

CS5234 : Combinatorial and Graph Algorithms  
Homework Set #3  
[Amortized Analysis, Binomial & Fibonacci Heaps]  
(Shorter homework in view of BAP-1 assignment)

**OUT:** 18-Sep-2007

**DUE:** 09-Oct-2007

**Course Web-Site:** <http://www.comp.nus.edu.sg/~cs5234/2007-08/>

**IMPORTANT NOTE: Read “Remarks about Homework”.**

**Solve all the S-Problems in every homework set.**

- Start each of the problems on a separate sheet of paper.
- Make sure your name and matric number is on each sheet.
- To hand the homework in, **staple them together** and hand to me during class or drop them into the CS5234 envelope outside room (COM1 03-41) by the due date.

When asked to “give an algorithm” to solve a problem, your write-up should NOT be just the code. Instead, it should be a short essay. The first paragraph should summarize the problem and what your results are. The body of the essay should consist of the following:

- A description of the algorithm in English and, if helpful, pseudo-code.
- At least one worked example or diagram to show more precisely how your algorithm works.
- A proof (or indication) of the correctness of the algorithm
- An analysis of the running time of the algorithm.

Remember, your goal is to communicate. Full credit will be given only to correct solutions which are described clearly. Convolved and obtuse descriptions will receive low marks.

---

*Routine Practice Problems -- do not turn these in -- but make sure you know how to do them.*

**R1. Exercises 19.1-1, 19.1-2** on p. 461 of [CLRS01] [**Simple Analysis**]

**R2. Exercises 19.2-2, 19.2-3, 19.2-5** of [CLRS01] [**Exercising Binomial-heap operations**]

**R3. Exercises 20.2-1** of [CLRS01] [**Exercising Fib-Heaps**]

**R4. Exercises 20.4-1** of [CLRS01] [**Long F-Heaps**] Prof. Pinocchio claims that the height of an  $n$ -node Fibonacci heap is  $O(\lg n)$ . Show that the professor is mistaken by exhibiting, for any positive integer  $n$ , a sequence of Fibonacci-heap operations that creates a Fibonacci heap consisting of just one tree that is a linear chain of  $n$  nodes.

---

**Standard-Problems -- solve these problems and turn them in by the due-date.**
**S1. [Amortized Analysis of Binomial Heaps]**

(a) Show that a Binomial-heap of  $n$  elements can be built by  $n$  successive insertions in  $O(n)$  time using any method (aggregate, accounting, potential) you like.

(b) Using the ideas developed above, or otherwise, show that the amortized times for INSERT, DELETE-MIN, and MELD for Binomial-heaps are  $O(1)$ ,  $O(\lg n)$ , and  $O(\lg n)$ , respectively.

(c) Suppose that a Binomial-heap of  $n = 2^k - 1$  elements is built. Alternately perform  $m$  INSERT and DELETE-MIN pairs. Clearly, each operation takes  $O(\lg n)$  time. Why does this *not* contradict the amortized bound of  $O(1)$  for insertion?

**S2. Problem 20-1 of [CLRS01] [Alternative implementation of deletion]**

Professor Pisano has proposed the following variant of the FIB-HEAP-DELETE procedure, claiming that it runs faster when the node being deleted is not the node pointed to by  $\text{min}[H]$ .

```

PISANO-DELETE( $H, x$ )
1.  if  $x = \text{min}[H]$ 
2.      then FIB-HEAP-EXTRACT-MIN( $H$ )
3.      else  $y \leftarrow p[x]$ 
4.          if  $y \neq \text{NIL}$ 
5.              then CUT( $H, x, y$ )
6.                  CASCADING-CUT( $H, y$ )
7.                  add  $x$ 's child list into the root list of  $H$ 
8.                  remove  $x$  from the root list of  $H$ 

```

(a) The professor's claim that this procedure runs faster is based partly on the assumption that line 7 can be performed in  $O(1)$  actual time. What is wrong with this assumption.

(b) Give a good upper bound on the actual time of PISANO-DELETE when  $x$  is not  $\text{min}[H]$ . Your bound should be in terms of  $\text{degree}[x]$  and the number  $c$  of calls to the CASCADING-CUT procedure.

(c) Suppose that we call PISANO-DELETE( $H, x$ ), and let  $H'$  be the Fibonacci heap that results. Assuming that node  $x$  is not a root, bound the potential of  $H'$  in terms of  $\text{degree}[x]$ ,  $c$ ,  $t(H)$  and  $\text{min}[H]$ .

(d) Conclude that the amortized time for PISANO-DELETE is asymptotically no better than FIB-HEAP-DELETE when  $x \neq \text{min}[H]$ .

**S3. Problem 19-2 of [CLRS] [MST algorithm using Binomial-Heaps]**

Chapter 23 [CLRS] presents two algorithms to solve the problem of finding a minimum spanning tree of an undirected graph. Here, we shall see how binomial heaps can be used to devise a different minimum-spanning-tree algorithm.

We are given a connected, undirected graph  $G = (V, E)$  with a weight function  $w : E \rightarrow \mathbf{R}$ . We call  $w(u, v)$  the weight of edge  $(u, v)$ . We wish to find a minimum spanning tree for  $G$ : an acyclic subset  $T \subseteq E$  that connects all the vertices in  $V$  and whose total weight (given by  $w(T) = \sum_{(u,v) \in T} w(u, v)$ ) is minimized.

The following pseudocode, which can be proven correct using techniques from Section 23.1, constructs a minimum spanning tree  $T$ . It maintains a partition  $\{V_i\}$  of the vertices of  $V$  and, with each set  $V_i$ , a set  $E_i \subseteq \{(u, v) : u \in V_i \text{ or } v \in V_i\}$  of edges incident on vertices in  $V_i$ .

```

MST( $G$ )
1.   $T \leftarrow \phi$ 
2.  for each vertex  $v_i \in V[G]$ 
3.      do  $V_i \leftarrow \{v_i\}$ 
4.           $E_i \leftarrow \{(v_i, v) \in E[G]\}$ 
5.  while there is more than one set  $V_i$ 
6.      do choose any set  $V_i$ 
7.          extract the minimum-weight edge  $(u, v)$  from  $E_i$ 
8.          assume without loss of generality that  $u \in V_i$  and  $v \in V_j$ 
9.          if  $i \neq j$ 
10.             then  $T \leftarrow T \cup \{(u, v)\}$ 
11.                  $V_i \leftarrow V_i \cup V_j$ , destroying  $V_j$ 
12.                  $E_i \leftarrow E_i \cup E_j$ 

```

Describe how to implement this algorithm using binomial heaps to manage the vertex and edge sets. Do you need to change the representation of a binomial heap? Do you need to add operations beyond the mergeable-heap operations given in Figure 19.1? Given the running time of your implementation.

---

**Advanced Problems** -- Try these for challenge and fun. There is no deadline for A-problems.

**A3.** Propose an algorithm to insert  $m$  nodes into a binary heap on  $n$  elements in  $O(m + \lg n)$  time. Prove your time bound. (Note: This is an *implicit* (array-based) binary heap.)

**A4.** Problem 17.3 of [CLRS] [**Amortized Weight-Balanced Trees**]