Automata, Logic and Games: Theory and Application

2. Parity Games, Tree Automata, and S2S

Luke Ong

University of Oxford

TACL Summer School
University of Salerno, 14-19 June 2015

Lecture Course Outline

Part 1: Foundations. Ideas and some technical details.

1. Büchi Automata and S1S: Legacy of Church & Büchi
2. Parity Games, Tree Automata, and S2S: Legacy of Rabin

Part 2: Active research topic. Mainly ideas.

Higher-Order Model Checking

Lecture Outline

1. Parity Games
2. Binary Trees and Tree Automata
3. S2S and Rabin's Tree Theorem
4. Nondeterministic Parity Tree Automata (NPT)
5. Alternating Parity Tree Automata (APT)
6. Closure Properties of APT
7. Proof of Rabin's Tree Theorem

Parity Game $\langle V_V, V_R, E, v_0, \Omega \rangle$ $(V_V \cap V_R = \varnothing)$

A parity game is played over a digraph $\langle V_V \cup V_R, E \rangle$ by V (Verifier) and R (Refuter).

V_V (resp. V_R) is the set of vertices owned by V (resp. R).

Each vertex v has a priority $\Omega(v)$, where $\Omega : V_V \cup V_R \to \{ 0, \cdots, p \}$ with $p \geq 0$

Rules

- A token is placed on the start vertex v_0.
- If the token is on $v \in V_V$, then V chooses an outgoing edge $(v, v') \in E$, and moves the token onto v'; similarly if $v \in V_R$.

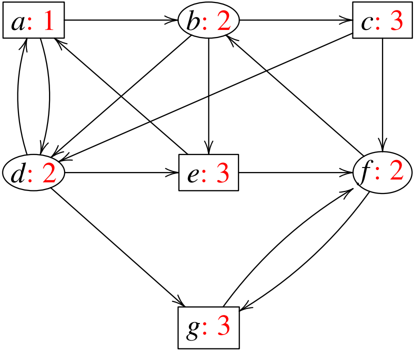
Let π be the maximal path in the digraph tracced out by a play.

Who wins π?

- If π is finite, the player who owns the last vertex loses.
- If π is infinite, V wins if π satisfies parity: the least infinitely-occurring priority in π is even; otherwise R wins.

Example: parity game

Circled vertices belong to Verifier; boxed belong to Refuter. Priorities are red numbers.



Recall: Verifier (circle) wins if the least infinitely-occurring priority is even.
Does Verifier have a winning strategy from f? from d? from a?

[Ans: Verifier has a winning strategy from a, i.e., $b \mapsto c,\ f \mapsto g,\ d \mapsto g$.]

Parity games: a central topic in algorithmic verification

A strategy is memoryless (= history-free = positional) if it depends, not on the history, but only on the last vertex of the play.

Theorem (Martin 1975, Mostowski 1991, Emerson & Jutla 1991)

Parity games are (memoryless) determined: from every vertex, exactly one of V *and* R *has a (memoryless) winning strategy.*

PARITY (Given a finite parity game and a start vertex, does V have a winning strategy?) is in **NP** $\cap$ **co-NP**.

Conjecture

PARITY in **P**.

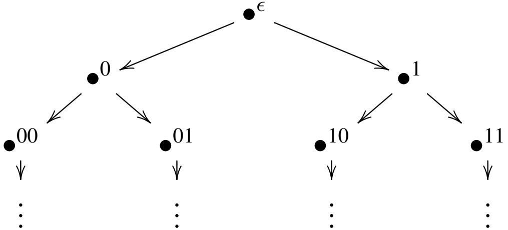
Parity games are ubiquitous in algorithmic verification

Standard qualitative model checking problems about reactive systems (Does a transition system satisfy a given temporal or modal property?) reduce to PARITY.

Σ-labelled (infinite, full) binary trees

A Σ-labelled binary tree is a function $t : \{ 0, 1 \}^* \to \Sigma$.

I.e. the label of the tree t at node $u \in \{ 0, 1 \}^*$ is $t(u) \in \Sigma$.



Write $\mathfrak{T}^\omega_\Sigma$ for the collection of Σ-labelled binary trees.

A tree language is just a subset of $\mathfrak{T}^\omega_\Sigma$.

A path of a tree is a sequence $\pi = u_0\, u_1\, u_2 \cdots$ of tree nodes whereby $u_0 = \epsilon$ (the root of the tree) and $u_{i+1} = u_i\, 0$ or $u_{i+1} = u_i\, 1$, for every $i \geq 0$.

A nondeterministic tree automaton A (for Σ-labelled binary trees) is a quintuple, $(Q, \Sigma, q_0, \Delta, Acc)$, where

- Q is the finite set of states, q_0 is the initial state
- $\Delta \subseteq Q \times \Sigma \times Q \times Q$ is the transition relation, and
- Acc is the acceptance condition (such as Büchi, Muller, Parity, etc.).

The automaton is deterministic if for every $q \in Q$ and $a \in \Sigma$, there is at most one transition (i.e. quadruple) in Δ the first two components of which are q and a.

Run-tree and acceptance by a Büchi tree automaton

A run-tree is a Q-labelled binary tree such that the root is labelled q_0, and the labels respects the transition relation.

Formally a run-tree of a tree automaton A over a tree t is an assignment of states to tree, i.e. a function $\rho : \{ 0, 1 \}^* \to Q$, such that

- $\rho(\epsilon) = q_0$, and
- for all $u \in \{ 0, 1 \}^*$, $(\rho(u), t(u), \rho(u\,0), \rho(u\,1)) \in \Delta$.

A Büchi tree automaton $A = (Q, \Sigma, q_0, \Delta, F)$ accepts a tree t just if there exists a Büchi-accepting run-tree ρ of A over t, i.e., in every path of ρ, a final state from F occurs infinitely often.

The tree language recognised by the tree automaton A, denoted $L(A)$, is the set of trees accepted by A.

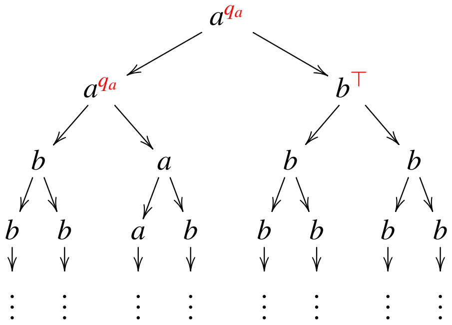
Example

Let T_1 be the set of $\{ a, b \}$-labelled binary trees t such that t has a path with infinitely many a's.

The Büchi tree automaton $(\{ q_a, q_b, \top \}, \{ a, b \}, q_a, \Delta, \{ q_a, \top \})$ where

$$\Delta \; : \; \begin{cases} (q_a/q_b, a) & \mapsto & \{ (q_a, \top), (\top, q_a) \} \\ (q_a/q_b, b) & \mapsto & \{ (q_b, \top), (\top, q_b) \} \\ (\top, a/b) & \mapsto & \{ (\top, \top) \} \end{cases}$$

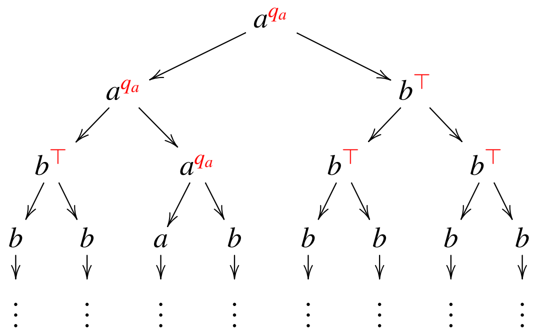
recognises the language T_1 where $*$ "matches every symbol".



$A = (\{\, q_a, q_b, \top \,\}, \{\, a, b \,\}, q_a, \Delta, \{\, q_a, \top \,\})$ where

$$\Delta \ : \left\{ \begin{array}{rcl} (q_a/q_b, a) & \mapsto & \{\, (q_a, \top), (\top, q_a) \,\} \\ (q_a/q_b, b) & \mapsto & \{\, (q_b, \top), (\top, q_b) \,\} \\ (\top, a/b) & \mapsto & \{\, (\top, \top) \,\} \end{array} \right.$$

recognises $T_1 := \{\, t \mid t$ has a path with infinitely many a's $\}$.

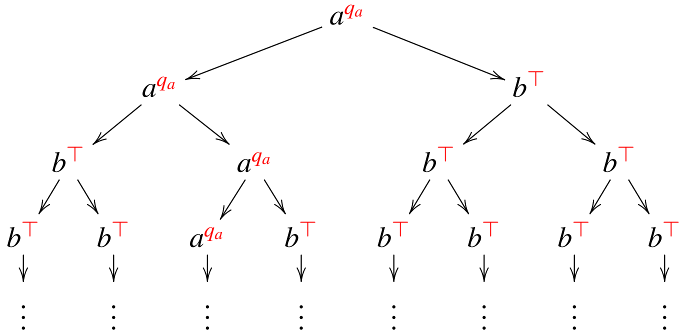


$A = (\{ q_a, q_b, \top \}, \{ a, b \}, q_a, \Delta, \{ q_a, \top \})$ where

$$\Delta : \begin{cases} (q_a/q_b, a) & \mapsto & \{ (q_a, \top), (\top, q_a) \} \\ (q_a/q_b, b) & \mapsto & \{ (q_b, \top), (\top, q_b) \} \\ (\top, a/b) & \mapsto & \{ (\top, \top) \} \end{cases}$$

recognises $T_1 := \{ t \mid t$ has a path with infinitely many a's $\}$.

A Büchi-accepting run-tree

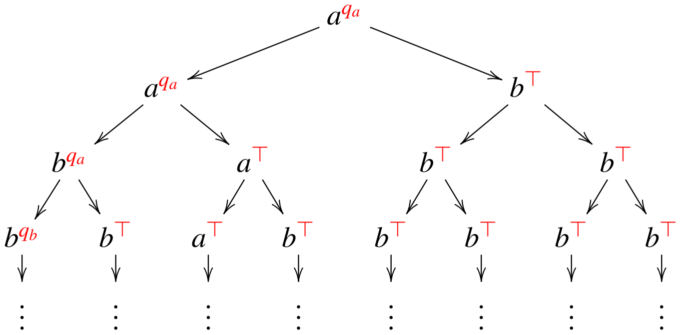


$A = (\{ q_a, q_b, \top \}, \{ a, b \}, q_a, \Delta, \{ q_a, \top \})$ where

$$\Delta \; : \; \begin{cases} (q_a/q_b, a) & \mapsto \quad \{ (q_a, \top), (\top, q_a) \} \\ (q_a/q_b, b) & \mapsto \quad \{ (q_b, \top), (\top, q_b) \} \\ \quad (\top, a/b) & \mapsto \quad \{ (\top, \top) \} \end{cases}$$

recognises $T_1 := \{ t \mid t$ has a path with infinitely many a's $\}$.

A wrongly guessed run-tree!



$A = (\{ q_a, q_b, \top \}, \{ a, b \}, q_a, \Delta, \{ q_a, \top \})$ where

$$\Delta \ : \ \begin{cases} (q_a/q_b, a) & \mapsto \quad \{ (q_a, \top), (\top, q_a) \} \\ (q_a/q_b, b) & \mapsto \quad \{ (q_b, \top), (\top, q_b) \} \\ (\top, a/b) & \mapsto \quad \{ (\top, \top) \} \end{cases}$$

recognises $T_1 := \{ t \mid t$ has a path with infinitely many a's $\}$.

The Logical System S2S (Monadic Second-order Logic of 2 Successors)

The logical system ***S2S*** is defined over

- first-order variables $x, y, \cdots$ ranging over $\{0, 1\}^*$ (nodes in the full binary tree) and
- second-order variables $X, Y, \cdots$ ranging over $2^{\{0,1\}^*}$ (sets of nodes of the full binary tree).

Terms are built up from first-order variables and ϵ by the two successors, represented as concatenation with 0 and 1 respectively.

Let s and t be terms. The atomic formulas are

- $s \in X$ "s is in X"
- $s \leq t$ "s is a prefix of t"
- $s = t$ "s is equal to t".

S2S formulas are built up from the atomic formulas using the standard boolean connectives, and closed under first- and second-order quantifiers $\exists$ and $\forall$.

The logical structure of the infinite full binary tree is $\mathbf{t}_2 = (\mathbb{B}^*, \epsilon, S_0, S_1)$ where S_i is the i-th successor function: $S_0(u) = u\,0$ and $S_1(u) = u\,1$ for $u \in \mathbb{B}^*$.

S2S-formulas $\varphi(X_1, \cdots, X_n)$, with free 2nd-order variables from $X_1, \cdots, X_n$, are interpreted in expanded structures $\widehat{t} = (\mathbf{t}_2, P_1, \cdots, P_n)$ where each $P_i \subseteq \mathbb{B}^*$. Write

$$\widehat{t} \vDash \varphi(X_1, \cdots, X_n)$$

just if $\widehat{t}$ satisfies $\varphi(\overline{X})$.

We identify $\widehat{t}$ with the infinite tree $t \in \mathfrak{T}^{\omega}_{\mathbb{B}^n}$ such that for each $u \in \mathbb{B}^*$, we have

$$t(u) = (b_1, \cdots, b_n) \quad \text{where } b_i = 1 \;\leftrightarrow\; u \in P_i.$$

Given an S2S formula $\varphi(\overline{X})$ the tree language defined by $\varphi(\overline{X})$ is the set

$$L(\varphi(X_1, \cdots, X_n)) := \{\, t \in \mathfrak{T}^{\omega}_{\mathbb{B}^n} \mid \widehat{t} \vDash \varphi(\overline{X}) \,\}.$$

Example. Take $\varphi(X_1) := \exists Y.\mathit{infinite}(Y) \wedge \forall y.(y \in Y \to y \in X_1)$, where $\mathit{infinite}(Y)$ says that Y is an infinite set. Then

$$L(\varphi(X_1)) := \{\, t \in \mathfrak{T}^{\omega}_{\mathbb{B}} : \widehat{t} \text{ has infinitely many positions where } P_1 \text{ holds} \,\}.$$

Our aim is to prove:

Theorem (Rabin 1969)

For $n \geq 0$, a tree language $T \subseteq \mathfrak{T}^{\omega}_{\mathbb{B}^n}$ is S2S-definable if, and only if, it is recognisable by a nondeterministic parity tree automaton (NPT).

Proof steps: (familiar pattern from Büchi's S1S)
$\Rightarrow$: closure of NPT under complementation ($\neg$), union ($\vee$), and projection ($\exists X$).
$\Leftarrow$: encode existence of an accepting run-tree as a S2S formula

Corollary (Rabin Tree Theorem)

Since non-emptiness of NPT is decidable, the theory S2S is decidable.

Evolution of proof: Rabin (1969), Gurevich & Harrington (1982), Löding (2011).

Notation

Following Vardi and Kupferman, we use acronym $X\ Y\ Z$ where

- X ranges over automaton modes: deterministic, nondeterministic and alternating,
- Y ranges over acceptance / winning conditions: Büchi, Muller, Rabin, Streett, parity, and weak,
- Z ranges over input structures: words and trees.

For example, DMW and NPT are shorthand for deterministic Muller word automaton and nondeterministic parity tree automaton respectively.

Parity Tree Automata

A ***nondeterministic parity tree automaton*** is a tuple $A = (Q, \Sigma, q_0, \Delta, \Omega)$ with priority map $\Omega : Q \to \{ 0, \cdots, k \}$.

A accepts a tree t just if there is a run-tree ρ of A over t such that for every path of ρ, the least priority that occurs infinitely often is Verifier.

Example

Recall T_1 is the set of $\{ a, b \}$-labelled binary trees t such that t has a path with infinitely many a's.

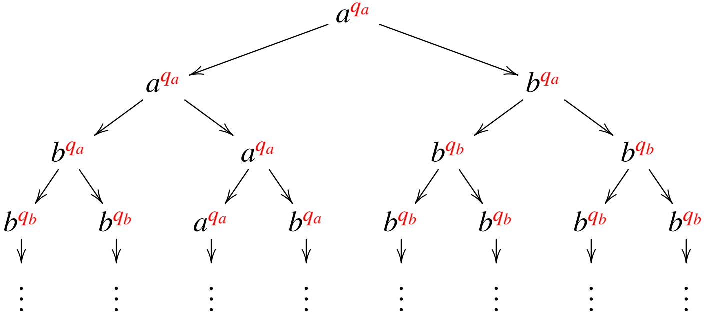
The nondeterministic parity tree automaton $(\{ q_a, q_b, \top \}, \{ a, b \}, q_a, \Delta, \Omega)$ where

$$\Delta \; : \; \begin{cases} (q_a/q_b, a) & \mapsto \quad \{ (q_a, \top), (\top, q_a) \} \\ (q_a/q_b, b) & \mapsto \quad \{ (q_b, \top), (\top, q_b) \} \\ \quad (\top, a/b) & \mapsto \quad \{ (\top, \top) \} \end{cases}$$

and $\Omega : q_a \mapsto 0, \top \mapsto 0; q_b \mapsto 1$ recognises the language T_1.

Indeed, every nondeterministic Büchi automaton can be viewed as a nondeterministic parity automaton with priorities $\{ 0, 1 \}$.

E.g. This tree is not in $T_2 := \{ t \mid \text{every path of } t \text{ has only finitely many } a\text{'s} \}$.



Deterministic parity $(\{ q_a, q_b \}, \{ a, b \}, q_a, \Delta, \Omega)$ where

$$\Delta \; : \; \begin{cases} (q_a/q_b, a) & \mapsto \quad \{ (q_a, q_a) \} \\ (q_a/q_b, b) & \mapsto \quad \{ (q_b, q_b) \} \end{cases}$$

and $\Omega : q_a \mapsto 1; q_b \mapsto 2$ recognises T_2.

N.B. For each path π of t, there are infinitely many b's (resp. a's) on π iff the corresponding path of the unique run-tree over t has infinitely many q_b's (resp. q_a's).

Acceptance Parity Game

Given a NPT $A = (Q, \Sigma, q_I, \Delta, \Omega)$ and a tree t, we define a parity game, called the ***acceptance parity game***, $\mathcal{G}_{A,t} = (V_\mathrm{V}, V_\mathrm{R}, E, (\epsilon, q_I), \lambda, \Omega')$ as follows.

- $V_\mathrm{V} = \{ 0, 1 \}^* \times Q$
- $V_\mathrm{R} = \{ 0, 1 \}^* \times (Q \times Q)$
- for each vertex $(v, q) \in V_\mathrm{V}$, for each transition $(q, a, q_0, q_1) \in \Delta$ with $t(v) = a$, we have $((v, q), (v, (q_0, q_1)) \in E$
- for each vertex $(v, (q_0, q_1)) \in V_\mathrm{R}$, we have

$$((v, (q_0, q_1)), (v\,0, q_0)), ((v, (q_0, q_1)), (v\,1, q_1)) \in E.$$

- $\Omega' : (v, q) \mapsto \Omega(q)$ and $(v, (q_0, q_1)) \mapsto \max(\Omega(q_0), \Omega(q_1))$.

Lemma

Verifier has a winning strategy in $\mathcal{G}_{A,t}$ from vertex (ϵ, q_I) if and only if $t \in L(A)$.

Given a NPT $A = (Q, \Sigma, q_I, \Delta, \Omega)$, we define a parity game, called the ***non-emptiness parity game***, $\mathcal{G}_A = (V_{\mathrm{V}}, V_{\mathrm{R}}, E, q_I, \lambda, \Omega')$ as follows.

- $V_{\mathrm{V}} = Q$
- $V_{\mathrm{R}} = \Delta$
- for each vertex $q \in V_{\mathrm{V}}$, and for each transition $(q, a, q_0, q_1) \in \Delta$, we have $(q, (q, a, q_0, q_1)) \in E$
- for each vertex $(q, a, q_0, q_1) \in V_{\mathrm{R}}$, we have

$$((q, a, q_0, q_1), q_0), ((q, a, q_0, q_1), q_1) \in E$$

- $\Omega' : q \mapsto \Omega(q)$ and $(q, a, q_0, q_1) \mapsto \Omega(q)$.

Lemma

Verifier has a winning strategy in $\mathcal{G}_A$ from vertex q_I if and only if $L(A) \neq \varnothing$.

Theorem (Rabin Basis Theorem)

1. *The emptiness problem for NPT is decidable.*
2. *If an NPT accepts some tree then it accepts a regular tree.*

Let $A = (Q, \Sigma, q_I, \Delta, \Omega)$ be an NPT.

1. The non-emptiness game $\mathcal{G}_A$ is a parity game on a finite graph, hence solvable: there is an algorithm to determine the winning region for each player.
   By the previous lemma, Verifier has a winning strategy from q_I iff $L(A) \neq \varnothing$.
2. Omitted.

Alternating (tree) automata

- An automaton mode, introduced by Chandra & Kozen (1986?). Complexity theory motivation.
- Generalises determinism and nondeterminism.
- Natural duality: counterpart of 2-player game

Define the set $\mathcal{B}^+(X)$ of positive boolean formulas consisting of formulas built up from atoms in X using $\vee$ and $\wedge$.

The transition function $\delta : Q \times \Sigma \to \mathcal{B}^+(\{0, 1\} \times Q)$ of an alternating tree automaton describes which moves are controlled by players:

- disjunctions by Verifier
- conjunction by Refuter.

and negation swaps the rôle of player.

Example. $\delta(q, a) := (0, q) \wedge (1, q) \wedge \big((0, q_b) \vee (1, q_b)\big)$

Assume in state q at node x in t with $t(x) = a$. Refuter could choose $(0, q_b) \vee (1, q_b)$, then Verifier could choose $(1, q_b)$.

Automaton would then move to the right successor of x (the node $x\,1$), change

Alternating parity tree automata

An ***alternating parity tree automaton*** A (for Σ-labelled binary trees) is a tuple $(Q, \Sigma, q_I, \delta, \Omega)$ where

- Q is the finite set of states, q_I is the initial state
- $\delta : Q \times \Sigma \to \mathcal{B}^+(\{ 0, 1 \} \times Q)$ is the transition function, and
- $\Omega : Q \to \{ 0, \ldots, k \}$ is the priority function.

We define the ***language*** $L(A)$ recognised by A to be the set of trees t such that Verifier has a winning strategy in the (alternating) acceptance parity game $\mathcal{G}_{A,t}$ starting from vertex (ϵ, q_I).

Acceptance Game for Alternating Parity Automata

Given APT $A = (Q, \Sigma, q_I, \delta, \Omega)$ and tree t, the ***acceptance parity game*** $\mathcal{G}_{A,t} = (V_V, V_R, E, (\epsilon, q_I), \Omega')$ is defined as follows.

Let $U := \{ 0, 1 \}^* \times Q$ and $V := U \cup (\{ 0, 1 \}^* \times \mathcal{B}^+(\{ 0, 1 \} \times Q))$.

- V_V is the set of nodes of the form (x, q), or nodes of the form (x, ψ) for ψ a disjunction or a single atom;
- V_R is the set of nodes of the form (x, ψ) for ψ a conjunction;
- for all $q \in Q$, $x \in \{ 0, 1 \}^*$, $d \in \{ 0, 1 \}$, and $\psi \in \mathcal{B}^+(\{ 0, 1 \} \times Q)$:

$$\begin{array}{rl} ((x, q), (x, \delta(q, t(x))) \in E & \\ ((x, \psi_1 \wedge \psi_2), (x, \psi_i)) \in E & \text{for } i \in \{ 1, 2 \} \\ ((x, \psi_1 \vee \psi_2), (x, \psi_i)) \in E & \text{for } i \in \{ 1, 2 \} \\ ((x, (d, q)), (x\,d, q)) \in E & \end{array}$$

$$\Omega' : \begin{cases} (x, q) & \mapsto \quad \Omega(q) \\ (x, \psi) & \mapsto \quad \max \Omega(Q) \end{cases}$$

Example

Consider $T_3 := \{ t \in \mathfrak{T}^{\omega}_{\{a,b\}} :$ below every a-node there is a b-node $\}$.
Define an APT $A := (\{ q, q_b, \top \}, \{ a, b \}, q, \delta, \Omega)$ where

$$\delta : \begin{cases} (q,a) & \mapsto & (0,q) \wedge (1,q) \wedge \big((0,q_b) \vee (1,q_b)\big) \\ (q,b) & \mapsto & (0,q) \wedge (1,q) \\ (q_b,a) & \mapsto & (0,q_b) \vee (1,q_b) \\ (q_b,b),(\top,a/b) & \mapsto & (0,\top) \end{cases}$$

and $\Omega : q, \top \mapsto 0; q_b \mapsto 1$.

Consider some tree $t \in \mathfrak{T}^{\omega}_{\{a,b\}}$.

In state q, Refuter chooses a path in t. If he sees a, he can switch to q_b, which represents a challenge to Verifier to witness a b below the current a-node.

In state q_b, Verifier chooses a path. If she sees b, then she moves to a sink state $\top$ with priority 0, so Verifier wins. If not, then Verifier remains forever in state q_b with priority 1, so Verifier loses.

Special types of alternating automata

Nondeterministic tree automaton

Alternating automaton where every transition is of the form $\bigvee_i (0, q_0^i) \wedge (1, q_1^i)$.

In other words, a nondeterministic automaton is an alternating automaton that sends exactly one state to each successor.

Deterministic tree automaton

Alternating automaton where every transition is of the form $(0, q_0) \wedge (1, q_1)$.

Theorem (Closure under Complementation)

Given an APT A, there is an algorithm to construct an APT for $\mathfrak{T}^{\omega}_{\{a,b\}} \setminus L(A)$.

Dualise the transition function and acceptance condition of $A = (Q, \Sigma, q_I, \delta, \Omega)$.

- Dual of $\delta : Q \times \Sigma \to (\mathcal{B}^+(\{0, 1\} \times Q)$ is the transition function $\widetilde{\delta}$ obtained by exchanging $\vee$ and $\wedge$ in all formulas in δ.
- Dual of the parity condition Ω, is $\widetilde{\Omega} : q \mapsto \Omega(q) + 1$.

Dualisation switches the rôles of Refuter and Verifier in the acceptance games. Thanks to determinacy of parity games, $\widetilde{A} := (Q, \Sigma, q_I, \widetilde{\delta}, \widetilde{\Omega})$ recognises $\mathfrak{T}^{\omega}_{\{a,b\}} \setminus L(A)$.

Lemma (Closure under Union and Intersection)

Given APT A_1 and A_2, there is an algorithm to construct an APT for $L(A_1) \cup L(A_2)$ and an APT for $L(A_1) \cap L(A_2)$.

Straightforward for alternating automata (Exercise).

Given $s \times t \in \mathfrak{T}^{\omega}_{\Gamma \times \Sigma}$ where $s \times t : u \mapsto (s(u), t(u))$, the ***$\Sigma$-projection*** of $s \times t$ is the tree t, and the Σ-projection of the language $T \subseteq \mathfrak{T}^{\omega}_{\Gamma \times \Sigma}$ is denoted $\pi_\Sigma(T)$.

Lemma (Closure under Projection)

Given an APT recognising the language T of $(\Gamma \times \Sigma)$-labelled binary trees, there is an algorithm to construct an APT recognising $\pi_\Sigma(T)$.

Straightforward for nondeterministic automata, but challenging for alternating automata!

Goal. Prove that APT can be simulated by NPT.

Simulation of APT by NPT

Theorem (Simulation of APT by NPT)

Given an APT A, there is an algorithm to construct an NPT B such that $L(A) = L(B)$.

The proof relies on two fundamental results:

1. Memoryless determinacy of parity games, and
2. Complementation and "determinising" NBW to DPW.

Informally, on input t,

- B guesses an annotation of t with a memoryless strategy for Verifier in the acceptance game $\mathcal{G}_{A,t}$, and
- B runs a DPW on each branch of this annotated tree in order to check that the strategy is winning.

Theorem (Rabin 1969)

A tree language $T \subseteq \mathfrak{T}^{\omega}_{\mathbb{B}^n}$ is S2S-definable if and only if it is recognisable by a NPT.

$\Leftarrow$: Given an NPT, write an equivalent S2S-formula: formula asserts the existence of a run-tree of APT.

$\Rightarrow$: Given S2S-formula, construct an equivalent NPT by induction on the structure of the formula, using closure (and equivalence between APT and NPT) under complementation, union and projection.

Theorem (Rabin's Tree Theorem)

Since non-emptiness of NTP is decidable, so is the theory S2S.