On the Introduction of Time in Distributed Blackboard Rules

Jean-Marie Jacquet, Isabelle Linden, Mihail-Octavian Staicu

University of Namur, Namur - Belgium

September 11^{th}, 2013

Contents

UNIVERSITÉ DE NAMUR

Outline

Blackboard rules usage

Used to

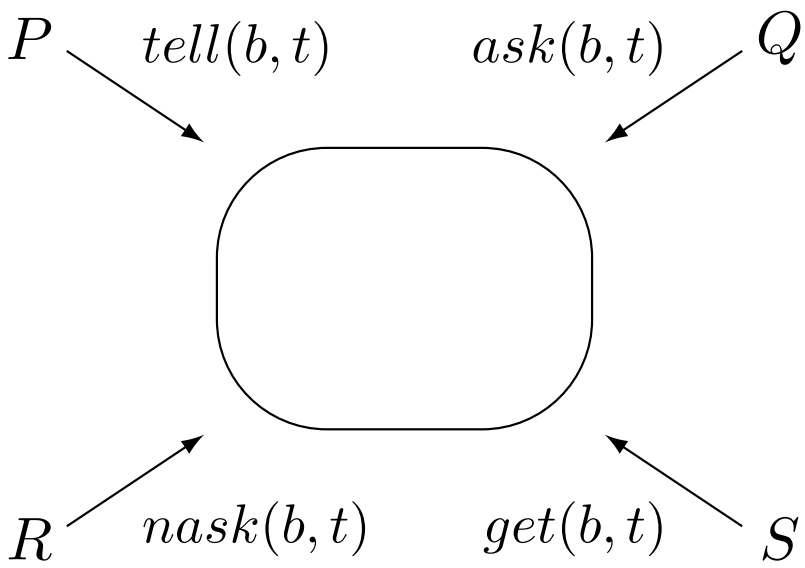
- provide a more dynamic behaviour to coordination languages
- observe the occurrence of events
- monitor the content of the blackboard and its changes
- enable reactivity: **LHS** (condition) $\longrightarrow$ **RHS** (action)

Coordination languages and time

Introducing time helps to

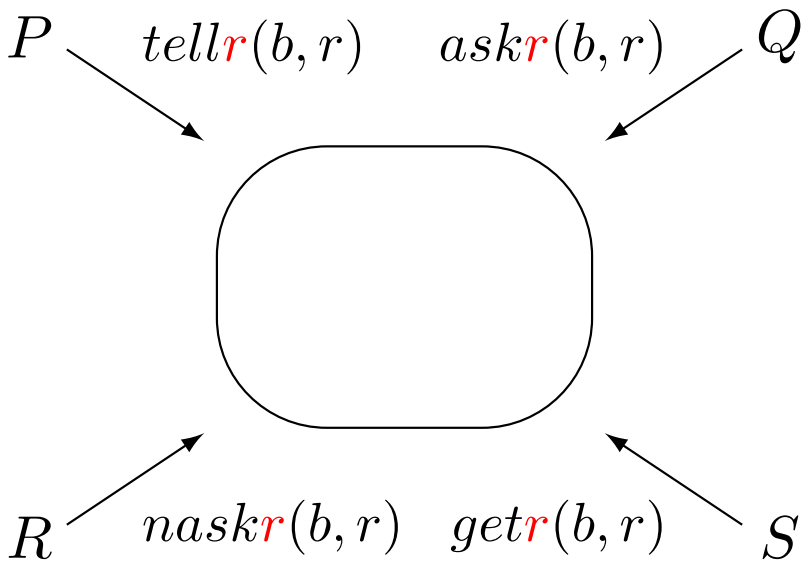
- model real-life coordination applications
- control the availability of data on the blackboard
- give different meanings to blackboard contexts
- observe ordered sequences of events

Bach basic model



$$
\begin{aligned}
C &::= tell(b,t) \mid ask(b,t) \mid nask(b,t) \mid get(b,t) \\
A &::= C \mid A \,;\, A \mid A \,||\, A \mid A \,+\, A
\end{aligned}
$$

Bach extended model



$$C \quad ::= \quad tellr(b,r) \mid askr(b,r) \mid naskr(b,r) \mid getr(b,r)$$

Outline

UNIVERSITÉ DE NAMUR

Reactive blackboards

- Action triggers based on blackboard content (*context-awareness*)
- in **primitive**

  $in(b, u)$: the tuple u is present on the blackboard b
- nin **primitive**

  $nin(b, u)$: the tuple u is absent from the blackboard b
- Establishing relations (links) between remote blackboards
- **R-rules**

$$in(b_1, u_1), \cdots, in(b_n, u_n), nin(b_{n+1}, u_{n+1}), \cdots, nin(b_m, u_m) \longrightarrow$$
$$in(b_{m+1}, u_{m+1}), \cdots, in(b_p, u_p), nin(b_{p+1}, u_{p+1}), \cdots, nin(b_q, u_q)$$

- In practice: $b_1 = \ldots = b_n = b_{n+1} = \ldots = b_m$

Examples (1)

Traffic management in a smart city

- highway leading to the city center with two lanes in each direction
- dynamic lane allocation according to incoming/outgoing traffic

Examples (1)

Traffic management in a smart city

- highway leading to the city center with two lanes in each direction
- dynamic lane allocation according to incoming/outgoing traffic

A possible solution

- $in(\text{inHeavyTraffic}, [06:10]) \longrightarrow$
  $in(\text{enter1}), in(\text{enter2}), in(\text{enter3}), in(\text{exit4})$

Examples (1)

Traffic management in a smart city

- highway leading to the city center with two lanes in each direction
- dynamic lane allocation according to incoming/outgoing traffic

A possible solution

- $in(\text{inHeavyTraffic}, [06:10]) \longrightarrow$
  $in(\text{enter1}), in(\text{enter2}), in(\text{enter3}), in(\text{exit4})$
- $in(\text{outHeavyTraffic}, [16:20]), nin(\text{concert}),$
  $nin(\text{sportEvent}) \longrightarrow$
  $in(\text{enter1}), in(\text{exit2}), in(\text{exit3}), in(\text{exit4})$

Examples (1)

Traffic management in a smart city

- highway leading to the city center with two lanes in each direction
- dynamic lane allocation according to incoming/outgoing traffic

A possible solution

- $in(\text{inHeavyTraffic}, [06:10]) \longrightarrow$
  $in(\text{enter1}), in(\text{enter2}), in(\text{enter3}), in(\text{exit4})$
- $in(\text{outHeavyTraffic}, [16:20]), nin(\text{concert}),$
  $nin(\text{sportEvent}) \longrightarrow$
  $in(\text{enter1}), in(\text{exit2}), in(\text{exit3}), in(\text{exit4})$
- $in(\text{inLowTraffic}), in(\text{outLowTraffic}),$
  $\longrightarrow$
  $in(\text{enter1}), in(\text{enter2}), in(\text{exit3}), in(\text{exit4})$

Examples (2)

Environmental monitoring in a smart city

- small weather stations monitor the quality of air in the city center
- divert traffic if pollution levels rise above accepted limits

Examples (2)

Environmental monitoring in a smart city

- small weather stations monitor the quality of air in the city center
- divert traffic if pollution levels rise above accepted limits

A possible solution

- $in(\text{qualityAlarm}, [10:16]), nin(\text{wind}) \longrightarrow in(\text{blockCirculation})$

Examples (2)

Environmental monitoring in a smart city

- small weather stations monitor the quality of air in the city center
- divert traffic if pollution levels rise above accepted limits

A possible solution

- $in(\text{qualityAlarm}, [10:16]), nin(\text{wind}) \longrightarrow in(\text{blockCirculation})$
- $nin(\text{qualityAlarm}, [10:16]) \longrightarrow in(\text{allowCirculation})$

Examples (3)

Cultural aspects

- customize subscriptions to different events
- receive on Saturday all the announcements related to "$theater$" published from Monday to Friday

Examples (3)

Cultural aspects

- customize subscriptions to different events
- receive on Saturday all the announcements related to "$theater$" published from Monday to Friday

A possible solution

- $in(\text{eventManager}, \text{theater}, [1:5]) \longrightarrow in(\text{user}, \text{theater}, 6)$

Examples (3)

Cultural aspects

- customize subscriptions to different events
- receive on Saturday all the announcements related to $"theater"$ published from Monday to Friday

A possible solution

- $in(\text{eventManager}, \text{theater}, [1:5]) \longrightarrow in(\text{user}, \text{theater}, 6)$
- $tellr(\text{eventManager},$
  $''in(\text{eventManager}, \text{theater}, [1:5]) \longrightarrow in(\text{user}, \text{theater}, 6)''$
  $)$

Examples (3)

Cultural aspects

- customize subscriptions to different events
- receive on Saturday all the announcements related to $"theater"$ published from Monday to Friday

A possible solution

- $in(\text{eventManager}, \text{theater}, [1:5]) \longrightarrow in(\text{user}, \text{theater}, 6)$
- $tellr(\text{eventManager},$
  $''in(\text{eventManager}, \text{theater}, [1:5]) \longrightarrow in(\text{user}, \text{theater}, 6)''$
  $)$
- $getr(\text{eventManager},$
  $''in(\text{eventManager}, \text{theater}, [1:5]) \longrightarrow in(\text{user}, \text{theater}, 6)''$
  $)$

Generic representation (1)

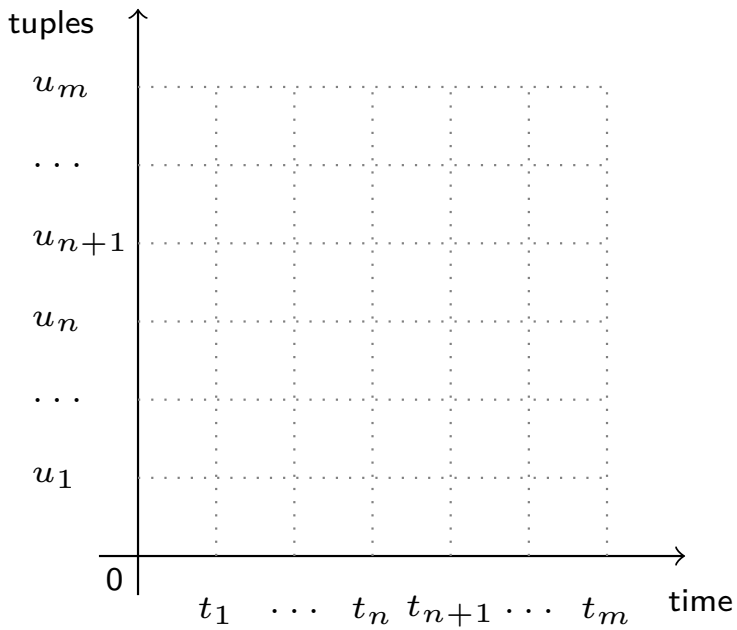
$$in(b, u_1, t_1), \ldots, in(b, u_1, t_n), \ldots, in(b, u_n, t_1), \ldots, in(b, u_n, t_n)$$
$$nin(b, u_{n+1}, t_{n+1}), \ldots, nin(b, u_{n+1}, t_m),$$
$$\ldots, nin(b, u_m, t_{n+1}), \ldots, nin(b, u_m, t_m) \longrightarrow$$
$$in(b, u_{m+1}, t_{m+1}), \cdots, in(b, u_p, t_p),$$
$$nin(b, u_{p+1}, t_{p+1}), \cdots, nin(b, u_q, t_q)$$

Consistency condition: $min(t_{m+1}, \ldots, t_q) \geq max(t_1, \ldots, t_m)$

Generic representation (1)

$$in(b, u_1, t_1), \ldots, in(b, u_1, t_n), \ldots, in(b, u_n, t_1), \ldots, in(b, u_n, t_n)$$
$$nin(b, u_{n+1}, t_{n+1}), \ldots, nin(b, u_{n+1}, t_m),$$
$$\ldots, nin(b, u_m, t_{n+1}), \ldots, nin(b, u_m, t_m) \longrightarrow$$
$$in(b, u_{m+1}, t_{m+1}), \cdots, in(b, u_p, t_p),$$
$$nin(b, u_{p+1}, t_{p+1}), \cdots, nin(b, u_q, t_q)$$

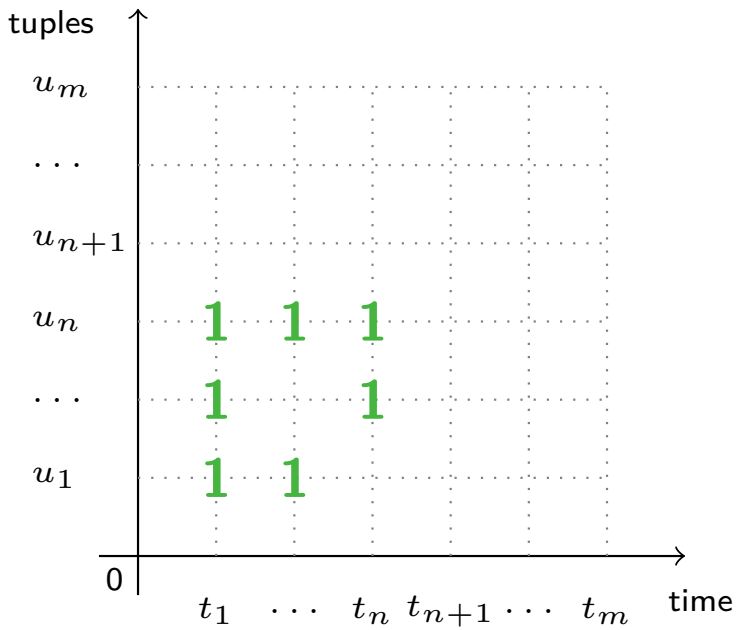
Consistency condition: $min(t_{m+1}, \ldots, t_q) \geq max(t_1, \ldots, t_m)$



Generic representation (1)

$$in(b, u_1, t_1), \ldots, in(b, u_1, t_n), \ldots, in(b, u_n, t_1), \ldots, in(b, u_n, t_n)$$
$$nin(b, u_{n+1}, t_{n+1}), \ldots, nin(b, u_{n+1}, t_m),$$
$$\ldots, nin(b, u_m, t_{n+1}), \ldots, nin(b, u_m, t_m) \longrightarrow$$
$$in(b, u_{m+1}, t_{m+1}), \cdots, in(b, u_p, t_p),$$
$$nin(b, u_{p+1}, t_{p+1}), \cdots, nin(b, u_q, t_q)$$

Consistency condition: $min(t_{m+1}, \ldots, t_q) \geq max(t_1, \ldots, t_m)$

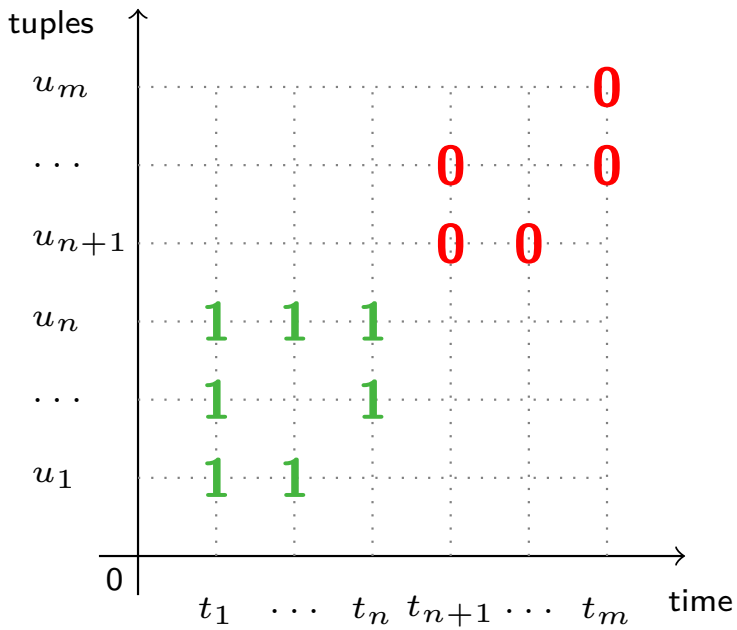


1 - information present

Generic representation (1)

$$in(b, u_1, t_1), \ldots, in(b, u_1, t_n), \ldots, in(b, u_n, t_1), \ldots, in(b, u_n, t_n)$$
$$nin(b, u_{n+1}, t_{n+1}), \ldots, nin(b, u_{n+1}, t_m),$$
$$\ldots, nin(b, u_m, t_{n+1}), \ldots, nin(b, u_m, t_m) \longrightarrow$$
$$in(b, u_{m+1}, t_{m+1}), \cdots, in(b, u_p, t_p),$$
$$nin(b, u_{p+1}, t_{p+1}), \cdots, nin(b, u_q, t_q)$$

Consistency condition: $min(t_{m+1}, \ldots, t_q) \geq max(t_1, \ldots, t_m)$

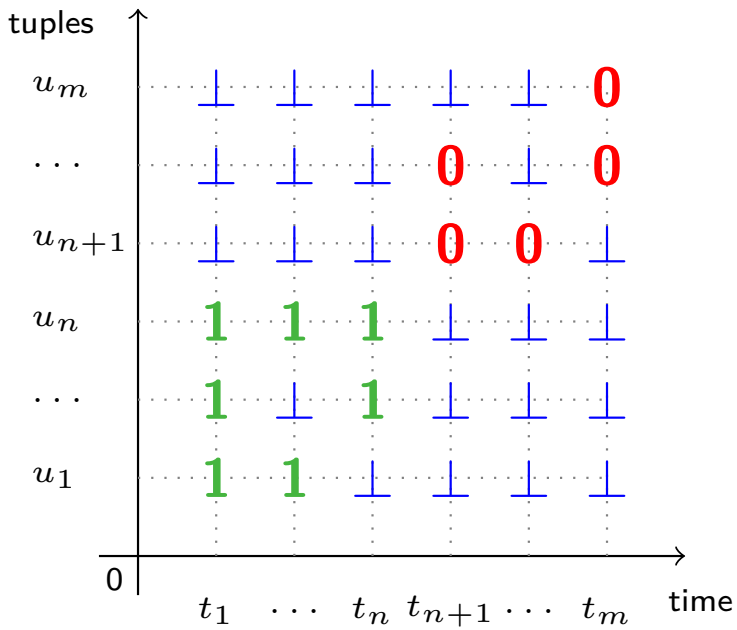


1 - information present **0** - information absent

Generic representation (1)

$$in(b, u_1, t_1), \ldots, in(b, u_1, t_n), \ldots, in(b, u_n, t_1), \ldots, in(b, u_n, t_n)$$
$$nin(b, u_{n+1}, t_{n+1}), \ldots, nin(b, u_{n+1}, t_m),$$
$$\ldots, nin(b, u_m, t_{n+1}), \ldots, nin(b, u_m, t_m) \longrightarrow$$
$$in(b, u_{m+1}, t_{m+1}), \cdots, in(b, u_p, t_p),$$
$$nin(b, u_{p+1}, t_{p+1}), \cdots, nin(b, u_q, t_q)$$

Consistency condition: $min(t_{m+1}, \ldots, t_q) \geq max(t_1, \ldots, t_m)$



1 - information present **0** - information absent ⊥ - no check

Generic representation (2)

- Model the activation condition in the form of a matrix

$$AM = \begin{cases} am_{i,j} = 1 & ,1 \leq i,j \leq n \text{ and } \exists\, in(b_i, u_i, t_j) \in LHS \\ am_{i,j} = 0 & ,n+1 \leq i,j \leq m \text{ and } \exists\, nin(b_i, u_i, t_j) \in LHS \\ am_{i,j} = \perp & ,1 \leq i,j \leq m \text{ in the remaining positions} \end{cases}$$

Generic representation (2)

- Model the activation condition in the form of a matrix

$$AM = \begin{cases} am_{i,j} = 1 & ,1 \leq i,j \leq n \text{ and } \exists\, in(b_i, u_i, t_j) \in LHS \\ am_{i,j} = 0 & ,n+1 \leq i,j \leq m \text{ and } \exists\, nin(b_i, u_i, t_j) \in LHS \\ am_{i,j} = \perp & ,1 \leq i,j \leq m \text{ in the remaining positions} \end{cases}$$

- Blackboard evolution handled through a counter matrix

$$CM = \begin{cases} cm_{i,j} = 0 & ,1 \leq i,j \leq n \text{ and } \exists\, in(b_i, u_i, t_j) \in LHS \\ cm_{i,j} = 0 & ,n+1 \leq i,j \leq m \text{ and } \exists\, nin(b_i, u_i, t_j) \in LHS \\ cm_{i,j} = \perp & ,1 \leq i,j \leq m \text{ in the remaining positions} \end{cases}$$

Generic representation (2)

- Model the activation condition in the form of a matrix

$$AM = \begin{cases} am_{i,j} = 1 & ,1 \leq i,j \leq n \text{ and } \exists\, in(b_i, u_i, t_j) \in LHS \\ am_{i,j} = 0 & ,n+1 \leq i,j \leq m \text{ and } \exists\, nin(b_i, u_i, t_j) \in LHS \\ am_{i,j} = \perp & ,1 \leq i,j \leq m \text{ in the remaining positions} \end{cases}$$

- Blackboard evolution handled through a counter matrix

$$CM = \begin{cases} cm_{i,j} = 0 & ,1 \leq i,j \leq n \text{ and } \exists\, in(b_i, u_i, t_j) \in LHS \\ cm_{i,j} = 0 & ,n+1 \leq i,j \leq m \text{ and } \exists\, nin(b_i, u_i, t_j) \in LHS \\ cm_{i,j} = \perp & ,1 \leq i,j \leq m \text{ in the remaining positions} \end{cases}$$

- $tell(u) \Rightarrow cm++$

Generic representation (2)

- Model the activation condition in the form of a matrix

$$AM = \begin{cases} am_{i,j} = 1 & ,1 \leq i,j \leq n \text{ and } \exists\, in(b_i, u_i, t_j) \in LHS \\ am_{i,j} = 0 & ,n+1 \leq i,j \leq m \text{ and } \exists\, nin(b_i, u_i, t_j) \in LHS \\ am_{i,j} = \perp & ,1 \leq i,j \leq m \text{ in the remaining positions} \end{cases}$$

- Blackboard evolution handled through a counter matrix

$$CM = \begin{cases} cm_{i,j} = 0 & ,1 \leq i,j \leq n \text{ and } \exists\, in(b_i, u_i, t_j) \in LHS \\ cm_{i,j} = 0 & ,n+1 \leq i,j \leq m \text{ and } \exists\, nin(b_i, u_i, t_j) \in LHS \\ cm_{i,j} = \perp & ,1 \leq i,j \leq m \text{ in the remaining positions} \end{cases}$$

- $tell(u) \Rightarrow cm++$
- $get(u)$, expired $u \Rightarrow cm--$

Time variants

Different approaches may be taken

- **discrete time points**: strict
- **continous time intervals**: flexible
- **absolute time**: single ativation
- **relative time**: repeated activation

Discrete time

- tuples should be present or absent as specific **time points**
- activation occurs when

$$\begin{cases} cm_{1,i} \geq am_{1,i} & , 1 \leq i \leq n, \\ cm_{1,j} = 0 & , n+1 \leq j \leq m, \end{cases}$$

Continuous time (1)

- tuples should be present or absent within specific **time intervals**
- Rule representation

$$in(b, u_1, [t_{1,1} : t_{1,2}]), \ldots, in(b, u_n, [t_{n,1} : t_{n,2}]),$$
$$nin(b, u_{n+1}, [t_{n+1,1} : t_{n+1,2}]), \ldots, nin(b, u_m, [t_{m,1} : t_{m,2}]) \longrightarrow$$
$$in(b_{m+1}, u_{m+1}, t_{m+1}), \cdots, in(b_p, u_p, t_p),$$
$$nin(b_{p+1}, u_{p+1}, t_{p+1}), \cdots, nin(b_q, u_q, t_q)$$

Continuous time (1)

- $AM = \begin{pmatrix} 1 & \dots & 1 & 0 & \dots & 0 \\ t_{1,1} & \dots & t_{n,1} & t_{n+1,1} & \dots & t_{m,1} \\ t_{1,2} & \dots & t_{n,2} & t_{n+1,2} & \dots & t_{m,2} \end{pmatrix}$
- $CM = \begin{pmatrix} bc_1 & \dots & bc_n & bc_{n+1} & \dots & bc_m \\ fi_1 & \dots & fi_n & fi_{n+1} & \dots & fi_m \\ li_1 & \dots & li_n & li_{n+1} & \dots & li_m \end{pmatrix}$
- Activation condition

$$\begin{cases} cm_{1,i} \geq am_{1,i} & , 1 \leq i \leq n, \\ cm_{1,j} = 0 & , n+1 \leq j \leq m, \\ cm_{2,k} \geq am_{2,k} & , 1 \leq k \leq m, \\ cm_{3,l} \leq am_{3,l} & , 1 \leq l \leq m. \end{cases}$$

Outline

Implementation

How to proceed . . .

Implementation

How to proceed . . .

- Can the system capture an event at the right time with pin-point accuracy?

Implementation

How to proceed ...

- Can the system capture an event at the right time with pin-point accuracy?
- Can we a-priori predict when pieces of information become available?

Implementation

How to proceed ...

- Can the system capture an event at the right time with pin-point accuracy?
- Can we a-priori predict when pieces of information become available?
- Shouldn't we have some degree of flexibility?

Implementation

How to proceed . . .

- Can the system capture an event at the right time with pin-point accuracy?
- Can we a-priori predict when pieces of information become available?
- Shouldn't we have some degree of flexibility?

The continuous approach - more suitable in real-life scenarios

Architecture

- Generic client-server

Architecture

- Generic client-server
- Server-side
  - blackboard and its operations
  - communication
  - rule space
  - request space

Architecture

- Generic client-server
- Server-side
  - blackboard and its operations
  - communication
  - rule space
  - request space
- Client-side
  - parse string representations of agents
  - dispatch requests to the designated blackboards

Adding time (1)

- Syntactic shortcut for mulptiple instances

$$in(b,u) \ldots in(b,u) \Rightarrow in_c(b,u)$$

Adding time (1)

- Syntactic shortcut for mulptiple instances

$$in(b,u)\ldots in(b,u) \Rightarrow in_c(b,u)$$

- Activation matrix

$$AM = \begin{pmatrix} c_1 & \ldots & c_n & 0 & \ldots & 0 \\ t_{1,1} & \ldots & t_{n,1} & t_{n+1,1} & \ldots & t_{m,1} \\ t_{1,2} & \ldots & t_{n,2} & t_{n+1,2} & \ldots & t_{m,2} \end{pmatrix}$$

Adding time (1)

- Syntactic shortcut for mulptiple instances

$$in(b,u)\ldots in(b,u) \Rightarrow in_c(b,u)$$

- Activation matrix

$$AM = \begin{pmatrix} c_1 & \ldots & c_n & 0 & \ldots & 0 \\ t_{1,1} & \ldots & t_{n,1} & t_{n+1,1} & \ldots & t_{m,1} \\ t_{1,2} & \ldots & t_{n,2} & t_{n+1,2} & \ldots & t_{m,2} \end{pmatrix}$$

- Counter matrix

$$CM = \begin{pmatrix} dict_1 & \ldots & dict_n & dict_{n+1} & \ldots & dict_m \end{pmatrix}$$

$$dict_i = \{< instance_1, timestamp_1 >, \ldots, < instance_{last}, timestamp_{last} >\}$$

Adding time (2)

- Valid instances for activation

$$bc_i = |VI|,\ VI = \{instance_i \mid t_{i,1} \leq timestamp_i \leq t_{i,2}, 1 \leq i \leq m\}$$

Adding time (2)

- Valid instances for activation

$$bc_i = |VI|,\ VI = \{instance_i \mid t_{i,1} \leq timestamp_i \leq t_{i,2}, 1 \leq i \leq m\}$$

- Activation condition

$$\begin{cases} bc_i \geq c_i & , 1 \leq i \leq n, \\ bc_j = 0 & , n+1 \leq j \leq m, \end{cases}$$

Adding time (2)

- Valid instances for activation

$$bc_i = |VI|,\ VI = \{instance_i \mid t_{i,1} \leq timestamp_i \leq t_{i,2}, 1 \leq i \leq m\}$$

- Activation condition

$$\begin{cases} bc_i \geq c_i & , 1 \leq i \leq n, \\ bc_j = 0 & , n+1 \leq j \leq m, \end{cases}$$

- Activation possibilities

$$\prod_{k=1}^{n} \binom{bc_k}{c_k}$$

Outline

Future work

- Explore the operational semantics
- Expressiveness studies
- Implementation improvements
- Performance testing

Any questions?

Thank you for your attention!