**Design and Analysis of Algorithms: Homework**

**Released/Deadline**: Dec 17, 2020/Jan 15, 2021

**Send to**: li.ning@siat.ac.cn

**Title Format**: homework - student name

**Q1.** Prove that *n*5 = *O*(2*n*).

Proof:

Assume

*T(n)= n5, f(n)= 2n*

For bit-O annotation, we need to find C>0 and N>0 that

*T(n) ≤C·f(n), ∀n>N.*

Let C=1, N=50, then

*n5≤1·2n , ∀ n≥50.*

So

*n5 = O(2n).*

**Q2.** Prove that 100*n* + log*n* = *O*(*n* + (log*n*)2).

Proof:

Assume

*T(n)= 100n + logn, f(n)= n + (logn)2*

Let C=100, N=3, then

*Cf(n)-T(n)= 100(logn)2-logn*

We have (logn)2≥logn≥loge=1 when n≥3, then

*100(logn)2-logn > 0, ∀ n≥3.*

Means

*Cf(n) ≥T(n), ∀ n≥3.*

So

*100n + logn = O(n + (logn)2)*.

**Q3.** Consider, with constants *d >* 0 and *ai >* 0 for all *i*. Prove that for any *k* ≥ *d*, it holds that

*T*(*n*) = *O*(*nk*)*.*

**Q4.** Show that for any real constant *a >* 0 and *b >* 0, it holds that

(*n* + *a*)*b* = *O*(*nb*)*.*

You can assume that *a* and *b* are all integers.

**Q5.** Describe the details of applying counting-sort with the following numbers,

1*,*5*,*4*,*7*,*2*,*2*,*1*,*1*,*4*,*3*,*1*,*2*,*9

which are selected from {0*,*1*,*2*,*··· *,*9}.

**Q6.** Recall that given a sequence *A* = *a*0*a*1 ···*an*−1 of size *n*, a subsequence has the form of *ai*0*ai*1 ···*aik*−1 with *i*0 *< i*1 *<* ··· *< ii*−1. Given a string *B* = *b*0*b*1 ···*bm*−1 with *m < n*, describe an algorithm to determine if *B* is a subsequence of *A*. Analyze the complexity of your algorithm in the big-*O* notation.

**Q7.** *T*(*n*) = 10 · *T*(*n/*3) + *n*2*.*5. Determine the order of *T*(*n*), in the big-Θ notation.

**Q8.** Let *F*(*n*) be the number of “hello” printed by algorithm *f*(*n*). For example, *F*(0) = *F*(1) = 0, *F*(2) = 1.

**Algorithm1:**

*f*

(

*n*

)

**if**

*n>*

1

**then**

print“Hello”;

*f*

(

*n/*

2)

;

*f*

(

*n/*

2)

;

Determine the order of *F*(*n*) in the big-Θ notation

**Q9.** Give the pseudocode of the dynamic programming algorithm for solving the longest common subsequence problem. **Q10.** Give the pseudocode of the dynamic programming algorithm for solving the sequence alignment problem.

**Q11.** Calculate the length of the shortest paths from *s* to *t*, using Dijkstra’s algorithm (for the directed graph) step by step.

*s*

*A*

*t*

*B*

*C*

*D*

*E*

8

3

2

3

1

1

6

2

1

1

2

2

1

**Q12.** Find a minimum spanning spanning tree.

*A*

*B*

*C*

*D*

*E*

*F*

1

3

3

5

1

6

2

4

1

**Q13.** Find a maximum bipartite matching.

*A*

*B*

*C*

*D*

*~~H~~*

*I*

*J*

*~~K~~*

2