CSE548, AMS542: Analysis of Algorithms, Fall 2016 **Date:** Nov 30

Final In-Class Exam

(7:05 PM – 8:20 PM : 75 Minutes)

- This exam will account for either 15% or 30% of your overall grade depending on your relative performance in the midterm and the final. The higher of the two scores (midterm and final) will be worth 30% of your grade, and the lower one 15%.
- There are three (3) questions, worth 75 points in total. Please answer all of them in the spaces provided.
- There are 16 pages including four (4) blank pages and one (1) page of appendix. Please use the blank pages if you need additional space for your answers.
- The exam is *open slides* and *open notes*. But *no books* and *no computers* (no laptops, tablets, capsules, cell phones, etc.).

GOOD LUCK!

Question	Pages	Score	Maximum
1. The Lazy Deletion Filter	2–5		30
2. Randomized $\frac{3}{2}$-Approximate 3-way Max-Cut	7–11		35
3. Exam Scores	13		10
Total			75

NAME: ______________________________

$\text{INIT}^{(Q)}(\ )$ 1. $Q.queue \leftarrow \emptyset,\ \ Q.filter \leftarrow \emptyset$ $\{Q.queue$ *and* $Q.filter$ *are basic priority queues*$\}$

$\text{INSERT}^{(Q)}(\ x\ )$ $\{$*insert key* x *into* $Q\}$ 1. $\text{INSERT}^{(Q.queue)}(\ x\ )$	$\text{DELETE}^{(Q)}(\ x\ )$ $\{$*delete key* x *from* $Q\}$ 1. $\text{INSERT}^{(Q.filter)}(\ x\ )$

$\text{MINIMUM}^{(Q)}(\ )$ $\{$*return the smallest key in* $Q\}$ 1. $x \leftarrow \text{MINIMUM}^{(Q.queue)}(\ ),\ \ x' \leftarrow \text{MINIMUM}^{(Q.filter)}(\ )$ $\{x$ *is the smallest key in* Q*, and* x' *is the smallest key with a pending* DELETE *request*$\}$ 2. ***while*** $x \neq \text{NIL}$ ***and*** $x = x'$ ***do*** $\left\{x = x' \neq \textit{NIL}\right.$ *means that* $\text{DELETE}^{(Q)}(\ x\ )$ *was issued for* $\left.x\right\}$ 3. $\text{EXTRACT-MIN}^{(Q.queue)}(\ )$ $\{$*remove* x *from* $Q.queue\}$ 4. $\text{EXTRACT-MIN}^{(Q.filter)}(\ )$ $\{$*remove* $\text{DELETE}^{(Q)}(\ x\ )$ *from* $Q.filter\}$ 5. $x \leftarrow \text{MINIMUM}^{(Q.queue)}(\ ),\ \ x' \leftarrow \text{MINIMUM}^{(Q.filter)}(\ )$ $\{$*next smallest key and pending* DELETE$\}$ 6. ***return*** x $\{x$ *is the smallest key in* Q *for which* $\text{DELETE}^{(Q)}(\ )$ *was not issued*$\}$

$\text{EXTRACT-MIN}^{(Q)}(\ )$ $\{$*extract and return the smallest key in* $Q\}$ 1. $x \leftarrow \text{MINIMUM}^{(Q)}(\ )$ $\{x$ *is the smallest key in* Q *for which* $\text{DELETE}^{(Q)}(\ x\ )$ *was not issued*$\}$ 2. $\text{EXTRACT-MIN}^{(Q.queue)}(\ )$ $\{$*remove* x *from* $Q\}$ 3. ***return*** x

Figure 1: Using two instances ($Q.queue$ and $Q.filter$) of the given basic priority queue to create a new priority queue Q that supports INSERT, DELETE, MINIMUM and EXTRACT-MIN operations.

QUESTION 1. [30 Points] The Lazy Deletion Filter. I have a basic priority queue implementation that supports only INSERT, MINIMUM and EXTRACT-MIN operations in $\mathcal{O}(1)$, $\mathcal{O}(1)$ and $\mathcal{O}(\log n)$ worst-case time, respectively, where n is the number of items currently in it. If the queue is empty both MINIMUM and EXTRACT-MIN return NIL.

I have an application that requires a DELETE operation in addition to the three operations mentioned above, but unfortunately, I cannot change the given priority queue implementation to add the DELETE operation[1].

Figure 1 shows how I have used the given basic priority queue implementation as a blackbox to create a new priority queue Q that supports all four operations I need. The trick is to use one basic priority queue $Q.queue$ to perform INSERT and EXTRACT-MIN operations as usual, and another basic priority queue $Q.filter$ to store all pending DELETE operations. Whenever I access a key x from $Q.queue$, I check $Q.filter$ to see if a $\text{DELETE}^{(Q)}(\ x\ )$ operation was issued, and if so, I discard x. Thus $Q.filter$ acts as a filter to lazily remove deleted keys from $Q.queue$.

Priority queue Q assumes that for any given key value x:

(i) $\text{INSERT}^{(Q)}(\ x\ )$ will not be performed more than once during Q's lifetime,

(ii) $\text{DELETE}^{(Q)}(\ x\ )$ will not be issued more than once during Q's lifetime, and

(iii) $\text{DELETE}^{(Q)}(\ x\ )$ operation will not be issued unless x already exists in $Q.queue$.

[1] I only have a pre-compiled library, not the source code.

Suppose my application first initializes Q by calling $\text{INIT}^{(Q)}(\ )$ and then performs an intermixed seqeuence of INSERT, DELETE, MINIMUM and EXTRACT-MIN operations among which exactly N (≥ 1) are INSERT operations. Then answer the following questions.

1(a) [**8 Points**] What is the worst-case cost of each of the following operations: (i) $\text{INSERT}^{(Q)}(\ x\ )$, ($ii$) $\text{DELETE}^{(Q)}(\ x\ )$, ($iii$) $\text{MINIMUM}^{(Q)}(\ )$ and (iv) $\text{EXTRACT-MIN}^{(Q)}(\ )$? Justify your answers.

1(b) [**4 Points**] In order to find the amortized costs of the operations performed on Q we will use the following potential function:

$$\Phi\,(\;Q_i\;) = c\log N \times \text{number of items in } Q.queue \text{ after the } i\text{-th operation},$$

where, Q_i is the state of Q after the i-th ($i \geq 0$) operation is performed on it assuming that Q was initially empty, and c is a positive constant.

Argue that this potential function guarantees that the total amortized cost will always be an upper bound on the total actual cost.

1(*c*) [**18 Points**] Use the potential function given in part 1(*b*) to find the amortized cost of each of the following operations: (*i*) INSERT$^{(Q)}$(x), (*ii*) DELETE$^{(Q)}$(x), (*iii*) MINIMUM$^{(Q)}$( ) and (*iv*) EXTRACT-MIN$^{(Q)}$( ).

Use this page if you need additional space for your answers.

QUESTION 2. [35 Points] Randomized $\frac{3}{2}$-Approximate 3-way Max-Cut. Suppose you are given an undirected graph $G = (V, E)$ with vertex set V and edge set E, where $|V| = n$ and $|E| = m$. Now you divide V into three pairwise disjoint subsets V_1, V_2 and V_3 such that $V_1 \cup V_2 \cup V_3 = V$. For any edge $(u, v) \in E$, let $u \in V_i$ and $v \in V_j$ for some $i, j \in [1, 3]$. Then we say that (u, v) is a *cut edge* provided $i \neq j$. Let $E_c \subseteq E$ be the set of all cut edges of G, and let $m_c = |E_c|$. We will call E_c the *cut set*. Figure 2 shows an example.

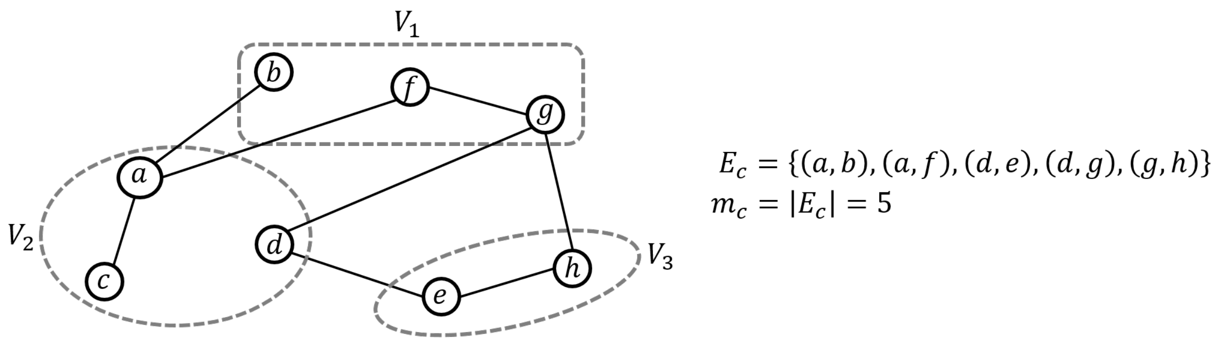

Figure 2: A 3-way cut example.

The *3-way Max-Cut* problem asks one to find subsets V_1, V_2 and V_3 to maximize m_c. A randomized approximation algorithm for solving the problem is given in Figure 3 below.

```
APPROX-3-WAY-MAX-CUT( V, E )
1. V1 ← ∅,  V2 ← ∅,  V3 ← ∅
2. for each vertex v ∈ V do
3.     choose a Vk from {V1, V2, V3} uniformly at random     {i.e., k takes each value from
                                                               {1, 2, 3} with probability 1/3}
4.     Vk ← Vk ∪ {v}
5. Ec ← ∅
6. for each edge (x, y) ∈ E do
7.     if x ∈ Vi and y ∈ Vj and i ≠ j then                   {1 ≤ i, j ≤ 3}
8.         Ec ← Ec ∪ {(x, y)}                                {(x, y) is a cut edge}
9. return ⟨V1, V2, V3, Ec⟩
```

Figure 3: Approximating 3-way Max-Cut.

2(a) [**7 Points**] Show that the expected approximation ratio of APPROX-3-WAY-MAX-CUT given in Figure 3 is $\frac{3}{2}$.

2(b) [**8 Points**] Show that for the cut set E_c returned by APPROX-3-WAY-MAX-CUT:

$$\Pr\left\{m_c \geq \frac{2m}{3}\right\} \geq \frac{3}{m+3}.$$

2(*c*) [**10 Points**] Explain how you will use APPROX-3-WAY-MAX-CUT as a subroutine to design an approximation algorithm with

$$\Pr\left\{m_c \geq \frac{2m}{3}\right\} \geq 1 - \frac{1}{e},$$

where, m_c is the size of the cut set returned by the algorithm.

You must describe your algorithm (briefly in words) and prove the probability bound.

2(d) [**10 Points**] Explain how you will use your algorithm from part (c) as a subroutine to design another approximation algorithm that returns a cut set of size at least $\frac{2m}{3}$ with high probability in m. You must describe your algorithm (briefly in words) and prove the probability bound.

Use this page if you need additional space for your answers.

Question 3. [**10 Points**] **Exam Scores.** After grading the last midterm exam I made a sorted list of n anonymous scores public. That was, indeed, a complete list of the scores obtained by all n students of the class. This time I plan to release a smaller list L. I will use the algorithm shown in Figure 4 for constructing L.

1. $L \leftarrow \emptyset$
2. ***for*** each student x in the class ***do***
3. include x's score in L with probablity $\frac{1}{n^{\frac{1}{3}}}$

Figure 4: Making the list L of scores to release.

3(a) [**10 Points**] Show that $Pr\left\{ \ |L| < n^{\frac{2}{3}} + n^{\frac{1}{2}} \ \right\} \geq 1 - \frac{1}{e^{\frac{n^{\frac{1}{3}}}{3}}}$.

Use this page if you need additional space for your answers.

Use this page if you need additional space for your answers.

Appendix I: Useful Tail Bounds

Markov's Inequality. Let X be a random variable that assumes only nonnegative values. Then for all $\delta > 0$, $Pr\left[X \geq \delta\right] \leq \frac{E[X]}{\delta}$.

Chebyshev's Inequality. Let X be a random variable with a finite mean $E[X]$ and a finite variance $Var[X]$. Then for any $\delta > 0$, $Pr\left[|X - E[X]| \geq \delta\right] \leq \frac{Var[X]}{\delta^2}$.

Chernoff Bounds. Let $X_1, \ldots, X_n$ be independent Poisson trials, that is, each X_i is a 0-1 random variable with $Pr[X_i = 1] = p_i$ for some p_i. Let $X = \sum_{i=1}^{n} X_i$ and $\mu = E[X]$. Following bounds hold:

Lower Tail:

- for $0 < \delta < 1$, $Pr\left[X \leq (1-\delta)\mu\right] \leq \left(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\right)^{\mu}$
- for $0 < \delta < 1$, $Pr\left[X \leq (1-\delta)\mu\right] \leq e^{-\frac{\mu\delta^2}{2}}$
- for $0 < \gamma < \mu$, $Pr\left[X \leq \mu - \gamma\right] \leq e^{-\frac{\gamma^2}{2\mu}}$

Upper Tail:

- for any $\delta > 0$, $Pr\left[X \geq (1+\delta)\mu\right] \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu}$
- for $0 < \delta < 1$, $Pr\left[X \geq (1+\delta)\mu\right] \leq e^{-\frac{\mu\delta^2}{3}}$
- for $0 < \gamma < \mu$, $Pr\left[X \geq \mu + \gamma\right] \leq e^{-\frac{\gamma^2}{3\mu}}$