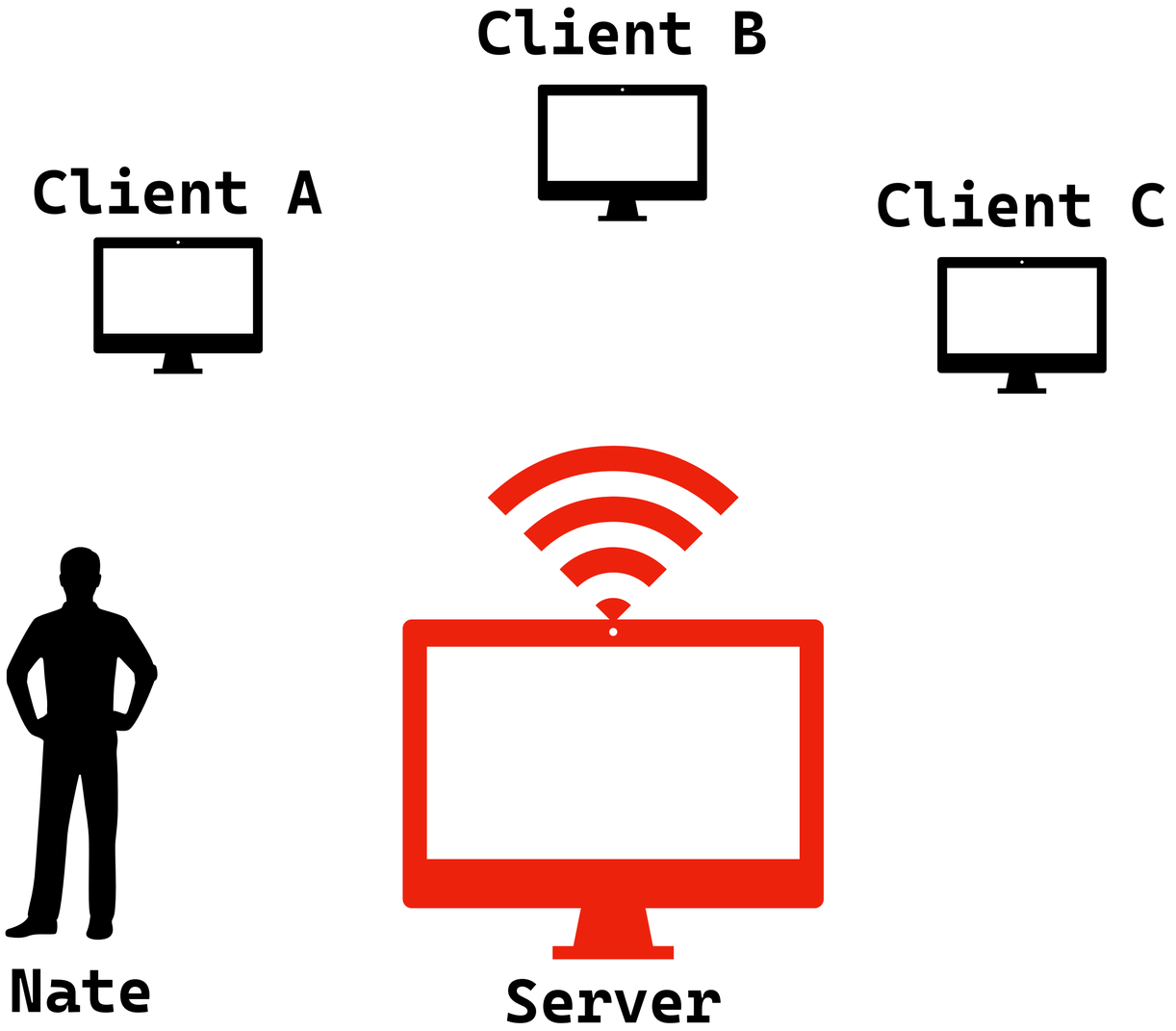
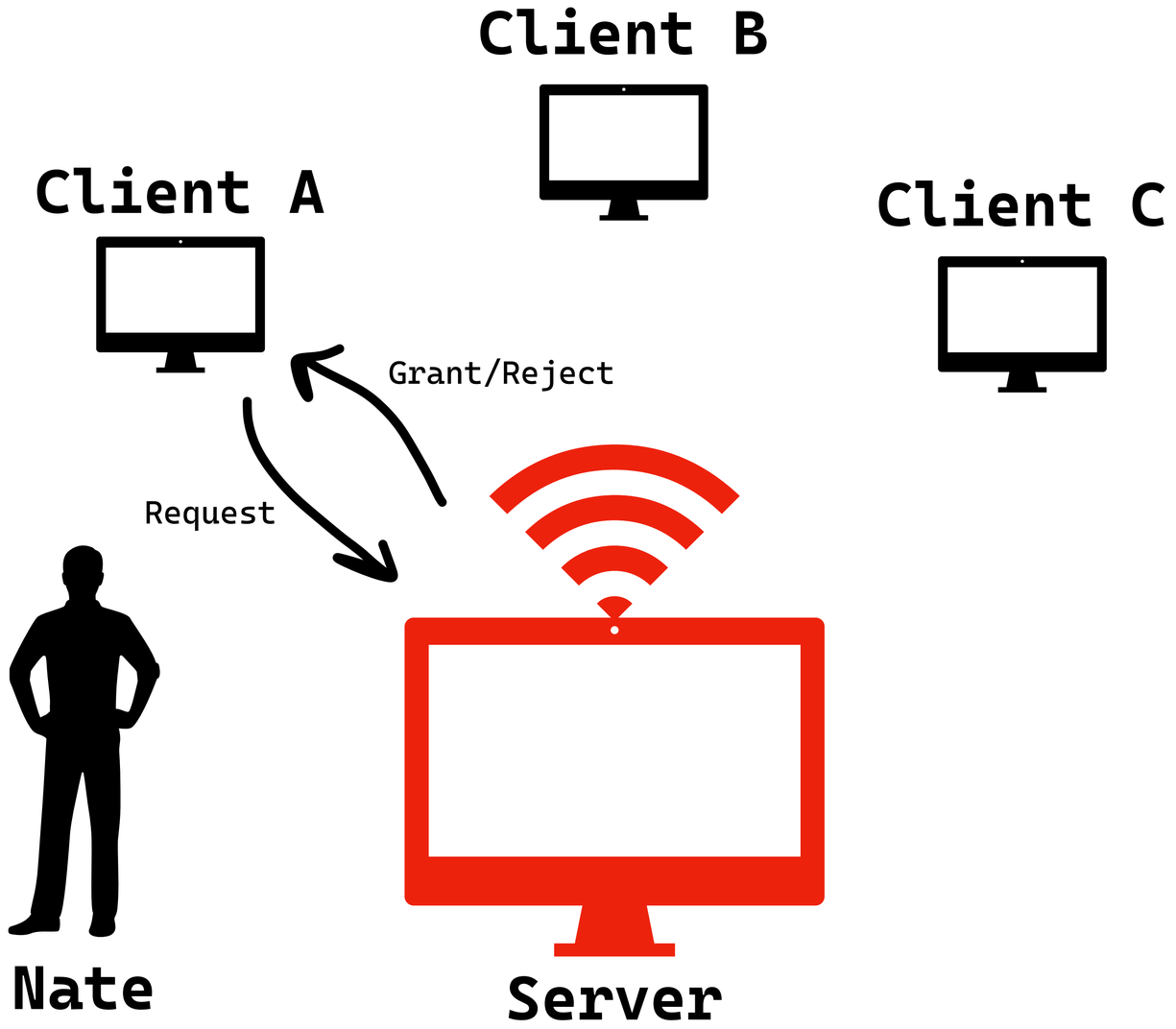
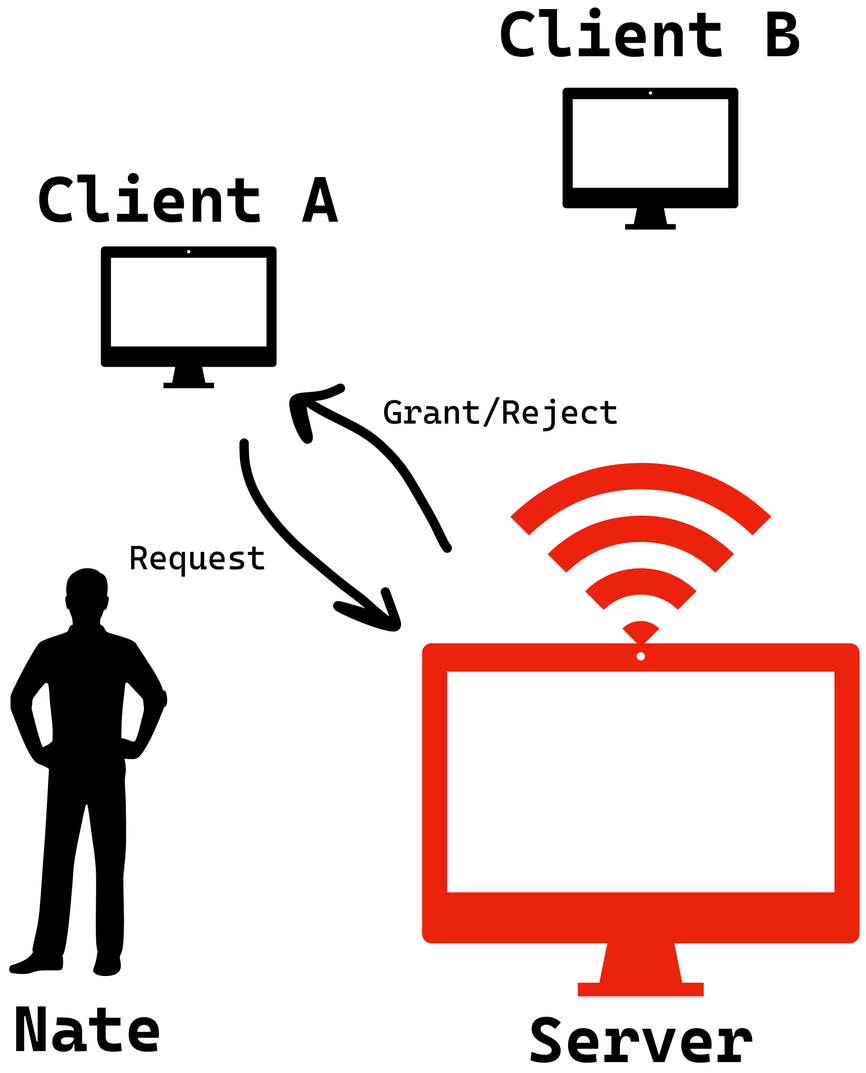
Scalable MLTL Runtime Monitoring and Satisfiability via Bit-vector Encoding

Christopher Johannsen, Phillip H. Jones,
Kristin Y. Rozier, Tichakorn Wongpirmasorn

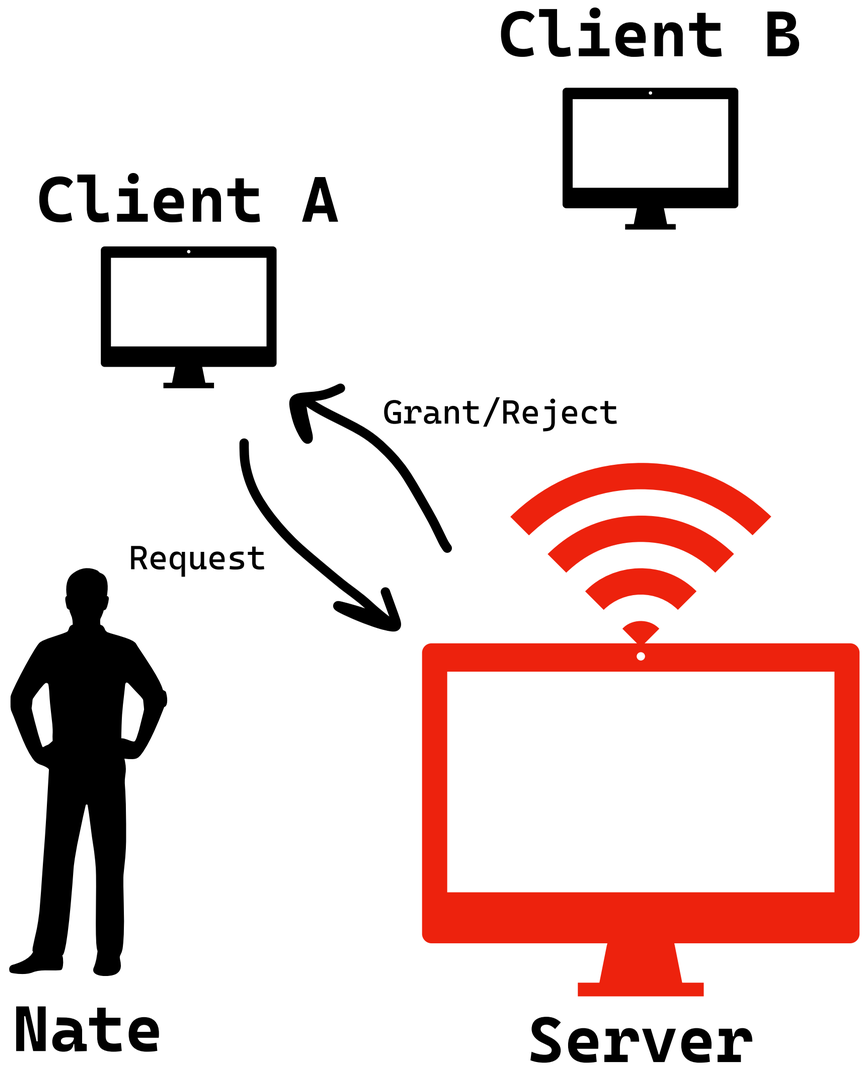
IOWA STATE UNIVERSITY
OF SCIENCE AND TECHNOLOGY


Client B
Client A
Client C
Nate
Server


Client B
Client A
Client C
Grant/Reject
Request
Nate
Server

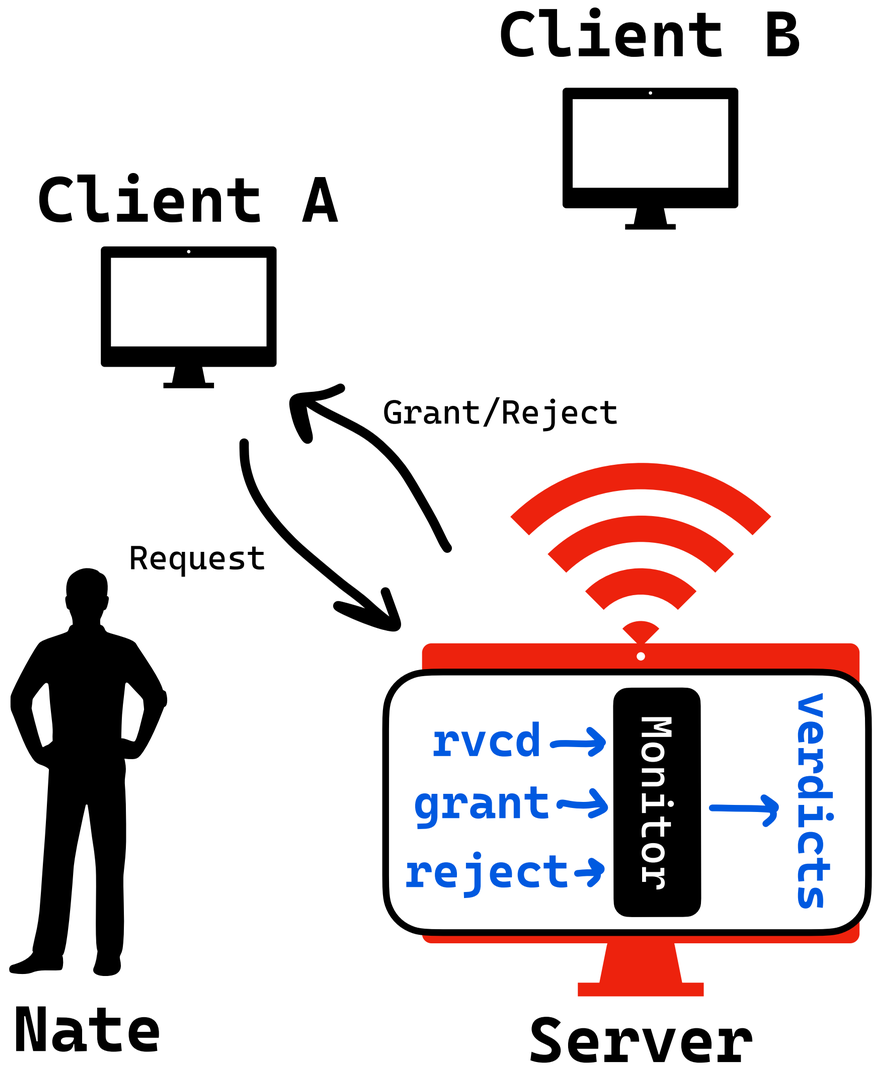


If a request is received by the Server, the request shall be granted or rejected within 225ms.



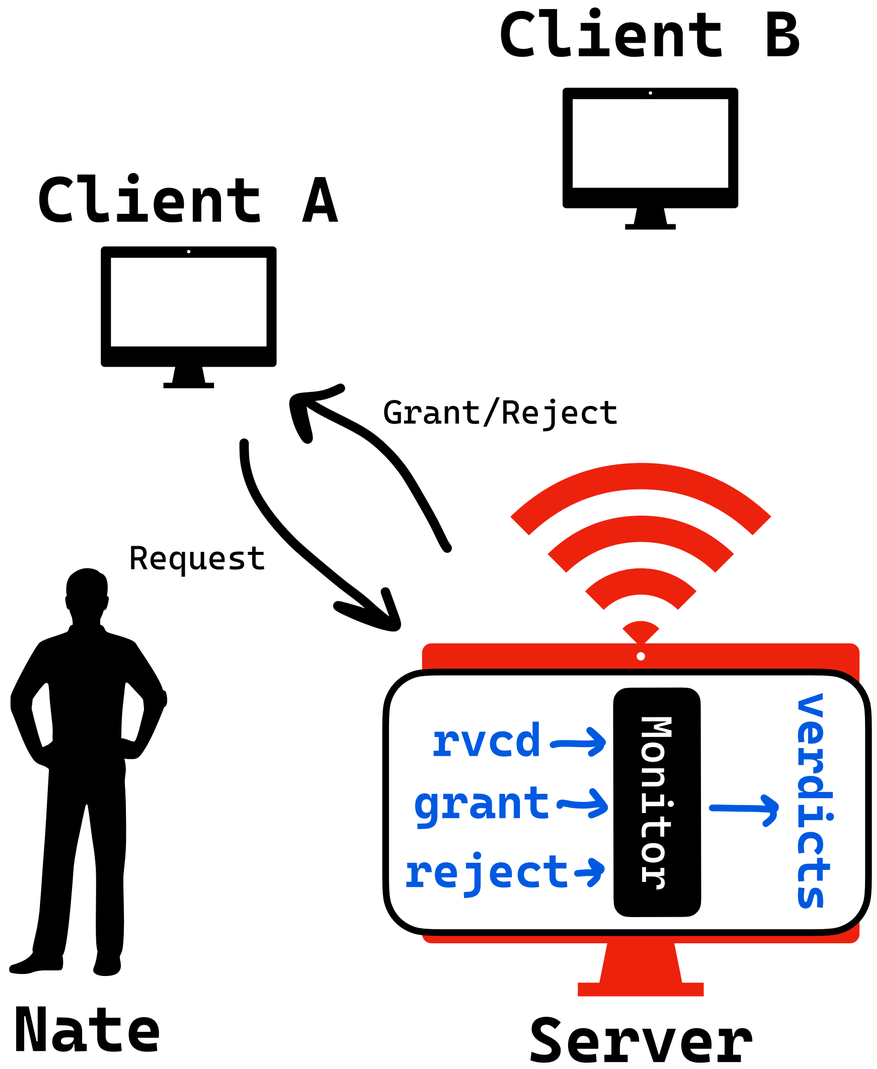
If a request is received by the Server, the request shall be granted or rejected within 225ms.

$$\text{rcvd} \rightarrow \Diamond_{[0,225]}(\text{grant} \vee \text{reject})$$



If a request is received by the Server, the request shall be granted or rejected within 225ms.

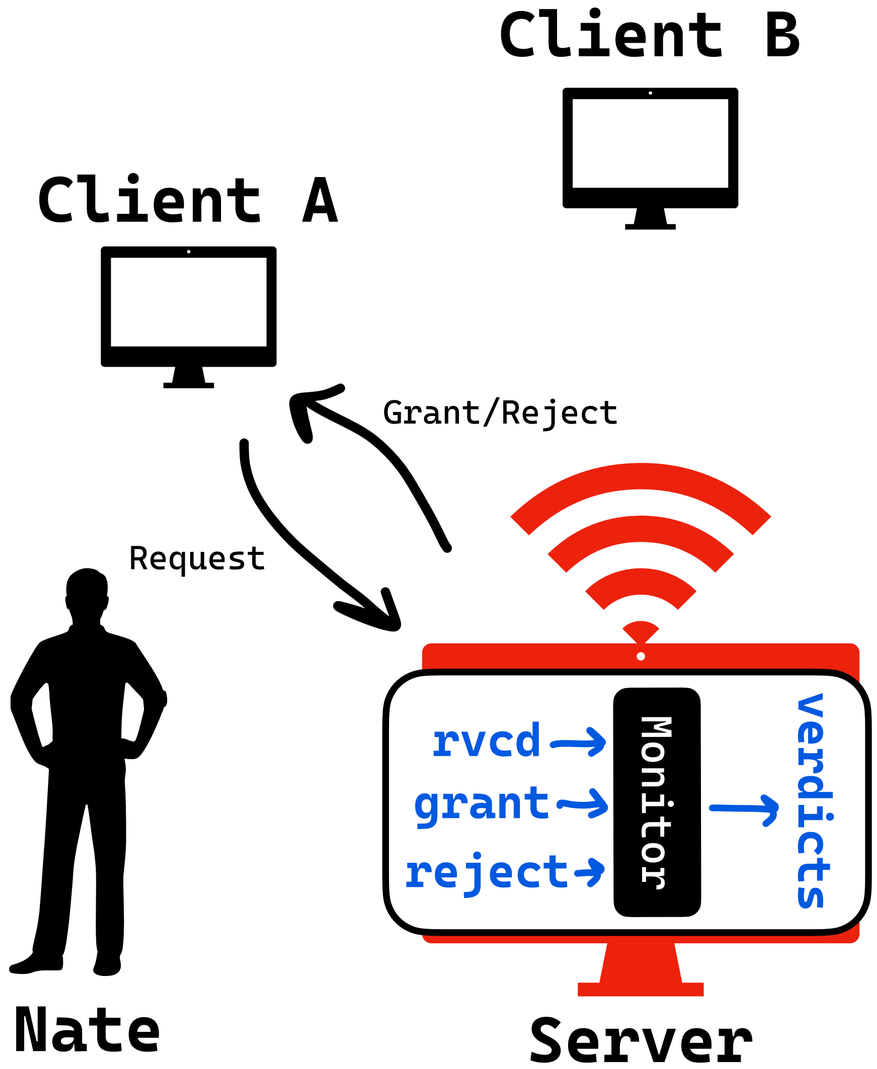
$$\text{rcvd} \rightarrow \Diamond_{[0,225]}(\text{grant} \vee \text{reject})$$



If a request is received by the Server, the request shall be granted or rejected within 225ms.

$$\text{rcvd} \rightarrow \Diamond_{[0,225]}(\text{grant} \vee \text{reject})$$

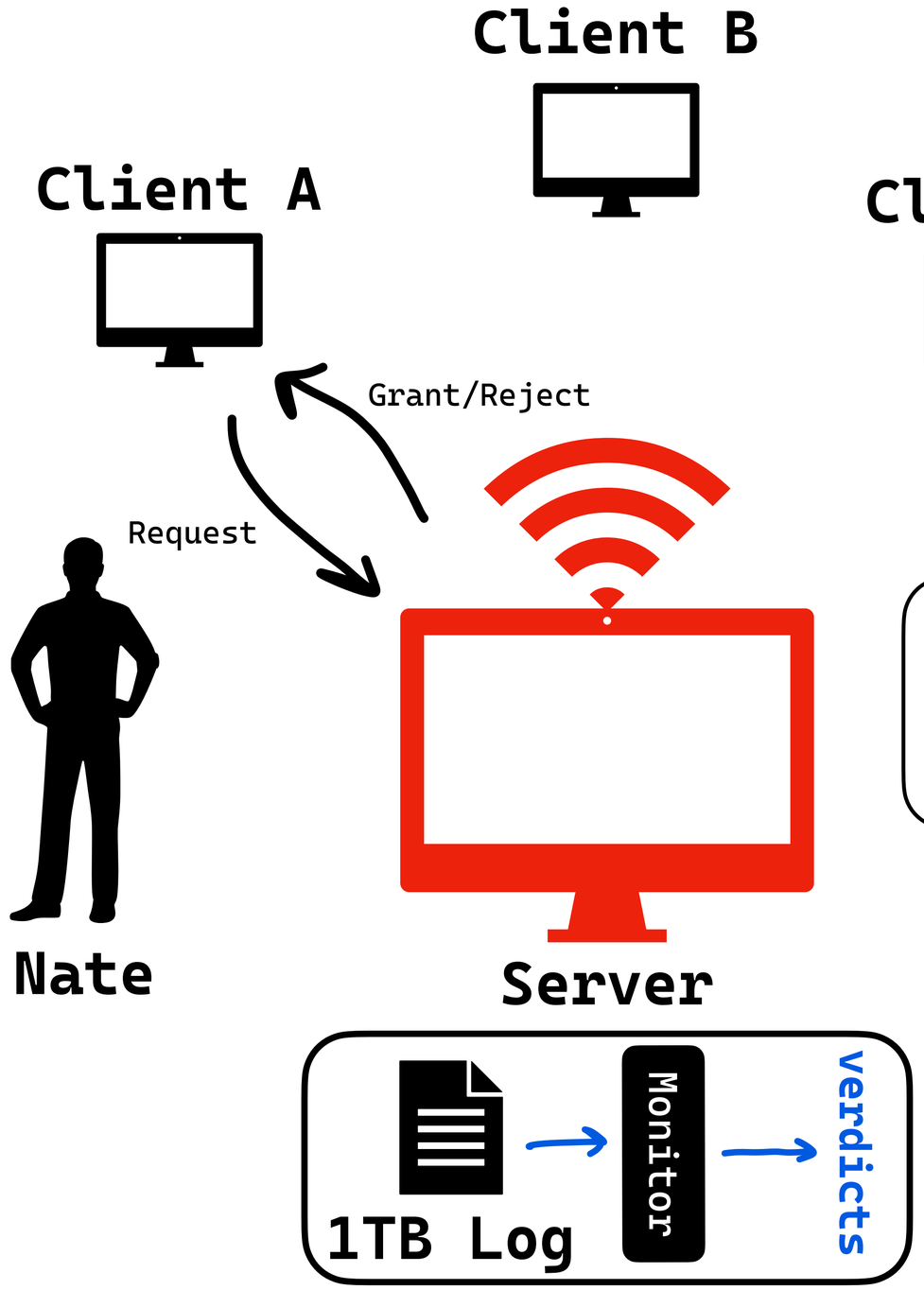
Required Throughput: 1/ms (1,000/s)



If a request is received by the Server, the request shall be granted or rejected within 225us.

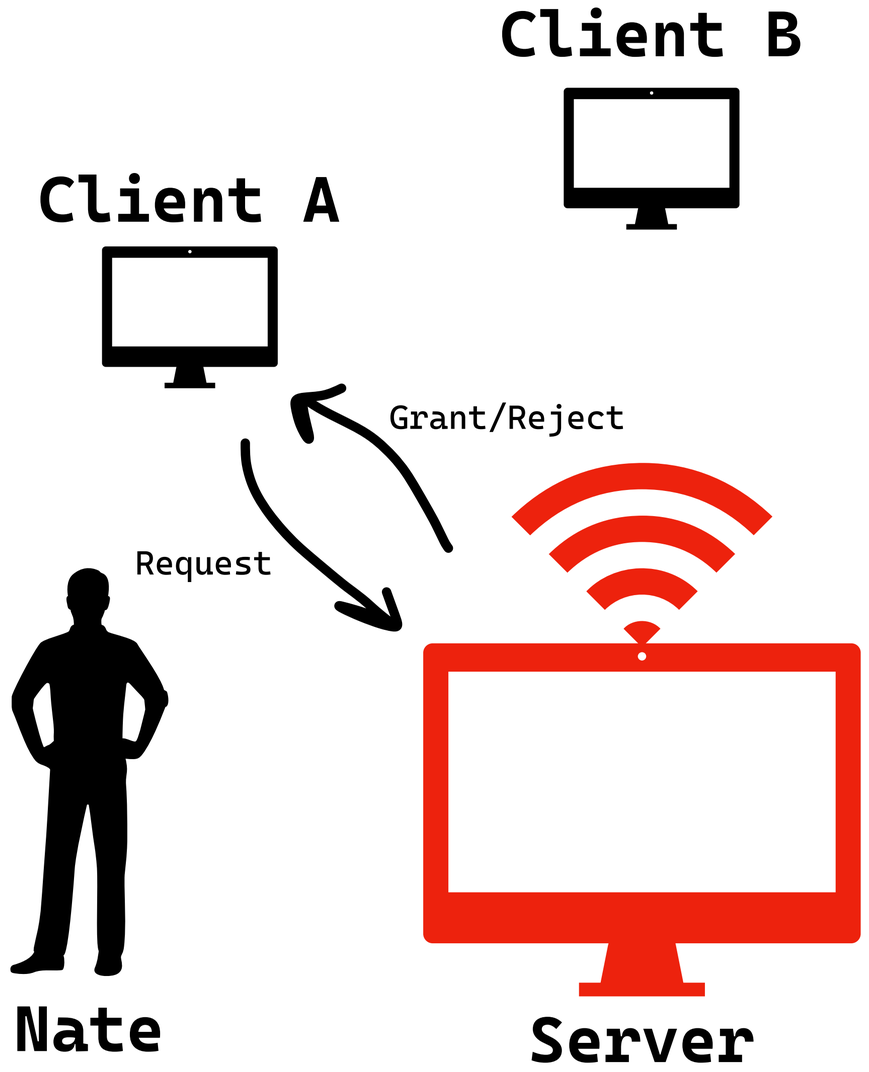
$$\text{rcvd} \rightarrow \Diamond_{[0,225]}(\text{grant} \vee \text{reject})$$

Required Throughput: 1/us (1,000,000/s)



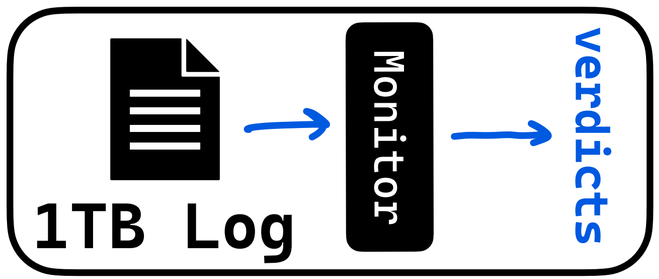
If a request is received by the Server, the request shall be granted or rejected within 225ms.

$$\text{rcvd} \rightarrow \Diamond_{[0,225]}(\text{grant} \vee \text{reject})$$



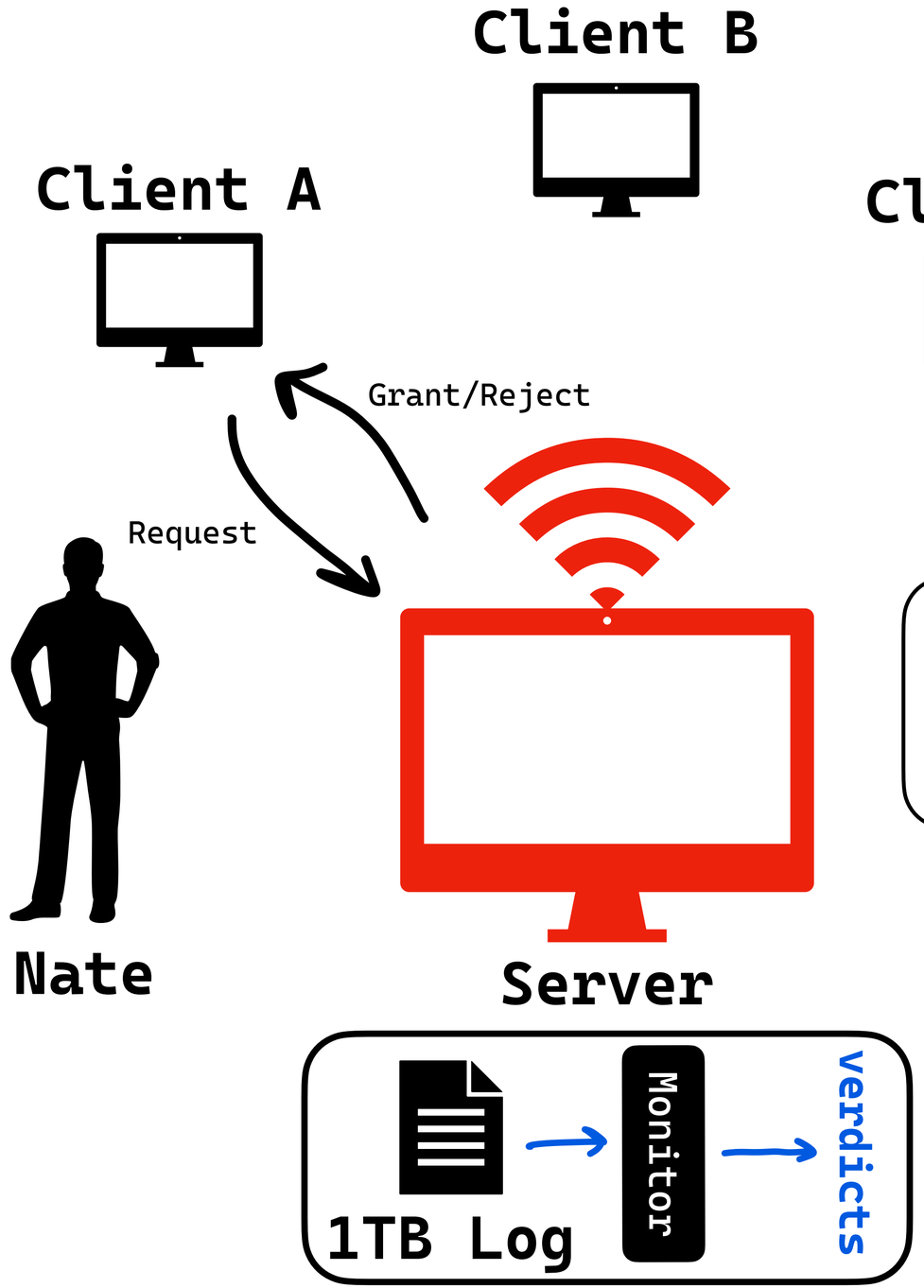
If a request is received by the Server, the request shall be granted or rejected within 225ms.

$$\text{rcvd} \rightarrow \Diamond_{[0,225]}(\text{grant} \vee \text{reject})$$



1TB / (1,000,000/s) ~= 277 hours

1TB / (100,000,000/s) ~= 2.77 hours



If a request is received by the Server, the request shall be granted or rejected within 225ms.

$$\text{rcvd} \rightarrow \Diamond_{[0,225]}(\text{grant} \vee \text{reject})$$

Need a monitoring algorithm that optimizes for **throughput**.

Mission-time Linear Temporal Logic (MLTL)

- LTL over finite traces with bounded temporal operators
- Set of atomic propositions $p, q \in \mathcal{AP}$
- Logical operators: $\neg$, $\wedge$, $\vee$, $\rightarrow$

Mission-time Linear Temporal Logic (MLTL)

- LTL over finite traces with bounded temporal operators
- Set of atomic propositions $p, q \in \mathcal{AP}$
- Logical operators: $\neg$, $\wedge$, $\vee$, $\rightarrow$

		0	1	2	3	4	5	6
Eventually	$\Diamond_{[1,3]}\, p$				p			
Globally	$\Box_{[2,5]}\, q$			q	q	q	q	
Until	$p\, \mathcal{U}_{[1,4]}\, q$		p	p	p	q		
Release	$p\, \mathcal{R}_{[2,5]}\, q$			q	q	q	$p\ q$	

- **MLTL monitoring problem**: Given an MLTL formula φ and trace π, compute the value of $\pi, \tau \models \varphi$ for each τ such that $0 \leq \tau < |\pi|$.

- **MLTL monitoring problem**: Given an MLTL formula φ and trace π, compute the value of $\pi, \tau \models \varphi$ for each τ such that $0 \leq \tau < |\pi|$.

τ	0	1	2	3	4	5	6	7	8
π			p					p	p
$\pi, \tau \models \Diamond_{[0,2]} p$									

- **MLTL monitoring problem**: Given an MLTL formula φ and trace π, compute the value of $\pi, \tau \models \varphi$ for each τ such that $0 \leq \tau < |\pi|$.

τ	0	1	2	3	4	5	6	7	8
π			p					p	p
$\pi, \tau \models \Diamond_{[0,2]} p$	T								

- **MLTL monitoring problem**: Given an MLTL formula φ and trace π, compute the value of $\pi, \tau \models \varphi$ for each τ such that $0 \leq \tau < |\pi|$.

τ	0	1	2	3	4	5	6	7	8
π			p					p	p
$\pi, \tau \models \Diamond_{[0,2]} p$	T	T							

- **MLTL monitoring problem**: Given an MLTL formula φ and trace π, compute the value of $\pi, \tau \models \varphi$ for each τ such that $0 \leq \tau < |\pi|$.

τ	0	1	2	3	4	5	6	7	8
π			p					p	p
$\pi, \tau \models \Diamond_{[0,2]} p$	T	T	T						

- **MLTL monitoring problem**: Given an MLTL formula φ and trace π, compute the value of $\pi, \tau \models \varphi$ for each τ such that $0 \leq \tau < |\pi|$.

τ	0	1	2	3	4	5	6	7	8
π			p					p	p
$\pi, \tau \models \Diamond_{[0,2]} p$	T	T	T	F					

- **MLTL monitoring problem**: Given an MLTL formula φ and trace π, compute the value of $\pi, \tau \models \varphi$ for each τ such that $0 \leq \tau < |\pi|$.

τ	0	1	2	3	4	5	6	7	8
π			p					p	p
$\pi, \tau \models \Diamond_{[0,2]} p$	T	T	T	F	F	T	T	T	T

- **MLTL monitoring problem**: Given an MLTL formula φ and trace π, compute the value of $\pi, \tau \models \varphi$ for each τ such that $0 \leq \tau < |\pi|$.

τ	0	1	2	3	4	5	6	7	8
π			p					p	p
$\pi, \tau \models \Diamond_{[0,2]} p$	T	T	T	F	F	T	T	T	T

- **Require**: MLTL monitoring algorithm that maximizes throughput.
  - **Fast CPU Operations**: Bitwise ($\sim$, $\&$, $|$, $\hat{}$, $\gg$, $\ll$)
  - **Slow* CPU Operations**: Arithmetic ($\times$, $\div$), Branching

- **Function** $bv(\varphi, \pi)$**:**
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

- **Function** $bv(\varphi, \pi)$**:**
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$bv(p, \pi) = b \text{ where } b[i] = (p \in \pi[i])$$

$$bv(\neg\varphi, \pi) = \sim bv(\varphi, \pi)$$

$$bv(\varphi \wedge \psi, \pi) = bv(\varphi, \pi) \,\&\, bv(\psi, \pi)$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

τ	0	1	2	3	4	5	6	7	8
π			$p\,q$	q				$p\,q$	p

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

τ	0	1	2	3	4	5	6	7	8
π			$p\,q$	q				$p\,q$	p

$$\tau = \quad 012345678$$

$$bv(p,\pi) = 0b001000011$$

$$bv(q,\pi) = 0b001100010$$

$$bv(\neg p, \pi) = \sim bv(p,\pi) =$$

$$bv(p \wedge q\ ,\ \pi) = bv(p,\pi)\ \&\ bv(q,\pi) =$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

τ	0	1	2	3	4	5	6	7	8
π			$p\,q$	q				$p\,q$	p

$$\tau = \quad 012345678$$

$$bv(p,\pi) = 0b001000011$$

$$bv(q,\pi) =$$

$$bv(\neg p, \pi) = \sim bv(p,\pi) = 0b110111100$$

$$bv(p \wedge q\ ,\ \pi) = bv(p,\pi)\ \&\ bv(q,\pi) =$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

τ	0	1	2	3	4	5	6	7	8
π			$p\,q$	q				$p\,q$	p

$$\tau = \quad 012345678$$

$$bv(p,\pi) = 0b001000011$$

$$bv(q,\pi) = 0b001100010$$

$$bv(\neg p, \pi) = \sim bv(p,\pi) =$$

$$bv(p \wedge q\ ,\ \pi) = bv(p,\pi)\ \&\ bv(q,\pi) = 0b001000010$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

τ	0	1	2	3	4	5	6	7	8
π			$p\,q$	q				$p\,q$	p

$$\tau = \quad 012345678$$

$$bv(p,\pi) = 0b001000011$$

$$bv(q,\pi) = 0b001100010$$

$$bv(\neg p, \pi) = \sim bv(p,\pi) =$$

$$bv(p \wedge q\ ,\ \pi) = bv(p,\pi)\ \&\ bv(q,\pi) = 0b001000010$$

$\pi, 7 \models p \wedge q$

- **Function** $bv(\varphi, \pi)$**:**
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$
$$b_1 = b_0 \mid (b_0 \ll 1)$$
$$b_2 = b_1 \mid (b_1 \ll 2)$$
$$b_3 = b_2 \mid (b_2 \ll 4)$$
$$\dots$$
$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_k \mid (b_k \ll 2^k)$$

- **Function** $bv(\varphi, \pi)$**:**
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \vDash \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$
$$b_1 = b_0 \mid (b_0 \ll 1)$$
$$b_2 = b_1 \mid (b_1 \ll 2)$$
$$b_3 = b_2 \mid (b_2 \ll 4)$$
$$\dots$$
$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_k \mid (b_k \ll 2^k)$$

- **Function** $bv(\varphi, \pi)$**:**
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \vDash \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$
$$b_1 = b_0 \mid (b_0 \ll 1)$$
$$b_2 = b_1 \mid (b_1 \ll 2)$$
$$b_3 = b_2 \mid (b_2 \ll 4)$$
$$\dots$$
$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_k \mid (b_k \ll 2^k)$$

Largest $k \in \mathbb{N}$ such that $2^k < (b - a + 1)$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$
$$b_1 = b_0 \mid (b_0 \ll 1)$$
$$b_2 = b_1 \mid (b_1 \ll 2)$$
$$b_3 = b_2 \mid (b_2 \ll 4)$$
$$\dots$$
$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_k \mid (b_k \ll 2^k)$$

Largest $k \in \mathbb{N}$ such that $2^k < (b - a + 1)$

Size must be power of 2

$(b - a + 1 = 2^k)$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$
$$b_1 = b_0 \mid (b_0 \ll 1)$$
$$b_2 = b_1 \mid (b_1 \ll 2)$$
$$b_3 = b_2 \mid (b_2 \ll 4)$$
$$\dots$$
$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_k \mid (b_k \ll 2^k)$$

Largest $k \in \mathbb{N}$ such that $2^k < (b - a + 1)$

Size must be power of 2

$(b - a + 1 = 2^k)$

$$\Diamond_{[a,b]}\varphi \equiv \Diamond_{[c,d]}\Diamond_{[e,f]}\varphi$$

Where $a = c + e$ and $b = d + f$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$
$$b_1 = b_0 \mid (b_0 \ll 1)$$
$$b_2 = b_1 \mid (b_1 \ll 2)$$
$$b_3 = b_2 \mid (b_2 \ll 4)$$
$$\dots$$
$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_k \mid (b_k \ll 2^k)$$

Largest $k \in \mathbb{N}$ such that $2^k < (b - a + 1)$

Size must be power of 2

$(b - a + 1 = 2^k)$

$$\Diamond_{[a,b]}\varphi \equiv \Diamond_{[c,d]}\Diamond_{[e,f]}\varphi$$

Where $a = c + e$ and $b = d + f$

$$\Diamond_{[0,9]}\varphi \equiv \Diamond_{[0,7]}\Diamond_{[0,2]}\varphi$$
$$\equiv \Diamond_{[0,7]}\Diamond_{[0,1]}\Diamond_{[0,1]}\varphi$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$
$$b_1 = b_0 \mid (b_0 \ll 1)$$
$$b_2 = b_1 \mid (b_1 \ll 2)$$
$$b_3 = b_2 \mid (b_2 \ll 4)$$
$$\dots$$
$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_k \mid (b_k \ll 2^k)$$

$\log(k)$

Size must be power of 2

$(b - a + 1 = 2^k)$

$\Diamond_{[a,b]}\varphi \equiv \Diamond_{[c,d]}\Diamond_{[e,f]}\varphi$

Where $a = c + e$ and $b = d + f$

$\log(k)$

$\Diamond_{[0,9]}\varphi \equiv \Diamond_{[0,7]}\Diamond_{[0,2]}\varphi$

$\equiv \Diamond_{[0,7]}\Diamond_{[0,1]}\Diamond_{[0,1]}\varphi$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

τ	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
π								p												p

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

τ	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
π								p												p

$$\begin{aligned}
\tau = &\quad 0123456789\ 1\ 0123456789 \\
bv(p,\pi) = b_0 = &\quad \texttt{0b0000000100}\ \ \texttt{0000000001} \\
b_1 = b_0 \mid (b_0 \ll 1) = &\quad \texttt{0b0000001100}\ \ \texttt{0000000011} \\
b_2 = b_1 \mid (b_1 \ll 2) = &\quad \texttt{0b0000111100}\ \ \texttt{0000001111} \\
bv(\Diamond_{[0,7]} p, \pi) = b_3 = b_2 \mid (b_2 \ll 4) = &\quad \texttt{0b1111111100}\ \ \texttt{0011111111}
\end{aligned}$$

- **Function** $bv(\varphi, \pi)$**:**
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \vDash \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$

$$b_1 = b_0 \mid (b_0 \ll 1)$$

$$b_2 = b_1 \mid (b_1 \ll 2)$$

$$b_3 = b_2 \mid (b_2 \ll 4)$$

$$\dots$$

$$b_{k+1} = b_k \mid (b_k \ll 2^k)$$

$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_{k+1} \mid (b_{k+1} \ll leftover)$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$b_0 = bv(\varphi, \pi) \ll a$$
$$b_1 = b_0 \mid (b_0 \ll 1)$$
$$b_2 = b_1 \mid (b_1 \ll 2)$$
$$b_3 = b_2 \mid (b_2 \ll 4)$$
$$\dots$$
$$b_{k+1} = b_k \mid (b_k \ll 2^k)$$
$$bv(\Diamond_{[a,b]}\varphi, \pi) = b_{k+1} \mid (b_{k+1} \ll leftover)$$

$$(b - a + 1) - 2^k$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$
\begin{aligned}
b_{\psi_0} &= bv(\psi, \pi) \ll a & b_{\varphi_0} &= bv(\varphi, \pi) \ll a \\
b_{\psi_1} &= b_{\psi_0} \,|\, (b_{\varphi_0} \,\&\, (b_{\psi_0} \ll 1)) & b_{\varphi_1} &= b_{\varphi_0} \,\&\, (b_{\varphi_0} \ll 1) \\
b_{\psi_2} &= b_{\psi_1} \,|\, (b_{\varphi_1} \,\&\, (b_{\psi_1} \ll 2)) & b_{\varphi_2} &= b_{\varphi_1} \,\&\, (b_{\varphi_1} \ll 2) \\
b_{\psi_3} &= b_{\psi_2} \,|\, (b_{\varphi_2} \,\&\, (b_{\psi_2} \ll 4)) & b_{\varphi_3} &= b_{\varphi_2} \,\&\, (b_{\varphi_2} \ll 4) \\
&\dots & &\dots \\
b_{\psi_{k+1}} &= b_{\psi_k} \,|\, (b_{\varphi_k} \,\&\, (b_{\psi_k} \ll 2^k)) & b_{\varphi_{k+1}} &= b_{\varphi_k} \,\&\, (b_{\varphi_k} \ll 2^k) \\
bv(\varphi \mathcal{U}_{[a,b]} \psi, \pi) &= b_{\psi_{k+1}} \,|\, (b_{\varphi_{k+1}} \,\&\, (b_{\psi_{k+1}} \ll leftover))
\end{aligned}
$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \vDash \varphi$

$$
\begin{aligned}
b_{\psi_0} &= bv(\psi, \pi) \ll a & b_{\varphi_0} &= bv(\varphi, \pi) \ll a \\
b_{\psi_1} &= b_{\psi_0} \,|\, (b_{\varphi_0} \,\&\, (b_{\psi_0} \ll 1)) & b_{\varphi_1} &= b_{\varphi_0} \,\&\, (b_{\varphi_0} \ll 1) \\
b_{\psi_2} &= b_{\psi_1} \,|\, (b_{\varphi_1} \,\&\, (b_{\psi_1} \ll 2)) & b_{\varphi_2} &= b_{\varphi_1} \,\&\, (b_{\varphi_1} \ll 2) \\
b_{\psi_3} &= b_{\psi_2} \,|\, (b_{\varphi_2} \,\&\, (b_{\psi_2} \ll 4)) & b_{\varphi_3} &= b_{\varphi_2} \,\&\, (b_{\varphi_2} \ll 4) \\
& & \dots & \\
b_{\psi_{k+1}} &= b_{\psi_k} \,|\, (b_{\varphi_k} \,\&\, (b_{\psi_k} \ll 2^k)) & b_{\varphi_{k+1}} &= b_{\varphi_k} \,\&\, (b_{\varphi_k} \ll 2^k)
\end{aligned}
$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \vDash \varphi$

$$b_{\psi_0} = bv(\psi, \pi) \ll a \qquad b_{\varphi_0} = bv(\varphi, \pi) \ll a$$
$$b_{\psi_1} = b_{\psi_0} | (b_{\varphi_0} \,\&\, (b_{\psi_0} \ll 1)) \qquad b_{\varphi_1} = b_{\varphi_0} \,\&\, (b_{\varphi_0} \ll 1)$$
$$b_{\psi_2} = b_{\psi_1} | (b_{\varphi_1} \,\&\, (b_{\psi_1} \ll 2)) \qquad b_{\varphi_2} = b_{\varphi_1} \,\&\, (b_{\varphi_1} \ll 2)$$
$$b_{\psi_3} = b_{\psi_2} | (b_{\varphi_2} \,\&\, (b_{\psi_2} \ll 4)) \qquad b_{\varphi_3} = b_{\varphi_2} \,\&\, (b_{\varphi_2} \ll 4)$$
$$\ldots$$
$$b_{\psi_{k+1}} = b_{\psi_k} | (b_{\varphi_k} \,\&\, (b_{\psi_k} \ll 2^k)) \qquad b_{\varphi_{k+1}} = b_{\varphi_k} \,\&\, (b_{\varphi_k} \ll 2^k)$$
$$bv(\varphi \mathcal{U}_{[a,b]} \psi, \pi) = b_{\psi_{k+1}} | (b_{\varphi_{k+1}} \,\&\, (b_{\psi_{k+1}} \ll leftover))$$

- **Function** $bv(\varphi, \pi)$:
  - **Input: MLTL formula** φ**, trace** π
  - **Output: Bit-vector** b **of width** $|\pi|$**, where each bit** $b[i]$ **represents whether** $\pi, i \models \varphi$

$$
\begin{aligned}
b_{\psi_0} &= bv(\psi, \pi) \ll a & b_{\varphi_0} &= bv(\varphi, \pi) \ll a \\
b_{\psi_1} &= b_{\psi_0} \,|\, (b_{\varphi_0} \,\&\, (b_{\psi_0} \ll 1)) & b_{\varphi_1} &= b_{\varphi_0} \,\&\, (b_{\varphi_0} \ll 1) \\
b_{\psi_2} &= b_{\psi_1} \,|\, (b_{\varphi_1} \,\&\, (b_{\psi_1} \ll 2)) & b_{\varphi_2} &= b_{\varphi_1} \,\&\, (b_{\varphi_1} \ll 2) \\
b_{\psi_3} &= b_{\psi_2} \,|\, (b_{\varphi_2} \,\&\, (b_{\psi_2} \ll 4)) & b_{\varphi_3} &= b_{\varphi_2} \,\&\, (b_{\varphi_2} \ll 4) \\
&\dots & &\dots \\
b_{\psi_{k+1}} &= b_{\psi_k} \,|\, (b_{\varphi_k} \,\&\, (b_{\psi_k} \ll 2^k)) & b_{\varphi_{k+1}} &= b_{\varphi_k} \,\&\, (b_{\varphi_k} \ll 2^k) \\
bv(\varphi \mathcal{U}_{[a,b]} \psi, \pi) &= b_{\psi_{k+1}} \,|\, (b_{\varphi_{k+1}} \,\&\, (b_{\psi_{k+1}} \ll leftover))
\end{aligned}
$$

- **MLTL Satisfiability Problem**: Given a formula φ, determine if there exist a trace π and time τ such that $\pi, \tau \models \varphi$.

$$\exists \pi \,.\, bv(\varphi, \pi) > 0$$

- **MLTL Satisfiability Problem**: Given a formula φ, determine if there exist a trace π and time τ such that $\pi, \tau \models \varphi$.

$$\exists \pi \,.\, bv(\varphi, \pi) > 0$$

- For each proposition p_i in formula φ, declare uninterpreted bit-vector b_{p_i} of width $cplen(\varphi)$.
  - $cplen(\varphi) \simeq$ sum of upper bounds of temporal ops in φ.
  - $cplen(\Diamond_{[0,3]}\Diamond_{[0,2]}p) = 5 + 1 = 6$.
- Check if $bv\left(\varphi, \pi\right)$ is **SAT**, where $bv(p_i, \pi) = b_{p_i}$.

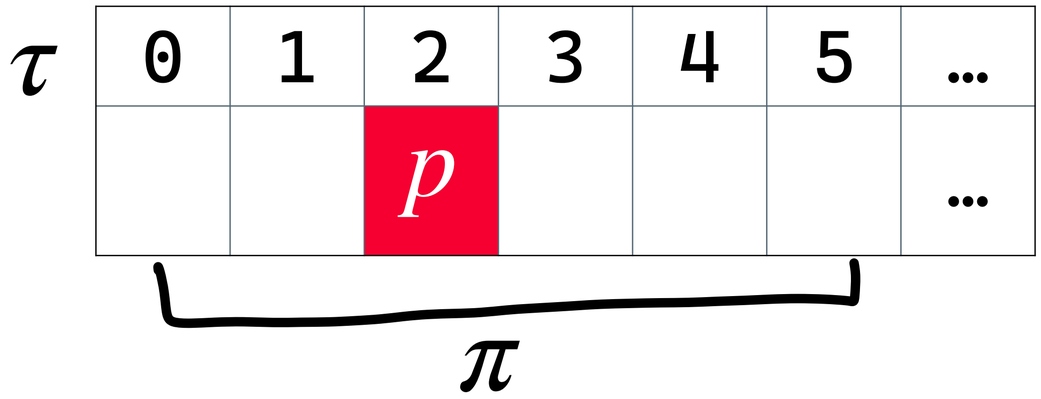
- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.

- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.

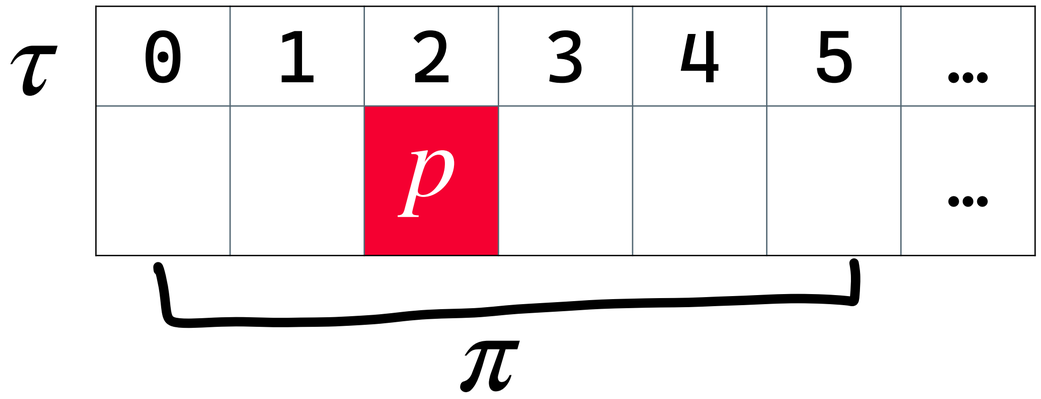
- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.

τ	0	1	2	3	4	5	…
			p				…

- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.

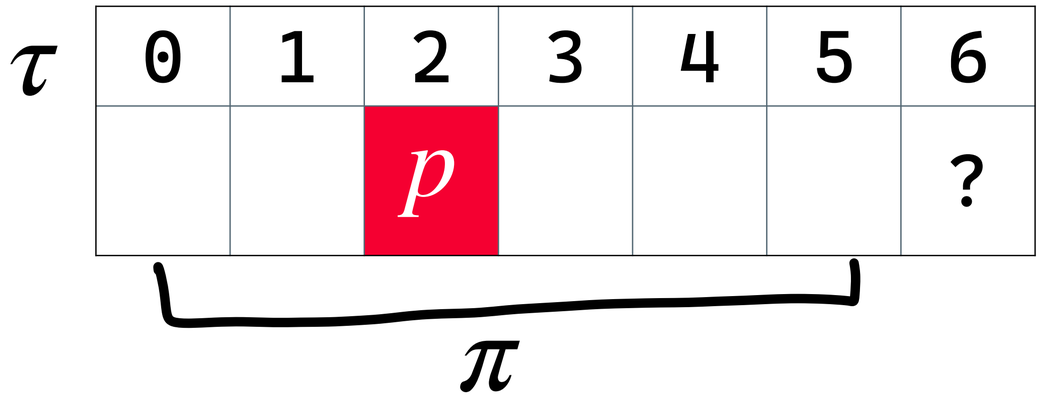


- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.



$$\tau = \quad 012345$$
$$bv(p, \pi) = 0b001000$$
$$bv(\Diamond_{[0,2]} p, \pi) = 0b111000$$

- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.

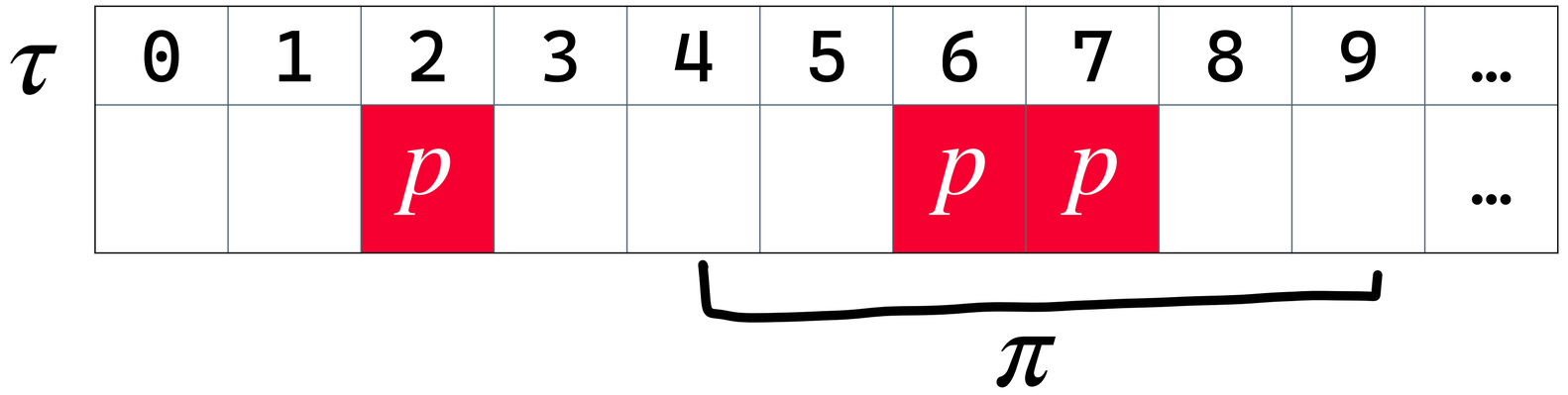


$$\tau = \quad 012345$$
$$bv(p, \pi) = 0b001000$$
$$bv(\Diamond_{[0,2]} p, \pi) = 0b111000$$

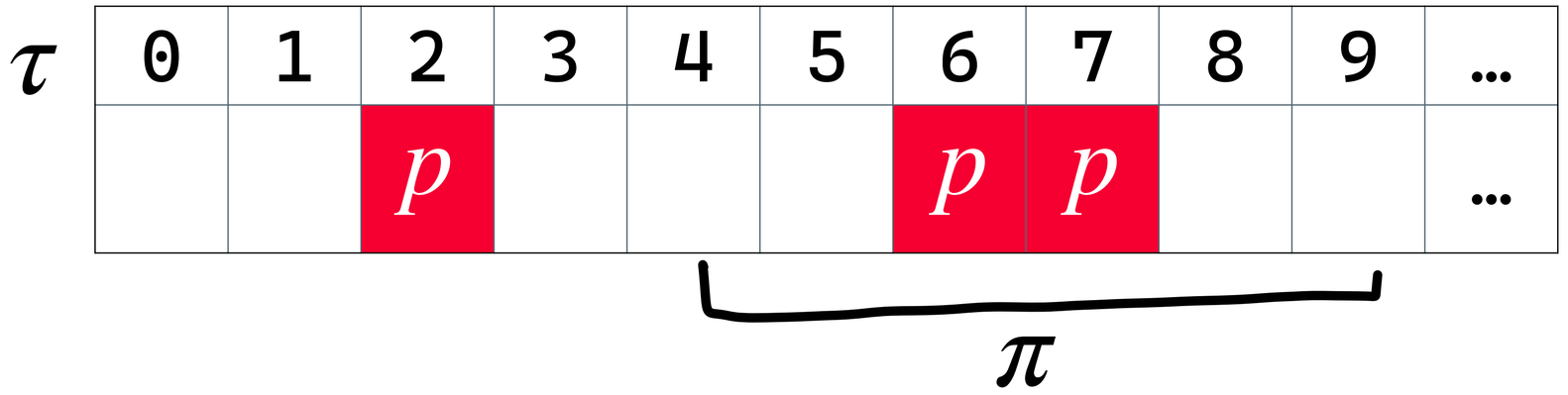
- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.

τ	0	1	2	3	4	5	6	7	8	9	…
			p				p	p			…

- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.

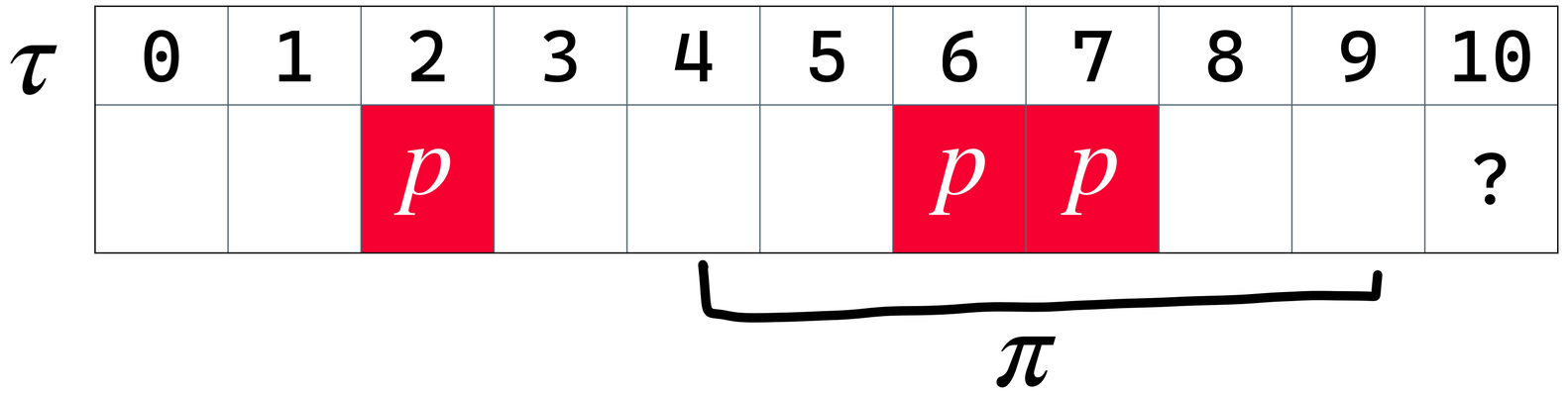


- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.



$$\tau = \quad 456789$$
$$bv(p, \pi) = 0b001100$$
$$bv(\Diamond_{[0,2]} p, \pi) = 0b111100$$

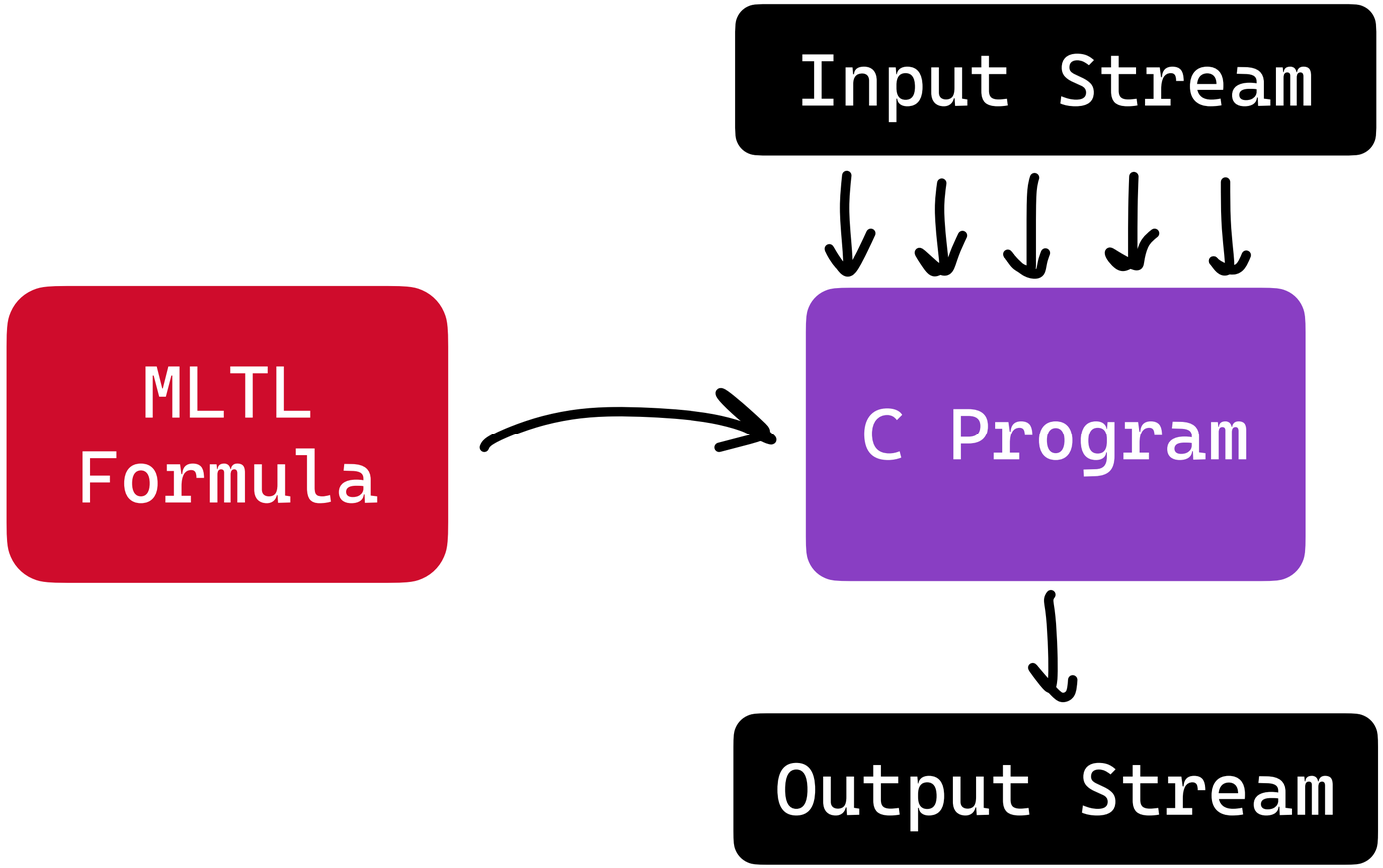
- **MLTL Online Monitoring**: Trace is not given up-front, each proposition is a never-ending stream of inputs.
  - Buffer values of p, treat this buffer as π, evaluate $bv(\varphi, \pi)$.
  - Shift in new values of p, re-evaluate $bv(\varphi, \pi)$, repeat.
- Let $\varphi = \Diamond_{[0,2]} p$, assume we evaluate $n = 4$ verdicts at a time.



$$\tau = \quad 4567\,89$$
$$bv(p, \pi) = 0b00\,11\,00$$
$$bv(\Diamond_{[0,2]} p, \pi) = 0b11\,11\,00$$

Implementation

Stream Analyzer via Bitwise Reasoning

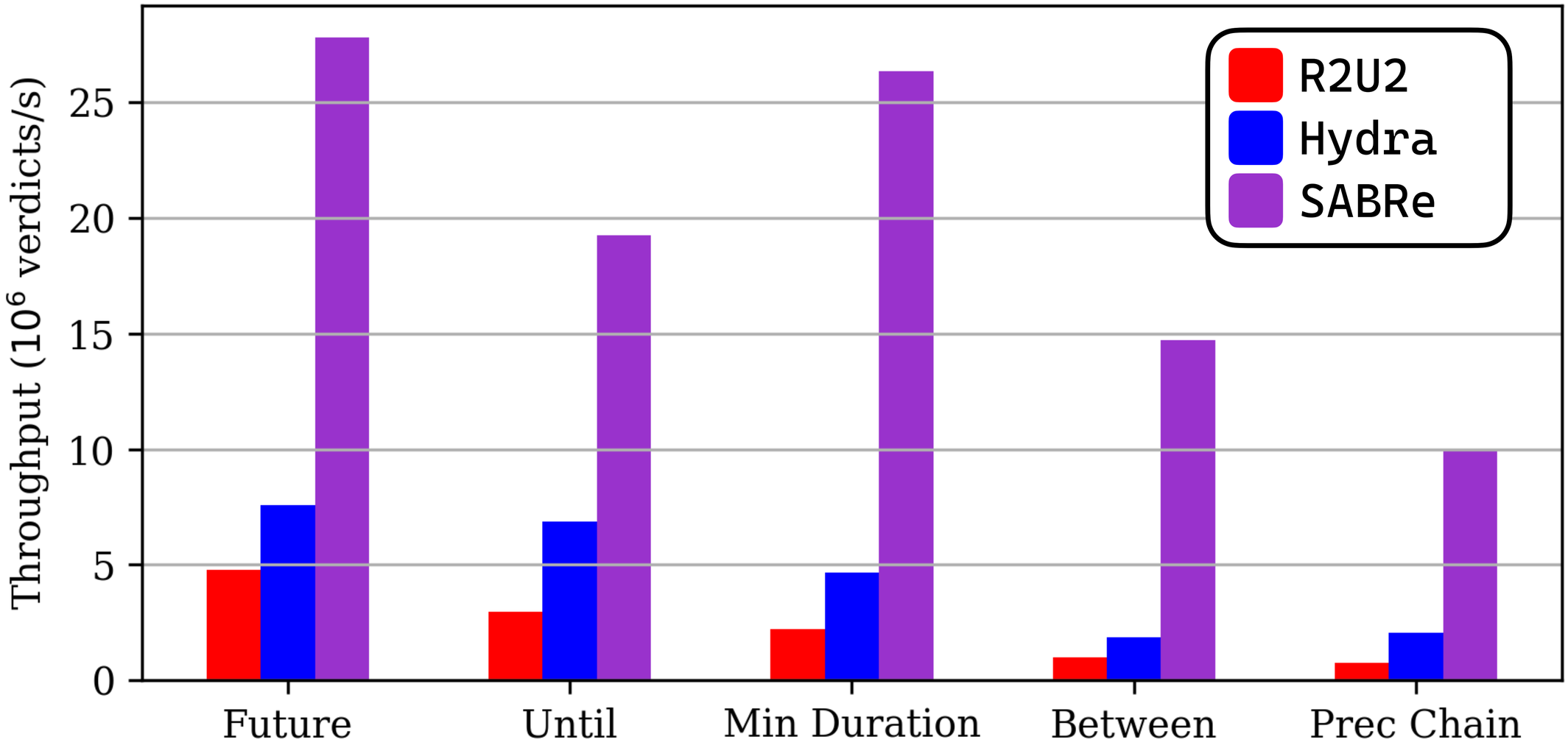


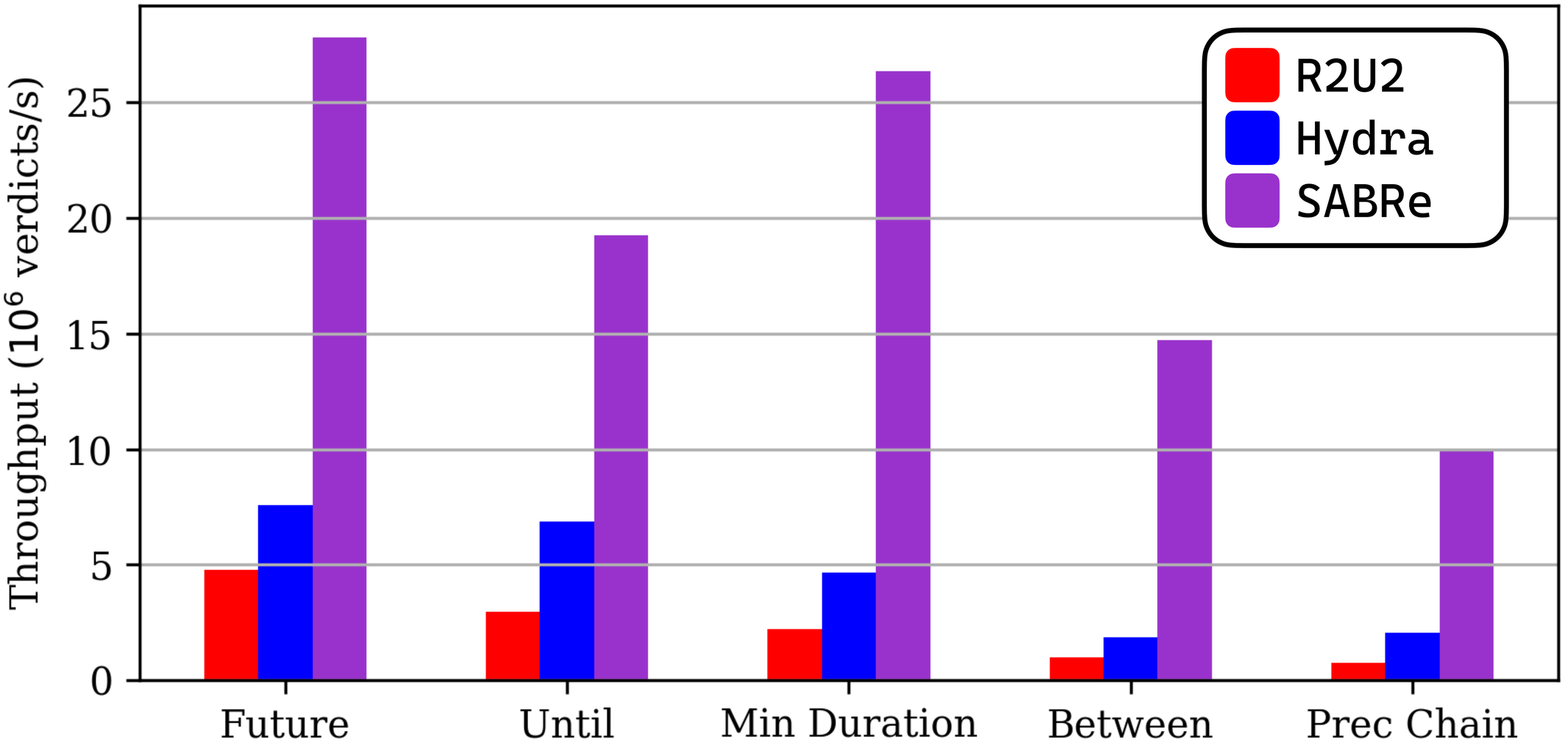
Experimental Setup

- Took 5 pattern-instantiated MLTL formulas with interval sizes from 10 to 30.
- Generated random traces of length 1,000,000.
- Ran each tool over each formula/trace pair 10 times and took the median throughput (**total time/verdicts computed**)
- Competition:
  - R2U2: Online, real-time MLTL Monitor
  - Hydra: Offline MTL Monitor

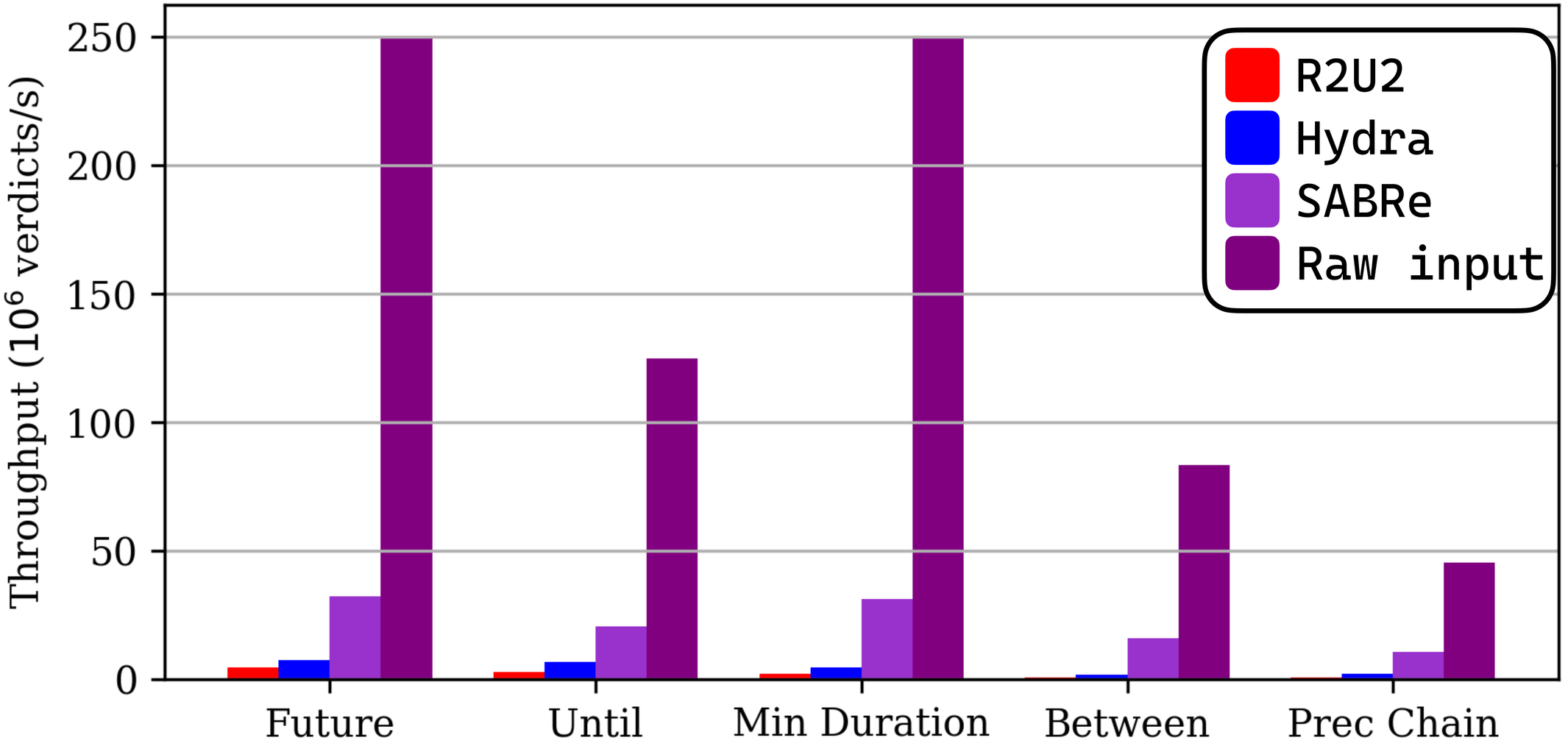
Experimental Setup

Name	Formula
Future	$\Diamond_{[0,10]} p$
Until	$p \mathcal{U}_{[0,10]} q$
Min Duration	$p \vee ((\neg p \mathcal{U}_{[0,10]} p) \vee \Box_{[0,10]} \neg p)$
Between	$(q \wedge \Box_{[0,10]} \neg r \wedge \Diamond_{[10,30]} r) \rightarrow ((\neg r \mathcal{U}_{[10,20]} (p \wedge \neg r)) \vee \Box_{[10,20]} \neg p)$
Precedence Chain	$\Diamond_{[10,10]} (q_0 \wedge \Diamond_{[0,5]} (q_1 \wedge \Diamond_{[0,5]} q_2 \wedge (\Diamond_{[0,5]} q_3))) \rightarrow \Diamond_{[0,10]} p$


R2U2
Hydra
SABRe
Throughput (10^6 verdicts/s)
0
5
10
15
20
25
Future
Until
Min Duration
Between
Prec Chain

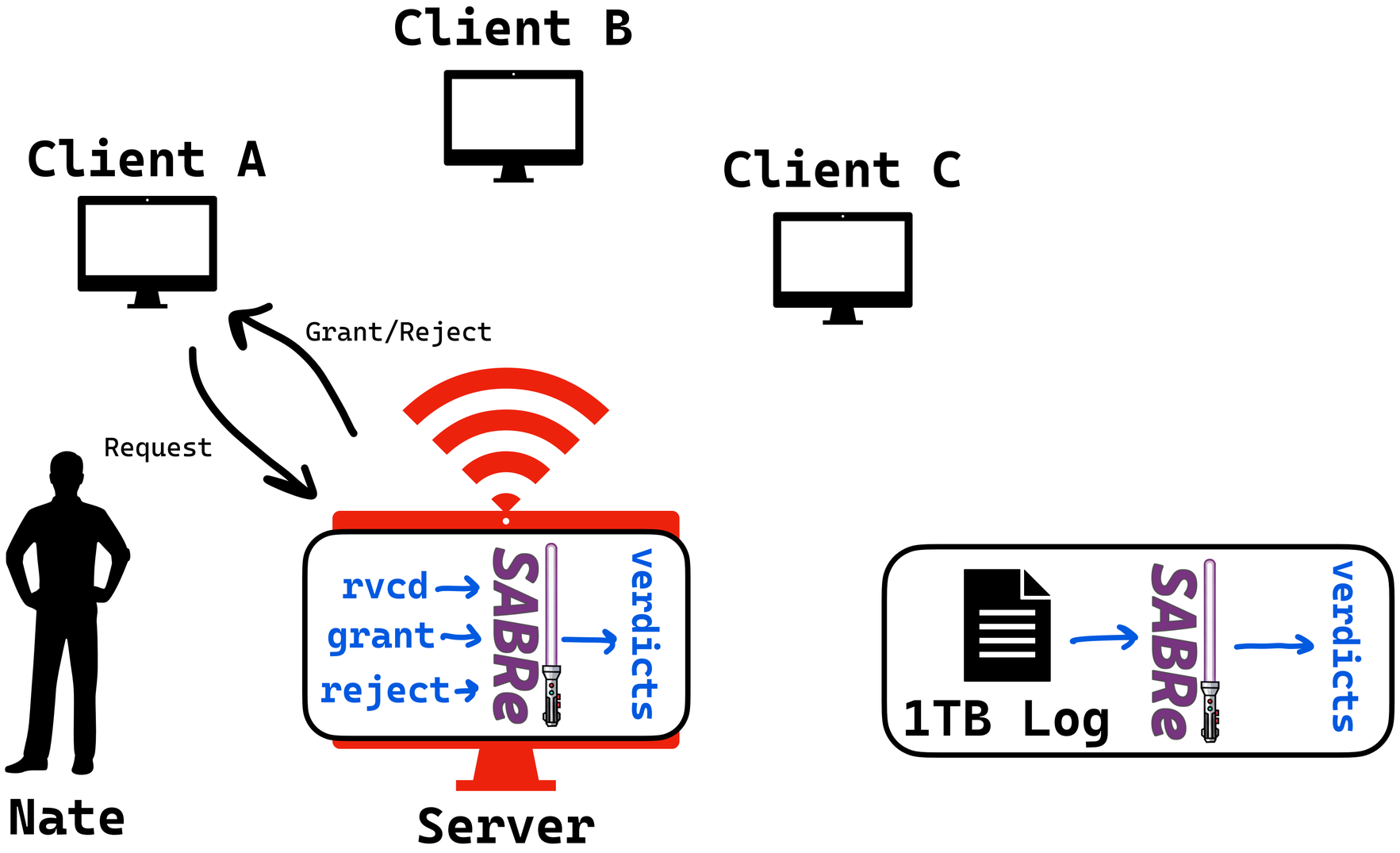


10.52x faster than **R2U2** **4.96x** faster than **Hydra**



10.52x faster than **R2U2**

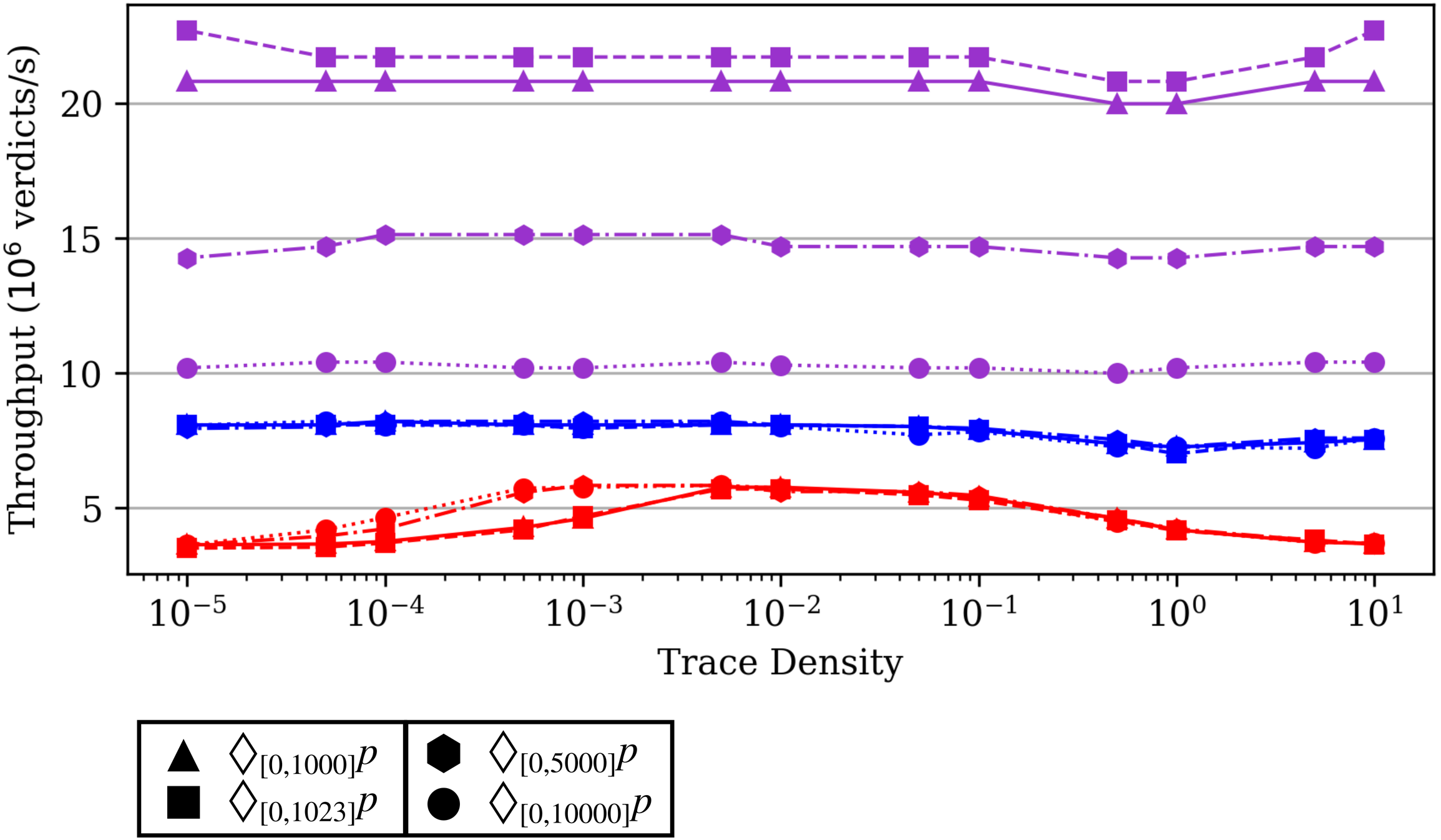
4.96x faster than **Hydra**

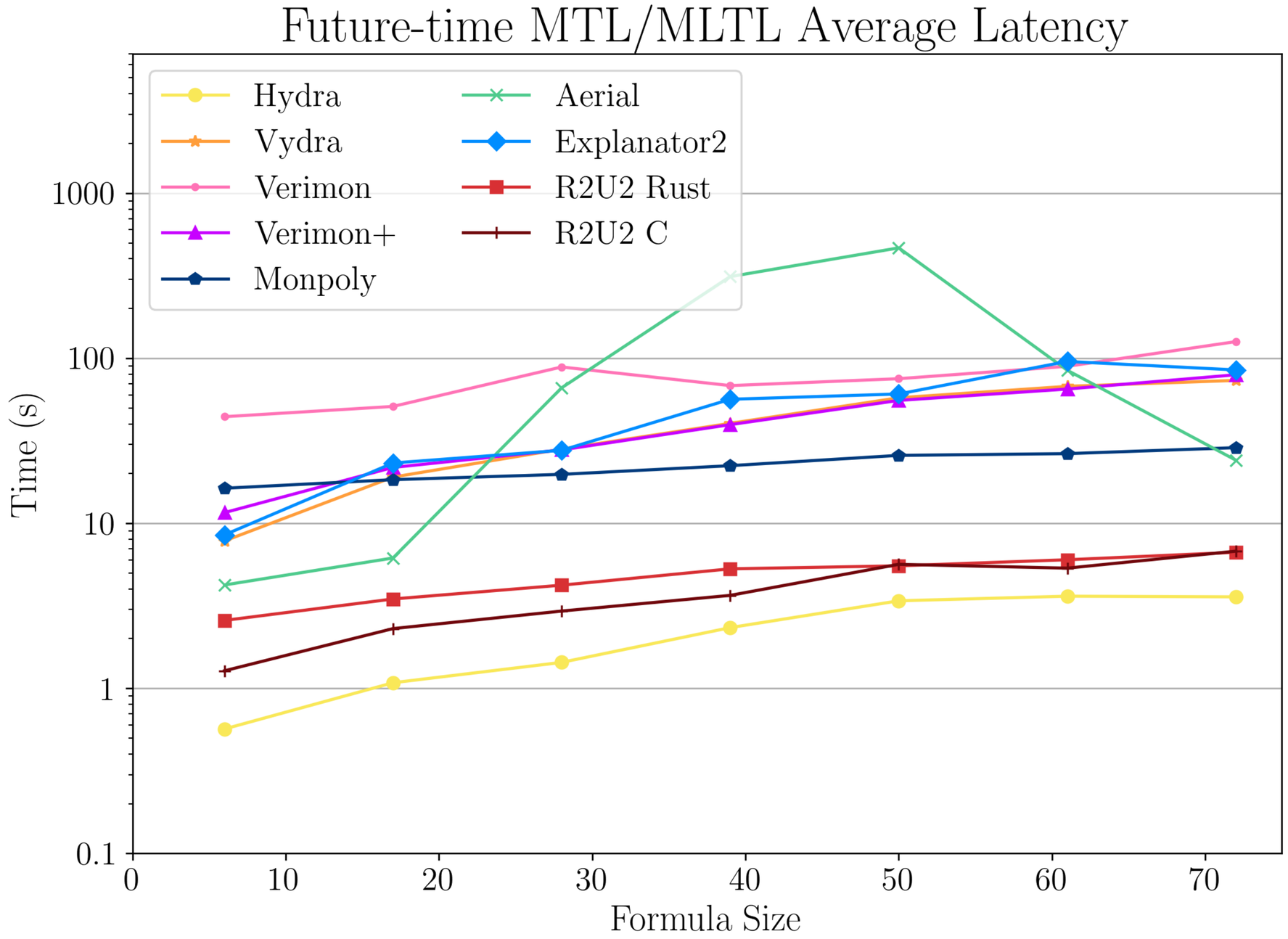


SABRe enables high-throughput monitoring of MLTL formulas.

Summary

- Defined a **bit-vector encoding** for MLTL formulas and traces.
- Applied the encoding to the MLTL **monitoring** and **satisfiability** problems.
- Implemented monitoring algorithm in the new tool **SABRe**.
- Experimental evaluation shows superior throughput compared to state-of-the-art online MLTL monitors (**10x higher**).


Throughput (10^6 verdicts/s)
20
15
10
5
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
Trace Density
$\Diamond_{[0,1000]}p$
$\Diamond_{[0,1023]}p$
$\Diamond_{[0,5000]}p$
$\Diamond_{[0,10000]}p$


Future-time MTL/MLTL Average Latency
Hydra
Vydra
Verimon
Verimon+
Monpoly
Aerial
Explanator2
R2U2 Rust
R2U2 C
Time (s)
1000
100
10
1
0.1
0
10
20
30
40
50
60
70
Formula Size

Definition 2 (MLTL Semantics)**.** The satisfaction of an MLTL formula by a trace π at timestamp τ is defined recursively as:

$\pi, \tau \models p$ iff $p \in \pi[\tau]$ $\quad\quad$ $\pi, \tau \models \neg\varphi$ iff $\pi, \tau \not\models \varphi$

$\pi, \tau \models \varphi \wedge \psi$ iff $\pi, \tau \models \varphi$ and $\pi, \tau \models \psi$

$\pi, \tau \models \Diamond_{[l,u]}\varphi$ iff $\tau + l < |\pi|$ and $\pi, i \models \varphi$ for some $i < |\pi|$ such that $l + \tau \leq i \leq u + \tau$

$\pi, \tau \models \varphi \; \mathcal{U}_{[l,u]} \; \psi$ iff $\tau + l < |\pi|$ and $\pi, i \models \psi$ for some $i < |\pi|$ such that $l + \tau \leq i \leq u + \tau$ and $\pi, j \models \varphi$ for all j such that $l + \tau \leq j < i$