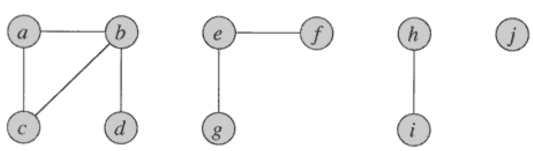
Introduction to Algorithms
Second Edition
by
Cormen, Leiserson, Rivest & Stein

Chapter 21



(a)

Edge processed	Collection of disjoint sets								
initial sets	$\{a\}$	$\{b\}$	$\{c\}$	$\{d\}$	$\{e\}$	$\{f\}$	$\{g\}$	$\{h\}$	$\{i\}$	$\{j\}$
(b,d)	$\{a\}$	$\{b,d\}$	$\{c\}$		$\{e\}$	$\{f\}$	$\{g\}$	$\{h\}$	$\{i\}$	$\{j\}$
(e,g)	$\{a\}$	$\{b,d\}$	$\{c\}$		$\{e,g\}$	$\{f\}$		$\{h\}$	$\{i\}$	$\{j\}$
(a,c)	$\{a,c\}$	$\{b,d\}$			$\{e,g\}$	$\{f\}$		$\{h\}$	$\{i\}$	$\{j\}$
(h,i)	$\{a,c\}$	$\{b,d\}$			$\{e,g\}$	$\{f\}$		$\{h,i\}$		$\{j\}$
(a,b)	$\{a,b,c,d\}$				$\{e,g\}$	$\{f\}$		$\{h,i\}$		$\{j\}$
(e,f)	$\{a,b,c,d\}$				$\{e,f,g\}$			$\{h,i\}$		$\{j\}$
(b,c)	$\{a,b,c,d\}$				$\{e,f,g\}$			$\{h,i\}$		$\{j\}$

(b)

Figure 21.1 **(a)** A graph with four connected components: $\{a, b, c, d\}$, $\{e, f, g\}$, $\{h, i\}$, and $\{j\}$. **(b)** The collection of disjoint sets after each edge is processed.

```
CONNECTED-COMPONENTS(G)
    for each vertex v ∈ V[G]
        do MAKE-SET(v)
    for each edge (u, v) ∈ E[G]
        do if FIND-SET(u) ≠ FIND-SET(v)
            then UNION(u, v)
```

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

```
SAME-COMPONENT(u, v)
    if FIND-SET(u) = FIND-SET(v)
        then return TRUE
        else return FALSE
```

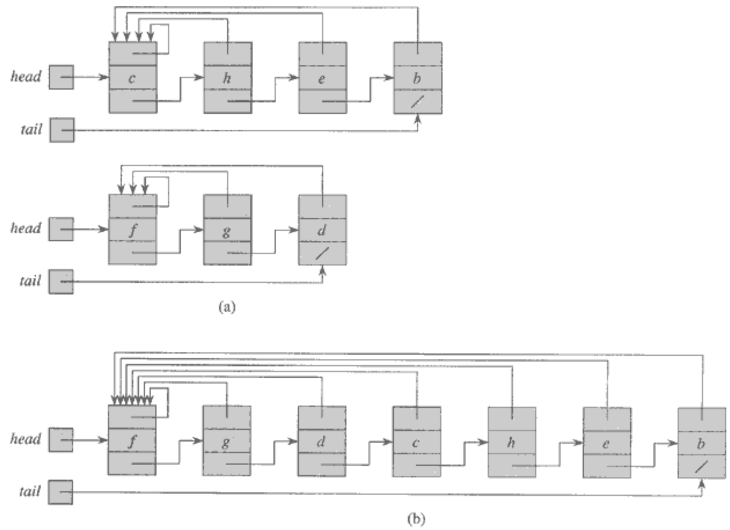


Figure 21.2 **(a)** Linked-list representations of two sets. One contains objects b, c, e, and h, with c as the representative, and the other contains objects d, f, and g, with f as the representative. Each object on the list contains a set member, a pointer to the next object on the list, and a pointer back to the first object on the list, which is the representative. Each list has pointers *head* and *tail* to the first and last objects, respectively. **(b)** The result of UNION(e, g). The representative of the resulting set is f.

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Operation	Number of objects updated
MAKE-SET(x_1)	1
MAKE-SET(x_2)	1
$\vdots$	$\vdots$
MAKE-SET(x_n)	1
UNION(x_1, x_2)	1
UNION(x_2, x_3)	2
UNION(x_3, x_4)	3
$\vdots$	$\vdots$
UNION(x_{n-1}, x_n)	$n - 1$

Figure 21.3 A sequence of $2n - 1$ operations on n objects that takes $\Theta(n^2)$ time, or $\Theta(n)$ time per operation on average, using the linked-list set representation and the simple implementation of UNION.

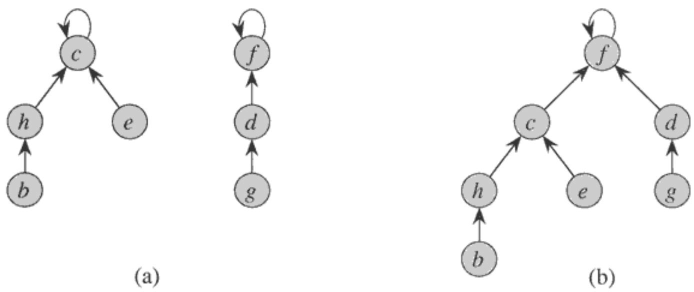


Figure 21.4 A disjoint-set forest. **(a)** Two trees representing the two sets of Figure 21.2. The tree on the left represents the set $\{b, c, e, h\}$, with c as the representative, and the tree on the right represents the set $\{d, f, g\}$, with f as the representative. **(b)** The result of UNION(e, g).

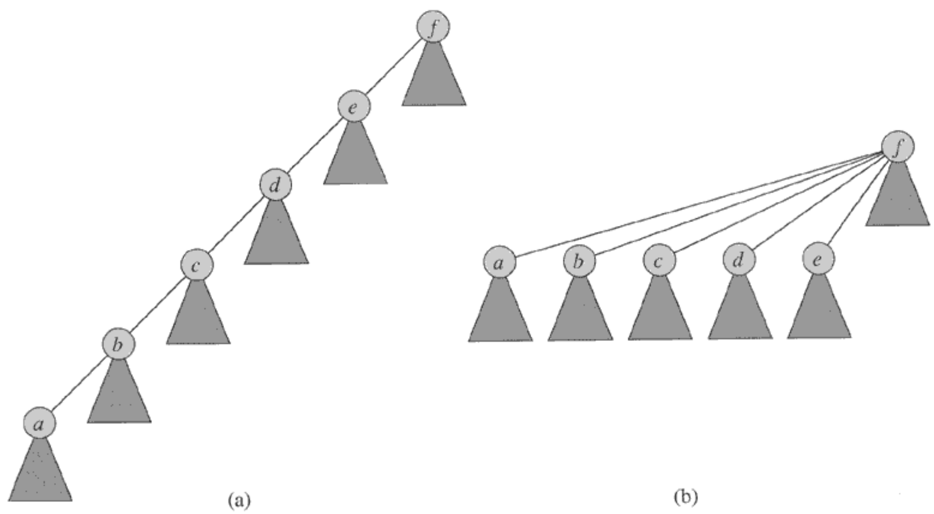


Figure 21.5 Path compression during the operation FIND-SET. Arrows and self-loops at roots are omitted. **(a)** A tree representing a set prior to executing FIND-SET(a). Triangles represent subtrees whose roots are the nodes shown. Each node has a pointer to its parent. **(b)** The same set after executing FIND-SET(a). Each node on the find path now points directly to the root.

MAKE-SET(x)

1 $p[x] \leftarrow x$

2 $rank[x] \leftarrow 0$

UNION(x, y)

1 LINK(FIND-SET(x), FIND-SET(y))

LINK(x, y)

if $rank[x] > rank[y]$
    then $p[y] \leftarrow x$
    else $p[x] \leftarrow y$
        if $rank[x] = rank[y]$
          then $rank[y] \leftarrow rank[y] + 1$

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

FIND-SET(x)

if $x \neq p[x]$
    then $p[x] \leftarrow$ FIND-SET$(p[x])$
return $p[x]$