COMP5711 Final Solution

Q1 (a) No. Consider a sequence of n MULTIPUSH operations each pushing n items into the stack. The total running time is $\Omega(n^2)$, so the amortized cost is $\Omega(n)$ not $O(1)$.

(b) Denote T as the expected running time of original Quicksort. We run the Quicksort for $2T$ time long and stop it . The algorithm will fail only when it need more than $2T$ time for the Las Vegas version which has expected running time T. According to Markov inequality, the fail probability is less than 1/2 when using a large constant c.

(c) Consider a random variable X which equals 1 with probability $1/k$ and 0 otherwise. Then we have $Pr(X \geq kE[X]) = Pr(X \geq 1) = \frac{1}{k}$.

(d) (1) 1

(2) 2

(3) n-1

(e) No. Although vertex cover can be solved in $f(k) \cdot poly(n)$ time, it is now known whether $f(n-k) \cdot poly(n)$ is polynomial in n or not.

Q2 (a) In the worst case, all the stacks containing the n elements are full, and in order to push one more element, we have to move each of these n elements from its current stack to the next one. Hence, the cost is $\Theta(n)$.

(b) Since each S_i can hold up to 2^i elements, the first k stacks can hold up to $2^0+2^1+\cdots+2^{k-1} = 2^k - 1$ elements. The maximum number of elements in the multi-stack is n, so we only use the first $k = \lceil \log_2(n+1) \rceil$ stacks. Then an element is pushed onto the multi-stack, we assign it $2\lceil \log_2(n+1) \rceil + 1$ credits. One credit is used to pay the cost of pushing to S_0, and the other $2\lceil \log_2(n+1) \rceil$ credits are stored with the element. When an element is move from its current stack to the next stack, we deduct 2 of its credit to pay the cost of pop (from the current stack) and push (to the next stack) operations. Since each element is moved at most $k = \lceil \log_2(n+1) \rceil$ times, its stored credits are enough to pay all the costs. In summary of the amortized cost of a push operation is $2\lceil \log_2(n+1) \rceil + 1 = O(\log n)$.

Q3 (a) Denote the number of steps needed of visiting an unvisited vertex conditioned on i vertices having been visited as T_i. By a similar analysis as in Coupon Collector problem. We have that the probability of visiting an unvisited vertex is $\frac{n-i}{n-1}$. So the waiting time $E[T_i] = \frac{n-1}{n-i}$. Then the expected total number of steps is $E[\sum_{i=1}^{n-1} T_i] = (n-1)H_{n-1}$.

(b) Due to the symmetry of complete graph, the probability of visiting v at the next step is always $\frac{1}{n-1}$ if the current vertex is not v. So the waiting time is $n-1$.

Q4 (a) This is pair-wise independent. For two distinct $x = x_1, \ldots, x_n$ and $x' = x'_1, \ldots, x'_n$ and $y, y' \in \{0,1\}$, we need to prove $Pr[a_0 + a_1x_1 + \ldots + a_nx_n = y \mod 2$ and $a_0 + a_1x'_1 + \ldots + a_nx'_n = y' \mod 2] = 1/4$.

$$\begin{cases} a_0 + a_1x_1 + \ldots + a_nx_n = y \mod 2 \\ a_0 + a_1x'_1 + \ldots + a_nx'_n = y' \mod 2 \end{cases}$$

There are 2^{n-1} solutions out of all possible 2^{n+1} choice of $a_0, \ldots, a_n$. Thus, $Pr[a_0 + a_1x_1 + \ldots + a_nx_n = y$ and $a_0 + a_1x'_1 + \ldots + a_nx'_n = y'] = \frac{2^{n-1}}{2^{n+1}}$. This is pair-wise independent.

(b) This is universal but not pair-wise independent. First, we show that it is universal. That is, for two distinct $x = x_1, \ldots, x_n$ and $x' = x'_1, \ldots, x'_n$, we need to prove that $Pr[a_0 + a_1x_1 + \ldots + a_nx_n = a_0 + a_1x'_1 + \ldots + a_nx'_n \mod 2] \leq 1/2$. In total we have 2^n hash functions, and

2^{n-1} of them causes a collision. Thus, $Pr[a_0 + a_1x_1 + \ldots + a_nx_n = a_0 + a_1x'_1 + \ldots + a_nx'_n \mod 2] = 1/2$. It is universal.

It is not pair-wise. Consider the input: $x_1 = x2 = \ldots x_n = 0$. It will always map to 0. Any other string will map to zero with probability 1/2. Hence the probability that they both take 0 is 1/2, not 1/4 as desired. So it is not pair-wise independent.

Q5 If $c = 1$, we only need to check whether $\mathcal{S}$ is a set cover of U, and whether $|\mathcal{S}| \leq k$. Suppose $c \geq 2$. Let a be an arbitrary element in U. We claim the following.

> The instance $(U, \mathcal{S})$ has a set cover of size at most k if and only if, for some $S_a \in \mathcal{S}$ containing a, the instance $(U \setminus S_a, \mathcal{S} \setminus \{S_a\})$ has a set cover of size at most $k - 1$.

We first prove the if direction. If for some $S_a \in \mathcal{S}$ containing a, the instance $(U \setminus S_a, \mathcal{S} \setminus \{S_a\})$ has a set cover $\mathcal{C}$ of size at most $k - 1$, then clearly $\mathcal{C} \cup \{S_a\}$ is a set cover of U, whose size is at most k. Next we prove the only if direction. Suppose that the instance $(U, \mathcal{S})$ has a set cover $\mathcal{C}$ of size at most k. In $\mathcal{C}$, there must be a subset $S_a \in \mathcal{S}$ that contains a. Clearly, $\mathcal{C} \setminus \{S_a\}$ is a set cover of $U \setminus S_a$, whose size is at most $k - 1$.

With the above claim, we can design our algorithm as follows. We pick an arbitrarily element a from U. For each $S_a \in S$ that contains a, we recursively determine whether the instance $(U \setminus S_a, S \setminus \{S_a\})$ has a set cover of size at most k. The original instance has a set cover of size at most k if and only if at least one reduced instances have a set cover of size at most $k - 1$.

We denote by $T(n, m, k)$ the running time for an instance with $|U| = n$, $|\mathcal{S}| = m$ and the parameter k. Recall our assumption that for each element $a \in U$, at most c subsets in $\mathcal{S}$ contain a. We have the following recurrence for $T(n, m)$.

$$T(n, m, k) \leq cT(n - 1, m - 1, k - 1) + c(m + n),$$
$$T(n, m, 0) = 1.$$

Solving the recurrence gives $T(n, m, k) = O(c^k(m + n))$.

Q6 Let's fix a bin. Let X be the number of balls in this bin. We have

$$E[X] = 2 \ln n + 2c.$$

By Chernoff bound, we have

$$Pr(X \leq 0) = Pr(X \leq (1 - 1)E[X]) \leq \exp\left(-\frac{E[X]}{2}\right) = \exp\left(-\ln n - c\right) = \frac{1}{e^c n}.$$

Since we have n bins, applying the union bound gives

$$Pr(\text{some bin is empty}) \leq n \cdot Pr(X \leq 0) = \frac{1}{e^c}.$$

Hence, the probability that we fill all the binds is at least $1 - \frac{1}{e^c}$.