Fine-Grained Complexity of Solving Systems of Polynomial Equations (over small finite fields)

Ryan Williams MIT

Polynomial Systems over Finite Fields

Let $k \in \mathbb{Z}^+$ and p be prime.

Degree-k F_p System Solvability (DkSp)

Given: Set S of polys $q_1, \dots, q_m \in F_p[x_1, \dots, x_n]$, $deg(q_i) \leq k$

Decide: Is $Z(S) = \{a \in F_p^n \mid \forall i, q_i(a) = 0\}$ empty?

This talk: think of p and k as small (constant), n as large, $p \ll k$

D1Sp $\in$ **P** **(Gaussian elimination)**

DkSp $\in$ **P** for $m = 1$ **(Finding a root of one degree-k polynomial)**

D2S2 is **NP-hard** **(Reduction from NAE-3-SAT)**

For small n and k, several algorithms are known

(e.g. **[Kayal'14]** runs in $\boldsymbol{poly(m, k^{exp(n)}, log(p))}$ **time**)

For our case: best algorithm for DkSp (up until ~2 years ago) $\approx \boldsymbol{p^n}$

Thm [LPTWY'17] DkSp can be solved in $p^{n-n/O(k)}$ time (for $p \leq 2^{O(k)}$)

Degree-k F_p System Solvability (DkSp)
Given: Set S of polys $q_1, \dots, q_m \in F_p[x_1, \dots, x_n]$
Decide: Is $Z(S) = \{a \in F_p^n \mid \forall i, q_i(a) = 0\}$ em

Derandomized Algorithm:
Use ϵ-biased generators to choose the polys,
use mod-amplifying polys over $\mathbb{Z}$ to compose

Thm [LPTWY'17] DkSp can be solved in $p^{n-n/O(k)}$ time (for $p \le 2^{O(k)}$)

General Idea: Approach the problem in a "circuit complexity way"

Given: Set $S = \{q_i(\vec{y}, \vec{x})\}$ with δn y-vars and $n - \delta n$ x-vars ($\delta \in (0,1)$)

Define an $(n - \delta n)$-input circuit:

$$C_S(\vec{x}) := \bigvee_{a \in F_p^{\delta n}} \left(\bigwedge_{i=1}^{m} [q_i(a, \vec{x}) = 0] \right)$$

Obs: $\left(\exists\, b \in F_p^{n-\delta n}\right)[C_S(b) = 1]$
$\Leftrightarrow (\exists\, (a,b))[(a,b) \in Z(S)]$

Lemma [Adapting Razborov-Smolensky 87/88]: Can *randomly* reduce C_S to an F_p-poly Q_S of degree $\approx p\delta nk$ such that for all $b \in F_p^{n-\delta n}$,

$$C_S(b) = 1 \Rightarrow Pr[Q_S(b) \neq 0] > \frac{2}{3}$$

$$C_S(b) = 0 \Rightarrow Pr[Q_S(b) = 0] < \frac{1}{3}$$

(1) doesn't take too long for "small" p
(Q has degree $\approx pn/100$)
(2) can be done in $\approx p^{n-\delta n}$ time by a divide-and-conquer approach

Algorithm. Set $\delta = 1/(100k)$.
Given S, try for $10\, n \log(p)$ times:
(1) Construct random Q_S.
(2) Eval $Q_S(b)$ on all $b \in F_p^{n-\delta n}$
Return **"solution"**
$\Leftrightarrow \exists b\; Q_S(b) \neq 0$ for > ½ trials

Counting Solutions to Poly Systems

Let $k \in \mathbb{Z}^+$ and p be prime.

#DkSp (Counting Solutions to Degree-k Systems over F_p)

Given: Set S of polys $q_1, \ldots, q_m \in F_p[x_1, \ldots, x_n]$, $deg(q_i) \leq k$

Output: cardinality of $Z(S) = \{a \in F_p^n \mid \forall i, q_i(a) = 0\}$

#D1Sp $\in$ **P** **(Gaussian elimination)**

#D2Sp $\in$ **P** for $m = 1$ [Carlitz69,Woods98,…]

[LPTWY'17] $p^{n-n/O_p(k)}$-time det. algorithm for #DkSp

#D2S2 is **#P-hard** (reduction from NAE-3-SAT)

#D3S2 remains **#P-hard** **even for $m = 1$** ([EK'90])

… but the reduction (from 3SAT) blows up # of variables

How hard is it to count zeroes of an $O(1)$-degree F_2-polynomial?

Might we expect a 1.99^n time algorithm?

Recall: ***finding*** a zero of one polynomial is relatively easy!

Strong Hardness of Counting

Thm [with Brynmor Chapman]
For all $\epsilon > 0$, $c > 1$, there's a deterministic $p^{\epsilon n}$-time reduction
from **#D$\boldsymbol{k}$S$\boldsymbol{p}$** with n vars and cn polynomials
to counting zeroes of **ONE** degree-$\boldsymbol{O(ck/\epsilon)}$ poly with n vars

Corollary Counting solutions to a system of degree-k polynomials
is fine-grained equivalent to
counting solutions to ***one*** degree-O(k) polynomial!

YAARS (Yet Another Approach to Refuting SETH?)

To solve $\boldsymbol{k}$-SAT in $\boldsymbol{1.999^n}$ time, it suffices to count the zeroes of a given $\boldsymbol{O_k(1)}$-degree polynomial in $\boldsymbol{n}$ variables over $\boldsymbol{F_2}$, in $\boldsymbol{O(1.99^n)}$ time.

Thm For all $\epsilon > 0$, there's a det. $p^{\epsilon n}$-time reduction
from **#D*k*S*p*** with n vars and cn polynomials
to counting zeroes of **ONE** degree-$O(ck/\epsilon)$ poly with n vars

First, assume the number of polynomials in our system is $m = \epsilon n$

Reduction:

Input $q_1, \ldots, q_{\epsilon n}$

Let $\{v_1, \ldots, v_{p^{\epsilon n}}\} = F_p^{\epsilon n}$

$Z := 0, N := 0$

For all $i = 1, \ldots, p^{\epsilon n}$,

set $P_i(x) := \sum_j v_i[j] \cdot q_j(x)$ ← **degree *k***

$Z = Z + (\#\text{zeroes of } P_i(x))$ ← **oracle call**

$N = N + (\#\text{zeroes of } 1 - P_i(x))$ ← **oracle call**

Output $(Z - N)/p^{\epsilon n}$

Analysis: Let $A = \{a \mid \forall i, q_i(a) = 0\}$ be the set of solutions to the system

Every $a \in A$ is a zero of P_i, and **not** of $1 - P_i$, **for all *i*** — ***Every a ∈ A* contributes 1**

Every $a \notin A$ is a zero of P_i for ***exactly*** $1/p$ of the i,
and is a zero of $1 - P_i$ for ***exactly*** $1/p$ of the i — ***Every a ∉ A* contributes 0**

So under our assumption, the output is the correct count!

Thm For all $\epsilon > 0$, there's a det. $p^{\epsilon n}$-time reduction
from **#D$\boldsymbol{k}$S$\boldsymbol{p}$** with n vars and cn polynomials
to counting zeroes of **ONE** degree-$O(ck/\epsilon)$ poly with n vars

Now we reduce to the case where the number of polys = $\boldsymbol{\epsilon n}$ …

Reduction:

Input $q_1, \ldots, q_{cn}$**, each of deg.** $\boldsymbol{k}$

• • •

Output $\boldsymbol{P_1, \ldots, P_{\epsilon n}}$**, each of deg.** $O(ck/\epsilon)$

Goal: Number of sols to $q_1 = 0, \ldots, q_{cn} = 0$

= Number of sols to $\boldsymbol{P_1 = 0, \ldots, P_{\epsilon n} = 0}$

Thm For all $\epsilon > 0$, there's a det. $p^{\epsilon n}$-time reduction from **#DkSp** with n vars and cn polynomials to counting zeroes of **ONE** degree-$O(ck/\epsilon)$ poly with n vars

Now we reduce to the case where the number of polys = ϵn …

Reduction:

First try:

Input $q_1, \ldots, q_{cn}$, each of deg. k

Partition the set of polys into groups $G_1, \ldots, G_{\epsilon n}$, where each G_i has $O(c/\epsilon)$ polys.

For all $i = 1, \ldots, \epsilon n$

$$P_i(x) := 1 - \prod_{q_j \in G_i} \left(1 - q_j(x)^{p-1}\right)$$

← **Simple version with degree $O(ckp/\epsilon)$**

Output $P_1, \ldots, P_{\epsilon n}$, each of deg. $O(ckp/\epsilon)$

Goal: Number of sols to $q_1 = 0, \ldots, q_{cn} = 0$

= Number of sols to $P_1 = 0, \ldots, P_{\epsilon n} = 0$

Analysis: For all $a \in F_p^n$, and all i, $P_i(a) = 0 \Leftrightarrow$ for all $q_j \in G_i$, $q_j(a) = 0$

So a is a solution to the original system $\Leftrightarrow$ a is a solution to the new system!

Final Reduction: Run the above reduction to get ϵn polys, then run the reduction from the previous slide

Thm For all $\epsilon > 0$, there's a det. $p^{\epsilon n}$-time reduction
from **#D*k*S*p*** with n vars and cn polynomials
to counting zeroes of **ONE** degree-$O(ck/\epsilon)$ poly with n vars

Now we reduce to the case where the number of polys = $\boldsymbol{\epsilon n}$ …

Reduction:

Input $q_1, \ldots, q_{cn}$**, each of deg.** $\boldsymbol{k}$
Partition the set of polys into groups $G_1, \ldots, G_{\epsilon n}$,
where each G_i has $\boldsymbol{O(c/\epsilon)}$ polys.
For all $\boldsymbol{i = 1, \ldots, \epsilon n}$

$$\boldsymbol{P_i(x) \coloneqq 1 - \prod_{p_j \in G_i}\left(1 - q_j(x)^{p-1}\right)}$$

Output $P_1, \ldots, P_{\epsilon n}$**, each of deg.** $O(ck\boldsymbol{p}/\epsilon)$
Goal: Number of sols to $q_1 = 0, \ldots, q_{cn} = 0$
= Number of sols to $\boldsymbol{P_1 = 0, \ldots, P_{\epsilon n} = 0}$

To improve the degree of the reduction to $\boldsymbol{O(ck/\epsilon)}$:

Brynmor's Lemma: Given 2^t polynomials $\{q_i\}$ of degree d over any prime field F_p, we can construct a polynomial P of degree $2^t d$ so that for all $a \in F_p^n$,

$$\boldsymbol{P(a) = 0} \Leftrightarrow \text{for all } j,\ q_j(a) = 0$$

No dependence on $\boldsymbol{p}$. Degree upper bound is tight!

Brynmor's Lemma: Given 2^t polynomials $\{q_i\}$ of degree d over any prime field F_p, we can construct a polynomial P of degree $2^t d$ so that for all $a \in F_p^n$,

$$P(a) = 0 \Leftrightarrow \text{for all } j,\ q_j(a) = 0$$

No dependence on p. Degree upper bound is tight!

Proof: WLOG $p > 2$.

Induction on t. Base case ($t = 0$) is trivial.

By induction, there are polynomials f and g of degree $2^{t-1}d$ such that $f(a) = g(a) = 0 \Leftrightarrow$ for all j, $q_j(a) = 0$ (apply f to half of the system, and g to the other half).

Let $\beta \in F_p - \{0\}$ so that β is not a perfect square (not a QR mod p).

Take $P(x) = f^2(x) - \beta g^2(x)$. Note P has degree $2^t d$.

Let $a \in F_p^n$. Since β is a *not* a QR mod p,

either $\beta g^2(a) = 0$, or $\beta g^2(a)$ is a (nonzero) non-QR mod p.

On the other hand, either $f^2(a)$ is 0 or it is a (nonzero) QR mod p.

It follows that $P(a) = 0$ iff $f^2(a) = \beta g^2(a) = 0$ iff $f(a) = g(a) = 0$.

QED

(Unconditional) Lower Bounds from Fine-Grained Counting

$\sum \circ \boldsymbol{POLYd}[\boldsymbol{p}]$:

Real-valued linear combinations of functions $f: \{0,1\}^n \to \{0,1,\ldots,p-1\}$
where for every f there is a degree-d polynomial $q(x)$ such that
$\forall x \in \{0,1\}^n$, $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{q}(\boldsymbol{x})$ **mod** $\boldsymbol{p}$

Case of $\boldsymbol{d} = 2, \boldsymbol{p} = 2$ is already very interesting!

Compelling Conjecture ["Degree-Two Uncertainty Principle"]:
$\boldsymbol{AND}$ **(on $\boldsymbol{n}$ inputs) requires $\boldsymbol{n}^{\omega(1)}$-size** $\sum \circ \boldsymbol{POLY2}[2]$

Known: $\boldsymbol{AND}$ requires $\Omega(2^n)$-size $\sum \circ \boldsymbol{POLY1}[2]$
$\boldsymbol{AND}$ has $O(2^{n/2})$-size $\sum \circ \boldsymbol{POLY2}[2]$
No non-trivial lower bounds were known for $\sum \circ \boldsymbol{POLY2}[\boldsymbol{p}]$

Using algorithm for #D$\boldsymbol{d}$S$\boldsymbol{p}$:
Thm [W'18] $\forall \boldsymbol{d}, \boldsymbol{k}, \forall \boldsymbol{p}$ prime, $\exists \boldsymbol{f_k} \in \boldsymbol{NP}$ without $\boldsymbol{n^k}$-size $\sum \circ \boldsymbol{POLYd}[\boldsymbol{p}]$

Recall: It is a *major* open problem to prove
$\exists \boldsymbol{f} \in \boldsymbol{NP}$ without $\boldsymbol{n^k}$-size (unrestricted) circuits

Two Open Questions

1. Improve the $p^{n-\frac{n}{O(k)}}$ running time for DkSp?

Some heuristic reasons to believe that $p^{n-\frac{n\,log(k)}{O(k)}}$ time is possible…
If that is true, then the "Super Strong ETH" is false!

2. Is #DkSp with *one* polynomial $\equiv$ #DkSp in general?

Our $2^{\epsilon n}$-time reduction from #DkSp to one polynomial
blows up the degree by an $O\left(\frac{1}{\epsilon}\right)$ factor…

Note: If the answer is "yes" for $k = 2$
with a sub-exptime reduction, then ETH is false

Thank you!