On efficient approximability of NP-hard optimization problems

Johan Håstad

Royal Institute of Technology

Stockholm, SWEDEN

Outline

- A complete example.
- A small tour of problems.
- A discussion of which problems admit nontrivial approximation.

Today's main example

$$
\left\{
\begin{array}{ccccccccccc}
x_1 & + & x_2 & + & x_3 & & & = & 1 & & \\
x_1 & + & x_2 & & & & & = & 1 & & \\
x_1 & + & x_2 & + & & & x_4 & = & 1 & & \\
 & & x_2 & + & & & x_4 & = & 0 & & \text{mod } 2 \\
x_1 & + & & & x_3 & + & x_4 & = & 0 & & \\
 & & x_2 & + & x_3 & + & x_4 & = & 1 & & \\
x_1 & + & & & x_3 & & & = & 0 & &
\end{array}
\right.
$$

Satisfy as many equations as possible (or $\geq K$).

Max-Lin-2, m equations n variables.

Obvious algorithms

Try to satisfy all using Gaussian elimination.

Time $\approx mn^2$.

Try all possible assignments and take the best.

Time $\approx mn2^n$.

Easy heuristics

Pick values for variables randomly.

Determine x_i sequentially, consider only equations reduced to the form $x_i = c$.

Satisfies $m/2$ equations on average.

Satisfies $\geq m/2$ equations.

Simple fact

Theorem: Max-Lin-2 in its decision form is NP-complete.

Proof: Straightforward reduction from 3SAT.

Approximation algorithms.

C-approximation

Maximization problems:

Found value $\geq$ (Optimal value)/ C.

Minimization problems:

Found value $\leq$ C(Optimal value).

This is required for all inputs!

Simple theorem

Theorem: Max-Lin-2 admits an efficient 2-approximation algorithm.

Proof: We can satisfy $m/2$ equations and the optimum satisfies at most all m equations.

First real theorem

Theorem: [Hå97] $\forall\delta > 0$, unless NP=P, Max-Lin-2 does not admit a (2-δ)-approximation algorithm.

phrased differently

Theorem: $\forall\delta > 0$, it is NP-hard to (2-δ)-approximate Max-Lin-2.

Even when we allow only 3 variables in each equation.

Proof outline

Use a supposed good approximation algorithm to solve an NP-hard problem like SAT in polynomial time.

$$
\begin{tikzcd}
\boxed{\varphi \text{ satisfiable?}} \arrow[r] & \boxed{L \text{ mod 2 system}} \arrow[d] \\
\boxed{\alpha \text{ assignment for } \varphi} & \boxed{x \text{ good solution for } L} \arrow[l]
\end{tikzcd}
$$

Key tool: Probabilistically checkable proofs.

Probabilistically Checkable Proofs (PCP)

Statement to be verified: Formula φ is satisfiable.

Proof: A written proof π.

Verifier: V.

Completeness c: If indeed φ is satisfiable then exists π that convinces V with probability c.

Soundness s: If φ is not satisfiable then no π convinces V with probability $\geq s$.

Normally $c = 1$ and s is a small constant.

A comparison

Traditional proofs: Perfect completeness and soundness ($c = 1$ and $s = 0$) and the verifier reads the entire proof.

New situation: We want reasonable values of c and s when the verifier only reads very small portion of the proof, like 3 bits.

PCP optimization problem

Given a formula φ find the proof π that maximizes the probability that the verifier accepts.

Easy observation: If we can solve this optimization problem within a factor c/s, then we can determine whether φ is satisfiable!

Design the PCP to make this optimization problem equal to Max-Lin-2!

The desired PCP

It should have the following properties

- It should be small.
- Checking should consist of computing exclusive ors.

The correspondence

Bit b in the proof	Variable x_b.
Checking that xor of b_1, b_2, b_3 is 1 with probability p	Equation $x_{b_1} + x_{b_2} + x_{b_3} = 1$ with weight p

A remarkable proof

Input: Formula φ on 3-CNF. It has n variables and m clauses.

Proof: A written proof π of size polynomial in n and m.

Verifier: V. Randomly chooses 3 bits in the proof, computes their exclusive or and rejects or accepts depending on the outcome.

Completeness: If indeed φ is satisfiable then exists π that convinces V with probability $1 - \epsilon$.

Soundness: If φ is not satisfiable then no π convinces V with probability $\geq \frac{1}{2} + \epsilon$.

A less remarkable proof

Input: Formula φ on 3-CNF. It has n variables and m clauses.

Proof: A written proof π of size exponential in n and m.

Verifier: V. Randomly reads 1 bit.

Completeness: If indeed φ is satisfiable then exists π that convinces V with probability 1.

Soundness: If φ is not satisfiable then no syntactically correct π convinces V with probability $\geq \frac{1}{2}$.

The less remarkable proof

$$\varphi \text{ satisfied} \Leftrightarrow P_i(x) = 0,\ i = 1, 2, \ldots m,$$

where P_i are degree 3 polynomials over $GF[2]$. If first clause $(x_1 \vee x_2 \vee x_3)$ then

$$P_1(x) = (1 + x_1)(1 + x_2)(1 + x_3).$$

π consists of table $T(Q)$ for all degree 3 polynomials Q over $GF[2]$.

For a correct proof $T(Q) = Q(\vec{x}_0)$ where $\vec{x}_0$ is a satisfying assignment of φ.

Syntactically correct $T(Q) = Q(\vec{x}_0)$ for some $\vec{x}_0$.

Checking

Checking: Pick random r_i and set

$P = \sum_{i=1}^{m} r_i P_i$. Accept iff $T(P) = 0$.

$$\forall_i P_i(\vec{x}_0) = 0 \Rightarrow P(\vec{x}_0) = 0$$

and

$$\exists_i P_i(\vec{x}_0) = 1 \Rightarrow Pr[P(\vec{x}_0) = 0] = \frac{1}{2}.$$

Making the proof more remarkable

Need to take care of

- Syntactical correctness.

- Size of proof.

- Acceptance criteria.

By reading $O(1)$ bits one can check approximate syntactical correctness. Read values for P' and $P' \oplus P$ instead of value for P.

Very clever recursive techniques [ALMSS]. Details are complicated.

Adding an extra layer of recursion to change top level and using a new coding of strings.

The PCP theorem

Theorem: [ALMSS] Any statement in NP can be verified by a PCP of polynomial size where the verifier reads $O(1)$ bits and uses $O(\log n)$ random bits. The test has completeness 1 and soundness 1/2.

The long code [BGS]

To code $x \in \{0,1\}^v$ write down $f(x)$ for any $f : \{0,1\}^v \mapsto \{0,1\}$.

We code v bits by 2^{2^v} bits.

Extremely useful for verifying since any property of x can be read in one bit. Used for strings of constant length.

Approximate syntactical correctness can be checked efficiently and many times comes for free.

One word about the proof

Given all the previous results [ALMSS, R] the final analysis is an argument about discrete Fourier transforms where we essentially need only Plancerel's equality.

Theorem again

Theorem: $\forall\delta > 0$, it is NP-hard to $(2\text{-}\delta)$-approximate Max-Lin-2.

Even when we allow only 3 variables in each equation.

Remain true in any Abelian group, inapproximability is almost size of group.

Approximation properties of other NP-hard optimization problems

Problems do come in different flavors.

Some hardness results follow from old reductions. Sharp results often require special purpose PCPs.

Algorithmic result are usually special purpose. Key new technique: Semidefinite programming [GW94].

Other problem 1: Max-3-SAT

Problem: Given a 3-CNF formula satisfy the maximal number of clauses.

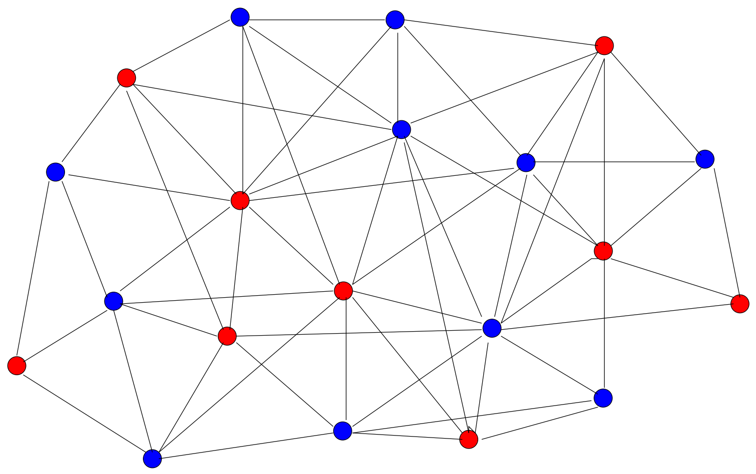
$$\varphi = (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4)$$

Upperbound: $8/7$. Easy when all clauses are of length exactly 3 [J74], otherwise semi-definite programming [KZ99].

Lowerbound: $8/7 - \epsilon$. Straightforward reduction from Max-Lin-2.

Other problem 2: Max-Cut

Problem: Given a graph G partition the vertices into two sets to cut the maximal number of vertices.



Trivial upperbound: 2 . A random partition cuts half the edges.

Good upperbound: 1.138. Semidefinite programming [GW94].

Lowerbound: $17/16 - \epsilon$. Reduction from Max-Lin-2 found by computer, verified by hand.

Semi-definite programming

Max-cut equals

$$\max \sum_{(i,j)\in E} \frac{(1-x_i x_j)}{2}, \quad x_i \in \{-1,1\}$$

Relax

$$\max \sum_{(i,j)\in E} \frac{(1-Y_{ij})}{2},$$

where Y is symmetric positive semi-definite matrix with $Y_{ii} = 1$.

$Y = V^T V, Y_{ij} =< v_i, v_j >$.

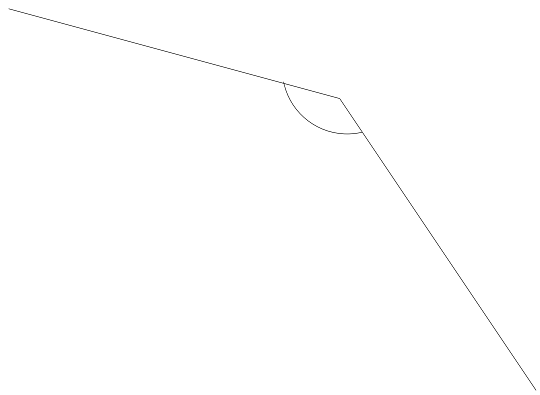
Almost optimal Y can be found efficiently.

Pick a random r and set $x_i = sign(< r, v_i >)$.

Analysis

$(i,j) \in E$ contribution to objective function

$(1 - Y_{ij})/2 = (1 - < v_i, v_j >)/2 = (1 - \cos\theta)/2$



Probability of being cut, i.e.

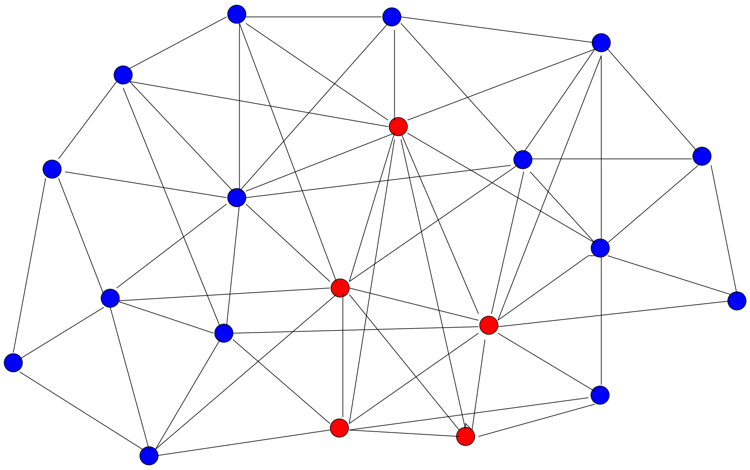
$sign(< r, v_i >) \neq sign(< r, v_j >)$ is θ/π.

$$\max_{\theta} \frac{\pi(1 - \cos\theta)}{2\theta} \approx 1.138$$

gives approximation ratio.

Other problem 3: Clique (Independent set)

Problem: Given a graph G with n vertices, find the largest number of pairwise connected vertices.

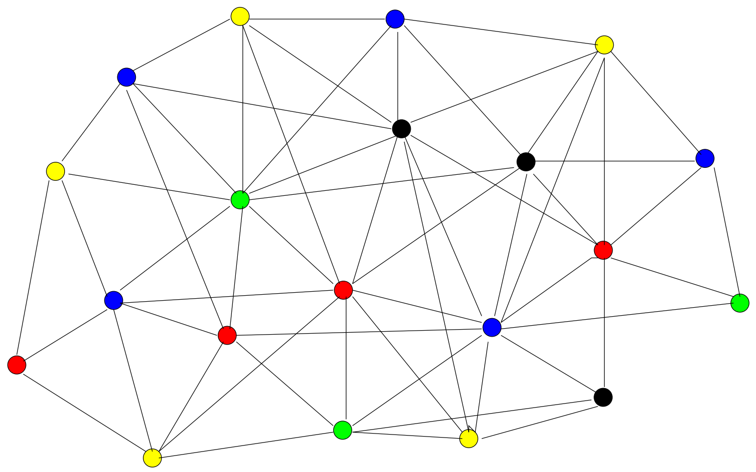


Upperbound: $O\left(\frac{n}{(\log n)^2}\right)$. Combinatorial algorithm removing subgraphs.[BH92]

Lowerbound: $n^{1-\epsilon}$ [Hå96], $n^{1-o(1)}$ [EH00]. Special purpose PCP.

Other problem 4: Graph-coloring

Problem: Given a graph G with n vertices, color the vertices with the minimal number of colors



Upperbound: $O\left(\frac{n(\log\log n)^2}{(\log n)^3}\right)$.
Combinatorial algorithm. [Ha93]

Lowerbound: $n^{1-\epsilon}$. An additional idea on top of clique lower bound. [FK96].

Other problem 5: Set-cover

Problem: Given a collection of subset of $[n]$, cover the union with as few sets as possible.

$$(1,2,3),(4,5,6)(7,8,9),(1,4,7,8),$$
$$(2,5,9),(3,5,8),(2,6,7)$$

Upperbound: $\ln n + 1$. Greedy algorithm [J74].

Lowerbound: $(1 - o(1)) \ln n$. Special purpose PCP [F96].

Greedy set cover

Algorithm: Take the set containing the largest number of remaining elements.

Analysis: If the smallest cover has k members the number of remaining elements after i steps is at most $(1-\frac{1}{k})^i n$. With $i = k \ln n$ we have covered all.

Other problem 6: Traveling Salesman Problem

Problem: Given n "cities" find the shortest tour that visits all.

Upperbound: Euclidean plane $(1+\epsilon)$ Geometric dynamic programming [A96]. Only Δ-inequality, 3/2. Based on maximal spanning tree [C76].

Lowerbound: Cannot be approximated without the Δ-inequality. With Δ $129/128$ [KP00]. Obtain by a, complicated, reduction from Max-Lin-2.

Fundamental question

Which NP-hard optimization problems admit nontrivial efficient approximation algorithms?

Non-trivial approximation

Linear equations	No
Max-3Sat	No
Max-Cut	Yes
Clique	No
Coloring	No
Set cover	No
TSP	Yes

Problematic issue

What do we mean by nontrivial approximation?

Easy case: Constraint satisfaction

Constraint satisfaction

Constraint: $f : \{0,1\}^k \mapsto \{0,1\}$.

Clauses: k-tuples of literals.

Objective: Satisfy as many as possible.

Examples: Linear equations 3 unknowns, Max-3Sat, Max-Cut, Set-splitting of fixed sized sets

Triviality definition

Constraint: $W(f) = Prob_x[f(x) = 1]$

An approximation algorithm with smaller ratio than $W(f)^{-1}$ is nontrivial.

Beats picking a random assignment.

First results

k=2

All problems allow nontrivial approximation

k=3

Complete classification. [Z]
Only trivial if $f(x) = 1 \Rightarrow x_1 + x_2 + x_3 = c$

Hopes

1. Could it be that all predicates, even on larger domains admit nontrivial approximation for $k = 2$?

2. Simple classification, possibly in terms of the Fourier transform of f?

True for linear equations mod m
[AEH]

Could be, but not as simple as for
$k = 3$. Might be non-monotone.

Perfect completeness

What happens if we study only instances where we can satisfy all constraints?

Can we efficiently find an assignment that satisfies a larger fraction than $W(f)$?

Perfect completeness; results

k=2	Can find optimal solutions
Max-3Lin	Can find optimal solutions
Max-3SAT	Nothing nontrivial
Not-Two	Unknown whether non-trivial possible

Open question 1

Can we beat a factor 2 for Vertex Cover? Strongest lower bound 7/6.

In the PCP-world, one free bit, $1-\epsilon$-completeness, ϵ-soundness, $O(\log n)$ random bits.

Possible with $O(n)$ random bits.

Open question 2

Understand exactly which constraint satisfaction problems admit nontrivial approximation?

Final words

We have (surprisingly) tight results for many problems, but few problems where both upper and lower bounds as non-trivial.

What is the Max-Cut constant?