Logic and Computation II

Part 6. Automata on infinite objects

Kazuyuki Tanaka

BIMSA

May 6, 2025

北京雁栖湖应用数学研究院
Beijing Institute of Mathematical Sciences and Applications

Logic and Computation II

- **Part 4. Modal logic**
- **Part 5. Modal μ-calculus**
- **Part 6. Automata on infinite objects**
- **Part 7. Recursion-theoretic hierarchies**

Part 6. Schedule (tentative)

- Apr.15, (1) Second-order arithmetic and analytical hierarchy
- Apr.17, (2) Büchi automata
- Apr.22, (3) Safra's theorem
- Apr.24, (4) The decidability of S1S
- May 6, (5) Tree automata
- May 8, (6) The decidability of S2S
- May 13, (7) Finite model theory
- May 15, (8) Parity games

Recap

- Let Ω be a finite set (alphabet) and Ω^ω be the set of ω-words $a_0 a_1 a_2 \cdots$ on Ω.
- A **run** of a nondeterministic automaton $M = (Q, \Omega, \delta, Q_0, \mathrm{Acc})$ on an input $\alpha = a_0 a_1 a_2 \cdots \in \Omega^\omega$ is an infinite sequence of states $q_0 q_1 q_2 \cdots \in Q^\omega$ satisfying:

$$q_0 \in Q_0, \quad (q_i, a_i, q_{i+1}) \in \delta \ (i \geq 0).$$

- By $\mathrm{Inf}(\sigma)$, we denote the set of states that appear infinitely many times in σ.
- A run σ is accepted by an **NBA** with Büchi condition $(F \subseteq Q)$ if $\mathrm{Inf}(\sigma) \cap F \neq \varnothing$; an **NMA** with Muller condition $(\mathcal{F} \subseteq \mathcal{P}(Q))$ if $\mathrm{Inf}(\sigma) \in \mathcal{F}$; an **NRA** with Rabin condition $(\mathcal{F} = \{(G_i, R_i) \mid (1 \leq i \leq k)\}, G_i, R_i \subset Q)$, if there exists i such that $\mathrm{Inf}(\sigma) \cap G_i \neq \varnothing$ and $\mathrm{Inf}(\sigma) \cap R_i = \varnothing$.
- A deterministic automaton with a Büchi/Muller/Rabin condition is called a **DBA**/**DMA**/**DRA**. Then, we have DBA $<$ NBA $=$ DRA $=$ NRA $=$ DMA $=$ NMA
- S1S is the MSO theory of $(\mathbb{N} \cup \mathcal{P}(\mathbb{N}), x+1, \in)$.
- We proved that S1S and NBA have equivalent expressive power. The decision problem of S1S can be reduced to the emptiness problem of NBA.

§6.5. Introducing tree automata

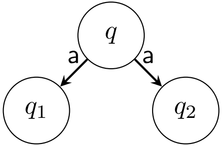
- Fix a finite set of symbols (or labels) Ω.
- An (Ω-)**labeled tree** is an infinite complete binary tree $\{0,1\}^*$ with each vertex labeled by a symbol in Ω. It can be viewed as a function $t : \{0,1\}^* \to \Omega$.

We define a tree automaton that accepts labeled trees.

Definition 6.18

The **tree automaton** $M = (Q, \Omega, \delta, Q_0, Acc)$:

- Q: a set of states,
- $\delta \subseteq Q \times \Omega \times Q^2$: a transition relation,
- $Q_0 \subseteq Q$: a set of initial states, and
- Acc: an acceptance conditions, such as Büchi , Rabin, Muller.

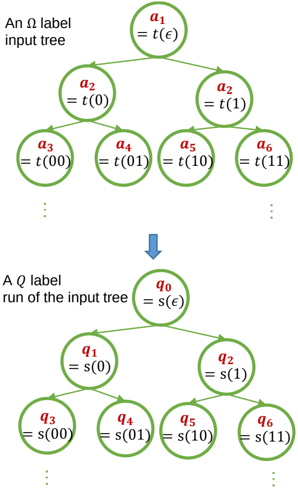


$(q, a, q_1, q_2) \in \delta$ means that by reading a, the state changes from q to (q_1, q_2) at once.

M is **deterministic** if δ is a function ($\delta : Q \times \Omega \to Q^2$) and Q_0 is a singleton set. However, for tree automata, deterministic ones are rarely used.

Run-trees of tree automata

- To determine the acceptance of the input tree, we define a run-tree representing the state transitions.
- For an input Ω-labelled tree $t : \{0,1\}^* \to \Omega$, a **run-tree** of M is a Q-labelled tree $s : \{0,1\}^* \to Q$ such that
  - $s(\epsilon) \in Q_0$, where ϵ is empty and represents the root of the binary tree.
  - for any $u \in \{0,1\}^*$, $(s(u), t(u), s(u0), s(u1)) \in \delta$.
- If M is deterministic then there is only one run-tree for any input tree.
- To simplify the discussion, assume that for any input, a run-tree can be constructed. (Such an automaton is said to be **complete**). This modification is easily done by adding new meaningless states.



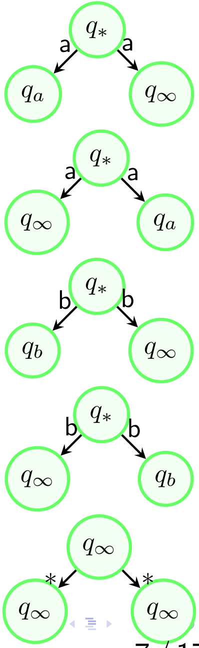
- A (infinite) **path** through the binary tree $\{0,1\}^*$ is a function $f : \mathbb{N} \to \{0,1\}^*$ such that $f(0) = \epsilon$ and $f(n+1)$ is a child (an immediate successor) of $f(n)$ for all n.
- For a Q-labelled tree s and an infinite path α through $\{0,1\}^*$, $s(\alpha)$ denotes the ω-sequence of states (labels) on path α in s.
- An input tree t is accepted by a tree automaton M if there is a run-tree s in which all of the paths $s(\alpha)$ satisfy (one of) the following acceptance conditions.
  - If M is a **Büchi tree automaton** (BTA), then the acceptance condition Acc is $F(\subseteq Q)$: an input tree $t \in L(M)$ if there is a run-tree s in which all its infinite paths $s(\alpha)$ satisfying $\mathrm{Inf}(s(\alpha)) \cap F \neq \varnothing$.
  - If M is a **Muller tree automaton** (MTA), Acc is $\mathcal{F}(\subseteq \mathcal{P}(Q))$: an input tree $t \in L(M)$ if there is a run-tree s in which all its infinite paths $s(\alpha)$ satisfying that $\mathrm{Inf}(s(\alpha)) \in \mathcal{F}$.
  - If M is a **Rabin tree automata**(RTA), Acc is $\mathcal{F} = \{(G_i, R_i) \mid 1 \leq i \leq k\}$, where $G_i, R_i \subset Q$: an input tree $t \in L(M)$ if there is a run-tree s s.t in all its infinite paths $s(\alpha)$ there exists i satisfying $\mathrm{Inf}(s(\alpha)) \cap G_i \neq \varnothing$ and $\mathrm{Inf}(s(\alpha)) \cap R_i = \varnothing$.

Example

- Let $\Omega = \{a, b\}$. Let T_1 be the set of Ω-labelled trees with at least one path in which a appears infinitely many times.
- A BTA $M = (Q, \Omega, \delta, Q_0, F)$ is defined as follows.
  $Q = \{q_a, q_b, q_\infty\}$, $\quad Q_0 = \{q_a\}$, $\quad F = \{q_a, q_\infty\}$,
  $\delta(q_y, x) = \{(q_x, q_\infty), (q_\infty, q_x)\}$, $\quad \delta(q_\infty, x) = \{(q_\infty, q_\infty)\}$,
  where x, y are any combination of a, b.
- Therefore, the acceptance of the input tree t is determined by whether or not q_a appears infinitely in a nondeterministically selected path.
- Thus M accepts language T_1.

Remarks from the viewpoint of analytical hierarchy

The accepting language of any deterministic tree automaton can be expressed as a Π^1_1 statement ($\because$ Its run-tree is uniquely determined). Since T_1 is (fnc-)Σ^1_1 and cannot be simplified any further, it cannot be accepted by any deterministic tree automaton.



$* = a$ or b

- We will prove the decidability of S2S, a monadic second-order theory of 2 successors, by using the expressive equivalence between S2S and MTA.
- The standard model of S2S is

$$(\{0,1\}^* \cup \mathcal{P}(\{0,1\}^*), S_0(x), S_1(x), \in),$$

where $S_i(x)$ is a kind of successor function, i.e., $S_i(w)$ is $w\hat{}i$ for any $w \in \{0,1\}^*$ $(i = 0,1)$. (Note: $w\hat{}i$ is also written as $w \cdot i$ or simply $w\ i$.)

- Let P_a be the set of nodes with label $a \in \Omega$, i.e., $P_a = t^{-1}(a)$. If an S2S formula φ (in an extended language with $\{P_a : a \in \Omega\}$) holds in the structure $(\{0,1\}^* \cup \mathcal{P}(\{0,1\}^*), S_0(x), S_1(x), \in, P_a)_{a\in\Omega}$, we say that the formula φ holds for t.
- Then there is a two-way translation between an MTA M and an S2S formula φ, and for any Ω-labeled tree t,

"M accepts t" is equivalent to "φ satisfies t".

Lemma 6.19

The class of languages accepted by MTA is closed under set union and projections.

Proof

- Let $M_1 = (Q_1, \Omega, \delta_1, Q_0^1, \mathcal{F}_1)$ and $M_2 = (Q_2, \Omega, \delta_2, Q_0^2, \mathcal{F}_2)$ be MTA's. We may assume $Q_1 \cap Q_2 = \varnothing$.
  Then, an MTA that accepts $L(M_1) \cup L(M_2)$ is
  $$N = (Q_1 \cup Q_2, \Omega, \delta_1 \cup \delta_2, Q_0^1 \cup Q_0^2, \mathcal{F}_1 \cup \mathcal{F}_2).$$
- Suppose that a set L of $\Omega_1 \times \Omega_2$-labeled trees is accepted by an MTA $M = (Q, \Omega_1 \times \Omega_2, \delta, Q_0, \mathcal{F})$. An MTA $N = (Q, \Omega_1, \delta', Q_0, \mathcal{F})$ that accepts the projection of L onto Ω_1 is defined as,
  $$(p, a, q_1, q_2) \in \delta' \Leftrightarrow \text{there exists } b \in \Omega_2 \text{ such that } (p, (a, b), q_1, q_2) \in \delta.$$

□

- The difficulty of equivalence of MTA and S2S lies in proving the class of languages accepted by MTA is closed under complement.
- Since MTA is different from DTA and DRA, it is even more difficult to prove its closure under complement than the ω-language case.
- To simplify the original argument of Rabin (1969), Y. Gurevich and L. Harrington (1982) brought in the idea of infinite games and gave an elegant proof.
- They call a strategy that has only bounded memory a **forgetful strategy**, and use the fact that certain games have such winning strategies to simplify the treatment of complements significantly.

ABSTRACT. In 1969 Rabin introduced tree automata and proved one of the deepest decidability results. If you worked on decision problems you did most probably use Rabin's result. But did you make your way through Rabin's cumbersome proof with its induction on countable ordinals? Building on ideas of our predecessors--and especially those of Büchi--we give here an alternative and transparent proof of Rabin's result. Generalizations and further results will be published elsewhere.

The idea to use games is not new. It was aired by McNaughton and exploited in Landweber 1967, Büchi & Landweber 1969 and especially in Büchi 1977 where the complementation problem was reduced (for an able reader) to a certain determinancy result. Our §2 gives such a reduction too. Our §3 provides the necessary determinancy result. When this solution had been reported in several places including Purdue Büchi kindly sent us a manuscript, Büchi 1981. To be sure Büchi proved the determinancy result, and he certainly was the first to do so. His proof still is, however, a very complicated induction on countable ordinals, much more difficult than our §3.

Our games form a special case of games studied in set theory. The most relevant set-theoretic paper is Davis 1964. However the determinancy results of Davis 1964 and other set-theoretic papers do not suffice for our purposes because we are interested only in very special memory-restricted strategies.

Parity condition of PTA

- Subsequently, Emerson and Jutla (1988), McNoughton (1993), Zielonka (1998) and others further simplified the proof by discovering and utilizing the relation between parity tree automata and memoryless (positional) strategies of parity games.

- A function $\pi : Q \to \{0, 1, \ldots, k\}$ is called a **priority function**. A **parity tree automaton** (PTA) is equipped with a priority function as its accepting condition. An input tree is accepted by a PTA, if there exists a run-tree where in each path, the smallest priority of the states appearing infinitely many times is even.

Theorem 6.20

PTA and MTA accept the same languages.

Proof.

It is easy to see that the languages accepted by a PTA can be accepted by a MTA such that $F \in \mathcal{F}$ iff F is a set of states whose smallest priority is even.

- Conversely, given an MTA $M = (Q, \Omega, \delta, Q_0, \mathcal{F})$, we want to construct a PTA $M' = (Q', \Omega, \delta', Q'_0, \pi)$ which accepts the same language.
- Let Q' be the set of permutations of $Q \cup \{\natural\}$ (where $\natural \notin Q$). An element of Q' denotes a **Last Appearing Record** of the states so that the rightmost q corresponds to the current state of M, and $\natural$ represents the place where such q appeared just before now.
- Thus, if $\delta(p, a, r_1, r_2)$ in M and $q_1 \dots q_m \natural q_{m+1} \dots q_n \in Q'$ and $q_n = p, q_i = r_1, q_j = r_2$,

$$\delta'(q_1 \dots q_m \natural q_{m+1} \dots q_n, a, q_1 \dots q_{i-1} \natural q_{i+1} \dots q_n q_i,\ q_1 \dots q_{j-1} \natural q_{j+1} \dots q_n q_j).$$

- Also, the definition of a priority function π is as follows. For $u \natural v \in Q'$,

$$\pi(u \natural v) = \begin{cases} 2|u|, & \{q \in Q : v \text{ contains } q\} \in \mathcal{F} \\ 2|u| + 1, & \{q \in Q : v \text{ contains } q\} \notin \mathcal{F} \end{cases}$$

- Then, $\pi : Q' \to \{0, 1, \dots, 2|Q| + 1\}$.
- Q'_0 can be Q', but a more efficient choice is the set of sequences in Q' with the rightmost belonging to Q_0.

- We compare the run-trees of MTA M and PTA M' for the same input tree.
- A state q that appears finitely (infinitely) many times in a path of the run-tree of M also occurs finitely (infinitely) many times to the right of $\natural$ in the corresponding path of the run-tree of M'.
- Therefore, from a certain time onwards, the states that appear finitely are fixed in a sequence u on the left side of $\natural$, and the states that appears infinitely and $\natural$ are permuted repeatedly. We fix such a sequence u and let V be the set of states not in u.
- If $\natural$ comes to the leftmost in the sequence, that is, if it comes immediately after u, it has the lowest priority. Such cases always occur infinitely. So, V is the set of states appearing infinitely many times. Hence, if a path satisfies the acceptance of M, i.e., $V \in \mathcal{F}$, the lowest priority of states of M' appearing infinitely many time is even, and so it also satisfies the acceptance condition for M'.
- Conversely, consider a path satisfying the acceptance condition of M'. Since the states appearing infinitely with the lowest priority is $u\natural v$ for a sequence v from V, the path also satisfies the acceptance condition of M because the lowest priority must be even.
- Therefore, the accepted tree languages of M and M' are the same. □

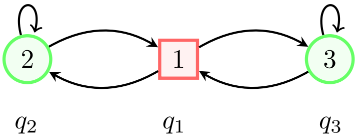
Parity games

A parity game $G = (V_\mathrm{I}, V_\mathrm{II}, E, \pi)$ is a game on a directed graph $(V_\mathrm{I} \cup V_\mathrm{II}, E)$ with a priority function $\pi : V_\mathrm{I} \cup V_\mathrm{II} \to \{0, 1, \cdots, k\}$:

- The set of vertices is partitioned into V_I and V_II ($V_\mathrm{I} \cap V_\mathrm{II} = \varnothing$).
- Two players, player I and II, move a token along the edges of the graph, which results in a path $\rho = v_0 v_1 \cdots$, called a **play**.
- At a vertex $v \in V_\mathrm{I}$ (V_II), it is player I (II)'s turn to choose some v' such that $(v, v') \in E$.
- A strategy for player I is a mapping $\sigma : (V_\mathrm{I} \cup V_\mathrm{II})^{<\omega} V_\mathrm{I} \to V_\mathrm{I} \cup V_\mathrm{II}$.
  A strategy for player II is a mapping $\tau : (V_\mathrm{I} \cup V_\mathrm{II})^{<\omega} V_\mathrm{II} \to V_\mathrm{I} \cup V_\mathrm{II}$.
- The winner of a finite play is the player whose opponent is unable to move.
- Parity winning condition: Player I wins with an infinite play if the smallest parity that occurs infinitely often in the play is even. II wins otherwise
- σ is a winning strategy for player I if whenever he follows σ the resulting play satisfies the parity condition.

Example

Consider the following parity game $G = (V_{\mathsf{I}}, V_{\mathsf{II}}, E, \pi)$, where $V_{\mathsf{I}} = \{q_2, q_3\}$ and $V_{\mathsf{II}} = \{q_1\}$, $\pi(q_i) = i$ for $i = 1, 2, 3$.

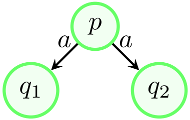


Assume the game starts from q_1, show that player II has a winning strategy.

- A game G is **determined** if one of the two players has a winning strategy.
- A game G is **positionally determined** if one of the two players has a memoryless winning strategy.
- A memoryless strategy for player I is a mapping $\sigma : V_{\mathsf{I}} \to V_{\mathsf{I}} \cup V_{\mathsf{II}}$.
  A memoryless strategy for player II is a mapping $\tau : V_{\mathsf{II}} \to V_{\mathsf{I}} \cup V_{\mathsf{II}}$.
- As we'll show later, parity games are positionally determined.

Characterize a run tree as an infinite game

- Given a PTA $M = (Q, \Omega, \delta, Q_0, \pi)$ and an input tree t, we construct an infinite game $G(M, t)$ in which two players alternately move as follows:
  (1) Player I (Automaton) chooses next pair of states (q_1, q_2) from $\delta(p, a)$.
  (2) Player II (Path Finder) chooses either 0 or 1 for the next direction.



- The **goal of the Path Finder** is to find a path $\alpha \in \{0, 1\}^*$ in the run-tree s that does not satisfy the acceptance condition, whereas the **goal of the Automaton** is to find the Q labels of the run-tree so that the label sequence satisfies the acceptance conditions.
- Player I (automaton) wins in $G(M, t)$ if the label string $s(\alpha)$ produced by the two players satisfies the accepting condition of M.
- Thus "M accepts $t \Leftrightarrow$ The automaton has a winning strategy in $G(M, t)$."
- Assume the determinacy of this game (one of players has a winning strategy),
  "M does not accept $t \Leftrightarrow$ The path finder has a winning strategy in $G(M, t)$."
- For the moment, we also assume the following (which we will prove in next week).
  "The parity game has a memoryless winning strategy."

Thank you for your attention!