CS630 Homework #1

Qilin Ye

January 25, 2025

Solution to problem 1. (1) $\mathbb{E}[X \mid X_1 \text{ is even}] = \mathbb{E}[X_1 \mid X_1 \text{ is even}] + \mathbb{E}[X_2 \mid X_1 \text{ is even}] = 4 + 3.5 = 7.5$.

(2) Conditioned on $\{X_1 = X_2\}$, both variables can simultaneously take any value in $\{1, \cdots, 6\}$ with equal probability, so the answer is the average of $\{2, 4, \cdots, 12\}$, or 7.

(3) Given $X_1 + X_2 = 9$, X_1 can only be $\{3, 4, 5, 6\}$ with equal probabilities. Hence $\mathbb{E}[X_1 \mid X = 9] = 4.5$.

(4) Since X_1 and X_2 are i.i.d. and $X_1 + X_2$ symmetrical, $\mathbb{E}[X_1 - X_2 \mid X]$ is simply the zero variable.

Solution to problem 2. There are two ways to interpret the problem. The first one is that we have unlimited supply, n general types, and nk exact types of coupons, where the experiment terminates once at least one coupon from each general type has been drawn. In this scenario, the problem is equivalent to the standard coupon collector's problem: for each of the n general types, the probability of being chosen chosen in each draw is $k/nk = 1/n$. Therefore, the expected number of draws remains unchanged: $\Theta(n \log n)$.

The more challenging part is where we assume the supply is finite: given kn coupons total, we draw without replacement and stop until we pick something from each set. To this end, let T be the number of draws throughout the experiment; clearly $\mathbb{E}T = \sum_{t=0}^{nk-1} \mathbb{P}(T > t)$. Define

$$\mathcal{E}_{i,t} := \{\text{at least } i \text{ sets remain uncollected after } t \text{ draws}\}.$$

Since $\{T > t\}$ means at least one set remains uncollected after t draws, by inclusion-exclusion

$$\begin{aligned}\mathbb{P}(T > t) &= \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} \mathbb{P}(\mathcal{E}_{i,t}) \\ &= \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} \binom{(n-i)k}{t} \binom{nk}{t}^{-1},\end{aligned}$$

and so

$$\mathbb{E}T = \sum_{t=0}^{nk-1} \mathbb{P}(T > t) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} \sum_{t=0}^{nk-1} \binom{(n-i)k}{t} \binom{nk}{t}^{-1}.$$

Now we consider the inner sum. Consider a hypergeometric random variable with nk total candidates and ik "good" candidates. The inner sum represents the expectation of first success (using again $\mathbb{E}X = \sum_t \mathbb{P}(X > t)$), and this quantity is well-known to be $(nk+1)/(ik+1)$. Therefore,

$$\boxed{\mathbb{E}T(n,k) = \mathbb{E}T = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} \frac{nk+1}{ik+1}.}$$

Now let $k \to \infty$ so that $(nk+1)/(ik+1) \to n/i$, and

$$\lim_{k\to\infty} \mathbb{E}T(n,k) = \sum_{j=1}^{n}(-1)^{j+1}\binom{n}{j}\frac{n}{j} = n\sum_{j=1}^{n}(-1)^{i+1}\binom{n}{j}\frac{1}{j} = nH_n,$$

where the last inequality is because the inclusion-exclusion expansion of the harmonic series matches the alternating sum structure. Indeed, we see as $k- > \infty$ we recover the result of the standard result.

Solution to problem 3. (1) We consider the famous Fisher-Yates algorithm:

Algorithm 1: Fisher-Yates

```
input: array a[1, ···, n]
for k = n, n − 1, ···, 1 do
    randomly generate j ∈ {1, ···, k}
    swap A[j] with A[k]
return A as shuffled array
```

It is easy to see that this output is uniformly distributed among all permutations on $\{1,\cdots,n\}$. Let $\sigma : \{1,\cdots,n\} \to \{1\cdots,n\}$ be given. It is clear that the outputted A has $A[n] = \sigma(n)$ with probability $1/n$: $A[n]$ is uniquely determined by the randomly selected element in the first iteration and unchanged since. Likewise, *conditioned on* $A[n] = \sigma(n)$, for $A[n-1]$ to match $\sigma(n-1)$, the second iteration must have chosen $\sigma(n-1)$, which happens with probability $1/(n-1)$. Iteratively, we see

$$\mathbb{P}(A[k] = \sigma(k) \mid A[j] = \sigma(j) \text{ for all } k+1 \leqslant j \leqslant n) = \frac{1}{k}.$$

Therefore,

$$\mathbb{P}(A[k] = \sigma(k) \text{ for all } k) = \mathbb{P}(A[n] = \sigma(n)) \times \mathbb{P}(A[n-1] = \sigma(n-1))$$
$$\times \cdots \times \mathbb{P}(A[1] = \sigma(1) \mid A[j] = \sigma(j) \text{ for all } 2 \leqslant j \leqslant n) = \frac{1}{n!},$$

proving the claim.

(2) By linearity of expectation and uniform randomness of each a_i,

$$\mathbb{E}\left[\sum_{i=1}^{n}|a_i - i|\right] = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}|j-i| = \frac{n^2-1}{3}.$$

This measures the expected "distance" between a random and the identity n-permutation.

(3) Given a permutation σ, a pair (i,j) with $i \neq j$ is out-of-order if $i < j$ and $\sigma(i) > \sigma(j)$. Observe that the number of iterations BubbleSort executes is precisely the number of out-of-order pairs: the identity permutation has zero out-of-order pairs, while a BubbleSort swap reduces the number of out-of-order pairs by precisely 1.

Under a random permutation, for every $i \neq j$, $\mathbb{P}((i,j)$ is out-of-order$) = \mathbb{P}((i,j)$ is not out-of-order$) = 1/2$. By linearity of expectation, summing over all $i \neq j$, $\mathbb{E}[\text{number of swaps}] = 1/2 \cdot n(n-1)/2 = n(n-1)/4$.

I'm not sure if (2) can be used to calculate (3). Specifically, the total displacement and the total number of out-of-order pairs are not equivalent concepts. One can easily come up with counterexamples in S_4, where the former is different from twice the latter.

Solution to problem 4. Given a group, with probability $(1-p)^k$ only one test is needed; otherwise, $k+1$ tests are needed. Thus for each group the expected number of tests is

$$(1-p)^k + (k+1)(1-(1-p)^k) = 1 + k(1-(1-p)^k).$$

Summing over all n/k groups we see

$$\mathbb{E}[\text{number of tests}] = \frac{n}{k} + n(1-(1-p)^k).$$

The exact minima can be found by solving

$$\frac{\partial}{\partial k}\mathbb{E}[\text{number of tests}] \sim -\frac{1}{k^2} - (1-p)^k \log(1-p) = 0,$$

which may or may not yield an elementary closed-form solution.
To prove the final claim, we consider the ratio $\mathbb{E}[\text{number of tests}]/n = 1 + 1/k - (1-p)^k$. Call this $f(p)$. Observe $f(1) > 1$ and $f(0) = 1/k < 1$. Clearly f is continuous, so for sufficiently small p, $f(p) < 1$, indicating that batch testing is indeed in expectation more efficient than the vanilla brute force testing.

***Solution to problem** 5*. Like in Karger's algorithm, in each iteration we randomly select an edge and contract its two vertices. We repeat this process until there are r components remaining. The only difference is that we terminate the algorithm once there are only r supernodes remaining; observe this is consistent with min-cut being a special case with $r = 2$.
We modify the proofs accordingly. The first part is to upper bound the size of the minimum r-way cut $|\mathcal{K}|$. We randomly pick $r-1$ vertices $v_1, \cdots, v_{r-1}$ and induce a r-way cut $\mathcal{C}$ consisting of $\{v_1\}, \cdots, \{v_{r-1}\}, V\backslash\{v_1, \cdots, v_{r-1}\}$. For an edge to not cross the cut, both endpoints must be in $V\backslash\{v_1, \cdots, v_{r-1}\}$. This happens with probability

$$\mathbb{P}(e \notin \mathcal{C}) = \left(1 - \frac{r-1}{n}\right)\left(1 - \frac{r-1}{n-1}\right).$$

Therefore,

$$|\mathcal{K}| \leqslant \mathbb{E}[|\mathcal{C}|] = m \cdot \left[1 - \left(1 - \frac{r-1}{n}\right)\left(1 - \frac{r-1}{n-1}\right)\right].$$

In the first contraction, the probability that $\mathcal{K}$ survives is at least

$$\mathbb{P}(\mathcal{K}\text{ survives first contraction}) = 1 - \frac{|\mathcal{K}|}{m} \geqslant \left(1 - \frac{r-1}{n}\right)\left(1 - \frac{r-1}{n-1}\right).$$

The rest is some algebraic manipulation.

$$\begin{aligned}\mathbb{P}(\mathcal{K}\text{ survives all contractions}) &\geqslant \prod_{j=r+1}^{n}\left(1 - \frac{r-1}{j}\right)\left(1 - \frac{r-1}{j-1}\right) = \prod_{j=r+1}^{n}\left(1 - \frac{r-1}{j}\right)\prod_{k=r}^{n-1}\left(1 - \frac{r-1}{k}\right) \\ &= \prod_{j=r+1}^{n}\frac{j-r+1}{j} \cdot \prod_{k=r}^{n-1}\frac{k-r+1}{k} \\ &= r \cdot \left[\frac{1}{r} \cdot \prod_{j=r+1}^{n}\frac{j-r+1}{j}\right] \cdot \prod_{k=r}^{n-1}\frac{k-r+1}{k} = r \cdot \binom{n}{r-1}^{-1}\binom{n-1}{r-1}^{-1}.\end{aligned}$$

As a sanity check, plugigng in $r = 2$ indeed gives the intended $\binom{n}{2}^{-1}$.

Solution to problem 6. PART 1. The intuition behind this LP is best seen in the following form:

$$\max \frac{\sum_{e \in E} y(e)}{\sum_{v \in V} x(v)} \qquad \text{subject to} \qquad \begin{cases} y(e) \leqslant \min(x(u), x(v)) & \text{for all } e = (u,v) \in E \\ x(v), y(e) \in \{0,1\} & \text{for all } v \in V, e \in E. \end{cases} \tag{*}$$

In this modified program (note it is not linear due to the denominator, so not even an integer LP), we explicitly assign binary labels to edges and vertices so a value 1 corresponds to the membership of the subgraph. The only constraint is that an edge belongs to the subgraph only if both endpoints are, and this is guaranteed by $y(e) \leqslant \min(x(u), x(v))$. Thus an optimal solution of (*) naturally induces the densest subgraph: include edges and vertices whose values are 1.
Relaxing $x(v), y(e) \in \{0,1\}$ into $x(v), y(e) \geqslant 0$, the new optimal value should bound that of (*) from above, while the problem is now a scale-free version of DENSEST SUBGRAPH. Since normalizing the denominator by setting $\sum_{v \in V} x(v) = 1$ will not alter the optimal objective value, solving the denominator-free gives an upper bound to the optimal density, as claimed.

PART 2.

(1) This follows from $y(e) \leqslant \min(x(u), x(v))$. Thus if $e = (u,v) \in E_r$ then $\min(x(u), x(v)) \geqslant y(e) \geqslant r$.

(2) This is essentially Lebesgue vs. Riemann integral. Since E is finite, $y : E \to [0,1]$ is measurable with respect to the discrete σ-algebra, so $y(e) = \int_0^1 \mathbf{1}[y(e) \geqslant r] \, \mathrm{d}r$. Then,

$$\sum_{e \in E} y(e) = \sum_{e \in E} \int_0^1 \mathbf{1}[y(e) \geqslant r] \, \mathrm{d}r = \int_0^1 \sum_{e \in E} \mathbf{1}[y(e) \geqslant r] \, \mathrm{d}r = \int_0^1 |E_r| \, \mathrm{d}r$$

as E is finite and $|E_r| = \sum_{e \in E} \mathbf{1}[y(e) \geqslant r]$.

(3) Analogous to above.

(4) The first claim follows from (2) and (3). Assume for contradiction that $|E_r|/|V_r| < c = \sum_e y(e) / \sum_v x(v)$ for all $r \in [0,1]$. Then

$$\sum_{e \in E} y(e) = \int_0^1 |E_r| \, \mathrm{d}r < c \int_0^1 |V_r| \, \mathrm{d}r = c \sum_{v \in V} x(v) = \sum_{e \in E} y(e),$$

contradiction.

As LP solvers such as the ellipsoid method itself takes $\Omega(m+n)$ time, to find r we can simply brute force all possible thresholds (based on values of x and y) until we find one with sufficiently large density ratio $|E_r|/|V_r|$. The induced subgraph is simply (V_r, E_r). Plugging these values back into the LP, we see the objective equals $\sum_{e \in E} y(e)$, the global optimum. Therefore, (V_r, E_r) must be a densest subgraph that we seek.