Exact quantum query complexity of symmetric boolean functions

Andris Ambainis, **Jānis Iraids**

May 14, 2014

Query complexity

- $D(f)$ - deterministic query complexity

Query complexity

- $D(f)$ - deterministic query complexity
- $Q(f)$ - two-sided bounded error quantum query complexity

Query complexity

- $D(f)$ - deterministic query complexity
- $Q(f)$ - two-sided bounded error quantum query complexity
- $Q_E(f)$ - exact quantum query complexity (error probability is 0)

Exact quantum query complexity [1/2]

- Largest known gap between $Q(f)$ and $D(f)$:
  $Q(f) = O(\sqrt{D(f)})$, for example, Grover's algorithm

Exact quantum query complexity [1/2]

- Largest known gap between $Q(f)$ and $D(f)$: $Q(f) = O(\sqrt{D(f)})$, for example, Grover's algorithm
- In general there are many known instances of $Q(f) \ll D(f)$

Exact quantum query complexity [1/2]

- Largest known gap between $Q(f)$ and $D(f)$: $Q(f) = O(\sqrt{D(f)})$, for example, Grover's algorithm
- In general there are many known instances of $Q(f) \ll D(f)$
  - Formula evaluation

Exact quantum query complexity [1/2]

- Largest known gap between $Q(f)$ and $D(f)$: $Q(f) = O(\sqrt{D(f)})$, for example, Grover's algorithm
- In general there are many known instances of $Q(f) \ll D(f)$
  - Formula evaluation
  - Deciding of some graph properties

Exact quantum query complexity [1/2]

- Largest known gap between $Q(f)$ and $D(f)$: $Q(f) = O(\sqrt{D(f)})$, for example, Grover's algorithm
- In general there are many known instances of $Q(f) \ll D(f)$
  - Formula evaluation
  - Deciding of some graph properties
  - See Quantum Algorithm Zoo for more

Exact quantum query complexity [1/2]

- Largest known gap between $Q(f)$ and $D(f)$: $Q(f) = O(\sqrt{D(f)})$, for example, Grover's algorithm
- In general there are many known instances of $Q(f) \ll D(f)$
  - Formula evaluation
  - Deciding of some graph properties
  - See Quantum Algorithm Zoo for more
- The only known instance of $Q_E(f) \ll D(f)$ is the iterated NE_3 (not-equal) function where $Q_E(f) = O(D(f)^{.8675})$ [Ambainis, 2012]

Exact quantum query complexity [1/2]

- Largest known gap between $Q(f)$ and $D(f)$: $Q(f) = O(\sqrt{D(f)})$, for example, Grover's algorithm
- In general there are many known instances of $Q(f) \ll D(f)$
  - Formula evaluation
  - Deciding of some graph properties
  - See Quantum Algorithm Zoo for more
- The only known instance of $Q_E(f) \ll D(f)$ is the iterated NE_3 (not-equal) function where $Q_E(f) = O(D(f)^{.8675})$ [Ambainis, 2012]
- For total functions $D(f) = O(Q_E(f)^3)$ [Midrijānis, 2004]

Exact quantum query complexity [1/2]

- Largest known gap between $Q(f)$ and $D(f)$: $Q(f) = O(\sqrt{D(f)})$, for example, Grover's algorithm
- In general there are many known instances of $Q(f) \ll D(f)$
  - Formula evaluation
  - Deciding of some graph properties
  - See Quantum Algorithm Zoo for more
- The only known instance of $Q_E(f) \ll D(f)$ is the iterated NE_3 (not-equal) function where $Q_E(f) = O(D(f)^{.8675})$ [Ambainis, 2012]
- For total functions $D(f) = O(Q_E(f)^3)$ [Midrijānis, 2004]
- The corresponding result for bounded error setting: $D(f) = O(Q(f)^6)$ [Beals et al., 1998]

Exact quantum query complexity [2/2]

- $Q_E(f) \geq \frac{\deg f}{2}$ [Beals et al., 1998]

Exact quantum query complexity [2/2]

- $Q_E(f) \geq \frac{\deg f}{2}$ [Beals et al., 1998]
- For partial functions $Q_E(f)$ and $D(f)$ can be $\Theta(1)$ vs $\Theta(n)$ respectively [Deutsch, Jozsa, 1992]

Exact quantum query complexity [2/2]

- $Q_E(f) \geq \frac{\deg f}{2}$ [Beals et al., 1998]
- For partial functions $Q_E(f)$ and $D(f)$ can be $\Theta(1)$ vs $\Theta(n)$ respectively [Deutsch, Jozsa, 1992]
- Any function with $Q_E(f) < \frac{D(f)}{2}$ implies an asymptotic gap between $Q_E(f)$ and $D(f)$

Boolean functions

- Total boolean function f:

$$f : \{0,1\}^n \to \{0,1\}$$

Boolean functions

- Total boolean function f:

$$f : \{0,1\}^n \to \{0,1\}$$

- Function is called symmetric, if $f(x)$ only depends on the Hamming weight of x

Boolean functions

- Total boolean function f:

$$f : \{0,1\}^n \to \{0,1\}$$

- Function is called symmetric, if $f(x)$ only depends on the Hamming weight of x
- In general, we specify a symmetric function by a vector of $n+1$ binary entries: $SYM(f_0, f_1, \ldots, f_n)$

Boolean functions

- Total boolean function f:

$$f : \{0,1\}^n \to \{0,1\}$$

- Function is called symmetric, if $f(x)$ only depends on the Hamming weight of x
- In general, we specify a symmetric function by a vector of $n+1$ binary entries: $SYM(f_0, f_1, \ldots, f_n)$
- $\deg f$ – the polynomial degree of f is the smallest degree for a polynomial equal to f on $\{0,1\}^n$

Symmetric functions

- For non-constant symmetric functions $D(f) = n$

Symmetric functions

- For non-constant symmetric functions $D(f) = n$
- There exist symmetric functions with $Q_E(f) = \frac{D(f)}{2}$, for example, $PARITY_n$

Symmetric functions

- For non-constant symmetric functions $D(f) = n$
- There exist symmetric functions with $Q_E(f) = \frac{D(f)}{2}$, for example, $PARITY_n$
- In particular, we can calculate XOR of two variables exactly using 1 query

Symmetric functions

- For non-constant symmetric functions $D(f) = n$
- There exist symmetric functions with $Q_E(f) = \frac{D(f)}{2}$, for example, $PARITY_n$
- In particular, we can calculate XOR of two variables exactly using 1 query
- There exist symmetric functions with $Q_E(f) = \frac{D(f)}{2}$, that cannot be computed optimally with probabilistic XOR trees [Montanaro et al., 2011, Ambainis et al., 2013]

Symmetric functions

- For non-constant symmetric functions $D(f) = n$
- There exist symmetric functions with $Q_E(f) = \frac{D(f)}{2}$, for example, $PARITY_n$
- In particular, we can calculate XOR of two variables exactly using 1 query
- There exist symmetric functions with $Q_E(f) = \frac{D(f)}{2}$, that cannot be computed optimally with probabilistic XOR trees [Montanaro et al., 2011, Ambainis et al., 2013]
- For symmetric functions $\deg f \geq n - O(n^{.548})$ [von zur Gathen, Roche, 1993]

Symmetric functions

- For non-constant symmetric functions $D(f) = n$
- There exist symmetric functions with $Q_E(f) = \frac{D(f)}{2}$, for example, $PARITY_n$
- In particular, we can calculate XOR of two variables exactly using 1 query
- There exist symmetric functions with $Q_E(f) = \frac{D(f)}{2}$, that cannot be computed optimally with probabilistic XOR trees [Montanaro et al., 2011, Ambainis et al., 2013]
- For symmetric functions $\deg f \geq n - O(n^{.548})$ [von zur Gathen, Roche, 1993]
- It is conjectured that for symmetric functions $\deg f \geq n - O(1)$

Symmetric functions with $Q_E(f) = \frac{D(f)}{2}$

- The first example is $EXACT_k^n(x) = \begin{cases} 1 \text{ if } |x| = k \\ 0, \text{ otherwise} \end{cases}$

$$Q_E(EXACT_k^n) = \max\{k, n-k\}$$

Symmetric functions with $Q_E(f) = \frac{D(f)}{2}$

- The first example is $EXACT_k^n(x) = \begin{cases} 1 \text{ if } |x| = k \\ 0, \text{ otherwise} \end{cases}$

$$Q_E(EXACT_k^n) = \max\{k, n-k\}$$

- The second example is $TH_k^n(x) = \begin{cases} 1 \text{ if } |x| \geq k \\ 0, \text{ otherwise} \end{cases}$

$$Q_E(TH_k^n) = \max\{k, n-k+1\}$$

Symmetric functions with $Q_E(f) = \frac{D(f)}{2}$

- The first example is $EXACT_k^n(x) = \begin{cases} 1 \text{ if } |x| = k \\ 0, \text{ otherwise} \end{cases}$

$$Q_E(EXACT_k^n) = \max\{k, n-k\}$$

- The second example is $TH_k^n(x) = \begin{cases} 1 \text{ if } |x| \geq k \\ 0, \text{ otherwise} \end{cases}$

$$Q_E(TH_k^n) = \max\{k, n-k+1\}$$

- For these functions $Q_E(f) \to \frac{n}{2}$ when $k \to \frac{n}{2}$

Symmetric functions with $Q_E(f) = \frac{D(f)}{2}$

- The first example is $EXACT_k^n(x) = \begin{cases} 1 \text{ if } |x| = k \\ 0, \text{ otherwise} \end{cases}$

$$Q_E(EXACT_k^n) = \max\{k, n-k\}$$

- The second example is $TH_k^n(x) = \begin{cases} 1 \text{ if } |x| \geq k \\ 0, \text{ otherwise} \end{cases}$

$$Q_E(TH_k^n) = \max\{k, n-k+1\}$$

- For these functions $Q_E(f) \to \frac{n}{2}$ when $k \to \frac{n}{2}$
- For these functions an optimal algorithm using only the *XOR* "trick" does not exist

Semi-definite programming

- Quantum query complexity admits a SDP characterization [Barnum et al., 2003]

Semi-definite programming

- Quantum query complexity admits a SDP characterization [Barnum et al., 2003]
- We can search for functions with small $Q_E(f)$

Semi-definite programming

- Quantum query complexity admits a SDP characterization [Barnum et al., 2003]
- We can search for functions with small $Q_E(f)$
- Symmetric functions of 6 bits [Montanaro et al., 2011]

Function	1 query	2 queries	3 queries	4 queries	5 queries
(0,0,0,0,0,0,1)	0.663	0.900	0.980	0.997	0.9999
(0,0,0,0,0,1,0)	0.511	0.684	0.940	0.993	1
(0,0,0,0,0,1,1)	0.640	0.738	0.946	0.993	1
...	...	...	...	...	...
(0,0,0,1,0,0,0)	0.530	0.616	1	1	1
...	...	...	...	...	...
(0,0,1,0,1,0,0)	0.500	0.517	1	1	1
...	...	...	...	...	...
(0,1,0,1,0,1,0)	0.500	0.500	1	1	1
...	...	...	...	...	...
(0,1,1,1,1,1,0)	0.693	0.925	0.988	0.999	1

Summary of results

- Let $EXACT^n_{k,l}$ $(k < l)$ be a symmetric function, such that

$$EXACT^n_{k,l}(x) = \begin{cases} 1, & \text{if } |x| = k \text{ or } |x| = l \\ 0, & \text{otherwise} \end{cases}$$

Theorem

$$Q_E(EXACT^n_{k,l}) \leq \begin{cases} \max\{l, n-k\} \text{ when } d = 1 \\ \max\{l, n-k\} - 1 \text{ when } d = 2 \end{cases}$$

Conjecture:

$$Q_E(EXACT^n_{k,l}) \leq \max\{l, n-k\} - 1 \text{ when } d > 2$$

Summary of results

- Let $EXACT^n_{k,l}$ $(k < l)$ be a symmetric function, such that

$$EXACT^n_{k,l}(x) = \begin{cases} 1, & \text{if } |x| = k \text{ or } |x| = l \\ 0, & \text{otherwise} \end{cases}$$

- Let $d = l - k$ and $EXACT_{k,n-k} = EXACT^n_{k,n-k}$

Theorem

$$Q_E(EXACT^n_{k,l}) \leq \begin{cases} \max\{l, n-k\} \text{ when } d = 1 \\ \max\{l, n-k\} - 1 \text{ when } d = 2 \end{cases}$$

Conjecture:

$$Q_E(EXACT^n_{k,l}) \leq \max\{l, n-k\} - 1 \text{ when } d > 2$$

Computing $EXACT_{k,n-k}$

- The algorithm is recursive

Computing $EXACT_{k,n-k}$

- The algorithm is recursive
- Proof – by induction

Computing $EXACT_{k,n-k}$

- The algorithm is recursive
- Proof – by induction
- In what follows $x_i \in \{\pm 1\}$

Computing $EXACT_{k,n-k}$

- The algorithm is recursive
- Proof – by induction
- In what follows $x_i \in \{\pm 1\}$
- Consider a non-normalized state

$$|\psi_{k,n}\rangle = \sum_{i=1}^{n} x_i |0\rangle + \sqrt{r_{k,n}} \sum_{\substack{i=1 \\ j=i+1}}^{n} (x_i - x_j) |i,j\rangle$$

Computing $EXACT_{k,n-k}$

Lemma

If $EXACT^{n-2}_{k-1,n-k-1}$ is computable with $t-1$ queries starting in a non-normalized quantum state $|\psi_{k-1,n-2}\rangle$ then $EXACT^{n}_{k,n-k}$ is computable with t queries starting in state $|\psi_{k,n}\rangle$ where

$$r_{k,n} = \frac{\frac{n^2(n-2)^2}{\frac{1}{r_{k-1,n-2}}-1} + d^2}{(n^2-d^2)^2}.$$

A sketch of proof

A sketch of proof.

We will use two kinds of unitary transformations and their inverses:

$$U_\alpha(|\psi\rangle) = \sqrt{\alpha}\,|\psi^*\rangle + \sqrt{1-\alpha}\,|\psi^{**}\rangle$$

$$V_n(\sum_{i=1}^{n}\alpha_i\,|i\rangle) = \frac{\sum_{i=1}^{n}\alpha_i}{n}\,|0\rangle + \frac{\sum_{i\neq j}(\alpha_i-\alpha_j)\,|i,j\rangle}{n}$$

Using U_α and V_n and their inverses from $|\psi_{k,n}\rangle$ we construct

$$\left[(\sum_{i=1}^{n}x_i)^2 - d^2\right]|0\rangle + \sum_{i\neq j}(x_i-x_j)\,|\psi_{k-1,n-2}\rangle$$

A sketch of proof

Proof.

The state

$$\left[(\sum_{i=1}^{n} x_i)^2 - d^2\right] |0\rangle$$

has non-zero amplitude if and only if $EXACT_{k,n-k} = 0$. In this case we are done.

The other part

$$\sum_{i \neq j} (x_i - x_j) \, |\psi_{k-1,n-2}\rangle$$

has non-zero amplitude if $x_i \neq x_j$. Therefore we have reduced computing of $EXACT_{k,n-k}$ to $EXACT_{k-1,n-k-1}$. □

Creating $|\psi_{k,n}\rangle$

- We can prepare the state $|\psi_{k,n}\rangle + \sqrt{1 - r_{k,n}} \sum_{i \neq j} (x_i - x_j) |i', j'\rangle$ using 1 query and transformations U_α and V_n.

Creating $|\psi_{k,n}\rangle$

- We can prepare the state $|\psi_{k,n}\rangle + \sqrt{1-r_{k,n}} \sum_{i \neq j} (x_i - x_j) |i', j'\rangle$ using 1 query and transformations U_α and V_n.
- Measuring $\sqrt{1-r_{k,n}} \sum_{i \neq j} (x_i - x_j) |i', j'\rangle$ gives us $x_i \neq x_j$ and reduces the task to computing $EXACT_{k-1,n-k+1}$

Base cases

- When $d = 1$: $EXACT^1_{0,1}$ can be computed with $r_{0,1} = 0$ using 0 queries

Base cases

- When $d = 1$: $EXACT^1_{0,1}$ can be computed with $r_{0,1} = 0$ using 0 queries
- When $d = 2$: $EXACT^2_{0,2}$ can be computed with $r_{0,2} = 0$ using 0 queries (i.e., we can compute XOR starting in state $(x_1 + x_2)\,|0\rangle$)

Base cases

- When $d = 1$: $EXACT^1_{0,1}$ can be computed with $r_{0,1} = 0$ using 0 queries
- When $d = 2$: $EXACT^2_{0,2}$ can be computed with $r_{0,2} = 0$ using 0 queries (i.e., we can compute XOR starting in state $(x_1 + x_2)\,|0\rangle$)
- When $d = 3$: numerical experiments indicate that there exists a sufficiently good base case for $EXACT^5_{1,4}$

The goal

Theorem

$$Q_E(EXACT^n_{k,l}) \geq \max\{l, n-k\} - 1$$

Theorem

$$Q_E(EXACT^n_{k,l}) \geq \max\{l, n-k\} = \frac{n+1}{2} \text{ if } l = k+1 = \frac{n+1}{2}$$

Proof of $Q_E(AND_n) \geq n$

- (Polynomial method)

Proof of $Q_E(AND_n) \geq n$

- (Polynomial method)
- The amplitudes are polynomials in $x_1, \ldots, x_n$:

$$p_1(x_1, \ldots, x_n), \ldots, p_l(x_1, \ldots, x_n)$$

Proof of $Q_E(AND_n) \geq n$

- (Polynomial method)
- The amplitudes are polynomials in $x_1, \ldots, x_n$:

$$p_1(x_1, \ldots, x_n), \ldots, p_l(x_1, \ldots, x_n)$$

- If the algorithm performs t queries

$$\forall i : \deg(p_i) \leq t$$

Proof of $Q_E(AND_n) \geq n$

- (Polynomial method)
- The amplitudes are polynomials in $x_1, \ldots, x_n$:

$$p_1(x_1, \ldots, x_n), \ldots, p_l(x_1, \ldots, x_n)$$

- If the algorithm performs t queries

$$\forall i : \deg(p_i) \leq t$$

- There exists i, such that

$$p_i(x) = \begin{cases} a \neq 0, \text{ if } x = (1, 1, \ldots, 1), \\ 0, \text{ otherwise} \end{cases}$$

Proof of $Q_E(AND_n) \geq n$

- (Polynomial method)
- The amplitudes are polynomials in $x_1, \ldots, x_n$:

$$p_1(x_1, \ldots, x_n), \ldots, p_l(x_1, \ldots, x_n)$$

- If the algorithm performs t queries

$$\forall i : \deg(p_i) \leq t$$

- There exists i, such that

$$p_i(x) = \begin{cases} a \neq 0, \text{ if } x = (1, 1, \ldots, 1), \\ 0, \text{ otherwise} \end{cases}$$

- Symmetrize p_i: p_{sym}

Proof of $Q_E(AND_n) \geq n$ [cont.]

- Convert p_{sym} to univariate polynomial $p(s)$ with

$$s = x_1 + x_2 + \ldots + x_n$$

Proof of $Q_E(AND_n) \geq n$ [cont.]

- Convert p_{sym} to univariate polynomial $p(s)$ with

$$s = x_1 + x_2 + \ldots + x_n$$

- p has n zeros at $s = 0, 1, \ldots, n-1$

Proof of $Q_E(AND_n) \geq n$ [cont.]

- Convert p_{sym} to univariate polynomial $p(s)$ with

$$s = x_1 + x_2 + \ldots + x_n$$

- p has n zeros at $s = 0, 1, \ldots, n-1$
- $n \leq \deg(p) \leq t$

Proof of $Q_E(AND_n) \geq n$ [cont.]

- Convert p_{sym} to univariate polynomial $p(s)$ with

$$s = x_1 + x_2 + \ldots + x_n$$

- p has n zeros at $s = 0, 1, \ldots, n-1$
- $n \leq \deg(p) \leq t$
- QED

The case when $d > 1$

Lemma

If f is a symmetric function, such that $f(n) = 1$ and $|\{s|f(s) = 0, s \in \{0, 1, \ldots, n-1\}\}| = k$, then

$$Q_E(f) \geq k$$

Proof.

Almost the same as for AND_n. □

Theorem

$$Q_E(EXACT^n_{k,l}) \geq \max\{l, n-k\} - 1$$

- When n is odd and $l = k + 1 = \frac{n+1}{2}$, however, this lower bound is lower that $\frac{n}{2}$!

The case when $d = 1$

Our result:

Theorem

$$Q_E(EXACT^n_{k,l}) \geq \frac{n+1}{2}, \text{ if } l = k+1 = \frac{n+1}{2}$$

The case when $d = 1$

Lemma

If f is computed by a t query quantum algorithm, then there exist univariate polynomials that represent f as:

$$\begin{aligned} & f(s) = \\ =& p_t^2(s) + \\ & + s(n-s)p_{t-1}^2(s) + \ldots + \\ & + s(n-s)(s-1)(n-1-s)\ldots(s-t+1)(n-t+1-s)p_0^2(s) \end{aligned}$$

where p_i are polynomials with $\deg p_i \leq i$.

Proof.

Omitted. □

Applying the lemma [1/3]

Theorem

$$Q_E(EXACT^n_{k,l}) \geq \frac{n+1}{2}, \text{ if } l = k+1 = \frac{n+1}{2}$$

Proof.

- Take negation of $EXACT^n_{k,l}$

Applying the lemma [1/3]

Theorem

$$Q_E(EXACT^n_{k,l}) \geq \frac{n+1}{2}, \text{ if } l = k+1 = \frac{n+1}{2}$$

Proof.

- Take negation of $EXACT^n_{k,l}$
- Assume $\exists t = \frac{n-1}{2}$ query quantum algorithm computing it

Applying the lemma [1/3]

Theorem

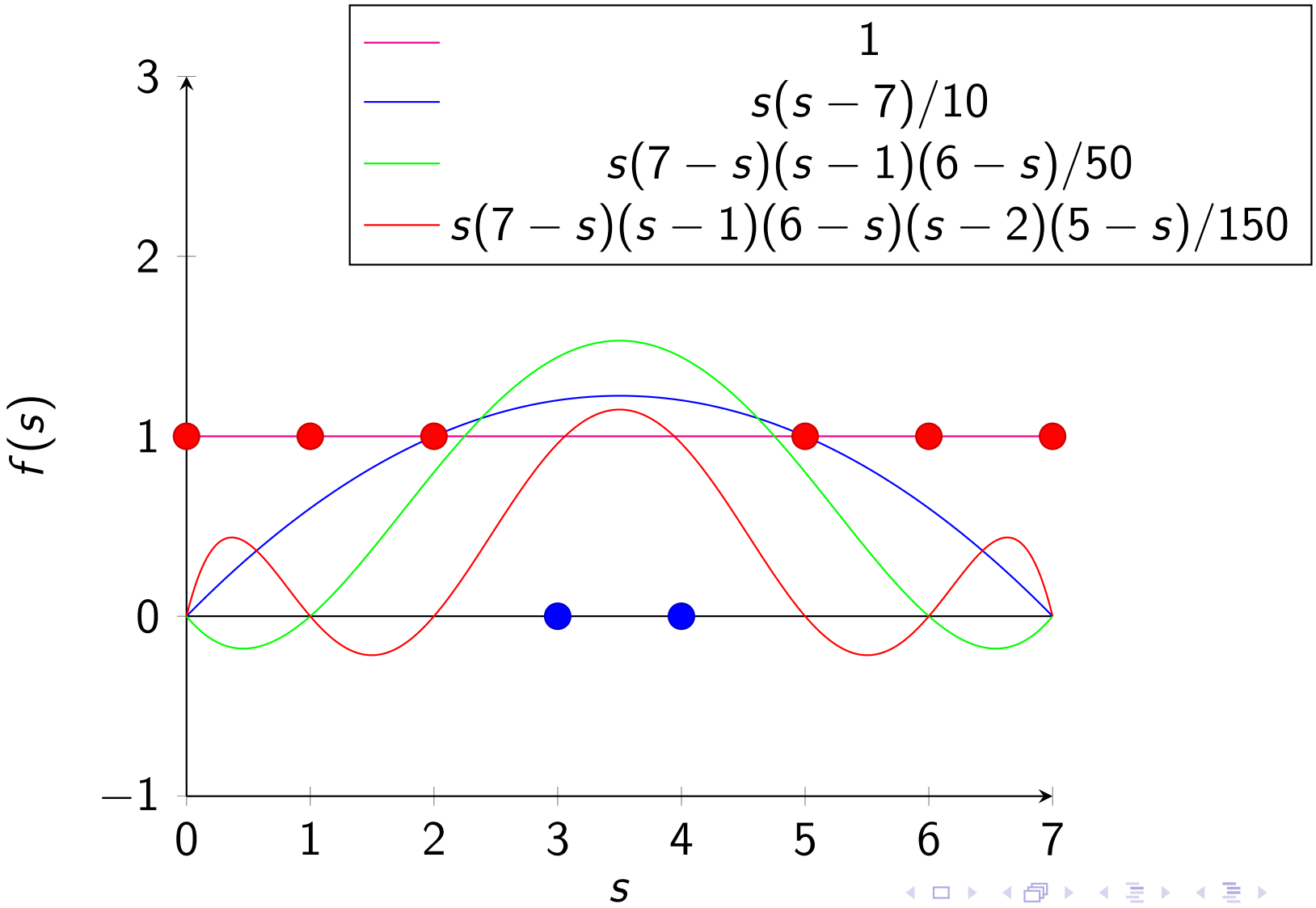
$$Q_E(EXACT^n_{k,l}) \geq \frac{n+1}{2}, \text{ if } l = k+1 = \frac{n+1}{2}$$

Proof.

- Take negation of $EXACT^n_{k,l}$
- Assume $\exists t = \frac{n-1}{2}$ query quantum algorithm computing it
- By lemma, there exists a representing polynomial of the form

$$\begin{aligned} & f(s) = \\ =& p_t^2(s) + \\ & + s(n-s)p_{t-1}^2(s) + \ldots + \\ & + s(n-s)(s-1)(n-1-s)\ldots(s-t+1)(n-t+1-s)p_0^2(s) \end{aligned}$$

Applying the lemma [2/3]



Applying the lemma [3/3]

Proof.

- $f(s)$ has $n-3$ extrema outside of $(k-1, l+1)$: between $(0,1)$, $(1,2)$, $\ldots$, $(k-2, k-1)$, $(l+1, l+2)$, $(l+2, l+3)$, $\ldots$, $(n-1, n)$

Applying the lemma [3/3]

Proof.

- $f(s)$ has $n-3$ extrema outside of $(k-1, l+1)$: between $(0,1)$, $(1,2)$, $\ldots$, $(k-2, k-1)$, $(l+1, l+2)$, $(l+2, l+3)$, $\ldots$, $(n-1, n)$
- $f(s)$ is non-negative in $(k-1, l+1)$ - it has three more extrema in $(k-1, l+1)$

Applying the lemma [3/3]

Proof.

- $f(s)$ has $n-3$ extrema outside of $(k-1, l+1)$: between $(0,1)$, $(1,2)$, $\ldots$, $(k-2, k-1)$, $(l+1, l+2)$, $(l+2, l+3)$, $\ldots$, $(n-1, n)$
- $f(s)$ is non-negative in $(k-1, l+1)$ - it has three more extrema in $(k-1, l+1)$
- $\deg f \geq (n-3)+3+1 = n+1$ - a contradiction!

$EXACT^n_{k,l}$

Theorem

$$Q_E(EXACT^n_{k,l}) \geq \max\{l, n-k\} - 1 \text{ if } d > 1$$

$$Q_E(EXACT^n_{k,l}) = \max\{l, n-k\} - 1 \text{ if } d = 2$$

$$Q_E(EXACT^n_{k,l}) = \max\{l, n-k\} \text{ if } d = 1 \text{ and } k = \frac{n-1}{2}$$

$$Q_E(EXACT^n_{k,l}) \leq \max\{l, n-k\} \text{ if } d = 1$$

Open problems

- What is the largest gap between $Q_E(f)$ and $D(f)$?

Open problems

- What is the largest gap between $Q_E(f)$ and $D(f)$?
- Are there symmetric functions with $Q_E(f) < \frac{n}{2}$?