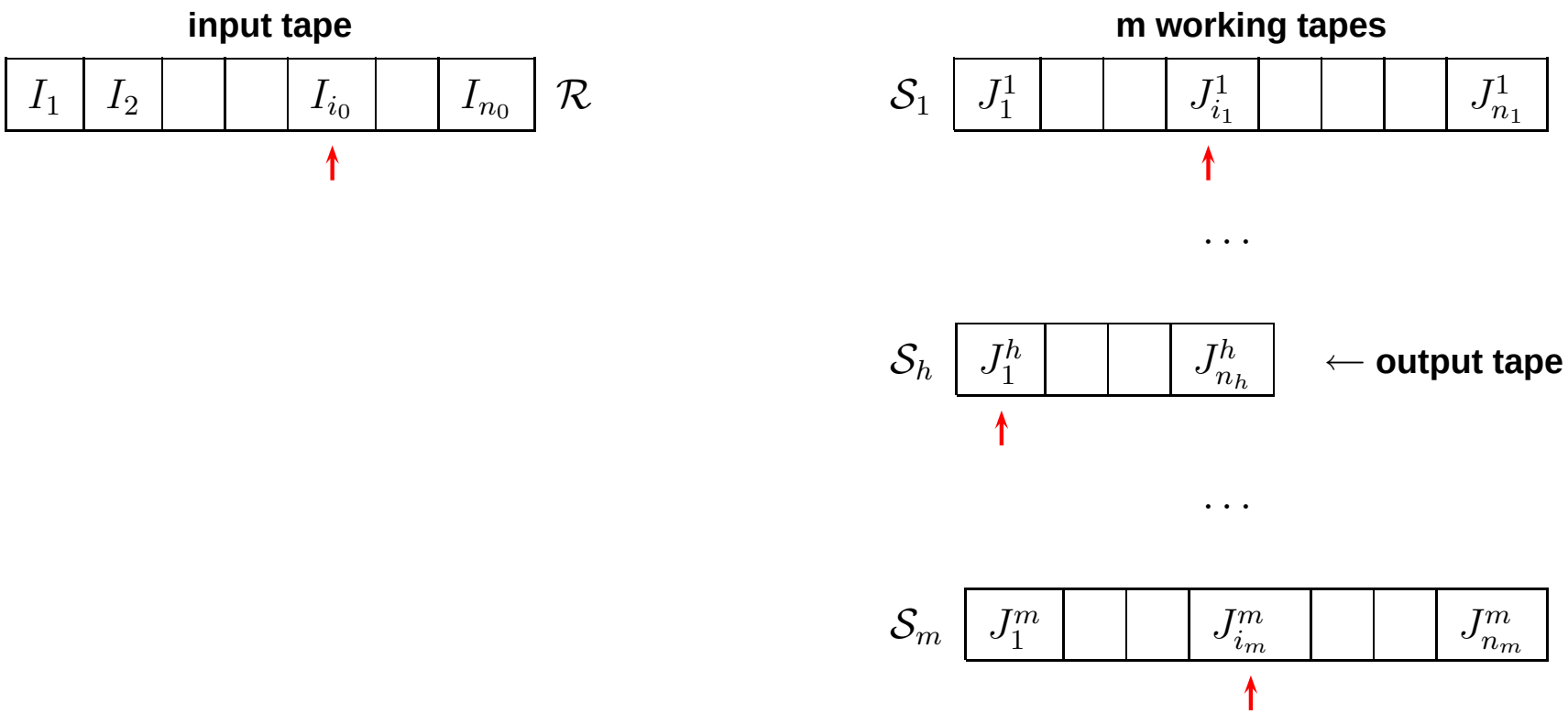
Relational Temporal Machines



↪ **tapes store temporal instances**

- input tape → input schema $\mathcal{R}$ — $I_1 \ldots I_{n_0}$ is a temporal instance over $\mathcal{R}$
- auxiliary tape k → auxiliary schema $\mathcal{S}_k$ — $J_1^k \ldots J_{n_k}^k$ is a temporal instance over $\mathcal{S}_k$

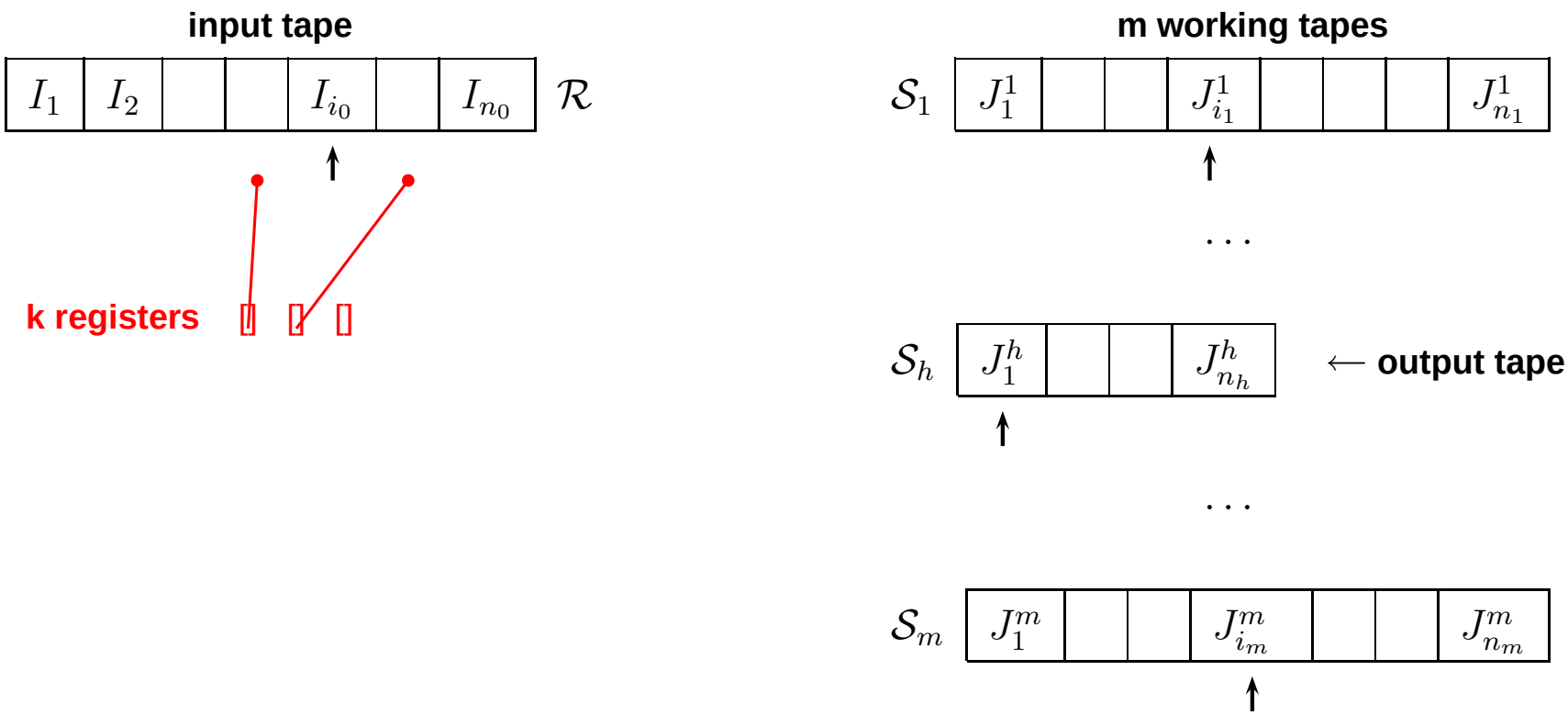
Relational Temporal Machines



↪ **cursors give access to relational instances** **cursor moves: left and right**

- input cursor → input tape → access to I_{i_0}
- auxiliary cursor k → auxiliary tape k → access to $J_{i_k}^k$

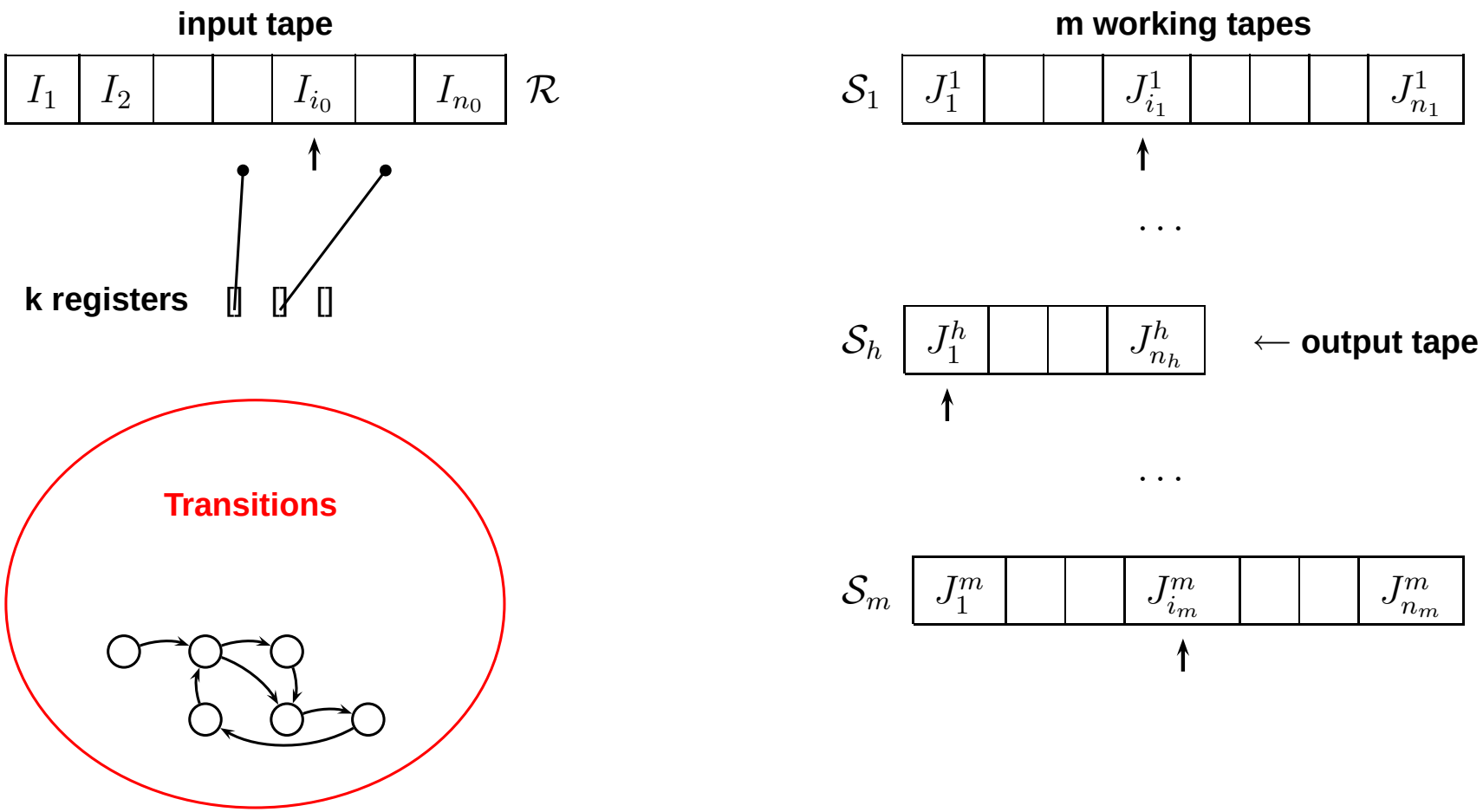
Relational Temporal Machines



↪ **Registers**

- on input tape only
- store time points

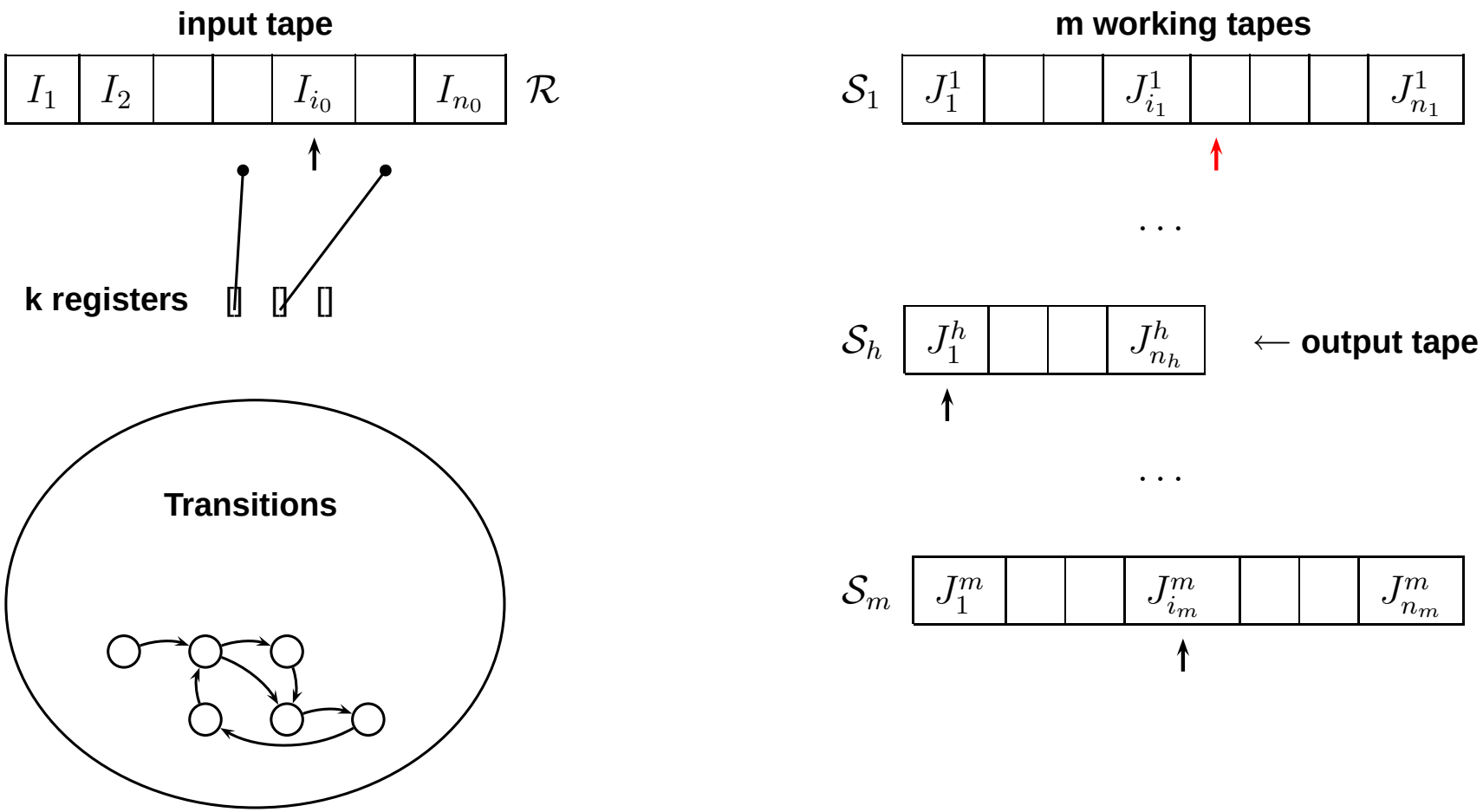
Relational Temporal Machines



↪ **Transition specification**

Q_B,	P,	s_1	$\longrightarrow$	s_2,	A
boolean FO query	**register indices**	**state**	$\longrightarrow$	**state**	**action**

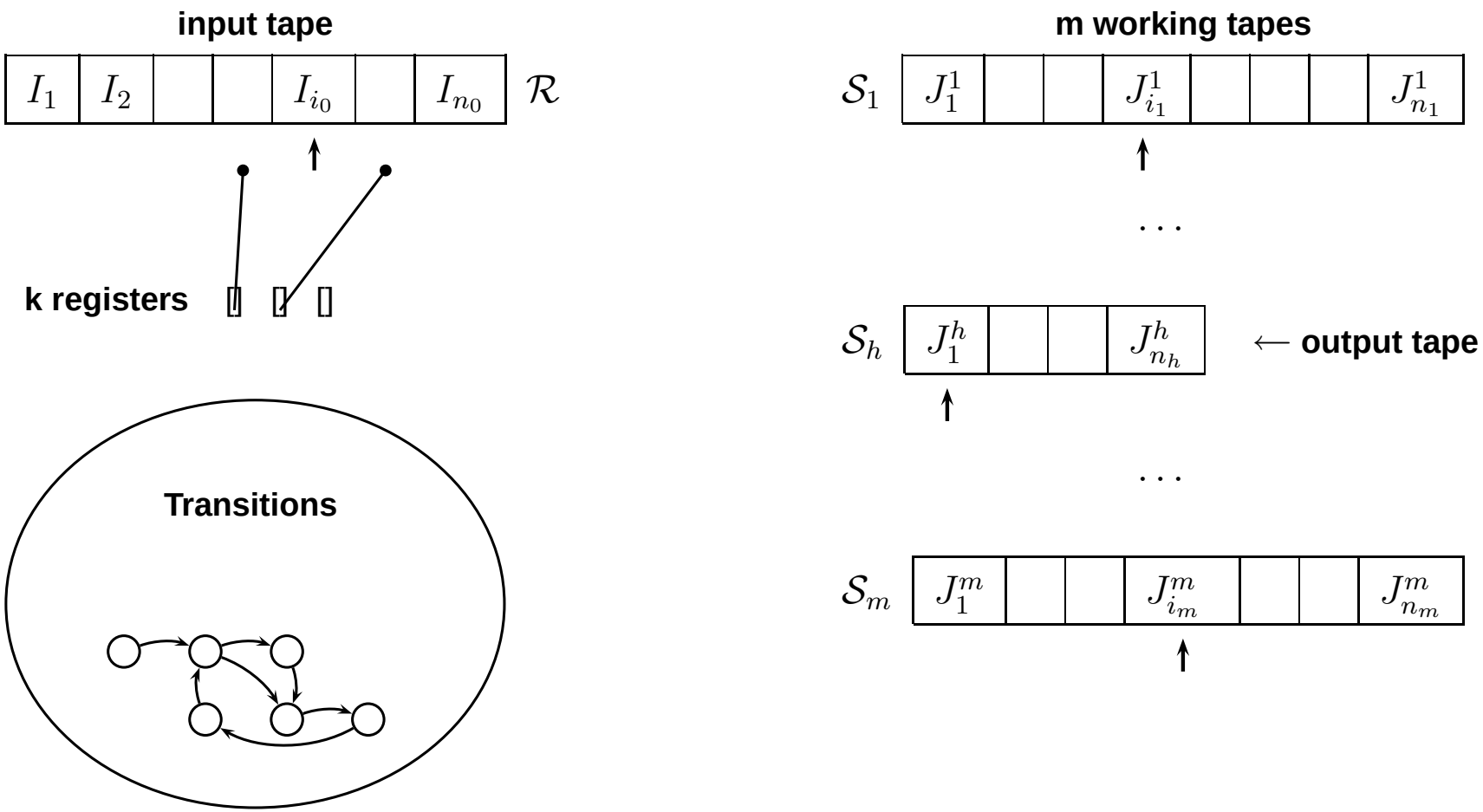
Relational Temporal Machines : cursor moves



↪ **Action specification**

- **cursor move: left , right**

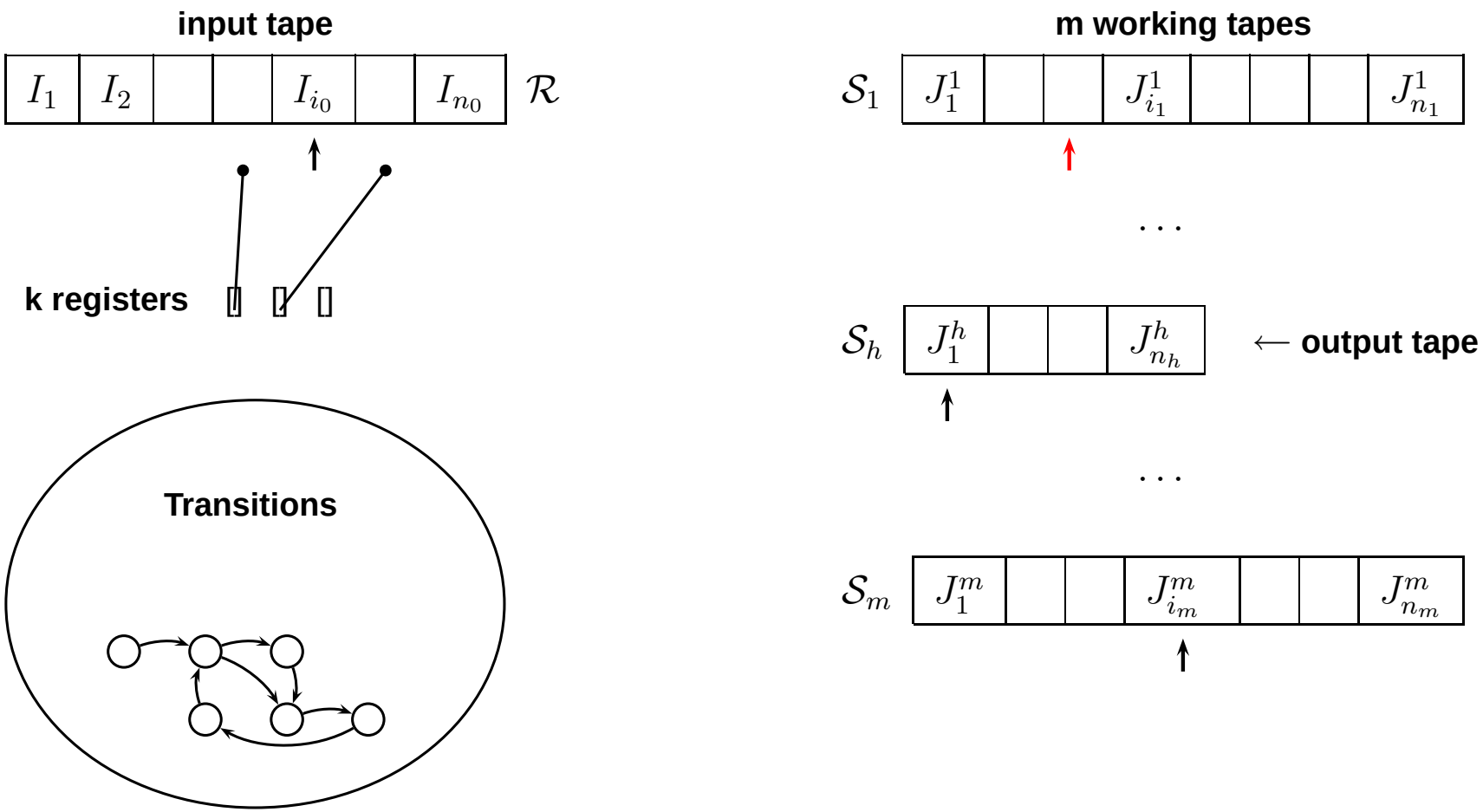
Relational Temporal Machines : cursor moves



↪ **Action specification**

- **cursor move: left , right**

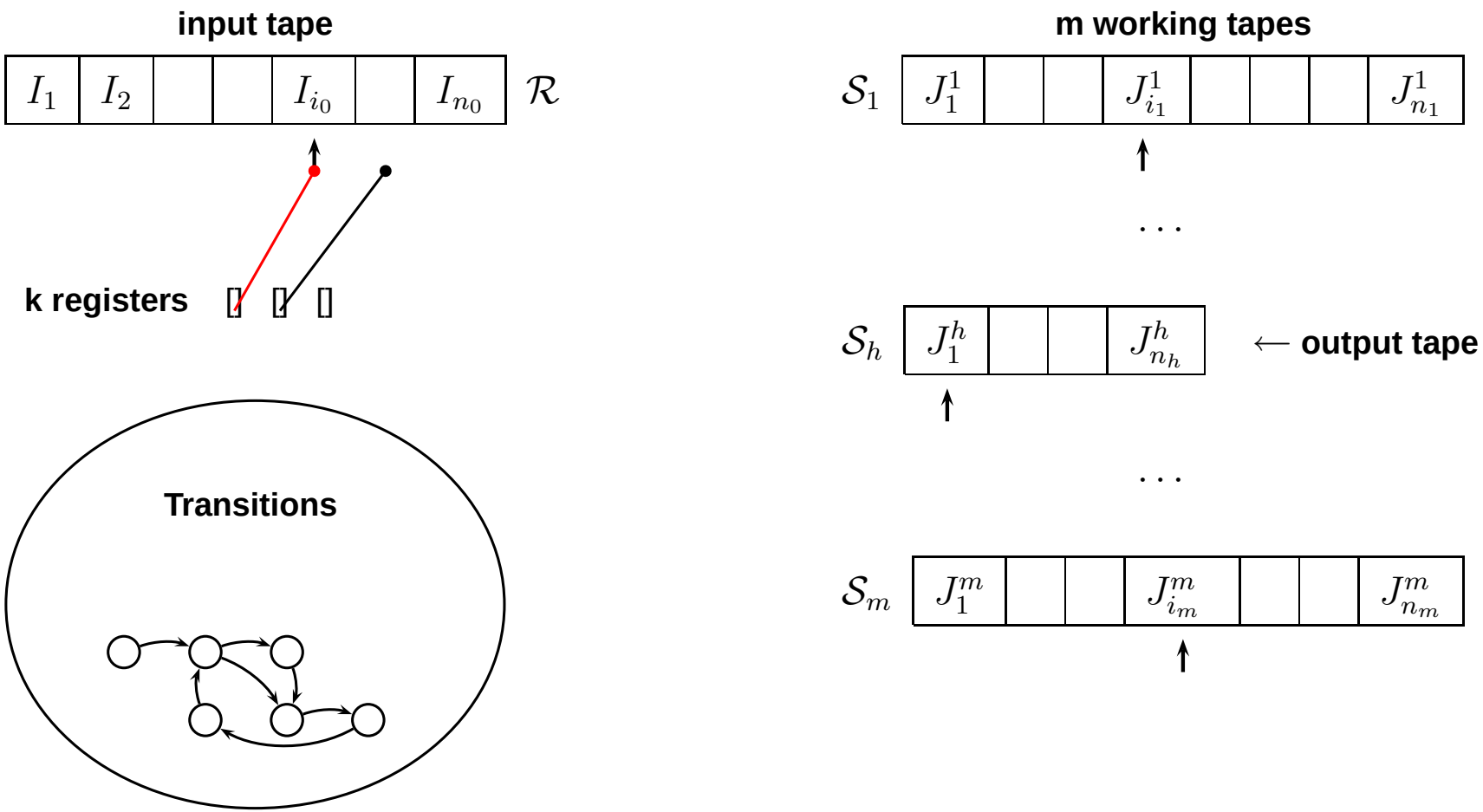
Relational Temporal Machines : cursor moves



↪ **Action specification**

- **cursor move: left , right**

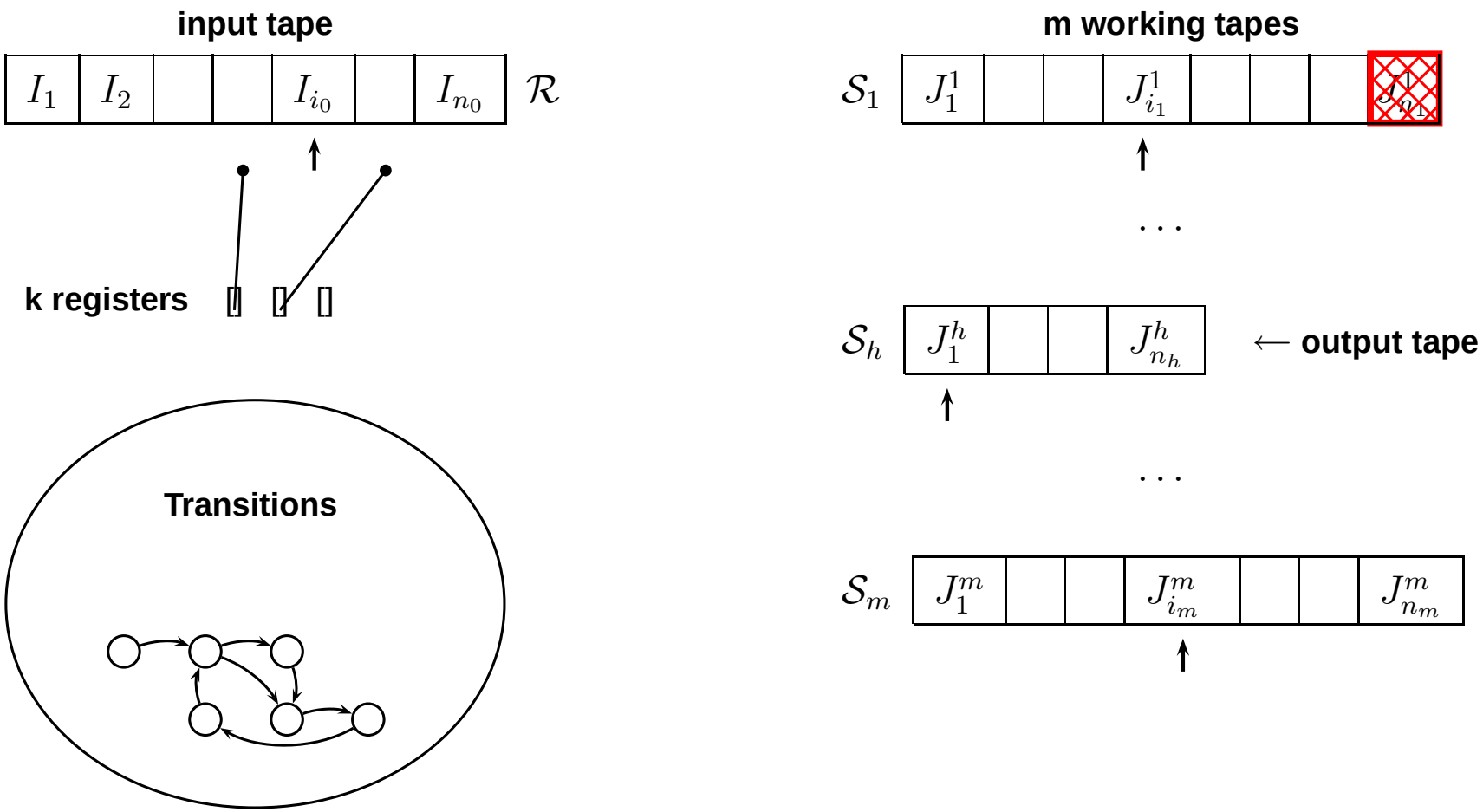
Relational Temporal Machines : register assignement



↪ **Action specification**

- **register assignement : input cursor value**

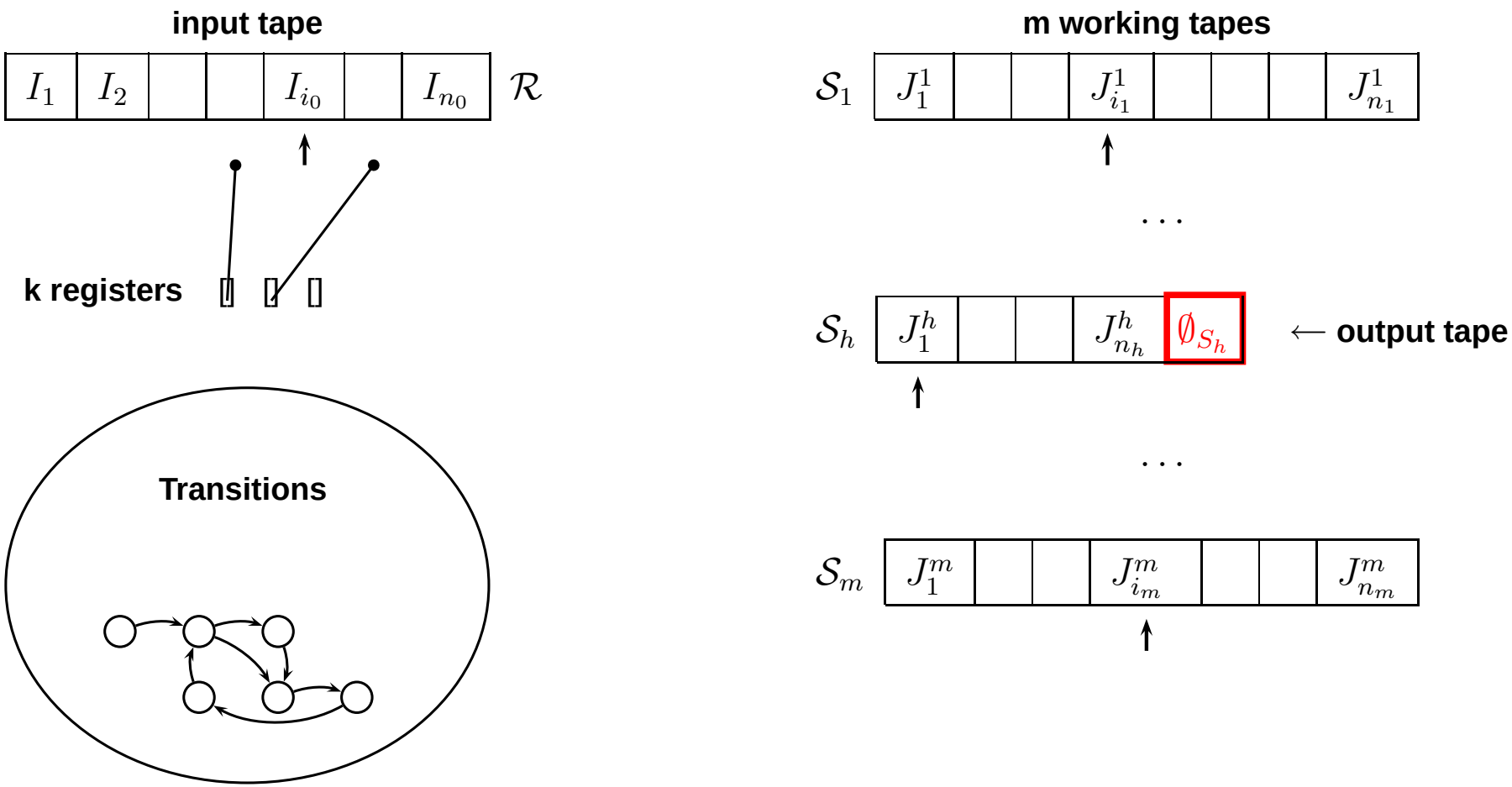
Relational Temporal Machines : remove tape cell



↪ **Action specification**

- **remove last cell of auxiliary tape** k

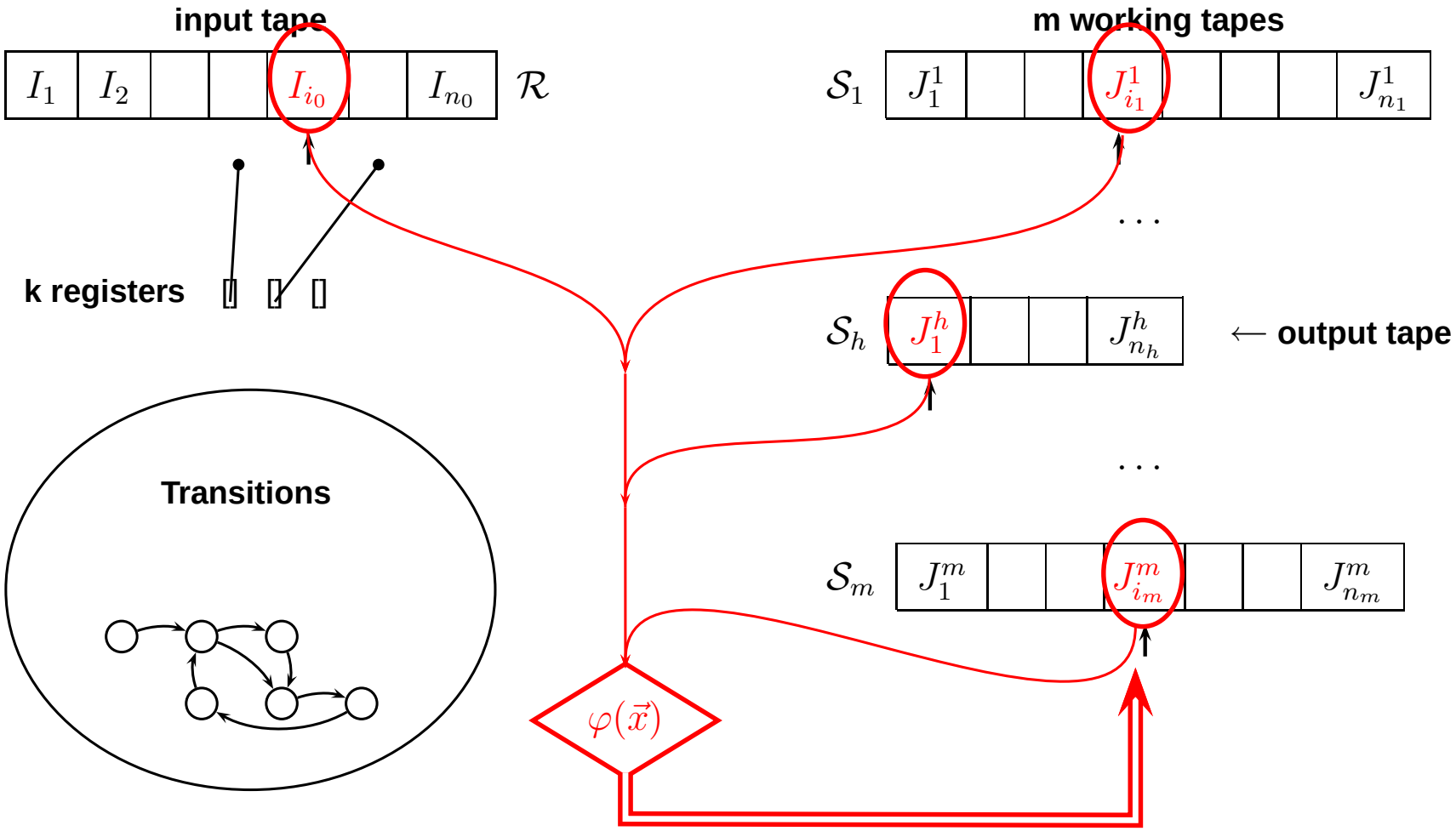
Relational Temporal Machines : add tape cell



↪ **Action specification**

- **add a new empty cell at the end of auxiliary tape** k

Relational Temporal Machines : update tape with FO query moves



↪ **Action specification** • **Update tape $\mathbf{k}$ with the FO query $\varphi(\vec{x})$**

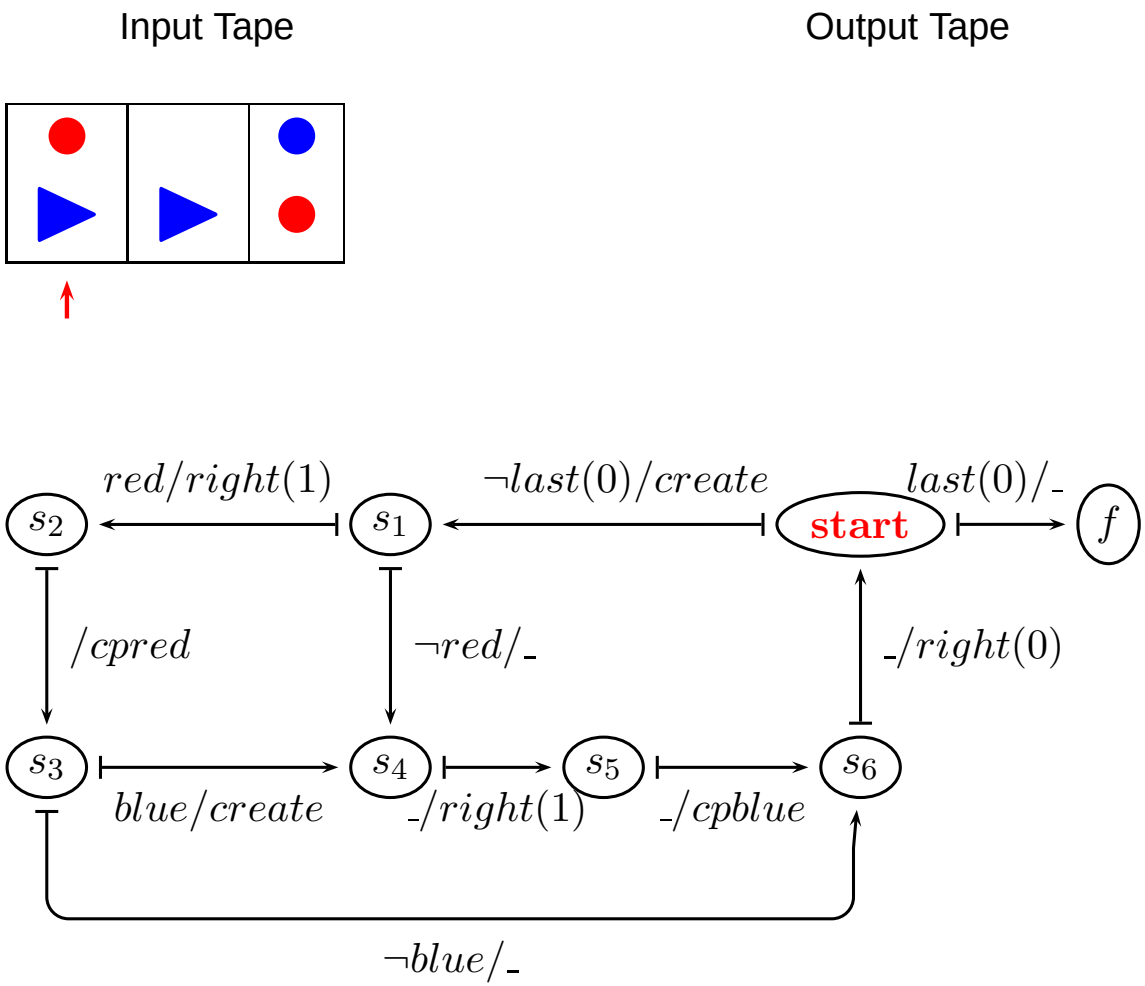
1. evalate $\varphi(\vec{x})$ over the instances pointed by input and auxiliary cursors
2. write result in the cell of tape k pointed by the cursor

Outline of the talk

- **Relational Temporal Machines (RTM)**
  - ▶ Abstract model capturing general temporal querying
    - ↪ Turing Machines [Turing 36], [Turing 37]
    - ↪ Relational Machines, Loosely coupled Generic Machine [Abiteboul, Vinau 95]

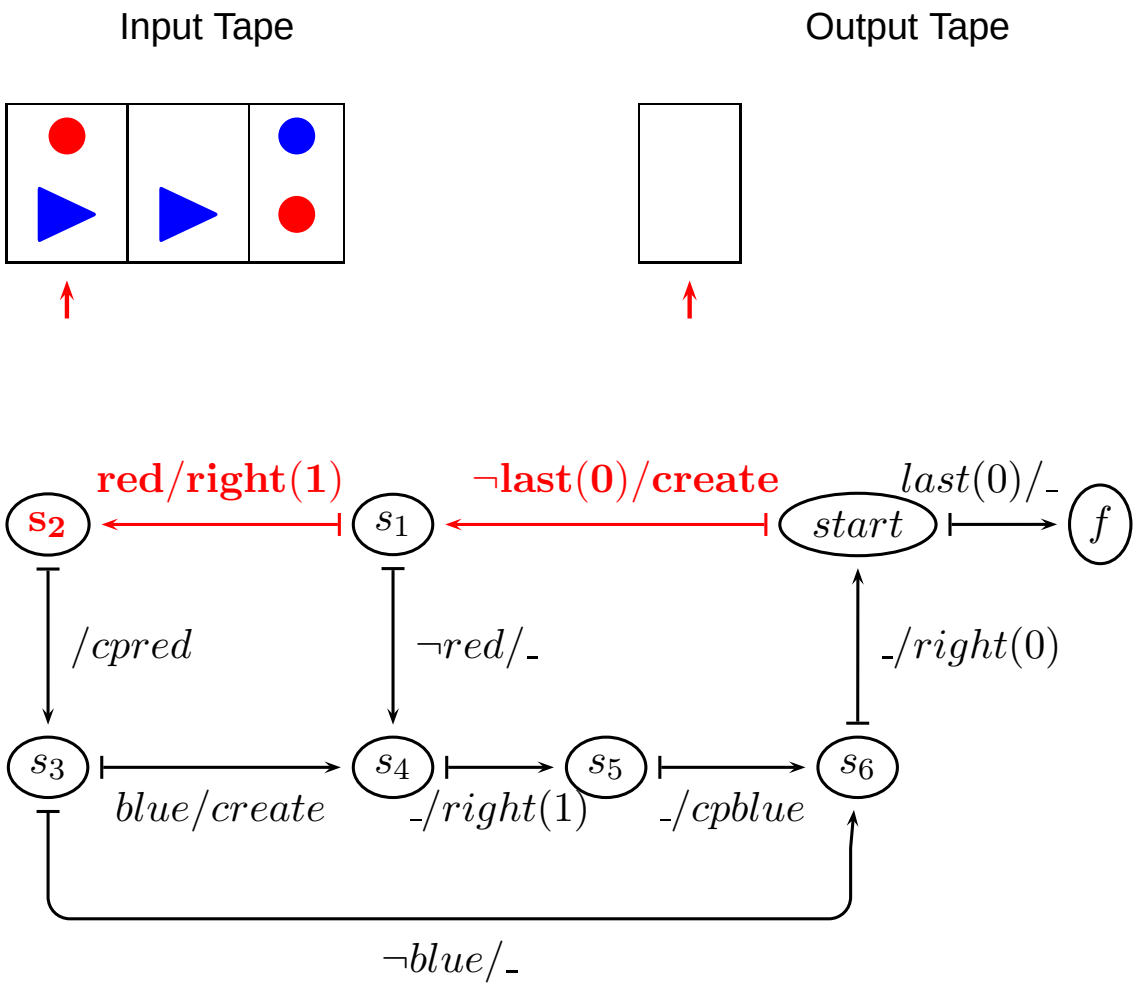
An example of RTM : color split

Relational Temporal Machines : the color split example



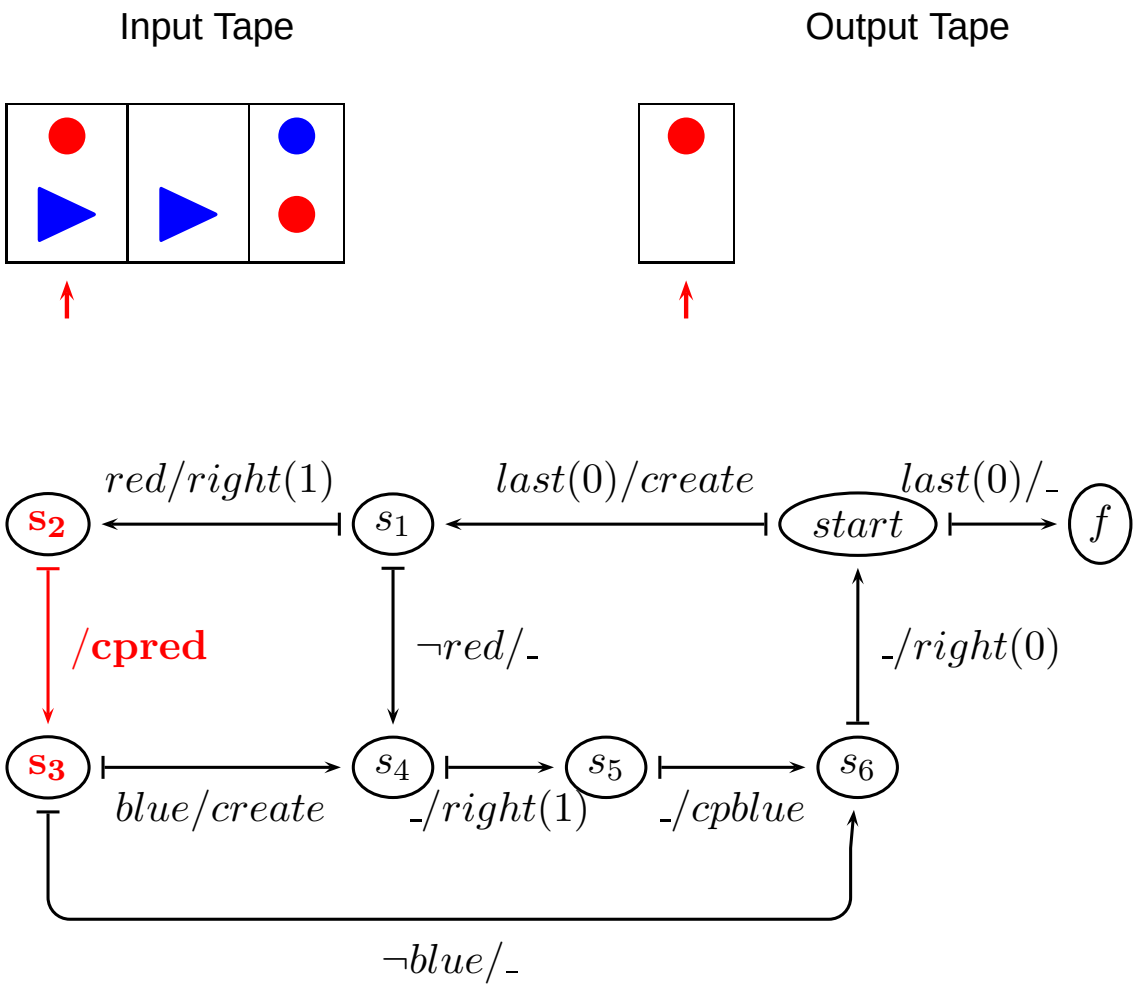
queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x, y) \wedge y{=}red$
cpblue	$P(x, y) \wedge y{=}blue$

Relational Temporal Machines : the color split example



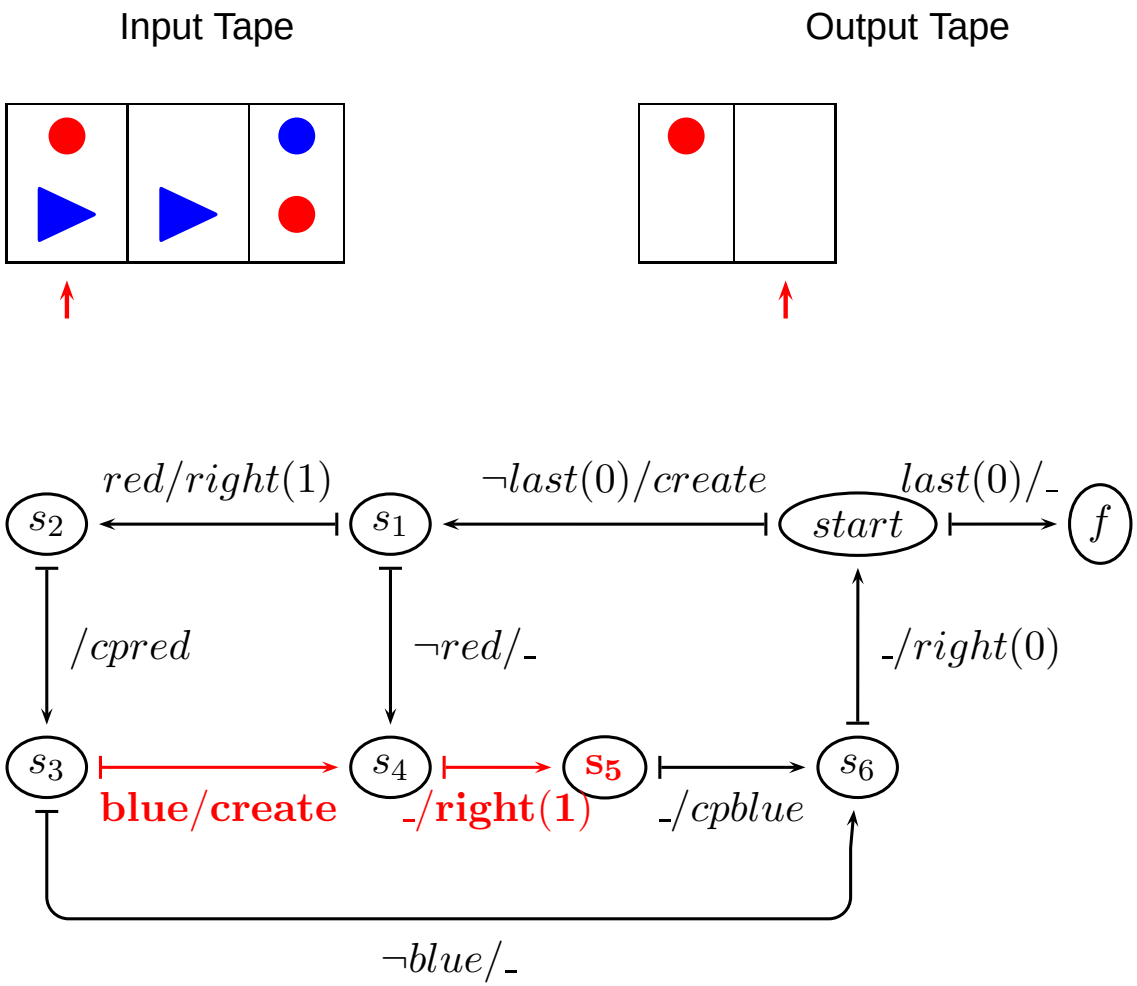
queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x,y) \wedge y{=}red$
cpblue	$P(x,y) \wedge y{=}blue$

Relational Temporal Machines : the color split example



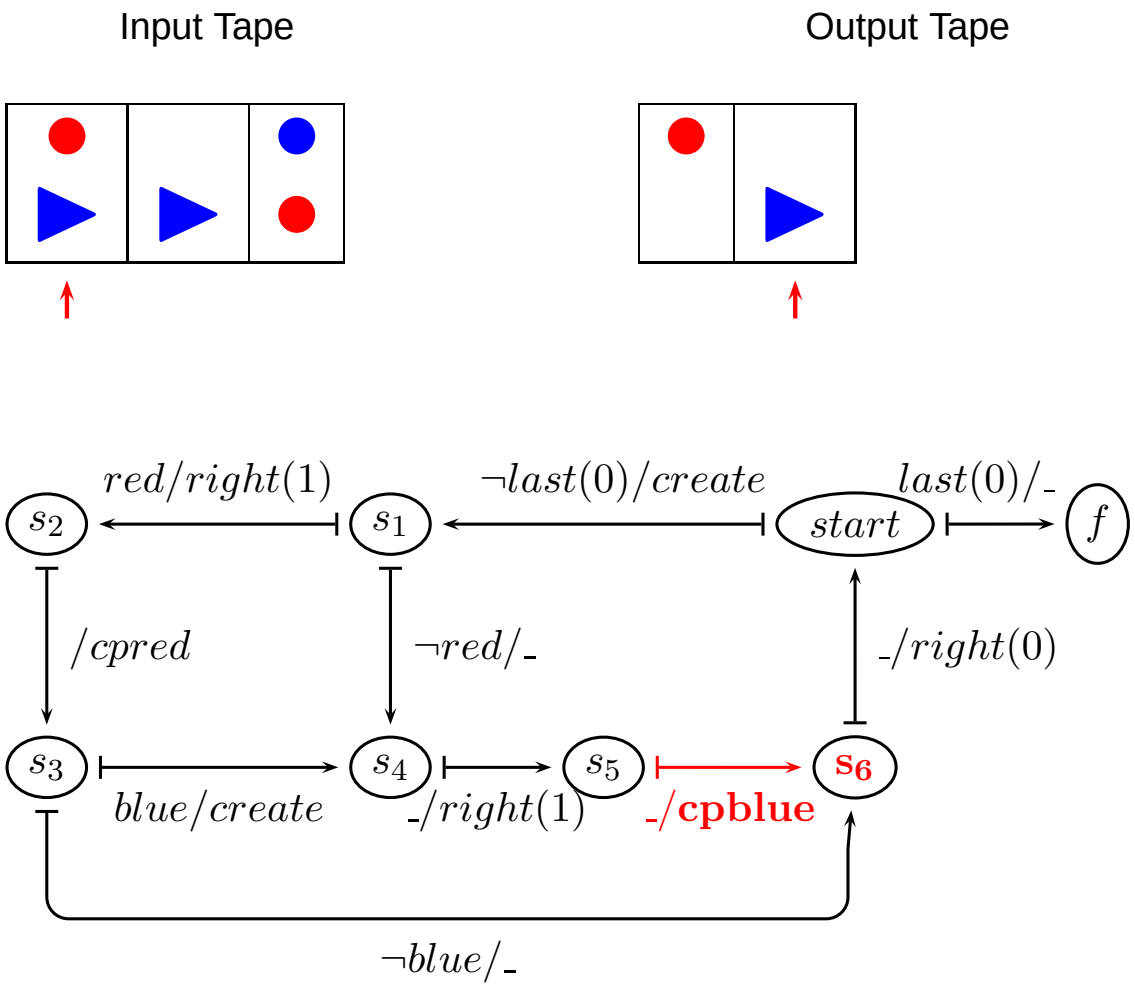
queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x, y) \wedge y{=}red$
cpblue	$P(x, y) \wedge y{=}blue$

Relational Temporal Machines : the color split example



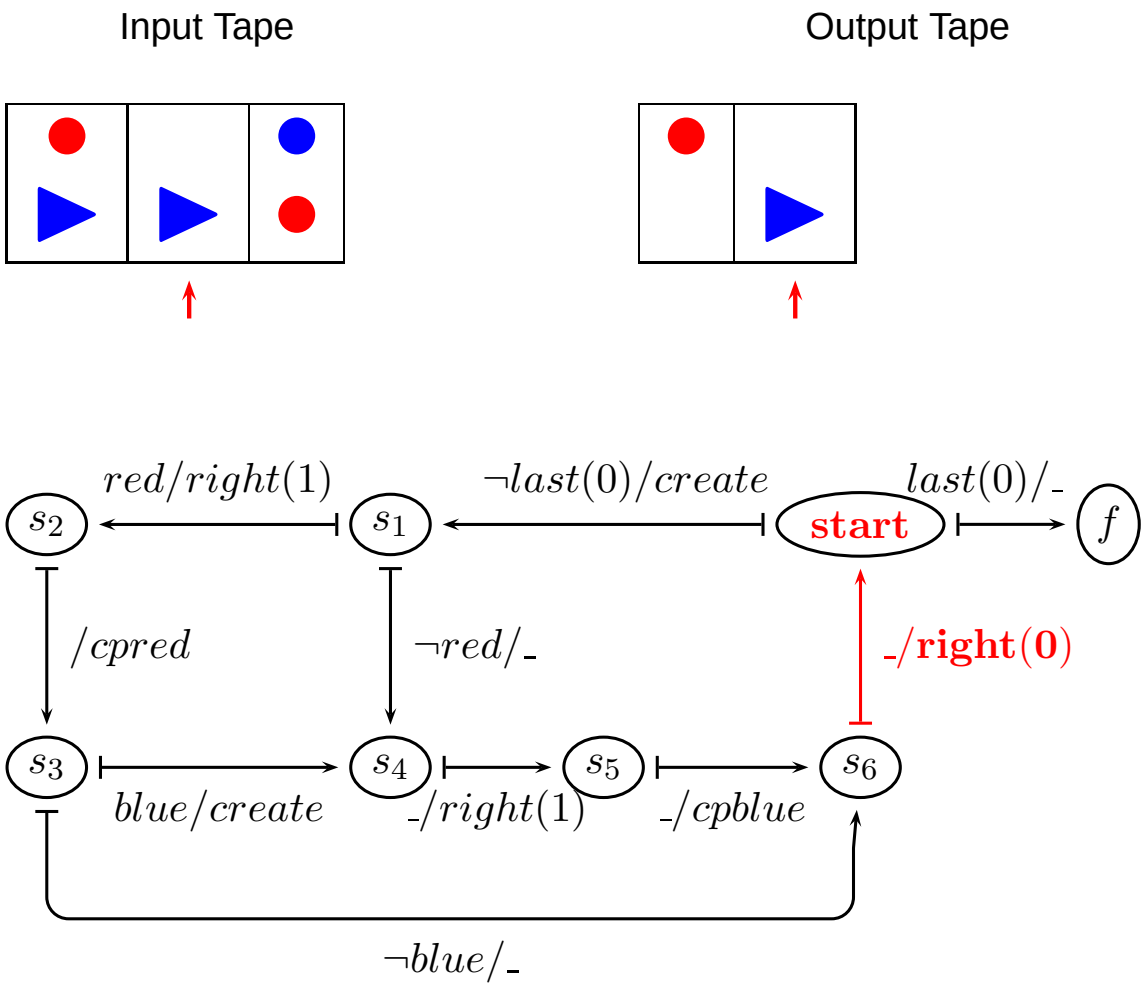
queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x,y) \wedge y{=}red$
cpblue	$P(x,y) \wedge y{=}blue$

Relational Temporal Machines : the color split example



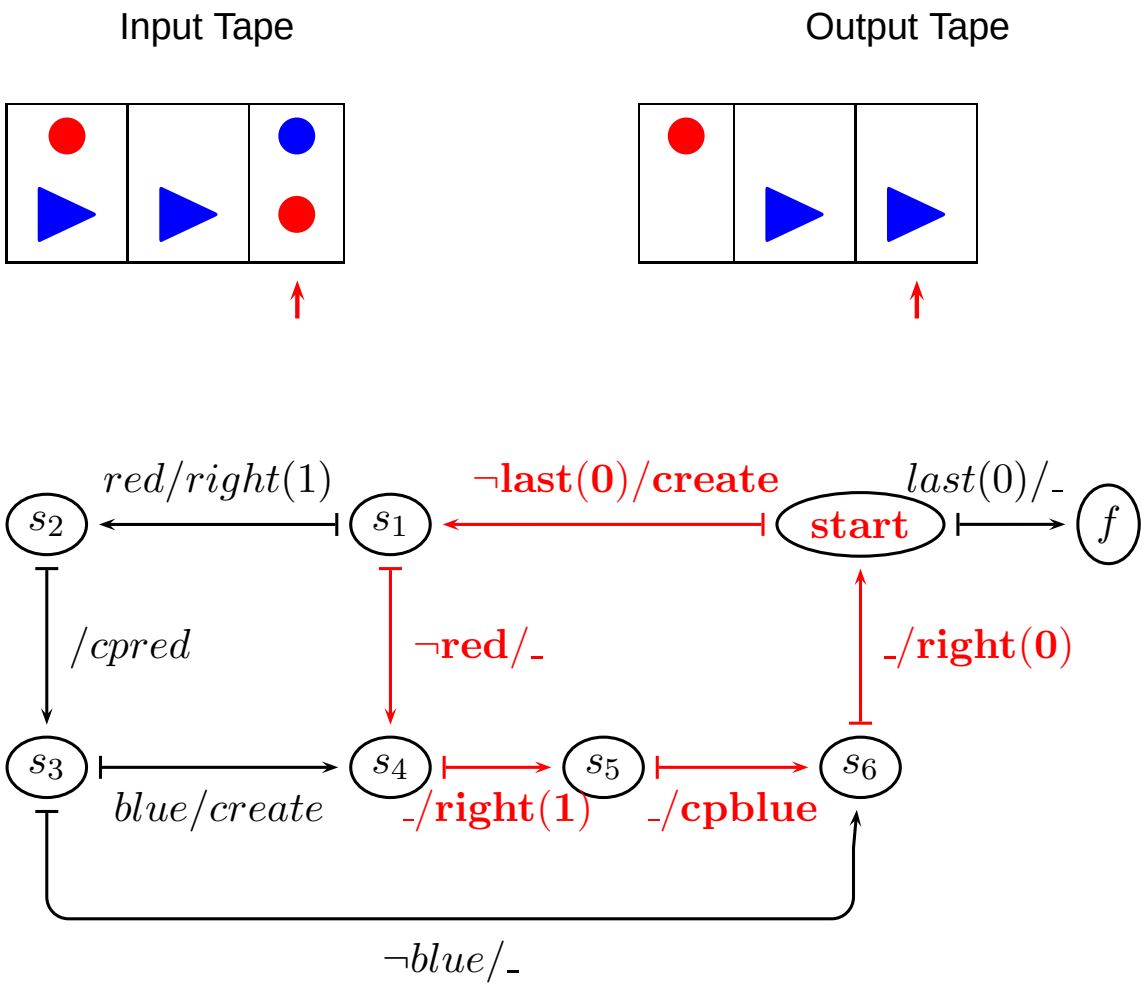
queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x,y) \wedge y{=}red$
cpblue	$P(x,y) \wedge y{=}blue$

Relational Temporal Machines : the color split example



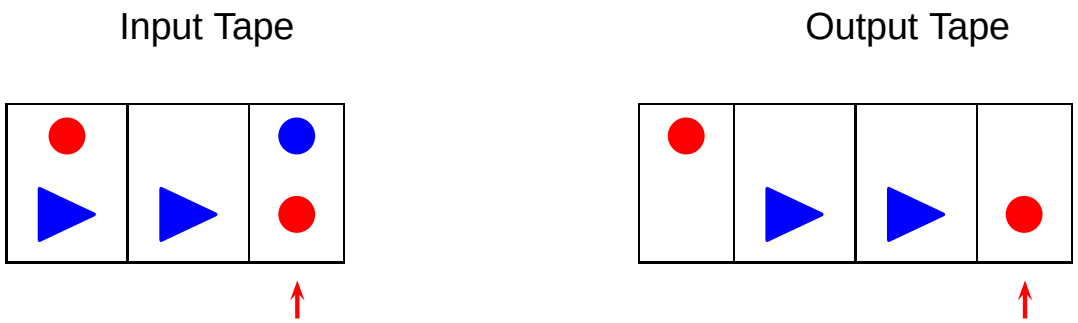
queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x, y) \wedge y{=}red$
cpblue	$P(x, y) \wedge y{=}blue$

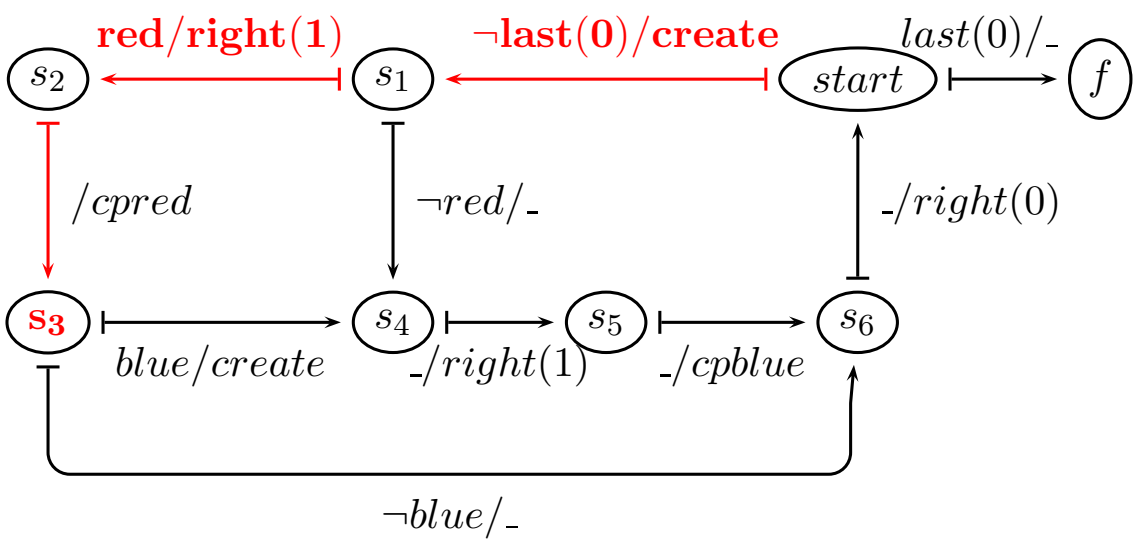
Relational Temporal Machines : the color split example



queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x,y) \wedge y{=}red$
cpblue	$P(x,y) \wedge y{=}blue$

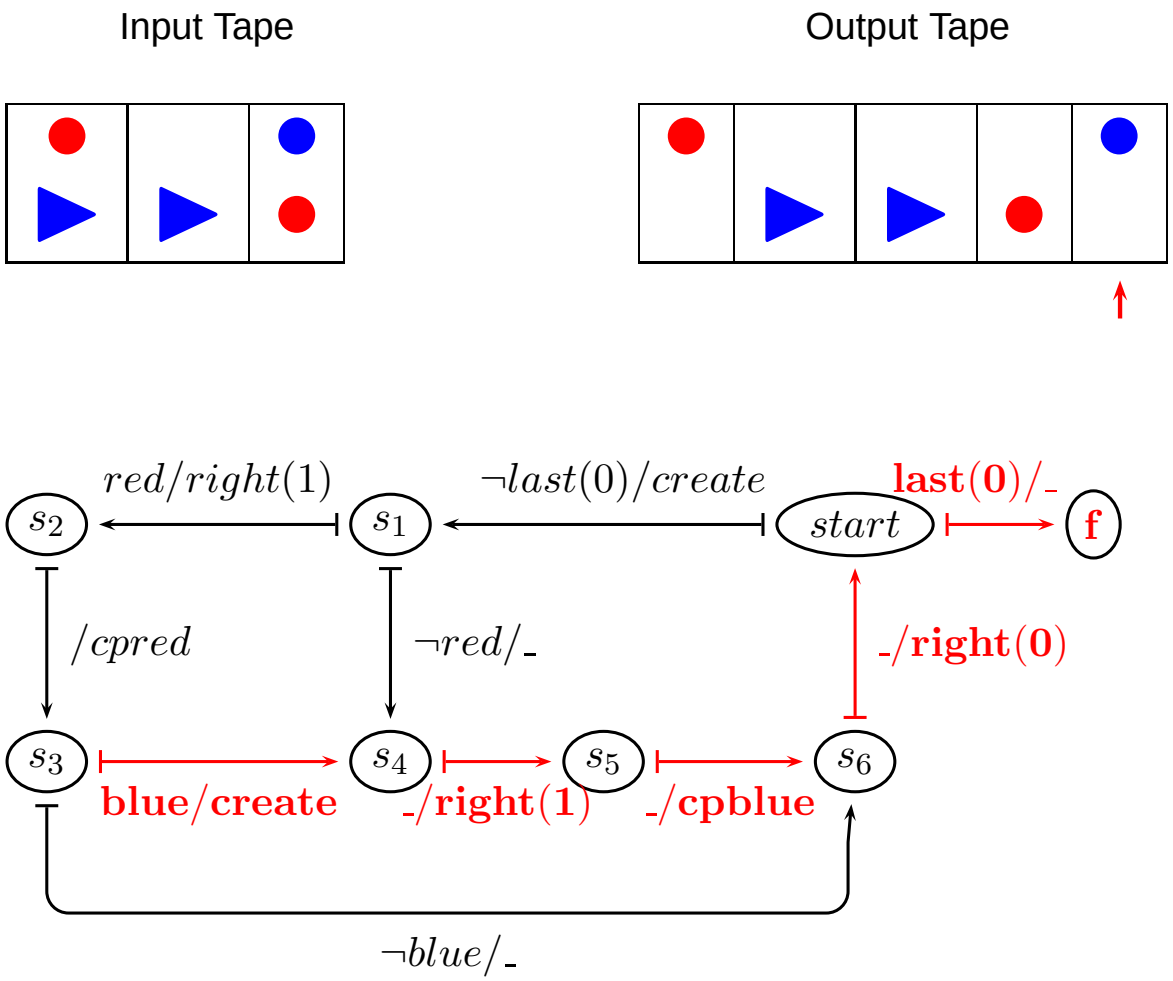
Relational Temporal Machines : the color split example





queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x,y) \wedge y{=}red$
cpblue	$P(x,y) \wedge y{=}blue$

Relational Temporal Machines : the color split example



queries and actions	
last(0)	checks end of input
red	$\exists x P(x, red)$
blue	$\exists x P(x, blue)$
cpred	$P(x,y) \wedge y{=}red$
cpblue	$P(x,y) \wedge y{=}blue$

Outline of the talk

- **Toward a ”normal form” of Relational Temporal Machines**

  Which components are really useful ?

  ▶ Extended one tape RTM

  ▶ One tape RTM

Relational Temporal Machines : which components are really useful ?

- Registers do not add any expressivity

$$\text{RTM}_k^m = \text{RTM}_0^{m+1}$$

(use an extra auxiliary tape to simulate register)

- Auxiliary tapes [1st result]

$$\text{RTM}_k^m = \text{RTM}_0^{1+}$$

Extended one-tape RTMS
- – one (temporal) auxiliary tape
- – a (static) store

("concatenate" the m auxiliary tapes $\longrightarrow$ one auxiliary tape

problem : m cursors ?

answer : use the store to simulate the (use of the) m cursors)

Relational Temporal Machines : which components are really useful ?

- Auxiliary tapes [2nd result]

$\text{RTM}_k^m = \text{RTM}_k^1$ one-tape RTMS

$\mathcal{M} \in \text{RTM}_k^m \quad \rightsquigarrow \quad \mathcal{N}_{\text{equiv}}; \mathcal{N}_{\text{load}}; \mathcal{N}_{\text{comp}}; \mathcal{N}_{\text{res}}$ with $\mathcal{N}_{\text{x}} \in \text{RTM}_k^1$

Proof : extension of [Abiteboul, Vinau 95] proof technique for *loosely coupled Generic Machine*.

$\underline{\mathcal{N}_{\text{equiv}}:}$ Encoding of a superset of the auxiliary relations computed when running the RTM_k^m $\mathcal{M}$.

Computation of a representation of an ordered partition $\{\delta_1 \ldots \delta_d\}$ of the set of r-ary tuples built from the active domain of the temporal input. $\mathcal{M}$ uses a finite number of FO queries.

$\underline{\mathcal{N}_{\text{load}}:}$ Encoding on the auxiliary tape of the ordered partition, the action tables and the input instance

The auxiliary tape is then used as a Turing machine tape.

$\underline{\mathcal{N}_{\text{comp}}:}$ Simulation of the run of the RTM $\mathcal{M}$.

$\mathcal{N}_{\text{comp}}$ is a Turing machine that works on the previous encoding stored on the auxiliary tape.

$\underline{\mathcal{N}_{\text{res}}:}$ Decoding of the result produced by the Turing machine $\mathcal{N}_{\text{comp}}$.

Outline of the talk

- RTM **complexity classes and temporal languages**

  Space complexity = size of auxiliary tapes

  ▶ T-WHILE and TS-WHILE versus linear and polynomial RTM

  ▶ Complete languages for constant RTM

Temporal query Languages : T-WHILE and TS-WHILE

Extensions of WHILE [Chandra,Harrel 82], see [Abiteboul, Herr, Van den Busche 95],[Bidoit, Objois 05].

TS-WHILE

Timestamped relations (Explicit time)

V^{st}	Frame	T
	●	1
	▶	1
	▶	2
	●	3
	●	3

TS-WHILE $\approx$ FO + assignments
+ while loops

T-WHILE

Sequences of relations (Implicit time)

V	Frame
	●
	▶

V	Frame
	▶

V	Frame
	●
	●

T-WHILE $\approx$ FO + assignments
+ while loops
+ temporal moves

Example: frame whose color appears everywhere

$\exists \mathbf{t} \mathbf{V}(\mathbf{f}, \mathbf{c}, \mathbf{t}) \wedge \forall \mathbf{t}' \exists \mathbf{f}' \mathbf{V}(\mathbf{f}', \mathbf{c}, \mathbf{t}')$

Q := V(f,c);
While true do
right;
Q := $[\mathbf{Q}(\mathbf{f}, \mathbf{c}) \wedge \exists \mathbf{f}' \mathbf{V}(\mathbf{f}', \mathbf{c})] \vee [\exists \mathbf{f}' \mathbf{Q}(\mathbf{f}', \mathbf{c}) \wedge \mathbf{V}(\mathbf{f}, \mathbf{c})]$

Temporal query Languages : more about T-WHILE

T-WHILE semantics can be described as follows ..... (in relationship with RTM)

input tape

I_1	I_2			I_{i_0}		I_n	$\mathcal{R}$

auxiliary tape

J_1	J_2			J_{i_0}		J_n	$\mathcal{S}$

store = output tape

K	$\mathcal{S}_o$

| **input tape** | = | **auxiliary tape** |

cursors on input and auxiliary tapes are **synchronized**

the **ouput** is one of the store relations

configuration = auxiliary tape + store + cursor value

While Q_B **Do**
$Body$

halting condition

iteration 1	$\longrightarrow$	configuration 1
$\ldots$	$\ldots$	$\ldots$
iteration k	$\longrightarrow$	configuration k
iteration $k+1$	$\longrightarrow$	configuration $k+1$

=

Relational Temporal Machines : complexity and temporal languages

- LinSpace-RTM = T-WHILE

(T-WHILE $\subseteq$ LinSpace-RTM easy.

LinSpace-RTM $\subseteq$ T-WHILE requires to simulate an auxiliary tape of length kn by a tape of length n.)

- PSpace-RTM = TS-WHILE

(TS-WHILE $\subseteq$ LinSpace-RTM requires to simulate:

generalized time stamped relations in an implicit manner, and

TS-FO*queries by FO queries

PSpace-RTM $\subseteq$ TS-WHILE is based on the converse encoding.)

Space complexity : How much auxiliary tape does a temporal computation needs ?

Based on extended one tape RTM

(we consider RTM whose output is always given by the store)

Relational Temporal Machines: Complexity and temporal languages

- RPM$_k$ $\subset$ RRM$_k$

(1. propositional-RPM = regular languages. 2. $a^n b^n$ is "recognized" by a RRM)

- TS-FO $\subsetneq$ RPM, • ETL $\subsetneq$ RPM, • T-WHILE$_{\text{shared}}$ $\subseteq$ RPM (strict inclusion claimed).

- T-WHILE$^{\text{bind}}_{\text{shared}}$ = RRM

T-WHILE$^{\text{bind}}_{\text{shared}}$ is T-WHILE **restricted** to shared auxiliary schema (no temporal auxiliary tape)
extended with temporal variables t_i

```
while not(first) do left end;
↓t';
while not (t'=t) do ( right; ↓t' ) end.
```

Constant Space RTM: no auxiliary temporal tape , just a store

registers needed $\longrightarrow$ general case : RRM$_k$
(Relational Register Machine)
stack discipline over registers : RPM$_k$
(Relational Pebble Machine)

Outline of the talk

- **Discussion and Future Work**

Relational Temporal Machines : Discussion and Future Work

- **Investigating features/parameters of** RTM
  - ▶ finite versus infinite (input) tape
  - ▶ restriction over cursor moves

    (right move only, restriction on reversal move, synchronization of input and auxiliary moves)
  - ▶ data model

    (relational + abstract data types, XML, specific domains)
  - ▶ ”internal” query language

    (inflationary, FO + aggregate, Xpath, ...)

  ⟶ applications (streaming, multimedia, ...)

- **Defining (declarative) languages**
  - ▶ SQTL (window + extensions of TL)
  - ▶ T-WHILE extensions
  - ▶ other logic based languages