6 Interactive Proofs

A proof is a way of convincing a party of a certain claim. When talking about proofs, we consider two parties: a *prover* and a *verifier*. Given an assertion, the prover's goal is to convince the verifier of it's validity, whereas the verifier's objective is to accept only a correct assertion. In mathematics, for instance, the prover provides a fixed sequence of claims and the verifier checks that they are all truthful and that they imply the theorem. In real life, however, the notion of a proof has a much wider interpretation. A proof is a process rather than a fixed object, by which the validity of the assertion is established. For instance, a job interview is a process in which the candidate tries to convince the employer that she should hire him. In order to make the right decision, the employer carries out an interactive process. Unlike a fixed set of questions, in an interview the employer can adapt her questions according to the answers of the candidate, and therefore extract more information, which leads to a better decision. This example exhibits the power of a proof process rather than a fixed proof. In particular, it shows the benefits of interaction between the parties.

In many contexts, finding a proof requires creativity and originality, and therefore attracts most of the attention. However, in our discussion of proof systems, we will focus on the task of the verifier – the verification process. Typically, the verification procedure is considered to be relatively easy while finding the proof is considered a harder task. The asymmetry between the complexity of verification and finding proofs is captured by the complexity class NP.

We can view NP as a proof system, where the only restriction is on the complexity of the verification procedure (the verification procedure must take at most a polynomial time). In fact, it is not difficult to see that for each language $L \in$ NP there exists a polynomial time recognizable relation R_L such that

$$L = \{x : \exists y \ (x, y) \in R_L\}$$

and $(x, y) \in R_L$ only if $|y| \leq poly(|x|)$. (Assignment??) In a proof system for an NP language L, a proof for the claim "$x \in L$" consists of the prover sending a witness y, and the verifier checking in polynomial time whether $(x, y) \in R_L$. Such a witness exists only if the claim is true. Hence, only true assertions can be proved by this system. Note that there is no restriction on the time complexity of finding the proof (or witness). A good *proof system* must have the following properties:

1. The verifier strategy is efficient (polynomial time in the NP case).

2. Correctness requirements:

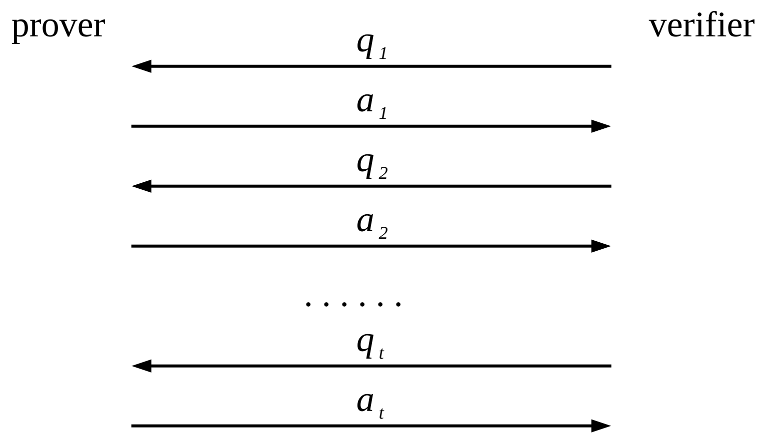
   - **Completeness:** For a true assertion, there is a convincing proof strategy (in the case of NP, if $x \in L$ then a witness y exists).

   - **Soundness:** For a false assertion, no convincing proof strategy exists (in the case of NP, if $x \in L$ then no witness y exists).

In the following we introduce the notion of *interactive proofs*. To do so, we generalize the requirements from a proof system, adding interaction and randomness.

Roughly speaking, an interactive proof is a sequence of questions and answers between the parties. The verifier asks the prover a question q_i and the prover answers with a message a_i.

At the end of the interaction, the verifier decides based on the knowledge he acquired in the process whether the claim is true or false.



6.1 The definition of IP

Following the above discussion, we define

Definition 6.1 *An* interactive proof system *for a language L is a two-party game between a verifier and a prover that interact on a common input in a way satisfying the following properties:*

1. *The verifier strategy is a probabilistic polynomial time procedure.*
2. *Correctness requirements:*
   - **Completeness:** *There exists a prover strategy P such that for every $x \in L$, when interacting on the common input x, the prover P convinces the verifier with probability at least* $2/3$.
   - **Soundness:** *For every $x \notin L$, when interacting on the common input x, any prover strategy P^* convinces the verifier with probability at most* $1/3$.

The prover strategy is computationally unbounded.

The complexity class IP consists of all the languages having an interactive proof system. We call the number of messages exchanged during the protocol between the two parties the number of *rounds* in the system. For every integer function $r(\cdot)$, the complexity class $\mathrm{IP}(r(\cdot))$ consists of all the languages that have an interactive proof system in which, on common input x, at most $r(|x|)$ rounds are used. For a set of integer functions R, we denote

$$\mathrm{IP}(R) = \bigcup_{r \in R} \mathrm{IP}(r(\cdot)) \; .$$

It is not difficult to check that BPP = IP(0) and NP $\subseteq$ IP(1). Furthermore, the number of rounds in IP can never be more than a polynomial in the length of the common input, since the verifier strategy must run in polynomial time. Therefore, if we denote by *poly* the set of all integer polynomial functions, then IP = IP(*poly*). Also, we can assume that the length of the messages exchanged between the prover and the verifier is a polynomial in the length of the common input, since in a polynomial number of steps the verifier can only read and write a polynomial number of symbols.

The following claim shows that the probabilities $2/3$ and $1/3$ in the completeness and soundness requirements can be replaced with probabilities as extreme as $1 - 2^{-p(\cdot)}$ and $2^{-p(\cdot)}$ for any polynomial $p(\cdot)$.

Claim 6.2 *Any language that has an interactive proof system, has one that achieves error probability of at most $2^{-p(\cdot)}$ for any polynomial $p(\cdot)$.*

Proof. We repeat the proof system sequentially for k times and take the majority vote. Let the random variable X denote the number of accepting votes. If the assertion holds, then X is the sum of k independent Bernoulli trials with probability of success at least $2/3$. An error in the new protocol only happens if $X \leq k/2$. Hence,

$$\begin{aligned}
\Pr[\text{error}] &\leq \Pr[X \leq k/2] \\
&\leq \sum_{i=0}^{k/2} \binom{k}{i} \left(\frac{2}{3}\right)^{i} \left(\frac{1}{3}\right)^{k-i} \\
&\leq 2 \cdot \binom{k}{k/2} \left(\frac{2}{3}\right)^{k/2} \left(\frac{1}{3}\right)^{k/2} \\
&\leq 2^k \cdot \left(\frac{2}{9}\right)^{k/2} = 2^k \cdot \left(\frac{\sqrt{2}}{3}\right)^{k} \\
&\leq \left(\frac{2.83}{3}\right)^{k} \leq 2^{k/12} .
\end{aligned}$$

Since k can be an arbitrary polynomial, the claim follows. $\square$

We note that introducing both interaction and randomness in the IP class is essential.

By adding interaction only, the interactive proof systems collapse to NP-proof systems, because for a deterministic verifier, the prover can predict the verifier's part on the interaction and send the full transcript as an NP-witness. The verifier then just has to check that the witness is a valid and accepting transcript of the original proof system.

By adding randomness only, we get a proof system in which the prover sends a witness and the verifier can run a BPP algorithm for checking its validity. We obtain the class IP(1), which seems to be simply a randomized (and perhaps stronger) version of NP.

6.2 Graph non-isomorphism

Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are called *isomorphic* (denoted $G_1 \cong G_2$) if there exists a one-to-one mapping $\pi : V_1 \to V_2$ such that $(u, v) \in E_1 \Leftrightarrow (\pi(u), \pi(v)) \in E_2$. The mapping π, if existing, is called an *isomorphism* between the graphs. If no such mapping exists, then the graphs are *non-isomorphic* (denoted $G_1 \ncong G_2$).

The *graph non-isomorphism problem* is defined as

$$GNI = \{(G_1, G_2) : \; G_1 \ncong G_2\} \; .$$

In the following we describe an interactive proof system for GNI:

- G_1 and G_2 are given as input to the verifier and the prover. Assume without loss of generality that $V_1 = V_2 = \{1, \ldots, n\}$.
- The verifier chooses $i \in \{1, 2\}$ and $\pi \in S_n$ (S_n is the group of all permutations on $\{1, \ldots, n\}$) uniformly at random. He applies the mapping π on the graph G_i to obtain a graph $H = (\{1, \ldots, n\}, E_H)$ where $E_H = \{(\pi(u), \pi(v)) : \; (u, v) \in E_i\}$ and sends the graph H to the prover.
- The prover sends $j \in \{1, 2\}$ to the verifier.
- The verifier accepts if and only if $j = i$.

Claim 6.3 *The protocol is an interactive proof system for GNI.*

Proof. We show that the above protocol is an interactive proof system with soundness probability at most $1/2$ rather than $1/3$ as in the formal definition. However, this is equivalent due to Claim 6.2.

First we note that the verifier's strategy can be easily implemented in probabilistic polynomial time. Hence, it remains to check the completeness and soundness of the proof system.

- Completeness: In case $G_1 \ncong G_2$, every graph can be isomorphic to at most one of G_1 or G_2 (otherwise, $G_1 \cong G_2$). It follows that the prover can always send the correct j and therefore Pr[verifier accepts] $= 1$.
- Soundness: In case $G_1 \cong G_2$ every graph H sent to the prover has an equal probability to come from G_1 or G_2. To prove this, let

  $$S_{G_k} = \{\sigma \in S_n : \; \sigma G_k = H\}$$

  for $k = 1, 2$ and assume $\tau \in S_n$ is an isomorphism between G_2 and G_1, i.e. $G_1 = \tau G_2$. For every $\sigma \in S_{G_1}$ it follows that $\sigma\tau \in S_{G_2}$, and for every $\sigma \in S_{G_2}$ it follows that $\sigma\tau^{-1} \in S_{G_1}$. Hence, $|S_{G_1}| = |S_{G_2}|$ and therefore, given that H was sent, the probability that the verifier chose $i = 1$ is equal to the probability of the choice $i = 2$. Since the prover does not know the random choices of the verifier, for every decision the prover makes he has success probability $1/2$, and therefore his total probability of success is $1/2$.

$\square$

The above protocol is implemented with only 2 rounds. Therefore,

Corollary 6.4 $GNI \in \mathrm{IP}(2)$.

Note that for the protocol for the GNI problem to work it is essential that the random choices of the verifier are kept *private*. If we require the random choices to be public, interactive proof systems are usually called *Arthur Merlin games*. It was shown by Goldwasser and Sipser that the power of interactive proofs and Arthur Merlin games is equivalent. Hence, it is also possible to construct a protocol for the GNI problem in which the verifier can reveal his random choices to the prover.

6.3 The complexity of IP

We have already seen that NP $\subseteq$ IP. Next we show a much stronger result.

Theorem 6.5 IP = PSPACE.

Proof. We will only show here that IP $\subseteq$ PSPACE. The proof for PSPACE $\subseteq$ IP is much more difficult and can be found in the paper by Shamir or Shen.

Let L be a language in IP. Then there is a verifier V that fulfills the conditions of an interactive proof system. That is, for any input x it holds that if $x \in L$ then there is a prover P such that V accepts x with probability at least 2/3, and if $x \notin L$ then there is no prover P such that V accepts x with probability at least 1/3. Hence, it suffices to find out what the probability of acceptance for V is, given a best possible prover strategy.

Since the verifier is polynomial time bounded, there must be a polynomial $p(\cdot)$ such that for any input of size n the number of rounds, the message sizes, and the number of computational steps of the verifier is at most $p(n)$. Without loss of generality, we assume that all random choices of the verifier are known to the prover (see the comments about Arthur Merlin games above). We also assume that the verifier plays first. Let q_i be the ith message sent by the verifier and a_i be the ith message sent by the prover. Furthermore, let r_i be the random coin tosses of the verifier between the $(i-1)$th and ith round. Since any fixed sequence $r_1, a_1, r_2, a_2, \ldots, r_i, a_i$ uniquely determines the questions $q_1, q_2, \ldots, q_i$ of the verifier, we will only consider its random choices and the answers of the prover in the following.

Given some fixed input x, $|x| = n$, let $\hat{P}_x$ denote the probability of acceptance for V, given a best possible prover strategy. Obviously,

$$\hat{P}_x = \sum_{r_1} \Pr[r_1] \cdot \hat{P}_x(r_1) ,$$

where $\hat{P}_x(r_1)$ is the maximum probability of acceptance given the random choices r_1. Furthermore,

$$\hat{P}_x(r_1) = \max_{a_1} \hat{P}_x(r_1, a_1) ,$$

where $\hat{P}_x(r_1, a_1)$ is the maximum probability of acceptance given r_1 and answer a_1. In general we get for any $i \geq 1$ that

$$\hat{P}_x(r_1, a_1, \ldots, r_i, a_i) = \sum_{r_{i+1}} \Pr[r_{i+1} \mid r_1, a_1, \ldots, r_i, a_i] \cdot \hat{P}_x(r_1, a_1, \ldots, r_i, a_i, r_{i+1})$$

and

$$\hat{P}_x(r_1, a_1, \ldots, r_i, a_i, r_{i+1}) = \max_{a_{i+1}} \hat{P}_x(r_1, a_1, \ldots, r_i, a_i, r_{i+1}, a_{i+1}) .$$

Since V decides whether to accept or reject after at most $p(n)$ rounds, there is for every possible sequence of random choices and answers an $i \leq p(n)$ such that $\hat{P}_x(r_1, a_1, \ldots, r_i, a_i) \in \{0, 1\}$.

It remains to show that $\hat{P}_x$ can be computed in polynomial space. For this we use the following algorithm:

Compute $\hat{P}_x$ in a depth-first-search manner, going through all possible choices for $r_1, a_1, \ldots, r_i, a_i$. For each round j, we store the current choice for r_j and for a_j. Furthermore, we store the current value of $\hat{P}_x(r_1, a_1, \ldots, r_j)$ (which holds the maximum $\hat{P}_x(r_1, a_1, \ldots, r_j, a_j)$ over all choices for a_j investigated so far) and of $\hat{P}_x(r_1, a_1, \ldots, r_j, a_j)$ (which holds the sum of $\Pr[r_{i+1} \mid r_1, a_1, \ldots, r_j, a_j] \cdot \hat{P}_x(r_1, a_1, \ldots, r_j, a_j, r_{j+1})$ over all r_{j+1} investigated so far). Since storing r_j, a_j, $\hat{P}_x(r_1, a_1, \ldots, r_j)$, and $\hat{P}_x(r_1, a_1, \ldots, r_j, a_j)$ each requires at most $O(p(n))$ space and the maximum number of rounds for which these items have to be stored is at most $p(n)$, the space requirement for the computation of $\hat{P}_x$ is bounded by $O(p(n)^2)$, which is a polynomial. □

6.4 Multiple-prover interactive proofs

In this section we consider an extension of the notion of an interactive proof system. Specifically, we consider the interaction of a verifier with several provers. The provers may share an a-priori selected strategy, but it is assumed that they cannot interact with each other during the time period in which they interact with the verifier.

Definition 6.6 *A* multiple-prover interactive proof system *for a language L is a game between a verifier and $k \geq 2$ provers that interact on a common input in a way satisfying the following properties:*

1. *The verifier strategy is a probabilistic polynomial time procedure.*
2. *The only interaction allowed is between the verifier and each of the provers.*
3. *A prover does not know the messages exchanged between the verifier and any other prover.*
4. *Correctness requirements:*
   - **Completeness:** *There exists a prover strategy P such that for every $x \in L$, when interacting on the common input x, the prover P convinces the verifier with probability at least* $2/3$.
   - **Soundness:** *For every $x \notin L$, when interacting on the common input x, any prover strategy P^* convinces the verifier with probability at most* $1/3$.

The prover strategies are computationally unbounded.

The complexity class MIP consists of all the languages having an multiple-prover interactive proof system. For every $k \geq 2$, the complexity class k-IP consists of all the languages that have a k-prover interactive proof system. The following result was shown by Ben-Or, Goldwasser, Kilian, and Wigderson.

Theorem 6.7 2-IP = MIP

Proof. The idea for the proof is as follows. Consider any k-prover interactive proof system between a verifier V and provers $P_1, \ldots, P_k$. In order to simulate the interaction, Verifier $\tilde{V}$ tosses all coins V ever uses and sends them to prover $\tilde{P}_1$. In return, $\tilde{P}_1$ sends $\tilde{V}$ the entire history of communication that would have occured for these coin tosses between V and the provers $P_1, \ldots, P_k$. If this is an accepting conversation for V, $\tilde{V}$ now uses $\tilde{P}_2$ to check the validity of the conversation. This is done by $\tilde{V}$ selecting at random an original prover P_i and simulating with $\tilde{P}_2$ the conversation between V and P_i on these coin tosses. If the conversation does not match the conversation sent by $\tilde{P}_1$ then $\tilde{V}$ rejects, otherwise the protocol is repeated k times (in series) and finally $\tilde{V}$ accepts. □

Let NEXP $= \bigcup_{k \geq 1} NTIME(2^{n^k})$. The next theorem was shown by Babai, Fortnow, and Lund. Since the proof is quite involved, we omit it here.

Theorem 6.8 MIP = NEXP

The theorem demonstrates that for the case of multiple provers, verifying a proof for a claim is significantly easier than proving the claim oneself.

6.5 References

- L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive proofs. *Computational Complexity* 1(1):3-40, 1991.
- M. Ben-Or, S. Goldwasser, J. Kilian, and A. Wigderson. Multi-prover interactive proofs: how to remove intractability assumptions. In *STOC 88*, pp. 113-131, 1988.
- O. Goldreich. Introduction to Complexity Theory. Lecture notes. Dept. of Computer Science and Applied Mathematics, Weizmann Institute.
- S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In *STOC 85*, pp. 291-304, 1985. (See also *SIAM Journal on Computing* 18:186-208, 1989.)
- S. Goldwasser and M. Sipser. Private coins versus public coins in interactive proof systems. In *STOC 86*, pp. 59-68, 1986.
- A. Shamir. IP = PSPACE. *Journal of the ACM* 39(4):869-877, 1992.
- A. Shen. IP = PSPACE: simplified proof. *Journal of the ACM* 39(4):878-880, 1992.