and $k_j < |S_2|$. Now note that $s_i \in F_1$ since $\alpha(0,i) \in W$. Moreover, neither of $\mathbf{s}_i(0), \ldots, \mathbf{s}_i(k_i - 1)$ equals $s_{0,2}$. Otherwise, there would be a prefix $\alpha(0,k)$ of $\alpha(0,i)$ such that $\alpha(k,\omega) \in L$, which would contradict the choice of i. Now note that t_{i+1} is

$$(s_{i+1}, T_2(\mathbf{s}_i s_{0,2}, \alpha(i)), \mu(T_2(\mathbf{s}_i s_{0,2}, \alpha(i))).$$

The computation of the sequential Rabin automaton $\mathcal{A}_2$:

$$\mathbf{r}_2 = s_{0,2}, T_2(s_{0,2}, \alpha(i)), T_2(T_2(s_{0,2}, \alpha(i)), \alpha(i+1)), \ldots$$

on $\alpha(i,\omega)$ is successful. Hence there exists an r such that $In(\mathbf{r}_2) \bigcap P_r \neq \emptyset$ and $In(\mathbf{c}_2) \bigcap N_r = \emptyset$. Moreover this run will never merge with the run

$$T_2(\mathbf{s}_i(m), \alpha(i)), T_2(T_2(\mathbf{s}_i(m), \alpha(i)), \alpha(i+1)), \ldots,$$

where $0 \leq m \leq k_i - 1$. Again, otherwise it would contradict the choice of i. Therefore, we conclude that there must exist a $k \leq k_i$ such that $In(\mathbf{c}) \bigcap P_{r,k} \neq \emptyset$ and $In(\mathbf{c}) \bigcap N_{r,k} = \emptyset$. This proves that α is accepted by the sequential Rabin automaton $\mathcal{A}$.

Suppose that α is accepted by $\mathcal{A}$. We need to show that $\alpha \in WL$. Let us consider the computation $\mathbf{r}$ of $\mathcal{A}$ on α:

$$\mathbf{r} = t_0, t_1, \ldots, t_i, t_{i+1}, \ldots,$$

where each t_j is $(s_j, \mathbf{s}_j, n_j)$ as in the notation above. There exist r, k such that $In(\mathbf{c}) \bigcap P_{r,k} \neq \emptyset$ and $In(\mathbf{c}) \bigcap N_{r,k} = \emptyset$. Take the least m such that for all $j > m$, $t_j \notin N_{r,k}$. It follows by the definition of $N_{r,k}$ that $\mathbf{s}_j(k)$ exists and does not belong to N_r. The sequence

$$\mathbf{s}_m(k)\mathbf{s}_{m+1}(k), \mathbf{s}_{m+2}(k) \ldots$$

is a computation of $\mathcal{A}_2$ on $\alpha(m,\omega)$ which begins from $\mathbf{s}_m(k)$. Hence there must exist an $i \leq m$ such that $\alpha(0,i) \in W$ and $\mathbf{s}_i(k), \ldots, \mathbf{s}_{m-1}(k), \mathbf{s}_m(k)$ is a computation of $\mathcal{A}_2$ on $\alpha(i,m)$ such that $\mathbf{s}_i(k) = s_{0,2}$. We conclude that $\alpha \in WL$. This proves the lemma.

3.9 Decidability

In this section, as we did for finite automata, we will discuss certain natural problems posed about Büchi automata. As we will see these problems are decidable. Decidability of these problems gives a powerful tool for applications of Büchi automata in logic, algebra, and computer science. In the next two sections we will discuss examples of such applications. Let us now to formulate the problems.

1. **(Emptiness Problem)** Does there exist an effective procedure which, given any Büchi automaton $\mathcal{A}$, decides whether some $\alpha \in \Sigma^\omega$ is accepted by $\mathcal{A}$?

2. **(Equality Problem)** Does there exist an effective procedure which, given Büchi automata $\mathcal{A}$ and $\mathcal{B}$, decides whether these automata accept the same ω-language?

3. **(Universality Problem)** Does there exist an effective procedure which, given any Büchi automaton $\mathcal{A}$, decides whether $L(\mathcal{A}) = \Sigma^\omega$?

As we see, these problems are reformulations of the questions we asked about finite automata. However, the solutions to these problems are not as easy as for finite automata, as some of them employ results obtained in the previous sections. In particular, the solutions to the last two problems are based on the proof of the complementation problem for Büchi automata. This section is devoted to giving positive solutions to all these problems.

Theorem 3.9.1 *The emptiness problem for Büchi automata is decidable.*

Proof. A basic tool for solving this problem is the transition graph representation for Büchi automata. Thus, let $\mathcal{A} = (S, I, T, F)$ be a Büchi automaton over Σ. By the Büchi characterization theorem of ω-languages the language $L(\mathcal{A})$ is equal to

$$\bigcup_{s \in I, s' \in F} W_{s,s'} W^{\omega}_{s',s'}.$$

Hence, the set $L(\mathcal{A})$ is not empty if and only if there exists a state $s \in F$ so that the following two conditions hold:

1. There exists a finite string u which transforms some initial state s_0 to s.

2. There exists a finite nonempty string v which transforms the state $s \in F$ into s itself.

In other words, in order to decide if the language $L(\mathcal{A})$ is empty or not, we have to look at the transition table that represents $\mathcal{A}$ and do the following. We have to find a path labeled by some $u \in \Sigma^\star$ that connects an initial state s_0 with a final state s, and then for each such final state s check whether or not there exists a loop labeled with a string $v \in \Sigma^\star$ which transforms the final state s to s itself. If for some $s \in F$ that is the case, then clearly $\mathcal{A}$ recognizes $uvvvvv\ldots$. Otherwise, $\mathcal{A}$ accepts no string. It is easy to see that one can construct an algorithm that decides whether or not there exist a state $s \in F$ and strings $u, v \in \Sigma^\star$ that satisfy the two properties above. The theorem is proved.

Exercise 3.9.1 *Prove the theorem above in more detail.*

Theorem 3.9.2 *The equality problem for Büchi automata is decidable.*

Proof. To decide the equality problem for Büchi automata, we need the following lemma.

Lemma 3.9.1 *There exists an algorithm which, given a Büchi automaton $\mathcal{A}$, constructs a Büchi automaton $\mathcal{A}'$ that accepts the complement of $L(\mathcal{A})$.*

Proof of the Lemma. Here is a step-by-step explanation of how the desired algorithm works.

1. Take the given Büchi automaton $\mathcal{A}$.
2. Consider the equivalence relation $\sim$ introduced in Büchi's proof of the complementation problem.
3. Compute the index of the equivalence relation $\sim$.
4. Using the definition of $\sim$, for each $\sim$-equivalence class effectively construct a finite automaton recognizing the class.
5. For every pair of $\sim$-equivalence classes V and W, effectively construct a Büchi automaton recognizing the language $VW^\omega \bigcap L(\mathcal{A})$.
6. Take all V, W equivalence classes such that $VW^\omega \bigcap L(\mathcal{A}) = \emptyset$.
7. Construct a Büchi automaton that accepts the union of all VW^ω such that $VW^\omega \bigcap L(\mathcal{A}) = \emptyset$.

Each step of the above algorithm can be effectively carried out. The last step of the algorithm provides the desired automaton, as follows from the proof of the complementation problem for Büchi automata. This proves the lemma.

Now in order to prove the theorem, consider two Büchi automata $\mathcal{A}_1$ and $\mathcal{A}_2$. By the lemma above we can effectively construct two Büchi automata $\mathcal{B}_1$ and $\mathcal{B}_2$ such that $\mathcal{B}_1$ accepts $L(\mathcal{A}_1) \setminus L(\mathcal{A}_2)$ and $\mathcal{B}_2$ accepts $L(\mathcal{A}_2) \setminus L(\mathcal{A}_1)$. Then $L(\mathcal{A}_1) = L(\mathcal{A}_2)$ if and only if $L(\mathcal{B}_1) = L(\mathcal{B}_2) = \emptyset$. This gives a desired proof of the theorem.

Theorem 3.9.3 *The universality problem for Büchi automata is decidable.*

Proof. The decision of the universality problem is as follows. Take a Büchi automaton $\mathcal{A}$. Then $L(\mathcal{A}) = \Sigma^\omega$ if and only if $\Sigma^\omega \setminus L(\mathcal{A})$ is empty. Thus we have to simply construct the automaton $\mathcal{C}$ which accepts $\Sigma^\omega \setminus L(\mathcal{A})$ and apply the first theorem to $\mathcal{C}$. The theorem is proved.

We now present several exercises. Solutions to these exercises require some analysis of the transition tables of Büchi automata.

Exercise 3.9.2 *Prove that there exists an effective procedure which, given any Büchi automaton $\mathcal{A}$, decides whether or not $\mathcal{A}$ accepts finitely many strings.*

Exercise 3.9.3 *Prove that there exists an effective procedure which, given any Büchi automaton $\mathcal{A}$, decides whether or not $\mathcal{A}$ accepts countably many strings.*

3.10 Büchi Automata and the Successor Function

Important and natural structures that arise in computer science and mathematics involve numbers. Perhaps, the most simple nontrivial and important mathematical structure which involves numbers is the set of natural numbers *together* with the **successor function.** We denote the successor function by S. The successor function S works in a simplest possible way. The function sends 0 to 1, 1 to 2, 2 to 3, etc. In general, S sends n to $n+1$. We write this as, $S(n) = n+1$. We denote this structure, that is the natural numbers together with the successor function S, by (ω, S). Let us call this structure the **successor structure**. Büchi was the first who observed a close relationship between automata on infinite inputs and some natural properties of the successor structure.

The goal of this section can be informally described as follows. We want to express properties of the structure (ω, S) using a formal language. Then we would like to have a procedure which, given any expression Φ of the language, tells us whether or not the property expressed by Φ holds in (ω, S). To do this we employ our results about Büchi automata. The reader has probably noticed that we would like to investigate the relationship between the successor structure and Büchi automata in the same manner as we studied the relationship between finite automata and the monadic second order logic of finite strings. In this section we shall use results about Büchi automata, their computations, computational properties and show how these are related to algebraic and logical properties of the successor structure.

3.10.1 ω-Strings as Structures. We will show how each infinite string α over Σ can be viewed as a mathematical structure richer than the successor structure (ω, S). Suppose that

$$\alpha = \sigma_0\sigma_1\sigma_2\ldots$$

is an ω-string. Consider the structure (ω, S). Take a σ from Σ. Now this σ defines a subset Q_σ on natural numbers in the following way:

A natural number n belongs to the subset Q_σ if and only if $\sigma_n = \sigma$.

Let us consider an example.

Example 3.10.1 *Suppose that $\Sigma = \{a, b, c\}$. Consider the word*

$$abcabcabcabc\ldots.$$

Then

$$Q_a = \{0, 3, 6, 9, \ldots\},\quad Q_b = \{1, 4, 7, 10, \ldots\},\quad Q_c = \{2, 5, 8, 11, \ldots\}.$$

In this example, the string $abcabcabc\ldots$ defines the structure which contains all natural numbers, the successor function, and the sets Q_a, Q_b, and Q_c. In other words, we have the following mathematical structure:

$$(\omega, S, Q_a, Q_b, Q_c).$$

This structure is indeed richer than (ω, S) just because it contains more information about natural numbers than the structure (ω, S) itself.

Now we can explain this more formally. Given an ω-string α over $\Sigma = \{a_1, \ldots, a_n\}$, we can *expand* (ω, S) by considering the structure

$$(\omega, S, Q_{a_1}, \ldots, Q_{a_n}),$$

where each Q_a contains all those natural numbers n such that $\alpha(n) = a$. We denote this structure by $(\omega, S)_\alpha$.

There is another way to look at the structures of the type $(\omega, S)_\alpha$. We explain this using the above example.

In the example, the alphabet is $\{a, b, c\}$. We can look at the alphabet $\{a, b, c\}$ as a subset of the alphabet $\{0, 1\}^2$ in such a way that

$$a = \begin{pmatrix}1\\0\end{pmatrix}, \quad b = \begin{pmatrix}0\\1\end{pmatrix}, \quad c = \begin{pmatrix}1\\1\end{pmatrix}.$$

Note that any Büchi recognizable language over the alphabet $\{a, b, c\}$ is also Büchi recognizable over the alphabet $\{0, 1\}^2$. Therefore, as our primary interest is in Büchi recognizable language, we can assume that $\Sigma = \{0, 1\}^2$. Now the ω-word $abcabcabcabc\ldots$ has become

$$\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\ldots.$$

This infinite word defines the pair (X_1, X_2) of subsets on the set ω of natural numbers as follows. A natural number n belongs (does not belong) to X_1 if and only if the first coordinate of the letter in the nth position of the above ω-string is 1 (0). Similarly, a natural number n belongs (does not belong) to X_2 if and only if the second coordinate of the letter in the nth position of the above ω-string is 1 (0). Hence we see that

$$X_1 = \{0, 2, 3, 5, 6, 8, 9, \ldots\}, \quad X_2 = \{1, 2, 4, 5, 7, 8, 10, 11, \ldots\}.$$

Note that the first (the second) row of the sequence

$$\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\ldots$$

is simply the characteristic function of the set X_1 (X_2). Therefore, the pair (X_1, X_2) defines the sequence

$$\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}\ldots$$

in the following way. Take natural number n. Define the letter

$$\begin{pmatrix}\epsilon_1\\\epsilon_2\end{pmatrix}$$

as follows: if $n \in X_1$, then $\epsilon_1 = 1$; otherwise $\epsilon_1 = 0$. Similarly, if $n \in X_2$, then $\epsilon_2 = 1$; otherwise $\epsilon_2 = 0$. Now note that this letter belongs to $\{0, 1\}^2$ and is exactly the letter in the nth position of the ω-sequence above.

Thus, informally we can conclude with the following comments about the example. If we know α, then we know the structure $(\omega, S)_\alpha$. If we know $(\omega, S)_\alpha$, we certainly know α. If we know α, we certainly know the pair (X_1, X_2). If we know the pair (X_1, X_2), then we know α. Roughly speaking, the three objects, α, $(\omega, S)_\alpha$, and (X_1, X_2), are essentially the same and possess, in the natural sense explained above, the same information.

3.10.2 Monadic Second Order Formalism. Our goal can informally be described as follows. First, we want to express properties of the structure (ω, S). A natural way to do this is to invent a formal language. The language for us will be an important tool to talk about properties of (ω, S) as well as structures of the type $(\omega, S)_\alpha$. Second, we want to have a procedure which, given any expression Φ of the language, tells us whether or not the property expressed by Φ holds in (ω, S).[5] In order to achieve our goal, we explain the language called the **monadic second order logic** of the successor S.

The symbols which we use in this logic (language) are the following.

1. We use the symbols

   $$x, y, z, \ldots,$$

   called **individual variables**, possibly with subscripts or superscripts. The intended meaning of these variables is that they run over the set of natural numbers.

2. We use the symbols

   $$X, Y, Z, \ldots,$$

   called **set variables**, possibly with subscripts or superscripts. The intended meaning of these variables is that they run over subsets of ω.

3. We use symbol $=$ to express **equality**; we use symbol $\in$ to express **membership**; we also use the symbol S whose meaning is simply the successor function S in ω.

4. We use logical connectives & (and), $\bigvee$ (or), $\neg$ (not), $\rightarrow$ (if ..., then ...), and quantifiers $\exists$ (there exist), $\forall$ (for all).

[5] The reader may want to review the section on the monadic second order theory of finite strings from the previous chapter. Then this section can be read quickly.

5. We also use nonlogical symbols (and), left and right parentheses.

Now, we would like to talk about properties of (ω, S) in terms of just introduced symbols. The properties are expressed by means of certain finite strings of the language. These strings will be called formulas. Formulas formalize the notion of expression. We first define the notion of term. Informally, the meaning of each term in the successor structure is a function that can be obtained from the successor function S by composition.

Basic Step. Each variable x is a term.

Inductive Step. If t is a term, then $S(t)$ is also a term.

Exercise 3.10.1 *Let x be a variable. For every $n \in \omega$, define $S^n(x)$ as follows:*

$$S^0(x) = x, \quad \textit{and} \quad S^{n+1}(x) = S(S^n(x)).$$

Show that t is a term if and only if $t = S^n(x)$ for some variable x and number n.

We now define, by induction, formulas of our language.

Basic Step. The expressions $t \in X$, $t_1 = t_2$, $X = Y$, where t_1 and t_2 are terms, X and Y are set variables, are **atomic formulas**.

Inductive Step. Suppose that the formulas Φ_1 and Φ_2 have been defined. Then all of the following expressions are formulas:

$$(\Phi_1 \& \Phi_2),\ (\Phi_1 \bigvee \Phi_2),\ (\Phi_1 \to \Phi_2),\ (\exists x \Phi_1),\ (\neg \Phi_1),\ (\exists X \Phi_1),\ (\forall x \Phi_1),\ (\forall X \Phi_1),$$

where x is an individual variable, and X is a set variable.

We give several examples to clarify the intended meaning of formulas.

Example 3.10.2 *The formula $X \subseteq Y$ is defined as*

$$\forall z(z \in X \to z \in Y).$$

Thus, when X and Y are thought to be A and B respectively, with $A, B \subseteq \omega$, then the formula $X \subset Y$ is satisfied under this interpretation of X and Y if and only if A is indeed a subset of B.

Example 3.10.3 *Consider the formula $Sub(X)$:*

$$\forall x(x \in X \to S(x) \in X).$$

Suppose that the interpretation of X is the set A_i:

$$\{i, i+1, i+2, \ldots\}.$$

Then certainly, for any number n if $n \in A_i$, then $n+1 \in A_i$. Hence one can say that A_i satisfies the formula $Sub(X)$ when X is interpreted as A_i. It is not hard to see that a subset B of natural numbers satisfies the formula $Sub(X)$ if and only if B coincides with A_i for some i or $B = \emptyset$.

Example 3.10.4 *We define the following formula denoted by $x \leq y$:*

$$\forall X(Sub(X) \,\&\, x \in X \to y \in X).$$

It is not hard to see that when we interpret x and y as numbers i and j, respectively, under this interpretation of x and y, the formula $x \leq y$ is true if and only if i is less than or equal to j.

Exercise 3.10.2 *Prove the statement of the example given above.*

We let the expression $X \subset Y$ stand for $X \subseteq Y \& X \neq Y$.

Example 3.10.5 *The formula $Fin(X)$ is defined as*

$$\exists Z(X \subset Z \,\&\, \exists y(y \notin Z) \,\&\, \forall x(x \notin Z \to S(x) \notin Z)).$$

This formula defines finite subsets of ω. Indeed, suppose that A is a finite subset. Let n be the maximal element of the set A. Then if we put Z to be $\{0, \ldots, n\}$, then $A \subset Z$, $n+1 \notin Z$ and for all numbers i if $i \notin Z$, then clearly $i+1 \notin Z$. Moreover, it is not hard to see that if, under the interpretation $X \to A$, the formula is satisfied, then A is finite.

Exercise 3.10.3 *Consider the formula*

$$\forall y((y \in X \to S(y) \notin X) \,\&\, (y \notin X \to S(y) \in X)).$$

What type of subsets X of natural numbers satisfy the formula?

3.10.3 Satisfiability. The examples given at the end of the previous subsection naturally lead us to a formalization of the notion of satisfiability in the structure (ω, S). Suppose that to each individual variable x_i and a set variable X_j an element $b_i \in L$ and a set B_i has been assigned, respectively. In other words, we have a fixed **interpretation**

$$intr : x_i \to b_i \quad intr : X_i \to B_i$$

of the variables. By induction on the complexity of the formula Φ, we define **satisfiability of Φ on $(\omega, S, \leq)$ under the interpretation.** We need one explanation. Consider the term $S^n(x)$. When $n = 0$, $S^0(x) = x$. If we think of x as a variable that runs over the set of natural numbers, then $S^0(x)$ can be thought of as the function mapping each x to itself. Suppose that $n = 1$. Then $S^1(x)$ is simply $S(x)$. Therefore, if we think of x as a variable that runs over the set of natural numbers, then $S^1(x)$ can be thought as a function that maps each x to its immediate successor $x + 1$. Similarly, the term $S^2(x)$ simply represents the function on ω that maps each x to $x + 2$. In general the term $S^m(x)$ represents the function on ω that maps each x to $x + m$. Now we are ready to explain the notion of satisfaction that relates formulas with their meaning in the successor structure (ω, S).

Basic Step.

- The structure (ω, S) **satisfies** the atomic formula $(S^m(x_i) \in X_j)$ if in the structure (ω, S) we indeed have $S^m(b_i) \in B_j$.
- The structure (ω, S) **satisfies** the atomic formula $S^n(x_i) = S^m(x_j)$ if $S^n(b_i) = S^m(b_j)$.
- The structure (ω, S) **satisfies** the atomic formula $(X_i = X_j)$ if $B_i = B_j$.

Inductive Step. We consider separately each case corresponding to the inductive step in the definition of formula.

1. (ω, S) **satisfies** $(\Phi_1 \bigvee \Phi_2)$ if (ω, S) satisfies either the formula Φ_1 or the formula Φ_2.
2. (ω, S) **satisfies** $(\Phi_1 \& \Phi_2)$ if (ω, S) satisfies both the formula Φ_1 and the formula Φ_2.
3. (ω, S) **satisfies** $(\Phi_1 \to \Phi_2)$ if the condition that (ω, S) satisfies Φ_1 implies that (ω, S) satisfies Φ_2.
4. (ω, S) **satisfies** $\neg(\Phi_1)$ if (ω, S) does not satisfy Φ_1.
5. (ω, S) **satisfies** $(\exists x_i \Phi_1)$ if there exists a $b \in \omega$ such that (ω, S) satisfies the formula Φ_1 under the new interpretation *intr'* which is the same as *intr* except its value on x_i is b.
6. (ω, S) **satisfies** $(\exists X_i \Phi_1)$ if there exists $B \subset \omega$ such that (ω, S) satisfies the formula Φ_1 under the new interpretation *intr'* which is the same as *intr* except its value on X_i is B.
7. (ω, S) satisfies $(\forall x_i \Phi_1)((\forall X_i \Phi_1))$ if and only if (ω, S) satisfies $\neg(\exists x_i(\neg\Phi_1))$ $(\neg(\exists X_i(\neg\Phi_1)))$.

For a formula Φ whose variables are among $x_1, \ldots, x_t$ and $X_1, \ldots, X_p$, we usually write $\Phi(X_1, \ldots, X_p, x_1, \ldots, x_t)$. We also write

$$(\omega, S) \models \Phi(B_1, \ldots, B_p, b_1, \ldots, b_t)$$

if the structure (ω, S) satisfies this formula under the interpretation *intr* such that $intr(x_i) = b_i$, $intr(X_j) = B_j$, where $i = 1, \ldots, t$, and $j = 1, \ldots, p$. We now

suggest that the reader go through the examples of the previous subsection to better understand the definition of satisfiability.

We now need a definition.

Definition 3.10.1 *We say that the formulas $\Phi(X_1, \ldots, X_m, y_1, \ldots, y_k)$ and $\Psi(X_1, \ldots, X_m, y_1, \ldots, y_k)$ are* **equivalent** *if for all subsets $B_1, \ldots, B_m \subset \omega$ and all elements $b_1, \ldots, b_k \in \omega$ the structure (ω, S) satisfies the formula $\Phi(B_1, \ldots, B_m, b_1, \ldots, b_k)$ if and only if (ω, S) satisfies $\Psi(B_1, \ldots, B_m, b_1, \ldots, b_k)$.*

The next exercise gives some examples of equivalent formulas.

Exercise 3.10.4

1. *Show that the formula $S^n(x) \in X$ is equivalent to the formula*

$$\exists x_1 \ldots \exists x_{n-1}(S(x) = x_1 \; \& \; S(x_1) = x_2 \; \& \; \ldots \; \& \; S(x_{n-1}) \in X).$$

2. *Show that the formula $S^n(x) = S^m(y)$ is equivalent to the formula*

$$\exists x_1 \ldots \exists x_n \exists y_1 \ldots \exists y_m (S(x) = x_1 \; \& \; S(x_1) = x_2 \; \& \; \ldots \; \& \; S(x_{n-1}) = y_m \; \& \; S(y) = y_1 \; \& \; S(y_1) = y_2 \; \& \; \ldots \; \& \; S(y_{m-1}) = y_m).$$

To investigate the connections between Büchi automata and the monadic second order logic of the successor structure (ω, S), we need a technical notion of **free variable**. Basically, we just repeat the appropriate notion from the section on the monadic second order logic of finite strings.

Basic Step. For atomic formulas $(S^n(x) \in X)$, $x = y$, $X = Y$, and $S^n(x) = S^m(y)$, the set of free variables are $\{x, X\}$, $\{x, y\}$, $\{X, Y\}$ and $\{x, y\}$, respectively.

Inductive Step. Suppose that the set of free variables $FV(\Phi_1)$ and $FV(\Phi_2)$ for the formulas Φ_1 and Φ_2 have been defined. Then

$$FV(\Phi_1 \tau \Phi_2) = FV(\Phi_1) \bigcup FV(\Phi_2), \quad FV(\neg\Phi_1) = FV(\Phi_1),$$
$$FV(Qx\Phi_1) = FV(\Phi_1) \setminus \{x\}, \quad FV(QX\Phi_1) = FV(\Phi_1) \setminus \{X\},$$

where $\tau \in \{\&, \bigvee, \to\}$, $Q \in \{\forall, \exists\}$, x is an individual variable, and X is a set variable.

A formula is a **sentence** if it has no free variables. Thus by the definition of satisfiability, in the structure (ω, S) any sentence Φ is either true or false. In other words, the truth value (satisfiability) of Φ in (ω, S) does not depend on any interpretation of variables. Here now is a central definition of this chapter.

Definition 3.10.2 *The* **monadic second order theory of the successor structure** *(ω, S) is the set of all sentences satisfiable (true) in the structure (ω, S). We denote this set of sentences by $S1S$.*

Our purpose is to show that the monadic second order theory of (ω, S) is decidable. In other words we will show that there exists an effective procedure which, for any given monadic second order sentence Φ, decides whether or not this sentence is true in the structure (ω, S).

3.10.4 From Büchi Automata to Formulas. Let $\mathcal{A} = (S, \{s_0\}, T, F)$ be a Büchi automaton over the alphabet Σ. We can suppose that $\Sigma = \{0, 1\}^m$ for some $m \in \omega$ as explained in Section 3.10.1. Let $B_1, \ldots, B_{m-k}$ be subsets of ω, and $b_{m-k+1}, \ldots, b_m$ be natural numbers. We sometimes denote the sequence

$$B_1, \ldots, B_{m-k}, b_{m-k+1}, \ldots, b_m$$

by $\bar{B}\bar{b}$. Each such sequence and natural number $i \in \omega$ define a symbol σ_i of the alphabet Σ:

$$(\epsilon_1, \ldots, \epsilon_m),$$

where for each j, $\epsilon_i = 1$ if and only if $i \in B_j$ or $i = b_j$. Therefore, the sequence $\bar{B}\bar{b}$ defines the ω-string

$$\sigma_0\sigma_1\sigma_2\sigma_3\ldots$$

of the alphabet $\Sigma = \{0, 1\}^m$. We denote this infinite string by

$$\mathbf{word}(\bar{B}, \bar{b}).$$

The opposite is also true: any sequence $\sigma_0\sigma_1\ldots\sigma_n\ldots$ from Σ^ω defines an m-tuple of sets $(X_1, \ldots, X_m)$, where for each i, $1 \leq i \leq m$, and each $n \in \omega$, the number n is in X_i (not in X_i) if and only if the ith component of σ_n is 1 (0). So, we see that there is a natural one-to-one correspondence between all infinite strings from Σ^ω and m-tuples of subsets of ω. This correspondence is clearly seen if the ith row in the infinite strings

$$\mathbf{word}(\bar{B}, \bar{b})$$

is identified with the characteristic function of the set B_i ($\{b_i\}$).

The following theorem shows that the notion of acceptance for Büchi automata is expressible in the monadic second order formalism.

Theorem 3.10.1 *There exists an algorithm which, given a Büchi automaton $\mathcal{A}$ over $\Sigma = \{0, 1\}^m$, constructs a formula $\Phi(X_1, \ldots, X_m)$, where $X_1, \ldots, X_m$ are set variables, such that for all subsets $\bar{B} = B_1, \ldots, B_m \subset \omega$ the following property holds:*

$$\mathcal{A} \text{ accepts } \mathbf{word}(\bar{B}) \text{ if and only if } (\omega, S) \text{ satisfies } \Phi(\bar{B}).$$

Proof. The proof is very similar to (even simpler than) the proof in which we transformed finite automata into monadic second order formulas. Let $\mathcal{A} =$

(S, I, T, F) be a given Büchi automaton. We can assume that the initial state I contains one state only. Since S is a finite set we can also suppose that

$$S = \{0, 1, \ldots, p\}$$

with 0 being the initial state. Let $\sigma \in \Sigma$ be of the form

$$\begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_m \end{pmatrix},$$

where each $\epsilon_i \in \{0, 1\}$. We define the formula $\Phi_\sigma(x, \bar{X})$, where $\bar{X}$ is $X_1, \ldots, X_m$, as follows:

$$\Phi_\sigma(x, X_1, \ldots, X_m) = x \in^{\epsilon_1} X_1 \;\&\; \ldots \;\&\; x \in^{\epsilon_m} X_m,$$

where $\in^\epsilon$ is $\in$ if $\epsilon = 1$, and $\notin$ otherwise.

Informally, this formula codes the letter $\sigma \in \Sigma$. We note the following fact. Let $B_1, \ldots, B_m$ be subsets of ω. Then for every $n \in \omega$ there must exist a $\sigma \in \Sigma$ such that $\Phi_\sigma(n, \bar{B})$ holds. It follows that the formula

$$\forall x \left(\bigvee_{\sigma \in \Sigma} \Phi_\sigma(x, \bar{B}) \right)$$

is satisfied in (ω, S). We also recall that for each ω-word

$$\alpha = \sigma_0 \ldots \sigma_t \ldots \in \Sigma^\omega$$

there exists a sequence $\bar{B} = B_1, \ldots, B_m$ of subsets of ω such that the infinite string α equals the string $\mathbf{word}(B_1, \ldots, B_m)$.

Let $Y_0, Y_1, \ldots, Y_p$ be set variables. The desired formula involves these set variables. The intention is that each Y_i represents the state i of the automaton $\mathcal{A}$. In the desired formula these set variables will be existentially quantified. We write down the following formula denoted by $D(\bar{Y})$, where $\bar{Y}$ is $Y_0, \ldots, Y_p$:

$$\&_{i \neq j}^{p} \neg\exists z (z \in Y_i \;\&\; z \in Y_j).$$

The intention is that we want to express the fact that the automaton $\mathcal{A}$ cannot be in two distinct states simultaneously.

We define another formula $\Phi_{\sigma,i}(x, \bar{X}, \bar{Y})$ and explain its meaning. The formula is

$$x \in Y_i \;\&\; \Phi_\sigma(x, \bar{X}) \rightarrow \bigvee_{(i,\sigma,j) \in T} (S(x) \in Y_j).$$

Informally, the formula corresponds to the fact that if $\mathcal{A}$ is in state i, the next input signal is σ, then the next state of the automaton is a j such that $(i, \sigma, j) \in T$. The following note concerns this formula. Suppose that $n \in \omega$ and $B_1, \ldots, B_m \subset \omega$. Let $\sigma \in \Sigma$ be such that (ω, S) satisfies $\Phi_\sigma(n, \bar{B}, \bar{Y})$. Then if the formula

$$\exists Y_0 \ldots \exists Y_p(\Phi_{\sigma,i}(n, \bar{B}, \bar{Y}))$$

were true, there would exist a j such that $S(n)$ belongs to Y_j and (i, σ, j) belongs to T.

Finally, the last formula we need is the formula $F(\bar{Y})$ defined as

$$\forall x \exists y (x \leq y \;\&\; x \neq y \;\&\; \left(\bigvee_{j \in F} y \in Y_j)\right).$$

We recall from Example 3.10.4 that the ordering $\leq$ of natural numbers is defined in the monadic second order logic. The formula $F(\bar{Y})$ is intended to code the notion of successful computation.

Now we write down the desired formula $\Phi(X_1, \ldots, X_m)$:

$$\exists Y_0 \ldots Y_p(\forall x(D(\bar{Y}) \;\&\; 0 \in Y_0 \;\&\; \&_{i \in S, \sigma \in \Sigma} \Phi_{\sigma,i}(x, \bar{X}, \bar{Y}) \& F(\bar{Y})).$$

Note that this formula uses the symbol 0 that represents the smallest natural number. This number satisfies the formula $\forall y(x \leq y)$, and therefore the expression $\Phi(X_1, \ldots, X_m)$ is a formula of the second order monadic logic. Now our goal is to show that the formula $\Phi(X_1, \ldots, X_m)$ satisfies the theorem. We need to prove that for all subsets $B_1, \ldots, B_m \subset \omega$ the automaton $\mathcal{A}$ accepts **word**$(\mathcal{B}_1, \ldots, B_m)$ if and only if (ω, S) satisfies $\Phi(B_1, \ldots, B_m)$.

Take subsets $B_1, \ldots, B_m$. Consider the ω-string

$$\mathbf{word}(B_1, \ldots, B_m),$$

which is

$$\sigma_0\sigma_1 \ldots \sigma_t \ldots.$$

Now let us assume that this string is an input to the automaton $\mathcal{A}$.

Suppose that $\mathcal{A}$ accepts the string **word**$(B_1, \ldots, B_m)$. Therefore, there exists a run

$$\mathbf{r} = s_0 s_1 \ldots s_{t-1} s_t s_{t+1} \ldots$$

on the string

$$\sigma_0\sigma_1 \ldots \sigma_t \ldots$$

such that $In(\mathbf{r}) \bigcap F \neq \emptyset$. We need to prove that

$$(\omega, S) \models \Phi(B_1, \ldots, B_m),$$

that is, we need to show that there are subsets $D_0, \ldots, D_p$ of the domain ω such that the interpretation $Y_i \to D_i, i = 0, \ldots, p$, makes the formula $\Phi(B_1, \ldots, B_m)$ true in ω.

Here is how we define each D_i, $i = 0, \ldots, p$:

$$D_i = \{j \in \omega \mid s_j = i\}.$$

In other words, we look at the jth state in the computation

$$\mathbf{r} = s_0 s_1 \ldots s_t s_{t+1} \ldots,$$

compare it with i, and put the number j into D_i if and only if the state s_j coincides with the state i. The formula $\Phi(B_1, \ldots, B_m)$ under this interpretation of $Y_i \to D_i$, $i = 0, \ldots, p$, is satisfied by (ω, S).

Now suppose that $\Phi(B_1, \ldots, B_m)$ is satisfied in (ω, S). We want to show that $\mathcal{A}$ accepts the string **word**$(B_1, \ldots, B_m)$. There are subsets $D_0, \ldots, D_p$ of ω such that the formula $\Phi(B_1, \ldots, B_m)$ becomes satisfied by (ω, S) when we interpret the variables $Y_0, \ldots, Y_p$ as $D_0, \ldots, D_p$, respectively.

For every $j \in \omega$, we consider the set D_{s_j} such that $j \in D_{s_j}$. Then using the definition of $\Phi(\bar{X})$, one can check that the sequence

$$s_0, s_1, \ldots, s_t, s_{t+1}, \ldots$$

is a computation of $\mathcal{A}$ on the string **word**$(B_1, \ldots, B_m)$ such that $In(\mathbf{c}) \bigcap F \neq \emptyset$. This proves the theorem.

Exercise 3.10.5 *Show that for every Büchi automaton $\mathcal{A}$ over the alphabet $\{0, 1\}^m$ there exists a formula $\Phi(X_1, \ldots, X_m)$ such that the following properties hold:*

1. *$\mathcal{A}$ accepts* **word**$(\bar{B})$ *if and only if (ω, S) satisfies $\Phi(\bar{B})$.*

2. *The formula $\Phi(X_1, \ldots, X_m)$ is of the form $\exists \bar{Y} \forall \bar{Z} \Psi(\bar{Z}, \bar{Y}, X_1, \ldots, X_m)$, where Ψ is a quantifier free formula.*

In relation to the theorem proved we would like to make the following comment. One might say that Büchi automata correspond to the class of nonterminating computer systems (e.g., operating systems, air traffic control systems, banking systems), and the monadic second order formulas represent logical specifications of the systems. Therefore, in this terminology the theorem above states that there exists a transformation from nonterminating operating systems to their logic specifications. The fundamental property of these specifications is in their *completeness*. In other words, one can provide a procedure which transforms each logic specification (formula) to a nonterminating computer system, that is to an appropriate Büchi automaton. We will prove this in the next section.

3.10.5 From Formulas to Büchi Automata. Our goal is to show the converse of the theorem proved in the previous subsection. In other words, given a formula of the second order monadic logic, we want to construct a Büchi automaton whose successful runs correspond to the interpretations of the variables that make the formula satisfied. We first consider several examples to explain the idea.

Example 3.10.6 *Consider the formula $x_2 \in X_1$. We define a Büchi automaton $\mathcal{A}$ whose transition table is presented in Figure* 3.19.

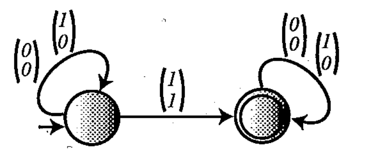

Figure 3.19: Automaton representing the formula $x_2 \in X_1$.

The connection between the automaton and the formula $x_2 \in X_1$ is the following. For all subsets $B_1 \subset \omega$ and numbers $b_2 \in \omega$, the automaton $\mathcal{A}$ accepts the string $\mathbf{word}(B_1, b_2)$ if and only if $b_2 \in B_1$. Indeed, first of all note that the first row in the string $\mathbf{word}(B_1, b_2)$ corresponds to the characteristic function of the set B_1. There exists only one symbol 1 in the second row of this string at the position b. Thus $b_2 \in B_1$ is detected if and only if the automaton enters the final state, and stays in the final state forever without failing to read the next inputs.

Example 3.10.7 *Consider the atomic formula $S(x_1) = x_2$. We define an automaton $\mathcal{A}$ whose graph presentation is given in Figure* 3.20.

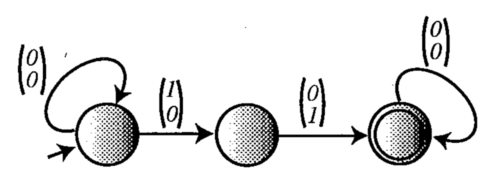

Figure 3.20: Automaton representing the formula $x_2 \in X_1$.

The connection between this automaton and the formula $S(x_1) = x_2$ is as in the previous example, that is for all $b_1, b_2 \in \omega$ the automaton $\mathcal{A}$ accepts the string $\mathbf{word}(b_1, b_2)$ if and only if (ω, S) satisfies $S(b_1) = b_2$. Indeed, take two numbers b_1 and b_2. Consider the infinite string $\mathbf{word}(b_1, b_2)$. The first row of this string contains only one symbol 1 at the position b_1. The second row of this string contains only one symbol 1 at the position b_2. Thus, the fact that $S(b_1) = b_2$ is detected if and only if the automaton enters the final state, and stays in the final state forever without failing to read the next inputs.

Now we give a definition. However, we should make some notations. From now on, $\bar{X}$ and $\bar{x}$ denote the sequence $X_1, \ldots, X_{m-k}$ of set variables and the sequence $x_{m-k+1}, \ldots, x_m$ of individual variables, respectively. We use similar notations for sequences of sets and elements in ω. For example $\bar{B} \subset A$ simply means that $B_1, \ldots, B_{m-k}$ is a sequence of subsets of A.

Definition 3.10.3 *Let $\Phi(\bar{X}, \bar{x})$ be a formula and let $\mathcal{A}$ be a Büchi automaton over $\{0, 1\}^m$. We say that the* **automaton $\mathcal{A}$ represents the formula** $\Phi(\bar{X}, \bar{x})$, *or equivalently the* **formula** $\Phi(\bar{X}, \bar{x})$ **represents the automaton** $\mathcal{A}$, *if for all subsets $\bar{B} \subset \omega$ and elements $\bar{b} \in \omega$, the automaton $\mathcal{A}$ accepts* $\mathbf{word}(\bar{B}, \bar{b})$ *if and only if (ω, S) satisfies $\Phi(\bar{B}, \bar{b})$.*

The two examples above show that the formulas $x_2 \in X_1$ and $S(x_1) = x_2$ are represented by automata. We also note that by Theorem 3.10.1 every Büchi automaton is represented by a formula. Our goal is to show that for any formula there exists a Büchi automaton that represents the formula.

Exercise 3.10.6 *Show that the formulas $Sub(X)$,*

$$\forall x(x \in X \to S(x) \in X),$$

and $x \le y$,

$$\forall X(Sub(X) \;\&\; x \in X \to y \in X),$$

are representable by Büchi automata.

We now prove the following lemma.

Lemma 3.10.1 *Suppose that Büchi automata $\mathcal{A}_1$ and $\mathcal{A}_2$ represent the formulas $\Phi_1(\bar{X}, \bar{x})$ and $\Phi_2(\bar{X}, \bar{x})$, respectively. Then there exist Büchi automata representing*

$$\Phi_1(\bar{X}, \bar{x}) \tau \Phi_2(\bar{X}, \bar{x}),\ \neg\Phi_1(\bar{X}, \bar{x}),\ Q_i X_i \Phi_1(\bar{X}, \bar{x}),\ \text{and}\ Q_j x_j \Phi_1(\bar{X}, \bar{x}),$$

where $\tau \in \{\bigvee, \&, \to\}$, $Q \in \{\forall, \exists\}$, $i = 1, \ldots, m-k$, $j = m-k+1, \ldots, m$.

Proof. Given Büchi automata $\mathcal{A}_1$ and $\mathcal{A}_2$, we can consider the automata $\mathcal{A}_1 \oplus \mathcal{A}_2$, $\mathcal{A}_1 \times \mathcal{A}_2$, and $\mathcal{A}_1^c$ that accept the union $L(\mathcal{A}_1) \bigcup L(\mathcal{A}_2)$, the intersection $L(\mathcal{A}_1) \bigcap L(\mathcal{A}_2)$, and the complement of $L(\mathcal{A}_1)$, respectively. Using the assumption that for each $i = 1, 2$ the automaton $\mathcal{A}_i$ represents $\Phi_i(\bar{X}, \bar{x})$, and using that Büchi recognizable languages are closed under union, intersection, and complementation, we see that the automata

$$\mathcal{A}_1 \oplus \mathcal{A}_2,\ \mathcal{A}_1 \times \mathcal{A}_2,\ \mathcal{A}_1^c$$

represent the formulas

$$\left(\Phi_1(\bar{X}, \bar{x}) \bigvee \Phi_2(\bar{X}, \bar{x})\right),\ \Phi_1(\bar{X}, \bar{x}) \& \Phi_2(\bar{X}, \bar{x}),\ \neg\Phi_1(\bar{X}, \bar{x}),$$

respectively. We give, for example, a proof that $\mathcal{A}_1 \oplus \mathcal{A}_2$ represents $(\Phi_1(\bar{X}, \bar{x}) \bigvee \Phi_2(\bar{X}, \bar{x}))$. Take subsets $\bar{B} \subset \omega$, and elements $\bar{b} \in \omega$. Consider the ω-string $\mathbf{word}(\bar{B}, \bar{b})$. Then, since each $\mathcal{A}_i$ represents $\Phi_i(\bar{X}, \bar{x})$, we see that (ω, S) satisfies $(\Phi_1(\bar{B}, \bar{b}) \bigvee \Phi_2(\bar{B}, \bar{b}))$ if and only if (ω, S) satisfies either $\Phi_1(\bar{B}, \bar{b})$ or $\Phi_2(\bar{B}, \bar{b})$. Thus, by hypothesis of the lemma, (ω, S) satisfies $(\Phi_1(\bar{B}, \bar{b}) \bigvee \Phi_2(\bar{B}, \bar{b}))$ if and only if either $\mathcal{A}_1$ accepts $\mathbf{word}(\bar{B}, \bar{b})$ or $\mathcal{A}_2$ accepts $\mathbf{word}(\bar{B}, \bar{b})$. Hence, (ω, S) satisfies $(\Phi_1(\bar{B}, \bar{b}) \bigvee \Phi_2(\bar{B}, \bar{b}))$ if and only if $\mathcal{A}_1 \oplus \mathcal{A}_2$ accepts $\mathbf{word}(\bar{B}, \bar{b})$.

Now suppose that $\Phi(\bar{X}, \bar{x})$ is $\exists X_i \Phi_1(\bar{X}, \bar{x})$. Without loss of generality, we can assume that $X_i = X_1$. Consider the Büchi automaton $\mathcal{A}_1$ over the alphabet $\Sigma = \{0, 1\}^m$ which represents the formula $\Phi_1(\bar{X}, \bar{x})$. The alphabet Σ is $\Sigma_1 \times \Sigma_2$, where $\Sigma_1 = \{0, 1\}$ and $\Sigma_2 = \{0, 1\}^{m-1}$. By the projection theorem about Büchi recognizable languages (see Theorem 3.2.3), there exists an automaton $\mathcal{B}$ which accepts the projection $pr_2(L(\mathcal{A}_1))$ of the set $L(\mathcal{A}_1)$. We show that $\mathcal{B}$ represents $\exists X_1 \Phi_1(\bar{X}, \bar{x})$. Indeed, take subsets $B_2, \ldots, B_{m-k} \subset \omega$, and elements $\bar{b} \in \omega$. Consider the ω-string $\mathbf{word}(B_2, \ldots, B_{m-k}, \bar{b})$. Then $\mathcal{B}$ accepts the string $\mathbf{word}(B_2, \ldots, B_{m-k}, \bar{b})$ if and only if there is a string $u \in (\{0, 1\})^{\omega}$ such that the string $(u, \mathbf{word}(B_2, \ldots, B_{m-k}, \bar{b})) \in (\{0, 1\}^m)^{\omega}$ belongs to $L(\mathcal{A}_1)$. Therefore, the automaton $\mathcal{B}$ accepts the string $\mathbf{word}(B_2, \ldots, B_{m-k}, \bar{b})$ if and only if there is a $B_1 \subset \omega$ such that the automaton $\mathcal{A}$ accepts the string $\mathbf{word}(B_1, B_2, \ldots, B_{m-k}, \bar{b})$. Thus we conclude that (ω, S) satisfies $\Phi_1(\bar{B}, \bar{b})$ if and only if (ω, S) satisfies $\exists X_1 \Phi_1(X_1, B_2, \ldots, B_{m-k}, \bar{b})$.

Similarly, one can prove that there exists an automaton that represents the formulas $Qx_j \Phi_1(\bar{B}, b_{m-k+1}, \ldots, x_j, \ldots, b_m)$ with $Q \in \{\forall, \exists\}$, and the formula $\forall X_j \Phi_1(\bar{B}, b_{m-k+1}, \ldots, x_j, \ldots, b_m)$. The lemma is proved.

Exercise 3.10.7 *Give a construction of an automaton that represents the formula* $\forall X_1 \Phi_1(X_1, \ldots, X_m)$ *if* $\Phi_1(X_1, \ldots, X_m)$ *is represented by a Büchi automaton* $\mathcal{A}$.

Lemma 3.10.2 *Let* Φ *be an atomic formula. There exists an automaton which represents* Φ.

Proof. The first two examples at the beginning of this subsection show that the lemma is true for the atomic formulas $x_2 \in X_1$ and $S(x_1) = x_2$. The atomic formula $x = y$ is represented by the automaton whose transition table is in Figure 3.21.

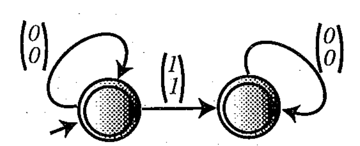

Figure 3.21: Automaton representing the formula $x = y$.

The atomic formula $X = Y$ is represented by the automaton whose graph is given in Figure 3.22.

We ask the reader to prove that the formula $S^n(x) = S^m(y)$ is presented by a Büchi automaton (see exercise below). The lemma is proved.

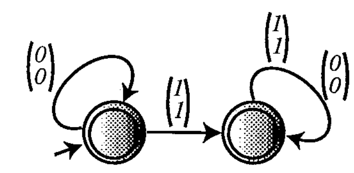

Figure 3.22: Automaton representing the formula $X = Y$.

Exercise 3.10.8 *Prove that there exists a Büchi automaton that represents the formula* $S^n(x) = S^m(y)$.

From the above lemmas we have the following theorem.

Theorem 3.10.2 *There exists a procedure which for every monadic second order formula* $\Phi(\bar{X}, \bar{x})$ *constructs a Büchi automaton* $\mathcal{A}$ *representing the formula.*

Proof. We first note that the operations of producing Büchi automata that accept the union, intersection, complements, and projections of Büchi recognizable languages can be carried out effectively. Now the proof of the theorem is by induction on the complexity of the formula $\Phi(\bar{X}, \bar{x})$. Lemma 3.10.2 proves the theorem for atomic formulas, that is for the basic step of our induction. Lemma 3.10.1 proves the inductive step and hence the theorem.

3.10.6 Decidability and Definability in $S1S$. As a corollary of the previous subsections, we get the following known theorem first proved by Büchi.

Theorem 3.10.3 *The monadic second order theory* $S1S$ *of the successor structure* (ω, S) *is decidable.*

Proof. Let Ψ be a sentence. We want to check whether or not (ω, S) satisfies Ψ. We can assume that Ψ is equivalent to a formula either of the form $\exists x \Phi(x)$ or of the form $\exists X \Phi(X)$. Consider the case when the sentence Ψ has the form $\exists X \Phi(X)$. The case when Ψ is of the form $\exists x \Phi(x)$ is treated similarly. There exists an automaton $\mathcal{A}$ representing $\Phi(X)$. We conclude that for any subset $B_1 \subset \omega$, the structure (ω, S) satisfies $\Phi(B_1)$ if and only if the Büchi automaton $\mathcal{A}$ accepts the string $\mathbf{word}(B_1)$. Thus, there exists a $B_1 \subset \omega$ such that (ω, S) satisfies $\Phi(B_1)$ if and only if there exists a string of the form $\mathbf{word}(B_1)$ accepted by $\mathcal{A}$. It follows that $\exists X \Phi(X)$ holds in (ω, S) if and only if $L(\mathcal{A}) \neq \emptyset$. Thus we have reduced the satisfiability problem for $\exists X \Phi(X)$ to the emptiness problem for a Büchi automaton representing $\Phi(X)$. Since the emptiness problem for Büchi automata is decidable, we see that $S1S$ is also decidable. The theorem is proved.

It turns out that the results proved above provide not only decidability of $S1S$ but also talk about definability issues. Consider the set $P(\omega)$ of all subsets of ω.

Definition 3.10.4 *We say that a subset $M \subset P(\omega)^m$ is* **definable** *if there exists a formula $\Phi(X_1, \dots, X_m)$ such that*

$$M = \{(B_1, \dots, B_m) \mid (\omega, S) \models \Phi(B_1, \dots, B_m)\}.$$

Clearly, every definable set M naturally defines an ω-language L_M as

$$L_M = \{\mathbf{word}(B_1, \dots, B_m) \mid (B_1, \dots, B_m) \in M\}.$$

We say that $L \subset \Sigma^\omega$ is **definable** if $L = L_M$ for some definable set $M \subset P(\omega)^m$. From the definitions and the theorems proved in the previous two sections we have the following definability result.

Theorem 3.10.4 *For any ω-language L over the alphabet Σ the following conditions are equivalent:*

1. *L is definable.*
2. *There exists a Büchi automaton $\mathcal{A}$ such that $L = L(\mathcal{A})$.* □

Exercise 3.10.9 *Check the correctness of the theorem above.*

3.11 An Application of the McNaughton Theorem

From the previous section we see that Büchi recognizability is equivalent to definability in monadic second order logic in a certain precise sense. A natural and somewhat informal question which arises is the following:

> *What is the relationship between Büchi recognizability and definability if we weaken the monadic second order logic?*

In this section we give one possible answer to this question. We recall that in defining the monadic second order logic of (ω, S) we used variables

$$X, Y, Z, \dots$$

which are set variables. The intention for these variables is that they run over subsets of ω. We now introduce a weaker fragment of the monadic second order logic of (ω, S).

The logic is defined in exactly the same way as we defined the monadic second order logic of (ω, S) with only one exception. In this logic we again use variables

$$X, Y, Z, \dots,$$

possibly with subscripts or superscripts, with the intention that they run over *subsets* of ω. However, when we quantify these set variables, then we always assume that the set variables under the quantifiers represent *finite sets*. Thus, the formula $\exists X \Phi(X)$ is simply intended to say that there exists a finite subset B of natural numbers such that $\Phi(B)$. We call this logic *the weak monadic second order logic* of (ω, S). Another *important* point here is the following. Set variables X that occur *freely* in a given formula can have any interpretation as subsets, in particular infinite subsets, of ω. Let us consider several examples.

Example 3.11.1 *Consider the formula: $\forall z(z \in Y \to z \in X)$. We denote this formula by $Y \subseteq X$.*

This formula is a formula of the weak monadic second order logic. When X is interpreted as an $A \subseteq \omega$ and Y is interpreted as a $B \subseteq \omega$, the formula is true if and only if B is a subset of A. Note that we do not require A and B to be finite since X, Y occur free in the formula.

Example 3.11.2 *Consider the formula $\exists Y(Y \subseteq X)$.*

Note that in this formula Y is in the scope of the existential quantifier. This formula reads as follows. There exists a finite set Y which is a subset of the set (not necessarily finite) X. Note that the formula is true under all interpretations of X.

Example 3.11.3 *Consider the formula $\forall X \exists Y(Y \subseteq X)$.*

Both variables in this formula are not free. Hence this formula states that for all finite sets X there exists a finite set Y which is a subset of X. Note that this sentence is true in the structure (ω, S).

Exercise 3.11.1 *Consider the following formula of the weak monadic second order logic:*

$$\exists Y(Y \subset X \;\&\; Y \neq \emptyset \;\&\; \forall y_1 \forall y_2(S(y_1) = y_2 \;\&\; y_2 \in Y \to y_1 \in Y)).$$

What does this formula say? For which X subsets of ω, is this formula true in the structure (ω, S)?

Exercise 3.11.2 *Consider the formula*

$$\exists Y(Y \subset X \;\&\; Y \neq \emptyset \;\&\; \forall y_1 \forall y_2(S(y_1) = y_2 \;\&\; y_1 \in Y \to y_2 \in Y)).$$

Does there exist an X subset of ω which makes this formula true in the weak second order monadic logic? Does there exist an X subset of ω which makes this formula true in the second order monadic logic?

We now can define the set of all sentences of the weak monadic second order logic that are true in the structure (ω, S). Here is the definition.

Definition 3.11.1 *Denote $WS1S$ to be the set of all sentences in the weak monadic second order logic which are true in (ω, S).*

Having defined the weak second order monadic logic, it is now natural to ask how this logic is related to the monadic second order logic. To partially answer this question, we note that there exists a formula $Fin(X)$ considered in Example 3.10.5 of the monadic second order logic that is true if and only if X is a finite set of natural numbers. Therefore, any formula Φ of the weak monadic second order logic can be transformed into an equivalent formula Ψ of the second order monadic logic. In order to explain this we give an example. Consider the formula

$$\exists Y \forall X (\Phi(X, Y, Z_1, Z_2)).$$

in the weak monadic second order logic, where $\Phi(X, Y, Z_1, Z_2)$ contains no quantifiers. Then we can transform this formula into the following formula of the second order monadic logic:

$$\exists Y (Fin(Y) \ \& \ \forall X (Fin(X) \rightarrow \Phi(X, Y, Z_1, Z_2))),$$

Thus, for all interpretations of variables the first formula is true in (ω, S) if and only if the second formula is true in (ω, S). We leave it to the reader to prove the following general statement which in particular shows that $WS1S$ can be embedded into $S1S$.

Exercise 3.11.3 *Prove that for any formula Φ of the weak second order monadic logic there exists a formula Ψ of the second order monadic logic so that for all interpretations of variables the formula Φ is true in (ω, S) if and only if Ψ is true in (ω, S).*

Now our goal is to show that the power of the weak second order monadic logic coincides with the power of the monadic second order logic. In order to make this statement precise we give the following definition.

Definition 3.11.2 *We say that a subset $M \subset P(\omega)^m$ is* **weakly definable** *if there exists a formula $\Phi(X_1, \ldots, X_m)$ of the weak second order monadic logic such that*

$$M = \{(B_1, \ldots, B_m) \mid (\omega, S) \models \Phi(B_1, \ldots, B_m)\}.$$

Now our goal will be to show that any definable set (in the monadic second order logic, see Definition 3.10.4) is also weakly definable.

Clearly, every (weakly definable) set $M \subset P(\omega)^m$ naturally defines an ω-language L_M over the alphabet $\{0, 1\}^m$ as follows:

$$L_M = \{\mathbf{word}(B_1, \ldots, B_m) \mid (B_1, \ldots, B_m) \in M\}.$$

We now give the following definition that singles out those languages that correspond to relations definable in the weak monadic second order logic.

Definition 3.11.3 *An ω-language $L \subset \Sigma^\omega$ is* **weakly definable** *if $L = L_M$ for some weakly definable set $M \subset P(\omega)^m$.*

Here is the desired result that uses the McNaughton theorem. The result states that the definability in the weak second order monadic logic is equivalent to the definability in the second order monadic logic.

Theorem 3.11.1 *An ω-language L is definable if and only if L is weakly definable.*

Proof. We have already outlined the proof that weak definability implies definability. We refer the reader to Exercise 3.11.3, the example considered before the exercise, and the definitions of definable and weakly definable languages in order to formally prove this part of the theorem.

We now need to prove that any definable language is in fact weakly definable. So let L be a definable language. Then there exists a Büchi automaton that recognizes the language L. By the McNaughton theorem (see Theorem 3.8.1), the language L is Müller recognizable. By the characterization theorem of Müller recognizable languages (see Theorem 3.7.3), L is a finite Boolean combination of ω-languages of the type $\vec{W}$, where each W is FA recognizable. If we can prove that each $\vec{W}$ is weakly definable, then by taking the boolean combination of the formulas (in the second order weak monadic logic) that define the language $\vec{W}$, we see that L is weakly definable. Therefore, it suffices to prove the theorem when L is simply $\vec{W}$, where W is an FA recognizable language.

Thus, assume that $L = \vec{W}$, where W is an FA recognizable language. The idea is the following. For the language W, consider a formula $\Phi(X_1, \ldots, X_m)$ of *the monadic second order theory of finite strings* such that W is definable (in the monadic second order logic of strings) by formula $\Phi(X_1, \ldots, X_m)$. We recall that $\Phi(X_1, \ldots, X_m)$ defines W if for any string $w \in (\{0, 1\}^m)^\star$ there exists a chain C and subsets $B_1, \ldots, B_m$ of the chain such that $w = \mathbf{word}(C, B_1, \ldots, B_m)$, and $w \in W$ if and only if C satisfies $\Phi(B_1, \ldots, B_m)$. Then one can obtain a formula $\Psi(X_1, \ldots, X_m, y)$ of the weak monadic second order logic of (ω, S) which simulates $\Phi(X_1, \ldots, X_m)$ in the following sense. The formula $\Psi(X_1, \ldots, X_m, y)$ states that for any $\alpha \in \Sigma^\omega$ determined by the sets $X_1, \ldots, X_m$ (that is, $\alpha = \mathbf{word}(X_1, \ldots, X_m)$) the infinite string α up to position y satisfies $\Phi(X_1, \ldots, X_m)$. Below we implement this idea formally.

Let $\mathcal{A}$ be a deterministic *finite automaton* recognizing W. The goal is to write down the formula that can talk about runs of the automaton $\mathcal{A}$ on infinite inputs. So assume that the initial state of $\mathcal{A}$ is 0 and the set S of states is $\{0, \ldots, p\}$. We define the formula $\Phi_\sigma(x, \bar{X})$, where $\bar{X}$ is $X_1, \ldots, X_m$, as follows:

$$\Phi_\sigma(x, X_1, \ldots, X_m) = x \in^{\epsilon_1} X_1 \& \ldots x \in^{\epsilon_m} X_m,$$

where for $\in^\epsilon$ is $\in$ if $\epsilon = 1$, and $\notin$ otherwise. Informally, this formula codes the word $\sigma \in \{0, 1\}^m$. We note the following fact. Let $B_1, \ldots, B_m$ be subsets of ω. Then for every $n \in \omega$ there must exist a $\sigma \in \Sigma$ such that $\Phi_\sigma(n, \bar{B})$ holds. It follows that the formula

$$\forall x \left(\bigvee_{\sigma \in \Sigma} \Phi_\sigma(x, \bar{B}) \right)$$

is satisfied in (ω, S). We also recall that for each ω-word

$$\alpha = \sigma_0 \ldots \sigma_t \ldots \in \Sigma^\omega$$

there exists a sequence $\bar{B} = B_1, \ldots, B_m$ of subsets of ω such that the infinite string α equals the string **word**$(B_1, \ldots, B_m)$.

Let $Y_0, Y_1, \ldots, Y_p$ be set variables. The desired formula involves these set variables. The intention is that each Y_i represents the state i of the automaton $\mathcal{A}$. In the desired formula these set variables will be existentially quantified. Since we are dealing with the weak monadic second order logic, these variables are supposed to run over finite subsets of ω. We write down the following formula and denote it by $D(\bar{Y})$, where $\bar{Y}$ is $Y_0, \ldots, Y_p$:

$$\&_{i \neq j}^{p} \neg \exists z (z \in Y_i \& z \in Y_j).$$

This formula informally states that $\mathcal{A}$ cannot be in two distinct states at the same time.

We define another formula $\Psi_\sigma(x, \bar{X}, \bar{Y})$ as follows:

$$\forall x' \left[(x \leq x') \& \Phi_\sigma(x, X_1, \ldots, X_m) \rightarrow \bigvee_{(0,\sigma,j) \in T} (x \in Y_j) \right].$$

Informally this formula expresses the fact that if σ is the first input signal of an ω-input, then the next state of the automaton $\mathcal{A}$ is a j such that $(0, \sigma, j) \in T$. Since T is a deterministic transition, we note that the state j is unique. The following note concerns this formula. Suppose that $B_1, \ldots, B_m \subset \omega$. Let $\sigma \in \Sigma$ be such that (ω, S) satisfies $\Phi_\sigma(0, \bar{B})$. Then if the formula

$$\exists Y_0 \ldots \exists Y_p (\Psi_\sigma(0, \bar{B}, \bar{Y}))$$

were true (note that the variables $Y_0, \ldots, Y_p$ are existentially quantified and hence run over finite subsets of ω), then there would exist a unique j such that $S(0)$ belongs to Y_j and $(0, \sigma, j)$ belongs to T.

We define another formula $\Phi_{\sigma,i}(x, \bar{X}, \bar{Y}, y)$ and explain its meaning. The formula is

$$x \in Y_i \ \& \ \Phi_\sigma(S(x), \bar{X}) \ \& \ x < y \rightarrow \bigvee_{(i,\sigma,j) \in T} (S(x) \in Y_j).$$

Informally, this formula codes the transition table T of $\mathcal{A}$ before reaching the position y.

Finally, the last formula we need is $F(y, \bar{Y})$, defined as

$$\bigvee_{j \in F} (y \in Y_j).$$

This formula is intended to say that if $\mathcal{A}$ has completed its computation up to position y, then the state of the computation just before reading y is a final state.

Now we write down the desired formula $\Psi(X_1, \ldots, X_m)$ in the weak second order monadic logic of the structure (ω, S):

$$\forall x \exists y ((x < y) \ \& \ \exists \bar{Y} (D(\bar{Y}) \ \& \ \forall x_1 \forall x_2 (\&_{i \in S, \sigma \in \Sigma} (\Psi_\sigma(x_1, \bar{X}, \bar{Y}) \\ \& \ \Phi_{\sigma,i}(x_2, \bar{X}, \bar{Y}, y) \ \& \ F(y, \bar{Y})))).$$

Now the reader can check that the language $\vec{W}$ is definable by this formula of the weak monadic second logic. We note that the proof of this is quite similar to the proof of Theorem 3.10.1 (see the exercise below). Thus, we conclude that the language L is weakly definable. The theorem is proved.

Exercise 3.11.4 *Check that the formula $\Psi(X_1, \ldots, X_m)$ provided at the end of the proof of the theorem above is indeed the desired one.*