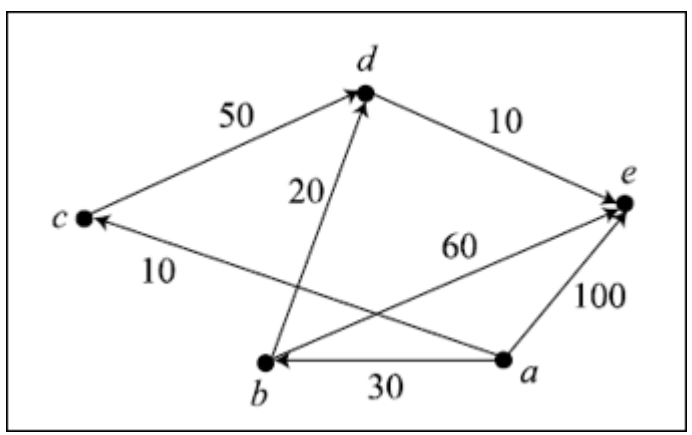
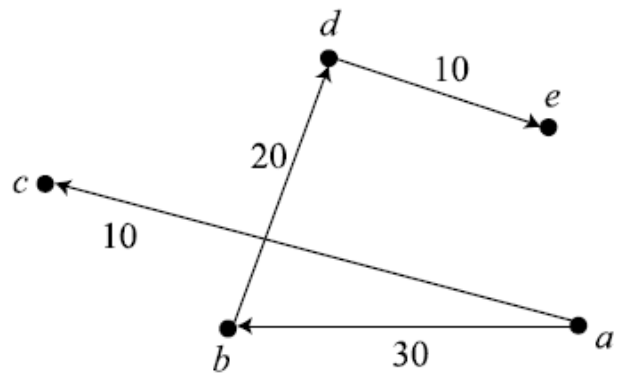
[표 9-1] 다익스트라 알고리즘을 이용한 최단경로 탐색 과정

단계	탐색 전 최단경로 집합 (S)과 남은 꼭짓점(V)	$D[b]$	$D[c]$	$D[d]$	$D[e]$	최단경로 꼭짓점	탐색 후 최단경로 집합 (S)과 남은 꼭짓점(V)
(1)	$S=\{a\}$ $V=\{b, c, d, e\}$	30	10	∞	100	c	$S=\{a, c\}$ $V=\{b, d, e\}$
(2)	$S=\{a, c\}$ $V=\{b, d, e\}$	30	10	60	100	b	$S=\{a, c, b\}$ $V=\{d, e\}$
(3)	$S=\{a, c, b\}$ $V=\{d, e\}$	30	10	50	90	d	$S=\{a, c, b, d\}$ $V=\{e\}$
(4)	$S=\{a, c, b, d\}$ $V=\{e\}$	30	10	50	60	e	$S=\{a, c, b, d, e\}$ $V=\{\ \}$





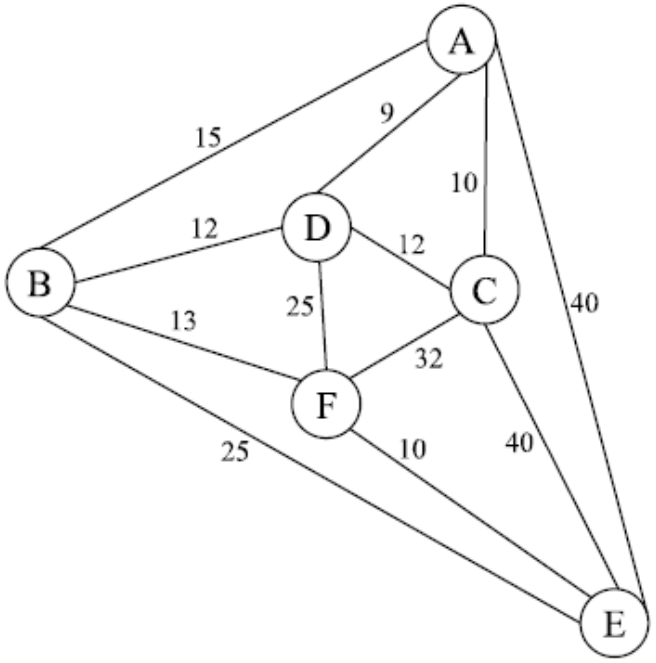
[그림 9-17] [그림 9-16]의 결과

3. 트리의 활용

- 프림 알고리즘(Prim Algorithm)
  - 주어진 가중치 그래프 G에서 최소 신장 트리를 프림 알고리즘으로 구하는 규칙
    - (1) 가중치가 가장 작은 모서리를 선택한다.
    - (2) (1)에서 선택된 모서리에 의해 연결된 꼭짓점들과 연결된 모든 모서리들 중 가중치가 가장 작은 모서리를 선택한다.
    - (3) 가중치가 같은 모서리는 임의로 하나를 선택한다.
    - (4) 선택된 모서리에 의해 순환이 형성되는 경우는 선택하지 않는다.
    - (5) 그래프 G에 포함된 모든 꼭짓점의 수가 n개일 때, n-1개의 모서리가 연결되면 종료

3. 트리의 활용

- [그림 10-21]의 그래프 G를 프림 알고리즘의 규칙으로 최소 신장 트리로 만들기
  - 각 모서리에 부여된 가중치는 비용

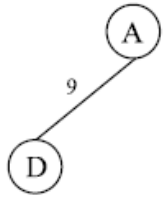


[그림 10-21] 프림 알고리즘의 예

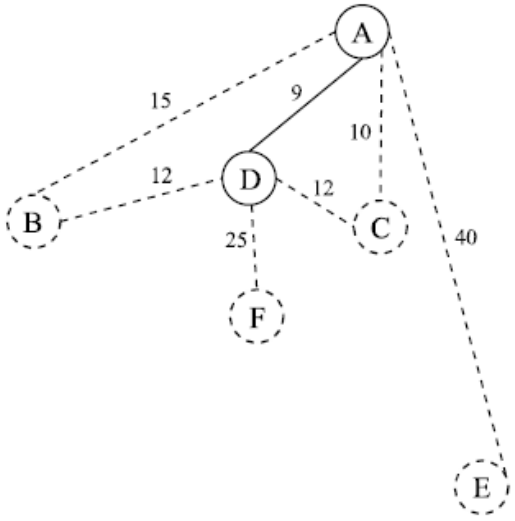
3. 트리의 활용

- [그림 10-21]의 그래프 G에 포함된 노드는 6개

(1) 모서리에 부여된 가중치 중 가장 작은 9를 선택하면 노드 A와 노드 D가 연결된다(노드 수 2개, 모서리 수 1개)

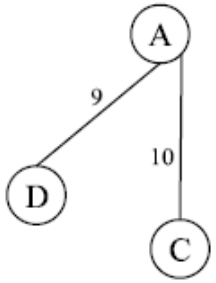


(2) 노드 A와 D에 연결된 모서리들과 노드들은 다음 점선으로 표시된 것과 같다.

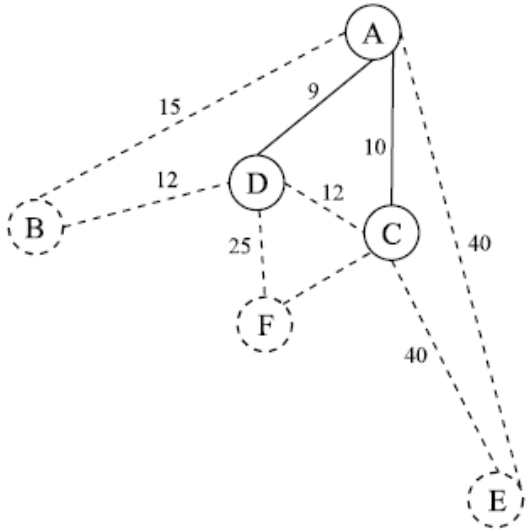


3. 트리의 활용

(3) (2)의 점선 중 가중치가 가장 낮은 10을 선택하면 노드 A와 노드 C가 연결된다(노드 수 3개, 모서리 수 2개).

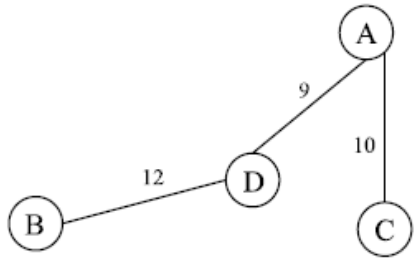


(4) 노드 A, C, D에 연결된 모서리들과 모드들은 다음 점선으로 표시된 것과 같다.

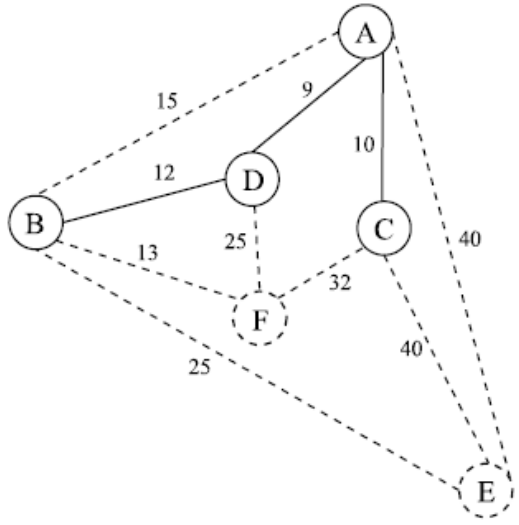


3. 트리의 활용

(5) (4)의 점선들 중 가중치가 가장 낮은 모서리는 노드 D와 C가 연결된 12와 노드 D와 B가 연결된 12이다. 이때 노드 D와 C를 연결하는 모서리를 선택하면 노드 A, D, C 간의 순환이 형성되므로 선택하지 않고 노드 D와 B를 연결하는 모서리를 선택한다(노드 수 4개, 모서리 수 3개).

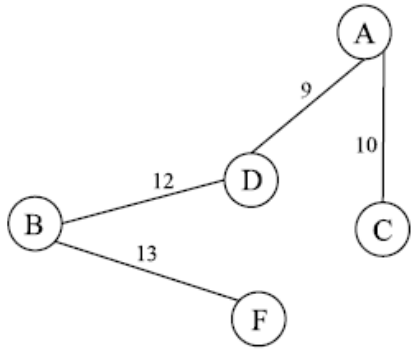


(6) 노드 A, B, C, D에 연결된 모서리들과 노드들은 점선으로 표시된 것과 같다. 노드 D와 C 간의 모서리는 (5)에서 순환을 형성하는 것을 확인했으므로 제거한다.

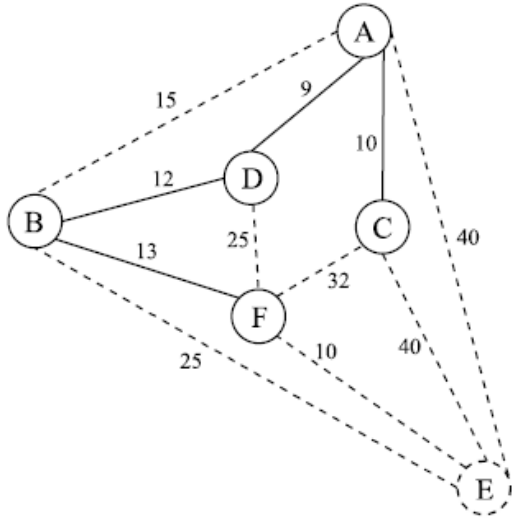


3. 트리의 활용

(7) (6)의 점선들 중 가중치가 가장 낮은 13을 선택하면 노드 B와 F가 연결된다(노드 수 5개, 모서리 수 4개).

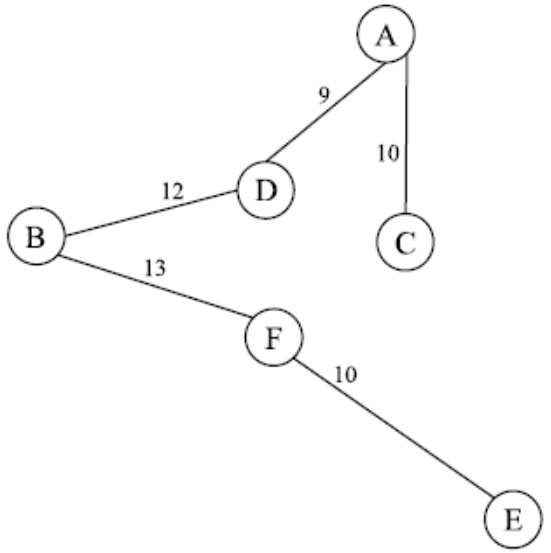


(8) 노드 A, B, C, D, F에 연결된 모서리들과 노드들은 다음의 점선으로 표시된 것과 같다.



3. 트리의 활용

(9) (8)의 점선들 중 가중치가 가장 낮은 10을 선택하면 노드 F와 E가 연결된다(노드 수 6개, 모서리 수 5개).



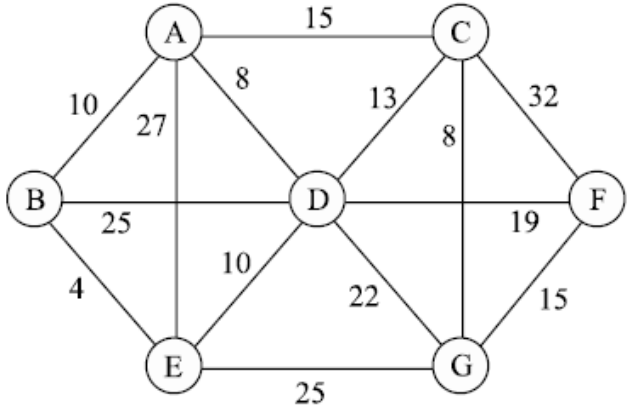
(10) 그래프 G에 포함된 6개의 노드가 $5(=6-1)$개의 모서리에 의해 연결된 최소 신장 트리가 완성되었다. 이 최소 신장 트리의 가중치는 다음과 같다.

$$9 + 10 + 12 + 13 + 10 = 54$$

3. 트리의 활용

예제 10-15

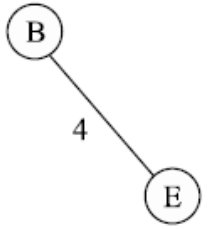
다음 그래프 G를 보고 프림 알고리즘을 이용해 최소 신장 트리를 작성하고 트리의 가중치를 구하라(부여된 가중치는 비용이다).



풀이

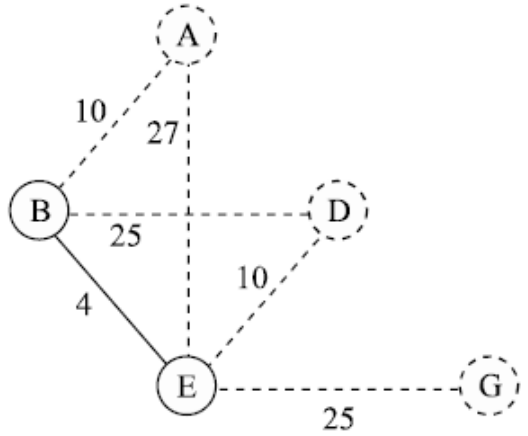
이 그래프의 노드의 수는 7개이다.

① 모서리에 부여된 가중치 중 가장 낮은 4를 선택하면 B와 E가 연결된다(노드 수 2개, 모서리 수 1개).

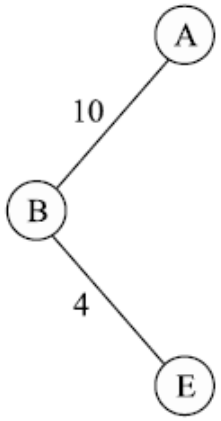


3. 트리의 활용

② 노드 B와 E에 연결된 모서리들과 노드들은 점선으로 표시된 것과 같다.

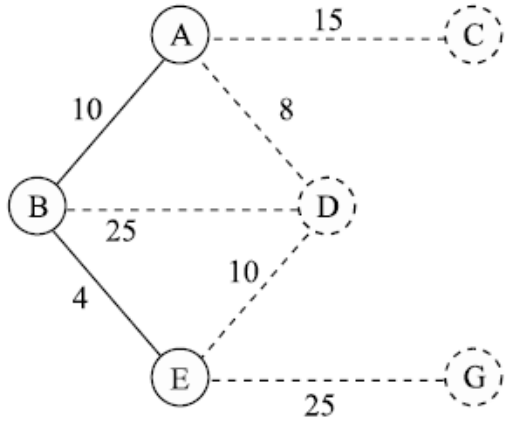


③ ②의 점선들 중 가중치가 가장 낮은 모서리는 B와 A, E와 D가 연결된 10이다. 이 중 B와 A를 선택하면 다음과 같다(노드 수 3개, 모서리 수 2개).

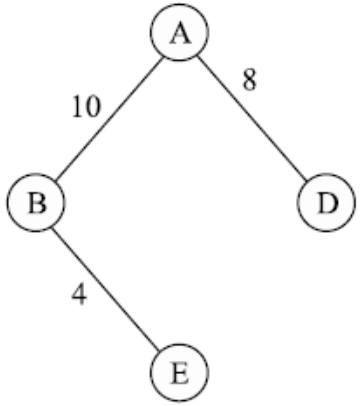


3. 트리의 활용

④ 노드 A, B, E에 연결된 모서리들과 노드들은 다음 점선으로 표시된 것과 같다. A와 E 사이의 모서리는 A, B, E 간에 순환이 생성되므로 제외하였다.

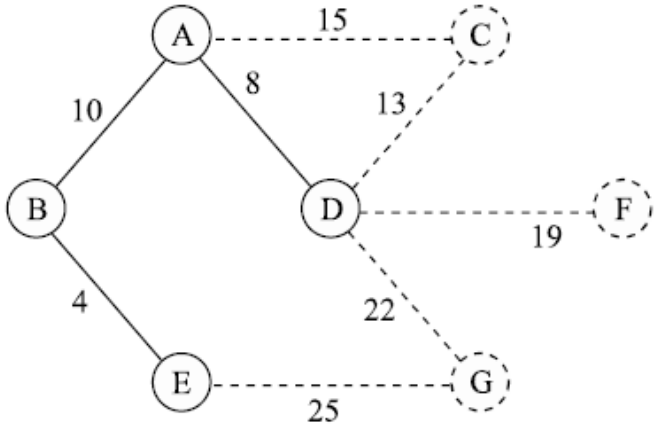


⑤ ④의 점선들 중에 가중치가 가장 낮은 모서리는 A와 D가 연결된 8이다(노드 수 4개, 모서리 수 3개).

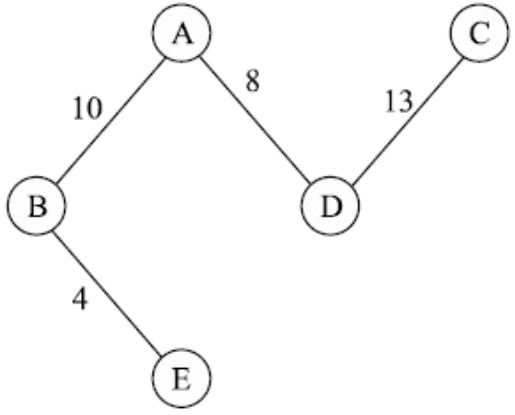


3. 트리의 활용

⑥ 노드 A, B, D, E에 연결된 모서리들과 노드들은 다음 점선으로 표시된 것과 같다. A와 E, B와 D, E와 D 간의 모서리는 A, B, E, E 사이에 순환이 생성되므로 제외하였다.

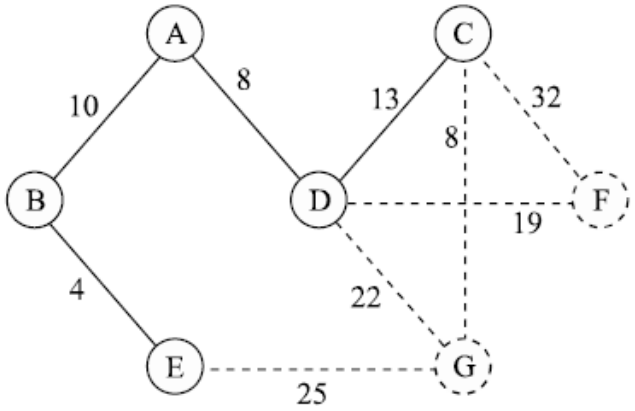


⑦ ⑥의 점선들 중 가중치가 가장 낮은 모서리는 D와 C가 연결된 13이다(노드 수 5개, 모서리 수 4개).

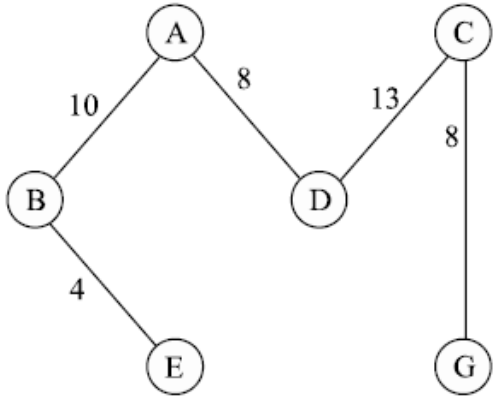


3. 트리의 활용

⑧ 노드 A, B, C, D, E에 연결된 모서리들과 노드들은 다음 점선으로 표시된 것과 같다. A와 C 간의 모서리는 A, C, D 사이에 순환을 생성하므로 제외하였다.

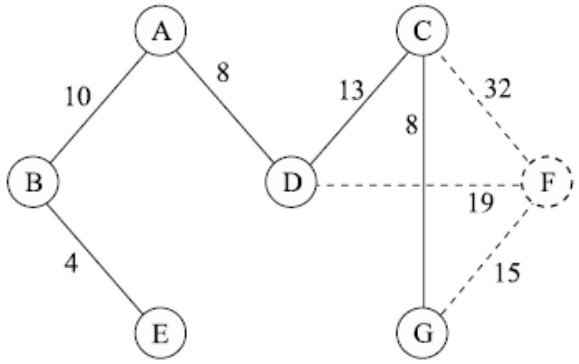


⑨ ⑧의 점선들 중 가중치가 가장 낮은 모서리는 C와 G가 연결된 8이다(노드 수 6개, 모서리 수 5개).

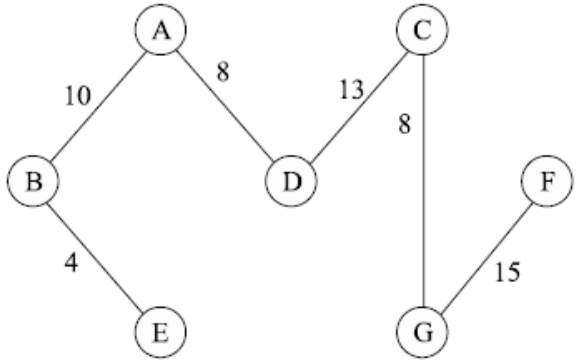


3. 트리의 활용

⑩ 노드 A, B, C, D, G, E에 연결된 모서리들과 노드들은 다음 점선으로 표시된 것과 같다. D와 G 간의 모서리는 C, D, G 간에 순환을 생성하므로 제외하였다.



⑪ ⑩의 점선들 중 가중치가 가장 낮은 모서리는 G와 F가 연결된 15이다(노드 수 7개, 모서리 수 6개).



⑫ 모든 노드 7개가 6(=7−1)개의 모서리와 연결되었으므로 알고리즘을 종료한다. 이때 최소 신장 트리의 비용은 $4+10+8+13+8+15=58$이다.

3. 트리의 활용

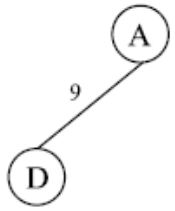
- 크루스칼 알고리즘(Kruskal Algorithm)
  - 주어진 가중치 그래프 G에서 최소 신장 트리를 크루스칼 알고리즘으로 구하려면 다음과 같은 규칙에 따름
    - (1) 가중치가 가장 작은 모서리를 차례로 선택한다.
    - (2) 가중치가 같은 모서리는 모두 선택한다.
    - (3) 선택된 모서리에 의해 순환이 형성되는 경우는 선택하지 않는다.
    - (4) 그래프 G에 포함된 모든 꼭짓점의 수가 n개일 때, n-1개의 모서리가 연결되면 종료
  - [그림 10-21]에서 예를 들었던 그래프를 이용해 크루스칼 알고리즘으로 최소 신장 트리를 만들어보기. 그래프 G에 포함된 노드는 6개
    - (1) 모든 노드에 연결된 모서리의 가중치를 정리하면 다음 표와 같음

[표 10-1] [그림 10-21]의 그래프 각 모서리의 가중치

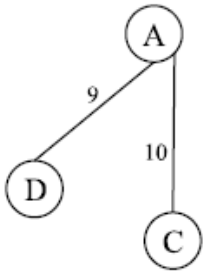
노드 연결	가중치	노드 연결	가중치
A-B	15	A-C	10
A-D	9	A-E	40
B-D	12	B-E	25
B-F	13	C-D	12
C-E	40	C-F	32
D-F	25	E-F	10

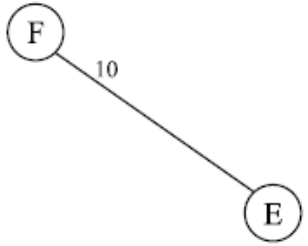
3. 트리의 활용

(2) (1)의 표에서 가중치가 가장 낮은 노드의 연결은 A-D이다(노드 수 2개, 모서리 수 1개).



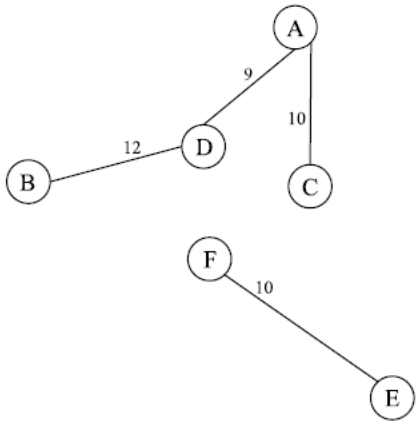
(3) (1)의 표에서 두 번째로 가중치가 낮은 노드의 연결은 A-C와 F-E이다(노드 수 5개, 모서리 수 3개).



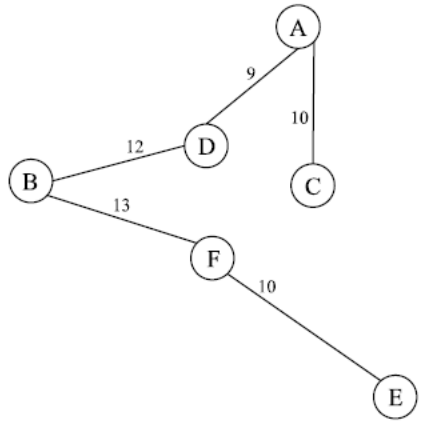


3. 트리의 활용

(4) (1)의 표에서 세 번째로 가중치가 낮은 노드의 연결은 B-D와 C-D이다. 그러나 C-D는 A-C-D 사이에 순환이 생성되므로 사용하지 않는다(노드 수 6개, 모서리 수 4개).



(5) (1)의 표에서 네 번째로 가중치가 낮은 노드의 연결은 B-F이다(노드 수 6개, 모서리 수 5개).



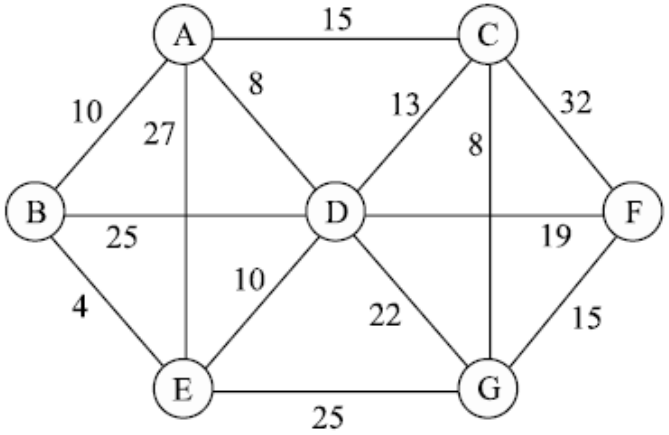
(6) 모든 노드(6개)가 5(=6-1)개의 모서리로 연결되었으므로 알고리즘을 종료한다. 이때 최소 신장 트리의 비용은 다음과 같다.

$$9 + 10 + 12 + 13 + 10 = 54$$

3. 트리의 활용

예제 10-16

[예제 10-15]에서 사용했던 그래프 G를 보고 크루스칼 알고리즘을 이용해 최소 신장 트리를 작성하고 트리의 가중치를 구하라(부여된 가중치는 비용이다).



풀이

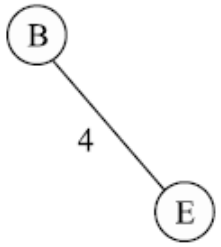
3. 트리의 활용

이 그래프의 노드의 수는 7개이다.

① 모든 노드에 연결된 모서리의 가중치를 정리하면 다음과 같다.

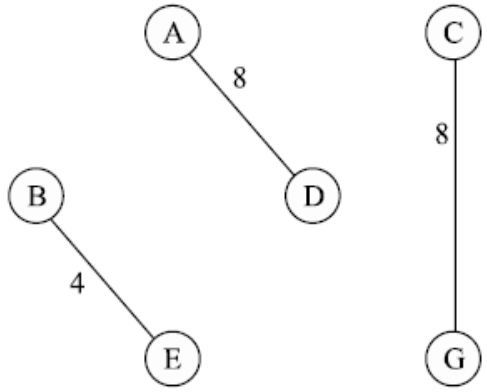
노드 연결	가중치	노드 연결	가중치
A–B	10	A–C	15
A–D	8	A–E	27
B–D	25	B–E	4
C–D	13	C–F	32
C–G	8	D–E	10
D–F	19	D–G	22
E–G	25	F–G	15

② ①의 표에서 가장 가중치가 낮은 노드의 연결은 B–E이다(노드 수 2개, 모서리 수 1개).

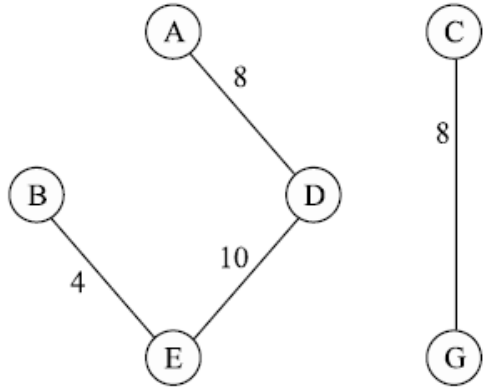


3. 트리의 활용

③ ①의 표에서 두 번째로 가중치가 낮은 노드의 연결은 A-D, C-G이다(노드 수 6개, 모서리 수 3개).

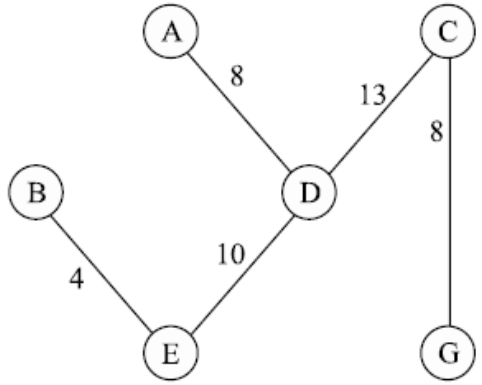


④ ①의 표에서 세 번째로 비용이 낮은 노드의 연결은 A-B와 D-E이다. 그러나 둘 다 선택하면 A, B, D, E 사이에 순환이 생성되므로 둘 중 하나만 선택한다. 여기서는 D-E를 선택한다(노드 수 6개, 모서리 수 4개).

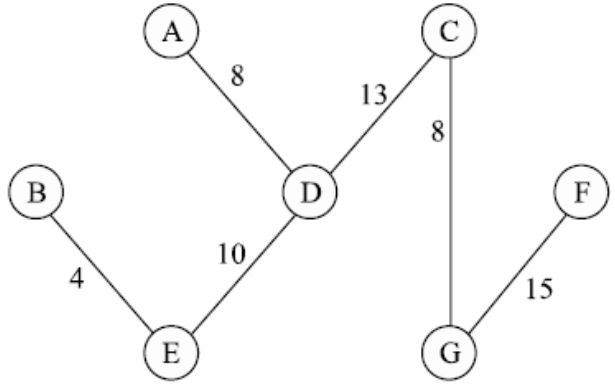


3. 트리의 활용

⑤ ①의 표에서 네 번째로 가중치가 낮은 노드의 연결은 C-D이다(노드 수 6개, 모서리 수 5개).



⑥ ①의 표에서 다섯 번째로 가중치가 낮은 노드의 연결은 A-C와 F-G이다. 그러나 A-C는 A, C, D 사이에 순환이 생성되므로 사용하지 않는다(노드 수 7개, 모서리 수 6개).



⑦ 모든 노드 7개가 6(=7-1)개의 모서리와 연결되었으므로 알고리즘을 종료한다. 이때 최소 신장 트리의 비용은 $4+10+8+13+8+15=58$이다.

3. 트리의 활용

❖ 허프만 코드

정의 10-26 허프만 알고리즘(Huffman Algorithm)

발생 빈도가 높은 문자는 적은 비트를 할당하고 발생 빈도가 낮은 문자에는 많은 비트를 할당하는 알고리즘

(1) 발생 빈도가 가장 낮은 두 문자를 선택하여 하나의 이진 트리를 생성한다.
  ① 왼쪽 노드에 빈도 수가 낮은 문자, 오른쪽 노드에 빈도가 높은 문자를 위치시킨다.
  ② 빈도 수가 같은 경우는 사전적으로 앞에 오는 문자를 왼쪽에 위치시킨다.
  ③ 두 문자의 빈도의 합을 그 문자들의 부모 노드로 한다.

(2) (1)의 과정을 모든 문자가 하나의 이진 트리로 묶일 때까지 반복한다.

(3) 생성된 이진 트리의 왼쪽 노드에는 0, 오른쪽 노드에는 1을 부여한다.

(4) 루트부터 해당 문자까지의 0 또는 1을 순서대로 나열한다.

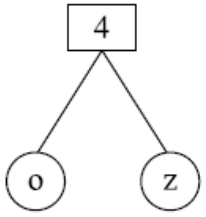
3. 트리의 활용

- 하나의 파일에 포함된 문자들의 빈도 수가 [표 10-2]와 같다고 할 때, 각 문자의 허프만 코드를 구해 보기

[표 10-2] 허프만 코드를 구하기 위한 문자들의 빈도 수

문자	b	e	g	j	l	o	s	u	z
빈도 수	15	8	23	8	30	2	17	5	2

(1) 빈도가 가장 낮은 문자는 o와 z이다. 두 문자의 빈도 수가 같고 o가 z보다 사전적 순서가 앞이므로 o를 왼쪽 노드, z를 오른쪽 노드로 한다.

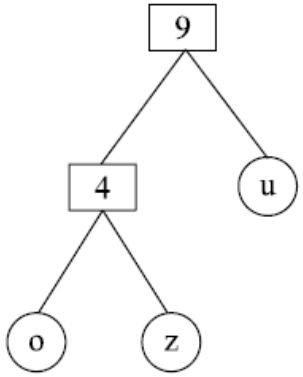


[표 10-2]를 다음과 같이 수정하였다.

문자	b	e	g	j	l	s	u	o / z
빈도 수	15	8	23	8	30	17	5	4

3. 트리의 활용

(2) 다음으로 빈도가 가장 낮은 문자는 (1)에서 만들어진 o/z 서브 트리와 u이다. u의 빈도 수가 o/z 서브 트리보다 높으므로 오른쪽 노드로 한다.

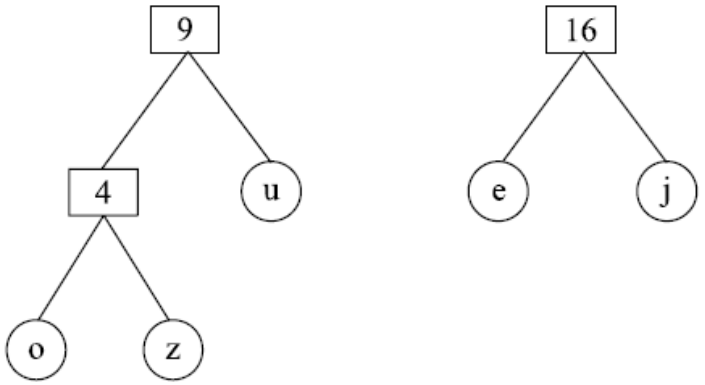


(1)의 표를 다음과 같이 수정하였다.

문자	b	e	g	j	l	s	o / z / u
빈도 수	15	8	23	8	30	17	9

3. 트리의 활용

(3) 다음으로 빈도가 낮은 문자는 e와 j이다. 그러므로 (2)에서 만들어진 o/z/u 서브 트리와 별도로 서브 트리가 만들어진다.

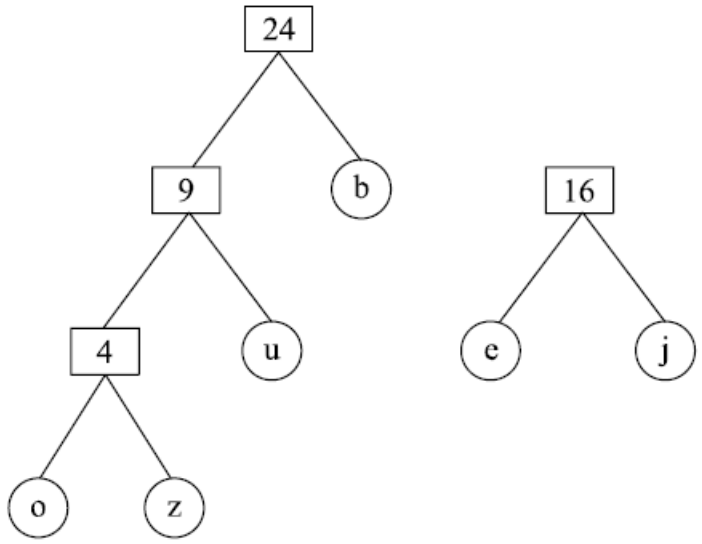


(2)의 표를 수정하면 다음과 같다.

문자	b	g	l	s	e / j	o / z / u
빈도 수	15	23	30	17	16	9

3. 트리의 활용

(4) 다음으로 빈도 수가 낮은 문자는 b와 o/z/u 서브 트리이다. b의 빈도 수가 o/z/u 서브 트리보다 높으므로 오른쪽 노드에 위치시킨다.

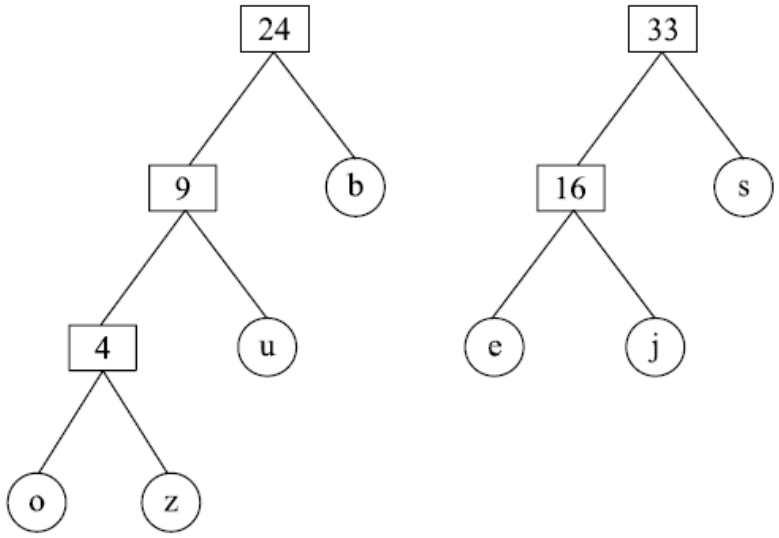


(3)의 표를 수정하면 다음과 같다.

문자	g	l	s	e / j	o / z / u / b
빈도 수	23	30	17	16	24

3. 트리의 활용

(5) 다음으로 빈도 수가 낮은 문자는 s와 e/j 서브 트리이다. e/j 서브 트리의 빈도 수가 더 낮으므로 왼쪽 서브 트리로 하고 s를 오른쪽 노드에 위치시킨다.

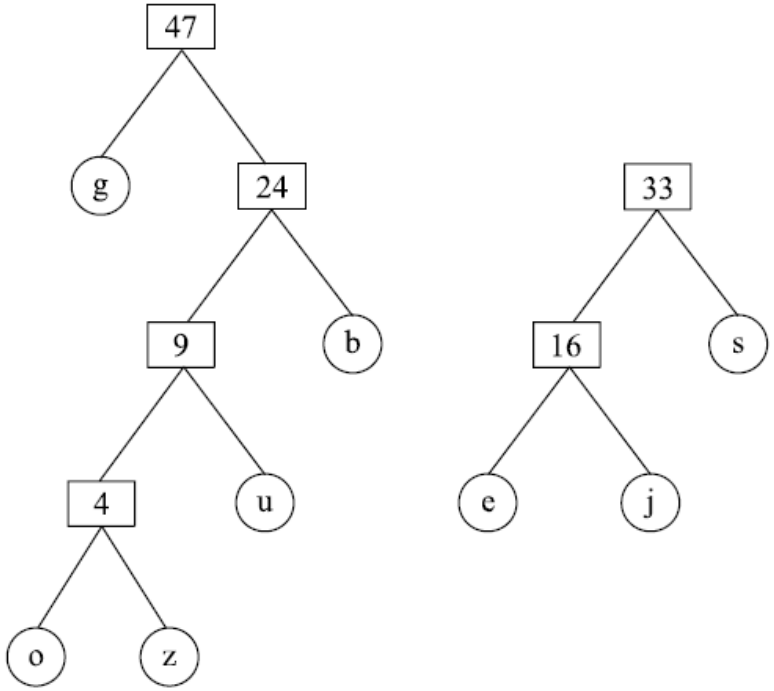


(4)의 표를 수정하면 다음과 같다.

문자	g	l	e / j /s	o / z / u / b
빈도 수	23	30	33	24

3. 트리의 활용

(6) 다음으로 빈도 수가 낮은 문자는 g와 o/z/u/b 서브 트리이다. g의 빈도 수가 더 낮으므로 왼쪽 노드에 위치시키고 o/z/u/b 서브 트리는 오른쪽 서브 트리로 한다.

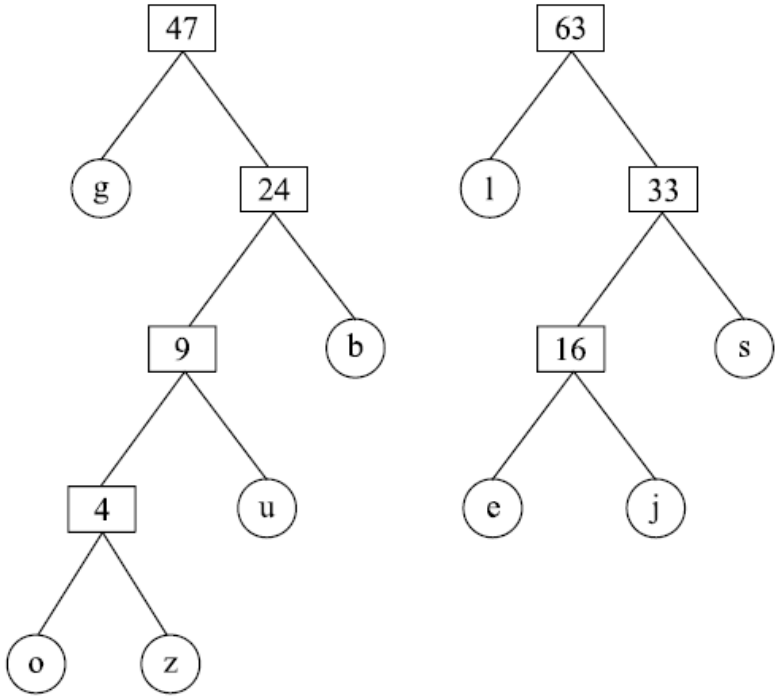


(5)의 표를 수정하면 다음과 같다.

문자	l	e / j /s	g / o / z / u / b
빈도 수	30	33	47

3. 트리의 활용

(7) 다음으로 빈도 수가 낮은 문자는 l과 e/j/s 서브 트리이다. l의 빈도 수가 더 낮으므로 l을 왼쪽 노드에 위치시키고, e/j/s 서브 트리를 오른쪽 서브 트리로 한다.

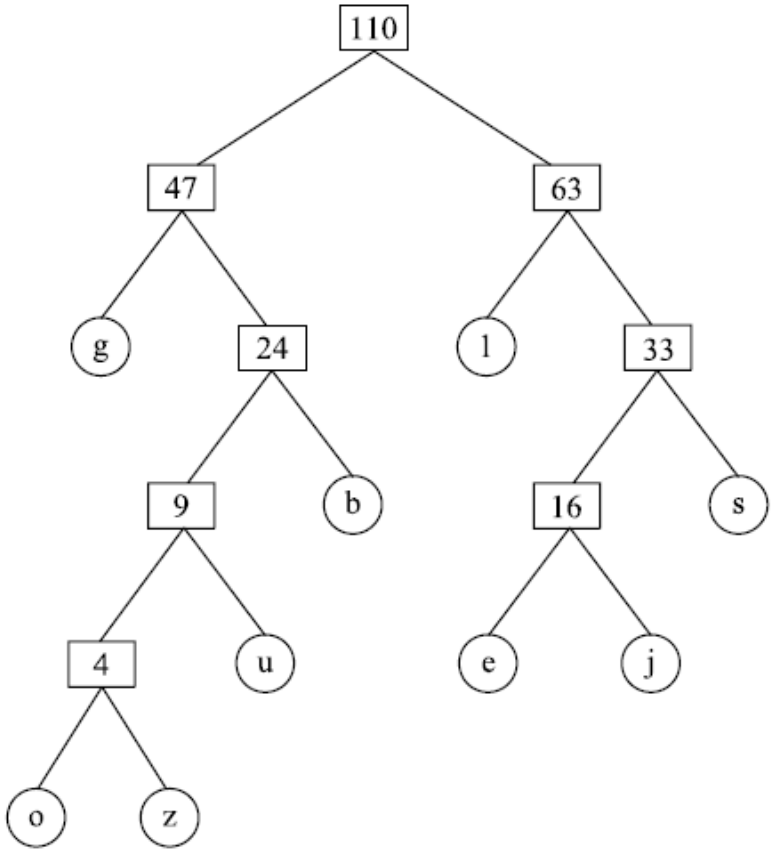


(6)의 표를 수정하면 다음과 같다.

문자	l / e / j /s	g / o / z / u / b
빈도 수	33	47

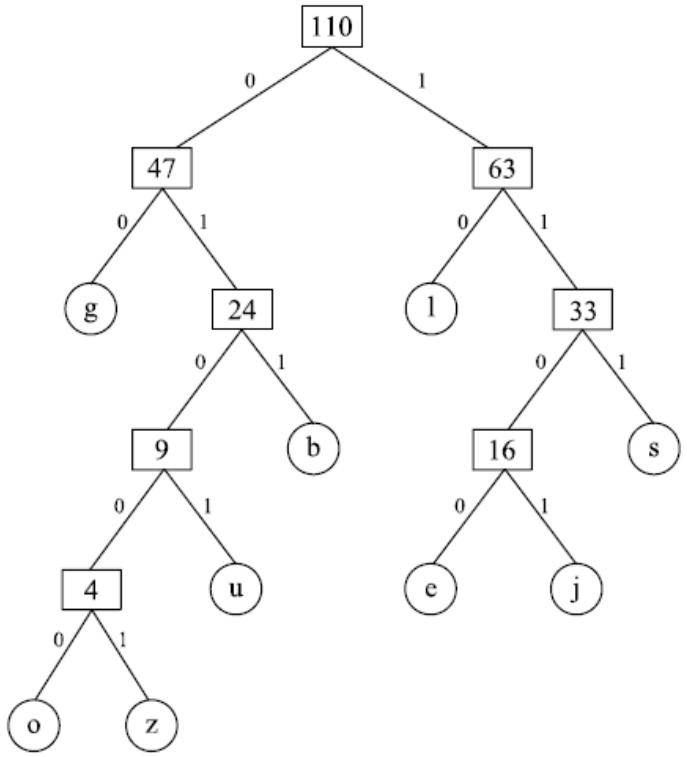
3. 트리의 활용

(8) 이제 두 개의 서브 트리를 하나의 루트 노드로 연결한다. 두 서브 트리 중 l/e/j/s 서브 트리의 빈도 수가 더 낮으므로 왼쪽 서브 트리로 하고, g/o/z/u/b 서브 트리는 오른쪽 서브 트리로 한다.



3. 트리의 활용

(9) 각 서브 트리마다 왼쪽 노드는 0을, 오른쪽 노드는 1을 부여한다.



(10) (9)의 결과를 이용해 각 문자에 코드를 부여한다. 예를 들면 레벨 2에 위치한 g의 경우, 루트 노드부터 부여된 값을 따라오면, 00의 코드를 갖게 된다. 이러한 원리로 각 문자에 부여된 허프만 코드는 다음과 같다.

트리레벨	2		3		4			5	
문자	g	l	b	s	u	e	j	o	z
허프만 코드	00	10	011	111	0101	1100	1101	01000	01001

3. 트리의 활용

예제 10-17

파일 내의 문자 빈도가 다음과 같을 때 질문에 답하라.

문자	a	b	c	d	e	f	g	h	i
빈도 수	5	7	13	9	16	2	7	10	4

(1) 허프만 알고리즘을 따라 허프만 트리를 만들어라.

(2) 각 문자에 대한 허프만 코드를 작성하라.

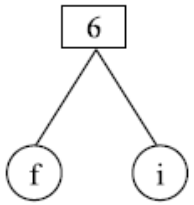
(3) (2)의 허프만 코드를 이용해 문자열 baeefcceehhibdgbgb를 코드로 작성하라.

(4) (2)의 허프만 코드를 이용해 000001111000011011000111010110010011111011코드에 대응되는 문자열을 작성하라.

풀이

(1) ① 가장 빈도가 낮은 문자는 f와 i이다. f의 빈도 수가 더 낮으므로 왼쪽 노드에, i를 오른쪽 노드에 위치시킨다.

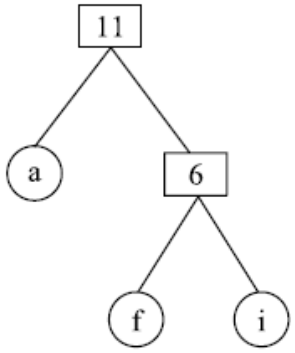
3. 트리의 활용



위의 표를 수정하면 다음과 같다.

문자	a	b	c	d	e	g	h	f / i
빈도 수	5	7	13	9	16	7	10	6

② 다음으로 빈도 수가 낮은 문자는 a와 f/i 서브 트리이다. a의 빈도 수가 더 낮으므로 a를 왼쪽 노드에 위치시키고, f/i 서브 트리를 오른쪽 서브 트리로 한다.

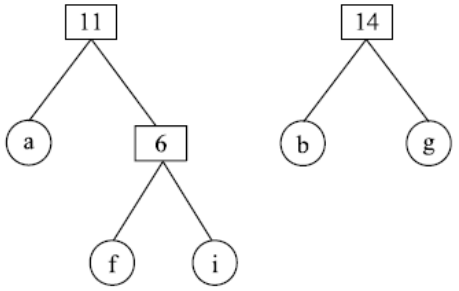


①의 표를 수정하면 다음과 같다.

문자	b	c	d	e	g	h	a / f / i
빈도 수	7	13	9	16	7	10	11

3. 트리의 활용

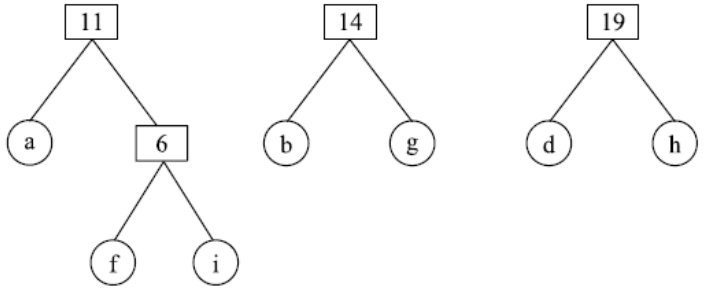
③ 다음으로 빈도 수가 낮은 문자는 b와 g이다. 두 문자의 빈도 수가 같으므로 사전적으로 앞에 오는 b를 왼쪽 노드에, g를 오른쪽 노드에 위치시킨다.



②의 표를 수정하면 다음과 같다.

문자	c	d	e	h	b / g	a / f / i
빈도 수	13	9	16	10	14	11

④ 다음으로 빈도 수가 낮은 문자는 d와 h이다. d의 빈도 수가 더 낮으므로 d를 왼쪽 노드에, h를 오른쪽 노드에 위치시킨다.

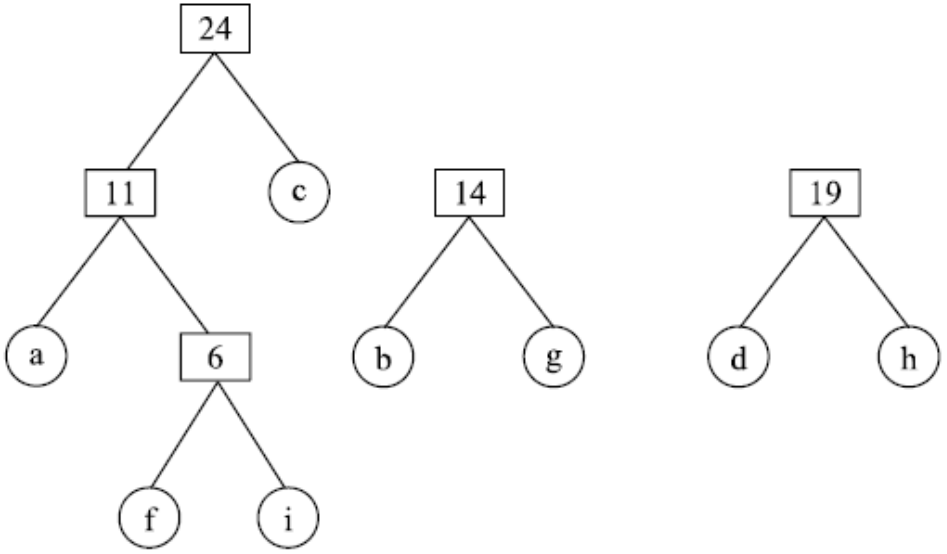


③의 표를 수정하면 다음과 같다.

문자	c	e	d / h	b / g	a / f / i
빈도 수	13	16	19	14	11

3. 트리의 활용

⑤ 다음으로 빈도 수가 낮은 문자는 c와 a/f/i 서브 트리이다. a/f/i 서브 트리의 빈도 수가 더 낮으므로 왼쪽 서브 트리로 하고, c는 오른쪽 노드에 위치시킨다.

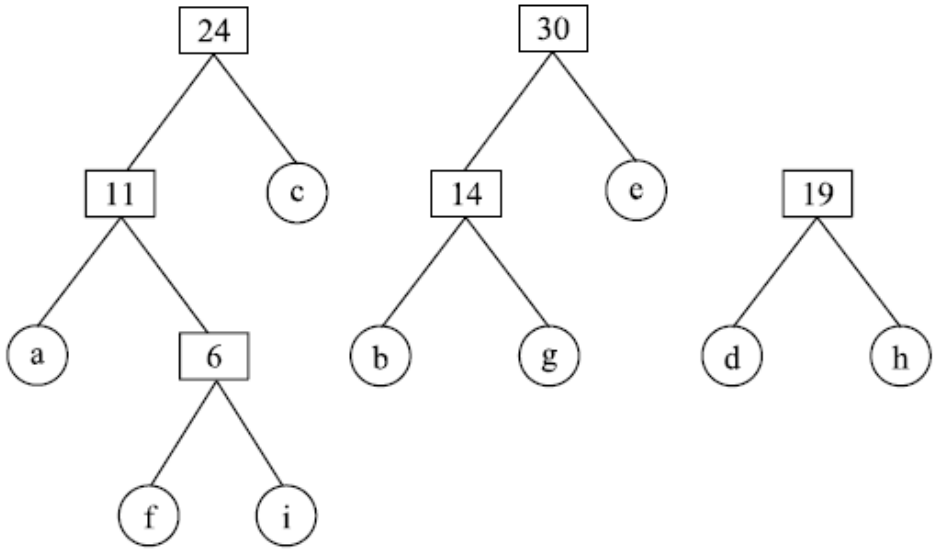


④의 표를 수정하면 다음과 같다.

문자	e	d / h	b / g	a / f / i / c
빈도 수	16	19	14	24

3. 트리의 활용

⑥ 다음으로 빈도 수가 낮은 문자는 e와 b/g 서브 트리이다. b/g 서브 트리의 빈도 수가 더 낮으므로 왼쪽 서브 트리로 하고, e는 오른쪽 노드에 위치시킨다.

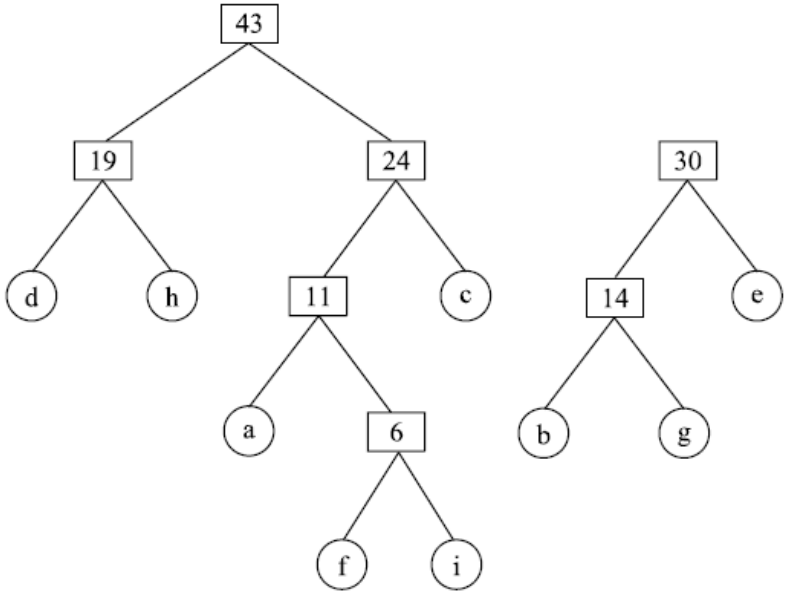


⑤의 표를 수정하면 다음과 같다.

문자	d / h	b / g / e	a / f / i / c
빈도 수	19	30	24

3. 트리의 활용

⑦ 다음으로 빈도 수가 낮은 서브 트리는 d/h 서브 트리와 a/f/i/c 서브 트리이다. d/h 서브 트리의 빈도 수가 더 낮으므로 왼쪽 서브 트리로 하고 a/f/i/c 서브 트리는 오른쪽 서브 트리로 한다.

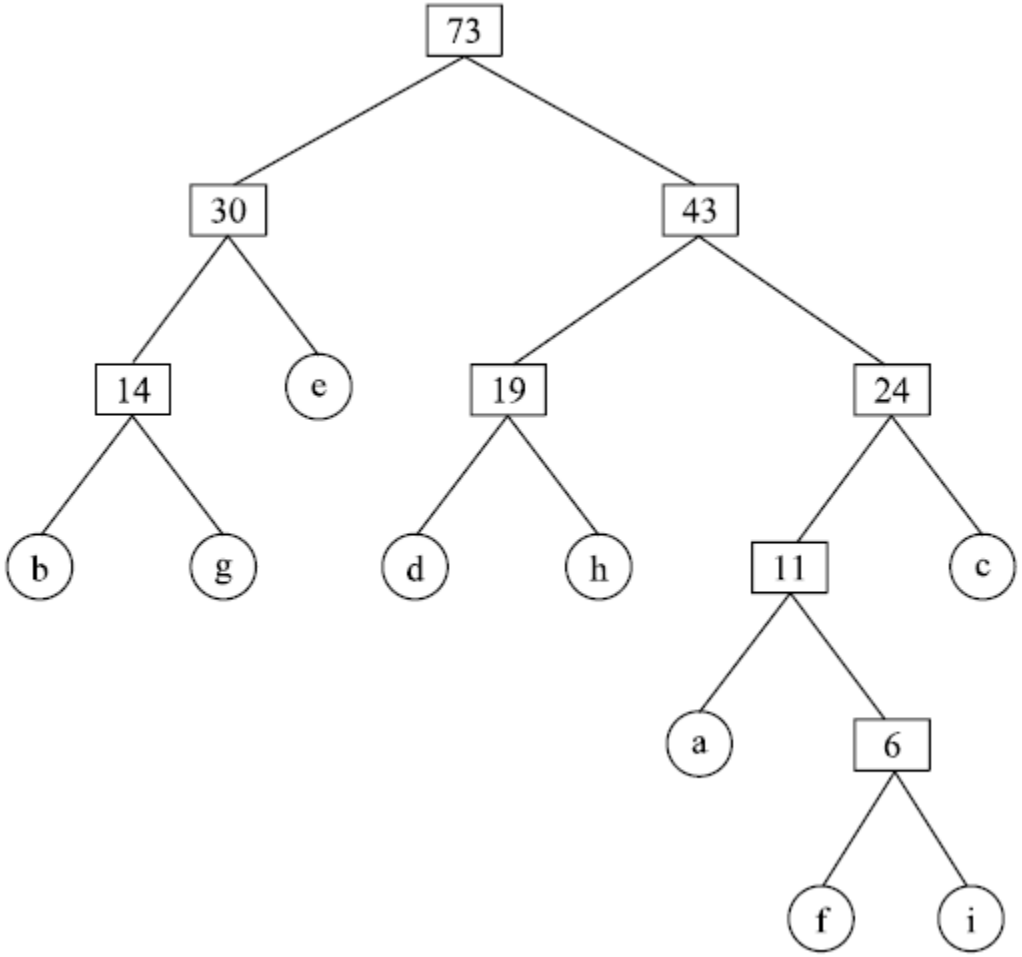


⑥의 표를 수정하면 다음과 같다.

문자	b / g / e	d / h /a / f / i / c
빈도 수	30	43

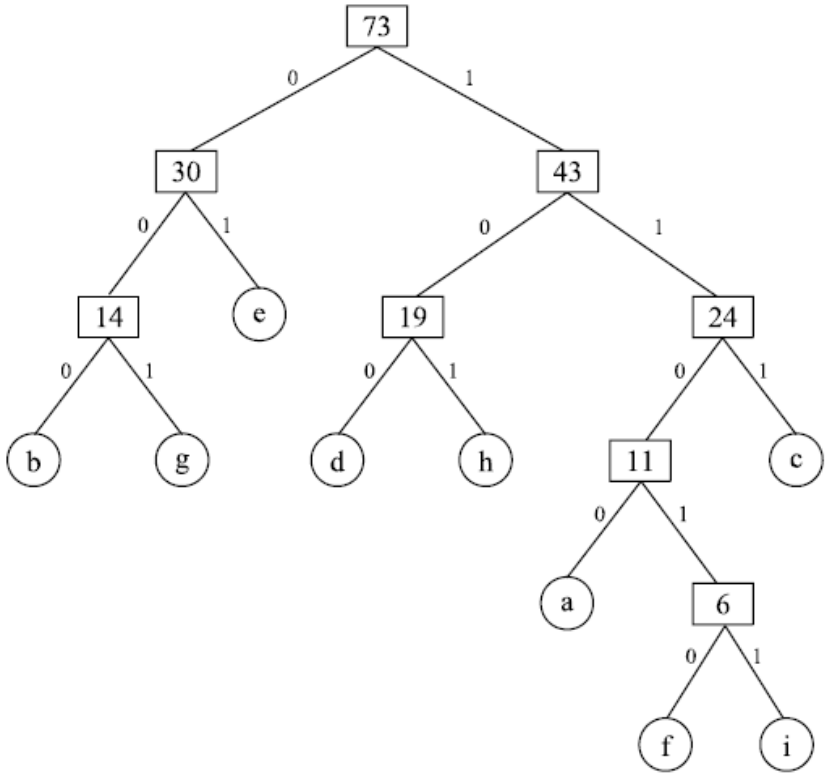
⑧ 남은 두 개의 서브 트리를 루트 노드로 연결한다. b/g/e 서브 트리의 빈도 수가 d/h/a/f/i/c 서브 트리의 빈도 수보다 낮으므로 왼쪽 서브 트리로 하고, d/h/a/f/i/c 서브 트리는 오른쪽 서브 트리로 한다.

3. 트리의 활용



3. 트리의 활용

(2) (1)에서 나온 트리의 각 노드에 0과 1을 부여한다.



위의 결과를 이용해 각 문자에 코드를 부여한다.

트리레벨	2	3					4	5	
문자	e	b	g	d	h	c	a	f	i
허프만 코드	01	000	001	100	101	111	1100	11010	11011

3. 트리의 활용

(3) (2)에서의 허프만 코드를 이용해 문자열 baeefcceehhibdgbgb를 코드로 작성하면 다음과 같다.

000110001011101011111101011011011011000100001000001000

(4) 000001111000011011000111010110010011111011을 문자로 바꾸려면, 앞에서부터 차례로 허프만 코드표에 대응되는 코드가 있는지를 확인하면 된다.

① 000으로 구성되는 문자는 b

000	001111000011011000111010110010011111011
b	

② 001로 시작되는 문자는 g

000	001	111000011011000111010110010011111011
b	g	

③ 111 세 개가 연속되는 문자는 c

000	001	111	000011011000111010110010011111011
b	g	c	

④ 000 세 개가 연속되는 문자는 b

000	001	111	000	011011000111010110010011111011
b	g	c	b	

3. 트리의 활용

⑤ 01로 구성된 문자는 e

000	001	111	000	01	10110001110101100100111111011
b	g	c	b	e	

⑥ 101로 구성된 문자는 h

000	001	111	000	01	101	1000111010110010011111011
b	g	c	b	e	h	

⑦ 100으로 구성된 문자는 d

000	001	111	000	01	101	100	011101011001001111011
b	g	c	b	e	h	d	

⑧ 01로 구성된 문자는 e

000	001	111	000	01	101	100	01	11010110010011111011
b	g	c	b	e	h	d	e	

⑨ 11010으로 구성된 문자는 f

000	001	111	000	01	101	100	01	11010	110010011111011
b	g	c	b	e	h	d	e	f	

⑩ 1100으로 구성된 문자는 a

000	001	111	000	01	101	100	01	11010	1100	10011111011
b	g	c	b	e	h	d	e	f	a	

3. 트리의 활용

⑪ 100으로 구성된 문자는 d

000	001	111	000	01	101	100	01	11010	1100	100	11111011
b	g	c	b	e	h	d	e	f	a	d	

⑫ 111로 구성된 문자는 c

000	001	111	000	01	101	100	01	11010	1100	100	111	11011
b	g	c	b	e	h	d	e	f	a	d	c	

⑬ 11011로 구성된 문자는 I

000	001	111	000	01	101	100	01	11010	1100	100	111	11011
b	g	c	b	e	h	d	e	f	a	d	c	i

∴ 0000011110000110110001110101100100111111011을 문자열로 바꾸면 bgcbehdefadci이다.