Time-optimal Strategies for Infinite Games

Martin Zimmermann

RWTH Aachen University

March 10th, 2010

DIMAP Seminar
Warwick University, United Kingdom
Model Checking: program $P$, specification $\varphi$, does

$P \models \varphi$ ?
Introduction

Model Checking: program $P$, specification $\varphi$, does

$$P \models \varphi$$

Synthesis: environment $E$, specification $\varphi$. Generate program $P$ such that

$$E \times P \models \varphi.$$
Introduction

Model Checking: program $P$, specification $\varphi$, does

$$P \models \varphi$$

Synthesis: environment $E$, specification $\varphi$. Generate program $P$ such that

$$E \times P \models \varphi$$

Synthesis as a game: no matter what the environment does, the program has to guarantee $\varphi$.

- Beautiful and rich theory based on infinite graph games.
- Typically: a player either wins or loses (zero-sum).
- Here: adding quantitative aspects to infinite games.
1. Infinite Games

2. Poset Games

3. Parametric LTL Games

4. Finite-time Muller Games

5. Conclusion
Definitions

An arena $\mathcal{A} = (V, V_0, V_1, E, v_0, l)$ consists of
- a finite directed graph $(V, E)$ without dead-ends,
- a partition $\{V_0, V_1\}$ of $V$ denoting the positions of Player 0 (circles) and Player 1 (squares),
- an initial vertex $v_0 \in V$,
- a labeling function $l : V \rightarrow 2^P$ for some set $P$ of atomic propositions.
- **Play** in $A$: infinite path $\rho_0 \rho_1 \rho_2 \ldots$ starting in $v_0$. 
Definitions cont’d

- **Play** in $\mathcal{A}$: infinite path $\rho_0\rho_1\rho_2\ldots$ starting in $v_0$.
- **Strategy** for Player $i \in \{0, 1\}$: mapping $\sigma : V^*V_i \rightarrow V$ such that $(s, \sigma(ws)) \in E$.
- $\sigma$ is **finite-state**: $\sigma$ computable by finite automaton with output.
- $\rho_0\rho_1\rho_2\ldots$ is **consistent** with $\sigma$: $\rho_{n+1} = \sigma(\rho_0\ldots\rho_n)$ for all $n$ such that $\rho_n \in V_i$. 
Definitions cont’d

- **Play** in $\mathcal{A}$: infinite path $\rho_0\rho_1\rho_2 \ldots$ starting in $v_0$.
- **Strategy** for Player $i \in \{0, 1\}$: mapping $\sigma : V^* V_i \rightarrow V$ such that $(s, \sigma(ws)) \in E$.
- $\sigma$ is **finite-state**: $\sigma$ computable by finite automaton with output.
- $\rho_0\rho_1\rho_2 \ldots$ is **consistent** with $\sigma$: $\rho_{n+1} = \sigma(\rho_0 \ldots \rho_n)$ for all $n$ such that $\rho_n \in V_i$.

**Game**: $G = (\mathcal{A}, \text{Win})$ with $\text{Win} \subseteq V^\omega$.

- $\rho$ winning for Player 0: $\rho \in \text{Win}$.
- $\rho$ winning for Player 1: $\rho \in V^\omega \setminus \text{Win}$. 
Definitions cont’d

- **Play** in $A$: infinite path $\rho_0\rho_1\rho_2\ldots$ starting in $v_0$.
- **Strategy** for Player $i \in \{0, 1\}$: mapping $\sigma : V^* V_i \to V$ such that $(s, \sigma(ws)) \in E$.
- $\sigma$ is **finite-state**: $\sigma$ computable by finite automaton with output.
- $\rho_0\rho_1\rho_2\ldots$ is **consistent** with $\sigma$: $\rho_{n+1} = \sigma(\rho_0\ldots\rho_n)$ for all $n$ such that $\rho_n \in V_i$.

**Game**: $G = (A, Win)$ with $Win \subseteq V^\omega$.
- $\rho$ winning for Player 0: $\rho \in Win$.
- $\rho$ winning for Player 1: $\rho \in V^\omega \setminus Win$.
- $\sigma$ **winning strategy** for Player $i$: all plays $\rho$ consistent with $\sigma$ are winning for Player $i$.
- $G$ **determined**: one player has a winning strategy.
1. Infinite Games

2. Poset Games

3. Parametric LTL Games

4. Finite-time Muller Games

5. Conclusion
Motivation

- Request-Response conditions are a typical requirement on reactive systems.
- There is a natural definition of waiting times and they allow time-optimal strategies.
Motivation

- Request-Response conditions are a typical requirement on reactive systems.
- There is a natural definition of waiting times and they allow time-optimal strategies.

Goal:

- Extend the Request-Response condition to partially ordered objectives.
- .. while retaining the notion of waiting times and the existence of time-optimal strategies.
Request-Response games

Request-response game: $(Δ, (Q_j, P_j)_{j=1,...,k})$ where $Q_j, P_j \subseteq V$. Player 0 wins a play if every visit to $Q_j$ (request) is responded by a later visit to $P_j$. 
Request-Response games

Request-response game: \((\mathcal{A}, (Q_j, P_j)_{j=1,\ldots, k})\) where \(Q_j, P_j \subseteq V\).

Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

\[
\begin{align*}
t_1 & : 0 \\
t_2 & : 0
\end{align*}
\]
Request-response game: \((\mathcal{A}, (Q_j, P_j))_{j=1,...,k}\) where \(Q_j, P_j \subseteq V\).

Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

```
\begin{align*}
Q_1 & : 0 & 1 \\
Q_2 & : 0 & 0
\end{align*}
```

Graphical representation of the game.
Request-Response games

Request-response game: $(\mathcal{A}, (Q_j, P_j)_{j=1,...,k})$ where $Q_j, P_j \subseteq V$. Player 0 wins a play if every visit to $Q_j$ (request) is responded by a later visit to $P_j$.

\[
\begin{array}{c}
Q_1 \\
Q_2 \\
\end{array} \rightarrow
\begin{array}{c}
P_1 \\
\end{array}
\rightarrow
\begin{array}{c}
Q_1 \\
Q_2 \\
P_2 \\
\end{array}
\]

\[
t_1 : \ 0 \ 1 \ 2 \\
t_2 : \ 0 \ 0 \ 0
\]
Request-Response games

Request-response game: \((\mathcal{A}, (Q_j, P_j)_{j=1,\ldots,k})\) where \(Q_j, P_j \subseteq V\). Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

\[\begin{array}{c}
Q_1 \quad Q_2 \\
| \quad | \\
P_1 \quad P_2
\end{array}\]

\[
t_1 : \quad 0 \quad 1 \quad 2 \quad 0 \\
t_2 : \quad 0 \quad 0 \quad 0 \quad 0 \quad 0
\]
Request-Response games

Request-response game: \((\mathcal{A}, (Q_j, P_j)_{j=1,\ldots,k})\) where \(Q_j, P_j \subseteq V\).

Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

\[
\begin{align*}
t_1 & : 0 \ 1 \ 2 \ 0 \ 0 \ 0 \\
t_2 & : 0 \ 0 \ 0 \ 0 \ 0 \ 1
\end{align*}
\]
Request-Response games

Request-response game: \( (A, (Q_j, P_j)_{j=1, \ldots, k}) \) where \( Q_j, P_j \subseteq V \). Player 0 wins a play if every visit to \( Q_j \) (request) is responded by a later visit to \( P_j \).

\[
\begin{array}{ccccccc}
0 & 1 & 2 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 2 \\
\end{array}
\]
Request-Response games

Request-response game: \((\mathcal{A}, (Q_j, P_j)_{j=1,\ldots,k})\) where \(Q_j, P_j \subseteq V\). Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

![Graph](image)

\[
t_1 : \quad 0 \quad 1 \quad 2 \quad 0 \quad 0 \quad 1 \quad 2 \\
t_2 : \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3
\]
Request-Response games

Request-response game: $(\mathcal{A}, (Q_j, P_j)_{j=1,\ldots,k})$ where $Q_j, P_j \subseteq V$. Player 0 wins a play if every visit to $Q_j$ (request) is responded by a later visit to $P_j$.

\begin{itemize}
    \item $t_1: 0\ 1\ 2\ 0\ 0\ 1\ 2\ 3$
    \item $t_2: 0\ 0\ 0\ 0\ 1\ 2\ 3\ 4$
\end{itemize}
Request-Response games

Request-response game: \((\mathcal{A}, (Q_j, P_j)_{j=1,...,k})\) where \(Q_j, P_j \subseteq V\). Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

![Diagram of a request-response game]

\[ t_1 : \quad 0 \quad 1 \quad 2 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \]
\[ t_2 : \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \]
Request-Response games

Request-response game: \((\mathcal{A}, (Q_j, P_j)_{j=1,\ldots,k})\) where \(Q_j, P_j \subseteq V\).
Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

\[
\begin{align*}
t_1 & : 0 & 1 & 2 & 0 & 0 & 1 & 2 & 3 & 4 & 5 \\
t_2 & : 0 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 5 & 0
\end{align*}
\]
Request-Response games

Request-response game: \((A, (Q_j, P_j)_{j=1,...,k})\) where \(Q_j, P_j \subseteq V\). Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

![Graph representation of request-response game]

\[
\begin{align*}
t_1 : & \quad 0 \quad 1 \quad 2 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 0 \\
t_2 : & \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 0 \quad 0
\end{align*}
\]
Request-Response games

Request-response game: \((\mathcal{A}, (Q_j, P_j)_{j=1,\ldots,k})\) where \(Q_j, P_j \subseteq V\). Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

\[
\begin{align*}
t_1 & : 0 \quad 1 \quad 2 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 0 \\
\end{align*}
\[
\begin{align*}
t_2 & : 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 0 \quad 0 \\
\end{align*}
\[
\begin{align*}
p_i = t_1 + t_2 & : 0 \quad 1 \quad 2 \quad 0 \quad 1 \quad 3 \quad 5 \quad 7 \quad 9 \quad 5 \quad 0 \\
\end{align*}
\]
Request-Response games

Request-response game: \((\mathcal{A}, (Q_j, P_j)_{j=1,\ldots,k})\) where \(Q_j, P_j \subseteq V\).
Player 0 wins a play if every visit to \(Q_j\) (request) is responded by a later visit to \(P_j\).

\[
\begin{align*}
\mathcal{A} &:= (Q_1, Q_2, P_1, P_2) \\
Q_1 &\rightarrow Q_2 \\
Q_2 &\rightarrow Q_1, P_1, P_2 \\
P_1 &\rightarrow Q_1 \\
P_2 &\rightarrow Q_2, P_1, P_2
\end{align*}
\]

\[
\begin{align*}
t_1 &:= 0, 1, 2, 0, 0, 1, 2, 3, 4, 5, 0 \\
t_2 &:= 0, 0, 0, 0, 1, 2, 3, 4, 5, 0, 0 \\
p_i &= t_1 + t_2 := 0, 1, 2, 0, 1, 3, 5, 7, 9, 5, 0 \\
\frac{1}{n} \sum_{i=1}^{n} p_i &:= 0, \frac{1}{2}, 1, \frac{3}{4}, \frac{4}{5}, \frac{7}{6}, \frac{12}{7}, \frac{19}{8}, \frac{28}{9}, \frac{34}{10}, \frac{34}{11}
\end{align*}
\]
Request-Reponse Games: Results

- **Waiting times**: start a clock for every request that is stopped as soon as it is responded (and ignore subsequent requests).

- Accumulated waiting time: sum up the clock values of a play prefix (quadratic influence of open requests).

- **Value of a play**: limit superior of the average accumulated waiting time.

- **Value of a strategy**: value of the worst play consistent with the strategy.
Request-Response Games: Results

- Waiting times: start a clock for every request that is stopped as soon as it is responded (and ignore subsequent requests).
- Accumulated waiting time: sum up the clock values of a play prefix (quadratic influence of open requests).
- Value of a play: limit superior of the average accumulated waiting time.
- Value of a strategy: value of the worst play consistent with the strategy.

Theorem (Horn, Thomas, Wallmeier)

If Player 0 has a winning strategy for an RR-game, then she also has an optimal winning strategy, which is finite-state and effectively computable.
Extending Request-Response Games
Generalize RR-games to express more complicated conditions, but retain notion of time-optimality.

Request: still a singular event.

Response: partially ordered set of events.
A Play

\[
\text{train go}
\]

\[
\begin{array}{cc}
\text{lower}_0 & \text{lower}_1 \\
\text{red}_0 & \text{red}_1 \\
\end{array}
\]

\{req\}
A Play

\{\text{req}\} \quad \{\text{red}_1\}

\text{train go}

\text{lower}_0 \quad \text{lower}_1

\text{red}_0 \quad \text{red}_1
A Play

\[
\text{train go} \\
\text{lower}_0 \quad \text{lower}_1 \\
\text{red}_0 \quad \text{red}_1 \\
\{\text{req}\} \quad \{\text{red}_1\} \quad \{\text{red}_0\}
\]
A Play

\[
\begin{align*}
\text{train} & \quad \text{go} \\
\text{lower}_0 & \quad \text{lower}_1 \\
\text{red}_0 & \quad \text{red}_1 \\
\{\text{req}\} & \quad \{\text{red}_1\} & \quad \{\text{red}_0\} & \quad \{\text{lower}_0\}
\end{align*}
\]
A Play

The diagram represents a game with the following states and transitions:

- **Train go**
- **lower_0**
- **lower_1**
- **red_0**
- **red_1**

The states are connected as follows:

- From **req**, transitions to **red_1**.
- From **red_1**, transitions to **red_0**.
- From **red_0**, transitions to **lower_0** and **lower_1**.
- From **lower_0**, transitions to **lower_1**.
- From **lower_1**, transitions to **red_0** and **red_1**.

This diagram explains the game's structure and possible moves between states.
Winning condition for Player 0: every request $q_j$ is responded by a later embedding of $\mathcal{P}_j$. 
Theorem

*Poset games are determined with finite-state strategies, i.e., in every poset game, one of the players has a finite-state winning strategy.*
Theorem

Poset games are determined with finite-state strategies, i.e., in every poset games, one of the players has a finite-state winning strategy.

Proof:
Reduction to Büchi games; memory is used
- to store elements of the posets that still have to be embedded,
- to deal with overlapping embeddings,
- to implement a cyclic counter to ensure that every request is responded by an embedding.

Size of the memory: exponential in the size of the posets $\mathcal{P}_j$. 
Waiting Times

As desired, a natural definition of waiting times is retained:

- Start a clock if a request is encountered...
- ... that is stopped as soon as the embedding is completed.
- Need a clock for every request (even if another request is already open).
Waiting Times

As desired, a natural definition of waiting times is retained:

- Start a clock if a request is encountered...
- ... that is stopped as soon as the embedding is completed.
- Need a clock for every request (even if another request is already open).

- Value of a play: limit superior of the average accumulated waiting time.
- Value of a strategy: value of the worst play consistent with the strategy.
- Corresponding notion of optimal strategies.
The Main Theorem

Theorem

If Player 0 has a winning strategy for a poset game $G$, then she also has an optimal winning strategy, which is finite-state and effectively computable.
The Main Theorem

**Theorem**

If Player 0 has a winning strategy for a poset game $G$, then she also has an optimal winning strategy, which is finite-state and effectively computable.

Proof:

- If Player 0 has a winning strategy, then she also has one of value less than a certain constant $c$ (from reduction). This bounds the value of an optimal strategy, too.
- For every strategy of value $\leq c$ there is another strategy of smaller or equal value, that also bounds all waiting times and bounds the number of open requests.
- If the waiting times and the number of open requests are bounded, then $G$ can be reduced to a mean-payoff game.
Further research and Open Problems

Size of the mean-payoff game: super-exponential in the size of the poset game (holds already for RR-games). Needed: tight bounds on the length of a non-self-covering sequence of waiting time vectors.
Further research and Open Problems

Size of the mean-payoff game: super-exponential in the size of the poset game (holds already for RR-games). Needed: tight bounds on the length of a non-self-covering sequence of waiting time vectors.

Also:

- Heuristic algorithms and approximatively optimal strategies.
- Lower bounds on the memory size of an optimal strategy.
- Direct computation of optimal strategies (without reduction to mean-payoff games).
- Other valuation functions for plays (e.g., discounting, \( \limsup \sum_{i=1}^{k} t_i \)).
- Tradeoff between size and value of a strategy.
Outline

1. Infinite Games

2. Poset Games

3. Parametric LTL Games

4. Finite-time Muller Games

5. Conclusion
Motivation

Here, we consider winning conditions in linear temporal logic (LTL). Advantages of LTL as specification language are

- compact, variable-free syntax,
- intuitive semantics,
- successfully employed in model checking tools.

Drawback: LTL lacks capabilities to express timing constraints.
Motivation

Here, we consider winning conditions in linear temporal logic (LTL). Advantages of LTL as specification language are

- compact, variable-free syntax,
- intuitive semantics,
- successfully employed in model checking tools.

Drawback: LTL lacks capabilities to express timing constraints.

Solution: Consider games with winning conditions in extensions of LTL that can express timing constraints.
LTL

Formulae of Linear temporal logic over $P$:

$$\varphi ::= p \mid \neg p \mid \varphi \land \varphi \mid \varphi \lor \varphi \mid X\varphi \mid F\varphi \mid G\varphi$$
Formulae of Linear temporal logic over $P$:

$$
\varphi ::= p \mid \neg p \mid \varphi \land \varphi \mid \varphi \lor \varphi \mid X\varphi \mid F\varphi \mid G\varphi
$$

LTL is evaluated at positions $i$ of infinite words $\rho$ over $2^P$:

$$(\rho, i) \models X\varphi: \quad \rho \quad | \quad \cdots \quad | \quad \cdots \quad \varphi \quad | \quad i \quad | \quad \cdots
$$

$$(\rho, i) \models F\varphi: \quad \rho \quad | \quad \cdots \quad | \quad \cdots \quad \varphi \quad | \quad i \quad | \quad \cdots
$$

$$(\rho, i) \models G\varphi: \quad \rho \quad | \quad \cdots \quad | \quad \varphi \quad | \quad \varphi \quad | \quad \varphi \quad | \quad \varphi \quad | \quad \varphi \quad | \quad i \quad | \quad \cdots
$$
Parametric LTL

Let $\mathcal{X}$ and $\mathcal{Y}$ be two disjoint sets of variables. PLTL adds bounded temporal operators to LTL:

- $F_{\leq x}$ for $x \in \mathcal{X}$,
- $G_{\leq y}$ for $y \in \mathcal{Y}$.
Let $\mathcal{X}$ and $\mathcal{Y}$ be two disjoint sets of variables. PLTL adds bounded temporal operators to LTL:

- $F_{\leq x}$ for $x \in \mathcal{X}$,
- $G_{\leq y}$ for $y \in \mathcal{Y}$.

Semantics defined w.r.t. variable valuation $\alpha: \mathcal{X} \cup \mathcal{Y} \rightarrow \mathbb{N}$.

$(\rho, i, \alpha) \models F_{\leq x} \varphi$:  
\[ \rho \quad \cdots \quad i \quad \vdots \quad i + \alpha(x) \]

$(\rho, i, \alpha) \models G_{\leq y} \varphi$:  
\[ \varphi \quad \varphi \quad \varphi \quad \varphi \quad \varphi \quad \varphi \quad i \quad \vdots \quad i + \alpha(y) \]
Parametric LTL Games

PLTL game \((\mathcal{A}, \varphi)\):

- \(\sigma\) is a winning strategy for Player 0 \(w.r.t.\ \alpha\) iff for all plays \(\rho\) consistent with \(\sigma\): \((\rho, 0, \alpha) \models \varphi\).
- \(\tau\) is a winning strategy for Player 1 \(w.r.t.\ \alpha\) iff for all plays \(\rho\) consistent with \(\tau\): \((\rho, 0, \alpha) \not\models \varphi\).
Parametric LTL Games

PLTL game \((A, \varphi)\):

- \(\sigma\) is a winning strategy for Player 0 w.r.t. \(\alpha\) iff for all plays \(\rho\) consistent with \(\sigma\): \((\rho, 0, \alpha) \models \varphi\).
- \(\tau\) is a winning strategy for Player 1 w.r.t. \(\alpha\) iff for all plays \(\rho\) consistent with \(\tau\): \((\rho, 0, \alpha) \not\models \varphi\).

The set of winning valuations for Player \(i\) is

\[ W^i_G = \{ \alpha \mid \text{Player } i \text{ has winning strategy for } G \text{ w.r.t. } \alpha \} \]

We are interested in the emptiness, finiteness, and universality problem for \(W^i_G\) and in finding optimal valuations in \(W^i_G\).
Winning condition $\mathbf{G}(q \rightarrow \mathbf{F}_{\leq x} p)$: “Every request $q$ is eventually responded by $p$”.

Player 0's goal: uniformly bound the waiting times between requests $q$ and responses $p$ by $\alpha(x)$.

\[ q \quad p \quad q \quad p \]

\[ \leq \alpha(x) \quad \leq \alpha(x) \]
Winning condition $\mathbf{G}(q \rightarrow \mathbf{F}_{\leq x} p)$: “Every request $q$ is eventually responded by $p$”.

- Player 0’s goal: uniformly bound the waiting times between requests $q$ and responses $p$ by $\alpha(x)$.

- Player 1’s goal: enforce waiting time greater than $\alpha(x)$. 
PLTL Games: Example

Winning condition $\mathbf{G}(q \rightarrow \mathbf{F}_{\leq x} p)$: “Every request $q$ is eventually responded by $p$”.

- Player 0’s goal: uniformly bound the waiting times between requests $q$ and responses $p$ by $\alpha(x)$.

- Player 1’s goal: enforce waiting time greater than $\alpha(x)$.

Note: the winning condition induces an optimization problem (for Player 0): minimize $\alpha(x)$. 
Theorem (Pnueli, Rosner ’89)

Determining the winner of an LTL game is $2\text{EXPTIME}$-complete.
Theorem (Pnueli, Rosner ’89)

Determining the winner of an LTL game is $2\text{EXPTIME}$-complete.

Theorem

Let $G$ be a PLTL game. The emptiness, finiteness, and universality problem for $W^i_G$ are $2\text{EXPTIME}$-complete.
PLTL: Results

Theorem (Pnueli, Rosner ’89)

Determining the winner of an LTL game is 2EXPTIME-complete.

Theorem

Let \( G \) be a PLTL game. The emptiness, finiteness, and universality problem for \( W^i_G \) are 2EXPTIME-complete.

So, adding bounded temporal operators does increase the complexity of solving games.
If $\varphi$ contains only $F_{\leq x}$ respectively only $G_{\leq y}$, then solving games is an **optimization problem**: which is the *best* valuation in $\mathcal{W}_G^0$?
If $\varphi$ contains only $F_{\leq x}$ respectively only $G_{\leq y}$, then solving games is an optimization problem: which is the best valuation in $W_0^G$?

**Theorem**

Let $\varphi_F$ be $G_{\leq y}$-free and $\varphi_G$ be $F_{\leq x}$-free, let $G_F = (A, \varphi_F)$ and $G_G = (A, \varphi_G)$. Then, the following values are computable:

- $\min_{\alpha \in W_0^{G_F}} \max_{x \in \text{var}(\varphi_F)} \alpha(x)$. 
If $\varphi$ contains only $F_{\leq x}$ respectively only $G_{\leq y}$, then solving games is an optimization problem: which is the best valuation in $W^0_G$?

**Theorem**

Let $\varphi_F$ be $G_{\leq y}$-free and $\varphi_G$ be $F_{\leq x}$-free, let $G_F = (A, \varphi_F)$ and $G_G = (A, \varphi_G)$. Then, the following values are computable:

- $\min_{\alpha \in W^0_{G_F}} \max_{x \in \text{var}(\varphi_F)} \alpha(x)$.
- $\min_{\alpha \in W^0_{G_F}} \min_{x \in \text{var}(\varphi_F)} \alpha(x)$.
- $\max_{\alpha \in W^0_{G_G}} \max_{y \in \text{var}(\varphi_G)} \alpha(y)$.
- $\max_{\alpha \in W^0_{G_G}} \min_{y \in \text{var}(\varphi_G)} \alpha(y)$.
If $\varphi$ contains only $F_{\leq x}$ respectively only $G_{\leq y}$, then solving games is an optimization problem: which is the best valuation in $W^0_G$?

**Theorem**

Let $\varphi_F$ be $G_{\leq y}$-free and $\varphi_G$ be $F_{\leq x}$-free, let $G_F = (A, \varphi_F)$ and $G_G = (A, \varphi_G)$. Then, the following values are computable:

- $\min_{\alpha \in W^0_{G_F}} \max_{x \in \text{var}(\varphi_F)} \alpha(x)$.
- $\min_{\alpha \in W^0_{G_F}} \min_{x \in \text{var}(\varphi_F)} \alpha(x)$.
- $\max_{\alpha \in W^0_{G_G}} \max_{y \in \text{var}(\varphi_G)} \alpha(y)$.
- $\max_{\alpha \in W^0_{G_G}} \min_{y \in \text{var}(\varphi_G)} \alpha(y)$.

**Proof idea:** obtain (double-exponential) upper bound $k$ on the optimal value by a reduction to an LTL game. Then, perform binary search in the interval $(0, k)$ to find the optimum.
Further research and Open Problems

- Again: tradeoff between size and quality of a finite-state strategy.
- Better algorithms for the optimization problems.
- Hardness results for the optimization problems.
1. Infinite Games

2. Poset Games

3. Parametric LTL Games

4. Finite-time Muller Games

5. Conclusion
Motivation

σ positional strategy: σ(ω) only depends on the last vertex of ω.
Motivation

\( \sigma \) positional strategy: \( \sigma(w) \) only depends on the last vertex of \( w \).

- Assume a game allows positional winning strategies for both players.
- Then, we can stop a play as soon as the first loop is closed.
- Winner is determined by infinite repetition of this loop.
Motivation

σ positional strategy: $\sigma(w)$ only depends on the last vertex of $w$.

- Assume a game allows positional winning strategies for both players.
- Then, we can stop a play as soon as the first loop is closed.
- Winner is determined by infinite repetition of this loop.

Is there an analogous notion for games with finite-state strategies? Here, we consider Muller games.
\[ \text{Inf}(\rho) = \{ v \in V \mid \exists \omega n \in \mathbb{N} \text{ such that } \rho_n = v \}. \]

**Muller game:** \((A, \mathcal{F}_0, \mathcal{F}_1)\) such that \(\{\mathcal{F}_0, \mathcal{F}_1\}\) is a partition of \(2^V \backslash \{\emptyset\}\). A play \(\rho\) is winning for Player \(i\), if \(\text{Inf}(\rho) \in \mathcal{F}_i\).
Muller Games

\[ \text{Inf}(\rho) = \{ v \in V \mid \exists \omega n \in \mathbb{N} \text{ such that } \rho_n = v \} . \]

**Muller game:** \((A, \mathcal{F}_0, \mathcal{F}_1)\) such that \(\{\mathcal{F}_0, \mathcal{F}_1\}\) is a partition of \(2^V \setminus \{\emptyset\}\). A play \(\rho\) is winning for Player \(i\), if \(\text{Inf}(\rho) \in \mathcal{F}_i\).

**Theorem**

*Muller games are determined with finite-state strategies of size \(|V| \cdot |V|!\).*
Finite-time Muller game: $(\mathcal{A}, \mathcal{F}_0, \mathcal{F}_1, k)$ such that $\{\mathcal{F}_0, \mathcal{F}_1\}$ is a partition of $2^V \setminus \{\emptyset\}$ and $k \geq 2$. A finite play $w$ is winning for Player $i$, if $F \in \mathcal{F}_i$, where $F$ is the first loop that is seen $k$ times in a row.
Finite-time Muller game: $(A, \mathcal{F}_0, \mathcal{F}_1, k)$ such that $\{\mathcal{F}_0, \mathcal{F}_1\}$ is a partition of $2^V \backslash \{\emptyset\}$ and $k \geq 2$. A finite play $w$ is winning for Player $i$, if $F \in \mathcal{F}_i$, where $F$ is the first loop that is seen $k$ times in a row.

**Example**

![Diagram](image)

Let $k = 2$: play
Finite-time Muller game: \((\mathcal{A}, \mathcal{F}_0, \mathcal{F}_1, k)\) such that \(\{\mathcal{F}_0, \mathcal{F}_1\}\) is a partition of \(2^V \setminus \{\emptyset\}\) and \(k \geq 2\). A finite play \(w\) is winning for Player \(i\), if \(F \in \mathcal{F}_i\), where \(F\) is the first loop that is seen \(k\) times in a row.

Example

\[\begin{array}{c}
\quad
\end{array}\]

Let \(k = 2\): play \(v_0\)
Finite-time Muller game: \((A, F_0, F_1, k)\) such that \(\{F_0, F_1\}\) is a partition of \(2^V \setminus \{\emptyset\}\) and \(k \geq 2\). A finite play \(w\) is winning for Player \(i\), if \(F \in F_i\), where \(F\) is the first loop that is seen \(k\) times in a row.

**Example**

![Diagram](image)

Let \(k = 2\): play \(v_0, v_2\)
Finite-time Muller Games

Finite-time Muller game: \((A, \mathcal{F}_0, \mathcal{F}_1, k)\) such that \(\{\mathcal{F}_0, \mathcal{F}_1\}\) is a partition of \(2^V \setminus \{\emptyset\}\) and \(k \geq 2\). A finite play \(w\) is winning for Player \(i\), if \(F \in \mathcal{F}_i\), where \(F\) is the first loop that is seen \(k\) times in a row.

Example

Let \(k = 2\): play \(v_0 \ v_2 \ v_0\)
Finite-time Muller game: $(\mathcal{A}, \mathcal{F}_0, \mathcal{F}_1, k)$ such that $\{\mathcal{F}_0, \mathcal{F}_1\}$ is a partition of $2^V\setminus\{\emptyset\}$ and $k \geq 2$. A finite play $w$ is winning for Player $i$, if $F \in \mathcal{F}_i$, where $F$ is the first loop that is seen $k$ times in a row.

Example

Let $k = 2$: play $v_0 \ v_2 \ v_0 \ v_1$
Finite-time Muller game: \((A, \mathcal{F}_0, \mathcal{F}_1, k)\) such that \(\{\mathcal{F}_0, \mathcal{F}_1\}\) is a partition of \(2^V \setminus \{\emptyset\}\) and \(k \geq 2\). A finite play \(w\) is winning for Player \(i\), if \(F \in \mathcal{F}_i\), where \(F\) is the first loop that is seen \(k\) times in a row.

**Example**

Let \(k = 2\): play \(v_0 \ v_2 \ v_0 \ v_1 \ v_1\)
Finite-time Muller game: \((A, F_0, F_1, k)\) such that \(\{F_0, F_1\}\) is a partition of \(2^V \setminus \{\emptyset\}\) and \(k \geq 2\). A finite play \(w\) is winning for Player \(i\), if \(F \in F_i\), where \(F\) is the first loop that is seen \(k\) times in a row.

Example

Let \(k = 2\): play \(v_0 \ v_2 \ v_0 \ v_1 \ v_1 \ v_0\).
Finite-time Muller Games

Finite-time Muller game: \((\mathcal{A}, \mathcal{F}_0, \mathcal{F}_1, k)\) such that \(\{\mathcal{F}_0, \mathcal{F}_1\}\) is a partition of \(2^V \setminus \{\emptyset\}\) and \(k \geq 2\). A finite play \(w\) is winning for Player \(i\), if \(F \in \mathcal{F}_i\), where \(F\) is the first loop that is seen \(k\) times in a row.

**Example**

\[
\begin{array}{c}
\text{\(v_1\)} \\
\text{\(v_0\)} \\
\text{\(v_2\)}
\end{array}
\]

Let \(k = 2\): play \(v_0 \ v_2 \ v_0 \ v_1 \ v_1 \ v_0\). \(F = \{v_0, v_1\}\) seen twice.
**Finite-time Muller Games**

Finite-time Muller game: \((A, \mathcal{F}_0, \mathcal{F}_1, k)\) such that \(\{\mathcal{F}_0, \mathcal{F}_1\}\) is a partition of \(2^V \setminus \{\emptyset\}\) and \(k \geq 2\). A finite play \(w\) is winning for Player \(i\), if \(F \in \mathcal{F}_i\), where \(F\) is the first loop that is seen \(k\) times in a row.

**Example**

![Diagram](image)

Let \(k = 2\): play \(v_0\ v_2\ v_0\ v_1\ v_1\ v_0\). \(F = \{v_0, v_1\}\) seen twice.

**Theorem**

*Finite-time Muller games are determined.*
Theorem

Let $\mathcal{A}$ be an arena and $k = |V|^2 \cdot |V|! + 1$. Player $i$ wins the Muller game $(\mathcal{A}, F_0, F_1)$ iff she wins the finite-time Muller game $(\mathcal{A}, F_0, F_1, k)$. 
Theorem

Let $A$ be an arena and $k = |V|^2 \cdot |V|! + 1$. Player $i$ wins the Muller game $(A, F_0, F_1)$ iff she wins the finite-time Muller game $(A, F_0, F_1, k)$.

Proof:

A finite-state winning strategy for Player $i$ does not see $F \in F_{1-i} k$ times in a row.
Further research and Open Problems

Conjecture

Player $i$ wins the Muller game $(A, F_0, F_1)$ iff she wins the finite-time Muller game $(A, F_0, F_1, 2)$. 
Conjecture

Player $i$ wins the Muller game $(\mathcal{A}, \mathcal{F}_0, \mathcal{F}_1)$ iff she wins the finite-time Muller game $(\mathcal{A}, \mathcal{F}_0, \mathcal{F}_1, 2)$.

Also:

- Is there a natural definition of eager strategies?
- Complexity of solving a finite-time Muller game? It is just a reachability game (albeit a large one), so simple algorithms exist.
- Starting with a winning strategy for a finite-time Muller game, can we construct a (finite-state) winning strategy for the Muller game.
Outline

1. Infinite Games
2. Poset Games
3. Parametric LTL Games
4. Finite-time Muller Games
5. Conclusion
Collaboration

Three suggestions from my side:

- Request-response games and Poset games
- PLTL games
- Finite-time Muller games
Collaboration

Three suggestions from my side:

- Request-response games and Poset games
- PLTL games
- Finite-time Muller games

Thank you!