Is your automaton good for playing games?

Marc Bagnol, Denis Kuperberg
CNRS, LIP, ENS Lyon

University of Warwick
30/10/2018
Games

Two Players:

- Eve
- Adam

Arena: finite graph $G = (V, E)$, with $V = V_a \cup V_b$.

Initial vertex: $v_0 \in V$.

Play: Infinite path: $v_0v_1v_2\cdots \in V_\omega$.

Winning Condition: $W \subseteq V_\omega$.

Eve wins a play $\pi$ if $\pi \in W$. 
Games

Two Players:

Eve

Adam

Arena: finite graph $G = (V, E)$, with $V = V_\circ \uplus V_\square$.

Initial vertex: $v_0 \in V$.

Play: Infinite path: $v_0 \rightarrow v_1 \rightarrow v_2 \cdots \in V_\omega$.

Winning Condition: $W \subseteq V_\omega$.

Eve wins a play $\pi$ if $\pi \in W$. 
Games

Two Players: Eve \quad Adam

Arena: finite graph $G = (V, E)$, with $V = V_0 \cup V_\square$.

Initial vertex: $v_0 \in V$. 

Diagram: A graph with vertices labeled $a$, $b$, and $c$, and edges connecting them.
Games

Two Players:

Arena: finite graph $G = (V, E)$, with $V = V_\circ \cup V_\blacksquare$.

Initial vertex: $v_0 \in V$.

Play: Infinite path: $v_0v_1v_2\cdots \in V^\omega$
Two Players: Eve, Adam

Arena: finite graph \( G = (V, E) \), with \( V = V_\bigcirc \cup V_\square \).

Initial vertex: \( v_0 \in V \).

Play: Infinite path: \( v_0 v_1 v_2 \cdots \in V^\omega \)

Winning Condition: \( W \subseteq V^\omega \).
Games

Two Players:

Eve
Adam

Arena: finite graph $G = (V, E)$, with $V = V_\circ \sqcup V_\square$.

Initial vertex: $v_0 \in V$.

Play: Infinite path: $v_0 v_1 v_2 \cdots \in V^\omega$

Winning Condition: $W \subseteq V^\omega$.

Eve wins a play $\pi$ if $\pi \in W$
ω-regular games

Winning condition $W$: an ω-regular language.

Example:

$W = (a^* ba^* c)^\omega$
ω-regular games

Winning condition $W$: an ω-regular language.

Example:

Why study these games?
$\omega$-regular games

Winning condition $W$: an $\omega$-regular language.

**Example:**

$$W = (a^* ba^* c)^\omega$$

Why study these games?

**Applications** in Logic, Verification, Proof theory, Complexity, ... 

*Church Synthesis Problem.*
Solving an \( \omega \)-regular game

**Input:** \( G \) game with \( \omega \)-regular winning condition \( W \subseteq V^\omega \).

**Question:** Who wins \( G \)? How?

**Solution:**
1. Build Det Parity automaton \( A_{Det} \) for \( W \),
2. Solve the parity game \( G' = A_{Det} \circ G \).

**Theorem:** \( A_{Det} \circ G \) has the same winner as \( G \).

**Problem:** Determinization is expensive. Maybe too strong?

**Definition (Henzinger, Piterman 2006):** \( A \) is Good-for-Games (GFG) if \( A \circ G \) has the same winner as \( G \), for any game \( G \) with winning condition \( L (A) \).

Also works for composition with infinite trees.
**Solving an $\omega$-regular game**

**Input:** $G$ game with $\omega$-regular winning condition $W \subseteq V^\omega$.

**Question:** Who wins $G$? How?

**Solution:**

1. Build Det Parity automaton $A_{Det}$ for $W$,
2. Solve the parity game $G' = A_{Det} \circ G$.

**Theorem**

$A_{Det} \circ G$ has same winner as $G$. 

Problem: Determinization is expensive. Maybe too strong?

Definition (Henzinger, Piterman 2006)

$A$ is Good-for-Games (GFG) if $A \circ G$ has same winner as $G$, for any game $G$ with winning condition $L(A)$.

Also works for composition with infinite trees.
Solving an \( \omega \)-regular game

**Input:** \( G \) game with \( \omega \)-regular winning condition \( W \subseteq V^\omega \).

**Question:** Who wins \( G \) ? How ?

**Solution:**

1. Build Det Parity automaton \( A_{Det} \) for \( W \),
2. Solve the parity game \( G' = A_{Det} \circ G \).

**Theorem**

\( A_{Det} \circ G \) has same winner as \( G \).

**Problem:** Determinization is expensive. Maybe too strong ?
Solving an $\omega$-regular game

**Input:** $G$ game with $\omega$-regular winning condition $W \subseteq V^\omega$.

**Question:** Who wins $G$? How?

**Solution:**

1. Build Det Parity automaton $A_{\text{Det}}$ for $W$,
2. Solve the parity game $G' = A_{\text{Det}} \circ G$.

**Theorem**

$A_{\text{Det}} \circ G$ has same winner as $G$.

**Problem:** Determinization is expensive. Maybe too strong?

**Definition (Henzinger, Piterman 2006)**

$A$ is Good-for-Games (GFG) if $A \circ G$ has same winner as $G$, for any game $G$ with winning condition $L(A)$. 
Solving an $\omega$-regular game

**Input:** $G$ game with $\omega$-regular winning condition $W \subseteq V^\omega$.

**Question:** Who wins $G$? How?

**Solution:**

1. Build Det Parity automaton $A_{\text{Det}}$ for $W$,
2. Solve the parity game $G' = A_{\text{Det}} \circ G$.

**Theorem**

$A_{\text{Det}} \circ G$ has same winner as $G$.

**Problem:** Determinization is expensive. Maybe too strong?

**Definition (Henzinger, Piterman 2006)**

$A$ is Good-for-Games (GFG) if $A \circ G$ has same winner as $G$, for any game $G$ with winning condition $L(A)$.

Also works for composition with infinite trees.
Definition of GFG via a game

A Parity automaton, we associate to it a **GFG game**:

Adam plays letters:

Eve: resolves non-deterministic choices for transitions
Definition of GFG via a game

A Parity automaton, we associate to it a **GFG game**:  
Adam plays letters:  \( a \)  
Eve: resolves non-deterministic choices for transitions
Definition of GFG via a game

A Parity automaton, we associate to it a **GFG game**:  
**Adam** plays letters: \( a \ a \)  
**Eve**: resolves non-deterministic choices for transitions
Definition of GFG via a game

A Parity automaton, we associate to it a GFG game:
Adam plays letters: $a$  $a$  $b$
Eve: resolves non-deterministic choices for transitions
Definition of GFG via a game

A Parity automaton, we associate to it a GFG game:

Adam plays letters: $a$ $a$ $b$ $c$

Eve: resolves non-deterministic choices for transitions
Definition of GFG via a game

A Parity automaton, we associate to it a **GFG game**:

Adam plays letters:  \( a \ a \ b \ c \ c \)

Eve: resolves non-deterministic choices for transitions
Definition of GFG via a game

A Parity automaton, we associate to it a **GFG game**: 

Adam plays letters: \(a \ a \ b \ c \ c \ldots = w\)

Eve: resolves non-deterministic choices for transitions

Eve wins if: \(w \in L \Rightarrow\) Run accepting.
Definition of GFG via a game

A Parity automaton, we associate to it a **GFG game**: Adam plays letters: \( a \ a \ b \ c \ c \ldots = w \)

Eve: resolves non-deterministic choices for transitions

\[ \begin{array}{c}
\text{a, b, c} \\
\text{a} \\
\text{b, c} \\
\text{b} \\
\text{c} \\
\text{a, b, c}
\end{array} \]

Eve wins if: \( w \in L \Rightarrow \) Run accepting.

A **GFG** \( \iff \) Eve wins the GFG game on \( A \).
Definition of GFG via a game

A Parity automaton, we associate to it a **GFG game**: Adam plays letters: \(a\ a\ b\ c\ c\ \ldots = w\)

Eve: resolves non-deterministic choices for transitions

Eve wins if: \(w \in L \Rightarrow \text{Run accepting.}\)

\(A \text{ GFG} \iff \text{Eve wins the GFG game on } A.\)

\(A \text{ GFG} \Rightarrow \text{GFG means that there is a strategy } \sigma_{\text{GFG}} : A^* \rightarrow Q, \text{ for accepting all words of } L(A).\)
A few facts on GFG automata

Fact
Every deterministic automaton is GFG.
A few facts on GFG automata

Fact
Every deterministic automaton is GFG.

Fact
There are some non-GFG automata: $(a + b)^* a^\omega$

Diagram:

```
1 ---- a ----> 2
```

Adam plays $aaa$... until Eve goes to 2, then $ba^\omega$.
A few facts on GFG automata

Fact
Every deterministic automaton is GFG.

Fact
There are some non-GFG automata:

Adam plays $aaa\ldots$ until Eve goes to 2, then $ba^\omega$. 

\[
\begin{array}{c}
\text{1} \\
\text{2}
\end{array}
\]

\[a, b \quad a \quad (a + b)^* a^\omega\]
A few facts on GFG automata

Fact

Every deterministic automaton is GFG.

Fact

There are some non-GFG automata: 

Adam plays $aaa\ldots$ until Eve goes to 2, then $ba^\omega$.

Fact

$\sigma_{\text{GFG}} \iff \text{Det automaton}$. 
A few facts on GFG automata

**Fact**

*Every deterministic automaton is GFG.*

**Fact**

*There are some non-GFG automata:*

\[
\begin{array}{c}
1 \\
\end{array} \xrightarrow{a} \begin{array}{c}
2 \\
\end{array}
\]

Adam plays \(aaa\ldots\) until Eve goes to 2, then \(ba^\omega\).

**Fact**

\(\sigma_{GFG} \leftrightarrow \text{Det automaton.}\)

**Theorem (Boker, K., Kupferman, Skrzypczak 2013)**

\(GFG\ for\ L + GFG\ for\ \overline{L} \rightarrow \text{Det for}\ L\)
A few facts on GFG automata

Fact
Every deterministic automaton is GFG.

Fact
There are some non-GFG automata: $1 \xrightarrow{a} 2 \quad (a + b)^* a^\omega$

Adam plays $aaa \ldots$ until Eve goes to 2, then $ba^\omega$.

Fact
$\sigma_{GFG} \iff \text{Det automaton.}$

Theorem (Boker, K., Kupferman, Skrzypczak 2013)
GFG for $L +$ GFG for $\bar{L} \rightarrow$ Det for $L$

Fact
If $A$ is GFG, no need to know its strategy $\sigma_{GFG}$ to solve $A \circ G$!
A few facts on GFG automata

Fact
Every deterministic automaton is GFG.

Fact
There are some non-GFG automata: \( (a + b)^* a^\omega \).

Adam plays aaa... until Eve goes to 2, then \( ba^\omega \).

Fact
\( \sigma_{GFG} \leftrightarrow \text{Det automaton} \).

Theorem (Boker, K., Kupferman, Skrzypczak 2013)
GFG for \( L \) + GFG for \( \overline{L} \) \( \rightarrow \) Det for \( L \)

Fact
If \( A \) is GFG, no need to know its strategy \( \sigma_{GFG} \) to solve \( A \circ G \)!
\( \rightarrow \) we can ignore the strategy and still use \( A \) as GFG in algorithms.
**GFG versus Deterministic**

**Lemma**: For Reachability/Safety conditions, \( \text{GFG} \approx \text{Det} \).
GFG versus Deterministic

Lemma: For Reachability/Safety conditions, $\text{GFG} \approx \text{Det}$.

Theorem (K., Skrzypczak 2015)

$\text{GFG}$ parity automata can be exponentially more succinct than deterministic ones.
**GFG versus Deterministic**

**Lemma**: For Reachability/Safety conditions, $\text{GFG} \approx \text{Det}$.

**Theorem (K., Skrzypczak 2015)**

*GFG parity automata can be exponentially more succinct than deterministic ones.*

$$L_n = \{ w \mid \text{Graph}(w) \text{ contains an } \infty \text{ path}\}.$$
**GFG versus Deterministic**

**Lemma:** For Reachability/Safety conditions, $\text{GFG} \approx \text{Det.}$

**Theorem (K., Skrzypczak 2015)**

**GFG** parity automata can be exponentially more succinct than deterministic ones.

$$w: \quad a \quad b \quad \# \quad a \quad \# \quad a \quad b \quad \# \quad \ldots$$

**Graph**($w$):

$$\begin{align*}
0 & \quad 0 & \quad 1 & \quad 1 & \quad 2 & \quad 2 & \quad 3 & \quad 3 & \quad 4 & \quad 4 & \quad 5 & \quad 5 & \quad \ldots \\
1 & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots \\
2 & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots \\
3 & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots
\end{align*}$$

**time:** 0 1 2 3 4 5 6 7 8 $\ldots$

$$L_n = \{w \mid \text{Graph}(w) \text{ contains an } \infty \text{ path} \}.$$  

Any Det Parity automaton for $L_n$ needs $\frac{2^n}{n+1}$ states!
A small GFG automaton for $L_n$

GFG coBüchi automaton:

Idea: Try paths one after the other.
Recognizing GFG automata

GFGness problem: input $A_{ND}$, is it GFG?
Recognizing GFG automata

GFGness problem: input $A_{\text{ND}}$, is it GFG?

Theorem (Löding)
The GFGness problem is in $P$ for reachability/safety automata.

Theorem (K., Skrzypczak 2015)
The GFGness problem is in $P$ for coBüchi automata.
Parity Games $\equiv$ GFGness for universal Parity automata.
Recognizing GFG automata

GFGness problem: input $A_{ND}$, is it GFG?

Theorem (Löding)
The GFGness problem is in $P$ for reachability/safety automata.

Theorem (K., Skrzypczak 2015)
The GFGness problem is in $P$ for coBüchi automata.
Parity Games $\equiv$ GFGness for universal Parity automata.

This talk:
Theorem (Bagnol, K. 2018)
The GFGness problem is in $P$ for Büchi automata.
Via the GFG game

Recall: $\mathcal{A}$ is GFG $\iff$ Eve wins the GFG game on $\mathcal{A}$. 

---

[Diagram description]

- Start state labeled $a, b, c$
- Transitions:
  - From $a, b, c$ to $a$
  - From $a$ to $b$
  - From $b$ to $c$
  - From $c$ back to $a, b, c$

---

Acceptance condition of the form "$u \in L \Rightarrow$ run accepting"
Via the GFG game

Recall: \( A \) is GFG \( \iff \) Eve wins the GFG game on \( A \).

Solve the GFG game?
Via the GFG game

Recall: $A$ is GFG $\iff$ Eve wins the GFG game on $A$.

Solve the GFG game?
Acceptance condition of the form “$u \in L \implies$ run accepting”
Via the GFG game

Recall: $A$ is GFG $\iff$ Eve wins the GFG game on $A$.

Solve the GFG game?
Acceptance condition of the form "$u \in L \Rightarrow$ run accepting"

→ We need a GFG automaton for $L$!
Via the GFG game

Recall: $A$ is GFG $\iff$ Eve wins the GFG game on $A$.

Solve the GFG game?

Acceptance condition of the form “$u \in L \Rightarrow \text{run accepting}$”

$\rightarrow$ We need a GFG automaton for $L$!

*Upper bound*: EXPTIME (by computing $A_{\text{Det}}$)
Abstracting the GFG game

The game $G_2$:

Adam plays letters:

Eve: moves one token  Adam: moves two tokens

Eve wins if:

1. or
2. accepts $\Rightarrow$ accepts.

Goal: Show that $Eve$ wins $G_2$ $\iff$ $Eve$ wins the GFG game.
Abstracting the GFG game

The game $G_2$:
- **Adam** plays letters: $a$
- **Eve**: moves one token  
  **Adam**: moves two tokens

Eve wins if:
1. or
2. accepts $\Rightarrow$ accepts.

Goal: Show that Eve wins $G_2$ if and only if Eve wins the GFG game.
Abstracting the GFG game

The game $G_2$:
Adam plays letters: $a$
Eve: moves one token Adam: moves two tokens

\begin{center}
\begin{tikzpicture}[node distance=1.5cm,auto,thick]
    \node (A0) {$a, b, c$};
    \node (A1) [right of=A0] {$a$};
    \node (A2) [right of=A1] {$b$};
    \node (A3) [right of=A2] {$c$};
    \node (A4) [right of=A3] {$a, b, c$};

    \path[->, bend left=30, red]
    (A0) edge (A1)
    (A1) edge (A2)
    (A2) edge (A3)
    (A3) edge (A4);
    \end{tikzpicture}
\end{center}
Abstracting the GFG game

The game $G_2$:
Adam plays letters: $a$, $a$
Eve: moves one token Adam: moves two tokens

Eve wins if:
1. or
2. accepts $\Rightarrow$ accepts.

Goal: Show that Eve wins $G_2 \iff$ Eve wins the GFG game.
Abstracting the GFG game

The game $G_2$:

Adam plays letters: $a \ a$

Eve: moves one token  Adam: moves two tokens

Eve wins if:

1 or 2 accepts $\Rightarrow$ accepts.

Goal: Show that Eve wins $G_2 \iff$ Eve wins the GFG game.
Abstracting the GFG game

The game $G_2$:
Adam plays letters: $a \ a \ b$
Eve: moves one token Adam: moves two tokens

Eve wins if:
1 or 2 accepts $\Rightarrow$ accept.

Goal: Show that Eve wins $G_2$ $\iff$ Eve wins the GFG game.
Abstracting the GFG game

The game $G_2$:

**Adam** plays letters: $a$, $a$, $b$

**Eve**: moves one token **Adam**: moves two tokens

\[ a, b, c \]

\[ a \]

\[ a, b, c \]

\[ b \]

\[ b, c \]

\[ c \]
Abstracting the GFG game

The game $G_2$:
Adam plays letters: $a \ a \ b \ c$
Eve: moves one token Adam: moves two tokens

Goal: Show that Eve wins $G_2 \iff$ Eve wins the GFG game.
Abstracting the GFG game

The game $G_2$:
- **Adam** plays letters: $a, a, b, c$
- **Eve**: moves one token  
  **Adam**: moves two tokens

Eve wins if:
1. or 2. accepts $\implies$ accepts.

Goal: Show that Eve wins $G_2$ $\iff$ Eve wins the GFG game.
Abstracting the GFG game

The game $G_2$:

- **Adam** plays letters: $a$ $a$ $b$ $c$ $\ldots = w$
- **Eve**: moves one token  
  **Adam**: moves two tokens

Eve wins if:  
1 or 2 accepts $\Rightarrow$ 1 accepts.

---

Eve wins if: □₁ or □₂ accepts $\Rightarrow$ □ accept
Abstracting the GFG game

The game $G_2$:

Adam plays letters: $a\ a\ b\ c\ \ldots\ =\ w$

Eve: moves one token  Adam: moves two tokens

\[ a, b, c \quad \quad a \quad \quad b \quad \quad a, b, c \]

\[ b, c \quad \quad c \]

Eve wins if: $\square_1$ or $\square_2$ accepts $\Rightarrow$ $\bigcirc$ accepts.

Goal: Show that Eve wins $G_2 \iff$ Eve wins the GFG game.
Proof sketch

Lemma

\(Eve\ wins\ G_2 \iff Eve\ wins\ G_k\ for\ all\ k.\)

Proof sketch: \(G_2 \implies G_3\)

- Play a virtual token \(\bullet\) against the first 2 \(\square\)
- Play \(G_2\) strategy against \(\bullet\) and last \(\square\)
Main proof sketch for $G_2 \iff \text{GFG}$

Assume:

- Adam wins the GFG game with finite-memory strategy $\tau_{\text{GFG}}$.
- Eve wins $G_2 \Rightarrow$ wins $G_k$ with strategy $\sigma_k$, for a big $k$. 
Main proof sketch for $G_2 \Leftrightarrow \text{GFG}$

Assume:

- Adam wins the GFG game with finite-memory strategy $\tau_{\text{GFG}}$.
- Eve wins $G_2 \Rightarrow$ wins $G_k$ with strategy $\sigma_k$, for a big $k$.

Build strategy for Eve against $\tau_{\text{GFG}}$:

- move $k$ virtual tokens $\bigcirc$ against $\tau_{\text{GFG}}$
- play $\sigma_k$ against these $k$ tokens

$\text{at most } M \text{ steps}$

$\geq N \text{ tokens}$

$\text{Büchi state}$

$\text{initial state}$
Building GFG automata

From a language $L$, how to obtain $A_{GFG}$ for $L$?
Building GFG automata

From a language $L$, how to obtain $A_{\text{GFG}}$ for $L$?

**Incremental construction** [K., Majumdar 2018]:
Partial $k$-Determinization of $A_{\text{ND}}$
Increase $k$ until $A$ is GFG.
Worst case: $k = |A|$, reach the full determinization construction.
Building GFG automata

From a language $L$, how to obtain $A_{GFG}$ for $L$ ?

**Incremental construction** [K., Majumdar 2018]:
Partial $k$-Determinization of $A_{ND}$
Increase $k$ until $A$ is GFG.
Worst case: $k = |A|$, reach the full determinization construction.

For **efficiency**: solution to the GFGness problem is needed !
So for now only efficient for coBüchi (and Büchi)
Building GFG automata

From a language $L$, how to obtain $A_{GFG}$ for $L$?

**Incremental construction** [K., Majumdar 2018]:
Partial $k$-Determinization of $A_{ND}$
Increase $k$ until $A$ is GFG.
Worst case: $k = |A|$, reach the full determinization construction.

For efficiency: solution to the GFGness problem is needed!
So for now only efficient for coBüchi (and Büchi)

**Directly from LTL formula** $\varphi$ [losti, K. (unpublished)]
Adapted for Church Synthesis
For now: Fragment of coBüchi languages
Worst-case blowup result: GFG is doubly exponential in $|\varphi|$.

Heuristic and modular, can be combined with other approaches.
Conclusion

Results

- **GFG** automata are intermediate between **Det** and **ND**.
- They are useful to solve complex games.
- They can bring an exponential advantage compared to determinism.
- For Büchi and coBüchi conditions, they are recognized efficiently.

Perspectives

- $G_2 \iff$ GFG for Parity automata ?
- Complexity of the Parity GFGness Problem ?
- New ways to build **GFG** automata ?
- Generalization to other models (alternating, pushdown, . . . ) ?