CSE3081 (2반): 알고리즘 설계와 분석

[숙제 1]

담당 교수: 임 인 성

20171682 임정호

1. **실험환경 기술**

OS: Windows 10 Pro

Cpu: Intel® Core™ i5-7200U CPU @ 2.50Ghz 2.71Ghz

Ram: 8.00GB

Compiler: Visual Studio 19 Release Mode/x64 Platform

1. **세 개의 알고리즘 (Algorithm 3, Algorithm 4, Algorithm 5) 각각에 대하여 자신이 사용한 방법을 몇 줄 이내의 문장으로 간략히 요약하라. 만약 세 개 모두 구현을 하지 못했다면, 어떤 알고리즘을 구현하였는지를 정확히 밝힐 것.**

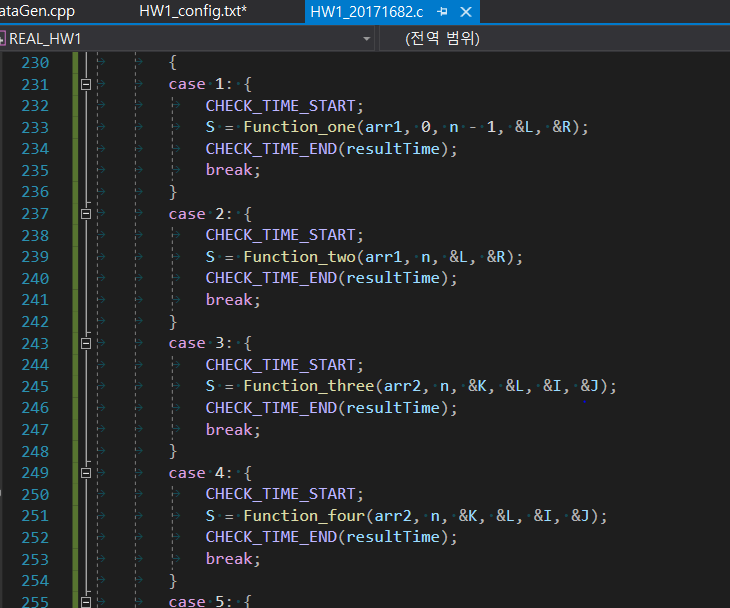
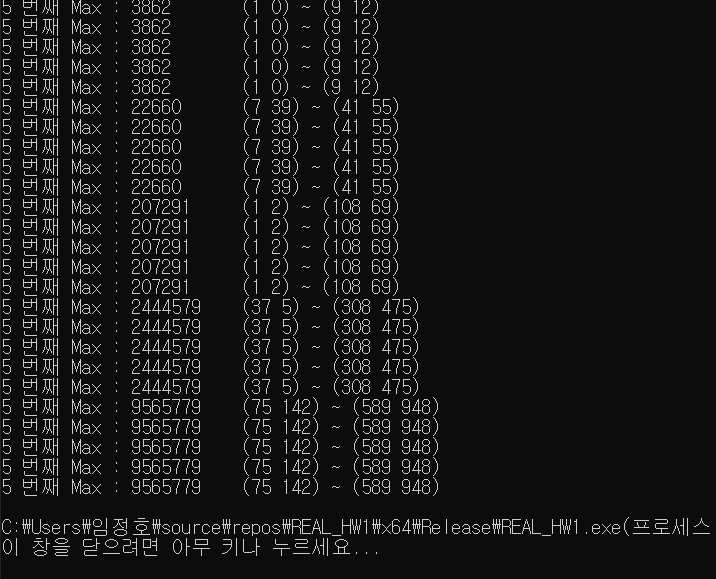
우선 세 개의 알고리즘들은 2D array에 대해 Maximum Sum Subrectangle Problem을 해결하는 알고리즘이다.

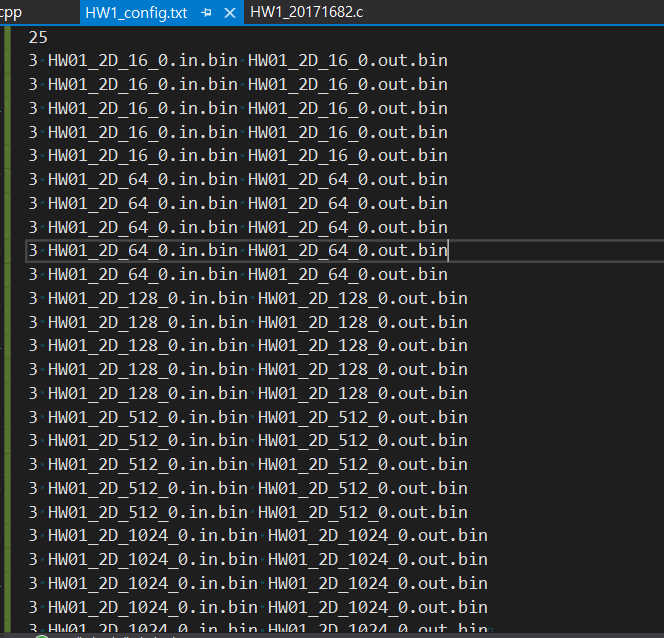
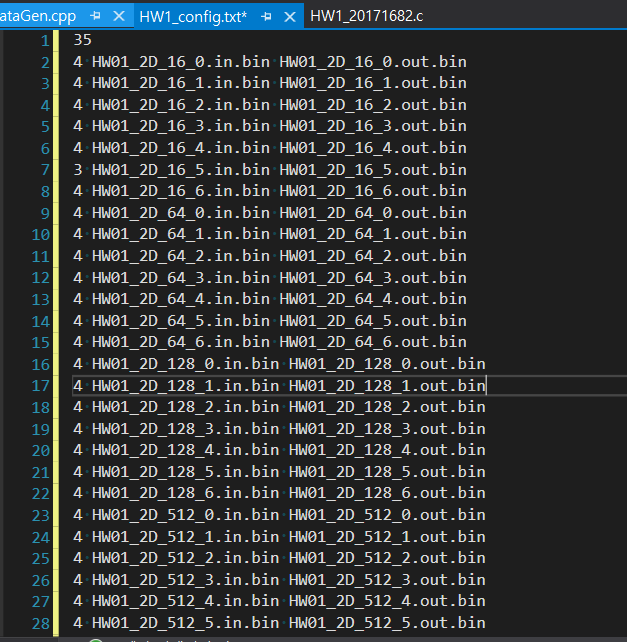
알고리즘 3의 경우 Summed Area Table 방법을 이용하는 알고리즘이다. (0,0)에서 (x,y)까지의 Subrectangle의 합을 구해 2차원 배열 T에 저장한다. (k,i)에서 (l,j)까지의 Subrectangle의 합 S(i,j,k,l)은 T(l,j)-(T(k-1,j)+T(l,i-1))+T(k-1,i-1)이므로 k와 i는 0부터 n-1까지 돌고, l과 j는 k, i부터 n-1까지 도는 4중 for문을 이용하여 최대 합을 찾을 수 있다.

알고리즘 4와 5의 경우, Subrectangle의 열은 i부터 j이란 점에 착안하여, i는 0, j는 i에서 각 n-1까지 도는 2중 for문으로 고정할 수 있다. 이 때 각 행의 i열부터 j열까지의 합을 배열 temp에 저장하면, 2차원 배열의 MSS 문제를 1차원 배열 temp의 MSS 문제로 단순화시킬 수 있다. 알고리즘 4와 알고리즘 5는 temp의 MSS를 구하는 방법의 차이로 둘의 시간 복잡도가 다르다.

1. **다음과 같이 실제 수행 시간에 대한 실험을 진행한 후, 그 결과를 보고서에 기술하라**
2. **실험 결과의 정확성을 높이는 방법**
3. 동일한 크기 n에 대해 최소 5개 이상의 서로 다른 데이터에 대하여 실험을 수행한 후 평균을 낸다.
4. 주어진 동일한 데이터에 대하여 다섯 번 이상 실험을 한 후 평균을 낸다.

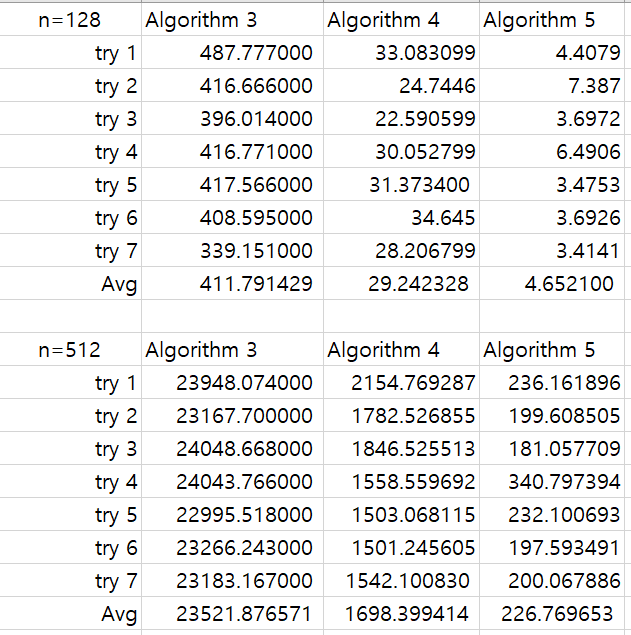
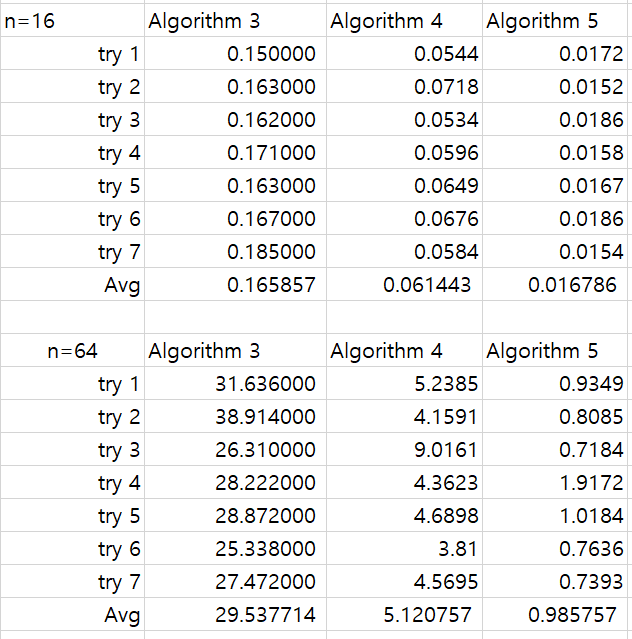
실험 결과의 정확성 및 신빙성을 높이기 위해 각 알고리즘 당 7개의 동일한 크기 n(16,64,128,512,1024)의 서로 다른 데이터의 bin파일들을 dataGen.cpp를 통해 생성하여 실험을 수행하였고, 이후 다시 동일한 데이터에 대하여 실험을 5번 반복 수행하였다. Timing.cpp를 활용하여 함수 호출 전후에 CHECK\_TIME\_START와 CHECK\_TIME\_END를 붙여 시간을 측정하였다.

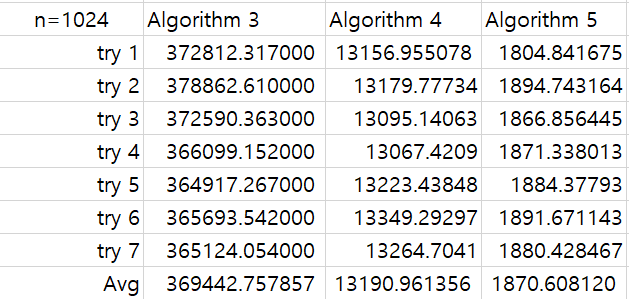




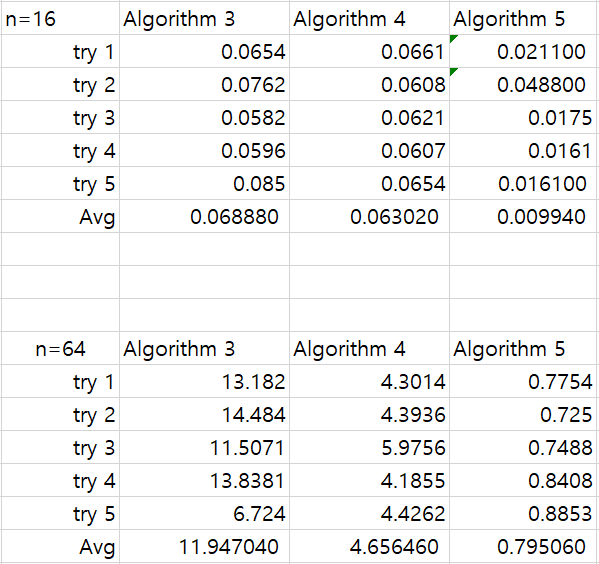
(동일한 n에 대한 서로 다른 데이터) (동일한 데이터에 대한 여러 번 수행)

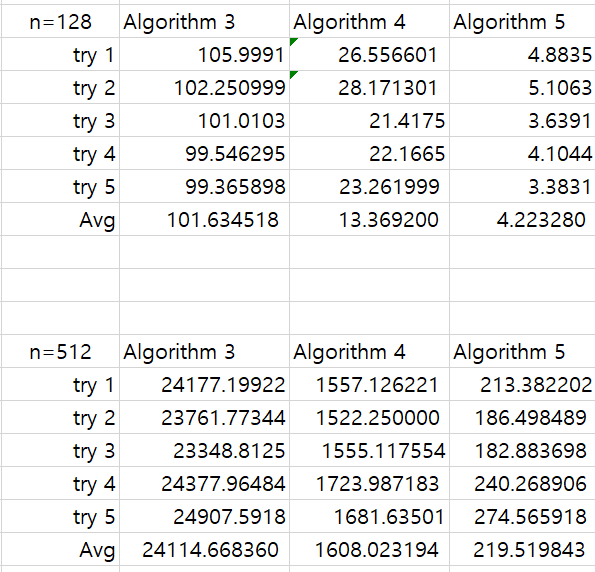
다음은 (a)의 수행 결과를 정리한 표이다. (단위는 ms이다.)

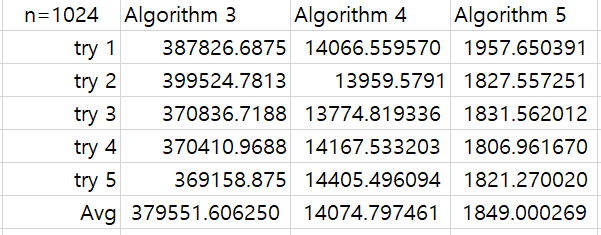




다음은 (b)의 수행 결과를 정리한 표이다. (단위는 ms이다.)







1. **세 알고리즘의 이론적인 시간 복잡도와 수행 시간 간의 관계 분석**

세 알고리즘의 이론적인 시간 복잡도는 O(N^4), O(N^3logN), O(N^3)이다. 이론적인 시간 복잡도이기 때문에 세 알고리즘의 실제 수행 결과를 상대적으로 비교해야 한다.

이론적으로는 Algorithm 5는 Algorithm 3에 비해 수행 시간이 N배 정도 차이가 날 것이고, Algorithm 4와 Algorithm 5는 logN배 정도 차이가 날 것으로 예상한다.

1. 같은 크기의 입력 값이더라도 수행 시간이 다 다르다는 것을 알 수 있었다. 그리고 N의 크기가 작을 때는 세 알고리즘의 수행 시간은 크게 차이 나지 않았지만, 크기가 커질수록 수행 시간의 차가 현명하게 난 점을 확인할 수 있다.

Algorithm 5와 Algorithm 3은 이론적으로 N배 차이가 나는데, 수행 결과를 비교해보면 값이 정확히 N배 차이가 나지 않는 것을 확인할 수 있다. N = 1024의 경우를 예를 들면, 369442.757857 / 1870.608120을 계산해보면 약 197.5가 나오는데 이는 N의 크기 1024와는 오차가 있음을 알 수 있다.

Algorithm 3의 경우 n=512의 수행시간과 n=1024일 때 수행 시간은 시간 복잡도가 O(n^4)이라 2^4배 만큼 차이가 날 거로 이론상 생각했지만 실제 계산 결과는 약 15.7배 정도 차이가 났다.

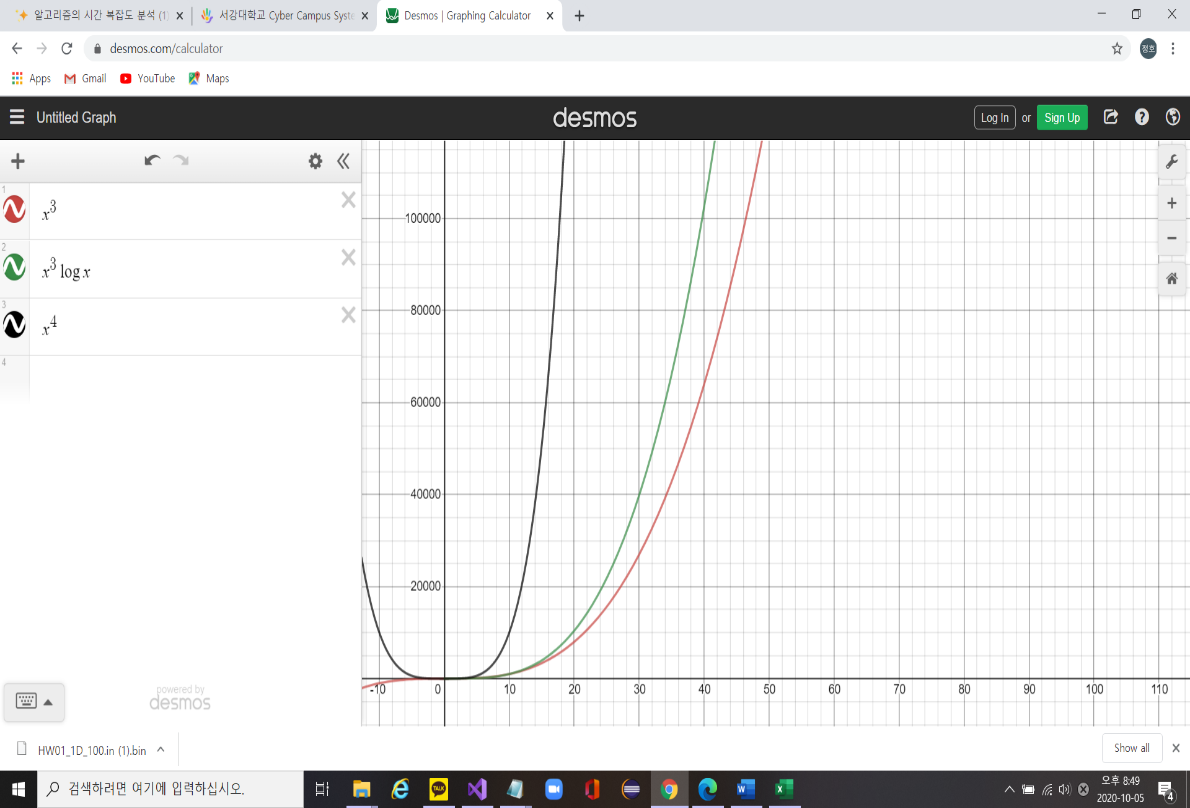
Algorithm 4의 경우 n=512의 수행시간과 n=1024일 때 수행 시간은 시간 복잡도가 O(n^3logN)이라 2^3\*(10/9)배(약 8.88)만큼 차이가 날 거로 이론상 생각했지만 실제 계산 결과는 약 7.76배 정도 차이가 났다.

Algorithm 5의 경우 n=512의 수행시간과 n=1024일 때 수행 시간은 시간 복잡도가 O(n^3)이라 2^3배 만큼 차이가 날 거로 이론상 생각했지만 실제 계산 결과는 약 8.27배 정도 차이가 났다.

.

1. 동일한 데이터를 반복 수행하였지만 수행 시간은 다 다르게 나왔음을 알 수 있었고, 위 알고리즘들을 비교할 때 이용한 N의 크기에 맞게 각 알고리즘들을 다시 비교해보았다.

Algorithm 3의 경우 계산 결과는 약 15.74배, Algorithm 4의 경우 약 8.75배 Algorithm 5의 경우 약 8.44배 정도 차이가 났음을 알 수 있었다.



1. **분석 및 의견**

이론 상의 예상 결과와 실제 계산 결과가 상이한 이유는 여러 가지 있다. 우선, 애초에 시간 복잡도를 계산할 때 상한으로 계산하기 때문이다. 예를 들면 O(n^4)의 알고리즘의 경우 O(n^4)보다 낮은 계산 과정이 여럿 있어도 추가로 계산해주지 않는다. 둘째로, 컴퓨터의 운영체제, 컴파일러, 하드웨어 등의 요소의 성능 차이로 수행 시간이 바뀔 수 있다. 또한 입력 데이터 값의 특성에도 달라질 수 있다. 동일한 데이터에 대해 반복 수행했을 때 실행할 때마다 수행 시간이 다른 이유는 CPU에 있는데, 알고리즘을 수행할 때마다, CPU는 매번 같은 일을 하고 있지 않기 때문이다.