Sabancı University
Faculty of Engineering and Natural Sciences

CS301 – Algorithms

Homework 2

Due: 08-11-2021 @ 23:55

Notes

- No homework will be accepted if it is not submitted using SUCourse.
- Write your solutions on a piece of paper and then upload your solutions to SUCourse.
- The files you upload must be the scanned form of your papers (not photographs). You can use scanner apps available for phones.
- LATE SUBMISSION POLICY:
  Late submission is allowed subject to the following conditions:
  - Your homework grade will be decided by multiplying what your normal grade (i.e. what you get from your answers) by a "submission time factor (STF)".
  - If you submit on time (i.e. before the deadline), your STF is 1. So, you don't lose anything.
  - If you submit late, you will lose 0.01 of your STF for every 5 mins of delay.
  - We will not accept any homework later than 500 mins after the deadline.
  - SUCourse's timestamp will be used for STF computation.
  - If you submit multiple times, the last submission time will be used.

---

1. Consider the Radix sort that is used to sort numbers.
   (a) Explain in at most 3 sentences, how would you modify the Radix sort in order to sort the list of words in alphabetical order.
   (b) Sort the following list of words using the modified algorithm and give the form of the array after every iteration of the main loop of the Radix sort. Keep the given format in your answers.
   init: [HUSNU, FURKAN, ALI, EMRE, EFE]
   iter0: . . .
   (c) Find the time complexity of the modified algorithm in big–O notation (as tight as possible).

2. Consider the following algorithm, which is exactly the same algorithm we considered in the class for the Selection problem with one difference; Instead of dividing the input (which has n elements) into groups of 5 elements, we divide the input into groups of k elements, where k is a parameter of the algorithm.

```
1     float WCL_Select (A, first, last, i, k) {
2         if (first == last)
3              return A[first]; // i=1 in this case
4
5         Divide n elements into groups of k elements; //g_1 g_2 ... g_(n/k)
6         Find the median m_i each group g_i;
7
8         // Use WCL_Select to find M, the median of the medians of all the groups
9         M = WCL_Select([ m_1, m_2, ..., m_(n/k)], 1, n/k, n/2k, k)
10
11        mid = NewPartition(A, first, last, M); // it partitions around M
12        mid_and_less = mid - first + 1;
13
14        if (mid_and_less == i) // we may be lucky
15            return A[mid];
16
17        if (i < mid_and_less) // it is in the left subarray
18            return (WCL_Select(A, first, mid-1, i, k));
19
20        return (WCL_Select(A, mid+1, last, i-mid_and_less, k));
21    }
```

(a) Derive the recurrence for the running time of the algorithm given above. Note that the recurrence will be parametric in k as well. After you derive this recurrence, substituting $k = 5$ in the recurrence should give the recurrence we had in the class.

(b) Now consider the algorithm above and suppose that we modify the line 9 as follows:

```
// Use WCL_Select to find M, the (1/4)th of the medians of all the groups
M = WCL_Select([ m_1, m_2, ..., m_(n/k)], 1, n/k, n/4k, k)
```

Informally, instead of using the median of the medians, we use (1/4)th of the medians as M.

For this modified algorithm,

1. Write down the recurrence for the running time.
2. Show that running time is super linear when $k = 5$.
   (Hint: You can show that the running time is in $\Omega(n \lg n)$ for this purpose)
3. Find the smallest odd integer value for k, for which the running time of this algorithm is linear. Write the concrete recurrence for the value of k you suggest and show that running time is linear in this case.