Bernd Finkbeiner Date: July 19, 2011

Automata, Games and Verification: Lecture 15

Full μ-calculus

We introduce a split function $split : \mathbb{B}^+(\omega \times cl(\varphi)) \rightarrow \mathbb{B}^+(\omega \times cl(\varphi))$ to avoid Boolean combinations as states:

- $split(true) = true$
- $split(false) = false$
- $split(\alpha \wedge \beta) = split(\alpha) \wedge split(\beta)$
- $split(\alpha \vee \beta) = split(\alpha) \vee split(\beta)$
- for $\psi = \Box\psi' \mid \Diamond\psi', \mu x.\psi', \nu x\psi'$: $split((c, \psi)) = (c, \psi)$
- $split((c, \psi_1 \wedge \psi_2)) = split((c, \psi_1)) \wedge split((c, \psi_2))$
- $split((c, \psi_1 \vee \psi_2)) = split((c, \psi_1)) \vee split((c, \psi_2))$

Example: $split((0, p \wedge q) \wedge (1, \Diamond(q \wedge r))) = (0, p) \wedge (0, q) \wedge (1, \Diamond(q \wedge r))$

Alternation-depth of a formula φ, $\alpha(\varphi)$:

- $\alpha(\bot) = \alpha(\top) = \alpha(p) = \alpha(\neg p) = 0$
- $\alpha(\varphi \wedge \psi) = \alpha(\varphi \vee \psi) = \max\{\alpha(\varphi), \alpha(\psi)\}$
- $\alpha(\Box\psi) = \alpha(\Diamond\psi) = \alpha(\psi)$
- $\alpha(\mu p.\psi) = \max(\{1, \alpha(\psi)\} \cup \{\alpha(\nu p'.\psi') + 1 \mid$
  $\nu p'.\psi'$ is a subformula of ψ, p occurs in $\psi'\})$
- $\alpha(\nu p.\psi) = \max(\{1, \alpha(\psi)\} \cup \{\alpha(\mu p'.\psi') + 1 \mid$
  $\mu p'.\psi'$ is a subformula of ψ, p occurs in $\psi'\})$

Translation from general guarded μ-calculus formula φ to alternating parity automaton $\mathcal{A}_\varphi$:

- $\delta(p, \sigma, k) = true$ if $p \in \sigma$
- $\delta(p, \sigma, k) = false$ if $p \notin \sigma$
- $\delta(\neg p, \sigma, k) = false$ if $p \in \sigma$
- $\delta(\neg p, \sigma, k) = true$ if $p \notin \sigma$
- $\delta(\varphi \wedge \psi, \sigma, k) = split(\delta(\varphi, \sigma, k) \wedge \delta(\psi, \sigma, k))$

- $\delta(\varphi \vee \psi, \sigma, k) = \mathit{split}(\delta(\varphi, \sigma, k) \vee \delta(\psi, \sigma, k))$
- $\delta(\Box\varphi, \sigma, k) = \mathit{split}(\bigwedge_{c=0}^{k-1}(c, \varphi))$
- $\delta(\Diamond\varphi, \sigma, k) = \mathit{split}(\bigvee_{c=0}^{k-1}(c, \varphi))$
- $\delta(\mu y.\psi(y), \sigma, k) = \mathit{split}(\delta(\psi(\mu y.\psi(y)), \sigma, k))$
- $\delta(\nu y.\psi(y), \sigma, k) = \mathit{split}(\delta(\psi(\nu y.\psi(y)), \sigma, k))$

parity condition:

- $c(\psi)$ is the smallest odd number $\geq \alpha(\psi) - 1$ if ψ is a μ-formula,
- $c(\psi)$ is the smallest even number $\geq \alpha(\psi) - 1$ if ψ is a ν-formula,
- $c(\psi) = 0$ otherwise.

24 Summary

Automata

1. *Branching Mode*
   deterministic – nondeterministic – universal – alternating
2. *Acceptance Mode*
   Büchi – co-Büchi – parity – Streett – Rabin – Muller
3. *Input*
   words – trees

Expressive Power

Word automata:

	Büchi	co-Büchi	parity	Muller
deterministic	−	−	+	+
nondeterministic	+	−	+	+
universal	−	+	+	+
alternating	+	+	+	+

Tree automata:

	Büchi	co-Büchi	parity	Muller
deterministic	−	−	−	−
nondeterministic	−	−	+	+
universal	−	−	+	+
alternating	−	−	+	+

Characterization Theorems

Definition 1 *An* ω-regular *language is a finite union of* ω*-languages of the form* $U \cdot V^\omega$ *where* $U, V \subseteq \Sigma^*$ *are regular languages.*

Theorem 1 (Büchi's Characterization Theorem (1962)) *An* ω*-language is* Büchi recognizable *iff it is* ω*-regular.*

Theorem 2 *An* ω*-language* $L \subseteq \Sigma^\omega$ *is recognizable by a* deterministic Büchi automaton *iff there is a regular language* $W \subseteq \Sigma^*$ *s.t.* $L = \overrightarrow{W}$.

Theorem 3 *A language* $\mathcal{L}$ *is recognizable by a* deterministic Muller *automaton iff* $\mathcal{L}$ *is a boolean combination of languages* $\overrightarrow{W}$ *where* $W \subseteq \Sigma^*$ *is regular.*

Translating Branching Modes

- McNaughton: *nondeterministic Büchi word* automaton → *deterministic Muller*
- Miyano and Hayashi: *alternating Büchi word* → *nondeterministic Büchi*
- not covered: Muller and Schupp *alternating Rabin tree* automaton → *nondeterministic Rabin tree* automaton

Translating Acceptance Modes

- Büchi, co-Büchi, parity → *parity, Rabin, Streett* (easy: special cases);
- Büchi, co-Büchi, Rabin, Streett, parity → *Muller* (easy but expensive);
- Rabin, Streett → *parity*: index appearence record.
- not covered: Muller → *parity*: latest appearence record;

Automata and Games

1. *Acceptance* game of nondeterministic/alternating *word/tree* automata,
2. *Emptiness* game of nondeterministic *word/tree* automata

Over 1-letter alphabet: emptiness game = acceptance game

Applications:

- language emptiness test
- complementation of alternating automata, tree automata

Determinacy

1. Reachability, Büchi, co-Büchi, parity games are *memoryless determined*.
2. Muller, Streett, Rabin games are *determined*, but not memoryless determined.

Corollary: memoryless runs suffice for alternating Büchi, co-Büchi, parity automata.

Logics

$$\text{LTL} \subsetneq \text{QPTL} \approx \text{S1S} \approx \text{WS1S}$$

$$\text{CTL} \subsetneq \text{CTL*} \subsetneq \mu\text{-calculus}$$

$$\text{WS2S} \subsetneq \text{S2S}$$

Theorem 4 *LTL, QPTL, S1S, WS1S, QPTL, CTL, CTL*, μ-calculus, WS2S, S2S are* decidable *logics.*

Formula satisfiable? → translate formula to automaton → check emptiness.

μ-calculus model checking

Does a given pointed Kripke structure (M, q_0) satisfy a μ-calculus formula φ?

μ-calculus formula φ

↓

Alternating parity *tree* automaton $\mathcal{A}_\varphi$

↓

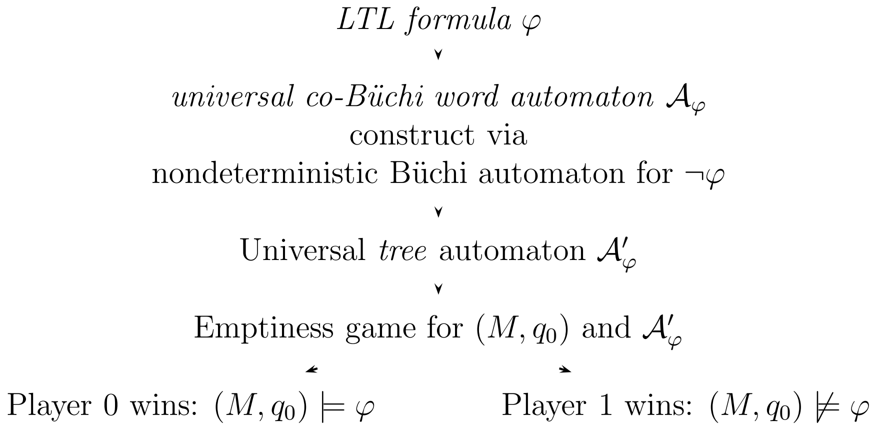
Emptyness game for (M, q_0) and $\mathcal{A}_\varphi$

↙ ↘

Player 0 wins: $(M, q_0) \models \varphi$ | Player 1 wins: $(M, q_0) \not\models \varphi$

LTL model checking

Does a given pointed Kripke structure (M, q_0) satisfy an LTL formula φ?



Alternative view on LTL model checking