1 Logics, Structures and Presentations

In this chapter we review the standard notions of first-order and monadic second-order logic. We introduce linear orders and trees and state a few elementary properties of these structures. We recall the correspondence between monadic second-order logic and automata on infinite words together with basic facts from automata theory . We introduce automatic structures using presentations by automata and characterize them both by first-order and by monadic second-order to first-order interpretations. Finally, we discuss the composition method for monadic second-order logic over linear orders and trees.

As most of these notions are standard, we assume that the reader is already familiar with them and thus we do not discuss them in detail, but concentrate instead on fixing the notation. More thorough introductions to logic and automata can be found in many textbooks and surveys, e.g. in [27, 82, 84].

1.1 First-Order and Monadic Second-Order Logic

A structure $\mathfrak{A} = (A, R_1, \ldots, R_n, f_1, \ldots, f_m)$ is given by a set A, called the domain of $\mathfrak{A}$, a number of relations R_i and a number of functions f_j. If we denote the arity of R_i by r_i and the arity of f_j by s_j then $R_i \subseteq A^{r_i}$ and $f_j : A^{s_j} \to A$. We say that $\mathfrak{A}$ is a *relational structure* if it contains no functions, only relations. Every structure can be coded as a relational structure by replacing each function f_j by its graph G_{f_j}, which is a relation of arity $s_j + 1$ such that $G_{f_j}(\overline{x}, y) \iff f_j(\overline{x}) = y$. The signature of the structure $\mathfrak{A}$ is denoted by $\sigma(\mathfrak{A}) = \{R_i^{(r_i)}\} \cup \{f_j^{(s_j)}\}$, where R_i and f_j are now just symbols with appropriate arities. Note that we only consider structures with finite signatures in this thesis, even though some of the results do not rely on this assumption.

Both first-order and monadic second-order logic formulas over a signature τ are built from atomic formulas using Boolean connectives and quantifiers. Atomic formulas in first-order logic (FO) have either the form $t_1 = t_2$ or

$R_i(t_1, \ldots, t_{r_i})$, where t_k are terms, i.e. either first-order variables, denoted $x, y, x_1, y_1, \ldots$, or expressions of the form $f_j(t_1, \ldots, t_{s_j})$. In monadic second-order logic (MSO) there are additional atomic formulas of the form $t \in X$, where t is again a term and X is a second-order variable. We use the standard Boolean connectives $\wedge, \vee, \neg$ to denote conjunction, disjunction and negation and we write $\varphi \to \psi$ for $(\neg\varphi) \vee \psi$ and $\varphi \leftrightarrow \psi$ for $(\varphi \to \psi) \wedge (\psi \to \varphi)$. In first-order logic it is possible to quantify over first-order variables using either existential or universal quantifiers, i.e. to write $\exists x\varphi$ or $\forall x\varphi$. In second-order logic there is the additional possibility to quantify over second-order variables, i.e. to write $\exists X\varphi$ or $\forall X\varphi$. When writing formulas we use the convention that negation and quantifiers bind stronger than $\wedge$ and $\vee$, which in turn bind stronger than $\to$ and $\leftrightarrow$, so that $\forall x Rx \wedge \neg Ry \to Rz$ is to be understood as $((\forall x Rx) \wedge (\neg Ry)) \to Rz$. We say that a variable in a formula φ that appears in scope of a quantifier is *bound* and in the other case it is *free*. When writing $\varphi(x_1, \ldots, x_n)$ (or sometimes shorter $\varphi(\overline{x})$) we mean that all free variables of φ are contained in $\{x_1, \ldots, x_n\}$.

Whether a formula $\varphi(\overline{x}, \overline{X})$ holds in a structure $\mathfrak{A}$, given an assignment $\theta : \overline{x} \to A$ for first-order variables and an assignment $\Theta : \overline{X} \to \mathcal{P}(A)$ for second-order ones, denoted $\mathfrak{A}, \theta, \Theta \models \varphi$, is defined inductively. First, we extend the assignment θ to all terms by putting $\theta(f_j(t_1, \ldots, t_{s_j})) = f_j(\theta(t_1), \ldots, \theta(t_{s_j}))$. Note that f_j on the left side of the equation is only a symbol, while on the right side it is a function of $\mathfrak{A}$. We will often abuse the notation in this way and we sometimes write $f_i^{\mathfrak{A}}$ to point out that we have the function (or relation) in the structure in mind, rather than the symbol. Moreover, we extend this notation to formulas, so given a formula $\varphi(\overline{x})$ we write $\varphi^{\mathfrak{A}}$ for the relation defined by $\varphi^{\mathfrak{A}}(\overline{a}) \iff \mathfrak{A}, \overline{x} \leftarrow \overline{a} \models \varphi$. In the inductive definition of the semantics below we write $\theta[x \leftarrow a]$ (or analogous for Θ) for the assignment $\theta' : \overline{x} \cup \{x\} \to A$ that maps x to a and every other variable x' to $\theta(x')$.

- $\mathfrak{A}, \theta, \Theta \models t_1 = t_2$ whenever $\theta(t_1) = \theta(t_2)$,
- $\mathfrak{A}, \theta, \Theta \models R_i(t_1, \ldots, t_{r_i})$ whenever $R_i^{\mathfrak{A}}(\theta(t_1), \ldots, \theta(t_{r_i}))$ holds,
- $\mathfrak{A}, \theta, \Theta \models t \in X$ whenever $\theta(t) \in \Theta(X)$,
- $\mathfrak{A}, \theta, \Theta \models \neg\varphi$ whenever it is not the case that $\mathfrak{A}, \theta, \Theta \models \varphi$,
- $\mathfrak{A}, \theta, \Theta \models \varphi \wedge \psi$ whenever $\mathfrak{A}, \theta, \Theta \models \varphi$ and $\mathfrak{A}, \theta, \Theta \models \psi$,
- $\mathfrak{A}, \theta, \Theta \models \varphi \vee \psi$ whenever $\mathfrak{A}, \theta, \Theta \models \varphi$ or $\mathfrak{A}, \theta, \Theta \models \psi$,
- $\mathfrak{A}, \theta, \Theta \models \exists x\varphi$ whenever $\mathfrak{A}, \theta[x \leftarrow a], \Theta \models \varphi$ for some $a \in A$,
- $\mathfrak{A}, \theta, \Theta \models \forall x\varphi$ whenever $\mathfrak{A}, \theta[x \leftarrow a], \Theta \models \varphi$ for all $a \in A$,
- $\mathfrak{A}, \theta, \Theta \models \exists X\varphi$ whenever $\mathfrak{A}, \theta, \Theta[X \leftarrow B] \models \varphi$ for some $B \subseteq A$,
- $\mathfrak{A}, \theta, \Theta \models \forall X\varphi$ whenever $\mathfrak{A}, \theta, \Theta[X \leftarrow B] \models \varphi$ for all $B \subseteq A$.

Sometimes we evaluate MSO formulas in the *weak semantics* and in such case only finite sets are substituted for second-order variables. In this setting the last two items above must be replaced by the following:

- $\mathfrak{A}, \theta, \Theta \models \exists X\varphi$ in the weak semantics if $\mathfrak{A}, \theta, \Theta[X \leftarrow B] \models \varphi$ for some *finite* $B \subseteq A$,

– $\mathfrak{A}, \theta, \Theta \models \forall X \varphi$ in the weak semantics if $\mathfrak{A}, \theta, \Theta[X \leftarrow B] \models \varphi$ for all *finite* $B \subseteq A$.

We often call formulas that we evaluate using the weak semantics *weak monadic second-order logic* (WMSO) formulas.

1.2 Linear Orders, Words and Trees

1.2.1 Linear Orders

Linear orders are a prominent example of structures that appear throughout this thesis. The standard ordering of natural numbers is denoted $(\omega, <)$ or $(\mathbb{N}, <)$, the orderings of integers and rational numbers are denoted $(\mathbb{Z}, <)$ and $(\mathbb{Q}, <)$, respectively. Recall that $(\mathbb{Q}, <)$ is dense, meaning that between any two elements $x < y$ there is another element z such that $x < z < y$. On the other hand, we say that a linear order $(L, <)$ is *scattered* if it does not embed any dense order, or equivalently if it does not embed $(\mathbb{Q}, <)$. Given a linear order $(I, <)$ and orders $(L_i, <_i)$ for every $i \in I$, the sum $\sum_I L_i$ is defined as the linear ordering of $\bigcup_{i \in I} L_i \times \{i\}$ such that

$$(l, i) < (l', i') \iff i < i' \text{ or } i = i' \text{ and } l <_i l'.$$

We write $(L_0, <_0) + (L_1, <_1)$ for the sum over $(\{0, 1\}, <)$. For example $(\mathbb{Z}, <)$ is isomorphic to $\omega^* + \omega$, where ω^* is the standard ordering of negative integers, $(\{-1, -2, -3, \ldots\}, <)$. Note that sometimes we use the same name for the structure and its universe when the meaning is clear from the context, as in the case of ω above. Linear orders with additional unary predicates are called *chains*. For a linear order L and $a, b \in L$ we use the standard notation for intervals, so for example $[a, b)$ is a left-closed and right-open interval. Moreover, we write $L|_{[a,b)}$ for the order $L \cap [a, b)$, and for $X \subseteq L$ we use analogous notation, i.e. $X|_{[a,b]}$ for $X \cap [a, b]$.

One can classify countable linear orders using the sum operation defined above in the following way. Every countable linear order can be written as a dense sum of scattered linear orders, i.e. as $\sum_{\mathbb{Q}} L_i$ where each $(L_i, <_i)$ is a scattered linear order. Moreover, Hausdorff classified countable scattered linear orders in classes VD_α defined inductively as follows. $\mathsf{VD}_0 = \{\mathbf{1}\}$ consists of the linear order having one element (we leave out the empty linear order). For each ordinal $\alpha > 0$, VD_α consists of those linear orders that can be written as a sum $\sum_{\mathbb{Z}} L_i$ with each $L_i \in \mathsf{VD}_\beta$ for some $\beta < \alpha$. Let VD be the union of all the VD_α. Hausdorff has shown that VD contains every countable scattered linear order, and the *Hausdorff-rank* of a linear order $L \in \mathsf{VD}$ is defined as the smallest α such that $L \in \mathsf{VD}_\alpha$.

A linear order is *complete* if every one of its subsets has a least upper bound. Given a linearly ordered set $(L, <)$ its *Dedekind cuts* are subsets $C \subseteq L$ that are downward closed. The *completion* of $(L, <)$ is the set $DC(L)$ of Dedekind cuts of L ordered by inclusion, containing $\emptyset$ if there is no least element in L

and excluding $\emptyset$ in the other case. Note that the completion of L, which we denote by $\overline{L}$ and consider only up to isomorphism, is a complete linear order with both endpoints, i.e. a least and a greatest element.

Every linear order is naturally equipped with the *order topology* generated by open intervals. This allows us to speak of neighborhoods, open sets, limit (alternatively condensation or accumulation) points, and all other topological notions on every linear order.

1.2.2 Words

For a given set A we denote by A^* the set of all finite sequences of elements of A, by A^ω the set of all infinite sequences of elements of A (i.e. functions $\omega \to A$), and $A^{\leq\omega} = A^* \cup A^\omega$. Elements of A^* are often called finite words and elements of A^ω are infinite words over A. For any sequence $s = s_0 s_1 s_2 \ldots \in A^{\leq\omega}$ we denote by $|s|$ the length of s (either a natural number or ω) and by $s|_n = s_0 \ldots s_{n-1}$ the finite sequence composed of the first n elements of s, with $s|_0 = \varepsilon$, the empty sequence. We write $s[n]$ for the $(n+1)$st element of s (as we start counting from 0), so $s[n] = s_n$ for $n \in \mathbb{N}$. Similarly, $s[n,m]$ is the factor $s[n]s[n+1]\cdots s[m]$ and $s[n,m)$ is defined as $s[n,m-1]$, therefore in our notation $s|_n = s[0,n)$.

Given a finite sequence s and a sequence $r \in A^{\leq\omega}$ we denote by $s \cdot r$ (or just sr) the concatenation of s and r. For the n-times concatenation $s \cdots s$ we use the symbol s^n. Moreover, we write $s \sqsubseteq t$ if s is a prefix of t, i.e. if there exists a sequence r such that $t = sr$, and in such case we denote the difference by $t - s = r$. A subset B of $A^{\leq\omega}$ is said to be prefix-closed if for every $t \in B$ and $s \sqsubseteq t$ it holds that $s \in B$. For an infinite sequence $s \in A^\omega$ the set of elements that appear infinitely often in this sequence is denoted by $\mathrm{Inf}(s)$.

We sometimes extend all notations introduced above to vectors of sequences, so for example if $\overline{s}$ is a vector then $\overline{s[n]}$ or equivalently $\overline{s}[n]$ is the vector consisting of the $(n+1)$st element of each sequence in $\overline{s}$. Moreover, given a function $f : A \to B$ and $u \in A^{\leq\omega}$ we denote by $f(u)$ the sequence $f(u[0])\ f(u[1])\ f(u[2])\ \ldots \in B^{\leq\omega}$.

1.2.3 Trees

A tree is a structure $\mathfrak{T} = (T, <, P_1, \ldots, P_n)$ where P_i's are unary predicates and $<$ is an irreflexive and transitive binary *ancestor* relation with a least element called the *root of* $\mathfrak{T}$ and such that for every $v \in T$ the set $\{u \in T \mid u < v\}$ of ancestors of v is finite and linearly ordered by $<$. We consider only finitely branching trees, i.e. we assume that in every tree $\mathfrak{T}$ the number of $v \in T$ with at most n ancestors is finite for every n.

Elements of a tree are referred to as *nodes*, maximal linearly ordered sets of nodes are called *branches*, ancestor-closed and linearly ordered sets of nodes are called *paths*, whereas *chains* are arbitrary linearly ordered subsets. An *antichain* is a set of pairwise incomparable nodes. Given a node v, the subtree of $\mathfrak{T}$ rooted in v is obtained by restricting the structure to the domain $T_v = \{u \in T \mid u \geq v\}$ and is denoted $\mathfrak{T}_v$. Similarly, we use this notation for every set $X \subseteq T$, i.e. $X_v = X \cap T_v$.

Given a finite set A we denote by $\mathfrak{T}(A)$ the complete tree over A, which is a structure with the universe A^* and $<$ interpreted as the standard prefix ordering. The tree $\mathfrak{T}(A)$ also includes successor labels, i.e. for each $a \in A$ there is a predicate P_a in the structure $\mathfrak{T}(A)$ such that $P_a(u)$ holds exactly when $u = va$, which allows to distinguish all immediate successors of any node $v \in T$. An important example of such a tree is the complete binary tree, $\mathfrak{T}(\{0,1\}) = (\{0,1\}^*, <, S_0, S_1)$ where $S_0(u)$ holds if $u = v0$ and $S_1(u)$ if $u = v1$. In the case of complete k-ary trees we may write $\mathfrak{T}(k)$ for $\mathfrak{T}(\{0,\dots,k-1\})$, so the complete binary tree is designated $\mathfrak{T}(2)$.

Given an indexed family of trees and an order on the index which also satisfies the requirements of a tree, we define the sum of this family. Intuitively, the sum can be understood as replacing each node in the index tree by the corresponding tree in the family, and is formally defined as follows.

Definition 1.1 (Tree sum). *Let $\mathfrak{I} = (I, <^{\mathfrak{I}})$ be an unlabeled tree and for each $i \in I$ let $\mathfrak{T}_i = (T_i, <^i, P_1^i, \dots, P_n^i)$ be a tree. The* tree sum *of $(\mathfrak{T}_i)_{i \in \mathfrak{I}}$, denoted $\sum_{i \in \mathfrak{I}} \mathfrak{T}_i$, is the tree*

$$\mathfrak{T} = \Big(\ \bigcup_{i \in I} \{i\} \times T_i\ ,\ <^{\mathfrak{T}},\ \bigcup_{i \in I} \{i\} \times {P_1}^i, \dots, \bigcup_{i \in I} \{i\} \times {P_n}^i\ \Big)\,,$$

where $(i,a) <^{\mathfrak{T}} (j,b)$ for $i,j \in I$, $a \in T_i$, $b \in T_j$ iff:

$$i <^{\mathfrak{I}} j \text{ and } a \text{ is the root of } \mathfrak{T}_i, \text{ or } \ i = j \text{ and } a <^i b\,.$$

Unless explicitly noted, we will not distinguish between $\mathfrak{T}_i$ and the isomorphic subtree $\{i\} \times \mathfrak{T}_i$ of $\mathfrak{T}$.

A particular special case of the sum we will be using is when the index structure $\mathfrak{I}$ consists of a single branch. Let $(I, <)$ be a linear order, which is finite or isomorphic to ω, and let $(\mathfrak{T}_i)_{i \in I}$ be an I-indexed sequence of trees. Then the sum $\mathfrak{T} = \sum_{i \in I} \mathfrak{T}_i$ is well defined, and $(I, <)$ forms a path (not necessarily maximal) in $\mathfrak{T}$.

In addition to the trees defined above, we also study trees over infinite words, called ω-trees. A complete ω-tree over a finite set A is defined in an analogous way to $\mathfrak{T}(A)$ as $\mathfrak{T}^\omega(A) = (A^{\leq\omega}, <, \{S_a\}_{a \in A})$, where $<$ is again the prefix relation and S_a are successor labels, i.e. $S_a(u)$ holds only for finite words $u = va$. Moreover, we sometimes extend the trees with the equal-length relation el, defined by $\mathsf{el}(u,w) \iff |u| = |w|$. We denote a tree $\mathfrak{T}$ extended with this binary relation $\mathfrak{T}_{\mathsf{el}}$, so for example $\mathfrak{T}^\omega_{\mathsf{el}}(\{0,1\})$ is the complete binary ω-tree extended with the equal-length relation, i.e. $(\{0,1\}^{\leq\omega}, <, S_0, S_1, \mathsf{el})$.

1.3 Automata on ω-Words

The order $(\omega, <)$ and the binary tree $\mathfrak{T}(2)$ play an important role in computer science and logic because the monadic second-order theory of both of these structures is decidable, as proved by Büchi [17] and Rabin [74] respectively. These proofs use the notion of an automaton, either a word automaton for $(\omega, <)$ or a tree automaton for $\mathfrak{T}(2)$, and establish a one-to-one correspondence between relations recognized by automata and the ones definable in monadic second-order logic.

An ω-word automaton $\mathcal{A}$ over a finite alphabet Σ is a tuple $(Q, \Delta, q_0, \mathcal{F})$ where Q is a finite set of states, Δ is a transition relation $\Delta \subseteq Q \times \Sigma \times Q$, $q_0 \in Q$ is an initial state and $\mathcal{F}$ is an acceptance condition. An automaton is *deterministic* if Δ is a function $Q \times \Sigma \to Q$. In the case of the standard finite-word automata, the acceptance condition consists only of a set of final states and a word is accepted if some run ends in a final state. For ω-word automata the acceptance condition $\mathcal{F}$ is a set of runs, i.e. infinite sequences of states, which are considered accepting for the automaton, so $\mathcal{F} \subseteq Q^\omega$. In practice $\mathcal{F}$ is described in a finite way and there are a few representations that are often used for this purpose.

- The *Büchi condition* is represented by a set $F \subseteq Q$ and $\mathcal{F} = \{s \in Q^\omega \mid \mathrm{Inf}(s) \cap F \neq \emptyset\}$.
- The *parity condition* is defined using a mapping $\Omega : Q \to \{0, \ldots, d\}$ and $\mathcal{F} = \{s \in Q^\omega \mid \min(\mathrm{Inf}(\Omega(s))) \text{ is even}\}$.
- The *Rabin condition* is given by a set of pairs $\{(E_1, F_1), \ldots, (E_k, F_k)\}$ and $\mathcal{F} = \{s \in Q^\omega \mid \mathrm{Inf}(s) \cap F_i \neq \emptyset \text{ and } \mathrm{Inf}(s) \cap E_i = \emptyset \text{ for some } i\}$.
- The *Streett condition*, dual to the Rabin condition, is again represented by a set of pairs $\{(E_1, F_1), \ldots, (E_k, F_k)\}$, but in this case $\mathcal{F} = \{s \in Q^\omega \mid \mathrm{Inf}(s) \cap F_i \neq \emptyset \text{ or } \mathrm{Inf}(s) \cap E_i = \emptyset \text{ for every } i\}$.
- The *Muller condition* is defined by listing $F \subseteq \mathcal{P}(Q)$ and $\mathcal{F} = \{s \in Q^\omega \mid \mathrm{Inf}(s) \in F\}$.

A *run* of an automaton $\mathcal{A}$ on a word $w \in \Sigma^\omega$ is defined as any sequence of states $q_0 q_1 \ldots \in Q^\omega$ in which $\Delta(q_i, w[i], q_{i+1})$ holds for all i. The word w is *accepted* by $\mathcal{A}$ if there is a run ρ of $\mathcal{A}$ on w that is accepting, i.e. $\rho \in \mathcal{F}$, and we denote by $L(\mathcal{A})$ the set of all words accepted by an automaton $\mathcal{A}$.

It is well-known that non-deterministic Büchi, parity, Rabin, Streett and Muller automata all recognize the same class of languages, the ω-regular languages. The deterministic variants have the same expressive power for all the representations of acceptance conditions introduced above except for the case of Büchi condition, as deterministic Büchi automata are strictly weaker than non-deterministic ones. Moreover, the class of ω-regular languages is closed under union, intersection and complementation.

1.3.1 Alternating Automata

In addition to the standard notion of automata, we use *alternating* automata as a tool in our proofs. The intuition behind alternating automata is that, unlike in the deterministic case where only one run on a given word is possible, there are more possibilities of transitions from each state for a given letter. But unlike non-deterministic automata, an alternating automaton does not only accept a word if there *exists* an accepting run among all possible ones, or if *all* possible runs are accepting (as in universal automata), but it allows to alternate such conditions with respect to states of the automaton and has both existential and universal branching choices.

To define alternating automata we have to consider, for a given set of states Q, the set $\mathcal{B}^+(Q)$ of all *positive Boolean formulas* over Q. By definition $\mathcal{B}^+(Q)$ is the set of all Boolean formulas built using elements of Q, the Boolean connectives $\wedge$ and $\vee$ and the constants $\top$ (true) and $\bot$ (false). Note that negation is not allowed. We say that a subset $X \subseteq Q$ *satisfies* a formula $\varphi \in \mathcal{B}^+(Q)$ if φ is satisfied by the assignment that assigns true to all elements of X and false to $Q \setminus X$.

An *alternating automaton* $\mathcal{A}$ over an alphabet Σ is a tuple $(Q, \delta, q_0, \mathcal{F})$, where Q is the set of states, q_0 is the initial state, $\mathcal{F}$ is the acceptance condition, but this time δ does not point to a single next state but specifies a positive Boolean formula as transition condition, $\delta : Q \times \Sigma \to \mathcal{B}^+(Q)$. Intuitively, a *correct run* of $\mathcal{A}$ on a word w is a tree labeled with Q where the successors of each node form a satisfying set for the Boolean condition related to the state in this node and to the corresponding letter in w.

To capture this intuition formally and simplify notation, we define runs as sets of infinite sequences of states, so a run ρ is a subset of Q^ω. When one thinks of runs as trees, our definition corresponds to defining runs directly as the set of branches of the run-tree. For a run ρ represented in this way we write $s_\rho(u)$ for the set of all states appearing in ρ that prolong $u \in Q^*$, i.e. the successors of u when thinking of a run as a tree,

$$s_\rho(u) = \{q \in Q \;:\; \exists v \; u \cdot q \cdot v \in \rho\}.$$

We define that ρ is a correct run of $\mathcal{A}$ on the word w if for each infinite branch $u \in \rho$ and each prefix $u|_i$ the successors after that prefix satisfy the corresponding Boolean constraint, i.e. if $s_\rho(u|_i)$ satisfies $\delta(u[i], w[i])$ for all i and $u \in \rho$. We say that $\mathcal{A}$ accepts a word w if there is a correct, non-empty run ρ on w starting from q_0 such that each branch $u \in \rho$ is accepted, i.e. $u \in \mathcal{F}$, and again we write $L(\mathcal{A})$ for the language recognized by $\mathcal{A}$.

Alternating automata may seem more powerful than deterministic ones and it is often much easier to express problems in terms of alternating automata than in terms of deterministic ones, but they are in fact equal in expressiveness to the standard automata.

Theorem 1.2. *Every language recognized by an alternating Büchi, parity, Rabin, Streett or Muller automaton is ω-regular.*

The above theorem can be proved by expressing acceptance of alternating automata in monadic second-order logic on infinite words and then going back from the logic to automata [17]. Alternatively, one can give an explicit construction which shows that (for all acceptance conditions except the Büchi condition) the size of the deterministic automaton constructed for a language recognized by an alternating one is at most doubly exponential in the size of the original automaton, as first shown in [66].

1.3.2 ω-Semigroups

There is a fundamental correspondence between recognizability of sets by finite-word automata and by finite semigroups. It has been extended to recognizability of ω-regular sets, first using Wilke algebras [87] and later based on the notion of *ω-semigroups*. The theory of ω-semigroups was first well presented in [71] and is thoroughly discussed in [72], we only mention what is most necessary.

An ω-semigroup $S = (S_f, S_\omega, \cdot, *, \pi)$ is a two-sorted algebra, where $(S_f, \cdot)$ is a semigroup, $* : S_f \times S_\omega \mapsto S_\omega$ is the *mixed product* satisfying for every $s, t \in S_f$ and every $\alpha \in S_\omega$ the equality

$$s * (t * \alpha) = (s \cdot t) * \alpha$$

and where $\pi : S_f^\omega \mapsto S_\omega$ is the *infinite product* satisfying

$$s_0 * \pi(s_1, s_2, \ldots) = \pi(s_0, s_1, s_2, \ldots)$$

as well as the associativity rule

$$\pi(s_0, s_1, s_2, \ldots) = \pi(s_0 s_1 \cdots s_{k_1}, s_{k_1+1} s_{k_1+2} \cdots s_{k_2}, \ldots)$$

for every sequence $(s_i)_{i \geq 0}$ of elements of S_f and every strictly increasing sequence $(k_i)_{i \geq 0}$ of indices. For $s \in S_f$ we denote $s^\omega = \pi(s, s, \ldots)$.

Morphisms of ω-semigroups are defined to preserve all three products as expected. There is a natural way to extend finite semigroups and their morphisms to ω-semigroups. As in semigroup theory, idempotents play a central role in this extension. An *idempotent* is a semigroup element $e \in S$ satisfying $ee = e$. For every element s in a finite semigroup the sub-semigroup generated by s contains a unique idempotent s^k. The least $k > 0$ such that s^k is idempotent for every $s \in S_f$ is called the *exponent* of the semigroup S_f. Another useful notion is absorption of semigroup elements: we say that s *absorbs* t *(on the right)* if $st = s$.

There is a natural extension of the free semigroup Σ^+ to the free ω-semigroup $(\Sigma^+, \Sigma^\omega)$ with $*$ and π determined by concatenation. An ω-semigroup $S = (S_f, S_\omega)$ *recognizes* a language $L \subseteq \Sigma^\omega$ via a morphism $\phi : (\Sigma^+, \Sigma^\omega) \to (S_f, S_\omega)$ if $\phi^{-1}(\phi(L)) = L$. This notion of recognizability coincides, as for finite words, with recognizability by non-deterministic Büchi automata and translations from Büchi automata to ω-semigroups and back can be done effectively.

Theorem 1.3 ([71]). *A language $L \subseteq \Sigma^\omega$ is ω-regular if and only if it is recognized by a finite ω-semigroup.*

This correspondence allows one to engage in an algebraic study of varieties of ω-regular languages, and also has the advantage of hiding complications of cutting apart and stitching together runs of Büchi automata. This is precisely the reason for which we use this algebraic framework. Most remarkably, one does not need to understand the exact relationship between automata and ω-semigroups and the technical details of the constructions behind Theorem 1.3 to use ω-semigroups to simplify calculations on ω-regular sets.

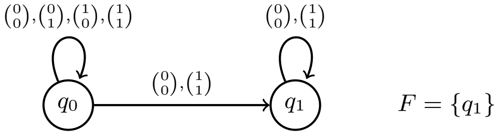
1.4 Automatic Structures

Before we define automatic presentations and automatic structures, let us introduce relations on finite and ω-words recognized by ω-word automata operating in a synchronized letter-to-letter fashion. Formally, R is an *ω-regular relation* of arity r over the domain Σ^ω if there exists an ω-automaton $\mathcal{A}$ over the alphabet Σ^r accepting the *convolution* $\otimes\overline{w}$ of ω-words $w_1, \ldots, w_r$ exactly when $R(w_1, \ldots, w_r)$ holds. The convolution is defined as

$$\otimes\overline{w}[i] = (w_1[i], \ldots, w_r[i]) \text{ for all } i.$$

For pairs of words w, w' we sometimes write $w \otimes w'$ or $\binom{w}{w'}$ for $\otimes(w, w')$.

Example 1.4. Words $u, v \in \Sigma^\omega$ have *equal ends*, written $u \sim_\mathrm{e} v$, if $u[n] = v[n]$ for all but a finitely many natural numbers n. This is an important example of an ω-regular equivalence relation. For $S, T \subseteq \mathbb{N}$ we extend the notation and write $S \sim_\mathrm{e} T$ if for all but finitely many $n \in \mathbb{N}$, $n \in S \iff n \in T$. The non-deterministic Büchi automaton depicted in Figure 1.1 accepts the equal-ends relation over $\{0, 1\}$.



$\binom{0}{0},\binom{0}{1},\binom{1}{0},\binom{1}{1}$
$\binom{0}{0},\binom{1}{1}$
$\binom{0}{0},\binom{1}{1}$
q_0
q_1
$F = \{q_1\}$

Fig. 1.1. An automaton for the equal ends relation

To define ω-regular relations over finite words one needs to add a padding end-of-word symbol $\Box \notin \Sigma$ to formally define the convolution of words of different length. For simplicity, we will sometimes identify a finite word $w \in \Sigma^*$ with its infinite padding $w^\Box = w\Box^\omega \in \Sigma_\Box^\omega$ where $\Sigma_\Box = \Sigma \cup \{\Box\}$. A language $L \subseteq \Sigma^*$ is regular, i.e. recognized by a standard finite-word automaton, exactly if the language $L^\Box = \{w^\Box \mid w \in L\}$ is ω-regular over $\Sigma_\Box$. Thus, we say that an r-ary relation $R \subseteq (\Sigma^*)^r$ is *regular* whenever its extension with padding is ω-regular over $\Sigma_\Box$. This is equivalent to defining a convolution for finite words by padding each word with $\Box$ to be as long as the longest one.

Definition 1.5 (Automatic Presentation)
For any relational structure $\mathfrak{A} = (A, R_1, \ldots, R_k)$, a tuple of ω-automata $\mathfrak{d} = (\mathcal{A}, \mathcal{A}_\approx, \mathcal{A}_1, \ldots, \mathcal{A}_k)$ together with a surjective naming function $f : L(\mathcal{A}) \to A$ constitutes an (ω-)automatic presentation *of $\mathfrak{A}$ over Σ if the following criteria are met:*

(i) the equivalence relation denoted $\approx$ and defined as
$$\{(u, w) \in L(\mathcal{A})^2 \mid f(u) = f(w)\}$$
is recognized by $\mathcal{A}_\approx$,
(ii) every automaton $\mathcal{A}_i$ recognizes a relation $\mathcal{R}_i \subseteq (\Sigma^\omega)^{r_i}$ with the same arity r_i as the relation R_i,
(iii) f is an isomorphism between $\mathfrak{A}_\mathfrak{d} = (L(\mathcal{A}), \mathcal{R}_1, \ldots, \mathcal{R}_k)/_\approx$ and $\mathfrak{A}$.

The presentation is said to be injective *whenever f is, in which case $\mathcal{A}_\approx$ can be omitted. Observe that the relation $\approx$ needs to be a congruence of the structure $(L(\mathcal{A}), \mathcal{R}_1, \ldots, \mathcal{R}_k)$ for item (iii) to hold.*

In case $L(\mathcal{A})$ only consists of words of the form $w^\Box$ where $w \in \Sigma^*$, we say that the presentation is (finite-word) automatic. We call a structure *(ω-)automatic* if it has an (ω-)automatic presentation.

There may exist different automatic presentations of a single structure, and different relations might be regular in each of these presentations. For example, for every number $p > 1$ there is a presentation of Presburger arithmetic $(\mathbb{N}, +)$ where numbers are coded in base p. The relation $|_2$ defined as $x|_2 y \iff x|y$ and $x = 2^n$ is a regular relation when numbers are coded in binary, but it is not regular in the presentation that uses ternary coding. Relations that are regular in each automatic presentation of a structure are called *intrinsically regular* and were first studied in [54, 55]. Every FO-definable relation is intrinsically regular, but on some structures there are intrinsically regular relations that are not definable in FO. Remarkably, this is not the case for Presburger arithmetic, where all relations that are intrinsically regular are definable in FO. An accessible survey of results on presentations of Presburger arithmetic is given in [16], and the general problem of different automatic presentations of a structure is studied in [5, 6].

The basic advantage of having an (ω-)automatic presentation of a structure lies in the fact that first-order formulas can be effectively evaluated using classical automata constructions. This is expressed by the following fundamental theorem.

Theorem 1.6 (Cf. [45, 53, 14]). *There is an effective procedure that given an (ω-)automatic presentation $\mathfrak{d}, f$ of a structure $\mathfrak{A}$, and given a FO-formula $\varphi(\overline{x})$ constructs an (ω-)automaton recognizing $f^{-1}(\varphi^\mathfrak{A})$. The FO-theory of every (ω-)automatic structure is decidable.*

1.5 Interpretations and Complete Structures

In this section, instead of explicitly representing a structure by a finite object, as done above using automata, we consider operations for transforming structures. More precisely, we fix an underlying family of structures and a class of operations that transform structures, and investigate the class of structures obtained by applying the transformations to structures in the underlying family. When the operations preserve decidability of first-order or monadic second-order logic, and structures in the underlying family have decidable first-order or monadic second-order theory, then we obtain a class of structures on which the corresponding logic is again decidable.

An important and well studied way of transforming structures is the model-theoretic interpretation, where one structure is interpreted in another one using formulas of either first-order or monadic second-order logic. Interpretations preserve decidability of the corresponding logics, or transform structures with decidable monadic second-order theory to structures with decidable first-order theory. We will show a few ways in which automatic structures can be characterized and extended by means of interpretations in trees and linear orders.

Definition 1.7. *Let $\mathfrak{A} = (A, R_1, \ldots, R_k)$ and $\mathfrak{B}$ be relational structures and let r_i be the arity of R_i. A tuple of first-order formulas over $\sigma(\mathfrak{B})$,*

$$\mathcal{I} = (\delta(\overline{x}), \varepsilon(\overline{x_1}, \overline{x_2}), \varphi_1(\overline{x}_1, \ldots, \overline{x}_{r_1}), \ldots, \varphi_k(\overline{x}_1, \ldots, \overline{x}_{r_k})),$$

where each vector $\overline{x}, \overline{x_i}$ is of the same length n is an n-dimensional FO interpretation of $\mathfrak{A}$ *in $\mathfrak{B}$ if $\mathcal{I}(\mathfrak{B}) = (\delta^{\mathfrak{B}}, \varphi_1^{\mathfrak{B}}, \ldots, \varphi_k^{\mathfrak{B}})/\varepsilon^{\mathfrak{B}}$ is isomorphic to $\mathfrak{A}$.*

In an analogous way, a tuple of MSO *or* WMSO *formulas over $\sigma(\mathfrak{B})$,*

$$\mathcal{J} = (\delta(X), \varepsilon(X_1, X_2), \varphi_1(X_1, \ldots, X_{r_1}), \ldots, \varphi_k(X_1, \ldots, X_{r_k})),$$

where X and X_i are single second-order variables, is an MSO*-to-*FO *or a* WMSO*-to-*FO *interpretation if $\mathcal{J}(\mathfrak{B}) = (\delta^{\mathfrak{B}}, \varphi_1^{\mathfrak{B}}, \ldots, \varphi_k^{\mathfrak{B}})/\varepsilon^{\mathfrak{B}}$ is isomorphic to $\mathfrak{A}$, with the formulas evaluated using the standard or the weak semantics respectively. If there exists an* FO*,* MSO*-to-*FO *or* WMSO*-to-*FO *interpretation of $\mathfrak{A}$ in $\mathfrak{B}$ we denote this by $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{B}$, $\mathfrak{A} \leq_{\mathsf{MSO}\to\mathsf{FO}} \mathfrak{B}$ or $\mathfrak{A} \leq_{\mathsf{WMSO}\to\mathsf{FO}} \mathfrak{B}$, respectively.*

Let ψ be a first-order formula over $\sigma(\mathfrak{A})$ and $\mathcal{I}$ an interpretation of $\mathfrak{A}$ in $\mathfrak{B}$. We construct the formula $\psi^{\mathcal{I}}$ by replacing every relation symbol R_i in ψ by the corresponding formula φ_i of $\mathcal{I}$, replacing every equality $t_1 = t_2$ in ψ by $\varepsilon(t_1, t_2)$ and relativizing the quantifiers in the following way. In the case of FO interpretations we replace $\exists x \varphi$ by $\exists \overline{x}(\delta(\overline{x}) \wedge \varphi)$ and $\forall x \varphi$ by $\forall \overline{x}(\delta(\overline{x}) \to \varphi)$, and in the second-order case we use second-order variables and thus $\exists X(\delta(X) \wedge \varphi)$ and $\forall X(\delta(X) \to \varphi)$, respectively. The standard interpretation lemma states that $\mathfrak{A} \models \psi$ exactly if $\mathfrak{B} \models \psi^{\mathcal{I}}$. This allows us to deduce decidability of the

FO-theory of $\mathfrak{A}$ from decidability of the FO, MSO or WMSO theory of $\mathfrak{B}$, given an FO, MSO-to-FO or WMSO-to-FO interpretation of $\mathfrak{A}$ in $\mathfrak{B}$, respectively.

A class $\mathcal{K}$ of structures is *closed under* FO*-interpretations* if for all $\mathfrak{B} \in \mathcal{K}$ whenever $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{B}$ then $\mathfrak{A} \in \mathcal{K}$ as well. For such a class $\mathcal{K}$ we say that a structure $\mathfrak{B}$ is *complete for* $\mathcal{K}$ if all $\mathfrak{A} \in \mathcal{K}$ are FO-interpretable in $\mathfrak{B}$. It follows from Theorem 1.6 that the class of (ω-)automatic structures is closed under FO-interpretations, because every relation defined in FO using only (ω-)regular relations is again (ω-)regular.

One way to characterize automatic structures by means of interpretations is to find complete automatic structures, and such structures were presented in [11, 14]. It was shown there that for any finite alphabet Σ with at least two letters, the complete tree over Σ extended with the equal-length relation, $\mathfrak{T}_{\mathsf{el}}(\Sigma)$, is complete for the class of automatic structures. Analogously, the complete ω-tree with the equal-length relation, $\mathfrak{T}^{\omega}_{\mathsf{el}}(\Sigma)$, is complete for the class of ω-automatic structures.

It is natural to ask whether Presburger arithmetic is a complete automatic structure. The answer is negative, but there are extensions of Presburger arithmetic that are complete. Let us define the following structures: $\mathfrak{N}_p = (\mathbb{N}, +, |_p)$ where $x|_p y \iff x|y$ and $x = p^n$ for some $n \in \mathbb{N}$, and $\mathfrak{R}_p = (\mathbb{R}, +, \leq, |_p, 1)$ where $x|_p y \iff y = kx$ and $x = p^l$ for some $k, l \in \mathbb{Z}$. It was shown in [11, 14] that for all integers $p \geq 2$ the extensions $\mathfrak{N}_p$ and $\mathfrak{R}_p$ of Presburger arithmetic and the real arithmetic are indeed complete, for the class of automatic and ω-automatic structures respectively.

Another way to characterize automatic structures, where WMSO-to-FO and MSO-to-FO interpretations are used, was first mentioned in [78] and more systematically introduced in [22]. This characterization extends the intimate connection between ω-automata over words and MSO over $(\omega, <)$, as well as between finite-word automata and WMSO over $(\omega, <)$, to (ω-)automatic structures. A structure $\mathfrak{A}$ is finite-word automatic if there is a WMSO-to-FO interpretation of $\mathfrak{A}$ in $(\omega, <)$, and a structure $\mathfrak{B}$ is ω-automatic if there is an MSO-to-FO interpretation of $\mathfrak{B}$ in $(\omega, <)$. Let us summarize the characterizations of automatic structures by means of interpretations in the following theorems.

Theorem 1.8 (Cf. [11, 14, 78, 22]). *For any relational structure* $\mathfrak{A}$ *the following statements are equivalent:*

- $\mathfrak{A}$ *is finite-word automatic,*
- $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{T}_{\mathsf{el}}(2)$*,*
- $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{N}_2$*,*
- $\mathfrak{A} \leq_{\mathsf{WMSO} \to \mathsf{FO}} (\omega, <)$*.*

Theorem 1.9 (Cf. [11, 14, 78, 22]). *For any relational structure* $\mathfrak{A}$ *the following statements are equivalent:*

- $\mathfrak{A}$ *is ω-automatic,*
- $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{T}^{\omega}_{\mathsf{el}}(2)$,
- $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{R}_2$,
- $\mathfrak{A} \leq_{\mathsf{MSO}\to\mathsf{FO}} (\omega, <)$.

The characterization of automatic structures by MSO-to-FO interpretations was used in [22] to define *generalized-automatic structures.* We say that $\mathfrak{A}$ is an (ω-)generalized-automatic structure if there is a WMSO-to-FO (or MSO-to-FO) interpretation of $\mathfrak{A}$ in some tree $\mathfrak{T}$. In particular, we say that the structure is (ω-)tree-automatic if this is the case for $\mathfrak{T}(2)$, the complete binary tree. By the result of Rabin and the interpretation lemma, (ω-)tree-automatic structures have a decidable first-order theory. Moreover, in chapter 7 we show that certain extensions of first-order logic collapse to FO on all ω-generalized-automatic structures.

1.6 Composition in Monadic Second-Order Logic

To study logic on linear orders and trees with arbitrary additional predicates it is convenient to depart from automata and use related methods from mathematical logic, especially the composition method. The history of the composition method starts with the introduction of Ehrenfeucht games [29], which are an intuitive formulation of Fraïssé's characterization of elementary equivalence, i.e. indistinguishability of relational structures by first-order formulas. These games were first defined for first-order logic and extended to weak monadic second-order logic [29]. Later, other logical systems were covered, such as modal, temporal and infinitary logics that we discuss in chapter 2. Here we focus on the extension of this method, now usually called the Ehrenfeucht-Fraïssé method, to full monadic second-order logic over linear orders and trees.

While Ehrenfeucht proved decidability of the first-order theory of countable ordinals using logical methods [28, 29], decidability of the full monadic second-order theory of these orderings was first shown by Büchi using automata [17, 18, 19]. Only later Shelah gave, in his celebrated and difficult paper [80], alternative proofs of Büchi's results (and many more) using an extension of the Ehrenfeucht-Fraïssé method to full monadic second-order logic, which he called the composition of monadic theories. This method was subsequently used by Gurevich and Shelah to obtain even more results, for example in [37, 41] and with Magidor in [40]. Theoretical computer scientists long preferred the automata theoretic approach, even after the composition method was well presented in Gurevich's survey [38]. It was only after the more accessible survey by Wolfgang Thomas [83] that the merits of the composition method started to be appreciated in theoretical computer science, which resulted in numerous papers. One example is the characterization of all extensions of $(\omega, <)$ by unary predicates that have a decidable monadic second-order theory [75].

The quantifier rank of a formula φ, denoted $\mathrm{qr}(\varphi)$, is the maximum depth of nesting of quantifiers in φ. For fixed n and l (and a fixed signature) we denote by $\mathrm{Form}_{n,m}$ the set of formulas of quantifier depth $\leq n$ and with free variables among $X_1, \ldots, X_m$.

For a structure $\mathfrak{A}$ and a tuple $\overline{U}$ of m subsets of $\mathfrak{A}$, the *monadic n-theory of $\overline{U}$*, $\mathrm{Th}^n(\mathfrak{A}, \overline{U})$, is the set of all MSO formulas $\varphi(\overline{X}) \in \mathrm{Form}_{n,m}$, having no more than n nested quantifiers in any subformula and no free variables other than $X_1, \ldots, X_m$, for which $\mathfrak{A} \models \varphi(\overline{U})$, i.e.

$$\mathrm{Th}^n(\mathfrak{A}, \overline{U}) = \{\, \varphi(\overline{X}) \in \mathrm{Form}_{n,m} \mid \mathfrak{A} \models \varphi(\overline{U}) \,\}.$$

For any $n, m > 0$, the set $\mathrm{Form}_{n,m}$ is infinite, but it only contains finitely many semantically distinct formulas, i.e. there are only finitely many n-theories in m variables. Moreover, every n-theory $\mathrm{Th}^n(\mathfrak{A}, \overline{U})$ is definable by a single MSO formula $\tau(\overline{X})$ having m free variables and quantifier depth at most n. Hintikka formulas are canonical formulas defining n-theories.

Lemma 1.10 (Hintikka Lemma [42]). *For every $n, m \in \mathbb{N}$ (and a fixed signature), we can compute a* finite *set $H_{n,m} \subseteq \mathrm{Form}_{n,m}$ such that:*

- *For every structure $\mathfrak{A}$ and $\overline{U} \subseteq \mathfrak{A}$ there is a* unique *$\tau \in H_{n,m}$ such that $\mathfrak{A} \models \tau(\overline{U})$.*
- *If $\tau_1, \tau_2 \in H_{n,m}$ and $\tau_1 \neq \tau_2$ then $\tau_1 \wedge \tau_2$ is unsatisfiable.*
- *If $\tau \in H_{n,m}$ and $\varphi \in \mathrm{Form}_{n,m}$, then either $\tau \models \varphi$ or $\tau \models \neg\varphi$. Furthermore, there is an algorithm that, given such τ and φ, decides which of these two possibilities holds.*

*Elements of $H_{n,m}$ are called (n,m)-*Hintikka formulas*.*

We say that a structure $\mathfrak{A}$ with labels (unary predicates) $\overline{U}$ has *type* $\tau \in H_{n,m}$, denoted $\mathrm{Tp}^n(\mathfrak{A}, \overline{U}) = \tau$, if $\mathfrak{A} \models \tau(\overline{U})$, i.e. if τ and $\mathrm{Th}^n(\mathfrak{A}, \overline{U})$ are equivalent. We sometimes speak of the n-type of a tuple of subsets $\overline{V} = V_1, \ldots, V_m$ of a given structure $\mathfrak{A}$ which already contains labels $\overline{U} = U_1, \ldots, U_l$. This is synonymous with the n-type $\tau \in H_{n,l+m}$ of the structure $(\mathfrak{A}, \overline{V})$ obtained by expansion of $\mathfrak{A}$ with the predicates interpreted as $\overline{V}$.

The essence of the composition method is that certain operations on structures, such as disjoint union and ordered sums of linear orders, can be projected to n-theories, i.e. there are corresponding operations mapping n-theories of constituent structures to the n-theory of the resulting structure. In other words, n-theories can be composed.

Here we state a very simple form of the composition method on linear orders and on trees, which can be proven directly using Ehrenfeucht-Fraïssé games. As mentioned before, there are more powerful theorems also known as the composition method, e.g. the effective ones presented later in chapters 6 and 7 and other, c.f. [80, 37, 41, 40].

Theorem 1.11 (Composition on linear orders)
Let $(I,<)$ *be a linear order, and* $\{\mathfrak{L}_i \mid i \in I\}$ *and* $\{\mathfrak{L}'_i \mid i \in I\}$ *two* I-*indexed sequences of chains such that* $\mathrm{Tp}^n(\mathfrak{L}_i) = \mathrm{Tp}^n(\mathfrak{L}'_i)$ *for all* $i \in I$. *Then* $\mathrm{Th}^n\left(\sum_{i\in I} \mathfrak{L}_i\right) = \mathrm{Th}^n\left(\sum_{i\in I} \mathfrak{L}'_i\right)$.

Theorem 1.12 (Composition on tree sums)
Let $\mathfrak{I} = (I, <^I)$ *be a fixed unlabeled tree. For every family* $\{\mathfrak{T}_i \mid i \in I\}$ *of trees, the theory* $\mathrm{Th}^n(\sum_{i\in\mathfrak{I}} \mathfrak{T}_i)$ *is uniquely determined by the theories* $\mathrm{Th}^n(\mathfrak{T}_i)$.