4주차 3차시 합병 정렬

【학습목표】

1. 분할 통치법 개념을 설명할 수 있다.
2. 합병 정렬을 설명할 수 있다.

학습내용1 : 분할 통치법

1. 분할 통치(정복)법

효율이 좋고 알고리즘을 설계하는 기법으로서 가장 널리 사용되고 있는 기법 중에 분할 통치법 (divide-and-conquer) 이라는 것이 있는데, 분할 통치법에서는 해결하려고 하는 문제 (크기를 n 이라고 한다) 를 크기가 보다 작은 여러 개의 부분 문제로 분할한다. 단, 이 때 크기가 작은 부분 문제에 대한 답으로부터 원래의 문제에 대한 해답을 쉽게 얻을 수 있게 분할할 필요가 있다. 이 기법에 대해서는 합병 정렬과, 퀵 정렬, 2진 탐색 등이 있다.
분할통치 방법의 세 단계는 원래의 문제를 소문제로 분할하는 분할 단계, 소문제의 해를 순환적으로 구하는 통치(정복)단계, 소문제의 해를 합하여 원래 문제에 대한 해를 만들어 내는 합병단계로 구성된다.

* 문제 : 탁구 대회를 열어서 가장 탁구를 잘 하는 사람과 가장 못하는 사람을 정하려고 한다. 실력이 좋은 사람이 항상 이긴다고 했을 때 몇 번 시합을 하면 되는가를 계산하는 것이 문제이다.

* 문제 세부내용 : 여기서 탁구 대회 참가 인원을 n 명이라고 하자. 이 문제를 해결하기 위해서 스포츠 경기시에 일반적으로 많이 이용하는 「토너먼트법」을 이용하는데, 첫 번째 토너먼트에서는 가장 강한 사람을 결정하고, 그 다음에는 나머지 n - 1명 중에서 마찬가지로 토너먼트법을 이용해서 가장 약한 사람을 정하는 방법을 이용하기로 하자 (그림 (1)). 그림 (1) 에서 굵은 선은 시합에서 이긴 사람을 나타내고, × 표시가 있는 것은 진 사람을 나타낸다.

* 그림 (1)

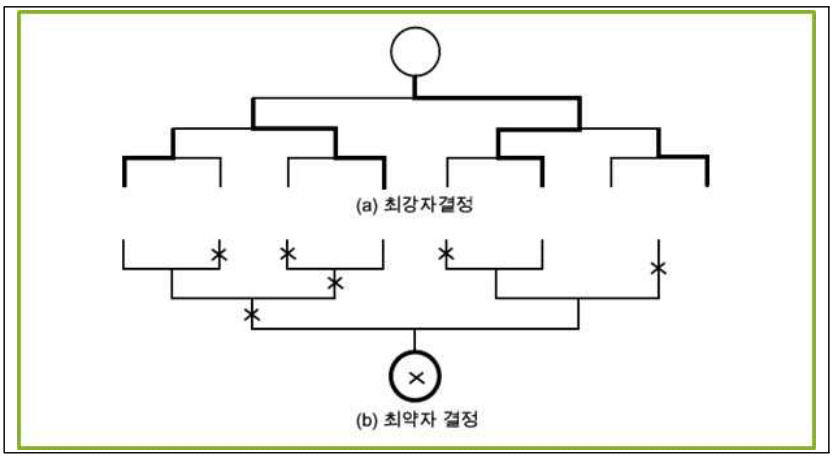
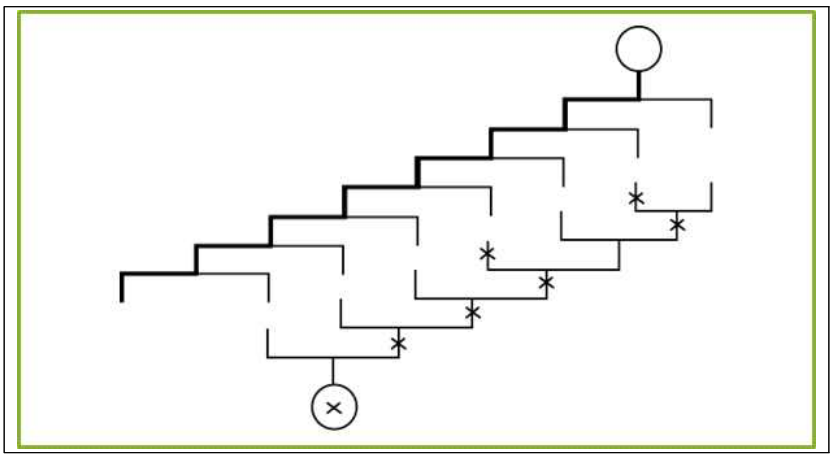


그림 (1) 과 같은 방법을 이용하는 경우에는 한번 시합을 할 때마다 한 사람씩 대상에서 제외되므로, n 명 중에서 가장 강한 사람을 정하기 위해서는 n - 1 번 시합을 하게 된다. 따라서 이 방법에서 필요한 시합수를 T1 (n) 이라고 하면, $T_1(n) = (n - 1) + (n - 2) = 2n - 3$ 이 된다.

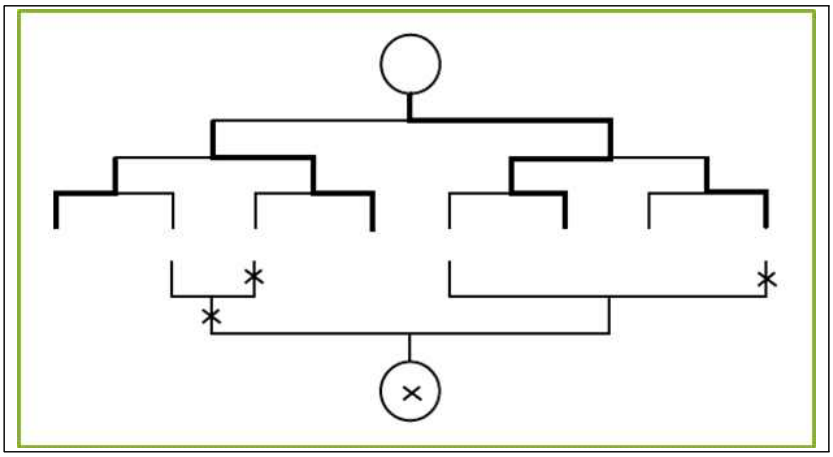
그러나, 조금만 곰곰히 생각해 보면 그림 1 과 같은 방법을 이용하는 경우에는 불필요한 시합을 많이 한다는 것을 금방 알 수 있다. 첫 번째 토너먼트에서 이긴 사람 (약 n / 2 명) 은 자기가 이긴 상대보다는 강하다는 것은 알고 있으므로 다음번에 실시하는 토너먼트에는 출전할 필요가 없다. 두 번째 토너먼트전은 첫 번째 토너먼트에서 한번도 이기지 못하고 진 사람만을 모아서 실시하면 되므로, 두 번째 토너먼트에 출전하는 사람수를 k 라고 하면 두 번째 토너먼트전을 운영하기 위해서는 k - 1 번 시합을 하면 된다.
이 방법으로 경기를 할 때의 시합수를 평가하기 위해서는 k 를 구해야 하는데, 이를 위해서는 첫 번째 토너먼트전을 구체적으로 어떻게 행하는지가 문제가 된다. 이 때 전년도 우승자에게 다른 사람이 차례로 한 사람씩 도전하는 방법을 택한다고 하고, 전년도 우승자가 모두에게 이기면 k 의 값은 n - 1 이 된다. 따라서, 이 방법을 이용하더라도 그림 (1)의 방법과 같이 2n - 3 번 시합을 해야 한다 (그림 (2)).

* 그림 (2) 비능률적인 경기 운영



- 그러나 n 명을 거의 반씩 나눠서 첫 번째 토너먼트를 하게 하면 k 의 값은 n / 2 이 된다 (그림 (3))

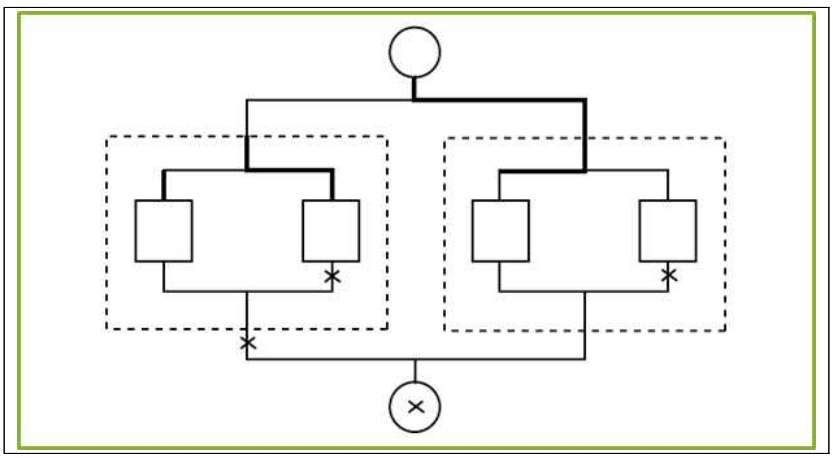
* 그림 (3) 능률이 좋은 경기 운영



이 때 계산을 간단하게 하기 위해 n을 짝수라 하고, 이 경우의 시합수를 T2(n) 으로 하면,
$T_2(n) = (n - 1) + (n/2 - 1) = (3/2)n - 2$이 되어, 그림 (1)의 방법보다도 n / 2 - 1 번이나 시합수가 적어진다.

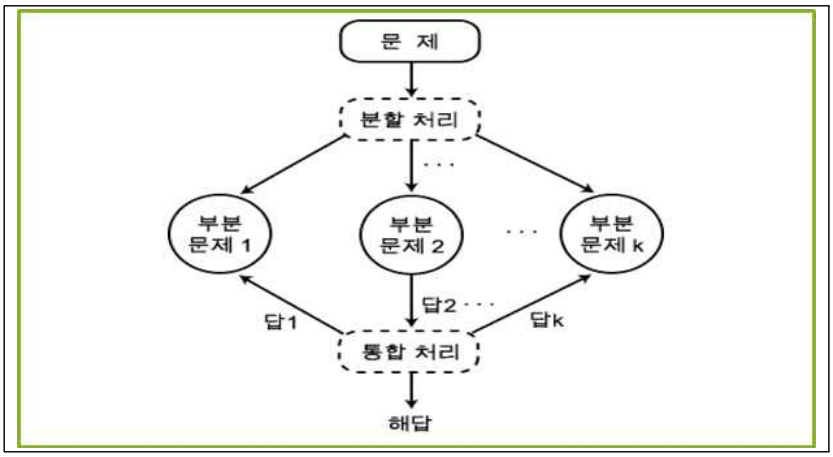
그러면, 이번에는 또 다른 방법을 생각해 보기로 하자.
우선 전체를 반으로 나눠서 각 팀에서 따로따로 최강자와 최약자를 결정한 후, 양 팀의 최장자끼리 시합을 하게 해서 전체의 최강자를 정하고, 최약자끼리 시합을 하게 해서 최약자를 정하면 된다 (그림 (4)).

* 그림 (4) 분할에 의한 최강 · 최약자 결정

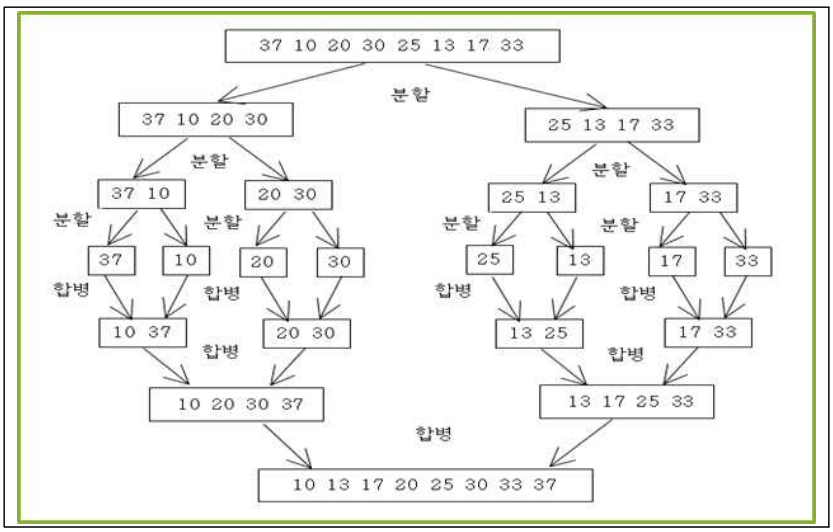


여기서 설명한 마지막 방법 (그림 (4)의 방법) 에서는 우선 전체를 반으로 나눠서 (분할) 각각에 대해 원래와 같은 문제를 해결하고 마지막에 간단한 처리를 해서 전체에 대한 해답을 구한다 (통합). 이와 같이 전체를 분할해서 각각을 따로따로 처리하고 마지막에 통합하는 방법을 「분할 통치법」이라고 한다 (그림 (5)).

* 그림 (5) 분할 통치법의 원리



* 분할과 합병의 예



학습내용2 : 합병 정렬

1. 합병 정렬이란?

합병 정렬(merge sort)은 단어의 뜻에서 처럼 두 개의 부분 파일을 서로 합치면서 정렬해 나가는 것을 말한다. 이를 위해서는 정렬되지 않은 데이터를 합병하기 위한 최소 단위로 먼저 분할하는 것이 필요하다.

1) 합병 정렬의 과정
예를 들어 8개의 데이터가 정렬되지 않은 상태로 있다면 먼저 해당 데이터를 2개의 부분 파일로 나눈다. 이 2개의 부분파일을 각각 또 하나의 정렬되지 않은 데이터로 보고 다시 각각을 두 개로 나눈다. 그러면 총 4개의 부분 파일이 된다. 이를 다시 각각에 대하여 분할하면 최소단위의 개별 데이터가 된다.
이제 분리된 데이터를 두 개씩 병합하면서 정렬한다. 최초 병합을 실행하면 각 2개의 데이터가 정렬되면서 총 4개의 부분 파일이 생성된다. 다시 4개의 부분 파일을 2개씩 묶으면서 정렬을 수행한다. 이 과정을 수행하면 총 4개로 구성된 2개의 부분 파일이 생성된다. 마지막으로 2개의 부분 파일을 하나로 병합하면서 정렬을 수행하면 최종적으로 정렬된 8개의

데이터를 얻을 수 있다.

합병 정렬의 수행시간은 최악의 경우에도 O(nlog n)이고 별도의 임시 기억장소가 필요하여 실제로는 수행시간이 많이 걸리며 외부정렬에 주로 활용된다.

2) 합병 정렬의 예

<원본 데이터>

1	7	6	4	3	8	5	2
12	59	34	23	17	61	31	14

<1단계>
먼저 전체 데이터를 정확히 2개로 분할하여 4개의 데이터를 가진 2개의 부분 파일로 나눈다.

1	7	6	4
12	59	34	23

3	8	5	2
17	61	31	14

<2단계>
두 개의 부분 파일을 다시 2개로 나눈다. 그러면 2개의 데이터로 이루어진 4개의 부분 파일이 생성된다.

1	7
12	59

6	4
34	23

3	8
17	61

5	2
31	14

<3단계>
아직 최소 단위가 아니므로 다시 4개의 부분 파일을 하나의 데이터로 분리한다. 여기까지 하면 분할 작업이 모두 끝나게 된다.

1
12

7
59

6
34

4
23

3
17

8
61

5
31

2
14

<4단계>
이제 인접한 데이터를 두 개씩 하나의 쌍으로 묶는다. 이때 묶으면서 데이터를 오름차순으로 정렬한다. 그러면 2개의 쌍으로 된 부분 파일이 4개 생성된다.

1	7
12	59

4	6
23	34

3	8
17	61

2	5
14	31

〈5단계〉
이번에는 2개의 데이터로 이루어진 부분 파일을 두 개를 다시 하나의 파일로 합쳐 4개로 이루어진 부분 파일을 생성한다. 그러면 총 4개로 이루어진 부분 파일 2개가 생성된다.

1	4	6	7
12	23	34	59

2	3	5	8
14	17	31	61

〈6단계〉
마지막으로 두 개의 부분 파일을 하나의 전체 파일로 합치면서 정렬을 한다. 이 과정을 거치면 오름차순으로 정렬된 데이터를 얻을 수 있다.

1	2	3	4	5	6	7	8
12	14	17	23	31	34	59	61

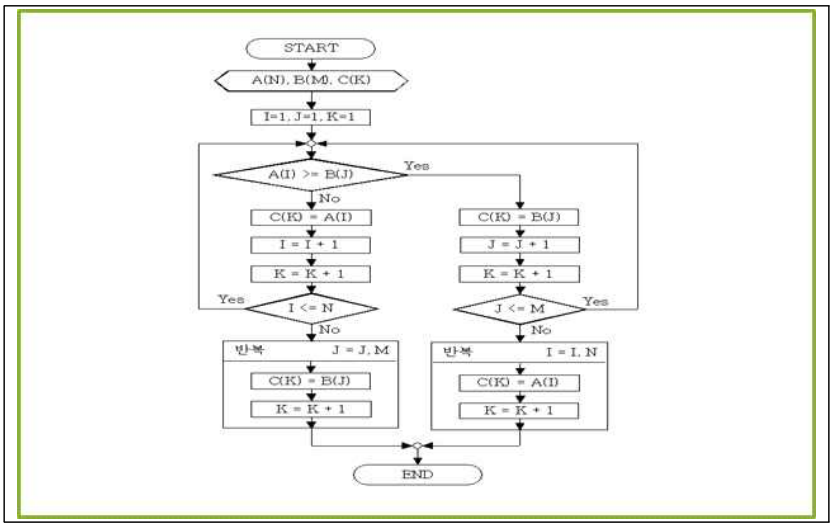
2. 합병정렬(merge sort)의 요약

정렬할 배열을 둘로 분할하여 왼쪽 부분배열에 대하여 합병정렬을 순환 호출하여 정렬하고 다음은 오른쪽 부분배열에 대하여 합병정렬을 순환 호출하여 정렬한 후 이 두 부분배열을 합병한다.
① 합병정렬은 퀵 정렬과 마찬가지로 분할정복 방식의 정렬 알고리즘이다.
② 합병정렬에서 키가 이동되는 것은 합병 과정에서만인데 합병과정에서 동일한 두 키의 상대적 위치는 변하지 않으므로 안정적인 알고리즘이다. 그러나 합병과정에서 버퍼 배열이 필요하다.
③ 합병(merge)함수에서 행하는 작업은 정렬된 두 부분배열을 선두 원소부터 차례로 비교하면서 순서에 맞게 합병한다.
④ 합병정렬 알고리즘의 시간복잡도는 평균/최악 모두 O(nlogn) 이다.

* 합병정렬의 실행 예
원본 → 71, 2, 38, 5, 7, 61, 11, 26, 53, 42
1단계 → (2, 71) (5, 38) (7, 61) (11, 26) (42, 53)
2단계 → (2, 5, 38, 71) (7, 11, 26, 61) (42, 53)
3단계 → (2, 5, 7, 11, 26, 38, 61, 71) (42, 53)
4단계 → 2, 5, 7, 11, 26, 38, 42, 53, 61, 71

분할통치 방법의 세 단계는 원래의 문제를 소문제로 분할하는 분할 단계, 소문제의 해를 순환적으로 구하는 정복단계, 소문제의 해를 합하여 원래 문제에 대한 해를 만들어 내는 합병단계로 구성된다.

* 합병정렬의 순서도



【학습정리】

1. 분할 통치(정복)법

효율이 좋고 알고리즘을 설계하는 기법으로서 가장 널리 사용되고 있는 기법 중에 분할 통치법 (divide-and-conquer) 이라는 것이 있는데, 분할 통치법에서는 해결하려고 하는 문제 (크기를 n 이라고 한다) 를 크기가 보다 작은 여러 개의 부분 문제로 분할한다. 단, 이 때 크기가 작은 부분 문제에 대한 답으로부터 원래의 문제에 대한 해답을 쉽게 얻을 수 있게 분할할 필요가 있다. 이 기법에 대해서는 합병 정렬과, 퀵 정렬, 2진 탐색 등이 있다.
분할통치 방법의 세 단계는 원래의 문제를 소문제로 분할하는 분할 단계, 소문제의 해를 순환적으로 구하는 통치(정복)단계, 소문제의 해를 합하여 원래 문제에 대한 해를 만들어 내는 합병단계로 구성된다.

2. 합병정렬(merge sort)

정렬할 배열을 둘로 분할하여 왼쪽 부분배열에 대하여 합병정렬을 순환 호출하여 정렬하고 다음은 오른쪽 부분배열에 대하여 합병정렬을 순환 호출하여 정렬한 후 이 두 부분배열을 합병한다.
① 합병정렬은 퀵 정렬과 마찬가지로 분할정복 방식의 정렬 알고리즘이다.
② 합병정렬에서 키가 이동되는 것은 합병 과정에서만인데 합병과정에서 동일한 두 키의 상대적 위치는 변하지 않으므로 안정적인 알고리즘이다. 그러나 합병과정에서 버퍼 배열이 필요하다.
③ 합병(merge)함수에서 행하는 작업은 정렬된 두 부분배열을 선두 원소부터 차례로 비교하면서 순서에 맞게 합병한다.
④ 합병정렬 알고리즘의 시간복잡도는 평균/최악 모두 O(nlogn) 이다.

* 합병정렬의 실행 예
원본 → 71, 2, 38, 5, 7, 61, 11, 26, 53, 42
1단계 → (2, 71) (5, 38) (7, 61) (11, 26) (42, 53)
2단계 → (2, 5, 38, 71) (7, 11, 26, 61) (42, 53)
3단계 → (2, 5, 7, 11, 26, 38, 61, 71) (42, 53)
4단계 → 2, 5, 7, 11, 26, 38, 42, 53, 61, 71