IA159 Formal Verification Methods

LTL$\rightarrow$BA via Very Weak Alternating BA

František Blahoudek

Department of Computer Science
Faculty of Informatics
Masaryk University

Outline & Sources

Outline

1. infinite words
2. Linear Temporal Logic (LTL)
3. nondeterministic Büchi automata (BA) and their variants
4. alternating automata (AA)
5. translation of LTL into BA via AA

Source

- P. Gastin and G. Oddoux: *Fast LTL to Büchi Automata Translation*, LNCS 2102, Springer, 2001.

Motivation – Infinite Behaviours

system behaviour as a sequence of sets of atomic propositions

$$\{\text{request}\}\{\text{request}\}\{\}\{\text{print}\}$$

Motivation – Infinite Behaviours

system behaviour as a sequence of sets of atomic propositions

$$\{\text{request}\}\{\text{request}\}\{\}\{\text{print}\}$$

infinite behaviours = infinite words (ω-words)

$$\{\}\{\text{request}\}\{\}\{\text{request}, \text{print}\}\{\}(\{\text{request}\}\{\text{print}\})^\omega$$

Motivation – Infinite Behaviours

system behaviour as a sequence of sets of atomic propositions

$$\{\text{request}\}\{\text{request}\}\{\}\{\text{print}\}$$

infinite behaviours = infinite words (ω-words)

$$\{\}\{\text{request}\}\{\}\{\text{request}, \text{print}\}\{\}(\{\text{request}\}\{\text{print}\})^\omega$$

$u = u(0)u(1)\ldots \in \Sigma^\omega$ — ω-word over the alphabet Σ

$u_i = u(i)u(i+1)\ldots$ — the *i*-th suffix of u

Motivation – Infinite Behaviours

system behaviour as a sequence of sets of atomic propositions

$$\{\text{request}\}\{\text{request}\}\{\}\{\text{print}\}$$

infinite behaviours = infinite words (ω-words)

$$\{\}\{\text{request}\}\{\}\{\text{request}, \text{print}\}\{\}(\{\text{request}\}\{\text{print}\})^\omega$$

$u = u(0)u(1)\ldots \in \Sigma^\omega$ — ω-word over the alphabet Σ
$u_i = u(i)u(i+1)\ldots$ — the i-th suffix of u

For reasoning about infinite behaviours we need

1. to express interesting properties, and (LTL)
2. to check the properties efficiently. (Büchi automata)

Syntax of LTL

Formulae of Linear Temporal Logic (LTL) in Positive Normal Form are defined by

$$\varphi \ ::= \ \top \ | \ \bot \ | \ a \ | \ \neg a \ | \ \varphi_1 \wedge \varphi_2 \ | \ \varphi_1 \vee \varphi_2 \ | \ \mathsf{X}\varphi \ | \ \varphi_1 \,\mathsf{U}\, \varphi_2 \ | \ \varphi_1 \,\mathsf{R}\, \varphi_2$$

where $\top, \bot$ stand for true, false respectively and a ranges over a countable set AP of atomic propositions.

Syntax of LTL

Formulae of Linear Temporal Logic (LTL) in Positive Normal Form are defined by

$$\varphi ::= \top \mid \bot \mid a \mid \neg a \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \mathsf{X}\varphi \mid \varphi_1 \,\mathsf{U}\, \varphi_2 \mid \varphi_1 \,\mathsf{R}\, \varphi_2$$

where $\top, \bot$ stand for true, false respectively and a ranges over a countable set AP of atomic propositions.

Abbreviations: $\mathsf{F}\varphi \equiv \top \,\mathsf{U}\, \varphi$ $\quad$ $\mathsf{G}\varphi \equiv \bot \,\mathsf{R}\, \varphi$

Syntax of LTL

Formulae of Linear Temporal Logic (LTL) in Positive Normal Form are defined by

$$\varphi \ ::= \ \top \ | \ \bot \ | \ a \ | \ \neg a \ | \ \varphi_1 \wedge \varphi_2 \ | \ \varphi_1 \vee \varphi_2 \ | \ \mathsf{X}\varphi \ | \ \varphi_1 \,\mathsf{U}\, \varphi_2 \ | \ \varphi_1 \,\mathsf{R}\, \varphi_2$$

where $\top, \bot$ stand for true, false respectively and a ranges over a countable set AP of atomic propositions.

Abbreviations: $\mathsf{F}\varphi \equiv \top \,\mathsf{U}\, \varphi$ $\qquad$ $\mathsf{G}\varphi \equiv \bot \,\mathsf{R}\, \varphi$

Temporal operators: terminology and intuitive meaning

$\mathsf{X}a$	next	$\bullet\ a\ \bullet\ \bullet\ \bullet\ \ldots$
$a \,\mathsf{U}\, b$	until	$a\ a\ \ldots\ a\ b\ \bullet\ \bullet\ \bullet\ \ldots$
$a \,\mathsf{R}\, b$	releases	$b\ b\ \ldots\ b\ \frac{a}{b}\ \bullet\ \bullet\ \bullet\ \ldots$ or $b\ b\ b\ b\ \ldots$
$\mathsf{F}a$	eventually	$\bullet\ \bullet\ \ldots\ \bullet\ a\ \bullet\ \bullet\ \ldots$
$\mathsf{G}a$	always	$a\ a\ a\ a\ \ldots$

Semantics of LTL

Let $\Sigma = 2^{AP'}$ where $AP' \subseteq AP$ is finite. The validity of an LTL formula φ for $u \in \Sigma^\omega$, written $u \models \varphi$, is defined as

$$
\begin{array}{lll}
u \models \top & & \\
u \models a & \text{iff} & a \in u(0) \\
u \models \neg a & \text{iff} & a \notin u(0) \\
u \models \varphi_1 \vee \varphi_2 & \text{iff} & u \models \varphi_1 \text{ or } u \models \varphi_2 \\
u \models \varphi_1 \wedge \varphi_2 & \text{iff} & u \models \varphi_1 \text{ and } u \models \varphi_2 \\
u \models \mathrm{X}\varphi & \text{iff} & u_1 \models \varphi \\
u \models \varphi_1 \,\mathrm{U}\, \varphi_2 & \text{iff} & \exists i \geq 0 \text{ such that} \\
 & & u_i \models \varphi_2 \text{ and } \forall\, 0 \leq j < i \,.\, u_j \models \varphi_1 \\
u \models \varphi_1 \,\mathrm{R}\, \varphi_2 & \text{iff} & \exists i \geq 0 \text{ such that} \\
 & & \quad u_i \models \varphi_1 \text{ and } \forall\, 0 \leq j \leq i \,.\, u_j \models \varphi_2, \\
 & & \text{or } \forall i \geq 0 \,.\, u_i \models \varphi_2
\end{array}
$$

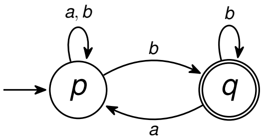
Given an alphabet Σ, an LTL formula φ defines the language

$$L^\Sigma(\varphi) = \{w \in \Sigma^\omega \mid w \models \varphi\}.$$

Büchi Automata

A Büchi automaton (BA) is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ defined precisely as a finite automaton, but

- a Büchi automaton is interpreted over infinite words, and
- a run is accepting if it visits some accepting state infinitely often.

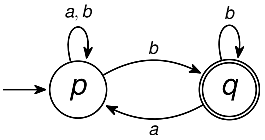


Accepts all infinite words over $\Sigma = \{a, b\}$ with infinitely many b.

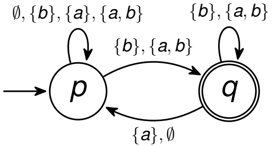
Büchi Automata

A Büchi automaton (BA) is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ defined precisely as a finite automaton, but

- a Büchi automaton is interpreted over infinite words, and
- a run is accepting if it visits some accepting state infinitely often.



Accepts all infinite words over $\Sigma = \{a, b\}$ with infinitely many b.

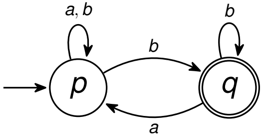


Accepts all infinite words over $\Sigma = 2^{\{a,b\}}$ where b appears in infinitely many sets.

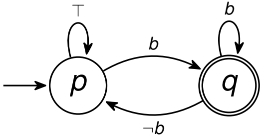
Büchi Automata

A Büchi automaton (BA) is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ defined precisely as a finite automaton, but

- a Büchi automaton is interpreted over infinite words, and
- a run is accepting if it visits some accepting state infinitely often.



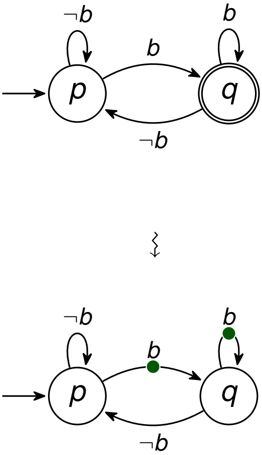
Accepts all infinite words over $\Sigma = \{a, b\}$ with infinitely many b.



Accepts all infinite words over $\Sigma = 2^{\{a,b\}}$ where b appears in infinitely many sets.

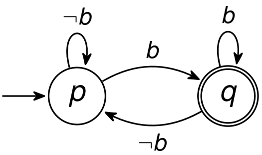
Extensions of Büchi Automata

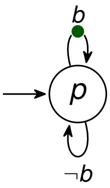
1. transition-based acceptance:
   - a run is accepting if it visits some accepting transition infinitely often



Extensions of Büchi Automata

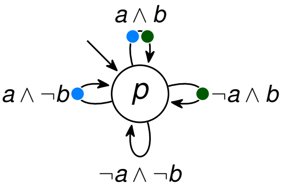
1. transition-based acceptance:
   - a run is accepting if it visits some accepting transition infinitely often





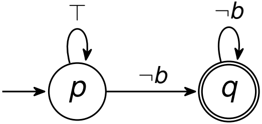
Extensions of Büchi Automata

1. transition-based acceptance:
   - a run is accepting if it visits some accepting transition infinitely often
2. generalized Büchi acceptance:
   - more sets of accepting states/transitions
   - a run is accepting if each set is visited infinitely often

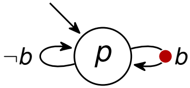


Extensions of Büchi Automata

1. transition-based acceptance:
   - a run is accepting if it visits some accepting transition infinitely often
2. generalized Büchi acceptance:
   - more sets of accepting states/transitions
   - a run is accepting if each set is visited infinitely often
3. co-Büchi acceptance
   - a run is accepting if it contains only finitely many accepting states/transitions



state-based Büchi



transition-based co-Büchi

TGBA to BA transformation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \{F_1, F_2, \ldots, F_k\})$ be a transition-based generalized Büchi automaton (TGBA) with $k \geq 2$ accepting sets. We build an equivalent state-based Büchi automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \delta_\mathcal{B}, q_\mathcal{B}, F)$ as follows.

TGBA to BA transformation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \{F_1, F_2, \ldots, F_k\})$ be a transition-based generalized Büchi automaton (TGBA) with $k \geq 2$ accepting sets. We build an equivalent state-based Büchi automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \delta_\mathcal{B}, q_\mathcal{B}, F)$ as follows.

- we have $k + 1$ copies of $\mathcal{A}$ (levels 0 to k)
  $Q_\mathcal{B} = Q \times \{0, \ldots, k\}$ $\qquad |Q_\mathcal{B}| \leq (k + 1) \cdot |Q|$

TGBA to BA transformation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \{F_1, F_2, \ldots, F_k\})$ be a transition-based generalized Büchi automaton (TGBA) with $k \geq 2$ accepting sets. We build an equivalent state-based Büchi automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \delta_\mathcal{B}, q_\mathcal{B}, F)$ as follows.

- we have $k + 1$ copies of $\mathcal{A}$ (levels 0 to k)
  $Q_\mathcal{B} = Q \times \{0, \ldots, k\}$ $\qquad |Q_\mathcal{B}| \leq (k + 1) \cdot |Q|$
- the initial state is on level 0 $q_\mathcal{B} = (q_0, 0)$

TGBA to BA transformation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \{F_1, F_2, \dots, F_k\})$ be a transition-based generalized Büchi automaton (TGBA) with $k \geq 2$ accepting sets. We build an equivalent state-based Büchi automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \delta_\mathcal{B}, q_\mathcal{B}, F)$ as follows.

- we have $k + 1$ copies of $\mathcal{A}$ (levels 0 to k)
  $Q_\mathcal{B} = Q \times \{0, \dots, k\}$ $\qquad |Q_\mathcal{B}| \leq (k + 1) \cdot |Q|$
- the initial state is on level 0 $q_\mathcal{B} = (q_0, 0)$
- all transitions from level 0 go to level 1
  $((q, 0), a, (p, 1)) \in \delta_\mathcal{B} \iff (q, a, p) \in \delta$

TGBA to BA transformation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \{F_1, F_2, \ldots, F_k\})$ be a transition-based generalized Büchi automaton (TGBA) with $k \geq 2$ accepting sets. We build an equivalent state-based Büchi automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \delta_\mathcal{B}, q_\mathcal{B}, F)$ as follows.

- we have $k+1$ copies of $\mathcal{A}$ (levels 0 to k)
  $Q_\mathcal{B} = Q \times \{0, \ldots, k\}$ $\qquad\qquad |Q_\mathcal{B}| \leq (k+1) \cdot |Q|$
- the initial state is on level 0 $q_\mathcal{B} = (q_0, 0)$
- all transitions from level 0 go to level 1
  $((q,0), a, (p,1)) \in \delta_\mathcal{B} \iff (q,a,p) \in \delta$
- on level $i > 2$ we wait for a transition from F_i and then move to level $(i+1)$ (or 0 if $i = k$)
  $((q,i), a, (p,i)) \in \delta_\mathcal{B} \iff (q,a,p) \in \delta \setminus F_i$
  $((q,i), a, (p,(i+1) \bmod (k+1))) \in \delta_\mathcal{B} \iff (q,a,p) \in \delta \cap F_i$

TGBA to BA transformation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \{F_1, F_2, \dots, F_k\})$ be a transition-based generalized Büchi automaton (TGBA) with $k \geq 2$ accepting sets. We build an equivalent state-based Büchi automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \delta_\mathcal{B}, q_\mathcal{B}, F)$ as follows.

- we have $k + 1$ copies of $\mathcal{A}$ (levels 0 to k)
  $Q_\mathcal{B} = Q \times \{0, \dots, k\}$ $\qquad |Q_\mathcal{B}| \leq (k + 1) \cdot |Q|$
- the initial state is on level 0 $q_\mathcal{B} = (q_0, 0)$
- all transitions from level 0 go to level 1
  $((q, 0), a, (p, 1)) \in \delta_\mathcal{B} \iff (q, a, p) \in \delta$
- on level $i > 2$ we wait for a transition from F_i and then move to level $(i + 1)$ (or 0 if $i = k$)
  $((q, i), a, (p, i)) \in \delta_\mathcal{B} \iff (q, a, p) \in \delta \smallsetminus F_i$
  $((q, i), a, (p, (i+1) \bmod (k+1))) \in \delta_\mathcal{B} \iff (q, a, p) \in \delta \cap F_i$
- the level 0 is accepting
  $F = Q \times \{0\}$

TGBA to BA transformation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \{F_1, F_2, \dots, F_k\})$ be a transition-based generalized Büchi automaton (TGBA) with $k \geq 2$ accepting sets. We build an equivalent state-based Büchi automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \delta_\mathcal{B}, q_\mathcal{B}, F)$ as follows.

- we have $k+1$ copies of $\mathcal{A}$ (levels 0 to k)
  $Q_\mathcal{B} = Q \times \{0, \dots, k\}$ $\qquad |Q_\mathcal{B}| \leq (k+1) \cdot |Q|$
- the initial state is on level 0 $q_\mathcal{B} = (q_0, 0)$
- all transitions from level 0 go to level 1
  $((q,0), a, (p,1)) \in \delta_\mathcal{B} \iff (q,a,p) \in \delta$
- on level $i > 2$ we wait for a transition from F_i and then move to level $(i+1)$ (or 0 if $i = k$)
  $((q,i), a, (p,i)) \in \delta_\mathcal{B} \iff (q,a,p) \in \delta \smallsetminus F_i$
  $((q,i), a, (p, (i+1) \bmod (k+1))) \in \delta_\mathcal{B} \iff (q,a,p) \in \delta \cap F_i$
- the level 0 is accepting
  $F = Q \times \{0\}$

TGBA to BA transformation

Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \{F_1, F_2, \ldots, F_k\})$ be a transition-based generalized Büchi automaton (TGBA) with $k \geq 2$ accepting sets. We build an equivalent state-based Büchi automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \delta_\mathcal{B}, q_\mathcal{B}, F)$ as follows.

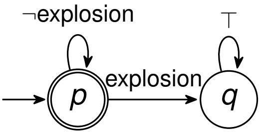
- we have $k + 1$ copies of $\mathcal{A}$ (levels 0 to k)
  $Q_\mathcal{B} = Q \times \{0, \ldots, k\}$ $\qquad |Q_\mathcal{B}| \leq (k + 1) \cdot |Q|$
- the initial state is on level 0 $q_\mathcal{B} = (q_0, 0)$
- all transitions from level 0 go to level 1
  $((q, 0), a, (p, 1)) \in \delta_\mathcal{B} \iff (q, a, p) \in \delta$
- on level $i > 2$ we wait for a transition from F_i and then move to level $(i + 1)$ (or 0 if $i = k$)
  $((q, i), a, (p, i)) \in \delta_\mathcal{B} \iff (q, a, p) \in \delta \smallsetminus F_i$
  $((q, i), a, (p, (i+1) \bmod (k+1))) \in \delta_\mathcal{B} \iff (q, a, p) \in \delta \cap F_i$
- the level 0 is accepting
  $F = Q \times \{0\}$

LTL to BA translations

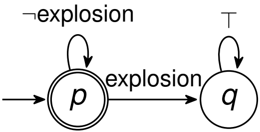
$\mathsf{G}\neg explosion$

LTL to BA translations

$\mathsf{G}\neg\mathrm{explosion} \rightsquigarrow$

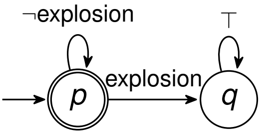


LTL to BA translations

$\mathsf{G}\neg\mathit{explosion} \rightsquigarrow$ 

- applications in automata-based LTL model checking, vacuity checking (checks trivial validity of a specification formula), ...

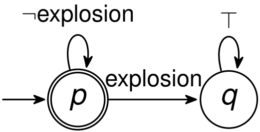
LTL to BA translations

$\mathbf{G}\neg explosion \rightsquigarrow$ 

- applications in automata-based LTL model checking, vacuity checking (checks trivial validity of a specification formula), ...
- many LTL→BA translations
  - LTL → generalized Büchi automata (GBA) → BA (Spin)
  - LTL → transition-based GBA (TGBA) → BA (Spot)
  - LTL → very weak alternating co-Büchi automata (VWAA) → → TGBA → BA (LTL2BA, LTL3BA)
  - ...

LTL to BA translations

$\mathsf{G}\neg explosion \rightsquigarrow$



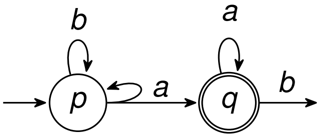
- applications in automata-based LTL model checking, vacuity checking (checks trivial validity of a specification formula), ...
- many LTL→BA translations
  - LTL → generalized Büchi automata (GBA) → BA (Spin)
  - LTL → transition-based GBA (TGBA) → BA (Spot)
  - LTL → very weak alternating co-Büchi automata (VWAA) → → TGBA → BA (LTL2BA, LTL3BA)
  - ...
- translations via alternating automata offer
  - size-reducing optimizations of alternating automata
  - smaller resulting BA (in some cases)

Alternating Automata

An alternating co-Büchi automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set of states,
- Σ is a finite alphabet,
- $\delta : Q \times \Sigma \rightarrow 2^{2^Q}$ is a transition function,
- $q_0 \in Q$ is an initial state,
- $F \subseteq Q$ is a set of co-Büchi-accepting states.

Alternating Automata – Example

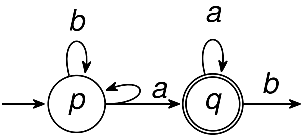


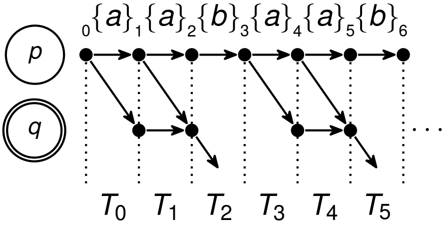
$$\delta(p, \{a, b\}) = \{\{p\}, \{p, q\}\}$$
$$\delta(q, \{b\}) = \{\emptyset\}$$
$$\delta(q, \emptyset) = \{\}$$

Alternating Automata – Runs

A run of $\mathcal{A}$ over a word $u = u(0)u(1)\ldots$ is a Directed Acyclic Graph (DAG) $G = (V, E)$ where

- $V = Q \times \{0, 1, 2, \ldots\}$
- only the state q_0 is in the level 0
- for any $(q, i) \in V$ it holds that
  - there is exactly one $P \in \delta(q, u(i))$ such that
  - for each $p \in P$ it holds that $((q, i), (p, i+1)) \in E$
- no other nodes and edges are in V and E





Alternating Automata – Accepting

A run is accepting iff each its infinite branch contains only finitely many states from F. [co-Büchi acceptance]

An automaton $\mathcal{A}$ accepts a word u iff there is an accepting run of $\mathcal{A}$ on u. We set

$$L(\mathcal{A}) = \{u \in \Sigma^\omega \mid \mathcal{A} \text{ accepts } u\}.$$

Very Weak Alternating Automata

Intuitively, an alternating automaton is very weak, written VWAA (or linear or 1-weak, written A1W) iff it contains no cycles except selfloops.

Formally, let $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ be an alternating automaton. Automaton $\mathcal{A}$ is very weak iff there exists a partial order $\preceq$ on Q such that for all $p, q \in Q$ and $\alpha \in \Sigma$ it holds:

$$p \in P, P \in \delta(q, \alpha) \implies p \preceq q$$

LTL $\rightarrow$ co-Büchi VWAA

LTL$\rightarrow$VWAA

The main ideas:

- states are subformulae of φ
- build bottom-up
- what needs to hold now and what in the next step

LTL$\rightarrow$VWAA

The main ideas:

- states are subformulae of φ
- build bottom-up
- what needs to hold now and what in the next step

Transition combination: Let $D, D' \subseteq 2^Q$ be two sets of state sets. We define their product $D \otimes D'$ as

$$D \otimes D' = \{P \cup P' \mid P \in D \text{ and } P' \in D'\}$$

Notes

- standard Büchi automata are alternating Büchi automata where each set in $\delta(p, l)$ is singleton
- VWAA automata have the same expressive power as LTL

LTL→VWAA

Input: an LTL formula φ and an alphabet $\Sigma = 2^{AP'}$ for some finite $AP' \subseteq AP$

Output: VWAA automaton $\mathcal{A} = (Q, \Sigma, \delta, \varphi, F)$ accepting $L^{\Sigma}(\varphi)$

LTL$\rightarrow$VWAA

Input: an LTL formula φ and an alphabet $\Sigma = 2^{AP'}$ for some finite $AP' \subseteq AP$

Output: VWAA automaton $\mathcal{A} = (Q, \Sigma, \delta, \varphi, F)$ accepting $L^{\Sigma}(\varphi)$

- $Q = \{\psi \mid \psi \text{ is a subformula of } \varphi\}$

LTL$\rightarrow$VWAA

Input: an LTL formula φ and an alphabet $\Sigma = 2^{AP'}$ for some finite $AP' \subseteq AP$

Output: VWAA automaton $\mathcal{A} = (Q, \Sigma, \delta, \varphi, F)$ accepting $L^{\Sigma}(\varphi)$

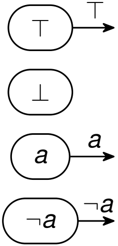
- $Q = \{\psi \mid \psi \text{ is a subformula of } \varphi\}$
- δ for $I \in \Sigma$ is defined as follows

$\delta(\top, I) = \{\emptyset\}$

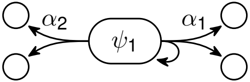
$\delta(\bot, I) = \emptyset$

$\delta(a, I) = \{\emptyset\}$ if $a \in I$, $\emptyset$ otherwise

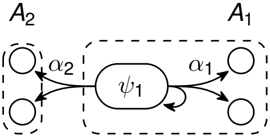
$\delta(\neg a, I) = \{\emptyset\}$ if $a \notin I$, $\emptyset$ otherwise



LTL→VWAA cont.

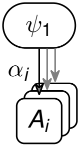


LTL→VWAA cont.



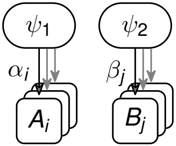
LTL→VWAA cont.

$A_i \in \delta(\psi_1, I) \iff I \models \alpha_i$



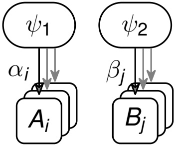
LTL→VWAA cont.

$$A_i \in \delta(\psi_1, I) \iff I \models \alpha_i$$

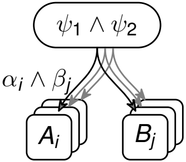


LTL→VWAA cont.

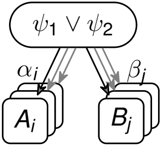
$$A_i \in \delta(\psi_1, I) \iff I \models \alpha_i$$



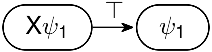
$$\delta(\psi_1 \wedge \psi_2, I) = \delta(\psi_1, I) \otimes \delta(\psi_2, I)$$



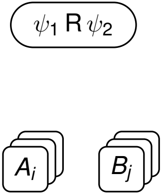
$$\delta(\psi_1 \vee \psi_2, I) = \delta(\psi_1, I) \cup \delta(\psi_2, I)$$



$$\delta(\mathsf{X}\psi_1, I) = \{\{\psi_1\}\}$$

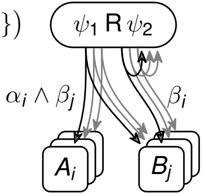


LTL→VWAA cont.



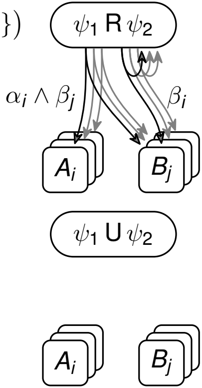
LTL→VWAA cont.

$$\delta(\psi_1 \,\mathsf{R}\, \psi_2, I) = \delta(\psi_2, I) \otimes \big(\delta(\psi_1, I) \cup \{\{\psi_1 \,\mathsf{R}\, \psi_2\}\}\big)$$



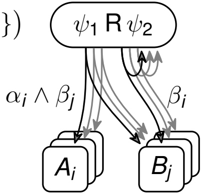
LTL→VWAA cont.

$$\delta(\psi_1 \mathrm{R}\, \psi_2, I) = \delta(\psi_2, I) \otimes \big(\delta(\psi_1, I) \cup \{\{\psi_1 \mathrm{R}\, \psi_2\}\}\big)$$

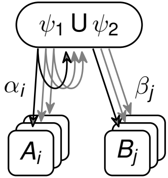


LTL→VWAA cont.

$$\delta(\psi_1 \,\mathsf{R}\, \psi_2, I) = \delta(\psi_2, I) \otimes \left(\delta(\psi_1, I) \cup \{\{\psi_1 \,\mathsf{R}\, \psi_2\}\}\right)$$

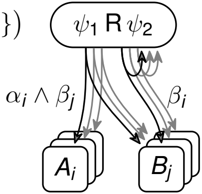


$$\delta(\psi_1 \,\mathsf{U}\, \psi_2, I) = \delta(\psi_2, I) \cup \left(\delta(\psi_1, I) \otimes \{\{\psi_1 \,\mathsf{U}\, \psi_2\}\}\right)$$

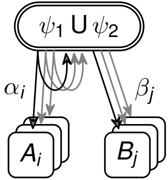


LTL→VWAA cont.

$$\delta(\psi_1 \mathsf{R}\, \psi_2, I) = \delta(\psi_2, I) \otimes \big(\delta(\psi_1, I) \cup \{\{\psi_1 \mathsf{R}\, \psi_2\}\}\big)$$



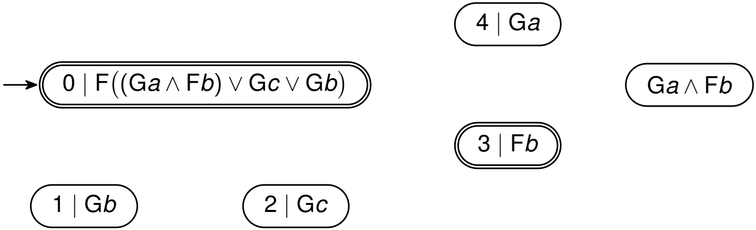
$$\delta(\psi_1 \mathsf{U}\, \psi_2, I) = \delta(\psi_2, I) \cup \big(\delta(\psi_1, I) \otimes \{\{\psi_1 \mathsf{U}\, \psi_2\}\}\big)$$



- $F = \{\psi_1 \mathsf{U}\, \psi_2 \mid \psi_1 \mathsf{U}\, \psi_2$ is a subformula of $\varphi\}$

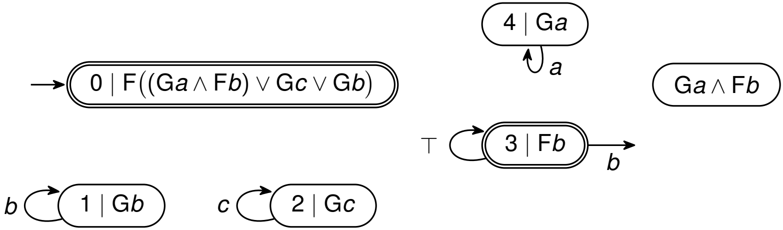
LTL→VWAA - Example

$$\mathsf{F}\big((\mathsf{G}a \wedge \mathsf{F}b) \vee \mathsf{G}c \vee \mathsf{G}b\big)$$



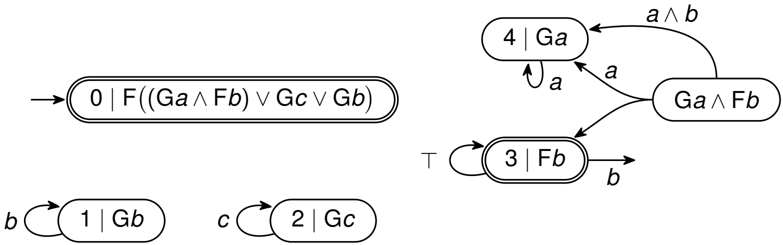
LTL$\to$VWAA - Example

$$\mathsf{F}\big((\mathsf{G}a \wedge \mathsf{F}b) \vee \mathsf{G}c \vee \mathsf{G}b\big)$$

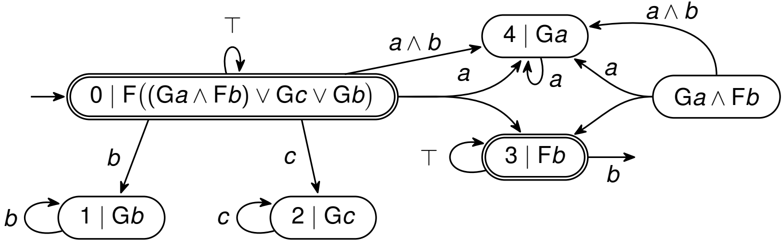


LTL→VWAA - Example

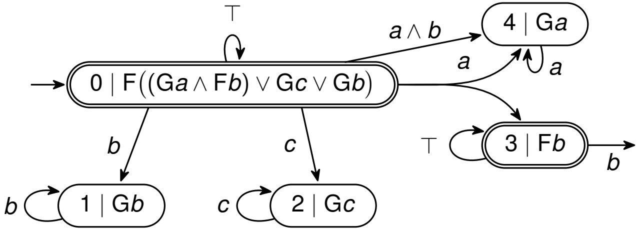
$$\mathsf{F}\big((\mathsf{G}a \wedge \mathsf{F}b) \vee \mathsf{G}c \vee \mathsf{G}b\big)$$



$$\mathsf{F}\big((\mathsf{G}a \wedge \mathsf{F}b) \vee \mathsf{G}c \vee \mathsf{G}b\big)$$



$$\mathsf{F}\big((\mathsf{G}a \wedge \mathsf{F}b) \vee \mathsf{G}c \vee \mathsf{G}b\big)$$



LTL$\rightarrow$VWAA

Note that every infinite branch of a run of $\mathcal{A}$ has a suffix with states of the form $\psi_1 \, \mathsf{U} \, \psi_2$ or $\psi_1 \, \mathsf{R} \, \psi_2$ (other states have no loops and can appear at most once on a branch). *F* is defined to ensure that ψ_2 eventually holds for each $\psi_1 \, \mathsf{U} \, \psi_2$.

Theorem

Given an LTL formula φ and an alphabet Σ, one can construct a VWAA $\mathcal{A}$ accepting $L^{\Sigma}(\varphi)$ such that the number of states of $\mathcal{A}$ is linear in the length of φ.

co-Büchi VWAA $\rightarrow$ TGBA

VWAA $\rightarrow$ TGBA

The key ideas:

- the TGBA tracks (selected) possible runs of the VWAA
- a run of the TGBA tracks states on each level of the run (DAG)

VWAA $\rightarrow$ TGBA

The key ideas:

- the TGBA tracks (selected) possible runs of the VWAA
- a run of the TGBA tracks states on each level of the run (DAG)
- states of the TGBA are sets (conjunction) of states

VWAA $\rightarrow$ TGBA

The key ideas:

- the TGBA tracks (selected) possible runs of the VWAA
- a run of the TGBA tracks states on each level of the run (DAG)
- states of the TGBA are sets (conjunction) of states
- once a state q is left by a branch, the branch never visits q again
- escaping f-transitions for an co-Büchi accepting state f

A transition $(q, l, P) \in \delta$ is q-escaping iff $q \notin P$.

VWAA $\rightarrow$ TGBA

Input: a co-Büchi VWAA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ with $k = |F|$
Output: TGBA $\mathcal{B} = (Q', \Sigma, \delta', q_0', \mathcal{F})$ accepting $L(\mathcal{A})$

VWAA $\rightarrow$ TGBA

Input: a co-Büchi VWAA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ with $k = |F|$
Output: TGBA $\mathcal{B} = (Q', \Sigma, \delta', q_0', \mathcal{F})$ accepting $L(\mathcal{A})$

- $Q' = 2^Q$
- $q_0' = \{q_0\}$

VWAA $\rightarrow$ TGBA

Input: a co-Büchi VWAA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ with $k = |F|$
Output: TGBA $\mathcal{B} = (Q', \Sigma, \delta', q'_0, \mathcal{F})$ accepting $L(\mathcal{A})$

- $Q' = 2^Q$
- $q'_0 = \{q_0\}$
- $\delta''(P, l) = \bigotimes_{p \in P} \delta(p, l)$ is an unoptimized tr. function

VWAA $\rightarrow$ TGBA

Input: a co-Büchi VWAA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ with $k = |F|$
Output: TGBA $\mathcal{B} = (Q', \Sigma, \delta', q_0', \mathcal{F})$ accepting $L(\mathcal{A})$

- $Q' = 2^Q$
- $q_0' = \{q_0\}$
- $\delta''(P, l) = \bigotimes_{p \in P} \delta(p, l)$ is an unoptimized tr. function
- $\mathcal{F}'' = \{T_f'' \subseteq \delta'' \mid f \in F\}$ where

$$T_f'' = \{(P_1, l, P_2) \mid f \notin P_2, \text{ or } (f, l, P') \in \delta, P' \subseteq P_2 \text{ and } f \notin P'\}$$

VWAA $\to$ TGBA

Input: a co-Büchi VWAA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ with $k = |F|$
Output: TGBA $\mathcal{B} = (Q', \Sigma, \delta', q'_0, \mathcal{F})$ accepting $L(\mathcal{A})$

- $Q' = 2^Q$
- $q'_0 = \{q_0\}$
- $\delta''(P, l) = \bigotimes_{p \in P} \delta(p, l)$ is an unoptimized tr. function
- $\mathcal{F}'' = \{T''_f \subseteq \delta'' \mid f \in F\}$ where

$$T''_f = \{(P_1, l, P_2) \mid f \notin P_2, \text{ or } (f, l, P') \in \delta, P' \subseteq P_2 \text{ and } f \notin P'\}$$

- $\preccurlyeq$ is a relation on transitions of δ'' where

$$t_1 \preccurlyeq t_2 \text{ iff } t_1 = (P, l, P_1) \text{ and } t_2 = (P, l, P_2) \text{ and } P_1 \subseteq P_2 \text{ and } t_1 \in T''_f \implies t_2 \in T''_f \text{ for all } f \in F$$

VWAA $\rightarrow$ TGBA

Input: a co-Büchi VWAA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ with $k = |F|$
Output: TGBA $\mathcal{B} = (Q', \Sigma, \delta', q'_0, \mathcal{F})$ accepting $L(\mathcal{A})$

- $Q' = 2^Q$
- $q'_0 = \{q_0\}$
- $\delta''(P, l) = \bigotimes_{p \in P} \delta(p, l)$ is an unoptimized tr. function
- $\mathcal{F}'' = \{T''_f \subseteq \delta'' \mid f \in F\}$ where

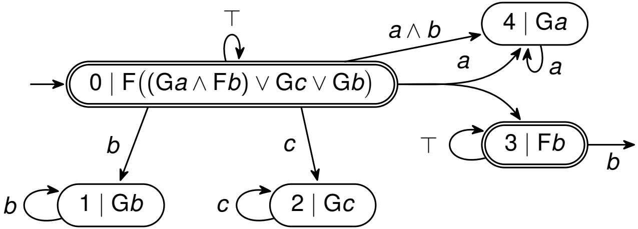
$$T''_f = \{(P_1, l, P_2) \mid f \notin P_2, \text{ or } (f, l, P') \in \delta, P' \subseteq P_2 \text{ and } f \notin P'\}$$

- $\preccurlyeq$ is a relation on transitions of δ'' where

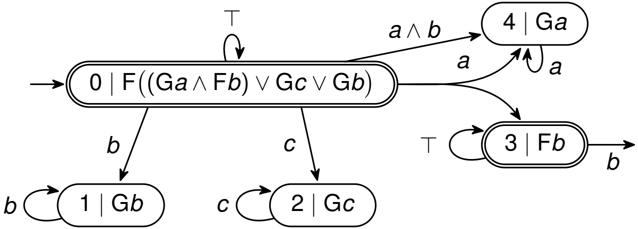
$$t_1 \preccurlyeq t_2 \text{ iff } t_1 = (P, l, P_1) \text{ and } t_2 = (P, l, P_2) \text{ and } P_1 \subseteq P_2 \text{ and } t_1 \in T''_f \implies t_2 \in T''_f \text{ for all } f \in F$$

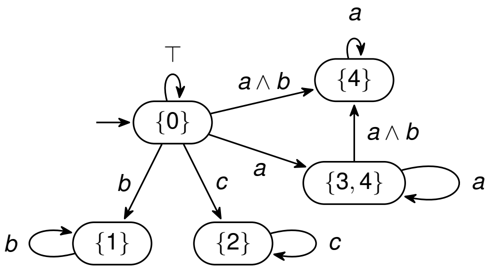
- δ' is the set of $\preccurlyeq$-minimal transitions of δ''
- $\mathcal{F} = \{T_f \cap \delta' \mid T_f \in \mathcal{F}''\}$

VWAA → TGBA – Example

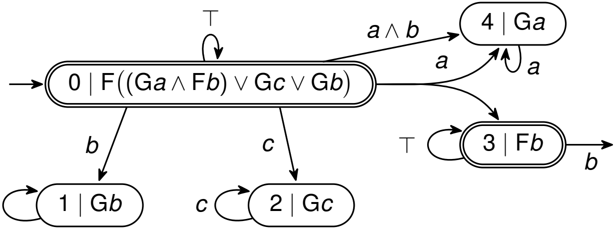


VWAA → TGBA – Example



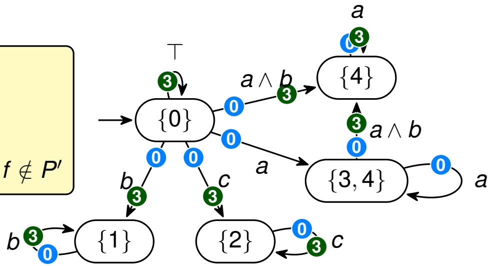


VWAA → TGBA – Example



$$P_1 \xrightarrow[f]{l} P_2$$

if $f \notin P_2$ or
$(f, l, P'), P' \subseteq P_2, f \notin P'$



LTL→A1W

Theorem

Given an co-Büchi VWAA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, one can construct a TGBA $\mathcal{B}$ with $2^{|Q|}$ states that accepts $L(\mathcal{A})$.

Corollary

Given an LTL formula φ and an alphabet Σ, one can construct a TGBA $\mathcal{B}$ accepting $L^{\Sigma}(\varphi)$ such that the number of states of $\mathcal{B}$ is $2^{|\varphi|}$. Consequently, one can construct a BA $\mathcal{C}$ that accepts $L^{\Sigma}(\varphi)$ and that has at most $|\varphi| \cdot 2^{|\varphi|}$ states.

Coming next week

Partial order reduction

- When can a state/transition be safely removed from a Kripke structure?
- What is a stuttering principle?
- Can we effectively compute the reduction?