Some of my interests and (old) results

Johan Håstad

KTH Teknikvetenskap

November 10, 2020

Questions

I am happy to take questions during the talk.

My world, Complexity theory

Efficient computation.

- P Polynomial time (in input length).
- BPP Probabilistic Polynomial time (still efficient).
- NP Non-deterministic Polynomial time.

I also (sometimes) care about actual computation in practice but let us ignore this today.

Randomness in computation

In practice for free and usually very low probability of error.

Whether needed is a very basic theoretical question.

Primality in deterministic polynomial time. Great progress in theory, of no importance in practice.

NP

The set of decision problems where a "yes"-answer can be verified in polynomial time.

Given a graph, can it be colored with three colors?

Given a set of linear inequalities in n variables, can it be satisfied by integers?

World view

We assume $P \neq NP$ and even $NP \not\subseteq BPP$.

Stronger assumptions that NP (or some particular problem in NP) cannot be solved in time

1. $2^{O((\log n)^k)}$.
2. $2^{O(n^\delta)}$ some $\delta > 0$.
3. $O(2^{cn})$ some $c > 0$.

Not needed in this talk but at least the first is not controversial and all are used.

Hard problems

NP-complete The hardest problems in NP.

NP-hard Possibly even harder problems, if in P then $NP = P$. Many times non-decision problems closely related to NP.

My current interests

1. Solving NP-hard problems approximately in polynomial time.
2. Cryptography and quantum computers.

I will today focus on approximation.

Interesting family of problems

Constraints on constant size subsets of Boolean variables.

Constraint Satisfaction Problems, CSPs.

Model problems for now 3-Sat, 3-Lin.

3-Sat

Satisfiability of 3-CNF formulas, i.e.

$$\varphi = (x_1 \vee \overline{x_7} \vee x_{12}) \wedge (x_2 \vee x_3 \vee x_8) \wedge \ldots (\overline{x_5} \vee \overline{x_{23}} \vee x_{99})$$

n variables, m clauses (i.e. disjunctions)

$$\varphi = \wedge_{i=1}^{m} C_i$$

For convenience let us assume that we have exactly three variables in each clause.

Basics for 3-Sat

Easy to verify a satisfying assignment.

Probably the most classical NP-complete problem, from Cook's original list in 1971.

No algorithm is known to run faster than 2^{cn} and work has been done trying to improve the value of c.

3-Lin

System of linear equations modulo 2 with at most three variables in each equation.

$$\begin{cases} x_1 + x_2 + x_3 = 1 \\ x_1 + x_2 = 1 \\ x_1 + x_2 + x_4 = 1 \\ x_2 + x_4 = 0 \\ x_1 + x_3 + x_4 = 0 \\ x_2 + x_3 + x_4 = 1 \\ x_1 + x_3 = 0 \end{cases} \mod 2$$

m equations n variables. Easy to solve by Gaussian elimination.

Max-CSP

New view. Given the set of constraints, maybe not all simultaneously satisfiable, try to satisfy as many as possible.

Optimization as opposed to decision.

Hope and fears

Hope for Max-3Sat: We know we cannot find the best solution but maybe we can find something reasonably good.

Fear for Max-3Lin: If we cannot satisfy all equations, the usefulness of Gaussian elimination seems doubtful.

Max-3-Lin vs Max-3-Sat

Observation: $(x_1 \vee x_2 \vee x_3)$ is true iff 4 of the equations

$$\begin{cases} x_1 + x_2 + x_3 = 1 \\ x_1 + x_2 = 1 \\ x_1 + x_3 = 1 \\ x_2 + x_3 = 1 \\ x_1 = 1 \\ x_2 = 1 \\ x_3 = 1 \end{cases} \quad \text{mod } 2$$

are satisfied (and otherwise none).

A reduction

Take 3-CNF $\varphi = \wedge_{i=1}^{m} C_i$ create $7m$ equations using last page giving system L.

Easy fact: φ is satisfiable iff we can simultaneously satisfy $4m$ equations of L.

Max-3-Lin is NP-hard!

A reduction

Take 3-CNF $\varphi = \wedge_{i=1}^{m} C_i$ create $7m$ equations using last page giving system L.

Easy fact: φ is satisfiable iff we can simultaneously satisfy $4m$ equations of L.

Max-3-Lin is NP-hard!

The fear was justified.

Performance measure for hope

Approximation ratio,

$$\alpha = \frac{Value(Found\ solution)}{Value(Best\ solution)}$$

worst case over all instances. $\alpha = 1$ the same as finding optimal, otherwise $\alpha < 1$.

For a randomized algorithm we allow expectation over internal randomness, worst case over inputs.

The mindless algorithm

Give each variable a random value without looking at the constraints.

The mindless algorithm

Give each variable a random value without looking at the constraints.

For Max-3-Sat we satisfy each clause with probability 7/8.

The mindless algorithm

Give each variable a random value without looking at the constraints.

For Max-3-Sat we satisfy each clause with probability 7/8.

Approximation ratio at least 7/8 (even deterministically).

The hope for Max-3-Sat

If a formula with m clauses is satisfiable then we can find an assignment that satisfies αm clauses where $\alpha > 7/8$.

The hope for Max-3-Sat

If a formula with m clauses is satisfiable then we can find an assignment that satisfies αm clauses where $\alpha > 7/8$.

It turns out that this hope cannot be fulfilled, but first a detour.

Two types of approximability results

- Positive results. Efficient algorithms with provable performance ratios.
- Negative results. Proving that certain tasks are NP-hard.

The favorite techniques

Algorithms: Semi-definite programming. Introduced in this context by Goemans and Williamson.

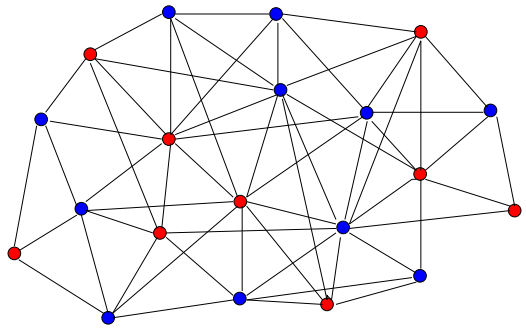
Lower bounds: The PCP-theorem and its consequences. Arora, Lund, Motwani, Sudan and Szegedy.

The favorite techniques

Algorithms: Semi-definite programming. Introduced in this context by Goemans and Williamson.

Max-Cut

Given a graph, partition the graph into two parts cutting as many edges as possible.



Basic NP-complete problem. Constraints: $x_i \neq x_j$ for any edge (i, j).

Max-Cut in formulas

The task is to maximize with $x_i \in \{-1, 1\}$ and edges E,

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

Max-Cut in formulas

The task is to maximize with $x_i \in \{-1, 1\}$ and edges E,

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

Relax by setting $y_{ij} = x_i x_j$ and requiring that Y is a symmetric positive semidefinite matrix with $y_{ii} = 1$.

Max-Cut in formulas

The task is to maximize with $x_i \in \{-1, 1\}$ and edges E,

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

Relax by setting $y_{ij} = x_i x_j$ and requiring that Y is a symmetric positive semidefinite matrix with $y_{ii} = 1$.

Optimal Y can be found in polynomial time.

View using $Y = V^T V$

Want to solve

$$\max_{x \in \{-1,1\}^n} \sum_{(i,j) \in E} \frac{1 - x_i x_j}{2}.$$

but as $Y = V^T V$ we instead maximize

$$\max_{\|v_i\|=1, i=1,\ldots n} \sum_{(i,j) \in E} \frac{1 - (v_i, v_j)}{2},$$

i.e. optimizing over vectors instead of real numbers.

Going vector to Boolean

The vector problem accepts a more general set of solutions. Gives higher objective value.

Key question: How to use the vector solution to get back a Boolean solution that does almost as well.

Rounding vectors to Boolean values

Simple but powerful idea of Goemans and Williamson.

Given vector solution v_i pick random vector r and set

$$x_i = \text{Sign}((v_i, r)),$$

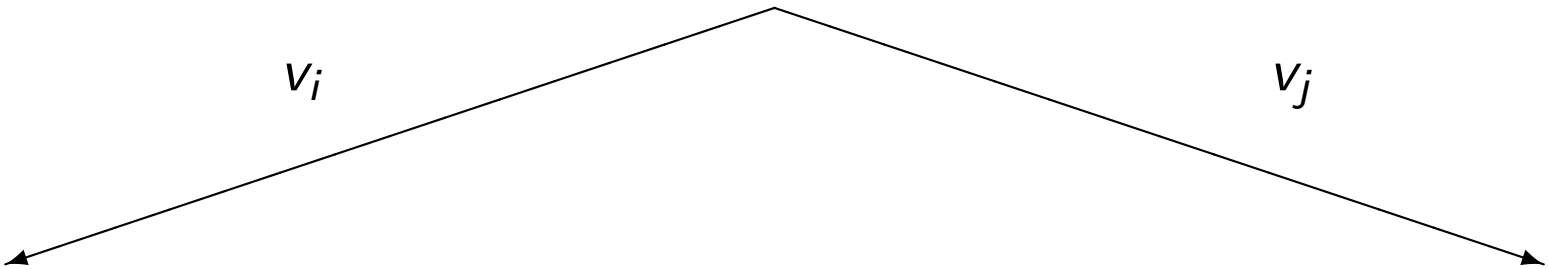
where (v_i, r) is the inner product.

Intuition of rounding

Contribution

$$\frac{1-(v_i, v_j)}{2}$$

to objective function large, implying angle between v_i, v_j large, $\text{Sign}((v_i, r)) \neq \text{Sign}((v_j, r))$ likely.



Analyzing GW

Do term by term, θ angle between vectors.
Contribution to semi-definite objective function

$$\frac{1-(v_i,v_j)}{2} = \frac{1-\cos\theta}{2}.$$

Probability of being cut

$$Pr[\text{Sign}((v_i,r)) \neq \text{Sign}((v_j,r))] = \frac{\theta}{\pi}.$$

Minimal quotient gives approximation ratio

$$\alpha_{GW} = \min_{\theta} \frac{2\theta}{\pi(1-\cos\theta)} \approx .8785$$

Switching sides

Let us turn to hardness results.

Proving NP-hardness results for approximability problems

Want to study problem X.

Proving NP-hardness results for approximability problems

Want to study problem X.

Given a Sat-formula φ, produce an instance, I of X such that:

φ satisfiable $\rightarrow$ $Value(I) \geq c$.

φ not satisfiable $\rightarrow$ $Value(I) \leq s$.

Proving NP-hardness results for approximability problems

Want to study problem X.

Given a Sat-formula φ, produce an instance, I of X such that:

φ satisfiable $\rightarrow$ $Value(I) \geq c$.

φ not satisfiable $\rightarrow$ $Value(I) \leq s$.

It is NP-hard to approximate our problem within $s/c + \epsilon$.

Running approximation algorithm on I tells us whether φ is satisfiable.

Inapproximability for Max-3-Sat

Given a Sat-formula φ, produce a different Sat-formula ψ with m clauses such that:

φ satisfiable $\rightarrow$ ψ satisfiable.

φ not satisfiable $\rightarrow$ Can only simultaneously satisfy only $(1-\epsilon)m$ of the clauses of ψ.

Gives inapproximability ratio $(1-\epsilon)$.

Probabilistically Checkable Proofs (PCPs)

A proof that 3-Sat formula φ is satisfiable.

Traditionally an assignment to the variables.

Checked by reading all variables and checking.

Probabilistically Checkable Proofs (PCPs)

A proof that 3-Sat formula φ is satisfiable.

Traditionally an assignment to the variables.

Checked by reading all variables and checking.

We want to read much less of the proof, only a constant number of bits.

Sought reduction gives PCP!

Proof: An assignment to variables of ψ.
Checking: Pick a random clause and read the variables that appear in the clause and check if it is satisfied.

Sought reduction gives PCP!

Proof: An assignment to variables of ψ.
Checking: Pick a random clause and read the variables that appear in the clause and check if it is satisfied.
Preserving satisfiability: φ satisfiable implies ψ satisfiable and we always accept.

Sought reduction gives PCP!

Proof: An assignment to variables of ψ.
Checking: Pick a random clause and read the variables that appear in the clause and check if it is satisfied.
Preserving satisfiability: φ satisfiable implies ψ satisfiable and we always accept.
Amplifying non-satisfiability: φ not satisfiable implies ψ only $(1-\epsilon)$-satisfiable and we reject with probability $\geq \epsilon$.

Sought reduction gives PCP!

Proof: An assignment to variables of ψ.
Checking: Pick a random clause and read the variables that appear in the clause and check if it is satisfied.
Preserving satisfiability: φ satisfiable implies ψ satisfiable and we always accept.
Amplifying non-satisfiability: φ not satisfiable implies ψ only $(1-\epsilon)$-satisfiable and we reject with probability $\geq \epsilon$.
Repeat a constant number of times to decrease fooling probability.

Thinking more carefully

Our type of reduction is equivalent to a good PCP.

The PCP theorem

PCP theorem: [ALMSS] There is a proof system for satisfiability that reads a constant number of bits such that

- Verifier always accepts a correct proof of correct statement.
- Verifier rejects any proof for incorrect statement with probability 1/2.

Translates to any NP statement by a reduction.

Proof of PCP theorem

Original proof: Algebraic techniques, properties of polynomials, proof composition, aggregation of queries, etc. Many details.

Proof of PCP theorem

Original proof: Algebraic techniques, properties of polynomials, proof composition, aggregation of queries, etc. Many details.

Second proof by Dinur (2005) that is essentially combinatorial. Relies on recursion and expander graphs.

Proof of PCP theorem

Original proof: Algebraic techniques, properties of polynomials, proof composition, aggregation of queries, etc. Many details.

Second proof by Dinur (2005) that is essentially combinatorial. Relies on recursion and expander graphs.

Implies that Max-3Sat is NP-hard to approximate within $1 - \epsilon$ for a (very small) constant ϵ.

Improving constants

A long story, one final point:

Theorem [H]: For any $\epsilon > 0$, it is NP-hard to approximate Max-3-Lin within $1/2 + \epsilon$.

Matches mindless algorithm up to ϵ. No nontrivial approximation in non-satisfiable case.

Improving constants

A long story, one final point:

Theorem [H]: For any $\epsilon > 0$, it is NP-hard to approximate Max-3-Lin within $1/2 + \epsilon$.

Matches mindless algorithm up to ϵ. No nontrivial approximation in non-satisfiable case.

Fear realized in worst possible way.

Ingredients in proof/construction

- Two prover games.
- Parallel repetition for two-prover games. [R]
- Coding strings by the long code. [BGS]
- Using discrete Fourier transforms in the analysis. [H]

Other problems

Independent Set: Given a graph on n vertices find the largest set of vertices not supporting any edge.

Coloring: Color the vertices with as few colors as possible.

Other problems

Independent Set: Given a graph on n vertices find the largest set of vertices not supporting any edge.

Coloring: Color the vertices with as few colors as possible.

Both hard to approximate within $n^{1-\epsilon}$ for any $\epsilon > 0$.

Famous conjecture in area

Khot in 2002 proposed the Unique Games Conjecture.

If true, the GW-algorithm for Max-Cut described is optimal.

Many problems

Take your favorite NP-hard optimization problem, how hard is it to approximate in polynomial time?

I do not remember all, but does anyone have a favorite problem?

Switcing topics

End of approximation part.

Cryptography and Complexity

The breaking of a cryptosystem is usually a problem in NP and we want(?) this to be hard.

An average case question and hence not possible to use NP-completeness which is worst case.

Asymmetric cryptography

Public key crypto-systems, Key Exchange, Signatures.

Requires extra structure.

Most used systems rely on either integer factorization or discrete logarithms in finite fields being difficult.

Quantum computers

Computers that use the quantumness of nature.

The memory of the computer can be in a superposition of states.

A (very) limited form of parallelism.

Quantum computers

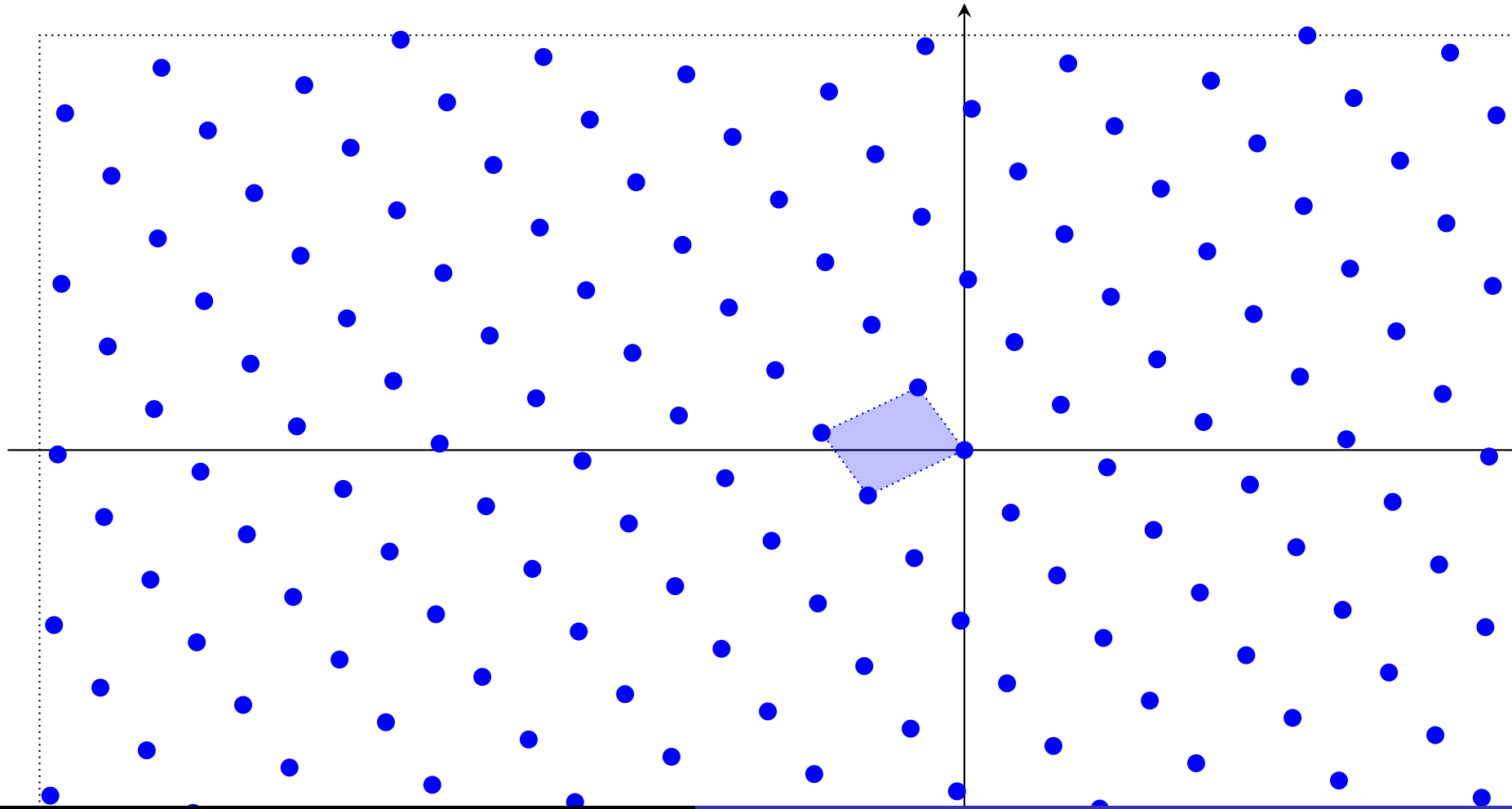
Computers that use the quantumness of nature.

The memory of the computer can be in a superposition of states.

A (very) limited form of parallelism.

Peter Shor discovered quantum algorithms that efficiently factor integers and compute discrete logarithms.

A lattice L



Replacement primitives

Finding the shortest vector in an integer lattice in high (200-1000) dimensions seems like a hard problem.

Crypto-systems have been proposed based on this.

Basic questions for lattice based crypto

1. Are given proposals secure from attacks by classical and/or quantum computers?
2. Find other constructions.

Our study of this is only starting and even if I had time I have little to say.

The end