Lecture 20: Nov 22
Complementation of Buchi automata

- How do we complement a Büchi automaton?

- Can we determinize?

- Subset construction does not work. What if a subset has repeating and nonrepeating states?

- Theorem: For a Büchi automaton $\mathcal{M}$, there exists a finite congruence $\sim$ over A^* such that both the ω-language of $\mathcal{M}$, and its complement, are unions of $L_1 \cdot L_2^\omega$ where L_1 and L_2 are equivalence classes of $\sim$

- It follows that Büchi automata are closed under complement

- This approach constructs an automaton for the complement with $2^{3n^3+4n^2+1}$ states

- Best known (and optimal) complementation construction: $2^{O(n\ log\ n)}$

ω-regular languages

- The ω-language $\mathcal{L}$ is a regular-safety language if there is a regular language L such that $\mathcal{L} = \mathit{safe}(L)$.

- Regular-guarantee, regular-response, and regular-persistence languages defined similarly

- The ω-language $\mathcal{L}$ is ω-regular if it is a boolean combination of regular-response and regular-persistence languages.

- All the regular subclasses have similar properties as the original classes

Expressiveness of ω-automata

- Theorem: An ω-language $\mathcal{L}$ is ω-regular iff it is accepted by a Büchi automaton

- Other types of automata such as Streett, Rabin, Muller have the same expressive power

- The same class can be defined using ω-regular expressions

$$\varphi := a \mid \varphi \cdot \varphi \mid \varphi + \varphi \mid \varphi^* \mid \varphi^\omega$$

- Deterministic Streett also accept all ω-regular languages, but deterministic Büchi are less expressive (cannot define $A^* a^\omega$)

- Theorem: An ω-language $\mathcal{L}$ is regular-response language iff it is accepted by a deterministic Büchi automaton

Linear Temporal Logic

- Introduced by Pnueli in 1977, LTL is a popular language for writing requirements
- An alternative to Büchi automata to write temporal requirements succinctly
- While CTL formulas are evaluated at states of structures, LTL formulas are interpreted at positions of a fixed ω-trace
- In CTL, time is branching: two varieties of $\bigcirc$: existential and universal
- In LTL, time is linear, only one $\bigcirc$
  No path-quantifiers as in CTL
- Temporal operators can be nested aribitrarily

Syntax and Semantics

- Syntax:

$$\varphi \ ::= \ p \mid \varphi \vee \varphi \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi\mathcal{U}\varphi$$

- Semantics: for a fixed ω-word $\underline{a}$ and position i, define $i \models_{\underline{a}} \varphi$

$i \models_{\underline{a}} p$	iff	$a_i \models p$;
$i \models_{\underline{a}} \varphi_1 \vee \varphi_2$	iff	$i \models_{\underline{a}} \varphi_1$ or $i \models_{\underline{a}} \varphi_2$;
$i \models_{\underline{a}} \neg\varphi$	iff	$i \not\models_{\underline{a}} \varphi$;
$i \models_{\underline{a}} \bigcirc\varphi$	iff	$i+1 \models_{\underline{a}} \varphi$;
$i \models_{\underline{a}} \varphi_1\mathcal{U}\varphi_2$	iff	there is a natural number $j \geq i$ such that $j \models_{\underline{a}} \varphi_2$ and for all $i \leq k < j$, $k \models_{\underline{a}} \varphi_1$.

- The word $\underline{a}$ satisfies φ if $0 \models_{\underline{a}} \varphi$

- Each LTL formula φ defines the ω-language $\mathcal{L}_\varphi$ containing ω-words that satisfy φ

- A fair structure $\mathcal{K}$ satisfies φ if every ω-trace of $\mathcal{K}$ satisfies φ

LTL Specs

- Send-receive protocol
  - Production of message m denoted $P?m$
  - Consumption of message m denoted $C?m$
- If m is produced then it is eventually consumed:

$$\Box(P?m \rightarrow \Diamond C?m)$$

- m is repeatedly produced iff repeatedly consumed

$$\Box\Diamond P?m \leftrightarrow \Box\Diamond C?m$$

- m is not consumed unless produced

$$(\neg C?m)\ \mathcal{W}\ P?m$$

Tableau construction

- Given an LTL formula φ, we can construct a multiBüchi automaton $\mathcal{M}_\varphi$ that accepts only the satisfying models of φ
- States of φ: sets of subformulas of φ
- Transitions of φ: propagation of next-time formulas
- Büchi constraints ensure eventual satisfaction of Until-formulas
- Size of tableau: $2^{|\varphi|}$
- Checking satisfiability: fair-emptiness of $\mathcal{M}_\varphi$
- Model checking $(\mathcal{K}, \varphi)$:
  - Construct tableau $\mathcal{M}_{\neg\varphi}$ for negation
  - Construct product $\mathcal{K} \times \mathcal{M}_{\neg\varphi}$
  - Check for fair-emptiness
- Satisfiability (or model checking) for LTL is PSPACE complete

Expressiveness

- Fairness can be specified within LTL
- LTL formulas define ω-regular languages
- Not all ω-regular languages are definable (e.g. every even state is a p-state)
- We can add past-operators
- Expressiveness of CTL and LTL is incomparable

- Logic CTL*: combination of LTL and CTL
- Expressiveness of LTL corresponds to
  - First-order fragment of S1S
  - Star-free ω-regular expressions