Logic, Automata, and Game Theory

With Applications to Computer Science

Jarren Briscoe

A research project paper presented for Formal Design

Formal Design
Dr. Brian Rague
Department of Computer Science
Weber State University
December 3, 2019

1 Abstract

This paper surveys the relations among formal systems, decidability, finite automata, and game theory. An emphasis is placed on monadic second-order logic which Büchi used to create ω-regular automata. Extending Büchi's work, Rabin found that the monadic second-order theory of the infinite complete binary tree was also decidable. Part of Rabin's proof showed that ω-tree automata could be equally expressive to ω-regular automata. Complementing Rabin's paper, a summary of Gurevich and Harrington's game-theoretic proof for ω-tree automata's closure under complementation is summarized [13, 11, 5].

2 Introduction

2.1 Formal systems

A formal system is a quintuple (P, O, G, A, I) where:

- P is the set of primitive variables (the alphabet).
- O is the set of operator symbols.
- G is the grammar which concerns syntax and grammar. If a sentence is compliant with G, it called a well-defined sentence.
- A is the null, finite, or countably infinite set of axioms.
- I is the set of inference rules. Inference rules are the set of premises that have deductively valid conclusions. If a conclusion is deductively valid, then there can be no other explanation or conclusion of the premises. This is opposed to inductive reasoning where the premises give inconclusive evidence. An inductively valid conclusion is the most likely case given our premises. "$\vdash$" is a common inference symbol.

2.2 History of formal systems

The 19^{th} century was a turning point in mathematics and logic as many paradoxes and non-intuitive properties emerged. Continuous, yet nowhere differentiable functions [25] and the Banach-Tarski paradox [24] were two

examples of developments that could not be argued by the traditional appeal to intuition [23].

While formal systems were still in their infancy, Gödel's incompleteness theorems struck a major blow to many researchers attempting to formulate a perfect formal system. Cleverly, Gödel proved that for any set of consistent axioms, there existed true yet unprovable propositions [22]. Complementing Gödel's work, Alan Turing found that no formal system could accept all languages (using self-referencing in his proof) [21]. Despite these revolutionary and terrible proofs, many problems are still solvable.

With the pressing topic of decidability in formal systems, Büchi began exploring automata and formal systems to discover the limits of what can be expressed and decidable. Finally in 1962, Büchi proved that ω-regular languages are precisely the ones definable in the monadic second-order theory of $\mathbb{N}$ under one successor function (S1S). Later, Rabin extended Büchi's work and showed that the monadic second-order theory of the infinite complete binary tree was also decidable (S2S).

2.3 Nth-order logics

Definitions Philosophers define logic as having ontological and epistemological commitments. Ontological commitments are the relationship between the language and the world while epistemological commitment is the logic's obligation to uphold the truth.

Zeroth-order logic More commonly known as propositional logic, zeroth-order logic's can only represent facts. Propositional logic is decidable since any zeroth-order formula can be evaluated using the truth-table method. The truth-table method allows many facts to be inferred. An example of inference:

$\{P \longrightarrow Q\} \wedge \{P \longrightarrow \neg Q\} \vdash \neg P$

First-order logic Quantifying over the subsets of propositional logic allows first-order logic to extend its ontological domain to {facts, objects, relations}. Examples of relations include safety and liveness properties. First-order logic is undecidable due to the halting problem [21]. However, the complete recursively enumerable subset of first-order theory is decidable.

Second-order logic Following the N^{th}-order trend, second-order logic quantifies over the subsets of first-order logic. This allows us to express properties of properties. Without second-order logic, the infiniteness of a randomly rooted tree could not be expressed. Define S as the subset of nodes in the tree, ϕ as the root node, and π as the parent-child relationship [19]. Then:

$\exists S\left[[\phi \in S] \wedge \left[\forall u \in S\left[\exists v \in S | \pi\left(v\right) = u\right]\right]\right]$

Behavior Exactly as first-order logic is undecidable, so is second-order logic. However, second-order logic is less well behaved than first-order logic. Each of the three theorems listed below are true for first-order logic and for all Henkin models (a model with a first-order structure (U) and a set of subsets over the domain of U), yet they fail when applied to the full second-order logic.

- Compactness theorem: a set of sentences (Q) has a model if and only if all sentences in Q also have a model.
- Lowenheim-Skolem theorem: if a sentence has an infinite Henkin model, then it is countably infinite.
- Completeness theorem: a sentence is provable if and only if the sentence holds for all Henkin models.

N^{th}-order logic N^{th}-order logics, also called higher-order logics (HOL), simply quantify the subsets of the $(N\text{-}1)^{th}$ logic. While HOL are technically stronger than second-order logic, second-order logic has Turing-equivalent decision problems; therefore, every logic in HOL is no more expressive than second-order logic [23].

As seen above, the more expressive a language is, the less well behaved it seems to be. Fortunately, there are fragments of N^{th}-order logic that are more expressible than propositional logic, and are decidable.

Table of some logics

Language	**Ontological commitment**	**Epistemological commitment**
Zeroth-order logic	facts	true, false, unknown
First-order logic	facts, objects, relations	true, false, unknown
Second-order logic	facts, objects, relations, relations of relations	true, false, unknown
N^{th}-order logic	quantifies over the subsets of $(N\text{-}1)^{th}$-order logic	true, false, unknown
First-order temporal logic	facts, objects, relations, time	true, false, unknown
Fuzzy logic	facts with degree of truth $\in$ [0,1]	some known interval within [0,1]

2.4 Fragments of Nth-order logic

2.4.1 Restricted quantifiers

Existential second-order logic (ESO) ESO restricts second-order logic by only allowing the existential quantifier over second-order expressions. However, ESO does not restrict quantifiers over first-order expressions. Fagin's theorem proves that ESO precisely coincides with NP [33]. Consequently, ESO is analogous to the existential acceptance condition of a nondeterministic Turing machine (TM).

Universal second-order logic (USO) The complement of ESO, USO precisely coincides with co-NP [33]. As the complement of ESO, USO is analogous to the universal acceptance condition of a TM (or equivalently the existential rejection condition).

ESO $\overset{?}{\longleftrightarrow}$ USO The well-known identity $\neg\exists\neg\phi(x) \equiv \forall\phi(x)$ cannot be used to easily translate ESO formula to a corresponding USO form (or vice versa) since ESO's closure under negation has not been proven nor disproven. Proving ESO $\neq$ USO would effectively prove NP $\neq$ co-NP; therefore P $\neq$ NP[19].

2.4.2 Monadic second-order logic (MSO)

MSO is a fragment of second-order logic that is restricted to a single predicate. Proving that MSO was decidable started with Büchi proving that the second-order theory of one successor function (S1S) was precisely the set of ω-regular languages [13]. Later Rabin concisely proved that the second-order theory of two successor functions (S2S) was decidable and therefore MSO was decidable.

Additionally, Courcelle proved that not only is MSO decidable, a graph property expressible by MSO can be decided in linear time—given the graph's tree decomposition has a constant treewidth [18].

3 Checking MSO properties of graphs

Application to ω-languages Since ω-regular expressions are precisely those described in S1S which is a fragment of MSO, all upper-bound complex-

ities described for MSO properties in this section can be applied to ω-regular properties.

3.1 Tree decomposition

Tree decomposition transforms a graph to a tree by mapping a subset of connected vertices to each tree node. Formally, a graph G = (V, E) maps into a tree decomposition D = (S, T) where

- S = $\{S_1, ..., S_n\}$. Each $S_i \in S$ corresponds to a subset (or a bag) of V.
- T is a tree whose nodes are the bags S_i. The following properties must hold for T:
  - The union of all bags in T equals V. $\bigcup_{S_i \in T} S_i = V$.
  - For all edges in E, the corresponding nodes (v,w) are in at least one bag. $\forall E(v, w) \in E.[\exists S_i | (v \wedge w) \in S_i)]$
  - If any two bags share common vertices, then every node in the path between the two bags also share the common vertices. This is known as the running intersection property or coherence [17]. $((v \in S_i \wedge v \in S_j) \wedge (\pi = UniquePath(S_i, S_j)) \rightarrow \forall S_k | (S_k \in \pi).S_i \cap S_j \subseteq S_k$

3.2 Courcelle's theorem

Treewidth definition The treewidth (w) of a tree decomposition is the size of the biggest bag minus 1. $w = (|S_i| - 1) | \{(\forall S_j \in T).(|S_j| \leq |S_i|)\}$.

Treewidth determination Determining if a given graph has treewidth at most k (for k is some variable) is NP-complete [16]. After [16]'s paper, Bodlaender found that if $k \in \mathbb{N}$ is instead a fixed constant, then a graph with treewidth k can be recognized in linear time. Bodlaender also proved that a tree decomposition can be done in linear time [15].

Courcelle's theorem Bodlaender's proof complemented Courcelle's theorem that proved the satisfiability of an ω-regular property (ϕ) given D (a tree decomposition with a constant treewidth k) can be done in O(n) [18].

Courcelle-Bodlaender's theorem $\phi \models G$ can be decided in linear time for a graph G that yields a constant treewidth. Proof: if $G \mapsto D$ can be accomplished in linear time [15] and $\phi \models D$ can be decided in linear time [18], then $\phi \models G$ can be decided in linear time.

4 ω-regular automata

4.1 ω-regular words

Büchi defined ω-regular words as words with a null or finite regular prefix and the suffix is some regular expression repeated ad infinitum.

Examples

- An ω-regular word with a null prefix: $(a \wedge b)^{\omega}$.
- An ω-regular word with a finite prefix: $a^{*}(a \wedge b)^{\omega}$.

In [13], Büchi proved that ω-regular expressions precisely coincided with S1S. Furthermore Büchi showed that languages over ω-words can be decided using a non-deterministic finite automata where the input is accepted if and only if the path visits some final state in the set of all final states infinitely often [13]. This extension to finite automata is known as Büchi automata.

Since Büchi's invention of Büchi automata, several other variants of ω-regular automata have been generalized and separate winning conditions have been created to produce ω-regular automata such as Muller automata [12], Rabin automata [11], and Streett automata [10].

4.2 Generalization of ω-automata

A finite ω-regular automaton A is a tuple $(Q, \Sigma, q_0, \delta, Acc)$ where

- Q is the set of states.
- Σ is the alphabet.
- $q_0 \subseteq Q$ is the set of initial states.
- $\delta \subseteq Q \times \Sigma \times Q$.
- Acc is the acceptance condition.

Conditions Let ρ be a run on A. The following are the most common acceptance conditions for ω-regular automata.

- Büchi condition: some final state in the set $F \subseteq Q$ occurs infinitely often in ρ [13].
- Muller condition: some set of final states in the family $\mathcal{F} \subseteq 2^Q$ occurs infinitely often in ρ [12].
- Rabin condition: Let Ω be a sequence of accepting pairs of states in Q. $\Omega = (E_1, F_1), \ldots, (E_n, F_n)$. Then for some pair (E_i, F_i) in Ω, E_i is visited *finitely* often in ρ while F_i is visited *infinitely* often [11]. A Rabin condition is also known as a "pairs condition".
- Streett condition: Let Ω be a sequence of accepting states as defined in the Rabin condition. For every pair (E_i, F_i) in Ω, if F_i is visited infinitely often, then E_i is also visited infinitely often [10]. A Street condition is also called a "complemented pairs condition."

Substituting the acceptance condition ACC, each of the ω-automata mentioned can be defined. A Büchi automaton NBA $= (Q, \Sigma, q_0, \delta, F)$. A Muller automaton MA $= (Q, \Sigma, q_0, \delta, \mathcal{F})$. A Rabin automaton RA $= (Q, \Sigma, q_0, \delta, \Omega)$. Finally, a Streett automaton SA $= (Q, \Sigma, q_0, \delta, \Omega)$. Note that the listed tuples do not entirely define the automata's behaviors; for example determinism has not yet been specified.

4.3 Expressivity

The first proof for the equivalence of Mueller automata and non-deterministic Büchi automata was McNaughton's theorem [8]. Since then, many proofs have emerged that can be summarized by the following theorem:

Equivalence Theorem Let determinism be abbreviated with "D" and non-determinism with "N". Let A be the set {NBA, DMA, NMA, DRA, NRA, DSA, NSA} and P be a transformation function. Then

$\forall a \forall b[[a \in A] \wedge [b \in A] \wedge [P(a) = b]]$.

In other words, for all the automata in A, there exists an algorithm P to transform that automata into another automata in A. As a corollary, all automata in A are equally expressive. The astute reader will have noticed

set A omitted DBA. While DBA may be morphed into any element of A, the converse is false since DBA is strictly less expressive than any element in A [8].

$DBA \subset NBA = DMA = NMA = DRA = NRA = DSA = NSA.$

Due to this equivalence theorem, any ω-automata in A can be used to express ω-regular expressions. This convenience is important since many proofs are not as easy if another ω-automata was used. Moreover, extensions of ω-regular automata do not imply the winning conditions always equivalent (ω-trees [8] and quantum ω-automata [2] are two examples).

5 Finite tree automata (FTA)

5.1 Expressivity

An FTA may be top-down or bottom-up and deterministic or non-deterministic. Non-deterministic trees may use ϵ to represent a null input to change the state and not the feed. A deterministic, bottom-up FTA is strictly less expressive than the other FTA which can express regular languages.

$DBFTA \subset NBFTA = NTFTA = DBFTA.$

5.2 Non-deterministic, bottom-up FTA (NBFTA)

An NBFTA over Σ is a quadruple (Q, Σ, Q_f, δ) where

- Q is a set of states.
- Σ is a ranked alphabet. A ranked alphabet has an arity $n \in \mathbb{N}$ associated with each symbol. In FTAs, arity corresponds the the amount of child nodes.
- $Q_f \subseteq Q$ is the set of final or accept states.
- $\forall \sigma \in \Sigma | (rank(\sigma) = n) : \delta_\sigma \subseteq Q^n \times Q$. Where $rank(\sigma)$ returns the arity of σ.

In an NBFTA, the leaves of the tree are the initial states which are often written as $Q_i \subseteq Q$. Since Q_i can be deduced by finding all states with an out-degree of zero (leaf nodes), they are sometimes omitted in the formal

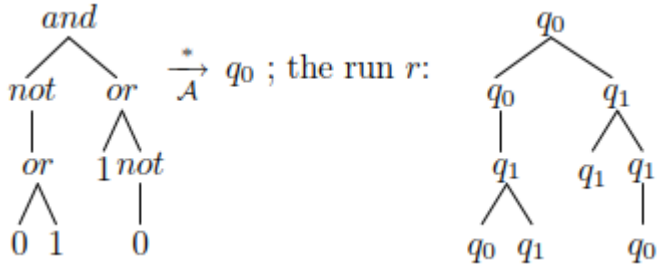
description. A word is accepted in some NBFTA A if and only if $q(t) \in Q_f$ for

$t \xrightarrow[A]{*} q(t)$.

Example Let $\Sigma = \{and, or, not, 0, 1\}$ where nodes {and, or} have a rank of 2 (rank of n is equivalent to saying arity of n or n-ary) and {not} is unary (1-ary). Consider the automaton $A = (Q, \Sigma, Q_f, \delta)$ where Σ is defined as above and the other elements are defined by $Q = \{q_0, q_1\}, Q_f = \{q_1\}$, and

$$\delta = \left\{ \begin{array}{cc} 0 \to q_0, & 1 \to q_1 \\ not(q_0) \to q_1, & not(q_1) \to q_0 \\ and(q_0, q_0) \to q_0, & and(q_0, q_1) \to q_0 \\ and(q_1, q_0) \to q_0, & and(q_1, q_1) \to q_1 \\ or(q_0, q_0) \to q_0, & or(q_0, q_1) \to q_1 \\ or(q_1, q_0) \to q_1, & or(q_1, q_1) \to q_1 \end{array} \right\}$$

To parse this tree, start from the leaf nodes and rewrite Σ with the respective state $q \in Q$ using the transition function δ [7].



6 Game theory and ω-automata

6.1 A brief history of game theory

Definitions

- A zero-sum game is a game where the sum of winnings equal the sum of losses when all players are considered.
- Perfect information is when all parties have complete and accurate information about the game and player strategies.

- A cooperative game is where players can help each other.

Before Neumann published [29, 28] on perfect information, zero-sum, two-player games, game theory had no decent formal platform. Later, Nash proved that for any non-cooperative game with a finite set of actions there existed at least one optimal solution for all players. Now called the Nash equilibrium, the optimal solution is defined as a game strategy where failure to comply results in an opponent gaining an advantage (assuming the opponent is perfectly rational) [27].

6.2 ω-games

Definitions Define an n-player game arena as $\Gamma = (Q, P, Q_{init}, \Sigma, \delta)$ where Q is a set of finite states, Q_{init} is an element of Q and the initial state, Σ is a finite set of actions, $\delta : Q \times \Sigma \times Q$ is the transition function, and $P = \{Q_1, Q_2, \ldots, Q_n\}$ partitions Q into n sets, one for each player. In game theory, states are also called game positions [3].

An ω-game is a pair (Γ, W) where $\Gamma = (Q, P, Q_{init}, \Sigma, \delta)$ and W is the set of winning conditions. Each player's winning condition is a subset in Q^ω. Given a run ρ, examples of winning conditions include:

- Reachability condition: $Z \subseteq Q$ must be visited in ρ.
- Safety condition: ρ must never leave $Z \subseteq Q$.
- Büchi condition: $Z \subseteq Q$ must be visited infinitely often.
- co-Büchi condition: any $q \in Q$ may be visited finitely often, however only a select $Z \subseteq Q$ may be visited infinitely often.
- Similarly, the Muller, Rabin, and Streett conditions can be applied as winning conditions.

Strategy A strategy strat(p, p') is a function that return a position which is reachable by p' and is found by an immediate outgoing transition from p. If strat(p, p') meets a winning condition and is inevitable, it is a winning strategy. Büchi-Landweber proved that for every game with an ω-winning condition (W), there exists a Muller automaton (with finite states by definition) that produces a winning strategy for some player [8].

7 Game theory and ω-trees

7.1 MSO on infinite trees

Rabin's theorem [11] extended Büchi's work [13] and proved the second-order monadic theory of the rooted infinite binary tree is decidable. Also called the monadic theory of two successor functions (S2S), it allows one to quantify over any sets of vertices. Consequently, Rabin extended MSO-formulas from ω-regular words to infinite trees.

Rabin's theorem was proven by creating ω-tree automata (ω-TA) with acceptance conditions taken from the many used for ω-regular automata and inputting infinite binary trees to the ω-TA. Recall that FTAs have a ranked alphabet which dictates how many children each node has. Rabin's proof used infinite trees where every node's rank equals two.

Expressivity ω-regular expressions can be decided with an ω-TA; however, not all languages accepted by an ω-TA can be recognized by an ω-regular automata [11].

7.2 ω-tree automata

Muller tree automaton (MTA) An MTA is a tuple $(Q, \Sigma, Q_0, \delta, \mathcal{F})$ where $(Q, \Sigma, Q_0, \mathcal{F})$ are the same as the original Muller automaton described previously. However, now $\delta \subseteq Q \times A \times Q \times Q$. Let ϵ be the null action and t be an infinite binary tree where each node is labeled with some letter in Σ. Then a run (ρ) over t is an infinite binary tree where each node is labeled with a state in Q satisfying $(\rho(w), t(w), \rho(w_0), \rho(w_1)) \in \delta$ and $\rho(\epsilon) = Q_0$. The run ρ is successful if and only if for each path $\pi \in \{0,1\}^\omega$ the Muller acceptance condition ($\mathcal{F}$) is satisfied.

Other tree automata The techniques demonstrated to create an MTA are done correspondingly to acquire Büchi, Rabin, and Streett tree automata. Muller, Rabin, and Streett tree automata all share Tree-ω over Σ (T^ω_Σ) expressiveness. Büchi tree automata (BTA) are a strict subset of the other tree automata discussed, and BTA cannot express thing such as $\{t \in T^\omega_{\{a,b\}} |$ each path through t carries finitely many b$\}$ [8].

Parity tree automaton (PTA) A PTA (also called a Rabin chain tree automaton) is a tuple: $PTA = (Q, \Sigma, Q_0, \delta, \Omega)$. (Q, Σ, Q_0, δ) are the same as described in a Muller tree automaton, and

$$\Omega : E_1 \supset F_1 \supset E_2 \supset F_2 \supset \cdots \supset E_n \supset F_n$$

Ω condition Recall the Rabin finite automata's acceptance condition: given a set of pairs $(E_1, F_1), \ldots, (E_n, F_n)$ and a run ρ, there exists some pair (E_i, F_i) where E_i is visited finitely often and F_i is visited infinitely often. Similarly, PTAs accept a tree if every path contains some pair of sets (E_i, F_i) such that every $e \in E_i$ is visited finitely often and every $f \in F_i$ e is visited infinitely often.

Parity condition Let π be a path in the run ρ, A be the set of accepted paths, (E_i, F_i) are as described in the Ω condition, and $\exists^\omega$ be read as "there exists infinitely often". Then the parity condition can be summarized as a labeling algorithm followed by an acceptance algorithm.

Labeling algorithm:

$(\forall s_k \in \{E_i \backslash F_{i-1}\} \text{ do } k := 2i - 1) \wedge (\forall s_k \in \{F_i \backslash E_i\} \text{ do } k := 2i)$

Acceptance algorithm:

$\exists \pi \in \rho | [min(k \in \{\bigcup_{k \geq 0}(s_k | s_k \in \{\pi | \exists^\omega s_k\})\}) \mod 2 \equiv 0] \longleftrightarrow \rho$ is accepted.

In English, the parity condition labels each state in $E_i \backslash F_{i-1}$ by $2i-1$ and each state in $F_i \backslash E_i$ by $2i$. Then a set of states on the path π is accepting if and only if the minimum index of states in π is even. The parity condition is equivalent to the Ω condition [8].

PTA = MTA Any PTA can be transformed into a Muller tree automata and vice versa. Therefore, PTA accepts the same tree languages as the Muller tree automata [6]. While Rabin was the first to introduce tree automata and their closure under complementation using induction on countable ordinals [11], Gurevich and Harrington found an alternative proof using game theory and PTAs which some people consider as far more comprehensible than Rabin's proof [5].

7.3 ω-TA closure under complementation

7.3.1 The game

Definitions Let t be an infinite binary tree and the input to some PTA α. t and α will be considered for the infinite, zero-sum, perfect information, two-player game Γ. Player one and player two will be called Automaton and Pathfinder, respectively. For Automaton to win, he must realize the acceptance condition with the constructed state sequence; if Automaton cannot win then Pathfinder wins.

Game play The game positions V are colored into subsets $\{V_0, V_1\}$ and controlled by Automaton and Pathfinder, respectively. Each game position is of the form (tree node of t, tree label of the tree node, $q \in Q$). Edges of the game graph are defined by $E \subseteq (V_0 \times V_1) \cup (V_1 \times V_0)$—this just indicates any transition from V_1 goes to V_0 and vice versa. Additionally, every vertice's outgoing degree is greater than 0 (to prevent deadlock) and finite. Player actions are limited to the transitions on the game graph available given the current game position.

Starting from the game position $(\epsilon, t(\epsilon), q_0)$, Automaton makes the first move by selecting a transition τ which is the quadruple

$(q, a, q', q'') \in \delta | [q' = left(q) \wedge q'' = right(q)]$.

After Automaton's move, Pathfinder dictates which direction (left or right) to take that is consistent with Automaton's transition selection. On the second round, Automaton again chooses a transition, this time he is restricted to choosing $\tau \in \delta$ such that the transition is compatible with his previous choices. Then Pathfinder selects another direction from the current node. This pattern repeats until a winner is found. If C is defined as the finite set of partitions that was previously used to partition V into $\{V_0, V_1\}$, then the winning condition $W \subseteq C^\omega$.

Strategies and acceptance For both Automaton and Pathfinder, a strategy from position p is a function that guides the player to position p' by giving a position that is reachable by p' and is a child of p. A winning strategy from p is a strategy is a guide that guarantees the player will win if state p is reached, no matter what actions the opponent takes. If the input tree t has a successful run on a PTA then there is a winning strategy (W_A) for Automaton in Γ. Therefore, PTA's acceptance condition can be translated

to

$\exists W_A | (\epsilon, t(\epsilon), q_0) \longleftrightarrow$ the PTA accepts t.

To prove ω-TA is closed under complementation, it must be shown that if a given PTA α does not accept input t, there exists a PTA β that accepts t. The game theoretic equivalent is: if Automaton does not have a winning strategy in Γ_α then Automaton has a winning strategy in a separate game Γ_β.

The new acceptance condition can prove closure under complementation in two steps. First, it is shown there exists a winning strategy for one player given any initial position. Second, Pathfinder's strategy can be translated to an Automaton strategy.

Winning strategies Recall that Büchi and Landweber proved that for every game with an ω-winning condition (W), there exists a Muller automaton that produces a winning strategy for some player. Memoryless strategies are strategies that do not require any history of the game to be remembered—only the current game position and options. For any given PTA α and input t, either Pathfinder or Automaton has a memoryless winning strategy for Γ_α [6].

Review Many graphs, automata, and trees are currently being considered. Recall that t is the infinite input tree that we are feeding into some PTA α. The game Γ is a graph of game positions partitioned into $\{V_0, V_1\}$ to be controlled by Automaton and Pathfinder, respectively. Strategies are ω-regular automata and each player has one. With memoryless strategies, the automata are simple—every state has precisely one out-edge.

7.3.2 Proof of closure under complementation

This section will prove that for any given PTA α over Σ, a Muller tree automata β that only accepts $T^\omega_\Sigma \backslash T(\alpha)$ can be constructed. We start by introducing three logically equivalent statements: a word is rejected from a PTA, Pathfinder has a memoryless winning strategy (P_w), and Automaton does not have a winning strategy (A_w). P_w can be reconstructed to "β accepts t" for some PTA β. Since all of these statements are logically equivalent, only one must be proven.

Pathfinder's actions are ($\{0, 1\}$, Σ, δ). The set $\{0, 1\}$ corresponds to the left and right children of the current node of an ω-TA. Recall that δ is

constructed by Automaton creating transitions $\tau \in \delta$ once per turn. Another way to describe Pathfinder's actions is with a family $(f_w : \Sigma \times \delta \to \{0,1\})$ where $w \in \{0,1\}^*$. Let the finite set of actions composing f_w be called A. Since A is finite, a tree (s) labeled with A can encode P_w such that s(w) = f_w. Now a corresponding tree (s_t) can be labeled with (A, Σ) for all s_t(w) = (s(w),t(w)) for $w \in \{0,1\}^*$.

With s_t, we can write another equivalency for P_w: there exists an A-labeled (action-labeled) tree (s) such that for any sequence τ_{seq} of transitions τ Automaton makes, and for the unique path $\pi \in \{0,1\}^\omega | \tau_{seq}$ the Ω accepting condition is not met.

Equivalently:
(1) $\exists s | (s\_t \models ($
(2) $\forall \pi \in \{0,1\}^\omega : ($
(3) $\forall \tau_{seq} \in \delta^\omega : ($
(4) $((s|\pi) \mapsto (\tau_{seq} \wedge t|\pi)) = \pi) \to (\{q_i, q_j, \dots\} | \tau_{seq} \not\models \Omega)))))$

Part (4) can be read as "if mapping the sequence of actions in s given by π to the transition sequence τ_{seq} and the sequence of tree labels in t given by π reconstructs π, then the state sequence produced by $(q_i, q_j, \dots)$ does not satisfy Ω." Part (4) is a property of ω-words over $A \times \Sigma \times \delta \times \{0,1\}$ and can therefore be checked by a sequential Muller automaton M_4 which is independent of t.

From part (4), part (3)'s dimensions can be deduced. Due to the universal quantification of δ, part (3) is a property of ω-words over $A \times \Sigma \times \{0,1\}$. Since closure under complementation of Muller automata has been proven, the identity $\forall\phi() \equiv \neg\exists\neg\phi()$ can be used, and a separate deterministic Muller automata M_3 can verify part (3).

Similarly, part (2) describes a property with $A \times \Sigma$ dimensions—a property of (A, Σ)-labeled trees. A deterministic Muller tree automaton M_2 can verify part (2) by simulating M_3 along each path. Since M_3 is deterministic, one can combine all M_3 runs into a single run of M_2 [8].

Finally, the Muller tree automaton β can be constructed to validate part 1. β will guess a tree s on the input tree t and descend s_t like M_2 did. Recall that M_4 is independent of the input tree t considered. As a result, β accepts a tree t if and only if $t \in T_\Sigma^\omega \backslash T(\alpha)$ [8, 5, 1].

8 Conclusion

Formal systems and logic vary in degree of expressivity and behavior. Propositional and monadic second-order logic (MSO) were found to be decidable using ω-regular automata or ω-tree automata. While the proof for propositional logic's decidability was trivial, MSO's decidability question took years of proofs, most notably [13, 11].

ω-regular words are precisely S1S and can be decided with ω-regular automata such as Büchi automata [13]. Additionally, there exists an ω-tree automata that can accept any word in {S1S, S2S,. . . ,SnS} which is the language of MSO [11]. Applications of MSO being decidable extend to game theory and graph theory [5, 18].

9 Future work

Behavioral game theory The rational approach cited by Neumann is not always the approach found in society. Consider a two-player, one-shot game called the ultimatum game. One player, the proposer, is given some money, say $10. The proposer is then tasked to offer a portion of the money to the other player (the responder). If the responder rejects the proposer's offer, neither player gets any money. This is a one-shot game so there is no negotiating. Rationally, the responder should accept any amount of money– even a penny is better than nothing. Experimentally, researchers find this is not the case. [31] found responders often rejected offers that were below 25-30% of the money offered to the proposer. Additionally, some proposers would offer more than what they thought would be the responder's minimum acceptance to be "fair".

Robert Axelrod ran many simulations and discovered that the winner was often the one who cooperates on the first step then reflects whatever the opponent does on subsequent steps[30]. Similar results were obtained by natural selection which helps explain the cooperation phenomena we see among mammals even if doing so may be irrational (as seen in the ultimatum game above). Traditional game theory assumes "perfectly rational" players— as shown above, the rational approach is not always the best choice.

It would be interesting to employ a machine learning technique (such as a genetic algorithm) over possible ω-winning conditions with imperfect information. Further work could also be done on infinite stochastic games

that do not have winning conditions but Nash equilibriums expressible as ω-regular conditions.

10 Appendix

10.1 Some formula-related symbols

π: a path.
(V, E): (Vertices, Edges).
$\mapsto$: maps to.
$\exists^{\omega}$ There exists infinitely many.

10.2 Common abbreviations

PTA: parity tree automata (synonym for a Rabin chain tree automata)
MTA: Muller tree automata
ω-TA: ω-tree automata
MSO: monadic second-order logic
SnS: monadic second-order theory of n successor function(s)

References

[1] M. Vardi, T. Wilke, "Automata: From Logics to Algorithms".

[2] A. S. Bhatia, A. Kumar. "Quantum ω-Automata over Infinite Words and Their Relationships". *International Journal of Theoretical Physics*, vol. 58(3), DOI: 10.1007/s10773-018-3983-0.

[3] K. Chatterjee, L. Doyen, E. Filiot, and J.-F. Raskin, "Doomsday equilibria for omega-regular games," *Information and Computation*, vol. 254, pp. 296–315, 2017.

[4] A. W. Mostowski, "Games with forbidden positions", Uniwersystet Gdański, Instytyt Matematyki 1991.

[5] Y. Gurevich and L. Harrington, "Trees, automata, and games," in 1982, . DOI: 10.1145/800070.802177.

[6] A. W. Mostowski, "Hierarchies of weak automata and weak monadic formulas," *Theoretical Computer Science*, vol. 83, (2), pp. 323-335, 1991.

[7] H. Comon et al. "Tree Automata Techniques and Applications", 2008.

[8] W. Thomas, "Languages, Automata, and Logic," *Handbook of Formal Languages*, pp. 389–455, 1997.

[9] M. Y. Vardi, "An automata-theoretic approach to linear temporal logic," *Logics for Concurrency Lecture Notes in Computer Science*, pp. 238–266, 1996.

[10] R. S. Streett, "Propositional Dynamic Logic of looping and converse," emphProceedings of the thirteenth annual ACM symposium on Theory of computing - STOC 81, 1981.

[11] M. O. Rabin, "Decidability of second-order theories and automata on infinite trees.," *Transactions of the American Mathematical Society*, vol. 141, pp. 1–1, 1969.

[12] D. E. Muller, "Infinite sequences and finite machines," *Proceedings of the Fourth Annual Symposium on Switching Circuit Theory and Logical Design (swct 1963)*, 1963.

[13] J. R. Büchi, "On a Decision Method in Restricted Second Order Arithmetic," *The Collected Works of J. Richard Büchi*, pp. 425–435, 1990.

[14] J. R. Büchi, "Weak Second-Order Arithmetic and Finite Automata," *The Collected Works of J. Richard Büchi*, pp. 398–424, 1990.

[15] H. L. Bodlaender, "A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth," SIAM Journal on Computing, vol. 25, no. 6, pp. 1305–1317, 1996.

[16] S. Arnborg, D. G. Corneil, and A. Proskurowski, "Complexity of Finding Embeddings in a k-Tree," *SIAM Journal on Algebraic Discrete Methods*, vol. 8, no. 2, pp. 277–284, 1987.

[17] R. Diestel, *Graph theory*. Berlin: Springer, 2006.

[18] B. Courcelle, "The monadic second-order logic of graphs. I. Recognizable sets of finite graphs," *Information and Computation*, vol. 85, (1), pp. 12-75, 1990.

[19] A. Holroyd, A. Levy, M. Podder, and J. Spencer, "Second order logic on random rooted trees" 2017.

[20] N. Immerman, "Languages which capture complexity classes," in 1983, . DOI: 10.1145/800061.808765.

[21] A. Turing, "On computable numbers, with an application to the Entscheidungsproblem", *Proc. London Maths. Soc.*, vol. s2-42, pp. 230-265, 1936.

[22] M. Smullyan, *Gödel's Incompleteness Theorems*, 1992.

[23] J. Väänänen, "Second-Order Logic and Foundations of Mathematics," *Bulletin of Symbolic Logic*, vol. 7, no. 4, pp. 504–520, 2001.

[24] S. Banach and A. Tarski, "Sur la décomposition des ensembles de points en parties respectivement congruentes," *Fundamenta Mathematicae*, vol. 6, pp. 244–277, 1924.

[25] K. Weierstrass, "Über continuirliche Functionen eines reellen Arguments, die für keinen Werth des letzteren einen bestimmten Differentialquotienten besitzen," *Mathematische Werke*, pp. 71–74, 1872.

[26] G. Boole, "An investigation of the laws of thought on which are founded the mathematical theories of logic and probabilities / by George Boole.," 1854.

[27] J. F. Nash, "Non-Cooperative Games", *PhD thesis*, 1950.

[28] J. Von Neumann, "On the Theory of Games of Strategy", 1928.

[29] J. Von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*. ([Third]. ed.) New York: Science Editions, 1964.

[30] S. Wolfram, *A new kind of science*. USA: Wolfram Media, 2002.

[31] J. Henrich, "Does Culture Matter in Economic Behavior? Ultimatum Game Bargaining among the Machiguenga of the Peruvian Amazon," *The American Economic Review*, vol. 90, (4), pp. 973-979, 2000.

[32] R. Fagin, L. J. Stockmeyer, and M. Y. Vardi, "On Monadic NP vs Monadic co-NP," *Information and Computation*, vol. 120, (1), pp. 78-92, 1995.

[33] R. Fagin, "Generalized first-order spectra and polynomial-time recognizable sets". *Complexity of Computation*, vol. 7, pp. 43-73, 1974.