Games and Automata: From Boolean to Quantitative Verification

- Habilitation thesis defense -

Laurent Doyen
CNRS

ENS Cachan, March 13th, 2012
Outline

- Antichain Algorithms
  Finite automata, Büchi automata, alternating automata, partial-observation games, QBF
- Quantitative Games
  Energy games, mean-payoff games, partial-observation, energy parity, multi-dimensional
- Quantitative Languages
  Automata-based model, complexity, expressiveness, closure properties, mean-payoff automaton expression

Context and perspective of a selection of results
Model-checking

\[ M \models \varphi \]

Check if a Model \textit{satisfies} a Property?

...in an \textit{automated} way

[Clarke, Emerson, Pnueli, Sifakis,...]
Model-checking

What kind of properties?
Model-checking

What kind of properties?

Avoid failures!
Model-checking

What kind of properties?

Ensure responsiveness!
What kind of models?
Model-checking

What kind of models?

Reactive systems:
- Non-terminating
- Safety-critical
- Data abstraction
Model-checking

\[ M \models \varphi \]
Example
Example

\[ r \in \{r_1, r_2\} \quad g \in \{g_1, g_2\} \quad \models \varphi \]
Server

\[ r \in \{r_1, r_2\} \]

\[ g \in \{g_1, g_2\} \]

\[ 1 \rightarrow 2 \]

\[ \Rightarrow \]

« Every request is eventually granted, no simultaneous grants »
Example

Every request is eventually granted, no simultaneous grants

\[ M \models \varphi \]

\omega\text{-automaton}
Example

Every request is eventually granted, no simultaneous grants

Closure properties
Expressiveness
Decidability

ω-automaton
Example

Every request is eventually granted, no simultaneous grants

Closure properties
Expressiveness
Decidability

ω-automaton

\( \square (r_i \rightarrow \Diamond g_i) \land \square \neg (g_1 \land g_2) \)

Translation to automata

LTL
Example

For every request, there is eventually granted, no simultaneous grants.

Translation to automata

Closure properties
Expressiveness
Decidability

$\omega$-automaton

Trace inclusion

LTL

Yes/No answer
Example

- Server
- \( g \in \{g_1, g_2\} \)
- \( r \in \{r_1, r_2\} \)

« Every request is eventually granted, no simultaneous grants »

Closure properties
Expressiveness
Decidability

\( \omega \)-automaton

LTL

\( M \models \varphi \)

\( \square(r_i \rightarrow \Diamond g_i) \land \square \neg(g_1 \land g_2) \)

Translation to automata

Automata-based approach to model-checking

[Vardi, Wolper,...]
Outline

From Boolean to quantitative verification
Outline

From **Boolean** to **quantitative** verification

- **Boolean** automata-based Verification
  1. Techniques to speed up well-known verification algorithms by orders of magnitude

- **Quantitative** Verification
  2. A surprising complexity result in game theory
  3. A robust and decidable class of quantitative languages
Algorithm ?

\[ M \models \varphi \quad L(M) \subseteq L(\varphi) \]
Algorithm ?

\[ M \models \varphi \quad L(M) \subseteq L(\varphi) \]

Translation to automata

\[ L(M) \subseteq L(A_\varphi) \]
Algorithm?

\[
M \models \varphi \quad L(M) \subseteq L(\varphi)
\]

Translation to automata

\[
L(M) \subseteq L(A_\varphi)
\]

\[
L(M) \cap L(A_\varphi)^c = \emptyset
\]

Closure properties

\[
L(M \times A_\varphi^c) = \emptyset
\]
Algorithm?

\[ M \models \varphi \quad L(M) \subseteq L(\varphi) \]

Translation to automata

\[ L(M) \subseteq L(A_\varphi) \]

\[ L(M) \cap L(A_\varphi)^c = \emptyset \]

Closure properties

\[ L(M \times A_\varphi^c) = \emptyset \]

This problem is PSPACE-complete
Translation to automata

\[ M \models \varphi \quad L(M) \subseteq L(\varphi) \]

\[ L(M) \subseteq L(A_\varphi) \]

\[ L(M) \cap L(A_\varphi)^c = \emptyset \]

Closure properties

\[ L(M \times A_\varphi^c) = \emptyset \]

This problem is PSPACE-complete

even if \( A_\varphi \) is given explicitly, even over finite words, and even if \( L(M) = \Sigma^* \)

\[ L(A_\varphi^c) = \emptyset \]
Efficient Algorithm?

(over finite words) \( L(A^c) = \emptyset \)

iff there is no path from initial to accepting states in \( A^c \).
Efficient Algorithm?

(over finite words) \( L(A^c) = \emptyset \)

iff there is no path from initial to accepting states in \( A^c \).

Subset construction (state-explosion problem)
Subset Construction

\[ A \]

\[
\begin{align*}
1 & \xrightarrow{a} 2 \\
1 & \xrightarrow{a, b} 2 \\
2 & \xrightarrow{a, b} 2 \\
2 & \xrightarrow{a} 3 \\
3 & \xrightarrow{b} 4 \\
3 & \xrightarrow{a} 2 \\
4 & \xrightarrow{a, b} 4 \\
4 & \xrightarrow{a, b} 4
\end{align*}
\]

\[ L(A^c) \equiv \emptyset \]
Subset Construction

$L(A^c) \equiv \emptyset$

\[
\begin{align*}
A & \\
1 & \xrightarrow{a} 1 \\
2 & \xrightarrow{a, b} 2 \\
3 & \xrightarrow{a} 2 \\
3 & \xrightarrow{b} 4 \\
4 & \xrightarrow{a, b} 4
\end{align*}
\]

\[
\{1\} \xrightarrow{?} \subseteq \{3, 4\}
\]
Subset Construction

$L(A^c) \equiv \emptyset$

\[
\begin{array}{ccccccc}
A & \longrightarrow & 1 & \overset{a}{\longrightarrow} & 2 & \overset{a, b}{\longrightarrow} & 3 \overset{a}{\longrightarrow} & 4 \overset{a, b}{\longrightarrow} & \{3, 4\} \\
& & \{1\} & \overset{?}{\longrightarrow} & \{1, 2\} & \{1\} & \overset{b}{\longrightarrow} & \{2\}
\end{array}
\]
Subset Construction

\[ L(A^c) \equiv \emptyset \]

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

\[
\begin{array}{c}
1 \xrightarrow{a} 2 \xrightarrow{a,b} 3 \xrightarrow{b} 4 \\
2 \xrightarrow{a,b} 1 \\
3 \xrightarrow{a} 2 \\
4 \xrightarrow{a,b} 3 \\
\end{array}
\]
Subset Construction

\[ L(A^c) \cong \emptyset \]
Subset Construction

\[ L(A^c) \equiv \emptyset \]
Subset Construction

\[ L(A^c) \equiv \emptyset \]
Pruning is sound: either

\[ \{1, 2\} \rightarrow \{3, 4\} \] or
Subset Construction

\[ L(A^c) \equiv \emptyset \]

Pruning is sound: either

- \( \{1, 2\} \rightarrow \{3, 4\} \) or
- \( \{1\} \rightarrow \{3, 4\} \)
Subset Construction

\[ A \]

\[ \begin{array}{cccc}
1 & \overset{a}{\rightarrow} & 2 & \overset{a, b}{\rightarrow} \\
a, b & \rightarrow & a & \overset{a}{\rightarrow} \\
& \rightarrow & b & \overset{a}{\rightarrow} \\
& & & \rightarrow \\
& & & \rightarrow \\
4 & \overset{a, b}{\rightarrow}
\end{array} \]

\[ L(A^c) \equiv \emptyset \]

\[ \{1\} \overset{?}{\rightarrow} \subseteq \{3, 4\} \]

\[ \begin{array}{cccc}
\{1\} & \overset{a}{\rightarrow} & \{2\} & \overset{\times}{\rightarrow} \\
& \overset{b}{\rightarrow} & \{2\} & \overset{b}{\rightarrow} \\
& & \{2\} & \rightarrow \\
& & \rightarrow \\
& & \rightarrow
\end{array} \]
Subset Construction

\[ L(A^c) \equiv \emptyset \]

\[
\begin{array}{c}
A \\
\begin{array}{c}
1 \\
2 \\
3 \\
4 \\
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\{1\} \\
\{2\} \\
\{3, 4\} \\
\{1\} \\
\{2\} \\
\{3, 4\} \\
\end{array}
\]

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

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

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

\[
\begin{array}{c}
a \\
\text{?} \\
\subseteq \{3, 4\} \\
a \\
b \\
\end{array}
\]

\[
\begin{array}{c}
a \\
\text{?} \\
\subseteq \{3, 4\} \\
a \\
b \\
\end{array}
\]
Subset Construction

$L(A^c) \equiv \emptyset$

Hence, $L(A^c) = \emptyset$
Is there a (finite) path from Init to Final?
Reachability

Is there a (finite) path from Init to Final?
Structure in graphs

Init

Final
Structure in graphs

Graph is partially ordered...
Structure in graphs

Graph is monotone...
Structure in graphs

\[ \{1, 2\} \quad \cdots \quad \cdots \quad \overset{w}{\longrightarrow} \quad \cdots \]

\[ \{1\} \quad \cdots \quad \cdots \quad \overset{w}{\longrightarrow} \quad \cdots \]

Key property
Structure in graphs

\[
\begin{array}{cccccc}
\{1, 2\} & \cdot & \cdot & \cdot & w & \cdot \\
\{1\} & \cdot & \cdot & \cdot & w & \cdot \\
\end{array}
\]

Two interpretations:

\(\subseteq\) is a **forward** simulation relation in \(A^c\)

\(\subseteq\) is a **backward** simulation relation in \(A^c\)
Structure in graphs

\{1, 2\} \quad \cdots \quad \cdots \quad \overset{w}{\rightarrow} \quad \cdots \\
\{1\} \quad \cdots \quad \cdots \quad \overset{w}{\rightarrow} \quad \cdots \\

Key property

Two interpretations:

\subseteq \text{ is a forward simulation relation in } A^c

Use \subseteq \text{ to prune the search}
Structure in graphs

\[
\{1, 2\} \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \\
\{1\} \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \downarrow^{w} \\
\text{Key property}
\]

Two interpretations:

\( \subseteq \) is a forward simulation relation in \( A^c \)

Use \( \subseteq \) to prune the search
Structure in graphs

\{1, 2\} \quad \therefore \quad \therefore \quad \therefore \quad \xrightarrow{w} \quad \therefore \quad \therefore \quad \therefore \quad \therefore \quad \therefore \\
\{1\} \quad \therefore \quad \therefore \quad \therefore \quad \xrightarrow{w} \quad \therefore \\

Key property

Two interpretations:

\subseteq \text{ is a forward simulation relation in } A^c

Use \subseteq \text{ to prune the search}
Structure in graphs

\{
1, 2\}

\begin{array}{cccccccc}
\cup & \cup & \cup & \cup & \cup & \cup & \cup & \cup \\
\end{array}

![](image)

Key property

Two interpretations:

\( \subseteq \) is a \textbf{forward} simulation relation in \( A^c \)

Use \( \subseteq \) to \textbf{prune} the search
Structure in graphs

\{1, 2\} \quad \ldots \quad \ldots \quad \ldots \quad \xrightarrow{w} \quad \ldots \\
\{1\} \quad \ldots \quad \ldots \quad \xrightarrow{w} \quad \ldots \\

Key property

Two interpretations:

\subseteq is a forward simulation relation in \( A^c \)

Use \( \subseteq \) to prune the search
Structure in graphs

Two interpretations:

\( \subseteq \) is a **forward** simulation relation in \( A^c \)

Use \( \subseteq \) to **prune** the search
Structure in graphs

Two interpretations:

$\subseteq$ is a **forward** simulation relation in $A^c$

Use $\subseteq$ to **prune** the search
Structure in graphs

Two interpretations:
\( \subseteq \) is a \textit{forward} simulation relation in \( A^c \)

- Use \( \subseteq \) to \textit{prune} the search
- Antichain of \textit{promising} states
Structure in graphs

\[
\{1, 2\} \quad \:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\\subseteq \:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\\subseteq 

\text{Key property}

Two interpretations:

\subseteq \text{ is a forward simulation relation in } A^c

\subseteq \text{ is a backward simulation relation in } A^c
Structure in graphs

\[
\begin{align*}
\{1, 2\} & \quad \cup \quad \cup \quad \cup \quad \overset{w}{\rightarrow} \quad \cup \\
\{1\} & \quad \cup \quad \cup \quad \overset{w}{\rightarrow} \quad \cup
\end{align*}
\]

Two interpretations:

\(\subseteq\) is a \textbf{forward} simulation relation in \(A^c\)

\(\subseteq\) is a \textbf{backward} simulation relation in \(A^c\)

iff \(\text{post}(.)\) preserves \(\subseteq\)-upward closure

\(\text{post}^*(.)\) computes a sequence of \(\subseteq\)-upward sets
Structure in graphs

Two interpretations:

\( \subseteq \) is a forward simulation relation in \( A^c \)

\( \subseteq \) is a backward simulation relation in \( A^c \)

iff \( \text{post}(.) \) preserves \( \subseteq \)-upward closure

\( \text{post}^*(.) \) computes a sequence of \( \subseteq \)-upward sets
Structure in graphs

\[
\begin{align*}
\{1, 2\} & \quad \uparrow \quad \uparrow \quad \uparrow \quad w \quad \rightarrow \quad . \\
\{1\} & \quad \uparrow \quad \uparrow \quad \uparrow \quad w \quad \rightarrow \quad .
\end{align*}
\]

Key property

Two interpretations:
\(\subseteq\) is a **forward** simulation relation in \(A^c\)

\(\subseteq\) is a **backward** simulation relation in \(A^c\)

iff post(.) preserves \(\subseteq\)-upward closure

post*(.) computes a sequence of \(\subseteq\)-upward sets
Structure in graphs

\{1, 2\} \quad \cdot \quad \cdot \quad \cdot \quad \xrightarrow{w} \quad \cdot \\
\{1\} \quad \cdot \quad \cdot \quad \cdot \quad \xrightarrow{w} \quad \cdot \\
\subseteq \quad \cdot \quad \cdot \quad \cdot \\
\subseteq \quad \cdot \\

Two interpretations:

\subseteq \text{ is a forward simulation relation in } A^c

\subseteq \text{ is a backward simulation relation in } A^c

\text{iff post}(.) \text{ preserves } \subseteq\text{-upward closure}

\text{post}^*(.) \text{ computes a sequence of } \subseteq\text{-upward sets}
Structure in graphs

\[
\begin{array}{cccccc}
\{1, 2\} & . & . & . & \xrightarrow{w} & . \\
\{1\} & . & . & . & \xrightarrow{w} & \cup I
\end{array}
\]

Key property

Two interpretations:

\(\subseteq\) is a forward simulation relation in \(A^c\)

\(\subseteq\) is a backward simulation relation in \(A^c\)

iff post(.) preserves \(\subseteq\)-upward closure

post*(.) computes a sequence of \(\subseteq\)-upward sets
Structure in graphs

Two interpretations:

\( \subseteq \) is a **forward** simulation relation in \( A^c \)

\( \subseteq \) is a **backward** simulation relation in \( A^c \)

iff post(.) preserves \( \subseteq \)-upward closure

post*(.) computes a sequence of \( \subseteq \)-upward sets

Antichains as a **symbolic** representation
(minimal elements)
Structure in graphs

Two interpretations:

\( \subseteq \) is a **forward** simulation relation in \( A^c \)

\( \subseteq \) is a **backward** simulation relation in \( A^c \)

iff \( \text{post}(.) \) preserves \( \subseteq \)-upward closure

\( \text{post}^*(.) \) computes a sequence of \( \subseteq \)-upward sets

Antichains as a **symbolic** representation

(minimal elements)
Structure in graphs

\[
\begin{align*}
\{1, 2\} & \quad \cup \quad \cup \quad \cup \\
\{1\} & \quad \cup \quad \cup \quad \cup
\end{align*}
\]

Key property

Two interpretations:

\(\subseteq\) is a **forward** simulation relation in \(A^c\)

\(\subseteq\) is a **backward** simulation relation in \(A^c\)

iff \(\text{post}(\cdot)\) preserves \(\subseteq\)-upward closure

\(\text{post}^*(\cdot)\) computes a sequence of \(\subseteq\)-upward sets

Antichains as a **symbolic** representation (minimal elements)
Two interpretations:

\( \subseteq \) is a **forward** simulation relation in \( A^c \)

\( \subseteq \) is a **backward** simulation relation in \( A^c \)

iff post(.) preserves \( \subseteq \)-upward closure

post*(.) computes a sequence of \( \subseteq \)-upward sets

Antichains as a **symbolic** representation
(minimal elements)
Structure in graphs

Two interpretations:

\( \subseteq \) is a **forward** simulation relation in \( A^c \)

\( \subseteq \) is a **backward** simulation relation in \( A^c \)

iff \( \text{post}(.) \) preserves \( \subseteq \)-upward closure

\( \text{post}^*(.) \) computes a sequence of \( \subseteq \)-upward sets

Antichains as a **symbolic** representation
(minimal elements)
Structure in graphs

Two interpretations:

\( \subseteq \) is a **forward** simulation relation in \( A^c \)

Promising states

\( \subseteq \) is a **backward** simulation relation in \( A^c \)

Symbolic representation
Structure in graphs

Two interpretations:

\(\subseteq\) is a \textbf{forward} simulation relation in \(A^c\)

Promising states

\(\subseteq\) is a \textbf{backward} simulation relation in \(A^c\)

Symbolic representation

Here the two interpretations coincide!
Two interpretations:

\( \subseteq \) is a **forward** simulation relation in \( A^c \)

Promising states

\( \subseteq \) is a **backward** simulation relation in \( A^c \)

Symbolic representation

Works with ANY **forward** simulation!

Works with ANY **backward** simulation!
## Antichains everywhere!

Partial-observation Reachability/Parity games

| Finite automata (language inclusion, universality) | HSCC’06, CSL’06, CONCUR’08, Inf&Comp’10 |
| Büchi automata (language inclusion, universality) | CAV’06 |
| LTL satisfiability and model-checking | TACAS’07, LMCS’09 |
| QBF | TACAS’08 |
|  | ATVA’11 |

...
Antichains everywhere!

Partial-observation Reachability/Parity games

Finite automata (language inclusion, universality)
Büchi automata (language inclusion, universality)
LTL satisfiability and model-checking
QBF

HSCC’06, CSL’06,
CONCUR’08, Inf&Comp’10
CAV’06
TACAS’07, LMCS’09
TACAS’08
ATVA’11

J-F. Raskin  M. De Wulf  N. Maquet  T. Henzinger  D. Berwanger
Antichains everywhere!

Partial-observation Reachability/Parity games
Finite automata (language inclusion, universality)
Büchi automata (language inclusion, universality)
LTL satisfiability and model-checking
QBF

HSCC’06, CSL’06, CONCUR’08, Inf&Comp’10
CAV’06
TACAS’07, LMCS’09
TACAS’08
ATVA’11

...

Finite Tree Automata [Bouajjani et al. 08]
Program Termination [Vardi et al. 09]
Minimizing Alternating Büchi [Abdulla et al. 09]
LTL synthesis [Raskin et al. 09]
Büchi universality [Vardi et al. 10]
Simulation Subsumption [Abdulla et al. 10,11]
Tools

ALASKA
LTL SATISFIABILITY, MODEL CHECKING AND ALTERNATING AUTOMATA EMPTINESS USING
ATVA’08

Alpaca
A TOOL SOLVING IMPERFECT INFORMATION PARITY GAMES USING
TACAS’09

Acacia
LTL REALIZABILITY CHECK AND WINNING STRATEGY SYNTHESIS USING
Raskin et al.

http://www.antichains.be
Tools

NFA universality
Tools

Reachability/Parity games with imperfect information

Finite automata (language inclusion, universality)
Büchi automata (language inclusion, universality)

LTL satisfiability and model checking

LTL synthesis

50 times faster than nuSMV...

LTL model-checking
Outline

From **Boolean** to **quantitative** verification

- **Boolean** Verification
  1. Techniques to speed up well-known verification algorithms by orders of magnitude

- **Quantitative** Verification
  2. A surprising complexity result in game theory
  3. A robust and decidable class of quantitative languages
Model-checking

\[ M \models \varphi \]

Check if a Model satisfies a Property?

...in an automated way

[Clarke, Emerson, Sifakis, ...]
Check if a Model satisfies a Property?

...in an automated way

[Clarke, Emerson, Sifakis,...]
From graphs to games

« Every request is eventually granted, no simultaneous grants »
From graphs to games

« Every request is eventually granted, no simultaneous grants »

(Part of) the Model is not given
From graphs to games

(Part of) the Model is not given

Construct a correct system
(typically reduces to game solving)

[Church, Büchi, Landweber, Rabin, Pnueli,...]
From graphs to games

"Every request is eventually granted, no simultaneous grants"

(Part of) the Model is not given

\[\text{Construct a correct system}
\]

(typically reduces to game solving)

[Church, Büchi, Landweber, Rabin, Pnueli, ...]
From Boolean to Quantitative spec

« Every request is eventually granted, no simultaneous grants »

Solution 1: grant within $10^6$ years
Solution 2: grant even if no request
From Boolean to Quantitative spec

Every request is eventually granted, no simultaneous grants

Solution 1: grant within $10^6$ years

Solution 2: grant even if no request

Boolean specs do not distinguish correct systems
From Boolean to Quantitative spec

« Every request is eventually granted, no simultaneous grants »

Solution 1: grant within $10^6$ years

Solution 2: grant even if no request

Switch to Quantitative Spec

« Minimize delays for pending requests, minimize number of grants »
From Boolean to Quantitative spec

Wrong solution 1: no grant at all
Wrong solution 2: 99% request granted

Boolean specs do not distinguish wrong systems either!
From Boolean to Quantitative spec

- Server
- Clients

« Every request is eventually granted, no simultaneous grants »

Wrong solution 1: no grant at all
Wrong solution 2: 99% request granted

Switch to Quantitative Spec « Maximize average number of granted requests »
From Boolean to...

Boolean acceptance conditions separate good and bad runs:

\[ \{0,1\}^\omega \rightarrow \{0,1\} \]

E.g., (co)Büchi, Muller, parity, etc.
From Boolean to...

Boolean acceptance conditions separate good and bad runs:

\[ \{0,1\}^\omega \rightarrow \{0,1\} \]

E.g., (co)Büchi, Muller, parity, etc.

Quantitative value functions assign value to runs:

\[ \mathbb{R}^\omega \rightarrow \mathbb{R} \]
Some value functions

For $v = v_0v_1 \ldots (v_i \in \mathbb{R})$, let

- $\text{Sup}(v) = \sup\{v_n \mid n \geq 0\}$;
- $\text{LimSup}(v) = \limsup_{n \to \infty} v_n$;
- $\text{LimInf}(v) = \liminf_{n \to \infty} v_n$;

($v_i \in \{0,1\}$) (reachability) (Büchi) (coBüchi)
Some value functions

For $v = v_0 v_1 \ldots \ (v_i \in \mathbb{R})$, let

- $\text{Sup}(v) = \sup \{v_n \mid n \geq 0\}$;  \hspace{1cm} (reachability)

- $\text{LimSup}(v) = \limsup_{n \to \infty} v_n$;  \hspace{1cm} (Büchi)

- $\text{LimInf}(v) = \liminf_{n \to \infty} v_n$;  \hspace{1cm} (coBüchi)

- $\text{LimAvg}(v) = \limsup_{n \to \infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i$; aka $\text{MeanPayoff}(v)$

- given $0 < \lambda < 1$, $\text{Disc}_\lambda(v) = \sum_{i=0}^{\infty} \lambda^i \cdot v_i$.  

Outline

From **Boolean** to **quantitative** verification

- **Boolean Verification**
  1. Techniques to speed up well-known verification algorithms by orders of magnitude

- **Quantitative Verification**
  2. Mean-payoff parity games are in $NP \cap coNP$
  3. A robust and decidable class of quantitative languages
Example

Mean-payoff parity games
Example

Mean-payoff \textit{parity} games

\( \omega \)-regular specifications
  (reactivity, liveness,...)
Mean-payoff parity games

ω-regular specifications (reactivity, liveness,...)

• Memoryless strategies
• NP \cap \text{coNP}
Example

Mean-payoff parity games

Quantitative specification (cost optimization,...)

- Memoryless strategies
- NP \cap coNP

\omega-regular specifications (reactivity, liveness,...)

- Memoryless strategies
- NP \cap coNP
Example

Mean-payoff Büchi games

Visit $q_0$ infinitely often, and maximize mean-payoff
Example

Mean-payoff Büchi games

Visit $q_0$ infinitely often, and maximize mean-payoff

Optimal strategy: spend more and more time in $q_1$

0, $-1$, 0, 0, $-1$, 0, 0, 0, $-1$, 0, 0, 0, 0, $-1$, 0, 0, 0, 0, 0, $-1$, 0, ...

Requires infinite memory...
Example

Mean-payoff parity games

- Memoryless strategies
- still in \( \text{NP} \cap \text{coNP} \)
Example

Mean-payoff parity games

- Memoryless strategies
- still in NP $\cap$ coNP

1. Reduction to parity games with positive counter
2. Finite-memory strategies suffice
Example

Mean-payoff parity games

- Memoryless strategies
- still in $\text{NP} \cap \text{coNP}$

1. Reduction to parity games with positive counter
2. Finite-memory strategies suffice
3. Winning strategies can be decomposed into memoryless strategies, and combined using counters.
4. Decomposition can be guessed in NP
Example

Mean-payoff parity games

- Memoryless strategies
- still in $\text{NP} \cap \text{coNP}$

ICALP'10

K. Chatterjee
Outline

From *Boolean* to *quantitative* verification

- **Boolean Verification**
  1. Techniques to speed up well-known verification algorithms by orders of magnitude

- **Quantitative Verification**
  2. Mean-payoff parity games are in $\text{NP} \cap \text{coNP}$
  3. A robust and decidable class of quantitative languages
Quantitative Languages
Long-term goal

Is there a Quantitative Framework with
- an appealing mathematical formulation,
- useful expressive power, robustness and
- good algorithmic properties?

(Like the boolean theory of $\omega$-regularity.)

Note: “Quantitative” is more than “timed” and “probabilistic”

[Henzinger,...]
A quantitative language is a function:

\[ L : \Sigma^\omega \rightarrow \mathbb{R} \]

\(L(w)\) can be interpreted as:

- the amount of some resource needed by the system to produce \(w\) (power, energy, time consumption),
- a reliability measure (the average number of “faults” in \(w\)).
A **quantitative** language is a function:

\[ L : \sum^\omega \rightarrow \mathbb{R} \]

Classical **Boolean** languages are the special case where

\[ L : \sum^\omega \rightarrow \{0, 1\} \]

\( L(w) \) can be interpreted as:

- the amount of some resource needed by the system to produce \( w \) (power, energy, time consumption),
- a reliability measure (the average number of “faults” in \( w \)).
Languages & Automata

Boolean languages are generated by finite automata.
Languages & Automata

**Boolean** languages are generated by **finite** automata.

\[ q_0 \xrightarrow{a} q_1 \]

**Quantitative** languages are generated by **weighted** automata,

\[ L_A(w) = \]

- A is deterministic: value of (unique) run
- A is non-deterministic: \( \sup \) of run values
- A is universal: \( \inf \) of run values
- A is alternating: value of game-outcome run (\( \sup \inf \))

\[ q_0 \xrightarrow{\gamma = 3} q_1 \]
## Quantitative Languages

<table>
<thead>
<tr>
<th></th>
<th>det.</th>
<th>nondet.</th>
<th>univ.</th>
<th>alt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sup</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimSup</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimInf</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimAvg</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disc_\lambda</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

20 classes of quantitative languages...
Quantitative Languages

1. Decision problems
2. Expressiveness
3. Closure properties
Decision problems

Given weighted automata $A$, $B$ and $\nu \in \mathbb{Q}$
decide

Quant. emptiness $\exists w : L_A(w) \geq \nu$
Quant. universality $\forall w : L_A(w) \geq \nu$
Given weighted automata $A$, $B$ and $\nu \in \mathbb{Q}$ decide

Quant. emptiness $\exists w : L_A(w) \geq \nu$

Quant. universality $\forall w : L_A(w) \geq \nu$

Quant. inclusion $\forall w : L_A(w) \leq L_B(w)$

Quant. equivalence $\forall w : L_A(w) = L_B(w)$
Decision problems

Given weighted automata $A, B$ and $\nu \in \mathbb{Q}$

decide

Quant. emptiness $\exists w : L_A(w) \geq \nu$

Quant. universality $\forall w : L_A(w) \geq L_B(w)$

Quant. inclusion $\forall w : L_A(w) \leq L_B(w)$

Quant. equivalence $\forall w : L_A(w) = L_B(w)$
Decision problems

Given weighted automata $A$, $B$ and $\nu \in \mathbb{Q}$ decide

Quant. emptiness $\exists w : L_A(w) \geq \nu$
Quant. universality $\forall w : L_A(w) \geq \nu$
Quant. inclusion $\forall w : L_A(w) \leq L_B(w)$
Quant. equivalence $\forall w : L_A(w) = L_B(w)$

Undecidable for LimAvg.
Open question for Disc.

CSL’08, CSL’10, ToCL’10
Quantitative Languages

1. Decision problems
2. Expressiveness
3. Closure properties
Expressiveness

Compare classes of quantitative languages defined by weighted automata

$O(20 \times 20)$ comparisons...
Expressiveness

Compare classes of quantitative languages defined by weighted automata

$O(20 \times 20)$ comparisons...

\textbf{LimAvg} and \textbf{Disc}_\lambda cannot be determinized.

LICS'09, LMCS'10
Quantitative Languages

1. Decision problems
2. Expressiveness
3. Closure properties
Operations

L₁, L₂ : \Sigma^\omega \rightarrow \mathbb{R}

Operations on quantitative languages:

- \text{max}(L₁,L₂) \quad L₁ \cup L₂
- \text{min}(L₁,L₂) \quad L₁ \cap L₂
- \text{complement}(L₁) = 1-L₁ \quad \Sigma^\omega \setminus L₁
- L₁ + L₂
Operations on quantitative languages:

- \( \max(L_1, L_2) \)
- \( \min(L_1, L_2) \)
- \( \text{complement}(L_1) = 1 - L_1 \)
- \( L_1 + L_2 \)

Note \( L_1 \leq L_2 \) iff \( L_1 + (1 - L_2) \leq 1 \)
### LimAvg Automata

<table>
<thead>
<tr>
<th>LimAvg</th>
<th>Closure properties</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>max</td>
</tr>
<tr>
<td>Deterministic</td>
<td>×</td>
</tr>
<tr>
<td>Nondeterministic</td>
<td>✓</td>
</tr>
<tr>
<td>Alternating</td>
<td>✓</td>
</tr>
</tbody>
</table>

LICS’09, FCT’09
## LimAvg Automata

<table>
<thead>
<tr>
<th>LimAvg</th>
<th>Closure properties</th>
<th>Decision problems</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>max</td>
<td>min</td>
</tr>
<tr>
<td>Deterministic</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Nondeterministic</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Alternating</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

LICS’09, FCT’09
Beyond Weighted Automata
## LimAvg Automata

<table>
<thead>
<tr>
<th>LimAvg</th>
<th>Closure properties</th>
<th></th>
<th>Decision problems</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>max</td>
<td>min</td>
<td>Sum</td>
</tr>
<tr>
<td>Deterministic</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Nondeterministic</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Alternating</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Expressions</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
LimAvg Automaton Expressions

LimAvg-automaton expressions are defined by:

\[ E ::= A \mid \text{max}(E,E) \mid \text{min}(E,E) \mid \text{Sum}(E,E) \]

where \( A \) is a deterministic LimAvg-automaton.
LimAvg Automaton Expressions

LimAvg-automaton expressions are defined by:

\[ E ::= A \mid \max(E,E) \mid \min(E,E) \mid \text{Sum}(E,E) \]

where \( A \) is a deterministic LimAvg-automaton.

E.g.: \( \max(A_1 + A_2, \min(A_3, A_4)) \)
LimAvg Automaton Expressions

LimAvg-automaton expressions are defined by:

\[ E ::= A \mid \max(E,E) \mid \min(E,E) \mid \text{Sum}(E,E) \]

where \( A \) is a deterministic LimAvg-automaton.

Closure properties:

<table>
<thead>
<tr>
<th>LimAvg</th>
<th>Closure properties</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>max</td>
</tr>
<tr>
<td>Deterministic</td>
<td>×</td>
</tr>
<tr>
<td>Nondeterministic</td>
<td>✓</td>
</tr>
<tr>
<td>Alternating</td>
<td>✓</td>
</tr>
<tr>
<td>Expressions</td>
<td>✓</td>
</tr>
</tbody>
</table>
LimAvg Automaton Expressions

LimAvg-automaton expressions are defined by:

\[ E ::= A \mid \max(E, E) \mid \min(E, E) \mid \text{Sum}(E, E) \]

where \( A \) is a deterministic LimAvg-automaton.

**Decision problems:** all questions reduce to quant. emptiness

\[ \exists w : E(w) \geq \nu \]
Solve decision problems using the value set:

E.g.: \( E = \max(A_1 + A_2, \min(A_3, A_4)) \)

Value Set = \{ (L_{A_1}(w), L_{A_2}(w), L_{A_3}(w), L_{A_4}(w)) \mid w \in \Sigma^\omega \} \subseteq \mathbb{R}^4

How to compute this set?
Value set

Solve decision problems using the value set:

E.g.: \( E = \max(A_1 + A_2, \min(A_3, A_4)) \)

Value Set = \{ (L_{A_1}(w), L_{A_2}(w), L_{A_3}(w), L_{A_4}(w)) \mid w \in \Sigma^\omega \} \subseteq \mathbb{R}^4

How to compute this set?

Uses arguments in computational geometry, yields 4EXPTIME complexity for emptiness.
Solve decision problems using the value set:

E.g.: \( E = \max(A_1 + A_2, \min(A_3, A_4)) \)

Value Set = \{ (L_{A_1}(w), L_{A_2}(w), L_{A_3}(w), L_{A_4}(w)) | w \in \Sigma^\omega \} \subseteq \mathbb{R}^4

\( E(\Sigma^\omega) = \{ \max(x+y, \min(z,t)) | (x,y,z,t) \in \text{Value Set} \} \)

is a finite union of intervals.

Find maximum of \( E(\Sigma^\omega) \) to solve emptiness
# LimAvg Automaton Expressions

<table>
<thead>
<tr>
<th>LimAvg</th>
<th>Closure properties</th>
<th>Decision problems</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>max</td>
<td>min</td>
</tr>
<tr>
<td>Deterministic</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Nondeterministic</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Alternating</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Expressions</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

## Closure properties
- **LimAvg-automaton expression**
- **Quant. inclusion**
- **LimAvg-automaton expression**

## Expressiveness

## Decidability

\[ M \models \varphi \]
# LimAvg Automaton Expressions

<table>
<thead>
<tr>
<th>LimAvg</th>
<th>Closure properties</th>
<th>Decision problems</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>max</td>
<td>min</td>
</tr>
<tr>
<td>Deterministic</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Nondeterministic</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Alternating</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Expressions</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Closure properties**
- Maximality
- Minimality
- Sum
- Complementation

**Decision problems**
- Emptiness
- Universality
- Inclusion
- Equivalence

---

**CONCUR’10**
- K. Chatterjee
- H. Edelsbrunner
- T. Henzinger
- P. Rannou
Conclusion
Conclusion – Key results

1. Efficient antichain algorithms

2. Quantitative games
   Mean-payoff parity games in $\text{NP} \cap \text{coNP}$

3. Quantitative generalization of languages
   LimAvg automaton expressions: robust and decidable
1. Efficient **antichain** algorithms

Can we predict the performance of antichain algorithms?

Complexity theory beyond worst-case...
2. Quantitative games

Mean-payoff parity games in $\text{NP} \cap \text{coNP}$

- Multi-dimensional mean-payoff games – complexity
- New classes of quantitative stochastic games
  in progress, PhD thesis of Mahsa Shirmohammadi
- New classes of games on counter systems
  in progress, PhD thesis of Julien Reichert
3. Quantitative generalization of languages

LimAvg automaton expressions: robust and decidable

- Discounted-sum “expressions”?
- Incorporate Boolean conditions
- Theory of quantitative regularity
  - analogous of Borel hierarchy
  - safety vs. liveness
  - logical characterization
The work in this thesis has been carried out in the following teams:

- Tom Henzinger (EPFL, 2006-2008)
- Jean-François Raskin (ULB, 2009)
- Alain Finkel (LSV, 2009-now)
Credits

With the following co-authors (students in blue):

- Dietmar Berwanger
- Thomas Brihaye
- Lubos Brim
- Véronique Bruyère
- Jakub Chaloupka
- Krishnendu Chatterjee
- Aldric Degorre
- Martin De Wulf
- Marc Ducobu
- Herbert Edelsbrunner
- Gilles Geeraerts
- Raffaella Gentilini
- Hugo Gimbert
- Tom Henzinger
- Barbara Jobstmann
- Axel Legay
- Nicolas Maquet
- Nicolas Markey
- Thierry Massart
- Dejan Nickovic
- Joël Ouaknine
- Tatjana Petrov
- Sangram Raje
- Philippe Rannou
- Jean-François Raskin
- Julien Reichert
- Mahsa Shirmohammadi
- Rohit Singh
- Szymon Torunczyk
- James Worrell
Thank you!

Questions?