Quantum Algorithms by Convex Optimization

Kıvanç Uyanık

IzTech,
KOBIT$|1\rangle$

February 3rd 2017

Outline

1. Introduction
2. Architecture
3. SDP - QQC Correspondence
   - Convex Optimization
   - Representations
   - Theorem
   - Numerical Results
4. Improvements, Applications and Results
   - Our contribution
   - Extensions and applications to known algorithms

Outline

1. Introduction
2. Architecture
3. SDP - QQC Correspondence
   - Convex Optimization
   - Representations
   - Theorem
   - Numerical Results
4. Improvements, Applications and Results
   - Our contribution
   - Extensions and applications to known algorithms

Introduction

- Feynman: exponential classical resources for the simulation of quantum systems.
- Deutsch: 1 quantum query vs. 2 classical queries for a very special problem
- Grover: $O(\sqrt{n})$ quantum queries vs. $O(n)$ classicals queries
- Simon: Exponential speedup as compared to nondeterministic classical algorithms
- Shor: Applicable to cryptography
- ...
- What's next?
- Can we automatize the process?

Outline

1. Introduction
2. Architecture
3. SDP - QQC Correspondence
   - Convex Optimization
   - Representations
   - Theorem
   - Numerical Results
4. Improvements, Applications and Results
   - Our contribution
   - Extensions and applications to known algorithms

Preliminaries

Bit strings and functions

- Bit string: $\boldsymbol{x}$: $\boldsymbol{x} = x_n x_{n-1} x_{n-2} \ldots x_1$, where $x_i \in \{0,1\}$, $1 \leq i \leq n$.
- Hamming weight: $|\boldsymbol{x}| \equiv \sum x_i$.
- Function $f : S \longrightarrow T$, where $S \subseteq \Sigma^n$, Σ and T are finite sets.
- f is partial when $S \subsetneq \Sigma^n$, total when $S = \Sigma^n$, Boolean when $\Sigma = \{0,1\}$.
- f is a decision function if $T = \{0,1\}$.

Assume all functions are Boolean.

Example

Deutsch - Jozsa algorithm: $T = \{0,1\}$, $S \subsetneq \{0,1\}^n$ is the set of constant and balanced bit strings.

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if } |\boldsymbol{x}| = \frac{n}{2}, \quad \text{(balanced case)} \\ 0 & \text{if } |\boldsymbol{x}| = 0 \text{ or } 1 \quad \text{(constant case).} \end{cases}$$

Preliminaries

Hilbert spaces, operators and matrices

- K a finite set, $\mathscr{H}_K$ Hilbert Space associated with K. Ortohonormal basis $\{|k\rangle\}$.
- A an operator on $\mathscr{H}$ if $A: \mathscr{H} \longrightarrow \mathscr{H}$.
- $A = A^{\dagger}$: Hermitian
- $\langle\psi|A\psi\rangle \geq 0 \ \ \forall|\psi\rangle \in \mathscr{H}$: Positive semidefinite
- $\langle A\psi|A\phi\rangle = \langle\psi|\phi\rangle \ \ \forall|\psi\rangle, |\phi\rangle \in \mathscr{H}$: Unitary
- $A^2 = A$: Projection
- $P_z : \sum P_i = \mathbb{1}$: Complete set of orthogonal projectors
- $M = \{m_{xy} : m_{xy} = \langle\psi_x|\psi_y\rangle\}$, Gram matrices. (Here$\{|\psi_i\rangle\}$ are an indexed family of vectors in $\mathscr{H}$. $M \geq 0$)

Registers

- input register: holds the input bit string $\boldsymbol{x} \in \{0,1\}^n$
- query register: holds an integer i such that $0 < i \leq n$.
- ancilla: acts as a working memory, no priory conditions.

State of the memory: $|\beta\rangle = \sum \beta_{x,i,w} |x,i,w\rangle$

It can be written as: $|\Psi\rangle = \sum_{x \in S} |x\rangle |\psi_x\rangle$ where $|\psi_x\rangle = \sum_i |i\rangle |\psi_{x,i}\rangle$

Operators

- Oracle

$$O|\mathbf{x}\rangle|i,w\rangle = (-1)^{x_i}|\mathbf{x}\rangle|i,w\rangle \tag{1}$$

  $i=0$, null query: No phase is introduced regardless of the input. Alternatively

$$O_{\mathbf{x}}|i,w\rangle = (-1)^{x_i}|i,w\rangle \tag{2}$$

  Note the difference between (2) and the conventional definition $O_f|x,w\rangle = (-1)^{f(x)}|x,w\rangle$.

- Intermediate unitaries $\left\{U^{(j)}\right\}$
- Orthogonal projection operators $\{P_z\}$, $\sum_z P_z = \mathbb{1}$

Quantum algorithm and query complexity

Input bit string is $\boldsymbol{x} = x_n x_{n-1} \ldots x_1$, corresponding oracle: $O_{\boldsymbol{x}}$, t queries

Algorithm:

1. Initialize the registers to $|0,0\rangle$

2. Apply the first unitary $U^{(0)}$

3. Alternatively apply O_x and $U^{(j)}$'s t times

4. Apply the projection operators $\{P_z\}$ (make a measurement) and output the result with an error ε.

Final state: $|\psi_{final}\rangle = U^{(t)} O_{\boldsymbol{x}} U^{(t-1)} O_{\boldsymbol{x}} \cdots U^{(1)} O_{\boldsymbol{x}} U^{(0)} |0,0\rangle$

Query complexity is t !

Outline

1 Introduction

2 Architecture

3 SDP - QQC Correspondence
- Convex Optimization
- Representations
- Theorem
- Numerical Results

4 Improvements, Applications and Results
- Our contribution
- Extensions and applications to known algorithms

Optimization

Definition

Optimization is the mathematical process of selecting the best element with regard to some criteria from the set of available alternatives[1].

It has the form:

$$\begin{array}{ll} \text{minimize} & f_0(\boldsymbol{x}) \\ \text{subject to} & f_i(\boldsymbol{x}) \leq b_i\,, \quad i = 1, \ldots, m. \end{array} \tag{3}$$

$\boldsymbol{x} = (x_1, \ldots, x_n)$: optimization variable,
$f_0 : \mathbb{R}^n \to \mathbb{R}$: objective function,
$f_i : \mathbb{R}^n \to \mathbb{R}$: constraint functions,
b_i: bounds.

Optimization

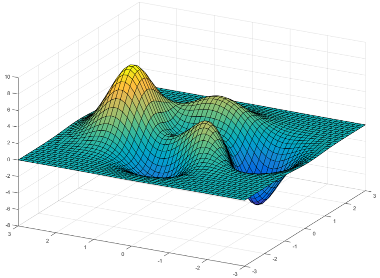

Figure: A surface with a few local optima: MATLAB peaks() function.

Linear and convex optimization

Definition

Optimization problem is called a linear program if the objective and constraint functions $f_0, \ldots f_m$ are linear,

$$f_i(\alpha \boldsymbol{x} + \beta \boldsymbol{y}) = \alpha f_i(\boldsymbol{x}) + \beta f_i(\boldsymbol{y}).$$

Definition

More generally, an optimization problem is called convex if the objective and constraint functions $f_0, \ldots f_m$ are convex,

$$f_i(\alpha \boldsymbol{x} + \beta \boldsymbol{y}) \leq \alpha f_i(\boldsymbol{x}) + \beta f_i(\boldsymbol{y}). \tag{4}$$

In convex optimization, optimal point is unique!

Semidefinite programming

A semidefinite program has the form:

$$\begin{aligned} \text{minimize} \quad & E * X \\ \text{subject to} \quad & \mathscr{A} * X = b \\ & X \succeq 0 \end{aligned} \tag{5}$$

E, X: symmetric matrices,
$\mathscr{A} : \mathbb{R}^{n \times n} \to \mathbb{R}^n$: linear operator,
b: vector,
$P * Q$: pairwise product of P and Q matrices.

Disciplined convex programming and CVX

Definition

Disciplined convex programming is a methodology for constructing convex optimization problems proposed by Michael Grant, Stephen Boyd, and Yinyu Ye[2].

DCP ruleset: a set of conventions or rules for converting a convex optimization problem to a numerically solvable form.

A convex problem can be rejected if it violates the ruleset!

Definition

CVX is a modeling system for constructing and solving disciplined convex programs on MATLAB.

A simple CVX example

Example

Least-squares problem with bounds

$$\begin{array}{ll} \text{minimize} & \|A\mathbf{x} - b\|_2 \\ \text{subject to} & l_i \leq x_i \leq u_i \end{array}$$

CVX code:

```
cvx_begin
    variable x(n)
    minimize( norm(A*x-b) )
    subject to
        l <= x <= u
cvx_end
```

Quantum query complexity and error

Let $f : S \to T$ and $\varepsilon \in \left[0, \frac{1}{2}\right)$.
Algorithm computes f within error $\varepsilon \leftrightarrows$ Probability of output $f(x)$ is at least $1 - \varepsilon$

$$\pi_x(f(x)) \geq 1 - \varepsilon$$

$\varepsilon = 0$: zero error case
Complexity of QA: number t of queries.

System: $QA(f, t, \varepsilon)$, partial Boolean function f, an integer t, a real number $\varepsilon \in \left[0, \frac{1}{2}\right)$
Question:

Is there a t-step $QA(f, t, \varepsilon)$ that computes f within error ε ?

A semidefinite program to represent QA

Semidefinite program: $SDP(f,t,\varepsilon)$, Find $S \times S$ real symmetric positive definite matrices $M^{(t)}$, $M_i^{(j)}$ and $\Gamma_z : z \in T$ satisfying [3]

$$\sum_{i=0}^{n} M_i^{(0)} = E_0 \quad (6)$$

$$\sum_{i=0}^{n} M_i^{(j)} = \sum_{i=0}^{n} E_i * M_i^{(j-1)} \quad \text{for } 1 \le j \le t \quad (7)$$

$$\sum_{z\in T} \Gamma_z = \sum_{i=0}^{n} E_i * M_i^{(t-1)} \quad (8)$$

$$\Delta_z * \Gamma_z = (1-\varepsilon) \quad (9)$$

where $E_i[x,y] = (-1)^{x_i+y_i}$, $\Delta_z = \mathrm{diag}\left(\delta_{f(x),z}\right)$, $F[x,y] = 1 - \delta_{f(x),f(y)}$

Correspondence theorem

Theorem

(Barnum, Saks and Szegedy [3]) Let $f : S \to T$ be a partial boolean function with domain $S \subseteq \{0,1\}^n$. Let t be a natural number and $\varepsilon \geq 0$. There is a t-step QA that computes f within error ε if and only if $SDP(f,t,\varepsilon)$ is feasible.

$$QA(f,t,\varepsilon) \Longleftrightarrow SDP(f,t,\varepsilon) \tag{10}$$

A recipe for quantum algorithms (Montanaro et al.[4])

1. Construct a SDP for the problem.
2. Write a CVX code to solve the SDP and run it.
3. Using the matrices $M^{(t)}$, $M_i^{(j)}$ and Γ_z , derive a sequence of intermediate states $\left|\psi_x^{(j)}\right\rangle$ of the quantum computer.
4. Using **Lemma 5** of [4] to generate all the intermediate unitary operators $U^{(j)}$ and the final projection operators P_z.

A numerical result

Function: $\text{EXACT}_2^4(\boldsymbol{x}) = \begin{cases} 1 & \text{if } |\boldsymbol{x}| = 2 \\ 0 & \text{otherwise} \end{cases}$

Design a 2-query quantum algorithm that evaluates EXACT_2^4 with zero error, (i.e. $t = 2, \varepsilon = 0$).

No ancilla, no output register. Only 5 dimensional input register. (Montanaro et al. [4])
Initial state: $|\psi\rangle = \frac{1}{2}\sum_{i=1}^{4}|i\rangle$,
Apply $O_x U O_x$,

$$U = \frac{1}{2}\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & \omega & \omega^2 \\ 1 & 1 & 0 & \omega^2 & \omega \\ 1 & \omega & \omega^2 & 0 & 1 \\ 1 & \omega^2 & \omega & 1 & 0 \end{pmatrix} \tag{11}$$

and $\omega = e^{2\pi i/3}$. Can we generalize it to $\text{EXACT}_{n/2}^n$? Partially...

Outline

1. Introduction
2. Architecture
3. SDP - QQC Correspondence
   - Convex Optimization
   - Representations
   - Theorem
   - Numerical Results
4. Improvements, Applications and Results
   - Our contribution
   - Extensions and applications to known algorithms

Error minimization: a CVX code for the problem

Task: minimize error ε(epss) for the function f, (2bits) with $t = 1$ query.

CVX code:

```
cvx_begin
    variable m*0's and g*'s symmetric, variable epss
    minimize( epss );
    subject to
        m00 + m10 + m20 == E0
        g0 + g1 == E0 .* m00 + E1 .* m10 + E2 .* m20;
        diag(g0) >= (1-epss)*(1-f);
        diag(g1) >= (1-epss)*f;
        m*0 == semidefinite(2^n); g* == semidefinite(2^n);
cvx_end
```

(*: 0,1,2 for m's and 0,1 for g's)

Generalizations

- Total vs partial functions
  $f : \{0,1\}^n \longrightarrow T$ becomes $f : S \longrightarrow T$, $S \subsetneq \{0,1\}^n$
  $E_i[x,y] = (-1)^{x_i + x_j}$ becomes $E_i[x(k), y(k)] = (-1)^{x(k)_i + x(k)_j}$, $k \in I$, I is an index set for S
- Boolean vs non-Boolean functions
  $f : S \longrightarrow \{0,1\}$ becomes $f : S \longrightarrow T$
  f vs $(1-f)$ becomes distinguishing all f_i, $i \in T$ from each other.

EXACT_2^4, EXACT_3^6 and $\text{EXACT}_{2,4}^6$

- EXACT_2^4: Trace minimization and angle manipulation leads to Montanaro's “inspired” result.
  $\max\left(\text{rank}\left(M_i^{(j)}\right)\right) = 2$ real dimensions $\longrightarrow$ 1 complex dimensions.
- EXACT_3^6 and $\text{EXACT}_{2,4}^6$
  $\max\left(\text{rank}\left(M_i^{(j)}\right)\right) = 6$ real dimensions $\longrightarrow \lceil \log_2 6 \rceil = 3$ qubit ancilla instead of 6.

Deutsch - Jozsa algorithm

Task:
Evaluate

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if } |\boldsymbol{x}| = \frac{n}{2}\,, \\ 0 & \text{if } |\boldsymbol{x}| = 0 \text{ or } 1\,. \end{cases}$$

using only $t = 1$ calls.

Code finds an algorithm with $t = 1$ for $n = 2$, $n = 4$ and $n = 6$.
$n = 8$ and beyond becomes too complex.

Grover's algorithm

Task:
Distinguish all $f_i(\boldsymbol{x}), 1 \leq i \leq m$ from each other. $\boldsymbol{x}$ is a bit string with only one 1 and the rest is 0.
Complexity of Grover's original algorithm: $\frac{\pi}{4}\sqrt{m}$.

m	Grover $\left\lceil \frac{\pi}{4}\sqrt{m} \right\rceil$	CVX
2-4	2	2
5-6	3	2
7-8	3	3
9-13	4	3
14-25	4	4

Table: Comparison of query complexities for the Grover's problem

Weight decision - I [6, 7, 8]

Task:

Let ρ_1 and ρ_2, $0 \leq \rho_1 < \rho_2 \leq 1$ be two weights. Evaluate

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if } |\boldsymbol{x}| = n\rho_1\,, \\ 0 & \text{if } |\boldsymbol{x}| = n\rho_2\,. \end{cases}$$

We found (some, not all) algorithms that distinguish weights for $n \leq 10$

Weight decision - II

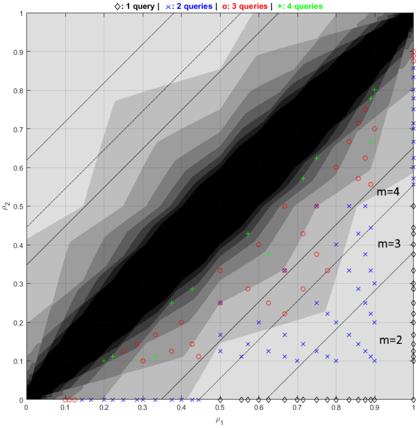

Figure: Comparison of the results by (Choi, Braunstein 2011), (Uyanık, Turgut 2013) and this work.

Pros and Cons

Pros

- QQC - SDP correspondence is easy to implement via CVX.
- Quick and exhaustive search.
- Many applications: Deutsch - Jozsa, Grover and Weight decision algorithms

Cons

- Only for a small number of qubits. Complexity of the convex optimization problem increases rapidly
- Mostly useful for existence proofs or inspiration
- It would have been very nice if we had a rank constraint

What to do next?

- Other applications, special problems
- Rank constraint, can we implement it with some other method?

[1] Boyd S., Vandenberghe L., Convex Optimization. Cambridge University Press. (2004)

[2] CVX Research web site, (2016), last accessed 2016.10.21, http://cvxr.com/

[3] Barnum, H., Saks, M., Szegedy, M.: Quantum query complexity and semi-definite programming. In: Proceedings of 18th Annual IEEE Conference on Computational Complexity, pp. 179–193 (2003)

[4] Montanaro, A.; Jozsa, R. & Mitchison, G. On Exact Quantum Query Complexity, Algorithmica, 71, 775-796, (2015).

[5] L.K. Grover, A fast quantum mechanical algorithm for database search, Proc. 28th Ann. ACM Symp. on the Theory of Computing, 1996, ACM Press, New York, pp. 212–219. Journal version, Quantum Mechanics helps in searching for a needle in a haystack, appeared in Physical Review Letters, 79 (1997) 325–328.

[6] Braunstein, S.L., Choi, B.-S., Ghosh, S., Maitra, S.: Exact quantum algorithm to distinguish Boolean functions of different weights. J. Phys. A: Math. Theor. 40, 8441–8454 (2007)

[7] Choi, B.S., Braunstein, S.L., Quantum Information Processing, Vol.10, p177-188 (2011).

[8] Uyanik, K., Turgut, S., Quantum Information Processing, Vol.12, p3395-3409 (2013).

[9] D. Deutsch, Proc. Roy. Soc. London A, **400**, 97–117 (1985).

[10] D. Deutsch and R. Jozsa, Proc. Roy. Soc. London A, **439**, 553–558 (1992).

[11] D. Simon, Proc. 35th Ann. Symp. on Foundations Comput. Sci., 116–123 (1994). Journal version appeared in SIAM J. Comput. **26**, 1474–1483 (1997) .

[12] P.W. Shor, Proc. 35th Ann. Symp. on Foundations of Comp. Sci., 124–134, (1994). Journal version appeared in SIAM J. Comput. **26**, 1484–1509 (1997).