Quantum Computation **(CMU 18-859BB, Fall 2015)**

Lecture 14: Reichardt's Theorem I: Definition of Span Programs

October 26, 2015

Lecturer: Ryan O'Donnell *Scribe: Will Griffin*

1 REVIEW: Super-Basic Adversary Method

In the Super-Basic Adversary Method, we take some decision problem $F : \{0,1\}^N \to \{0,1\}$ and define two sets, $Y \subseteq F^{-1}(1)$ and $Z \subseteq F^{-1}(0)$.
Then we define a relation $R \subseteq Y \times Z, R = \{(y,z) : d(y,z) = 1\}$, so R is the set of pairs of $y \in Y, z \in Z$ such that the distance between y and z is 1, so y is only one bit different from z.
Then let $R(y)$ be the set of all y strings in the pairs in R, $R(z)$ the set of all z strings in R, and define m, m' such that:
$m \leq |R(y)|$
$m' \leq |R(z)|$
Then F requires $\Omega(\sqrt{mm'})$ quantum queries.

Unfortunately, this does not provide a good lower bound for many algorithms.

2 Basic Adversary Method

The idea is the same, but the relation between strings in Y and Z is different. Instead of being restricted to picking Y, Z where the Hamming weights of strings in Y and Z differ by 1, choose an arbitrary relation $R \subseteq Y \times Z$.

Theorem 2.1. *Basic Adversary Method:*
For some decision problem $F : \{0,1\}^N \to \{0,1\}$, let $Y \subseteq F^{-1}(1)$ and $Z \subseteq F^{-1}(0)$, and let $R \subseteq Y \times Z$ be a binary relation. If:

$\forall y \in Y$ there exist at least m distinct $z \in Z$ such that $(y,z) \in R$

$\forall z \in Z$ there exist at least m' distinct $y \in Y$ such that $(y,z) \in R$

$\forall y \in Y$, $i \in \{0,1,2,\ldots,N\}$ there are at most l $z \in Z$ such that $(y,z) \in R$ and $y_i \neq z_i$

$\forall z \in Z$, $i \in \{0,1,2,\ldots,N\}$ there are at most l' $y \in Y$ such that $(y,z) \in R$ and $y_i \neq z_i$

Then F requires $\Omega\left(\sqrt{\frac{mm'}{ll'}}\right)$ quantum queries.

The proof is similar to the proof for the super-basic adversary method, and can be found in [HLS07].

Example: Given oracle access to $w \in \{0,1\}^N$, we want to distinguish between strings with Hamming weight 0 or Hamming weight greater than k. First choose the sets Y and Z: let Z be the string of all 0 bits, and Y the set of strings with Hamming weight k, and let $R = Y \times Z$. Then $m = 1$, $m' = \binom{N}{k}$ and $l = 1$. Since the string in Z is all $0s$, l' is the number of strings in Y that have a 1 bit on coordinate i, which is the number of strings of length $N-1$ with Hamming weight $k-1$, so $l' = \binom{N-1}{k-1}$. Then the lower bound is $\sqrt{\frac{\binom{N}{k}}{\binom{N-1}{k-1}}} = \sqrt{\binom{N}{k}}$.
Another example of a lower bound that can be found with this method is graph connectivity. Given query access to the adjacency matrix of a graph G with n vertices, the basic adversary method can show that checking if G is connected requires $n^{3/2}$ queries. This is also the upper bound, which you may have shown in homework 3.

3 Weighted Adversary Method

The basic adversary method does not always give optimal lower bounds for algorithms. This leads to the next generalization: the Weighted Adversary Method. It is similar, and based on the same idea - define the sets Y and Z in the same way, and use a relation R between the elements, but this time, (y, z) pairs in R are given different weights in a weight matrix. Harder to distinguish pairs are given higher weights to get good results.

Definition 3.1. Let Γ be a weight matrix such that $\Gamma[y, z]$ is the weight of (y, z), and $\Gamma[y, z] \in \mathbb{R} \geq 0$, and $\Gamma[y, z] = 0$ if and only if $F(y) = F(z)$. Γ should be symmetric.

Let $\| \Gamma \|$ be the maximum of the absolute values of the eigenvalues of Γ. $\| \Gamma \|$ is the analogue to $\sqrt{mm'}$ in the basic adversary method.

There is also an analog of the query distinguishability, $\sqrt{ll'}$, in the basic adversary.
For $i \in \{0, 1, \ldots n\}$ let Γ_i be the zero-out of Γ except for those $[y, z]$ entries where $y_i \neq z_i$.
Then $max\{\| \Gamma_i \|, i \in 0, \ldots, n\}$ is the analog of $\sqrt{ll'}$.

Theorem 3.2. *Quantum Query lower bound for F is $\frac{\|\Gamma\|}{max\{\|\Gamma_i\|, i \in 0,\ldots,n\}}$ [HLS07]*

This proof is more difficult.
Fun Fact: for 2:1 collision problem, this method gives a constant time lower bound.
Other Fun Fact: After the weighted adversary method, there were several different new

adversary methods developed, that appear different. They were all proven to be essentially equivalent, see [SS06]. This suggests that the general adversary method is very robust and powerful.

4 Negative Weights Adversary Method

Exactly the same as the Weighted Adversary method, but allow the weights $\Gamma[y, z]$ to take on negative values. See [HLS07]
Let $ADV^{\pm}(F)$ be the best lower bound achievable using the Negative Weights Adversary Method.
Fun Fact: Given the truth table of F, $ADV^{\pm}(F)$ is computable in polynomial time in the size of the truth table of F using semi-definite programming.

5 Reichardt's Theorem

Theorem 5.1. *Reichardt's Theorem: Quantum Query complexity of F is $ADV^{\pm}(F)$, so the Negative Weights Adversary method gives an upper and lower bound for the quantum query complexity fo F. [Rei09]*

Proof. (Not really. All the details are in next lecture.) $ADV^{\pm}(F)$ is a maximization problem, since it is trying to find the weight matrix that gives the highest lower bound. This can be seen as an SDP (semi-definite program). This allows it to be transformed into the dual of the SDP, which is a minimization problem. See: http://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15859-f11/www/notes/lecture12.pdf for notes on SDPs.
Reichardt observed that finding the dual of the SDP is interpretable as finding the lowest complexity Span Program that computes F. This is the easy part. Then Richardt showed that any Span Program for F of complexity T can be converted into a quantum query algorithm for F using $O(T)$ queries. In other words, he showed that a quantum algorithm could evaluate the span program in $\emptyset(T)$ queries (actually it uses $O(T)log(T)$, so it's tight within a logarithmic factor -[Rei09]). This is the hard part of the proof. □

6 Span Programs

Span programs are a linear-algebraic model of computation from before quantum computation was a real topic though you need a different definition for their complexity for quantum computation than for classical.
As Reichardt showed, we can design quantum algorithms by designing span programs, and without loss of generality, these can be optimal.
In these notes, span programs are defined for decision functions $F : \{0, 1\}^N \to \{0, 1\}$, but they can be more generally defined [KW93].
Definition: A Span Program P operating on strings $w \in \{0, 1\}^N$ consists of a bunch of real

input vectors in $\mathbb{R}^d$ grouped into $2N$ sets, named $w_1 = 0, w_1 = 1, w_2 = 0, w_2 = 1, \ldots, w_N = 0, w_N = 1$, plus a target vector $\tau \in \mathbb{R}^d$.
How P computes:
First it partitions the $2N$ sets of vectors into 2 new sets, $Avail$, and $Unavail$, each of size N based on w: For each $i = 0, 1, \ldots N$, if $w_i = 0$, the set named $w_i = 0$ goes into $Avail$ and the set $w_i = 1$ goes into $Unavail$, and if $w_i = 1$, $w_i = 1$ goes into $Avail$ and $w_i = 0$ into $Unavail$. Then $P(w) = 1$ if τ is in the span of the vectors in $Avail$. We say P computes F if $P(w) = F(w)$ on all inputs.
Example: Majority for 3 bits:
$d = 3$
P:
$w_1 = 0 : \emptyset$
$w_1 = 1 : \begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}$
$w_2 = 0 : \emptyset$
$w_2 = 1 : \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}$
$w_3 = 0 : \emptyset$
$w_3 = 1 : \begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}$
$T = \begin{pmatrix}1\\1\\1\end{pmatrix}$
Since any two of the $w_i = 1$ sets form a basis for $\mathbb{R}^3$, but none of them individually contain T in their span, $P(w)$ will compute majority.
Example 2: OR (Search)
$d = 1$
P:
$w_i = 0 : \emptyset \forall i$
$w_i = 1 : \big(1\big) \forall i$
$\tau = \big(() \, 1\big)$

7 Span Program Complexity for Quantum Algorithms

Given a span program P, let I be the set of its input vectors. Then let V be the matrix formed by all the vectors in I:

$$V = \left(\begin{pmatrix} V_{11} \\ \vdots \\ V_{1d} \end{pmatrix} \cdots \begin{pmatrix} V_{|I|1} \\ \vdots \\ V_{|I|d} \end{pmatrix} \right)$$

I is partitioned into $Avail(w)$ and $Unavail(w)$ as described before.
Suppose P computes a function F.
Definition: A *Positive Witness* for some $y \in F^{-1}(1)$ is a linear combination of vectors in $Avail(y)$ that make τ. We can write this as a column vector $\alpha \in \mathbb{R}^{|I|}$ such that $V\alpha = \tau, \alpha_i = 0 \forall i \in Unavail(z)$.
Definition: A *Negative Witness* for some $z \in F^{-1}(0)$ is a row vector $\beta \in \mathbb{R}^d$ such that $\langle \beta, V_i \rangle = 0 \forall i \in Avail(z)$ and $\langle \beta, \tau \rangle = 1$.
The negative witness works because it forces β to be in the nullspace of the available vectors, so no linear combination of them can ever form β, but since $\langle \beta, \tau \rangle = 1$ those vectors would need to form β to have a linear combination of them form τ. Forcing $\beta = 1$ is important for the complexity definition, since otherwise we could make the Positive Witness size arbitrarily small by multiplying τ by a small constant. Forcing this requirement on β ensures that the complexity is independent of constant multiplication of τ.

Definition 7.1. An *Extended Span Program ESP* is some span program P and a witness for every input, so:
$ESP = (P, (\alpha_y)_{y \in F^{-1}(1)}, (\beta_z)_{z \in F^{-1}(0)})$

Definition 7.2. The size of a positive witness α is: $size(\alpha) = |\alpha|^2$

Definition 7.3. The size of a negative witness β is $size(\beta) = \sum_{i=1}^{|I|} |\langle B, V_i \rangle|^2$.

Let $T_1 = max\{size(\alpha_y)\}, y \in F^{-1}(1)$
Let $T_0 = max\{size(\beta_z)\}, z \in F^{-1}(0)$

Definition 7.4. The complexity T of P is $\sqrt{T_0 T_1}$.

Theorem 7.5. *Given an extended span program $(P, (\alpha_y)_{y \in F^{-1}(1)}, (\beta_z)_{z \in F^{-1}(0)})$ for a function F, there is a quantum algorithm for F making $O(T)$ queries.*
Also, the minimum T over all span programs computing F = "dual SDP" = $ADV^{\pm}(F)$, so the best possible span program gives the best possible quantum algorithm. [Rei09]

Example: Majority for 3 bits:

$$V = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

For $w = 111$, the best α is $\begin{pmatrix} 0.5 \\ \vdots \\ 0.5 \end{pmatrix}$, and its size is $6 * 0.5^2 = 1.5$.

For $w = 110$, the last two columns of V are unavailable, so $\alpha = \begin{pmatrix} 1 \\ 0.5 \\ 0.5 \\ 1 \\ 0 \\ 0 \end{pmatrix}$ gives the best value, with $size(\alpha) = 2.5$, and $w = 101$ or 011 gives the same value by permuting the rows in α. Since these are all the positive witnesses, $T_1 = 2.5$.

For $w = 100$, we need a negative witness. Since there are 2 of the 3 standard basis vectors of $\mathbb{R}^3$ in $Avail(w)$, the only choice is the third basis vector, $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$, which satisfies $\langle \beta, \tau \rangle = 1$, and has size 2, and the same bound holds for the other strings with Hamming weight 1.

For $w = 000$, any β with $\langle \beta, \tau \rangle$ works, and several choices give a size of 2 (or example, all values $1/\sqrt{3}$, or the β from $w = 100$) and since this matches the best bound bound from the other negative witness, $T_0 = 2$.

Then $T = \sqrt{T_1 T_0} = sqrt5$.

Example: OR function (unstructured search)

For any input string y with $F(y) = 1$, $\alpha_y = \begin{pmatrix} 0 \vdots 1 \vdots 0 \end{pmatrix}$, with the 1 on any coordinate with $y_i = 1$, so $size(\alpha) = 1$, and $T_1 = 1$.

The only possible negative witness is $\beta = 1$, and $\langle \beta, V_i \rangle = 1$ for N vectors in V, since V has N 1 vectors and nothing else, so $T_0 = N$.

Then the quantum query complexity for unstructured search is (SURPRISE!) $\sqrt{N}$.

References

[HLS07] P. Hoyer, T. Lee, and R Spalek. Negative weights make adversaries stronger. *Proceedings of the 39th ACM Symposium on the Theory of Computing*, pages 526–535, November 2007.

[KW93] M. Karchmer and A. Wigderson. On span programs. *Proceedings of the 8th IEEE Structure in Complexity Theory*, pages 102–111, 1993.

[Rei09] Ben W. Reichardt. Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function. *Foundations of Computer Science, 2009*, pages 544–551, October 2009.

[SS06] R. Spalek and M. Szegedy. All quantum adversary methods are equivalent. *Theory of Computing, Volume 2 (2006)*, pages 1–18, January 2006.