1 Element Distinctness

The search for an item in an unsorted list can be interpreted as the search for a collision. Given a set of \( n \) numbers \( \{x_1, \ldots, x_n\} \), a collision is a pair \((i, j)\) so that \( x_i = x_j, i \neq j \).

A deterministic algorithm would solve this problem as follows: First it would sort the list (time: \( O(n \cdot \log(n)) \)) and then scan it to spot a collision. This algorithm can be shown to be optimal in the number of comparisons.

1.1 Cast as a graph search algorithm

This problem can be reformulated as a search problem on a graph \( G \). The vertices of \( G \) are all possible \( r \)-subsets of \( [n] \). Two vertices are connected whenever the corresponding subsets differ in a single element (i.e. the intersection has size \( r - 1 \)). This graph is a Johnson Graph with parameters \( n, r, \) and \( r - 1 \). The interesting vertices (called the “marked” vertices) are those that contain two indices \( i \) and \( j \) so that \( i \neq j \) and \( x_i = x_j \) (i.e. a collision).

Example: Let \( n = 10, r = 3 \). Then the number of vertices is \( \binom{10}{3} \) and the number of edges is \( \frac{1}{2} \cdot \binom{10}{3} \cdot \frac{10}{r(n-r)} = \frac{1}{2} \cdot \binom{10}{3} \cdot \frac{10}{3 \cdot 7} \).

2 A random walk algorithm

Given the formulation as a graph search problem, the following classical (probabilistic) algorithm finds a collision using random walks on the graph:

1. Start in a uniformly random vertex
   (pick \( r \) elements from \([n]\) and the corresponding numbers \( x_i \) from the list, sort the numbers)

2. Repeat for \( T_1 \) steps:
   (a) Random Walk on the graph \( G \) for \( T_2 \) steps:
      • Pick \( i \in R, j \in \bar{R} \) (where \( R \) is the subset corresponding to the current vertex)
      • insert \( j \) into \( R \) and \( x_j \) into the sorted list
      • delete \( i \) from \( R \) and \( x_i \) from the sorted list
   (b) If the state reached in (a) contains a collision, stop and output the pair.

3. If no collision is found, output "no collision".

The algorithm starts at a node \( x \), takes \( T_2 \) random steps leading to a new node \( x' \) and checks whether this new node contains a collision. If not, it repeats this procedure a maximum of \( T_1 \) times. Hence, it will check at most \( T_1 \) different nodes (i.e. subsets) for collisions.

Analysis of the complexity:

• Cost of 2(a): \( \log(r) \) per step in the random walk (so \( T_2 \cdot \log(r) \) in a total)
• $T_1 \approx$ expected number of samples of uniformly random $r$-subsets needed to locate a collision, assuming that the nodes reached are uniformly distributed. We have $p = \Pr[$subset has a collision$] \approx \left(\frac{r}{n}\right)^2$. So, $T_1 = O(n^2 r^2)$.

• $T_2 \approx r$ steps (roughly the time required to randomize any fixed $r$-subset by performing a random walk starting at that vertex)

### 2.1 Formal Analysis using Probability Transition Matrices

Let $P = (P_{XY})$ be the probability transition matrix of a random walk on a graph $G$. Assume $G$ is regular, undirected, non-bipartite and connected. For the Johnson Graph we have:

$$P_{XY} = \begin{cases} 
\frac{1}{\text{degree}} & \text{if } |X \cap Y| = r - 1 \\
0 & \text{otherwise}
\end{cases}$$

Properties of $P$:

1. $P$ has a left 1-eigenvector, the uniform distribution over the vertices and this is the unique 1-eigenvector

2. Every other eigenvector has eigenvalue $< 1$ in magnitude

The Spectral Gap of a matrix $P$ is defined as $\delta(P) = 1 - |\lambda_2(P)|$ where $\lambda_2(P)$ is the second largest eigenvalue of $P$ (in magnitude). The following theorem corresponds to Proposition 1 in the original paper:

**Theorem 16.1:** Let $P$ be a symmetric, ergodic random walk on state space $X$, with spectral gap $\delta(P) = \delta$. Let $M$ be a subset of $X$ (the marked elements) so that $|M| \geq \epsilon |X| = \epsilon N$. Then an algorithm analogous to the one above finds a marked element in time $O\left(\frac{1}{\delta^2}\right)$ with probability $\geq \frac{2}{3}$ (if one exists).

The proof shows that roughly it holds that $T_1 \approx \frac{1}{\epsilon}$ and $T_2 \approx \frac{1}{\delta}$ (the complete proof is given in the paper).

### 2.2 Cost of the random walk algorithm for Element Distinctness

For Element Distinctness, given a Johnson Graph with parameters $n, r, r - 1$, where $r = O(n)$, we have $\epsilon \approx \frac{r^2}{n^2}$ and $\delta = \frac{1}{r}$. This gives us

$$\text{Total Cost} = r \cdot \log(r) + T_1 \cdot (T_2 \cdot \log(r)) = r \cdot \log(r) + \frac{n^2}{r^2} \cdot (r \cdot \log(r))$$

Optimized over $r$ this gives $O(n \cdot \log(n))$ (for $r = n$), i.e. the same deterministic algorithm presented in the beginning. However, the random walk algorithm can be speeded up using quantum algorithms.

### 3 Quantum Walk

Let $P$ be a symmetric, ergodic random walk on state space $X$ (nodes of a graph). With $W(P)$ we denote the corresponding quantum walk where $W$ is a unitary operator (following Szegedy ’04). $W(P)$ is defined as follows:

**State Space:** Instead of nodes, the state space is spanned by pairs $|x,y\rangle$ where $x$ & $y$ are neighbours in the underlying graph (i.e. $(x,y)$ is an edge).

**Transition:** For the basis state $|x,y\rangle$, a step of the quantum walk is given by $W(P) = R_2 \cdot R_1$ where

(a) $R_1$: mix the right hand point $y$ using the Grover diffusion operator on the $d$ neighbours of $x$

(b) $R_2$: if $y'$ is the new right end point, similarly ”mix” the left endpoints over neighbours of $y'$
As in the case of the classical random walk, we can extract properties of $W(P)$. For that purpose we define $|P_\pi\rangle = \sum_y \frac{1}{\sqrt{d}} |y\rangle$ (where $y$ are neighbours of $x$) and let $I_x$ be the identity over the subspace $|x\rangle\langle x| \otimes \mathbb{C}^X$.

1. $R_1$ and $R_2$ are reflection operators:
   - $R_1$ is the reflection through the states $|x\rangle\langle P_x|$: $R_1 = \sum_x (2|x\rangle\langle P_x| - I_x)$
   - $R_2$ is the reflection through the states $|P_y\rangle\langle y|$: $R_2 = \sum_y (2|P_y\rangle\langle y| - I_y)$

2. (Spectrum of $W(P)$)
   - (a) $W(P)$ has a unique 1-eigenvector: $|\pi\rangle = \sum_{x\in X} \frac{1}{\sqrt{N}} |x\rangle\langle P_x|$ (where $x\in X$)
   - (b) For every eigenvalue $\lambda$ of $P$, $|\lambda| \in [0,1)$, $W(P)$ has eigenvalues $e^{\pm 2i\theta}$, where $\theta = \sqrt{\frac{1}{\lambda}}$.

Observation:

Hence, the phase gap of $W(P)$ between the 1-eigenvector and the eigenvector corresponding to the second largest eigenvalue is $\delta' = |0 - \theta| = |\theta|$. Using the above derived properties we get

$$\theta = \sqrt{\frac{1}{\lambda_2(P)}}$$

$$\cos(\theta) = \lambda_2 = 1 - \delta$$

With the approximation $\cos(\theta) \approx 1 - \frac{\theta^2}{2}$ we get a phase gap of $\theta \approx \sqrt{2\delta}$. This implies that we can distinguish between the 1-eigenvector and the remaining eigenvectors using phase estimation. The cost of the phase estimation is $\frac{1}{\text{phase gap}} \approx \frac{1}{\sqrt{\delta}}$ applications of $W(P)$.

4 The Quantum Algorithm for Element Distinctness

Given the quantum walk we can modify the classical random walk algorithm. Our desired final state is $|\mu\rangle = \sum_{x\in M} \frac{1}{\sqrt{N}} |x\rangle|P_x\rangle$ (normalized). As in Grover’s search algorithm, the following algorithm approximates $|\mu\rangle$ by two different reflections in the 2-dimensional subspace spanned by $|\pi\rangle$ and $|\mu\rangle$:

1. Start with: $|\pi\rangle = \sum_{x\in X} \frac{1}{\sqrt{N}} |x\rangle|P_x\rangle$

2. Repeat for T steps:
(a) Reflection through $|\pi\rangle$:
- for any basis vector $|x\rangle$ check if $x \in M$
- if yes, flip phase

(b) Reflection through $|\mu^\perp\rangle$:
- run phase estimation on current state (which is a linear combination of eigenvectors)
- if the estimate for the phase is $\neq 0$, flip the sign of that state
- undo phase estimation

The angle $\omega$ between $|\pi\rangle$ and $|\mu^\perp\rangle$ is given by $\sin(\omega) = \langle \pi | \mu \rangle = \sqrt{\epsilon} = \sqrt{\frac{M}{N}}$. The product of the two reflections above is a rotation by an angle of $2\omega$. Therefore, after $T = O(1/\omega) = O(1/\sqrt{\epsilon})$ iterations of this rotation starting with state $|\pi\rangle$, we will have approximated the target state $|\mu\rangle$.

The cost of the phase estimation in step 2(b) is $\frac{1}{\sqrt{\delta}}$. The cost of error reduction (through repetitions) is $\sim \log(T) \sim \log \left( \frac{1}{\sqrt{\epsilon}} \right)$. Therefore, the total cost is $\frac{1}{\sqrt{\delta}} \cdot \frac{1}{\sqrt{\epsilon}} \cdot \log \left( \frac{1}{\sqrt{\epsilon}} \right)$. The last term, $\log \left( \frac{1}{\sqrt{\epsilon}} \right)$, can be eliminated by using a recursive version of Grover search.

4.1 Applied to Element Distinctness

When we apply this result to the problem of Element Distinctness, we get

$$\text{Total Cost} = r \log(r) + \frac{1}{\sqrt{\epsilon}} \cdot \frac{1}{\sqrt{\delta}} \cdot \log(r)$$

where $\epsilon \approx \frac{\delta^2}{n^2}, \delta = \frac{1}{r}$. Optimizing over $r$ we get $r = n^{\frac{2}{3}}$ and a runtime complexity of $O(n^{\frac{2}{3}} \log(n))$. 