CSA E0 235: Cryptography 30 Sep 2019

Lecture 15

Instructor: Arpita Patra *Submitted by: Rameesh Paul*

1 Introduction

In this lecture we will discuss **Zero Knowledge Proofs**, a completely different paradigm for proving things compared to our traditional Euclidean style proofs. The idea was introduced by Turing Award winners Shafi Goldwasser, Silvio Micali along with Charles Rackoff in their landmark paper, "The Knowledge Complexity of Interactive Proof-Systems" published in STOC-1985.
Traditional proof systems have two shortcomings -

- Proof itself needs to be disclosed to the verifier
- Running time of the verifier needs to grow with the length of the proof

In this lecture we will see how we can overcome first limitation through Zero Knowledge proofs. To realize this we will allow the prover and the verifier to be probabilistic and also allow them to interact with each other. Put more formally it means that they can be modelled as interactive probabilistic polynomial time Turing Machines.

1.1 Motivation for Zero Knowledge

The problem of Zero Knowledge is interesting and self-motivated in itself. However in this section we still present a few motivating problems from Computer Science domain.

- **Factoring large numbers** - The problem of factoring large numbers, when they are product of equal sized primes is believed to be a hard problem. Let's say Alice claims to know factors of a large number N.

  $$N = p.q; \quad p, q : 512\text{-bit primes}$$

  The goal of Alice is to convince Bob that she knows such p, q. An obvious way to do that would be to send p, q itself to Bob. But this could really be catastrophic for Alice. Say a scenario where N is a part of public key of Alice for encryption and the decryption key depends on p, q.
  What Zero-Knowledge accomplishes is that it allows Alice to convince Bob that she indeed knows such p, q without actually revealing them.

- **Graph isomorphism** - The problem of Graph isomorphism is also believed to be a hard problem for large graphs. Now let's say Alice claims to know the two graphs

  $$G_1 = (V_1, E_1) \text{ is isomorphic to } G_2 = (V_2, E_2)$$

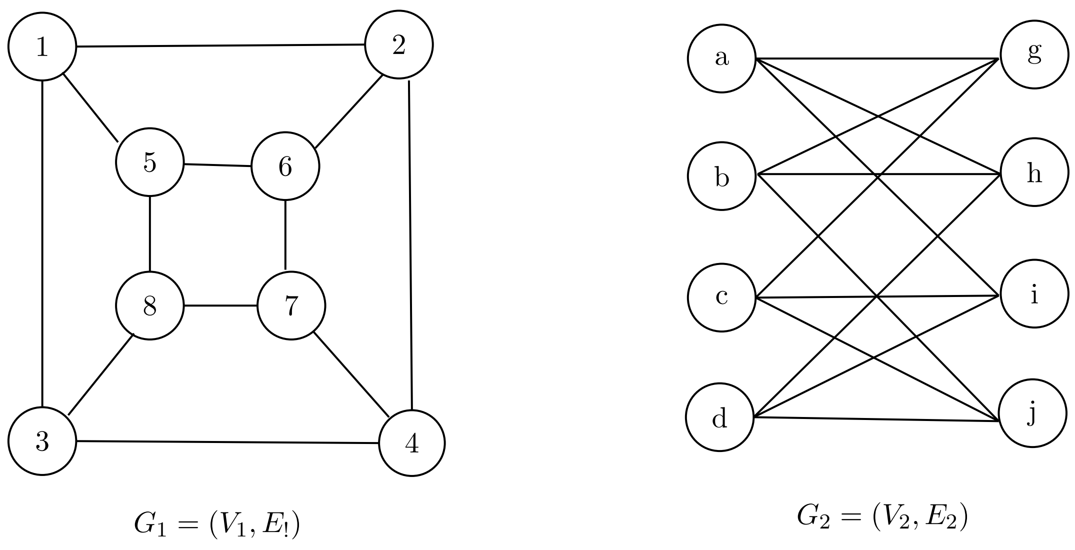


Figure 1: Isomorphic Graphs

An obvious way again would be for Alice to send the bijection

$$\pi : V_1 \to V_2 : (u, v) \in E_1 \Leftrightarrow (\pi(u), \pi(v)) \in E_2$$

However Zero Knowledge allows us to prove this without the bijection.

1.2 Applications of Zero Knowledge

Zero Knowledge has many more potential applications in solving big problems outside Computer Science domain and we state a few of them

- **Nuclear disarmament** - Global nuclear warhead has been on an incline with Russia and US possessing most of it. Efforts are being made to persuade these countries to dismantle their nuclear arms. However the details of the nuclear warhead are secrets which must not be revealed. So how does one prove that it actually did dismantle the warhead. Zero Knowledge is one way to go about it.
- **Cryptocurrency** - Zero Knowledge finds its use in verifying the validity of a transaction without learning any details of the transaction like the amount involved and identity of the parties involved.

2 Formal Definition of Zero Knowledge

Let us try to build a formal definition for Zero Knowledge in this section. Classically proofs are strings that demonstrate the validity of a statement. As an example let L be an NP-Complete language and R_L be an appropriate NP-relation for L. Then by definition of NP there exists a polynomial time algorithm A that outputs 1 upon receiving $(x, w) \in R_L$, where string x is the statement and string w is the proof.

On similar lines we define Zero Knowledge Proofs as a binary relation R and $(x, w) \in R$ where

- x is an instance of some computationally difficult problem
- w is the witness for problem instance x

We stress here that the problem has to be computationally difficult for Zero Knowledge to be of any use as otherwise parties can solve the problem themselves locally.

As an example consider the Graph Isomorphism problem, we can write the relation as

$$R_{GI} = \{\underbrace{(G_1 = (V_1, E_1), G_2 = (V_2, E_2)}_{\text{x : instance of Graph Isomorhpism}}, \underbrace{\pi : V_1 \to V_2 : (u, v) \in E_1 \Leftrightarrow (\pi(u), \pi(v)) \in E_2}_{\text{w : witness of Graph Isomorphism}}\}$$

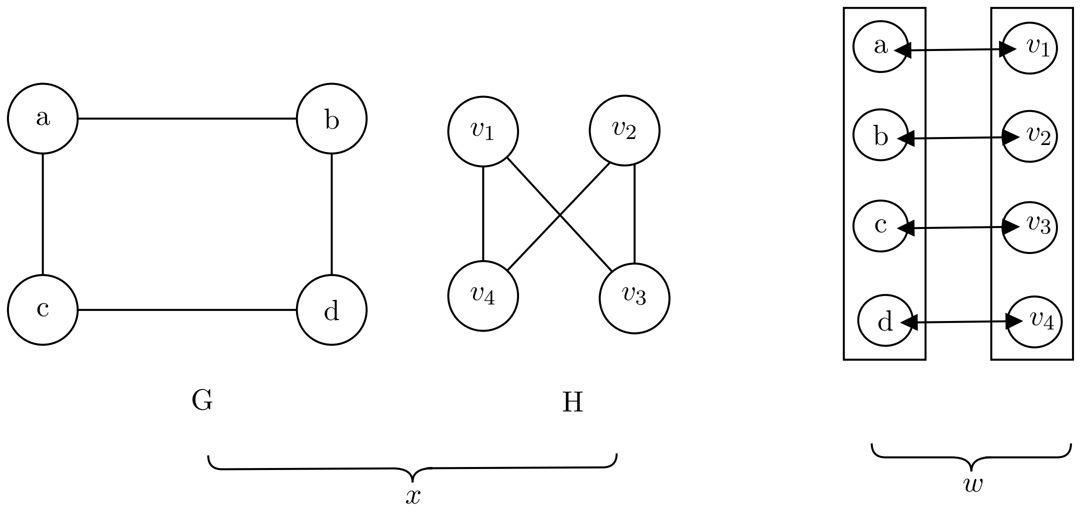


Figure 2: Instance and witness for Graph Isomorphism problem

Additionally we also require that if the graphs in the instance x are not isomorphic then no such witness w shall be possible.

2.1 Zero Knowledge Proof System

We relax our notion of proof systems (as we discussed in the preceding section) by allowing the parties to be interactive. We can consider the string we discussed earlier that was static to be now written in this interaction dynamically. As a further relaxation we also allow the proofs to fail with some negligible probability.

In our Zero Knowledge proof system parlance we will call the two parties as Prover(P) and Verifier(V) as PPT interactive protocols. Party P, the prover has the witness and wants to prove to V using interaction.

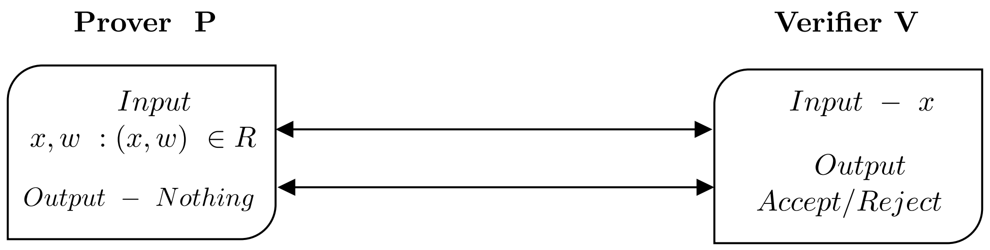


Figure 3: Interactive Zero Knowledge Proof System

We also want that the verifier(V) should accept only if x is a true statement and reject when no such w exists.

2.2 Properties required for Zero Knowledge

Even for our classical proof system we require a few properties so that the proof holds good. Here we discuss the properties for Zero Knowledge proofs after the relaxations discussed in preceding sections.

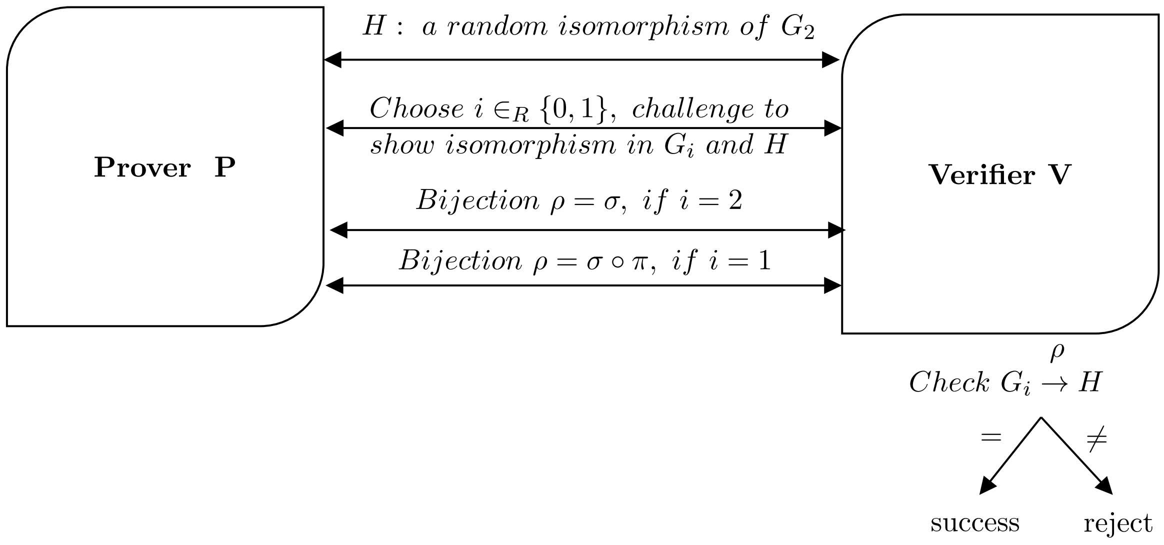
- **Completeness** - Completeness says that if both the prover(P) and Verifier (V) are honest then protocol should output accept with high probability
- **Soundness** - Soundness implies that for a corrupt prover (who does not have a witness (w), the protocol V should output Reject with high probability
- **Zero-knowledge** - For the case when prover is honest but the verifier is corrupt. Zero knowledge property implies that the probability distribution of the views generated is independent of w. The definition of this notion follows what is known as the **simulation paradigm**.

  Loosely speaking, this states that a Verifier(V) learns nothing from a proof if it can efficiently generate everything that it saw in the proof, by itself. In other words, the Verifier's view in the protocol can be simulated, given only its input. Notice that this implies that if the Verifier learned something from the proof, then it could have learned this by itself (without seeing the proof).

Here we would like to draw our attention to the fact that the first two properties hold true for a proof in classical proof system also. So the zero knowledge property is the key idea here.

3 Zero Knowledge Proof for Graph Isomorphism

In this section we discuss Zero Knowledge proof for a specific problem of Graph Isomorphism. We discussed the problem formally in previous section. So we directly start with the proof. The goal is to check if prover indeed knows an isomorphism between G_1 and G_2 without learning the isomorphism. As discussed for soundness, a cheating prover who does not know an isomorphism should get caught with high probability. The statement to be proved is isomorphism between two graphs G_1, G_2



$H:$ a random isomorphism of G_2
Choose $i \in_R \{0,1\}$, challenge to show isomorphism in G_i and H
Bijection $\rho = \sigma$, if $i = 2$
Bijection $\rho = \sigma \circ \pi$, if $i = 1$
Prover P
Verifier V
Check $G_i \xrightarrow{\rho} H$
$=$
$\neq$
success
reject

Figure 4: Zero Knowledge Protocol for Graph Isomorphism

For an honest prover, it has the required bijection for both cases ($i = 1, 2$) as ($\rho = \sigma, \rho = \sigma \circ \pi$). However an honest verifier can be cheated again by a dishonest prover with probability $\frac{1}{2}$. This happens when the verifier chooses $i = 1$ and prover just outputs the randomly chosen bijection(ρ) and since i is chosen randomly, this happens with probability $\frac{1}{2}$. This is not going to be acceptable and so we will need to boost the probability by repeating the experiment 'k' times and Verifier accepts only if all rounds are successful. We discuss this formally next in context of the properties of Zero Knowledge Proofs for Graph Isomorphism -

- Completeness - For an honest verifier and prover it is clear that each round will be successful and verifier outputs accept at the end of the protocol
- Soundness - Because of the experiment being repeated 'k' times, a cheating Prover(P) can pass all of the rounds in the extreme case when $i = 1$ in all k experiments which can happen with probability $\frac{1}{2^k}$ which is negligible.
- Zero Knowledge - The secret isomorphism π remains private in both the cases. For $i = 1$, the bijection π does not appear in the interaction so nothing about π is revealed. For $i = 2$ the composition $\pi \circ \rho$ leaks nothing about π. The leakage is actually absolutely 0 and this is what is known as **perfect zero knowledge**.

  The argument above is not a formal argument. For a more formal proof of this, one has to argue how a simulator can generate both these distributions in the real world. A curious reader is advised to know more about this from [2]

  Also notice that we have managed to build Zero Knowledge primitive without relying on any of cryptographic primitives.

4 Zero knowledge Proof for NP-Complete Class

The zero knowledge proof we presented above for Graph Isomorphism was interesting because checking if two graphs are isomorphic is considered to be a hard problem. Also the hardness of the problem in Zero Knowledge should be tameable enough to allow verification of an instance efficiently. NP-Complete class of problems are the one's which perfectly fit the bill for this description. Graph Isomorphism has not been proven to be an NP-Complete problem. So we consider this more general class of NP-complete problems and how to prove these in Zero-Knowledge. We will focus our attention on one particular NP-complete problem in this section: 3-coloring. A graph is said to be 3-colorable if each of its vertices can be a assigned one of the colors from c_1, c_2, c_3, such that no pair of adjacent vertices are assigned the same color.

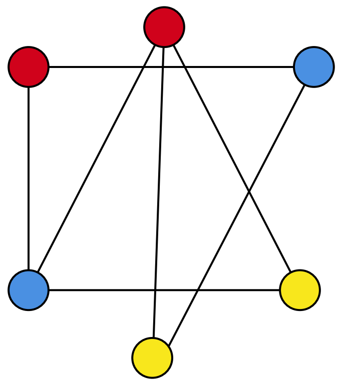


Figure 5: An instance of a 3-colorable problem

In our formal notation the graph coloring problem can be represented as

$$R_{3-col} = \underbrace{(G = (V, E))}_{\text{instance of 3-color problem}}, \underbrace{\pi : V \to c_1, c_2, c_3 : (u, v) \in E \implies (\pi(u) \neq \pi(v))))}_{\text{w is witness of 3-color problem}}$$

The goal again here is for the Prover(P) to convince the Verifier(V) that it knows the 3-color assignment π without learning anything about π. For NP-Complete problems it is known that we cannot have perfect Zero Knowledge, so we can only have **computational Zero Knowledge**. As discussed earlier since Zero Knowledge is in computational sense, we use **commitments** and our security relies on **hiding** and **binding** property of commitments. To read more about these commitment schemes, the reader is suggested to refer to the scribes for Lecture-6 at [3]. Here we use an instantiation of such commitment scheme to build our Zero Knowledge proof system for 3-coloring problem. In the figure above what we have shown is one round of interaction between Prover and Verifier. Similar to our Graph Isomorphism example the probability would need some boosting and we will repeat for k rounds.

Completeness for 'honest prover and verifier case' holds due to success of each round. In the corrupt prover case the strategy that prover can use is to guess the edge which the verifier might ask to open and color that edge with different colors and rest of the graph

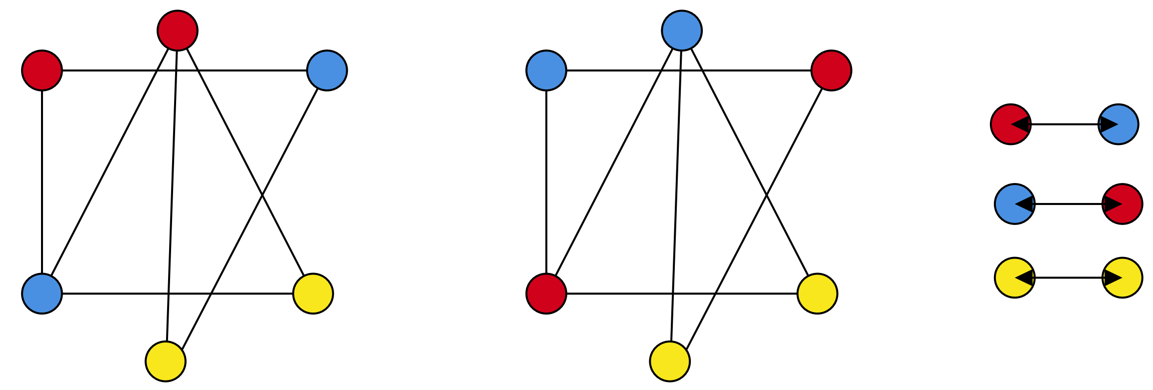


Figure 6: New 3-coloring as per random permutation π

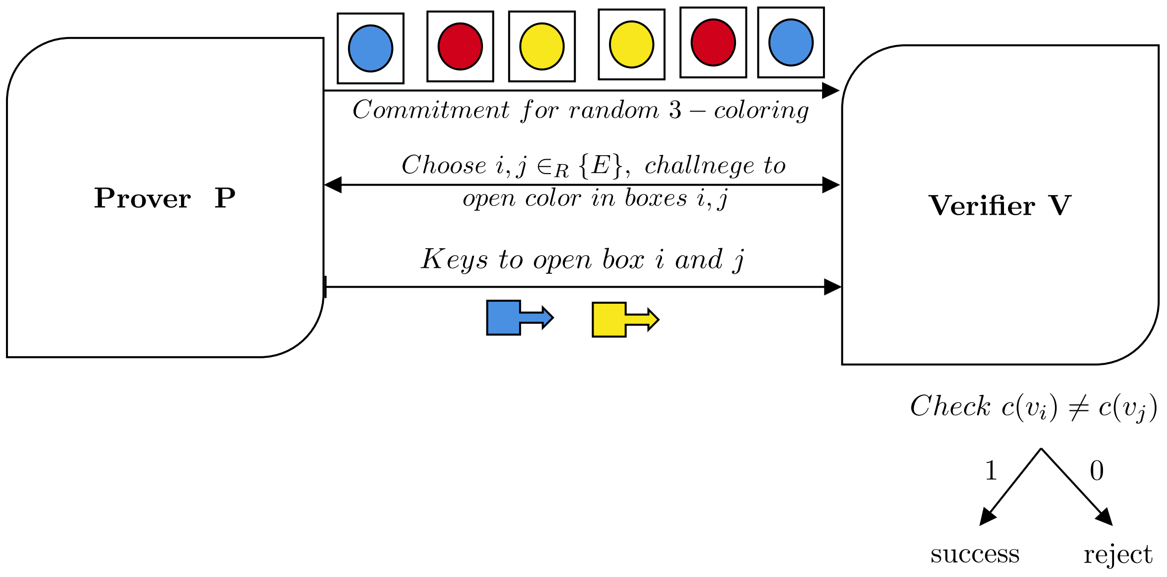

$Commitment\ for\ random\ 3-coloring$
Prover P
$Choose\ i,j \in_R \{E\},\ challnege\ to$
$open\ color\ in\ boxes\ i,j$
Verifier V
$Keys\ to\ open\ box\ i\ and\ j$
$Check\ c(v_i) \neq c(v_j)$
1
0
success
reject

Figure 7: Zero Knowledge Protocol for 3-coloring problem

arbitrarily. The strategy fails only with probability $\frac{1}{|E|}$ and hence the success probability of a round is $1 - \frac{1}{|E|}$. This can be boosted by repeating the experiment for k times until $(1 - \frac{1}{|E|})^k$ becomes negligible . Zero knowledge requires that a corrupt Verifier shall not be able to glean any information about original 3-coloring.

The proof for zero-knowledge in the case of a corrupt verifier relies on the hiding property of the underlying commitment scheme. Also as part of the protocol, color of the vertices of only one edge is revealed and the verifier only gets to know that the color of the vertices is different which it anyway knows.

Another point that we should mention here is that the 3-coloring is randomly chosen for each of the rounds otherwise a cheating Verifier can query at all the edges one by one and get to know the coloring for the whole graph.

5 Zero Knowledge Proof for Actively Secure Protocols

Zero Knowledge proofs are used extensively as a primitive for building actively secure protocols out of passively secure protocols. This is done by making parties prove each step of the computation. Parties have their private input and randomness. Each party proves that based on communication previously received and it's current state, the computation has been done as per the protocol. The witness is the local private state used by the party in each step of the computation. Each of such computations is an instance of an NP-statement which can be proved in a Zero Knowledge fashion (due to Goldreich,Micali and Wigderson) and hence we can build a generic compiler for securely computing as per the protocol.

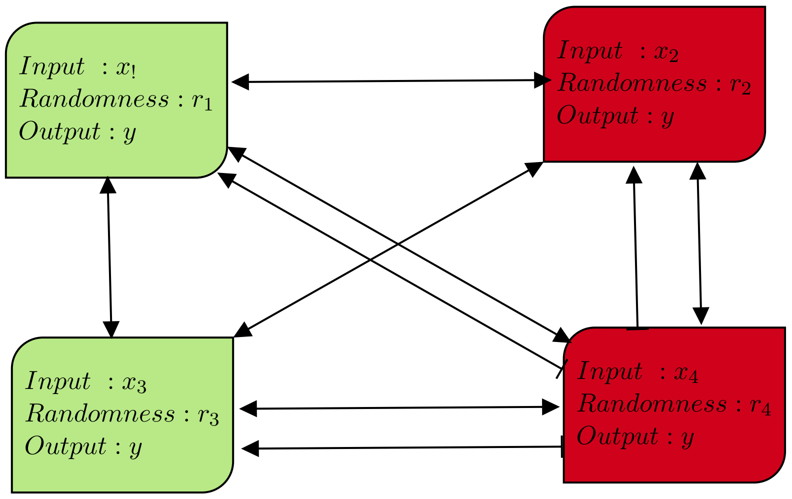


Figure 8: Actively Secure Protocols from Passively Secure Protocol

The honest parties(shown in green above) can in this way force the corrupt parties(marked in red above) out of the communication and now the active security reduces to passive security. This notion of active and private security is the same as we discussed in our lecture on Secure Multi Party Computation(MPC). So Zero Knowledge proofs are one of the ways we can handle active corruption in MPC protocols. Zero Knowledge is actually itself a special instance of another MPC protocol.

6 Conclusion

To conclude the lecture, we studied a fundamental primitive in cryptographic protocols and especially Secure Multiparty Computation(MPC). The primitive is a part of more general category of protocols called Interactive protocols. Zero Knowledge has applications in other domains like Complexity Theory also.

The current trend in Zero Knowledge proofs is in designing generic proofs for better efficiency in terms of number of rounds of communication and communication-complexity. What we discussed was it takes 3 rounds of interaction and if the experiment is repeated k times it scales to $3k$ rounds. Interestingly, there exists a non-interactive one shot proof using a technique known as **Fiat-Shamir Transform**. Optimizations are also done for proof size and time to verify for an efficient verifier.

There is also interest in designing efficient Zero Knowledge proofs for specialized tasks especially when it is used in a real life application. As an example it is used in digital signatures for cryptocurrency.

References

[1] The Knowledge Complexity of Interactive Proof-Systems” STOC-1985

[2] http://u.cs.biu.ac.il/ lindell/89-856/main-89-856.pdf

[3] https://www.csa.iisc.ernet.in/ cris/resources/e0_235/ppt/AP_Lecture6.pdf

[4] https://people.eecs.berkeley.edu/ sanjamg/classes/cs276-fall14/scribe/lec07.pdf