UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR
THEORETISCHE INFORMATIK

PCP Teil 2

Oberseminar

Sebastian Berndt

Heute

- Kurze Wiederholung

Heute

- Kurze Wiederholung
- Nicht-Approximierbarkeit von MAX-3-LIN

Heute

- Kurze Wiederholung
- Nicht-Approximierbarkeit von MAX-3-LIN
- Unique Games Conjecture

Heute

- Kurze Wiederholung
- Nicht-Approximierbarkeit von MAX-3-LIN
- Unique Games Conjecture
- Nicht-Approximierbarkeit von MAX-CUT

Wiederholung

PCP

Die Klasse $\mathrm{PCP}_{c,s}[r(n), q(n)]$ enthält alle Sprachen $\mathcal{L} \subseteq \{0,1\}^*$, für die es einen Verifier V gibt mit:

- V erhält eine *Instanz* $x \in \{0,1\}^n$ und einen *Beweis* $\pi \in \{0,1\}^*$ (es reicht $|\pi| \leq 2^{r(n)} \cdot q(n)$)

Wiederholung

PCP

Die Klasse $\mathrm{PCP}_{c,s}[r(n), q(n)]$ enthält alle Sprachen $\mathcal{L} \subseteq \{0,1\}^*$, für die es einen Verifier V gibt mit:

- V erhält eine *Instanz* $x \in \{0,1\}^n$ und einen *Beweis* $\pi \in \{0,1\}^*$ (es reicht $|\pi| \leq 2^{r(n)} \cdot q(n)$)
- V läuft in Zeit $\mathrm{poly}(n)$

Wiederholung

PCP

Die Klasse $\text{PCP}_{c,s}[r(n), q(n)]$ enthält alle Sprachen $\mathcal{L} \subseteq \{0,1\}^*$, für die es einen Verifier V gibt mit:

- V erhält eine *Instanz* $x \in \{0,1\}^n$ und einen *Beweis* $\pi \in \{0,1\}^*$ (es reicht $|\pi| \leq 2^{r(n)} \cdot q(n)$)
- V läuft in Zeit $\text{poly}(n)$
- V darf $\mathcal{O}(r(n))$ Münzwürfe durchführen

Wiederholung

PCP

Die Klasse $\mathrm{PCP}_{c,s}[r(n), q(n)]$ enthält alle Sprachen $\mathcal{L} \subseteq \{0,1\}^*$, für die es einen Verifier V gibt mit:

- V erhält eine *Instanz* $x \in \{0,1\}^n$ und einen *Beweis* $\pi \in \{0,1\}^*$ (es reicht $|\pi| \leq 2^{r(n)} \cdot q(n)$)
- V läuft in Zeit $\mathrm{poly}(n)$
- V darf $\mathcal{O}(r(n))$ Münzwürfe durchführen
- V darf $\mathcal{O}(q(n))$ Bits des Beweises π lesen

Wiederholung

PCP

Die Klasse $\mathrm{PCP}_{c,s}[r(n), q(n)]$ enthält alle Sprachen $\mathcal{L} \subseteq \{0,1\}^*$, für die es einen Verifier V gibt mit:

- V erhält eine *Instanz* $x \in \{0,1\}^n$ und einen *Beweis* $\pi \in \{0,1\}^*$ (es reicht $|\pi| \leq 2^{r(n)} \cdot q(n)$)
- V läuft in Zeit $\mathrm{poly}(n)$
- V darf $\mathcal{O}(r(n))$ Münzwürfe durchführen
- V darf $\mathcal{O}(q(n))$ Bits des Beweises π lesen
- $\forall x \in \mathcal{L} : \exists \pi : \Pr[V^\pi(x) = 1] \geq c$ (*Completeness*)

Wiederholung

PCP

Die Klasse $\mathrm{PCP}_{c,s}[r(n), q(n)]$ enthält alle Sprachen $\mathcal{L} \subseteq \{0,1\}^*$, für die es einen Verifier V gibt mit:

- V erhält eine *Instanz* $x \in \{0,1\}^n$ und einen *Beweis* $\pi \in \{0,1\}^*$ (es reicht $|\pi| \leq 2^{r(n)} \cdot q(n)$)
- V läuft in Zeit $\mathrm{poly}(n)$
- V darf $\mathcal{O}(r(n))$ Münzwürfe durchführen
- V darf $\mathcal{O}(q(n))$ Bits des Beweises π lesen
- $\forall x \in \mathcal{L} : \exists \pi : \Pr[V^\pi(x) = 1] \geq c$ (*Completeness*)
- $\forall x \notin \mathcal{L} : \forall \pi : \Pr[V^\pi(x) = 1] \leq s$ (*Soundness*)

Wiederholung

PCP-Theorem (Arora, Lund, Motwani, Sudan, Szegedy)

$$\mathrm{PCP}_{1,1/2}[\log n, 1] = \mathcal{NP}$$

Wiederholung

PCP-Theorem (Arora, Lund, Motwani, Sudan, Szegedy)

$$\mathrm{PCP}_{1,1/2}[\log n, 1] = \mathcal{NP}$$

ε-GAP-MAX-3SAT

Gegeben: 3SAT-Formel φ mit m Klauseln

Gesucht:
- Ist φ erfüllbar? (JA)
- Gilt für alle β, dass höchstens $(1 - \varepsilon) \cdot m$ Klauseln gleichzeitig erfüllt sind? (NEIN)

Wiederholung

PCP-Theorem (Arora, Lund, Motwani, Sudan, Szegedy)

$$\mathrm{PCP}_{1,1/2}[\log n, 1] = \mathcal{NP}$$

ε-GAP-MAX-3SAT

Gegeben: 3SAT-Formel φ mit m Klauseln

Gesucht:
- Ist φ erfüllbar? (JA)
- Gilt für alle β, dass höchstens $(1 - \varepsilon) \cdot m$ Klauseln gleichzeitig erfüllt sind? (NEIN)

Nicht-Approximierbarkeit

Das PCP-Theorem ist äquivalent zur Aussage, dass es eine Konstante ε gibt, so dass ε-GAP-MAX-3SAT $\mathcal{NP}$-hart ist.

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert
2. Baue einfachen Verifier V (Outer Verifier)

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert
2. Baue einfachen Verifier V (Outer Verifier)
3. Modifiziere V zu V^* (Inner Verifier)

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot\cup Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e \colon \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X \colon X \to \Sigma_X, \ell_Y \colon Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot\cup Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e\colon \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X\colon X \to \Sigma_X, \ell_Y\colon Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Beispiel

3SAT-Formel $\varphi = \bigwedge_{i=1}^m K_i$ mit Variablen $x_1, \dots, x_n$

- $X = \{K_1, K_2, \dots, K_m\}$

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot{\cup} Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e \colon \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X \colon X \to \Sigma_X, \ell_Y \colon Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Beispiel

3SAT-Formel $\varphi = \bigwedge_{i=1}^{m} K_i$ mit Variablen $x_1, \ldots, x_n$

- $X = \{K_1, K_2, \ldots, K_m\}$
- $Y = \{x_1, x_2, \ldots, x_n\}$

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot\cup Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e \colon \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X \colon X \to \Sigma_X, \ell_Y \colon Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Beispiel

3SAT-Formel $\varphi = \bigwedge_{i=1}^{m} K_i$ mit Variablen $x_1, \ldots, x_n$

- $X = \{K_1, K_2, \ldots, K_m\}$
- $Y = \{x_1, x_2, \ldots, x_n\}$
- $E = \{(K, x) \mid x \in K\}$

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot{\cup} Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e \colon \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X \colon X \to \Sigma_X, \ell_Y \colon Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Beispiel

3SAT-Formel $\varphi = \bigwedge_{i=1}^{m} K_i$ mit Variablen $x_1, \ldots, x_n$

- $X = \{K_1, K_2, \ldots, K_m\}$
- $Y = \{x_1, x_2, \ldots, x_n\}$
- $E = \{(K, x) \mid x \in K\}$
- $\Sigma_X = [1..7]$

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot\cup Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e : \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X : X \to \Sigma_X, \ell_Y : Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Beispiel

3SAT-Formel $\varphi = \bigwedge_{i=1}^{m} K_i$ mit Variablen $x_1, \ldots, x_n$

- $X = \{K_1, K_2, \ldots, K_m\}$
- $Y = \{x_1, x_2, \ldots, x_n\}$
- $E = \{(K, x) \mid x \in K\}$
- $\Sigma_X = [1..7]$
- $\Sigma_Y = \{0, 1\}$

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot\cup Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e \colon \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X \colon X \to \Sigma_X, \ell_Y \colon Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Beispiel

3SAT-Formel $\varphi = \bigwedge_{i=1}^{m} K_i$ mit Variablen $x_1, \ldots, x_n$

- $X = \{K_1, K_2, \ldots, K_m\}$
- $Y = \{x_1, x_2, \ldots, x_n\}$
- $E = \{(K, x) \mid x \in K\}$
- $\Sigma_X = [1..7]$
- $\Sigma_Y = \{0, 1\}$
- $\pi_{(K,x)}$ wandelt Belegung von K in Belegung von x um

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot\cup Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e \colon \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X \colon X \to \Sigma_X, \ell_Y \colon Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Beispiel

3SAT-Formel $\varphi = \bigwedge_{i=1}^{m} K_i$ mit Variablen $x_1, \ldots, x_n$

- $X = \{K_1, K_2, \ldots, K_m\}$
- $Y = \{x_1, x_2, \ldots, x_n\}$
- $E = \{(K, x) \mid x \in K\}$
- $\Sigma_X = [1..7]$
- $\Sigma_Y = \{0, 1\}$
- $\pi_{(K,x)}$ wandelt Belegung von K in Belegung von x um

Label Cover

LABEL-COVER

Gegeben: Bipartiter, regulärer Graph $G = (X \dot{\cup} Y, E)$, Alphabete Σ_X, Σ_Y, Menge von $\Pi = \{\pi_e \colon \Sigma_X \to \Sigma_Y \mid e \in E\}$

Gesucht: Zwei Abbildungen $\ell_X \colon X \to \Sigma_X, \ell_Y \colon Y \to \Sigma_Y$, so dass für alle $(x, y) \in E$ gilt $\pi_{(x,y)}(\ell_X(x)) = \ell_Y(y)$.

Beispiel

3SAT-Formel $\varphi = \bigwedge_{i=1}^{m} K_i$ mit Variablen $x_1, \ldots, x_n$

- $X = \{K_1, K_2, \ldots, K_m\}$
- $Y = \{x_1, x_2, \ldots, x_n\}$
- $E = \{(K, x) \mid x \in K\}$
- $\Sigma_X = [1..7]$
- $\Sigma_Y = \{0, 1\}$
- $\pi_{(K,x)}$ wandelt Belegung von K in Belegung von x um

Satz

Es gibt eine Konstante ε', so dass ε'-GAP-MAX-LABEL-COVER $\mathcal{NP}$-hart ist.

3-LIN

3-LIN

Gegeben: Menge von Variablen $x_1, \dots, x_n$ und Gleichungen C der Form $x \oplus y \oplus z = \{0, 1\}$

Gesucht: Belegung $\beta\colon \{x_1, x_2, \dots, x_n\} \to \{0, 1\}$, so dass alle Gleichungen erfüllt sind

3-LIN

3-LIN

Gegeben: Menge von Variablen $x_1, \ldots, x_n$ und Gleichungen C der Form $x \oplus y \oplus z = \{0, 1\}$

Gesucht: Belegung $\beta\colon \{x_1, x_2, \ldots, x_n\} \to \{0, 1\}$, so dass alle Gleichungen erfüllt sind

Satz (Johan Håstad, 2001)

Wenn es einen $(\log n, 1)$-Verifier V mit Soundness s und Completeness c gibt, der genau drei Bits x, y, z liest und akzeptiert, falls $x \oplus y \oplus z = 0$, so kann MAX-3-LIN nicht besser als mit Rate s/c approximiert werden. (außer $\mathcal{P} = \mathcal{NP}$)

3-LIN

3-LIN

Gegeben: Menge von Variablen $x_1, \ldots, x_n$ und Gleichungen C der Form $x \oplus y \oplus z = \{0, 1\}$

Gesucht: Belegung $\beta \colon \{x_1, x_2, \ldots, x_n\} \to \{0, 1\}$, so dass alle Gleichungen erfüllt sind

Satz (Johan Håstad, 2001)

Wenn es einen $(\log n, 1)$-Verifier V mit Soundness s und Completeness c gibt, der genau drei Bits x, y, z liest und akzeptiert, falls $x \oplus y \oplus z = 0$, so kann MAX-3-LIN nicht besser als mit Rate s/c approximiert werden. (außer $\mathcal{P} = \mathcal{NP}$)

Beweis

- Variablen $x_1, x_2, \ldots$

3-LIN

3-LIN

Gegeben: Menge von Variablen $x_1, \dots, x_n$ und Gleichungen C der Form $x \oplus y \oplus z = \{0, 1\}$

Gesucht: Belegung $\beta\colon \{x_1, x_2, \dots, x_n\} \to \{0, 1\}$, so dass alle Gleichungen erfüllt sind

Satz (Johan Håstad, 2001)

Wenn es einen $(\log n, 1)$*-Verifier* V *mit Soundness* s *und Completeness* c *gibt, der genau drei Bits* x, y, z *liest und akzeptiert, falls* $x \oplus y \oplus z = 0$*, so kann* MAX-3-LIN *nicht besser als mit Rate* s/c *approximiert werden. (außer* $\mathcal{P} = \mathcal{NP}$*)*

Beweis

- Variablen $x_1, x_2, \dots$
- Simuliere Münzwürfe und konstruiere Gleichungen $C = \{x_i \oplus x_j \oplus x_k = 0 \mid \Pr[V \text{ liest gleichzeitig Bits } i, j, k] > 0\}$

3-LIN

3-LIN

Gegeben: Menge von Variablen $x_1, \ldots, x_n$ und Gleichungen C der Form $x \oplus y \oplus z = \{0, 1\}$

Gesucht: Belegung $\beta\colon \{x_1, x_2, \ldots, x_n\} \to \{0, 1\}$, so dass alle Gleichungen erfüllt sind

Satz (Johan Håstad, 2001)

Wenn es einen $(\log n, 1)$*-Verifier* V *mit Soundness* s *und Completeness* c *gibt, der genau drei Bits* x, y, z *liest und akzeptiert, falls* $x \oplus y \oplus z = 0$, *so kann MAX-3-LIN nicht besser als mit Rate* s/c *approximiert werden.* *(außer* $\mathcal{P} = \mathcal{NP}$*)*

Beweis

- Variablen $x_1, x_2, \ldots$
- Simuliere Münzwürfe und konstruiere Gleichungen $C = \{x_i \oplus x_j \oplus x_k = 0 \mid \Pr[V \text{ liest gleichzeitig Bits } i, j, k] > 0\}$
- $\Pr[V \text{ akzeptiert}] = \frac{\text{Anzahl erfüllter Gleichungen}}{\text{Anzahl der Gleichungen}}$

Outer Verifier

LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für LABEL-COVER:

Outer Verifier

LABEL-COVER Instanz $G = (X \dot\cup Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für LABEL-COVER:

Verifier

- Wähle $(x, y) \in E$ uniform

Outer Verifier

LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für LABEL-COVER:

Verifier

- Wähle $(x, y) \in E$ uniform
- Lies $\ell_X(x), \ell_Y(y)$ aus dem Beweis

Outer Verifier

LABEL-COVER Instanz $G = (X \dot\cup Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für LABEL-COVER:

Verifier

- Wähle $(x, y) \in E$ uniform
- Lies $\ell_X(x), \ell_Y(y)$ aus dem Beweis
- Teste, ob $\pi(\ell_X(x)) = \ell_Y(y)$

Outer Verifier

LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für LABEL-COVER:

Verifier

- Wähle $(x, y) \in E$ uniform
- Lies $\ell_X(x), \ell_Y(y)$ aus dem Beweis
- Teste, ob $\pi(\ell_X(x)) = \ell_Y(y)$

Outer Verifier

LABEL-COVER Instanz $G = (X \dot\cup Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für LABEL-COVER:

Verifier

- Wähle $(x, y) \in E$ uniform
- Lies $\ell_X(x), \ell_Y(y)$ aus dem Beweis
- Teste, ob $\pi(\ell_X(x)) = \ell_Y(y)$

Wenn alle Kanten erfüllt sind, ist die Wahrscheinlichkeit, dass diese erfüllt ist, 1.

Outer Verifier

LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für LABEL-COVER:

Verifier

- Wähle $(x, y) \in E$ uniform
- Lies $\ell_X(x), \ell_Y(y)$ aus dem Beweis
- Teste, ob $\pi(\ell_X(x)) = \ell_Y(y)$

Wenn alle Kanten erfüllt sind, ist die Wahrscheinlichkeit, dass diese erfüllt ist, 1.
Sind maximal $(1 - \varepsilon) \cdot |E|$ der Kanten erfüllt, ist die Wahrscheinlichkeit, dass diese erfüllt ist, höchstens $(1 - \varepsilon)$.

Outer Verifier

LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für LABEL-COVER:

Verifier

- Wähle $(x, y) \in E$ uniform
- Lies $\ell_X(x), \ell_Y(y)$ aus dem Beweis
- Teste, ob $\pi(\ell_X(x)) = \ell_Y(y)$

Wenn alle Kanten erfüllt sind, ist die Wahrscheinlichkeit, dass diese erfüllt ist, 1.
Sind maximal $(1 - \varepsilon) \cdot |E|$ der Kanten erfüllt, ist die Wahrscheinlichkeit, dass diese erfüllt ist, höchstens $(1 - \varepsilon)$.
Wir lesen $\log |\Sigma_X| + \log |\Sigma_Y|$ Bits!

3-Bit Verifier / Inner Verifier

Idee

Wir codieren $\ell_X(x), \ell_Y(y)$ so, dass wir für den Test $\pi(\ell_X(x)) = \ell_Y(y)$ nur drei Bits lesen müssen!

3-Bit Verifier / Inner Verifier

Idee

Wir codieren $\ell_X(x), \ell_Y(y)$ so, dass wir für den Test $\pi(\ell_X(x)) = \ell_Y(y)$ nur drei Bits lesen müssen!

Long Code

Sei $\mathcal{B}_\Sigma = \{f\colon \Sigma \to \{0,1\}\}$ Menge der booleschen Funktionen ($|\mathcal{B}_\Sigma| = 2^{|\Sigma|}$) mit $\mathcal{B}_\Sigma = \{f_0, f_1, \dots, f_{2^{|\Sigma|}-1}\}$.
Wir codieren $a \in \Sigma$ mit $\mathrm{Long}_a = f_0(a)f_1(a)\dots f_{2^{|\Sigma|}-1}(a)$, d.h. $\mathrm{Long}_a[f] = f(a)$ für alle $f \in \mathcal{B}_\Sigma$.

3-Bit Verifier / Inner Verifier

Neuer 3-Bit Verifier:

Verifier

- Wähle zufällig $(x, y) \in E$

3-Bit Verifier / Inner Verifier

Neuer 3-Bit Verifier:

Verifier

- Wähle zufällig $(x, y) \in E$
- Wähle zufällig $f \in B_{\Sigma_X}, g \in B_{\Sigma_Y}$, mit $\Pr[f(\sigma) = 1] = 1/2 = \Pr[g(\sigma') = 1]$

3-Bit Verifier / Inner Verifier

Neuer 3-Bit Verifier:

Verifier

- Wähle zufällig $(x, y) \in E$
- Wähle zufällig $f \in B_{\Sigma_X}, g \in B_{\Sigma_Y}$, mit $\Pr[f(\sigma) = 1] = 1/2 = \Pr[g(\sigma') = 1]$
- Teste, ob $\mathrm{Long}_{\ell_X(x)}[f] \oplus \mathrm{Long}_{\ell_Y(y)}[g] \oplus \mathrm{Long}_{\ell_X(x)}[(g \circ \pi_{(x,y)}) \oplus f] = 0$

3-Bit Verifier / Inner Verifier

Neuer 3-Bit Verifier:

Verifier

- Wähle zufällig $(x, y) \in E$
- Wähle zufällig $f \in B_{\Sigma_X}, g \in B_{\Sigma_Y}$, mit $\Pr[f(\sigma) = 1] = 1/2 = \Pr[g(\sigma') = 1]$
- Teste, ob $\mathrm{Long}_{\ell_X(x)}[f] \oplus \mathrm{Long}_{\ell_Y(y)}[g] \oplus \mathrm{Long}_{\ell_X(x)}[(g \circ \pi_{(x,y)}) \oplus f] = 0$

3-Bit Verifier / Inner Verifier

Neuer 3-Bit Verifier:

Verifier

- Wähle zufällig $(x, y) \in E$
- Wähle zufällig $f \in B_{\Sigma_X}, g \in B_{\Sigma_Y}$, mit $\Pr[f(\sigma) = 1] = 1/2 = \Pr[g(\sigma') = 1]$
- Teste, ob $\mathrm{Long}_{\ell_X(x)}[f] \oplus \mathrm{Long}_{\ell_Y(y)}[g] \oplus \mathrm{Long}_{\ell_X(x)}[(g \circ \pi_{(x,y)}) \oplus f] = 0$

Completeness

Falls $\pi(\ell_X(x)) = \ell_Y(y)$:

3-Bit Verifier / Inner Verifier

Neuer 3-Bit Verifier:

Verifier

- Wähle zufällig $(x, y) \in E$
- Wähle zufällig $f \in B_{\Sigma_X}, g \in B_{\Sigma_Y}$, mit $\Pr[f(\sigma) = 1] = 1/2 = \Pr[g(\sigma') = 1]$
- Teste, ob $\mathrm{Long}_{\ell_X(x)}[f] \oplus \mathrm{Long}_{\ell_Y(y)}[g] \oplus \mathrm{Long}_{\ell_X(x)}[(g \circ \pi_{(x,y)}) \oplus f] = 0$

Completeness

Falls $\pi(\ell_X(x)) = \ell_Y(y)$:

Soundness

Idee: Wenn der Test mit zu hoher Wahrscheinlichkeit gelingt, gibt uns Long_ℓ ein Labeling, dass sehr viele Kanten erfüllt.
Beweis: Fourier-Analyse von Long_ℓ

Konsequenzen

Satz

Wir erhalten $c = 1 - \varepsilon$, $s = 1/2 + \varepsilon$. *Eine bessere Approximation als* $2 - \varepsilon$ *ist nicht möglich!*

Konsequenzen

Satz

Wir erhalten $c = 1 - \varepsilon$, $s = 1/2 + \varepsilon$. Eine bessere Approximation als $2 - \varepsilon$ ist nicht möglich!

Weiteres Fortführen der Ideen:

Satz (Johan Håstad, 2001)

Es gibt keinen Algorithmus, der MAX-3SAT besser als mit Rate $8/7$ approximieren kann. (außer $\mathcal{P} = \mathcal{NP}$)

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert
2. Baue einfachen Verifier V (Outer Verifier)

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert
2. Baue einfachen Verifier V (Outer Verifier)
3. Modifiziere V zu V^* (Inner Verifier)

Unique Games

UNIQUE-LABEL-COVER

Sei $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$ Instanz von LABEL-COVER. Sind alle π_e Permutationen, so ist es eine Instanz von UNIQUE-LABEL-COVER.

Unique Games

UNIQUE-LABEL-COVER

Sei $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$ Instanz von LABEL-COVER. Sind alle π_e Permutationen, so ist es eine Instanz von UNIQUE-LABEL-COVER.

Frage

Wie schwer ist UNIQUE-LABEL-COVER?

Unique Games

UNIQUE-LABEL-COVER

Sei $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$ Instanz von LABEL-COVER. Sind alle π_e Permutationen, so ist es eine Instanz von UNIQUE-LABEL-COVER.

Frage

Wie schwer ist UNIQUE-LABEL-COVER?

ε-GAP-MAX-UNIQUE-LABEL-COVER

Gegeben: UNIQUE-LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$

Gesucht:
- Sind mindestens $(1 - \varepsilon) \cdot |E|$ Kanten erfüllbar? (JA)
- Gilt für alle ℓ, dass höchstens $\varepsilon \cdot |E|$ Kanten gleichzeitig erfüllt sind? (NEIN)

Unique Games

UNIQUE-LABEL-COVER

Sei $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$ Instanz von LABEL-COVER. Sind alle π_e Permutationen, so ist es eine Instanz von UNIQUE-LABEL-COVER.

Frage

Wie schwer ist UNIQUE-LABEL-COVER?

ε-GAP-MAX-UNIQUE-LABEL-COVER

Gegeben: UNIQUE-LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$

Gesucht:

- Sind mindestens $(1 - \varepsilon) \cdot |E|$ Kanten erfüllbar? (JA)
- Gilt für alle ℓ, dass höchstens $\varepsilon \cdot |E|$ Kanten gleichzeitig erfüllt sind? (NEIN)

Unique Games Conjecture (Subhash Khot, 2002)

Für alle $\varepsilon > 0$ gibt es ein $s = s(\varepsilon)$, so dass ε-GAP-MAX-UNIQUE-LABEL-COVER $\mathcal{NP}$-hart für Instanzen mit Alphabetgröße s ist.

Unique Games Conjecture

Zustand 2005:

Name	Rate	LB	LB UGC
MIN-VERTEX-COVER	2	1.36	$2 - \varepsilon$
MAX-CUT	$1/\alpha_{GW} \approx 1.139$	$17/16 \approx 1.0625$	$1/\alpha_{GW} - \varepsilon$
MIN-K-UNIFORM-HYPERGRAPH-VC	k	$k - 1 - \varepsilon$	$k - \varepsilon$
MAX-2SAT	$1/\alpha_{LLZ} \approx 1.064$	1.048	$1/\alpha_{LLZ} - \varepsilon$
CSP mit integrality gap α	α	?	$\alpha - \varepsilon$

Unique Games Conjecture

Zustand 2005:

Name	Rate	LB	LB UGC
MIN-VERTEX-COVER	2	1.36	$2-\varepsilon$
MAX-CUT	$1/\alpha_{GW} \approx 1.139$	$17/16 \approx 1.0625$	$1/\alpha_{GW} - \varepsilon$
MIN-K-UNIFORM-HYPERGRAPH-VC	k	$k-1-\varepsilon$	$k-\varepsilon$
MAX-2SAT	$1/\alpha_{LLZ} \approx 1.064$	1.048	$1/\alpha_{LLZ} - \varepsilon$
CSP mit integrality gap α	α	?	$\alpha-\varepsilon$

Satz (Raghavendra, 2008)

Unter der UGC ist jedes CSP entweder $\mathcal{NP}$-hart oder in Polynomialzeit lösbar.

Ein Beispiel

MAX-CUT

Gegeben: Graph $G = (V, E)$

Gesucht: Partition $V = V_1 \dot\cup V_2$, so dass die Anzahl an Kanten zwischen V_1 und V_2 maximal ist.

Ein Beispiel

MAX-CUT

Gegeben: Graph $G = (V, E)$

Gesucht: Partition $V = V_1 \dot{\cup} V_2$, so dass die Anzahl an Kanten zwischen V_1 und V_2 maximal ist.

Satz (Khot, Kindler, Mossel, O'Donnel, 2005)

Wenn es einen $(\log n, 1)$*-Verifier V gibt mit Soundness s und Completeness c gibt, der genau zwei Bits* b, b' *liest und akzeptiert, falls* $b \neq b'$*, so impliziert die UGC, dass MAX-CUT nicht besser als mit Wert* s/c *approximiert werden kann.*

Ein Beispiel

MAX-CUT

Gegeben: Graph $G = (V, E)$

Gesucht: Partition $V = V_1 \dot\cup V_2$, so dass die Anzahl an Kanten zwischen V_1 und V_2 maximal ist.

Satz (Khot, Kindler, Mossel, O'Donnel, 2005)

Wenn es einen $(\log n, 1)$-Verifier V gibt mit Soundness s und Completeness c gibt, der genau zwei Bits b, b' liest und akzeptiert, falls $b \neq b'$, so impliziert die UGC, dass MAX-CUT nicht besser als mit Wert s/c approximiert werden kann.

Beweis

- π Beweis für V

Ein Beispiel

MAX-CUT

Gegeben: Graph $G = (V, E)$

Gesucht: Partition $V = V_1 \dot{\cup} V_2$, so dass die Anzahl an Kanten zwischen V_1 und V_2 maximal ist.

Satz (Khot, Kindler, Mossel, O'Donnel, 2005)

Wenn es einen $(\log n, 1)$*-Verifier* V *gibt mit Soundness* s *und Completeness* c *gibt, der genau zwei Bits* b, b' *liest und akzeptiert, falls* $b \neq b'$*, so impliziert die UGC, dass MAX-CUT nicht besser als mit Wert* s/c *approximiert werden kann.*

Beweis

- π Beweis für V
- Konstruiere $G = (V, E)$

Ein Beispiel

MAX-CUT

Gegeben: Graph $G = (V, E)$

Gesucht: Partition $V = V_1 \dot\cup V_2$, so dass die Anzahl an Kanten zwischen V_1 und V_2 maximal ist.

Satz (Khot, Kindler, Mossel, O'Donnel, 2005)

Wenn es einen $(\log n, 1)$-Verifier V gibt mit Soundness s und Completeness c gibt, der genau zwei Bits b, b' liest und akzeptiert, falls $b \neq b'$, so impliziert die UGC, dass MAX-CUT nicht besser als mit Wert s/c approximiert werden kann.

Beweis

- π Beweis für V
- Konstruiere $G = (V, E)$
- $V = \{v_1, v_2, \ldots, v_{|\pi|}\}$

Ein Beispiel

MAX-CUT

Gegeben: Graph $G = (V, E)$

Gesucht: Partition $V = V_1 \dot\cup V_2$, so dass die Anzahl an Kanten zwischen V_1 und V_2 maximal ist.

Satz (Khot, Kindler, Mossel, O'Donnel, 2005)

Wenn es einen $(\log n, 1)$-Verifier V gibt mit Soundness s und Completeness c gibt, der genau zwei Bits b, b' liest und akzeptiert, falls $b \neq b'$, so impliziert die UGC, dass MAX-CUT nicht besser als mit Wert s/c approximiert werden kann.

Beweis

- π Beweis für V
- Konstruiere $G = (V, E)$
- $V = \{v_1, v_2, \ldots, v_{|\pi|}\}$
- $E = \{(v_i, v_j) \mid \Pr[V \text{ liest Bits } i \text{ und } j] > 0\}$

Ein Beispiel

MAX-CUT

Gegeben: Graph $G = (V, E)$

Gesucht: Partition $V = V_1 \dot\cup V_2$, so dass die Anzahl an Kanten zwischen V_1 und V_2 maximal ist.

Satz (Khot, Kindler, Mossel, O'Donnel, 2005)

Wenn es einen $(\log n, 1)$-Verifier V gibt mit Soundness s und Completeness c gibt, der genau zwei Bits b, b' liest und akzeptiert, falls $b \neq b'$, so impliziert die UGC, dass MAX-CUT nicht besser als mit Wert s/c approximiert werden kann.

Beweis

- π Beweis für V
- Konstruiere $G = (V, E)$
- $V = \{v_1, v_2, \ldots, v_{|\pi|}\}$
- $E = \{(v_i, v_j) \mid \Pr[V \text{ liest Bits } i \text{ und } j] > 0\}$
- $\Pr[V \text{ akzeptiert}] = \frac{\text{Anzahl der Kanten im Cut}}{\text{Anzahl der Kanten}}$

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot\cup Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot\cup Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}, \pi' = \pi_{(x,y')}$

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot\cup Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}, \pi' = \pi_{(x,y')}$
- Lies $\ell(x), \ell(y), \ell(y')$ aus dem Beweis

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}, \pi' = \pi_{(x,y')}$
- Lies $\ell(x), \ell(y), \ell(y')$ aus dem Beweis
- Teste, ob $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}, \pi' = \pi_{(x,y')}$
- Lies $\ell(x), \ell(y), \ell(y')$ aus dem Beweis
- Teste, ob $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}, \pi' = \pi_{(x,y')}$
- Lies $\ell(x), \ell(y), \ell(y')$ aus dem Beweis
- Teste, ob $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$

Wenn $(1 - \varepsilon) \cdot |E|$ der Kanten erfüllt, ist die Wahrscheinlichkeit, dass beide erfüllt sind, mindestens $1 - 2\varepsilon$.

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot{\cup} Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}, \pi' = \pi_{(x,y')}$
- Lies $\ell(x), \ell(y), \ell(y')$ aus dem Beweis
- Teste, ob $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$

Wenn $(1 - \varepsilon) \cdot |E|$ der Kanten erfüllt, ist die Wahrscheinlichkeit, dass beide erfüllt sind, mindestens $1 - 2\varepsilon$.
Sind maximal $\varepsilon \cdot |E|$ der Kanten erfüllt, ist die Wahrscheinlichkeit, dass beide erfüllt sind, höchstens ε.

Outer Verifier

UNIQUE-LABEL-COVER Instanz $G = (X \dot\cup Y, E), \Sigma, \Pi = \{\pi_e\}_{e \in E}$
Konstruiere typischen Verifier für UNIQUE-LABEL-COVER:

Verifier

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}, \pi' = \pi_{(x,y')}$
- Lies $\ell(x), \ell(y), \ell(y')$ aus dem Beweis
- Teste, ob $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$

Wenn $(1 - \varepsilon) \cdot |E|$ der Kanten erfüllt, ist die Wahrscheinlichkeit, dass beide erfüllt sind, mindestens $1 - 2\varepsilon$.
Sind maximal $\varepsilon \cdot |E|$ der Kanten erfüllt, ist die Wahrscheinlichkeit, dass beide erfüllt sind, höchstens ε.
Wir lesen $3 \cdot \log(|\Sigma|)$ Bits!

2-Bit Verifier / Inner Verifier

Idee

Wir codieren $\ell(x), \ell(y), \ell(y')$ so, dass wir für den Test $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$ nur zwei Bits lesen müssen!

2-Bit Verifier / Inner Verifier

Idee

Wir codieren $\ell(x), \ell(y), \ell(y')$ so, dass wir für den Test $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$ nur zwei Bits lesen müssen!

Auch hier wieder Long Code!

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi^{-1}_{(x,y)}, \pi' = \pi^{-1}_{(x,y')}$

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}^{-1}, \pi' = \pi_{(x,y')}^{-1}$
- Wähle $f \in B_{|\Sigma|}$, so dass $\Pr[f(\sigma) = 0] = 1/2$

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi^{-1}_{(x,y)}, \pi' = \pi^{-1}_{(x,y')}$
- Wähle $f \in B_{|\Sigma|}$, so dass $\Pr[f(\sigma) = 0] = 1/2$
- Wähle $f_\rho \in B_{|\Sigma|}$, so dass $\Pr[f_\rho(\sigma) = 0] = 1/2 + 1/2\rho > 1/2$

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi^{-1}_{(x,y)}, \pi' = \pi^{-1}_{(x,y')}$
- Wähle $f \in B_{|\Sigma|}$, so dass $\Pr[f(\sigma) = 0] = 1/2$
- Wähle $f_\rho \in B_{|\Sigma|}$, so dass $\Pr[f_\rho(\sigma) = 0] = 1/2 + 1/2\rho > 1/2$
- Teste, ob $\mathrm{Long}_{\ell(y)}[f \circ \pi] = \mathrm{Long}_{\ell(y')}[(f \circ \pi') \cdot f_\rho]$

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi^{-1}_{(x,y)}, \pi' = \pi^{-1}_{(x,y')}$
- Wähle $f \in B_{|\Sigma|}$, so dass $\Pr[f(\sigma) = 0] = 1/2$
- Wähle $f_\rho \in B_{|\Sigma|}$, so dass $\Pr[f_\rho(\sigma) = 0] = 1/2 + 1/2\rho > 1/2$
- Teste, ob $\mathrm{Long}_{\ell(y)}[f \circ \pi] = \mathrm{Long}_{\ell(y')}[(f \circ \pi') \cdot f_\rho]$

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi^{-1}_{(x,y)}, \pi' = \pi^{-1}_{(x,y')}$
- Wähle $f \in B_{|\Sigma|}$, so dass $\Pr[f(\sigma) = 0] = 1/2$
- Wähle $f_\rho \in B_{|\Sigma|}$, so dass $\Pr[f_\rho(\sigma) = 0] = 1/2 + 1/2\rho > 1/2$
- Teste, ob $\mathrm{Long}_{\ell(y)}[f \circ \pi] = \mathrm{Long}_{\ell(y')}[(f \circ \pi') \cdot f_\rho]$

Completeness

Falls $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$:

2-Bit Verifier / Inner Verifier

Verifier mit Parameter $0 < \rho < 1$

- Wähle $x \in X$ uniform
- Wähle zufällig $(x, y), (x, y') \in E$
- Sei $\pi = \pi_{(x,y)}^{-1}, \pi' = \pi_{(x,y')}^{-1}$
- Wähle $f \in B_{|\Sigma|}$, so dass $\Pr[f(\sigma) = 0] = 1/2$
- Wähle $f_\rho \in B_{|\Sigma|}$, so dass $\Pr[f_\rho(\sigma) = 0] = 1/2 + 1/2\rho > 1/2$
- Teste, ob $\mathrm{Long}_{\ell(y)}[f \circ \pi] = \mathrm{Long}_{\ell(y')}[(f \circ \pi') \cdot f_\rho]$

Completeness

Falls $\pi(\ell(x)) = \ell(y)$ und $\pi'(\ell(x)) = \ell(y')$:

Soundness

Idee: Wenn der Test mit zu hoher Wahrscheinlichkeit gelingt, gibt uns Long_ℓ ein Labeling, dass sehr viele Kanten erfüllt.
Beweis: Fourier-Analyse von Long_ℓ und "Majority is Stablest"

Konsequenzen

Satz

Wir erhalten $c = 1/2 + 1/2\rho$ *und* $s = \arccos(\rho)/\pi + \varepsilon$*. Eine bessere Approximation als* $s/c \approx 1/\alpha_{GW} - \varepsilon \approx 1.139 - \varepsilon$ *ist nicht möglich.*
$(\alpha_{GW} = \min_{0<\Theta\leq\pi} \frac{2}{\pi} \cdot \frac{\Theta}{1-\cos\Theta})$

Konsequenzen

Satz

Wir erhalten $c = 1/2 + 1/2\rho$ *und* $s = \arccos(\rho)/\pi + \varepsilon$. *Eine bessere Approximation als* $s/c \approx 1/\alpha_{GW} - \varepsilon \approx 1.139 - \varepsilon$ *ist nicht möglich.*
$(\alpha_{GW} = \min_{0<\Theta\leq\pi} \frac{2}{\pi} \cdot \frac{\Theta}{1-\cos\Theta})$

Satz (Goemans & Williamson, 1995)

Es gibt einen randomisierten Approximationsalgorithmus für $MAX - CUT$ *mit Approximationsrate* $1/\alpha_{GW} \approx 1.139$.

Konsequenzen

Satz

Wir erhalten $c = 1/2 + 1/2\rho$ *und* $s = \arccos(\rho)/\pi + \varepsilon$*. Eine bessere Approximation als* $s/c \approx 1/\alpha_{GW} - \varepsilon \approx 1.139 - \varepsilon$ *ist nicht möglich.* $(\alpha_{GW} = \min_{0<\Theta\leq\pi} \frac{2}{\pi} \cdot \frac{\Theta}{1-\cos\Theta})$

Satz (Goemans & Williamson, 1995)

Es gibt einen randomisierten Approximationsalgorithmus für $MAX - CUT$ *mit Approximationsrate* $1/\alpha_{GW} \approx 1.139$.

Satz (Mahajan, Ramesh, 1995)

Es gibt einen deterministischen Approximationsalgorithmus für $MAX - CUT$ *mit Approximationsrate* $1/\alpha_{GW} \approx 1.139$.

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert
2. Baue einfachen Verifier V (Outer Verifier)

Allgemeines Vorgehen

1. Zeige, dass die Existenz eines bestimmten Verifiers V^* die Nicht-Approximierbarkeit impliziert
2. Baue einfachen Verifier V (Outer Verifier)
3. Modifiziere V zu V^* (Inner Verifier)