6.876/18.426: Advanced Cryptography March 12, 2003

Lecture 10: Non-Interactive ZK Proofs for all of NP

1 Recap

Last lecture we defined the notion of a non-interactive proof system (NIPS), and constructed a non-interactve *zero-knowledge* (NIZK) proof system for a "special" language of pairs: $L = \{(n, y) : n \in 2PP \wedge y \in QNR \cap J_{+1}\}^1$.

Today, we'll see how to use the proof system constructed for this "special" language L as a stepping stone in building a computational NIZK proof system for *any* NP language.

2 (Bounded) NIZK Proof Systems Revisited

Recall that we say that (P, V) is a (perfect) NIZK proof system for language L if the following conditions hold:

1. **Completeness:** $\forall x \in L$, $\forall w \in W_x$, $Pr[\sigma \leftarrow \{0,1\}^{|x|^c}; \pi \leftarrow \mathsf{P}(x, \sigma, w) : \mathsf{V}(x, \sigma, \pi) =$ "YES" $) > 1 - negl(|\sigma|)$

2. **(Strong) Soundness:** $\forall \mathsf{P}'$, $Pr[\sigma \leftarrow \{0,1\}^{n^c}; (x', \pi') \leftarrow \mathsf{P}'(\sigma) : \mathsf{V}(x', \sigma, \pi') =$ "YES"$) < negl(|\sigma|)$

   Note that here, $|x|$ and n are polynomially related whenever P' is PPT. However, technically, we should enforce this property even for unbounded P'.

3. **ZKness:** $\exists \mathsf{S}_{PPT}$, $\forall x \in L, \forall a$, $\mathsf{S}(x, a) = VIEW(x, a)$.

Remarks

Recall that σ denotes a common random reference string, which is in open view of both parties, and is fixed outside of their control (e.g. the Rand reference table).

Note throughout we may think of P as being PPT, since he receives a witness $w \in W_x$ as input, where W_x denotes the set of all witnesses for the theorem $x \in L$. Alternatively, we may think of P as being of unbounded power as usual, who then can compute a witness from the common input.

The statement of soundness above is the strengthened form from last lecture. That is we assume that the malicious prover P' possesses an algorithm which, when given a particular reference string σ, will generate a pair (x', π'), such that π' is P's attempt at a "proof"

[1]Here and following we omit the understood modulus n and just write $J_{+1} = \{y : (\frac{y}{n}) = 1\}$ for the set of all remainders mod n with Jacobi symbol +1).

(with respect to σ) of the *false* theorem $x' \notin L$; i.e, *after* seeing σ the malicious prover P' gets to choose the particular false theorem x' whose validity he will try to convince V of.

"Bounded" NIPS

We will say that a proof system (P, V) satisfying the above statements of (1) Completeness and (2) Soundness is a bounded NIPS (non-interactive proof system) for language L. This highlights the fact that the length of any theorem x which we may prove in such a system is bounded above by a (polynomial) function of the length of the reference string σ. Or, the other way around, if we want to prove a theorem x of a certain length, this will require access to a sufficiently long reference string σ. For example, in last lecture in the proof system for the special language L, in order to prove $(n, y) \in L$ we assumed that the given σ was long enough so that we could subdivide it into many sections each of length $|n|$.

3 Applications of NIZK on the horizon

The power of the ZK protocols we've seen previously depended on *interaction* and *randomness* (unpredictability of the verifier's coin flips). In contrast, the concept of a NIZK proof is very attractive: the verifier V never talks, and hence we don't need to refer at all to a malicious V' in the definition of ZKness. Herein lies a great advantage of NIZK proof systems. Indeed, the $VIEW$ referred to in the ZKness definition above is given without machine parameters, since the $VIEW$ of a NIZK proof essentially contains: $(\sigma, \pi_\sigma(x))$, where $\pi_\sigma(x)$ is a proof of the common input theorem x, given with respect to the σ, the externally fixed, publicly viewable reference string. This freedom will prove to be extremely useful in constructing secure multi-party computation protocols, something to be seen later in the course.

4 Observations on NIZK for L and beyond

Up to now we have just seen a single example of a NIZK proof system, the one from last lecture for the language of pairs:

$L = \{(n, y) : n \in 2PP \wedge y \in QNR \cap J_{+1}\}$.

In this lecture we will see how to generalize from a NIZK proof system for the special language L to one for *any* NP language.

4.1 Limitations of NIZK for L and beyond

1. Boundedness: n must be short relative to σ.
2. This L is just one "special" language.
3. The quantification of σ gives less than ideal strength to the definition.

Let's examine the above limitations one by one.

Boundedness

We will not focus on refining the notion of boundedness today, though we note it may be useful to think of $|\sigma| \leq poly(|x|, k)$, i.e. with an additional dependency on some independent security parameter k.

"Specialness" of L

We'd like to be able to extend such an attractive concept as NIZK beyond the special language L. Recall the previous progression we saw for general ZK, from ZK for ISO, then NISO, and later 3-colorability, which gave us all of NP. Here again, the choice of the "right" NP-complete language is particularly important in order to make the difficulties in reduction go away. In the case of NIZK, we'll choose $3SAT$ as our intermediate step, for reasons perhaps mysterious at present.

Timing of σ

Finally, consider the timing (relative to the theorem x to be proved) of when the reference string σ must be fixed.

For completeness, it doesn't matter whether σ is generated before or after x is specified, since all parties are honest.

As for soundness, note that the definition of soundness above says that no malicious prover P', even if first given σ, will be able to find a false theorem x' that it can generate an acceptable proof for, except with negligible probability.

As for zero-knowledgeness, recall that the simulator S we constructed in the last lecture *had* to be given the freedom to determine σ *after* the theorem x to be proved was fixed; the σ it produced depended explicitly on x. Specifically, given $(n, y) \in L$, S constructed a σ (with the correct distribution) in part by squaring random elements of Z_n^* and multiplying roughly half of these squares by y.

In fact the requirement that σ be able to be chosen independently by the simulator S *after* the theorem x is fixed seems to be an *intrinsic* limitation of NIZK proof systems in general, although we don't know how to prove that this is the case. In particular this would spell problems for NIZK proofs for languages containing statements about σ itself. That is, we would get into trouble trying to simulate a proof of something like "σ factors as the product of three distinct primes".

5 A Bounded NIZK Proof System for $3SAT$

Recall that the language $3SAT$ consists of all satisfiable formulas ϕ which consist of conjunctions of 3-literal disjunctive clauses: $3SAT = \{\phi : \phi = C_1 \wedge C_2 \wedge \cdots \wedge C_r\}$, where each clause C_s is of the form: $(X_{s_1} \vee X_{s_2} \vee X_{s_3})$, and where each literal X_{s_t} represents either the boolean variable x_{s_t} or its negation $\overline{x}_{s_t}$.

Given a formula $\phi \in 3SAT$ with satisfying assignment (witness) $w = (x_1, x_2, \ldots, x_m)$, the NIZK prover's task is to construct a proof π using the reference string σ which will convince V that $\phi \in 3SAT$, without disclosing any knowledge about the witness w.

In order to realize this, we resort to a useful tool: reduce to a previously solved problem. Namely, P will subdivide his proof in two parts: $\pi = (\pi_1, \pi_2)$, and will divide the reference string into two equal halves: $\sigma||\tau$. The first proof, the *auxiliary proof* π_1, will refer to σ

and will be a NIZK proof of membership for some pair: $(n, y) \in L$, where L is the special language we have already seen. The second part, π_2, will refer to τ and will use the (n, y) pair to convince V that a special (knowledge-hiding) encoding e of w does in fact correspond to a satisfying assignment of ϕ.

Encoding an assignment

We create the encoding $e = (e_1, e_2, \ldots e_m)$ of the assignment $w = (x_1, x_2, \ldots, x_m)$ by selecting $e_i \in J_{+1} \subset Z_n^*$ as follows: If $x_i = 0$ (FALSE), then we set e_i to be a random square mod n. If $x_i = 1$ (TRUE), then we set e_i to be a random non-square mod n which is also in J_{+1}. Note that if the Quadratic Residuosity Assumption (QRA) holds, then disclosing this encoding will reveal no knowledge about the witness w.

Given such an encoding e, V can check in polynomial time that e is properly formed (i.e. $\forall i, e_i \in J_{+1}$), and thus corresponds to *some* assignment on the variables. P must provide evidence that the assignment to which e corresponds actually satisfies ϕ.

5.1 A Number Theory Aside

Here we recall the definition of the Jacobi symbol of $z \in Z_n^*$ modulo composite n as an extension of Legendre symbols (RHS below), where the modulus n prime factors as $n = p_1^{h_1} \cdots p_k^{h_k}$:

$$\left(\frac{z}{n}\right) = \prod_{i=1}^{k} \left(\frac{z}{p_i}\right)^{h_i}.$$

Consider the subset of Jacobi symbol $+1$ elements $J_{+1} \subset Z_n^*$. Next, consider triples of elements $(a_1, a_2, a_3) \in J_{+1} \times J_{+1} \times J_{+1}$. We define an equivalence relation on triples from this special set. We write $(a_1, a_2, a_3) \approx (b_1, b_2, b_3)$ iff a_i has the same "quadratic character" (i.e. "square" vs. "non-square") as b_i for $i = 1, 2, 3$.

A nice characteristic of this equivalence relation is that for $n \in 2PP$,

$$(a_1, a_2, a_3) \approx (b_1, b_2, b_3) \Leftrightarrow \exists \sqrt{a_1 b_1}, \sqrt{a_2 b_2}, \sqrt{a_3 b_3} \bmod n.$$

Thus, producing three such roots as on the RHS above would constitute a short proof of the equivalence relation. The equivalence above holds for $n = p_1^{h_1} p_2^{h_2} \in 2PP$ because for such moduli n, any element $z \in J_{+1}$ must have Jacobi symbols modulo p_1, p_2 of either $(+1, +1)$ or $(-1, -1)$, depending on whether z is a square or non-square, respectively; i.e. for $n \in 2PP$, J_{+1} consists of $1/2$ squares and $1/2$ non-squares.

Next we will see how P can leverage such proofs of this equivalence relation to convince V that his encoding e actually corresponds to a satisfying assignment of the formula ϕ.

5.2 A Bounded NIZK proof system for $3SAT$ (continued)

Proving satisfaction of a clause in ϕ from the encoding

By the nature of $3SAT$, P must convince V that every clause C_h in ϕ contains at least one true literal. Suppose for example that $C_h = (x_i \vee \overline{x}_j \vee x_k)$. Under the encoding from above, P's task is equivalent to proving that at least one of e_i, e_k is a non-square, or e_j is a square. This in turn is equivalent to proving that at least one of (e_i, ye_j, e_k) is a non-square. This follows from the fact that V has already been convinced by π_1 that $(n, y) \in L$. Consequently, the conditions $n \in 2PP$ and $y \in QNR \cap J_{+1}$ imply that ye_j will be a non-square mod n iff e_j is a square[2]. Thus, V should believe that multiplying any e_t from the encoding by y will "flip" its quadratic character, from square to non-square or vice-versa.

So P may complete the second half π_2 of his proof as follows. For each 3-literal disjunctive clause $(X_{s_1} \vee X_{s_2} \vee X_{s_3})$ in ϕ, P will use the second half of the reference string, τ, to give evidence that at least one element of the triple $\{e_{s_1}$ (or ye_{s_1}), e_{s_2} (or ye_{s_2}), e_{s_3} (or ye_{s_3})$\}$ is a non-square, where each choice, e_{s_i} (or ye_{s_i}), is made as in the example above on the basis of whether $X_{s_i} = x_i$ or $X_{s_i} = \overline{x}_i$, respectively.

Details of P's strategy

To accomplish this, P first subdivides τ into m sections of equal length $\tau_1, \ldots, \tau_m$, where m is the number of clauses in ϕ. τ itself must be chosen long enough so the rest of the proof has a high probability of going through (Completeness exponentially close to 1).

Now wlog, fix a clause C_h from ϕ to consider, and for simplicity of notation let us assume it has a particular form: $C_h = (x_i \vee \overline{x}_j \vee x_k)$.

P can construct a NIZK proof that C_h contains at least one true literal as follows:

1. Divide the subsection $\tau_h \subset \tau$ into a sequence of consecutive triples. Each triple should consist of three strings, each of length $|n|$.

2. "Purify" this sequence of triples by completely removing every triple in which any one of the three strings corresponds to the binary representation of a number not in $J_{+1} \subset Z_n^*$. This will leave a refined sequence of triples: $\tau'_h = \tau_{h1}, \ldots, \tau_{h\alpha}$, for some α. Furthermore, each of these remaining triples should contain three numbers that are distributed uniformly at random over J_{+1}.

3. For every 3-square triple $(sq_1, sq_2, sq_3) \in \tau'_h$, write down a proof (z_1, z_2, z_3) of its quadratic character, such that $z_i^2 = sq_i$ for $i = 1, 2, 3$. On average this will identify a subset S_1 of $\alpha/8$ triples from τ'_h (since as noted above, for odd $n \in 2PP$, J_{+1} will consist of 1/2 squares and 1/2 non-squares).

4. Consider the triple $E_h = (e_i, ye_j, e_k)$ derived from the clause we are considering $C_h = (x_i \vee \overline{x}_j \vee x_k)$. Since the encoding e is of a true satisfying assignment on the variables, we know in particular that the triple E_h above must contain at least one non-square mod n. Therefore the triple E_h lies in a quadratic character equivalence class different from the 3-square class containing the $\alpha/8$ triples identified in the previous step.

   So for every triple in τ' from the same equivalence class as E_h, write down a proof of the equivalence $\approx E_h$. For example, suppose $(x_i, \overline{x}_j, x_k) = (0, 0, 1)$. Then $e_i =$ (square), $ye_j =$ (square), $e_k =$ (non-square), and so $E_h \in$ (square, square, non-square). So for every triple in τ' of the form $(sq_1, sq_2, \overline{sq}_3)$ write down a proof of

[2]Note that for general n this equivalence does *not* hold, since in general the product of two non-squares mod n might produce another non-square. For example, if $n \in 3PP$, we might have two non-squares z_1, z_2 whose Jacobi symbols relative to the three prime factors are given by the triples $(+1, +1, -1)$ and $(-1, +1, +1)$; their product $z_1 z_2$, having triple $(-1, +1, -1)$, is also a non-square.

equivalence $\approx E_h$ as: $(\sqrt{sq_1e_i}, \sqrt{sq_2ye_j}, \sqrt{sq_3e_k}$ (see number theory notes above). In general this will identify a subset S_2 of roughly $\alpha/8$ *additional* triples from τ'_h belonging to some equivalence class *other* than the 3-square class from the previous step, and so $S_2 \cap S_1 = \emptyset$. Indeed the equivalence class of E_h must be different than the 3-square class, since C_h will be satisfied iff at least one *non-square* is present in E_h.

If P follows the above sequence of steps for each clause C_i in ϕ, and for each clause uses as purified reference string the corresponding (fresh) section τ'_i from τ, then if the above conditions are met, V will be convinced that each clause is satisfied, i.e. $\phi \in 3SAT$.

(Note it is important that no portion of the reference string τ is reused for proving equivalences for two different clauses, since this might reveal knowledge of correlation between truth values of literals from these two clauses)

Conditions on V's acceptance

We stipulate that in order for V to accept that a given C_h is satisfied, the sizes of the two disjoint sets of triples S_1, S_2 identified by P in the last two steps above must satisfy $\alpha/8 - \alpha/32 \leq |S_1|, |S_2| \leq \alpha/8 + \alpha/32$. As long as $\phi \in 3SAT$, this condition will be met with probability exponentially close to 1 given a sufficiently long choice of τ.

6 (P, V) is a bounded NIZK proof system for 3SAT

Completeness

Completeness holds with probability exponentially close to 1. The first part of the proof, π_1, was seen last lecture to be NIZK with exponentially small failure rate.

As for the second part of the proof, π_2, we see from the above steps that failure will occur when for some clause C_h the corresponding string τ_h happens to contain too few (or too many) representatives of the required equivalence classes, (3-square and the class corresponding to the clause in question). But this will only occur with exponentially small probability.

Soundness

There are two possible scenarios under which a malicious P' could successfully deceive V.

First, note that P$'$ can almost always succeed in deceiving V (into accepting a false theorem $\phi' \notin 3SAT$) if after P$'$ looks at σ he is then able to come up with a "proof" π'_1 which fools V into accepting some invalid pair $(n', y') \notin L$. For example, after fooling V into accepting the invalid pair $(n', y') \notin L$, P$'$ then may construct the false theorem $\phi' \notin 3SAT$ whose 8 clauses are all the possible triples of literals for the 3 variables x_1, x_2, x_3. This formula is not satisfiable, and yet if P$'$ assigns all three variables the value 1 (TRUE), he can almost always fool V into accepting. Indeed, by the first deception V will incorrectly believe that $y' \in QNR$, so P should be able to (deceitfully) persuade V that each of the 8 clauses' encodings (after *purported* inversions, where appropriate) belongs to an equivalence class distinct from that of 3-square triples, namely the class of 3 non-squares. (This will be possible as long as each subsection of τ contains roughly the expected number of triples from the 3 non-square equivalence class; hence the "almost always" condition above on the success of P$'$ under these conditions).

However, the above scenario is very unlikely to occur, because by the strong soundness of the NIZK proof system for L, even after a σ is fixed, the probability that such a false pair (n', y') together with a "proof" π_1' that deceives V is found by the polynomial-time subroutine of P' is exponentially small.

Second, now assuming that P' is using a valid pair $(n, y) \in L$, the only way he could succeed at deceiving V is if for every unsatisfied clause C_h in ϕ an extraordinary number (roughly $2\alpha/8$, *twice* the expected value) of 3-square triples happen to be present in the substring τ_h. This is necessary, since to fake a proof that E_h contains at least one non-square, *all* of the distinct $|S_1| + |S_2| \approx 2\alpha/8$ triples from τ_h' identified in the last two steps of P's algorithm above would in fact have to belong to the *same* (3-square) equivalence class. The possibility of this event is equivalent to flipping a biased coin for which $Pr[HEADS] = 1/8$ some number of times and observing that a $1/4$ fraction of the outcomes are $HEADS$. This probability will be exponentially small for a large enough number of coin tosses, i.e. for a sufficiently long reference string τ_h.

Zero-Knowledgeness: The Simulator S

Here we describe an efficient simulator algorithm S which outputs a view whose distribution is indistinguishable (by non-uniform circuits under the QRA assumption) from that generated by the real NIZK prover P.

Consider the task facing the simulator S. Given only a formula ($\phi \in 3SAT$), S must produce a simulated view: $\mathsf{S}(\phi) = (\sigma || \tau, \pi_1, \pi_2)$.

S proceeds as follows:

1. Generate an invalid pair $(n, y) \notin L$ by randomly selecting an appropriate length $n \in 2PP$, but choosing a random *square* $y \in J_{+1}$, (rather than y a random *non*-square $\in J_{+1}$).

2. Use the NIZK simulator S_L which exists for language L to compute $(\sigma, \pi_1) = \mathsf{S}_L(n, y)$. Note that because the QR decision problem $y \overset{?}{\in} QR_n$ is 50/50 hard for *each instance* $y \in J_{+1}$ (or all but negligibly many), then intuitively the output distribution of $\mathsf{S}_L(y)$ for random $y \in QR_n$ should be indistinguishable from that of $\mathsf{S}_L(x)$ for random $y \in J_{+1} \cap QNR_n$. So the output (σ, π_1) of $\mathsf{S}_L(n, y)$ in this step should be indistinguishable from an auxiliary proof given by P. The difference is that the σ which $\mathsf{S}_L(n, y)$ generates in this step using $y \in QR_n$ will contain only elements of J_{+1} which are squares.

3. Generate an encoding $(\hat{e}_1, \ldots, \hat{e}_m)$, consisting entirely of random *non*-square $\hat{e}_i \in J_{+1}$. Recall this would correspond to a uniform setting of TRUE on all the variables in ϕ, but by the QRA, V will not be able to discern the extremely "special" nature of this artificial encoding.

4. Generate a sufficiently long random string τ and subdivide it into m sections, one for each clause in ϕ.

5. From here, proceed just at the prover P would from above. For example, again consider wlog some clause in ϕ of the form: $C_h = (x_i \vee \overline{x}_j \vee x_k)$. To simulate a NIZK proof

that the clause C_h contains at least one true literal, use the refined section τ'_h, divided into consecutive triples of elements of Z_n^*. First identify a (roughly) 1/8 fraction of 3-square triples as above by providing square roots (S is able to do this because himself generated n and so knows its prime factorization).

6. Identify a roughly 1/8 disjoint fraction of triples consisting of 3 *non*-squares. Then prove that $\hat{E}_h = (\hat{e}_i, y\hat{e}_j, \hat{e}_k)$ lies in the same equivalence class as each of these new triples (S again uses the factorization of n). Note that regardless of the structure of the 3-literal clause in question, by the choice of this special encoding (all non-square $\hat{e}_t$) and $y \in QR_n$, the corresponding $\hat{E}_h$ will *always* consist of three non-squares.

   The concatenation of these subproofs for each clause C_h in ϕ will give the second half of the simulated proof π_2, which refers to the random string τ.

7. Output $(\sigma||\tau, \pi_1, \pi_2)$.

Proof of Indistinguishability
The details of the indistinguishability proof are somewhat complex. The essential ingredient is the QRA. Here we just recall as mentioned above that the first half of the simulated view, (σ, π_1), intuitively should be indistinguishable from that resulting from a genuine auxiliary proof of P, by virtue of the fact that Quadratic Residusity is a hard decision problem for almost all instances. That is, we need the fact that the output of $S_L(n, y)$ with y a random square mod n will be indistinguishable from $S_L(n, y)$ with y a random non-square in J_{+1}. This is a subtle but crucial point in the proof.
Indistinguishability of the second half of the simulated view, (τ, π_2), can also be argued under the QRA.
Note, however, that we can only prove one theorem with a single σ. Otherwise, clues could leak about correlations between clauses: for instance, if the first clause in each had different truth values of their literals, an observer would notice this.

We have thus motivated the fact that the system (P, V) described above is bounded NIZK for the (carefully chosen!) NP-complete language $3SAT$. Thus we have seen that the attractive concept of bounded NIZK can be implemented for any NP language, under the QRA.

7 NIZK for NP under general complexity assumptions

Note that undergirding today's construction is the Quadratic Residuosity Assumption (QRA), by way of the auxiliary proof of membership for the special language L from last lecture. It has been shown in [FLS90] and [BeYu96] that NIZK proof systems for any NP language can be constructed under the general complexity assumption of one-way trapdoor permutations (whose existence is implied for example by the QRA used here). It remains an open question whether the same result is possible assuming only general one-way Functions.

A question was raised: Isn't there a Black-Box separation between one-way permutations and NIZK? More to come on this perhaps in future lectures.

A similar progression from special to general assumptions may also be observed in the development of other cryptographic tools. Often it is the case that:

1. An interesting notion is latched onto.
2. The notion is instantiated, perhaps using some very ad hoc assumptions.
3. Further work shows how the same tool can be constructed under some "well-established" number-theoretic assumption.
4. Finally, one tries to relax the construction to the most general assumptions possible, such as any abstract one-way function or trapdoor permutation.

As an example of the above process, consider some historical instances of PRG's and PRF's, together with their assumptions (discrete log, factoring, any OWF). Also, consider the recent development of verifiable random functions.

Please note: some section headings and structure of the above notes were informed by [BlDeMiPe91].

References

[BeYu96] M. Bellare and M. Yung. Certifying Permutations: Non-Interactive Zero-Knowledge Based on any Trapdoor Permutation. Journal of Cryptology. Vol. 9, No. 1, Winter 1996, pp. 149-166.

[BlDeMiPe91] M. Blum, A. De Santis, S. Micali, and G. Persiano. Non-Interactive Zero Knowledge. SIAM J. Comput. Vol. 20, No. 6, pp. 1084-1118, 1991.

[FLS90] U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Based on a Single Random String. Proc. 31st Ann. Symp. Found. Comput. Sci. pp. 308-317, 1990.