Improvements in Quantum SDP-Solving

András Gilyén

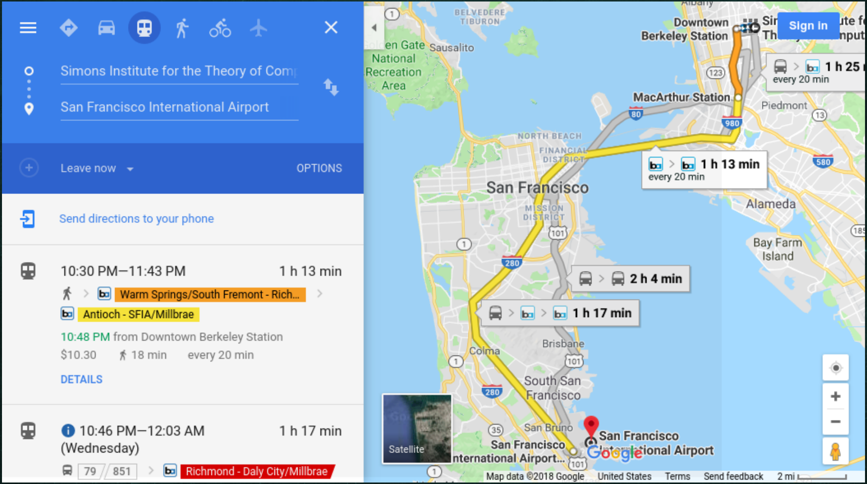
CWI / QuSoft / University of Amsterdam

Research Center for Quantum Software

June 13, 2018

Optimization is cool

Optimization is cool



Optimization is cool

One of the greatest successes in computer science

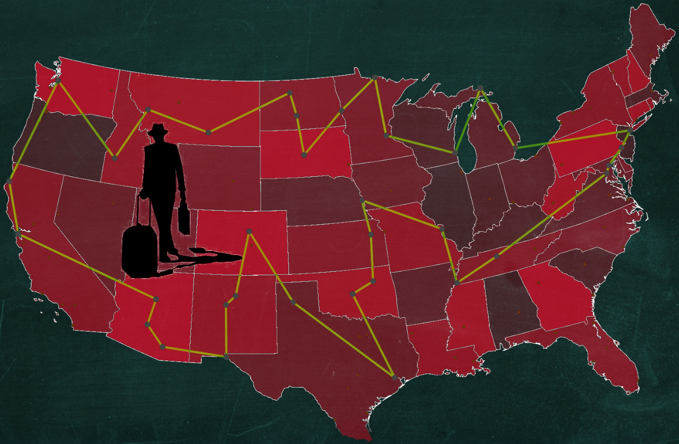
Optimization is cool

One of the greatest successes in computer science

Important practical applications in

- Route planning
- Scheduling
- Resource allocation
- Power management
- Design
- ⋮

Quantum optimization?



Quantum optimization?

Quantum algorithms for optimization:

Proven advantage

- Grover search
- Quantum Walks
- Backtracking
- Shortest path
- Minimum weight spanning tree

Heuristics

- Quantum annealing
- Adiabatic algorithms
- QAOA
- VQE
- Quantum machine learning

Quantum optimization?

Quantum algorithms for optimization:

Proven advantage

- Grover search
- Quantum Walks
- Backtracking
- Shortest path
- Minimum weight spanning tree

Heuristics

- Quantum annealing
- Adiabatic algorithms
- QAOA
- VQE
- Quantum machine learning

What about Linear Programs (LPs) and Semidefinite Programs (SDPs)?

LPs and SDPs

A generalization of Linear programs (LPs).

LPs and SDPs

A generalization of linear programs (LPs). Let $x \in \mathbb{R}^n$

$$\mathrm{OPT} = \min \quad \langle c, x \rangle$$

LPs and SDPs

A generalization of Linear programs (LPs). Let $x \in \mathbb{R}^n$

$$\begin{aligned} \mathrm{OPT} = \min \quad & \langle c, x \rangle \\ \text{s.t.} \quad & \langle a_j, x \rangle \leq b_j \quad \text{for all } j \in [m], \\ & x \geq 0 \end{aligned}$$

LPs and SDPs

A generalization of Linear programs (LPs). Let $X \in \mathbb{R}^{n \times n}$

$$\begin{aligned} \mathrm{OPT} = \min \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \langle A_j, X \rangle \le b_j \quad \text{for all } j \in [m], \\ & X \ge 0 \end{aligned}$$

LPs and SDPs

A generalization of Linear programs (LPs). Let $X \in \mathbb{R}^{n \times n}$

$$\begin{aligned} \mathrm{OPT} = \min \quad & \mathrm{Tr}(CX) \\ \text{s.t.} \quad & \mathrm{Tr}(A_j X) \leq b_j \quad \text{for all } j \in [m], \\ & X \succeq 0 \end{aligned}$$

Assumptions ans formalization

LPs and SDPs

A generalization of Linear programs (LPs). Let $X \in \mathbb{R}^{n \times n}$

$$\begin{aligned} \mathrm{OPT} = \min \quad & \mathrm{Tr}(CX) \\ \text{s.t.} \quad & \mathrm{Tr}(A_j X) \le b_j \quad \text{for all } j \in [m], \\ & X \succeq 0 \end{aligned}$$

Assumptions ans formalization

- $n \times n$ variable matrix X, with m constraints.

LPs and SDPs

A generalization of Linear programs (LPs). Let $X \in \mathbb{R}^{n\times n}$

$$\begin{aligned} \mathrm{OPT} = \min \quad & \mathrm{Tr}(CX) \\ \text{s.t.} \quad & \mathrm{Tr}(A_j X) \leq b_j \quad \text{for all } j \in [m], \\ & X \succeq 0 \end{aligned}$$

Assumptions ans formalization

- $n \times n$ variable matrix X, with m constraints.
- Assume $\|C\|, \|A_j\| \leq \frac{1}{2}$ and s-sparse.

LPs and SDPs

A generalization of linear programs (LPs). Let $X \in \mathbb{R}^{n\times n}$

$$\begin{aligned} \mathrm{OPT} = \min \quad & \mathrm{Tr}(CX) \\ \text{s.t.} \quad & \mathrm{Tr}(A_j X) \leq b_j \quad \text{for all } j \in [m], \\ & X \succeq 0 \end{aligned}$$

Assumptions ans formalization

- $n \times n$ variable matrix X, with m constraints.
- Assume $\|C\|, \|A_j\| \leq \frac{1}{2}$ and s-sparse.
- A priori known bounds $\mathrm{Tr}\,(X) \leq R$ and $\sum_{j=0}^{m} y_j \leq r$.

LPs and SDPs

A generalization of Linear programs (LPs). Let $X \in \mathbb{R}^{n\times n}$

$$\begin{aligned} \mathrm{OPT} = \min \quad & \mathrm{Tr}(CX) \\ \text{s.t.} \quad & \mathrm{Tr}(A_jX) \le b_j \quad \text{for all } j \in [m], \\ & X \succeq 0 \end{aligned}$$

Assumptions ans formalization

- $n \times n$ variable matrix X, with m constraints.
- Assume $\|C\|, \|A_j\| \le \frac{1}{2}$ and s-sparse.
- A priori known bounds $\mathrm{Tr}\,(X) \le R$ and $\sum_{j=0}^{m} y_j \le r$.
- Goal: additive ε-approximation of the optimum.

LPs and SDPs

A generalization of Linear programs (LPs). Let $X \in \mathbb{R}^{n\times n}$

$$\begin{aligned} \mathrm{OPT} = \min \quad & \mathrm{Tr}(CX) \\ \text{s.t.} \quad & \mathrm{Tr}(A_j X) \leq b_j \quad \text{for all } j \in [m], \\ & X \succeq 0 \end{aligned}$$

Assumptions ans formalization

- $n \times n$ variable matrix X, with m constraints.
- Assume $\|C\|, \|A_j\| \leq \frac{1}{2}$ and s-sparse.
- A priori known bounds $\mathrm{Tr}(X) \leq R$ and $\sum_{j=0}^{m} y_j \leq r$.
- Goal: additive ε-approximation of the optimum.

Examples: MAXCUT, Lovász theta number, Sum-of-Squares, General Adversary bound, ...

Classical solvers

- Simplex algorithm for linear programs. (Dantzig, 1947)

Classical solvers

- Simplex algorithm for linear programs. (Dantzig, 1947)
- Ellipsoid method in polynomial time. (Khachiyan, 1979)

Classical solvers

- Simplex algorithm for linear programs. (Dantzig, 1947)
- Ellipsoid method in polynomial time. (Khachiyan, 1979)
- Also works for SDPs! (Grötschel, Lovász, Schrijver, 1988)

Classical solvers

- Simplex algorithm for linear programs. (Dantzig, 1947)
- Ellipsoid method in polynomial time. (Khachiyan, 1979)
- Also works for SDPs! (Grötschel, Lovász, Schrijver, 1988)
- State of the art methods: (Lee, Sidford, Wong, 2015)

$$\mathcal{O}\left(m(m^2+n^\omega+mns)\log^{\mathcal{O}(1)}(mnR/\varepsilon)\right),$$

Classical solvers

- Simplex algorithm for linear programs. (Dantzig, 1947)
- Ellipsoid method in polynomial time. (Khachiyan, 1979)
- Also works for SDPs! (Grötschel, Lovász, Schrijver, 1988)
- State of the art methods: (Lee, Sidford, Wong, 2015)

$$\mathcal{O}\left(m(m^2 + n^\omega + mns)\log^{\mathcal{O}(1)}(mnR/\varepsilon)\right),$$

- Arora and Kale (2008):
  Worse error-dependence, better in n and m in certain cases.

Quantum solvers

So far quantum algorithms are based on ideas of Arora-Kale.
Nice speed-ups in n, m but heavy dependence on $1/\delta := (Rr)/\varepsilon$.

- 2016 Sep.: Brandão and Svore: $\widetilde{O}\left(\sqrt{mn}\frac{s^2}{\delta^{18}}\right)$

Quantum solvers

So far quantum algorithms are based on ideas of Arora-Kale.
Nice speed-ups in n, m but heavy dependence on $1/\delta := (Rr)/\varepsilon$.

- 2016 Sep.: Brandão and Svore: $\widetilde{O}\left(\sqrt{mn}\frac{s^2}{\delta^{18}}\right)$
- 2017 May.: van Apeldoorn, G., Gribling, de Wolf: $\widetilde{O}\left(\sqrt{mn}\frac{s^2}{\delta^{8}}\right)$

Quantum solvers

So far quantum algorithms are based on ideas of Arora-Kale.
Nice speed-ups in n, m but heavy dependence on $1/\delta := (Rr)/\varepsilon$.

- 2016 Sep.: Brandão and Svore: $\widetilde{\mathcal{O}}\left(\sqrt{mn}\frac{s^2}{\delta^{18}}\right)$
- 2017 May.: van Apeldoorn, G., Gribling, de Wolf: $\widetilde{\mathcal{O}}\left(\sqrt{mn}\frac{s^2}{\delta^{8}}\right)$
- 2017 Oct.: Brandão, Kalev, Li, Lin, Svore, Wu: $\widetilde{\mathcal{O}}\left(\sqrt{m}\,\mathrm{poly}\left(\frac{B}{\delta^{8}}\right)\right)$

  [in a quantum input model]

Quantum solvers

So far quantum algorithms are based on ideas of Arora-Kale.
Nice speed-ups in n, m but heavy dependence on $1/\delta := (Rr)/\varepsilon$.

- 2016 Sep.: Brandão and Svore: $\widetilde{\mathcal{O}}\left(\sqrt{mn}\frac{s^2}{\delta^{18}}\right)$
- 2017 May.: van Apeldoorn, G., Gribling, de Wolf: $\widetilde{\mathcal{O}}\left(\sqrt{mn}\frac{s^2}{\delta^{8}}\right)$
- 2017 Oct.: Brandão, Kalev, Li, Lin, Svore, Wu: $\widetilde{\mathcal{O}}\left(\sqrt{m}\,\mathrm{poly}\left(\frac{B}{\delta^{8}}\right)\right)$
  [in a quantum input model]
- 2018 Apr.: Brandão, Kalev, Li, Lin, Svore, Wu: $\widetilde{\mathcal{O}}\left((\sqrt{m}+\sqrt{n})\frac{s^2}{\delta^{12}}\right)$

Quantum solvers

So far quantum algorithms are based on ideas of Arora-Kale.
Nice speed-ups in n, m but heavy dependence on $1/\delta := (Rr)/\varepsilon$.

- 2016 Sep.: Brandão and Svore: $\widetilde{\mathcal{O}}\left(\sqrt{mn}\frac{s^2}{\delta^{18}}\right)$
- 2017 May.: van Apeldoorn, G., Gribling, de Wolf: $\widetilde{\mathcal{O}}\left(\sqrt{mn}\frac{s^2}{\delta^8}\right)$
- 2017 Oct.: Brandão, Kalev, Li, Lin, Svore, Wu: $\widetilde{\mathcal{O}}\left(\sqrt{m}\,\mathrm{poly}\left(\frac{B}{\delta^8}\right)\right)$
  [in a quantum input model]
- 2018 Apr.: Brandão, Kalev, Li, Lin, Svore, Wu: $\widetilde{\mathcal{O}}\left((\sqrt{m}+\sqrt{n})\frac{s^2}{\delta^{12}}\right)$
- 2018 Apr.: van Apeldoorn and G.: $\widetilde{\mathcal{O}}\left((\sqrt{m}+\sqrt{n})\frac{s}{\delta^5}\right)$

SDP feasibility problem

$$\begin{aligned} &\min \operatorname{Tr}(CX) \\ &\operatorname{Tr}(A_j X) \le b_j \quad \text{for all } j \in [m] \\ &\operatorname{Tr}(X) = 1 \end{aligned}$$

SDP feasibility problem

$$\mathrm{Tr}(CX) \leq \alpha$$

$$\mathrm{Tr}(A_j X) \leq b_j \quad \text{for all } j \in [m]$$

$$\mathrm{Tr}(X) = 1$$

SDP feasibility problem

$$\mathrm{Tr}\,(A_j X) \leq b_j \quad \text{for all } j \in [m]$$
$$\mathrm{Tr}\,(X) = 1$$

SDP feasibility problem

$$\mathrm{Tr}\,(A_j X) \le b_j \quad \text{for all } j \in [m]$$
$$\mathrm{Tr}\,(X) = 1$$

Find X with $\mathrm{Tr}\,(X) = 1$ such that

$$\mathrm{Tr}\,(A_j X) \le b_j + \delta \quad \text{for all } j \in [m]$$

SDP feasibility problem

$$\mathrm{Tr}\left(A_j X\right) \leq b_j \quad \text{for all } j \in [m]$$
$$\mathrm{Tr}\left(X\right) = 1$$

Find X with $\mathrm{Tr}\left(X\right) = 1$ such that

$$\mathrm{Tr}\left(A_j X\right) \leq b_j + \delta \quad \text{for all } j \in [m]$$

or conclude that the problem is infeasible.

SDP feasibility problem

$$\mathrm{Tr}(A_j X) \leq b_j \quad \text{for all } j \in [m]$$
$$\mathrm{Tr}(X) = 1$$

Find X with $\mathrm{Tr}(X) = 1$ such that

$$\mathrm{Tr}(A_j X) \leq b_j + \delta \quad \text{for all } j \in [m]$$

or conclude that the problem is infeasible.
[In case the problem is infeasible, but a δ-approximation exists we allow both solutions.]

Basic iterative algorithm

Basic iterative algorithm

Set $y^{(0)} := 0 \in \mathbb{R}^m$

Basic iterative algorithm

Set $y^{(0)} := 0 \in \mathbb{R}^m$

For $t = 0 \ldots \frac{\log(n)}{\delta^2}$

Basic iterative algorithm

Set $y^{(0)} := 0 \in \mathbb{R}^m$

For $t = 0 \ldots \frac{\log(n)}{\delta^2}$

- Let $H^{(t)} := \sum_{j=1}^m y_j^{(t)} A_j$ and $X := \frac{e^{-H^{(t)}}}{\mathrm{Tr}\left(e^{-H^{(t)}}\right)}$

Basic iterative algorithm

Set $y^{(0)} := 0 \in \mathbb{R}^m$

For $t = 0 \ldots \frac{\log(n)}{\delta^2}$

- Let $H^{(t)} := \sum_{j=1}^{m} y_j^{(t)} A_j$ and $X := \frac{e^{-H^{(t)}}}{\mathrm{Tr}\left(e^{-H^{(t)}}\right)}$
- Try to find $j \in [m]$: $\mathrm{Tr}\,(A_j X) > b_j$.
  Once found j set $y^{(t+1)} := y^{(t)} + \delta e_j$

Basic iterative algorithm

Set $y^{(0)} := 0 \in \mathbb{R}^m$

For $t = 0 \ldots \frac{\log(n)}{\delta^2}$

- Let $H^{(t)} := \sum_{j=1}^m y_j^{(t)} A_j$ and $X := \frac{e^{-H^{(t)}}}{\mathrm{Tr}\left(e^{-H^{(t)}}\right)}$
- Try to find $j \in [m]$: $\mathrm{Tr}\left(A_j X\right) > b_j$.
  Once found j set $y^{(t+1)} := y^{(t)} + \delta e_j$
- If for all $j \in [m]$: $\mathrm{Tr}\left(A_j X\right) \leq b_j + \delta$
  Can simply output "(approximately) feasible"

Basic iterative algorithm

Set $y^{(0)} := 0 \in \mathbb{R}^m$

For $t = 0 \ldots \frac{\log(n)}{\delta^2}$

- Let $H^{(t)} := \sum_{j=1}^m y_j^{(t)} A_j$ and $X := \frac{e^{-H^{(t)}}}{\mathrm{Tr}\left(e^{-H^{(t)}}\right)}$
- Try to find $j \in [m]$: $\mathrm{Tr}\left(A_j X\right) > b_j$.
  Once found j set $y^{(t+1)} := y^{(t)} + \delta e_j$
- If for all $j \in [m]$: $\mathrm{Tr}\left(A_j X\right) \leq b_j + \delta$
  Can simply output "(approximately) feasible"

Conclude that the problem is infeasible

Basic iterative algorithm

Set $y^{(0)} := 0 \in \mathbb{R}^m$

For $t = 0 \dots \frac{\log(n)}{\delta^2}$

- Let $H^{(t)} := \sum_{j=1}^{m} y_j^{(t)} A_j$ and $X := \frac{e^{-H^{(t)}}}{\mathrm{Tr}\left(e^{-H^{(t)}}\right)}$
- Try to find $j \in [m]$: $\mathrm{Tr}\left(A_j X\right) > b_j$.
  Once found j set $y^{(t+1)} := y^{(t)} + \delta e_j$
- If for all $j \in [m]$: $\mathrm{Tr}\left(A_j X\right) \leq b_j + \delta$
  Can simply output "(approximately) feasible"

Conclude that the problem is infeasible

Proof of correctness by Lee, Raghavendra and Steurer '15.
(Very similar to the algorithm of Arora and Kale '08.)

Basic iterative algorithm

Set $y^{(0)} := 0 \in \mathbb{R}^m$

For $t = 0 \ldots \frac{\log(n)}{\delta^2}$

- Let $H^{(t)} := \sum_{j=1}^m y_j^{(t)} A_j$ and $X := \frac{e^{-H^{(t)}}}{\mathrm{Tr}\left(e^{-H^{(t)}}\right)}$
- Try to find $j \in [m]$: $\mathrm{Tr}\,(A_j X) > b_j$.
  Once found j set $y^{(t+1)} := y^{(t)} + \delta e_j$
- If for all $j \in [m]$: $\mathrm{Tr}\,(A_j X) \leq b_j + \delta$
  Can simply output "(approximately) feasible"

Conclude that the problem is infeasible

Proof of correctness by Lee, Raghavendra and Steurer '15.
(Very similar to the algorithm of Arora and Kale '08.)

Application to quantum SDP-solving Brandão et al. '16,'17.

Preparing X

Suppose we can query the position and value of the non-zero elements of the sparse matrices A_j, then we can implement

$$U_{\mathrm{Select}} = \sum_{j=1}^{m} |j\rangle\langle j| \otimes U_j, \text{ such that } U_j = \begin{bmatrix} A_j & \cdot \\ \cdot & \cdot \end{bmatrix},$$

using $\widetilde{O}(s)$ queries and gates.

Preparing X

Suppose we can query the position and value of the non-zero elements of the sparse matrices A_j, then we can implement

$$U_{\text{Select}} = \sum_{j=1}^{m} |j\rangle\langle j| \otimes U_j, \text{ such that } U_j = \begin{bmatrix} A_j & \cdot \\ \cdot & \cdot \end{bmatrix},$$

using $\widetilde{O}(s)$ queries and gates.

Let us store $y^{(t)}$ in QRAM using the data structure of Kerenidis and Prakash. ($y^{(t)}$ is sparse $\Rightarrow$ QRAM is small.)

Preparing X

Suppose we can query the position and value of the non-zero elements of the sparse matrices A_j, then we can implement

$$U_{\mathrm{Select}} = \sum_{j=1}^{m} |j\rangle\langle j| \otimes U_j, \text{ such that } U_j = \begin{bmatrix} A_j & \cdot \\ \cdot & \cdot \end{bmatrix},$$

using $\widetilde{\mathcal{O}}(s)$ queries and gates.

Let us store $y^{(t)}$ in QRAM using the data structure of Kerenidis and Prakash. ($y^{(t)}$ is sparse $\Rightarrow$ QRAM is small.)

$$U_{\mathrm{Select}} \xrightarrow[\widetilde{\mathcal{O}}(1)]{\mathrm{LCU}} \begin{bmatrix} \delta \sum_{j=1}^{m} y_j^{(t)} A_j & \cdot \\ \cdot & \cdot \end{bmatrix}$$

Preparing X

Suppose we can query the position and value of the non-zero elements of the sparse matrices A_j, then we can implement

$$U_{\mathrm{Select}} = \sum_{j=1}^{m} |j\rangle\langle j| \otimes U_j, \text{ such that } U_j = \begin{bmatrix} A_j & \cdot \\ \cdot & \cdot \end{bmatrix},$$

using $\widetilde{\mathcal{O}}(s)$ queries and gates.

Let us store $y^{(t)}$ in QRAM using the data structure of Kerenidis and Prakash. ($y^{(t)}$ is sparse $\Rightarrow$ QRAM is small.)

$$U_{\mathrm{Select}} \overset{\mathrm{LCU}}{\underset{\widetilde{\mathcal{O}}(1)}{\Longrightarrow}} \begin{bmatrix} \delta H^{(t)} & \cdot \\ \cdot & \cdot \end{bmatrix}$$

Preparing X

Suppose we can query the position and value of the non-zero elements of the sparse matrices A_j, then we can implement

$$U_{\text{Select}} = \sum_{j=1}^{m} |j\rangle\langle j| \otimes U_j, \text{ such that } U_j = \begin{bmatrix} A_j & \cdot \\ \cdot & \cdot \end{bmatrix},$$

using $\widetilde{\mathcal{O}}(s)$ queries and gates.

Let us store $y^{(t)}$ in QRAM using the data structure of Kerenidis and Prakash. ($y^{(t)}$ is sparse $\Rightarrow$ QRAM is small.)

$$U_{\text{Select}} \underset{\widetilde{\mathcal{O}}(1)}{\overset{\text{LCU}}{\Longrightarrow}} \begin{bmatrix} \delta H^{(t)} & \cdot \\ \cdot & \cdot \end{bmatrix} \underset{\widetilde{\mathcal{O}}(1/\delta)}{\overset{\text{SVT}}{\Longrightarrow}} \begin{bmatrix} e^{-H^{(t)}} & \cdot \\ \cdot & \cdot \end{bmatrix}$$

Preparing X

Suppose we can query the position and value of the non-zero elements of the sparse matrices A_j, then we can implement

$$U_{\mathrm{Select}} = \sum_{j=1}^{m} |j\rangle\langle j| \otimes U_j, \text{ such that } U_j = \begin{bmatrix} A_j & \cdot \\ \cdot & \cdot \end{bmatrix},$$

using $\widetilde{\mathcal{O}}(s)$ queries and gates.

Let us store $y^{(t)}$ in QRAM using the data structure of Kerenidis and Prakash. ($y^{(t)}$ is sparse $\Rightarrow$ QRAM is small.)

$$U_{\mathrm{Select}} \underset{\widetilde{\mathcal{O}}(1)}{\overset{\mathrm{LCU}}{\Longrightarrow}} \begin{bmatrix} \delta H^{(t)} & \cdot \\ \cdot & \cdot \end{bmatrix} \underset{\widetilde{\mathcal{O}}(1/\delta)}{\overset{\mathrm{SVT}}{\Longrightarrow}} \begin{bmatrix} e^{-H^{(t)}} & \cdot \\ \cdot & \cdot \end{bmatrix} \underset{\widetilde{\mathcal{O}}(\sqrt{n})}{\overset{\mathrm{Amp.}}{\Longrightarrow}} X$$

Preparing X

Suppose we can query the position and value of the non-zero elements of the sparse matrices A_j, then we can implement

$$U_{\mathrm{Select}} = \sum_{j=1}^{m} |j\rangle\langle j| \otimes U_j, \text{ such that } U_j = \begin{bmatrix} A_j & \cdot \\ \cdot & \cdot \end{bmatrix},$$

using $\widetilde{\mathcal{O}}(s)$ queries and gates.

Let us store $y^{(t)}$ in QRAM using the data structure of Kerenidis and Prakash. ($y^{(t)}$ is sparse $\Rightarrow$ QRAM is small.)

$$U_{\mathrm{Select}} \underset{\widetilde{\mathcal{O}}(1)}{\overset{\mathrm{LCU}}{\Longrightarrow}} \begin{bmatrix} \delta H^{(t)} & \cdot \\ \cdot & \cdot \end{bmatrix} \underset{\widetilde{\mathcal{O}}(1/\delta)}{\overset{\mathrm{SVT}}{\Longrightarrow}} \begin{bmatrix} e^{-H^{(t)}} & \cdot \\ \cdot & \cdot \end{bmatrix} \underset{\widetilde{\mathcal{O}}(\sqrt{n})}{\overset{\mathrm{Amp.}}{\Longrightarrow}} X$$

Preparation of X has $\widetilde{\mathcal{O}}\left(\frac{\sqrt{n}s}{\delta}\right)$ query and time complexity.

Violated constraints

Decide if $\mathrm{Tr}\left(A_j X\right) \leq b_j \pm \delta$

Violated constraints

Decide if $\mathrm{Tr}(A_j X) \le b_j \pm \delta$

- Can be done using $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X

Violated constraints

Decide if $\mathrm{Tr}\,(A_j X) \leq b_j \pm \delta$

- Can be done using $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X
- with query and gate complexity $\widetilde{\mathcal{O}}\left(\frac{s}{\delta^2}\right)$.

Now we use the quantum OR lemma of Harrow et al. '17 using its fast implementation due to Brandão et al. '17.

Violated constraints

Decide if $\mathrm{Tr}(A_j X) \leq b_j \pm \delta$

- Can be done using $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X
- with query and gate complexity $\widetilde{\mathcal{O}}\left(\frac{s}{\delta^2}\right)$.

Now we use the quantum OR lemma of Harrow et al. '17 using its fast implementation due to Brandão et al. '17.

Violated constraints

Decide if $\operatorname{Tr}(A_j X) \leq b_j \pm \delta$

- Can be done using $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X
- with query and gate complexity $\widetilde{\mathcal{O}}\left(\frac{s}{\delta^2}\right)$.

Now we use the quantum OR lemma of Harrow et al. '17 using its fast implementation due to Brandão et al. '17.

- Find $j \in [m]$ such that $\operatorname{Tr}(A_j X) \geq b_j$

Violated constraints

Decide if $\mathrm{Tr}\left(A_j X\right) \leq b_j \pm \delta$

- Can be done using $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X
- with query and gate complexity $\widetilde{\mathcal{O}}\left(\frac{s}{\delta^2}\right)$.

Now we use the quantum OR lemma of Harrow et al. '17 using its fast implementation due to Brandão et al. '17.

- Find $j \in [m]$ such that $\mathrm{Tr}\left(A_j X\right) \geq b_j$
- or conclude that for all $j \in [m]$ we have $\mathrm{Tr}\left(A_j X\right) \leq b_j + \delta$

Violated constraints

Decide if $\mathrm{Tr}\,(A_j X) \leq b_j \pm \delta$

- Can be done using $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X
- with query and gate complexity $\widetilde{\mathcal{O}}\left(\frac{s}{\delta^2}\right)$.

Now we use the quantum OR lemma of Harrow et al. '17 using its fast implementation due to Brandão et al. '17.

- Find $j \in [m]$ such that $\mathrm{Tr}\,(A_j X) \geq b_j$
- or conclude that for all $j \in [m]$ we have $\mathrm{Tr}\,(A_j X) \leq b_j + \delta$

The above problem can be solved with $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X with query and gate complexity $\widetilde{\mathcal{O}}\left(\frac{\sqrt{m}s}{\delta^2}\right)$.

Violated constraints

Decide if $\mathrm{Tr}(A_j X) \leq b_j \pm \delta$

- Can be done using $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X
- with query and gate complexity $\widetilde{\mathcal{O}}\left(\frac{s}{\delta^2}\right)$.

Now we use the quantum OR lemma of Harrow et al. '17 using its fast implementation due to Brandão et al. '17.

- Find $j \in [m]$ such that $\mathrm{Tr}(A_j X) \geq b_j$
- or conclude that for all $j \in [m]$ we have $\mathrm{Tr}(A_j X) \leq b_j + \delta$

The above problem can be solved with $\widetilde{\mathcal{O}}\left(\frac{1}{\delta^2}\right)$ copies of X with query and gate complexity $\widetilde{\mathcal{O}}\left(\frac{\sqrt{m}s}{\delta^2}\right)$.

The overall query and gate complexity is $\widetilde{\mathcal{O}}\left(\frac{\sqrt{m}s}{\delta^2} + \frac{\sqrt{n}s}{\delta^3}\right)$.

Applications

Shadow tomography

Applications

Shadow tomography

- Given samples of $\rho \in \mathbb{C}^{n\times n}$, and m meas. operators M_j

Applications

Shadow tomography

- Given samples of $\rho \in \mathbb{C}^{n \times n}$, and m meas. operators M_j
- find $y \in \mathbb{R}^m$ such that $H := \sum_{j=1}^m y_j M_j$ satisfies
  for all $j \in [m]$ that $\left| \mathrm{Tr}\left(M_j \rho\right) - \mathrm{Tr}\left(M_j \frac{e^{-H}}{\mathrm{Tr}\left(e^{-H}\right)} \right) \right| \le \delta.$

Applications

Shadow tomography

- Given samples of $\rho \in \mathbb{C}^{n\times n}$, and m meas. operators M_j
- find $y \in \mathbb{R}^m$ such that $H := \sum_{j=1}^m y_j M_j$ satisfies
  for all $j \in [m]$ that $\left|\mathrm{Tr}\left(M_j \rho\right) - \mathrm{Tr}\left(M_j \frac{e^{-H}}{\mathrm{Tr}\left(e^{-H}\right)}\right)\right| \leq \delta.$

Aaronson showed how to solve using $\mathcal{O}\left(\frac{\log^4(m)\log(n)}{\delta^4}\right)$ samples.

Applications

Shadow tomography

- Given samples of $\rho \in \mathbb{C}^{n\times n}$, and m meas. operators M_j
- find $y \in \mathbb{R}^m$ such that $H := \sum_{j=1}^m y_j M_j$ satisfies for all $j \in [m]$ that $\left|\mathrm{Tr}\left(M_j\rho\right) - \mathrm{Tr}\left(M_j \frac{e^{-H}}{\mathrm{Tr}\left(e^{-H}\right)}\right)\right| \leq \delta.$

Aaronson showed how to solve using $\mathcal{O}\left(\frac{\log^4(m)\log(n)}{\delta^4}\right)$ samples.
Our SDP solver recovers it in a gate efficient way incurring $\widetilde{\mathcal{O}}\left(\sqrt{m}\right)$ gate and query complexity overhead.

Applications

Shadow tomography

- Given samples of $\rho \in \mathbb{C}^{n\times n}$, and m meas. operators M_j
- find $y \in \mathbb{R}^m$ such that $H := \sum_{j=1}^{m} y_j M_j$ satisfies

  for all $j \in [m]$ that $\left|\mathrm{Tr}\left(M_j\rho\right) - \mathrm{Tr}\left(M_j \frac{e^{-H}}{\mathrm{Tr}\left(e^{-H}\right)}\right)\right| \leq \delta.$

Aaronson showed how to solve using $\mathcal{O}\left(\frac{\log^4(m)\log(n)}{\delta^4}\right)$ samples.
Our SDP solver recovers it in a gate efficient way incurring $\widetilde{\mathcal{O}}\left(\sqrt{m}\right)$ gate and query complexity overhead.

Further applications

Applications

Shadow tomography

- Given samples of $\rho \in \mathbb{C}^{n\times n}$, and m meas. operators M_j
- find $y \in \mathbb{R}^m$ such that $H := \sum_{j=1}^{m} y_j M_j$ satisfies
  for all $j \in [m]$ that $\left|\mathrm{Tr}\left(M_j\rho\right) - \mathrm{Tr}\left(M_j \frac{e^{-H}}{\mathrm{Tr}\left(e^{-H}\right)}\right)\right| \leq \delta.$

Aaronson showed how to solve using $\mathcal{O}\left(\frac{\log^4(m)\log(n)}{\delta^4}\right)$ samples.
Our SDP solver recovers it in a gate efficient way incurring $\widetilde{\mathcal{O}}\left(\sqrt{m}\right)$ gate and query complexity overhead.

Further applications

- Quantum state discrimination with maximal total success probability.

Applications

Shadow tomography

- Given samples of $\rho \in \mathbb{C}^{n\times n}$, and m meas. operators M_j
- find $y \in \mathbb{R}^m$ such that $H := \sum_{j=1}^m y_j M_j$ satisfies
  for all $j \in [m]$ that $\left|\mathrm{Tr}\left(M_j\rho\right) - \mathrm{Tr}\left(M_j \frac{e^{-H}}{\mathrm{Tr}\left(e^{-H}\right)}\right)\right| \le \delta.$

Aaronson showed how to solve using $\mathcal{O}\left(\frac{\log^4(m)\log(n)}{\delta^4}\right)$ samples.
Our SDP solver recovers it in a gate efficient way incurring $\widetilde{\mathcal{O}}\left(\sqrt{m}\right)$ gate and query complexity overhead.

Further applications

- Quantum state discrimination with maximal total success probability.
- Optimal measurement design.

Summary

Quantum SDP solver

Summary

Quantum SDP solver

- Query and gate complexity $\widetilde{O}\left(\left(\sqrt{m}+\frac{\sqrt{n}}{\delta}\right)\frac{s}{\delta^4}\right)$.

Summary

Quantum SDP solver

- Query and gate complexity $\widetilde{O}\left(\left(\sqrt{m}+\frac{\sqrt{n}}{\delta}\right)\frac{s}{\delta^4}\right)$.

Matching lower bounds for LPs (and hence SDPs)

Summary

Quantum SDP solver

- Query and gate complexity $\widetilde{O}\left(\left(\sqrt{m}+\frac{\sqrt{n}}{\delta}\right)\frac{s}{\delta^4}\right)$.

Matching lower bounds for LPs (and hence SDPs)

- $\Omega\left(\sqrt{m}+\sqrt{n}\right)$ in sparse matrix access input model.

Summary

Quantum SDP solver

- Query and gate complexity $\widetilde{O}\left(\left(\sqrt{m}+\frac{\sqrt{n}}{\delta}\right)\frac{s}{\delta^4}\right)$.

Matching lower bounds for LPs (and hence SDPs)

- $\Omega\left(\sqrt{m}+\sqrt{n}\right)$ in sparse matrix access input model.
- $\Omega\left(\sqrt{m}/\delta\right)$ in block-encoding input model.

Summary

Quantum SDP solver

- Query and gate complexity $\widetilde{O}\left(\left(\sqrt{m}+\frac{\sqrt{n}}{\delta}\right)\frac{s}{\delta^4}\right)$.

Matching lower bounds for LPs (and hence SDPs)

- $\Omega\left(\sqrt{m}+\sqrt{n}\right)$ in sparse matrix access input model.
- $\Omega\left(\sqrt{m}/\delta\right)$ in block-encoding input model.

Open questions/future research

Summary

Quantum SDP solver

- Query and gate complexity $\widetilde{O}\left(\left(\sqrt{m}+\frac{\sqrt{n}}{\delta}\right)\frac{s}{\delta^4}\right)$.

Matching lower bounds for LPs (and hence SDPs)

- $\Omega\left(\sqrt{m}+\sqrt{n}\right)$ in sparse matrix access input model.
- $\Omega\left(\sqrt{m}/\delta\right)$ in block-encoding input model.

Open questions/future research

- Problem specific fine-tuned algorithms?

Summary

Quantum SDP solver

- Query and gate complexity $\widetilde{O}\left(\left(\sqrt{m}+\frac{\sqrt{n}}{\delta}\right)\frac{s}{\delta^4}\right)$.

Matching lower bounds for LPs (and hence SDPs)

- $\Omega\left(\sqrt{m}+\sqrt{n}\right)$ in sparse matrix access input model.
- $\Omega\left(\sqrt{m}/\delta\right)$ in block-encoding input model.

Open questions/future research

- Problem specific fine-tuned algorithms?
- Tight quantum bounds for the δ dependence?

Summary

Quantum SDP solver

- Query and gate complexity $\widetilde{\mathcal{O}}\left(\left(\sqrt{m}+\frac{\sqrt{n}}{\delta}\right)\frac{s}{\delta^4}\right)$.

Matching lower bounds for LPs (and hence SDPs)

- $\Omega\left(\sqrt{m}+\sqrt{n}\right)$ in sparse matrix access input model.
- $\Omega\left(\sqrt{m}/\delta\right)$ in block-encoding input model.

Open questions/future research

- Problem specific fine-tuned algorithms?
- Tight quantum bounds for the δ dependence?
- Speed-ups using other, e.g., interior point methods?

Sources of images

- © Lucas Surtin (http://unisci24.com)
- © Google maps
- © Gurobi.com
- © ScienceBuzz.org