On the efficient approximability of constraint satisfaction problems

Johan Håstad

KTH Numerical Analysis
and Computer Science

July 13, 2007

My world

Efficient computation.

- P Polynomial time
- BPP Probabilistic Polynomial time (still efficient)
- NP Non-deterministic Polynomial time

Randomness in computation

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

Whether needed is a very basic theoretical question.

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.

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, assumed hard.

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

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)

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

Basics for 3-Sat

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, Gaussian elimination does not seem to do anything interesting.

Max-3-Lin vs Max-3-Sat

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

$$\left\{\begin{array}{ccccccc} x_1 & + & x_2 & + & x_3 & = & 1 \\ x_1 & + & x_2 & & & = & 1 \\ x_1 & & & + & x_3 & = & 1 \\ & & x_2 & + & x_3 & = & 1 \\ x_1 & & & & & = & 1 \\ & & x_2 & & & = & 1 \\ & & & & x_3 & = & 1 \end{array}\right. \qquad \bmod 2$$

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 with looking at the constraints.

The mindless algorithm

Give each variable a random value with looking at the constraints.

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

The mindless algorithm

Give each variable a random value with looking at the constraints.

For Max-3-Sat with each clause of length 3 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.

The basic question

For which types of constraints can we beat the random mindless algorithm and on what instances?

- As soon as optimal value is significantly better than random, i.e. $(1+\epsilon)$ times random fraction.
- When the optimal value is (very) large, i.e. $(1-\epsilon)m$.
- When we can satisfy all constraints, satisfiable instances.

Two branches

- Positive results. Efficient algorithms with provable performance ratios.
- Negative results. Proving that certain tasks are NP-hard, or possibly hard given some other complexity assumption.

The favorite techniques

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

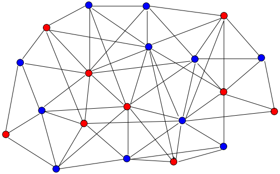
Lower bounds: The PCP-theorem and its consequences. Arora, Lund, Motwani, Sudan and Szegedy.

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.



Famous 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 $y_{ii} = 1$.

Positive semidefinite matrices?

Y symmetric matrix is positive semidefinite iff one of the following is true

- All eigenvalues $\lambda_i \geq 0$.
- $z^T Y z \geq 0$ for any vector $z \in R^n$.
- $Y = V^T V$ for some matrix V.

$y_{ij} = x_i x_j$ is in matrix language $Y = xx^T$.

By a result by Alizadeh we can to any desired accuracy solve

$$\max \sum_{ij} c_{ij} y_{ij}$$

subject to

$$\sum_{ij} a_{ij}^k y_{ij} \leq b^k$$

and Y positive semidefinite.

By a result by Alizadeh we can to any desired accuracy solve

$$\max \sum_{ij} c_{ij} y_{ij}$$

subject to

$$\sum_{ij} a_{ij}^{k} y_{ij} \leq b^{k}$$

and Y positive semidefinite.

Intuitive reason, set of PSD is convex and we should be able to find optimum of linear function (as is true for LP).

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.

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

Great suggestion by GW.

Given vector solution v_i pick random vector r and set

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

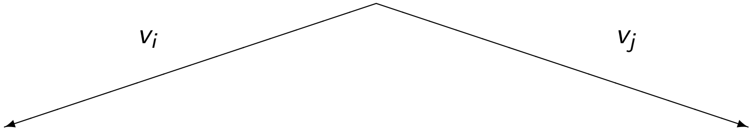
where (v_i, r) is the inner product.

Intuition of rounding

Contribution

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

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

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

Minimal quotient gives approximation ratio

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

Immediate other application

Original GW-paper derived same bound for approximating Max-2-Sat.

Improved [LLZ] to $\approx .9401$ (not analytically proved).

"Obvious" semi-definite program. More complicated rounding.

Immediate other application

Original GW-paper derived same bound for approximating Max-2-Sat.

Improved [LLZ] to $\approx .9401$ (not analytically proved).

"Obvious" semi-definite program. More complicated rounding.

Many other applications some using many additional ideas.

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 $s/c + \epsilon$.

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 $1/2$.

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.

Interesting new 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.

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

These basic proofs give BAD inapproximability constants

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]

Classifying CSPs

1. Hard to approximate better than random mindless algorithm on satisfiable instances.
2. Hard to do better than random mindless algorithm on (almost) satisfiable instances.
3. Can beat random mindless algorithm as soon as optimal is significantly better than random.
4. Have an approximation constant better than achieved by random mindless algorithm, but not in previous class.

Classifying CSPs

1. Hard to approximate better than random mindless algorithm on satisfiable instances. Approximation resistant on satisfiable instances, (Max-3-Sat).
2. Hard to do better than random mindless algorithm on (almost) satisfiable instances.
3. Can beat random mindless algorithm as soon as optimal is significantly better than random.
4. Have an approximation constant better than achieved by random mindless algorithm, but not in previous class.

Classifying CSPs

1. Hard to approximate better than random mindless algorithm on satisfiable instances. Approximation resistant on satisfiable instances, (Max-3-Sat).
2. Hard to do better than random mindless algorithm on (almost) satisfiable instances. Approximation resistant, (Max-3-Lin).
3. Can beat random mindless algorithm as soon as optimal is significantly better than random.
4. Have an approximation constant better than achieved by random mindless algorithm, but not in previous class.

Classifying CSPs

1. Hard to approximate better than random mindless algorithm on satisfiable instances. Approximation resistant on satisfiable instances, (Max-3-Sat).
2. Hard to do better than random mindless algorithm on (almost) satisfiable instances. Approximation resistant, (Max-3-Lin).
3. Can beat random mindless algorithm as soon as optimal is significantly better than random. Fully approximable, (Max-Cut).
4. Have an approximation constant better than achieved by random mindless algorithm, but not in previous class.

Classifying CSPs

1. Hard to approximate better than random mindless algorithm on satisfiable instances. Approximation resistant on satisfiable instances, (Max-3-Sat).
2. Hard to do better than random mindless algorithm on (almost) satisfiable instances. Approximation resistant, (Max-3-Lin).
3. Can beat random mindless algorithm as soon as optimal is significantly better than random. Fully approximable, (Max-Cut).
4. Have an approximation constant better than achieved by random mindless algorithm, but not in previous class. Somewhat approximation resistant.

Constraints of 2 variables

All such predicates are fully approximable, even over larger domains.

Semi-definite programming is all powerful.

Constraints of 3 variables

- Approximation resistant iff we accept either all strings of even parity or all strings of odd parity.
- Fully approximable (class 3) iff un-correlated with parity of all three variables.

Other (nontrivial) cases belong to class 4.

Constraints of 3 variables

- Approximation resistant iff we accept either all strings of even parity or all strings of odd parity.
- Fully approximable (class 3) iff un-correlated with parity of all three variables. Reduces to (real) sum of predicates on two variables.

Other (nontrivial) cases belong to class 4.

Constraints of 3 variables

- Approximation resistant iff we accept either all strings of even parity or all strings of odd parity.
- Fully approximable (class 3) iff un-correlated with parity of all three variables. Reduces to (real) sum of predicates on two variables.

Other (nontrivial) cases belong to class 4.
Approximation resistance applies to satisfiable instances if accepts at least 6 inputs.

Unknown width 3

What happens with the "not two ones" predicate on satisfiable instances.

Could we do better than random?

Not true for just $(1-\epsilon)$-satisfiable instances!

Unknown width 3

What happens with the "not two ones" predicate on satisfiable instances.

Could we do better than random?

Not true for just $(1-\epsilon)$-satisfiable instances!

Parity is different for satisfiable and almost satisfiable instances!

Width 4

Partial classification by Hast.

400 essentially different predicates.

- 79 approximation resistant.
- 275 not approximation resistant.
- 46 not classified.

# Acc	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Non-res	1	4	6	19	27	50	50	52	27	26	9	3	1	0	0
Res	0	0	0	0	0	0	0	16	6	22	11	15	4	4	1
Unkn	0	0	0	0	0	0	6	6	23	2	7	1	1	0	0

Satisfiability ignored.

Large width

General facts, assume width k

- Accepts very few inputs, non-trivially approximable.
- Exists rather sparse approximation resistant predicates.
- The really dense predicates are approximation resistant.

General result on sparse predicates

Any k-ary Boolean predicate can be approximated within $ck2^{-k}$ [CMM].

No predicate with $\leq ck$ accepting configurations is approximation resistant.

General result on sparse predicates

Any k-ary Boolean predicate can be approximated within $ck2^{-k}$ [CMM].

No predicate with $\leq ck$ accepting configurations is approximation resistant.

Uses semi-definite programming.

Sparse resistant predicates

For any l_1 and l_2 there are predicates on $k = l_1 + l_2 + l_1 l_2$ Boolean variables that accept $2^{l_1+l_2}$ vectors and are approximation resistant.

Only $2^{O(\sqrt{k})}$ accepted inputs.

Very dense predicates

If $k \geq l_1 + l_2 + l_1 l_2$ any predicate on k Boolean variables that rejects fewer than $2^{l_1 l_2}$ inputs is approximation resistant [Hast].

This is $2^{o(k)}$ but still a reasonable number. For small k constants can be improved.

General fact?

It seems like the more inputs a predicate accepts the more likely it is to be approximation resistant.

General fact?

It seems like the more inputs a predicate accepts the more likely it is to be approximation resistant.

Approximation resistance is not a monotone property. Have example P, Q,

$$P(x) \rightarrow Q(x)$$

P approximation resistant.

Q not approximation resistant.

Asymptotic question

For large k is a random predicate of Boolean variables approximation resistant?

Asymptotic question

For large k is a random predicate of Boolean variables approximation resistant?

Probably ...

Unique games conjecture.

Made by Khot.

CSP of width 2 over large domains. $C_i(x_a, x_b)$, for each value of x_a exists a unique value of x_b to satisfy the constraint and vice versa.

Unique games conjecture.

Made by Khot.

CSP of width 2 over large domains. $C_i(x_a, x_b)$, for each value of x_a exists a unique value of x_b to satisfy the constraint and vice versa.

Conjecture: Hard to distinguish $(1-\epsilon)$-satisfiable from ϵ-satisfiable.

Unique games conjecture.

Made by Khot.

CSP of width 2 over large domains. $C_i(x_a, x_b)$, for each value of x_a exists a unique value of x_b to satisfy the constraint and vice versa.

Conjecture: Hard to distinguish $(1-\epsilon)$-satisfiable from ϵ-satisfiable.

True? A new complexity class?

Consequences of UGC

Many, some central:

Constant α_{GW} sharp for Max-Cut [KKMO].

Vertex Cover is hard to approximate within $2 - \epsilon$ [KR].

Optimal constant $\approx .9401$ for Max-2-Sat [A].

Random predicates are approximation resistant [H].

Summing up

We have a huge classification problem ahead of us.

NP-completeness of decision problem is almost universally true and understood since the 1970-ies [S].

Approximation resistance on satisfiable instances means that efficient computation cannot do anything. Much stronger notion of hardness.

Summing up

We have a huge classification problem ahead of us.

NP-completeness of decision problem is almost universally true and understood since the 1970-ies [S].

Approximation resistance on satisfiable instances means that efficient computation cannot do anything. Much stronger notion of hardness.

Open: Settle the unique games conjecture!

Summing up

We have a huge classification problem ahead of us.

NP-completeness of decision problem is almost universally true and understood since the 1970-ies [S].

Approximation resistance on satisfiable instances means that efficient computation cannot do anything. Much stronger notion of hardness.

Open: Settle the unique games conjecture!

Most basic fact: Max-3-Sat is approximation resistant on satisfiable instances.