Question 1

Estimating all-pairs distances exactly

Algorithm 1: Monte Carlo Algorithm for exact distance

Input: G, partial distance matrix M
Output: complete distance matrix M with error probability $< 1 - 1/n^2$
$D \leftarrow n$ x n matrix to store distance
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ **to** n **do**
    if $M[i][j] = \#$ **then**
      $D[i][j]$ = infinity
    else
      $D[i][j] = M[i][j]$
  end
end
for $k \leftarrow 1$ **to** K **do**
  $w \leftarrow$ Random vertex in $[1:n]$
  $dist = bfs$(G, w)
  for $i \leftarrow 1$ **to** n **do**
    for $j \leftarrow 1$ **to** n **do**
      $D[i][j] = min(D[i][j], dist[i] + dist[j])$
    end
  end
end
return M

Complexity analysis

In above algorithm $bfs(G, s)$ returns an array storing shortest distance of s to each vertex in G. Time complexity for bfs is $O(n+m)$ (m = number of vertices in graph). In worst case there can be atmost $\binom{n}{2}$ edges, i.e worst case complexity of bfs is $O(n^2)$. Time taken to update D in one iteration is also $O(n^2)$. Since outer loop runs for K times, total complexity is $O(n^2K)$. In below analysis it is show that $K = O(\log n)$, hence complexity of above algorithm is $\boldsymbol{O(n^2)\log n}$.

Error Analysis

Error will occur if there are ≥ 1 entries in D which are not equal shortest path distance.

$$P(error) = P(M[1][1] \text{ wrong } \cup M[1][2] \text{ wrong } \ldots \cup M[n][n] \text{ wrong })$$

Applying Boole's inequality we get:

$$P(error) \leq n^2 * P(M[i][j] \text{ wrong }) \text{ (distance between } i, j > n/100 \text{)}$$

Note that there is $\leq$ sign in above line since the points for which distance $\leq n/100$ matrix M stores correct value (since the value is taken directly from partial distance matrix).

$M[i][j]$ computed wrong

We know that distance between i, j is greater than $n/100$, i.e number of nodes between shortest path from i to j is $\geq n/100$. So now consider first iteration of first loop.
Note that w is choosen randomly, hence the probability that w belongs to shortest path from i to j is $\geq \dfrac{n/100}{n} = \dfrac{1}{100}$.
If w belong to shortest path then $D[i][j]$ will be updated correctly and will contain shortest path distance (since we are doing bfs from w, we get shortest path value from $w \to i, w \to j$, $M[i][j]$ is simple the sum

of these 2 values).
Probability that $D[i][j]$ is wrong $(P(\delta)) \leq \dfrac{99}{100}$
Probability that $D[i][j]$ is wrong even after K iteration $\leq P(\delta)^K = \left(\dfrac{99}{100}\right)^K$
Using this value in above equation we get

$$\begin{aligned} P(error) &\leq n^2 P(\delta) \\ &\leq n^2 \left(\frac{99}{100}\right)^K \end{aligned}$$

We want that error probability is $\leq 1/n^2$, or

$$\begin{aligned} P(error) &\leq \frac{1}{n^2} \\ n^2 \left(\frac{99}{100}\right)^K &\leq \frac{1}{n^2} \\ \left(\frac{99}{100}\right)^K &\leq \frac{1}{n^4} \\ K \log\left(\frac{100}{99}\right) &\geq 4 \log n \\ K &\geq \boldsymbol{\frac{4}{\log(100/99)} \log n} \end{aligned}$$

Taking $\boldsymbol{K = 400 \log n}$ we get that all entries of the distance matrix are correct with probability exceeding $1 - 1/n^2$.

Question 2

Rumour Spreading

We will partition the experiment into following three stages. Let X is the number of persons knowing the rumour at any time. The stages are -

1. $X < c\log n$
2. $c\log n < X < \frac{n}{2}$
3. $\frac{n}{2} < X < n$

Expected no. of days to spread the rumour is the sum of expected no. of days spent in each of these stages.
(1) Expected no. days spent in stage 1 -
Let p is the probability that no new person comes to know about the rumour at the end of some day. Let k be the no. of persons knowing the rumour at the start of the day.
Thus,

$$p = \left(\frac{k}{n}\right)^k$$

For $1 \le k < \frac{n}{2}$, $p \le \left(\frac{1}{2}\right)^k \le \frac{1}{2}$
So, expected no. of days for at least one person to know the rumour is $\frac{1}{1-p} \le 2$.
Thus the expected no. of days for $c\log n$ persons to know the rumour is less than or equal to $2c\log n$.
Expected no. of days spent in first stage,

$$N_1 = 2c\log n$$

(2) Expected no. days spent in stage 2 -
Let the label of the person called by i^{th} person be x_i. Clearly, $x_1, x_2, ..., x_k$ are independent.
Consider $f(x_1, x_2, ..., x_k)$: No. of persons not knowing the rumor at the end of day, if the no. of persons not knowing the rumor at the beginning of the day was r.
By linearity of expectation and since $n - \log n < r$,

$$E[f] = r\left(\frac{n-1}{n}\right)^{n-r} \le \frac{r}{c}, c > 1$$

Clearly, $|f(A) - f(A_0)| \le 1$ for all A, A_0 that differ only at the i^{th} coordinate.
Thus, f satisfies Lipshitz condition with parameters $c_1, c_2, ...c_k$, $c_i = 1$.
Take p such that $1 < p < c$, call a day good if $f < \frac{r}{p}$ at the end of the day. Expected no. of good days required for this stage is $O(\log n)$

$$P[\text{day is bad}] = P\left[f \ge \frac{r}{p}\right]$$

$$P[\text{day is bad}] = P\left[f - \frac{r}{c} \ge \frac{r}{p} - \frac{r}{c}\right]$$

$$P[\text{day is bad}] \le P\left[|f - E[f]| \ge \frac{ru}{c}\right], u > 0$$

Using the method of bounded difference,

$$P[\text{day is bad}] \le \exp\left(\frac{-u^2r^2}{2c^2\sum {c_i}^2}\right)$$

Since $r < \frac{n}{2}$

$$P[\text{day is bad}] < \exp\left(\frac{-u^2n^2}{8c^2n}\right)$$

$$P[\text{day is bad}] \le \exp(-kn)$$

Thus probability that a day is bad is inverse exponential in n and hence the expected no. of days spent in second stage = $O(\log n)$.

(3) Expected no. days spent in stage 3 -
Let k be the no. of persons knowing the rumour at the start of the day and let r be the no. of persons not knowing the rumour at the start of the day ($r = n - k$).
Let p_i be the probability that i^{th} person does not know the rumour at the end of the day.

$$p_i = \left(\frac{n-1}{n}\right)^k$$

Since $k > \frac{n}{2}$,

$$p_i \leq \left(1 - \frac{1}{n}\right)^{\frac{n}{2}} \approx \frac{1}{\sqrt{e}}$$

Let R_i be the random variable which takes value 1 if the i^{th} person does not know the rumour at the end of the day and 0 otherwise.
Let R be the no. of persons not knowing the rumour at the end of the day.
Now by linearity of expectation,

$$E[R] = \sum_{i=1}^{r} E[R_i]$$

$$E[R] = \sum_{i=1}^{r} p_i + 0(1 - p_i)$$

$$E[R] \leq \frac{r}{\sqrt{e}}$$

Define a day to be good if the no. of persons not knowing the rumour reduces by more than $\frac{1}{\sqrt{2}}$ from the start of the day.
Thus the expected no. of good days required for spreading the rumour to n people is $2 \log n$.
Let p be the probability that a day is bad,

$$p = P\left[R \geq \frac{r}{\sqrt{2}}\right]$$

Using Markov's inequality,

$$p \leq \frac{\sqrt{2}E[R]}{r}$$

$$p \leq \frac{\sqrt{2}}{\sqrt{e}} \leq \frac{7}{8}$$

and so probability of a day being good is at least $\frac{1}{8}$.
Thus the Expected no. of days spent in third stage,

$$N_3 = 16 \log n = q \log n$$

Question 3

Approximate Ham-Sandwich Cut

Algorithm 1: Las-Vegas Algorithm for $(1 + 1/2)$-approximate ham-sandwich cut of P

Input: set P of $2n$ points, n red and n blue
Output: $(1 + 1/2)$-approximate ham-sandwich cut of P

```
repeat
    K ← random sample containing k/2 red, and k/2 blue points.
    L = ExactHamCut(K, k)
        /* count red points that lie on either side, with reference to origin.  */
    c_red_l ← 0
    c_red_r ← 0
    for i ← 1 to n do
        if P[i], (0,0) lie on same side of L then
            c_red_l ← c_red_l + 1
        else
            c_red_r ← c_red_r + 1
    end
    if n/4 ≤ c_red_l ≤ 3n/4 && n/4 ≤ c_red_r ≤ 3n/4 then
        red ← True
    else
        red ← False
        /* count blue points that lie on either side, with reference to origin.  */
    c_blue_l ← 0
    c_blue_r ← 0
    for i ← n + 1 to 2n do
        if P[i], (0,0) lie on same side of L then
            c_blue_l ← c_blue_l + 1
        else
            c_blue_r ← c_blue_r + 1
    end
    if n/4 ≤ c_blue_l ≤ 3n/4 && n/4 ≤ c_blue_r ≤ 3n/4 then
        blue ← True
    else
        blue ← False
    if blue && red then
        return L
until True;
```

Algorithm 1: ExactHamCut

```
Input: set X of m points, m/2 blue, m/2 red
Output: Exact Ham-Sandwich cut for X, m
R ← set of red points (|R| = m/2)
B ← blue points
for i ← 1 to m − 1 do
    for j ← i + 1 to m do
        L ← line joining X[i], X[j]
                                    /* Check if L is an exact ham-sandwich cut. */
        /* count red points that lie on either side, with reference to origin. */
        c_red_l ← 0
        c_red_r ← 0
        for i ← 1 to m/2 do
            if R[i], (0,0) lie on same side of L then
                c_red_l ← c_red_l + 1
            else
                c_red_r ← c_red_r + 1
        end
        if c_red_l ≤ m/4 && c_red_r ≤ m/4 then
            red ← True
        else
            red ← False
        /* count blue points that lie on either side, with reference to origin. */
        c_blue_l ← 0
        c_blue_r ← 0
        for i ← 1 to m/2 do
            if B[i], (0,0) lie on same side of L then
                c_blue_l ← c_blue_l + 1
            else
                c_blue_r ← c_blue_r + 1
        end
        if c_blue_l ≤ m/4 && c_blue_r ≤ m/4 then
            blue ← True
        else
            blue ← False
        if blue && red then
            return L
    end
end
```

Overview

We take random sample of size k, containing equal number of red and blue points, compute their exact ham sandwich cut, and check if this cut is an approximate ham sadwich cut for global set P as well. Is yes the we report it, else repeat. Since there are only $\binom{k}{2}$ possible cuts, for deterministic algorithm we just check all there possible pair in $O(k)$.

Complexity analysis

Time complexity of exact ham-sandwich cut above is $O(k^3)$ (since we need to check $\binom{k}{2}$ lines). From below analysis we know that for $k = 40 \log n$, the probability that algorithm will not terminate in 1 iteration is $\leq \dfrac{1}{n^4}$. Complexity of Las Vegas algorithm for approximate ham cut above is in fact a geometric random variable with success probability $\geq 1 - \dfrac{1}{n^4}$. Expected number of iteration ($E(t)$) is given by:

$$\begin{aligned}
E(t) &= \frac{1}{\text{Success probability}} \\
&\leq \frac{1}{1 - \dfrac{1}{n^4}} \\
&\leq \frac{n^4}{n^4 - 1} \\
\text{or } E(t) &= O(1)
\end{aligned}$$

In one iteration we call ExactHamCut ($O(k^3)$) for a set of points of size k, and then we check for all points ($O(n)$). So overall complexity of one iteration is $O(k^3 + n) = O(\log^3 n + n)$. Since expected number of iterations is $O(1)$, expected total complexity of algorithm is given by: $O(\log^3 n + n) = \boldsymbol{O(n)}$.

Error Analysis:

Error will occur when more than $\frac{3n}{4}$ red or $\frac{3n}{4}$ blue points lie on either side of the line which is the exact ham-sandwich cut of k points.
Without loss of generality assume that more than $\frac{3n}{4}$ red points lie on one side of the line.
This means that when selecting $\frac{k}{2}$ red points uniformly randomly, $\frac{k}{4}$ red points got sampled from a cluster of less than $\frac{n}{4}$ points and the other $\frac{k}{4}$ red points were sampled from the remaining points on the other side of the line.
Thus the error probability will not be more than the probability of getting more than $\frac{k}{4}$ tails in $\frac{k}{2}$ tosses of a biased coin having probability of head as $\frac{3}{4}$.
Thus,

$$P[\text{error}] \leq 2 * P[\text{more than } \frac{k}{4} \text{ tails occur in } \frac{k}{2} \text{ tosses}]$$

Taking into account the blue points, we have multiplied by 2.

$$P[\text{error}] \leq 2 * \sum_{i=\frac{k}{4}}^{\frac{k}{2}} \binom{\frac{k}{2}}{i} \frac{1}{4}^{i} \frac{3}{4}^{\frac{k}{2}-i}$$

$$P[\text{error}] \leq 2 * \binom{\frac{k}{2}}{\frac{k}{4}} \frac{3}{4}^{\frac{k}{2}} \sum_{i=\frac{k}{4}}^{\frac{k}{2}} \frac{1}{3}^{i}$$

$$P[\text{error}] \leq 2 * \binom{\frac{k}{2}}{\frac{k}{4}} \frac{3}{4}^{\frac{k}{2}} \frac{1}{3}^{\frac{k}{4}} \frac{3}{2}$$

$$P[\text{error}] \leq 3 * 4^{\frac{k}{4}} \frac{3}{4}^{\frac{k}{2}} \frac{1}{3}^{\frac{k}{4}}$$

$$P[\text{error}] \leq 3 * \frac{3}{4}^{\frac{k}{4}}$$

$$P[\text{error}] \leq \frac{1}{2}^{\frac{k}{10}}, k > 10$$

To get an error probability less than n^{-4} take $k = 40 \log n$.