The Complexity of Ising Models with Complex Weights

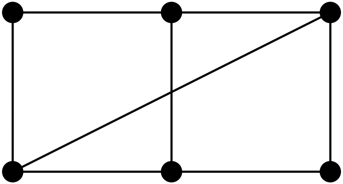
Leslie Ann Goldberg[1] and Heng Guo [2]

[1]University of Oxford
[2]University of Wisconsin-Madison

Ann Arbor, MI
Dec 6th 2014

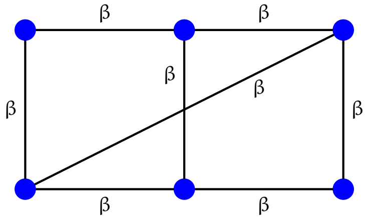
Ising Model

Edge interaction $\begin{bmatrix} \beta & 1 \\ 1 & \beta \end{bmatrix}$



Ising Model

Edge interaction $\begin{bmatrix} \beta & 1 \\ 1 & \beta \end{bmatrix}$

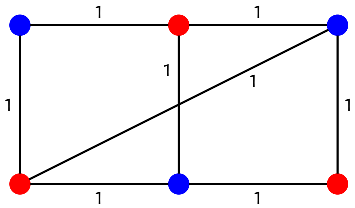


$$\Pr(\sigma) \sim w(\sigma)$$

$$w(\sigma) = \beta^8$$

Ising Model

Edge interaction $\begin{bmatrix} \beta & 1 \\ 1 & \beta \end{bmatrix}$

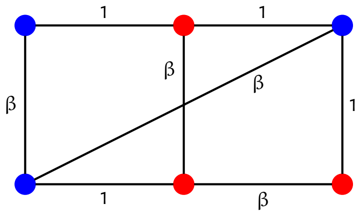


$$\Pr(\sigma) \sim w(\sigma)$$

$$w(\sigma) = \beta^0 = 1$$

Ising Model

Edge interaction $\begin{bmatrix} \beta & 1 \\ 1 & \beta \end{bmatrix}$

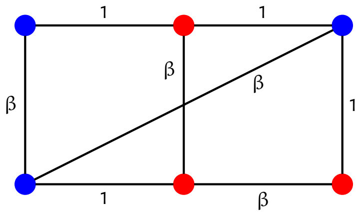


$$\Pr(\sigma) \sim w(\sigma)$$

$$w(\sigma) = \beta^4$$

Ising Model

Edge interaction $\begin{bmatrix} \beta & 1 \\ 1 & \beta \end{bmatrix}$



Partition function (normalizing factor):

$$Z_G(\beta) = \sum_{\sigma:V\to\{0,1\}} w(\sigma)$$

where $w(\sigma) = \beta^{m(\sigma)}$, $m(\sigma)$ is the number of monochromatic edges under σ.

Complexity Results

Exact evaluation of $Z_G(\beta)$:

- #**P**-hard unless $\beta = 0, \pm 1, \pm i$. [Jaeger, Vertigan, Welsh 90]

Complexity Results

Exact evaluation of $Z_G(\beta)$:

- #**P**-hard unless $\beta = 0, \pm 1, \pm i$. [Jaeger, Vertigan, Welsh 90]

Approximation:

- FPRAS if $\beta > 1$ and **NP**-hard if $0 < \beta < 1$. [Jerrum, Sinclair 93]

Complexity Results

Exact evaluation of $Z_G(\beta)$:

- #**P**-hard unless $\beta = 0, \pm 1, \pm \mathtt{i}$. [Jaeger, Vertigan, Welsh 90]

Approximation:

- FPRAS if $\beta > 1$ and **NP**-hard if $0 < \beta < 1$. [Jerrum, Sinclair 93]
- For bounded degree graphs and $0 < \beta < 1$,
  - FPTAS below the uniqueness threshold.
    [Sinclair, Srivastava, Thurley 12] and [Li,Lu,Yin 12,13]
  - **NP**-hard beyond it. [Sly, Sun 12]

Complexity Results

Exact evaluation of $Z_G(\beta)$:

- #**P**-hard unless $\beta = 0, \pm 1, \pm \mathtt{i}$. [Jaeger, Vertigan, Welsh 90]

Approximation:

- FPRAS if $\beta > 1$ and **NP**-hard if $0 < \beta < 1$. [Jerrum, Sinclair 93]
- For bounded degree graphs and $0 < \beta < 1$,
  - FPTAS below the uniqueness threshold. [Sinclair, Srivastava, Thurley 12] and [Li,Lu,Yin 12,13]
  - **NP**-hard beyond it. [Sly, Sun 12]

Note that we can amplify any constant approximation into an FPRAS.

Complexity Results

Exact evaluation of $Z_G(\beta)$:

- #**P**-hard unless $\beta = 0, \pm 1, \pm \mathtt{i}$. [Jaeger, Vertigan, Welsh 90]

Approximation:

- FPRAS if $\beta > 1$ and **NP**-hard if $0 < \beta < 1$. [Jerrum, Sinclair 93]
- For bounded degree graphs and $0 < \beta < 1$,
  - FPTAS below the uniqueness threshold.
    [Sinclair, Srivastava, Thurley 12] and [Li,Lu,Yin 12,13]
  - **NP**-hard beyond it. [Sly, Sun 12]

Note that we can amplify any constant approximation into an FPRAS.

In this talk we will focus on approximating $Z_G(\beta)$ for $\beta \in \mathbb{C}$.

Quantum Connection

IQP (Instantaneous Quantum Polynomial-time) [Shepherd, Bremner 09] is a subclass of **BQP**.

Quantum Connection

IQP (Instantaneous Quantum Polynomial-time) [Shepherd, Bremner 09] is a subclass of **BQP**.

An **IQP** circuit satisfies:

- each qubit line starts and ends with an H gate;
- all other gates are diagonal.

Quantum Connection

IQP (Instantaneous Quantum Polynomial-time) [Shepherd, Bremner 09] is a subclass of **BQP**.

An **IQP** circuit satisfies:

- each qubit line starts and ends with an H gate;
- all other gates are diagonal.

Lemma (Fuiji, Morimae, 13)

Given an **IQP** circuit C and an output $\mathbf{x}$, there is a graph G such that the marginal probability of $\mathbf{x}$ equals to $|Z_G(e^{\pi \mathtt{i}/4})|$ up to an easy to compute factor.

Implication on Complex Ising

Lemma (Fuiji, Morimae, 13)

Given an **IQP** circuit C and an output $\mathbf{x}$, there is a graph G such that the marginal probability of $\mathbf{x}$ equals to $|Z_G(e^{\pi \mathrm{i}/4})|$ up to an easy to compute factor.

Implication on Complex Ising

Lemma (Fuiji, Morimae, 13)

Given an **IQP** circuit C and an output $\mathbf{x}$, there is a graph G such that the marginal probability of $\mathbf{x}$ equals to $|Z_G(e^{\pi \mathtt{i}/4})|$ up to an easy to compute factor.

Lemma (Bremner, Jorza, Shepherd 11)

Sampling $\mathbf{x}$ with multiplicative errors classically in polynomial time implies that the polynomial hierarchy collapses to the third level.

Implication on Complex Ising

Lemma (Fuiji, Morimae, 13)

Given an **IQP** circuit C and an output $\mathbf{x}$, there is a graph G such that the marginal probability of $\mathbf{x}$ equals to $|Z_G(e^{\pi \mathrm{i}/4})|$ up to an easy to compute factor.

Lemma (Bremner, Jorza, Shepherd 11)

Sampling $\mathbf{x}$ with multiplicative errors classically in polynomial time implies that the polynomial hierarchy collapses to the third level.

Combining above two results together, $|Z_G(e^{\pi \mathrm{i}/4})|$ cannot be approximated efficiently unless the polynomial hierarchy collapses.

Implication on Complex Ising

Lemma (Fuiji, Morimae, 13)

Given an **IQP** circuit C and an output $\mathbf{x}$, there is a graph G such that the marginal probability of $\mathbf{x}$ equals to $|Z_G(e^{\pi \mathrm{i}/4})|$ up to an easy to compute factor.

Lemma (Bremner, Jorza, Shepherd 11)

Sampling $\mathbf{x}$ with multiplicative errors classically in polynomial time implies that the polynomial hierarchy collapses to the third level.

Combining above two results together, $|Z_G(e^{\pi \mathrm{i}/4})|$ cannot be approximated efficiently unless the polynomial hierarchy collapses.

But is the quantum machinery necessary to study $Z_G(\beta)$?

Approximate Complex Numbers

Given a complex number z, one may approximate $|z|$ and $\arg(z)$.

Approximate Complex Numbers

Given a complex number z, one may approximate $|z|$ and $\arg(z)$.

Definition (Ziv's measure [Ziv 82])

The distance between two complex numbers z and z' should be measured as

$$d(z, z') = \frac{|z - z'|}{\max(|z|, |z'|)},$$

where $d(0, 0) = 0$.

Approximate Complex Numbers

Given a complex number z, one may approximate $|z|$ and $\arg(z)$.

Definition (Ziv's measure [Ziv 82])

The distance between two complex numbers z and z' should be measured as

$$d(z, z') = \frac{|z - z'|}{\max(|z|, |z'|)},$$

where $d(0, 0) = 0$.

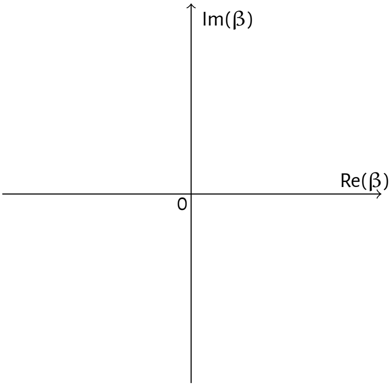
Hardness of approximating $|z|$ or $\arg(z)$ implies hardness under Ziv's measure.

Complex Ising

Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

Complex Ising

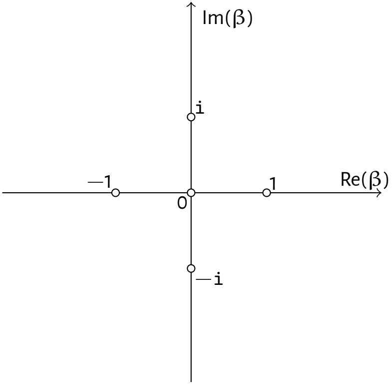
Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.



Complex Ising

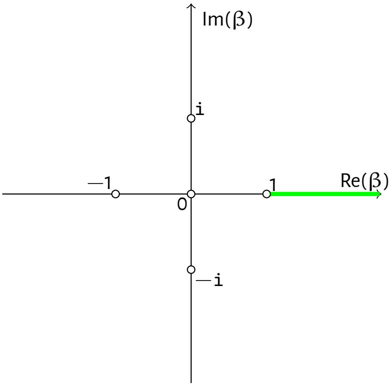
Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

- $\beta \in \{0, \pm 1, \pm \mathtt{i}\}$, tractable. [JVW90].



Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

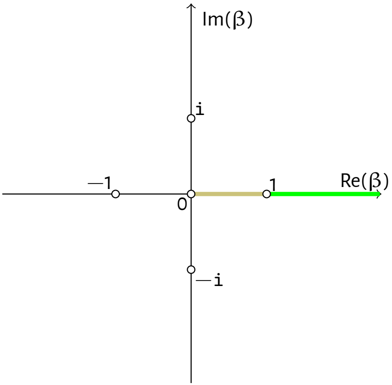
- $\beta \in \{0, \pm 1, \pm \mathtt{i}\}$, tractable. [JVW90].
- $\beta \in (1, \infty)$, FPRAS. [JS93]



Complex Ising

Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

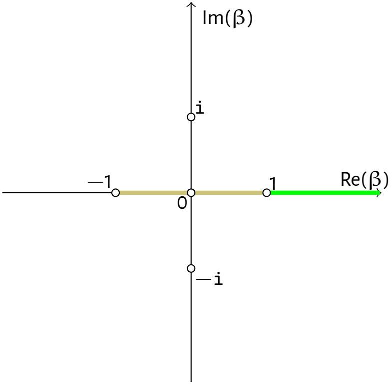
- $\beta \in \{0, \pm 1, \pm \mathtt{i}\}$, tractable. [JVW90].
- $\beta \in (1, \infty)$, FPRAS. [JS93]
- $\beta \in (0, 1)$, **NP**-hard. [JS93]



Complex Ising

Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

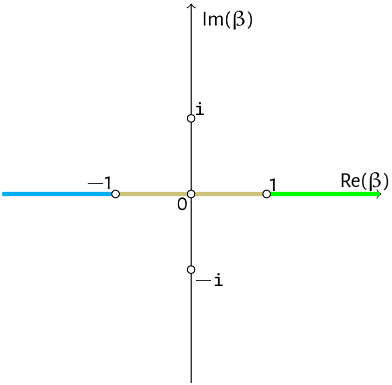
- $\beta \in \{0, \pm 1, \pm i\}$, tractable. [JVW90].
- $\beta \in (1, \infty)$, FPRAS. [JS93]
- $\beta \in (0, 1)$, **NP**-hard. [JS93]
- $\beta \in (-1, 0)$, **NP**-hard. [GJ08]



Complex Ising

Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

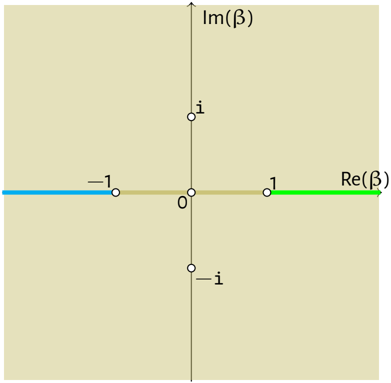
- $\beta \in \{0, \pm 1, \pm i\}$, tractable. [JVW90].
- $\beta \in (1, \infty)$, FPRAS. [JS93]
- $\beta \in (0, 1)$, **NP**-hard. [JS93]
- $\beta \in (-1, 0)$, **NP**-hard. [GJ08]
- $\beta \in (-\infty, -1)$, #PM. [GJ08]



Complex Ising

Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

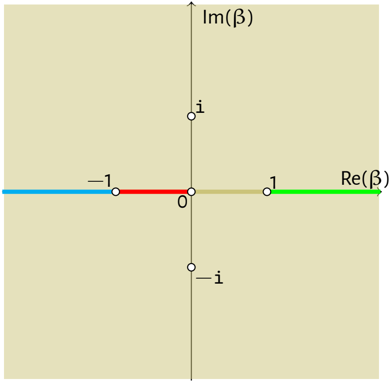
- $\beta \in \{0, \pm 1, \pm \mathtt{i}\}$, tractable. [JVW90].
- $\beta \in (1, \infty)$, FPRAS. [JS93]
- $\beta \in (0, 1)$, **NP**-hard. [JS93]
- $\beta \in (-1, 0)$, **NP**-hard. [GJ08]
- $\beta \in (-\infty, -1)$, #PM. [GJ08]
- $\beta \notin \mathbb{R} \cup \{\mathtt{i}, -\mathtt{i}\}$, **NP**-hard. [GG14]



Complex Ising

Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

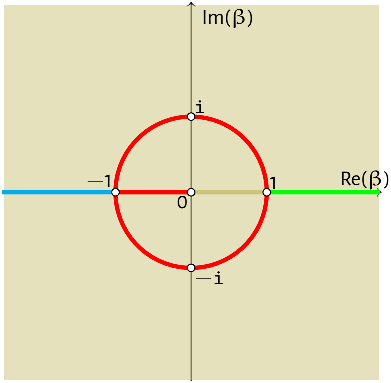
- $\beta \in \{0, \pm 1, \pm \mathtt{i}\}$, tractable. [JVW90].
- $\beta \in (1, \infty)$, FPRAS. [JS93]
- $\beta \in (0, 1)$, **NP**-hard. [JS93]
- $\beta \in (-1, 0)$, **NP**-hard. [GJ08]
- $\beta \in (-\infty, -1)$, #PM. [GJ08]
- $\beta \notin \mathbb{R} \cup \{\mathtt{i}, -\mathtt{i}\}$, **NP**-hard. [GG14]
- $\beta \in (-1, 0)$, **#P**-hard. [GG14]



Complex Ising

Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

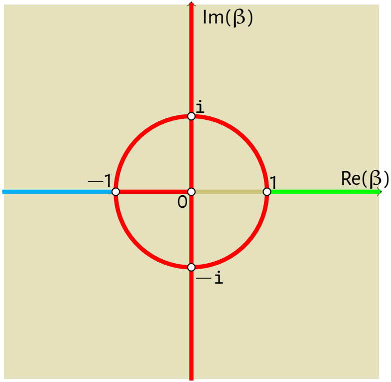
- $\beta \in \{0, \pm 1, \pm \mathtt{i}\}$, tractable. [JVW90].
- $\beta \in (1, \infty)$, FPRAS. [JS93]
- $\beta \in (0, 1)$, **NP**-hard. [JS93]
- $\beta \in (-1, 0)$, **NP**-hard. [GJ08]
- $\beta \in (-\infty, -1)$, #PM. [GJ08]
- $\beta \notin \mathbb{R} \cup \{\mathtt{i}, -\mathtt{i}\}$, **NP**-hard. [GG14]
- $\beta \in (-1, 0)$, #**P**-hard. [GG14]
- $|\beta| = 1$, $\beta \notin \{\pm 1, \pm \mathtt{i}\}$, #**P**-hard. [GG14]



Complex Ising

Our main result is the approximation complexity of $|Z_G(\beta)|$ for $\beta \in \mathbb{C}$.

- $\beta \in \{0, \pm 1, \pm i\}$, tractable. [JVW90].
- $\beta \in (1, \infty)$, FPRAS. [JS93]
- $\beta \in (0, 1)$, **NP**-hard. [JS93]
- $\beta \in (-1, 0)$, **NP**-hard. [GJ08]
- $\beta \in (-\infty, -1)$, #PM. [GJ08]
- $\beta \notin \mathbb{R} \cup \{i, -i\}$, **NP**-hard. [GG14]
- $\beta \in (-1, 0)$, #**P**-hard. [GG14]
- $|\beta| = 1$, $\beta \notin \{\pm 1, \pm i\}$, #**P**-hard. [GG14]
- Re(β)=0, $\beta \notin \{0, \pm i\}$, #**P**-hard. [GG14]



#**P**-hardness

To get #**P**-hardness of approximation is non-trivial, as any problem in #**P** can be approximated using an **NP**-oracle within polynomial time.

#**P**-hardness

To get #**P**-hardness of approximation is non-trivial, as any problem in #**P** can be approximated using an **NP**-oracle within polynomial time.

If $Z_G(\beta) = 0$, even the approximation requires the exact answer.
We relax our problem so that if $Z_G(\beta) = 0$, we accept any return.
Our hardness results hold for these relaxed versions.

#**P**-hardness

To get #**P**-hardness of approximation is non-trivial, as any problem in #**P** can be approximated using an **NP**-oracle within polynomial time.

If $Z_G(\beta) = 0$, even the approximation requires the exact answer.
We relax our problem so that if $Z_G(\beta) = 0$, we accept any return.
Our hardness results hold for these relaxed versions.

We reduce #Minimum Cardinality (s, t)-Cut [Provan, Ball 83] to approximating $|Z_G(\beta)|$ for any $\beta \in (-1, 0)$.

#**P**-hardness

To get #**P**-hardness of approximation is non-trivial, as any problem in #**P** can be approximated using an **NP**-oracle within polynomial time.

If $Z_G(\beta) = 0$, even the approximation requires the exact answer.
We relax our problem so that if $Z_G(\beta) = 0$, we accept any return.
Our hardness results hold for these relaxed versions.

We reduce #MINIMUM CARDINALITY (s, t)-CUT [Provan, Ball 83] to approximating $|Z_G(\beta)|$ for any $\beta \in (-1, 0)$.

The key part of the #**P**-hardness proof is a bisection argument.
This idea has been used to show hardness of determining signs of Tutte polynomials (at real points). [Goldberg, Jerrum, 12]

The Reduction

- Given a graph G, suppose $C = \#\text{Min-}(s,t)\text{-Cut}$.

The Reduction

- Given a graph G, suppose $C = $ #Min-(s,t)-Cut.

  We may assume (s,t) is not in G. Introduce a new edge $e = (s,t)$.

The Reduction

- Given a graph G, suppose $C = $ #Min-(s, t)-Cut.
  We may assume (s, t) is not in G. Introduce a new edge $e = (s, t)$.
- We want to put a weight x on e and a fixed weight γ on every other edge.

The Reduction

- Given a graph G, suppose $C = \#\text{Min-}(s,t)\text{-Cut}$.
  We may assume (s,t) is not in G. Introduce a new edge $e = (s,t)$.
- We want to put a weight x on e and a fixed weight γ on every other edge.
  - Using edge weight β, we build gadgets to implement γ.
    We can also approximate any $x \in (-1,0)$ exponentially accurately.

The Reduction

- Given a graph G, suppose $C = \#\text{Min-}(s,t)\text{-Cut}$.
  We may assume (s,t) is not in G. Introduce a new edge $e = (s,t)$.
- We want to put a weight x on e and a fixed weight γ on every other edge.
  - Using edge weight β, we build gadgets to implement γ.
    We can also approximate any $x \in$ (-1,0) exponentially accurately.
- Call the graph G_x. Let $f(x) = Z_{G_x}(\gamma)$.

The Reduction

- Given a graph G, suppose $C = \#\text{Min-}(s,t)\text{-Cut}$.
  We may assume (s,t) is not in G. Introduce a new edge $e = (s,t)$.
- We want to put a weight x on e and a fixed weight γ on every other edge.
  - Using edge weight β, we build gadgets to implement γ.
    We can also approximate any $x \in (-1,0)$ exponentially accurately.
- Call the graph G_x. Let $f(x) = Z_{G_x}(\gamma)$.
  Notice that $f(x)$ is a linear function in x.
  Let x_0 be the root of $f(x)$.

The Reduction

- Given a graph G, suppose $C = \#\text{Min-}(s,t)\text{-Cut}$.
  We may assume (s,t) is not in G. Introduce a new edge $e = (s,t)$.
- We want to put a weight x on e and a fixed weight γ on every other edge.
  - Using edge weight β, we build gadgets to implement γ.
    We can also approximate any $x \in (-1,0)$ exponentially accurately.
- Call the graph G_x. Let $f(x) = Z_{G_x}(\gamma)$.
  Notice that $f(x)$ is a linear function in x.
  Let x_0 be the root of $f(x)$.
- Our choice of γ guarantees that $f(0) > 0$, $f(-1) < 0$.

The Reduction

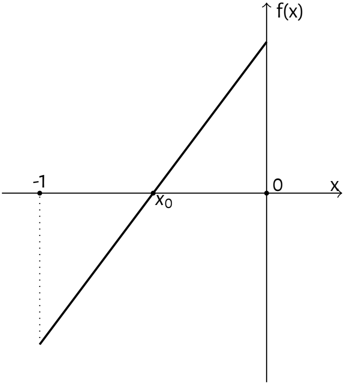
- Given a graph G, suppose $C = \#\text{Min-}(s,t)\text{-Cut}$.
  We may assume (s,t) is not in G. Introduce a new edge $e = (s,t)$.
- We want to put a weight x on e and a fixed weight γ on every other edge.
  - Using edge weight β, we build gadgets to implement γ.
    We can also approximate any $x \in (-1,0)$ exponentially accurately.
- Call the graph G_x. Let $f(x) = Z_{G_x}(\gamma)$.
  Notice that $f(x)$ is a linear function in x.
  Let x_0 be the root of $f(x)$.
- Our choice of γ guarantees that $f(0) > 0$, $f(-1) < 0$.
  Moreover if we can approximate x_0 accurately enough, C can be computed exactly.

Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$.
We recursively shrink the interval containing x_0.

Bisection with an Oracle of Approximating Norms

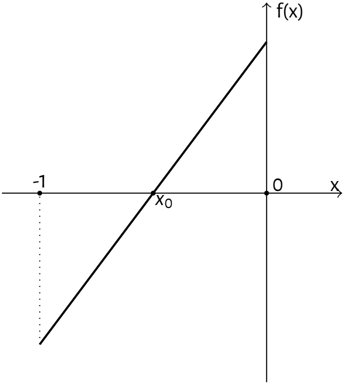
The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$.
We recursively shrink the interval containing x_0.



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$.
We recursively shrink the interval containing x_0.

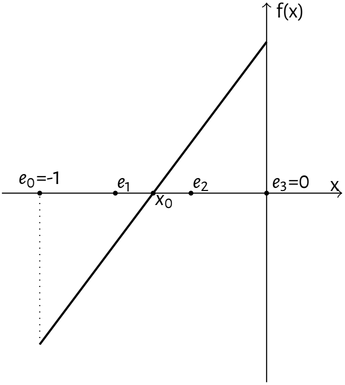
- We begin with the interval (-1,0).



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$. We recursively shrink the interval containing x_0.

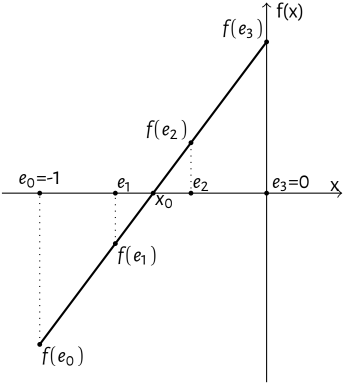
- We begin with the interval (-1,0).
- Divide the current interval into 3 subintervals equally.



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$.
We recursively shrink the interval containing x_0.

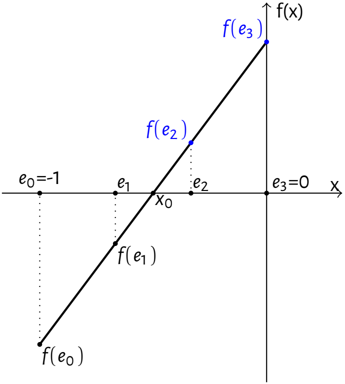
- We begin with the interval (-1,0).
- Divide the current interval into 3 subintervals equally.
- Evaluate $|f(x)|$ approximately at the 4 endpoints.



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$.
We recursively shrink the interval containing x_0.

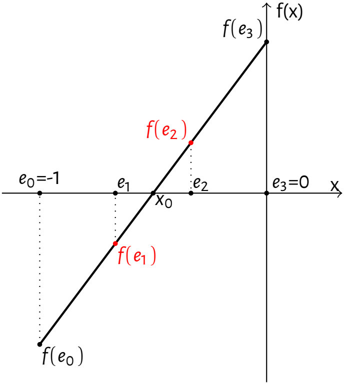
- We begin with the interval (-1,0).
- Divide the current interval into 3 subintervals equally.
- Evaluate $|f(x)|$ approximately at the 4 endpoints.
- If two points x_1, x_2 are on the same side of x_0, then the accuracy K guarantees that the ordering of $g(x_1)$ and $g(x_2)$ is the same as that of $|f(x_1)|$ and $|f(x_2)|$.



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$.
We recursively shrink the interval containing x_0.

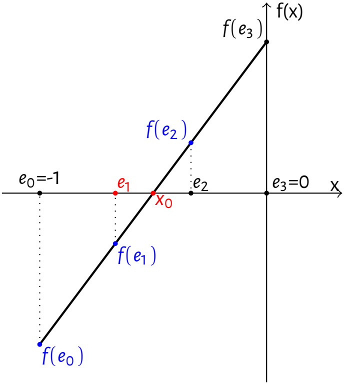
- We begin with the interval (-1,0).
- Divide the current interval into 3 subintervals equally.
- Evaluate $|f(x)|$ approximately at the 4 endpoints.
- If two points x_1, x_2 are on the same side of x_0, then the accuracy K guarantees that the ordering of $g(x_1)$ and $g(x_2)$ is the same as that of $|f(x_1)|$ and $|f(x_2)|$.
- Otherwise the order may be wrong, but it happens at most once.



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$. We recursively shrink the interval containing x_0.

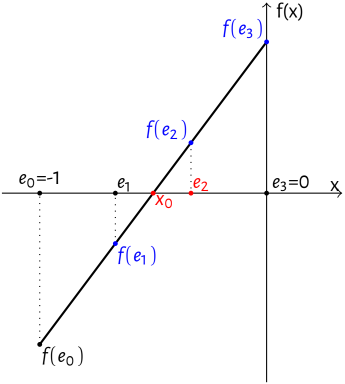
- We begin with the interval (-1,0).
- Divide the current interval into 3 subintervals equally.
- Evaluate $|f(x)|$ approximately at the 4 endpoints.
- If two points x_1, x_2 are on the same side of x_0, then the accuracy K guarantees that the ordering of $g(x_1)$ and $g(x_2)$ is the same as that of $|f(x_1)|$ and $|f(x_2)|$.
- Otherwise the order may be wrong, but it happens at most once.
- If $g(e_0) > g(e_1) > g(e_2)$, then $e_1 < x_0$.



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$.
We recursively shrink the interval containing x_0.

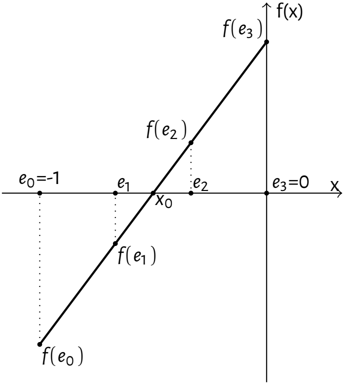
- We begin with the interval (-1,0).
- Divide the current interval into 3 subintervals equally.
- Evaluate $|f(x)|$ approximately at the 4 endpoints.
- If two points x_1, x_2 are on the same side of x_0, then the accuracy K guarantees that the ordering of $g(x_1)$ and $g(x_2)$ is the same as that of $|f(x_1)|$ and $|f(x_2)|$.
- Otherwise the order may be wrong, but it happens at most once.
- If $g(e_0) > g(e_1) > g(e_2)$, then $e_1 < x_0$.
  If $g(e_1) < g(e_2) < g(e_3)$, then $e_2 > x_0$.



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$.
We recursively shrink the interval containing x_0.

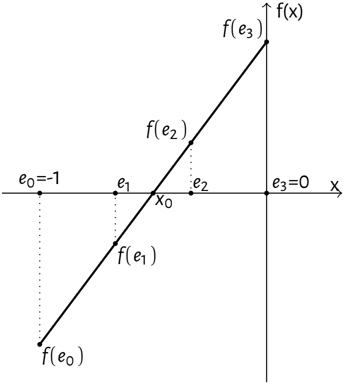
- We begin with the interval (-1,0).
- Divide the current interval into 3 subintervals equally.
- Evaluate $|f(x)|$ approximately at the 4 endpoints.
- If two points x_1, x_2 are on the same side of x_0, then the accuracy K guarantees that the ordering of $g(x_1)$ and $g(x_2)$ is the same as that of $|f(x_1)|$ and $|f(x_2)|$.
- Otherwise the order may be wrong, but it happens at most once.
- If $g(e_0) > g(e_1) > g(e_2)$, then $e_1 < x_0$.
  If $g(e_1) < g(e_2) < g(e_3)$, then $e_2 > x_0$.
- At least one of the cases is true, so we can shrink the interval by $\frac{2}{3}$.



Bisection with an Oracle of Approximating Norms

The oracle returns $|f(x)|$ up to some constant K. Call the approximation $g(x)$. We recursively shrink the interval containing x_0.

- We begin with the interval (-1,0).
- Divide the current interval into 3 subintervals equally.
- Evaluate $|f(x)|$ approximately at the 4 endpoints.
- If two points x_1, x_2 are on the same side of x_0, then the accuracy K guarantees that the ordering of $g(x_1)$ and $g(x_2)$ is the same as that of $|f(x_1)|$ and $|f(x_2)|$.
- Otherwise the order may be wrong, but it happens at most once.
- If $g(e_0) > g(e_1) > g(e_2)$, then $e_1 < x_0$.
  If $g(e_1) < g(e_2) < g(e_3)$, then $e_2 > x_0$.
- At least one of the cases is true, so we can shrink the interval by $\frac{2}{3}$.



We divide the interval into more subintervals so that we don't need an exact evaluation of $|f(x)|$ at x_0.

Further results

- Hardness results of approximating $\arg(Z_G(\beta))$.

Further results

- Hardness results of approximating $\arg(Z_G(\beta))$.
- Given an oracle computing the sign of Tutte polynomial at $(-e^{2\pi i/5}, -e^{8\pi i/5})$ over planar graphs, all problems in **BQP** can be solved classically in polynomial time [Bordewich, Freedman, Lovász, Welsh 05].

Further results

- Hardness results of approximating $\arg(Z_G(\beta))$.
- Given an oracle computing the sign of Tutte polynomial at $(-e^{2\pi i/5}, -e^{8\pi i/5})$ over planar graphs, all problems in **BQP** can be solved classically in polynomial time [Bordewich, Freedman, Lovász, Welsh 05].

  We showed that to determine this sign is #**P**-hard over general graphs.

Further results

- Hardness results of approximating $\arg(Z_G(\beta))$.
- Given an oracle computing the sign of Tutte polynomial at $(-e^{2\pi i/5}, -e^{8\pi i/5})$ over planar graphs, all problems in **BQP** can be solved classically in polynomial time [Bordewich, Freedman, Lovász, Welsh 05].

  We showed that to determine this sign is #**P**-hard over general graphs.
- A complete classification of approximating partition functions of Ising models with external fields, when both the edge weight and the field are roots of unity.

Complex Ising with Fields

Edge weight β, external field λ:

$$Z_G(\beta;\lambda) = \sum_{\sigma:V\to\{0,1\}} w(\sigma)$$

where $w(\sigma) = \beta^{m(\sigma)}\lambda^{c_1(\sigma)}$, $m(\sigma)$ is the number of monochromatic edges under σ, and $c_1(\sigma)$ is the number of “blue” vertices.

Complex Ising with Fields

Edge weight β, external field λ:

$$Z_G(\beta;\lambda) = \sum_{\sigma:V\to\{0,1\}} w(\sigma)$$

where $w(\sigma) = \beta^{m(\sigma)}\lambda^{c_1(\sigma)}$, $m(\sigma)$ is the number of monochromatic edges under σ, and $c_1(\sigma)$ is the number of "blue" vertices.

Theorem

Let β and λ be two roots of unity. Then the following holds:

- If $\beta = \pm 1$, or $\beta = \pm \mathtt{i}$ and $\lambda \in \{1, -1, \mathtt{i}, -\mathtt{i}\}$, $Z_G(\beta;\lambda)$ can be computed exactly in polynomial time.
- Otherwise $|Z_G(\beta;\lambda)|$ is #**P**-hard to approximate.

Proof Ideas

- The general idea to reduce from previous cases.

Proof Ideas

- The general idea to reduce from previous cases.
- Given a graph G, we build a graph G' such that $Z_{G'}(\beta;\lambda)$ is close to $Z_G(\beta)$.

Proof Ideas

- The general idea to reduce from previous cases.
- Given a graph G, we build a graph G' such that $Z_{G'}(\beta;\lambda)$ is close to $Z_G(\beta)$.
- We can build gadgets to make $|Z_{G'}(\beta;\lambda) - Z_G(\beta)|$ exponentially small, but we need to convert additive distances into multiplicative distances.

Proof Ideas

- The general idea to reduce from previous cases.
- Given a graph G, we build a graph G' such that $Z_{G'}(\beta;\lambda)$ is close to $Z_G(\beta)$.
- We can build gadgets to make $|Z_{G'}(\beta;\lambda) - Z_G(\beta)|$ exponentially small, but we need to convert additive distances into multiplicative distances.
- If $|Z_G(\beta)| = 0$, then it is impossible.
  The non-zero relaxation is necessary to make the reduction go through.

Proof Ideas

- The general idea to reduce from previous cases.
- Given a graph G, we build a graph G' such that $Z_{G'}(\beta;\lambda)$ is close to $Z_G(\beta)$.
- We can build gadgets to make $|Z_{G'}(\beta;\lambda) - Z_G(\beta)|$ exponentially small, but we need to convert additive distances into multiplicative distances.
- If $|Z_G(\beta)| = 0$, then it is impossible.
  The non-zero relaxation is necessary to make the reduction go through.
- So we can assume $|Z_G(\beta)| \neq 0$. All we need is a lower bound of $|Z_G(\beta)|$.
  (Even an exponential one suffices.)

Proof Ideas

- The general idea to reduce from previous cases.
- Given a graph G, we build a graph G' such that $Z_{G'}(\beta; \lambda)$ is close to $Z_G(\beta)$.
- We can build gadgets to make $|Z_{G'}(\beta; \lambda) - Z_G(\beta)|$ exponentially small, but we need to convert additive distances into multiplicative distances.
- If $|Z_G(\beta)| = 0$, then it is impossible.
  The non-zero relaxation is necessary to make the reduction go through.
- So we can assume $|Z_G(\beta)| \neq 0$. All we need is a lower bound of $|Z_G(\beta)|$.
  (Even an exponential one suffices.)
- If β is rational, this is straightforward by a granularity argument.
  If β is algebraic, we need to use some basic transcendental number theory.

Thank You!

Papers and slides available on my homepage:
`www.cs.wisc.edu/~hguo/`