Week 3

The Adversary Method

3.1 Primal form

The adversary method, first introduced by Ambainis [Amb03], can be viewed as a generalization of the hybrid method. Recall that the quantum hybrid method says that a T-query quantum Q which distinguishes x from y to error ϵ must satisfy

$$\sum_{i\in[n]:x_i\neq y_i} m_i \geq \frac{(1-2\epsilon)^2}{4T}$$

where m_i is a measure of many times the quantum algorithm queried bit i when run on x. Let us denote this by $m_i(x)$ instead of just m_i, to highlight the dependence on the input string (the algorithm may query in different places depending on the input, with the exception of the first query). Then one version of the (positive) adversary argument gives the following generalization:

$$\sum_{i\in[n]:x_i\neq y_i} \sqrt{m_i(x)m_i(y)} \geq \frac{(1-2\epsilon)^2}{4} \tag{3.1}$$

for all pairs (x,y) that are distinguished by Q to error ϵ. In particular, applying Cauchy-Schwartz and using $\sum_i m_i(y) \leq T$, we can rederive the quantum hybrid method from (3.1) except with a worse upfront constant and a worse dependent on ϵ (neither of which matters if we do not care about constant factors and set $\epsilon = 1/3$).

The (positive) adversary method is usually introduced in terms of a query complexity measure, analogous to fractional certificate complexity. We define it as follows.

Definition 3.1 (Positive quantum adversary, primal version)**.** *Let f be a (possibly partial) Boolean function. We define the positive quantum adversary complexity of f, denoted* $\mathrm{Adv}(f)$ *or* $\mathrm{Adv}^+(f)$*, as minimum solution of the following optimization problem. For each $x \in \mathrm{Dom}(f)$ and $i \in [n]$, we have a weight $w_{x,i} \geq 0$. The constraint is that for all $x, y \in \mathrm{Dom}(f)$ with $f(x) \neq f(y)$, we have*

$$\sum_{i\in[n]:x_i\neq y_i} \sqrt{w_{x,i}w_{y,i}} \geq 1.$$

The objective function we wish to minimize is the maximum, over $x \in \mathrm{Dom}(f)$, of $\sum_{i\in[n]} w_{x,i}$.

We note that the positive quantum adversary is similar in spirit to fractional certificate complexity. For the latter, we assign a fractional certificate to each input x; if we denote its weights by

$w_{x,i}$, then we would require $w_{x,i} \geq 0$ and $\mathrm{FC}(f)$ would be the minimum of $\max_{x \in \mathrm{Dom}(f)} \sum_{i \in [n]} w_{x,i}$. So far this is exactly the same as $\mathrm{Adv}(f)$. Where they differ is in the constraint placed on the weights: fractional certificate complexity required $\sum_{i \in [n]: x_i \neq y_i} w_{x,i} \geq 1$ for all $x, y \in \mathrm{Dom}(f)$ with $f(x) \neq f(y)$, while the positive adversary bound replaced the summand $w_{x,i}$ with $\sqrt{w_{x,i} w_{y,i}}$. We have the following theorem.

Theorem 3.2. *Let f be a (possibly partial) Boolean function. Then*

$$\mathrm{Q}_\epsilon(f) \geq \frac{(1-2\epsilon)^2}{4} \mathrm{Adv}^+(f).$$

Proof. Let Q be a quantum algorithm solving f to error ϵ using $T = \mathrm{Q}_\epsilon(f)$ queries. To prove this theorem, we only need to come up with a feasible set of weights $w_{x,i}$ such that $\sum_i w_{x,i} \leq 4T/(1-2\epsilon)^2$ for all $x \in \mathrm{Dom}(f)$. We will pick $w_{x,i} = 4m_i(x)/(1-2\epsilon)^2$, where $m_i(x)$ is defined as in the hybrid argument (representing the expected number of queries Q makes to bit i when run on x). Then $\sum_i w_{x,i} = 4/(1-2\epsilon)^2 \cdot \sum_i m_i(x) = 4T/(1-2\epsilon)^2$, and $w_{x,i} \geq 0$. We also have

$$\sum_{i: x_i \neq y_i} \sqrt{w_{x,i} w_{y,i}} = \frac{4}{(1-2\epsilon)^2} \sum_{i: x_i \neq y_i} \sqrt{m_i(x) m_i(y)},$$

which means that to complete the proof, we only need to prove (3.1).

Proving this will proceed along similar lines to the hybrid argument. The main difference is that we will split up $\left\| |\psi^x_{T+1}\rangle - |\psi^y_{T+1}\rangle \right\|^2$ into a telescoping sum instead of splitting up $\left\| |\psi^x_{T+1}\rangle - |\psi^y_{T+1}\rangle \right\|$ into such a sum. Recall from the hybrid argument that

$$\left\| |\psi^x_{T+1}\rangle - |\psi^y_{T+1}\rangle \right\|^2 \geq 2 - 4\sqrt{\epsilon(1-\epsilon)}.$$

We write

$$\begin{aligned}
2 - 4\sqrt{\epsilon(1-\epsilon)} &\leq \left\| |\psi^x_{T+1}\rangle - |\psi^y_{T+1}\rangle \right\|^2 \\
&= \sum_{t=1}^{T} \left\| |\psi^x_{t+1}\rangle - |\psi^y_{t+1}\rangle \right\|^2 - \left\| |\psi^x_t\rangle - |\psi^y_t\rangle \right\|^2 \\
&= \sum_{t=1}^{T} 2\Re\left(\langle \psi^x_t | \psi^y_t \rangle\right) - 2\Re\left(\langle \psi^x_{t+1} | \psi^y_{t+1} \rangle\right) \\
&= 2\Re\left(\sum_{t=1}^{T} \langle \psi^x_t | \psi^y_t \rangle - \langle \psi^x_{t+1} | (U^x)^\dagger U_t^\dagger U_t U^y | \psi^y_{t+1} \rangle \right) \\
&= 2\Re\left(\sum_{t=1}^{T} \langle \psi^x_t | I - (U^x)^\dagger U^y | \psi^y_t \rangle \right).
\end{aligned}$$

Here we used $\left\| |\psi^x_0\rangle - |\psi^y_0\rangle \right\|^2 = 0$ in the second line. The third line followed from the identity

$$\left\| |\psi^x_t\rangle - |\psi^y_t\rangle \right\|^2 = 2 - 2\Re(\langle \psi^x_t | \psi^y_t \rangle)$$

which we showed in the proof of the hybrid argument. The fourth line uses the definition $|\psi^x_{t+1}\rangle = U_t U^x |\psi^x_t\rangle$, and the last line followed using the fact that U_t is unitary so $U_t^\dagger U_t = I$.

Next, just as in the hybrid argument, we break up the analysis by the query register $|i\rangle$. That is, let Π_i be the projector onto the query register being $|i\rangle$, as in the proof of the hybrid argument.

Then we have $|\psi_t^x\rangle = \sum_{i\in[n]} \Pi_i |\psi_t^x\rangle$. Moreover, the projection matrix Π_i satisfies $\Pi_i^\dagger = \Pi_i$, $\Pi_i^2 = \Pi_i$, and $\Pi_i U^x = U^x \Pi_i$ (which follows from the fact that U^x does not change the query register). Also note that $(U^x)^\dagger U^y \Pi_i = \Pi_i$ when $x_i = y_i$, since when the query register is i, the matrices U^x and U^y have the same action, so $(U^x)^\dagger U^y$ behaves as identity. This also means that $\Pi_i(I - (U^x)^\dagger U^y) = 0$. Finally, another property is that $\Pi_i \Pi_j = 0$ when $i \neq j$. Putting this together, we get

$$\langle\psi_t^x|I - (U^x)^\dagger U^y|\psi_t^y\rangle = \langle\psi_t^x| \sum_i \Pi_i (I - (U^x)^\dagger U^y) \sum_j \Pi_j |\psi_t^y\rangle = \sum_i \sum_j \langle\psi_t^x|\Pi_i\Pi_j(I - (U^x)^\dagger U^y)|\psi_t^y\rangle$$

$$= \sum_{i:x_i\neq y_i} \langle\psi_t^x|\Pi_i(I - (U^x)^\dagger U^y)|\psi_t^y\rangle\,.$$

This means that by continuing the chain from above, we actually have

$$\begin{aligned}
\left\| |\psi_{T+1}^x\rangle - |\psi_{T+1}^y\rangle \right\|^2 &= 2\Re\left(\sum_{t=1}^T \sum_{i:x_i\neq y_i} \langle\psi_t^x|\Pi_i(I - (U^x)^\dagger U^y)|\psi_t^y\rangle\right) \\
&= 2\sum_{i:x_i\neq y_i} \sum_{t=1}^T \Re(\langle\psi_t^x|\Pi_i(I - (U^x)^\dagger U^y)|\psi_t^y\rangle) \\
&\leq 2\sum_{i:x_i\neq y_i} \sum_{t=1}^T |\,\langle\psi_t^x|\Pi_i(I - (U^x)^\dagger U^y)|\psi_t^y\rangle\,| \\
&\leq 2\sum_{i:x_i\neq y_i} \sum_{t=1}^T |\,\langle\Pi_i\psi_t^x|\Pi_i\psi_t^y\rangle\,| + |\,\langle U^x\Pi_i\psi_t^x|U^y\Pi_i\psi_t^y\rangle\,| \\
&\leq 2\sum_{i:x_i\neq y_i} \sum_{t=1}^T \| \,|\Pi_i\psi_t^x\rangle\,\| \cdot \| \,|\Pi_i\psi_t^y\rangle\,\| + \| \,|U^x\Pi_i\psi_t^x\rangle\,\| \cdot \| \,|U^y\Pi_i\psi_t^y\rangle\,\| \\
&= 4\sum_{i:x_i\neq y_i} \sum_{t=1}^T \| \,|\Pi_i\psi_t^x\rangle\,\| \cdot \| \,|\Pi_i\psi_t^y\rangle\,\| \\
&= 4\sum_{i:x_i\neq y_i} \sum_{t=1}^T \sqrt{m_i^t(x) m_i^t(y)} \\
&\leq 4\sum_{i:x_i\neq y_i} \sqrt{m_i(x) m_i(y)}.
\end{aligned}$$

Here the second line follows from rearranging the sums, the third from replacing the real part of a complex number with the magnitude of the complex number, the fourth from triangle inequality, the fifth from Cauchy-Schwartz, the sixth using the fact that the unitaries U^x and U^y preserve the norm of vectors, the seventh from the definition of $m_i^t(x)$, and the last using Cauchy-Schwartz again. We therefore conclude that

$$\sum_{i:x_i\neq y_i} \sqrt{m_i(x) m_i(y)} \geq \frac{1 - 2\sqrt{\epsilon(1-\epsilon)}}{2}.$$

Finally, the desired result follows from $1 - 2\sqrt{\epsilon(1-\epsilon)} \geq (1-2\epsilon)^2/2$, which we showed in the proof of the hybrid argument. □

The lower bound on $\sum_{i:x_i \neq y_i} \sqrt{m_i(x) m_i(y)}$ is remarkable. For one thing, it gives (by Cauchy-Schwartz) the weaker statement

$$\sqrt{\sum_{i:x_i \neq y_i} m_i(x) \sum_{i:x_i \neq y_i} m_i(y)} \geq \frac{(1-2\epsilon)^2}{4}.$$

This weaker statement is already quite significant: recall that in the hybrid argument we could only lower bound $\sum_{i:x_i \neq y_i} m_i(x)$ by $\Omega(1/T)$ rather than $\Omega(1)$, and this is actually tight (by Grover search). However, the adversary method says that although $\sum_{i:x_i \neq y_i} m_i(x)$ might be as small as $1/T$ for a quantum algorithm distinguishing x from y, and although $\sum_{i:x_i \neq y_i} m_i(y)$ might also be as small as $1/T$, they cannot both be small at one—their product is $\Omega(1)$! In particular, if $\sum_{i:x_i \neq y_i} m_i(x)$ is $O(1/T)$, then not only must $\sum_{i:x_i \neq y_i} m_i(y)$ be $\Omega(1)$, but in fact it must be $\Omega(T)$.

To put it another way: suppose a quantum algorithm Q distinguishes x from y, and let $B \subseteq [n]$ be the set of bits i with $x_i \neq y_i$. Then if Q queries inside B with only small probability mass when run on x, then it must query inside B with very large probability mass (greater than 1) when run on y, corresponding to making repeated, "redundant" queries inside B. In particular, Grover search distinguishes the all-zero input 0^n from the input 10^{n-1} which has a 1. It does so in $O(\sqrt{n})$ total queries, placing probability mass only $O(1/\sqrt{n})$ on each $i \in [n]$ when run on 0^n. However, when Grover search runs on 10^{n-1}, it must place probability mass $\Omega(\sqrt{n})$ on the first bit $i = 1$, effectively querying it again and again. This is very strange behavior! It has to do with the fact that quantum algorithms don't query the input in the normal way (in which exactly one bit is revealed and can be remembered forever after), but rather query in superposition.

3.2 Dual form

How do we use the quantum adversary method to prove lower bounds? Well, recall what we did for fractional certificate complexity: we took its dual, fractional block sensitivity, which is a maximization problem rather than a minimization problem. We could then show lower bounds on fractional certificate complexity (and hence on $\mathrm{Q}(f)$ after taking a square root) by giving feasible solutions to the maximization problem.

We do a similar thing for the quantum adversary method. Unfortunately, the quantum adversary bound is not a linear program: the constraints involve terms $\sqrt{w_{x,i} w_{y,i}}$, which is a square root of product of variables, rather than a linear combination of variables. However, rather miraculously, it turns out that the adversary bound can be converted into what's called a *semidefinite program*. Semidefinite programs have variables that are placed inside a symmetric matrix X, and can have constraints that are linear in the entries of X, plus the additional special constraint $X \succeq 0$ (which says that X is a positive semidefinite matrix).

It turns out that we can define duality for semidefinite programs as well, so that for each minimization problem we can define an analogous maximization problem whose objective value lower bounds that of the original program. While strong duality (guaranteeing that the optimal solutions of a program and its dual are exactly the same) does not always hold for semidefinite programs, it "usually" holds: one only needs to show that the program has some nice properties in order to conclude that strong duality will be satisfied.

We are clearly skipping a lot of details here (see [ŠS06], which makes things explicit), but the upshot is that we can convert the adversary bound to a semidefinite program (SDP), take the dual to get a maximization SDP, show that strong duality holds, and then convert this dual SDP into a more readable form. Once we do all this, we get an alternative definition for the quantum adversary, which is called the spectral version.

Definition 3.3 (Positive quantum adversary, spectral version)**.** *Let f be a (possibly partial) Boolean function. Then* $\mathrm{Adv}^+(f)$ *can also be defined as*

$$\max_{\Gamma \geq 0} \min_{i \in [n]} \frac{\|\Gamma\|}{\|\Gamma \circ D_i\|},$$

where Γ ranges over non-negative real symmetric matrices with rows and columns indexed by $\mathrm{Dom}(f)$ *which satisfy $\Gamma[x,y] = 0$ whenever $f(x) = f(y)$. Here D_i is the $\{0,1\}$-matrix with $D_i[x,y] = 1$ if and only if $x_i \neq y_i$, the notation $\circ$ denotes the Hadamard (entrywise) product, and the norm $\|\cdot\|$ is the spectral norm (i.e. $\|A\|$ is the maximum value of $u^T A v$ over vectors u and v with $\|u\|_2 = \|v\|_2 = 1$).*

As noted, this definition of the adversary bound exactly equals the previous (primal) definition, although we do not prove this fact (see [ŠS06] for a proof).

This spectral version has many nice properties, but unfortunately, it is still too cumbersome to give easy lower bounds on quantum query complexity. After all, to use it we would not only have to give a matrix Γ for the function f, we would also have to lower bound the spectral norm $\|\Gamma\|$ and upper bound the spectral norms $\|\Gamma \circ D_i\|$ for all i.

Luckily, there is yet another equivalent version of the quantum adversary bound.

Definition 3.4 (Positive quantum adversary, weighted form)**.** *Let f be a (possibly partial) Boolean function. Then* $\mathrm{Adv}^+(f)$ *can also be defined as the maximum of the following optimization problem. The variables are weights $w(x,y) \geq 0$ and $w'(x,y,i) \geq 0$ for all $x, y \in \mathrm{Dom}(f)$ with $f(x) \neq f(y)$ and for all $i \in [n]$ such that $x_i \neq y_i$. They must satisfy $w(x,y) = w(y,x)$ and $w'(x,y,i)w'(y,x,i) \geq w(x,y)^2$ for all such x, y, and i. Define $wt(x) \coloneqq \sum_{y:f(y)\neq f(x)} w(x,y)$ and $wt'(x,i) \coloneqq \sum_{y:f(y)\neq f(x), y_i \neq x_i} w'(x,y,i)$. The objective function is*

$$\min_{x,y,i:w(x,y)>0, x_i \neq y_i} \sqrt{\frac{wt(x)wt(y)}{wt'(x,i)wt'(y,i)}}.$$

Once again, we omit the proof that this equals the other definitions of the positive adversary bound (see [ŠS06] for the proof).

To use this definition, we will almost always actually pick $w'(x,y,i) = w(x,y)$ for all x, y, and i. Note that doing so is always allowed. This substantially simplifies the bound; at this point, the weights $w(x,y)$ can be interpreted as weights on the edges of a complete bipartite graph, the vertex sets corresponding to 0-inputs and 1-inputs of f (if the outputs of f are not Boolean, we this generalizes nicely to a k-partite graph). Then $wt(x)$ is the total weight of the edges adjacent to x (essentially the degree of x is this weighted graph). Further, $wt'(x,i)$ is the total weight of the edges adjacent to x that connect x to an input y with $x_i \neq y_i$. This can be interpreted as the degree of x in the subgraph where we've deleted all edges $\{x,y\}$ that have $x_i = y_i$.

In other words, feasible solutions to the weighted version of the positive adversary take the form of a weighted bipartite graph G; if we denote by $\deg(x,G)$ the (weighted) degree of x in G, and if we denote by G_i the subgraph of G where we only keep edges $\{x,y\}$ if $x_i \neq y_i$, then the weighted adversary has the form

$$\max_G \min_{x,y,i:w(x,y)>0, x_i \neq y_i} \sqrt{\frac{\deg(x,G)\deg(y,G)}{\deg(x,G_i)\deg(y,G_i)}}.$$

Of course, this is not the full weighted adversary; for that, we would need to reintroduce the weights $w'(x,y,i)$, which function as *reweighting* of the edge $\{x,y\}$ inside the subgraph G_i. Instead

of assigning this edge the weight $w(x,y)$ in the subgraph G_i (the same weight as in G), we can instead pick two directional weights, one from the x node and another from the y node, such that the geometric mean of these two weights is at least $w(x,y)$. When determining the degree of x in G_i, we only add up its directional weights $w'(x,y,i)$, which can be smaller than $w(x,y)$ (at the cost of making the directional weight for y, the weight $w'(y,x,i)$, larger than $w(x,y)$). While this complication is necessary to ensure that this definition of the adversary bound equals the previous ones, in practice we rarely use it and just set $w'(x,y,i) = w(x,y)$.

Finally, we can weaken the adversary bound even further, making it even simpler to use. Instead of minimizing over all triples (x,y,i), we can simply let M be the minimum non-zero degree of a 0-input in G, let M' be the minimum non-zero degree of a 1-input in G, and let L_i and L'_i be the *maximum* degrees of 0-inputs and 1-inputs (repectively) in the subgraph G_i. Then we can lower bound the adversary bound by

$$\max_G \min_i \sqrt{\frac{MM'}{LL'}}.$$

In fact, we will often pick the weights in G to be $\{0,1\}$ weights. Then G is just any (unweighted) bipartite graph. Moreover, we can discard all the vertices with degree 0, as they don't affect anything. So the goal is just to pick two sets $X \subseteq f^{-1}(0)$ and $Y \subseteq f^{-1}(1)$, as well as a bipartite graph on (X,Y), such that

1. every 0-input in X is connected to at least M 1-inputs,
2. every 1-input in Y is connected to at least M' 0-inputs,
3. for each i, every 0-input in X is connected to at most L 1-inputs that disagree with it on bit i,
4. for each i, every 1-input in Y is connected to at most L 0-inputs that disagree with it on bit i.

Once we've picked such a graph, we will have $\mathrm{Adv}^+(f) \geq \sqrt{MM'/LL'}$.

3.3 Applications

The adversary bound is one of the most useful quantum lower bound techniques. If you only learn one technique from this course, it should be this one.

3.3.1 Reproving previous results

Let's first use it to reprove the lower bound on OR_n. To do so, we first pick the sets X and Y mentioned above. We will set $X = \{0^n\}$, and set Y to be the set of all strings in $\{0,1\}^n$ of Hamming weight 1. Note that we deliberately picked X and Y to be as "close" to each other as possible. We then connect 0^n to every input in Y by an edge in our bipartite graph. Now, we will have $M = n$, since the only 0-input we've included has degree n. We will have $M' = 1$, since every 1-input we've included has degree 1. Further, for each index $i \in [n]$, it is clear that there is exactly one input in Y that disagrees with 0^n on i, and for each $y \in Y$ there is at most 1 input in X that disagrees with y on i. Hence $L = L' = 1$. This means that $\mathrm{Adv}^+(f) \geq \sqrt{n \cdot 1/1 \cdot 1} = \sqrt{n}$, so $\mathrm{Q}_\epsilon(\mathrm{OR}_n) \geq (1-2\epsilon)^2/4 \cdot \sqrt{n}$.

Since we only used the Hamming weight 0 and Hamming weight 1 strings for this lower bound, it also works to lower bound $\mathrm{Q}(\textsc{PromiseOR}_n)$, giving the same bound. This also shows that

$\mathrm{Adv}^+(\textsc{PromiseOR}_n) \geq \sqrt{n}$, which by the reduction shown in the first week implies that $\mathrm{Adv}^+(f) \geq \sqrt{\mathrm{bs}(f)}$ for all f.

Next, we will show that the adversary bound is at least $\sqrt{\mathrm{fbs}(f)}$, the lower bound we established last week. To do this, we recall that $\mathrm{fbs}(f)$ is the maximum over z of $\mathrm{fbs}(f,z)$, so pick z maximizing the latter. Then there is a weight scheme over the sensitive blocks of z, $\alpha_B \geq 0$, such that $\sum_B \alpha_B = \mathrm{fbs}(f)$ and for each $i \in [n]$, $\sum_{B:i\in B} \alpha_B \leq 1$. We now pick the adversary weight scheme $w(x,y)$. We set $w(x,y) = 0$ if $x \neq z$ and $y \neq z$. Otherwise, if y is such that $f(y) \neq f(z)$, we set $w(z,y) = w(y,z) = \alpha_B$ where $B = \{i \in [n] : y_i \neq z_i\}$. Then $wt(z) = \sum_y w(z,y) = \sum_B \alpha_B = \mathrm{fbs}(f)$ and for all y such that $f(y) \neq f(z)$, $wt(y) = w(z,y)$. Moreover, for all i we set $w'(x,y,i) = w'(y,x,i) = w(x,y)$; then $wt'(z,i) = \sum_{y:y_i\neq z_i} w(z,y) = \sum_{B:i\in B} \alpha_B \leq 1$ and for all y with $f(y) \neq f(z)$, we have $wt'(y,i) = 0$ if $y_i = z_i$ and $wt'(y,i) = w(z,y)$ if $y_i \neq z_i$. The objective value of this solution is

$$\min_{x,y,i:w(x,y)>0,x_i\neq y_i} \sqrt{\frac{wt(x)wt(y)}{wt'(x)wt'(y)}} \geq \min_{y,i:w(z,y)>0,y_i\neq z_i} \sqrt{\frac{\mathrm{fbs}(f)\cdot w(x,y)}{1\cdot w(x,y)}} = \sqrt{\mathrm{fbs}(f)},$$

as desired.

3.3.2 New applications

Example 1. Consider the function $\mathrm{AND}_n \circ \mathrm{OR}_n$, which is defined as the composition of AND and OR; this is a function on strings of length n^2, where the input string is interpreted as n strings $x^1, x^2, \ldots, x^n$ of length n each, and the function evaluates to $\mathrm{AND}_n(\mathrm{OR}_n(x^1), \mathrm{OR}_n(x^2), \ldots, \mathrm{OR}_n(x^n))$. If we arranged the input bits in a matrix with the columns being x^j, this function would ask whether there is a 1 in every column (it would output 1 if yes, and 0 if there is an all-0 column). What is the quantum query complexity of $\mathrm{AND}_n \circ \mathrm{OR}_n$?

Note that the certificate complexity of $\mathrm{AND}_n \circ \mathrm{OR}_n$ is n, because we can prove an input is a 0-input by showing an all-0 column and we can prove it is a 1-input by showing a 1 in every column. So $\mathrm{C}(\mathrm{AND}_n \circ \mathrm{OR}_n) \leq n$, and hence $\mathrm{fbs}(\mathrm{AND}_n \circ \mathrm{OR}_n) \leq n$ and $\mathrm{bs}(\mathrm{AND}_n \circ \mathrm{OR}_n) \leq n$. On the other hand, the sensitivity of $\mathrm{AND}_n \circ \mathrm{OR}_n$ is at least n, so we conclude that all these measures are n for $\mathrm{AND}_n \circ \mathrm{OR}_n$. In terms of quantum lower bounds, this means we can achieve $\Omega\sqrt{n}$ using block sensitivity or fractional block sensitivity, but no better.

Now let's try using the adversary bound. We pick the hard set of 1-inputs Y to be the inputs that have exactly one 1 in each column. We pick the hard set of 0 inputs X to be the inputs that have exactly one all-0 column, and exactly one 1 in each other column. We put an edge between $x \in X$ and $y \in Y$ in our bipartite graph if x and y disagree on exactly one bit. Now, for each 0-input x, flipping any bit in the all-1 column gives a 1-input to which it is adjacent; hence $M = n$. Similarly, for each 1-input y, flipping any of its 1s gives a 0-input to which it is adjacent, so $M' = n$. On the other hand, for each bit i, any $x \in X$ is adjacent to at most one $y \in Y$ for which $x_i \neq y_i$, and vice versa, so $L = L' = 1$. This means $\mathrm{Adv}^+(\mathrm{AND}_n \circ \mathrm{OR}_n) \geq \sqrt{n \cdot n/1 \cdot 1} = n$, so $\mathrm{Q}(\mathrm{AND}_n \circ \mathrm{OR}_n) = \Omega(n)$. This lower bound is tight (Grover search can be modified to compute $\mathrm{AND}_n \circ \mathrm{OR}_n$ in $O(n)$ queries).

Example 2. What is the quantum query complexity of $\textsc{Parity}_n$, the parity function on n bits? It is not hard to see that $\mathrm{s}(\textsc{Parity}_n) = n$, and hence the block sensitivity and fractional block sensitivity are also n. However, this only gives a $\Omega(\sqrt{n})$ quantum lower bound. In fact, the sensitivity measures (including fractional block sensitivity) can only every give lower bounds of $\Omega(\sqrt{n})$ or less. Can we show that the quantum query complexity of parity is $\Omega(n)$?

We use the adversary method. Let X be the set of all 0-inputs, let Y be the set of all 1-inputs, and place edges on pairs $\{x,y\}$ which have Hamming distance 1 from each other. Then each input

has degree n in this graph. On the other hand, if we restrict to a single bit i, then each x only has a single neighbor that disagrees with it on i (since all its neighbors disagree on exactly one bit, which means the only one that disagrees on i is x^i). Hence we have $M = M' = n$ and $L = L' = 1$, so $\mathrm{Adv}^+(\textsc{Parity}) \geq \sqrt{n \cdot n/(1 \cdot 1)} = n$. This gives the $\Omega(n)$ lower bound on quantum query complexity.

Example 3. We define the "permutation inversion" function f as follows. f will take inputs that are strings of length n over an alphabet of length n; that is, $\mathrm{Dom}(f) \subseteq [n]^n$. It will be a partial function, with the promise that the input is a permutation. That is, each $x \in \mathrm{Dom}(f)$ will have the symbols $1, 2, \ldots, n$ occurring exactly once each. The task is to find the 1 symbol. Actually, to make it a decision problem, we will just set $f(x) = 0$ if the index i such that $x_i = 1$ satisfies $i \leq n/2$, and set $f(x) = 1$ if $i > n/2$. What is the quantum query complexity of f?

Intuitively, solving f requires finding the 1 symbol, which is essentially an unstructured search problem. The best algorithm for this should be Grover search, which uses around $\sqrt{n}$ queries. However, proving that there is no better quantum algorithm is tricky. The certificate complexity of f is 1, because to prove the value of $f(x)$ we just need to reveal the position x_i with $x_i = 1$. Even certifying that an input is not in the promise of f is easy: every string not in $\mathrm{Dom}(f)$ has some symbol $j \in [n]$ occurring at least twice, so revealing $x_{i_1} = x_{i_2} = j$ certifies that $x \notin \mathrm{Dom}(f)$. This means that $\mathrm{C}(f) = O(1)$ even if we were to change the definition of $\mathrm{C}(f)$ to count the cost of certifying that a string is not in the domain.

As before, this means that $\mathrm{bs}(f)$ and $\mathrm{fbs}(f)$ are too small to give a good lower bound (in this case, they are $O(1)$). Instead, we will use the adversary method. We will let the set of hard 0-instances X be the set of all 0-inputs, and similarly Y will be the set of all 1-inputs. We will connect $x \in X$ and $y \in Y$ by an edge if they differ on exactly two bits. Note that since $f(x) = 1 = 0$ and $f(y) = 1$, this means that the bits i and j where they differ must satisfy $x_i = 1$ and $y_j = 1$ and $i \leq n/2 < j$, with $x_j = y_i$ (since both strings are permutations).

Note that a string $x \in X$ is connected to around $n/2$ 1-inputs, because to get a 1-input from x we could swap its 1 with any position $j > n/2$. similarly, every string $y \in Y$ is connected to around $n/2$ 0-inputs (we ignore the off-by-one issues due to rounding). Hence $M = M' = n/2$. On the other hand, consider the subgraph in which we keep only edges $\{x, y\}$ in which $x_i \neq y_i$. Assume without loss of generality that $i \leq n/2$. Then for any $y \in Y$, its degree in this subgraph is only 1, because if j is the position with $y_j = 1$, the only 0-input that y is connected to which disagrees with y at i is the string x which is y with y_i and y_j swapped. Since there is only one such string, we have $L' = 1$ and $L \leq n/2$ for this subgraph. In the case $i > n/2$, we would get $L = 1$ and $L' \leq n/2$ instead. Hence in all subgraphs, we have $LL' \leq n/2$, and we conclude that

$$\mathrm{Adv}^+(f) = \Omega\left(\sqrt{\frac{n/2 \cdot n/2}{1 \cdot n/2}}\right) = \Omega(\sqrt{n}).$$

This is tight due to Grover search.

References

[Amb03] Andris Ambainis. "Polynomial degree vs. quantum query complexity". In: *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science (FOCS 2003)*. 2003 (p. 1).

[ŠS06] Robert Špalek and Mario Szegedy. "All Quantum Adversary Methods are Equivalent". In: *Theory of Computing* 2.1 (2006). DOI: 10.4086/toc.2006.v002a001. URL: http://www.theoryofcomputing.org/articles/v002a001 (pp. 4, 5).