Master of Sorting

20151547 박재상

Method1 : Bubble Sort

Method2 : Quick Sort

Method3 : Merge Sort

Method4 : My Sort

우선 본인이 만든 random sequence testcase를 통해서 각각의 sorting방식이 얼마나 시간이 걸리는지 측정해보았다. 실험을 진행한 환경은 마지막에 사진으로 첨부하겠다.

각각의 시간의 단위는 ms이다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| number | Method 1 | Method 2 | Method 3 | Method 4 |
| 5 | 0.006 | 0.007 | 0.008 | 0.006 |
| 10 | 0.008 | 0.008 | 0.009 | 0.008 |
| 50 | 0.038 | 0.015 | 0.043 | 0.014 |
| 100 | 0.125 | 0.025 | 0.112 | 0.024 |
| 1000 | 9.98 | 0.28 | 9.4 | 0.3 |
| 5000 | 177 | 1.52 | 155 | 1.85 |
| 10000 | 676 | 3.43 | 532 | 3.74 |
| 50000 | 12851 | 22 | 9224 | 24.686 |
| 100000 | 53034 | 53 | 36345 | 56 |

다음은 reverse sorted testcase를 이용하여 각각의 sorting방식이 얼마나 시간이 걸리는지 측정해보았다. 이 testcase는 1부터 n까지의 수를 거꾸로 나열한 것이다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| number | Method 1 | Method 2 | Method 3 | Method 4 |
| 5 | 0.009 | 0.007 | 0.007 | 0.006 |
| 10 | 0.008 | 0.007 | 0.008 | 0.019 |
| 50 | 0.036 | 0.026 | 0.039 | 0.025 |
| 100 | 0.07 | 0.044 | 0.118 | 0.076 |
| 1000 | 12 | 6.3 | 9.2 | 6.35 |
| 5000 | 195 | 115 | 158 | 150 |
| 10000 | 676 | 368 | 531 | 518 |
| 50000 | 11855 | 6325 | 9258 | 8972 |
| 100000 | 47618 | 24570 | 36482 | 24809 |

각각의 number에 대해서 5개의 testcase를 만들어서 실험을 진행하였고, 결과값은 평균값으로 작성하였다. 물론 여러가지 변수들이 있겠지만(실험 시간에 따른 서버사용자의 변화, 와이파이로 연결된 불안정한 네트워크상황)

Method1 (Bubble Sort)의 경우에는 random으로 만든 sequence와 sort하기 가장 어려운 reverse sequence와의 차이가 거의 없고 심지어 reverse sequence가 더 빠르게 진행되었다.

Method3은 reverse sequence와 random sequence testcase간의 차이가 거의 없었다.

Method2는 Method1,3과 시간차이가 확연하게 드러났다. 본인이 random testcase를 만들 때,

rand()%1000-500;으로 각각의 값들을 만들어냈는데, number가 커질수록 method1,3과의 차이가 확연하게 드러났다. 괜히 quicksort가 아니다. 다만 reverse sequence의 경우에는 다른 sort algorithm보다는 빨랐지만, random testcase보다는 확연하게 느려진 모습이다.

본인이 만든 Method4는 Method1~3을 직접 구현하고 실험을 먼저 진행했는데, quicksort의 성능에 너무나도 놀라서 이를 응용하기로했다.

number값이 1000000이하이면 그냥 quicksort를 진행했다. number값이 1000000보다 크다면, 충분히 큰 상황이기 때문에, 전체적인 sequence의 경향성을 읽어보는 과정을 추가했다. Reverse sequence인 상황에서 quicksort의 성능이 급격히 떨어지기 때문에, 전체적인 sequence가 reverse sequence의 경향에 가깝다면, 입력받은 sequence를 중간점을 기준으로 대칭시켰다.

입력받은 sequence를 A라고 한다면,

for(int i=0;i<N/2;i++){

swap(A,i,N-i);

}의 과정을 추가 시켜서 진행하였다.

