Rabin's Tree Theorem and Applications

Shane Steinert-Threlkeld

November 20, 2013

STANFORD UNIVERSITY

Plan

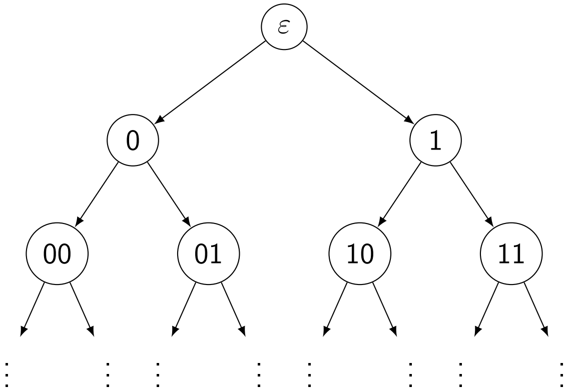
The Main Idea

In this talk, we will do two main things:

1. Prove *Rabin's Tree Theorem*
2. Show how to use this theorem to prove the decidability of other logics.

To do (1), we will introduce infinite automata both on strings and on trees.

Rabin's Tree Theorem



Theorem 1.1 (Rabin [1969])

The monadic second-order theory of the infinite binary tree is decidable.

The Infinite Binary Tree

Theorem 1.2 (Rabin [1969])

The monadic second-order theory of the infinite binary tree is decidable.

The infinite binary tree is the structure

$$T^2 = (\{0,1\}^*, s_0, s_1)$$

of all finite sequences of 0s and 1s where

$$s_0(w) = w0$$
$$s_1(w) = w1$$

are the two successor functions. We use ε to denote the empty sequence.

Monadic Second-Order Logic

Theorem 1.3 (Rabin [1969])

The monadic second-order theory of the infinite binary tree is decidable.

Monadic second-order logic extends first-order logic with variables for and quantification over monadic predicates. That is, we add atomic formulas of the form

$$Xx$$

and quantified formulas of the form

$$\exists X \varphi$$

where X will be interpreted as a *subset* of the domain of discourse.

Monadic Second-Order Theories

Theorem 1.4 (Rabin [1969])

The monadic second-order theory of the infinite binary tree is decidable.

The monadic second-order theory of a structure $\mathfrak{A}$ is the set of all monadic second-order sentences (in the appropriate signature) φ such that $\mathfrak{A} \models \varphi$.
So, the monadic second-order theory of the infinite binary tree is the set of all monadic second-order sentences φ such that $T_2 \models \varphi$.
We call this theory **S2S**.

Decidability

Theorem 1.5 (Rabin [1969])

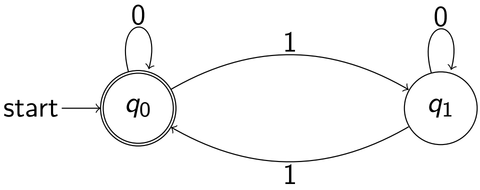
The monadic second-order theory of the infinite binary tree is decidable.

The subject of this whole course. Intuitively, there is an algorithm that, when given a sentence φ, answers "yes" or "no" depending on whether $\varphi \in$ **S2S** or not.
Slightly more formally, let

$$S_2 = \{n \in \mathbb{N} \mid n = \#\varphi \text{ and } T_2 \models \varphi\}$$

Then we have that χ_{S_2} is a recursive function.

Finite Automaton Example

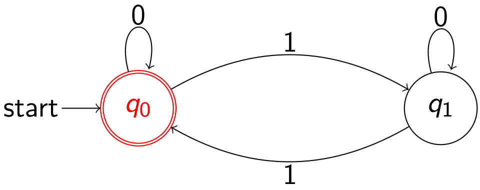


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \restriction 5, s)$: 0 → 1 → 2 → 3 → 4

run: q_0

Finite Automaton Example

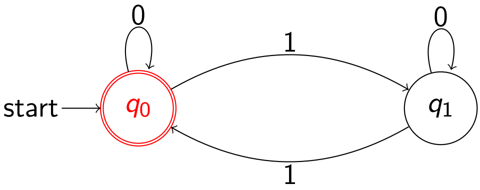


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \restriction 5, s)$:

run: q_0

Finite Automaton Example

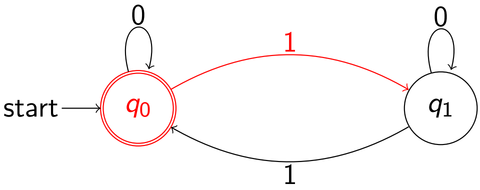


word: $\quad$ 1 $\quad$ 1 $\quad$ 0 $\quad$ 1 $\quad$ 0

$\mathfrak{A} = (\omega \restriction 5, s)$:

run: q_0

Finite Automaton Example

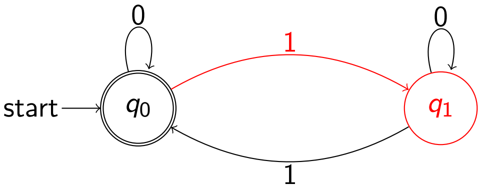


word: $\quad$ 1 $\quad$ 1 $\quad$ 0 $\quad$ 1 $\quad$ 0

$\mathfrak{A} = (\omega \upharpoonright 5, s)$: $\quad 0 \to 1 \to 2 \to 3 \to 4$

run: $\quad q_0$

Finite Automaton Example

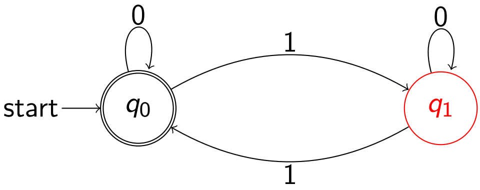


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \upharpoonright 5, s)$: $0 \to 1 \to 2 \to 3 \to 4$

run: q_0

Finite Automaton Example

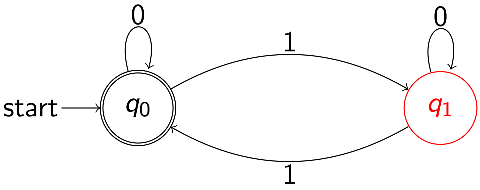


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \restriction 5, s)$: $0 \to 1 \to 2 \to 3 \to 4$

run: q_0 q_1

Finite Automaton Example

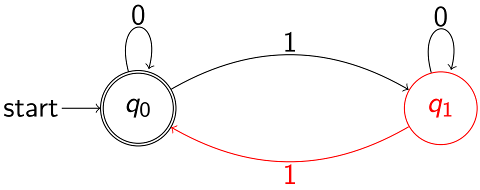


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \restriction 5, s)$: $0 \to 1 \to 2 \to 3 \to 4$

run: q_0 q_1

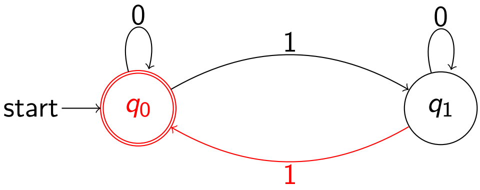
Finite Automaton Example



word:	1	1	0	1	0
$\mathfrak{A} = (\omega \restriction 5, s)$:	0	1	2	3	4

run: q_0 q_1

Finite Automaton Example

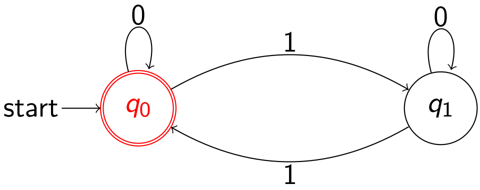


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \restriction 5, s)$: $0 \to 1 \to 2 \to 3 \to 4$

run: q_0 q_1

Finite Automaton Example

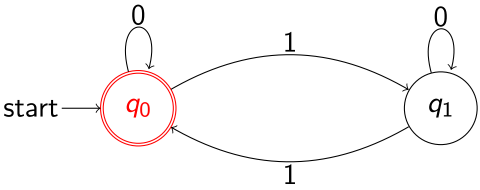


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \upharpoonright 5, s)$: $0 \to 1 \to 2 \to 3 \to 4$

run: q_0 q_1 q_0

Finite Automaton Example

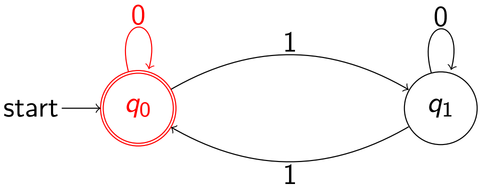


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \restriction 5, s)$: 0 1 2 3 4

run: q_0 q_1 q_0

Finite Automaton Example

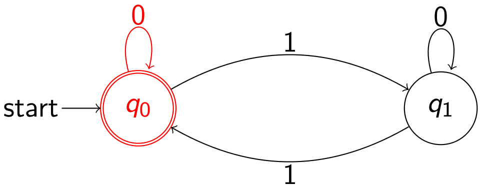


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \upharpoonright 5, s)$: $0 \to 1 \to 2 \to 3 \to 4$

run: q_0 q_1 q_0

Finite Automaton Example

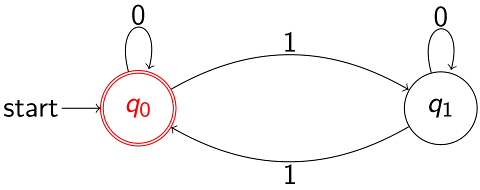


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \restriction 5, s)$: 0 → 1 → 2 → 3 → 4

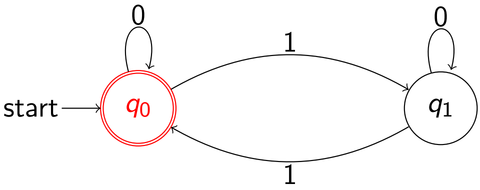
run: q_0 q_1 q_0

Finite Automaton Example



word:		1	1	0	1	0
$\mathfrak{A} = (\omega \restriction 5, s)$:		0	1	2	3	4
run:	q_0	q_1	q_0	q_0		

Finite Automaton Example

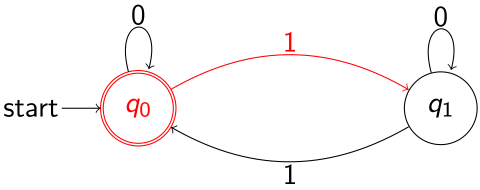


word: 1 1 0 1 0

$\mathfrak{A} = (\omega \restriction 5, s)$: $0 \to 1 \to 2 \to 3 \to 4$

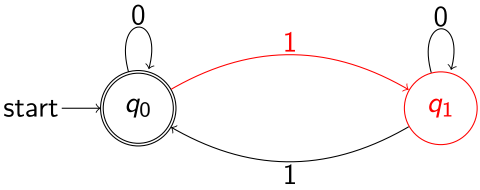
run: q_0 q_1 q_0 q_0

Finite Automaton Example



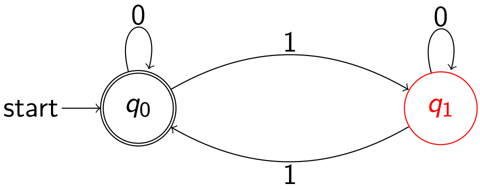
word:		1	1	0	1	0
$\mathfrak{A} = (\omega \restriction 5, s)$:		0	1	2	3	4
run:	q_0	q_1	q_0	q_0		

Finite Automaton Example



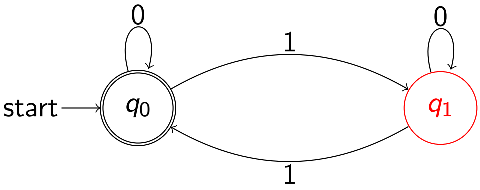
word:		1	1	0	1	0
$\mathfrak{A} = (\omega \restriction 5, s)$:		0	1	2	3	4
run:	q_0	q_1	q_0	q_0		

Finite Automaton Example



word:		1	1	0	1	0
$\mathfrak{A} = (\omega \restriction 5, s)$:		$0 \to$	$1 \to$	$2 \to$	$3 \to$	4
run:	q_0	q_1	q_0	q_0	q_1	

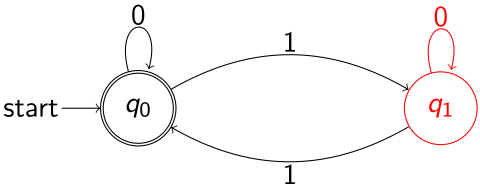
Finite Automaton Example



word:		1	1	0	1	0
$\mathfrak{A} = (\omega \upharpoonright 5, s)$:		0	1	2	3	4
run:	q_0	q_1	q_0	q_0	q_1	

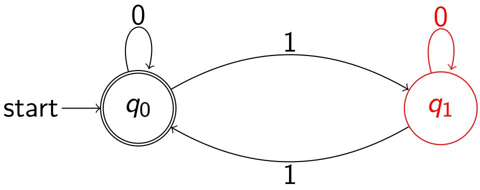
$$0 \to 1 \to 2 \to 3 \to 4$$

Finite Automaton Example



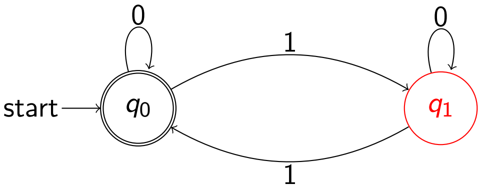
word:		1	1	0	1	0
$\mathfrak{A} = (\omega \restriction 5, s)$:		0	1	2	3	4
run:	q_0	q_1	q_0	q_0	q_1	

Finite Automaton Example



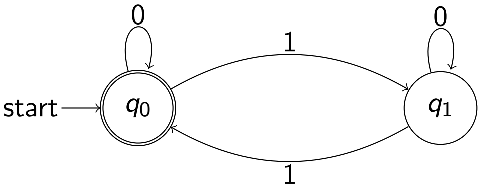
word:		1	1	0	1	0
$\mathfrak{A} = (\omega \upharpoonright 5, s)$:		$0 \to$	$1 \to$	$2 \to$	$3 \to$	4
run:	q_0	q_1	q_0	q_0	q_1	

Finite Automaton Example



word:		1	1	0	1	0
$\mathfrak{A} = (\omega \upharpoonright 5, s)$:		0	1	2	3	4
run:	q_0	q_1	q_0	q_0	q_1	q_1

Finite Automaton Example



word:		1	1	0	1	0	
$\mathfrak{A} = (\omega \restriction 5, s)$:		$0 \to$	$1 \to$	$2 \to$	$3 \to$	4	
run:	q_0	q_1	q_0	q_0	q_1	q_1	$\notin F$

Regular Languages

Recall that finite automata (whether deterministic or non-deterministic) recognize the *regular languages*. Given an alphabet Σ, the regular languages in Σ are the smallest collection of elements of $\mathcal{P}(\Sigma^*)$ s.t.

- $\emptyset$ is regular
- $\{a\}$ is regular for each $a \in \Sigma$
- $A \cup B$, $A \cdot B$ and A^* are regular if A, B are regular

One can show that the regular languages are also closed under intersection and complement (from which closure under relative complement follows). Note that $\{\varepsilon\} = \emptyset^*$ is regular.

Finite Automaton Lessons

Although I will assume familiarity with the basics of finite automata theory, I wanted to do the example that way to emphasize a few points which will make generalizing to infinite objects and trees easier to understand:

Finite Automaton Lessons

Although I will assume familiarity with the basics of finite automata theory, I wanted to do the example that way to emphasize a few points which will make generalizing to infinite objects and trees easier to understand:

- *Words* are just labelings of a particular structure

Finite Automaton Lessons

Although I will assume familiarity with the basics of finite automata theory, I wanted to do the example that way to emphasize a few points which will make generalizing to infinite objects and trees easier to understand:

- *Words* are just labelings of a particular structure
- *Runs* of an automaton are labelings of that same structure with states, subject to

$$r_{i+1} \in \delta\left(w_i, r_i\right)$$

Finite Automaton Lessons

Although I will assume familiarity with the basics of finite automata theory, I wanted to do the example that way to emphasize a few points which will make generalizing to infinite objects and trees easier to understand:

- *Words* are just labelings of a particular structure
- *Runs* of an automaton are labelings of that same structure with states, subject to

$$r_{i+1} \in \delta\left(w_i, r_i\right)$$

- A run is *accepted* iff a certain property holds of it; in the finite automaton case:

$$r_{len(r)} \in F$$

Plan

Büchi's Theorem

$\mathfrak{A} = (\omega, s)$: $\quad 0 \to 1 \to 2 \to 3 \to 4 \to \cdots$

Let $T_1 = (\omega, s)$ be the structure of the natural numbers and the successor function. Denote the monadic second-order theory of this structure by **S1S**. As a warm-up to Rabin's theorem, we will first prove:

Theorem 2.1 (Büchi [1962])

S1S *is decidable.*

Preliminary Definitions

Any finite set Σ will be called an *alphabet*.
By Σ^ω we denote the set of ω-sequences $w = w_0 w_1 w_2 \ldots$ of elements of Σ, i.e. functions $w : \omega \to \Sigma$.
For $U \subseteq \Sigma^*$, U^ω is the set of ω-words $u = u_0 u_1 u_2 \ldots$ s.t. $u_i \in U$.
An ω-*language* in Σ is a subset of Σ^ω.
Given an element $w \in \Sigma^\omega$, let

$$Inf(w) := \{\sigma \in \Sigma \mid \sigma \text{ occurs infinitely many times in } w\}$$

Büchi Automaton

Definition 2.2

A *non-deterministic Büchi automaton* for alphabet Σ is a tuple $A = (Q, Q_0, \delta, F)$ where

- Q is a finite set of *states*
- $Q_0 \subseteq Q$ is the set of *initial states*
- $F \subseteq Q$ is the set of *final states*
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ is the *transition function*

A *run* of A on an ω-word w is a $q \in Q^\omega$ s.t. $w_0 \in Q_0$ and

$$q_{i+1} \in \delta\,(q_i, w_i)$$

A accepts w iff there is a run q of A on w s.t.

$$Inf\,(q) \cap F \neq \emptyset$$

Some Complexity Results

Given an automaton A,

$$L(A) = \{w \in \Sigma^\omega \mid A \text{ accepts } w\}$$

I record here a few interesting complexity results; we need only their decidability. The proofs run through connections with temporal logics.

Theorem 2.3 (Sistla et al. [1987])

The emptiness problem for Büchi automata – given A, does $L(A) = \emptyset$? – is NLOGSPACE*-complete.*

Theorem 2.4 (Vardi and Wolper [1994])

The universality problem for Büchi automata – given A, does $L(A) = \Sigma^\omega$? – is PSPACE*-complete.*

A Key Result

An ω-language L is called ω-regular iff it is of one of the forms:

- U^ω for a regular language U
- UL for regular language A and ω-regular B
- $L \cup L'$ for L, L' ω-regular

Theorem 2.5

L is ω-regular iff there is a non-deterministic Büchi automaton A s.t. $L = L(A)$.

We will soon prove the $\Rightarrow$ direction as a series of closure lemmas.

Note on Accepting Conditions

An automaton is determinstic if $|Q_0| = 1$ and

$$|\delta(q, \sigma)| = 1$$

for every $q \in Q$ and $\sigma \in \Sigma$.

It's well-known that deterministic finite automata are as powerful as non-deterministic automata. This, however, is *not true* about Büchi automata: there are non-deterministic Büchi automata which accept languages not accepted by any deterministic Büchi automaton.

There are other kinds of infinite automata – Rabin, Streett, Muller – which differ just based on their acceptance conditions. All of these also accept the ω-regular languages and, interestingly, have equally powerful deterministic versions.

Closure Properties

Lemma 2.6

If $U \subseteq \Sigma^*$ is regular, then U^ω is accepted by a n.d. Büchi automaton.

Proof.

Because $U^\omega = (U \setminus \{\varepsilon\})^\omega$ and $U \setminus \{\varepsilon\}$ is regular if U is, we can assume w.l.o.g. that $\varepsilon \notin U$.
Let A be a finite automaton recognizing U with no transitions leading into q_0. (Because $\varepsilon \notin U$, $q_0 \notin F$.) Now, let B be an automaton identical to A, except without its final states, with $F = \{q_0\}$ and all (q_1, a, f) transitions (for $f \in F(A)$) replaced by (q_1, a, q_0) transitions.
(Helpful to draw a picture of this.) $\square$

Closure Properties (cont.)

Lemma 2.7

If $U \subseteq \Sigma^*$ is regular and $L \subseteq \Sigma^\omega$ is ω-regular, then UL is accepted by a n.d. Büchi automaton.

Proof.

Let A be a finite automaton accepting U and B a non-deterministic Büchi automaton accepting L (our inductive hypothesis). Let C be the disjoint union of A and B, with all (q, a, f) transitions in A replaced by transitions (q, a, q_0) for each $q_0 \in Q_0(B)$.
(Again, draw a picture.) $\square$

Closure Properties (cont.)

Lemma 2.8

If L, L' are ω-regular, then $L \cup L'$ is accepted by a n.d. Büchi automaton.

Proof.

Let A, A' be non-deterministic Büchi automata accepting L and L' respsectively. WLOG, assume $Q(A)$ and $Q(A')$ are disjoint. Then, simply take the union of all components to get an automaton accepting $L \cup L'$. $\square$

Closure Properties (cont.)

Lemma 2.9

If L, L' are ω-regular, then $L \cap L'$ is accepted by a n.d. Büchi automaton.

Proof.

Let A, A' be non-deterministic Büchi automata accepting L and L' repsectively. WLOG, assume $Q(A)$ and $Q(A')$ are disjoint. Let $C = (Q \times Q' \times \{0,1,2\}, Q_0 \times Q_0' \times \{0\}, \delta'', F'')$ where

$$F'' = Q \times Q' \times \{2\}$$
$$\delta''(\langle q, q', i\rangle, a) := \delta(q, a) \times \delta'(q', a) \times \{j\} \text{ where}$$
$$\begin{cases} j = 1 & i = 0 \text{ and } q \in F \\ j = 2 & i = 1 \text{ and } q' \in F' \\ j = 0 & i = 2 \\ j = i & \text{otherwise} \end{cases}$$

So: we start with the third component of the state being 0. Once $q \in F$ is reached, flipped to 1. Then, once $q' \in F'$ reached, flipped to 2. Then, immediately back to 0. So, a state with third component 2 (i.e. a state in F'' is reached infinitely often iff both A and A' reach final states infinitely often. □

Closure Under Complement

To prove that the ω-regular languages are closed under complement, we need two theorems:

Theorem 2.10 (Büchi)

$L \subseteq \Sigma^\omega$ is ω-regular iff it can be represented as a finite union of sets UV^ω where $U, V \subseteq \Sigma^*$ are regular.

Let $[X]^k$ denote the set of k-element subsets of a given set X.

Theorem 2.11 (Ramsey)

For every finite set M, $k \in \omega$, and $f : [\omega]^k \to M$, there is an infinite $X \subseteq \omega$ s.t. $f(x) = f(y) \ni M$ for all $x, y \in [X]^k$.

NB: $k = 1$ is the pigeonhole principle. We'll use $k = 2$.

Closure Under Complement (cont.)

Theorem 2.12

If L is ω-regular, then so too is $\overline{L} := \Sigma^\omega \setminus L$.

Strategy: Given an automaton A over Σ, define a congruence relation (an equivalence relation compatible with concatenation) $\sim_A$ over Σ^*. Show that the equivalence classes are regular languages. Then, represent $L(A)$ and $\overline{L(A)}$ as finite unions of sets UV^ω where U and V are $\sim_A$-equivalence classes. Then use the previous theorem of Büchi.

Closure Under Complement (cont.)

Define: $q \stackrel{w,F}{\to} q'$ iff there is a run of A on w from q to q' s.t. at least one state of the run is in F.
Now, for $u, v \in \Sigma^*$, define $u \sim_A v$ iff for all states q, q' of A:

$$q \stackrel{u}{\to} q' \Leftrightarrow q \stackrel{v}{\to} q' \text{ and } q \stackrel{u,F}{\to} q' \Leftrightarrow q \stackrel{v,F}{\to} q'$$

Closure Under Complement (cont.)

Lemma 2.13

1. *$\sim_A$ is a congruence relation with a finite number of equivalence classes ('of finite index')*
2. *Each $\sim_A$-class is a regular langauge*

Proof.

(1): clearly a congruence. Equivalence classes correspond to pairs of functions $w_1 : Q \to \mathcal{P}(Q)$ and $w_2 : Q \times Q \to \mathcal{P}(Q)$ of which there are finitely many.

(2): define $W_{qq'} = \left\{ w \in \Sigma^* \mid q \xrightarrow{w} q' \right\}$ and similarly for $W^E_{qq'}$. Both are clearly regular. For $w \in \Sigma^*$, we have that

$$[w]_{\sim_A} = \bigcap \left\{ W_{qq'}, W^E_{qq'}, \overline{W_{qq'}}, \overline{W^E_{qq'}} \mid w \in \text{ each} \right\}$$

which is regular. □

Closure Under Complement (cont.)

Say that $\sim$ an equivalence relation over Σ^* saturates an ω-language L if for any pair of equivalence classes U and V,

$$UV^\omega \cap L \neq \emptyset \Rightarrow UV^\omega \subseteq L$$

Note: if $\sim$ saturates L, it also saturates $\overline{L}$.

Closure Under Complement (cont.)

Lemma 2.14

Let A be a n.d. Büchi automaton. Then $\sim_A$ saturates $L(A)$.

Proof.

Let U, V be $\sim_A$ equiv classes and suppose $UV^\omega \cap L(A) \neq \emptyset$. Then there is $w = uv_1v_2 \cdots \in UV^\omega \cap L(A)$ where $u \in U$, $v_i \in V \setminus \{\varepsilon\}$. Because $w \in L(A)$, there is a sequence of states $(q_i)_{i\in\omega}$ s.t. $q_0 \in Q_0$ and

$$q_0 \xrightarrow{u} q_1 \xrightarrow{v_1} q_2 \xrightarrow{v_2} q_3 \xrightarrow{v_3} \cdots$$

and for infinitely many i, $q_i \xrightarrow{v_i,F} q_{i+1}$. Now, take $w' = u'v'_1v'_2 \cdots \in UV^\omega$. We have $u \sim_A u'$ and $v_i \sim_A v'_i$. Thus $q_0 \xrightarrow{u'} q_1 \xrightarrow{v'_1} q_2 \xrightarrow{v'_2} q_3 \xrightarrow{v'_3} \cdots$ and for infinitely many i, $q_i \xrightarrow{v'_i,F} q_{i+1}$. Hence $w' \in L(A)$, as required. □

Closure Under Complement (cont.)

Lemma 2.15

Let $\sim$ be a congruence relation over Σ^* of finite index. Then, for every ω-word w, there are $\sim$-classes U, V s.t. $w \in UV^\omega$.

Proof.

Define $f_w : [\omega]^2 \to \Sigma^*/\sim$ by $f_w(\{i,j\}) = [w_i \dots w_{j-1}]_\sim$. Since $\sim$ is of finite index, by Ramsey's theorem, there is an infinite set $X \subseteq \omega$ s.t. all words $w_k \dots w_{l-1}$ for $k, l \in X$ are $\sim$-equiv. In particular, there is an infinite sequence $i_0 < i_1 < \dots \in X$ s.t. all segments $w_{i_j} \dots w_{i_{j+1}}$ belong to the same $\sim$-class. Let V be that class, and let U be the $\sim$-class of $w_0 \dots w_{i_0 - 1}$ $(= f_w(\{0, i_0\}))$. Then $w \in UV^\omega$. $\square$

Closure Under Complement (cont.)

Theorem 2.16

If L is ω-regular, then so too is $\overline{L} := \Sigma^\omega \setminus L$.

Proof.

Given A accepting L, $\sim_A$ saturates $L(A)$ and $\overline{L(A)}$ (two lemmas previous). By the previous lemma,

$$\overline{L(A)} = \bigcup \{UV^\omega \mid U, V \sim_A \text{-classes and } UV^\omega \cap L(A) = \emptyset\}$$

Because $\sim_A$ has finite index, this is a finite union. By the earlier (unproved) theorem of Büchi, it follows that $\overline{L(A)}$ is ω-regular. $\square$

The Main Theorem of This Section

Theorem 2.17 (Büchi [1962])

S1S *is decidable.*

Strategy: associate every formula $\varphi(X_1, \ldots, X_n)$ with a Büchi automaton A_φ and an ω-word w (over a fairly complicated alphabet) s.t. the formula holds in T_1 iff A_φ accepts w.

The ω-word

Let $V_1, \ldots, V_n \subseteq \omega$. We define an ω-word $W(V_1, \ldots, V_n)$ over the alphabet $\{0,1\}^n$ by

$$w_{i_j} = \chi_{V_j}(i)$$

for $i \in \omega, j \in \{1, \ldots, n\}$. As an example: let V_1 be the odds and V_2 the evens. We can visualize $W(V_1, V_2)$ as:

$$\begin{array}{ccccc} w_0 & w_1 & w_2 & w_3 & \ldots \\ 0 & 1 & 0 & 1 & \ldots \\ 1 & 0 & 1 & 0 & \ldots \end{array}$$

The Main Theorem

Theorem 2.18

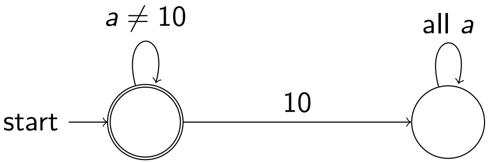
For every formula $\varphi(X_1, \ldots, X_n)$ in the monadic logic of one successor, one can effectively construct a n.d. Büchi automaton A_φ in alphabet $\{0,1\}^n$ such that for all $V_1, \ldots, V_n \subseteq \omega$,

$$T_1 \models \varphi[V_1, \ldots, V_n] \text{ iff } A_\varphi \text{ accepts } W(V_1, \ldots, V_n)$$

The proof will be by induction on formulas. First, we reformulate the language as a first-order langauge with binary relations $\subseteq$ and S. Variables range over subsets of ω, $\subseteq$ has its usual interpretation and $S(U, V)$ holds iff $U = \{m\}$ and $V = \{m+1\}$ for some $m \in \omega$.

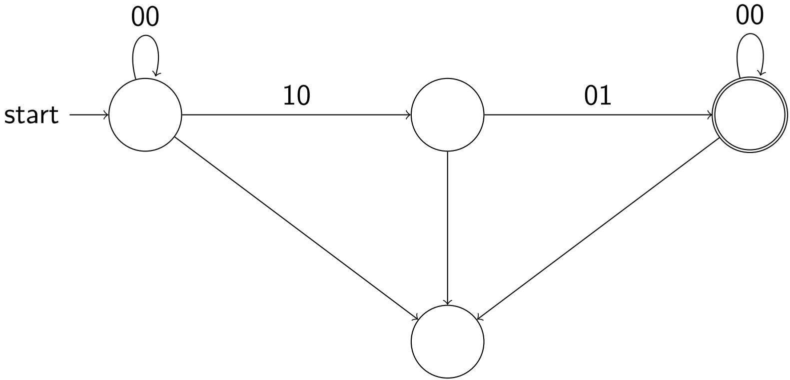
Base Case

Base case 1: φ is $X \subseteq Y$. We need an automaton that accepts all ω-words over $\{0,1\}^2$ that do not contain the letter 10 (corresponding to an element in X but not Y).



Base Case

Base case 2: φ is $S(X, Y)$. The automaton is:



Inductive Step

The negation, disjunction, and conjunction cases follow from the closure of the ω-regular languages under complement, union, and intersection respectively.

Now consider $\varphi\left(\vec{Y}\right) = \exists X \psi\left(X, \vec{Y}\right)$. By the IH, we have $A_\psi = (Q, Q_0, \delta, F)$ recognizing $W(U, \vec{V})$ whenever $T_1 \models \psi\left[U, \vec{V}\right]$. A_φ is just like A_ψ except that it has transition function

$$\delta'(q, \vec{a}) = \delta(q, 0\vec{a}) \cup \delta(q, 1\vec{a})$$

Intuitively, A_φ guesses a component for U and then runs A_ψ.

The Main Result

Thus, we have proved Theorem 2.18: For every formula $\varphi(X_1, \ldots, X_n)$ in the monadic logic of one successor, one can effectively construct a n.d. Büchi automaton A_φ in alphabet $\{0,1\}^n$ such that for all $V_1, \ldots, V_n \subseteq \omega$, $T_1 \models \varphi[V_1, \ldots, V_n]$ iff A_φ accepts $W(V_1, \ldots, V_n)$.

Corollary 2.19

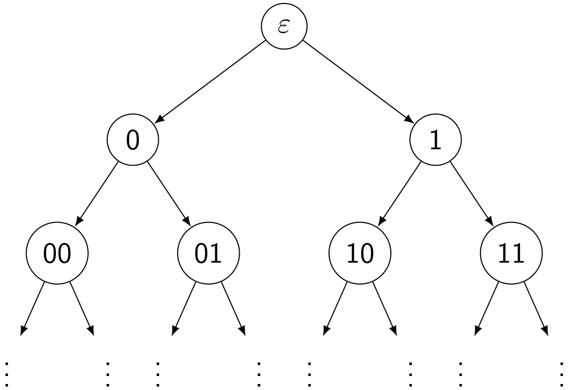
S1S *is decidable.*

Proof.

A sentence φ can be put in prenex form $\exists X_1 \ldots X_n \psi$. This is true iff $T_1 \models \psi[V_1, \ldots, V_n]$ for some assignment of V_i to X_i. By the above theorem, this holds iff $L(A_\psi) \neq \emptyset$, which we saw earlier is decidable. □

Plan

Rabin's Theorem



Theorem 3.1 (Rabin [1969])

S2S *is decidable.*

Strategy

The strategy for proving Rabin's Theorem resembles very closely the strategy for Büchi's decidability theorem.

First, we define automata which run on infinite trees (though we won't do so in full generality). Then, we prove that the emptiness problem is decidable, various closure properties (again, complementation will be the difficult one), and a theorem associating such automata to formulas in the language of **S2S**.

Note that the method I will use, which runs through a Forgetful Determinacy Theorem, is not Rabin's original. This method originates with Gurevich and Harrington [1982]. I will follow, with some modifications, Börger et al. [1997].

Tree Automata

A Σ-tree is a labeling $T : \{0,1\}^* \to \Sigma$.

Definition 3.2

A Σ-*tree automaton* is a quadruple $A = (Q, Q_0, \delta, \mathcal{F})$ where:

- Q is a finite set of *states*
- $Q_0 : \Sigma \to \mathcal{P}(Q)$ is the *initial table*
- $\delta : Q \times \Sigma \times \{0,1\} \to \mathcal{P}(Q)$ is the *transition function*
- $\mathcal{F} \subseteq \mathcal{P}(Q)$ is the *set of final collections of states*

Accepting Condition

To define the acceptance condition, we introduce a game $\Gamma(A, T)$ between the Automaton A and a player called Pathfinder P.
Automaton chooses $q_0 \in Q_0(T(\varepsilon))$. The players alternate. At odd numbered turns, Pathfinder chooses a direction $d_n \in \{0, 1\}$.
Automaton chooses a state

$$q_{n+1} \in \delta(q_n, T(d_0 \dots d_n), d_n)$$

Together, these define an infinite sequence $q_0 d_0 q_1 d_1 q_2 d_2 \dots$, called a *play* of the game. A finite prefix of a play is called a *position* of the game.
Automaton wins a play iff $Inf\left((q_i)_{i\in\omega}\right) \in \mathcal{F}$.
The automaton A accepts T iff Automaton has a winning strategy for $\Gamma(A, T)$.

Preliminary Definitions

The *node* of a position p is $Node(p) := (p_{2i+1})_{i \leq len(p)}$: the node of the binary tree that is currently being played.

Preliminary Definitions

The *node* of a position p is $Node(p) := (p_{2i+1})_{i \leq len(p)}$: the node of the binary tree that is currently being played.
Given $v \in \{0,1\}^*$ and Σ-tree T, the *v-residue of T* is the Σ-tree T_v s.t. $T_v(w) = T(vw)$.

Preliminary Definitions

The *node* of a position p is $Node(p) := (p_{2i+1})_{i \leq len(p)}$: the node of the binary tree that is currently being played.

Given $v \in \{0,1\}^*$ and Σ-tree T, the *v-residue of T* is the Σ-tree T_v s.t. $T_v(w) = T(vw)$.

Now, we define the *latest appearance record* $LAR(p)$. $LAR(\varepsilon)$ is a list of all states in some order. Pathfinder does not change LAR: $LAR(pd) = LAR(p)$ for $d \in \{0,1\}$ and p a position where Automaton has just moved. If $p = wq$ for $q \in Q$, then $LAR(p) = rq$ where r is the result of removing q from $LAR(w)$.

Intuitively: $LAR(p)$ lists the states in p without repetition in order of their latest appearance.

Preliminary Definitions

The *node* of a position p is $Node(p) := (p_{2i+1})_{i \leq len(p)}$: the node of the binary tree that is currently being played.

Given $v \in \{0,1\}^*$ and Σ-tree T, the *v-residue of T* is the Σ-tree T_v s.t. $T_v(w) = T(vw)$.

Now, we define the *latest appearance record* $LAR(p)$. $LAR(\varepsilon)$ is a list of all states in some order. Pathfinder does not change LAR: $LAR(pd) = LAR(p)$ for $d \in \{0,1\}$ and p a position where Automaton has just moved. If $p = wq$ for $q \in Q$, then $LAR(p) = rq$ where r is the result of removing q from $LAR(w)$.

Intuitively: $LAR(p)$ lists the states in p without repetition in order of their latest appearance.

A *strategy* for either player in $\Gamma(A, T)$ is a function from positions of that player to legal moves from that position (either states q or directions d).

Forgetful Determinacy

The key theorem to the approach we take is:

Theorem 3.3 (Gurevich and Harrington [1982])

One of the players has a strategy f for winning $\Gamma(A, T)$ s.t. the following 'forgetfulness' condition holds:
If p and q are positions from which the winner moves, such that $LAR(p) = LAR(q)$ and $T_{Node(p)} = T_{Node(q)}$, then $f(p) = f(q)$.

Proof.

Long and very hard. Börger et al. [1997], pp. 329-337 contains a proof (of a slightly more general version) which follows Zeitman [1994] and Yakhnis and Yakhnis [1990]'s improvements of the original proof. □

Emptiness Problem

First, the key decidable problem that we will use later.

Theorem 3.4

Given a Σ-tree automaton A, it is decidable whether $L(A) = \emptyset$.

Proof.

Let B be the $\{0\}$-tree automaton with the same states and final collection as A, but with $Q_0'(0) := \bigcup_{a \in \Sigma} Q_0(a)$ and $\delta'(q, 0, i) := \bigcup_{a \in \Sigma} \delta(q, a, i)$. Clearly, B accepts the unique $\{0\}$-tree T iff A accepts some Σ-tree.
By Forgetful Determinacy, a player has a forgetful winning strategy for $\Gamma(B, T)$. Let $f_1, \ldots, f_m$ be all of the forgetful strategies for Automaton and $g_1, \ldots, g_n$ those for Pathfinder. (Why only finitely many?) Plays eventually become periodic, so one can check each f_i against each g_j to determine whether B accepts T. $\square$

Closure Properties

Using constructions very analogous to those for Büchi automata, one can show that

Theorem 3.5

The class of languages accepted by Σ-tree automata are closed under union.
Moreover, given a $(\Sigma_1 \times \Sigma_2)$-tree automaton A, there is a Σ_1-tree automaton B that accepts T iff there is a Σ_2-tree T' s.t. A accepts (T, T').

These will be the key inductive steps in a later proof, along with closure under complement.

Complementation Theorem

We now prove the very important

Theorem 3.6

Given a Σ-tree automaton A, one can effectively constructe another one $\overline{A}$ s.t. $\overline{A}$ accepts T iff A rejects T. In other words, $L(\overline{A}) = \overline{L(A)}$.

Preliminary Definitions

First, some prelminary definitions, then two lemmas.
Let T be a Σ-tree, and g any forgetful strategy for Pathfinder.
WLOG, assume g is determinstic (i.e. $|g(p)| = 1$ for all positions).
Let R be set of all *a priori* possible LARs for A, i.e. lists of states containing each state at most once. Then, g can be viewed as $g : \{0,1\}^* \times R \to \{0,1\}$ since Pathfinder only moves on nodes.

Preliminary Definitions

First, some prelminary definitions, then two lemmas.
Let T be a Σ-tree, and g any forgetful strategy for Pathfinder. WLOG, assume g is determinstic (i.e. $|g(p)| = 1$ for all positions). Let R be set of all *a priori* possible LARs for A, i.e. lists of states containing each state at most once. Then, g can be viewed as $g : \{0,1\}^* \times R \to \{0,1\}$ since Pathfinder only moves on nodes. Call Δ be the set of all functions $h : R \to \{0,1\}$. View g as a Δ-tree G where

$$G(w) = \lambda r.g(w,r)$$

If we combine the labels of tree T and G, we have a $(\Sigma \times \Delta)$-tree denoted (T, G).

First Lemma

Lemma 3.7

Given A, one can effectively construct a $(\Sigma \times \Delta)$-tree automaton B s.t. Pathfinder wins $\Gamma(A, T)$ via the forgetful strategy g iff Automaton wins all plays of the game $\Gamma(B, (T, G))$.

For non-empty $r \in R$, let $last(r) := r_{len(r)}$ and let $u(r, q)$ be the LAR obtained from r by removing q and appending it to the end (so $last(u(r, q)) = q$).

Proof of Lemma 3.7

The construction: $B = (R \cup \{win\}, Q_0', \delta', \mathcal{F}')$ where:

- $Q_0'(ah) = Q_0(a)$
- $R_0 \in \mathcal{F}'$ iff either $win \in R_0$ or $\{last(r) \mid r \in R_0\} \notin \mathcal{F}$
- Transitions:

$$\delta'(win, ah, d) := win \text{ all } a, h, d$$

$$\delta'(r, ah, d) := \begin{cases} \{win\} & h(r) \neq d \\ \{u(r,q) \mid q \in \delta(last(r), a, d)\} & h(r) = d \end{cases}$$

So: if Pathfinder ever deviates from strategy G – when $h(r) \neq d$, this automaton goes to state *win* and never leaves. As long as Pathfinder plays strategy G, B simulares the old automaton A.

Proof of Lemma 3.7 (cont.)

We show Pathfinder wins $\Gamma(A, T)$ with g iff Automaton always wins $\Gamma(B, (T, G))$.

$\Rightarrow$: If Pathfinder ever deviates from G, Automaton clearly wins. If Pathfinder sticks to G, Automaton wins because the sequence of states corresponds to a sequence of LARs of a winning play in A for Pathfinder; these are exactly what is in $\mathcal{F}'$.

$\Leftarrow$: suppose Automaton A wins $\Gamma(A, T)$ with f against g. If Automaton B plays f in $\Gamma(B, (T, G))$, Pathfinder wins since the sequence of states played here will have final components corresponding to a winning collection in A.

Second Lemma

Lemma 3.8

For every Σ-tree automaton A, one can effectively construct another B which accepts a tree T iff Automaton wins all plays of $\Gamma(A, T)$.

Second Lemma

Lemma 3.8

For every Σ-tree automaton A, one can effectively construct another B which accepts a tree T iff Automaton wins all plays of $\Gamma(A, T)$.

That Automaton wins all plays of $\Gamma(A, T)$ means that each path $(d_i) \in \{0, 1\}^\omega$ satisfies:

(*) For all sequences $(q_i) \in Q^\omega$ s.t. $q_0 \in Q_0(T(\varepsilon))$ and $q_{n+1} \in \delta(q_n, T(d_0 \dots d_n), d_n)$, $Inf(q_i) \in \mathcal{F}$.

Second Lemma

Lemma 3.8

For every Σ-tree automaton A, one can effectively construct another B which accepts a tree T iff Automaton wins all plays of $\Gamma(A, T)$.

That Automaton wins all plays of $\Gamma(A, T)$ means that each path $(d_i) \in \{0, 1\}^\omega$ satisfies:

(*) For all sequences $(q_i) \in Q^\omega$ s.t. $q_0 \in Q_0(T(\varepsilon))$ and $q_{n+1} \in \delta(q_n, T(d_0 \ldots d_n), d_n)$, $Inf(q_i) \in \mathcal{F}$.

But (*) is expressible by an S1S-formula $\varphi\left(X, \vec{Y}\right)$ where X encodes (d_i) and $\vec{Y}$ encodes the sequence of labels (note: there will be one Y_i for each $a \in \Sigma$).

Proof of Lemma 3.8

By Theorem 2.18, there is a n.d. Büchi automaton C in alphabet $\{0,1\} \times \Sigma$ that accepts the pair of $d_0 d_1 d_2 \ldots$ and $T(\varepsilon)\, T(d_0)\, T(d_0 d_1) \ldots$ iff they satisfy (*).

Proof of Lemma 3.8

By Theorem 2.18, there is a n.d. Büchi automaton C in alphabet $\{0,1\} \times \Sigma$ that accepts the pair of $d_0 d_1 d_2 \dots$ and $T(\varepsilon)\, T(d_0)\, T(d_0 d_1) \dots$ iff they satisfy (*).
Now define the Σ-tree automaton B:

- $Q_B := Q_C$
- $Q_{B0}(a) := \bigcup_{q \in Q_{C0}} \bigcup_{i \in \{0,1\}} \delta_C(q, ia)$
- $\delta_B(q, a, d) := \delta_C(q, da)$
- $\mathcal{F}_B := \{X \subseteq Q_B \mid X \cap F_C \neq \emptyset\}$

Proof of Lemma 3.8

By Theorem 2.18, there is a n.d. Büchi automaton C in alphabet $\{0,1\} \times \Sigma$ that accepts the pair of $d_0 d_1 d_2 \dots$ and $T(\varepsilon)\, T(d_0)\, T(d_0 d_1) \dots$ iff they satisfy (*).
Now define the Σ-tree automaton B:

- $Q_B := Q_C$
- $Q_{B0}(a) := \bigcup_{q \in Q_{C0}} \bigcup_{i \in \{0,1\}} \delta_C(q, ia)$
- $\delta_B(q, a, d) := \delta_C(q, da)$
- $\mathcal{F}_B := \{X \subseteq Q_B \mid X \cap F_C \neq \emptyset\}$

Now, Automaton wins $\Gamma(B, T)$ iff for every (d_i) chosen by Pathfinder, $T(\varepsilon), d_0 T(d_0), d_1 T(d_0 d_1), \dots$ is accepted by C, iff A wins all plays of $\Gamma(A, T)$.

Finishing Proof of Complementation

Now, we finish the proof of the Complimentation Theorem 3.6. Given A, use Lemmas 3.7 and 3.8 to construct a $(\Sigma \times \Delta)$-automaton C that accepts (T, G) iff Pathfinder wins $\Gamma(A, T)$ by strategy g.
Where $C = (Q, Q_0, \delta, \mathcal{F})$, let $D = (Q, Q_0', \delta', \mathcal{F})$ be the Σ-tree automaton with

$$Q_0' := \bigcup_{b \in \Delta} Q_0(ab)$$

$$\delta'(q, a, d) := \bigcup_{b \in \Delta} \delta(q, ab, d)$$

D accepts a Σ-tree T iff there is a Δ-tree G s.t. C accepts (T, G) iff A rejects T.

Proving Decidability

The strategy will be identical to the Büchi case. We start by proving the analog of Theorem 2.18.
We reformulate S2S in a first-order langauge with binary predicates $\subseteq$, S_1, and S_2 where variables range over subsets of $\{0,1\}^*$. The interpretation of $\subseteq$ is standard, while $S_i(X,Y)$ iff $X=\{w\}$ and $Y=\{wi\}$.
Let $\Sigma=\{0,1\}$. For every tuple $V_1,\ldots,V_n$ of subsets of $\{0,1\}^*$, we define a Σ_n-tree $T(V_1,\ldots,V_n)$ by

$$T(V_1,\ldots,V_n)(w) := (\chi_{V_1}(w),\ldots,\chi_{V_n}(w))$$

Proof of Theorem 3.9

Our main theorem is:

Theorem 3.9

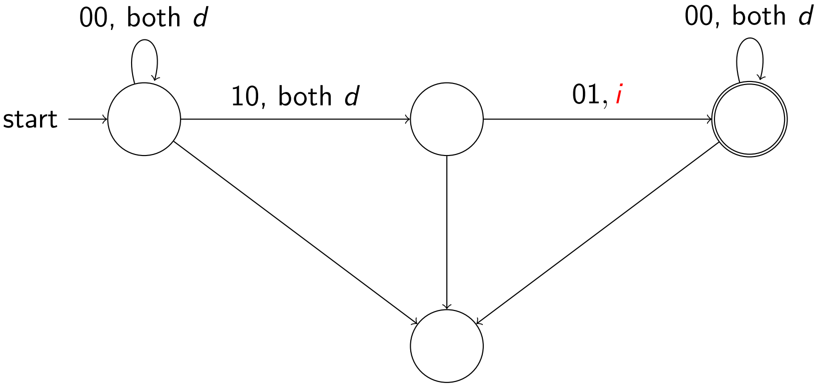
For every S2S-formula $\varphi(X_1, \ldots, X_n)$, one can effectively construct a Σ_n-tree automaton A_φ such that for all $V_1, \ldots V_n \subseteq \{0,1\}^*$,

$$T_2 \models \varphi[V_1, \ldots, V_n] \text{ iff } A_\varphi \text{ accepts } T(V_1, \ldots, v_n)$$

This is proved, as before, by induction on φ.
Base case 1: φ is $X \subseteq Y$. Take the same construction as in Theorem 2.18, where all transitions take place for both $d \in \{0,1\}$.

Proof of Theorem 3.9

Base case 2: φ is $S_i(X, Y)$. The automaton is a very slight modification of the Büchi one:



Proof of Theorem 3.9

Inductive step: negation is given by the Complementation Theorem 3.6. Disjunction and existential quantification were asserted in Theorem 3.5. We here provide a construction for the latter.
Consider $\varphi = \exists X \psi \left(X, \vec{Y} \right)$. By the IH, we have a $\{0,1\}^{n+1}$-tree automaton $A_\psi = (Q, Q_0, \delta, \mathcal{F})$ recognizing $T \left(U, \vec{V} \right)$ whenever $T_2 \models \psi \left[U, \vec{V} \right]$. A_φ is just like A_ψ except that it has transition function

$$\delta' \left(q, \vec{a}, d \right) := \delta \left(q, 0\vec{a}, d \right) \cup \delta \left(q, 1\vec{a}, d \right)$$

which intuitively 'guesses' a component U and runs A_ψ.

The Main Result

Corollary 3.10 (Rabin's Theorem)

S2S *is decidable.*

Proof.

A sentence in the language of S2S has a prenex form $\varphi := \exists X_1 \ldots X_n \psi$. This is true iff $T_2 \models \psi[V_1, \ldots, V_n]$ for some assignment of V_i to X_i. By the previous Theorem, this holds iff $L(A_\psi) \neq \emptyset$. We can check this since the emptiness problem for tree automata is decidable (Theorem 3.4). $\square$

Plan

Using Rabin's Theorem

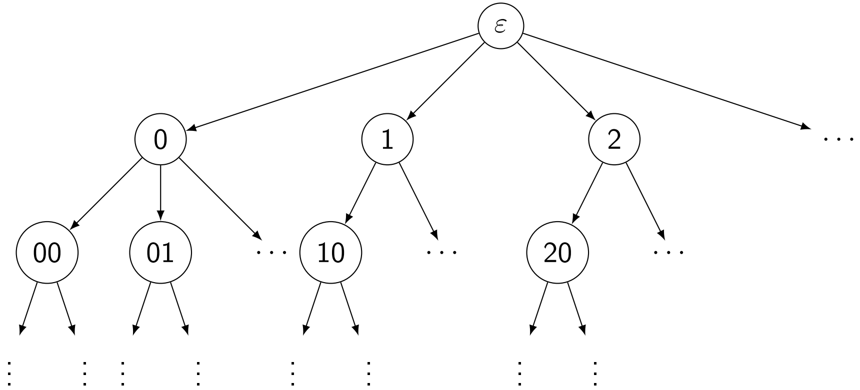
In this section, we show how to use Rabin's Theorem to prove that *other theories* are decidable. The basic strategy is to take models of the other theory (whether a single model or a class of models), embed them in T_2 in a way that is definable and then define a satisfaction-preserving translation.

We will look at:

1. **SωS**: the monadic second-order theory of ω-successors
2. **S4**: the modal logic of reflexive and transitive Kripke frames
   We will also mention that Rabin's Theorem can be used to prove modal logics decidable when more traditional methods (i.e. the finite model property) do not work.

I will conclude by mentioning some other decidability applications.

The Theory SωS



Theorem 4.1 (Rabin [1969])

The monadic second-order theory of T_ω is decidable.

The Idea of the Proof

In Rabin's original paper, but more detailed presentation in chapter 6 of Khoussainov and Nerode [2001]. We are interested in the structure

$$T_\omega = (\omega^*, (s_i)_{i\in\omega}, \leq, \preceq)$$

where the s_i are the usual successor functions, $\leq$ is the prefix ordering on the tree, and $\preceq$ is the lexicographic ordering. Note that these two are definable in **S2S**, but are not definable here in terms of just the successor functions, so we must include them. The idea:

1. Construct definable $D \subseteq T_2$, f_i on D, and relations $\leq_1$, $\preceq_1$ on D s.t.
2. $T_\omega \cong (D, (f_i)_{i\in\omega}, \leq_1, \preceq_1)$
3. Define a satisfiability-preserving translation between **SωS** and **S2S**

The Sub-structure of T_2

The relevant sub-structure of T_2, denoted $\mathfrak{D}$ is:

$$D = \{\varepsilon\} \cup \{1^{n_1}01^{n_2}0\ldots1^{n_k}0 \mid 1 \leq k, 1 \leq i \leq k, 1 \leq n_i\}$$
$$f_i = w \mapsto w1^{i+1}0$$
$$\leq_1 = \leq\restriction D$$
$$\preceq_1 = \preceq\restriction D$$

Theorem 4.2

$T_\omega \cong \mathfrak{D}$

Proof.

The mapping is $n_1n_2\ldots n_k \mapsto 1^{n_1+1}01^{n_2+1}0\ldots1^{n_k+1}0$. It's easy to check that this is an isomorphism. $\square$

Definability of $\mathfrak{D}$

Lemma 4.3

D is definable

Proof.

$x \in D$ iff $x = \varepsilon$ or $s_1(\varepsilon) \leq x$ and there is a proper prefix y of x s.t. $s_0(x) = y$ (i.e. y ends in 0) and for every proper prefix y_1 of x, if $s_0(x_1) < y$, then $s_1(s_0(x_1)) < y$ (i.e. non-terminal 0s are followed by 1s). Thus, D is defined by:

$$\varphi(x) := x = \varepsilon \vee [s_1(\varepsilon) \leq x \wedge \exists y \, (y < x \wedge s_0(y) = x) \wedge \forall y_1 \, (s_0(y_1) < x \rightarrow s_1(s_0(y_1) < x))]$$

But ε, $<$, $\leq$ are all definable in **S2S**. □

Clearly, $\leq_1$ and $\preceq_1$ are therefore definable.

Definability of $\mathfrak{D}$ (cont.)

To prove that the f_i are definable, we introduce a preliminary definition. If $w \in D$, call the nodes $w1^n0$ for $n \geq 1$ the *D-immediate successors of w*. We then have:

Lemma 4.4

1. *The D-immediate successors of w are in D*
2. *The set of D-immediate successors of w is definable.*
3. $w10 \preceq_1 w110 \preceq_1 w1110 \preceq_1 \cdots$

Proof.

(2) is the only non-obvious one. But y is a D-immediate successor of x is defined by:

$$\varphi(x, y) := x <_1 y \wedge \forall z \in D \left(x \leq_1 z \wedge z \leq_1 y \rightarrow z = x \vee z = y\right)$$

□

Definability of $\mathfrak{D}$ (cont.)

Recall the definition:

$$f_i := w \mapsto w1^{i+1}0$$

We convert this into an inductive definition which will be definable.

- $f_0(x) = y$ iff $x, y \in D$ and y is the smallest (w/r/t $\preceq_1$) D-immediate successor of x s.t. $x \preceq_1 y$.
- $f_{i+1}(x) = y$ iff $x, y \in D$ and y is the smallest (w/r/t $\preceq_1$) D-immediate successor of x s.t. $y \neq f_k(x)$ for all $k \leq i$.

Because f_0 is clearly definable and f_{i+1} is if all the f_k for $k \leq i$ are, it follows that all f_i are by induction on i.

Proving Decidability

We can now prove Theorem 4.1 from Rabin [1969]: The monadic second-order theory of T_ω is decidable.
We will take any sentence φ in the language of the structure T_ω and define a translation φ^t s.t. $T_\omega \models \varphi$ iff $T_2 \models \varphi^t$. This will reduce the decidabillity of **SωS** to the decidability of **S2S**.

$$(Xt)^t = Xt^t$$
$$(t_1 = t_2)^t = t_1^t = t_2^t$$
$$(x \leq y)^t = x \leq_1 y$$
$$(x \preceq y)^t = x \preceq_1 y$$

and $(\cdot)^t$ commutes with the connectives as expected.

Proving Decidability (cont.)

The quantifier cases are as expected:

$$(\exists x \varphi)^t = \exists x \left(x \in D \wedge \varphi^t\right)$$
$$(\exists X \varphi)^t = \exists X \left(X \subseteq D \wedge \varphi^t\right)$$

It's easy to check that $(\cdot)^t$ preserves satisfiability.

Proving Decidability (cont.)

The quantifier cases are as expected:

$$(\exists x \varphi)^t = \exists x \left(x \in D \wedge \varphi^t\right)$$
$$(\exists X \varphi)^t = \exists X \left(X \subseteq D \wedge \varphi^t\right)$$

It's easy to check that $(\cdot)^t$ preserves satisfiability.

Corollary 4.5

SnS, *for any $n \in \omega$ is decidable.*

Proof.

T_n is definable as a subset of T_ω by

$$\varphi(X) := X\varepsilon \wedge \forall x \left(Xx \wedge x \neq \varepsilon \rightarrow \exists y \left(Xy \wedge \bigvee_{0 \leq i \leq n} x = s_i(y) \right)\right)$$

$\square$

The Logic **S4**

Here, I follow section 6.3 of Blackburn et al. [2002].
S4 is the modal logic of reflexive, transitive frames. That is, it is the smallest set of formulas in the basic modal language containing

1. all propositional tautologies
2. (Dual): $\diamond p \leftrightarrow \neg\Box\neg p$
3. (K): $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$
4. (T): $p \rightarrow \diamond p$
5. (4): $\diamond\diamond p \rightarrow \diamond p$

and closed under modus ponens, uniform substitution, and necessitation (from p infer $\Box p$).
A logic satisfying (1), (2), and (3) and all the closure properties above is called *normal*.

Some Facts About S4

Theorem 4.6

S4 *is sound and strongly complete with respect to the class of reflexive, transitive models.*

Theorem 4.7

If a normal modal logic is sound and strongly complete w/r/t a first-order definable class of models M*, then it is also sound and strongly complete w/r/t the class of countable models in* M.

Corollary 4.8

S4 *is sound and complete w/r/t the class of countable, reflexive, transitive trees.*

Proof.

By the above theorems and the technique of *tree unraveling*. □

Proving S4's Decidability

Theorem 4.9

S4 *is decidable*

The strategy will be to identify models of **S4** with subtrees of T_ω and then write down an SωS sentence asserting **S4**-satisfiability of a formula.

$S \subseteq T_\omega$ is an *initial subtree* if $\varepsilon \in S$ and $y \in S$ and $x \leq y$ imply that $x \in S$. Let $\leq_S := \leq \restriction S$.

Proving S4's Decidability

Lemma 4.10

Let $\left(\vec{W}, \vec{R}\right)$ be the tree unravling of some countable frame (W, R) around point w and let R^* be the reflexive transitive closure of $\vec{R}$. Then $\left(\vec{W}, R^*\right) \cong (S, \leq_S)$ for some initial subtree S of T_ω.

Proof.

We inductively define an isomorphism f:

- $f(\langle w \rangle) = \varepsilon$ where $\langle w \rangle$ is the root of $\left(\vec{W}, R^*\right)$.
- Now, suppose for $\vec{u} \in \vec{W}$, $f(\vec{u}) = m$. The set $R^u = \left\{\vec{s} \in \vec{W} \mid \vec{u}\vec{R}\vec{s}\right\}$ is countable, so fix an enumeration of it. Define: $f(R_i^u) = s_i(m) = s_i(f(\vec{u}))$.

It's easy to check that this is an isomorphism. □

Proving S4's Decidability

As before, we have to steps left: (1) show that the class of initial subtrees is definable and (2) define an appropriate translation from the modal language to the language of T_ω.

For (1), we have

$$\begin{aligned} \mathrm{IST}\,(X) :=& \exists x\,(\mathrm{Root}(x) \wedge Xx) \wedge \\ & \forall yz\,((Xz \wedge y \leq z) \rightarrow Xy) \end{aligned}$$

where $\mathrm{Root}(x) := \neg\exists y\,(y < x)$.

$\leq_S$ is clearly defined by $Sx \wedge Sy \wedge x \leq y$.

Proving S4's Decidability

The translation $(\cdot)^t_{x,S}$ is essentially identical to the *standard translation* ST_x, except for the modality clause:

$$(\diamond\varphi)^t_{x,S} = \exists y \left(x \leq_S y \wedge (\varphi)^t_{y,S} \right)$$

Note that we need the free set variable S because we are not mapping to a unique substructure of T_ω.

Proving S4's Decidability

Now, we complete the proof. Let φ be a modal formula using propositional letters $p_1, \ldots, p_n$. Define the formula

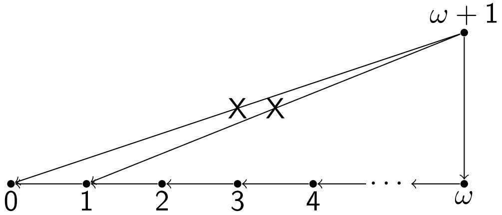
$$
\begin{aligned}
SatS4\,(\varphi) :=\; & \exists S \exists P_1 \ldots \exists P_n \exists x (\\
& \mathrm{IST}(S) \wedge P_1 \subseteq S \wedge \cdots \wedge P_n \subseteq S \wedge \\
& Sx \wedge (\varphi)^t_{x,S})
\end{aligned}
$$

One can check that $T_\omega \models SatS4\,(\varphi)$ iff $\varphi \in$ **S4** since the latter holds iff φ is satisfied at some node in a countable, reflexive transitive tree.

Thus, decidability of **S4** is reduced to the decidability of **S**ω**S**.

Other Modal Logics

Now, **S4** can be proved decidable by other methods (e.g. by having the finite model property).
The logic **KvB** is the logic of a *general frame* $\mathfrak{J}$ based on the frame:



with a certain collection of admissible sets on it.
KvB is not the logic of any class of frames and therefore does not have the finite model property. Nevertheless, the methods used here can be applied to it to show that **KvB** is decidable.

Other Applications

Rabin's Theorem has also been used to prove the following decidable:

1. The monadic second order theory of all countable (well-ordered) linearly ordered sets.
2. The first-order theory of Cantor's discontinuum. Cantor's discontinuum: $\{0,1\}^\omega$ with the product topology, which is isomorphic to the subset of $(0,1)$ given by the usual definition.
3. The second-order theory of all countable Boolean algebras (where set variables range over ideals).
4. Other modal logics: the modal μ-calculus, the computational tree logic CTL*.

References I

Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, June 2002.

Egon Börger, Erich Grädel, and Yuri Gurevich. *The Classical Decision Problem*. Perspectives in Mathematical Logic. Springer, Berlin, 1997.

Julius Richard Büchi. On a decision method in restricted second-order arithmetic. In *1960 International Congress for Logic, Methodology, and Philosophy of Science*, pages 1–11. Stanford University Press, 1962.

References II

Yuri Gurevich and Leo Harrington. Trees, Automata, and Games. In *Proceedings of the 14th Annual ACM Syposium on the Theory of Computing (STOC '82)*, pages 60–65, 1982. ISBN 0897910672.

Bakhadyr Khoussainov and Anil Nerode. *Automata Theory and its Applications*, volume 21 of *Progress in Computer Science and Applied Logic*. Springer, New York, 2001. ISBN 9781461266457.

Michael O Rabin. Decidability of Second-Order Theories and Automata on Infinite Trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

A Prasad Sistla, Moshe Y Vardi, and Pierre Wolper. The Complementation Problem for Büchi Automata with Applications to Temporal Logic. *Theoretical Computer Science*, 49:217–237, 1987.

References III

Moshe Y Vardi and Pierre Wolper. Reasoning about Infinite Computations. *Information and Computation*, 115:1–37, 1994.

Alexander Yakhnis and Vladimir Yakhnis. Extension of Gurevich-Harrington's Restricted Memory Determinacy Theorem: A Criterion for the Winning Player and an Explicit Class of Winning Strategies. *Annals of Pure and Applied Logic*, 48:277–297, 1990.

Suzanne Zeitman. Unforgettable Forgetful Determinacy. *Journal of Logic and Computation*, 4(3):273–283, 1994.

Thank You

Questions?