Solving Multivariate Polynomial Systems

Presented by: Bo-Yin Yang

work with Lab of Yang and Cheng, and
Charles Bouillaguet, ENS

Institute of Information Science
and TWISC, Academia Sinica
Taipei, Taiwan
mschen@crypto.tw

Feb. 9, 2010

Most "Practical" Generic Solution for $\mathbb{F}_2$?

How to solve a "generic" system of m equations, n variables over $\mathbb{F}_2$, not too overdetermined, n mid-sized, say 50 or 70.

Most “Practical” Generic Solution for $\mathbb{F}_2$?

How to solve a “generic” system of m equations, n variables over $\mathbb{F}_2$, not too overdetermined, n mid-sized, say 50 or 70.

$\mathcal{MQ}$ (Multivariate Quadratics) Problem $\mathcal{MQ}(q; n, m)$

Want a solution for n variables in m quadratic equations, All coefficients and variables in $\mathbb{F}_q$, if one exists.

- NP-Hard even when $q = 2$

Most "Practical" Generic Solution for $\mathbb{F}_2$?

How to solve a "generic" system of m equations, n variables over $\mathbb{F}_2$, not too overdetermined, n mid-sized, say 50 or 70.

$\mathcal{MQ}$ (Multivariate Quadratics) Problem $\mathcal{MQ}(q; n, m)$

Want a solution for n variables in m quadratic equations, All coefficients and variables in $\mathbb{F}_q$, if one exists.

- NP-Hard even when $q = 2$
- Conjectured Probabilistically Hard [Berbain-Gilbert-Patarin]

Most "Practical" Generic Solution for $\mathbb{F}_2$?

How to solve a "generic" system of m equations, n variables over $\mathbb{F}_2$, not too overdetermined, n mid-sized, say 50 or 70.

$\mathcal{MQ}$ (Multivariate Quadratics) Problem $\mathcal{MQ}(q; n, m)$

Want a solution for n variables in m quadratic equations, All coefficients and variables in $\mathbb{F}_q$, if one exists.

- NP-Hard even when $q = 2$
- Conjectured Probabilistically Hard [Berbain-Gilbert-Patarin]

There has been many recent advances in system-solving, what is the best practical way to solve this? $\mathbf{F_4}$? $\mathbf{F_5}$? XL variant?

Solving for 32 $\mathbb{F}_2$ Variables from Quadratics

MAGMA-2.15 results

More equations $\longrightarrow$ Easier system:

- 32 Equations: 55 Gigabytes, 2.2 days on 2.2GHz Opteron core
- 64 Equations: 2.5 Gigabytes, 3 hours on 2.2GHz Opteron core
- 320 Equations: 0.2 Gigabytes, 4.1 seconds on a 2.2GHz Opteron core

Can we do better?

Solving for 32 $\mathbb{F}_2$ Variables from Quadratics

`MAGMA-2.15` results

More equations $\longrightarrow$ Easier system:

- 32 Equations: 55 Gigabytes, 2.2 days on 2.2GHz Opteron core
- 64 Equations: 2.5 Gigabytes, 3 hours on 2.2GHz Opteron core
- 320 Equations: 0.2 Gigabytes, 4.1 seconds on a 2.2GHz Opteron core

Can we do better?

A Smarter Brute-Force

We implemented an idea brought to us by Charles Bouillaguet

- 2.2 GHz Opteron core: 3.579 seconds
- nVidia GTX 280 (1.296 GHz): 0.05 seconds

Larger/Higher Systems?

MAGMA-2.15 performance

Solves 20 cubics in 20 $\mathbb{F}_2$ variables in about 9000 s on Opteron 2.2GHz; runs out of memory with 23 variables and 23 equations.

Larger/Higher Systems?

MAGMA-2.15 performance

Solves 20 cubics in 20 $\mathbb{F}_2$ variables in about 9000 s on Opteron 2.2GHz; runs out of memory with 23 variables and 23 equations.

36 $\mathbb{F}_2$ vars in 36+ eqs, 1 core, C2Q 9550 2.83 GHz

- Quadratics: 9.0 s
- Cubics: 23.2 s
- Quartics: 55.8 s

Larger/Higher Systems?

MAGMA-2.15 performance

Solves 20 cubics in 20 $\mathbb{F}_2$ variables in about 9000 s on Opteron 2.2GHz; runs out of memory with 23 variables and 23 equations.

36 $\mathbb{F}_2$ vars in 36+ eqs, 1 core, C2Q 9550 2.83 GHz

- Quadratics: 9.0 s
- Cubics: 23.2 s
- Quartics: 55.8 s

48 $\mathbb{F}_2$ vars in 48+ eqs, GTX 280

- Quadratics: 41 mins
- Cubics: 73 mins
- Quartics: 280 mins

Why?

$o(1) \neq 0$

$\mathbf{F_4}$ solving n $\mathbb{F}_2$ equations in as many variables should take $2^{(0.78+o(1))n}$ time if we can apply sparse matrix techniques. But won't beat brute-force in number of logical ops until $n = 200$.

Memory Effects

Systems with Large Memory are slower: some claims $O(M^{1/2})$ slowdown.

Better Enumeration thru Gray Code

Via tracking all successive differentials, solving for n $\mathbb{F}_2$ variables from degree-d systems takes $O(2^n \cdot d \cdot \mathrm{polylog}(n))$ time.

What Now?

Thanks to the Other People

Charles Bouillaguet of ENS, Hsieh-Chung Kevin Chen, Ming-Shing Chen, Chen-Mou Cheng, Tony Chou, Ruben Niederhagen at our Lab.

Take-Away Point

For not-too-overdetermined $\mathbb{F}_2$ systems in the practical range, Brute Force works better than $\mathbf{F_4}$ and other Gröbner basis solvers. This affects, for example, security guarantees of `QUAD` stream ciphers.

Future

Results will be submitted and on ePrint archive soon.