Quantum Algorithms 3/7/2023

Lecture 13: Quantum Walk Search

Instructor: Dieter van Melkebeek

In this lecture we establish the quadratic speed-up that quantum walks offer over random walks for hitting a good vertex. We first establish a fast forwarding procedure for an unconstrained walk in the block encoding framework and then apply it to an unconstrained walk that mimics the hitting constraint by interpolating between the original walk and its absorbing variant. Technical ingredients include Chebyshev expansions and a method known as Linear Combination of Unitaries (LCU).

1 Recap

We review the paradigm of a quantum walk and the notion of a block encoding. We start with the latter.

1.1 Block encoding

Definition 1. *A block encoding of a matrix M acting on m qubits is a unitary (circuit) A acting on $\ell + m$ qubits such that*

$$A = \begin{bmatrix} M & * \\ * & * \end{bmatrix}$$

The block encoding can be used as a probabilistic encoding of M with a success indicator as follows:

1. Apply A to state $|0^\ell\rangle |\psi\rangle$ representing 0 in all ancillas and $|\psi\rangle$ being the state to which we want to apply M.

2. Measure the first register (first ℓ qubits).

3. If the outcome is 0^ℓ, the second register is in the state $M|\psi\rangle / ||M|\psi\rangle||_2$. For this encoding to be useful, $||M|\psi\rangle||_2^2$, the probability of observing 0^ℓ, should be sufficiently large.

1.2 From random to quantum walks

Consider a graph $G = (V, E)$ with positive edge weights and no vertices of degree 0. In the classical setting, we start from a vertex chosen from an initial distribution σ. In each step, when at vertex u, we move to vertex v with probability proportional to the weight of the corresponding edge. This can be modeled as a Markov chain with transition matrix T, where:

$$T_{vu} = \Pr[\text{move to } v | \text{at } u]$$

There is a unique stationary distribution π provided that G is connected, and convergence to the distribution is guaranteed provided G is connected and non-bipartite.

We can use a random walk to find a good vertex, i.e., a vertex v such that $f(x) = 1$, where $f : V \mapsto \{0,1\}$ is given as a black-box. There can be several good vertices. The hitting time $H(T, f)$ is the expected number of steps needed to hit a good vertex when the initial vertex is picked according to the stationary distribution π. In case there is no good vertex, $H(T, f) = \infty$.

In a quantum walk, we act on directed edges, namely $|u, v\rangle$ where u represents the previous or next vertex and v represents the current one. Each step of the quantum walk can be decomposed into two parts: a coin flip C, where we decide the next vertex to go to, and a swap, where we actually "move" from one vertex to the next. While there are several choices for the coin flip, the Grover coin leads to the quadratic speed-up for search. Specifically, the two steps can be formalized as:

- Grover coin C: Reflect the first component of $|u, v\rangle$ about $|N_v\rangle \doteq \sum_{u'} \sqrt{T_{u'v}}\,|u'\rangle$. Assuming a unitary U such that $U : |0^n\rangle\,|v\rangle \mapsto |N_v\rangle\,|v\rangle$, C can be effectuated as $C = UR_{|0^n*\rangle}U^*$.
- Swap S: Swap vertex u and v to realize $|u, v\rangle \mapsto |v, u\rangle$.

In contrast to random walks, the asymptotic behavior of quantum walks can depend on the start state. For the quadratic speed-up result for search, we start from the following invariant state:

$$|\Pi\rangle \doteq \sum_v \sqrt{\pi_v}\,|N_v\rangle\,|v\rangle = \sum_u \sqrt{\pi_u}\,|u\rangle\,|N_u\rangle\,, \tag{1}$$

where $|N_v\rangle \doteq \sum_u \sqrt{T_{uv}}\,|u\rangle$.

We focus on three contributions to the cost of a quantum walk algorithm for search:

- Setup cost s: the cost to create the state $|\Pi\rangle$. This can be done by first creating the pure state $|\sqrt{\pi}\rangle \doteq \sum_v \sqrt{\pi_v}\,|v\rangle$ and then applying U to $|0^n\rangle\,|\sqrt{\pi}\rangle$.
- Update cost u: the cost of a (controlled) application of the walk operator SC, where C is the Grover coin. That is, the cost of $SUR_{|0^n*\rangle}U^*$ (or a controlled version).
- Check cost c: the cost of an application of U_f.

The above costs reflect a natural way of casting quantum walk algorithms in terms of three black-boxes (setup, update, and check) but ignore other quantum gates that the algorithm may use (and are typically independent of T and f).

With the above notation, we are ready to state the result that is the main topic of this lecture.

Theorem 1. *There exists a quantum algorithm that, for any random walk with transition matrix T on a graph $G = (V, E)$ with positive edge weights and no vertices of degree zero, and for any $f : V \to \{0,1\}$, outputs $v \in V$ with $f(v) = 1$ in expected cost $\tilde{O}(s + \sqrt{H(T, f)}(u + c))$.*

Here is the proof outline in the case of a symmetric T:

- We construct a block encoding of a matrix M that is close to T^t using only $O(\sqrt{t})$ steps of the quantum walk. This process is referred to as fast forwarding.
- We use the block encoding of M to approximately compute and measure $T^t\,|B\rangle$ for a random $t \in [O(\sqrt{H(T, f)})]$, where $|B\rangle$ denotes the normalized projection of $|\pi\rangle$ onto the bad components. We do this in case an initial check for a good vertex in the second register fails, in which case the system is in state $U\,|0^n\rangle\,|B\rangle$.
- We replace T by an interpolation between T and the absorbing version of T to guarantee good success probability.

2 Fast Forwarding

In general, the fast forwarding property refers to the geometric symmetrization $\widetilde{T}$ of T rather than T itself. Recall $\widetilde{T}_{uv} \doteq \sqrt{T_{uv} \cdot T_{vu}}$. Note that when T is symmetric, $\widetilde{T} = T$.

Lemma 2 (Fast Forwarding Lemma). *There exists a quantum algorithm that, for any random walk with transition matrix T, $t \in \mathbb{N}$, and $\epsilon > 0$, realizes a block encoding of a matrix M with $\|\widetilde{T}^t - M\|_2 \leq \epsilon$ and the following complexity:*

- $q = O(\sqrt{t \cdot \log(1/\epsilon)})$ *controlled applications of the quantum walk operator SC with coin $C = UR_{|0^n*\rangle}U^*$, where U is a unitary mapping $|0^n\rangle\,|v\rangle$ to $|N_v\rangle\,|v\rangle \doteq \sum_u \sqrt{T_{uv}}\,|u\rangle\,|v\rangle$.*
- *One more application of U and U^* each.*
- $\ell = O(\log q)$ *ancillas for the block encoding.*
- $O(q\ell \log(1/\epsilon))$ *elementary quantum gates, the parameters of which can be computed classically in time* $\mathrm{poly}(q, \log(1/\epsilon))$.

The lemma roughly says that we can approximately simulate t steps of the random walk with only about $O(\sqrt{t})$ steps of the quantum walk. The proof consists of three steps.

1. The transition matrix for d steps of the quantum walk, $(SC)^d$, block encodes $T_d(\widetilde{T})$ (up to a basis change by U), where T_d denotes the Chebyshev polynomial of degree d.
2. $\widetilde{T}^t$ can be closely approximated by a linear combination of $T_d(\widetilde{T})$ for d around $\Theta(\sqrt{t})$. This follows from well-known properties of the Chebyshev polynomials.
3. An efficient method to block encode a linear combination of unitaries, known as the LCU method.

2.1 Iterates of the quantum walk operator as block encodings

The block encoding perspective enters the analysis of the iterates of SC naturally. Recall that the quantum walk acts on two registers of vertices, with basis states of the form $|u\rangle\,|v\rangle$. In the end, we are interested in the weight distribution induced by the second register; the first register we introduced in order to make the simulation of a random walk on a quantum computer possible. For each basis state $|v\rangle$ of the second register, we start the first register in the state $|N_v\rangle$. As the coin operator C reflects about that very state, the state is invariant under C. It therefore makes sense to analyze what happens in the subspace spanned by the states $|N_v\rangle\,|v\rangle$. Since $|N_v\rangle\,|v\rangle = U\,|0^n\rangle\,|v\rangle$, this means analyzing the quantities

$$(U\,|0^n\rangle\,|u\rangle)(SC)^d(U\,|0^n\rangle\,|v\rangle = \langle 0^n|\,\langle u|\,(U^*(SC)^dU)\,|0^n\rangle\,|v\rangle\,,$$

i.e., the $N \times N$ matrix that $U^*(SC)^dU$ block encodes.

Single step. Let us start with $d = 1$. Since $C = UR_{|0^n*\rangle}U^*$, we have that $U^*(SC)U = U^*SUR_{|0^n*\rangle}$, which acts the same as U^*SU on $|0^n*\rangle$. Note that, as S is a reflection, so is U^*SU. We denote the latter reflection as

$$R \doteq U^*SU$$

and compute its top left corner.

$$\begin{aligned}
R_{0^nu,0^nv} &= \langle 0^n| \langle u| R |0^n\rangle |v\rangle \\
&= \langle 0^n| \langle u| U^*SU |0^n\rangle |v\rangle && \text{(definition of } R\text{)} \\
&= (U |0^n\rangle |u\rangle)^* S(U |0^n\rangle |v\rangle) && \text{(rearranging terms)} \\
&= (|N_u\rangle |u\rangle)^* S(|N_v\rangle |v\rangle) && \text{(defining property of } U\text{)} \\
&= (|N_u\rangle |u\rangle)^* (|v\rangle |N_v\rangle) && (S : |u, v\rangle \mapsto |v, u\rangle) \\
&= \sqrt{T_{vu}} \cdot \sqrt{T_{uv}} && \text{(definition of } N_u \text{ and } N_v\text{)} \\
&= \widetilde{T}_{uv} && \text{(definition of } \widetilde{T}\text{)}
\end{aligned}$$

We conclude:

Fact 3. *R block encodes $\widetilde{T}$, and so does $RR_{|0^n\rangle} = U^*(SC)U$.*

In other words, up to a basis transformation by the unitary U, one step of the quantum walk block encodes the geometric symmetrization of T.

More steps. Let M_d denote the $N \times N$ matrix that $(RR_{|0^n*\rangle})^d$ block encodes, i.e.,

$$(RR_{|0^n*\rangle})^d = \begin{bmatrix} M_d & * \\ * & * \end{bmatrix}.$$

In terms of such block matrices, we can write $P_{|0^n*\rangle} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$, $R_{|0^n*\rangle} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ and

$$P_{|0^n*\rangle}(RR_{|0^n*\rangle})^d P_{|0^n*\rangle} = \begin{bmatrix} M_d & 0 \\ 0 & 0 \end{bmatrix}.$$

We have

$$\begin{aligned}
&P_{|0^n*\rangle}(RR_{|0^n*\rangle})^{d+1} P_{|0^n*\rangle} \\
&\quad = P_{|0^n*\rangle} \left(R(2P_{|0^n*\rangle} - I)\right) (RR_{|0^n*\rangle})^d P_{|0^n*\rangle} \\
&\quad = 2(P_{|0^n*\rangle} R P_{|0^n*\rangle}) \left((RR_{|0^n*\rangle})^d P_{|0^n*\rangle}\right) \\
&\qquad\quad - \left(P_{|0^n*\rangle}(RR) R_{|0^n*\rangle}\right) \left((RR_{|0^n*\rangle})^{d-1} P_{|0^n*\rangle}\right) \\
&\quad = 2 \begin{bmatrix} M_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} M_d & 0 \\ * & 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} M_{d-1} & 0 \\ * & 0 \end{bmatrix} \\
&\quad = \begin{bmatrix} 2M_1M_d - M_{d-1} & 0 \\ 0 & 0 \end{bmatrix},
\end{aligned}$$

where the first line follows from writing $R_{|0^n*\rangle} = 2P_{|0^n*\rangle} - I$, the second from linearity and rearranging, and the third from Fact 3 and that, as a reflection, R is its own inverse. Thus, we have the recurrence

$$M_{d+1} = 2 \cdot M_1 \cdot M_d - M_{d-1}$$

for $d \geq 1$, where $M_0 = I$. A similar recurrence defines the Chebyshev polynomials $T_d(x) \in \mathbb{R}[x]$:

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{d+1}(x) &= 2 \cdot x \cdot T_d(x) - T_{d-1}(x) \text{ for } d \geq 1. \end{aligned} \tag{2}$$

We conclude that $M_d = T_d(M_1)$. The above derivation is valid in more general settings where the number of ancillas of the block encoding can differ from the number of qubits on which the encoded matrix is acting. For future reference, we state the more general result.

Lemma 4. *If a reflection R block encodes a matrix M, then for every $d \in \mathbb{N}$, the unitary $(RR_{|0^\ell*\rangle})^d$ block encodes $T_d(M)$ with the same number of ancillas, where T_d denotes the Chebyshev polynomial of degree d.*

Combining the generic lemma with Fact 3, we have shown that d steps of our quantum walk block encode T_d applied to the geometric symmetrization of T up to a basis change with U.

Corollary 5. *For every $d \in \mathbb{N}$, $U^*(SC)^d U$ block encodes $T_d(\widetilde{T})$.*

2.2 Chebyshev polynomials

Chebyshev polynomials have many interesting properties, some of which can be used as alternate definitions. We derive the ones that we need based on our definition (2), and state a few more. First of all, induction on d shows that T_d is indeed a polynomial of degree d. Second, T_d is the polynomial that expresses $\cos(d\theta)$ as a function of $\cos(\theta)$.

Fact 6. $\cos(d\theta) = T_d(\cos(\theta))$ *for all* $\theta \in \mathbb{R}$.

Proof. This property also follows by induction on d. For $d = 0$, $\cos(0 \cdot \theta) = \cos(0) = 1 = T_0(\cos\theta)$, and the case $d = 1$ holds because T_1 is the identity function. For $d \geq 1$ we have

$$\begin{aligned} T_{d+1}(\cos(\theta)) &= 2\cos(\theta)T_d(\cos(\theta)) - T_{d-1}(\cos(\theta)) && \text{(recurrence for } T_d\text{)} \\ &= 2\cos(\theta)\cos(d\theta) - \cos((d-1)\theta) && \text{(inductive hypothesis)} \\ &= 2\cos(\theta)\cos(d\theta) - (\cos(d\theta)\cos(\theta) + \sin(d\theta)\sin(\theta)) && \text{(trigonometric identity)} \\ &= \cos(\theta)\cos(d\theta) - \sin(d\theta)\sin(\theta) && \text{(rearranging)} \\ &= \cos((d+1)\theta) && \text{(trigonometric identity),} \end{aligned}$$

where we twice used the trigonometric identity $\cos(\alpha + \beta) = \cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta)$. □

Based on Fact 6, an explicit expression for $T_d(x)$ can be obtained from de Moivre's formula

$$\cos(d\theta) + i\sin(d\theta) = e^{id\theta} = (e^{i\theta})^d = (\cos(\theta) + i\sin(\theta))^d.$$

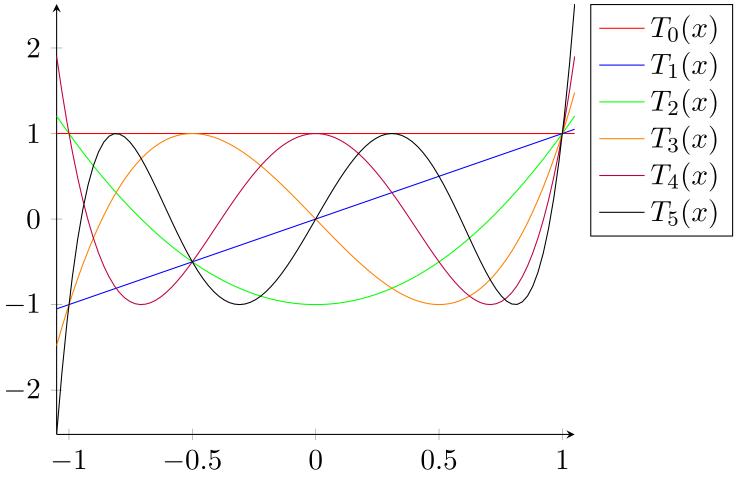

Figure 1: Plot of the first few Chebyshev polynomials

Applying the binomial theorem, rewriting $\sin^2(\theta)$ as $1 - \cos^2(\theta)$, and taking the real parts on both sides yields

$$T_d(x) = \sum_{k=0}^{\lfloor d/2 \rfloor} \binom{d}{2k} (-1)^k (1-x^2)^k x^{d-2k}.$$

Fact 6 implies that $T_d(x) \in [-1,1]$ for $x \in [-1,1]$. Among all polynomials of degree d whose graph is contained within the box $[-1,1] \times [-1,1]$, T_d is extremal in several respects. For example, T_d achieves the largest maximum derivative in absolute value over $[-1,1]$, as well as the largest absolute value at any point $x \in \mathbb{R}$ outside of $[-1,1]$. The plot in Figure 1 illustrates these properties.

Chebyshev approximations. Any polynomial of degree t can be expressed exactly as a linear combination of the Chebyshev polynomials of degree up to t. We are particularly interested in the Chebyshev expansion of x^t. Using the convention that $T_{-d} \doteq T_d$ for positive integers d, we can write the expansion as follows.

Fact 7. *For every $t \in \mathbb{N}$,*

$$x^t = E_{s_t}[T_{s_t}(x)], \tag{3}$$

where s_t is the sum of t independent uniform ± 1 random variables.

Proof. Rearranging the recurrence (2), we have that

$$xT_d(x) = \frac{1}{2}(T_{d+1}(x) + T_{d-1}(x)) = E_\Delta[T_{d+\Delta}(x)], \tag{4}$$

where $\Delta \in_u \pm 1$. The relationship holds for integers $d \geq 1$ by the recurrence. It extends to $d < 0$ by the convention that $T_d \doteq T_{-d}$, and also holds for $d = 0$ because $T_0(x) = 1$ and $T_{-1}(x) = T_1(x) = x$. Thus, the relationship holds for all integers d.

The expansion (3) then follows by induction on t. The base case $t = 0$ holds because $s_0 = 0$ and $T_0(x) = x^0$. For the induction step we have for $t \geq 0$

$$x^{t+1} = x \cdot x^t = x \cdot E_{s_t}[T_{s_t}(x)] = E_{s_t}[xT_{s_t}(x)] = E_{s_t}[E_\Delta[T_{s_t+\Delta}(x)]] = E_{s_{t+1}}[T_{s_{t+1}}(x)],$$

where the second equality follows from the induction hypothesis, the third from linearity of expectation, the fourth from (4), and the last one because $s_t + \Delta$ is a sum of $t+1$ independent ± 1 random variables. □

Note that the right-hand side of (3) is a linear combination of $T_d(x)$ where d ranges over the integers with $|d| \leq t$, namely $\sum_{d=-t}^{t} q_{t,d} T_d(x)$, where

$$q_{t,d} \doteq \Pr[s_t = d] = \Pr[\text{sum of } t \text{ independent uniform} \pm 1 \text{ random variables equals } d]. \tag{5}$$

By concentration of measure, most of the weight in the expansion (3) lies on degree d up to $\Theta(\sqrt{t})$. This allows us to approximate x^t closely as a linear combination of Chebyshev polynomials of degree at most $a = \Theta(\sqrt{t})$, namely as the truncated Chebyshev expansion

$$p_{t,a}(x) \doteq \sum_{|d|<a} q_{t,d} T_d(x). \tag{6}$$

We can bound the error of the approximation as follows for $x \in [-1, 1]$:

$$\begin{aligned}
|x^t - p_{t,a}(x)| &= |\sum_{|d|\geq a} q_{t,d} T_d(x)| \\
&\leq \sum_{|d|\geq a} q_{t,d} |T_d(x)| && \text{(triangle inequality)} \\
&\leq \sum_{|d|\geq a} q_{t,d} && (|T_d(x)| \leq 1 \text{ for } x \in [-1,1]) \\
&= \Pr[|s_t| \geq a] && \text{(equation (5))} \\
&\leq 2 \cdot \exp\left(-a^2/(2t)\right) && \text{(Chernoff bound).}
\end{aligned}$$

We conclude:

Fact 8. *For any $a \geq \sqrt{2t \cdot \ln(2/\epsilon)}$, $|x^t - p_{t,a}(x)| \leq \epsilon$ for every $x \in [-1, 1]$, where $p_{t,a}$ is defined by (6) and (5).*

Polynomial approximations to scalars generically extend to matrices that have a full basis of eigenvectors as long as the eigenvalues fall within the domain of the approximation. If the basis is orthonormal, the error in 2-norm of the matrix approximation is bounded by the error in the scalar approximation.

Fact 9. *Suppose that $M \in \mathbb{C}^{N\times N}$ has a full orthonormal basis of eigenvectors, and $p, q \in \mathbb{C}[z]$. If $|p(\lambda) - q(\lambda)| \leq \epsilon$ for every eigenvalue λ of M, then $\|p(M) - q(M)\|_2 \leq \epsilon$.*

Proof. Let $V \in \mathbb{C}^{N\times N}$ be a unitary matrix such that the columns of V form an orthonormal eigenbasis for M: $M = V^* \operatorname{Diag}(\lambda) V$, where we use $\operatorname{Diag}(\lambda)$ as a shorthand for $\operatorname{Diag}(\lambda_1, \ldots, \lambda_N)$, the diagonal matrix with the eigenvalues of M on the diagonal. Note that $p(M)$ and $q(M)$ share the eigenbasis V with M.

$$\begin{aligned}
\|p(M) - q(M)\|_2 &= \|V^* p(\mathrm{Diag}(\lambda))V - V^* q(\mathrm{Diag}(\lambda))V\|_2 && \text{(shared eigenbasis)} \\
&= \|V^*(p-q)(\mathrm{Diag}(\lambda))V\| && \text{(linearity)} \\
&= \|(p-q)(\mathrm{Diag}(\lambda))\|_2 && \text{(unitary invariance of 2-norm)} \\
&= \|\,\mathrm{Diag}((p-q)(\lambda_1), \dots, (p-q)(\lambda_N)\|_2 && \text{(polynomial of diagonal matrix)} \\
&= \max_{1 \leq j \leq N} |p(\lambda_j) - q(\lambda_j)| && \text{(2-norm of diagonal matrix)} \\
&\leq \epsilon && \text{(hypothesis)} \qquad \square
\end{aligned}$$

Consider a matrix M that is block encoded by a reflection R. Since R is Hermitian, so is M. Thus, M has a full orthonormal basis of eigenvectors and all its eigenvalues are real. Moreover, as a block encoded matrix, all its eigenvalues have absolute value at most 1. Hermitian matrices always have a full orthonormal basis of eigenvectors and all their eigenvalues are real. By Lemma 4, Fact 8, and linearity we conclude:

Lemma 10. *If a reflection R block encodes a matrix M, then for every $t \in \mathbb{N}$, $\epsilon > 0$ and $a \geq \sqrt{2t \cdot \ln(2/\epsilon)}$, $\sum_{|d|<a} q_{t,d}(RR_{|0^\ell *\rangle})^d$ block encodes a matrix M' such that $\|M^t - M'\|_2 \leq \epsilon$, where $q_{t,d}$ is given by* (5).

In combination with Fact 3, we obtain:

Corollary 11. *For any $t \in \mathbb{N}$, $\epsilon > 0$, and $a \geq \sqrt{2t \cdot \ln(2/\epsilon)}$, $U^* \left(\sum_{|d|<a} q_{t,d}(SC)^d\right) U$ block encodes a matrix M such that $\|\widetilde{T}^t - M\|_2 \leq \epsilon$, where $q_{t,d}$ is given by* (5).

2.3 Linear Combination of Unitaries

Corollary 11 gives us a linear combination $L \doteq \sum_{|d|<a} q_{t,d}(SC)^d$ of unitaries such that $U^* L U$ block encodes a good approximation M to T^t. We have a unitary circuit for each of the constituting unitaries $(SC)^d$, namely as $(SUR_{|0^n *\rangle}U^*)^d$. If we were able to efficiently construct a block encoding A for L, we would be home free: $(I \otimes U^*)A(I \otimes U)$ then block encodes M.

Next lecture we will develop a technique to obtain a block encoding for a generic linear combination of unitaries. The technique is known as the Linear Combination of Unitaries method, or LCU for short. It comes at the cost of $\ell = O(\log a)$ additional ancillas and only provides a block encoding for the linear combination up to a scalar. The resulting unitary circuit for A uses $q = O(a)$ controlled applications of the quantum walk operator SC, and $O(a\ell \log(1/\epsilon))$ other elementary quantum gates, the parameters of which can be computed classically in time $\mathrm{poly}(a, \log(1/\epsilon))$. This way we achieve complexity bounds stated in the Fast Forwarding Lemma.

The LCU construction and proof will be presented in the next lecture. In preparation, solve the following special case.

Exercise #10: Construct a block encoding for the sum $U_1 + U_2$ of two unitaries on n qubits, up to some scalar. You can assume access to a selector operator U on $n + 1$ qubits, where the first qubit selects which of U_1 or U_2 to apply: $U |b\rangle |\psi\rangle = |b\rangle U_{b+1} |\psi\rangle$. Aim for a number of ancillas that is as small as possible, and a scalar that is as large as possible.

3 Interpolating Walks

To find a good vertex using a random walk, we start from a vertex chosen from the initial distribution $\sigma = \pi$, and keep walking until we hit a good vertex. This means that we check f before each step of the walk. The underlying transition matrix is an absorbing version T_f of the transition matrix T of the unconstrained random walk, obtained by replacing the columns in T corresponding to good vertices ($f(v) = 1$) by the point distribution at that vertex.

In an attempt to speed up the process on a quantum computer, we could run the quantum walk starting from $|\Pi\rangle$, evaluate the success predicate in every step by applying U_f on the second register and a clean ancilla, measure the ancilla, and measure the second register when the measurement of the ancilla indicates success. This approach mimics the classical process closely. The Fast Forwarding Lemma does not apply, though, as the Markov chain T_f is not reversible.

The approach we follows is to create $|0^n\rangle\,|\sqrt{\pi}\rangle$ and perform the initial success check as above, but in case of failure apply the block encoding from the Fast Forwarding Lemma to the state $|0^n\rangle\,|B\rangle$, where $|B\rangle$ is the normalized projection of $|\sqrt{\pi}\rangle$ onto the bad vertices. The initial check takes care of cases where the weight of $|\sqrt{\pi}\rangle$ on the good vertices is fair, so in the rest of the analysis we only need to worry about cases where the weight is small, or equivalently, when the component of $|B\rangle$ along $|\sqrt{\pi}\rangle$ is fair. As the component along $|\sqrt{\pi}\rangle$ is invariant under $\widetilde{T}$, this means that $\|\widetilde{T}^t\,|B\rangle\,\|_2$ remains fair for any t, so the application of the block encoding of M to $|B\rangle$ has a fair probability of success.

The question is what t we should use. By the definition of the hitting time H and Markov's inequality, with probability of at least 50%, the classical process ends within the first $2H$ steps. The Fast Forwarding Lemma suggests we should not set t higher than $\Theta(\sqrt{H})$, but the precise choice matters and is unclear. Similar to our version of Grover search with unknown initial weight, we could pick t uniformly at random between 0 and an upper bound of $\Theta(\sqrt{H})$. This works provided there are many values of t in that range where the second register of the quantum system has a fair amount of weight on the good vertices. This may not be the case for the Markov chain defined by T, but can be obtained by adequately *interpolating* between T and the absorbing version T_f.

Interpolating walks. We consider the Markov chain with transition matrix $T(a) \doteq (1-a)T + aT_f$, where a ranges over $[0,1]$. For any parameter setting other than $a = 1$, $T(a)$ is reversible. Indeed, $T(a)$ corresponds to a random walk on the original weighted graph $G = (V, E)$ but with an additional self-loop of weight $c \cdot w(v)$ at every good vertex v, where $c = \frac{a}{1-a}$. The effect in the quantum walk shows up in the the operator U, which now also depends on the parameter a.

Exercise: Show how to efficiently implement $U(a)$ for $a \in [0,1)$ using two applications of U_f and a controlled application of U.

The higher a, the longer the walk stays at a good vertex once reached, and the more choices of $t \in \Theta(\sqrt{H})$ result in fair success for the quantum walk to hit a good vertex. On the other hand, higher values of a result in the stationary distribution $\pi(a)$ having more weight on the good vertices and less on the bad vertices, and therefore the component of $|B\rangle$ along $|\sqrt{\pi(a)}\rangle$ being smaller, and thus the success probability $\|\widetilde{T(a)}^t\,|B\rangle\,\|_2^2$ of the application of the block encoding being smaller. Adequate values of a balance between these two effects. A detailed analysis shows that we can pick a at random from a small range and ensure a decent probability of choosing a value that achieves both goals and therefore guarantees overall success with a pair probability.

Lemma 12. *If $\mu \doteq \Pr_{v\ \pi}[f(v) = 1] \leq \frac{1}{10}$, then for any integer $\tau \in \Omega(\sqrt{H})$, the expected weight of $\widetilde{T(a)}^{t} |B\rangle$ on the vertices v with $f(v) = 1$ is $\Omega(\frac{1}{\log(\tau)})$, where $a \doteq 1 - \frac{1}{2^r}$, $r \in_u [O(\log(\tau))]$ and $t \in_u [\tau]$.*

4 Algorithm

Taken together, we have designed the following subroutine for quantum walk search.

1. Create the superposition $|\pi\rangle \doteq \sum_v \sqrt{\pi_v}\,|v\rangle$.
2. Evaluate goodness in a fresh ancilla by running U_f on the register and the ancilla, and measuring the ancilla. If we measure 1, we measure the register, output the result and stop. Otherwise the register is in state $|B\rangle$.
3. Pick a and t for τ as in Lemma 12.
4. Run fast forwarding for the random walk with transition matrix $T(a)$. This yields a block encoding A of a matrix M.
5. Apply A to $|0^n\rangle\,|B\rangle$.
6. Evaluate goodness as above. of the ancilla again. If we measure 1, measure the second register,, and output the result. Otherwise, we report failure.

We set $\tau = \Theta(\sqrt{H})$. A single run of the subroutine costs $O(s + \sqrt{H}(u + c))$ and has a success probability of at least $\min(\frac{1}{10}, \Omega(\frac{1}{\log(H)}))$. Using amplitude amplification, we can boost the confidence to 2/3 at a multiplicative cost of $\sqrt{\log(H)}$. This gives the complexity bounds from Theorem 1 and completes our discussion of how a quantum walk can perform search for a good item in a number of steps that is roughly only the square root of the classical hitting time.