5주차 2차시 분포에 의한 정렬과 외부 정렬

【학습목표】

1. 계수 정렬을 이해할 수 있다.
2. 기수 정렬, 버킷 정렬을 이해할 수 있다.
3. 외부 정렬을 이해할 수 있다.

학습내용1 : 계수 정렬

1. 계수 정렬이란?

계수 정렬(Counting Sort)은 입력키가 어떤 범위, 예를 들어 1부터 k사이의 작은 정수 범위에 있다는 것을 알고 있을 때에만 적용할 수 있는 방법으로 어떤 입력키 x의 정렬위치는 x보다 작은 키가 몇 개나 입력에 나타나는지를 알면 결정할 수 있다. 예를 들어 입력 키들이 숫자일 때 입력에 10이라는 키가 있고 이보다 작은 키가 5개 있다면 10은 정렬 순서에서 6번째 위치한다. 계수 정렬에서는 입력 키들이 범위 k 내의 각 값에 대하여 입력 키가 실제로 입력에 나타나는 횟수를 계산한다.
계수 정렬 알고리즘의 시간 복잡도는 O(n+k)로 보통 k=O(n)일 때에만 사용되므로 실질적으로는 시간 복잡도가 O(n)이다. 계수 정렬의 한 가지 중요한 성질은 동일한 키의 상대적 순서가 바뀌지 않는 안정적인 정렬이란 것이다.

1) 계수 정렬을 그림으로 알아보면 다음과 같다.

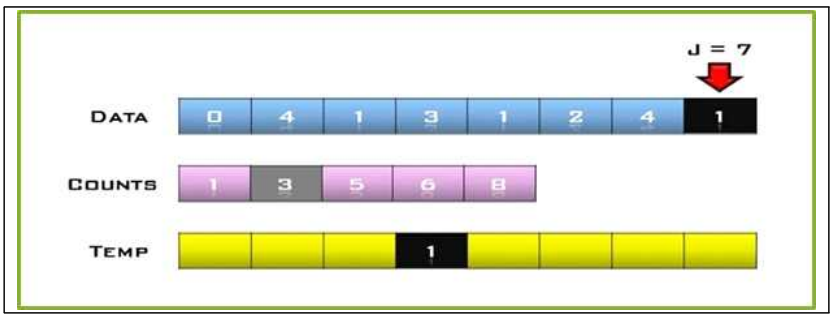
〈1단계〉

Data에서 각 항목들의 발생횟수를 센다. 횟수들은 정수 항목들로 직접 인덱스되는 카운트 배열 count 에 저장한다.

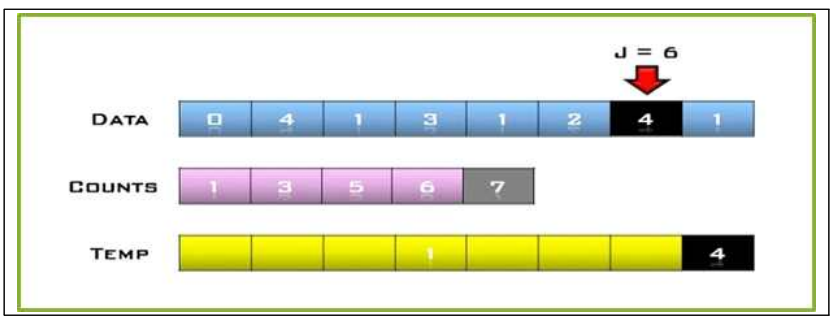
정렬된 집합에서 각 항목의 앞에 위치할 항목의 개수를 반영하기 위하여 카운트들은 조정한다.

〈2단계〉

temp에 1을 삽입하고 Counts[1]을 감소시킨 후

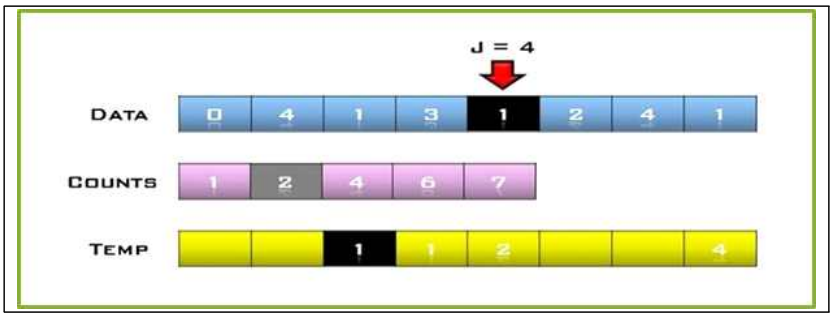


temp에 4를 삽입하고 Counts[4]를 감소시킨 후

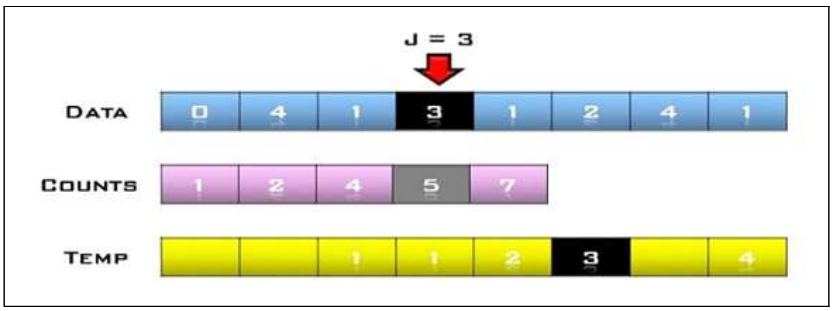


temp에 2를 삽입하고 Counts[2]를 감소시킨 후

temp에 1을 삽입하고 Counts[1]을 감소시킨 후



temp에 3을 삽입하고 Counts[3]을 감소시킨 후



temp에 0을 삽입하고 Counts[0]을 감소시킨다.

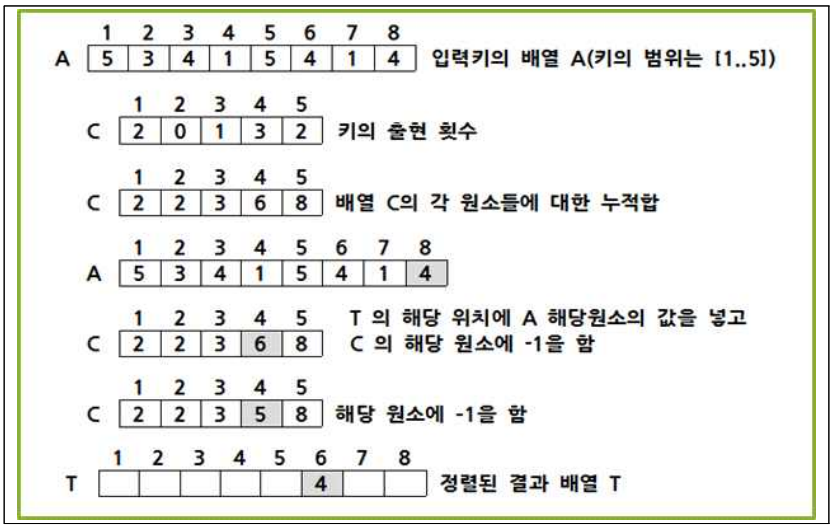
2. 계수정렬(counting sort)의 요약

계수정렬은 입력 키가 어떤 범위, 예를 들어 1부터 k사이의 정수 범위에 있다는 것을 알고 있을 때에만 적용할 수 있는 방법이다. 입력 키의 출현 횟수를 세어서 정렬하는 방식으로 중요한 성질은 동일한 키의 상대적 순서가 바뀌지 않는 안정적 정렬이라는 점이다.

① 계수정렬, 알고리즘의 시간복잡도는 평균/최악 모두 O(n)이다.

② 계수정렬에서 입력 데이터 외에 추가로 필요로 하는 배열은 계수를 위한 배열과 데이터 출력을 위한 배열이다.

* 계수 정렬의 실행 예



학습내용2 : 기수 정렬, 버킷 정렬

1. 기수 정렬이란?

기수 정렬(radix Sort)은 전체 키를 여러 자리로 나누어 각 자리마다 계수 정렬과 같은 안정적인 알고리즘을 적용하는 방법이다. 기수 정렬을 d 자리수 숫자들에 대하여 계수 정렬로 정렬한다면 각 자리에 대하여 O(n) 시간이 걸리므로 전체로는 O(dn) 시간이 걸리는데 d를 상수로 취급할 수 있다면 결론적으로는 O(n)의 시간이 걸린다고 볼 수 있다. 기수 정렬은 각 자리수를 정렬할 때 계수 정렬을 이용했는데 이 계수 정렬은 정렬과정에서 전체 정렬 데이터 개수 만큼의 메모리와 진법 크기 만큼의 메모리가 추가로 필요한 비제자리 정렬이므로 메모리의 여유가 많지 않을 경우에는 실제로 적용하기 어려운 정렬 방법이다.

1) 기수 정렬의 예

<원본 데이터>

5	1	6	2	4	3	7	10	9	8
50	4	55	10	20	15	60	99	80	74

〈1단계〉
1의 자리를 기준으로 각 큐에 계수 정렬을 기준으로 데이터를 담는다. 즉, 1의 자리가 0인 데이터를 큐에 담고 계수 정렬을 하면 10, 20, 50, 60, 80 이 될 것이며, 1의 자리가 4인 데이터를 큐에 담고 계수 정렬을 하면 4, 74가 되며, 1의 자리가 5인 데이터를 정렬하면 15, 55가 되고, 1의 자리가 9인 99가 남는다. 그리고 나서의 순서를 보면 다음과 같다.

2	4	5	7	9	1	8	3	6	10
10	20	50	60	80	4	74	15	55	99

〈2단계〉
이제 10의 자리를 기준으로 다시 계수 정렬을 실시한다. 먼저 한 자리수 인 4가 맨 앞에 오며, 10의 자리가 1인 10, 15가 오고, 10의 자리가 2인 20, 10의 자리가 5인 50, 55가 오며, 10의 자리가 6인 60이 오고, 10의 자리가 7인 74, 10의 자리가 8인 80, 10의 자리가 9인 99가 온다. 따라서 다음과 같이 데이터가 정렬된다.

1	2	3	4	5	6	7	8	9	10
4	10	15	20	50	55	60	74	80	99

2. 기수정렬(radix sort)의 요약

키의 각 자릿수에 대하여 낮은 자리에서 높은 자리의 순으로 버킷정렬을 적용하여 정렬하는 방법 각 자릿수 별로 계수정렬과 같은 안정적 정렬 알고리즘을 적용하여 정렬하는 방식으로 낮은 자리수 부터 먼저 정렬한다.

* 기수 정렬의 실행 예

다음과 같은 입력 키들을 정렬한다고 하자.
567, 654, 124, 457, 830, 911, 555
이들 세 자리 숫자들에 대하여 최하위 자리 수 에 대하여 계수 정렬을 적용하여 정렬한다. 그 결과는 다음과 같다.
830, 911, 654, 124, 555, 567, 457
다음은 그 위 자리들에 대하여 계수정렬을 계속 행한다.
911, 124, 830, 654, 555, 457, 567
124, 457, 555, 567, 654, 830, 911

① 기수정렬, 알고리즘의 시간복잡도는 평균/최악 모두 O(n)이다.
② 버킷정렬을 적용할 수 있는 조건은 키의 범위를 미리 알 수 있고 입력키가 균등하게 분포되는 경우이다.

3. 버킷 정렬이란?

버킷 정렬(bucket Sort)은 계수 정렬을 버킷 정렬로 부르는 경우도 있으나 여기서의 버킷 정렬은 계수 정렬을 일반화한 것으로 간주할 수 있다. 계수 정렬은 키 값이 작은 범위 안에 들어올 때 적용할 수 있는 방법이지만 버킷 정렬은 키 값의 범위뿐만이 아니라 그 범위 내에서 키 값이 확률적으로 균등하게 분포된다고 가정할 수 있을 때 적용할 수 있는 방법이다.

1) 버킷 정렬의 과정
편의상 키 값이 0과 1사이라고 가정한다. n개의 키가 구간[0,1]을 n등분하고 이들 각각을 하나의 버킷으로 한다. 각 키를 크기에 따라 각 버킷에 분배한다면, 키가 구간 내에서 균등하게 분포한다고 가정했으므로 하나의 버킷에는 하나의 키만 들어있을 확률이 높다. 그리고 만양 한 버킷에 여러 키가 들어가게 된다면 같은 버킷에 속하는 키끼리는 간단한 정렬 알고리즘으로 정렬한다. 이러한 버킷에 대하여 버킷 순서에 따라 데이터를 출력하면 정렬된 결과를 얻을 수 있다.

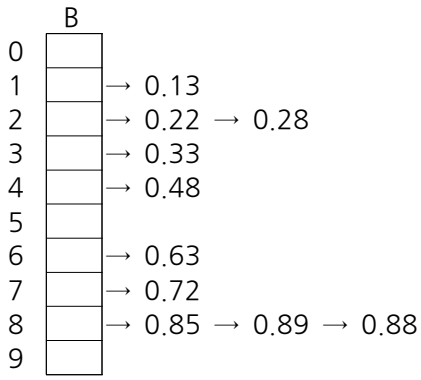
계수정렬에서는 키 값이 작은 범위 안에 들어올 때 적용할 수 있는 방법이지만 버킷정렬은 키 값의 범위뿐만이 아니라 그 범위 내에서 키 값이 확률적으로 균등하게 분포된다고 가정할 수 있는 방법이다.

* 버킷 정렬의 특징
① 정해진 구간을 n등분하였을 때 그 각 부분구간을 버킷(bucket)이라 한다.
② O(n)의 성능을 갖는 정렬 방법
③ n개의 버킷을 두고 각 버킷에는 (입력키 범위/n)에 해당하는 입력 데이터를 연결 리스트 형식으로 만들어 정렬하는 방식

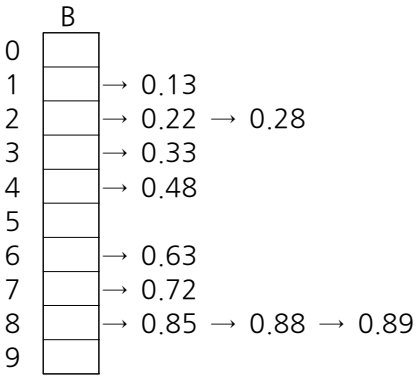
* 버킷정렬의 이해

설명의 편의상 키 값이 0과 1사이라고 가정하자. n개의키가 있을 때 구간 [0,1)을 n등분하고 이들 각각을 하나의 버킷으로 한다. 각 키를 크기에 따라 각 버킷에 분배한다면 키가 구간 내에서 균등하게 분포한다고 가정했으므로 하나의 버킷에는 하나의 키만 들어있을 확률이 높다. 만약 한 버킷에 여러 키가 들어가게 된다면 같은 버킷에 속하는 키끼리는 간단한 정렬 알고리즘으로 정렬한다. 이러한 버킷에 대하여 버킷 순서에 따라 데이터를 출력하면 정렬된 결과를 얻을 수 있다.

A: 0.85, 0.33, 0.28, 0.13, 0.48, 0.22, 0.63, 0.89, 0.72, 0.88의 10개 데이터가 있다고 하면 구간 크기가 0.1인 10개의 버킷을 만든다. A의 각 원소값에 따라 적절한 버킷에 넣고 같은 버킷에 들어갈 값은 연결 리스트에 의해 연결시킨다.



다음은 각 리스트 B[i]에 대하여 삽입정렬을 행한다. 그러면 그 결과는 다음 그림과 같다.



이제 리스트의 각 원소를 순서대로 연결하면 정렬이 완료된다.

학습내용3 : 외부 정렬

1. 외부 정렬이란?

주기억장치의 용량이 적기 때문에 자료의 양이 많은 대형 파일은 한 번에 내부 정렬로 처리하는 것이 힘들다. 이런 경우 대형 파일을 여러 개의 부분 파일로 나누어 보조기억 장치에 저장한다. 그런 다음 각각의 부분 파일을 내부 정렬에 의해 정렬하고 합병시켜 정렬한다. 이와 같이 순차적 접근(sequential access)방식의 자기 테이프나 직접 접근(direct access)방식의 자기 디스크와 같은 보조기억 장치를 이용하여 정렬하는 방식을 외부 정렬(external sort)이라 한다.
정렬하고자 하는 자료를 보조 기억 공간에 놓아 누고 일부의 자료씩 주 메모리로 가져가 정렬한 다음 병합하여 정렬하는 방식으로 자료의 양이 많을 때 사용하며 아래와 같은 종류가 있다.

1) 외부 정렬의 종류
① balanced merge sort
② cascade merge sort
③ polyphase merge sort
④ oscillating merge sort

【학습정리】

1. 계수정렬(counting sort)
계수정렬은 입력 키가 어떤 범위, 예를 들어 1부터 k사이의 정수 범위에 있다는 것을 알고 있을 때에만 적용할 수 있는 방법이다. 입력 키의 출현 횟수를 세어서 정렬하는 방식으로 중요한 성질은 동일한 키의 상대적 순서가 바뀌지 않는 안정적 정렬이라는 점이다.

2, 기수정렬(radix sort)
키의 각 자릿수에 대하여 낮은 자리에서 높은 자리의 순으로 버킷정렬을 적용하여 정렬하는 방법 각 자릿수 별로 계수정렬과 같은 안정적 정렬 알고리즘을 적용하여 정렬하는 방식으로 낮은 자리수 부터 먼저 정렬한다.

3. 버킷정렬(bucket Sort)
설명의 편의상 키 값이 0과 1사이라고 가정하자. n개의키가 있을 때 구간 [0,1)을 n등분하고 이들 각각을 하나의 버킷으로 한다. 각 키를 크기에 따라 각 버킷에 분배한다면 키가 구간 내에서 균등하게 분포한다고 가정했으므로 하나의 버킷에는 하나의 키만 들어있을 확률이 높다. 만약 한 버킷에 여러 키가 들어가게 된다면 같은 버킷에 속하는 키끼리는 간단한 정렬 알고리즘으로 정렬한다. 이러한 버킷에 대하여 버킷 순서에 따라 데이터를 출력하면 정렬된 결과를 얻을 수 있다.