Rabin’s Tree Theorem and Applications

Shane Steinert-Threlkeld

November 20, 2013
Plan

1. Introduction
   - Explaining Rabin’s Theorem
   - A Perspective on Finite Automata

2. Infinite Automata and S1S
   - The Warm-up: S1S
   - Infinite Automata
   - Decidability of S1S

3. Infinite Tree Automata and S2S
   - Infinite Tree Automata
   - Closure Under Complement
   - Decidability of S2S

4. Applications
   - Decidability of $\omega$S
   - Decidability of Modal Logics
   - References
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.
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 \( \varepsilon \) 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])

\textit{The monadic second-order theory of the infinite binary tree is decidable.}

The monadic second-order theory of a structure $\mathcal{A}$ is the set of all monadic second-order sentences (in the appropriate signature) $\varphi$ such that $\mathcal{A} \models \varphi$.

So, the monadic second-order theory of the infinite binary tree is the set of all monadic second-order sentences $\varphi$ such that $T_2 \models \varphi$.

We call this theory $S2S$. 
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 \( \varphi \), 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 \( \chi_{S_2} \) is a recursive function.
Finite Automaton Example

\[ A = (\omega \upharpoonright 5, s) : \]

\[ \text{word:} \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \]

\[ \text{run:} \quad q_0 \]
A Perspective on Finite Automata

Finite Automaton Example

\[ \mathcal{A} = (\omega \upharpoonright 5, s) : \]

**Run:** \( q_0 \)

**Word:** 1 1 0 1 0 0

**Diagram:**
- Start state: \( q_0 \)
- Transitions:
  - From \( q_0 \) on 0 to \( q_0 \)
  - From \( q_0 \) on 1 to \( q_1 \)
  - From \( q_1 \) on 1 to \( q_0 \)
  - From \( q_1 \) on 0 to \( q_1 \)
Finite Automaton Example

\[ A = (\omega \upharpoonright 5, s): \]

\[ \text{word: } \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \]

\[ \text{run: } \quad q_0 \]
Finite Automaton Example

\[ \mathcal{A} = (\omega \upharpoonright 5, s) : \]

\[
\begin{array}{cccccc}
q_0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 2 & 3 & 4 & \\
\end{array}
\]

word: 1 1 0 1 0

run: \( q_0 \)
Finite Automaton Example

\[ \mathcal{A} = (\omega \upharpoonright 5, s) : \]

- **word:** 1 1 0 1 0 0
- **run:** q₀
Finite Automaton Example

\[ A = (\omega \upharpoonright 5, s): \]

word: 
1 1 0 1 0

run: 
q_0 \rightarrow q_1
Finite Automaton Example

\[ A = (\omega \upharpoonright 5, s): \]

- **word:** 1 1 0 1 0
- **run:** \( q_0 \) \( q_1 \)
A Perspective on Finite Automata

**Finite Automaton Example**

\[ \mathcal{A} = (\omega \upharpoonright 5, s) : \]

- **word:** \[1 \ 1 \ 0 \ 1 \ 0\]
- **run:** \[q_0 \quad q_1\]

Diagram:

- **start state:** \( q_0 \)
- **transitions:**
  - \( q_0 \) to \( q_1 \) on input 1
- **final states:** \( q_1 \)

Graph:

- **nodes:** \( q_0, q_1 \)
- **edges:**
  - \( q_0 \) to \( q_1 \) on input 1

Sequence:

- \(0, 1, 1, 0, 1, 0\)
Finite Automaton Example

**A** = \((\omega \upharpoonright 5, s)\):

- **word:** 1 1 0 1 0 0
- **run:** \(q_0 \rightarrow q_1\)
Finite Automaton Example

\[ A = (\omega \downarrow 5, s): \]

- word: 1 1 0 1 0
- run: 0 1 2 3 4
- \[ q_0 \rightarrow q_1 \rightarrow q_0 \]
A Perspective on Finite Automata

Finite Automaton Example

\[ A = (\omega \upharpoonright 5, s) : \]

<table>
<thead>
<tr>
<th>word:</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>run:</th>
<th>( q_0 )</th>
<th>( q_1 )</th>
<th>( q_0 )</th>
</tr>
</thead>
</table>

\[ 0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \]
Finite Automaton Example

\[
\mathcal{A} = (\omega \upharpoonright 5, s): \quad \begin{array}{cccccc}
0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 2 & 3 & 4
\end{array}
\]

\[
\text{word:} \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \\
\text{run:} \quad q_0 \quad q_1 \quad q_0
\]
Finite Automaton Example

\[
\mathcal{A} = (\omega \upharpoonright 5, s):
\]

<table>
<thead>
<tr>
<th>word:</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
</table>

\[
\begin{align*}
\text{run:} & \quad q_0 \quad q_1 \quad q_0 \\
\end{align*}
\]
Finite Automaton Example

\[ A = (\omega \upharpoonright 5, s) : \]

**word:** 1 1 0 1 0 1 0

**run:** \( q_0 \) \( q_1 \) \( q_0 \) \( q_0 \)
Finite Automaton Example

\( \mathcal{A} = (\omega \upharpoonright 5, s) : \)

- **word:** 1 1 0 1 0
- **run:** \( q_0 \) \( q_1 \) \( q_0 \) \( q_0 \)
A Perspective on Finite Automata

**Finite Automaton Example**

\[ \mathcal{A} = (\omega \upharpoonright 5, s) : \]

\[ \text{word: } \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \]

\[ \begin{align*}
\text{run: } & \quad q_0 \quad q_1 \quad q_0 \quad q_0 \\
\end{align*} \]
Finite Automaton Example

\( A = (\omega \upharpoonright 5, s) : \)

- **word:** 1 1 0 1 0
- **run:** \( q_0 \) \( \rightarrow \) \( q_1 \) \( \rightarrow \) \( q_0 \) \( \rightarrow \) \( q_0 \)
A Perspective on Finite Automata

Finite Automaton Example

\[ \mathcal{A} = (\omega \upharpoonright 5, s): \]

\[ \text{word: } 1 \ 1 \ 0 \ 1 \ 0 \]

\[ \text{run: } q_0 \ q_1 \ q_0 \ q_0 \ q_1 \]
Finite Automaton Example

\[ \mathcal{A} = (\omega \upharpoonright 5, s) : \]

- **word:** 1 1 0 1 0
- **run:** \( q_0 \) \( q_1 \) \( q_0 \) \( q_0 \) \( q_1 \)
**Finite Automaton Example**

\[ A = (\omega \upharpoonright 5, s) : \]

**word:**
\[ 1 \ 1 \ 0 \ 1 \ 0 \]

**run:**
\[ q_0 \ q_1 \ q_0 \ q_0 \ q_1 \]
Finite Automaton Example

A finite automaton is a mathematical model used to recognize patterns in strings of symbols. The automaton has a set of states, a start state, a set of inputs, and a transition function that determines which state the automaton moves to given an input.

In this example, the automaton \( \mathcal{A} = (\omega \upharpoonright 5, s) \) is defined over the alphabet \( \{0, 1\} \) with the start state \( s \). The word accepted by this automaton is \( 110100 \).

The transition run is as follows:

- Start at state \( q_0 \)
- Move to state \( q_1 \) after reading '1'
- Move back to state \( q_0 \) after reading '1'
- Move to state \( q_0 \) after reading '0'
- Move to state \( q_0 \) after reading '1'
- Move to state \( q_1 \) after reading '0'

The automaton transitions through the states according to the input sequence, ending in state \( q_1 \).

A perspective on finite automata provides a foundation for understanding more complex automata like infinite automata and tree automata.
Finite Automaton Example

\[ \mathcal{A} = (\omega \upharpoonright 5, s) : \]

\[ \begin{array}{ccccccc}
0 & 1 & 1 & 0 & 1 & 0 \\
\end{array} \]

run: \[ q_0 \rightarrow q_1 \rightarrow q_0 \rightarrow q_0 \rightarrow q_1 \rightarrow q_1 \]
Finite Automaton Example

\[ A = (\omega \upharpoonright 5, s) \):

- **word:** 1 1 0 1 0 0
- **run:** \( q_0 \) \( q_1 \) \( q_0 \) \( q_0 \) \( q_1 \) \( q_1 \) \( \notin F \)
Recall that finite automata (whether deterministic or non-deterministic) recognize the *regular languages*. Given an alphabet $\Sigma$, the regular languages in $\Sigma$ are the smallest collection of elements of $P(\Sigma^*)$ 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.
虽然我会假设熟悉有限自动机的基本概念，但我想用这种方式做例子，以强调几个将这些概念推广到无限对象和树上更容易理解的点：

**有限自动机教训**

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:
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

\[ r_{i+1} \in \delta (w_i, r_i) \]
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

  \[ r_{i+1} \in \delta(w_i, r_i) \]

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

  \[ r_{\text{len}(r)} \in F \]
Plan

1. Introduction
   - Explaining Rabin’s Theorem
   - A Perspective on Finite Automata

2. Infinite Automata and S1S
   - The Warm-up: S1S
   - Infinite Automata
   - Decidability of S1S

3. Infinite Tree Automata and S2S
   - Infinite Tree Automata
   - Closure Under Complement
   - Decidability of S2S

4. Applications
   - Decidability of $S_\omega S$
   - Decidability of Modal Logics
   - References
The Warm-up: S1S

Büchi’s Theorem

\[ \mathcal{A} = (\omega, s) : \]

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 \( \textbf{S1S} \). As a warm-up to Rabin’s theorem, we will first prove:

**Theorem 2.1 (Büchi [1962])**

\( \textbf{S1S} \textit{ is decidable.} \)
Any finite set $\Sigma$ will be called an *alphabet*. By $\Sigma^\omega$ we denote the set of $\omega$-sequences $w = w_0w_1w_2\ldots$ of elements of $\Sigma$, i.e. functions $w : \omega \to \Sigma$.

For $U \subseteq \Sigma^*$, $U^\omega$ is the set of $\omega$-words $u = u_0u_1u_2\ldots$ s.t. $u_i \in U$.

An *$\omega$-language* in $\Sigma$ is a subset of $\Sigma^\omega$.

Given an element $w \in \Sigma^\omega$, let

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

Definition 2.2

A non-deterministic Büchi automaton for alphabet $\Sigma$ 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
- $\delta : Q \times \Sigma \rightarrow \mathcal{P}(Q)$ is the transition function

A run of $A$ on an $\omega$-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.

$$\text{Inf}(q) \cap F \neq \emptyset$$
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 $\mathsf{NLOGSPACE}$-complete.*

**Theorem 2.4** (Vardi and Wolper [1994])

*The universality problem for Büchi automata – given $A$, does $L(A) = \Sigma^\omega$? – is $\mathsf{PSPACE}$-complete.*
An ω-language $L$ is called ω-regular iff it is of one of the forms:

- $U^\omega$ 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.
An automaton is deterministic 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 $\omega$-regular languages and, interestingly, have equally powerful deterministic versions.
**Lemma 2.6**

If $U \subseteq \Sigma^*$ is regular, then $U^\omega$ 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.

(Helper to draw a picture of this.)
**Lemma 2.7**

If $U \subseteq \Sigma^*$ is regular and $L \subseteq \Sigma^\omega$ is $\omega$-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.)
Lemma 2.8

If $L, L'$ are $\omega$-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 $L'$ respectively. WLOG, assume $Q(A)$ and $Q(A')$ are disjoint. Then, simply take the union of all components to get an automaton accepting $L \cup L'$. 
Lemma 2.9

If $L, L'$ are $\omega$-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 $L'$ respectively. 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

$$F'' = Q \times Q' \times \{2\}$$

$$\delta'' ((q, q', i), a) := \delta(q, a) \times \delta'(q', a) \times \{j\} \quad \text{where}$$

$$\begin{cases} 
  j = 1 & i = 0 \text{ and } q \in F \\
  j = 2 & i = 1 \text{ and } q' \in F' \\
  j = 0 & i = 2 \\
  j = i & \text{otherwise}
\end{cases}$$

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.

$\Box$
Closure Under Complement

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

**Theorem 2.10 (Büchi)**

$L \subseteq \Sigma^\omega$ is $\omega$-regular iff it can be represented as a finite union of sets $UV^\omega$ 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 $f : [\omega]^k \rightarrow M$, 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 $k = 2$. 
**Theorem 2.12**

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

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

\[
q \xrightarrow{u} q' \iff q \xrightarrow{v} q' \quad \text{and} \quad q \xrightarrow{u,F} q' \iff q \xrightarrow{v,F} q'
\]
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 language

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 \( W_{qq'}^E \). Both are clearly regular. For \( w \in \Sigma^* \), we have that

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

which is regular.
Say that $\sim$ an equivalence relation over $\Sigma^*$ saturates an $\omega$-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 $\overline{L}$. 


**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 $w' = u'v_1'v_2'\cdots \in UV^\omega$. 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.
Lemma 2.15

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

Proof.

Define $f_w : [\omega]^2 \to \Sigma^*/\sim$ by $f_w ([i,j]) = [w_i \ldots w_{j-1}]_\sim$. Since $\sim$ is of finite index, by Ramsey's theorem, there is an infinite set $X \subseteq \omega$ s.t. all words $w_k \ldots w_l$ for $k, l \in X$ are $\sim$-equiv. In particular, there is an infinite sequence $i_0 < i_1 < \cdots \in X$ s.t. all segments $w_{i_j} \ldots w_{i_{j+1}}$ belong to the same $\sim$-class. Let $V$ be that class, and let $U$ be the $\sim$-class of $w_0 \ldots w_{i_0-1}$ ($= f_w ([0, i_0])$). Then $w \in UV^\omega$. 
Theorem 2.16

If $L$ is $\omega$-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 $\omega$-regular. □
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 $A_\varphi$ and an $\omega$-word $w$ (over a fairly complicated alphabet) s.t. the formula holds in $T_1$ iff $A_\varphi$ accepts $w$. 
Let $V_1, \ldots, V_n \subseteq \omega$. We define an $\omega$-word $W(V_1, \ldots, V_n)$ over the alphabet $\{0, 1\}^n$ by

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

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:

$$\begin{align*}
w_0 & \quad w_1 & \quad w_2 & \quad w_3 & \quad \ldots \\
0 & \quad 1 & \quad 0 & \quad 1 & \quad \ldots \\
1 & \quad 0 & \quad 1 & \quad 0 & \quad \ldots
\end{align*}$$
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 $A_\varphi$ in alphabet $\{0, 1\}^n$ such that for all $V_1, \ldots, V_n \subseteq \omega$,

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

The proof will be by induction on formulas. First, we reformulate the language as a first-order language with binary relations $\subseteq$ and $S$. Variables range over subsets of $\omega$, $\subseteq$ has its usual interpretation and $S(U, V)$ holds iff $U = \{m\}$ and $V = \{m + 1\}$ for some $m \in \omega$. 
Base case 1: $\varphi$ is $X \subseteq Y$. We need an automaton that accepts all $\omega$-words over $\{0, 1\}^2$ that do not contain the letter 10 (corresponding to an element in $X$ but not $Y$).
Base case 2: \( \varphi \) is \( S(X, Y) \). The automaton is:
Decidability of S1S

Inductive Step

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

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

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

Intuitively, $A_\varphi$ guesses a component for $U$ and then runs $A_\psi$. 
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 \( A_\varphi \) in alphabet \( \{0, 1\}^n \) such that for all \( V_1, \ldots, V_n \subseteq \omega \),
\[
T_1 \models \varphi [V_1, \ldots, V_n] \text{ iff } A_\varphi \text{ accepts } W (V_1, \ldots, V_n).
\]

Corollary 2.19

**S1S is decidable.**

Proof.

A sentence \( \varphi \) 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

1. Introduction
   - Explaining Rabin’s Theorem
   - A Perspective on Finite Automata

2. Infinite Automata and S1S
   - The Warm-up: S1S
   - Infinite Automata
   - Decidability of S1S

3. Infinite Tree Automata and S2S
   - Infinite Tree Automata
   - Closure Under Complement
   - Decidability of S2S

4. Applications
   - Decidability of $S_\omega S$
   - Decidability of Modal Logics
   - References
Rabin’s Theorem

Theorem 3.1 (Rabin [1969])

*S2S* is decidable.
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].
A $\Sigma$-tree is a labeling $T : \{0, 1\}^* \rightarrow \Sigma$.

**Definition 3.2**

A $\Sigma$-tree automaton is a quadruple $A = (Q, Q_0, \delta, F)$ where:
- $Q$ is a finite set of states
- $Q_0 : \Sigma \rightarrow \mathcal{P}(Q)$ is the initial table
- $\delta : Q \times \Sigma \times \{0, 1\} \rightarrow \mathcal{P}(Q)$ is the transition function
- $F \subseteq \mathcal{P}(Q)$ 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

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

Together, these define an infinite sequence $q_0 d_0 q_1 d_1 q_2 d_2 \ldots$, called a play of the game. A finite prefix of a play is called a position of the game.

Automaton wins a play iff $\inf \ ( (q_i)_{i \in \omega} ) \in \mathcal{F}$.

The automaton $A$ accepts $T$ iff Automaton has a winning strategy for $\Gamma (A, T)$. 
Preliminary Definitions

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

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

Given $\nu \in \{0, 1\}^*$ and $\Sigma$-tree $T$, the $\nu$-residue of $T$ is the $\Sigma$-tree $T_\nu$ s.t. $T_\nu(w) = T(\nu w)$. 

Preliminary Definitions

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

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

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

Intuitively: $\text{LAR}(p)$ lists the states in $p$ without repetition in order of their latest appearance.
Preliminary Definitions

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

Given $v \in \{0, 1\}^*$ and $\Sigma$-tree $T$, the $v$-residue of $T$ is the $\Sigma$-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 $T_{\text{Node}}(p) = T_{\text{Node}}(q)$, 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 $\Sigma$-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 $\Sigma$-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 $T$. 

\[ \square \]
Using constructions very analogous to those for Büchi automata, one can show that

**Theorem 3.5**

*The class of languages accepted by *$\Sigma$*-tree automata are closed under union.*

*Moreover, given a $(\Sigma_1 \times \Sigma_2)$-tree automaton $A$, there is a $\Sigma_1$-tree automaton $B$ that accepts $T$ iff there is a $\Sigma_2$-tree $T'$ s.t. $A$ accepts $(T, T')$.  

These will be the key inductive steps in a later proof, along with closure under complement.*
We now prove the very important

**Theorem 3.6**

*Given a $\Sigma$-tree automaton $A$, one can effectively construct another one $\overline{A}$ s.t. $\overline{A}$ accepts $T$ iff $A$ rejects $T$. In other words, $L(\overline{A}) = \overline{L(A)}$.***
First, some preliminary definitions, then two lemmas.
Let $T$ be a $\Sigma$-tree, and $g$ any forgetful strategy for Pathfinder. WLOG, assume $g$ is deterministic (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.
First, some preliminary definitions, then two lemmas.

Let $T$ be a $\Sigma$-tree, and $g$ any forgetful strategy for Pathfinder.

WLOG, assume $g$ is deterministic (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 $g : \{0, 1\}^* \times R \rightarrow \{0, 1\}$ since Pathfinder only moves on nodes.

Call $\Delta$ be the set of all functions $h : R \rightarrow \{0, 1\}$. View $g$ as a $\Delta$-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 $last(r) := r_{\text{len}}(r)$ 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: \( B = (R \cup \{\text{win}\}, Q_0', \delta', F') \) where:

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

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

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

So: if Pathfinder ever deviates from strategy \( G \) – when \( h(r) \neq d \), this automaton goes to state \( \text{win} \) and never leaves. As long as Pathfinder plays strategy \( G \), \( B \) simulates 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 \( F' \).

\( \Leftarrow \): 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 $\Sigma$-tree automaton $A$, one can effectively construct another $B$ which accepts a tree $T$ iff Automaton wins all plays of $\Gamma (A, T)$.
Lemma 3.8

For every \( \Sigma \)-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:

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

Lemma 3.8

For every $\Sigma$-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 $q_{n+1} \in \delta (q_n, T (d_0 \ldots d_n), d_n), \inf (q_i) \in F$.

But (*) is expressible by an S1S-formula $\varphi (X, \vec{Y})$ 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 $d_0d_1d_2\ldots$ and $T(\varepsilon)T(d_0)T(d_0d_1)\ldots$ 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 $d_0 d_1 d_2 \ldots$ and $T(\epsilon) T(d_0) T(d_0 d_1) \ldots$ iff they satisfy (*).

Now define the $\Sigma$-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, $T(\epsilon), T(d_0), T(d_0 d_1), \ldots$ is accepted by $C$, iff $A$ wins all plays of $\Gamma(A, T)$. 
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 \( d_0d_1d_2 \ldots \) and \( T(\varepsilon) T(d_0) T(d_0d_1) \ldots \) iff they satisfy (*).

Now define the \( \Sigma \)-tree automaton $B$:

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

Now, Automaton wins $\Gamma (B, T)$ iff for every \( (d_i) \) chosen by Pathfinder, \( T(\varepsilon), d_0 T(d_0), d_1 T(d_0d_1), \ldots \) is accepted by $C$, iff $A$ wins all plays of $\Gamma (A, T)$. 
Now, we finish the proof of the Complementation 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 $C = (Q, Q_0, \delta, F)$, let $D = (Q, Q'_0, \delta', F)$ be the $\Sigma$-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 $\Sigma$-tree $T$ iff there is a $\Delta$-tree $G$ s.t. $C$ accepts $(T, G)$ iff $A$ rejects $T$. 

54 / 82
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 language 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 \( \Sigma_n \)-tree \( T (V_1, \ldots, V_n) \) by

\[
T (V_1, \ldots, V_n) (w) := (\chi_{V_1} (w), \ldots, \chi_{V_n} (w))
\]
Our main theorem is:

**Theorem 3.9**

*For every S2S-formula* $\varphi(X_1, \ldots, X_n)$, *one can effectively construct a* $\Sigma_n$-*tree automaton* $A_\varphi$ *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 $\varphi$.

**Base case 1**: $\varphi$ 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: \( \varphi \) 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 \( \varphi = \exists X \psi (X, \vec{Y}) \). By the IH, we have a \( \{0, 1\}^{n+1} \)-tree automaton \( A_\psi = (Q, Q_0, \delta, F) \) recognizing \( T(U, \vec{V}) \) whenever \( T_2 \models \psi [U, \vec{V}] \). \( A_\varphi \) is just like \( A_\psi \) except that it has transition function

\[
\delta'(q, \vec{a}, d) := \delta(q, 0\vec{a}, d) \cup \delta(q, 1\vec{a}, d)
\]

which intuitively ‘guesses’ a component \( U \) and runs \( A_\psi \).
Corollary 3.10 (Rabin’s Theorem)

$\mathsf{S2S}$ is decidable.

Proof.

A sentence in the language of $\mathsf{S2S}$ 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).
Plan

1. Introduction
   - Explaining Rabin’s Theorem
   - A Perspective on Finite Automata

2. Infinite Automata and S1S
   - The Warm-up: S1S
   - Infinite Automata
   - Decidability of S1S

3. Infinite Tree Automata and S2S
   - Infinite Tree Automata
   - Closure Under Complement
   - Decidability of S2S

4. Applications
   - Decidability of $S_\omega S$
   - Decidability of Modal Logics
   - References
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. $S_{\omega}$: the monadic second-order theory of $\omega$-successors
2. $S_4$: 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 $S_\omega S$

Theorem 4.1 (Rabin [1969])

The monadic second-order theory of $T_\omega$ is decidable.
In Rabin’s original paper, but more detailed presentation in chapter 6 of Khoussainov and Nerode [2001]. We are interested in the structure

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

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 $S\omega S$ and $S2S$
The Sub-structure of $T_2$

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

$$D = \{ \varepsilon \} \cup \{ 1^{n_1}01^{n_2}0 \ldots 1^{n_k}0 \mid 1 \leq k, 1 \leq i \leq k, 1 \leq n_i \}$$

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

$\leq_1 = \leq \upharpoonright D$

$\preceq_1 = \preceq \upharpoonright D$

Theorem 4.2

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

Proof.

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

**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:

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

But $\varepsilon$, $<$, $\leq$ are all definable in $\mathbf{S2S}$. Clearly, $\leq_1$ and $\preceq_1$ are therefore definable.
Definability of $\mathcal{D}$ (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:

$$\varphi(x, y) := x <_1 y \land \forall z \in D (x \leq_1 z \land z \leq_1 y \rightarrow z = x \lor z = y)$$
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 \( i \).
We can now prove Theorem 4.1 from Rabin [1969]: The monadic second-order theory of $T_\omega$ is decidable.

We will take any sentence $\varphi$ in the language of the structure $T_\omega$ and define a translation $\varphi^t$ s.t. $T_\omega \models \varphi$ iff $T_2 \models \varphi^t$. This will reduce the decidability of $S\omega S$ 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.
The quantifier cases are as expected:

\[(\exists x \varphi)^t = \exists x (x \in D \land \varphi^t)\]
\[(\exists X \varphi)^t = \exists X (X \subseteq D \land \varphi^t)\]

It’s easy to check that \((\cdot)^t\) preserves satisfiability.
The quantifier cases are as expected:

\[(\exists x \varphi)^t = \exists x \left( x \in D \land \varphi^t \right)\]

\[(\exists X \varphi)^t = \exists X \left( X \subseteq D \land \varphi^t \right)\]

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

**Corollary 4.5**

\(\text{SnS}, \text{ for any } n \in \omega \text{ is decidable.}\)

**Proof.**

\(T_n\) is definable as a subset of \(T_\omega\) by

\[\varphi (X) := X\varepsilon \land \forall x \left( Xx \land x \neq \varepsilon \rightarrow \exists y \left( Xy \land \bigvee_{0 \leq i \leq n} x = s_i (y) \right) \right)\]
The Logic \textbf{S4}

Here, I follow section 6.3 of Blackburn et al. [2002]. \textbf{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): $\Box (p \to q) \to (\Box p \to \Box q)$
4. (T): $p \to \diamond p$
5. (4): $\diamond \diamond p \to \diamond p$

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 \textit{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.
The strategy will be to identify models of $\mathbf{S4}$ with subtrees of $T_{\omega}$ and then write down an $S_{\omega}S$ sentence asserting $\mathbf{S4}$-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 $\leq_S := \leq | S$. 

**Theorem 4.9**

$\mathbf{S4}$ is decidable
Lemma 4.10

Let \((\vec{W}, \vec{R})\) 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 \((\vec{W}, R^*) \cong (S, \leq_S)\) for some initial subtree \(S\) of \(T_\omega\).

Proof.

We inductively define an isomorphism \(f\):

- \(f(\langle w \rangle) = \varepsilon\) where \(\langle w \rangle\) is the root of \((\vec{W}, R^*)\).

- Now, suppose for \(\vec{u} \in \vec{W}, f(\vec{u}) = m\). The set \(R^u = \{ \vec{s} \in \vec{W} \mid \vec{u}\vec{R}\vec{s}\}\) is countable, so fix an enumeration of it. Define: \(f(R^u_i) = s_i(m) = s_i(f(\vec{u}))\).

It's easy to check that this is an isomorphism.
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 $T_\omega$.

For (1), we have

$$\text{IST } (X) := \exists x \ (\text{Root}(x) \land Xx) \land$$

$$\forall yz ((Xz \land y \leq z) \rightarrow Xy)$$

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

$\leq_S$ is clearly defined by $Sx \land Sy \land x \leq y$. 
The translation \((\cdot)^t_{x,S}\) is essentially identical to the standard translation \(ST_x\), except for the modality clause:

\[
(\diamond \varphi)^t_{x,S} = \exists y \left( x \leq_S y \land (\varphi)^t_{y,S} \right)
\]

Note that we need the free set variable \(S\) because we are not mapping to a unique substructure of \(T_\omega\).
Now, we complete the proof. Let $\varphi$ be a modal formula using propositional letters $p_1, \ldots, p_n$. Define the formula

$$\text{Sat}_{S4}(\varphi) := \exists S \exists P_1 \ldots \exists P_n \exists x (\text{IST}(S) \land P_1 \subseteq S \land \cdots \land P_n \subseteq S \land Sx \land (\varphi)_x^t, S)$$

One can check that $T_\omega \models \text{Sat}_{S4}(\varphi)$ iff $\varphi \in S4$ since the latter holds iff $\varphi$ is satisfied at some node in a countable, reflexive transitive tree.

Thus, decidability of $S4$ is reduced to the decidability of $S_\omega S$. 
Now, $\text{S4}$ can be proved decidable by other methods (e.g. by having the finite model property).

The logic $\text{KvB}$ is the logic of a general frame $\mathfrak{J}$ based on the frame:

$\omega + 1$

with a certain collection of admissible sets on it. $\text{KvB}$ 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 $\text{KvB}$ is decidable.
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 \( \mu \)-calculus, the computational tree logic CTL*.
References I


References II


Questions?