Bernd Finkbeiner Date: June 21, 2011

Automata, Games and Verification: Lecture 11

17 McNaughton's Algorithm

McNaughton($\mathcal{G}$)

1. $c :=$ highest color in $\mathcal{G}$
2. if $c = 0$ or $V = \emptyset$
   then return $(V, \emptyset)$
3. set σ to c mod 2
4. set $W_{1-\sigma}$ to $\emptyset$
5. repeat
   (a) $\mathcal{G}' := \mathcal{G} \setminus Attr_\sigma(\alpha^{-1}(c))$
   (b) $(W'_0, W'_1) := McNaughton(\mathcal{G}')$
   (c) if $(W'_{1-\sigma} = \emptyset)$ then
      i. $W_\sigma := V \setminus W_{1-\sigma}$
      ii. return (W_0, W_1)
   (d) $W_{1-\sigma} := W_{1-\sigma} \cup Attr_{(1-\sigma)}(W'_{1-\sigma})$
   (e) $\mathcal{G} := \mathcal{G} \setminus Attr_{(1-\sigma)}(W'_{1-\sigma})$

18 Tree Automata

Binary Tree: $T = \{0,1\}^*$.
Notation: T_Σ : set of all binary Σ-trees

Definition 1 *A tree automaton (over binary Σ-trees) is a tuple $\mathcal{A} = (S, s_0, M, \varphi)$:*

- *S: finite set of states*
- $s_0 \in S$
- $M = S \times \Sigma \times S \times S$
- *φ: acceptance condition (Büchi, parity, ...)*

Definition 2 *A run of a tree automaton $\mathcal{A}$ on a Σ-tree v is a S-tree (T, r), s.t.*

- $r(\epsilon) = s_0$

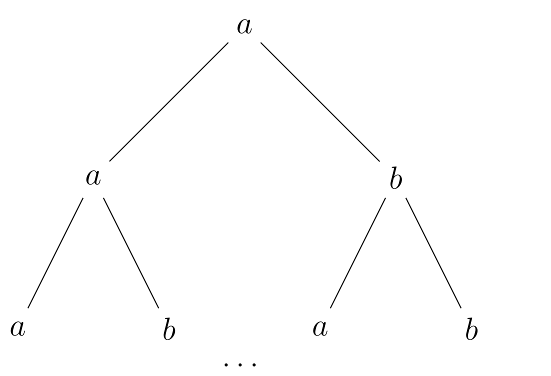
- $(r(q), v(q), r(q0), r(q1)) \in M$ *for all* $q \in \{0,1\}^*$

Definition 3 *A run is accepting if every branch is accepting (by* φ*). A* Σ*-tree is accepted if there exists an accepting run.*
$\mathcal{L}(A) :=$ *set of accepted* Σ*-trees.*

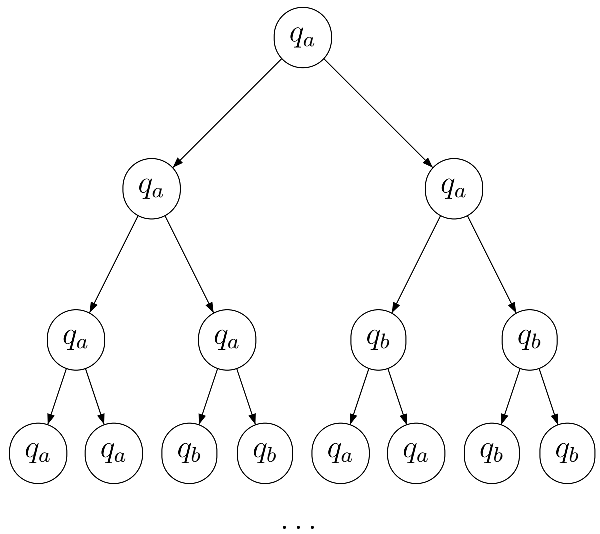
Example: $\{a, b\}$-trees with infinitely many bs on each path.

$\mathcal{A} = (S, s_0, M, c); \Sigma = \{a, b\};$
$S = \{q_a, q_b\}; s_0 = q_a;$
$M = \{(q_a, a, q_a, q_a), (q_b, a, q_a, q_a), (q_a, b, q_b, q_b), (q_b, b, q_b, q_b)\};$
Büchi $F = \{q_b\}$.

Σ-tree:



run:

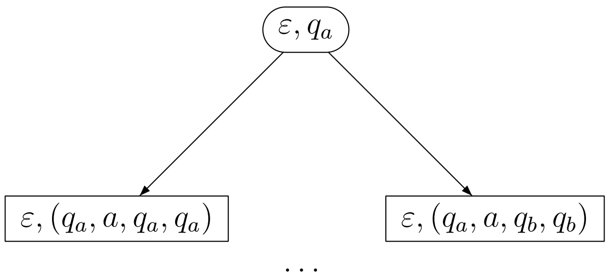


Theorem 1 *A parity tree automaton* $\mathcal{A} = (S, s_0, M, c)$ *accepts an input tree* t *iff Player 0 wins the parity game* $\mathcal{G}_{\mathcal{A},t} = (V_0, V_1, E, c')$ *from position* (ε, s_0)*.*

- $V_0 = \{(w, q) \mid w \in \{0,1\}^*, q \in S\}$;
- $V_1 = \{(w, \tau) \mid w \in \{0,1\}^*, \tau \in M\}$;

- $E = \{((w,q),(w,\tau)) \mid \tau = (q, t(w), q'_0, q'_1), \tau \in M\}$
  $\cup \{((w,\tau),(w',q')) \mid \tau = (q, \sigma, q'_0, q'_1) \ and$
  $((w' = w0 \ and \ q' = q'_0) \ or \ (w' = w1 \ and \ q' = q'_1))\}$;
- $c'(w,q) = c(q)$ *if* $q \in S$;
- $c'(w,\tau) = 0$ *if* $\tau \in M$.

Example:



✚

Proof:

- Given an accepting run r construct a winning strategy f_0:

$$f_0(w,q) = (w, (r(w), t(w), r(w0), r(w1))$$

- Given a memoryless winning strategy f_0 construct an accepting run $r(\varepsilon) = s_0$ $\forall w \in \{0,1\}^*$
  - $r(w0) = q$ where $f_0(w, r(w)) = (w, (\_, \_, q, \_))$
  - $r(w1) = q$ where $f_0(w, r(w)) = (w, (\_, \_, \_, q))$

■

Lemma 1 *For each parity tree automaton $\mathcal{A}$ over Σ-trees there exists a parity tree automaton $\mathcal{A}'$ over $\{1\}$-trees, such that $\mathcal{L}(\mathcal{A}) = \emptyset$ iff $\mathcal{L}(\mathcal{A}') = \emptyset$.*

Proof:

- $S' = S$;
- $s'_0 = s_0$;
- $M' = \{(q, 1, q_0.q_1) \mid (q, \sigma, q_0, q_1) \in M, \sigma \in \Sigma\}$
- $c' = c$

■

Theorem 2 *The language of a parity tree automaton $\mathcal{A} = (S, s_0, M, c)$ is non-empty iff Player 0 wins the parity game $\mathcal{G}_{\mathcal{A},t} = (V_0, V_1, E, c')$ from position s_0.*

- $V_0 = S$;
- $V_1 = M$;
- $E = \{(q, \tau) \mid \tau = (q, 1, q'_0, q'_1), \tau \in M\}$
  $\cup \{(\tau, q') \mid \tau = (q, 1, q'_0, q'_1)$ *and*
  $(q' = q'_0$ *or* $q' = q'_1)\}$;
- $c'(q) = c(q)$ *for* $q \in S$;
- $c(\tau) = 0$ *for* $\tau \in M$.

Theorem 3 *Büchi tree automata are structly weaker than parity tree automata.*

Proof:

- Consider the tree language $T = \{t \in T_{\{a,b\}} \mid$ every branch of t has only finitely many $b\}$
- T is recognized by a parity tree automaton. For example by $\mathcal{A} = (S, s_0, M, c)$ with $S = \{q_a, q_b\}; s_0 = q_a$; $M = \{(q_a, a, q_a, q_a), (q_b, a, q_a, q_a), (q_a, b, q_b, q_b), (q_b, b, q_b, q_b)\}$; $c(q_a) = 0, c(q_b) = 1$.
- T is not recognized by any Büchi tree automaton. Assume, by way of contradiction, that there is a Büchi tree automaton $\mathcal{A} = (S, s_0, M, F)$ such that $\mathcal{L}(\mathcal{A}) = T$.
  - Let $n = |S|$.
  - Consider the input tree t_n, where b appears exactly at nodes $1^+0, 1^+01^+0, \ldots, (1^+0)^n$.
  - $t_n \in T \Rightarrow$ there exists an accepting run r of $\mathcal{A}$ on t_n.
  - On the branch consisting of the finite prefixes of 1^ω there are infinitely many visits to $F \Rightarrow \exists m_0 \in \omega$ such that $r(1^{m_0}) \in F$.
  - Analogously, on the branch consisting of the finite prefixes of $1^{m_0}01^\omega$, there are infinitely many visits to $F \Rightarrow \exists m_1 \in \omega$ such that $r(1^{m_0}01^{m_1}) \in F$.

- Repeating this argument, we obtain $n+1$ positions $1^{m_0}, 1^{m_0}01^{m_1}, \ldots, 1^{m_0}01^{m_1}0\ldots01^{m_n}$ where F is visited.
- There must exist two different nodes u, v on the path to $1^{m_0}01^{m_1}0\ldots01^{m_n}$ such that u is a prefix of v and $r(u) = r(v) \in F$. The path from u to v contains a left turn and therefore contains a node labeled with b.
- We construct a new input tree t_n and a run tree r' by repeating the path from u to v infinitely often:
  * let $v = u \cdot \pi$.
  * $t'_n(x) = t_n(u \cdot y)$ if $x = u \cdot \pi^* \cdot y$ for some shortest $y \in \{0,1\}^*$
    $t'_n(x) = t_n(x)$ otherwise
  * $r'(x) = r(u \cdot y)$ if $x = u \cdot \pi^* \cdot y$ for some shortest $y \in \{0,1\}^*$
    $r'(x) = r(x)$ otherwise
  * r' is accepting: the branch consisting of the finite prefixes of $u \cdot \pi^\omega$ has infinitely many visits to F; all other branches have the same labeling as in r after some finite prefix. Since r is accepting, these branches thus must also visit F infinitely often.
  * Hence t'_n is accepted by $\mathcal{A}$, but $t'_n \notin T$, because the branch consisting of the finite prefixes of $u \cdot \pi^\omega$ has infinitely many bs. Contradiction.

■