CS340 – Advanced Data Structures and Algorithm Design – Fall 2020
Assignment 4 – October 2, 2020

Dr. Sandra Zilles, Department of Computer Science, University of Regina

answer key

Problem 1 (2+4 marks).

(a) see hand-written sheets

(b) Placing all single-node heaps on a queue takes $O(N)$ time. [0.5 marks]

In the first step, $N/2$ pairs of heaps of size 1 are dequeued (for $O(1)$ per pair) and merged (for $O(\log(2))$ per pair). In the second step, $N/4$ pairs of heaps of size 2 are dequeued (for $O(1)$ per pair) and merged (for $O(\log(4))$ per pair). In the third step, $N/8$ pairs of heaps of size 4 are dequeued (for $O(1)$ per pair) and merged (for $O(\log(8))$ per pair). ... Finally, in the last step, $N/N = 1$ pair of heaps of size $N/2$ is dequeued (for $O(1)$) and then merged (for $O(\log(N))$). [2 marks]

In total, we get a worst case runtime of

$$O(N/2\log(2) + N/4\log(4) + \cdots + N/N\log(N)) = O(N\sum_{i=1}^{\log(N)} \frac{1}{2^i}\log(2^i)) = O(N\sum_{i=1}^{\log(N)} \frac{i}{2^i}) \text{ which is in } O(N)\,.$$

[1.5 marks]

Problem 2 (4 marks).

Proof by induction on k.

induction base: $k = 1$. In any binomial tree in B_1, the root has only one child, which consists of a single node. This child is a binomial tree in B_0. [1 mark]

inductive hypothesis: Suppose that, for some fixed k, in any binomial tree in B_k, the root has trees from B_0, B_1, ..., B_{k-1} as children. [0.5 marks]

inductive step: $k \rightsquigarrow k+1$. Consider a binomial tree $\mathcal{T}$ in B_{k+1}.

By definition, such a tree consists of a binomial tree $\mathcal{T}_1$ from B_k, to the root of which we append a binomial tree $\mathcal{T}_2$ from B_k. [1 mark]

The children of $\mathcal{T}$ are therefore $\mathcal{T}_2$ (which is a tree from B_k) and all the children of $\mathcal{T}_1$ (which, by inductive hypothesis, are trees from B_0, B_1, ... B_{k-1}). [1 mark]

Therefore, $\mathcal{T}$ has trees from B_0, B_1, ..., B_k as children. [0.5 marks]

Problem 3 (5 marks). Since `a[i]` and `a[i+k]` were in the wrong order initially, the pair $(i, i+k)$ is an inversion. If we exchange these two elements, they will be in the correct order. Hence at least one inversion is removed when swapping `a[i]` and `a[i+k]`. [0.5 marks]

Since we swap only `a[i]` and `a[i+k]`, each inversion removed by that swap must contain the index i or the index $i+k$ (or both). [1 mark]

The elements in `a[0..i-1]` cannot be involved in any removed inversion, because their relative position to `a[i]` and `a[i+k]` is not changed by the swap. The same holds for the elements in `a[i+k+1..MAX]`. [1 mark]

Hence every removed inversion, except for the pair $(i, i+k)$, contains exactly one of the indices i and $i+k$ and exactly one of the indices in $\{i+1, \ldots, i+k-1\}$. Since the latter set has $k-1$ elements, we can form $2(k-1) = 2k-2$ such inversions. Adding the one inversion $(i, i+k)$ gives the desired upper bound of $2k-1$ on the number of removed inversions. [1.5 marks]

An array meeting the lower bound would for instance have the elements $1,\ 2,\ 3,\ \ldots,\ k$ in positions $i,\ i+1,\ i+2,\ \ldots,\ i+k-1$, and then the element 0 in position $i+k$. The relevant inversions before swapping are

$$(i, i+k),\ (i+1, i+k),\ (i+2, i+k),\ \ldots, (i+k-1, i+k)\,.$$

After the swap, $(i, i+k)$ is no longer an inversion, but the pairs

$$(i+1, i+k),\ (i+2, i+k),\ \ldots, (i+k-1, i+k)$$

remain inversions. [0.5 marks]

An array meeting the upper bound would for instance have the elements $k+1,\ 2,\ 3,\ \ldots,\ k,\ 1$ in positions $i,\ i+1,\ i+2,\ \ldots,\ i+k$. The relevant inversions before swapping are

$$(i, i+1),\ (i, i+2),\ (i, i+3),\ \ldots, (i, i+k)$$

as well as

$$(i+1, i+k),\ (i+2, i+k),\ \ldots, (i+k-1, i+k)$$

for a total of $2k-1$ inversions. After the swap, we get the elements in `a[i..i+k]` all in order, so none of these inversions remain. [0.5 marks]

Problem 4 (4+5 marks).

(a) see hand-written sheets

(b) [give partial marks for partially correct code: 2 marks for the first correct sorting algorithm with the first version of a gap sequence, 1 more mark for the version with the seond gap sequence, and 1 more mark for the version with the third gap sequence. 1 mark for the input/output examples including number of comparisons.]

CS 340 2013 10 Assignment 4

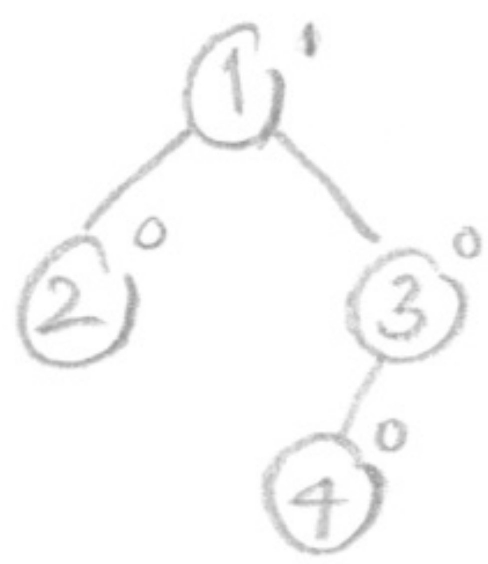
Problem 1(a) [2 marks]

insert 1:

insert 2:

insert 3:

insert 4:



insert 5:

insert 6:

insert 7:

[subtract 1 mark for each mistake]

Problem 4(a) gap sequence 7, 3, 1 [0.5 marks]

										# comparisons [1.5 marks]
g=7	77	17	66	19	30	24	64	14	23	1
	14	17	66	19	30	24	64	77	23	1
g=3	14	17	66	19	30	24	64	77	23	1
	14	17	66	19	30	24	64	77	23	1
	14	17	66	19	30	24	64	77	23	1
	14	17	24	19	30	66	64	77	23	1
	14	17	24	19	30	66	64	77	23	1
	14	17	24	19	30	66	64	77	23	2
g=1	14	17	23	19	30	24	64	77	66	1
	14	17	23	19	30	24	64	77	66	1
	14	17	23	19	30	24	64	77	66	2
	14	17	19	23	30	24	64	77	66	1
	14	17	19	23	30	24	64	77	66	2
	14	17	19	23	24	30	64	77	66	1
	14	17	19	23	24	30	64	77	66	1
	14	17	19	23	24	30	64	77	66	2
	14	17	19	23	24	30	64	66	77	

[2 marks]

total: 20 comparisons