Sevag Gharibian

Lecture 11: Grover search and approximate counting

"The way to find a needle in a haystack is to sit down."
— Beryl Markham.

"After that, work and hope. But never hope more than you work."
— Also Beryl Markham.

Contents

1 Introduction

Shor's factoring algorithm and Grover's search algorithm are like soccer teammates on different ends of the field — the former, like a striker, is highly specialized to do one thing efficiently (i.e. score), while the latter, akin to a sweeper, has the broad job of neutralizing all threats which slip past the rest of the team. Indeed, Shor's algorithm gives an *exponential* speedup for the *specific* problem of factoring, whereas Grover search can be thrown at just about *anything* to yield a *quadratic* speedup.

Discovered by Lov Grover in 1996, Grover search is more specifically a quantum algorithm for solving the following general problem: Given query access to an oracle containing N items, one of which is "marked", find the marked item. For example, the "oracle" could be implemented by a 3-SAT formula ϕ, indexed by n-bit assignments x, and the aim is to find a satisfying assignment x (which would be considered "marked"). Grover's algorithm solves this problem with high probability using $O(\sqrt{N})$ queries to the database (which turns out to be optimal for a quantum algorithm), whereas a classical algorithm would require $\Omega(N)$ queries in the worst case. Thus, Grover search solves 3-SAT in $O(\sqrt{2^n})$ time.

We begin in Section 2 by defining the unstructured search problem. Section 3 then gives Grover's algorithm, and Section 4 discusses a recent approach for bootstrapping Grover's algorithm for approximate counting.

2 The unstructured search problem

We begin by formalizing the unstructured search problem.

Definition 1 (Unstructured search (SEARCH))**.**

- *Input: Query access to an oracle U_f for $f : \{0,1\}^n \mapsto \{0,1\}$ an unknown Boolean function, meaning the ability to compute (at unit cost) map*

$$|x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle \tag{1}$$

for any $x, y \in \{0,1\}^n$, and $\oplus$ the bit-wise XOR operation.

- *Output: An index $x \in \{0,1\}^n$ such that $f(x) = 1$, if one exists.*

Note that no assumptions about U_f are made, other than the requirement that we have superposition query access to the input/output behavior of f.

Exercise 2. SEARCH is often alternatively formulated as follows: The oracle U_f is replaced with an unknown string $z \in \{0,1\}^{2^n}$, such that each query to U_f is equivalent to accessing a single bit of z. The output is to compute the OR function on string z, i.e. $\mathrm{OR}(z) = \bigvee_{i=1}^{2^n} z_i$. Explain why this formulation of SEARCH is equivalent to Definition 1.

Application to 3-SAT and the Exponential-Time Hypothesis. As alluded to in the introduction, we may view a 3-SAT formula ϕ as a function $f : \{0,1\}^n \mapsto \{0,1\}$, i.e. which maps n-bit assignment x to $\phi(x)$. Thus, finding a satisfying assignment to ϕ is a special case of SEARCH with $f = \phi$. As we shall see shortly, SEARCH can be solved in $O(\sqrt{2^n})$ time quantumly, and this turns out to be optimal. Thus, quantumly one can apply Grover search as a black box to 3-SAT to find a satisfying assignment (if one exists) in $O(\sqrt{2^n})$ time. Does this saying anything about the complexity of 3-SAT itself?

Maybe, maybe not. It is generally believed that 3-SAT cannot be solved in subexponential time, at least on a classical computer. This is the premise of the *Exponential-Time Hypothesis* of Impagliazzo and Paturi from 1999, stated as follows.

Claim 3 (Exponential-Time Hypothesis (ETH))**.** *There exists a constant $\epsilon > 0$ such that 3-SAT requires time $\Omega(2^{\epsilon n})$ on a deterministic Turing machine.*

While the runtime of Grover's algorithm does not contradict ETH, if one *does* believe ETH, then it is perhaps not too surprising that $O(\sqrt{2^n})$ turns out to be the optimal worst-case runtime for a black-box quantum search algorithm.

Is ETH true? Again, this is not clear, but assuming the truth of the ETH has led to matching runtime *lower bounds* in an area known as *fine-grained complexity*. Roughly, the latter aims to pin down the precise complexity of problems which are *known to be in* P (e.g. is the optimal worst-case runtime, say, $O(n^3)$ or $O(n^2)$?). For example, in the Orthogonal Vectors Problem (OV), given sets of vectors $A, B \subseteq \{0,1\}^d$ with $|A| = |B| = n$, one is asked whether there exist $|v\rangle \in A$ and $|w\rangle \in B$ such that $\langle v|w\rangle = 0$?

Exercise 4. Show that OV can be solved in $O(n^2 d)$ time.

It turns out that, assuming a stronger variant of ETH known as the Strong Exponential Time Hypothesis, the naive polynomial runtime of Exercise 4 is the best possible. The interested reader is referred to the accessible survey of Bringmann [Bri19] for details.

3 Grover's algorithm

Define $N := 2^n$. We now show how to solve SEARCH in $O(\sqrt{N})$ time on a quantum computer. For clarity, recall that typically when one discusses problems with black-box access to an oracle (as in SEARCH), the relevant cost model is *query complexity* (i.e. each query has unit cost, and this is the *only* cost we care about). This is also the cost model we adopt here. We begin by revisiting our old friend, the phase kickback trick. Viewing this trick geometrically, in particular, will kickstart the development of the remainder of Grover's algorithm.

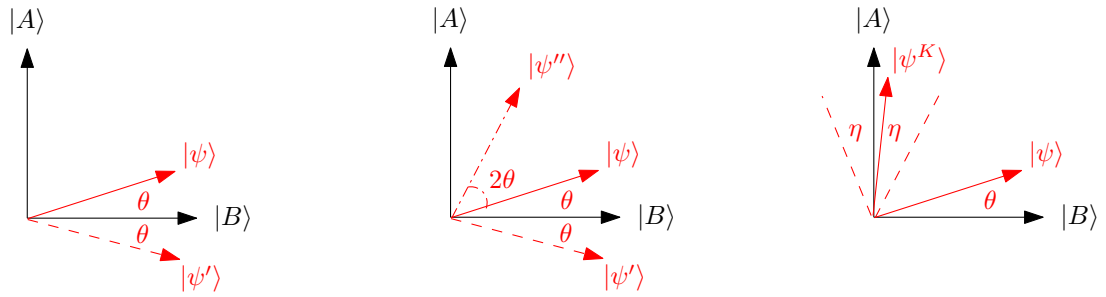

Figure 1: (Left) A reflection of $|\psi\rangle$ about $|B\rangle$ to $|\psi'\rangle$. (Middle) A reflection of $|\psi'\rangle$ about $|\psi\rangle$. For clarity, the angle 2θ is between $|\psi''\rangle$ and $|\psi'\rangle$. (Right) The final state $|\psi^K\rangle = (R_\psi R_B)^K|\psi\rangle$ after running the Grover iterate K times.

Phase kickback. The starting point is the phase kickback trick used in conjunction with oracle U_f. Recall this means using mapping $|x\rangle|-\rangle \mapsto (-1)^{f(x)}|x\rangle|-\rangle$, or for brevity,

$$|x\rangle \mapsto (-1)^{f(x)}|x\rangle. \tag{2}$$

It will be remarkably helpful to visualize phase kickback *geometrically* in the case of SEARCH. This is done, roughly, by considering superpositions over marked and unmarked items (for clarity, a "marked" ("unmarked") item x satisfies $f(x) = 1$ ($f(x) = 0$)). For this, let $A, B \subseteq \{0,1\}^n$ be the sets of marked and unmarked items, respectively, and define

$$|A\rangle := \frac{1}{\sqrt{|A|}} \sum_{x \in A} |x\rangle \quad \text{and} \quad |B\rangle := \frac{1}{\sqrt{|B|}} \sum_{x \in B} |x\rangle. \tag{3}$$

Thus, $|A\rangle$ ($|B\rangle$) is an equal superposition over all marked (unmarked) items. The geometric interpretation follows from the next exercise.

Exercise 5. Show that $U_f|A\rangle = -|A\rangle$ and $U_f|B\rangle = |B\rangle$.

In words, restricted to the 2D space defined by Span$(|A\rangle, |B\rangle)$, U_f acts as a reflection about $|B\rangle$, as depicted in Figure 5. And this is no coincidence — digging more deeply into this view will lead us directly to Grover's algorithm (though, for clarity, this geometric view was only discovered *after* Grover's original work). To see this, let us remind ourselves of the second tool we have at our disposal — preparing some initial start state $|\psi\rangle$. Ideally, this state $|\psi\rangle$ should also lie in the span of $|A\rangle$ and $|B\rangle$, so that it "fits" into the 2D picture of Figure 5.

Exercise 6. Take a moment to guess what might be a "reasonable" choice of $|\psi\rangle$, given our current state of knowledge. What do we know about the location of the marked items? What choice of $|\psi\rangle$ might lie in the span of $|A\rangle$ and $|B\rangle$?

The start state, $|\psi\rangle$. Since *a priori* we have no reason to believe any particular item x is marked, a naive choice of start state is

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} |x\rangle = \sqrt{\frac{|A|}{N}}|A\rangle + \sqrt{1 - \frac{|A|}{N}}|B\rangle. \tag{4}$$

Exercise 7. Prove the second equality above.

In other words, $|\psi\rangle \in \text{Span}(|A\rangle, |B\rangle)$, as desired, and we may depict it as in Figure 5 (Middle). The angle θ therein is given in the following exercise.

Exercise 8. Show that for start state $|\psi\rangle$ in Equation (4), $\cos\theta = \sqrt{1 - \frac{|A|}{N}}$.

The goal. To guide the rest of the algorithm, we now pause and step back to restate our goal in the 2D picture of Figure 5. Given the ability to prepare $|\psi\rangle$, we wish to rotate $|\psi\rangle$ counter-clockwise up to to $|A\rangle$, and subsequently measure in the standard basis, thus obtaining some marked item $x \in A$. Since this is just a rotation map, it is certainly possible, in that there exists a $2^n \times 2^n$ unitary matrix performing this rotation. The question is: *Can this mapping be computed using* $\text{poly}(n)$ *queries to* U_f *(and ideally,* $\text{poly}(n)$ *auxiliary gates)*?

Two reflections make a rotation. As the saying goes, beggars can't be choosers, and indeed to attain our goal, we must make do with what few tools the generality of SEARCH affords us: We can reflect about $|B\rangle$, and we can prepare $|\psi\rangle$. The key observation is that since we can efficiently prepare $|\psi\rangle$, i.e. $|\psi\rangle = H^{\otimes n}|0^n\rangle$, we can also *reflect* about $|\psi\rangle$.

Exercise 9. In Lecture 9, we saw that for $|\psi\rangle$, a reflection about $|\psi\rangle$ is achieved by unitary $U_\psi = 2|\psi\rangle\langle\psi| - I$. Show how to implement U_ψ for our choice of $|\psi\rangle$ from Equation (4). (Hint: Begin by showing how to reflect about $|0^n\rangle$, i.e. by implementing $U_{0^n} = 2|0^n\rangle\langle 0^n| - I$.))

Geometrically, this means we can map $|\psi'\rangle$ to $|\psi''\rangle$ in Figure 5. In other words, by first reflecting about $|B\rangle$, and then about $|\psi\rangle$, we can effect a counter-clockwise rotation of 2θ in Figure 5. Magic! Formally, we shall repeatedly apply the pair of reflections (often dubbed the "Grover iterate")

$$(2|\psi\rangle\langle\psi| - I)(2|B\rangle\langle B| - I) =: R_\psi R_B, \tag{5}$$

where R_B is effected by querying the oracle U_f, and R_ψ by undoing the preparation procedure for $|\psi\rangle$, reflecting about $|0^n\rangle$, and then re-doing the preparation for $|\psi\rangle$ (as per Exercise 9).

The number of iterations required. We can now state Grover's algorithm as follows:

1. Prepare $|\psi\rangle = H^{\otimes n}|0^n\rangle$.
2. Apply the Grover iterate, $R_\psi R_B$, K times.
3. Measure in the standard basis to obtain string $x \in \{0,1\}^n$.

If there are no solutions, i.e. $M = 0$, this procedure always outputs x satisfying $f(x) = 0$, as expected. If $M > 0$, on the other hand, the question is what to set K, the number of loop iterations, to? Note that it suffices for the algorithm to succeed with any fixed constant success probability p, as then independently repeating the algorithm drives down the overall error probability exponentially. To simplify the analysis, set $p = 1/2$. Without loss of generality, we may assume $|A|/N \leq 1/2$ (as otherwise classically choosing uniformly random inputs to U_f yields success probability at least $1/2$). By Exercise 9, we hence have $\theta \leq \pi/4$.

Exercise 10. Show that for Step 3 of Grover's algorithm to output $x \in \{0,1\}^n$ satisfying $f(x) = 1$ with probability at least $1/2$, it suffices in Figure 5 (Right) that $|\psi^K\rangle$ makes angle at most $\eta \leq \pi/4$ with $|A\rangle$. (Hint: Recall your high school formula that the overlap between real vectors $|v\rangle$ and $|w\rangle$ equals $\langle v|w\rangle = \| |v\rangle \|_2 \| |w\rangle \|_2 \cos\eta$ for η the angle between $|v\rangle$ and $|w\rangle$.)

Thus, by Exercise 8, we start with $|\psi\rangle$ at $\theta = \arccos(\sqrt{(1 - |A|)/N}) \leq \pi/4$, and we wish to end up at $|\psi^K\rangle$ with $\eta \in [-\pi/4, \pi/4]$.

Exercise 11. Given that $|\psi\rangle$ starts at angle $\theta \leq \pi/4$, do we ever need to wrap around the circle (when applying the Grover iterate) in order to land in range $\eta \in [-\pi/4, \pi/4]$?

Exercise 12. Using your answer from the previous exercise, show that setting

$$K = \left\lceil \frac{1}{2\theta} \left(\arccos \sqrt{\frac{|A|}{N}} - \frac{\pi}{4} \right) \right\rceil \tag{6}$$

suffices to succeed with probability at least $1/2$, assuming $|A| > 0$.

Exercise 13. Show that $\sin\theta = \sqrt{|A|/N}$ and that $\theta \geq \sin\theta$ for $|\theta| \leq 1$. Using these facts, show that

$$K \leq \left\lceil \frac{\pi}{8} \sqrt{\frac{N}{|A|}} \right\rceil \in O\left(\sqrt{\frac{N}{|A|}} \right) \tag{7}$$

queries to U_f suffice to find a marked item with probability at least $1/2$.

Exercise 14. How many auxiliary gates (i.e. gates other than queries to U_f) does Grover's algorithm use?

In closing, given query access to an oracle U_f for which $|A|$ out of $N = |A| + |B|$ items are marked, a marked item can be found quantumly using $O(\sqrt{|A|/N})$ queries. There is a slight catch, however — the exact number of queries required, as given by Equation (6), requires knowledge of $|A|$. Luckily, it turns out that not only do quantum algorithms allow us to check if $|A| > 0$, but additionally to estimate $|A|$ itself to within multiplicative error. This is known as *quantum approximate counting*, covered next.

4 Approximate counting

In Section 2, we defined the input to SEARCH as an oracle U_f, and the output was to find a marked item x, i.e. satisfying $f(x) = 1$. This can be generalized to the much more difficult question: Can we *count* the number of marked items?

4.1 A brief history of counting

Recall that in SEARCH we place no requirements on the complexity of *implementing* U_f, making SEARCH extremely general. However, here on Earth, one typically requires U_f to have an efficient implementation. For example, if $f : \{0,1\}^n \mapsto \{0,1\}$ is efficiently implementable by a deterministic Turing machine M_f, then assuming in SEARCH we are also given a description of M_f (as opposed to just query access), SEARCH is NP-complete. (For example, as done in Section 2, U_f could evaluate a 3-SAT formula.) If we now ask the more general question *"how many $x \in \{0,1\}^n$ satisfy $f(x) = 1$?"*, we obtain precisely the complexity class #P, which is believed much harder than NP. For example, while NP is (by definition) the first level of the Polynomial-Time Hierarchy (PH), Toda's theorem tells us $\mathrm{P}^{\#\mathrm{P}}$ contains *all* of PH. Thus, the ability to *count* solutions allows us to solve NP and a whole lot more.

Given the importance of #P to complexity theory, it is worth taking a moment to understand what quantum approximate counting shall buy us, in comparison to what is possible classically (again, assuming f is efficiently implementable by a Turing machine). Let M denote the number of $x \in \{0,1\}^n$ satisfying $f(x) = 1$. The classic result for approximate counting is Stockmeyer's algorithm, which shows how to approximate M within a multiplicative factor of 2 in randomized polynomial time, *assuming* one has access to an NP oracle. (This factor of 2 can then easily be boosted to $1+(1/p(|x|))$ for any desired fixed polynomial p.) In contrast, in this section we shall make a tradeoff — by adding quantum computation to the picture, we no longer need an NP oracle, but now the runtime is worst-case exponential: $O(\sqrt{M/N})$ to be precise

(assuming we want a constant probability of success). And this tradeoff in some sense is necessary — in the worst case, approximating the Permanent of a matrix (a classic #P-complete problem dating back to Valiant's original 1979 paper on #P) within a constant multiplicative error is *still* #P-*hard* [AA11]. And it is nowadays generally believed quantum computers cannot efficiently solve NP-hard problems, never mind #P-hard problems.

4.2 Quantum approximate counting via Grover search

Returning to the setting where U_f is a black-box about which we make no assumptions, there are nowadays multiple approaches for quantumly approximately counting $M := |\{x \in \{0,1\} \mid f(x) = 1\}|$. A classic approach is to run QPE on the Grover iterate (a theme which reappears in more general quantum walk frameworks), as it turns out the *eigenvalues* of the iterate encode M. However, here we shall review a conceptually simpler, more recent approach due to Aaronson and Rall [AR20], which does away with the QPE machinery and whittles the solution down to requiring just a single tool — Grover search itself.

4.2.1 Intuition

The basic idea. Let $p = M/N$, i.e. the fraction of satisfying assignments. Naively, there is a simple classical algorithm for estimating p — simply pick x uniformly at random, and evaluate $f(x)$. By definition, this succeeds with probability p, and so $1/p$ trials are expected before a satisfying assignment is found.

Exercise 15. Take a moment to Google for "geometric distribution". Show how to model the sampling experiment above as a geometric distribution. What is the expected value and variance for this distribution? What is the probability that the number of trials needed to see a success deviates significantly from $1/p$? (Hint: Use Chebyshev's inequality, which unlike Markov's inequality, takes the variance into account.)

Of course, in the worst case, $1/p \in O(N)$, and by now we are spoiled with getting faster $O(\sqrt{N})$ runtimes via Grover search. Thus, roughly, the quantum algorithm we discuss will carefully mimic (a refinement of) the idea above in conjunction with Grover search. In the remainder of this section, we state the algorithm, and sketch its proof of correctness. The interested reader is referred to [AR20] for full details.

Quantizing the basic idea. Recall from Exercise 13 that in Grover search, the angle made by start state $|\psi\rangle$ with $|B\rangle$ is $\theta = \arcsin(\sqrt{|A|/N})$, or in the terminology of this section, $\theta = \arcsin(\sqrt{M/N})$. One can generalize the analysis of Section 3 to show that by making $O(r)$ queries to U_f, Grover search finds a marked item with probability $p = \sin^2(r\theta)$. The smaller M is, the smaller θ is, and hence the larger r needs to be to make p large, as expected.

Exercise 16. More accurately, denoting the Grover iterate as G, we have

$$G^{(r-1)/2} = \frac{\sin(r\theta)}{\sqrt{M}} \sum_{x \in A} |x\rangle + \frac{\cos(r\theta)}{\sqrt{N-M}} \sum_{x \in B} |x\rangle. \tag{8}$$

Confirm that the probability of extracting a marked item after $O(r)$ uses of G is indeed p.

The beauty of [AR20] is now that we can forget about the word "quantum", and simply think of p as a probability arising from some abstract sampling experiment E with parameter r (we henceforth write $E(r)$ where appropriate). The question is: *Given the ability to choose r in this experiment $E(r)$, how many runs of E do we need to estimate p (thus allowing us to extract θ, which encodes the number of solutions M)*?

The high-level outline for achieving this with a quadratic speedup consists of two steps:

1. (Rough estimate) Repeat $E(r)$ using exponentially increasing values of r until "sufficiently many" marked items are found. This gives a rough estimate of $K_- \leq \theta \leq K_+$.

2. (Finetuning the estimate) Iteratively cut down the interval $[K_-, K_+]$ to zoom in on θ.

The second step, in particular, will require a careful choice of r each time $E(r)$ is run to avoid cutting the candidate interval $[K_-, K_+]$ too much, which in particular is a danger if $\theta \approx K_-$ or $\theta \approx K_+$. This shall rely on the Steady Hands Lemma 20, to be stated shortly.

4.2.2 Algorithm statement and analysis

The main theorem of [AR20] is the following.

Theorem 17. *Fix any desired $\epsilon, \delta > 0$. Given oracle access to U_f (as in* SEARCH*), there exists a quantum algorithm which:*

1. *outputs $\widetilde{M}$ satisfying $(1-\epsilon)M < \widetilde{M} < (1+\epsilon)M$,*
2. *succeeds with probability at least $1-\delta$,*
3. *uses $O\left(\sqrt{\frac{N}{M}}\frac{1}{\epsilon}\frac{1}{\log\delta}\right)$ queries to U_f,*
4. *uses $O(\log N)$ qubits/space.*

Statement of the algorithm. The algorithm is stated below. It assumes $\theta \leq \pi/1000$, which is without loss of generality since we can always "extend" U_f with dummy indices x which always lead to $f(x) = 0$. For now, we use abstract names c_i to denote parameters in the algorithm, so as to avoid excess detail obscuring the main pedagogical ideas.

1. Set $t = 0$.
2. (Rough estimate) Loop:
   (a) Let r be be the largest odd integer less than or equal to c_1^t for $c_1 > 1$.
   (b) Run $E(r)$ c_2 times, recording each of the c_2 items produced.
   (c) If at least $1/3$ of the recorded items x are marked, let $t^* = t$ and exit the loop.
   (d) Set $t = t + 1$.
3. Set $\theta_{\min} := \frac{5}{8}\left(\frac{1}{c_1}\right)^{t^*+1}$ and $\theta_{\max} := \frac{5}{8}\left(\frac{1}{c_1}\right)^{t^*-1}$, where recall $c_1 > 1$.
4. Set $t = 0$.
5. (Finetuning the estimate) Loop:
   (a) Use the Steady Hands Lemma to choose r.
   (b) Run $E(r)$ c_3 times, recording each of the c_3 items produced.
   (c) If at least half the items recorded were marked, increase the lower bound via
   $$\theta_{\min} = \frac{\theta_{\max}}{\Delta}, \tag{9}$$
   where $\Delta := 0.1 + 0.9(\theta_{\max}/\theta_{\min})$. Otherwise, decrease the upper bound via
   $$\theta_{\max} = \Delta\theta_{\min}. \tag{10}$$
   (d) If $\theta_{\max} \leq (1 + \frac{\epsilon}{5})\theta_{\min}$, exit the loop.
   (e) Set $t = t + 1$.
6. Return $\widetilde{M} := N\sin^2(\theta_{\max})$.

Formally, setting parameters $c_1 = 12/11$, $c_2 = 10^5 \ln(120/\delta)$, $c_3 = 10^3 \ln(100(0.9)^t/(\delta\epsilon))$ suffices for the proof of Theorem 17.

Analysis sketch. We now briefly sketch the proof of Theorem 17.

Checkpoint 1: After first loop terminates. The claim here, as suggested by line 3 of the algorithm, is that with probability at least $1 - (\delta/2)$,

$$\frac{5}{8}\left(\frac{1}{c_1}\right)^{t^*+1} = \frac{5}{8}\left(\frac{11}{12}\right)^{t^*+1} \leq \theta \leq \frac{5}{8}\left(\frac{11}{12}\right)^{t^*-1} = \frac{5}{8}\left(\frac{1}{c_1}\right)^{t^*-1}. \tag{11}$$

The claim is seen by focusing on a "threshold" t_0 for t, around which the probability of seeing at least $1/3$ of the recorded items marked in Step 2(c) jumps from "insignificant" to "extremely likely". Formally, set t_0 as the largest integer satisfying

$$\left(\frac{12}{11}\right)^{t_0}\theta \leq \frac{5}{8}. \tag{12}$$

Then, one can show that if $t < t_0$, the probability p of $E(r)$ returning a marked item is at most

$$p = \sin^2(r\theta) \leq 0.33\left(\frac{12}{11}\right)^{2(1+t-t_0)} < \frac{1}{3}. \tag{13}$$

Thus, over m trials, when $t < t_0$ we expect to see strictly less than $1/3$ of the sampled items being marked. Formally, one applies the Chernoff-Hoeffding bound to conclude that since the number of trials is $c_2 = 10^5 \ln(120/\delta)$, the probability of seeing at least $1/3$ of the samples being marked is "small", i.e. at most $\delta/4$.

Exercise 18. The Hoeffding bound states the following. Let X_1 through X_n be independent random variables, each satisfying $0 \leq X_i \leq 1$, whose arithmetic mean is $\overline{X} = (X_1 + \cdots + X_n)/n$. Then, the bound states

$$\Pr\left(\left|\overline{X} - E[\overline{X}]\right| \geq t\right) \leq 2e^{-2nt^2}. \tag{14}$$

In words, we converge exponentially quickly to the true expectation of $\overline{X}$ by drawing many samples and taking their arithmetic mean. Use the Chernoff bound to show that if a biased coin flip lands HEADS with probability $p-\epsilon$ for fixed $\epsilon > 0$, then it is highly unlikely over n coin flips to have at least pn HEADS instances.

Conversely, as soon as t is "large enough" (formally, $t = t_0 + 1$ suffices), one can show $p > 0.336 > 1/3$, so now a Chernoff bound suffices to conclude that, with probability at least $1 - (\delta/4)$, over $1/3$ of the samples will be marked.

Checkpoint 2: After second loop terminates. By line 5(d), we are guaranteed that if we reach line 6, then

$$\frac{\theta_{\max}}{\theta_{\min}} \leq 1 + \frac{\epsilon}{5}, \tag{15}$$

implying any $\widetilde{\theta} \in [\theta_{\min}, \theta_{\max}]$ satisfies

$$\left(1 - \frac{\epsilon}{5}\right)\theta \leq \widetilde{\theta} \leq \left(1 + \frac{\epsilon}{5}\right)\theta. \tag{16}$$

Exercise 19. Observing that line 6 of the algorithm chooses $\widetilde{\theta} = \theta_{\max}$, show that $\widetilde{M}$ satisfies the first claim of Theorem 17.

Roughly, to show that we indeed reach line 6 eventually, observe that after line 3, we have $\Delta = 0.1 + 0.9(\theta_{\max}/\theta_{\min}) \approx 0.1 + 0.9(1.19) > 1$. Thus, intuitively each run of lines 5c and 5d will eliminate a constant fraction of the search space, implying line 5(d) will eventually cause us to exit the second loop, as desired. The only catch is that, each time we remove such a fraction of the search space, we must ensure we do not

accidentally "skip" θ, i.e. we must always maintain $\theta \in [\theta_{\min}, \theta_{\max}]$. This is ensured by the following lemma, which shows how to pick r in line 5(a) to avoid the situation $\theta \notin [\theta_{\min}, \theta_{\max}]$. Note that the remaining probability of failure of $\delta/2$ in the algorithm arises by applying the union bound over all uses of the following lemma.

Lemma 20 (Steady Hands Lemma)**.** *Assume $0 < \theta_{\min} \le \theta \le \theta_{\max} \le \pi/1000$, and that $\theta_{\max}/\theta_{\min} \le 5/4$. Then, there exists an odd integer r such that, running $E(r)$ at least $1000 \ln(1/\delta)$ times and updating $\theta_{\min}$ and $\theta_{\max}$ according to the following rules preserves $\theta_{\min} \le \theta \le \theta_{\max}$ with probability at least $1 - \delta$:*

1. *If the majority of samples are marked, update $\theta_{\min} = \theta_{\max}/\Delta$.*
2. *If the majority of samples are unmarked, update $\theta_{\max} = \Delta\theta_{\min}$.*

Further, $r \in \Theta(1/\theta)$, where recall r controls the number of applications of the Grover iterate in $E(r)$.

We will not prove this lemma explicitly, but the rough idea is to select[1] r satisfying

$$r\theta_{\min} \approx \frac{\pi}{2}\left(\frac{\theta_{\min}}{\theta_{\max} - \theta_{\min}}\right) \qquad \text{and} \qquad r\theta_{\max} \approx r\theta_{\min} + \frac{\pi}{2}. \tag{17}$$

In words, this not only means r iterations of the Grover iterate "separate" the starting angles $\theta_{\min}$ and $\theta_{\max}$ by an additive angle of approximately $\pi/2$ radians, but that one can additionally show this choice of r aligns $r\theta_{\min}$ with "approximately" $|B\rangle$ (unmarked items) and $r\theta_{\max}$ with $|A\rangle$ (marked items) (recall $|A\rangle$ and $|B\rangle$ have an angle of $\pi/2$ radians). Since before we apply Lemma 20, we assume as a precondition that $\theta_{\min} \le \theta \le \theta_{\max}$, this means that if $\theta \approx \theta_{\min}$ ($\theta \approx \theta_{\max}$), after r iterations we have $r\theta \approx r\theta_{\min} \approx 0$ modulo 2π ($r\theta \approx r\theta_{\max} \approx \pi/2$ modulo 2π), so we are likely to sample an unmarked (marked) element. Thus, repeating $E(r)$ sufficiently many times and applying a Chernoff bound will ensure that if θ is close to (say) threshold $\theta_{\min}$, then Lemma 20 is smart enough to recognize this and to instead update threshold $\theta_{\max}$.

Formally, if $\theta_{\min} \le \theta \le \theta_{\max}/\Delta$ (Case 1 of Lemma 20), one can show $E(r)$ outputs a marked item with probability

$$p = \sin^2(r\theta) \le 0.47 < \frac{1}{2}. \tag{18}$$

Conversely, if $\Delta\theta_{\min} \le \theta \le \theta_{\max}$ (Case 2 of Lemma 20), one can show $E(r)$ outputs a marked item with probability

$$p = \sin^2(r\theta) \ge 0.6 > \frac{1}{2}. \tag{19}$$

Thus, by the Chernoff-Hoeffding bound, cases 1 and 2 of Lemma 20 will correctly update $\theta_{\max}$ or $\theta_{\min}$, respectively, with high probability.

Query complexity. Finally, let us sketch the number of queries to U_f each stage of the algorithm requires.

Exercise 21. Recall the first loop is run at most $t_0 + 1$ times, for t_0 defined in Equation (12). Recalling that each run of the loops uses $10^5 \ln(120/\delta)$ calls to U_f, show that the total number of queries required by the first loop scales as

$$O\left(\frac{1}{\theta}\log\frac{1}{\delta}\right) \in O\left(\sqrt{\frac{N}{M}}\log\frac{1}{\delta}\right). \tag{20}$$

A somewhat similar argument shows that the second loop requires

$$O\left(\frac{1}{\theta\epsilon}\log\frac{1}{\delta}\right) \in O\left(\sqrt{\frac{N}{M}}\frac{1}{\epsilon}\log\frac{1}{\delta}\right) \tag{21}$$

queries, which dominates the first loop's runtime, and hence leads to the claimed overall query cost of Theorem 17.

[1] Formally, one selects r as the closest integer to $2\pi k/\theta_{\min}$, where k is the closest integer to $\theta_{\min}/(4(\theta_{\max} - \theta_{\min}))$.

References

[AA11] Scott Aaronson and Alex Arkhipov. The computational complexity of linear optics. In *Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 333–342, 2011.

[AR20] Scott Aaronson and Patrick Rall. *Quantum Approximate Counting, Simplified*, pages 24–32. 2020.

[Bri19] Karl Bringmann. Fine-Grained Complexity Theory (Tutorial). In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science (STACS 2019)*, volume 126 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 4:1–4:7, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.