On Synthesizing Controllers from Bounded-Response Properties

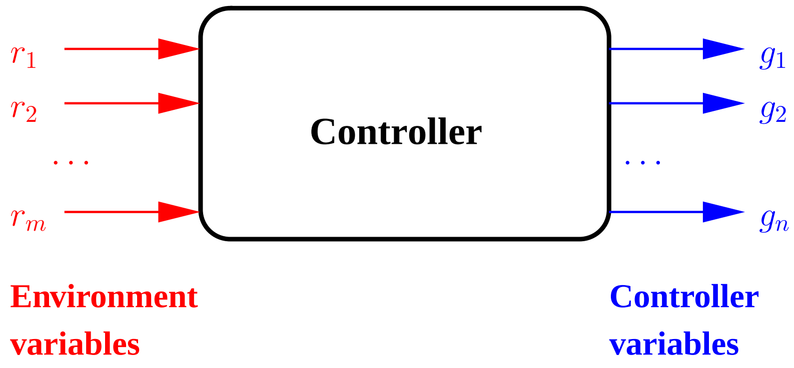
Oded Maler
Verimag

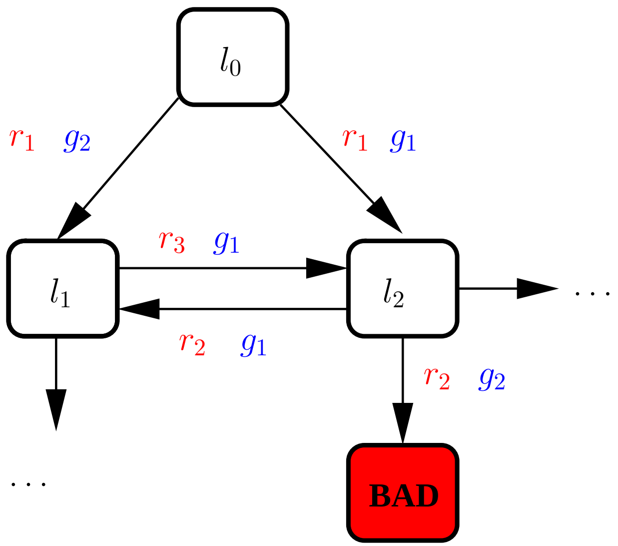
Dejan Ničković
Verimag

Amir Pnueli
Weizmann Institute
NYU

Overview

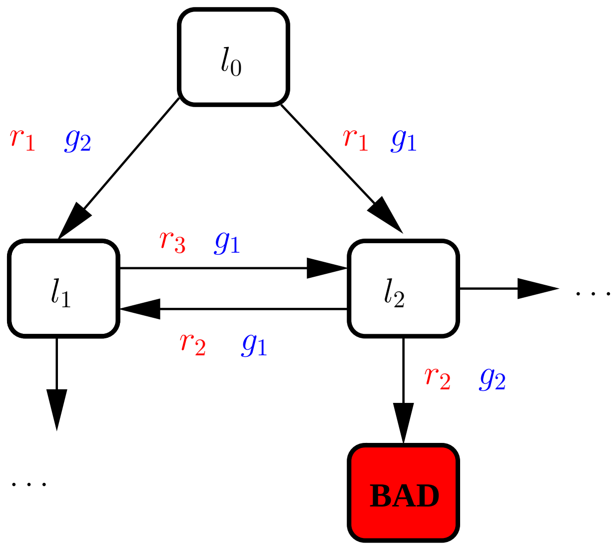
- Introduction
- Property-based Synthesis
  - Bounded-response Properties
- MTL-B
  - Syntax and Semantics
  - Non-Determinism
- From MTL-B to Deterministic Temporal Testers
  - Pastification of MTL-B formulae
  - Bounded-variability assumption
- Application to Synthesis: Arbiter Example
  - Specification in MTL-B
  - Experimental Results
- Conclusion





- Automatic controller synthesis from high-level specifications
  - Problem posed in [Chu63]
  - Theoretically solved in [BL69,TB73]

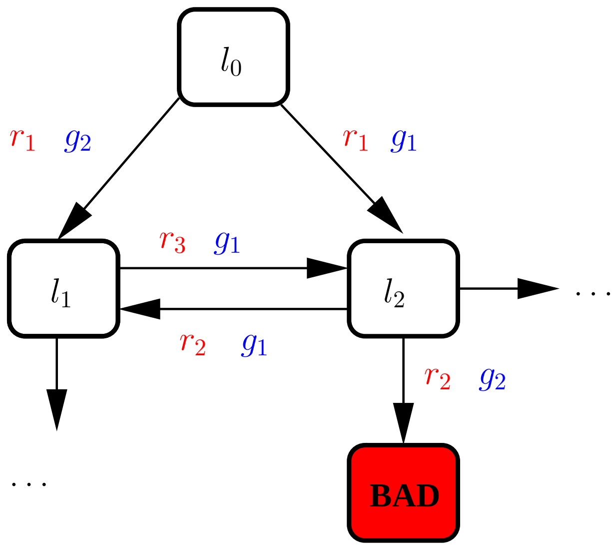
Introduction



$$\Box(r_1 \rightarrow r_1 \mathcal{S} g_1)$$
$$\Box(g_1 \rightarrow \Diamond_{[0,1]}\, r_1)$$
$$\Box(\overline{g_1} \rightarrow \Diamond_{[0,2]}\, \overline{r_1})$$
...

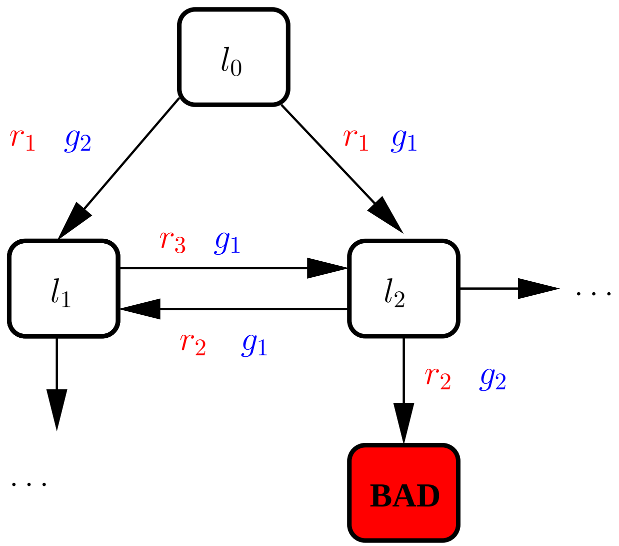
- Synthesizing controllers from temporal logic formulae [PR89]
  - Recent improvements [PPS06,PP06]
- Property-based synthesis problem:
  Given a temporal property φ defined over two distinct alphabets A and B, build a finite-state transducer (controller) from A^ω to B^ω such that all of its behaviors satisfy φ.
- We are interested in controller synthesis from **real-time** temporal logic specifications

Introduction



$$\Box(r_1 \rightarrow r_1 \mathcal{S} g_1)$$
$$\Box(g_1 \rightarrow \Diamond_{[0,1]} r_1)$$
$$\Box(\overline{g_1} \rightarrow \Diamond_{[0,2]} \overline{r_1})$$
$$\ldots$$

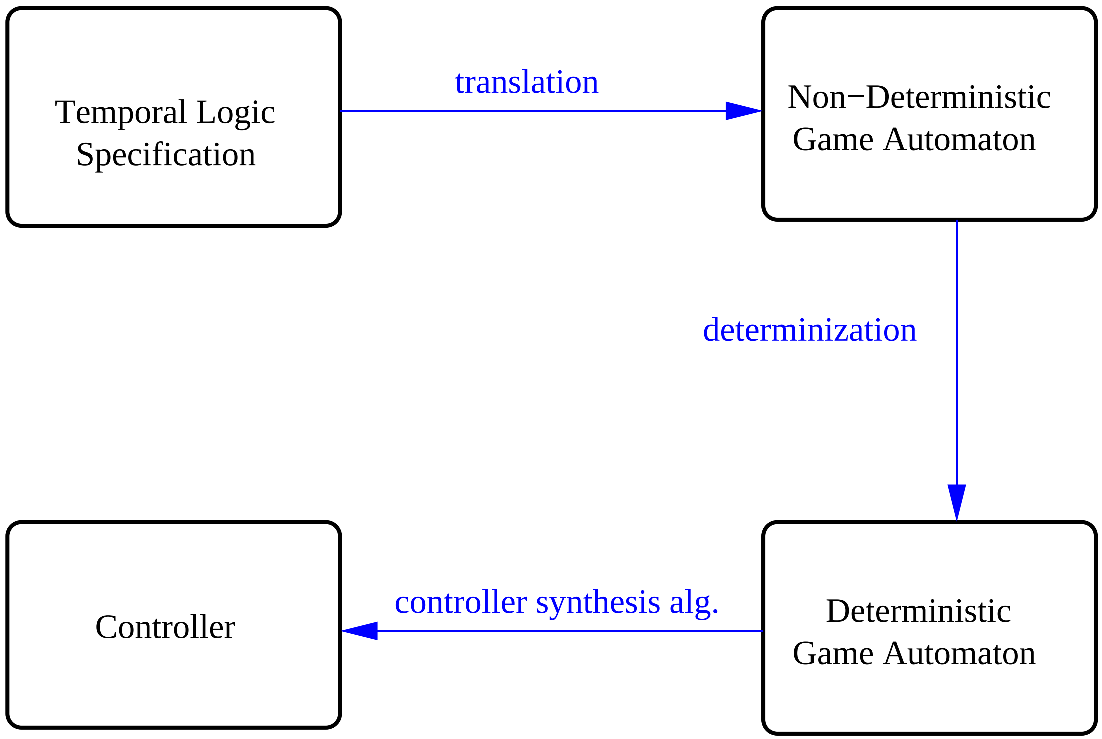
- Synthesizing controllers from temporal logic formulae [PR89]
  - Recent improvements [PPS06,PP06]
- Property-based synthesis problem:
  Given a temporal property φ defined over two distinct alphabets A and B, build a finite-state transducer (controller) from A^ω to B^ω such that all of its behaviors satisfy φ.
- We are interested in controller synthesis from **real-time** temporal logic specifications



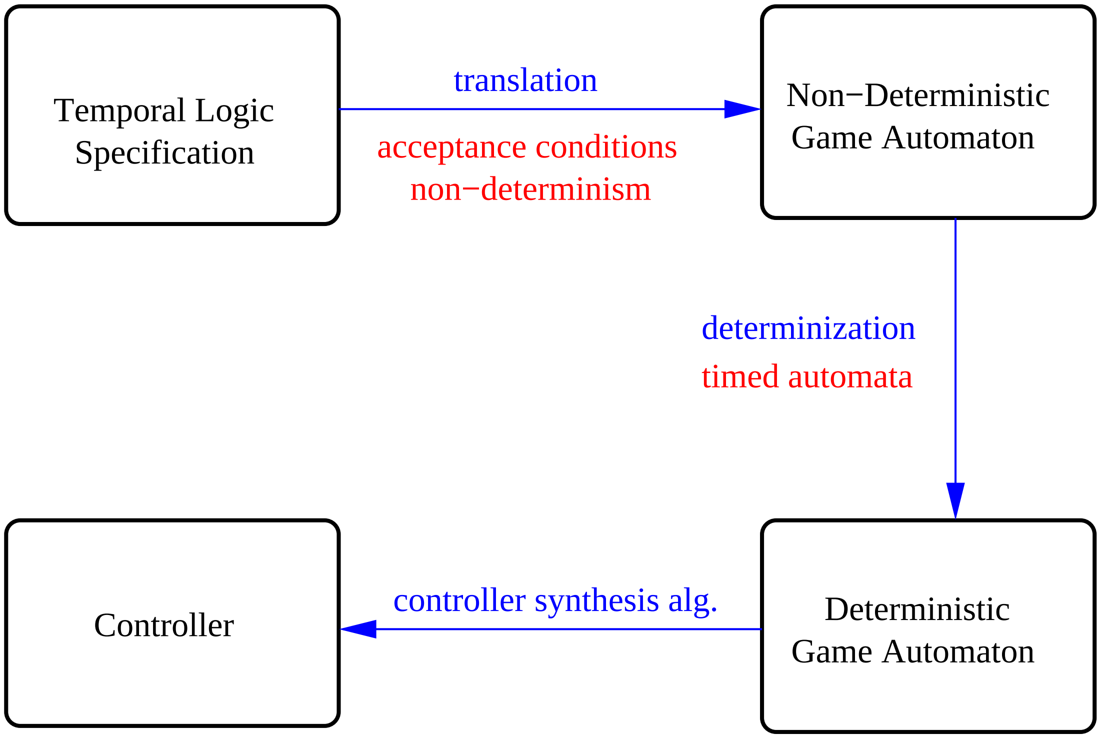
$$\Box(r_1 \rightarrow r_1 \mathcal{S} g_1)$$
$$\Box(g_1 \rightarrow \Diamond_{[0,1]} r_1)$$
$$\Box(\overline{g_1} \rightarrow \Diamond_{[0,2]} \overline{r_1})$$
$$\ldots$$

- Synthesizing controllers from temporal logic formulae [PR89]
  - Recent improvements [PPS06,PP06]
- Property-based synthesis problem:
  Given a temporal property φ defined over two distinct alphabets A and B, build a finite-state transducer (controller) from A^ω to B^ω such that all of its behaviors satisfy φ.
- We are interested in controller synthesis from **real-time** temporal logic specifications

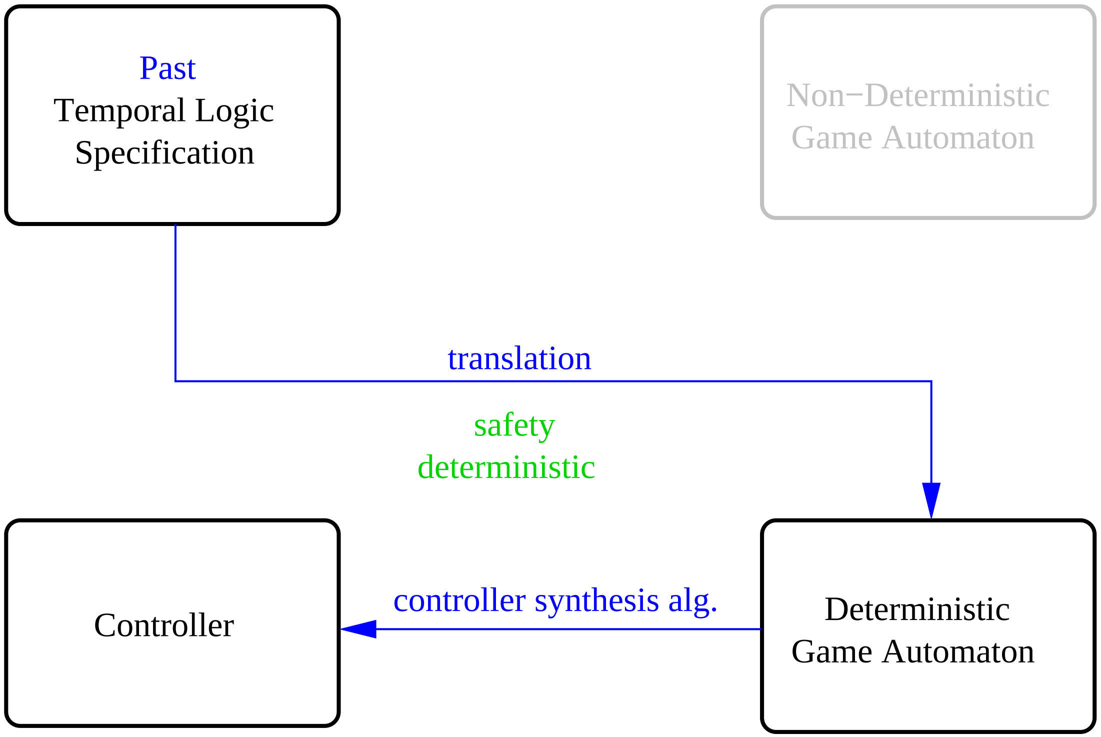
Temporal Logic and Controller Synthesis



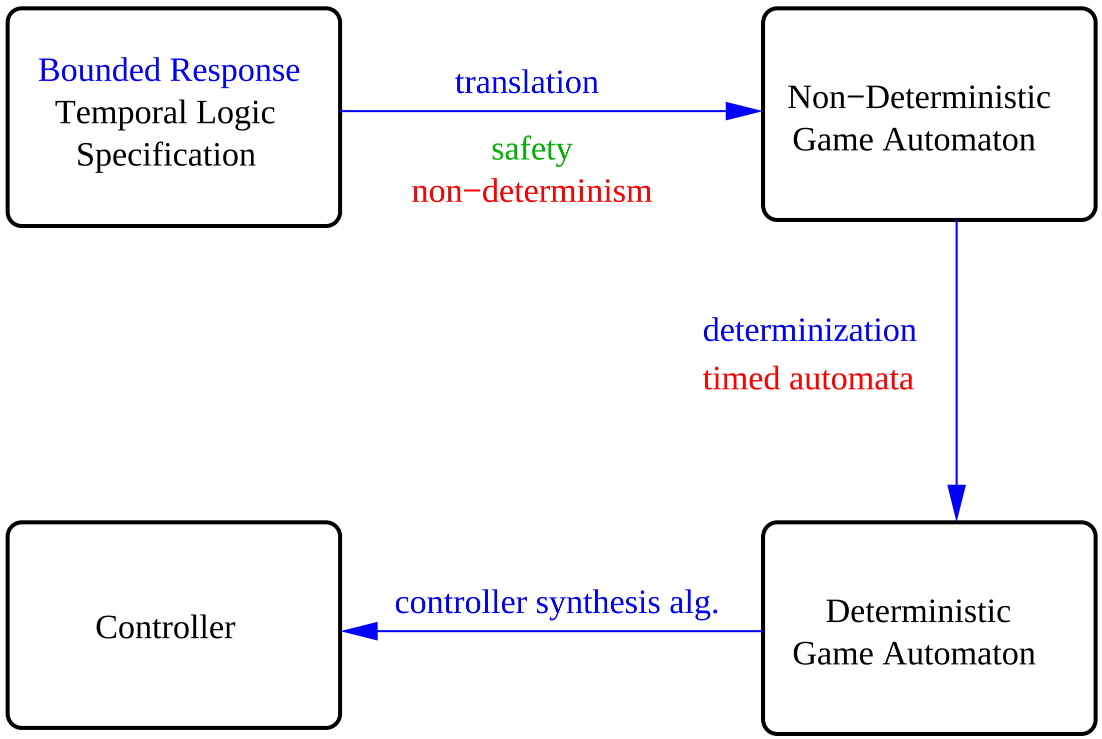
Temporal Logic and Controller Synthesis



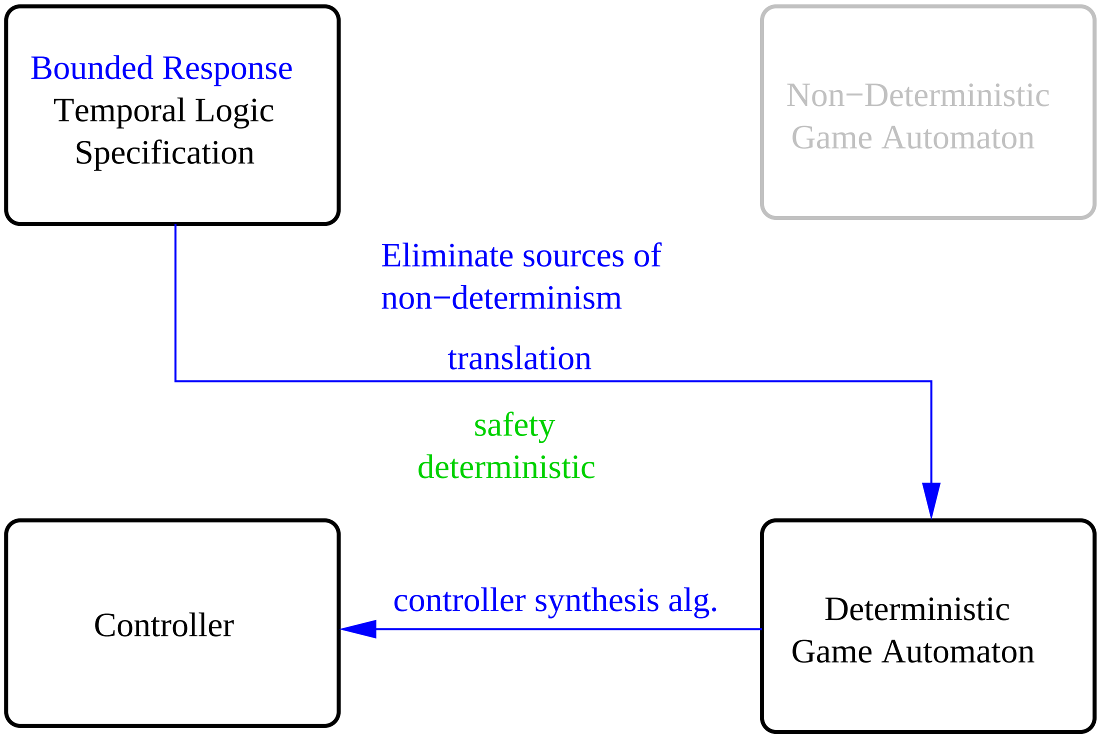
Temporal Logic and Controller Synthesis



Temporal Logic and Controller Synthesis



Temporal Logic and Controller Synthesis



Motivation for Bounded-Response Properties

- **Bounded-response** correspond to **safety** properties
  - $\longrightarrow$ Limited scope wrt more general **liveness** properties
- Liveness properties abstract away the upper bound requirement of occurrence of events
  - But many applications require specifying explicitly such upper bound:
    - Hard real-time systems
    - Scheduling problems
    - ...
- We choose **Bounded Response Metric Temporal Logic** - MTL-B as the specification formalism
  - MTL [Koy90] without **unbounded until**
  - Punctual operators (unlike MITL [AFH96])
  - Allows specifying non-trivial properties
  - Can be interpreted both in **discrete** and **dense** time
  - We consider specifications of type $\Box\varphi$ where φ is an MTL-B formula

Motivation for Bounded-Response Properties

- **Bounded-response** correspond to **safety** properties
  - $\longrightarrow$ Limited scope wrt more general **liveness** properties
- Liveness properties abstract away the upper bound requirement of occurrence of events
  - But many applications require specifying explicitly such upper bound:
    - Hard real-time systems
    - Scheduling problems
    - . . .
- We choose **Bounded Response Metric Temporal Logic** - MTL-B as the specification formalism
  - MTL [Koy90] without **unbounded until**
  - Punctual operators (unlike MITL [AFH96])
  - Allows specifying non-trivial properties
  - Can be interpreted both in **discrete** and **dense** time
  - We consider specifications of type $\Box\varphi$ where φ is an MTL-B formula

Motivation for Bounded-Response Properties

- **Bounded-response** correspond to **safety** properties
  - $\longrightarrow$ Limited scope wrt more general **liveness** properties
- Liveness properties abstract away the upper bound requirement of occurrence of events
  - But many applications require specifying explicitly such upper bound:
    - Hard real-time systems
    - Scheduling problems
    - ...
- We choose **Bounded Response Metric Temporal Logic** - MTL-B as the specification formalism
  - MTL [Koy90] without **unbounded until**
  - Punctual operators (unlike MITL [AFH96])
  - Allows specifying non-trivial properties
  - Can be interpreted both in **discrete** and **dense** time
  - We consider specifications of type $\Box\varphi$ where φ is an MTL-B formula

Motivation for Bounded-Response Properties

- **Bounded-response** correspond to **safety** properties
  - $\longrightarrow$ Limited scope wrt more general **liveness** properties
- Liveness properties abstract away the upper bound requirement of occurrence of events
  - But many applications require specifying explicitly such upper bound:
    - Hard real-time systems
    - Scheduling problems
    - ...
- We choose **Bounded Response Metric Temporal Logic** - MTL-B as the specification formalism
  - MTL [Koy90] without **unbounded until**
  - Punctual operators (unlike MITL [AFH96])
  - Allows specifying non-trivial properties
  - Can be interpreted both in **discrete** and **dense** time
  - We consider specifications of type $\Box\varphi$ where φ is an MTL-B formula

Motivation for Bounded-Response Properties

- **Bounded-response** correspond to **safety** properties
  - $\longrightarrow$ Limited scope wrt more general **liveness** properties
- Liveness properties abstract away the upper bound requirement of occurrence of events
  - But many applications require specifying explicitly such upper bound:
    - Hard real-time systems
    - Scheduling problems
    - ...
- We choose **Bounded Response Metric Temporal Logic** - MTL-B as the specification formalism
  - MTL [Koy90] without **unbounded until**
  - Punctual operators (unlike MITL [AFH96])
  - Allows specifying non-trivial properties
  - Can be interpreted both in **discrete** and **dense** time
  - We consider specifications of type $\Box\varphi$ where φ is an MTL-B formula

Motivation for Bounded-Response Properties

- **Bounded-response** correspond to **safety** properties
  - $\longrightarrow$ Limited scope wrt more general **liveness** properties
- Liveness properties abstract away the upper bound requirement of occurrence of events
  - But many applications require specifying explicitly such upper bound:
    - Hard real-time systems
    - Scheduling problems
    - ...
- We choose **Bounded Response Metric Temporal Logic** - MTL-B as the specification formalism
  - MTL [Koy90] without **unbounded until**
  - Punctual operators (unlike MITL [AFH96])
  - Allows specifying non-trivial properties
  - Can be interpreted both in **discrete** and **dense** time
  - We consider specifications of type $\Box\varphi$ where φ is an MTL-B formula

Motivation for Bounded-Response Properties

- **Bounded-response** correspond to **safety** properties
  - $\longrightarrow$ Limited scope wrt more general **liveness** properties
- Liveness properties abstract away the upper bound requirement of occurrence of events
  - But many applications require specifying explicitly such upper bound:
    - Hard real-time systems
    - Scheduling problems
    - ...
- We choose **Bounded Response Metric Temporal Logic** - MTL-B as the specification formalism
  - MTL [Koy90] without **unbounded until**
  - Punctual operators (unlike MITL [AFH96])
  - Allows specifying non-trivial properties
  - Can be interpreted both in **discrete** and **dense** time
  - We consider specifications of type $\Box\varphi$ where φ is an MTL-B formula

MTL-B: *Syntax and Semantics*

- **Syntax:**

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{S}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{S} \varphi_2 \mid \varphi_1 \mathcal{P}_{[a,b]} \varphi_2$$

- **Semantics:**

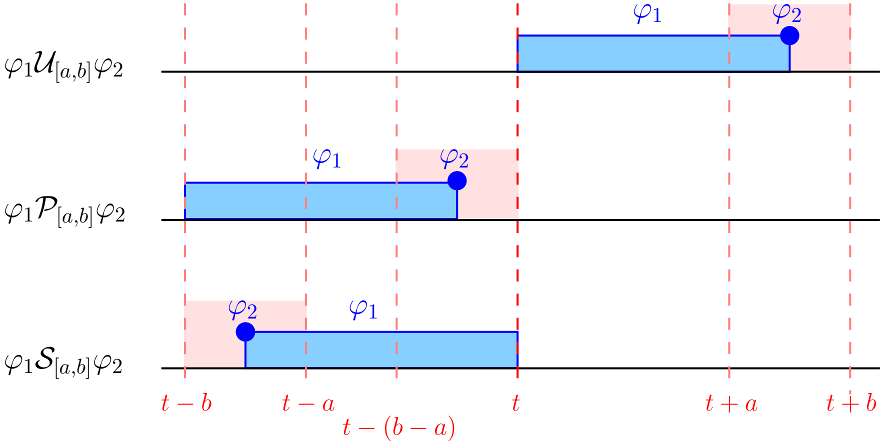
$$\begin{array}{lcl} & \ldots & \\ (\xi, t) \models \varphi_1\ \mathcal{U}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \oplus [a,b]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t''[t, t'], (\xi, t'') \models \varphi_1 \\ (\xi, t) \models \varphi_1\ \mathcal{P}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \ominus [0, b-a]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t'' \in [t-b, t'], (\xi, t'') \models \varphi_1 \\ (\xi, t) \models \varphi_1\ \mathcal{S}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \ominus [a,b]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t'' \in [t, t'], (\xi, t'') \models \varphi_1 \\ & \ldots & \end{array}$$

- **Syntax:**

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{S}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{S} \varphi_2 \mid \varphi_1 \mathcal{P}_{[a,b]} \varphi_2$$

- **Semantics:**

$$\begin{array}{lcl} & \cdots & \\ (\xi, t) \models \varphi_1\ \mathcal{U}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \oplus [a,b]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t''[t, t'], (\xi, t'') \models \varphi_1 \\ (\xi, t) \models \varphi_1\ \mathcal{P}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \ominus [0, b-a]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t'' \in [t-b, t'], (\xi, t'') \models \varphi_1 \\ (\xi, t) \models \varphi_1\ \mathcal{S}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \ominus [a,b]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t'' \in [t, t'], (\xi, t'') \models \varphi_1 \\ & \cdots & \end{array}$$



- **Syntax:**

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{S}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{S} \varphi_2 \mid \varphi_1 \mathcal{P}_{[a,b]} \varphi_2$$

- **Semantics:**

$$\begin{array}{lcl} & \ldots & \\ (\xi, t) \models \varphi_1\ \mathcal{U}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \oplus [a, b]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t''[t, t'], (\xi, t'') \models \varphi_1 \\ (\xi, t) \models \varphi_1\ \mathcal{P}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \ominus [0, b-a]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t'' \in [t-b, t'], (\xi, t'') \models \varphi_1 \\ (\xi, t) \models \varphi_1\ \mathcal{S}_{[a,b]}\ \varphi_2 & \leftrightarrow & \exists\ t' \in t \ominus [a, b]\ (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t'' \in [t, t'], (\xi, t'') \models \varphi_1 \\ & \ldots & \end{array}$$

- **Notes:**
  - “Handshake” semantics of bounded until
  - **Precedes** operator $\sim$ past equivalent of bounded until
- **Derived operators:** $\diamondminus_{[a,b]}, \boxminus_{[a,b]}, \Diamond_{[a,b]}, \Box_{[a,b]}$

MTL-B: *Syntax and Semantics*

- **Syntax:**

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{S}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{S} \varphi_2 \mid \varphi_1 \mathcal{P}_{[a,b]} \varphi_2$$

- **Semantics:**

$$\begin{array}{rcl} & \ldots & \\ (\xi, t) \models \varphi_1 \; \mathcal{U}_{[a,b]} \; \varphi_2 & \leftrightarrow & \exists\, t' \in t \oplus [a,b] \; (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t''[t, t'], (\xi, t'') \models \varphi_1 \\ (\xi, t) \models \varphi_1 \; \mathcal{P}_{[a,b]} \; \varphi_2 & \leftrightarrow & \exists\, t' \in t \ominus [0, b-a] \; (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t'' \in [t-b, t'], (\xi, t'') \models \varphi_1 \\ (\xi, t) \models \varphi_1 \; \mathcal{S}_{[a,b]} \; \varphi_2 & \leftrightarrow & \exists\, t' \in t \ominus [a,b] \; (\xi, t') \models \varphi_2 \text{ and} \\ & & \forall t'' \in [t, t'], (\xi, t'') \models \varphi_1 \\ & \ldots & \end{array}$$

- **Notes:**
  - “Handshake” semantics of bounded until
  - **Precedes** operator $\sim$ past equivalent of bounded until
- **Derived operators:** $\diamondminus_{[a,b]}, \boxminus_{[a,b]}, \diamond_{[a,b]}, \Box_{[a,b]}$

MTL-B *and Non-Determinism*

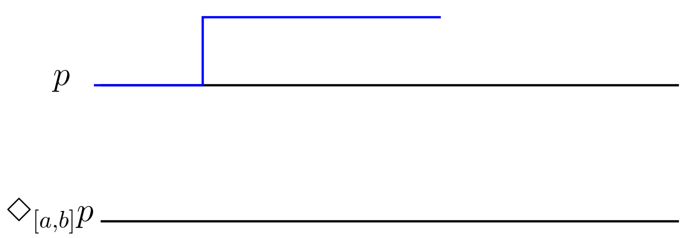
- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$
  - **Past** fragments of temporal logics have **causal** semantics
- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1 p$ - perfect shift register for p

MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$
  - **Past** fragments of temporal logics have **causal** semantics
- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1 p$ - perfect shift register for p

MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$

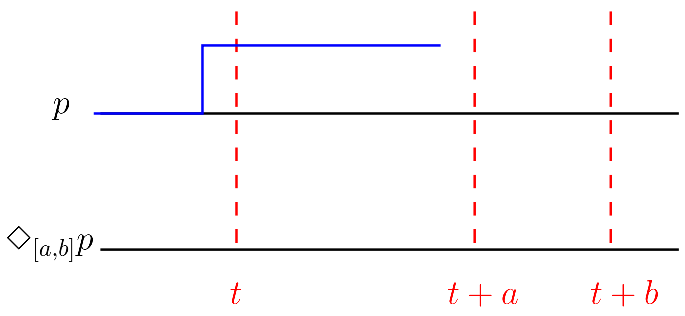


  - **Past** fragments of temporal logics have **causal** semantics

- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1\, p$ - perfect shift register for p

MTL-B *and Non-Determinism*

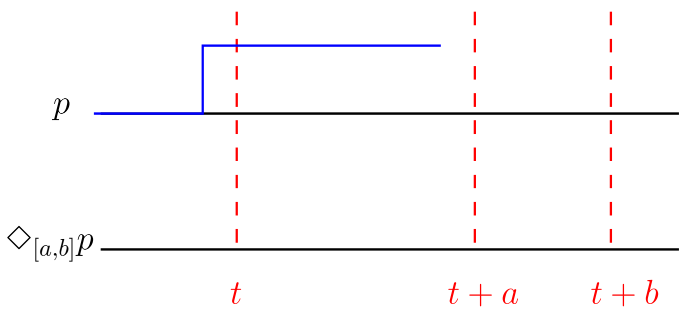
- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$



  - **Past** fragments of temporal logics have **causal** semantics
- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1 p$ - perfect shift register for p

MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$

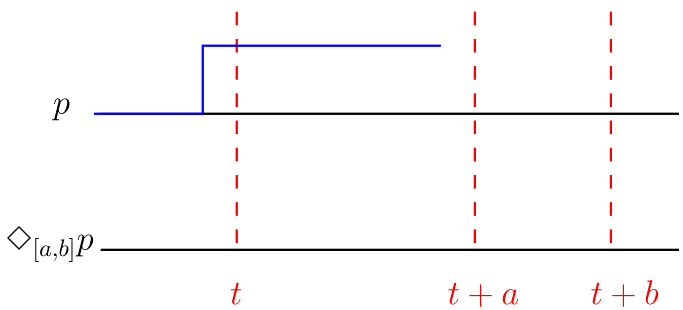


  - **Past** fragments of temporal logics have **causal** semantics

- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\ominus_1 p$ - perfect shift register for p

MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$

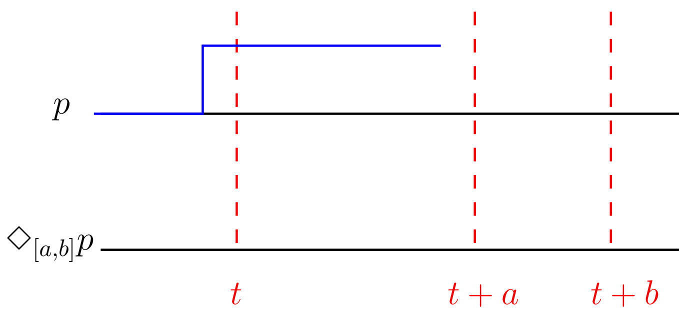


  - **Past** fragments of temporal logics have **causal** semantics

- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1\, p$ - perfect shift register for p

MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$

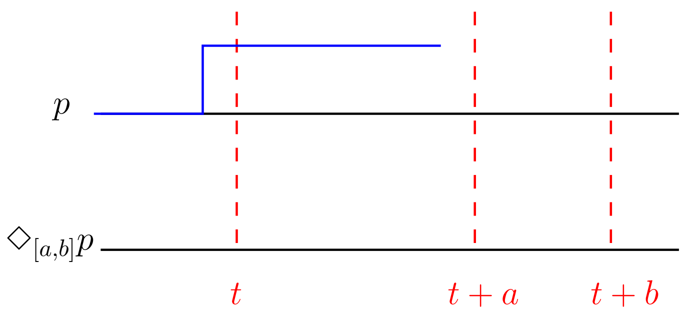


  - **Past** fragments of temporal logics have **causal** semantics

- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1 p$ - perfect shift register for p

MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$

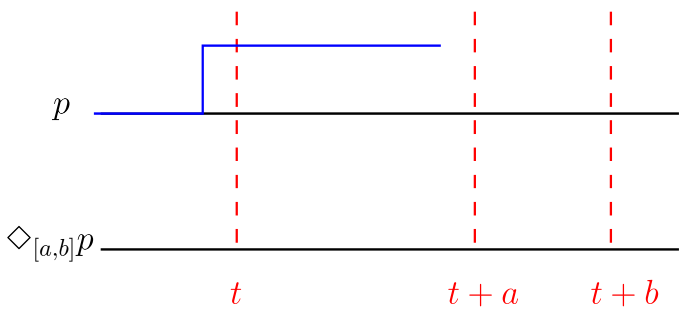


  - **Past** fragments of temporal logics have **causal** semantics

- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1 p$ - perfect shift register for p

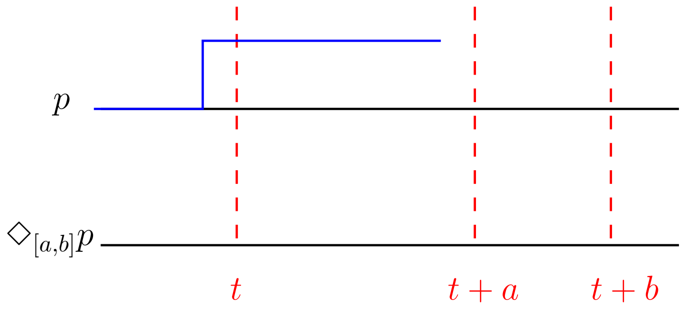
MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$



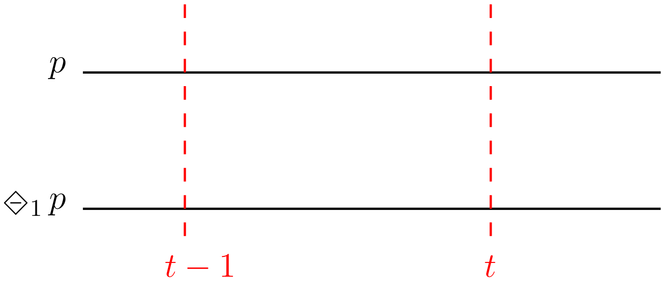
  - **Past** fragments of temporal logics have **causal** semantics
- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1\, p$ - perfect shift register for p

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$



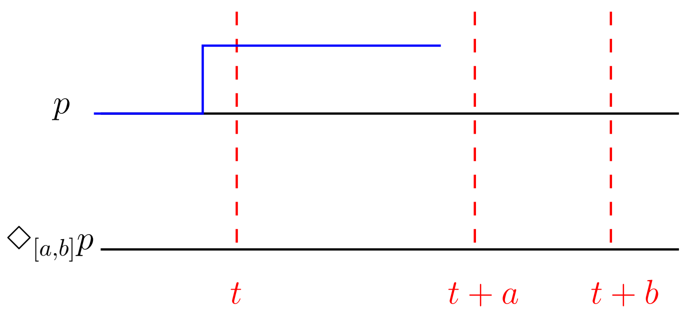
  - **Past** fragments of temporal logics have **causal** semantics

- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1\, p$ - perfect shift register for p



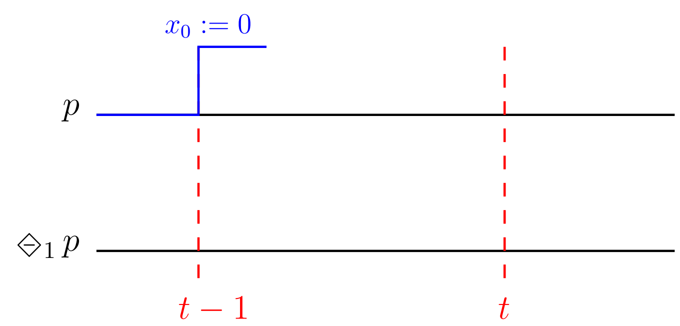
MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$



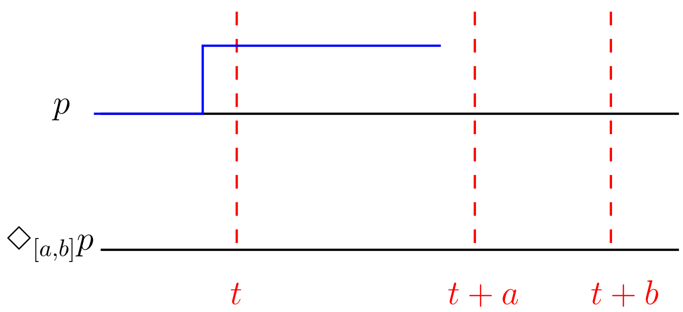
  - **Past** fragments of temporal logics have **causal** semantics

- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1\, p$ - perfect shift register for p



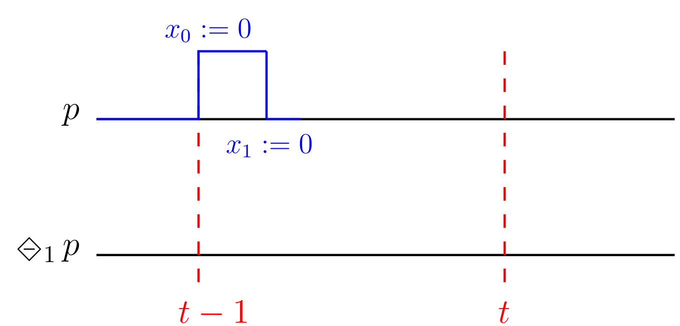
MTL-B *and Non-Determinism*

- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$

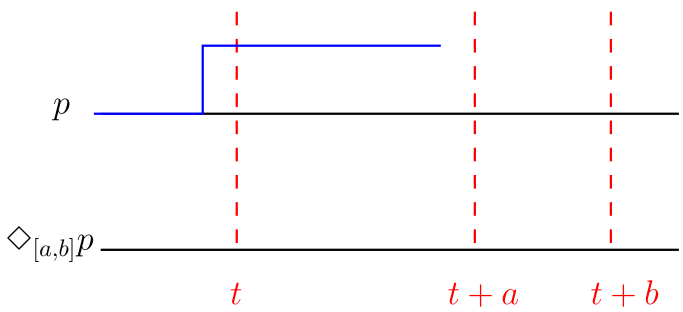


  - **Past** fragments of temporal logics have **causal** semantics

- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1 p$ - perfect shift register for p

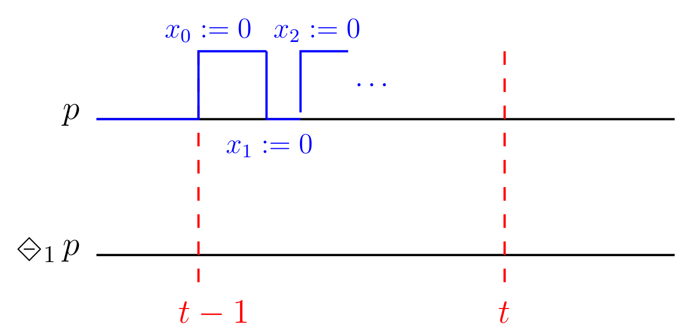


- Two sources of non-determinism
- Acausality
  - Semantics of **future** temporal logics **acausal**
    - Satisfiability of φ at time t depends on the input signal value at time $t' \geq t$



  - **Past** fragments of temporal logics have **causal** semantics

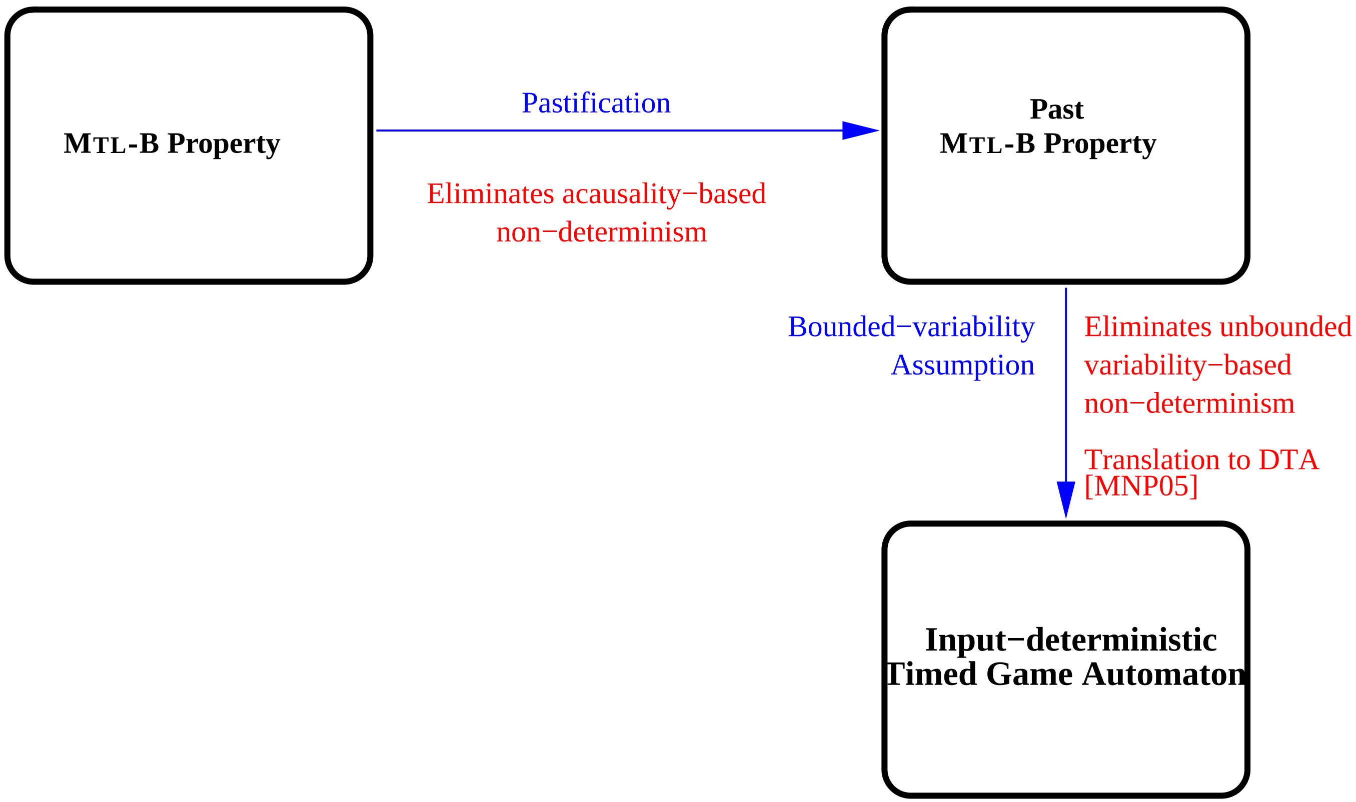
- Unbounded Variability
  - **No bound** on the **variability** of input signals
  - $\rightarrow$ remember **unbounded number of events**
    - **Example:** $\diamondminus_1 p$ - perfect shift register for p



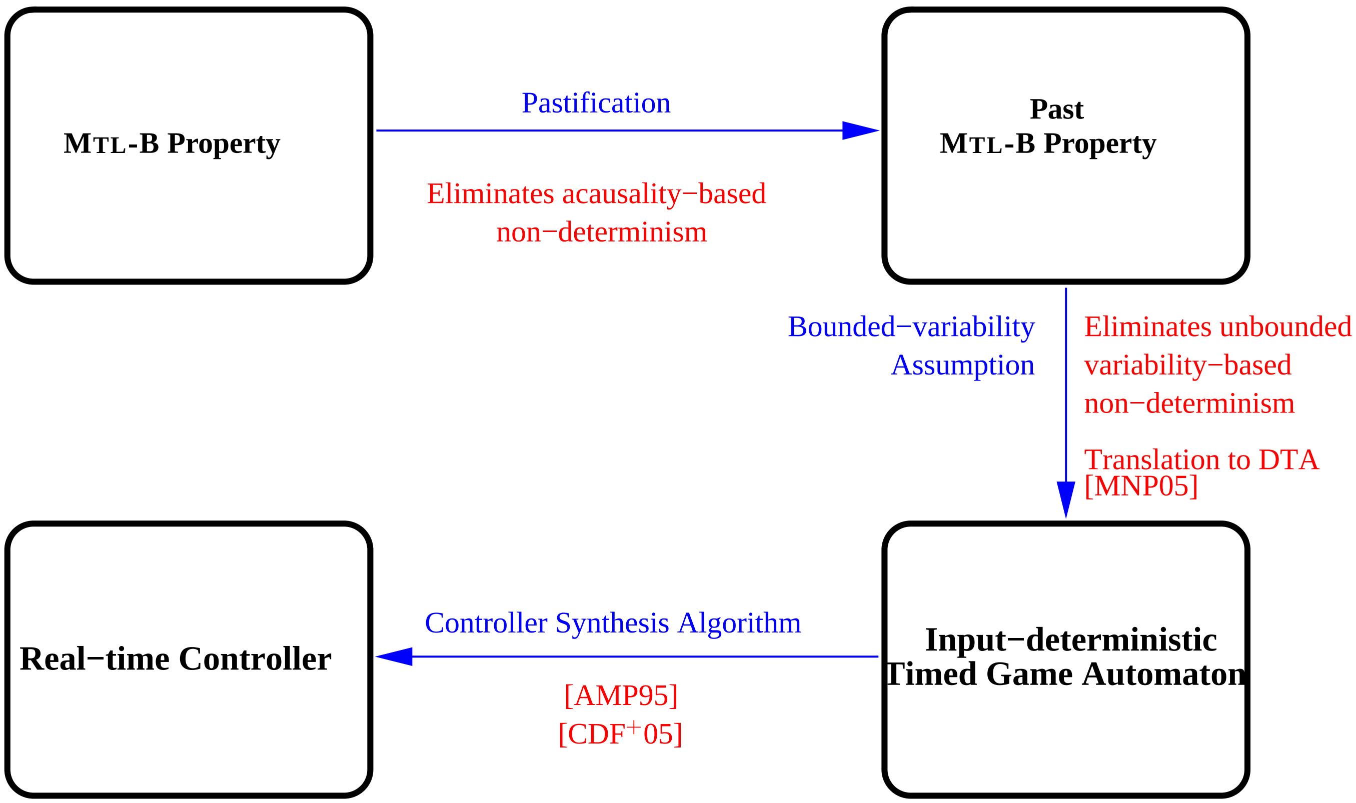
From MTL-B *to Deterministic Timed Automata: Overview*

MTL-B Property

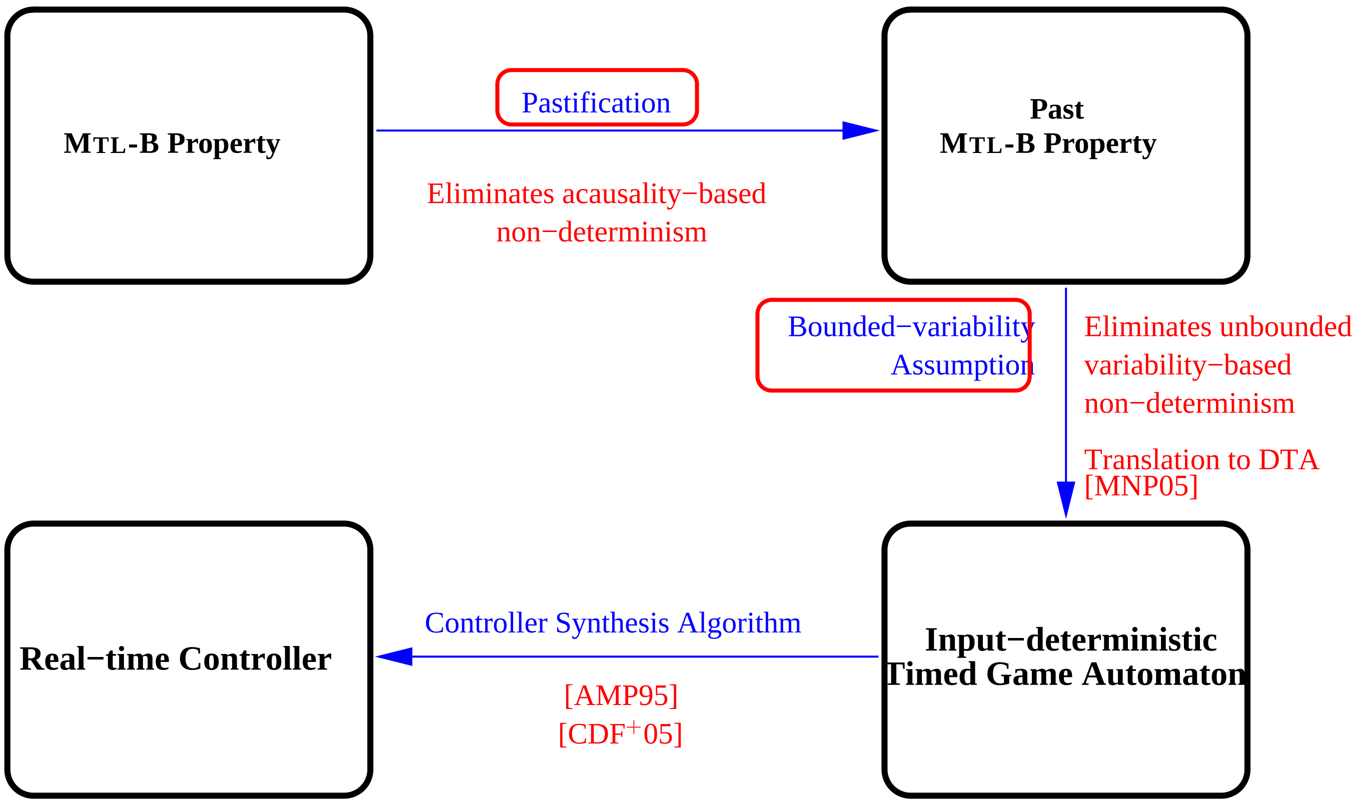
From MTL-B to Deterministic Timed Automata: Overview


MTL-B Property
Pastification
Eliminates acausality−based non−determinism
Past MTL-B Property
Bounded−variability Assumption
Eliminates unbounded variability−based non−determinism
Translation to DTA [MNP05]
Input−deterministic Timed Game Automaton

From MTL-B *to Deterministic Timed Automata: Overview*



From MTL-B *to Deterministic Timed Automata: Overview*



Pastification of MTL-B *formulae*

- **Key idea:** Change the time direction from **future** to **past**
  - MTL-B formula fully determined withing a bounded horizon
  - $\rightarrow$ Eliminate the “predictive” aspect of the semantics
- **Example:** $\varphi = p \rightarrow \Diamond_{[1,2]} \Box_{[0,2]} q$
- What would be the “equivalent” past formula ψ that describes the same pattern from $t+4$?
  - $\psi = \diamondminus_{4} p \rightarrow \diamondminus_{[0,1]} \boxminus_{[0,2]} q$

Pastification of MTL-B *formulae*

- **Key idea:** Change the time direction from **future** to **past**
  - MTL-B formula fully determined withing a bounded horizon
  - $\longrightarrow$ Eliminate the "predictive" aspect of the semantics
- **Example:** $\varphi = p \rightarrow \Diamond_{[1,2]} \Box_{[0,2]} q$
- What would be the "equivalent" past formula ψ that describes the same pattern from $t + 4$?
  - $\psi = \diamondminus_{4} p \rightarrow \diamondminus_{[0,1]} \boxminus_{[0,2]} q$

Pastification of MTL-B *formulae*

- **Key idea:** Change the time direction from **future** to **past**
  - MTL-B formula fully determined withing a bounded horizon
  - $\longrightarrow$ Eliminate the "predictive" aspect of the semantics
- **Example:** $\varphi = p \rightarrow \Diamond_{[1,2]} \Box_{[0,2]} q$

$\rightarrow$	t	$t+1$	$t+2$	$t+3$	$t+4$	$\rightarrow$
	$\overline{p}*$	$**$	$**$	$**$	$**$	
$\ldots$	$p*$	$*q$	$*q$	$*q$	$**$	$\ldots$
	$p*$	$**$	$*q$	$*q$	$*q$	

- What would be the "equivalent" past formula ψ that describes the same pattern from $t+4$?
  - $\psi = \ominus_4 p \rightarrow \ominus_{[0,1]} \boxminus_{[0,2]} q$

Pastification of MTL-B *formulae*

- **Key idea:** Change the time direction from **future** to **past**
  - MTL-B formula fully determined withing a bounded horizon
  - $\rightarrow$ Eliminate the "predictive" aspect of the semantics
- **Example:** $\varphi = p \rightarrow \Diamond_{[1,2]} \Box_{[0,2]} q$

$\rightarrow$	t	$t+1$	$t+2$	$t+3$	$t+4$	$\rightarrow$
	$\bar{p}*$	$**$	$**$	$**$	$**$	
$\ldots$	$p*$	$*q$	$*q$	$*q$	$**$	$\ldots$
	$p*$	$**$	$*q$	$*q$	$*q$	

- What would be the "equivalent" past formula ψ that describes the same pattern from $t+4$?
  - $\psi = \diamondminus_{4}\, p \rightarrow \diamondminus_{[0,1]} \boxminus_{[0,2]} q$

- **Key idea:** Change the time direction from **future** to **past**
  - MTL-B formula fully determined withing a bounded horizon
  - $\longrightarrow$ Eliminate the "predictive" aspect of the semantics
- **Example:** $\varphi = p \rightarrow \Diamond_{[1,2]} \Box_{[0,2]}\, q$

$\rightarrow$	t	$t+1$	$t+2$	$t+3$	$t+4$	$\rightarrow$
	$\overline{p}*$	$**$	$**$	$**$	$**$	
$\ldots$	$p*$	$*q$	$*q$	$*q$	$**$	$\ldots$
	$p*$	$**$	$*q$	$*q$	$*q$	
$\leftarrow$	$t-4$	$t-3$	$t-2$	$t-1$	t	$\leftarrow$

- What would be the "equivalent" past formula ψ that describes the same pattern from $t+4$?
  - $\psi = \ominus\!\!\!\!\Diamond_{4}\, p \rightarrow \ominus\!\!\!\!\Diamond_{[0,1]} \boxminus_{[0,2]}\, q$

Temporal Depth of an MTL-B *formula*

- Each future MTL-B formula admits a number $D(\varphi)$ indicating its **temporal depth**
  - The satisfaction of φ by a signal ξ from any position t is fully determined within the interval $[t, t + D(\varphi)]$

$$
\begin{array}{lcl}
D(p) & = & 0 \\
D(\neg\varphi) & = & D(\varphi) \\
D(\varphi_1 \vee \varphi_2) & = & \max\{D(\varphi_1), D(\varphi_2)\} \\
D(\varphi_1 \mathcal{U}_{[a,b]} \varphi_2) & = & b + \max\{D(\varphi_1), D(\varphi_2)\}
\end{array}
$$

- Syntax-dependent **upper-bound** on the actual depth
  - **Example:** $D(\Box_{[a,b]}\mathsf{T}) = b$

Temporal Depth of an MTL-B *formula*

- Each future MTL-B formula admits a number $D(\varphi)$ indicating its **temporal depth**
  - The satisfaction of φ by a signal ξ from any position t is fully determined within the interval $[t, t + D(\varphi)]$

$$
\begin{array}{lcl}
D(p) & = & 0 \\
D(\neg\varphi) & = & D(\varphi) \\
D(\varphi_1 \vee \varphi_2) & = & \max\{D(\varphi_1), D(\varphi_2)\} \\
D(\varphi_1 \mathcal{U}_{[a,b]} \varphi_2) & = & b + \max\{D(\varphi_1), D(\varphi_2)\}
\end{array}
$$

- Syntax-dependent **upper-bound** on the actual depth
  - **Example:** $D(\Box_{[a,b]} \mathsf{T}) = b$

Pastify Operator

- **Relation between φ and $\psi = \Pi(\varphi, d)$:**

$$(\xi, t) \models \varphi \quad \leftrightarrow \quad (\xi, t+d) \models \psi$$

- **Definition:** The operator Π on future MTL-B formulae φ and a displacement $d \geq D(\varphi)$ is defined recursively as:

$$\begin{array}{lcl} \Pi(p, d) & = & \diamondminus_{d}\, p \\ \Pi(\neg\varphi, d) & = & \neg\Pi(\varphi, d) \\ \Pi(\varphi_1 \vee \varphi_2, d) & = & \Pi(\varphi_1, d) \vee \Pi(\varphi_2, d) \\ \Pi(\varphi_1 \mathcal{U}_{[a,b]} \varphi_2, d) & = & \Pi(\varphi_1, d-b)\mathcal{P}_{[a,b]}\Pi(\varphi_2, d-b) \\ \Pi(\Diamond_{[a,b]}\, \varphi, d) & = & \diamondminus_{[0,b-a]}\, \Pi(\varphi, d-b) \end{array}$$

- **Equisatisfaction of $\Box\,\varphi$ and $\Box\,\psi$:**

$$\xi \models \Box\,\varphi \quad \leftrightarrow \quad \xi \models \Box\,\psi$$

Pastify Operator

- **Relation between φ and $\psi = \Pi(\varphi, d)$:**

$$(\xi, t) \models \varphi \quad \leftrightarrow \quad (\xi, t+d) \models \psi$$

- **Definition:** The operator Π on future MTL-B formulae φ and a displacement $d \geq D(\varphi)$ is defined recursively as:

$$\begin{array}{lcl}
\Pi(p, d) & = & \diamondminus_{d}\, p \\
\Pi(\neg\varphi, d) & = & \neg\Pi(\varphi, d) \\
\Pi(\varphi_1 \vee \varphi_2, d) & = & \Pi(\varphi_1, d) \vee \Pi(\varphi_2, d) \\
\Pi(\varphi_1 \mathcal{U}_{[a,b]} \varphi_2, d) & = & \Pi(\varphi_1, d-b) \mathcal{P}_{[a,b]} \Pi(\varphi_2, d-b) \\
\Pi(\diamond_{[a,b]}\, \varphi, d) & = & \diamondminus_{[0,b-a]}\, \Pi(\varphi, d-b)
\end{array}$$

- **Equisatisfaction of $\Box\,\varphi$ and $\Box\,\psi$:**

$$\xi \models \Box\,\varphi \quad \leftrightarrow \quad \xi \models \Box\,\psi$$

Pastify Operator

- **Relation between φ and $\psi = \Pi(\varphi, d)$:**

$$(\xi, t) \models \varphi \quad \leftrightarrow \quad (\xi, t+d) \models \psi$$

- **Definition:** The operator Π on future MTL-B formulae φ and a displacement $d \geq D(\varphi)$ is defined recursively as:

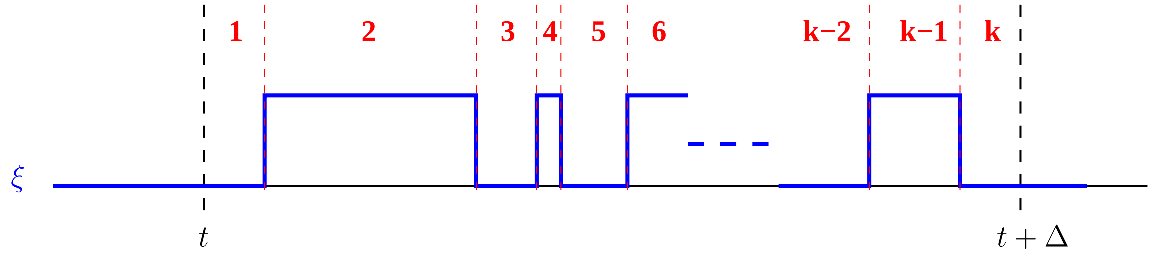
$$\begin{array}{lcl}
\Pi(p, d) & = & \diamondminus_{d}\, p \\
\Pi(\neg\varphi, d) & = & \neg\Pi(\varphi, d) \\
\Pi(\varphi_1 \vee \varphi_2, d) & = & \Pi(\varphi_1, d) \vee \Pi(\varphi_2, d) \\
\Pi(\varphi_1 \mathcal{U}_{[a,b]} \varphi_2, d) & = & \Pi(\varphi_1, d-b) \mathcal{P}_{[a,b]} \Pi(\varphi_2, d-b) \\
\Pi(\Diamond_{[a,b]}\, \varphi, d) & = & \diamondminus_{[0,b-a]}\, \Pi(\varphi, d-b)
\end{array}$$

- **Equisatisfaction of $\Box\,\varphi$ and $\Box\,\psi$:**

$$\xi \models \Box\,\varphi \quad \leftrightarrow \quad \xi \models \Box\,\psi$$

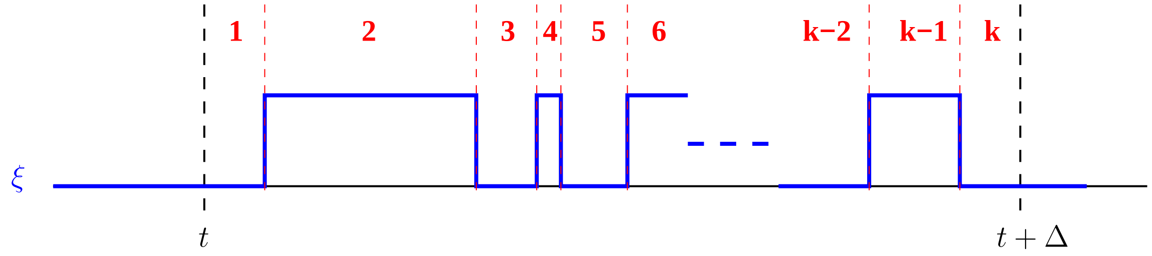
Bounded Variability of Input Signals

- **Definition:**
- A signal ξ is of (Δ, k)-bounded variability if for every interval of the form $[t, t + \Delta]$ the number of changes in the value of ξ is at most k



- The bounded variability is preserved by MTL-B operators

- **Definition:**
- A signal ξ is of (Δ, k)-bounded variability if for every interval of the form $[t, t+\Delta]$ the number of changes in the value of ξ is at most k



- The bounded variability is preserved by MTL-B operators

Temporal testers for MTL-B *formulae*

- Temporal testers for LTL proposed in [KP05]
  - Compositional basis for automata construction corresponding to LTL formulae
  - Extension to real-time temporal logics
    - Past-MITL [MNP05]
    - MITL [MNP06]
- Temporal testers for Past-MITL are **deterministic**
  - Under the bounded variability assumption, deterministic temporal tester construction naturally extends to past MTL-B operators such as $\diamondminus_d$ or $\mathcal{S}_d$
- How to build a deterministic temporal tester for $\mathcal{P}_{[a,b]}$ operator?

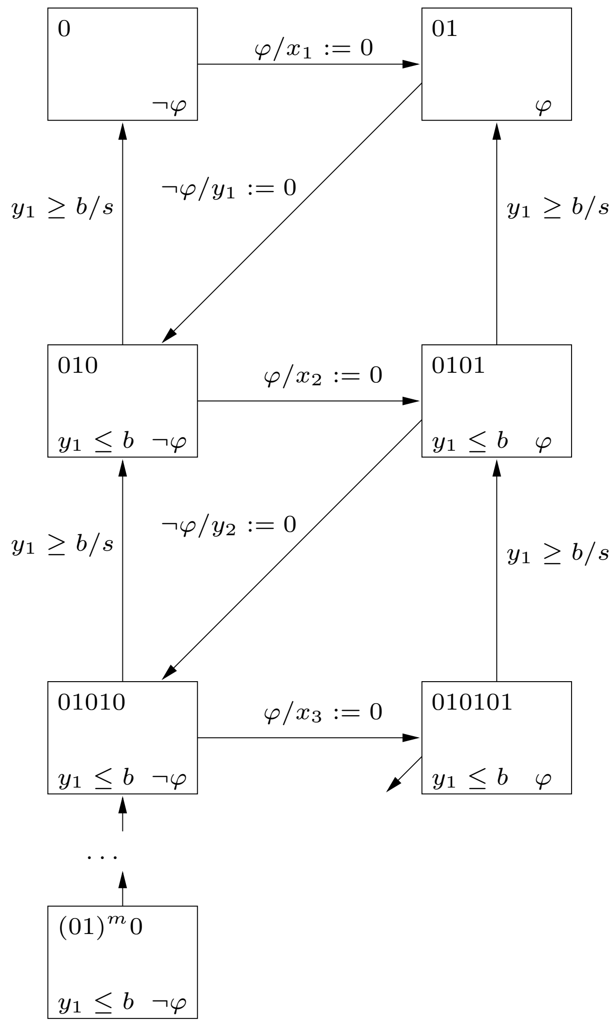
Temporal testers for MTL-B *formulae*

- Temporal testers for LTL proposed in [KP05]
  - Compositional basis for automata construction corresponding to LTL formulae
  - Extension to real-time temporal logics
    - Past-MITL [MNP05]
    - MITL [MNP06]
- Temporal testers for Past-MITL are **deterministic**
  - Under the bounded variability assumption, deterministic temporal tester construction naturally extends to past MTL-B operators such as $\diamondminus_d$ or $\mathcal{S}_d$
- How to build a deterministic temporal tester for $\mathcal{P}_{[a,b]}$ operator?

Temporal testers for MTL-B *formulae*

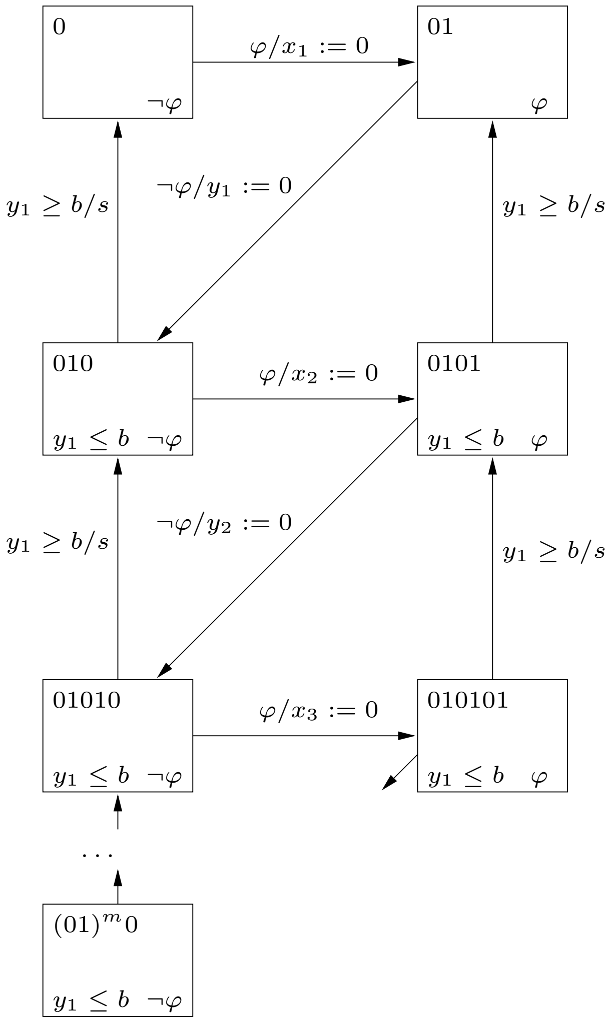
- Temporal testers for LTL proposed in [KP05]
  - Compositional basis for automata construction corresponding to LTL formulae
  - Extension to real-time temporal logics
    - Past-MITL [MNP05]
    - MITL [MNP06]
- Temporal testers for Past-MITL are **deterministic**
  - Under the bounded variability assumption, deterministic temporal tester construction naturally extends to past MTL-B operators such as $\diamondminus_d$ or $\mathcal{S}_d$
- How to build a deterministic temporal tester for $\mathcal{P}_{[a,b]}$ operator?

- **Event recorder** [MNP05]
  - The core of the tester-based translation from Past MITL to timed automata
  - Takes φ as input and $\diamondminus_{[a,b]} \varphi$ as output
  - The automaton outputs 1 whenever $x_1 \geq a$
- Trivial extension for $\diamondminus_{b} \varphi$ with the bounded variability assumption



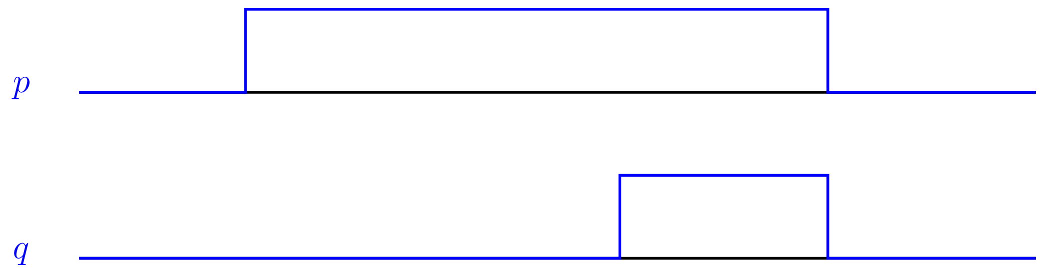
Deterministic Temporal Tester for $\diamondminus_{[a,b]} \varphi$

- **Event recorder** [MNP05]
  - The core of the tester-based translation from Past MITL to timed automata
  - Takes φ as input and $\diamondminus_{[a,b]} \varphi$ as output
  - The automaton outputs 1 whenever $x_1 \geq a$
- Trivial extension for $\diamondminus_b \varphi$ with the bounded variability assumption



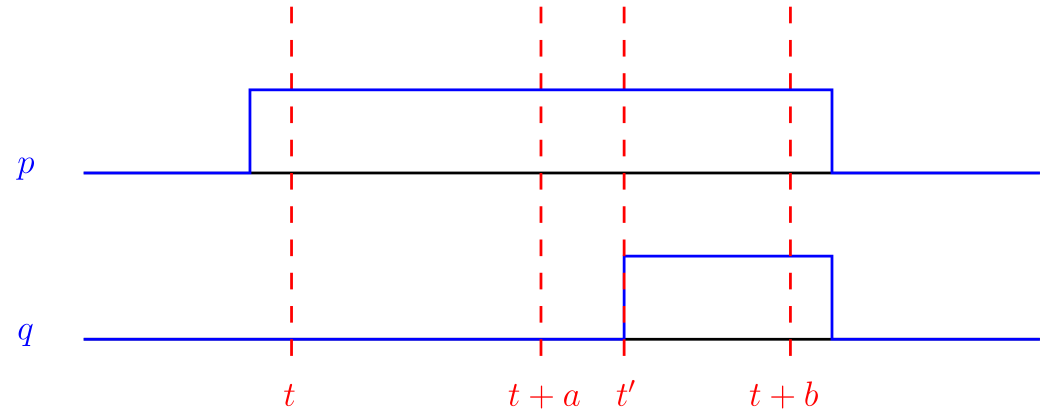
Deterministic Temporal Tester for $\varphi_1 \mathcal{P} \varphi_2$

- **Observation** [MN04]: If p is a signal of $(b, 1)$-bounded variability, then
  - $(\xi, t) \models p\, \mathcal{U}_{[a,b]} q$ iff $(\xi, t) \models p \wedge \Diamond_{[a,b]}(p \wedge q)$
  - $(\xi, t) \models p\, \mathcal{P}_{[a,b]} q$ iff $(\xi, t) \models \diamondminus_b\, p \wedge \diamondminus_{[0,b-a]}(p \wedge q)$



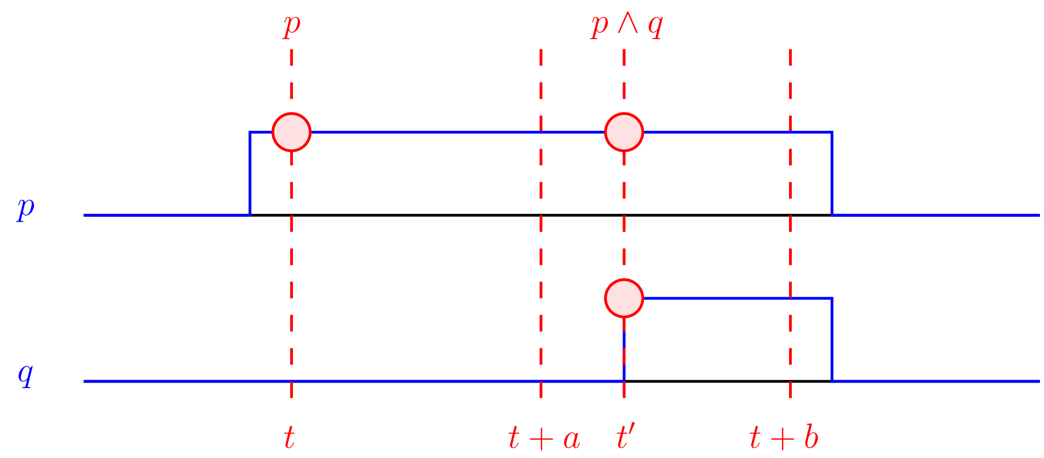
Deterministic Temporal Tester for $\varphi_1 \mathcal{P} \varphi_2$

- **Observation** [MN04]: If p is a signal of $(b, 1)$-bounded variability, then
  - $(\xi, t) \models p\,\mathcal{U}_{[a,b]} q$ iff $(\xi, t) \models p \wedge \Diamond_{[a,b]} (p \wedge q)$
  - $(\xi, t) \models p\,\mathcal{P}_{[a,b]} q$ iff $(\xi, t) \models \ominus\!\!\!\!\Diamond_b\, p \wedge \ominus\!\!\!\!\Diamond_{[0,b-a]} (p \wedge q)$

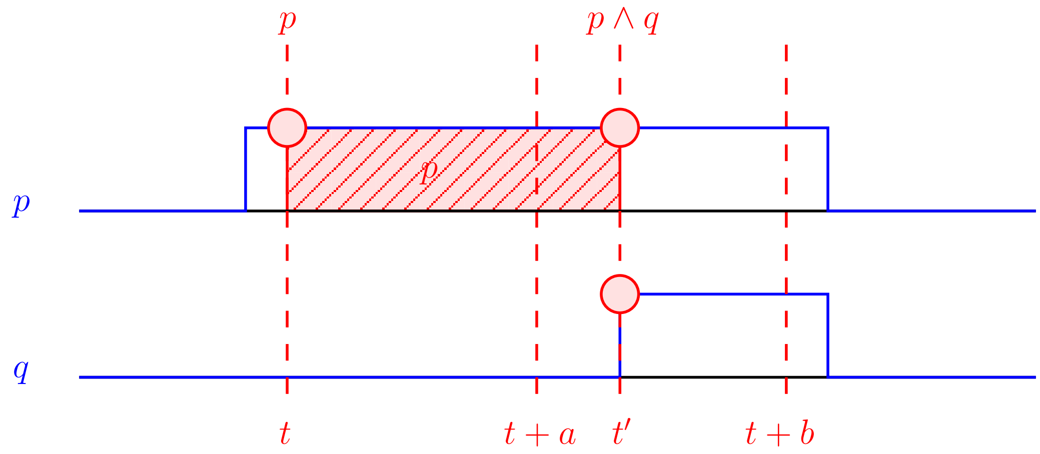


Deterministic Temporal Tester for $\varphi_1 \mathcal{P} \varphi_2$

- **Observation** [MN04]: If p is a signal of $(b, 1)$-bounded variability, then
  - $(\xi, t) \models p\,\mathcal{U}_{[a,b]} q$ iff $(\xi, t) \models p \wedge \diamondsuit_{[a,b]}(p \wedge q)$
  - $(\xi, t) \models p\,\mathcal{P}_{[a,b]} q$ iff $(\xi, t) \models \diamondminus_{b}\, p \wedge \diamondminus_{[0,b-a]}(p \wedge q)$

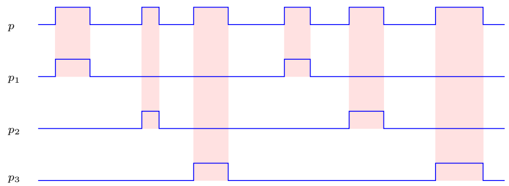


- **Observation** [MN04]: If p is a signal of $(b,1)$-bounded variability, then
  - $(\xi, t) \models p\,\mathcal{U}_{[a,b]} q$ iff $(\xi, t) \models p \wedge \Diamond_{[a,b]}(p \wedge q)$
  - $(\xi, t) \models p\,\mathcal{P}_{[a,b]} q$ iff $(\xi, t) \models \diamondminus_{b}\, p \wedge \diamondminus_{[0,b-a]}(p \wedge q)$



Deterministic Temporal Tester for $\varphi_1 \mathcal{P} \varphi_2$

- Any signal p of (b, k) variability ($k > 1$), can be decomposed into k signals $p_1, p_2, \ldots, p_k$, such that:
  - $p = p_1 \vee p_2 \vee \ldots \vee p_k$
  - $p_i \wedge p_j$ always **false** for every $i \neq j$
  - p_i is of $(b, 1)$-variability



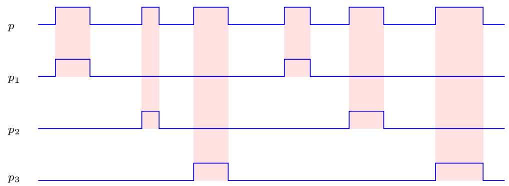
- For such p_i's we have:

$$
\begin{array}{lcl}
(\xi, t) \models p\, \mathcal{U}_{[a,b]} q & \leftrightarrow & (\xi, t) \models \bigvee_{i=1}^{k} p_i\, \mathcal{U}_{[a,b]} q \\
(\xi, t) \models p\, \mathcal{P}_{[a,b]} q & \leftrightarrow & (\xi, t) \models \bigvee_{i=1}^{k} p_i\, \mathcal{P}_{[a,b]} q
\end{array}
$$

- The splitting of p can be achieved trivially using an automaton realizing a **counter modulo** k.

Deterministic Temporal Tester for $\varphi_1 \mathcal{P} \varphi_2$

- Any signal p of (b, k) variability ($k > 1$), can be decomposed into k signals $p_1, p_2, \ldots, p_k$, such that:
  - $p = p_1 \vee p_2 \vee \ldots \vee p_k$
  - $p_i \wedge p_j$ always **false** for every $i \neq j$
  - p_i is of $(b, 1)$-variability



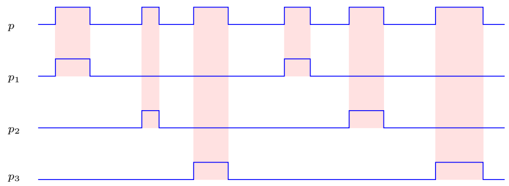
- For such p_i's we have:

$$
\begin{array}{lcl}
(\xi, t) \models p\, \mathcal{U}_{[a,b]} q & \leftrightarrow & (\xi, t) \models \bigvee_{i=1}^{k} p_i\, \mathcal{U}_{[a,b]} q \\
(\xi, t) \models p\, \mathcal{P}_{[a,b]} q & \leftrightarrow & (\xi, t) \models \bigvee_{i=1}^{k} p_i\, \mathcal{P}_{[a,b]} q
\end{array}
$$

- The splitting of p can be achieved trivially using an automaton realizing a **counter modulo** k.

Deterministic Temporal Tester for $\varphi_1 \mathcal{P} \varphi_2$

- Any signal p of (b, k) variability ($k > 1$), can be decomposed into k signals $p_1, p_2, \ldots, p_k$, such that:
  - $p = p_1 \vee p_2 \vee \ldots \vee p_k$
  - $p_i \wedge p_j$ always **false** for every $i \neq j$
  - p_i is of $(b, 1)$-variability



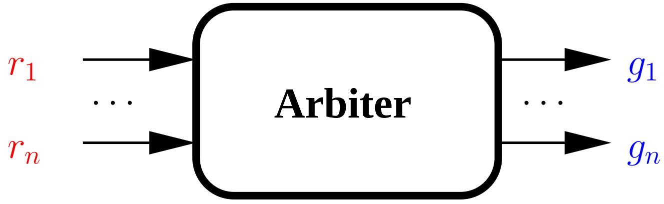
- For such p_i's we have:

$$
\begin{array}{lcl}
(\xi, t) \models p\, \mathcal{U}_{[a,b]} q & \leftrightarrow & (\xi, t) \models \bigvee_{i=1}^{k} p_i\, \mathcal{U}_{[a,b]} q \\
(\xi, t) \models p\, \mathcal{P}_{[a,b]} q & \leftrightarrow & (\xi, t) \models \bigvee_{i=1}^{k} p_i\, \mathcal{P}_{[a,b]} q
\end{array}
$$

- The splitting of p can be achieved trivially using an automaton realizing a **counter modulo** k.

Synthesis of an Arbiter

- Architecture of an arbiter

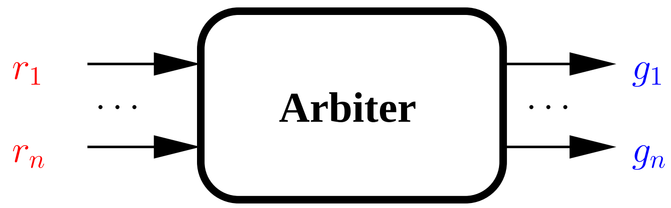


- Typical timed interaction between the arbiter an a client i

- Communication protocol between the arbiter an a client i

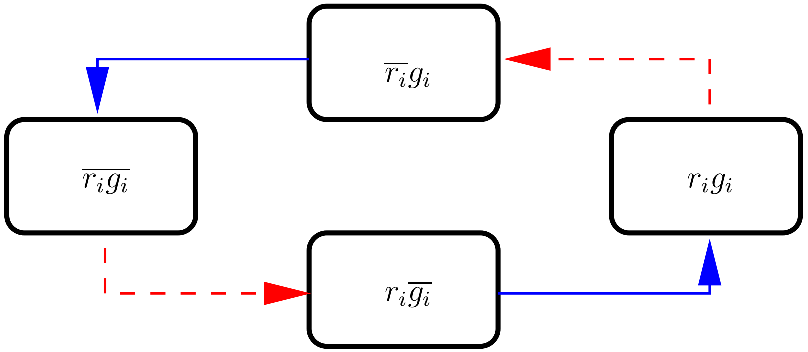
Synthesis of an Arbiter

- Architecture of an arbiter

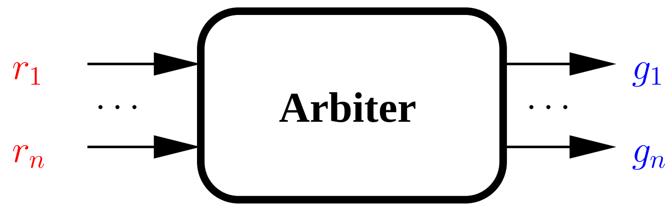


- Typical timed interaction between the arbiter an a client i

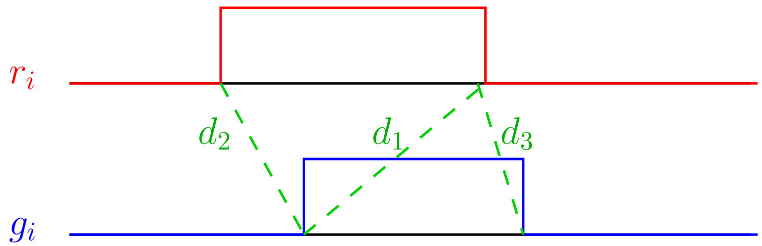
- Communication protocol between the arbiter an a client i



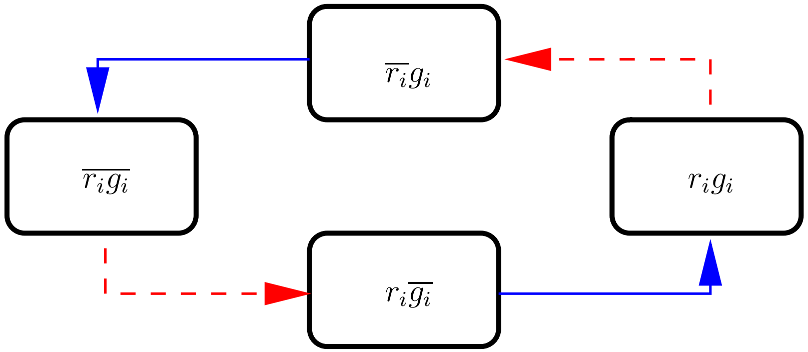
- Architecture of an arbiter



- Typical timed interaction between the arbiter an a client i



- Communication protocol between the arbiter an a client i



Synthesis of an Arbiter: MTL-B *Specification*

- Initial conditions
  - $I^E : \bigwedge_i \overline{r_i}$
  - $I^C : \bigwedge_i \overline{g_i}$
- Safety requirements
  - $S^E : \bigwedge_i r_i \rightarrow r_i \mathcal{S} (\overline{r_i} \wedge \overline{g_i}) \wedge \bigwedge_i (\overline{r_i} \rightarrow \overline{r_i} \mathcal{B}(r_i \wedge g_i)$
  - $S^C : \bigwedge_i (g_i \rightarrow g_i \mathcal{S}(r_i \wedge \overline{g_i})) \wedge \bigwedge_i (\overline{g_i} \rightarrow \overline{g_i} \mathcal{B}(\overline{r_i} \wedge g_i))$
- Bounded liveness requirements
  - $L^E : \bigwedge_i (g_i \rightarrow \Diamond_{[0,d_1]} \overline{r_i})$
  - $L^C : \bigwedge_i (r_i \rightarrow \Diamond_{[0,d_2]} g_i) \wedge \bigwedge_i (\overline{r_i} \rightarrow \Diamond_{[0,d_3]} \overline{g_i})$
- Main formula
  - $(I^E \rightarrow I^C) \wedge \Box(\boxminus(\Pi(S^E) \wedge \Pi(L^E)) \rightarrow (\Pi(S^E) \wedge \Pi(L^C)))$

Synthesis of an Arbiter: MTL-B *Specification*

- Initial conditions
  - $I^E : \bigwedge_i \overline{r_i}$
  - $I^C : \bigwedge_i \overline{g_i}$
- Safety requirements
  - $S^E : \bigwedge_i r_i \rightarrow r_i \mathcal{S} \left(\overline{r_i} \wedge \overline{g_i}\right) \wedge \bigwedge_i (\overline{r_i} \rightarrow \overline{r_i} \mathcal{B}(r_i \wedge g_i)$
  - $S^C : \bigwedge_i (g_i \rightarrow g_i \mathcal{S}(r_i \wedge \overline{g_i})) \wedge \bigwedge_i (\overline{g_i} \rightarrow \overline{g_i} \mathcal{B}(\overline{r_i} \wedge g_i))$
- Bounded liveness requirements
  - $L^E : \bigwedge_i (g_i \rightarrow \Diamond_{[0,d_1]} \overline{r_i})$
  - $L^C : \bigwedge_i (r_i \rightarrow \Diamond_{[0,d_2]} g_i) \wedge \bigwedge_i (\overline{r_i} \rightarrow \Diamond_{[0,d_3]} \overline{g_i})$
- Main formula
  - $(I^E \rightarrow I^C) \wedge \Box(\boxminus(\Pi(S^E) \wedge \Pi(L^E)) \rightarrow (\Pi(S^E) \wedge \Pi(L^C)))$

Synthesis of an Arbiter: MTL-B *Specification*

- Initial conditions
  - $I^E : \bigwedge_i \overline{r_i}$
  - $I^C : \bigwedge_i \overline{g_i}$
- Safety requirements
  - $S^E : \bigwedge_i r_i \rightarrow r_i \mathcal{S} (\overline{r_i} \wedge \overline{g_i}) \wedge \bigwedge_i (\overline{r_i} \rightarrow \overline{r_i} \mathcal{B}(r_i \wedge g_i)$
  - $S^C : \bigwedge_i (g_i \rightarrow g_i \mathcal{S}(r_i \wedge \overline{g_i})) \wedge \bigwedge_i (\overline{g_i} \rightarrow \overline{g_i} \mathcal{B}(\overline{r_i} \wedge g_i))$
- Bounded liveness requirements
  - $L^E : \bigwedge_i (g_i \rightarrow \Diamond_{[0,d_1]} \overline{r_i})$
  - $L^C : \bigwedge_i (r_i \rightarrow \Diamond_{[0,d_2]} g_i) \wedge \bigwedge_i (\overline{r_i} \rightarrow \Diamond_{[0,d_3]} \overline{g_i})$
- Main formula
  - $(I^E \rightarrow I^C) \wedge \Box(\boxminus(\Pi(S^E) \wedge \Pi(L^E)) \rightarrow (\Pi(S^E) \wedge \Pi(L^C)))$

Synthesis of an Arbiter: MTL-B *Specification*

- Initial conditions
  - $I^E : \bigwedge_i \overline{r_i}$
  - $I^C : \bigwedge_i \overline{g_i}$
- Safety requirements
  - $S^E : \bigwedge_i r_i \rightarrow r_i \mathcal{S} (\overline{r_i} \wedge \overline{g_i}) \wedge \bigwedge_i (\overline{r_i} \rightarrow \overline{r_i} \mathcal{B}(r_i \wedge g_i)$
  - $S^C : \bigwedge_i (g_i \rightarrow g_i \mathcal{S}(r_i \wedge \overline{g_i})) \wedge \bigwedge_i (\overline{g_i} \rightarrow \overline{g_i} \mathcal{B}(\overline{r_i} \wedge g_i))$
- Bounded liveness requirements
  - $L^E : \bigwedge_i (g_i \rightarrow \Diamond_{[0,d_1]} \overline{r_i})$
  - $L^C : \bigwedge_i (r_i \rightarrow \Diamond_{[0,d_2]} g_i) \wedge \bigwedge_i (\overline{r_i} \rightarrow \Diamond_{[0,d_3]} \overline{g_i})$
- Main formula
  - $(I^E \rightarrow I^C) \wedge \Box(\boxminus(\Pi(S^E) \wedge \Pi(L^E)) \rightarrow (\Pi(S^E) \wedge \Pi(L^C)))$

Synthesis of an Arbiter: MTL-B *Specification*

- Initial conditions
  - $I^E : \bigwedge_i \overline{r_i}$
  - $I^C : \bigwedge_i \overline{g_i}$
- Safety requirements
  - $S^E : \bigwedge_i r_i \rightarrow r_i \mathcal{S}\,(\overline{r_i} \wedge \overline{g_i}) \wedge \bigwedge_i(\overline{r_i} \rightarrow \overline{r_i}\mathcal{B}(r_i \wedge g_i)$
  - $S^C : \bigwedge_i(g_i \rightarrow g_i\mathcal{S}(r_i \wedge \overline{g_i})) \wedge \bigwedge_i(\overline{g_i} \rightarrow \overline{g_i}\mathcal{B}(\overline{r_i} \wedge g_i))$
- Bounded liveness requirements
  - $L^E : \bigwedge_i(g_i \rightarrow \Diamond_{[0,d_1]}\,\overline{r_i})$
  - $L^C : \bigwedge_i(r_i \rightarrow \Diamond_{[0,d_2]}\,g_i) \wedge \bigwedge_i(\overline{r_i} \rightarrow \Diamond_{[0,d_3]}\,\overline{g_i})$
- Main formula
  - $(I^E \rightarrow I^C) \wedge \Box(\boxminus(\Pi(S^E) \wedge \Pi(L^E)) \rightarrow (\Pi(S^E) \wedge \Pi(L^C)))$

Synthesis of an Arbiter: Experimental Results

- Discrete time synthesis
- $d_3 = 1$

N	d_1	d_2	Size	Time	d_1	d_2	Size	Time	d_1	d_2	Size	Time
2	2	4	466	0.00	3	5	654	0.01	4	6	946	0.02
3	2	8	1382	0.14	3	10	2432	0.34	4	12	4166	0.51
4	2	12	4323	0.63	3	15	7402	1.12	4	18	16469	2.33
5	2	16	13505	1.93	3	20	26801	4.77	4	24	50674	10.50
6	2	20	43366	8.16	3	25	84027	22.55	4	30	168944	64.38
7	2	24	138937	44.38	3	30	297524	204.56	4	36	700126	1897.56

- Exponential growth of BDD nodes in N and d_2
  - Expected using discrete time

Synthesis of an Arbiter: Experimental Results

- Discrete time synthesis
- $d_3 = 1$

N	d_1	d_2	Size	Time	d_1	d_2	Size	Time	d_1	d_2	Size	Time
2	2	4	466	0.00	3	5	654	0.01	4	6	946	0.02
3	2	8	1382	0.14	3	10	2432	0.34	4	12	4166	0.51
4	2	12	4323	0.63	3	15	7402	1.12	4	18	16469	2.33
5	2	16	13505	1.93	3	20	26801	4.77	4	24	50674	10.50
6	2	20	43366	8.16	3	25	84027	22.55	4	30	168944	64.38
7	2	24	138937	44.38	3	30	297524	204.56	4	36	700126	1897.56

- Exponential growth of BDD nodes in N and d_2
  - Expected using discrete time

Conclusion

- Complete chain that allows to synthesize controllers automatically from real-time bounded-response temporal specifications
  - Bounded-response temporal property $\longrightarrow$ deterministic timed automaton
    - Pastification of MTL-B formulae
    - Bounded-variability assumption
- Future work
  - Focus on efficient symbolic algorithms in the spirit of [CDF+05]
  - Apply the synthesis algorithm to more complex specifications of real-time scheduling problems

Conclusion

- Complete chain that allows to synthesize controllers automatically from real-time bounded-response temporal specifications
  - Bounded-response temporal property $\longrightarrow$ deterministic timed automaton
    - Pastification of MTL-B formulae
    - Bounded-variability assumption
- Future work
  - Focus on efficient symbolic algorithms in the spirit of [CDF$^+$05]
  - Apply the synthesis algorithm to more complex specifications of real-time scheduling problems

Conclusion

- Complete chain that allows to synthesize controllers automatically from real-time bounded-response temporal specifications
  - Bounded-response temporal property $\rightarrow$ deterministic timed automaton
    - Pastification of MTL-B formulae
    - Bounded-variability assumption
- Future work
  - Focus on efficient symbolic algorithms in the spirit of [CDF+05]
  - Apply the synthesis algorithm to more complex specifications of real-time scheduling problems

References

[AFH96] R. Alur, T. Feder, and T.A. Henzinger, The Benefits of Relaxing Punctuality, *Journal of the ACM* **43**, 116–146, 1996 (first published in *PODC'91*).

[AMP95] E. Asarin, O. Maler and A. Pnueli, Symbolic Controller Synthesis for Discrete and Timed Systems, *Hybrid Systems II*, 1–20, LNCS 999, 1995.

[BL69] J.R. Büchi and L.H. Landweber, Solving Sequential Conditions by Finite-state Operators, *Trans. of the AMS* **138**, 295–311, 1969.

[CDF$^+$05] F. Cassez, A. David, E. Fleury, K.G. Larsen and D. Lime, Efficient On-the-Fly Algorithms for the Analysis of Timed Games, *CONCUR'05*, 66–80, 2005.

[Chu63] A. Church, Logic, Arithmetic and Automata, in *Proc. of the Int. Cong. of Mathematicians 1962*, 23–35, 1963.

[KP05] Y. Kesten and A. Pnueli, A Compositional Approach to CTL* Verification, *Theoretical Computer Science* **331**, 397–428, 2005.

[Koy90] R. Koymans, Specifying Real-time Properties with Metric Temporal Logic, *Real-time Systems* **2**, 255–299, 1990.

References

[MN04] O. Maler and D. Nickovic, Monitoring Temporal Properties of Continuous Signals, *FORMATS/FTRTFT'04*, 152–166, LNCS 3253, 2004.

[MNP05] O. Maler, D. Nickovic and A. Pnueli, Real Time Temporal Logic: Past, Present, Future, *FORMATS'05*, 2–16, LNCS 3829, 2005.

[MNP06] O. Maler, D. Nickovic and A. Pnueli, From MITL to Timed Automata, *FORMATS'06*, 274–289, LNCS 4202, 2006.

[MPS95] O. Maler, A. Pnueli and J. Sifakis, On the Synthesis of Discrete Controllers for Timed Systems, *STACS'95*, 229–242, LNCS 900, 1995.

[PPS06] N. Piterman, A. Pnueli and Y. Sa'ar, Synthesis of Reactive(1) Designs, *VMCAI'06*, 364–380, 2006.

[PP06] N. Piterman and A. Pnueli, Faster Solutions of Rabin and Streett Games, *LICS'06*, 275–284, 2006.

[RW89] P.J. Ramadge and W.M. Wonham, The Control of Discrete Event Systems, *Proc. of the IEEE* **77**, 81–98, 1989.