Distributed Runtime Monitoring

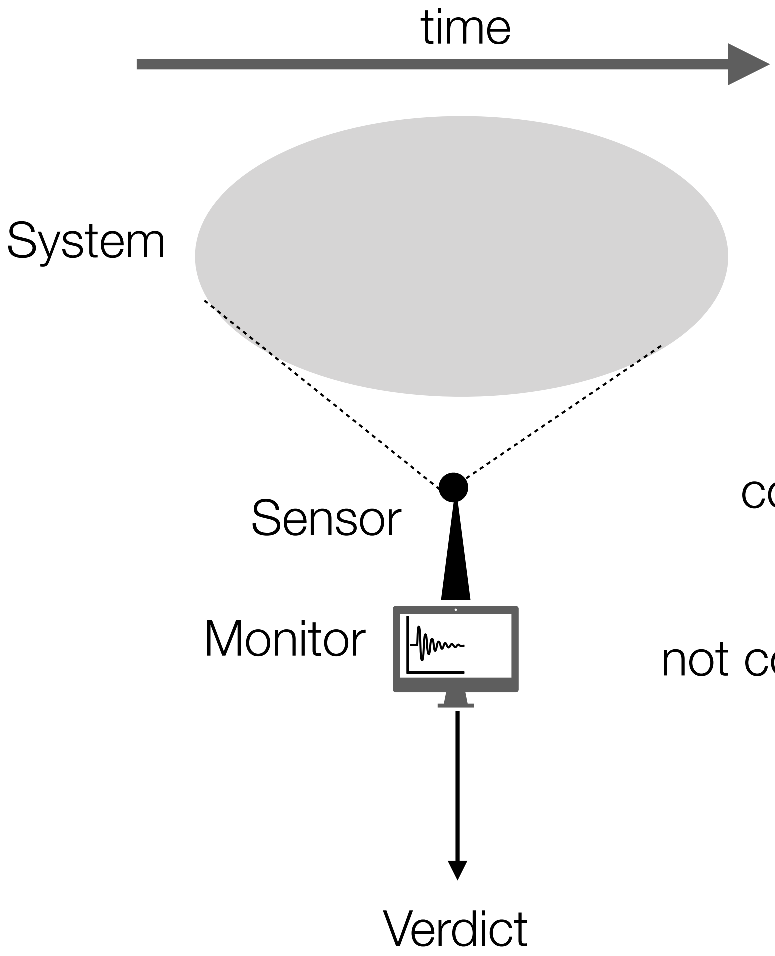
Pierre Fraigniaud
Institut de Recherche en Informatique Fondamentale (IRIF)
CNRS and Université Paris Cité

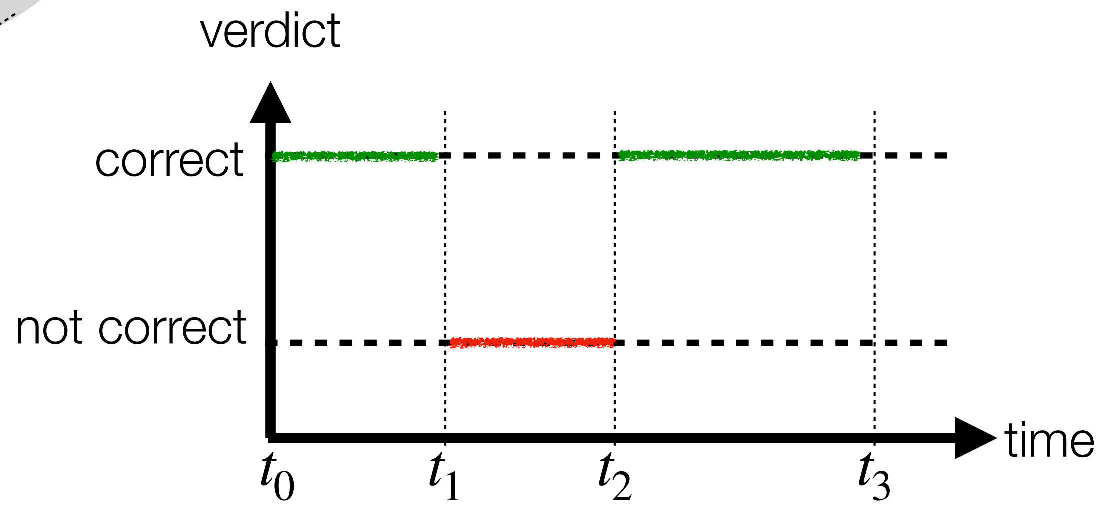
Joint work with:

- Borzoo Bonakdarpour, Michigan State University, U.S.A.
- Sergio Rajsbaum, UNAM, México
- David Rosenblueth, UNAM, México
- Corentin Travers, Bordeaux University, France

DISC Satellite Workshop FRIDA 2023, L'Aquila, October 9, 2023

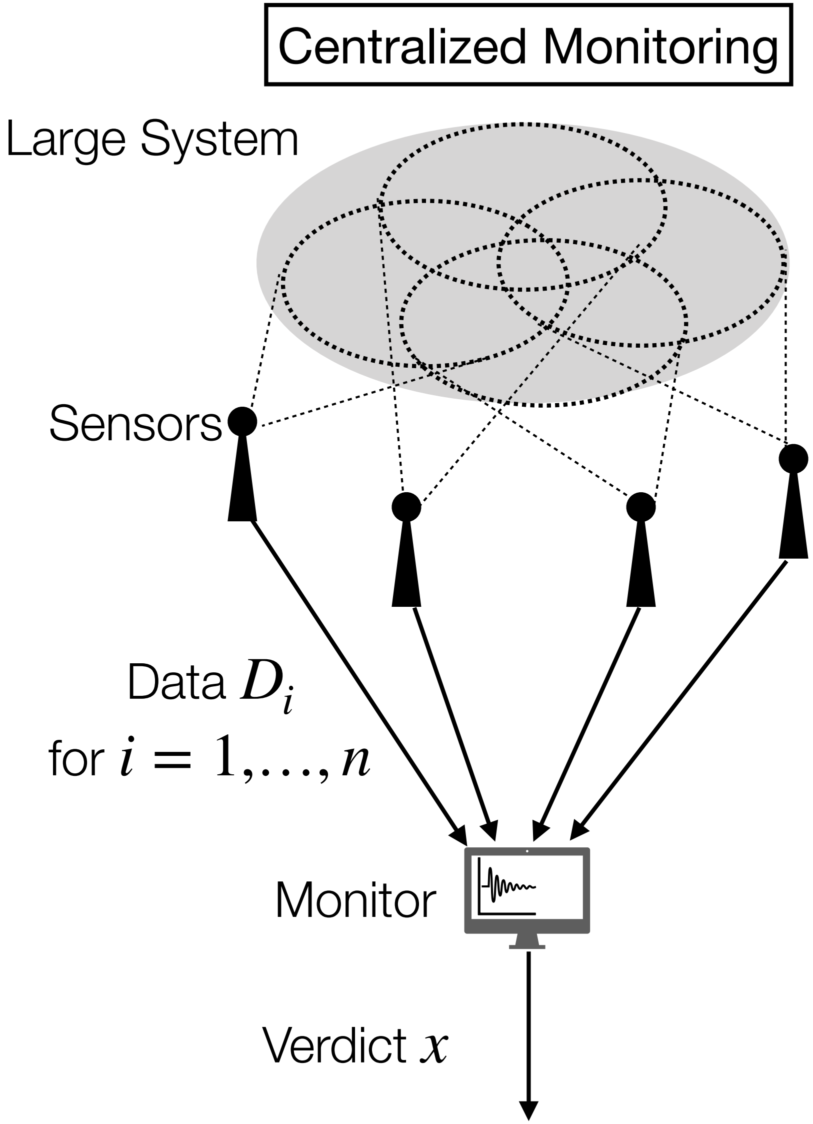
Runtime Monitoring



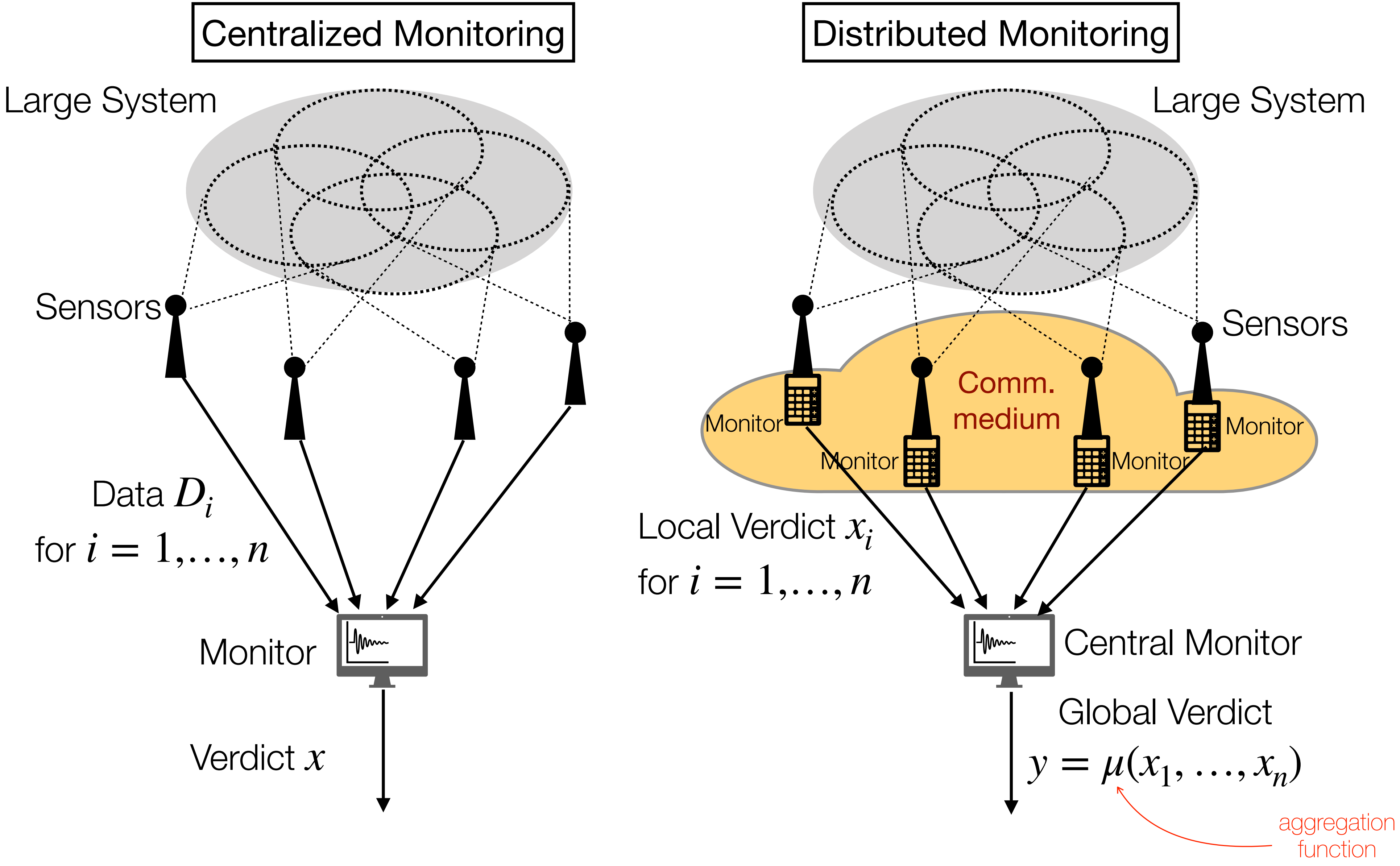


Runtime Monitoring Large Remote Systems

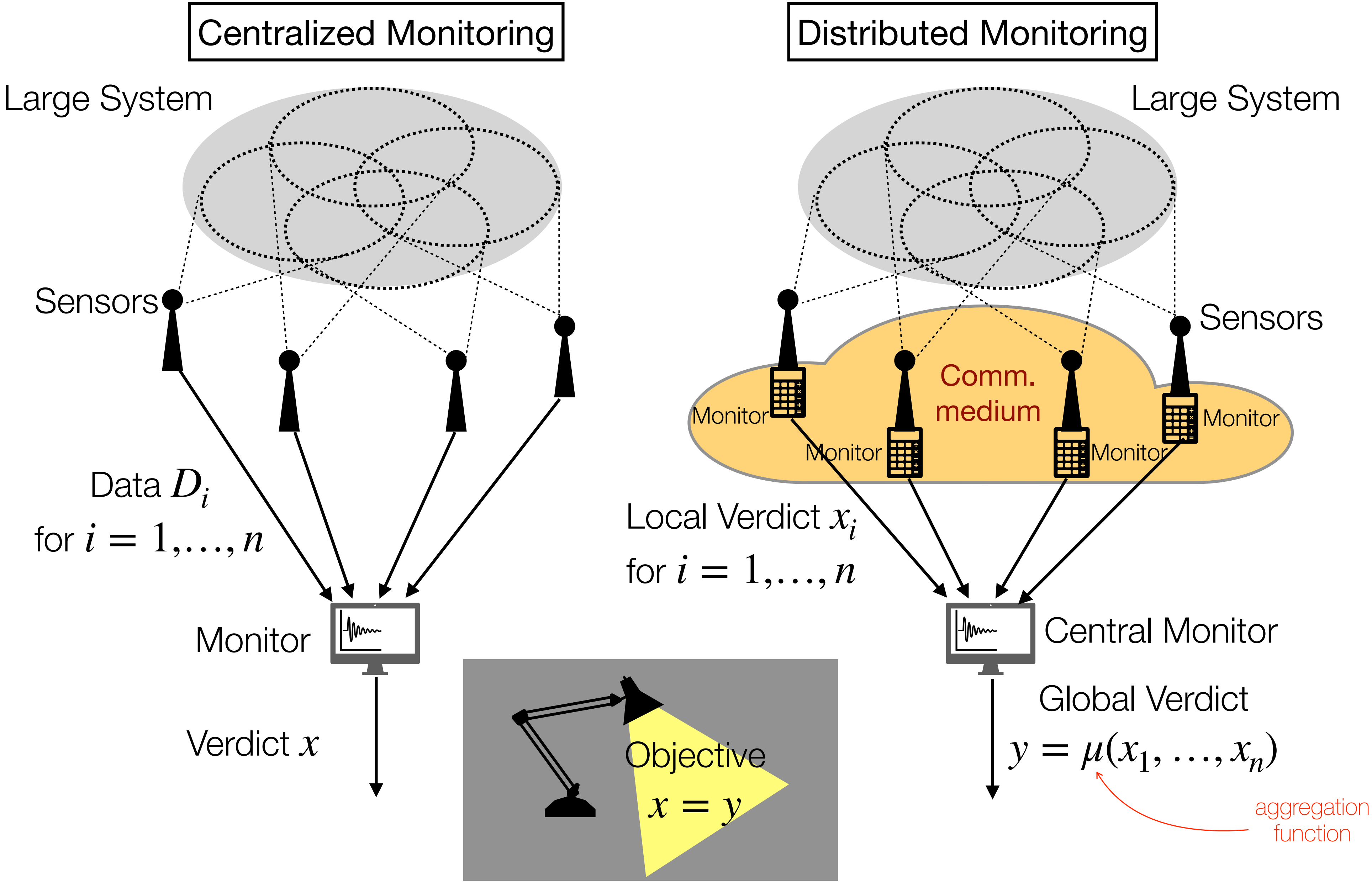
Runtime Monitoring Large Remote Systems



Runtime Monitoring Large Remote Systems



Runtime Monitoring Large Remote Systems



Linear Temporal Logic (LTL)

Linear Temporal Logic (LTL)

LTL applies to infinite traces $\sigma = s_0 s_1 s_2 \cdots$ where $s_i \in \Sigma = 2^{\mathsf{AP}}$

Linear Temporal Logic (LTL)

LTL applies to infinite traces $\sigma = s_0 s_1 s_2 \cdots$ where $s_i \in \Sigma = 2^{\mathsf{AP}}$

Logical operators ¬ and ∨, and temporal operator X (*next*) and U (*until*):

- $[\sigma \models \mathsf{X}\varphi] \iff [\sigma^1 \models \varphi]$ where $\sigma^i = s_i s_{i+1} \ldots$
- $[\sigma \models \varphi \mathsf{U} \psi] \iff \exists i \geq 0 : ([\sigma^i \models \psi]) \wedge (\forall 0 \leq j < i, [\sigma^j \models \varphi])$

Linear Temporal Logic (LTL)

LTL applies to infinite traces $\sigma = s_0 s_1 s_2 \cdots$ where $s_i \in \Sigma = 2^{\mathsf{AP}}$

Logical operators ¬ and ∨, and temporal operator X (*next*) and U (*until*):

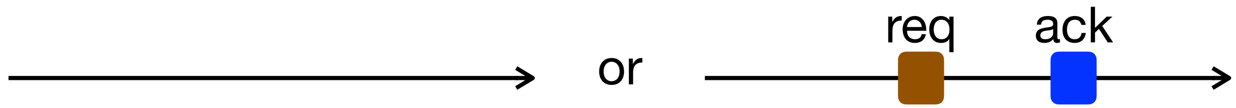
- $[\sigma \vDash \mathsf{X}\varphi] \iff [\sigma^1 \vDash \varphi]$ where $\sigma^i = s_i s_{i+1} \ldots$
- $[\sigma \vDash \varphi \mathsf{U} \psi] \iff \exists i \geq 0 : ([\sigma^i \vDash \psi]) \wedge (\forall 0 \leq j < i, [\sigma^j \vDash \varphi])$

Enables to build other logical operators (∧, →, ↔, true, false) and other temporal operators, such as:

- F (*finally*): $\mathsf{F}\,\psi \equiv \mathsf{true}\,\mathsf{U}\,\psi$
- G (*globally*): $\mathsf{G}\,\psi \equiv \neg(\mathsf{F}\neg\psi)$
- R (*release*): $\psi\,\mathsf{R}\,\varphi \equiv \neg(\neg\psi\,\mathsf{U}\,\neg\varphi)$

Example: Req./Ack.

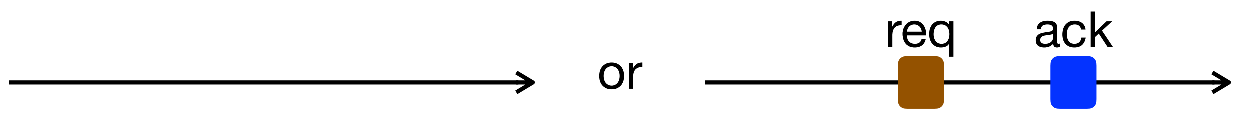
$$\varphi_{ra} = \mathsf{G}(\neg r \wedge \neg a) \vee \big((\neg a\,\mathsf{U}\,r) \wedge \mathsf{F}a\big)$$



- $\mathsf{G}(\neg a \wedge \neg r)$ = there are no req and no ack
- $(\neg a\,\mathsf{U}\,r) \wedge \mathsf{F}a$ = a req eventually occurs, not ack occur before that, and an ack must eventually occur.

Example: Req./Ack.

$$\varphi_{ra} = \mathsf{G}(\neg r \wedge \neg a) \vee \big((\neg a \,\mathsf{U}\, r) \wedge \mathsf{F}a\big)$$



- $\mathsf{G}(\neg a \wedge \neg r)$ = there are no req and no ack
- $(\neg a \,\mathsf{U}\, r) \wedge \mathsf{F}a$ = a req eventually occurs, not ack occur before that, and an ack must eventually occur.

$$\varphi_{ra2} = \Big(\mathsf{G}(\neg a_1 \wedge \neg r_1) \vee [(\neg a_1 \,\mathsf{U}\, r_1) \wedge \mathsf{F}a_1]\Big)$$

$$\wedge \Big(\mathsf{G}(\neg a_2 \wedge \neg r_2) \vee [(\neg a_2 \,\mathsf{U}\, r_2) \wedge \mathsf{F}a_2]\Big)$$

Finite LTL

Finite LTL (FLTL) is essentially LTL on finite traces $\alpha = s_0 s_1 \cdots s_t$

$$\left[\alpha \vDash_F \mathsf{N}\, \varphi\right] = \begin{cases} \left[\alpha^1 \vDash_F \varphi\right] & \text{if } \alpha^1 \neq \epsilon \\ \bot & \text{otherwise} \end{cases}$$

$$\left[\alpha \vDash_F \varphi \,\mathsf{U}\, \psi\right] = \begin{cases} \top & \text{if } \exists i \in \{0, \ldots, t\} : \Big(([\alpha^i \vDash_F \psi] = \top\ ) \\ & \quad \wedge (\forall j \in \{0, \ldots, i-1\}, [\alpha^j \vDash_F \varphi] = \top\ ) \Big) \\ \bot & \text{otherwise} \end{cases}$$

Multivalued Logic(s)

Multivalued Logic(s)

3-valued LTL (LTL_3):

$$[\alpha \models_3 \varphi] = \begin{cases} \top & \text{if} \quad \forall \sigma \in \Sigma^\omega : \alpha\sigma \models \varphi \\ \bot & \text{if} \quad \forall \sigma \in \Sigma^\omega : \alpha\sigma \not\models \varphi \\ ? & \text{otherwise} \end{cases}$$

Multivalued Logic(s)

3-valued LTL (LTL3):

$$[\alpha \models_3 \varphi] = \begin{cases} \top & \text{if} \quad \forall \sigma \in \Sigma^\omega : \alpha\sigma \models \varphi \\ \bot & \text{if} \quad \forall \sigma \in \Sigma^\omega : \alpha\sigma \not\models \varphi \\ ? & \text{otherwise} \end{cases}$$

4-valued LTL (RV-LTL):

$$[\alpha \models_4 \varphi] = \begin{cases} \top & \text{if} \quad [\alpha \models_3 \varphi] = \top \\ \bot & \text{if} \quad [\alpha \models_3 \varphi] = \bot \\ \top_p & \text{if} \quad [\alpha \models_3 \varphi] = ? \ \wedge \ [\alpha \models_F \varphi] = \top \\ \bot_p & \text{if} \quad [\alpha \models_3 \varphi] = ? \ \wedge \ [\alpha \models_F \varphi] = \bot \end{cases}$$

Multivalued Logic(s)

3-valued LTL (LTL_3):

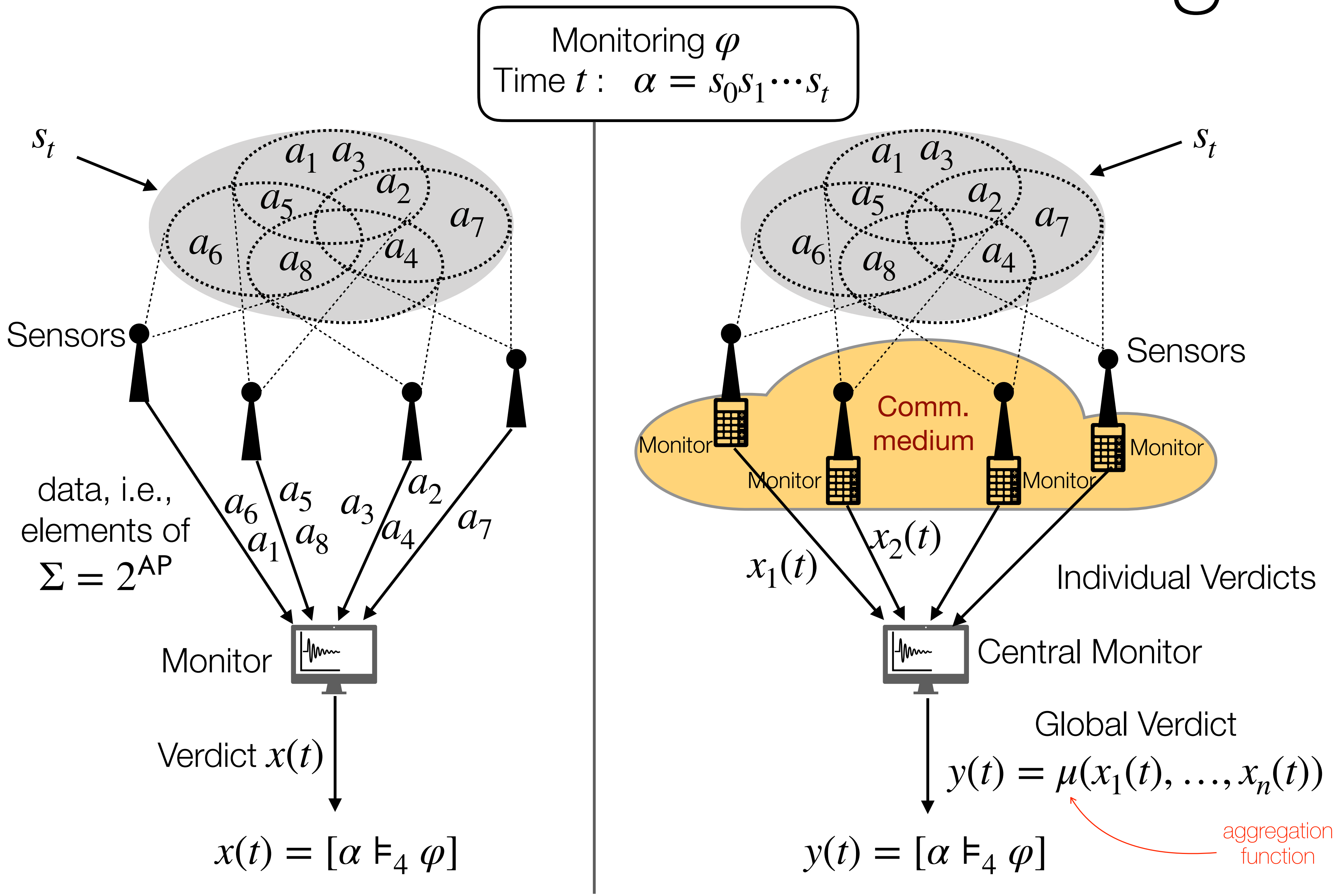
$$[\alpha \models_3 \varphi] = \begin{cases} \top & \text{if} \quad \forall \sigma \in \Sigma^\omega : \alpha\sigma \models \varphi \\ \bot & \text{if} \quad \forall \sigma \in \Sigma^\omega : \alpha\sigma \not\models \varphi \\ ? & \text{otherwise} \end{cases}$$

Runtime Verification LTL (RV-LTL)

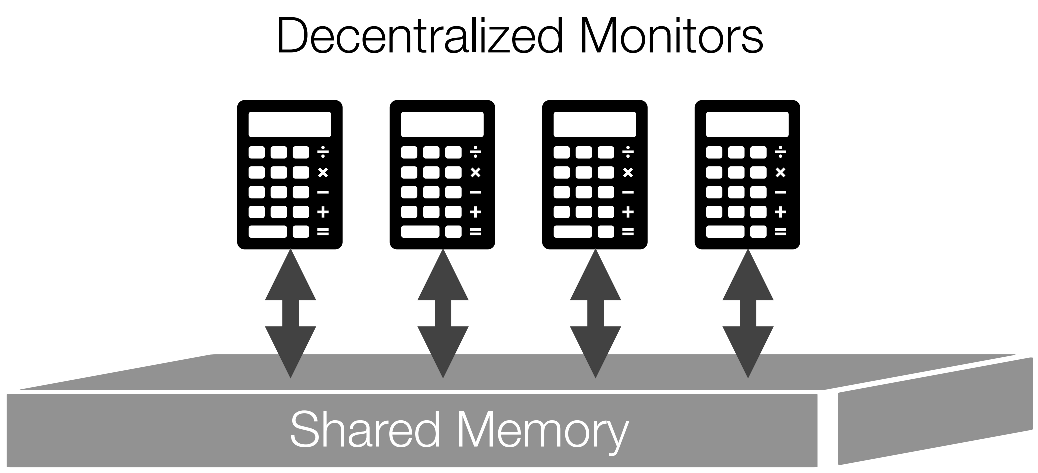
4-valued LTL (RV-LTL):

$$[\alpha \models_4 \varphi] = \begin{cases} \top & \text{if} \quad [\alpha \models_3 \varphi] = \top \\ \bot & \text{if} \quad [\alpha \models_3 \varphi] = \bot \\ \top_p & \text{if} \quad [\alpha \models_3 \varphi] = ? \wedge [\alpha \models_F \varphi] = \top \\ \bot_p & \text{if} \quad [\alpha \models_3 \varphi] = ? \wedge [\alpha \models_F \varphi] = \bot \end{cases}$$

Decentralized Monitoring



The Computational Model



Set of asynchronous crash-prone monitors

- Hypothesis 1: Shared memory with read/write accesses (we actually use IIS model)

- Hypothesis 2: Synchronization barrier between $\alpha' = s_0 s_1 \cdots s_{t-1}$ and $\alpha = s_0 s_1 \cdots s_t = \alpha' s_t$ for all $t \geq 1$

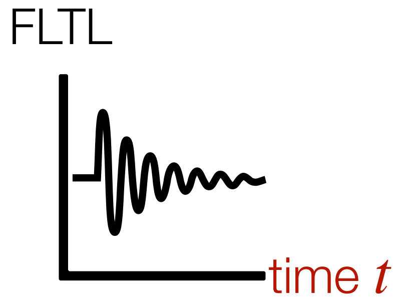
Monitors Get Partial Information

- Partial trace $\alpha = s_0 s_1 \cdots s_t$ where $\alpha' = s_0 s_1 \cdots s_{t-1}$ is fixed
- Monitors examine $s_t = \{a_1, \ldots, a_k\}$ for deciding $[\alpha \models_4 \varphi]$
- $w_i = \text{view}(p_i),\ i = 1, \ldots, n$, after some communication
- It may be the case that
  - $[\alpha' w_i \models_F \varphi] = \top$ for some $i \in [n]$
  - $[\alpha' w_j \models_F \varphi] = \bot$ for some $j \neq i$

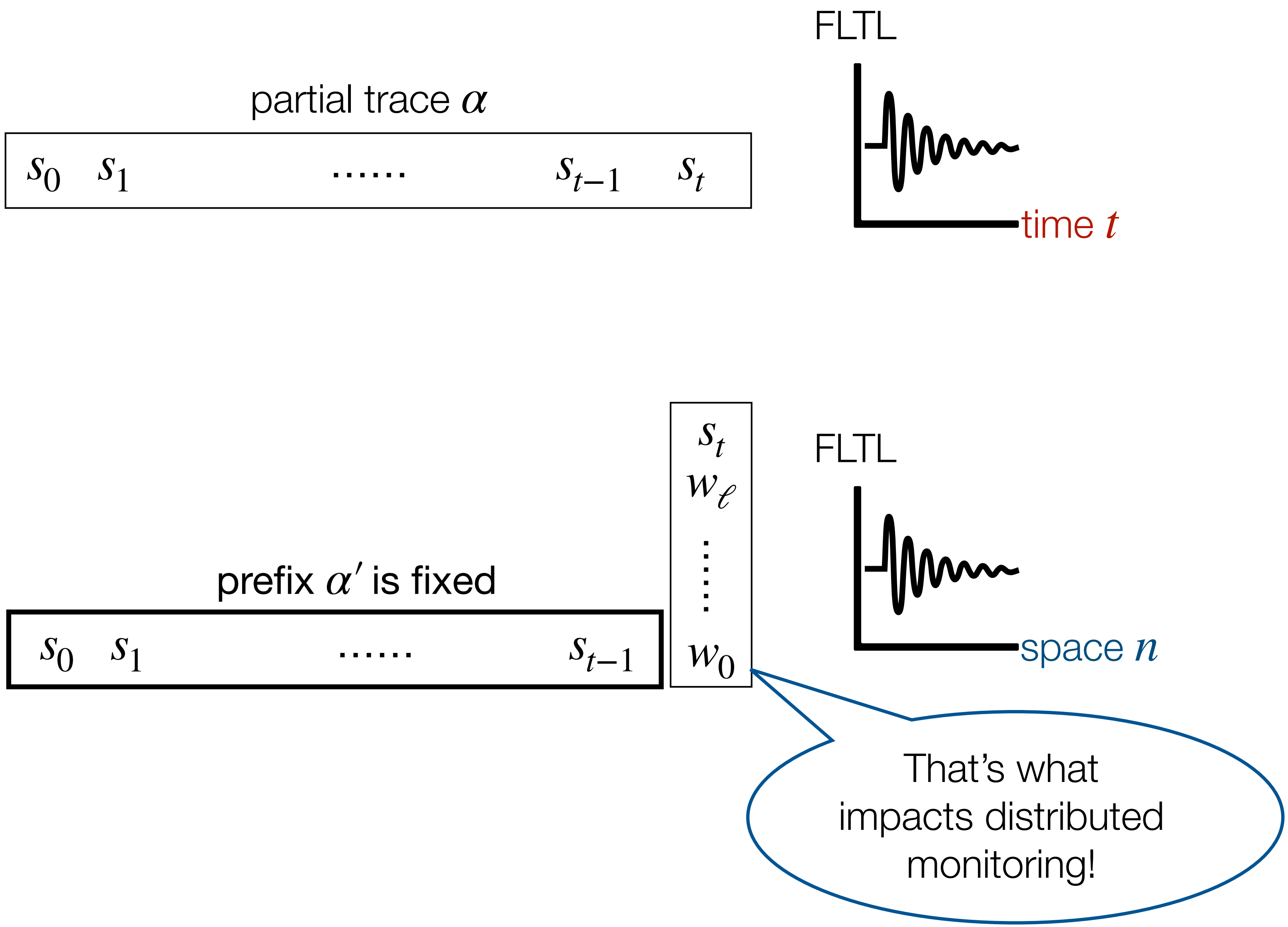
Evaluation of φ Evolves Across both Time and Space

partial trace α

s_0 s_1 …… s_{t-1} s_t



Evaluation of φ Evolves Across both Time and Space



Alternation Number

- The *alternation number* of an LTL formula φ with respect to a finite partial trace $\alpha = \alpha' s$, denoted by $AN(\varphi, \alpha)$, is the maximum integer $k \geq 0$ such that there exists a sequence of partial states (i.e., views) $w_0, \cdots, w_k$ with $w_0 = \emptyset$, $w_k = s$, and, for every $i \in \{0, \ldots, k-1\}$,

$$\left(w_i \subsetneq w_{i+1}\right) \quad \wedge \quad \left([\alpha' w_i \models_F \varphi] \neq [\alpha' w_{i+1} \models_F \varphi]\right)$$

- The alternation number of an LTL formula φ is

$$AN(\varphi) = \max \left\{AN(\varphi, \alpha) \mid \alpha \in \Sigma^{\star}\right\}$$

- Remark: $AN(\varphi) \leq |\mathsf{AP}|$

Example

$$\varphi_{ra2} = \Big(\mathsf{G}(\neg a_1 \wedge \neg r_1) \vee [(\neg a_1 \,\mathsf{U}\, r_1) \wedge \mathsf{F} a_1]\Big)$$

$$\wedge \Big(\mathsf{G}(\neg a_2 \wedge \neg r_2) \vee [(\neg a_2 \,\mathsf{U}\, r_2) \wedge \mathsf{F} a_2]\Big)$$

$s_0 = \{r_1, a_1, r_2, a_2\}$

- $w_0 = \emptyset$ ➜ $[w_0 \models_F \varphi_{ra2}] = \top$
- $w_1 = \{r_1\}$ ➜ $[w_1 \models_F \varphi_{ra2}] = \bot$
- $w_2 = \{r_1, a_1\}$ ➜ $[w_2 \models_F \varphi_{ra2}] = \top$
- $w_3 = \{r_1, a_1, r_2\}$ ➜ $[w_3 \models_F \varphi_{ra2}] = \bot$
- $w_4 = \{r_1, a_1, r_2, a_2\} = s_0$ ➜ $[w_4 \models_F \varphi_{ra2}] = \top$

$$AN(\varphi_{ra2}) = 4$$

Results

Theorem For every $k \geq 0$, there exists an LTL formula φ with $AN(\varphi) = k$ such that runtime verifying φ using distributed monitors requires a verdict set V with $|V| \geq k + 1$.

Theorem For every $k \geq 0$, and for every LTL formula φ with $AN(\varphi) = k$, there is distributed monitors that correctly monitor φ using verdict set

$$\mathbb{B}_{2\lceil k/2 \rceil + 4} = \{ \bot , \top , \bot_0 , \top_0 , \ldots, \bot_{\lceil k/2 \rceil} , \top_{\lceil k/2 \rceil} \}$$

Distributed LTL

Distributed LTL

Let $\alpha = \alpha' s$ be a finite partial trace

Distributed LTL

Let $\alpha = \alpha' s$ be a finite partial trace

$$
\left[\alpha \vDash_D \varphi\right] = \begin{cases}
\top & \text{if} \quad [\alpha \vDash_4 \varphi] = \top \\
\bot & \text{if} \quad [\alpha \vDash_4 \varphi] = \bot \\
\top_0 & \text{if} \quad [\alpha \vDash_4 \varphi] = \top_p \;\wedge\; (\forall w \subset s : [\alpha' w \vDash_D \varphi] = \top_0\,) \\
\bot_0 & \text{if} \quad [\alpha \vDash_4 \varphi] = \bot_p \;\wedge\; (\forall w \subset s : [\alpha' w \vDash_D \varphi] = \bot_0\,) \\
\top_i\,, i > 0 & \text{if} \quad [\alpha \vDash_4 \varphi] = \top_p \;\wedge\; (\,\exists w \subset s : [\alpha' w \vDash_D \varphi] = \bot_{i-1}\,) \\
 & \quad\quad \wedge\; (\forall w \subset s, \exists j < i : [\alpha' w \vDash_D \varphi] \in \{\, \top_j\,, \bot_j \,\} \cup \{\, \top_i \,\}) \\
\bot_i\; i > 0 & \text{if} \quad [\alpha \vDash_4 \varphi] = \bot_p \;\wedge\; (\,\exists w \subset s : [\alpha' w \vDash_D \varphi] = \top_{i-1}\,) \\
 & \quad\quad \wedge\; (\forall w \subset s, \exists j < i : [\alpha' w \vDash_D \varphi] \in \{\, \top_j\,, \bot_j \,\} \cup \{\, \bot_i \,\})
\end{cases}
$$

Distributed LTL

Let $\alpha = \alpha' s$ be a finite partial trace

$$[\alpha \vDash_D \varphi] = \begin{cases} \top & \text{if} \quad [\alpha \vDash_4 \varphi] = \top \\ \bot & \text{if} \quad [\alpha \vDash_4 \varphi] = \bot \\ \top_0 & \text{if} \quad [\alpha \vDash_4 \varphi] = \top_p \ \wedge \ (\forall w \subset s : [\alpha' w \vDash_D \varphi] = \top_0 \ ) \\ \bot_0 & \text{if} \quad [\alpha \vDash_4 \varphi] = \bot_p \ \wedge \ (\forall w \subset s : [\alpha' w \vDash_D \varphi] = \bot_0 \ ) \\ \top_i \, , i > 0 & \text{if} \quad [\alpha \vDash_4 \varphi] = \top_p \ \wedge \ (\ \exists w \subset s : [\alpha' w \vDash_D \varphi] = \bot_{i-1} \ ) \\ & \quad \wedge \ (\forall w \subset s, \exists j < i : [\alpha' w \vDash_D \varphi] \in \{ \ \top_j \, , \bot_j \ \} \cup \{ \ \top_i \ \}) \\ \bot_i \ i > 0 & \text{if} \quad [\alpha \vDash_4 \varphi] = \bot_p \ \wedge \ (\ \exists w \subset s : [\alpha' w \vDash_D \varphi] = \top_{i-1} \ ) \\ & \quad \wedge \ (\forall w \subset s, \exists j < i : [\alpha' w \vDash_D \varphi] \in \{ \ \top_j \, , \bot_j \ \} \cup \{ \ \bot_i \ \}) \end{cases}$$

$$AN(\varphi, \alpha) = \begin{cases} 0 & \text{if} \quad [\alpha \vDash_D \varphi] \in \{ \ \top \, , \bot \ \} \\ k & \text{if} \quad [\alpha \vDash_D \varphi] \in \{ \ \bot_k \, , \top_k \ \} \end{cases}$$

Reducing #Logical Values

DLTL^{+}

$\bot_0 < \top_0 < \bot_1 < \top_1 < \ldots < \top_{i-1} < \bot_i < \top_i < \bot_{i+1} < \ldots$

DLTL^{-}

$\top_0 < \bot_0 < \top_1 < \bot_1 < \ldots < \bot_{i-1} < \top_i < \bot_i < \top_{i+1} < \ldots$

Conclusion and Open Problems

- **Proof of concept:** Decentralized runtime monitoring of φ can be done, with verdicts in $\text{LTL}_{AN(\varphi)+O(1)}$

- **Conjecture:** For every φ, distributed monitoring of φ requires $AN(\varphi)$ different values.

- **Next step:** Getting rid of the synchronization barrier between $\alpha' = s_0 s_1 \ldots s_{t-1}$ and $\alpha = \alpha' s_t = s_0 s_1 \ldots s_{t-1} s_t$

Conclusion and Open Problems

- **Proof of concept:** Decentralized runtime monitoring of φ can be done, with verdicts in $\mathrm{LTL}_{AN(\varphi)+O(1)}$
- **Conjecture:** For every φ, distributed monitoring of φ requires $AN(\varphi)$ different values.
- **Next step:** Getting rid of the synchronization barrier between $\alpha' = s_0 s_1 \ldots s_{t-1}$ and $\alpha = \alpha' s_t = s_0 s_1 \ldots s_{t-1} s_t$

Grazie