1.4 Orderings

An ordering R is a binary relation on some set M. Depending on particular properties such as

$$\begin{array}{cl}
\text{(reflexivity)} & \forall x \in M\, R(x,x) \\
\text{(irreflexivity)} & \forall x \in M\, \neg R(x,x) \\
\text{(antisymmetry)} & \forall x,y \in M\, (R(x,y) \wedge R(y,x) \rightarrow x = y) \\
\text{(transitivity)} & \forall x,y,z \in M\, (R(x,y) \wedge R(y,z) \rightarrow R(x,z)) \\
\text{(totality)} & \forall x,y \in M\, (R(x,y) \vee R(y,x))
\end{array}$$

there are different types of orderings. The relation $=$ is the identity relation on M. The quantifier $\forall$ reads "for all", and the boolean connectives $\wedge$, $\vee$, and $\rightarrow$ read "and", "or", and "implies", respectively. For example, the above formula stating reflexivity $\forall x \in M\, R(x,x)$ is a shorthand for "for all $x \in M$ the relation $R(x,x)$ holds".

Actually, the definition of the above properties is informal in the sense that I rely on the meaning of certain symbols such as $\in$ or $\rightarrow$. While the former is assumed to be known from school math, the latter is "explained" above. So, strictly speaking this book is neither self contained, nor overall formal. For the concrete logics developed in subsequent chapters, I will formally define $\rightarrow$ but here, where it is used to state properties needed to eventually define the notion of an ordering, it remains informal. Although it is possible to develop the overall content of this book in a completely formal style, such an approach is typically impossible to read and comprehend. Since this book is about teaching a general framework to eventually generate automated reasoning procedures this would not be the right way to go. In particular, being informal starts already with the use of natural language. In order to support this "mixed" style, examples and exercises deepen the understanding and rule out potential misconceptions.

Now, based on the above defined properties of a relation, the usual notions with respect to orderings are stated below.

Definition 1.4.1 (Orderings)**.** A *(partial) ordering* $\succeq$ (or simply ordering) on a set M, denoted $(M, \succeq)$, is a reflexive, antisymmetric, and transitive binary relation on M. It is a *total ordering* if it also satisfies the totality property. A *strict (partial) ordering* $\succ$ is a transitive and irreflexive binary relation on M. A strict ordering is *well-founded*, if there is no infinite descending chain $m_0 \succ m_1 \succ m_2 \succ \ldots$ where $m_i \in M$.

Given a strict partial order $\succ$ on some set M, its respective partial order $\succeq$ is constructed by adding the identities $(\succ \cup =)$. If the partial order $\succeq$ extension of some strict partial order $\succ$ is total, then we call also $\succ$ total. As an alternative, a strict partial order $\succ$ is total if it satisfies the strict totality axiom $\forall x, y \in M\, (x \neq y \rightarrow (R(x,y) \vee R(y,x)))$. Given some ordering $\succ$ the respective ordering $\prec$ is defined by $a \prec b$ iff $b \succ a$.

Example 1.4.2. The well-known relation $\leq$ on $\mathbb{N}$, where $k \leq l$ if there is a j so that $k + j = l$ for $k, l, j \in \mathbb{N}$, is a total ordering on the naturals. Its strict subrelation $<$ is well-founded on the naturals. However, $<$ is not well-founded on $\mathbb{Z}$.

Definition 1.4.3 (Minimal and Smallest Elements)**.** Given a strict ordering $(M, \succ)$, an element $m \in M$ is called *minimal*, if there is no element $m' \in M$ so that $m \succ m'$. An element $m \in M$ is called *smallest*, if $m' \succ m$ for all $m' \in M$ different from m.

Note the subtle difference between minimal and smallest. There may be several minimal elements in a set M but only one smallest element. Furthermore, in order for an element being smallest in M it needs to be comparable to all other elements from M.

Example 1.4.4. In $\mathbb{N}$ the number 0 is smallest and minimal with respect to $<$. For the set $M = \{q \in \mathbb{Q} \mid 5 \leq q\}$ the ordering $<$ on M is total, has the minimal and smallest element 5 but is not well-founded.

If $<$ is the ancestor relation on the members of a human family, then $<$ typically will have several minimal elements, the currently youngest children of the family, but no smallest element, as long as there is a couple with more than one child. Furthermore, $<$ is not total, but well-founded.

Well-founded orderings can be combined to more complex well-founded orderings by lexicographic or multiset extensions.

Definition 1.4.5 (Lexicographic and Multiset Ordering Extensions)**.** Let $(M_1, \succ_1)$ and $(M_2, \succ_2)$ be two strict orderings. Their *lexicographic combination* $\succ_{\text{lex}} = (\succ_1, \succ_2)$ on $M_1 \times M_2$ is defined as $(m_1, m_2) \succ (m_1', m_2')$ iff $m_1 \succ_1 m_1'$ or $m_1 = m_1'$ and $m_2 \succ_2 m_2'$.

Let $(M, \succ)$ be a strict ordering. The *multiset extension* $\succ_{\text{mul}}$ to multisets over M is defined by $S_1 \succ_{\text{mul}} S_2$ iff $S_1 \neq S_2$ and $\forall m \in M\, [S_2(m) > S_1(m) \rightarrow \exists m' \in M\, (m' \succ m \wedge S_1(m') > S_2(m'))]$.

The definition of the lexicographic ordering extensions can be expanded to n-tuples in the obvious way. So it is also the basis for the standard lexicographic ordering on words as used, e.g., in dictionaries. In this case the M_i are alphabets, say *a*-*z*, where $a \prec b \prec \ldots \prec z$. Then according to the above definition *tiger* $\prec$ *tree*.

Example 1.4.6 (Multiset Ordering)**.** Consider the multiset extension of $(\mathbb{N}, >)$. Then $\{2\} >_{\text{mul}} \{1, 1, 1\}$ because there is no element in $\{1, 1, 1\}$ that is larger than 2. As a border case, $\{2, 1\} >_{\text{mul}} \{2\}$ because there is no element that has more occurrences in $\{2\}$ compared to $\{2, 1\}$. The other way round, 1 has more occurrences in $\{2, 1\}$ than in $\{2\}$ and there is no larger element to compensate for it, so $\{2\} \not>_{\text{mul}} \{2, 1\}$.

Proposition 1.4.7 (Properties of Lexicographic and Multiset Ordering Extensions)**.** Let $(M, \succ)$, $(M_1, \succ_1)$, and $(M_2, \succ_2)$ be orderings. Then

1. $\succ_{\text{lex}}$ is an ordering on $M_1 \times M_2$.

2. if $(M_1, \succ_1)$ and $(M_2, \succ_2)$ are well-founded so is $\succ_{\text{lex}}$.

3. if $(M_1, \succ_1)$ and $(M_2, \succ_2)$ are total so is $\succ_{\text{lex}}$.

4. $\succ_{\text{mul}}$ is an ordering on multisets over M.

5. if $(M, \succ)$ is well-founded so is $\succ_{\text{mul}}$.

6. if $(M, \succ)$ is total so is $\succ_{\text{mul}}$.

Please recall that multisets are finite.

T

The lexicographic ordering on words is not well-founded if words of arbitrary length are considered. Starting from the standard ordering on the alphabet, e.g., the following infinite descending sequence can be constructed: $b \succ ab \succ aab \succ \ldots$. It becomes well-founded if it is lexicographically combined with the length ordering, see Exercise **??**.

Lemma 1.4.8 (König's Lemma)**.** Every finitely branching tree with infinitely many nodes contains an infinite path.

1.5 Induction

More or less all sets of objects in computer science or logic are defined *inductively*. Typically, this is done in a bottom-up way, where starting with some definite set, it is closed under a given set of operations.

Example 1.5.1 (Inductive Sets)**.** In the following, some examples for inductively defined sets are presented:

1. The set of all Sudoku problem states, see Section 1.1, consists of the set of start states $(N; \top; \top)$ for consistent assignments N plus all states that can be derived from the start states by the rules Deduce, Conflict, Backtrack, and Fail. This is a finite set.

2. The set $\mathbb{N}$ of the natural numbers, consists of 0 plus all numbers that can be computed from 0 by adding 1. This is an infinite set.

3. The set of all strings Σ^* over a finite alphabet Σ. All letters of Σ are contained in Σ^* and if u and v are words out of Σ^* so is the word uv, see Section 1.2. This is an infinite set.

All the previous examples have in common that there is an underlying well-founded ordering on the sets induced by the construction. The minimal elements for the Sudoku are the problem states $(N; \top; \top)$, for the natural numbers it is 0 and for the set of strings it is the empty word. Now in order to prove a property of an inductive set it is sufficient to prove it (i) for the minimal element(s) and (ii) assuming the property for an arbitrary set of elements, to prove that it holds for all elements that can be constructed "in one step" out those elements. This is the principle of *Noetherian Induction*.

Theorem 1.5.2 (Noetherian Induction)**.** Let $(M, \succ)$ be a well-founded ordering, and let Q be a predicate over elements of M. If for all $m \in M$ the implication

if $Q(m')$, for all $m' \in M$ so that $m \succ m'$,	(induction hypothesis)
then $Q(m)$.	(induction step)

is satisfied, then the property $Q(m)$ holds for all $m \in M$.

Proof. Let $X = \{m \in M \mid Q(m) \text{ does not hold}\}$. Suppose, $X \neq \emptyset$. Since $(M, \succ)$ is well-founded, X has a minimal element m_1. Hence for all $m' \in M$ with $m' \prec m_1$ the property $Q(m')$ holds. On the other hand, the implication which is presupposed for this theorem holds in particular also for m_1, hence $Q(m_1)$ must be true so that m_1 cannot be in X - a contradiction. □

Note that although the above implication sounds like a one step proof technique it is actually not. There are two cases. The first case concerns all elements that are minimal with respect to $\prec$ in M and for those the predicate Q needs to hold without any further assumption. The second case is then the induction step showing that by assuming Q for all elements strictly smaller than some m, Q holds for m.

Now for context free grammars. Let $G = (N, T, P, S)$ be a context-free grammar (possibly infinite) and let q be a property of T^* (the words over the alphabet T of terminal symbols of G).

q holds for *all* words $w \in L(G)$, whenever one can prove the following two properties:

1. (*base cases*)
   $q(w')$ holds for each $w' \in T^*$ so that $X ::= w'$ is a rule in P.

2. (*step cases*)
   If $X ::= w_0 X_0 w_1 \ldots w_n X_n w_{n+1}$ is in P with $X_i \in N$, $w_i \in T^*$, $n \geq 0$, then for all $w'_i \in L(G, X_i)$, whenever $q(w'_i)$ holds for $0 \leq i \leq n$, then also $q(w_0 w'_0 w_1 \ldots w_n w'_n w_{n+1})$ holds.

Here $L(G, X_i) \subseteq T^*$ denotes the language generated by the grammar G from the non-terminal X_i.

Let $G = (N, T, P, S)$ be an *unambiguous* (why?) context-free grammar. A function f is well-defined on $L(G)$ (that is, unambiguously defined) whenever these 2 properties are satisfied:

1. (base cases)
   f is well-defined on the words $w' \in T^*$ for each rule $X ::= w'$ in P.

2. (step cases)
   If $X ::= w_0 X_0 w_1 \dots w_n X_n w_{n+1}$ is a rule in P then $f(w_0 w_0' w_1 \dots w_n w_n' w_{n+1})$ is well-defined, assuming that each of the $f(w_i')$ is well-defined.

1.6 Rewrite Systems

The final ingredient to actually start the journey through different logical systems is rewrite systems. Here I define the needed computer science background for defining algorithms in the form of rule sets. In Section 1.1 the rewrite rules Deduce, Conflict, Backtrack, and Fail defined an algorithm for solving 4×4 Sudokus. The rules operate on the set of Sudoku problem states, starting with a set of initial states $(N; \top; \top)$ and finishing either in a solution state $(N; D; \top)$ or a fail state $(N; \top; \bot)$. The latter are called *normal forms* (see below) with respect to the above rules, because no more rule is applicable to a solution state $(N; D; \top)$ or a fail state $(N; \top; \bot)$.

Definition 1.6.1 (Rewrite System)**.** A *rewrite system* is a pair $(M, \rightarrow)$, where M is a non-empty set and $\rightarrow \subseteq M \times M$ is a binary relation on M. Figure 1.4 defines the needed notions for $\rightarrow$.

$\rightarrow^0$	$= \{ (a,a) \mid a \in M \}$	*identity*
$\rightarrow^{i+1}$	$= \rightarrow^i \circ \rightarrow$	*$i+1$-fold composition*
$\rightarrow^+$	$= \bigcup_{i>0} \rightarrow^i$	*transitive closure*
$\rightarrow^*$	$= \bigcup_{i \geq 0} \rightarrow^i \; = \; \rightarrow^+ \cup \rightarrow^0$	*reflexive transitive closure*
$\rightarrow^=$	$= \rightarrow \cup \rightarrow^0$	*reflexive closure*
$\rightarrow^{-1}$	$= \{ (b,c) \mid c \rightarrow b \}$	*inverse*
$\leftrightarrow$	$= \rightarrow \cup \leftarrow$	*symmetric closure*
$\leftrightarrow^+$	$= (\leftrightarrow)^+$	*transitive symmetric closure*
$\leftrightarrow^*$	$= (\leftrightarrow)^*$	*refl. trans. symmetric closure*

Figure 1.4: Notation on $\rightarrow$

For a rewrite system $(M, \rightarrow)$ consider a sequence of elements a_i that are pairwise connected by the symmetric closure, i.e., $a_1 \leftrightarrow a_2 \leftrightarrow a_3 \dots \leftrightarrow a_n$. Then a_i is called a *peak* in such a sequence, if actually $a_{i-1} \leftarrow a_i \rightarrow a_{i+1}$.

Actually, in Definition 1.6.1 I overload the symbol $\rightarrow$ that has already denoted logical implication, see Section 1.4, with a rewrite relation. This overloading will remain throughout this book. The rule symbol

C

$\Rightarrow$ is only used on the meta level in this book, e.g., to define the Sudoku algorithm on problem states, Section 1.1. Nevertheless, these meta rule systems are also rewrite systems in the above sense. The rewrite symbol $\rightarrow$ is used on the formula level inside a problem state. This will become clear when I turn to more complex logics starting from Chapter 2.

Definition 1.6.2 (Reducible)**.** Let $(M, \rightarrow)$ be a rewrite system. An element $a \in M$ is *reducible*, if there is a $b \in M$ such that $a \rightarrow b$. An element $a \in M$ is *in normal form (irreducible)*, if it is not reducible. An element $c \in M$ is a *normal form* of b, if $b \rightarrow^* c$ and c is in normal form, denoted by $c = b{\downarrow}$. Two elements b and c are *joinable*, if there is an a so that $b \rightarrow^* a \;{}^*\!\!\leftarrow c$, denoted by $b \downarrow c$.

Traditionally, $c = b{\downarrow}$ implies that the normal form of b is unique. However, when defining logical calculi as abstract rewrite systems on states in subsequent chapters, sometimes it is useful to write $c = b{\downarrow}$ even if c is not unique. In this case, c is an arbitrary irreducible element obtained from reducing b.

Definition 1.6.3 (Properties of $\rightarrow$)**.** A relation $\rightarrow$ is called

Church-Rosser	if $b \leftrightarrow^* c$ implies $b \downarrow c$
confluent	if $b \;{}^*\!\!\leftarrow a \rightarrow^* c$ implies $b \downarrow c$
locally confluent	if $b \leftarrow a \rightarrow c$ implies $b \downarrow c$
terminating	if there is no infinite descending chain $b_0 \rightarrow b_1 \dots$
normalizing	if every $b \in A$ has a normal form
convergent	if it is confluent and terminating

Lemma 1.6.4. If $\rightarrow$ is terminating, then it is normalizing.

The reverse implication of Lemma 1.6.4 does not hold. Assuming this is a frequent mistake. Consider $M = \{a, b, c\}$ and the relation $a \rightarrow b$, $b \rightarrow a$, and $b \rightarrow c$. Then $(M, \rightarrow)$ is obviously not terminating, because we can cycle between a and b. However, $(M, \rightarrow)$ is normalizing. The normal form is c for all elements of M. Similarly, there are rewrite systems that are locally confluent, but not confluent, see Figure . In the context of termination the property holds, see Lemma 1.6.6.

Theorem 1.6.5. The following properties are equivalent for any rewrite system $(M, \rightarrow)$:

(i) $\rightarrow$ has the Church-Rosser property.
(ii) $\rightarrow$ is confluent.

Proof. (i) $\Rightarrow$ (ii): trivial.
(ii) $\Rightarrow$ (i): by induction on the number of peaks in the derivation $b \leftrightarrow^* c$. □

Lemma 1.6.6 (Newman's Lemma : Confluence versus Local Confluence)**.** Let $(M, \rightarrow)$ be a terminating rewrite system. Then the following properties are equivalent:

(i) $\rightarrow$ is confluent

(ii) $\rightarrow$ is locally confluent

Proof. (i) $\Rightarrow$ (ii): trivial.

(ii) $\Rightarrow$ (i): Since $\rightarrow$ is terminating, it is a well-founded ordering (see Exercise **??**). This justifies a proof by Noetherian induction where the property $Q(a)$ is "a is confluent". Applying Noetherian induction, confluence holds for all $a' \in M$ with $a \rightarrow^+ a'$ and needs to be shown for a. Consider the confluence property for a: $b \ {}^*\!\!\leftarrow a \rightarrow^* c$. If $b = a$ or $c = a$ the proof is done. For otherwise, the situation can be expanded to $b \ {}^*\!\!\leftarrow b' \leftarrow a \rightarrow c' \rightarrow^* c$ as shown in Figure 1.5. By local confluence there is an a' with $b' \rightarrow^* a' \ {}^*\!\!\leftarrow c'$. Now b', c' are strictly smaller than a, they are confluent and hence can be rewritten to a single a'', finishing the proof (see Figure 1.5). ☐

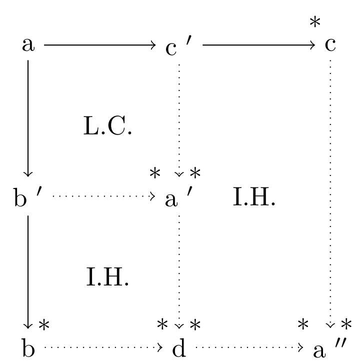

Figure 1.5: Proof of (ii) $\Rightarrow$ (i) of Newman's Lemma 1.6.6

Lemma 1.6.7. If $\rightarrow$ is confluent, then every element has at most one normal form.

Proof. Suppose that some element $a \in A$ has normal forms b and c, then $b \ {}^*\!\!\leftarrow a \rightarrow^* c$. If $\rightarrow$ is confluent, then $b \rightarrow^* d \ {}^*\!\!\leftarrow c$ for some $d \in A$. Since b and c are normal forms, both derivations must be empty, hence $b \rightarrow^0 d \ {}^0\!\!\leftarrow c$, so b, c, and d must be identical. ☐

Corollary 1.6.8. If $\rightarrow$ is normalizing and confluent, then every element b has a unique normal form.

Proposition 1.6.9. If $\rightarrow$ is normalizing and confluent, then $b \leftrightarrow^* c$ if and only if $b{\downarrow} = c{\downarrow}$.

Proof. Either using Theorem 1.6.5 or directly by induction on the length of the derivation of $b \leftrightarrow^* c$. ☐