Approximating Optimal Bounds in Prompt-LTL Realizability in Doubly-exponential Time

Joint work with Leander Tentrup and Martin Zimmermann

Alexander Weinert

Saarland University

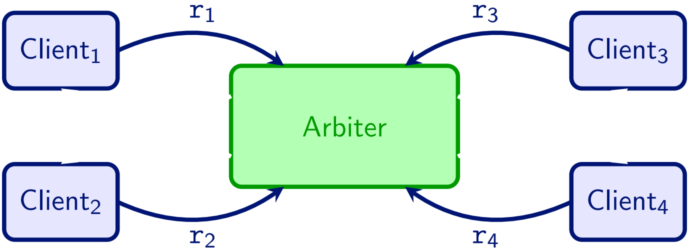
September, 16th 2016
GandALF '16

Realizability: a Toy Example

- Setting: an arbiter with 4 clients
-
-

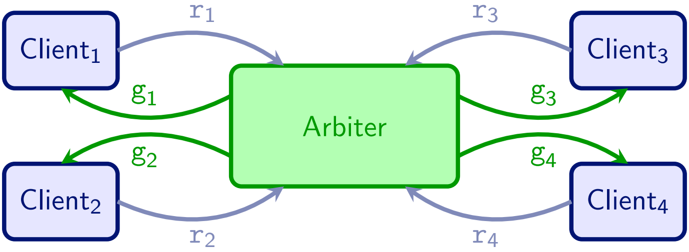
Realizability: a Toy Example

- Setting: an arbiter with 4 clients
- Requests $\mathtt{r}_i$ from client i (controlled by the environment)
-



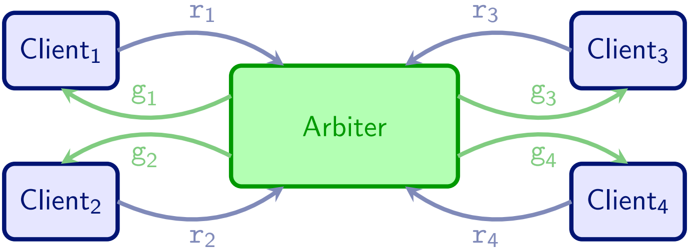
Realizability: a Toy Example

- Setting: an arbiter with 4 clients
- Requests $\mathtt{r}_i$ from client i (controlled by the environment)
- Grants $\mathtt{g}_i$ for client i (controlled by the system)



Realizability: a Toy Example

- Setting: an arbiter with 4 clients
- Requests $\mathtt{r}_i$ from client i (controlled by the environment)
- Grants $\mathtt{g}_i$ for client i (controlled by the system)



Goal: Formal specification of arbiter's behavior

Linear Temporal Logic

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F}\,\varphi$$

where p ranges over a finite set P of atomic propositions.

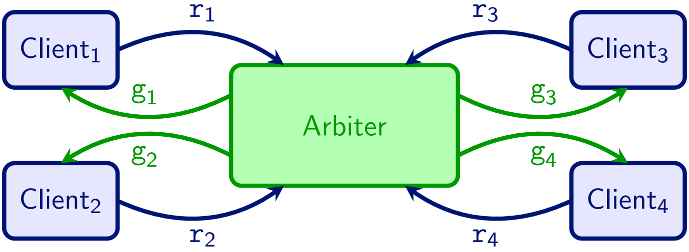
Linear Temporal Logic

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F}\,\varphi$$

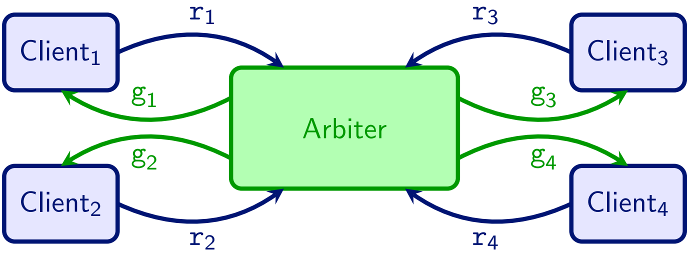
+ typical shorthands

where p ranges over a finite set P of atomic propositions.

Continuing the Example: Specification



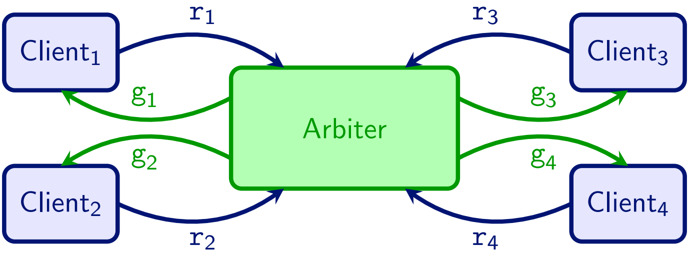
Continuing the Example: Specification



Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Continuing the Example: Specification



Specification:

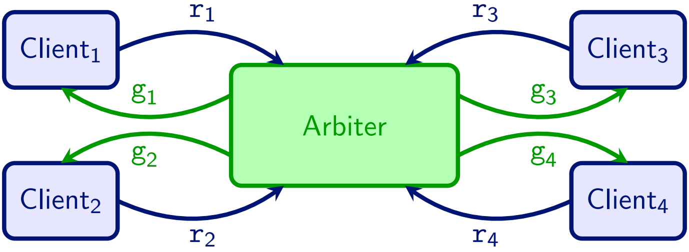
$$\bigwedge_{i=1}^{4} \mathbf{G}\,(r_i \rightarrow \mathbf{F}\, g_i)$$

Admissible execution:

Env:

Sys:

Continuing the Example: Specification



Specification:

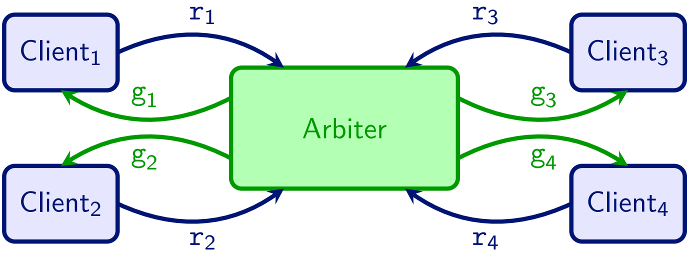
$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env: $\mathtt{r}_1$

Sys:

Continuing the Example: Specification



Specification:

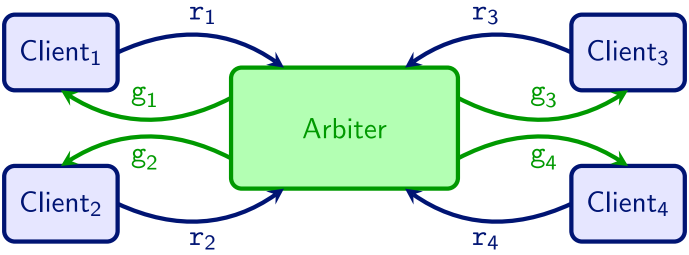
$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env: $\mathtt{r}_1$

Sys: $\mathtt{g}_1$

Continuing the Example: Specification



Specification:

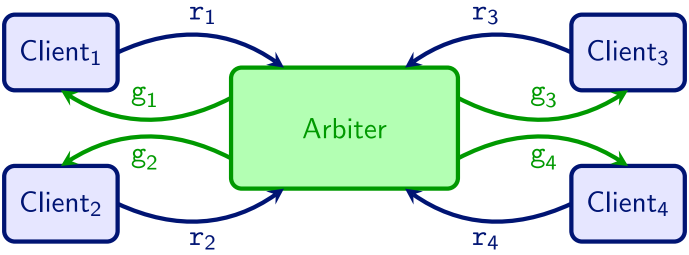
$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env: $\mathtt{r}_1$ $\mathtt{r}_1$

Sys: $\mathtt{g}_1$

Continuing the Example: Specification



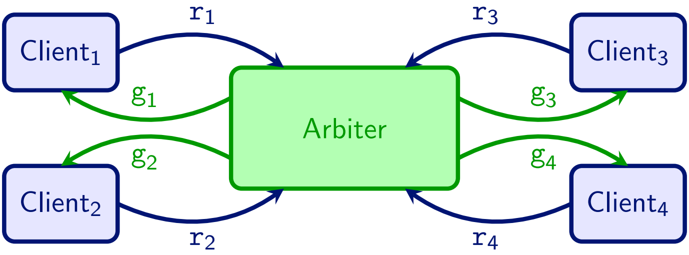
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	$-$

Continuing the Example: Specification



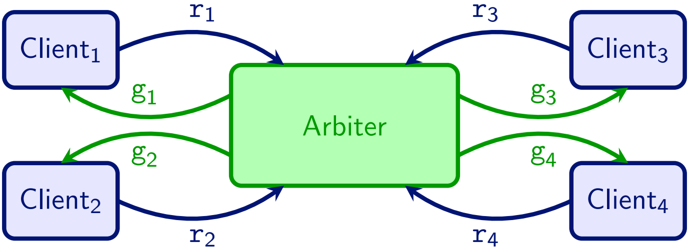
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	$-$
Sys:	$\mathtt{g}_1$	$-$	

Continuing the Example: Specification



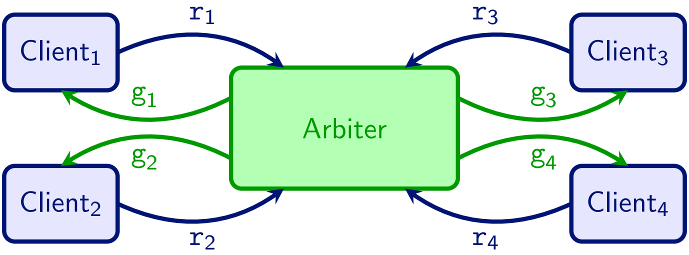
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	–
Sys:	$\mathtt{g}_1$	–	$\mathtt{g}_1$

Continuing the Example: Specification



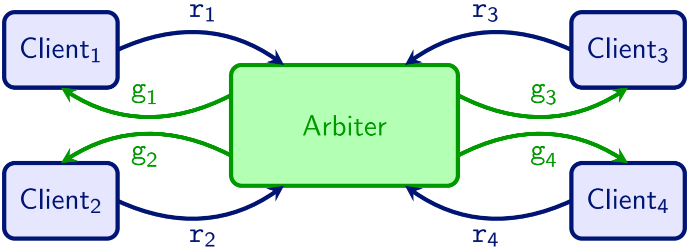
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	–	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	–	$\mathtt{g}_1$	

Continuing the Example: Specification



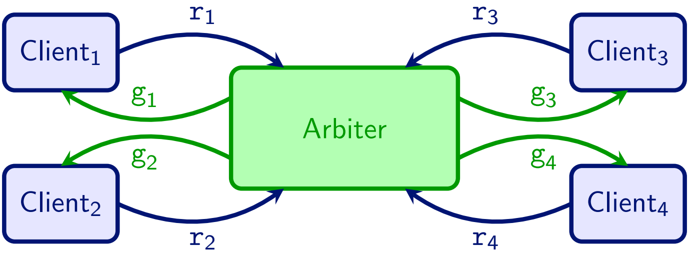
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	–	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	–	$\mathtt{g}_1$	–

Continuing the Example: Specification



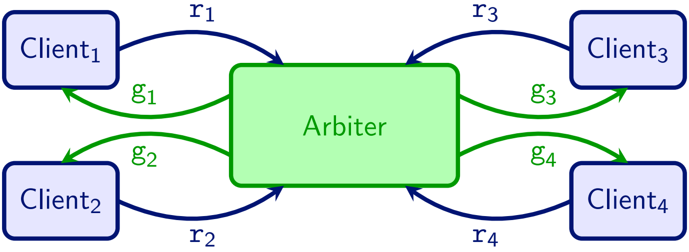
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(r_i \rightarrow \mathbf{F}\, g_i)$$

Admissible execution:

Env:	r_1	r_1	—	r_1	—
Sys:	g_1	—	g_1	—	—

Continuing the Example: Specification



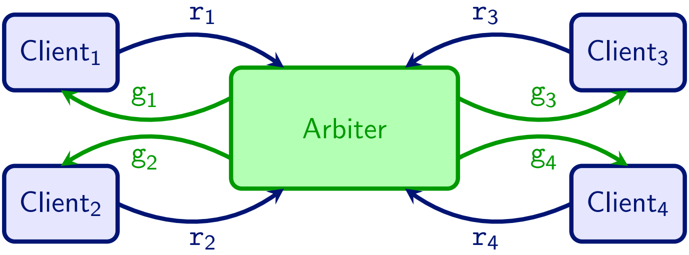
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	—	$\mathtt{r}_1$	—	—
Sys:	$\mathtt{g}_1$	—	$\mathtt{g}_1$	—	—	$\mathtt{g}_1$

Continuing the Example: Specification



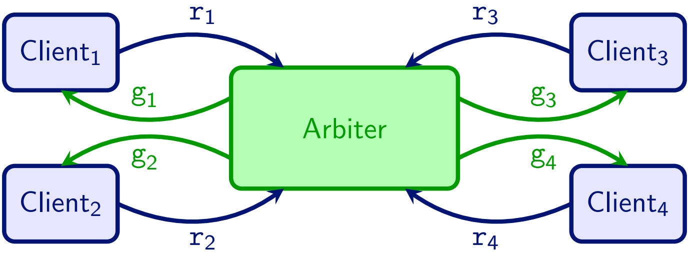
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	—	$\mathtt{r}_1$	—	—	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	—	$\mathtt{g}_1$	—	—	$\mathtt{g}_1$	

Continuing the Example: Specification



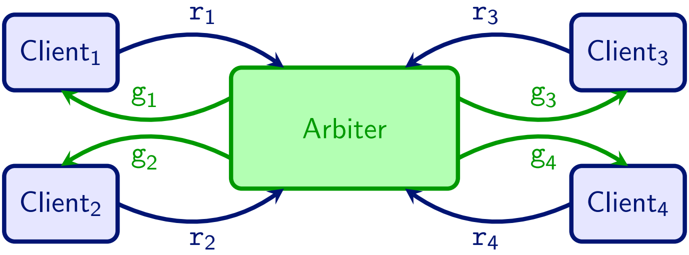
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	—	$\mathtt{r}_1$	—	—	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	—	$\mathtt{g}_1$	—	—	$\mathtt{g}_1$	—

Continuing the Example: Specification



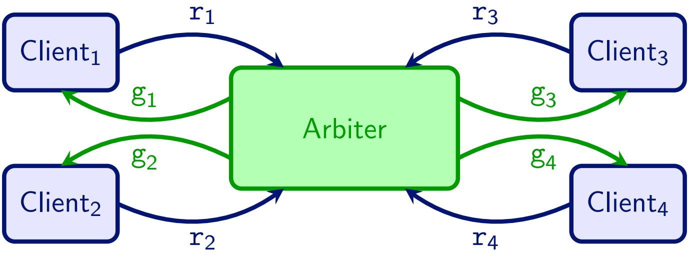
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(r_i \rightarrow \mathbf{F}\, g_i)$$

Admissible execution:

Env:	r_1	r_1	—	r_1	—	—	r_1	—	—
Sys:	g_1	—	g_1	—	—	g_1	—	—	—

Continuing the Example: Specification



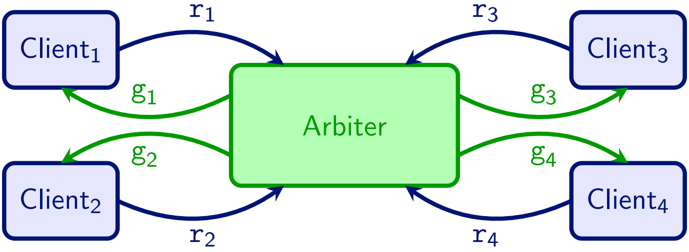
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	—	$\mathtt{r}_1$	—	—	$\mathtt{r}_1$	—	—	—
Sys:	$\mathtt{g}_1$	—	$\mathtt{g}_1$	—	—	$\mathtt{g}_1$	—	—	—	$\mathtt{g}_1$

Continuing the Example: Specification



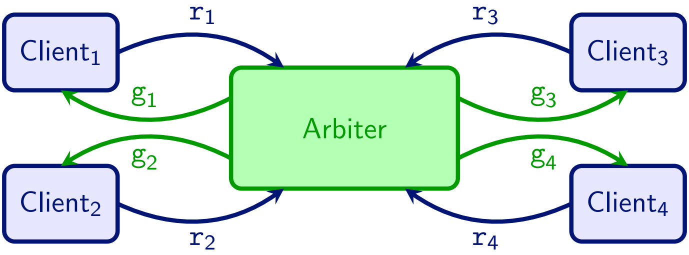
Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Admissible execution:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	–	$\mathtt{r}_1$	–	–	$\mathtt{r}_1$	–	–	–	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	–	$\mathtt{g}_1$	–	–	$\mathtt{g}_1$	–	–	–	$\mathtt{g}_1$	

Continuing the Example: Specification



Specification:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(r_i \rightarrow \mathbf{F}\, g_i)$$

Admissible execution:

Env:	r_1	r_1	—	r_1	—	—	r_1	—	—	—	r_1
Sys:	g_1	—	g_1	—	—	g_1	—	—	—	g_1	...

Prompt-LTL

Problem: $\mathbf{F}\,\varphi$ does not guarantee when φ holds true.

Prompt-LTL

Problem: $\mathbf{F}\,\varphi$ does not guarantee when φ holds true.

Solution: Add prompt-eventually operator $\mathbf{F_P}$:

Prompt-LTL

Problem: $\mathbf{F}\,\varphi$ does not guarantee when φ holds true.

Solution: Add prompt-eventually operator $\mathbf{F_P}$:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F}\,\varphi$$

Prompt-LTL

Problem: $\mathbf{F}\,\varphi$ does not guarantee when φ holds true.

Solution: Add prompt-eventually operator $\mathbf{F_P}$:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F}\,\varphi \mid \mathbf{F_P}\,\varphi$$

Prompt-LTL

Problem: $\mathbf{F}\,\varphi$ does not guarantee when φ holds true.
Solution: Add prompt-eventually operator $\mathbf{F_P}$:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F}\,\varphi \mid \mathbf{F_P}\,\varphi$$

Semantics: Given some word α

Prompt-LTL

Problem: $\mathbf{F}\,\varphi$ does not guarantee when φ holds true.
Solution: Add prompt-eventually operator $\mathbf{F_P}$:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F}\,\varphi \mid \mathbf{F_P}\,\varphi$$

Semantics: Given some word α, $k \in \mathbb{N}$

Prompt-LTL

Problem: $\mathbf{F}\,\varphi$ does not guarantee when φ holds true.
Solution: Add prompt-eventually operator $\mathbf{F_P}$:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F}\,\varphi \mid \mathbf{F_P}\,\varphi$$

Semantics: Given some word α, $k \in \mathbb{N}$

$(\alpha, k) \models \mathbf{F_P}\,\varphi$ if, and only if,

φ holds true within at most k steps

Prompt-LTL Example

Before:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Execution α**:**

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	–	$\mathtt{r}_1$	–	–	$\mathtt{r}_1$	–	–	–	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	–	$\mathtt{g}_1$	–	–	$\mathtt{g}_1$	–	–	–	$\mathtt{g}_1$	$\cdots$

Prompt-LTL Example

Before:

$$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$$

Execution α:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	–	$\mathtt{r}_1$	–	–	$\mathtt{r}_1$	–	–	–	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	–	$\mathtt{g}_1$	–	–	$\mathtt{g}_1$	–	–	–	$\mathtt{g}_1$	$\cdots$

Prompt-LTL Example

Before:

$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$

Execution α:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	–	$\mathtt{r}_1$	–	–	$\mathtt{r}_1$	–	–	–	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	–	$\mathtt{g}_1$	–	–	$\mathtt{g}_1$	–	–	–	$\mathtt{g}_1$	$\cdots$

$\alpha \models \bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$

Prompt-LTL Example

Before:

$\bigwedge_{i=1}^{4} \mathbf{G}\,(r_i \rightarrow \mathbf{F}\, g_i)$

Now:

$\bigwedge_{i=1}^{4} \mathbf{G}\,(r_i \rightarrow \mathbf{F_P}\, g_i)$

Execution α:

Env:	r_1	r_1	–	r_1	–	–	r_1	–	–	–	r_1
Sys:	g_1	–	g_1	–	–	g_1	–	–	–	g_1	…

$\alpha \models \bigwedge_{i=1}^{4} \mathbf{G}\,(r_i \rightarrow \mathbf{F}\, g_i)$

Prompt-LTL Example

Before:

$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$

Now:

$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F_P}\,\mathtt{g}_i)$

Execution α:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	–	$\mathtt{r}_1$	–	–	$\mathtt{r}_1$	–	–	–	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	–	$\mathtt{g}_1$	–	–	$\mathtt{g}_1$	–	–	–	$\mathtt{g}_1$	…
	0	1		2			3				

$\alpha \models \bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$

Prompt-LTL Example

Before:

$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$

Now:

$\bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F_P}\,\mathtt{g}_i)$

Execution α:

Env:	$\mathtt{r}_1$	$\mathtt{r}_1$	—	$\mathtt{r}_1$	—	—	$\mathtt{r}_1$	—	—	—	$\mathtt{r}_1$
Sys:	$\mathtt{g}_1$	—	$\mathtt{g}_1$	—	—	$\mathtt{g}_1$	—	—	—	$\mathtt{g}_1$	…
	0	1		2			3				

$\alpha \models \bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$

There exists no k such that

$(\alpha, k) \models \bigwedge_{i=1}^{4} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F_P}\,\mathtt{g}_i)$

Prompt-LTL Realizability

Theorem (Kupferman, Piterman, Vardi '07)

The following problem is $2\textsc{Exptime}$*-complete:*

Input: *Prompt-LTL formula* φ *over* $I \cup O$

Question: *Does there exist a strategy* $\sigma\colon (2^I)^+ \to 2^O$ *and a bound* k*, such that every word consistent with* σ *models* φ *w.r.t.* k*?*

Prompt-LTL Realizability

Theorem (Kupferman, Piterman, Vardi '07)

The following problem is $2\textsc{Exptime}$*-complete:*

Input: *Prompt-LTL formula* φ *over* $I \cup O$

Question: *Does there exist a strategy* $\sigma \colon (2^I)^+ \to 2^O$
and a bound k*, such that*
every word consistent with σ *models* φ *w.r.t.* k*?*

Now: Prompt-LTL realizability as optimization problem

Prompt-LTL Realizability

Theorem (Kupferman, Piterman, Vardi '07)

The following problem is 2EXPTIME*-complete:*

Input: *Prompt-LTL formula φ over $I \cup O$*

Question: *Does there exist a strategy $\sigma\colon (2^I)^+ \rightarrow 2^O$ and a bound k, such that every word consistent with σ models φ w.r.t. k?*

Now: Prompt-LTL realizability as optimization problem

Theorem (Z. '11)

The minimal k such that there exists a strategy $\sigma\colon (2^I)^+ \rightarrow 2^O$ such that every word consistent with σ models φ w.r.t. k can be determined in triply-exponential time.

Prompt-LTL Realizability

Theorem

The minimal k as defined previously can be approximated within a factor of 2 in doubly-exponential time.

Prompt-LTL Approximation

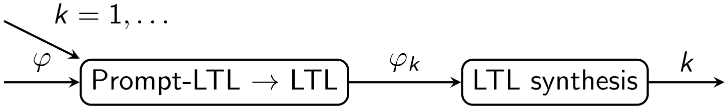
Prompt-LTL Approximation

Theorem (Kupferman, Piterman, Vardi '07)

For every Prompt-LTL formula φ *and each bound* $k \in \mathbb{N}$*,*
there exists an LTL formula φ_k*, such that*

- *if* φ_k *is realizable, then* $(\varphi, 2k)$ *is realizable, and*
- *if* (φ, k) *is realizable, then* φ_k *is realizable*

Prompt-LTL Approximation



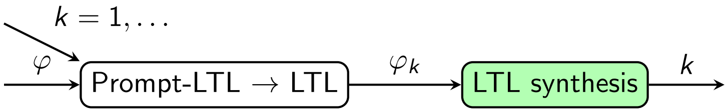
(Kupferman, Piterman, Vardi '07)

Theorem (Kupferman, Piterman, Vardi '07)

For every Prompt-LTL formula φ and each bound $k \in \mathbb{N}$, there exists an LTL formula φ_k, such that

- *if φ_k is realizable, then $(\varphi, 2k)$ is realizable, and*
- *if (φ, k) is realizable, then φ_k is realizable*

Prompt-LTL Approximation



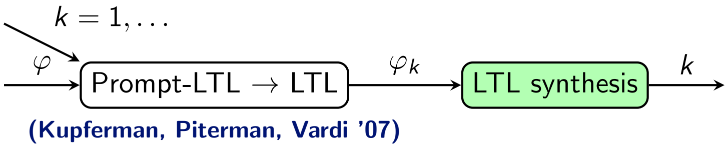
(Kupferman, Piterman, Vardi '07)

Theorem (Kupferman, Piterman, Vardi '07)

For every Prompt-LTL formula φ and each bound $k \in \mathbb{N}$, there exists an LTL formula φ_k, such that

- *if φ_k is realizable, then $(\varphi, 2k)$ is realizable, and*
- *if (φ, k) is realizable, then φ_k is realizable*

Prompt-LTL Approximation



Theorem (Kupferman, Piterman, Vardi '07)

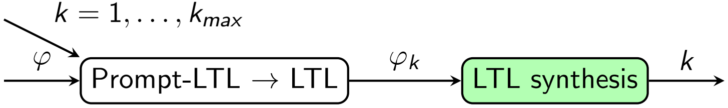
For every Prompt-LTL formula φ and each bound $k \in \mathbb{N}$, there exists an LTL formula φ_k, such that

- *if φ_k is realizable, then $(\varphi, 2k)$ is realizable, and*
- *if (φ, k) is realizable, then φ_k is realizable*

Theorem (Kupferman, Piterman, Vardi '07)

If (φ, k) is realizable for some $k \in \mathbb{N}$, then (φ, k') is realizable for some k' doubly exponential in $|\varphi|$.

Prompt-LTL Approximation



(Kupferman, Piterman, Vardi '07)

Theorem (Kupferman, Piterman, Vardi '07)

For every Prompt-LTL formula φ and each bound $k \in \mathbb{N}$, there exists an LTL formula φ_k, such that

- *if φ_k is realizable, then $(\varphi, 2k)$ is realizable, and*
- *if (φ, k) is realizable, then φ_k is realizable*

Theorem (Kupferman, Piterman, Vardi '07)

If (φ, k) is realizable for some $k \in \mathbb{N}$, then (φ, k') is realizable for some k' doubly exponential in $|\varphi|$.

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.
Idea: Use fresh proposition $r \notin P$, "color" α.

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.
Idea: Use fresh proposition $r \notin P$, "color" α.

$$\alpha = \alpha_0 \quad \alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad \alpha_5 \quad \alpha_6 \quad \alpha_7 \quad \alpha_8 \quad \alpha_9$$

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.
Idea: Use fresh proposition $r \notin P$, "color" α.

$\alpha =$	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9
	r	r	$\neg r$	$\neg r$	r	r	$\neg r$	$\neg r$	r	r

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.
Idea: Use fresh proposition $r \notin P$, "color" α.

$\alpha =$	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9
	r	r	$\neg r$	$\neg r$	r	r	$\neg r$	$\neg r$	r	r

1. Replace each $\mathbf{F_P}\,\psi$ by LTL formula $\mathrm{rel}(\mathbf{F_P}\,\psi)$ stating
   - "φ holds within one color change"

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.
Idea: Use fresh proposition $r \notin P$, "color" α.

$\alpha =$	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9
	r	r	$\neg r$	$\neg r$	r	r	$\neg r$	$\neg r$	r	r

1. Replace each $\mathbf{F_P}\,\psi$ by LTL formula $\mathrm{rel}(\mathbf{F_P}\,\psi)$ stating
   - "φ holds within one color change"
2. Add ψ_k stating
   - "The coloring changes after at most k steps"

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.
Idea: Use fresh proposition $r \notin P$, "color" α.

$\alpha =$	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9
	r	r	$\neg r$	$\neg r$	r	r	$\neg r$	$\neg r$	r	r

1. Replace each $\mathbf{F_P}\,\psi$ by LTL formula $\mathrm{rel}(\mathbf{F_P}\,\psi)$ stating
   - "φ holds within one color change"
2. Add ψ_k stating
   - "The coloring changes after at most k steps"

$$\varphi \rightsquigarrow \mathrm{rel}(\varphi) \land \psi_k$$

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.
Idea: Use fresh proposition $r \notin P$, "color" α.

$\alpha =$	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9
	r	r	$\neg r$	$\neg r$	r	r	$\neg r$	$\neg r$	r	r

1. Replace each $\mathbf{F_P}\,\psi$ by LTL formula $\mathrm{rel}(\mathbf{F_P}\,\psi)$ stating
   - "φ holds within one color change"
2. Add ψ_k stating
   - "The coloring changes after at most k steps"

(Prompt-LTL) → $\varphi \rightsquigarrow \mathrm{rel}(\varphi) \wedge \psi_k$ ← (LTL)

Construction of φ_k: Alternating Color

Given: Prompt-LTL formula φ, bound $k \in \mathbb{N}$.
Wanted: LTL formula φ_k.
Idea: Use fresh proposition $r \notin P$, "color" α.

$\alpha =$	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9
	r	r	$\neg r$	$\neg r$	r	r	$\neg r$	$\neg r$	r	r

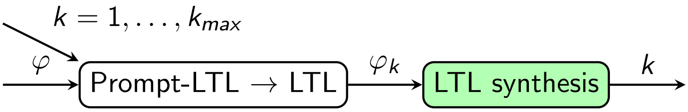
1. Replace each $\mathbf{F_P}\,\psi$ by LTL formula $\mathrm{rel}(\mathbf{F_P}\,\psi)$ stating
   - "φ holds within one color change"
2. Add ψ_k stating
   - "The coloring changes after at most k steps"

(Prompt-LTL) (LTL)

$$\varphi \rightsquigarrow \mathrm{rel}(\varphi) \wedge \psi_k$$

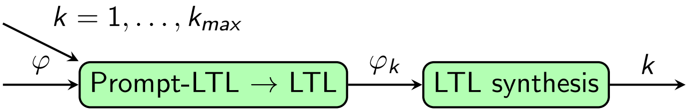
Correctness due to **(Kupferman, Piterman, Vardi '07)**

The Algorithm



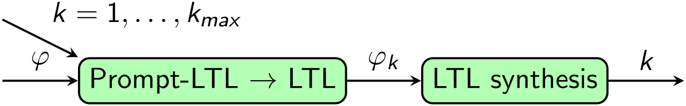
(Kupferman, Piterman, Vardi '07)

The Algorithm



(Kupferman, Piterman, Vardi '07)

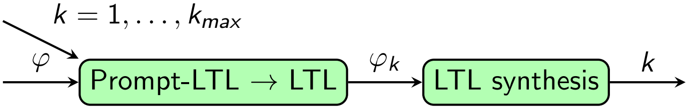
The Algorithm



(Kupferman, Piterman, Vardi '07)

if φ unrealizable **then**
    return "φ unrealizable"
for $k = 0, 1, 2, \ldots, 2^{2^{|\varphi|}}$ **do**
    if $\mathrm{rel}(\varphi) \wedge \psi_k$ realizable **then**
        return $2k$

The Algorithm



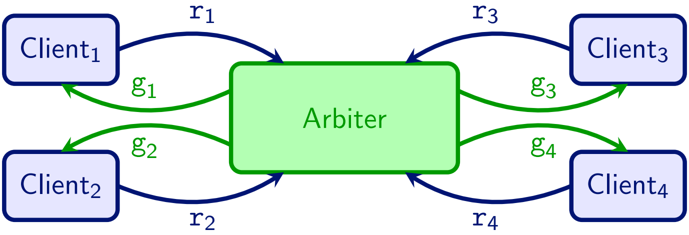
(Kupferman, Piterman, Vardi '07)

1: **if** φ unrealizable **then**
2: **return** "φ unrealizable"
3: **for** $k = 0, 1, 2, \ldots, 2^{2^{|\varphi|}}$ **do**
4: **if** $\mathrm{rel}(\varphi) \wedge \psi_k$ realizable **then**
5: **return** $2k$

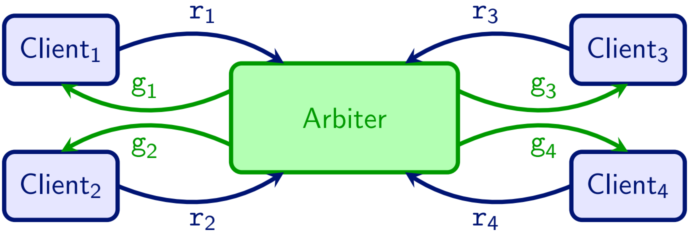
Run-time: doubly-exponential in $|\varphi|$:

- Lines 1 and 4: doubly-exponential time.
- At most doubly-exponentially many iterations.

Back to the Example



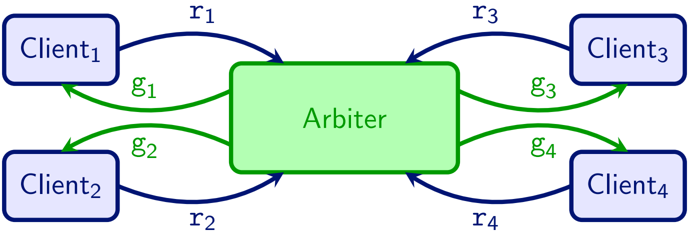
Back to the Example



Parameters:

- Number of clients: r
- Number of prioritized clients: r_p

Back to the Example

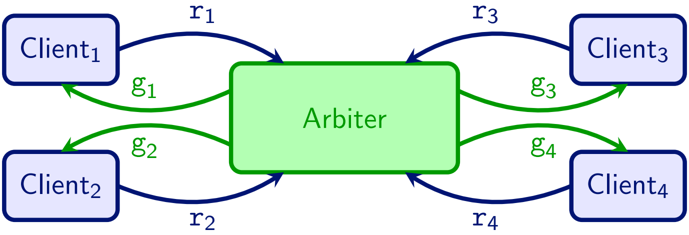


Parameters:

- Number of clients: r
- Number of prioritized clients: r_p

1. Answer every request of clients 1 through r_p promptly:
   $\bigwedge_{1 \leq i \leq r_p} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F_P}\,\mathtt{g}_i)$

Back to the Example

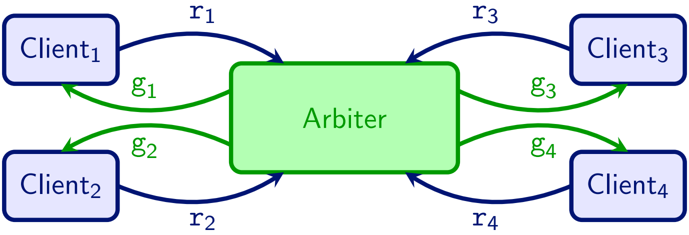


Parameters:

- Number of clients: r
- Number of prioritized clients: r_p

1. Answer every request of clients 1 through r_p promptly: $\bigwedge_{1 \leq i \leq r_p} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F_P}\,\mathtt{g}_i)$
2. Answer every other request eventually: $\bigwedge_{r_p < i} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$

Back to the Example



Parameters:

- Number of clients: r
- Number of prioritized clients: r_p

1. Answer every request of clients 1 through r_p promptly: $\bigwedge_{1 \leq i \leq r_p} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F_P}\,\mathtt{g}_i)$
2. Answer every other request eventually: $\bigwedge_{r_p < i} \mathbf{G}\,(\mathtt{r}_i \rightarrow \mathbf{F}\,\mathtt{g}_i)$
3. At most one grant at a time: $\mathbf{G} \bigwedge_{i \neq j} \neg(\mathtt{g}_i \wedge \mathtt{g}_j)$

LTL synthesis vs. Prompt-LTL synthesis

Resources	Prioritized Resources	LTL [s]	Prompt-LTL [s]
3	0		
	1		
	2		
	3		
4	0		
	1		
	2		
	3		
	4		

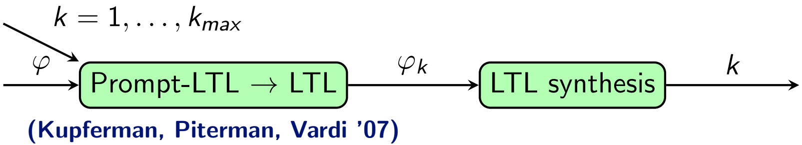
LTL synthesis vs. Prompt-LTL synthesis

Resources	Prioritized Resources	LTL [s]	Prompt-LTL [s]
3	0	0.26	
	1		
	2		
	3		
4	0	0.32	
	1		
	2		
	3		
	4		

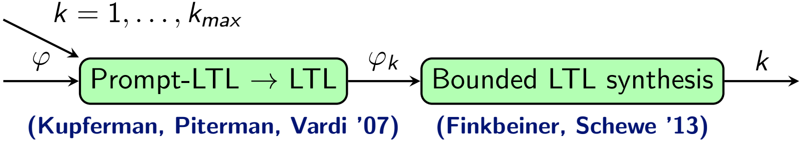
LTL synthesis vs. Prompt-LTL synthesis

Resources	Prioritized Resources	LTL [s]	Prompt-LTL [s]
3	0	0.26	0.37
	1		0.47
	2		0.64
	3		0.72
4	0	0.32	0.47
	1		1.32
	2		1.52
	3		1.72
	4		1.72

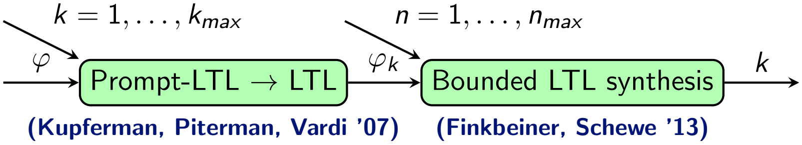
Bounded Prompt-LTL Approximation



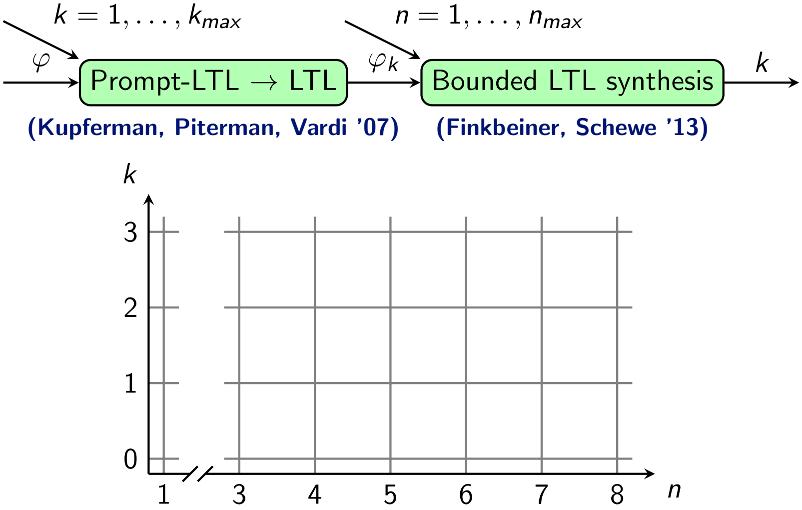
Bounded Prompt-LTL Approximation



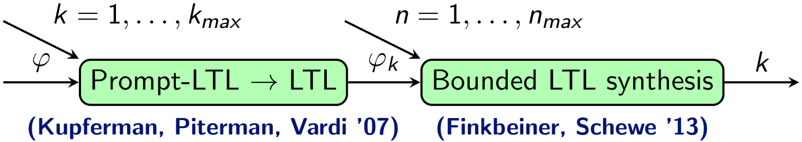
Bounded Prompt-LTL Approximation

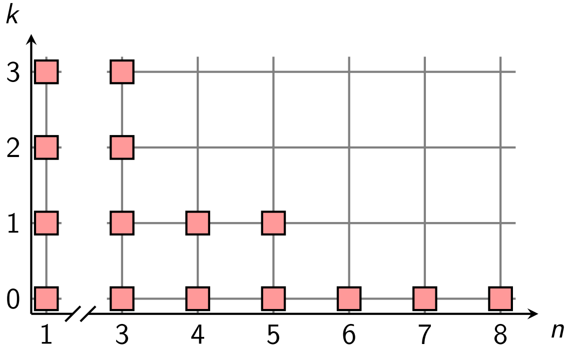


Bounded Prompt-LTL Approximation

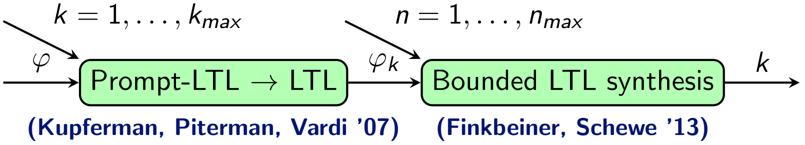


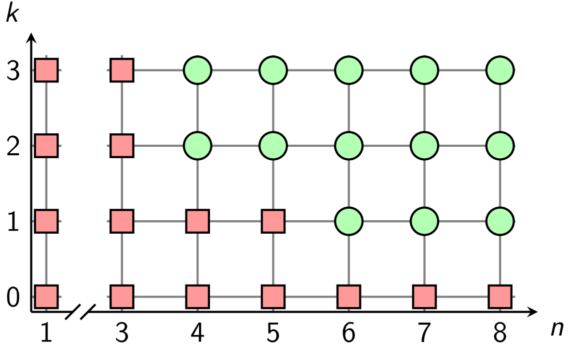
Bounded Prompt-LTL Approximation



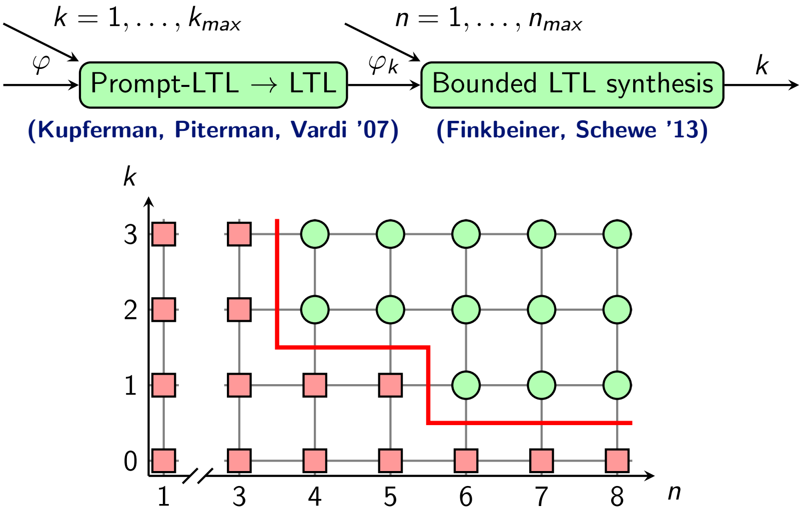


Bounded Prompt-LTL Approximation

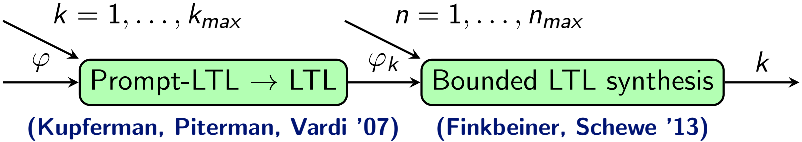


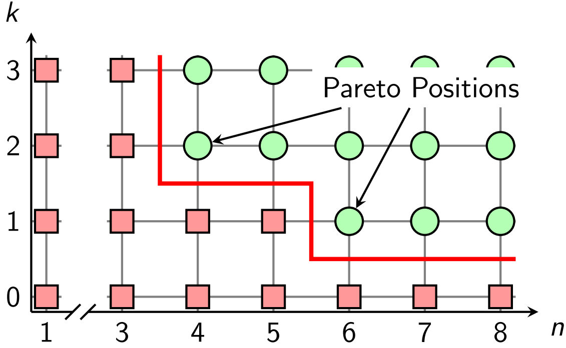


Bounded Prompt-LTL Approximation

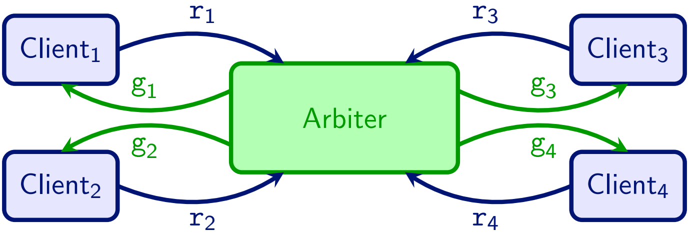


Bounded Prompt-LTL Approximation

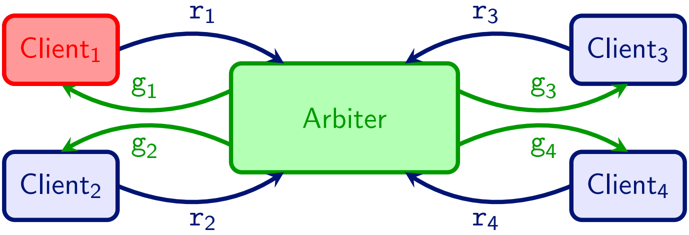




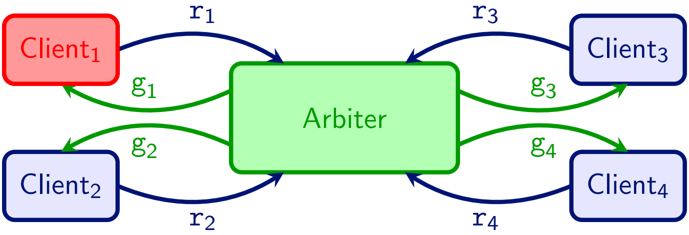
Strategies: Slow, but Small



Strategies: Slow, but Small

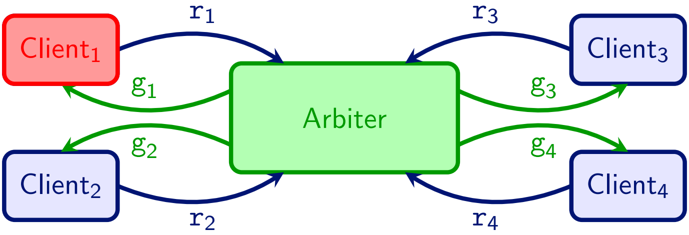


Strategies: Slow, but Small

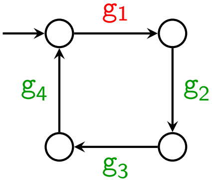


Always assume the worst: All requests in each step

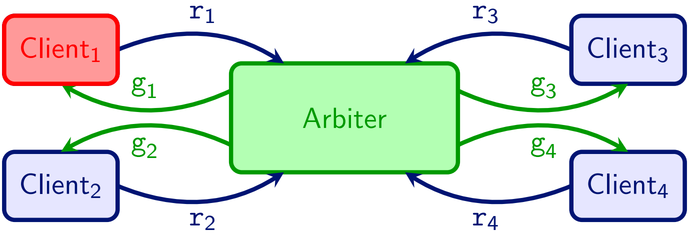
Strategies: Slow, but Small



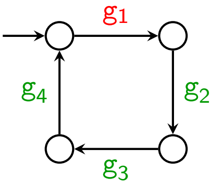
Always assume the worst: All requests in each step



Strategies: Slow, but Small

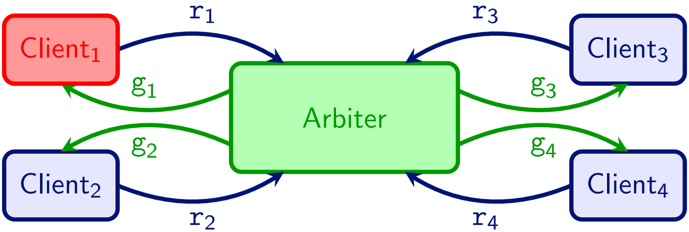


Always assume the worst: All requests in each step



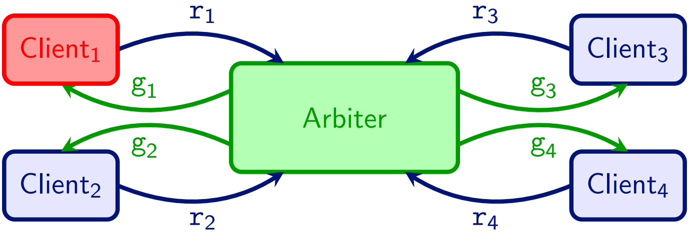
$\rightsquigarrow$ 4 states, maximal delay 3

Strategies: Fast, but Large

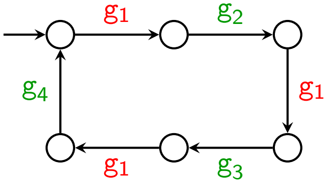


Always assume the worst: All requests in each step

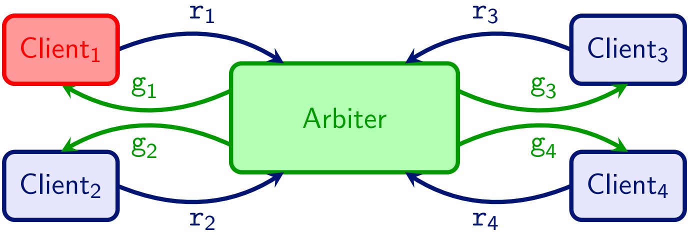
Strategies: Fast, but Large



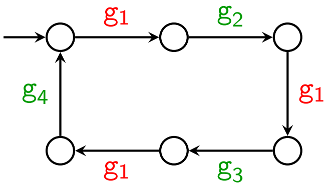
Always assume the worst: All requests in each step



Strategies: Fast, but Large



Always assume the worst: All requests in each step



$\rightsquigarrow$ 6 states, maximal delay 2

Pareto Positions

Theorem

There exists a family of Prompt-LTL formulas φ_b of size linear in b such that the output player has:

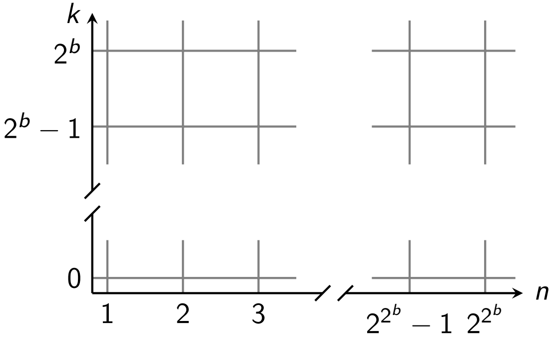
- *a positional strategy realizing φ_b w.r.t. $k = 2^b$, and*
- *a strategy of size $n = 2^{2^b}$ realizing φ_b w.r.t. $k = 0$.*

Pareto Positions

Theorem

There exists a family of Prompt-LTL formulas φ_b of size linear in b such that the output player has:

- *a positional strategy realizing φ_b w.r.t. $k = 2^b$, and*
- *a strategy of size $n = 2^{2^b}$ realizing φ_b w.r.t. $k = 0$.*

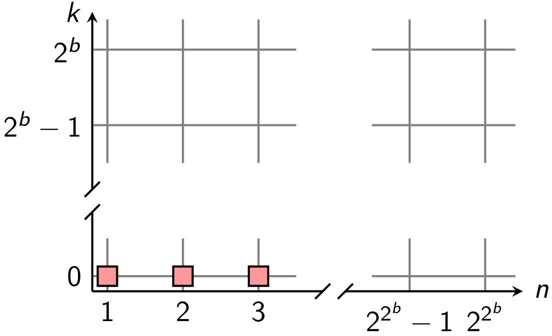


Pareto Positions

Theorem

There exists a family of Prompt-LTL formulas φ_b of size linear in b such that the output player has:

- *a positional strategy realizing φ_b w.r.t. $k = 2^b$, and*
- *a strategy of size $n = 2^{2^b}$ realizing φ_b w.r.t. $k = 0$.*

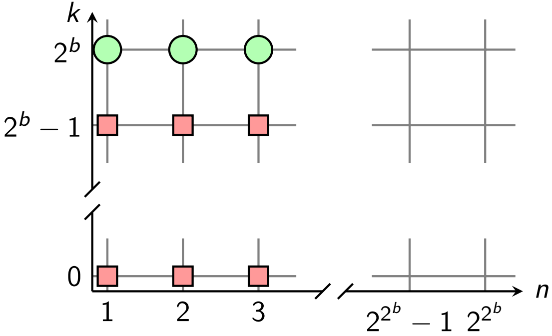


Pareto Positions

Theorem

There exists a family of Prompt-LTL formulas φ_b of size linear in b such that the output player has:

- *a positional strategy realizing φ_b w.r.t. $k = 2^b$, and*
- *a strategy of size $n = 2^{2^b}$ realizing φ_b w.r.t. $k = 0$.*

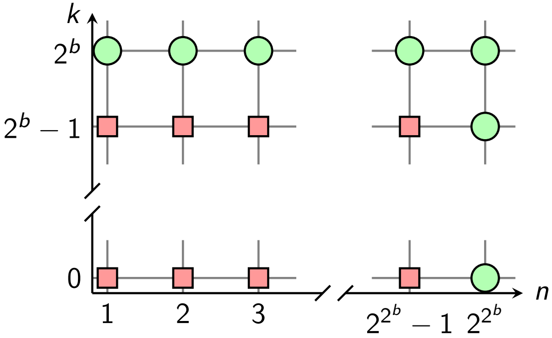


Pareto Positions

Theorem

There exists a family of Prompt-LTL formulas φ_b of size linear in b such that the output player has:

- *a positional strategy realizing φ_b w.r.t. $k = 2^b$, and*
- *a strategy of size $n = 2^{2^b}$ realizing φ_b w.r.t. $k = 0$.*



Conclusion

Our contribution:

- The first approximation algorithm for Prompt-LTL realizability with doubly-exponential running time
- Computes a realizing strategy
- Applicable to stronger logics as well
- Prototypical implementation
- Upper and lower bounds on tradeoff time vs. memory

Conclusion

Our contribution:

- The first approximation algorithm for Prompt-LTL realizability with doubly-exponential running time
- Computes a realizing strategy
- Applicable to stronger logics as well
- Prototypical implementation
- Upper and lower bounds on tradeoff time vs. memory

Take-away:

- Relaxing the optimality requirement for Prompt-LTL yields exponentially better runtime
- In general, memory can be traded for response time and vice versa.