Lecture 10— Extended Frege and Interpolation

Massimo Lauria — lauria.massimo@gmail.com
Office 1107, Ookayama West 8th Building

Friday — November 27th, 2015 (This document was updated on June 21, 2017)

We introduce a remarkably powerful proof system, called extended Frege. We show that if this proof system has an efficient algorithm for interpolation, then it is possible to break RSA.

The main result presented in the lecture is an example of this process of extracting computation from proofs.

Theorem (Krajíček, Pudlák 1998)**.** *If proof system Extended Frege has an efficient interpolant, then RSA crypto system can be broken.*

Extended Frege (EF) proofs

The extended frege proof is one of the strongest proof systems studied in proof complexity. Essentially is an inference system in which each line of the proof is a polynomial size circuit. There are several way to define it and we will see some of them.

The language of the proof systems seen so far are too weak to express a formula as a proof line, therefore in those systems we turn the problem of proving a tautology into the equivalent problem of refuting a CNF. In proof systems as EF lines can be formulas, therefore it is possible to discuss proofs of tautologies directly. A treatment of EF is in Krajíček book.[1]

[1] Chapter 4 in *Bounded Arithmetic, Propositional Logic, and Complexity Theory* (Cambridge University Press)

Frege system A Frege system is the textbook proof system (actually a family of proof systems) in propositional calculus. Example of such systems are the propositional parts of "Hilbert's system" and Gentzen sequent calculus LK.

Consider propositional variables $x_1, x_2, \ldots, x_n$ we have an inference system in which each line is a propositional formula over this variables. There are multiple ways to define a Frege system and we will see that it does not make much difference. In particular a common definition is made by

- A finite number of constant size axiom schemes $\frac{}{A_1}$; $\frac{}{A_2}$; …; $\frac{}{A_\ell}$ where each A_i is a formula on variables $p_1, \ldots, p_k$ over some fixed language;
- and a modus ponens rule

$$\frac{A \qquad A \to B}{B}\,.$$

A proof of a formula ϕ is a sequence of formulas $\phi_1, \ldots, \phi_T$ where $\phi_T = \phi$ and where each ϕ_i is either

- equal to some A_j were each variable p_i is substituted with arbitrary formulas over $x_1, x_2, \ldots, x_n$;
- obtained by modus ponens from two previous proof lines.

Definition 1 (Frege)**.** *A Frege system is a proof system as above where the sequence of axioms schemes is finite, sound and implicationally complete over its language.*[2]

The length of a proof is the number of lines, the size of a proof is the number of distinct subformulas contained in all formulas in the proof.

[2] A proof system is complete when it proves every tautology over its language. It is implicationally complete if for every tautology $A \to B$ the system can also prove B using A as non logical axiom.

A natural example of language is $\{0, 1, \neg, \vee, \wedge, \to, =, \neq\}$ An important fact about Frege system is that definition is flexible and robust. Indeed the specific axiom scheme is not important, as long as it satisfies the definition.[3]

[3]

Theorem 2. *(Reckhow, 1976) Any Frege proof system polynomially simulates any other Frege system on the same language. Moreover, if the language of a Frege system F_1 contains the language of a system F_2 then F_1 has at most a polynomial speed-up over F_2.*

The theorem refers, among others, also to the propositional part of sequent calculus LK.

Extended Frege Expressing computation by formulas is not always efficient. In general computation is expressed efficiently by circuits, which seem to be more powerful than formulas.

EF uses this intuition to provide a seemingly more powerful proof system than Frege, by the means of abbreviations.

Definition 3 (Extended Frege)**.** *Given any Frege system F, An* Extended Frege *proof of ϕ is a sequence of formulas $\phi_1, \ldots, \phi_T$ such that $\phi_T = \phi$ and every ϕ_i is either obtained from previous formulas using a rule in F, or ϕ_i has the form*[4]

$$y = \psi$$

where

- *variable y does not appear neither in ψ nor in ϕ_j for $j < i$;*
- *variable y does not appear in ϕ.*

The latter type of formula is called an extension axiom *and y is called an* extension variable*.*

[4] If $y = \psi$ is not allowed in the language of F, alternative formulations are either

$$(y \wedge \psi) \vee (\neg y \wedge \neg \psi)$$

or

$$(y \vee \neg \psi) \wedge (\neg y \vee \psi) \ .$$

The previous definition of Extended Frege is based on extension axioms. Nevertheless there are other definitions that are equivalent.

Definition 4 (Circuit Frege (informal))**.** *Circuit Frege is similar to a Frege system with some modifications. Consider the axiom schemes in the definition of Frege where, instead of formulas, $A_1, \ldots A_\ell$ are circuits and where during the application of an axiom scheme the variables p_i are substituted by circuits over $x_1, x_2, \ldots, x_n$.*

Fact 5. *Circuit Frege is polynomially equivalent to Extended Frege.*

Another interesting variant of Extended Frege is Extended Resolution. This is particularly interesting because some SAT solvers represent preprocessing and inprocessing techniques in some version of Extended Resolution.

Definition 6 (Extended Resolution)**.** *Consider a CNF ϕ over variables $x_1, \dots, x_n$. An Extended Resolution refutation of ϕ is essentially a resolution refutation where axioms are either*

- *initial clauses of ϕ; or*
- *a clause of the form $\neg y_i \vee E_i$; or*
- *a clause of the form $y_i \vee \neg \ell_i$ for some $\ell_i \in E_i$;*

where E_i is a clause (specified once for all in the proof) associated to y_i which contains only variables among $\{x_1, \dots, x_n\}$ and $\{y_1, \dots, y_{i-1}\}$.

Even if Resolution is not a Frege system, extension axioms make the system as powerful as extended Frege.

Fact 7. *Extended Resolution is polynomially equivalent to Extended Frege over CNF refutations.*

Interpolant

We recall the definition of interpolant, given already in Lecture 6. As in that case, we use a "disjoint NP-pair": a pair of NP predicates $\exists y A_0(x,y)$ and $\exists z A_1(x,z)$ with common variables and empty intersection.

Definition 8. *Given a formula $A_0(x,y) \wedge A_1(x,z)$, a function $I(x)$ with $\{0,1\}$ values* interpolates *the formula if for for every assignment $\vec{v}$,*

$$I(\vec{v}) = \begin{cases} 0 & \text{implies } A_1(\vec{v},y) \text{ is unsatisfiable;} \\ 1 & \text{implies } A_0(\vec{v},z) \text{ is unsatisfiable.} \end{cases} \tag{1}$$

Essentially

$$A_1(x,y) \longrightarrow I(x) \longrightarrow \neg A_0(x,z)\ . \tag{2}$$

The main result of the lecture is due to (Krajíček, Pudlák 1998)[5] and essentially consists in showing a disjoint pair A_0, A_1 based on RSA encryption scheme such that

- EF proves efficiently that $A_0 \wedge A_1$ is a contradiction;
- any interpolant between A_0 and A_1 can be used to efficiently break RSA.

[5] Jan Krajíček and Pavel Pudlák. Some consequences of cryptographical conjectures for s_2^1 and ef. *Information and Computation*, 140(1):82–94, 1998

RSA system

The RSA encryption scheme[6] is a public key encryption scheme that works as follow:

1. Pick two sufficiently large primes at random p and q;

[6] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978

2. Fix $N := p \cdot q$;

3. Pick $1 < e < N$ and fix (N, e) to be the public key;

4. Pick d such that $e \cdot d \equiv 1 \pmod{\varphi(N)}$ to be the private key;[7]

Encryption $E(x) := x^e \pmod{N}$;

Decryption $D(y) := y^d \pmod{N}$;.

where encryption and decryptions are defined over the invertible in $\mathbb{Z}_N$, which are the elements of $\mathbb{Z}_N$ coprime with N. When $y \equiv x^e \pmod{N}$ then

$$y^d \equiv x^{de} = x^{1+t\cdot\varphi(N)} \equiv x \pmod{N}. \tag{3}$$

Observe that the security of RSA hinges on the difficulty of computing inverses modulo $\varphi(N)$. We won't discuss about the security of the actual scheme in depth, but notice that knowing $\varphi(N)$ is equivalent to factoring N itself. Indeed

$$\varphi(N) = (p-1)(q-1) = pq - (p+q) + 1 = N - (p+q) + 1, \tag{4}$$

therefore we can express the degree two polynomial $(x-q)(x-q)$ as

$$(x-p)(x-q) = x^2 - (p+q)x + pq = x^2 - (N - \varphi(N) + 1)x + N. \tag{5}$$

Probabilistic encription of one bit Consider the encryption scheme that has as input a single bit b and

1. picks a random $1 < x < N$ which is equal to $b \pmod 2$;

2. RSA encrypts x into $y = E(x)$ using key (N, e).

Fix $n = \lceil \log N \rceil$. In (Alexi et al., 1988)[8] they show that any algorithm that guesses the bit b from y with probability (over random choice of x and internal random coin flips) at least $\frac{1}{2} + \frac{1}{\text{poly}(n)}$, can be use as a subroutine to get a randomized polynomial time algorithm that inverts RSA encryption with high probability.

Since we are discussing about circuits, we can reformulate the previous statement

Theorem 9 (Alexi et al., 1988)**.** *If there is a circuit of size polynomial in n that computes the bit b from the scheme above, then there is a polynomial size circuit that recorver x from input (N, e) and $E(x)$.*

[7] The Euler $\varphi(N)$ function computes the number of invertible elements in $1, \ldots, N$, Namely the elements that are coprime with N. For $N = p \cdot q$ where p and q are distinct primes $\varphi(N) = (p-1)(q-1)$. It is simple to see that for every $1 \leq x < N$ with $\gcd(x, N) = 1$,

$$x^{\varphi(N)} \equiv 1 \pmod{N}.$$

[8] Werner Alexi, Benny Chor, Oded Goldreich, and Claus P Schnorr. Rsa and rabin functions: Certain parts are as hard as the whole. *SIAM Journal on Computing*, 17(2):194–209, 1988

Conditional lower bounds

We will define two predicates $\mathcal{A}_0$ (i.e. Null input) and $\mathcal{A}_1$ (i.e. One input) which say respectively that the bit b encrypted is zero and one.

$$\begin{aligned}\mathcal{A}_i = \{(N,e,y) \text{ such that } \exists x, d < N \text{ where} \\ x \equiv b \pmod 2 \wedge \\ x^e \equiv y \pmod N \wedge \\ y^d \equiv x \pmod N \wedge \\ \gcd(y,N) = 1\} \end{aligned} \tag{6}$$

We define A_i to be the relation defining $\mathcal{A}_i$, namely

$$\begin{aligned}A_i(N,e,y;x,d) = x \equiv b \pmod 2 \wedge \\ x^e \equiv y \pmod N \wedge \\ y^d \equiv x \pmod N \wedge \\ \gcd(y,N) = 1 \end{aligned} \tag{7}$$

The interpolant $I(N,e,y)$ for the formula for

$$A_0(N,e,y;x_0,d_0) \wedge A_1(N,e,y;x_1,d_1) \tag{8}$$

separates the inputs (N,e,y) where y is the encryption of some odd x using (N,e), from the ones where y is encryption of some even x (see Figure 1).

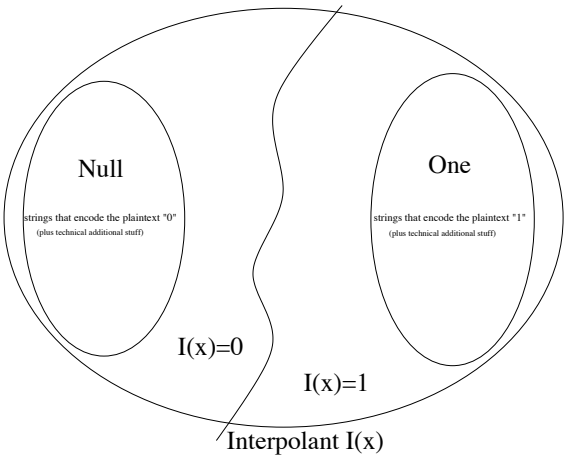

Figure 1: An interpolant is a separator circuit for the two sets Null and One. (Picture by Sebastian Müller)

The formula $A_0 \wedge A_1$ is a contradictory statement expressed in the language of arithmetic. The theory of bounded arithmetic called S_2^1 captures very well polynomial computation,. Such theory is a fragment of Peano Arithmetic.

- It's language allow to build functions of polynomial growth with respect to the length (in binary) of their inputs;
- It has a form of efficiently verifiable induction

$$\left[f(0) \wedge \forall x\, f\left(\left\lfloor \frac{x}{2} \right\rfloor\right) \to f(x)\right] \text{ implies } \forall x\, f(x)\,. \tag{9}$$

- Under certain conditions (met in particular by $A_0 \wedge A_1$) a S^1_2 proof Π of an arithmetic statement can be efficiently translated into a sequence of refutations $\{\Pi_n\}_n$ of size $n^{O(1)}$ where each such Π_n is an Extended Frege refutation of the propositional encoding of $A_0 \wedge A_1$ over numbers of length n.

Theorem 10 (Krajíček, Pudlák 1998)**.** *Theory S^1_2 refutes*

$$A_0(N, e, y; x_0, d_0) \wedge A_1(N, e, y; x_1, d_1) \ .$$

Corollary 11 (Krajíček, Pudlák 1998)**.** *For every $n > 0$, consider the propositional encoding of the arithmetic statement*

$$A_0(N, e, y; x_0, d_0) \wedge A_1(N, e, y; x_1, d_1)$$

over numbers of n bits. EF has a polynomial size refutation of this statement.

Corollary 12 (Krajíček, Pudlák 1998)**.** *If RSA is secure, then Extended Frege has no efficient interpolant, not even when N is promised to be the product of two primes.*

In the following we are going to show that $A_0 \wedge A_1$ is a contradiction, but we are not going to give all details of the proof in the theory S^1_2, otherwise we should define it formally. We will argue that (as the authors did in their paper) that the proof is explicit enough to be doable in S^1_2, and hence in Extended Frege.

Proof sketch of Theorem 10. Suppose we have x_0 and $x_1 \leq N$, d_0, d_1 and we have that $\gcd(y, N) = 1$, and

$$x_i^e \equiv y \pmod{N} \wedge y^{d_i} \equiv x_i \pmod{N} \tag{10}$$

for $i = 0, 1$. Now fix $r := d_0 \cdot e - 1$,

$$y^{r+1} \equiv y^{d_0 e} \equiv x_0^e \equiv y \pmod{N} \tag{11}$$

and since $\gcd(y, N) = 1$ then y is efficiently invertible with respect to N using Euclid's algorithm so we know that

$$y^r \equiv 1 \pmod{N} \ . \tag{12}$$

We use this to show that

$$x_i^r \equiv y^{d_i r} \equiv (y^r)^{d_i} \equiv 1 \pmod{N} \ , \tag{13}$$

for $i = 0, 1$. Now we can see that for both $i = 0, 1$ we get that

$$y^{d_0} \equiv x_i^{d_0 e} \equiv x_i^{r+1} \equiv x_i \pmod{N} \ , \tag{14}$$

for $i = 0, 1$. So we proved that $y^{d_0} \equiv x_0 \equiv x_1 \pmod{N}$ and since $1 < x_0, x_1 < N$ then $x_0 = x_1$ which contraddicts the hypothesis.

We didn't show that this proof is doable in S^1_2. An important point for the translation is that we don't mention prime numbers since the correctness of primality testing (despite the problem being in P) may not be provable in S^1_2. Instead we just used the correctness of Euclid algorithm for computing gcd, and properties of exponentiations like $a^{b+c} = a^b a^c$ and $a^{bc} = (a^b)^c$. □

Further extensions

Extended Frege is a very powerful proof system therefore is it quite believable that it has no interpolation. Actually it is debatable whether confidence in the security of RSA adds anything to this belief.

There have been other results along these lines, where interpolation circuits or automatization algorithm for proof systems weaker than EF have been used to solve hard computational problems.[9] This means that such algorithms are unlikely to exists.

[9] Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel, and Toniann Pitassi. Non-automatizability of bounded-depth frege proofs. *Computational Complexity*, 13(1-2):47–68, 2004; Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for frege systems. *SIAM J. Comput.*, 29(6):1939–1967, 2000; L. Huang and Toniann Pitassi. Automatizability and simple stochastic games. *Automata, Languages and Programming*, pages 605–617, 2011; and Albert Atserias and E. Maneva. Mean-payoff games and propositional proofs. *Information and Computation*, 2011

References

[ACGS88] Werner Alexi, Benny Chor, Oded Goldreich, and Claus P Schnorr. Rsa and rabin functions: Certain parts are as hard as the whole. *SIAM Journal on Computing*, 17(2):194–209, 1988.

[AM11] Albert Atserias and E. Maneva. Mean-payoff games and propositional proofs. *Information and Computation*, 2011.

[BDG+04] Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel, and Toniann Pitassi. Non-automatizability of bounded-depth frege proofs. *Computational Complexity*, 13(1-2):47–68, 2004.

[BPR00] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for frege systems. *SIAM J. Comput.*, 29(6):1939–1967, 2000.

[HP11] L. Huang and Toniann Pitassi. Automatizability and simple stochastic games. *Automata, Languages and Programming*, pages 605–617, 2011.

[KP98] Jan Krajíček and Pavel Pudlák. Some consequences of cryptographical conjectures for s^1_2 and ef. *Information and Computation*, 140(1):82–94, 1998.

[RSA78] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.