Deterministic ω-Automata for LTL: A safraless, compositional, and mechanically verified construction

Javier Esparza[1] Jan Křetínský[2] Salomon Sickert[1]

[1]Fakultät für Informatik, Technische Universität München, Germany

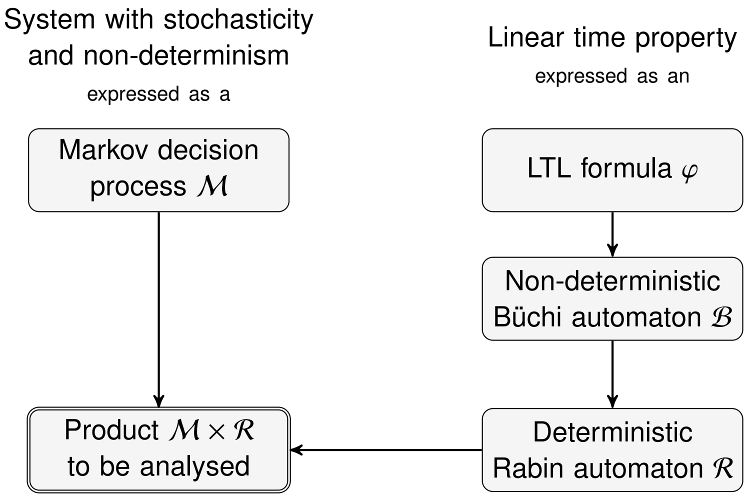
[2]IST Austria

May 11, 2015

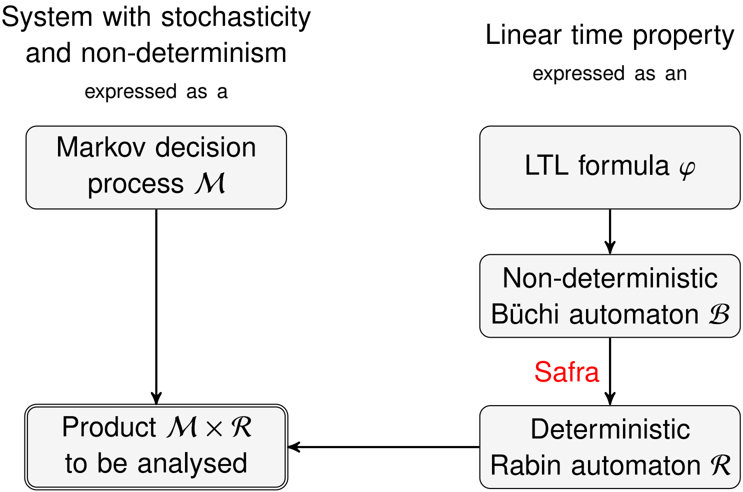
Deterministic ω-Automata for LTL:
A safraless, compositional, and mechanically verified construction

Deterministic ω-Automata for LTL:

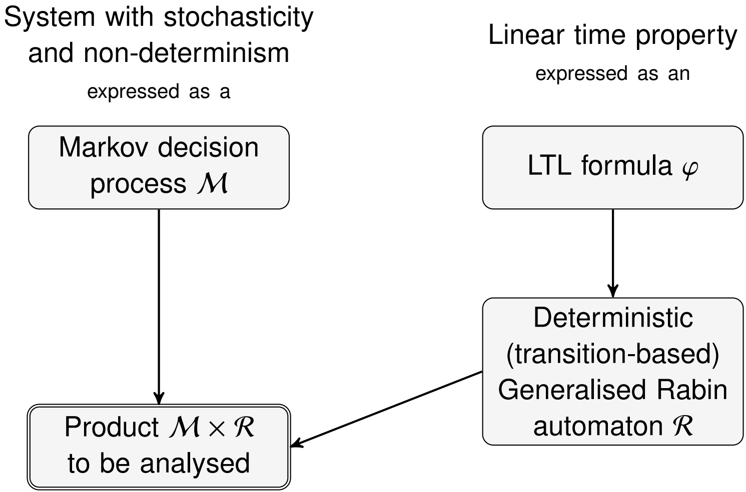
A safraless, compositional, and mechanically verified construction


System with stochasticity and non-determinism
expressed as a
Markov decision process $\mathcal{M}$
Linear time property
expressed as an
LTL formula φ
Non-deterministic Büchi automaton $\mathcal{B}$
Deterministic Rabin automaton $\mathcal{R}$
Product $\mathcal{M} \times \mathcal{R}$ to be analysed

Deterministic ω-Automata for LTL:
A safraless, compositional, and mechanically verified construction



Deterministic ω-Automata for LTL:
A safraless, compositional, and mechanically verified construction



Deterministic ω-Automata for LTL:

A safraless, compositional, and mechanically verified construction

Deterministic ω-Automata for LTL:
A safraless, compositional, and mechanically verified construction

- Directly yields a deterministic system

Deterministic ω-Automata for LTL:
A safraless, compositional, and mechanically verified construction

- Directly yields a deterministic system
- Product of several automata

Deterministic ω-Automata for LTL:
A safraless, compositional, and mechanically verified construction

- Directly yields a deterministic system
- Product of several automata
- Logical structure of the input formula is preserved
  - e.g.: “Which **G**-subformulae are eventually true?”

Deterministic ω-Automata for LTL:
A safraless, compositional, and mechanically verified construction

- Directly yields a deterministic system
- Product of several automata
- Logical structure of the input formula is preserved
  - e.g.: "Which **G**-subformulae are eventually true?"
- *Smaller* Systems[1]

[1] In most cases according to our experimental data; compared to the standard approach

Deterministic ω-Automata for LTL:
A safraless, compositional, and mechanically verified construction

---

- Directly yields a deterministic system
- Product of several automata
- Logical structure of the input formula is preserved
  - e.g.: “Which **G**-subformulae are eventually true?”
- *Smaller* Systems[1]
- Bonus: Construction and correctness theorem verified in Isabelle/HOL

[1] In most cases according to our experimental data; compared to the standard approach

Deterministic ω-Automata for LTL: A safraless, compositional, and mechanically verified construction

- Directly yields a deterministic system
- Product of several automata
- Logical structure of the input formula is preserved
  - e.g.: “Which **G**-subformulae are eventually true?”
- *Smaller* Systems[1]
- Bonus: Construction and correctness theorem verified in Isabelle/HOL with code extraction 50% done

[1] In most cases according to our experimental data; compared to the standard approach

Experimental Data

$\bigwedge_{i\in\{1,\ldots,n\}} \mathbf{GF}a_i \Rightarrow \mathbf{GF}b_i$	NBA LTL2BA	DRA ltl2dstar	DTGRA Rabinizer 3
$n = 1$	4		
$n = 2$	14		
$n = 3$	40		

Experimental Data

$\bigwedge_{i\in\{1,\ldots,n\}} \mathbf{GF}a_i \Rightarrow \mathbf{GF}b_i$	NBA `LTL2BA`	DRA `ltl2dstar`	DTGRA Rabinizer 3
$n = 1$	4	4	
$n = 2$	14	$> 10^4$	
$n = 3$	40	$> 10^6$	

Experimental Data

$\bigwedge_{i \in \{1,\ldots,n\}} \mathbf{GF} a_i \Rightarrow \mathbf{GF} b_i$	NBA	DRA	DTGRA
	LTL2BA	ltl2dstar	Rabinizer 3
$n = 1$	4	4	1
$n = 2$	14	$> 10^4$	1
$n = 3$	40	$> 10^6$	1

ω-Words and LTL

An ω-word is an infinite sequence: $w = a_0 a_1 a_2 a_3 \ldots$.

ω-Words and LTL

An ω-word is an infinite sequence: $w = a_0 a_1 a_2 a_3 \ldots$.

Definition (LTL Semantics, Negation-Normal-Form)

$$
\begin{array}{lllll}
\square & \models & \square & :: & \alpha\ \mathit{set\ word} \to \alpha\ \mathit{ltl} \to \mathbb{B} \\
w & \models & \mathbf{tt} & = & \mathit{True} \\
w & \models & \mathbf{ff} & = & \mathit{False} \\
w & \models & a & = & a \in w_0 \\
w & \models & \neg a & = & a \notin w_0 \\
w & \models & \varphi \wedge \psi & = & w \models \varphi \wedge w \models \psi \\
w & \models & \varphi \vee \psi & = & w \models \varphi \vee w \models \psi
\end{array}
$$

ω-Words and LTL

An ω-word is an infinite sequence: $w = a_0 a_1 a_2 a_3 \ldots$.

Definition (LTL Semantics, Negation-Normal-Form)

$$
\begin{array}{lllll}
\square & \models & \square & :: & \alpha\ \mathit{set}\ \mathit{word} \rightarrow \alpha\ \mathit{ltl} \rightarrow \mathbb{B} \\
w & \models & \mathbf{tt} & = & \mathit{True} \\
w & \models & \mathbf{ff} & = & \mathit{False} \\
w & \models & a & = & a \in w_0 \\
w & \models & \neg a & = & a \notin w_0 \\
w & \models & \varphi \wedge \psi & = & w \models \varphi \wedge w \models \psi \\
w & \models & \varphi \vee \psi & = & w \models \varphi \vee w \models \psi \\
w & \models & \mathbf{F}\varphi & = & \exists k.\ w_{k\infty} \models \varphi \\
w & \models & \mathbf{G}\varphi & = & \forall k.\ w_{k\infty} \models \varphi \\
w & \models & \psi\mathbf{U}\varphi & = & \exists k.\ w_{k\infty} \models \varphi \wedge \forall j < k.\ w_{j\infty} \models \psi \\
w & \models & \mathbf{X}\varphi & = & w_{1\infty} \models \varphi
\end{array}
$$

ω-Words and LTL

An ω-word is an infinite sequence: $w = a_0 a_1 a_2 a_3 \ldots$.

Definition (LTL Semantics, Negation-Normal-Form)

$$
\begin{array}{lllll}
\square & \models & \square & :: & \alpha \textit{ set word} \to \alpha \textit{ ltl} \to \mathbb{B} \\
w & \models & \mathbf{tt} & = & \textit{True} \\
w & \models & \mathbf{ff} & = & \textit{False} \\
w & \models & a & = & a \in w_0 \\
w & \models & \neg a & = & a \notin w_0 \\
w & \models & \varphi \wedge \psi & = & w \models \varphi \wedge w \models \psi \\
w & \models & \varphi \vee \psi & = & w \models \varphi \vee w \models \psi \\
w & \models & \mathbf{F}\varphi & = & \exists k.\ w_{k\infty} \models \varphi \quad \checkmark \\
w & \models & \mathbf{G}\varphi & = & \forall k.\ w_{k\infty} \models \varphi \\
w & \models & \psi\mathbf{U}\varphi & = & \exists k.\ w_{k\infty} \models \varphi \wedge \forall j < k.\ w_{j\infty} \models \psi \quad \checkmark \\
w & \models & \mathbf{X}\varphi & = & w_{1\infty} \models \varphi \quad \checkmark
\end{array}
$$

ω-Words and LTL

An ω-word is an infinite sequence: $w = a_0 a_1 a_2 a_3 \ldots$.

Definition (LTL Semantics, Negation-Normal-Form)

$$
\begin{array}{llllll}
\square & \models & \square & :: & \alpha\ \mathit{set\ word} \rightarrow \alpha\ \mathit{ltl} \rightarrow \mathbb{B} & \\
w & \models & \mathbf{tt} & = & \mathit{True} & \\
w & \models & \mathbf{ff} & = & \mathit{False} & \\
w & \models & a & = & a \in w_0 & \\
w & \models & \neg a & = & a \notin w_0 & \\
w & \models & \varphi \wedge \psi & = & w \models \varphi \wedge w \models \psi & \\
w & \models & \varphi \vee \psi & = & w \models \varphi \vee w \models \psi & \\
w & \models & \mathbf{F}\varphi & = & \exists k.\ w_{k\infty} \models \varphi & \checkmark \\
w & \models & \mathbf{G}\varphi & = & \forall k.\ w_{k\infty} \models \varphi & \times \\
w & \models & \psi\mathbf{U}\varphi & = & \exists k.\ w_{k\infty} \models \varphi \wedge \forall j < k.\ w_{j\infty} \models \psi & \checkmark \\
w & \models & \mathbf{X}\varphi & = & w_{1\infty} \models \varphi & \checkmark
\end{array}
$$

Unfolding Modal Operators

$$\begin{aligned} \mathbf{F}\varphi &\equiv \mathbf{X}\mathbf{F}\varphi \vee \varphi \\ \mathbf{G}\varphi &\equiv \mathbf{X}\mathbf{G}\varphi \wedge \varphi \\ \psi\mathbf{U}\varphi &\equiv \varphi \vee (\psi \wedge \mathbf{X}(\psi\mathbf{U}\varphi)) \end{aligned}$$

Co-Büchi Automata for **G**-free φ

$$\varphi = a \vee (b \mathbf{U} c)$$

Co-Büchi Automata for **G**-free φ

$$\varphi = a \vee (b\ \mathbf{U}\ c)$$

φ

Co-Büchi Automata for $\mathbf{G}$-free φ

$$\varphi = a \vee (b \; \mathbf{U} \; c)$$

$$\varphi \to a \vee c \vee (b \wedge \mathbf{X}(b\mathbf{U}c))$$

Co-Büchi Automata for **G**-free φ

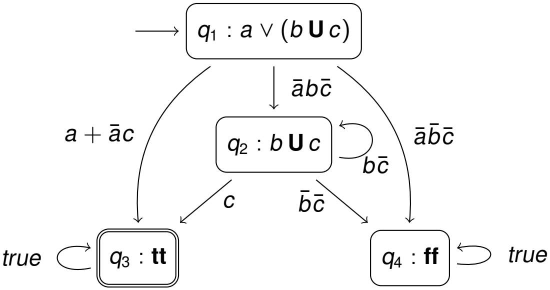
$$\varphi = a \vee (b \,\mathbf{U}\, c)$$

$$\varphi \rightarrow a \vee c \vee (b \wedge \mathbf{X}(b\mathbf{U}c)) \rightarrow_{\bar{a}b\bar{c}} b\mathbf{U}c$$

Co-Büchi Automata for **G**-free φ

$$\varphi = a \vee (b\ \mathbf{U}\ c)$$

$$\varphi \to a \vee c \vee (b \wedge \mathbf{X}(b\mathbf{U}c)) \to_{\bar{a}b\bar{c}} b\mathbf{U}c$$

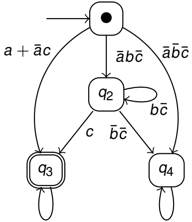


Tackling the **G**-Operator

- Relaxed case: $\mathbf{FG}\varphi$
  - $w \models \mathbf{FG}\varphi$ *iff* $w_{i\infty} \models \varphi$ for almost all i
- Reason: **G**-subformulae may be nested inside $\mathbf{X}, \mathbf{F}, \mathbf{U}$.

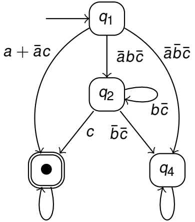
Automata for **FG**φ where φ is **G**-free

$$w = \ \ldots$$



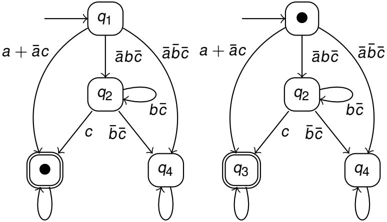
Automata for $\mathbf{FG}\varphi$ where φ is $\mathbf{G}$-free

$$w = abc \ \ldots$$



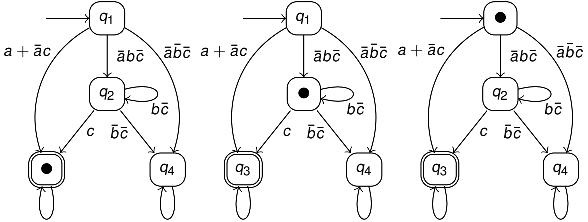
Automata for **FG**φ where φ is **G**-free

$$w = abc \ \ldots$$



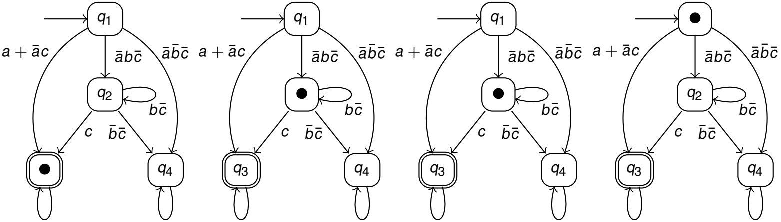
Automata for $\mathbf{FG}\varphi$ where φ is $\mathbf{G}$-free

$$w = abc\ \bar{a}b\bar{c}\ \ldots$$



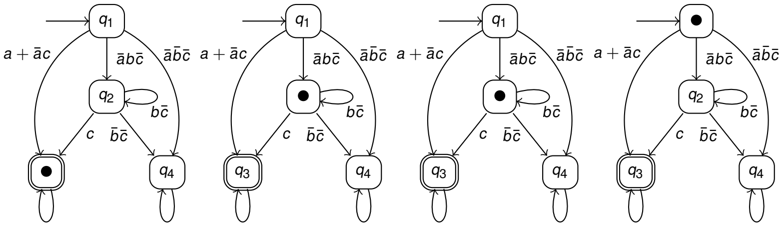
Automata for $\mathbf{FG}\varphi$ where φ is $\mathbf{G}$-free

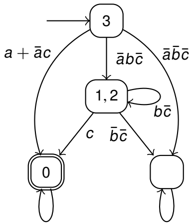
$$w = abc\ \bar{a}b\bar{c}\ \bar{a}b\bar{c}\ \ldots$$



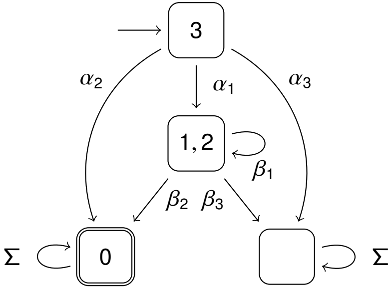
Automata for $\mathbf{FG}\varphi$ where φ is $\mathbf{G}$-free

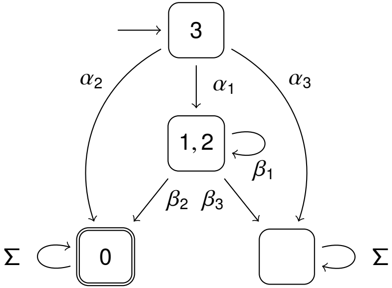
$$w = abc\ \bar{a}b\bar{c}\ \bar{a}b\bar{c}\ \ldots$$





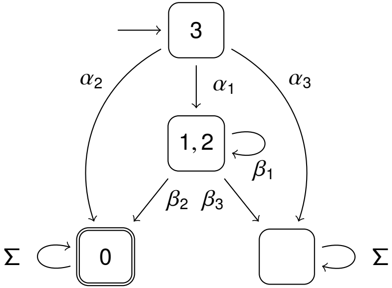
Mojmir Automata





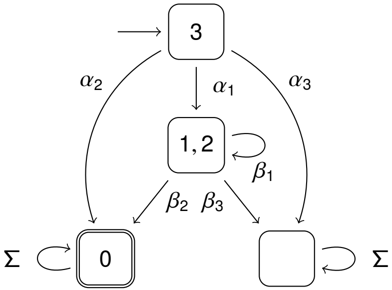
- In every step a new token is placed in the initial state and all other tokens are moved according to the transition function.

Mojmir Automata



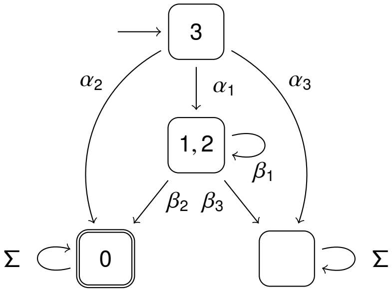
- In every step a new token is placed in the initial state and all other tokens are moved according to the transition function.
- Deterministic

Mojmir Automata



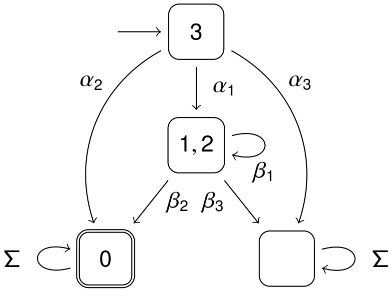
- In every step a new token is placed in the initial state and all other tokens are moved according to the transition function.
- Deterministic
- Accepts an ω-word w *iff* almost all tokens reach the final states

Mojmir Automata



- In every step a new token is placed in the initial state and all other tokens are moved according to the transition function.
- Deterministic
- Accepts an ω-word w *iff* almost all tokens reach the final states
  - Mojmir automata are "blind" to events that only happen finitely often

Mojmir Automata



- In every step a new token is placed in the initial state and all other tokens are moved according to the transition function.
- Deterministic
- Accepts an ω-word *w iff* almost all tokens reach the final states
  - Mojmir automata are "blind" to events that only happen finitely often

Going Further

- From Mojmir to Rabin Automata

Going Further

- From Mojmir to Rabin Automata
  - Unbounded number of tokens?

Going Further

- From Mojmir to Rabin Automata
  - Unbounded number of tokens?
    Abstraction with ranking functions for states and tokens

Going Further

- From Mojmir to Rabin Automata
  - Unbounded number of tokens?
    Abstraction with ranking functions for states and tokens
  - Mojmir acceptance ($\overset{\infty}{\forall}$) vs. Rabin acceptance (finite, $\overset{\infty}{\exists}$)?

Going Further

- From Mojmir to Rabin Automata
  - Unbounded number of tokens?
    Abstraction with ranking functions for states and tokens
  - Mojmir acceptance ($\overset{\infty}{\forall}$) vs. Rabin acceptance (finite, $\overset{\infty}{\exists}$)?
    Alternative definition for Mojmir acceptance

Going Further

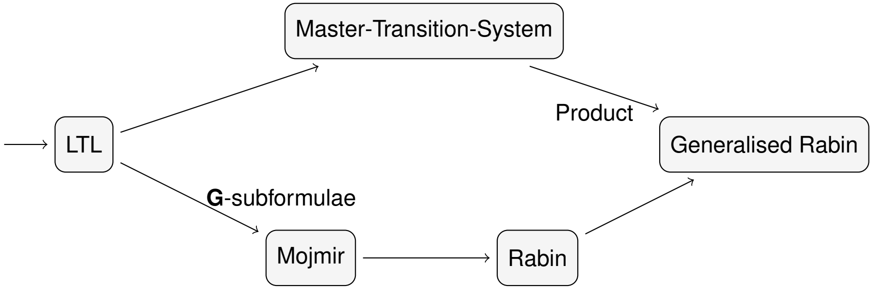
- From Mojmir to Rabin Automata
  - Unbounded number of tokens?
    Abstraction with ranking functions for states and tokens
  - Mojmir acceptance ($\overset{\infty}{\forall}$) vs. Rabin acceptance (finite, $\overset{\infty}{\exists}$)?
    Alternative definition for Mojmir acceptance
- Mojmir Automata for $\mathbf{FG}\varphi$ for arbitrary φ

Going Further

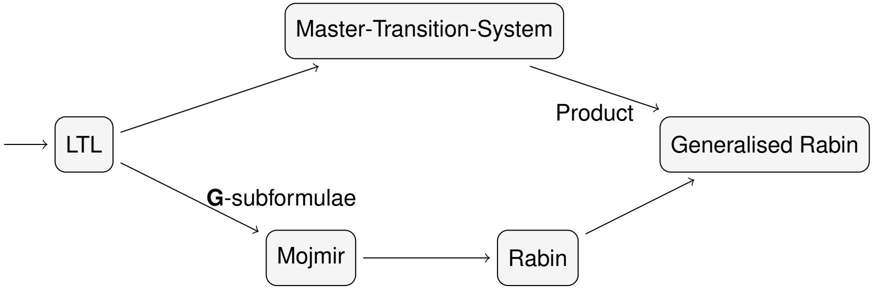
- From Mojmir to Rabin Automata
  - Unbounded number of tokens?
    Abstraction with ranking functions for states and tokens
  - Mojmir acceptance ($\overset{\infty}{\forall}$) vs. Rabin acceptance (finite, $\overset{\infty}{\exists}$)?
    Alternative definition for Mojmir acceptance

- Mojmir Automata for $\mathbf{FG}\varphi$ for arbitrary φ
  - Divide-and-conquer approach
  - Construct for every **G**-subformula a separate automaton
  - Instead of expanding **G**'s rely on the other automata
  - Intersection and Union of several Mojmir Automata

Overview of the Construction

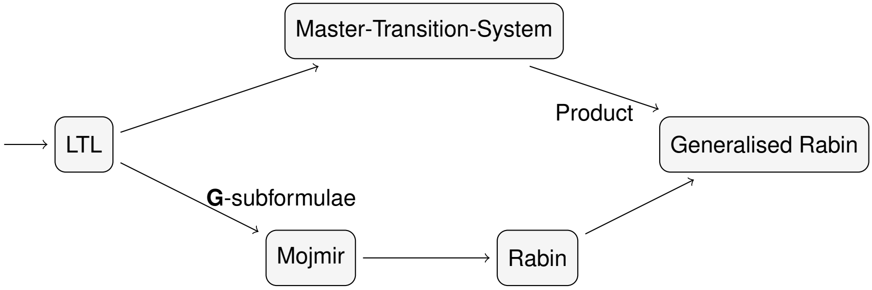


Overview of the Construction



- The Master-Transition-System tracks a finite prefix of the ω-word.

Overview of the Construction



- The Master-Transition-System tracks a finite prefix of the ω-word.
- Acceptance:
  1. Guess the set of eventually true **G**-subformulae
  2. Verify this guess using the Mojmir automata
  3. Accept *iff* almost all the time this guess entails the current state of the master-transition-system

Conclusion and Future Work

The presented translation . . .

- preservers the logical structure of the formula
- is compositional
  - Aggressive optimization can lead to huge space savings
  - Some optimizations are already verified
- yields small deterministic ω-automata

Conclusion and Future Work

The presented translation . . .

- preservers the logical structure of the formula
- is compositional
  - Aggressive optimization can lead to huge space savings
  - Some optimizations are already verified
- yields small deterministic ω-automata

Open Problems:

- Explore and formalize further optimizations
- Adapt construction to support:
  - Alternation-free linear-time μ-calculus (contains LTL)
  - Parity automata

Getting More Information

- Javier Esparza, Jan Kretínský: From LTL to Deterministic Automata: A Safraless Compositional Approach. CAV 2014: pages 192–208

From LTL to Deterministic Automata:
A Safraless Compositional Approach

Javier Esparza and Jan Křetínský*†

Institut für Informatik, Technische Universität München, Germany
IST Austria

2014

Getting More Information

- Javier Esparza, Jan Kretínský: From LTL to Deterministic Automata: A Safraless Compositional Approach. CAV 2014: pages 192–208
- Isabelle/HOL Formalisation
  - To be submitted to the “Archive of Formal Proofs” - `afp.sourceforge.net`
  - Available on request: `sickert@in.tum.de`

From LTL to Deterministic Automata:
A Safraless Compositional Approach

Javier Esparza and Jan Křetínský*†

Institut für Informatik, Technische Universität München, Germany
IST Austria

2014

Getting More Information

- Javier Esparza, Jan Kretínský: From LTL to Deterministic Automata: A Safraless Compositional Approach. CAV 2014: pages 192–208
- Isabelle/HOL Formalisation
  - To be submitted to the “Archive of Formal Proofs” - `afp.sourceforge.net`
  - Available on request: `sickert@in.tum.de`

Thank you for your attention!

From LTL to Deterministic Automata:
A Safraless Compositional Approach

Javier Esparza and Jan Křetínský*†

Institut für Informatik, Technische Universität München, Germany
IST Austria

2014