Topics in Timed Automata

B. Srivathsan

RWTH-Aachen

Software modeling and Verification group
Deterministic timed automata are **closed under complement**
Theorem (Lecture 2)

Deterministic timed automata are **closed under complement**

1. **Unique** run for every timed word

\[ w_1 \in \mathcal{L}(A) \quad w_2 \notin \mathcal{L}(A) \]
Theorem (Lecture 2)

Deterministic timed automata are closed under complement

1. **Unique** run for every timed word

2. **Complementation:** Interchange acc. and non-acc. states

\[
\begin{align*}
& \begin{array}{l}
  w_1 \in \mathcal{L}(A) \quad w_2 \notin \mathcal{L}(A) \\
  w_1 \notin \overline{\mathcal{L}(A)} \quad w_2 \in \overline{\mathcal{L}(A)}
\end{array}
\end{align*}
\]
Theorem (Lecture 1)

Non-deterministic timed automata are **not closed under complement**

Many runs for a timed word

\[ w_1 \in \mathcal{L}(A) \]

 Exists an acc. run

\[ w_2 \notin \mathcal{L}(A) \]

All runs non-acc.
Theorem (Lecture 1)

Non-deterministic timed automata are **not closed under complement**

**Many** runs for a timed word

\[ w_1 \in \mathcal{L}(A) \]

**Exists an acc. run**

\[ w_2 \notin \mathcal{L}(A) \]

**All** runs non-acc.

**Complementation:** interchange **acc/non-acc** + ask are **all runs acc.** ?
A timed automaton model with existential and universal semantics for acceptance
Lecture 5:

Alternating timed automata

Lasota and Walukiewicz. *FoSSaCS’05, ACM TOCL’2008*
Section 1:

Introduction to ATA
- \( X \): set of clocks

- \( \Phi(X) \): set of clock constraints \( \sigma \) (guards)

\[
\sigma : \begin{array}{c}
x < c \\
x \leq c \\
\sigma_1 \land \sigma_2 \\
\neg \sigma
\end{array}
\]

c is a non-negative integer

- Timed automaton \( A \): \((Q, Q_0, \Sigma, X, T, F)\)

\[
T \subseteq Q \times \Sigma \times \Phi(X) \times Q \times \mathcal{P}(X)
\]
\( T \subseteq Q \times \Sigma \times \Phi(X) \times Q \times \mathcal{P}(X) \)

\[ T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X)) \]

Diagram:

- \( q \)
- \( a, g \)
- \( q_1, r_1 \)
- \( q_2, r_2 \)
- \( q_3, r_3 \)
- \( q_4, r_4 \)
- \( q_5, r_5 \)
\[ T \subseteq Q \times \Sigma \times \Phi(X) \times Q \times \mathcal{P}(X) \]

\[ T : Q \times \Sigma \times \Phi(X) \rightarrow \mathcal{P}(Q \times \mathcal{P}(X)) \]
\[ T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X)) \]
\[ T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X)) \]

\[ \mathcal{B}^+(S) \text{ is all } \phi ::= S \mid \phi_1 \land \phi_2 \mid \phi_1 \lor \phi_2 \]

\[ T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{B}^+(Q \times \mathcal{P}(X)) \]
\[ T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X)) \]

\[ \mathcal{B}^+(S) \text{ is all } \phi ::= S \mid \phi_1 \land \phi_2 \mid \phi_1 \lor \phi_2 \]

\[ T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{B}^+(Q \times \mathcal{P}(X)) \]

\[ (q_1, r_1 \land q_2, r_2) \lor (q_3, r_3) \lor (q_4, r_4 \land q_5, r_5 \land q_6, r_6) \]
Alternating Timed Automata

An ATA is a tuple $A = (Q, q_0, \Sigma, X, T, F)$ where:

$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{B}^+(Q \times \mathcal{P}(X))$$

is a finite partial function.
Alternating Timed Automata

An ATA is a tuple $A = (Q, q_0, \Sigma, X, T, F)$ where:

$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{B}^+(Q \times \mathcal{P}(X))$$

is a finite partial function.

**Partition:** For every $q, a$ the set

$$\{ [\sigma] \mid T(q, a, \sigma) \text{ is defined} \}$$

gives a finite partition of $\mathbb{R}_{\geq 0}^X$
Acceptance

Accepting run from $q$ iff:

$$(q_1, r_1 \land q_2, r_2) \lor (q_3, r_3) \lor (q_4, r_4 \land q_5, r_5 \land q_6, r_6)$$
Acceptance

Accepting run from $q$ iff:

- accepting run from $q_1$ and $q_2$,
Acceptance

Accepting run from $q$ iff:

1. accepting run from $q_1$ and $q_2$,
2. or accepting run from $q_3$,
Acceptance

Accepting run from \( q \) iff:

- accepting run from \( q_1 \) \textbf{and} \( q_2 \),

- or accepting run from \( q_3 \),

- or accepting run from \( q_4 \) \textbf{and} \( q_5 \) \textbf{and} \( q_6 \)
$L$: timed words over \{$a\}$ containing **no two $a$'s** at distance 1
(Not expressible by non-deterministic TA)
$L$: timed words over $\{a\}$ containing no two $a$'s at distance 1 (Not expressible by non-deterministic TA)

**ATA:**

$$q_0, a, tt \mapsto (q_0, \emptyset) \land (q_1, \{x\})$$

$$q_1, a, x = 1 \mapsto (q_2, \emptyset)$$

$$q_1, a, x \neq 1 \mapsto (q_1, \emptyset)$$

$$q_2, a, tt \mapsto (q_2, \emptyset)$$

$q_0, q_1$ are acc., $q_2$ is non-acc.
Closure properties

- **Union, intersection**: use disjunction/conjunction

- **Complementation**: interchange
  1. acc./non-acc.
  2. conjunction/disjunction
Closure properties

- **Union, intersection**: use disjunction/conjunction

- **Complementation**: interchange
  1. acc./non-acc.
  2. conjunction/disjunction

No change in the number of clocks!
Section 2: The 1-clock restriction
- **Emptiness:** given $A$, is $\mathcal{L}(A)$ empty
- **Universality:** given $A$, does $\mathcal{L}(A)$ contain all timed words
- **Inclusion:** given $A, B$, is $\mathcal{L}(A) \subseteq \mathcal{L}(B)$
- **Emptiness:** given $A$, is $\mathcal{L}(A)$ empty
- **Universality:** given $A$, does $\mathcal{L}(A)$ contain all timed words
- **Inclusion:** given $A, B$, is $\mathcal{L}(A) \subseteq \mathcal{L}(B)$

Undecidable for **two clocks or more** (via Lecture 3)
- **Emptiness**: given $A$, is $\mathcal{L}(A)$ empty
- **Universality**: given $A$, does $\mathcal{L}(A)$ contain all timed words
- **Inclusion**: given $A, B$, is $\mathcal{L}(A) \subseteq \mathcal{L}(B)$

Undecidable for **two clocks or more** (via Lecture 3)

Decidable for **one clock** (via Lecture 4)
- **Emptiness:** given $A$, is $\mathcal{L}(A)$ empty
- **Universality:** given $A$, does $\mathcal{L}(A)$ contain all timed words
- **Inclusion:** given $A, B$, is $\mathcal{L}(A) \subseteq \mathcal{L}(B)$

Undecidable for **two clocks or more** (via Lecture 3)

Decidable for **one clock** (via Lecture 4)

Restrict to one-clock ATA
Theorem

Languages recognizable by 1-clock ATA and (many clock) TA are **incomparable**

→ proof on the board
Section 3:
Complexity
Lower bound

Complexity of emptiness of purely universal 1-clock ATA is not bounded by a primitive recursive function
Lower bound

Complexity of emptiness of purely universal 1-clock ATA is not bounded by a primitive recursive function

⇒ complexity of Ouaknine-Worrell algorithm for universality of 1-clock TA is non-primitive recursive
**Primitive recursive functions**

Functions \( f : \mathbb{N} \mapsto \mathbb{N} \)

Basic primitive recursive functions:

- **Zero function**: \( Z() = 0 \)
- **Successor function**: \( \text{Succ}(n) = n + 1 \)
- **Projection function**: \( P_i(x_1, \ldots, x_n) = x_i \)

Operations:

- **Composition**

- **Primitive recursion**: if \( f \) and \( g \) are p.r. of arity \( k \) and \( k + 2 \), there is a p.r. \( h \) of arity \( k + 1 \):

\[
\begin{align*}
    h(0, x_1, \ldots, x_k) &= f(x_1, \ldots, x_k) \\
    h(n + 1, x_1, \ldots, x_k) &= g(h(n, x_1, \ldots, x_k), n, x_1, \ldots, x_k)
\end{align*}
\]
Addition:

\[
\text{Add}(0, y) = y \\
\text{Add}(n + 1, y) = \text{Succ}(\text{Add}(n, y))
\]
Addition:

\[ Add(0, y) = y \]
\[ Add(n + 1, y) = Succ(Add(n, y)) \]

Multiplication:

\[ Mult(0, y) = Z() \]
\[ Mult(n + 1, y) = Add(Mult(n, y), y) \]
Addition:

\[ Add(0, y) = y \]
\[ Add(n + 1, y) = Succ(Add(n, y)) \]

Multiplication:

\[ Mult(0, y) = Z() \]
\[ Mult(n + 1, y) = Add(Mult(n, y), y) \]

Exponentiation \(2^n\):

\[ Exp(0) = Succ(Z()) \]
\[ Exp(n + 1) = Mult(Exp(n), 2) \]
Addition:

\[
\text{Add}(0, y) = y \\
\text{Add}(n + 1, y) = \text{Succ}(\text{Add}(n, y))
\]

Multiplication:

\[
\text{Mult}(0, y) = Z() \\
\text{Mult}(n + 1, y) = \text{Add}(\text{Mult}(n, y), y)
\]

Exponentiation \(2^n\):

\[
\text{Exp}(0) = \text{Succ}(Z()) \\
\text{Exp}(n + 1) = \text{Mult}(\text{Exp}(n), 2)
\]

Hyper-exponentiation (tower of \(n\) two-s):

\[
\text{HyperExp}(0) = \text{Succ}(Z()) \\
\text{HyperExp}(n + 1) = \text{Exp}(\text{HyperExp}(n))
\]
Recursive but not primitive rec.: Ackermann function, Sudan function
Coming next: a problem that has complexity non-primitive recursive
Channel systems

Finite state description of communication protocols
G. von Bochmann. 1978

On communicating finite-state machines
D. Brand and P. Zafiropulo. 1983

Example from Schnoebelen’2002
Theorem [BZ’83]

Reachability in channel systems is undecidable
Coming next: modifying the model for decidability
Lossy channel systems

Finkel’94, Abdulla and Jonsson’96

Messages stored in channel can be lost during transition
Lossy channel systems

Finkel’94, Abdulla and Jonsson’96

Messages stored in channel can be lost during transition

Theorem [Schnoebelen’2002]
Reachability for lossy one-channel systems is non-primitive recursive
Reachability problem for lossy one-channel systems can be reduced to emptiness problem for purely universal 1-clock ATA
1-clock ATA

- **closed** under boolean operations
- **decidable** emptiness problem
- expressivity **incomparable** to many clock TA
- **non-primitive recursive** complexity for emptiness
1-clock ATA

- closed under boolean operations
- decidable emptiness problem
- expressivity incomparable to many clock TA
- non-primitive recursive complexity for emptiness

Other results: Undecidability of:

- 1-clock ATA + \(\varepsilon\)-transitions
- 1-clock ATA over infinite words
Summary of Part 1 of the course

- Lecture 1: Expressiveness, $\varepsilon$-transitions
- Lecture 2: Determinization
- Lecture 3: Universality and inclusion
- Lecture 4: Restriction to one-clock
- Lecture 5: Alternating timed automata