1

INTRODUCTION

1.1 INTRODUCTION

Partial order and lattice theory play an important role in many disciplines of computer science and engineering. For example, they have applications in distributed computing (vector clocks and global predicate detection), concurrency theory (pomsets and occurrence nets), programming language semantics (fixed-point semantics), and data mining (concept analysis). They are also useful in other disciplines of mathematics such as combinatorics, number theory, and group theory.

This book differs from earlier books written on the subject in two aspects. First, this book takes a computational perspective—the emphasis is on algorithms and their complexity. While mathematicians generally identify necessary and sufficient conditions to characterize a property, this book focuses on efficient algorithms to test the property. As a result of this bias, much of the book concerns itself only with finite sets. Second, existing books do not dwell on applications of lattice theory. This book treats applications at par with the theory. In particular, I have given applications of lattice theory to distributed computing and combinatorics.

This chapter covers the basic definitions of partial orders.

Introduction to Lattice Theory with Computer Science Applications, First Edition. Vijay K. Garg.

1.2 RELATIONS

A partial order is simply a relation with certain properties. A **relation** R over any set X is a subset of $X \times X$. For example, let

$$X = \{a, b, c\}.$$

Then, one possible relation is

$$R = \{(a, c), (a, a), (b, c), (c, a)\}.$$

It is sometimes useful to visualize a relation as a graph on the vertex set X such that there is a directed edge from x to y iff $(x, y) \in R$. The graph corresponding to the relation R in the previous example is shown in Figure 1.1.

A relation is **reflexive** if for each $x \in X$, $(x, x) \in R$, i.e., each element of X is related to itself. In terms of a graph, this means that there is a self-loop on each node. If X is the set of natural numbers, $\mathcal{N}$, then "*x divides y*" is a reflexive relation. R is **irreflexive** if for each $x \in X$, $(x, x) \notin R$. In terms of a graph, this means that there are no self-loops. An example on the set of natural numbers, $\mathcal{N}$, is the relation "*x less than y*." Note that a relation may be neither reflexive nor irreflexive.

A relation R is **symmetric** if for all $x, y \in X$, $(x, y) \in R$ implies $(y, x) \in R$. An example of a symmetric relation on $\mathcal{N}$ is

$$R = \{(x, y) \mid x \; mod \; 5 = y \; mod \; 5\}. \tag{1.1}$$

A symmetric relation can be represented using an undirected graph. R is **antisymmetric** if for all $x, y \in X$, $(x, y) \in R$ and $(y, x) \in R$ implies $x = y$. For example, the relation *less than or equal to* defined on $\mathcal{N}$ is anti-symmetric. A relation R is **asymmetric** if for all $x, y \in X$, $(x, y) \in R$ implies $(y, x) \notin R$. The relation *less than* on $\mathcal{N}$ is asymmetric. Note that an asymmetric relation is always irreflexive.

A relation R is **transitive** if for all $x, y, z \in X$, $(x, y) \in R$ and $(y, z) \in R$ implies $(x, z) \in R$. The relations *less than* and *equal to* on $\mathcal{N}$ are transitive.

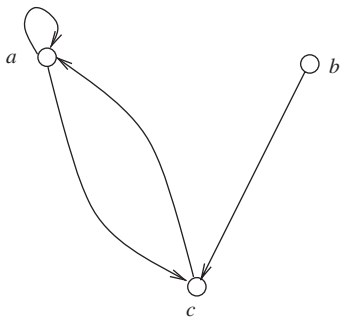

Figure 1.1 The graph of a relation.

A relation R is an **equivalence** relation if it is reflexive, symmetric, and transitive. When R is an equivalence relation, we use $x \equiv_R y$ (or simply $x \equiv y$ when R is clear from the context) to denote that $(x, y) \in R$. Furthermore, for each $x \in X$, we use $[x]_R$, called the **equivalence class of** x, to denote the set of all $y \in X$ such that $y \equiv_R x$. It can be seen that the set of all such equivalence classes forms a **partition** of X. The relation on $\mathcal{N}$ defined in (1.1) is an example of an equivalence relation. It partitions the set of natural numbers into five equivalence classes.

Given any relation R on a set X, we define its **irreflexive transitive closure**, denoted by R^+, as follows. For all $x, y \in X : (x, y) \in R^+$ iff there exists a sequence $x_0, x_1, ..., x_j, j \geq 1$ with $x_0 = x$ and $x_j = y$ such that

$$\forall i : 0 \leq i < j : (x_i, x_{i+1}) \in R.$$

Thus $(x, y) \in R^+$, iff there is a nonempty path from x to y in the graph of the relation R. We define the **reflexive transitive closure**, denoted by R^*, as

$$R^* = R^+ \cup \{(x, x) \mid x \in X\}.$$

Thus $(x, y) \in R^*$ iff y is reachable from x by taking a path with zero or more edges in the graph of the relation R.

1.3 PARTIAL ORDERS

A relation R is a **reflexive partial order** (or, a **nonstrict partial order**) if it is reflexive, antisymmetric, and transitive. The *divides* relation on the set of natural numbers is a reflexive partial order. A relation R is an **irreflexive partial order**, or a **strict partial order** if it is irreflexive and transitive. The *less than* relation on the set of natural numbers is an irreflexive partial order. When R is a reflexive partial order, we use $x \leq_R y$ (or simply $x \leq y$ when R is clear from the context) to denote that $(x, y) \in R$. A reflexive partially ordered set, **poset** for short, is denoted by $(X, \leq)$. When R is an irreflexive partial order, we use $x <_R y$ (or simply $x < y$ when R is clear from the context) to denote that $(x, y) \in R$. The set X together with the partial order is denoted by $(X, <)$. We use $P = (X, <)$ to denote a irreflexive poset defined on X.

The two versions of partial orders—reflexive and irreflexive—are essentially the same. Given an irreflexive partial order, we can define $x \leq y$ as $x < y$ or $x = y$, which gives us a reflexive partial order. Similarly, given a reflexive partial order $(X, \leq)$, we can define an irreflexive partial order $(X, <)$ by defining $x < y$ as $x \leq y$ and $x \neq y$.

A relation is a **total order** if R is a partial order and for all distinct $x, y \in X$, either $(x, y) \in R$ or $(y, x) \in R$. The natural order on the set of integers is a total order, but the *divides* relation is only a partial order.

Finite posets are often depicted graphically using **Hasse diagrams**. To define Hasse diagrams, we first define a relation **covers** as follows. For any two elements $x, y \in X$, y covers x if $x < y$ and $\forall z \in X : x \leq z < y$ implies $z = x$. In other words, if y covers x then there should not be any element z with $x < z < y$. We use $x <_c y$ to

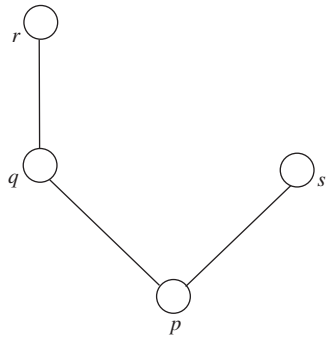

Figure 1.2 Hasse diagram.

denote that y covers x (or x is covered by y). We also say that y is an **upper cover** of x and x is a **lower cover** of y. A Hasse diagram of a poset is a graph with the property that there is an edge from x to y iff $x <_c y$. Furthermore, when drawing the graph on a Euclidean plane, x is drawn lower than y when y covers x. This allows us to suppress the directional arrows in the edges. For example, consider the following poset $(X, \leq)$,

$$X \stackrel{\text{def}}{=} \{p, q, r, s\}; \quad \leq \stackrel{\text{def}}{=} \{(p, q), (q, r), (p, r), (p, s)\}. \tag{1.2}$$

The corresponding Hasse diagram is shown in Figure 1.2. Note that we will sometimes use directed edges in Hasse diagrams if the context demands it. In general, in this book, we switch between the directed graph and undirected graph representations of Hasse diagrams.

Given a poset $(X, \leq_X)$ a **subposet** is simply a poset $(Y, \leq_Y)$, where $Y \subseteq X$, and

$$\forall x, y \in Y : x \leq_Y y \stackrel{\text{def}}{=} x \leq_X y.$$

Let $x, y \in X$ with $x \neq y$. If either $x < y$ or $y < x$, we say x and y are **comparable**. On the other hand, if neither $x < y$ nor $x > y$, then we say x and y are incomparable and write $x||y$. A poset $(Y, \leq)$ (or a subposet $(Y, \leq)$ of $(X, \leq)$) is called a **chain** if every distinct pair of elements from Y is comparable. Similarly, we call a poset an **antichain** if every distinct pair of elements from Y is incomparable. For example, for the poset represented in Figure 1.2, $\{p, q, r\}$ is a chain, and $\{q, s\}$ is an antichain.

A chain C of a poset $(X, \leq)$ is a **longest chain** if no other chain contains more elements than C. We use a similar definition for the **largest antichain**. The **height** of the poset is the number of elements in a longest chain, and the **width** of the poset is the number of elements in a largest antichain. For example, the poset in Figure 1.2 has height equal to 3 (the longest chain is $\{p, q, r\}$) and width equal to 2 (a largest antichain is $\{q, s\}$).

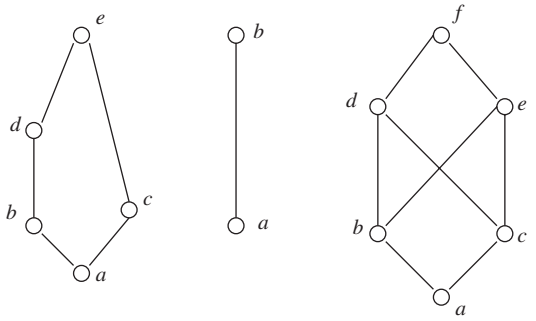

Figure 1.3 Only the first two posets are lattices.

Generalizing the notation for intervals on the real-line, we define an **interval** $[x, y]$ in a poset $(X, \leq)$ as

$$\{z \mid x \leq z \leq y\}.$$

The meanings of (x, y), $[x, y)$ and $(x, y]$ are similar. A poset is **locally finite** if all intervals are finite. Most posets in this book will be locally finite if not finite.

A poset is **well-founded** iff it has no infinite decreasing chain. The set of natural numbers under the usual $\leq$ relation is well-founded but the set of integers is not well-founded.

Poset $Q = (X, \leq_Q)$ **extends** the poset $P = (X, \leq_P)$ if

$$\forall x, y \in X : x \leq_P y \Rightarrow x \leq_Q y.$$

If Q is a total order, then we call Q a **linear extension** of P. For example, for the poset $P = (X, \leq)$ defined in Figure 1.2, a possible linear extension Q is

$$X \stackrel{\text{def}}{=} \{p, q, r, s\}; \quad \leq_Q \stackrel{\text{def}}{=} \{(p, q), (q, r), (p, r), (p, s), (q, s), (r, s)\}.$$

We now give some special posets that will be used as examples in the book.

- $\underline{n}$ denotes a poset which is a chain of length n. The second poset in Figure 1.3 is $\underline{2}$.
- We use A_n to denote the poset of n incomparable elements.

1.4 JOIN AND MEET OPERATIONS

We now define two operators on subsets of the set X—**meet** and **join**. The operator meet is also called **infimum** (or **inf**). Similarly, the operator join is also called **supremum** (or **sup**).

Let $Y \subseteq X$, where $(X, \leq)$ is a poset. For any $m \in X$, we say that $m = \inf Y$ iff

1. $\forall y \in Y : m \leq y$ and
2. $\forall m' \in X : (\forall y \in Y : m' \leq y) \Rightarrow m' \leq m$.

The condition (1) says that m is a lower bound of the set Y. The condition (2) says that if m' is any lower bound of Y, then it is less than or equal to m. For this reason, m is also called the **greatest lower bound** (*glb*) of the set Y. It is easy to check that the infimum of Y is unique whenever it exists. Observe that m is not required to be an element of Y.

The definition of sup is similar. For any $s \in X$, we say that $s = \sup\ Y$ iff

1. $\forall y \in Y : y \leq s$ and
2. $\forall s' \in X : (\forall y \in Y : y \leq s') \Rightarrow s \leq s'$.

Again, s is also called the **least upper bound** (*lub*) of the set Y. We denote the **glb** of $\{a, b\}$ by $a \sqcap b$ and **lub** of $\{a, b\}$ by $a \sqcup b$. In the set of natural numbers ordered by the *divides* relation, the *glb* corresponds to finding the greatest common divisor (gcd) and the *lub* corresponds to finding the least common multiple (lcm) of two natural numbers. The glb or the lub may not always exist. In Figure 1.2, the set $\{q, s\}$ does not have any upper bound. In the third poset in Figure 1.3, the set $\{b, c\}$ does not have any least upper bound (although both d and e are upper bounds).

The following lemma relates $\leq$ to the meet and join operators.

Lemma 1.1 (Connecting Lemma) *For all elements x, y of a poset,*

1. $x \leq y \equiv (x \sqcup y) = y$ *and*
2. $x \leq y \equiv (x \sqcap y) = x$.

Proof: For the first part, note that $x \leq y$ implies that y is an upper bound on $\{x, y\}$. It is also the lub because any upper bound of $\{x, y\}$ is greater than both x and y. Therefore, $(x \sqcup y) = y$. Conversely, $(x \sqcup y) = y$ means y is an upper bound on $\{x, y\}$. Therefore, $x \leq y$.

The proof for the second part is the dual of this proof.

■

Lattices, which are special kinds of posets, can be defined using the join and meet operators as shown in the following discussion.

Definition 1.2 (Lattice) *A poset $(X, \leq)$ is a* **lattice** *iff $\forall x, y \in X : x \sqcup y$ and $x \sqcap y$ exist.*

The first two posets in Figure 1.3 are lattices, whereas the third one is not.

If $\forall x, y \in X : x \sqcup y$ exists, then we call it a *sup semilattice*. If $\forall x, y \in X : x \sqcap y$ exists, then we call it an *inf semilattice*.

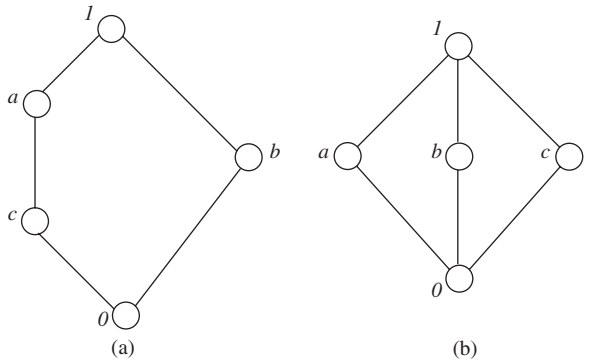

Figure 1.4 (a) Pentagon(N_5) and (b) diamond(M_3).

In our definition of lattice, we have required existence of *lub*'s and *glb*'s for sets of size two. This is equivalent to the requirement of existence of *lub*'s and *glb*'s for sets of finite size by using induction. A poset may be a lattice, but it may have sets of infinite size for which lub/glb may not exist. A simple example is that of the set of natural numbers. There is no lub of set of even numbers, even though glb and lub exist as min and max for any finite set. Another example is the set of rational numbers. For a finite subset of rational numbers, we can easily determine lubs(and glbs). However, consider the set $\{x \mid x \leq \sqrt{2}\}$. The lub of this set is $\sqrt{2}$, which is not a rational number. A lattice for which any set has lub and glb defined is called a *complete lattice*. An example of a complete lattice is the set of real numbers extended with $+\infty$ and $-\infty$.

Definition 1.3 (Distributive Lattice) *A lattice L is* **distributive** *if*

$$\forall a, b, c \in L : a \sqcap (b \sqcup c) = (a \sqcap b) \sqcup (a \sqcap c).$$

It is easy to verify that the above-mentioned condition is equivalent to

$$\forall a, b, c \in L : a \sqcup (b \sqcap c) = (a \sqcup b) \sqcap (a \sqcup c).$$

Thus, in a distributive lattice, $\sqcup$ and $\sqcap$ operators distribute over each other.

Any power-set lattice is distributive. The lattice of natural numbers with $\leq$ defined as the relation *divides* is also distributive. Some examples of nondistributive lattices are diamond (M_3) and pentagon (N_5) shown in Figure 1.4.

1.5 OPERATIONS ON POSETS

Given any set of structures, it is useful to consider ways of composing them to obtain new structures.

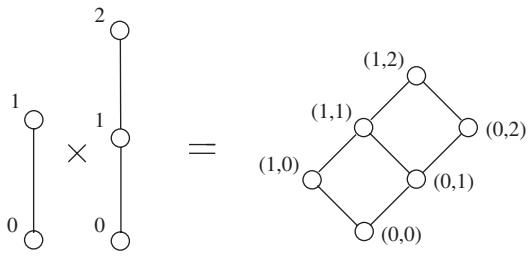

Figure 1.5 Cross product of posets.

Definition 1.4 (Disjoint Sum) *Given two posets P and Q, their* **disjoint sum**, *denoted by $P + Q$, is defined as follows. Given any two elements x and y in $P + Q$, $x \leq y$ iff*

(1) both x and y belong to P and $x \leq y$ in P or
(2) both x and y belong to Q and $x \leq y$ in Q.

The Hasse diagram of the disjoint sum of P and Q can be computed by simply placing the Hasse diagram of P next to Q.

Definition 1.5 (Cross Product of Posets) *Given two posets P and Q, the cross product denoted by $P \times Q$ is defined as*

$$(P \times Q, \leq_{P \times Q})$$

where

$$(p_1, q_1) \leq (p_2, q_2) \stackrel{\text{def}}{=} (p_1 \leq_P p_2) \wedge (q_1 \leq_Q q_2).$$

See Figure 1.5 for an example. The definition can be extended to an arbitrary indexing set.

Definition 1.6 (Linear Sum) *Given two posets P and Q, the* **linear sum** *(or ordinal sum) denoted by $P \oplus Q$ is defined as*

$$x \leq_{P \oplus Q} y \text{ iff } (x \leq_P y) \vee (x \leq_Q y) \vee [(x \in P) \wedge (y \in Q)].$$

1.6 IDEALS AND FILTERS

Let $(X, \leq)$ be any poset. We call a subset $Y \subseteq X$ a **down-set** if

$$\forall y, z \in X : z \in Y \wedge y \leq z \Rightarrow y \in Y.$$

Down-sets are also called **order ideals**. It is easy to see that for any x, $D[x]$ defined below is an order ideal. Such order ideals are called **principal order ideals**,

$$D[x] = \{y \in X \mid y \leq x\}.$$

For example, in Figure 1.2, $D[r] = \{p, q, r\}$. Similarly, we call $Y \subseteq X$ an **up-set** if

$$y \in Y \wedge y \leq z \Rightarrow z \in Y.$$

Up-sets are also called **order filters**. We also use the notation below to denote **principal order filters**

$$U[x] = \{y \in X | x \leq y\}.$$

In Figure 1.2, $U[p] = \{p, q, r, s\}$.

The following lemma provides a convenient relationship between principal order filters and other operators defined earlier.

Lemma 1.7 *1.* $x \leq y \equiv U[y] \subseteq U[x]$

2. $x = \sup Y \equiv U[x] = \cap_{y \in Y} U[y]$.

Proof: Left as an exercise.

■

In some applications, the following notation is also useful:

$$U(x) = \{y \in X | x < y\}$$

$$D(x) = \{y \in X | y < x\}.$$

We will call $U(x)$ the **upper-holdings** of x, and $D(x)$, the **lower-holdings** of x. We can extend the definitions of $D[x], D(x), U[x], U(x)$ to sets of elements, A. For example,

$$U[A] = \{y \in X | \exists x \in A : x \leq y\}.$$

1.7 SPECIAL ELEMENTS IN POSETS

We define some special elements of posets such as bottom and top elements, and minimal and maximal elements.

Definition 1.8 (Bottom Element) *An element x is a* **bottom element** *or a* **minimum** *element of a poset P if $x \in P$, and*

$$\forall y \in P : x \leq y.$$

For example, 0 is the bottom element in the poset of whole numbers and $\emptyset$ is the bottom element in the poset of all subsets of a given set W. Similarly, an element x is a **top element**, or a **maximum** element of a poset P if $x \in P$, and

$$\forall y \in P : y \leq x.$$

A bottom element of the poset is denoted by $\perp$ and the top element by $\top$. It is easy to verify that if bottom and top elements exist, they are unique.

Definition 1.9 (Minimal and Maximal Elements) *An element x is a* **minimal** *element of a poset P if*

$$\forall y \in P : y \not< x.$$

The minimum element is also a minimal element. However, a poset may have more than one minimal element. Similarly, an element x is a **maximal** *element of a poset P if*

$$\forall y \in P : y \not> x.$$

1.8 IRREDUCIBLE ELEMENTS

Definition 1.10 (Join-irreducible) *An element x is* **join-irreducible** *in P if it cannot be expressed as a join of other elements of P. Formally, x is join-irreducible if*

$$\forall Y \subseteq P : x = \sup Y \Rightarrow x \in Y.$$

Note that if a poset has $\perp$, then $\perp$ is not join-irreducible because when $Y = \{\}$, $\sup Y$ is $\perp$. The set of join-irreducible elements of a poset P will be denoted by $J(P)$. The first poset in Figure 1.3 has b,c and d as join-irreducible. The element e is not join-irreducible in the first poset because $e = b \sqcup c$. The second poset has b, and the third poset has b, c, d, and e as join-irreducible. It is easy to verify that $x \notin J(P)$ is equivalent to $x = \sup D(x)$. In a chain, all elements other than the bottom are join-irreducible.

By duality, we can define **meet-irreducible** elements of a poset, denoted by $M(P)$.

Loosely speaking, the set of (join)-irreducible elements forms a basis of the poset because all elements can be expressed using these elements. More formally,

Theorem 1.11 *Let P be a finite poset. Then, for any $x \in P$*

$$x = \sup(D[x] \cap J(P)).$$

Proof: Left as an exercise. ■

An equivalent characterization of join-irreducible elements is as follows.

Theorem 1.12 *For a finite poset P, $x \in J(P)$ iff $\exists y \in P : x \in minimal(P - D[y])$.*

Proof: First assume that $x \in J(P)$.

Let $LC(x)$ be the set of elements covered by x. If $LC(x)$ is singleton, then choose that element as y. It is clear that $x \in minimal(P - D[y])$.

Now consider the case when $LC(x)$ is not singleton (it is empty or has more than one element). Let Q be the set of upper bounds for $LC(x)$. Q is not empty because $x \in Q$. Further, x is not the minimum element in Q because x is join-irreducible. Pick any element $y \in Q$ that is incomparable to x. Since $D[y]$ includes $LC(x)$ and not x, we get that x is minimal in $P - D[y]$.

The converse is left as an exercise. ■

1.9 DISSECTOR ELEMENTS

We now define a subset of irreducibles called dissectors.

Definition 1.13 (Upper Dissector) *For a poset P, $x \in P$ is an* **upper dissector** *if there exists $y \in P$ such that*

$$P - U[x] = D[y].$$

In the above-mentioned definition, y is called a **lower dissector**. We will use dissectors to mean upper dissectors.

A dissector x decomposes the poset into two parts $U[x]$ and $D[y]$ for some y. In the first poset in Figure 1.3, b is an upper dissector because $P - U[b] = P - \{b, d, e\} = \{a, c\} = D[c]$. However, d is not an upper dissector because $P - U[d] = \{a, b, c\}$, which is not a principal ideal.

The following result is an easy implication of Theorem 1.12.

Theorem 1.14 *x is a dissector implies that x is join-irreducible.*

Proof: If x is an upper dissector, then there exists y such that x is minimum in $P - D[y]$. This implies that x is minimal in $P - D[y]$.

■

1.10 APPLICATIONS: DISTRIBUTED COMPUTATIONS

As mentioned earlier, posets have a wide range of applications. In this book, we will focus on applications to distributed computing and combinatorics.

Partial order play an important role in distributed computing because a distributed computation is most profitably modeled as a partially ordered set of events. A distributed program consists of two or more processes running on different processors and communicating via messages. We will be concerned with a single computation of such a distributed program. Each process P_i in that computation generates a sequence of *events*. There are three types of events—*internal*, *send*, and *receive*. It is clear how to order events within a single process. If event e occurred before f in the process, then e is ordered before f. How do we order events across processes? If e is the send event of a message and f is the receive event of the same message, then we can order e before f. Combining these two ideas, we obtain the following definition.

Definition 1.15 (Happened-Before Relation) *The* **happened-before** *relation ($\rightarrow$) on a set of events E is the smallest relation on E that satisfies*

1. *If event e occurred before event f in the same process, then $e \rightarrow f$,*
2. *If e is the send event of a message and f is the receive event of the same message, then $e \rightarrow f$, and*
3. *If there exists an event g such that $(e \rightarrow g)$ and $(g \rightarrow f)$, then $(e \rightarrow f)$.*

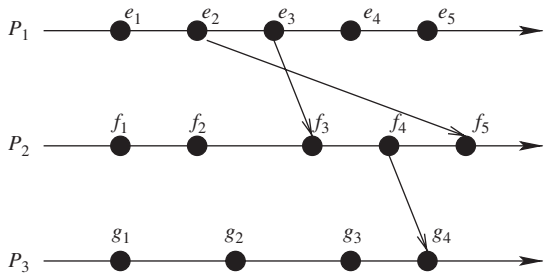

Figure 1.6 A computation in the happened-before model.

Formally, a *computation* in the happened-before model is defined as a tuple $(E, \rightarrow)$, where E is the set of all events and $\rightarrow$ is a partial order on events in E. Figure 1.6 illustrates a computation where $e_2 \rightarrow e_4$, $e_3 \rightarrow f_3$, and $e_1 \rightarrow g_4$.

Since a computation is a poset, all the concepts that we have defined for posets are applicable to distributed computations. For example, in distributed computing, the notion of a consistent global state is defined as follows.

Definition 1.16 *A* **consistent global state** *of a distributed computation* $(E, \rightarrow)$ *is any subset* $G \subseteq E$ *such that*

$$\forall e, f \in E : (f \in G) \wedge (e \rightarrow f) \Rightarrow (e \in G)$$

As the reader can verify, there is one-to-one correspondence between consistent global states and down-sets of a poset. Therefore, many results in lattice theory carry over to distributed computing. For example, the set of all down-sets of a poset forms a distributive lattice implies that the set of all consistent global states forms a distributive lattice.

1.11 APPLICATIONS: COMBINATORICS

We now discuss an example from combinatorics. Integer partitions have been studied in combinatorics for centuries. We show how we can use lattices to model sets of integer partitions.

Definition 1.17 (Partition) *An integer sequence* $\lambda = (\lambda_1, ..., \lambda_k)$ *is a* **partition** *of the integer* n *if*

1. $\forall i \in [1, k-1] : \lambda_i \geq \lambda_{i+1}$
2. $\sum_{i=1}^{k} \lambda_i = n$
3. $\forall i : \lambda_i \geq 1$.

For example, $(5, 2, 2)$ and $(4, 3, 2)$ are two of the partitions of 9. Integer partitions are conveniently visualized using Ferrer's diagrams.

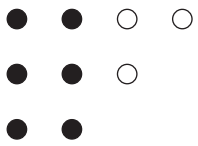

Figure 1.7 Ferrer's diagram for $(4, 3, 2)$ shown to contain $(2, 2, 2)$.

Definition 1.18 (Ferrer's Diagram) *A* **Ferrer's diagram** *for an integer partition $\lambda = (\lambda_1, ..., \lambda_k)$ of integer n is a matrix of dots where the ith row contains λ_i dots. Thus, row i represents the ith part and the number of rows represents the number of parts in the partition.*

We define an order on the partitions as follows: Given two partitions $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_m), \delta = (\delta_1, \delta_2, \dots, \delta_n)$, we say that $\lambda \geq \delta$ iff $m \geq n$ and $\forall i : 1 \leq i \leq n, \lambda_i \geq \delta_i$. This can also be viewed in terms of containment in the Ferrer's diagram, i.e. $\lambda \geq \delta$ if the Ferrer's diagram for δ is contained in the Ferrer's diagram of λ. For example, consider the partitions $(4, 3, 2)$ and $(2, 2, 2)$. The Ferrer's diagram for $(2, 2, 2)$ is contained in the Ferrer's diagram for $(4, 3, 2)$ as shown in Figure 1.7. Hence, $(4, 3, 2) \geq (2, 2, 2)$.

Definition 1.19 (Young's Lattice) *Given a partition λ,* **Young's lattice** *Y_λ is the poset of all partitions that are less than or equal to λ.*

The Young's lattice for $(3, 3, 3)$ is shown in Figure 1.8. Note that partitions less than a given partition are not necessarily partitions of the same integer.

Again, it follows easily from lattice theory that the Young's lattice is distributive. Furthermore, assume that we are interested in only those partitions in Y_λ which have distinct parts. The notion of slicing posets (explained in Chapter 10 can be used to analyze such subsets of integer partitions.

1.12 NOTATION AND PROOF FORMAT

We use the following notation for quantified expressions:

(*op* free-var-list : range-of-free-vars : expression)

where *op* is a universal or an existential quantifier, free-var-list is the list of variables over which the quantification is made, and the range-of-free-vars is the range of the variables. For example, $(\forall i : 0 \leq i \leq 10 : i^2 \leq 100)$ means that for all i such that $0 \leq i \leq 10$, $i^2 \leq 100$ holds. If the range of the variables is clear from the context, then we simply use

(*op* free-var-list : expression).

For example, if it is clear that i and j are integers, then we may write

$$\forall i : (\exists j : j > i).$$

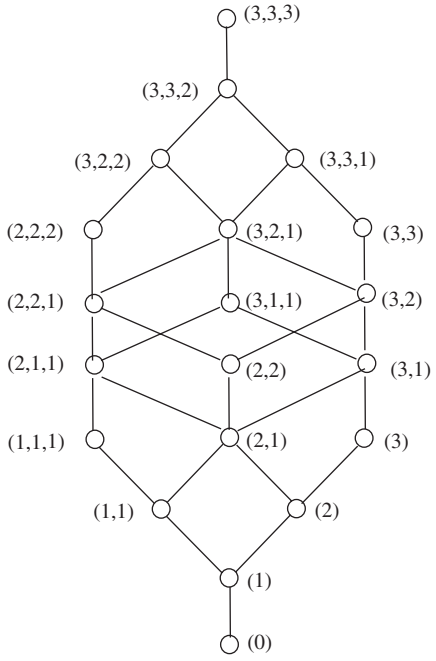

Figure 1.8 Young's lattice for $(3, 3, 3)$.

We use a calculational style of proofs for many of our theorems. For example, a proof that $[A \equiv C]$ is rendered in our format as

$$
\begin{array}{ll}
 & A \\
\equiv & \quad \{ \text{ hint why } [A \equiv B] \ \} \\
 & B \\
\equiv & \quad \{ \text{ hint why } [B \equiv C] \ \} \\
 & C.
\end{array}
$$

We use implication ($\Rightarrow$) instead of equivalence when proving $A \Rightarrow C$.

A predicate with free variables is assumed to be universally quantified for all possible values of free variables. We use the usual convention for binding powers of operators. In order of increasing binding powers, the operators are as follows:

$\equiv$
$\Rightarrow$
$\vee, \wedge$
$\neg$
$=, \neq, <, \leq$, and other relational operators over integers and sets,
arithmetic operators, and
function applications.

Operators that appear on the same line have the same binding power, and we use parentheses to show the order of application. We sometimes omit parentheses in

expressions when they use operators from different lines. Thus

$$x \leq y \wedge y \leq z \Rightarrow x \leq z$$

is equivalent to

$$\forall x, y, z : (((x \leq y) \wedge (y \leq z)) \Rightarrow (x \leq z)).$$

1.13 PROBLEMS

1.1. Give an example of a nonempty binary relation that is symmetric and transitive but not reflexive.

1.2. The *transitive closure* of a relation R on a finite set can also be defined as the smallest transitive relation on S that contains R. Show that the transitive closure is uniquely defined. We use "smaller" in the sense that R_1 is smaller than R_2 if $|R_1| < |R_2|$.

1.3. Show that if P and Q are posets defined on set X, then so is $P \cap Q$.

1.4. Draw the Hasse diagram of all natural numbers less than 10 ordered by the relation *divides*.

1.5. Show that if C_1 and C_2 are down-sets for any poset $(E, <)$, then so is $C_1 \cap C_2$.

1.6. Show the following properties of $\sqcup$ and $\sqcap$.

$$(L1) \quad a \sqcup (b \sqcup c) = (a \sqcup b) \sqcup c \qquad \text{(associativity)}$$

$$(L1)^\delta \quad a \sqcap (b \sqcap c) = (a \sqcap b) \sqcap c$$

$$(L2) \quad a \sqcup b = b \sqcup a \qquad \text{(commutativity)}$$

$$(L2)^\delta \quad a \sqcap b = b \sqcap a$$

$$(L3) \quad a \sqcup (a \sqcap b) = a \qquad \text{(absorption)}$$

$$(L3)^\delta \quad a \sqcap (a \sqcup b) = a$$

1.7. Prove Lemma 1.7.

1.8. Prove Theorem 1.11.

1.9. Complete the proof of Theorem 1.12.

1.14 BIBLIOGRAPHIC REMARKS

The first book on lattice theory was written by Garrett Birkhoff [Bir40, Bir48, Bir67]. The reader will find a discussion of the origins of lattice theory and an extensive bibliography in the book by Gratzer [Grä71, Grä03]. The book by Davey and Priestley

[DP90] provides an easily accessible account of the field. Some other recent books on lattice theory or ordered sets include books by Caspard, Leclerc, and Monjardet [CLM12], and Roman [Rom08]. The happened-before relation on the set of events in a distributed computation is due to Lamport [Lam78]. Reading [Rea02] gives a detailed discussion of dissectors and their properties. The proof format adopted for many theorems in this book is taken from Dijkstra and Scholten [DS90].