Algorithm homework 1

공과대학 산업공학과

2018-10786 서규혁

1. Measure the actual time usage of the two algorithms on inputs of various sizes and compute the ratio of the constants hidden in the asymptotic time complexities of the two algorithms.

Various sizes의 inputs들을 가지고 Deterministic Selection, Randomized Selection을 진행하였고 각각 알고리즘들이 수행되는동안의 시간들을 ms의 단위로 측정하였다. 본인이 actual time 을 측정하기 위해 사용한 예제Input들은 총 10개로 input1.txt, input2.txt … , input10.txt들이다. 각각의 size는 10,000개, 20,000개, … , 100,000개이며 각 input들의 element는 python의 random.randint(-10000,10000)을 통해 -10000부터 10000까지의 임의의 정수들로 구성되었다. i-th element를 알려주는 i또한 random.randint(1,10000)을 이용해 1부터 10000까지의 양의 정수들 중 임의로 하나가 선택되어지게 하였다.

측정한 결과는 아래의 표와 차트와 같다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Input Size** | **Selection 방식에 따른 소요 시간 측정 ( 단위 : ms )** | | | | | |
| deterministic 1차 | deterministic 2차 | deterministic 3차 | randomized  1차 | randomized 2차 | randomized 3차 |
| 10000 | 22.60ms | 19.68ms | 39.95ms | 6.75ms | 4.89ms | 4.68ms |
| 20000 | 49.78ms | 47.75ms | 45.11ms | 12.71ms | 24.89ms | 23.41ms |
| 30000 | 75.47ms | 102.31ms | 108.76ms | 23.78ms | 13.01ms | 14.90ms |
| 40000 | 80.55ms | 118.13ms | 86.37ms | 24.97ms | 20.94ms | 37.86ms |
| 50000 | 109.02ms | 116.19ms | 143.45ms | 43.62ms | 50.90ms | 31.75ms |
| 60000 | 224.45ms | 166.36ms | 131.53ms | 39.80ms | 46.48ms | 47.07ms |
| 70000 | 263.17ms | 158.25ms | 236.61ms | 30.62ms | 54.99ms | 39.31ms |
| 80000 | 176.15ms | 190.32ms | 181.96ms | 28.83ms | 49.78ms | 29.00ms |
| 90000 | 234.93ms | 168.16ms | 190.59ms | 64.27ms | 53.81ms | 48.44ms |
| 100000 | 230.50ms | 367.52ms | 230.62ms | 28.83ms | 37.07ms | 77.61ms |

Figure 1 같은 Input1,2, … , 10에 대해 같은 방식의 algorithm을 3번씩 테스트한 결과

|  |  |  |
| --- | --- | --- |
| **Input Size** | **Deterministic** | **Randomized** |
| 10000 | 27.41ms | 5.44ms |
| 20000 | 47.55ms | 20.34ms |
| 30000 | 95.51ms | 17.23ms |
| 40000 | 95.02ms | 27.92ms |
| 50000 | 122.89ms | 42.09ms |
| 60000 | 174.11ms | 44.45ms |
| 70000 | 219.34ms | 41.64ms |
| 80000 | 182.81ms | 35.87ms |
| 90000 | 197.89ms | 55.51ms |
| 100000 | 276.21ms | 47.84ms |

Figure 2 같은 input에 대해 3번씩 측정한 time을 average값으로 정리한 표

Figure 3 Figure2의 결과를 차트로 표현하고 추세선 식을 표현

알다시피 Deterministic Selection의 경우 Time complexity가 theta(n)을 따르고, Randomized Selection의 경우 average Time을 생각했을 때 Time complexity가 theta(n)을 따르는 것을 알 수 있다. 각각의 input들에 대해 3번씩 반복하여 평균 time을 구했고 이를 선형으로 추정하여 추세선을 구한 결과는 위의 차트에 표시되어있다. Deterministic Selection의 경우, Time complexity가 약 25.166의 ratio로 list의 elements 갯수에 asymptotic하게 비례하는것을 알 수 있다. Randomized Selection의 경우, Average Time complexity로 가정하면 약 4.6332의 ratio로 list의 elements 갯수에 asymptotic하게 비례하는것을 알 수 있다. 여기서 하나 주목해서 보면 좋을 부분은 R제곱값인 결정계수이다. 추세선의 설명력, 즉 추세선이 얼마나 해당 데이터를 잘 설명하고있는가의 정도를 보여준다. Deterministic의 경우 theta(n)을 따르므로 R제곱값이 약 0.9237로 선형 추세선이 해당 데이터를 잘 설명하고 있음을 보여준다. 그러나 Randomized Selection의 경우, average case로 가정하였기에 R제곱값이 약 0.8 근처에 머무는 것을 보여준다. 이는 Randomized Selection의 경우 partition의 기준이 되는 Pivot index가 무엇이 되느냐에 따라 partition을 통해 다음 step으로 넘어갈 때, partition 작업이 매우 훌륭했을수도 있고 원소 1개만 걸러내는 매우 비효율적인 작업일 수도 있기에 R제곱값이 떨어지는 것을 확인할 수 있다.

2. Explain how your checker program works

Checker program은 간단한 형태로 구현되어있다. Input으로 list A의 길이인 n, nrodml elements로 이루어진 list인 A, i-th element에서의 i, randomized selection의 결과로 나온 i-th element를 나타내는 random, deterministic selection의 결과로 나온 i-th element를 나타내는 deter, file naming에 사용될 output\_file\_name이다.

small\_r, big\_r, small\_d, big\_d가 check함수안에 선언된 변수인데, 각각 list A 내에서 random, deter값보다 작고 큰 원소들의 수를 카운트하는데에 사용된다.

j가 0부터 n-1까지 돌면서, A[j]와 random, deter값을 비교하고 small\_r, big\_r, small\_d, big\_d의 의미에 맞게 각각 카운트를 한다. For문이 종료되면 i와 small\_r, big\_r, small\_d, big\_d를 비교하여 옳고 그름을 판정하게 된다.

n에 영향을 받는 부분은 j가 range(n)의 for문을 도는 경우이므로 checker program은 n에 linear한 time complexity임을 알 수 있다.

3. Write down the environment you run your program and how to run it.

작업환경은 python 3.8.9이며 glob, random, time을 import했다.

해당 프로그램을 run하는 방법은, terminal을 켜고 main.py가 있는 경로에 접속한다. 해당 경로에서 $ python main.py 를 하면 main.py가 실행된다.