2020년도 2학기 컴퓨터공학설계및실험I
X주차 정렬 알고리즘 결과보고서

20201234 홍길동

1. 실습 목적

(해당 주차의 실습 목적이 무엇인지 명료하게 작성)

정렬 알고리즘 중 거품 정렬, 삽입 정렬, 퀵 정렬을 구현하고 다양한 입력 데이터로 실행해보며 소요 시간을 비교한다.

2. 실습 구현 내용

(해당 주차의 실습을 어떻게 구현했는지 함수 단위로 작성)

(1) void BubbleSort (int list[], int n)

거품 정렬을 사용하여 입력으로 들어오는 정수형 배열 list[]를 오름차순으로 정렬하는 함수.
input : 정렬하기 위한 정수형 리스트 (int) list[], 리스트의 크기 (int) n
return : 없음

(2) void InsertionSort (int list[], int n)

삽입 정렬을 사용하여 입력으로 들어오는 정수형 배열 list[]를 오름차순으로 정렬하는 함수.
input : 정렬하기 위한 정수형 리스트 (int) list[], 리스트의 크기 (int) n
return : 없음

(3) void QuickSort (int list[], int left, int right)

퀵 정렬을 사용하여 입력으로 들어오는 정수형 배열 list[]를 오름차순으로 정렬하는 함수.
input : 정렬하기 위한 정수형 리스트 (int) list[], 리스트의 왼쪽 끝 index (int) left, 리스트의 오른쪽 끝 index (int) right
return : 없음

(4) int partition (int list[], int left, int right)

pivot을 기준으로 리스트를 2개로 나누는 함수. 이때 pivot보다 작은 값은 왼쪽으로, pivot보다 큰 값은 오른쪽으로 옮긴다.
input : 나누기 위한 정수형 리스트 (int) list[], 리스트의 왼쪽 끝 index (int) left, 리스트의 오른쪽 끝 index (int) right
return : 현재 pivot의 index

3. 실습 환경

(실습을 수행한 컴퓨터의 사양 및 사용한 프로그램을 서술. 만약 CSPRO 환경이라면 간단히 CSPRO 서버로 서술)

APPLE Macbook Pro-13 2016
CPU : Intel Core i5-6267U 2.9Hz (Skylake)
RAM : 8GB 2133Mhz LPDDR3
GPU : Intel Iris Graphics 550 1536MB
OS : macOS 10.15 Catalina
IDE : Xcode Version 11.6

4. 실습 결과 및 분석

(실습 결과에 대해 이론적인 예상과 비교하여 분석)

세 정렬의 효율을 분석하기 위해, 입력 리스트를 다음과 같이 크게 3가지 경우로 나누어 각각의 정렬을 수행하는 시간을 비교하였다. 리스트의 크기는 10000으로 설정하였다.

Case 1	리스트의 숫자가 무작위로 나열된 경우
Case 2	리스트의 숫자가 역순으로 정렬되어 있는 경우
Case 3	리스트의 대부분이 이미 정렬되어 있고, 일부분만 위치만 바뀌어 있는 경우

(1) Case 1의 수행 결과

```
Choose Case : 1
Duration of Bubble Sort : 0.249286
Duration of Insertion Sort : 0.126365
Duration of Quick Sort : 0.099794
Program ended with exit code: 0
```

(2) Case 2의 수행 결과

```
Choose Case : 2
Duration of Bubble Sort : 0.249337
Duration of Insertion Sort : 0.126721
Duration of Quick Sort : 0.101182
Program ended with exit code: 0
```

(3) Case 3의 수행 결과

```
Choose Case : 3
Duration of Bubble Sort : 0.258601
Duration of Insertion Sort : 0.127517
Duration of Quick Sort : 0.100078
Program ended with exit code: 0
```

모든 Case를 통틀어서 퀵 정렬이 가장 빠른 수행 시간을 보였고, 버블 정렬이 가장 느린 수행 시간을 보여주었다. 이론적으로 버블 정렬과 삽입 정렬은 같은 시간 복잡도를 가지고 있지만, 실제 수행 시간은 두 배의 차이가 남을 확인할 수 있었다.

퀵 정렬의 경우 이론적으로는 시간 복잡도가 $O(n\log n)$으로 가장 빠르지만, 재귀 호출 방식을 사용하고 있으므로 시간 낭비가 있을 것으로 예상되었다. 하지만 그럼에도 불구하고 모든 리스트 상황에서 삽입 정렬보다 20% 정도의 빠른 수행 속도를 보여주었다.

5. 과제

(해당 주차에 주어진 과제의 수행 과정과 결과를 위와 같이 자세하게 서술)

6. 결론

(실습 및 과제를 통해 알게된 내용을 요약하여 서술)

이번 주차의 실습에서는 여러 정렬 알고리즘을 구현하여 그 성능을 비교하는 것이 목적이었다. 크게 버블 정렬, 삽입 정렬, 퀵 정렬의 수행 속도를 비교하였는데, 이론적으로나 실제 실행 결과를 보나 퀵 정렬이 가장 빠른 수행 속도를 보여줌을 알 수 있었다.

하지만 삽입 정렬은 $O(n^2)$의 시간 복잡도를 가지고 있음에도 불구하고 버블 정렬에 비해 상당히 빠른 수행 시간을 보여주었다. 일반적인 경우에는 퀵 정렬이 가장 효율이 좋기 때문에 퀵 정렬을 사용하는 것이 좋으나, 퀵 정렬은 pivot을 어떻게 정하는 지에 따라 효율이 변하는 문제도 있고, 구현 방법이 다른 정렬에 비해 복잡하다. 따라서 간단한 문제에서는 구현이 쉬운 삽입 정렬을 사용하는 것이 하나의 대안이 될 수도 있다.