CSCI-B 505 Applied Algorithms (3 cr.) № 4

Dr. H. Kurban
Computer Science
School of Informatics, Computing, and Engineering

Indiana University, Bloomington, IN, USA

October 7, 2021

Contents

Problem 1: Algorithm Design 1 2

Problem 2: Algorithm Design 2 2

Problem 3: Algorithm Design 3 2

Problem 4: Merge-sort 2

Problem 5: Heap Sort 3
- Heap-sort algorithm . . . 3
- max_heapify algorithm . . . 3
- build_max_heap algorithm . . . 4

Problem 6: Counting Sort 4
- Counting-sort algorithm . . . 4

Problem 7: Bucket Sort 5
- Bucket-sort algorithm . . . 5

Problem 1: Algorithm Design 1

Suppose you are given two sequences $\mathcal{D}_1$ and $\mathcal{D}_2$ of n elements, possibly containing duplicates, on which a total order relation is defined. Describe an efficient algorithm for determining if $\mathcal{D}_1$ and $\mathcal{D}_2$ contain the same set of elements. What is the running time of this method?

Problem 2: Algorithm Design 2

Given an array $\mathcal{D}$ of n integers in the range $[0, n^2 - 1]$, describe a simple method for sorting $\mathcal{D}$ in $O(n)$ time.

Problem 3: Algorithm Design 3

Given a sequence $\mathcal{D}$ of n elements, on which a total order relation is defined, describe an efficient method for determining whether there are two equal elements in $\mathcal{D}$. What is the running time of your method?

Problem 4: Merge-sort

Implement a bottom-up merge-sort for a collection of items by placing each item in its own queue, and then repeatedly merging pairs of queues until all items are sorted within a single queue.

Problem 5: Heap Sort

Implement the heap-sort algorithm given in algorithm 1. The max_heapify and build_max_heap procedures are described in algorithm 2 and algorithm 3, respectively.

Algorithm 1 Heap-sort algorithm

Input: $\mathcal{D}$, an unsorted sequence.

Output sorted $\mathcal{D}$.

build_max_heap($\mathcal{D}$)

input_length = len($\mathcal{D}$)

for *i = input_length* **downto** *2* **do**

  swap $\mathcal{D}[1]$ and $\mathcal{D}[i]$

  $\mathcal{D}$.heap_size = $\mathcal{D}$.heap_size -1

  max_heapify($\mathcal{D}, 1$)

end

Algorithm 2 max_heapify algorithm

Input: $\mathcal{D}$, a sequence, and an integer i

Output partially max_heapify applied $\mathcal{D}$

l =left_child(i)

r =right_child(i)

input_length = len($\mathcal{D}$)

$\mathcal{D}$.heap_size $=$ input_length

if $l \leq \mathcal{D}$*.heap_size and* $\mathcal{D}[l] > \mathcal{D}[i]$ **then**

  largest = l

else

  largest = i

end

if $r \leq \mathcal{D}$*.heap_size and* $\mathcal{D}[r] > \mathcal{D}$[*largest*] **then**

  largest = r

end

if *largest* $\neq i$ **then**

  swap $\mathcal{D}[i]$ and $\mathcal{D}$[largest]

  max_heapify($\mathcal{D}$, largest)

end

Algorithm 3 build_max_heap algorithm

Input: $\mathcal{D}$, a sequence.

Output Max-heap $\mathcal{D}$

input_length = len($\mathcal{D}$)

$\mathcal{D}$.heap_size = input_length

for *i* = $\lfloor$*input_length* $/2\rfloor$ **downto** *1* **do**

| max_heapify($\mathcal{D}, i$)

end

Problem 6: Counting Sort

Implement the counting-sort algorithm given in algorithm 4.

Algorithm 4 Counting-sort algorithm

/* $len(\mathcal{D}) = len(B), max(\mathcal{D}) = k$ */

Input: $\mathcal{D}$: an unsorted sequence, B : empty sequence, k : an integer.

Output sorted $\mathcal{D}$.

Create a new array $C[0 \ldots k]$

input_length $= len(\mathcal{D})$

for $i = 0 \to k$ **do**

| $C[i] = 0$

end

for $j = 1 \to$ *input_length* **do**

| $C[\mathcal{D}[j]] = C[\mathcal{D}[j]] + 1$

end

for $i = 1 \to k$ **do**

| $C[i] = C[i] + C[i-1]$

end

for $j =$ *input_length* **downto** 1 **do**

| $B[C[\mathcal{D}[j]]] = \mathcal{D}[j]$

| $C[\mathcal{D}[j]] = C[\mathcal{D}[j]] - 1$

end

Problem 7: Bucket Sort

Implement the bucket sort algorithm given in algorithm 5.

Algorithm 5 Bucket-sort algorithm

Input: $\mathcal{D}$, an unsorted sequence.

Output sorted $\mathcal{D}$.

input_length = len($\mathcal{D}$)

Create a new array $B[0 \ldots (\text{input\_length} - 1)]$

for $i = 0 \to (\textit{input\_length} - 1)$ **do**
  make $B[i]$ an empty list
end

for $i = 1 \to n$ **do**
  insert $\mathcal{D}[i]$ into list $B[\lfloor \text{input\_length} \times \mathcal{D}[i] \rfloor]$
end

for $i = 0 \to (\textit{input\_length} - 1)$ **do**
  sort list $B[i]$ with insertion sort
end

concatenate the lists $B[0], B[1], \ldots, B[n-1]$ together in order

Directions

Please follow the syllabus guidelines in turning in your homework. While testing your programs (Questions 4-7), run them with a variety of inputs, e.g. ordered and unordered sequences, etc., and discuss your findings. The first four questions and question 7 are worth 15 points each. Any question you choose among them is optional. If you answer all, one question will be counted as an extra credit question. Questions 5 and 6 are worth 20 points each . This homework is due Sunday, Oct 17, 2021 10:00pm. **OBSERVE THE TIME**. Absolutely no homework will be accepted after that time. All the work should be your own.