Universidade de São Paulo
Instituto de Matemática e Estatística
Bacharelado em Ciência da Computação

Sobre a Conjectura dos Jogos Únicos

Lucas Carvalho Daher

Monografia Final

mac 499 — Trabalho de
Formatura Supervisionado

Supervisora: Profª. Drª. Yoshiko Wakabayashi

São Paulo
2 de dezembro de 2019

Je veux m'abstraire

— Paul Verlaine

Agradecimentos

Agradeço a todos que me acompanharam e incentivaram nessa incursão pelo mundo da computação, em especial a minha professora e orientadora Yoshiko, aos professores Carlinhos, Coelho e Daniel e aos amigos Colombo e Breno.

Resumo

Lucas Carvalho Daher. **Sobre a Conjectura dos Jogos Únicos**. Monografia (Bacharelado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2019.

O texto apresenta uma introdução à Conjectura dos Jogos Únicos e suas aplicações na área de limites de inaproximabilidade. Especificamente, é apresentada uma prova de que para o problema do corte máximo o algoritmo de Goemans-Williamson é ótimo caso a conjectura seja verdadeira. Além disso, outros temas necessários para a compreensão do assunto são apresentados. O objetivo é que o trabalho seja auto contido, ou perto disso, para um aluno de graduação. Ou seja, com os conhecimentos que ele já possui e os apresentados aqui, ele deve compreender o texto todo pesquisando minimamente em outras fontes.

Palavras-chave: Unique Games Conjecture. Max Cut.

Abstract

Lucas Carvalho Daher. **About the Unique Games Conjecture**. Capstone Project Report (Bachelor). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2019.

This work presents an introduction to the Unique Games Conjecture and some of its applications in the field of inapproximability bounds. The main focus is presenting proof that the Goemans-Williamson algorithm is optimal for the MaxCut problem if the conjecture is true. We also present other topics necessary to understand the main result, aiming to make this work as self-contained as possible for an undergrad student.

Keywords: Unique Games Conjecture. Max Cut.

Sumário

Capítulo 1

Preliminares

Aqui definimos alguns problemas citados ao longo do texto com os quais alguns leitores talvez não estejam familiarizados:

Corte Máximo (MaxCut): Dado um grafo $G = (V, E)$, encontrar um corte Máximo, ou seja, encontrar um conjunto $S \subseteq V$ tal que o nÃºmero de arestas com um extremo em S e outro em seu complemento seja Máximo. (A versão com pesos nas arestas, procura um corte de peso Máximo.)

Cobertura Mínima (MinCov): Dado um grafo $G = (V, E)$, encontrar uma cobertura Mínima de vértices, ou seja, um conjunto $W \subseteq V$ de cardinalidade Mínima tal que toda aresta de G tenha pelo menos um extremo em W.

Max2Sat (Max2Sat): Dada uma fórmula booleana na forma normal conjuntiva, na qual cada cláusula contém no Máximo dois literais, encontrar uma valoração (Verdadeiro/Falso) para as variáveis que satisfaça o maior número de cláusulas.

Subgrafo Acíclico Máximo (MaxAcyclic): Dado um grafo $G = (V, E)$, com pesos nas arestas, encontrar em G um subgrafo acíclico cujas arestas tenham um peso total Máximo.

Capítulo 2

Introdução

O objetivo desse trabalho é introduzir o leitor a um dos maiores avanços recentes sobre razão de inaproximabilidade de vários problemas de otimização combinatória. Trata-se da **Unique Games Conjecture (UGC)**, que possibilita provar resultados fortes de inaproximabilidade, sob a hipótese de que a conjectura é verdadeira.

Este texto se destina a alunos no final da graduação ou na pós-graduação que tenham alguma base em teoria da computação, mas não dos tópicos aqui apresentados. Conhecimentos sobre os assuntos apresentados nas disciplinas MAC0450 ou MAC5727 (algoritmos de aproximação) ou equivalente são recomendados. Ao leitor não familiarizado com esse assunto, indicamos o texto Carvalho *et al.*, 2001.

A UGC, proposta em Khot, 2002, gerou resultados muito fortes de inaproximabilidade para vários problemas clássicos de otimização combinatória (alguns definidos logo abaixo). Dentre esses, mencionamos alguns exemplos notórios. Para todos esses problemas, a melhor razão de aproximação conhecida coincide com o limiar obtido via a UGC, que mencionamos a seguir.

- Razão de inaproximabilidade do problema do Corte Máximo (MaxCut) pode ser $\alpha_{MC} - \epsilon$, provado por Khot, Kindler *et al.*, 2007. O algoritmo com a melhor razão de aproximação conhecida, α_{MC}, foi obtido por Goemans e Williamson, 1995.
- razão de inaproximabilidade do problema Cobertura Mínima (MinCov) pode ser $2 - \epsilon$, provado por Khot e Regev, 2008. O algoritmo com a melhor razão de aproximação conhecida, 2, foi obtido por Bar-Yehuda e Even, 1981.
- razão de inaproximabilidade do problema Max2Sat pode ser $\alpha_{LLZ} - \epsilon$, provado por Austrin, 2007. O algoritmo com a melhor razão de aproximação que se conhece, α_{LLZ}, foi obtido por Lewin *et al.*, 2002.

- razão de inaproximabilidade do problema do Subgrafo Acíclico Máximo (MaxAcyclic) pode ser $2 - \epsilon$, provado por Guruswami *et al.*, 2008. A melhor razão de aproximação conhecida é 2 (resultado trivial).

Note o uso de *pode ser*, pois se trata de uma conjectura e se ela for provada ser falsa, esses resultados de inaproximabilidade não valem. (Claramente, se for provado que $P = NP$, esses resultados se tornam irrelevantes.)

2.1 O problema do Corte Máximo

O Problema do Corte Máximo é um problema clássico de otimização combinatória, provado ser NP-completo no primeiro artigo em que uma lista de 21 problemas desse tipo foi apresentada por Karp, 1972. Os últimos avanços sobre este problema, antes do resultado de Khot, Kindler *et al.*, 2007, foram:

- Algoritmo de aproximação proposto por Goemans e Williamson, 1995, que possui a melhor razão que conhecemos, $\alpha_{gw} \approx 0.878$ e que será provada ser ótima no Capítulo 7 (supondo a UGC).
- O melhor limite de inaproximabilidade que foi provado, supondo que a UGC seja falsa, é $\frac{16}{17} \approx 0.941$, provado por Håstad, 2001.

2.2 Visão geral do trabalho

Nesse trabalho nosso foco é provar uma razão de inaproximabilidade para o MaxCut, seguindo o modelo clássico de utilizar a UGC em conjunto com o sistema PCP (Probabilistically Checkable Proof), como é feita a maioria das provas de resultados de inaproximabilidade que dependem da UGC.

No Capítulo 3 apresentamos a UGC, e no Capítulo 4, apresentamos o PCP. Tópicos importantes para a prova de inaproximabilidade são apresentados nos Capítulos 5 e 6. Por fim, no Capítulo 7 provamos o resultado central sobre o MaxCut.

Capítulo 3

Unique Games Conjecture

3.1 Notas Iniciais

Essa conjectura foi proposta em Khot, 2002 e ainda está em aberto.

3.2 Unique Label Cover

O problema Unique Label Cover é central para a definição da conjectura. Nesse problema, temos um grafo bipartido, um conjunto de rótulos e uma restrição associada a cada aresta do grafo. Cada restrição vem na forma de uma bijeção. Isso implica que se temos, por exemplo, uma aresta uv e damos um rótulo x para a vértice u, existe um e apenas um rótulo que podemos dar para o vértice v que satisfaz a restrição. Definimos formalmente:

Definição 3.1. Sejam $G = (V \cup W, E)$ um grafo bipartido, $[M] := \{1, 2, \dots, M\}$ o universo de possíveis rótulos e σ um conjunto de bijeções $\sigma_{vw} : [M] \longrightarrow [M]$ para todo $vw \in E$. Definimos uma instância do Unique Label Cover como $\mathcal{L}(G, [M], \sigma)$. Queremos atribuir rótulos $l(v) : [M] \longrightarrow [M]$, para todo $v \in V \cup W$ de modo a satisfazer o máximo de arestas possíveis. Uma aresta vw é satisfeita se e somente se:

$$\sigma_{vw}(l(v)) = l(w).$$

Para uma instância $\mathcal{L}(G, [M], \sigma)$ dizemos que uma solução l^* é ótima se o número de arestas satisfeitas por l^* é maior ou igual ao número de arestas satisfeitas por qualquer outra solução l.

3.3 Unique Games Conjecture

Utilizamos o Unique Label Cover para definir a UGC:

Conjectura 3.2. Sejam $\eta,\ \gamma > 0$ números reais quaisquer. Existe $M = M(\eta,\ \gamma)$ tal que é NP-difícil distinguir se a solução ótima de uma instância do Unique Label Cover com conjunto de rótulos de tamanho M satisfaz uma fração de pelo menos $1 - \eta$ ou no máximo γ.

Uma consequência direta da conjectura é que para $\eta,\ \gamma$ quaisquer não existe um algoritmo polinomial com razão de aproximação maior ou igual a $\frac{1-\eta}{\gamma}$. Como podemos escolher η e γ arbitrariamente, concluímos que não existem algoritmos de aproximação para o Unique Label Cover com razão constante, ou seja, o problema não está na classe APX.

3.4 ULC-gap

Apresentamos agora uma variação do problema Unique Label Cover, que chamaremos de ULC-gap. Nessa versão, recebemos uma instância em que ou uma fração muito grande das arestas ou muito pequena é satisfeita e queremos apenas distinguir entre elas. Formalmente:

Definição 3.3. Sejam $\mathcal{L}(G, [M], \sigma)$ uma instância do Unique Label Cover e $\eta,\ \gamma > 0$. Queremos saber se uma solução ótima de $\mathcal{L}$ satisfaz uma fração de pelo menos $1 - \eta$ ou no máximo γ das arestas.

Capítulo 4

Provas Interativas

4.1 Notas iniciais

Essa sessão segue de perto Kohayakawa e Soares, 1995 com algumas adaptações para a linguagem e o uso que daremos nos próximos capítulos. Inicialmente, introduzimos a classe IP para familiarizar o leitor com o assunto e depois definimos a classe PCP, que será fundamental para provar a inaproximabilidade do MaxCut.

4.2 A classe IP

4.2.1 Intuição informal

Seja L um problema de decisão, ou seja, um problema em que para toda instância I temos duas respostas possíveis: sim ou não. Agora, suponhamos um jogo entre dois jogadores, que chamaremos de P (provador) e V (verificador), inicialmente eles recebem uma instância I do problema e P quer *provar* para V que a resposta para a instância é sim e V decide se acredita ou não na prova, ou seja, *verifica* se a prova está correta e decide se a resposta correta é sim ou não.

Esse jogo possui algumas particularidades: P possui capacidade de processamento e memória infinitas, enquanto V possui capacidades polinomiais. Ambos os jogadores têm acesso à instância do problema e cada jogador possui sua saída onde pode escrever e o outro jogador ler. V também possui acesso a um gerador de bits aleatórios.

O jogo se desenrola em rodadas, V faz o primeiro movimento: ele pode consultar o gerador de bits aleatórios e a instância I e, após isso, escreve algo em sua saída (ou nada). Após V fazer seu movimento, P faz o seu: ele pode ler o que foi escrito por V, consultar I e

então responder. E assim por diante. Após algum número de rodadas, V toma sua decisão sobre a resposta e o jogo termina.

Intuitivamente, podemos pensar que P possui uma prova de que a saída deve ser SIM e a cada rodada V pede informações sobre essa prova. Após receber essas informações, V pode se decidir positivamente, negativamente ou pedir mais informações.

Sempre que a instância é positiva, P consegue convencer V, ou seja, nunca há falsos negativos. Porém, quando a instância é negativa, P consegue enganar V uma fração das vezes, outra V não é convencido e responde NÃO.

4.2.2 Definição formal

Formalmente, temos que V e P são máquinas de Turing, sendo que V é limitada polinomialmente nas operações que pode realizar, enquanto P não. Ambas as máquinas têm acesso de leitura à fita de entrada, que apresenta a instância I. V também pode ler de uma fita de bits aleatórios e ler e escrever em sua fita de saída, que chamaremos de F_V. P pode apenas ler de F_V, mas pode escrever em sua própria fita F_P, que é lida por V.

Definição 4.1. Sejam L um problema de decisão, I uma instância qualquer de L de tamanho $|I| := n$ e $t(n)$ uma função. L faz parte da classe $IP(t(n))$ se e somente se:

- V e P trocam no máximo $t(n)$ mensagens utilizando as fitas F_V e F_P. Ou seja, o jogo acaba em no máximo $t(n)$ rodadas;
- sempre que a resposta correta for SIM, V será convencido disso;
- sempre que a resposta correta for NÃO, V será enganado por P uma fração de no máximo σ das vezes, para algum $\sigma < 1$;
- V escreve a primeira mensagem;

4.3 O sistema PCP

4.3.1 Intuição informal

No sistema PCP, temos uma situação parecida à descrita acima, com algumas diferenças. O verificador V continua sendo uma máquina de Turing, mas o provador será substituído por um oráculo, que possui um certificado, que chamaremos de $\Pi := \pi_1\pi_2 \ldots \pi_{|\Pi|}, \pi_i \in \{0, 1\}$, de que a resposta correta é SIM. Também continuamos com uma instância I de um problema L de decisão.

Agora, V possui acesso apenas a um gerador de bits aleatórios, que será utilizado para escolher posições em Π e a uma quantidade de bits de Π apenas uma vez. Ou seja, o jogo possui apenas uma rodada: inicialmente V pede os bits em posições específicas de Π, o oráculo retorna os valores e V já deve decidir a resposta correta.

Uma prova no sistema é caracterizada por dois valores: a quantidade de bits pedida ao gerador e a quantidade pedida ao oráculo.

4.3.2 Definição formal

Definição 4.2. Sejam V o verificador, I uma instância de um problema de decisão L, τ um gerador de números aleatórios e Π uma prova produzida pelo oráculo. Definimos uma instância pela tripla $V(I,\ \tau,\ \Pi)$.

Seja $\mathcal{P}_\tau$ a probabilidade vinculada ao gerador τ (único elemento não determinístico do sistema), então:

- para todo I cuja resposta é positiva, temos que $\mathcal{P}_\tau(V(I,\ \tau,\ \Pi) = \text{SIM}) = 1$;
- para todo I cuja resposta é negativa, temos que $\mathcal{P}_\tau(V(I,\ \tau,\ \Pi) = \text{SIM}) < \sigma$, para algum $\sigma < 1$;

Para caracterizar formalmente a quantidade de bits pedida ao oráculo e a τ, definimos duas funções reais $r(n)$ e $q(n)$ sobre os inteiros não negativos e dizemos que V é $(r(n),\ p(n))$-restrito se para $r'(n) = \mathcal{O}(r(n))$ e $p' = \mathcal{O}(p(n))$ quaisquer, a quantidade de bits que V pede a τ é limitada por $r'(n)$ e, analogamente, a quantidade de bits que V pede ao oráculo é limitada por $p'(n)$.

4.3.3 A classe NP

Em um dos resultados mais fortes da área de teoria da complexidade, foi provado por Arora *et al.*, 1998 que:

Teorema 4.3. $PCP(log\ n,\ 1) = NP$

Ou seja, a classe de problemas NP é equivalente à classe de problemas PCP em que o verificador pede um número de bits aleatórios para τ da ordem de $\mathcal{O}(\log\ n)$ e lê um número constante de bits de prova Π.

Capítulo 5

Códigos

5.1 Notas iniciais

Aqui vamos fazer uma rápida introdução ao tema da teoria de códigos, especificamente para apresentar o conceito de long codes, que será usada na prova de inaproximabilidade do max cut.

5.2 Hadamard Codes

Antes de falar sobre long codes, porém, é necessário nos familiarizarmos com Hadamard codes.

Definimos $x \in \{-1, 1\}^k$ e $y \in \{-1, 1\}^k$ como mensagens binárias de tamanho $|k|$. Usamos a notação ± 1, pois ela será mais conveniente adiante, porém, agora devemos tratar os vetores como booleanos em que -1 é equivalente a verdadeiro e 1, a falso.

Definimos o produto interno $\langle x, y \rangle : \{-1, 1\}^k \times \{-1, 1\}^k \longrightarrow \{-1, 1\}$ como:

$$\langle x, y \rangle = \bigvee_{i=1}^{k} x_i \wedge y_i$$

Agora definimos $Had(x)$ como o produto interno de x com todos os y possíveis:

Definição 5.1. $Had(x) = (\langle x, y \rangle)_{y \in \{-1,1\}^k}$

5.3 Long Codes

Agora, queremos definir o long code de $a \in \Sigma = \{1,\ 2, \dots, k\}$.

Primeiramente definimos e_a, tal que $|e_a| = k$ e

$$(e_a)_i = \begin{cases} -1, \text{ se } i \neq a \\ 1, \text{ caso contrário} \end{cases}$$

E finalmente:

Definição 5.2. $LONG(a) = Had(e_a)$

Note que com um long code mapeamos um elemento de um alfabeto de tamanho k, que pode ser pensado como equivalente a um vetor de tamanho $log_2(k)$, para um vetor de tamanho 2^k

5.4 Exemplo

Vamos agora mostrar um exemplo de long code: suponha um simples alfabeto $\{a,\ b,\ c\}$ e queremos calcular $LONG(b)$. Calculamos $e_b = \{-1,\ 1,\ -1\}$ e precisamos calcular o produto interno de e_b com os oito seguintes vetores:

$$\{-1, -1, -1\}$$
$$\{-1, -1,\ \ 1\}$$
$$\{-1,\ \ 1, -1\}$$
$$\{-1,\ \ 1,\ \ 1\}$$
$$\{\ \ 1, -1, -1\}$$
$$\{\ \ 1, -1,\ \ 1\}$$
$$\{\ \ 1,\ \ 1, -1\}$$
$$\{\ \ 1,\ \ 1,\ \ 1\}$$

Após algumas contas chegamos no seguinte resultado:

$$\begin{aligned} LONG(b) = \{&-1,\\ &-1,\\ &-1,\\ &-1,\\ &-1,\\ &1,\\ &-1,\\ &1\} \end{aligned}$$

5.5 Decodificando um long code

Queremos agora decodificar um long code. Especificamente, dado um long code $v \in \{-1,\ 1\}^k$ e uma letra $l \in \Sigma = \{1,\ 2, \dots, k\}$, queremos saber se v codifica l.

Supondo que estamos apenas analisando long codes válidos, basta olharmos uma e apenas uma posição de v para decidirmos se ele codifica l ou não: v codifica l se e somente se $v_l = 1$.

Definição 5.3. Sejam $v \in \{-1,\ 1\}^k$ e $l \in \Sigma = \{1,\ 2, \dots, k\}$. Definimos $f_l : \{-1,\ 1\}^k \longrightarrow \{-1,\ 1\}$ tal que:

$$f_l(v) := v_l$$

Capítulo 6

Majority Is Stablest

6.1 Notas Iniciais

Nesse capítulo apresentaremos os conceitos de estabilidade e influência, além de alguma matemática utilizando coeficientes de Fourier e chegaremos no principal: o Teorema Majority is Stablest (MIS). Este foi apresentado em Khot, Kindler *et al.*, 2007 (na versão original do artigo de 2004) como uma conjectura e provado em Mossel *et al.*, 2005.

Cabe dizer aqui que as provas para o MIS e sua extensão não serão apresentadas, pois são extensas, complexas e fogem completamente do escopo desse trabalho.

6.2 Estabilidade

Intuitivamente, queremos ter uma medida de quão estável uma função é a ruídos, isto é, supondo que um ruído possa afetar independentemente cada bit da entrada de uma função, qual a probabilidade desse ruído afetar a saída da função. Primeiramente definimos ruído:

Definição 6.1. Seja $\rho \in [-1, 1]$ e $x \in \{-1,\ 1\}^N$. Definimos ruído como uma função $g : \{-1,\ 1\}^N \times [-1, 1] \longrightarrow \{-1,\ 1\}^N$ tal que, se $y = g(x, \rho)$, então $E[x_i y_i] = \rho$, para todo $1 \leq i \leq N$.

De maneira equivalente, podemos pensar que $Pr(x_i \neq y_i) = \frac{1-\rho}{2}$.

Agora definimos estabilidade:

Definição 6.2. Sejam uma função $f : \{-1, 1\}^N \longrightarrow \mathcal{R}$, $x \in \{-1, 1\}^N$ uniformemente aleatório, $-1 \leq \rho \leq 1$ e $y = g(x, \rho)$. Definimos a estabilidade como:

$$\mathcal{S}_\rho(f) = E[f(x)f(y)].$$

6.3 Influência

Voltamos nossa atenção agora a funções $f : \{-1, 1\}^N \longrightarrow \{-1, 1\}$, que nesse trabalho serão utilizadas no contexto de decodificar long codes (ver 5.5).

Seja $x \in \{-1, 1\}^N$ um vetor qualquer. Queremos saber qual a influência de cada x_i, $1 \leq i \leq N$ em $f(x)$, ou seja, como a saída de $f(x)$ é alterada caso x_i se altere. Por exemplo, no caso de uma função f_l que decodifica um long code, para todo x_i com $i \neq l$, a influência de x_i é zero, ou seja, o valor de x_i nunca altera o resultado de $f_l(x)$.

Definição 6.3. Seja $f : \{-1, 1\}^N \longrightarrow \{-1, 1\}$, a influência da posição i, que denotaremos por Inf_i é:

$$Inf_i := Pr_x[f(x) \neq f(x^{\oplus i})].$$

Sendo que:

Definição 6.4. $x^{\oplus i} \in \{-1, 1\}^N$ é tal que:

$$(x^{\oplus i})_j = \begin{cases} x_j, & i \neq j \\ -x_j, & i = j \end{cases}$$

Isto é, invertemos o bit da posição i.

Podemos pensar que Inf_i indica a probabilidade de x_i ser decisivo no valor de $f(x)$ quando escolhemos um x qualquer.

6.4 MIS

Agora vamos unir os conceitos de estabilidade e influência para enunciar o Teorema Majority is Stablest (MIS).

Primeiramente, devemos definir a função maioria:

Definição 6.5. Para um $N \geq 1$ qualquer, definimos majority como uma função $f : \{-1, 1\}^N \longrightarrow \{-1, 1\}$ tal que:

$$f(x) = \begin{cases} 1, & \Sigma_{i=1}^{N} x_i \geq 0 \\ -1, & \text{caso contrário.} \end{cases}$$

Ou seja, é 1 se a maioria dos elementos do vetor de entrada for 1, ou em caso de empate; e é −1 em caso contrário.

Agora estamos prontos para enunciar o teorema. Intuitivamente, ele nos diz que a função balanceada mais estável em que cada elemento do vetor de entrada tem influência "pequena" é a função majority. Formalmente definimos:

Definição 6.6. Sejam $0 \leq \rho < 1$ e $\epsilon > 0$, então existe algum $\tau = \tau(\rho, \epsilon) > 0$, tal que para qualquer $f : \{-1, 1\}^N \longrightarrow \{-1, 1\}$, se $Inf_i < \tau$ para todo $1 \leq i \leq N$ e $E[f] = 0$, então:

$$Stab_\rho(f) \leq 1 - \frac{2}{\pi} cos^{-1}\rho + \epsilon.$$

Quando N tende ao infinito, a estabilidade de majority tende para $1 - \frac{2}{\pi} cos^{-1}\rho$, daí o nome do teorema.

A partir de 6.6 deduzimos o seguinte corolário:

Corolário 6.7. Sejam $-1 < \rho \leq 0$ e $\epsilon > 0$, então existe algum $\tau = \tau(\rho, \epsilon) > 0$, tal que para qualquer $f : \{-1, 1\}^N \longrightarrow \{-1, 1\}$, se $Inf_i < \tau$ para todo $1 \leq i \leq N$ e $E[f] = 0$, então:

$$\frac{1}{2} - \frac{1}{2} Stab_\rho(f) < \frac{cos^{-1}\rho}{\pi} + \epsilon.$$

6.5 Coeficientes de Fourier

Trabalharemos muito com funções do tipo $\{-1, 1\}^N \longrightarrow \{-1, 1\}$, então agora iremos deduzir algumas propriedades úteis de tais funções, especificamente como decompô-las em coeficientes de Fourier. E encerraremos o capítulo utilizando coeficientes de Fourier em uma extensão do MIS.

Consideremos o espaço vetorial $\mathcal{F}$ com todas as funções que nos interessam:

$$\mathcal{F} = \{f \mid f : \{-1, 1\}^N \longrightarrow \{-1, 1\}\}.$$

Esse espaço tem 2^N dimensões e uma ideia natural de base é usar funções binárias:

$$b_a(x) = \begin{cases} 1, & \text{se } x = a \\ 0, & \text{caso contrário} \end{cases}$$

para todo $a \in \{-1,\ 1\}^N$.

Nessa base, as coordenadas de $f \in \mathcal{F}$ são simplesmente a tabela de valores de $f(a)$ para todo $a \in \{-1,\ 1\}^N$. Note que as coordenadas serão apenas 1 e -1. Para entendermos como essa base funciona, vejamos um exemplo:

Exemplo 6.8. Suponha que ordenamos $a \in \{-1,\ 1\}^N$: $a_1, a_2, \dots, a_{2^N}$, então para algum a_j temos que:

$$f(a_j) = f(a_j) \cdot b_{a_j}(a_j) + \sum_{1 \le i \le 2^N, i \neq j} f(a_j) \cdot b_{a_i}(a_j)$$

$$f(a_j) = f(a_j) \cdot 1 + \sum_{1 \le i \le 2^N, i \neq j} f(a_j) \cdot 0$$

$$f(a_j) = f(a_j)$$

Agora, descreveremos outra base possível para esse espaço, que de fato será utilizada adiante. Primeiro, definimos um produto interno:

Definição 6.9.

$$\langle f, g \rangle = \frac{1}{2^N} \sum_{a \in \{-1,\ 1\}^N} f(a)g(a).$$

Definição 6.10. Sejam $a \in \{-1,\ 1\}^N$ e um conjunto $S \subset \{1, 2, \dots, N\}$. $\mathcal{X}_S : \{-1,\ 1\}^N \longrightarrow \{-1,\ 1\}$ é uma função linear tal que:

$$\mathcal{X}_S(a) = \prod_{i \in S} a_i.$$

Ou seja, S é um subconjunto dos índices de um vetor e $\mathcal{X}_S$ é o produto dos elementos nesses índices.

Proposição 6.11. O conjunto de funções $\mathcal{X}_S$ para todo $S \subset \{1, 2, \dots, N\}$ forma uma base ortonormal para nosso espaço $\mathcal{F}$ com o produto interno definido acima.

Prova 1. É fácil verificar a normalidade da base: como $\mathcal{X}_S(a) : \{-1,\ 1\}^N \longrightarrow \{-1,\ 1\}$, é óbvio que $|\mathcal{X}_S(a)| = 1$ e, portanto, também $\langle \mathcal{X}_S(a), \mathcal{X}_S(a) \rangle = 1$, para quaisquer S e a.

Queremos provar também ortogonalidade: sejam $S,\ S' \subset \{1, 2, \dots, N\}$ tais que $S \neq S'$, então:

$$\langle \mathcal{X}_S, \mathcal{X}_{S'} \rangle = \frac{1}{2^N} \sum_{a \in \{-1,\ 1\}^N} \mathcal{X}_S(a)\mathcal{X}_{S'}(a)$$

$$\langle \mathcal{X}_S, \mathcal{X}_{S'} \rangle = \frac{1}{2^N} \sum_{a \in \{-1,\ 1\}^N} \left(\prod_{i \in S} a_i \prod_{i \in S'} a_i \right)$$

$$\langle \mathcal{X}_S, \mathcal{X}_{S'} \rangle = \frac{1}{2^N} \sum_{a \in \{-1,\ 1\}^N} \left(\prod_{i \in S \cap S'} (a_i)^2 \prod_{i \in S \triangle S'} a_i \right),$$

onde $S \triangle S'$ é a diferença simétrica entre S e S', isto é, $S \triangle S' = (S \cup S') \setminus (S \cap S')$. É claro que $\prod_{i \in S \cap S'} (a_i)^2 = 1$. Então nos sobra:

$$\langle \mathcal{X}_S, \mathcal{X}_{S'} \rangle = \frac{1}{2^N} \sum_{a \in \{-1,\ 1\}^N} \left(\prod_{i \in S \triangle S'} a_i \right).$$

Finalmente, temos o "pulo do gato": fixamos um $j \in S \triangle S'$ e então podemos agrupar todos os vetores $a \in \{-1,\ 1\}^N$ em pares, chamemos de a e a', tais que:

$$a_i = \begin{cases} a'_i, & i \neq j \\ -a'_i, & i = j. \end{cases}$$

Para cada par, temos que

$$\prod_{i \in S \triangle S'} a_i + \prod_{i \in S \triangle S'} a'_i = 0.$$

Portanto,

$$\langle \mathcal{X}_S, \mathcal{X}_{S'} \rangle = \frac{1}{2^N} \sum_{a \in \{-1,\ 1\}^N} \left(\prod_{i \in S \triangle S'} a_i \right) = 0.$$

■

Queremos decompor uma $f \in \mathcal{F}$ qualquer utilizando nossa base, o que faremos da seguinte forma:

$$f(a) = \sum_{S \in [n]} \hat{f}(S)\mathcal{X}_S(a),$$

onde $[n] := \{1,\ 2, \dots,\ n\}$ e $\hat{f}(S)$ são os coeficientes de Fourier de f, definidos como:

$$\hat{f}(S) = \langle f,\ \mathcal{X}_S \rangle = \frac{1}{2^N} \sum_{a \in \{-1,\ 1\}^N} f(a)\mathcal{X}_S(a).$$

6.5.1 Coeficientes de Fourier e influência

Agora podemos encontrar uma nova fórmula para influência utilizando coeficientes de Fourier (e algumas manipulações algébricas):

$$Inf_i(f) = \sum_{S \ni i} \hat{f}(S)^2.$$

6.5.2 Coeficientes de Fourier e influência de k-ésimo grau

Agora definimos o conceito de influência de k-éismo grau:

Definição 6.12.

$$Inf_i^{\leq k}(f) := \sum_{S \ni i,\ |S| \leq k} \hat{f}(S)^2.$$

6.6 Extensão do MIS

Por fim, precisamos apresentar uma extensão do MIS:

Definição 6.13. Sejam $-1 < \rho \leq 0$ e $\epsilon > 0$, então existe algum $\tau = \tau(\rho, \epsilon) > 0$ e algum $k = k(\rho,\ \epsilon)$, tais que para qualquer $f : \{-1,\ 1\}^N \longrightarrow \{-1,\ 1\}$, se $Inf_i^{\leq k}(f) < \tau$ para todo $i \in [N]$ e $E[f] = 0$, então:

$$\frac{1}{2} - \frac{1}{2} Stab_\rho(f) < \frac{cos^{-1}\rho}{\pi} + \epsilon.$$

Capítulo 7

Redução do ULC-gap ao MaxCut

7.1 Notas iniciais

Quando nos referirmos a long codes aqui, estamos nos referindo na verdade à função de decodificação de um long code, especificamente à tabela verdade que representa a saída da função para cada entrada possível.

Chamamos de *completeness* a probabilidade do PCP ser aceito quando a resposta correta é SIM e de *soundness* a probabilidade dele ser aceito quando a resposta correta é NÃO.

7.2 Visão geral

Para provar o limite de inaproximabilidade, vamos construir um PCP para o ULC-gap (ver 3.3) em que o Verificador lê dois bits da prova e a aceita se e somente se eles forem diferentes.

Podemos modelar a prova como um grafo em que cada bit é um vértice e para cada par de bits que pode ser escolhido pelo verificador, inserimos uma aresta. Assim, geramos um gap *soundness*/*completeness*, que é exatamente o bound que queremos provar.

Mas, por que esse gap garante a inaproximabilidade? Pela UGC, é difícil distinguir entre as instâncias do ULC, mas seja I uma instância SIM. Então, ela deve ser aceita com probabilidade maior ou igual à razão de *completeness* c. Pensando na prova como uma instância do max cut, isso significa que o corte ótimo corta uma fracão maior ou igual a c das arestas. Agora, suponha que existe um algoritmo de aproximação melhor do que α_{GW},

então se executássemos esse algoritmo, acabaríamos com um corte que corta uma fração maior ou igual à razão de *soundness* s das arestas. Assim, saberíamos em tempo polinomial que I é uma instância SIM, contrariando a UGC.

7.3 O PCP

Antes de montarmos o PCP, é necessário tratar o grafo (bipartido) da instância do ULC, tornando-o regular no lado V, como descrito em KHOT e REGEV, 2008. Isso será importante na escolha das arestas. Podemos considerar essa operação como uma "caixa preta", pois é bem complexa, foge do escopo desse trabalho e nos interessemos apenas pelo seu resultado.

Agora, ao teste do PCP em si:

Começamos com uma instância do ULC-gap, o oráculo supostamente irá devolver o long code do rótulo correto para cada vértice de W, ou seja, irá devolvê-lo, se existir, ou tentar enganar o Verificador, caso contrário. Na verdade, em ambos os casos ele tenta convencer o Verificador de que ele devolve um long code referente ao rótulo que deveria ser aplicado ao vértice, mas quando ele realmente existe, isso é trivial: basta devolvê-lo.

O primeiro passo é o Verificador escolher um vértice $v \in V$ e duas arestas $vw,\ vw'$ aleatoriamente.

Aqui a importância do grafo ser regular em V: tornar a probabilidade de cada aresta ser escolhida uniforme. Como todo vértice $v \in V$ possui o mesmo grau, a probabilidade de cada aresta ser escolhida é a mesma.

Sejam $\sigma = \sigma_{v,w}$ e $\sigma' = \sigma_{v,w'}$ as bijeções para vw e vw', respectivamente.

Agora, o oráculo devolve f_w e $f_{w'}$, os supostos long codes de w e w', respectivamente.

O Verificador escolhe $x \in \{-1,\ 1\}^M$ aleatoriamente e $\mu \in \{-1,\ 1\}^M$ em que cada elemento de μ é escolhido independentemente com probabilidade $1/2 + 1/2\rho$ de ser 1 e $1/2 - 1/2\rho$ de ser -1. O teste é aceito se e somente se:

$$f_w(x \cdot \sigma) \neq f_{w'}((x \cdot \sigma')\mu).$$

7.4 Completeness

Proposição 7.1. Completeness é pelo menos $(1 - 2\eta)(1/2 - 1/2\rho)$.

Prova 2. Por hipótese, a instância do ULC satisfaz pelo menos $1 - \eta$ das arestas, então, ambas as arestas vw e vw' são satisfeitas com probabilidade pelo menos $(1 - 2\eta)$.

Sejam $i,\ j,\ j'$ rótulos para $v,\ w,\ w'$, respectivamente, que satisfaçam as arestas vw e vw', ou seja:

$$\sigma(j) = \sigma'(j') = i.$$

Então

$$f_w(x \cdot \sigma) = x_{\sigma(j)} = x_i$$
$$f_{w'}((x \cdot \sigma')\mu) = x_{\sigma'(j')}\mu_{j'} = x_i\mu_{j'}$$

É claro que $f_w(x \cdot \sigma) \neq f_{w'}((x \cdot \sigma')\mu)$ se e somente se $\mu_{j'} = -1$.

Finalmente,

$Pr[f_w(x\cdot\sigma) \neq f_{w'}((x\cdot\sigma')\mu)] \geq Pr[\text{vw e vw' são satisfeitas}]\cdot Pr[\mu_{j'} = -1] \geq (1-2\eta)(1/2-1/2\rho)$.

■

7.5 Soundness

Queremos provar que se para uma instância do label cover a solução ótima satisfaz uma fração menor do que γ das arestas, então o PCP é aceito com probabilidade menor ou igual a $s = \frac{arccos\rho}{\pi} + \epsilon$. Vamos provar a contrapositiva, ou seja, se o PCP é aceito com probabilidade maior do que s, então a instância do label cover deve satisfazer uma fração $\gamma' > \gamma$ das arestas.

Primeiramente, faremos uma análise com relação à escolha inicial de $v \in V$ (primeiro passo do PCP):

Proposição 7.2. Se, em geral, o PCP é aceito com probabilidade pelo menos $\frac{arccos\rho}{\pi} + \epsilon$, então uma fração de pelo menos $\frac{\epsilon}{2}$ de v é aceita com probabilidade maior ou igual a $\frac{arccos\rho}{\pi} + \frac{\epsilon}{2}$.

Prova 3. Provamos por contradição analisando o "pior caso": mesmo se a fração aceita com probabilidade maior do que $\frac{arccos\rho}{\pi} + \frac{\epsilon}{2}$ for toda aceita com probabilidade 1 e o resto com probabilidade exatamente $\frac{arccos\rho}{\pi} + \frac{\epsilon}{2}$, em geral teríamos probabilidade de aceitar igual a:

$$1 \cdot \frac{\epsilon}{2} + (\frac{arccos\rho}{\pi} + \frac{\epsilon}{2}) \cdot (1 - \frac{\epsilon}{2}) < \frac{arccos\rho}{\pi} + \epsilon.$$

■

Chamaremos os vértices de V que quando escolhidos fazem o PCP ser aceito com probabilidade maior do que $\frac{arccos\rho}{\pi} + \frac{\epsilon}{2}$ de *bons*.

Agora, vamos calcular a probabilidade do PCP ser aceito quando escolhemos um vértice bom. Lembramos que ele é aceito se e somente se:

$$f_w(x \cdot \sigma) \neq f_{w'}((x \cdot \sigma')\mu).$$

Então, se o PCP é aceito, $f_w(x\cdot\sigma)\cdot f_{w'}((x\cdot\sigma')\mu) = -1$ e se é rejeitado $f_w(x\cdot\sigma)\cdot f_{w'}((x\cdot\sigma')\mu) = 1$. Fazemos uma pequena manipulação algébrica para calcularmos a probabilidade de sucesso:

$$\mathbf{E}_{w,w',x,\mu}[\frac{1}{2} - \frac{1}{2} \cdot f_w(x \cdot \sigma) \cdot f_{w'}((x \cdot \sigma')\mu)]$$

Antes de continuarmos, faremos a seguinte definição: para todo vértice $v \in V$, seja função $g_v : \{-1,\ 1\}^N \longrightarrow \{-1,\ 1\}$ tal que:

$$g_v(x) = \mathbf{E}_{w\sim v}[f_w(x \cdot \sigma_{v,w})],$$

onde w é um vizinho aleatório de v.

A intuição dessa função é que se f_w for um long code válido e a restrição $\sigma_{v,w}$ for satisfeita para todo w, então g_v é o long code de v.

Continuando com a probabilidade do teste ser aceito dado que escolhemos um vértice bom, fazemos mais algumas manipulações:

$$\mathbf{E}_{w,w',x,\mu}[\frac{1}{2} - \frac{1}{2} \cdot f_w(x \cdot \sigma) \cdot f_{w'}((x \cdot \sigma')\mu)] =$$

$$\frac{1}{2} - \frac{1}{2}\mathbf{E}_{x,\mu}[E_{w,w'}[f_w(x \cdot \sigma) \cdot f_{w'}((x \cdot \sigma')\mu)]].$$

Pela independência de w e w':

$$\frac{1}{2} - \frac{1}{2}\mathbf{E}_{x,\mu}[\mathbf{E}_w[f_w(x \cdot \sigma)] \cdot \mathbf{E}_{w'}[f_{w'}((x \cdot \sigma')\mu)]] =$$

$$\frac{1}{2} - \frac{1}{2}E_{x,\mu}[g_v(x) \cdot g_v(x \cdot \mu)] =$$

$$\frac{1}{2} - \frac{1}{2}Stab_\rho(g_v).$$

Lembrando que essa é a probabilidade do teste ser aceito quando começamos escolhendo um vértice v *bom* e definição de um vértice bom, chegamos na seguinte desigualdade:

$$\frac{1}{2} - \frac{1}{2}Stab_{\rho}(g_v) \geq \frac{arccos\rho}{\pi} + \frac{\epsilon}{2}.$$

Agora, iremos aplicar a extensão do MIS (ver 6.13). Lembramos que ele foi apresentado na forma "se A, então B" e o caso que temos é "se não B, então não A". Concluímos que para todo vértice bom v existe pelo menos uma coordenada j tal que:

$$Inf_i^{\leq k}(g_v) \geq \tau.$$

Pela definição de influência:

$$\tau \leq \sum_{S \ni i,\ |S| \leq k} \hat{g_v}(S)^2.$$

Pela definição de g:

$$\tau \leq \sum_{S \ni i,\ |S| \leq k} \left(\mathbf{E}_w[\hat{f_w}(\sigma^{-1}(S))] \right)^2.$$

Pela desigualdade de Cauchy-Schwarz:

$$\tau \leq \sum_{S \ni i,\ |S| \leq k} \mathbf{E}_w[\hat{f_w}(\sigma^{-1}(S))^2].$$

Mais algumas manipulações algébricas:

$$\tau \leq \mathbf{E}_w \left[\sum_{S \ni i,\ |S| \leq k} \hat{f_w}(\sigma^{-1}(S))^2 \right]$$

$$\tau \leq \mathbf{E}_w[Inf^{\leq k}_{\sigma^{-1}(i)}(f_w)].$$

Fazendo uma análise análoga a que foi feita na prova 3, concluímos que uma fração de pelo menos $\frac{\tau}{2}$ do vizinhos w de v tem $Inf^{\leq k}_{\sigma^{-1}_{v,w}(i)}(f_w) \geq \tau/2$.

Agora, definimos um conjunto de rótulos candidatos para um w qualquer:

$$Cand[w] = \{i : \ Inf_i^{\leq k}(f_w) \geq \tau/2\}.$$

E escolheremos um rótulo aleatoriamente dessa lista. Mas antes, vamos analisar quantos elementos possui esse conjunto:

Primeiramente, notamos que:

Proposição 7.3. $\sum_i^m Inf_i^{\leq k}(f) \leq k$

Prova 4.

$$\sum_{i=1}^{m} Inf_i^{\leq k}(f) =$$

Por definição (6.12):

$$\sum_{i=1}^{m} \sum_{S \ni i, \ |S| \leq k} \hat{f}(S)^2 =$$

Pela independência de $\hat{f}(S)^2$ em relação a i:

$$\sum_{|S| \leq k} |S| \hat{f}(S)^2 \leq$$

Como, por definição, $|S| \leq k$:

$$k \sum_{|S| \leq k} \hat{f}(S)^2 \leq$$

Como $\hat{f}(S)^2 \geq 0$ para qualquer S:

$$k \sum_{S} \hat{f}(S)^2 =$$

$$kf(S)^2 =$$

$$k.$$

■

Agora, chegamos ao número de elementos do conjunto:

Proposição 7.4. $|Cand[w]| \leq \frac{2k}{\tau}$

Prova 5.

$$|Cand[w]| \cdot \tau/2 \leq \sum_{i \in Cand[w]} Inf_i^{\leq k}(f) \leq \sum_{i}^{m} Inf_i^{\leq k}(f) \leq k$$

$$|Cand[w]| \leq 2k/\tau.$$

■

Finalmente, é possível encontrar rótulos que satisfazem uma fração de pelo menos

$$\frac{\tau}{2} \cdot \frac{\tau}{2k}$$

das arestas adjacentes a cada v *bom*. Cada termo significa, respectivamente, a probabilidade de um vizinho w de v ter $Inf^{\leq k}_{\sigma^{-1}_{v,w}(i)}(f_w) \geq \tau/2$ e de escolher $\sigma^{-1}(i_v)$ de $Cand[w]$.

Como uma fração de pelo menos $\epsilon/2$ dos vértices v são bons, conseguimos encontrar uma solução para a instância do ULC tal que:

$$\gamma' \geq \frac{\epsilon}{2} \cdot \frac{\tau}{2} \cdot \frac{\tau}{2k},$$

concluindo a prova do limite de soundness. ■

Capítulo 8

Conclusão

Escrever esse texto foi uma longa jornada, com muitas idas e voltas, becos sem saída e mudanças de percurso. Grande parte da dificuldade se deve à falta de conteúdo sobre esse assunto em livros didáticos. Obrigando-nos a utilizar como principais fontes artigos e notas de aulas.

Inicialmente, tentamos seguir a prova para o limite de inaproximabilidade do Corte Máximo feita em Trevisan, 2012, porém essa prova contém vários "buracos"não explicados e não conseguimos desvendar alguns. Então, decidimos mudar para a prova original, apresentada em Khot, Kindler *et al.*, 2007.

Mesmo após a mudança, não foi fácil. Uma ajuda essencial foi a página do curso *CSE 533: The PCP Theorem and Hardness of Approximation* da Universidade de Washington, que pode ser acessada aqui. O curso basicamente constrói ao longo do semestre todos as bases necessárias para apresentar a mesma prova de inaproximabilidade do Corte Máximo que apresentamos aqui.

O curso entra em muitos detalhes que não foram possíveis aqui, principalmente sobre o PCP, e as notas de aula não são suficientes para compreender tudo (nem se propõem a isso), mas a estrutura utilizada foi um excelente guia.

A partir daí e de outros materiais, conseguimos desvendar o quebra cabeça e montar esse texto. Tratamos de um tema inerentemente complexo, mas nossa meta foi deixá-lo o mais simples e compreensível possível. Esperamos ter chegado perto.

Referências

[Arora *et al.* 1998] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan e Mario Szegedy. "Proof verification and the hardness of approximation problems". Em: *Journal of the ACM (JACM)* 45.3 (1998), pgs. 501–555 (citado na pg. 9).

[Austrin 2007] Per Austrin. "Balanced max 2-sat might not be the hardest". Em: *STOC*. Vol. 7. Citeseer. 2007, pgs. 189–197 (citado na pg. 3).

[Bar-Yehuda e Even 1981] Reuven Bar-Yehuda e Shimon Even. "A linear-time approximation algorithm for the weighted vertex cover problem". Em: *Journal of Algorithms* 2.2 (1981), pgs. 198–203 (citado na pg. 3).

[Carvalho *et al.* 2001] M. Carvalho *et al. Uma Introdução Sucinta a Algoritmos de Aproximação*. First. 2001, pgs. xvi+157 (citado na pg. 3).

[Guruswami *et al.* 2008] Venkatesan Guruswami, Rajsekar Manokaran e Prasad Raghavendra. "Beating the random ordering is hard: inapproximability of maximum acyclic subgraph". Em: *2008 49th Annual IEEE Symposium on Foundations of Computer Science*. IEEE. 2008, pgs. 573–582 (citado na pg. 4).

[Goemans e Williamson 1995] Michel X Goemans e David P Williamson. "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming". Em: *Journal of the ACM (JACM)* 42.6 (1995), pgs. 1115–1145 (citado nas pgs. 3, 4).

[Håstad 2001] Johan Håstad. "Some optimal inapproximability results". Em: *Journal of the ACM (JACM)* 48.4 (2001), pgs. 798–859 (citado na pg. 4).

[Karp 1972] Richard M Karp. "Reducibility among combinatorial problems". Em: *Complexity of computer computations*. Springer, 1972, pgs. 85–103 (citado na pg. 4).

[Khot, Kindler *et al.* 2007] Subhash Khot, Guy Kindler, Elchanan Mossel e Ryan O'Donnell. "Optimal inapproximability results for max-cut and other 2-variable csps?" Em: *SIAM Journal on Computing* 37.1 (2007), pgs. 319–357 (citado nas pgs. 3, 4, 15, 29).

[Khot 2002] Subhash Khot. "On the power of unique 2-prover 1-round games". Em: *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. ACM. 2002, pgs. 767–775 (citado nas pgs. 3, 5).

[Khot e Regev 2008] Subhash Khot e Oded Regev. "Vertex cover might be hard to approximate to within 2- ε". Em: *Journal of Computer and System Sciences* 74.3 (2008), pgs. 335–349 (citado nas pgs. 3, 22).

[Kohayakawa e Soares 1995] Yoshiharu Kohayakawa e José Augusto Soares. *Demonstrações transparentes e a impossibilidade de aproximações*. 1995 (citado na pg. 7).

[Lewin *et al.* 2002] Michael Lewin, Dror Livnat e Uri Zwick. "Improved rounding techniques for the max 2-sat and max di-cut problems". Em: *International Conference on Integer Programming and Combinatorial Optimization*. Springer. 2002, pgs. 67–82 (citado na pg. 3).

[Mossel *et al.* 2005] Elchanan Mossel, Ryan O'Donnell e Krzysztof Oleszkiewicz. "Noise stability of functions with low influences: invariance and optimality". Em: *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*. IEEE. 2005, pgs. 21–30 (citado na pg. 15).

[Trevisan 2012] Luca Trevisan. "On khot's unique games conjecture." Em: *Bulletin (New Series) of the American Mathematical Society* 49.1 (2012) (citado na pg. 29).