Games with Costs and Delays

Martin Zimmermann
Saarland University
June 20th, 2017
LICS 2017, Reykjavik, Iceland
**Büchi-Landweber:** The winner of a zero-sum two-player game of infinite duration with \( \omega \)-regular winning condition can be determined effectively.
Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with $\omega$-regular winning condition can be determined effectively.

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]
**Büchi-Landweber:** The winner of a zero-sum two-player game of infinite duration with \( \omega \)-regular winning condition can be determined effectively.

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

**I:** \( b \)

**O:** 
**Büchi-Landweber:** The winner of a zero-sum two-player game of infinite duration with $\omega$-regular winning condition can be determined effectively.

$$(\alpha(0)) \begin{pmatrix} \alpha(1) \\ \beta(0) \end{pmatrix} \begin{pmatrix} \beta(1) \\ \vdots \end{pmatrix} \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i$$

**I:** $b$

**O:** $a$
**Büchi-Landweber:** The winner of a zero-sum two-player game of infinite duration with $\omega$-regular winning condition can be determined effectively.

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[
I: \quad b \quad a
\]

\[
O: \quad a
\]
Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with $\omega$-regular winning condition can be determined effectively.

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\cdot
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\ldots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[\begin{array}{cccccc}
I: & b & a \\
O: & a & a \\
\end{array} \]
**Gale-Stewart Games**

**Büchi-Landweber:** The winner of a zero-sum two-player game of infinite duration with $\omega$-regular winning condition can be determined effectively.

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[I: \quad b \quad a \quad b\]

\[O: \quad a \quad a\]
**Gale-Stewart Games**

**Büchi-Landweber:** The winner of a zero-sum two-player game of infinite duration with $\omega$-regular winning condition can be determined effectively.

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[
\begin{array}{ccccccc}
I: & b & a & b & \cdots & \text{I wins!} \\
O: & a & a & \cdots
\end{array}
\]
Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with $\omega$-regular winning condition can be determined effectively.

$$\begin{pmatrix} \alpha(0) \\ \beta(0) \end{pmatrix} \begin{pmatrix} \alpha(1) \\ \beta(1) \end{pmatrix} \cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

$I$: $b$ $a$ $b$ $\cdots$  \hspace{1cm} \text{I wins!} \\

$O$: $a$ $a$ $\cdots$ 

- Many possible extensions... we consider two:
  - **Interaction**: one player may delay her moves.
  - **Winning condition**: quantitative instead of qualitative.
Allow Player $O$ to delay her moves:

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]
Allow Player $O$ to delay her moves:

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

$I$: $b$

$O$:
Delay Games

Allow Player $O$ to delay her moves:

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

$I$: \hspace{1cm} $b$ \hspace{1cm} $a$

$O$: \hspace{1cm}
Allow Player $O$ to delay her moves:

\[
\begin{pmatrix} \alpha(0) \\ \beta(0) \end{pmatrix} \begin{pmatrix} \alpha(1) \\ \beta(1) \end{pmatrix} \cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[
\text{I: } \ b \quad a \quad b
\]

\[
\text{O: }
\]
Delay Games

Allow Player $O$ to delay her moves:

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

$I$: $b\ a\ b$

$O$: $b$
Allow Player $O$ to delay her moves:

\[
\begin{pmatrix}
\alpha(0)
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1)
\beta(1)
\end{pmatrix} \ldots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

$I$: $b$ $a$ $b$ $b$

$O$: $b$
Allow Player O to delay her moves:

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[I: \quad b \quad a \quad b \quad b\]

\[O: \quad b \quad b\]
Allow Player $O$ to delay her moves:

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[
I: \quad b \quad a \quad b \quad b \quad a
\]

\[
O: \quad b \quad b
\]
Delay Games

Allow Player \( O \) to delay her moves:

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[ I: \quad b \quad a \quad b \quad b \quad a \]

\[ O: \quad b \quad b \quad a \]
Allow Player $O$ to delay her moves:

$$
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix} \cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
$$

\begin{align*}
I: & \quad b \quad a \quad b \quad b \quad a \quad a \\
O: & \quad b \quad b \quad a
\end{align*}
Allow Player $O$ to delay her moves:

$$
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix} \cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
$$

$I$: $b$ $a$ $b$ $b$ $a$ $a$

$O$: $b$ $b$ $a$ $a$
Allow Player \( O \) to delay her moves:

\[
(\alpha(0)) (\alpha(1)) \cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[\begin{array}{ccccccc}
I: & b & a & b & b & a & a & b \\
O: & b & b & a & a & b
\end{array}\]

O wins!
Allow Player $O$ to delay her moves:

$$
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
$$

$I$: $b$ $a$ $b$ $b$ $a$ $a$ $b$

$O$: $b$ $b$ $a$ $a$ $b$
Delay Games

Allow Player $O$ to delay her moves:

$$\begin{pmatrix} \alpha(0) \\ \beta(0) \end{pmatrix} \begin{pmatrix} \alpha(1) \\ \beta(1) \end{pmatrix} \ldots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i$$

$I$: b a b b a a b b
$O$: b b a a b
Allow Player $O$ to delay her moves:

\[
\begin{pmatrix}
\alpha(0) \\
\beta(0)
\end{pmatrix}
\begin{pmatrix}
\alpha(1) \\
\beta(1)
\end{pmatrix}
\cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\begin{align*}
I: & \quad b \ a \ b \ b \ a \ a \ b \ b \\
O: & \quad b \ b \ a \ a \ b \ b
\end{align*}
Delay Games

Allow Player $O$ to delay her moves:

\[
\left(\begin{array}{c}
\alpha(0) \\
\beta(0)
\end{array}\right) \left(\begin{array}{c}
\alpha(1) \\
\beta(1)
\end{array}\right) \ldots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i
\]

\[\begin{array}{cccccccc}
I: & b & a & b & b & a & a & b & b & \ldots \\
O: & b & b & a & a & b & b & \ldots \\
\end{array}\]

$O$ wins!
Delay Games

Allow Player $O$ to delay her moves:

$$(\alpha(0)) \left( \begin{array}{c} \alpha(1) \\ \beta(1) \end{array} \right) \cdots \in L, \text{ if } \beta(i) = \alpha(i + 2) \text{ for every } i$$

$I$: $b$ $a$ $b$ $b$ $a$ $a$ $b$ $b$ $\cdots$

$O$: $b$ $b$ $a$ $a$ $b$ $b$ $\cdots$

$O$ wins!

Typical questions:

- How often does Player $O$ have to delay to win?
- How hard is determining the winner of a delay game?
- Does the ability to delay allow Player $O$ to improve the quality of her strategies?
Previous Work

If winning conditions given by deterministic parity automata:

**Theorem (Klein, Z. ’15)**

- *If Player O wins delay game induced by \( A \), then also by delaying at most \( 2^{|A|^2} \) times.*
Previous Work

If winning conditions given by deterministic parity automata:

**Theorem (Klein, Z. ’15)**

- If Player $O$ wins delay game induced by $A$, then also by delaying at most $2^{|A|^2}$ times.
- Lower bound $2^{|A|}$ (already for safety automata).
Previous Work

If winning conditions given by deterministic parity automata:

**Theorem (Klein, Z. ’15)**

- *If Player O wins delay game induced by \( A \), then also by delaying at most \( 2|A|^2 \) times.*
- *Lower bound \( 2|A| \) (already for safety automata).*
- *Determining the winner is EXPTIME-complete (hardness already for safety automata).*
Previous Work

If winning conditions given by deterministic parity automata:

**Theorem (Klein, Z. ’15)**

- *If Player O wins delay game induced by A, then also by delaying at most $2|A|^2$ times.*
- *Lower bound $2|A|$ (already for safety automata).*
- *Determining the winner is EXPTIME-complete (hardness already for safety automata).*

**Note:**
This improved similar results by Holtmann, Kaiser, and Thomas with doubly-exponential upper bounds and no lower bounds.
Previous Work

If winning conditions given by formula in (quantitative) linear temporal logics:

**Theorem (Klein, Z. ’16)**

- If Player O wins delay game induced by $\varphi$, then also by delaying at most $2^{2|\varphi|}$ times.
- There is a matching lower bound.
- Determining the winner is 3EXPTIME-complete.

**Note:**
Quantitative conditions not harder than qualitative ones.
A strategy $\sigma$ for $O$ in a game induces a mapping $f_\sigma : \Sigma_I^\omega \rightarrow \Sigma_O^\omega$

$\sigma$ is winning $\iff \{ (f_\sigma(\alpha)) \mid \alpha \in \Sigma_I^\omega \} \subseteq L$  \hspace{1cm} ($f_\sigma$ uniformizes $L$)
Uniformization of Relations

- A strategy $\sigma$ for $O$ in a game induces a mapping $f_{\sigma} : \Sigma^\omega_I \rightarrow \Sigma^\omega_O$

- $\sigma$ is winning $\iff \{ (f_{\sigma}(\alpha)) \mid \alpha \in \Sigma^\omega_I \} \subseteq L$ (f$_{\sigma}$ uniformizes L)

Continuity in terms of strategies (in Cantor metric):

- Strategy without lookahead: $i$-th letter of $f_{\sigma}(\alpha)$ only depends on first $i$ letters of $\alpha$ (very strong notion of continuity).
Uniformization of Relations

- A strategy $\sigma$ for $O$ in a game induces a mapping $f_\sigma : \Sigma^\omega_I \rightarrow \Sigma^\omega_O$
- $\sigma$ is winning $\iff\{ (f_\sigma(\alpha)) \mid \alpha \in \Sigma^\omega_I \} \subseteq L$ \hspace{1cm} ($f_\sigma$ uniformizes $L$)

Continuity in terms of strategies (in Cantor metric):
- Strategy without lookahead: $i$-th letter of $f_\sigma(\alpha)$ only depends on first $i$ letters of $\alpha$ (very strong notion of continuity).
- Strategy with bounded delay: $f_\sigma$ Lipschitz-continuous.
Uniformization of Relations

- A strategy $\sigma$ for $O$ in a game induces a mapping $f_\sigma : \sum^\omega_i \rightarrow \sum^\omega_O$
- $\sigma$ is winning $\iff \{ (f_\sigma(\alpha)) \mid \alpha \in \sum^\omega_i \} \subseteq L$  ($f_\sigma$ uniformizes $L$)

Continuity in terms of strategies (in Cantor metric):
- Strategy without lookahead: $i$-th letter of $f_\sigma(\alpha)$ only depends on first $i$ letters of $\alpha$ (very strong notion of continuity).
- Strategy with bounded delay: $f_\sigma$ Lipschitz-continuous.
- Strategy with arbitrary (finite) delay: $f_\sigma$ (uniformly) continuous.
Uniformization of Relations

- A strategy $\sigma$ for $O$ in a game induces a mapping $f_\sigma : \Sigma^\omega_I \rightarrow \Sigma^\omega_O$
- $\sigma$ is winning $\iff \{ (f_\sigma(\alpha)) \mid \alpha \in \Sigma^\omega_I \} \subseteq L$ ($f_\sigma$ uniformizes $L$)

Continuity in terms of strategies (in Cantor metric):
- Strategy without lookahead: $i$-th letter of $f_\sigma(\alpha)$ only depends on first $i$ letters of $\alpha$ (very strong notion of continuity).
- Strategy with bounded delay: $f_\sigma$ Lipschitz-continuous.
- Strategy with arbitrary (finite) delay: $f_\sigma$ (uniformly) continuous.

Holtmann, Kaiser, Thomas: for $\omega$-regular $L$

\[
L \text{ uniformizable by continuous function } \iff
L \text{ uniformizable by Lipschitz-continuous function}
\]
Parity acceptance:

Almost every odd priority is followed by a larger even one.

$L(A) = a(b^* \alpha a)^* b^ω + \sum_{n \in \mathbb{N}} a(b^* \alpha a)^* b^ω$

Finitary parity acceptance:

There is a bound $n$ such that almost every odd priority is followed by a larger even one within $n$ steps.
Finitary Parity Automata

Parity acceptance: Almost every odd priority is followed by a larger even one.

\[ L(A) = a(b^* aaa)^* b^\omega + a(b^* aaa)^\omega \]
**Finitary Parity Automata**

**Parity acceptance:** Almost every odd priority is followed by a larger even one.

\[ L(A) = a(b^{\ast} {aaa})^{\ast} b^{\omega} + a(b^{\ast} {aaa})^{\omega} \]

**Finitary parity acceptance:** There is a bound \( n \) such that almost every odd priority is followed by a larger even one within \( n \) steps.

\[ L(A) = a(b^{\ast} {aaa})^{\ast} b^{\omega} + \sum_{n \in \mathbb{N}} a(b^{\leq n} {aaa})^{\omega} \]
Remark

Safety automata can be transformed into finitary parity automata of the same size.
Remark

Safety automata can be transformed into finitary parity automata of the same size.

Proof:

Turn all unsafe states into sinks with an odd color, all safe states get even color.
Remark

Safety automata can be transformed into finitary parity automata of the same size.

Proof:

Turn all unsafe states into sinks with an odd color, all safe states get even color.

Thus: exponential lower bounds on complexity and necessary lookahead for delay games with finitary parity conditions.
Results

If winning conditions given by deterministic finitary parity automata:

**Theorem**

- If Player O wins delay game induced by \( \mathcal{A} \), then also by delaying at most \( 2|\mathcal{A}|^6 \) times.
Results

If winning conditions given by deterministic finitary parity automata:

Theorem

- If Player O wins delay game induced by $A$, then also by delaying at most $2|A|^6$ times.
- Lower bound $2|A|$.
Results

If winning conditions given by deterministic finitary parity automata:

**Theorem**

- If Player O wins delay game induced by $A$, then also by delaying at most $2|A|^6$ times.
- Lower bound $2|A|$.
- Determining the winner is EXPTIME-complete.

**Note:**
Again, quantitative conditions not harder than qualitative ones.
Theorem

For every \( n > 0 \), there is a language \( L_n \) recognized by a finitary Büchi automaton with \( n + 2 \) states such that an optimal strategy without delay has cost \( n \), but an optimal strategy delaying once has cost \( 1 \).
**Theorem**

*For every* $n > 0$, *there is a language* $L_n$ *recognized by a finitary Büchi automaton with* $n + 2$ *states such that*

- *an optimal strategy without delay has cost* $n$, *but*
- *an optimal strategy delaying once has cost* $1$. 
Theorem
For every $n > 0$, there is a language $L'_n$ recognized by a finitary Büchi automaton with $O(n)$ states such that

- an optimal strategy delaying $2^n$ times has cost 0, and
- an optimal strategy delaying less than $2^n$ times has cost $n$. 
Theorem

For every $n > 0$, there is a language $L''_n$ recognized by a finitary Büchi automaton with $O(n^2)$ states such that for every $0 \leq j \leq n$:

$\text{an optimal strategy delaying } j \text{ times has cost } 2(n + 1) - j$. 

Theorem
For every $n > 0$, there is a language $L_n$ recognized by a finitary Büchi automaton with $O(n^2)$ states such that for every $0 \leq j \leq n$:
an optimal strategy delaying $j$ times has cost $2(n + 1) - j$. 

Martin Zimmermann  Saarland University  Games with Costs and Delays 12/14
Theorem

For every $n > 0$, there is a language $L'$ recognized by a finitary Büchi automaton with $O(n^2)$ states such that for every $0 \leq j \leq n$:

an optimal strategy delaying $j$ times has cost $2(n+1) - j$. 

Martin Zimmermann Saarland University Games with Costs and Delays 12/14
Theorem

For every $n > 0$, there is a language $L''_n$ recognized by a finitary Büchi automaton with $O(n^2)$ states such that for every $0 \leq j \leq n$:

an optimal strategy delaying $j$ times has cost $2(n+1) - j$.
Theorem
For every $n > 0$, there is a language $L_n$ recognized by a finitary Büchi automaton with $O(n^2)$ states such that for every $0 \leq j \leq n$:
an optimal strategy delaying $j$ times has cost $2(n + 1) - j$. 

Martin Zimmermann  
Saarland University  
Games with Costs and Delays  
12/14
Theorem

For every $n > 0$, there is a language $L''_n$ recognized by a finitary Büchi automaton with $\mathcal{O}(n^2)$ states such that for every $0 \leq j \leq n$: an optimal strategy delaying $j$ times has cost $2(n + 1) - j$. 
<table>
<thead>
<tr>
<th>acceptance</th>
<th>lookahead</th>
<th>complexity</th>
</tr>
</thead>
</table>

Theorem

Optimal strategies in delay games with Streett conditions with costs may require doubly-exponential lookahead.
<table>
<thead>
<tr>
<th>acceptance</th>
<th>lookahead</th>
<th>complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>parity</td>
<td>exp.</td>
<td><strong>EXPTIME</strong>-complete</td>
</tr>
<tr>
<td>finitary parity</td>
<td>exp.</td>
<td><strong>EXPTIME</strong>-complete</td>
</tr>
</tbody>
</table>

Theorem

Optimal strategies in delay games with Streett conditions with costs may require doubly-exponential lookahead.
More Results

<table>
<thead>
<tr>
<th>acceptance</th>
<th>lookahead</th>
<th>complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>parity</td>
<td>exp.</td>
<td><strong>EXPTIME</strong>-complete</td>
</tr>
<tr>
<td>finitary parity</td>
<td>exp.</td>
<td><strong>EXPTIME</strong>-complete</td>
</tr>
<tr>
<td>parity w. costs</td>
<td>exp.</td>
<td><strong>EXPTIME</strong>-complete</td>
</tr>
</tbody>
</table>

Theorem

Optimal strategies in delay games with Streett conditions with costs may require doubly-exponential lookahead.
Theorem
Optimal strategies in delay games with Streett conditions with costs may require doubly-exponential lookahead.

<table>
<thead>
<tr>
<th>acceptance</th>
<th>lookahead</th>
<th>complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>parity</td>
<td>exp.</td>
<td>EXPTIME-complete</td>
</tr>
<tr>
<td>finitary parity</td>
<td>exp.</td>
<td>EXPTIME-complete</td>
</tr>
<tr>
<td>parity w. costs</td>
<td>exp.</td>
<td>EXPTIME-complete</td>
</tr>
<tr>
<td>finitary Streett</td>
<td>exp./doubly-exp.</td>
<td>EXPTIME/2EXPTIME</td>
</tr>
<tr>
<td>Streett w. costs</td>
<td>exp./doubly-exp.</td>
<td>EXPTIME/2EXPTIME</td>
</tr>
</tbody>
</table>
### More Results

<table>
<thead>
<tr>
<th>Acceptance</th>
<th>Lookahead</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parity</td>
<td>Exp.</td>
<td><strong>EXPTIME</strong>-complete</td>
</tr>
<tr>
<td>Finitary Parity</td>
<td>Exp.</td>
<td><strong>EXPTIME</strong>-complete</td>
</tr>
<tr>
<td>Parity w. Costs</td>
<td>Exp.</td>
<td><strong>EXPTIME</strong>-complete</td>
</tr>
<tr>
<td>Finitary Streett</td>
<td>Exp./doubly-exp.</td>
<td><strong>EXPTIME/2EXPTIME</strong></td>
</tr>
<tr>
<td>Streett w. Costs</td>
<td>Exp./doubly-exp.</td>
<td><strong>EXPTIME/2EXPTIME</strong></td>
</tr>
</tbody>
</table>

### Theorem

*Optimal strategies in delay games with Streett conditions with costs may require doubly-exponential lookahead.*
Conclusion

- Quantitative delay games with parity conditions are not harder than qualitative ones.
- Lookahead allows to improve the quality of strategies.
Conclusion

- Quantitative delay games with parity conditions are not harder than qualitative ones.
- Lookahead allows to improve the quality of strategies.

Open Problems

- Close the gaps for Streett conditions (qualitative and quantitative).
- Study other tradeoffs, e.g., lookahead vs. memory size.
- Determine the complexity of finding optimal strategies (smallest cost or smallest lookahead).