Appendix B.

Sets, Etc.

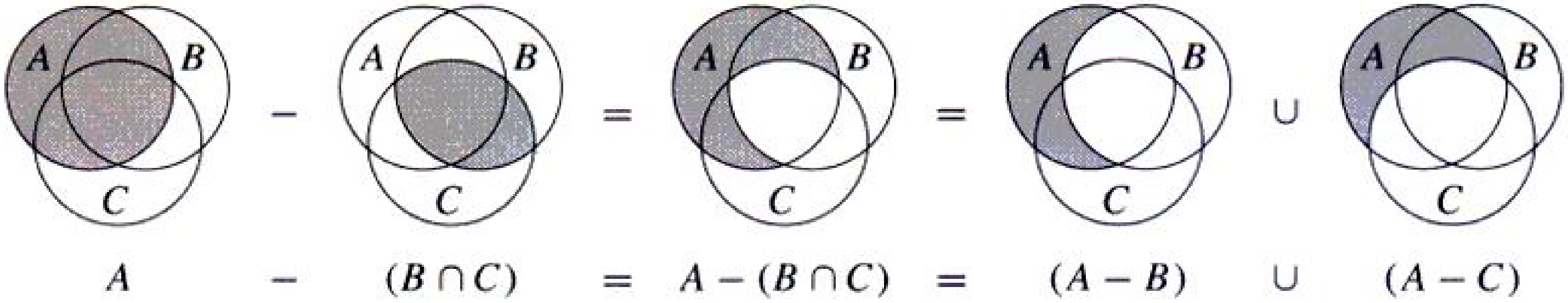

Figure B.1 A Venn diagram illustrating the first of DeMorgan's laws (B.2). Each of the sets *A, B,* and *C* is represented as a circle.

Graphs

- A ***directed graph*** (or digraph) $\boldsymbol{G}$ is a pair (V, E), where

  V is a finite set: the ***vertex set*** of G, i.e. a finite set of vertices

  and E is a binary relation on V: the ***edge set*** of G, i.e. a set of edges.

- An ***undirected graph*** $G=(V, E)$ is a pair (V, E), where

  V is the vertex set of G and

  E is the edge set of G which consists of unordered pairs of vertices.

  i.e. $e=\{u, v\} \in E$, where $u, v \in V$ and $u \neq v$. So, $(u, v)=(v, u)$.

- If $(u, v) \in E$ in $G=(V, E)$, a vertex v is adjacent to vertex u: $u \rightarrow v$
- When the graph is undirected, the adjacency relation is symmetric.
- When the graph is directed, the adjacency relation is not necessarily symmetric: $u \leftrightarrow v$
- If $(u,v) \in E$ in a directed graph $G=(V,E)$,
  - (u,v) is incident from or leaves vertex u and is incident to or enters vertex v.
  - If $(u,v) \in E$ in a undirected graph $G=(V,E)$, (u,v) is *incident on* vertices u *and* v.

.. continued

- In an undirected graph,
  - The ***degree of a vertex*** is the # of edges incident on it.
  - A vertex is isolated if its degree is 0.
- In a directed graph,
  - the ***out-degree*** of a vertex is the # of edges leaving it, and
  - the ***in-degree*** of a vertex is the # of edges entering it.
  - The ***degree*** of a vertex is its in-degree + its out-degree.
- A ***path*** of a ***length*** k from a vertex u to a vertex u' in a graph $G=(V, E)$ is

  a sequence $<v_0, v_1, v_2, \ldots, v_k>$ of vertices such that $u=v_0$, $u'=v_k$, and $(v_{i-1}, v_i) \in E$.
- If there is a path p from u to u', u' is ***reachable*** from u via p, if G is directed.
- An undirected graph is ***connected*** if every pair of vertices is connected by a path.
- The ***connected components*** of a graph are the equivalence classes of vertices under the 'is reachable from' relation.
- A directed graph is ***strongly connected*** if every two vertices are reachable from each other.
- The ***strongly connected components*** of a directed graph are the equivalence classes of vertices under the mutually reachable relation.

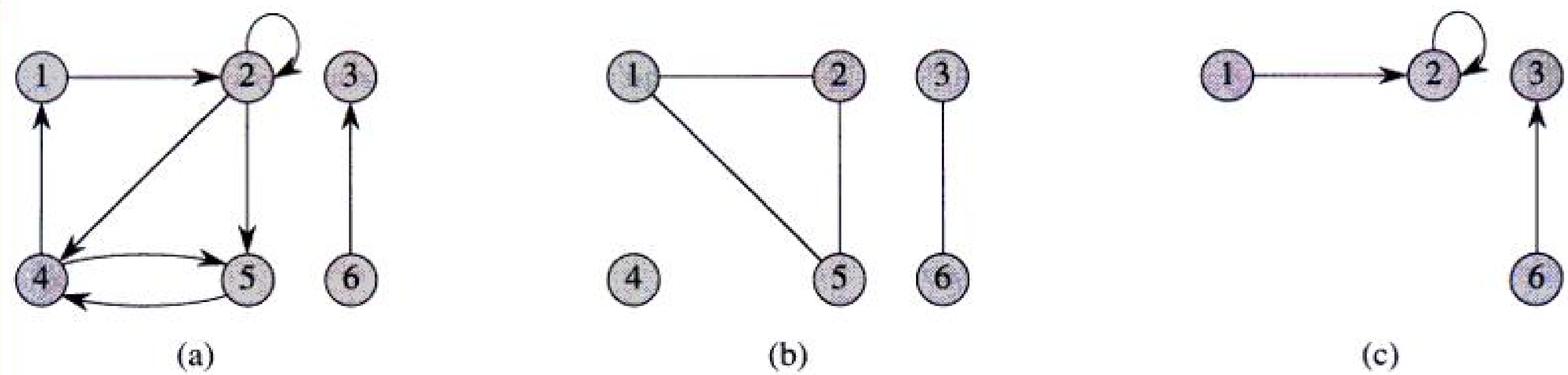

Figure B.2 Directed and undirected graphs.

- A ***directed graph*** $G = (V, E)$, where $V = \{1,2,3,4,5,6\}$

  and $E = \{(1,2), (2,2), (2,4), (2,5), (4,1), (4,5), (5,4), (6,3)\}$.

  The edge (2,2) is a self-loop.

(b) An ***undirected graph*** $G = (V,E)$, where $V = \{1,2,3,4,5,6\}$

and $E = \{(1,2), (1,5), (2,5), (3,6)\}$. The vertex 4 is isolated.

(c) The ***subgraph*** of the graph in part (a) induced by the vertex set {1,2,3, 6}.

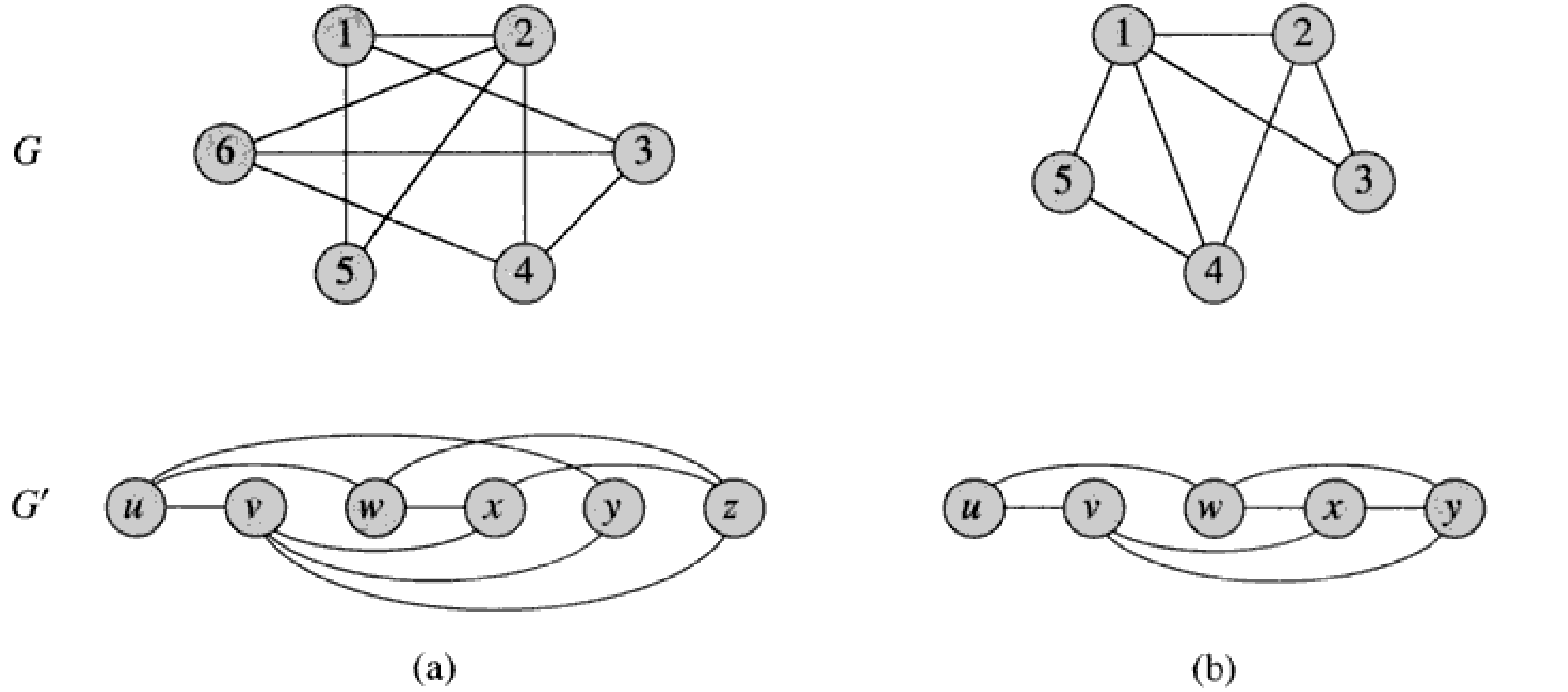

Figure B.3 **(a)** A pair of isomorphic graphs. The vertices of the top graph are mapped to the vertices of the bottom graph by $f(1) = u$, $f(2) = v$, $f(3) = w$, $f(4) = x$, $f(5) = y$, $f(6) = z$. **(b)** Two graphs that are not isomorphic, since the top graph has a vertex of degree 4 and the bottom graph does not.

Trees

(Free) *Trees*

A (free) ***tree*** is a connected, acyclic, undirected graph.

A ***forest*** is a graph which is acyclic and undirected, but possibly disconnected.

Theorem: Properties of Free Trees

Let G=(V, E) be an undirected graph.

The following statements are equivalent.

1. G is a free tree.
2. Any two vertices in G are connected by a unique simple path.
3. G is connected, but if any edge is removed from E, the resulting graph is disconnected.
4. G is connected, and $|E| = |V| - 1$.
5. G is acyclic, and $|E| = |V| - 1$.
6. G is acyclic, but if any edge is added to E, the resulting graph contains a cycle.

Trees

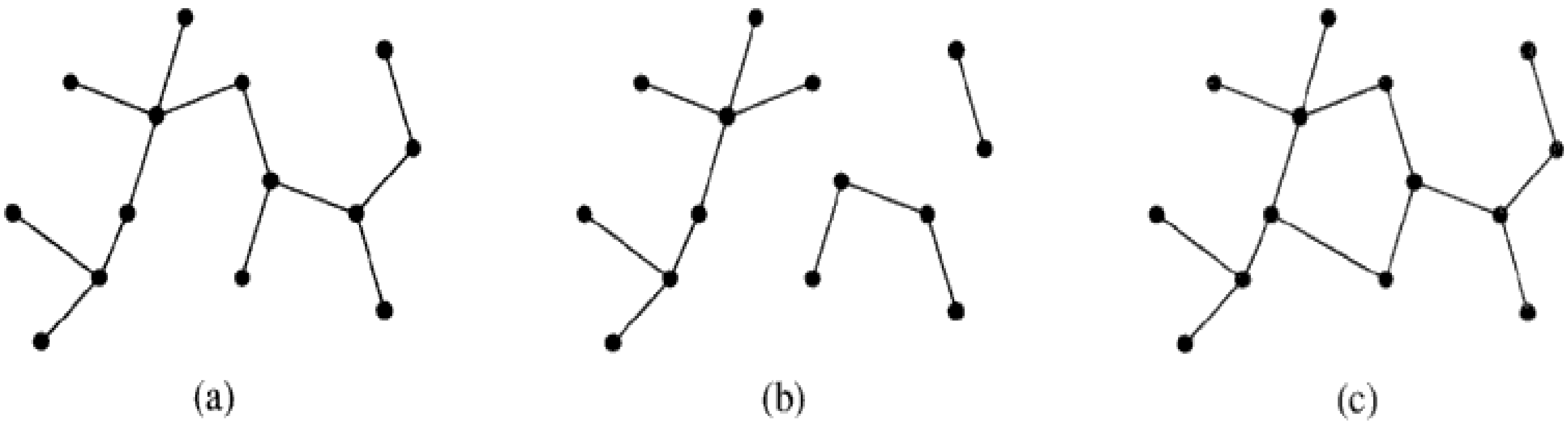

Figure B.4 **(a)** A free tree. **(b)** A forest. **(c)** A graph that contains a cycle and is therefore neither a tree nor a forest.

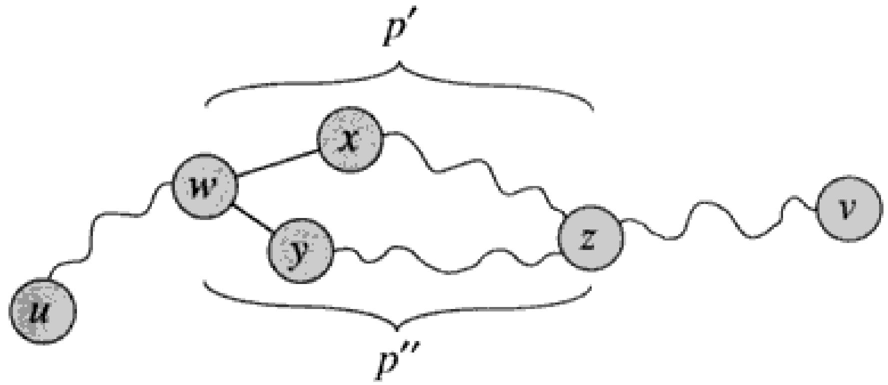

Figure B.5 A step in the proof of Theorem B.2: if (1) G is a free tree, then (2) any two vertices in G are connected by a unique simple path. Assume for the sake of contradiction that vertices u and v are connected by two distinct simple paths p_1 and p_2. These paths first diverge at vertex w, and they first reconverge at vertex z. The path p' concatenated with the reverse of the path p'' forms a cycle, which yields the contradiction.

Rooted and Ordered Trees

A ***rooted tree*** is a free tree in which one of the vertices is distinguished from the others.

The ***root***: the distinguished vertex.

A node: a vertex of a rooted tree.

A node x in a rooted tree T with root r.

An ***ancestor*** of x: any node y on the unique path from r to x.

x is a ***descendant*** of y if y is an ancestor of x.

The ***subtree*** rooted at x is the tree induced by descendants of x, rooted at x.

y is the ***parent*** of x and x is a ***child*** of y if the last edge on the path from the root r of a tree T to a node x is (y, x): the root has no parent.

Two nodes are ***siblings*** if they have the same parent.

A ***leaf*** or an ***external node*** is a node with no children.

An ***internal*** node is a nonleaf node.

.. continued

The ***degree*** of a node x is the number of children of x in T.

The ***depth*** of x in T is the length of the path from the root r to a node x.

The ***height*** of a node in a tree is the number of edges on the longest simple downward path from the node to a leaf, and is the height of its root.

An ***ordered tree*** is a rooted tree in which the children of each node are ordered.

I.e. If a node has k children, then there is a 1st child, a 2nd child, .., and a kth child.

Rooted and Ordered Trees

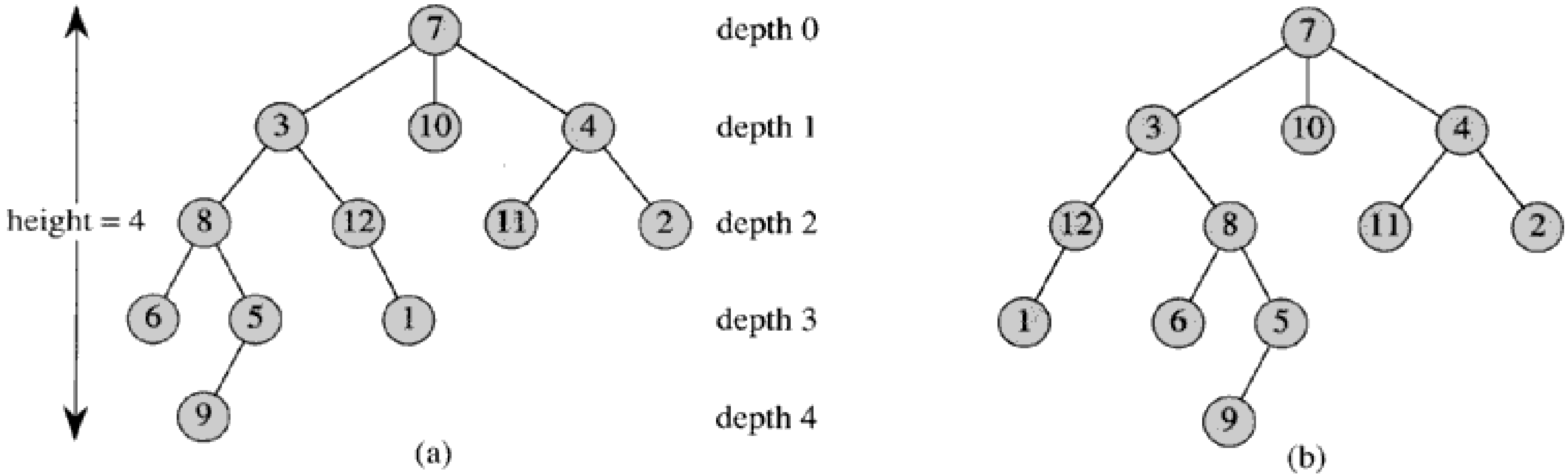

Figure B.6 Rooted and ordered trees. **(a)** A rooted tree with height 4. The tree is drawn in a standard way: the root (node 7) is at the top, its children (nodes with depth 1) are beneath it, their children (nodes with depth 2) are beneath them, and so forth. If the tree is ordered, the relative left-to-right order of the children of a node matters; otherwise it doesn't. **(b)** Another rooted tree. As a rooted tree, it is identical to the tree in (a), but as an ordered tree it is different, since the children of node 3 appear in a different order.

Binary and Positional Trees

A ***binary tree T*** is a tree which is

either ***empty*** (i.e. no nodes),

or consists of a node, called the ***root***,

and two distinct binary trees,

called its ***left subtree****(TL)* and its ***right subtree****(TR)* of T.

The empty tree(or null tree): the binary tree which contains no nodes (NIL).

The root of nonnull left (or right) subtree is called

the ***left*** ***(****or* ***right)*** ***child*** of the root of the entire tree.

-- the position of the child matters in a binary tree.

A ***full binary tree*** is a binary tree in which each node is either a leaf or has degree exactly 2.

.. continued

A ***k*-ary** tree is a binary tree in which every node has at most k children.

A complete k-ary tree is a k-ary tree in which all leaves has the same depth and all internodes have degree k.

The number of internal nodes of a complete k-ary tree of height h is:

$$1+k+k^2+\ldots+k^{h-1}=\sum_{i=0}^{h-1}k^i=\frac{k^h-1}{k-1}$$

Thus, the number of internal nodes of a complete binary tree is

$$2^h-1.$$

Binary Trees

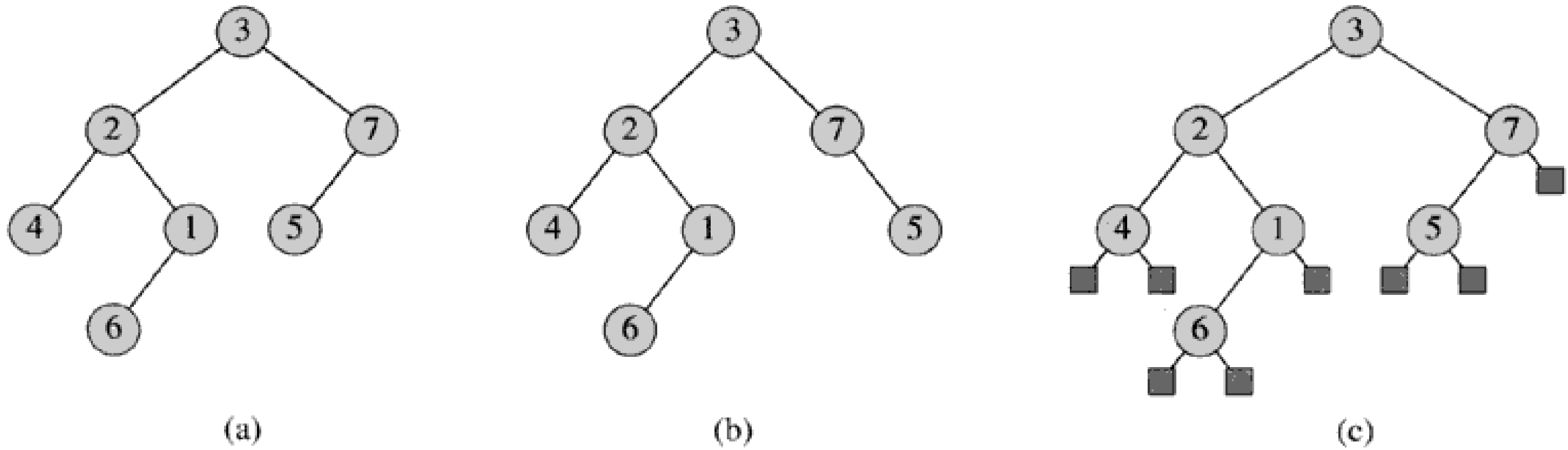

Figure B.7 Binary trees. **(a)** A binary tree drawn in a standard way. The left child of a node is drawn beneath the node and to the left. The right child is drawn beneath and to the right. **(b)** A binary tree different from the one in (a). In (a), the left child of node 7 is 5 and the right child is absent. In (b), the left child of node 7 is absent and the right child is 5. As ordered trees, these trees are the same, but as binary trees, they are distinct. **(c)** The binary tree in (a) represented by the internal nodes of a full binary tree: an ordered tree in which each internal node has degree 2. The leaves in the tree are shown as squares.

Example: A Complete Binary Tree

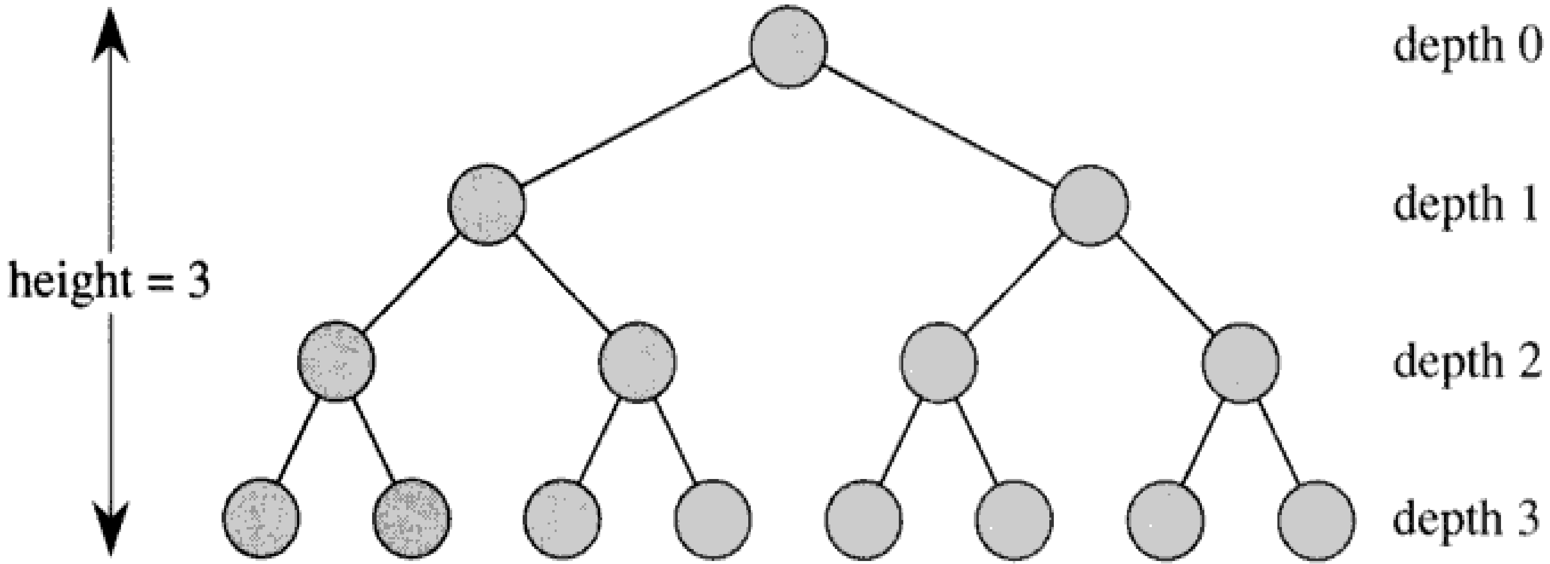

Figure B.8 A complete binary tree of height 3 with 8 leaves and 7 internal nodes.