Computational complexity

lecture 12

Approximation

Last time we have started talking about approximation.

This will be finished later...

Today, we spend the whole lecture on interactive proofs.

Interactive proofs

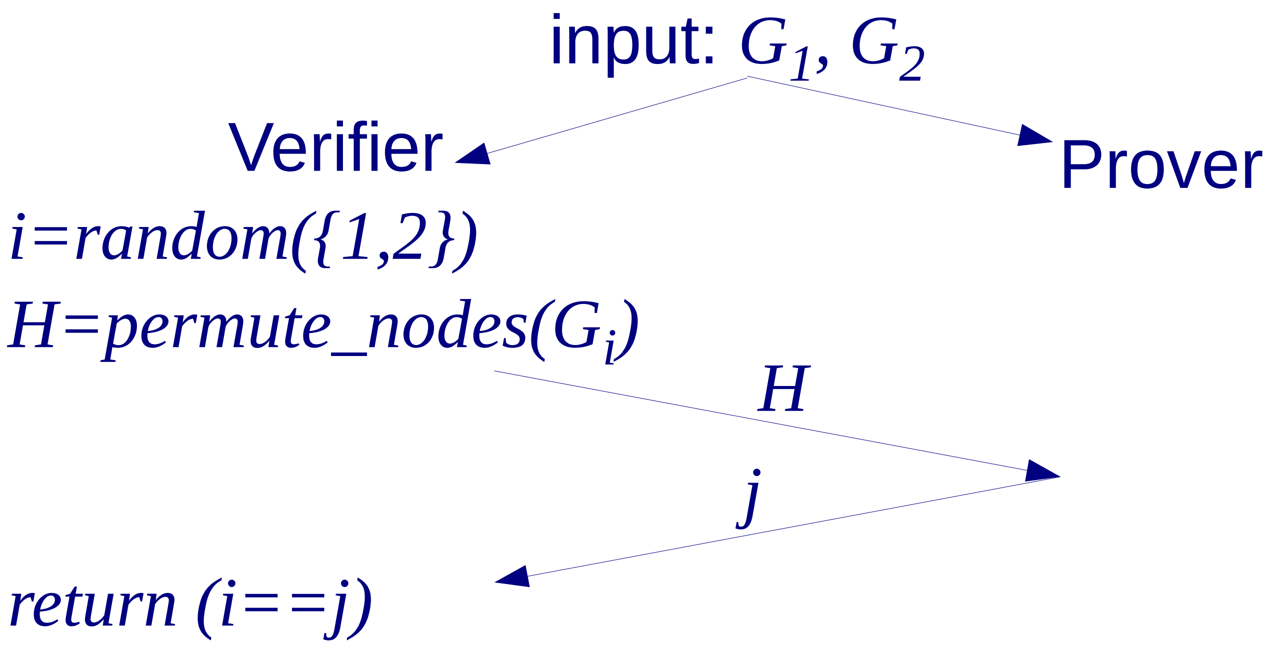
<u>Idea</u>:

- the class **NP** corresponds to a classical theorem proving – one presents a proof (a witness), and it should be possible to verify this proof in polynomial time
- the class **IP** – like on an oral exam: a verifier questions a prover, and in this way he can faster check what he knows

Interactive proofs

Example: graph nonisomorphism (are two given graph different?)
We do not know whether this problem is in **NP**, but it has an easy interactive proof:

- Two players: Verifier & Prover
- Verifier picks randomly one of the two given graphs, permutes randomly its nodes, and shows it to Prover
- Prover has to say, which graph he has received



$i = random(\{1,2\})$

$H = permute\_nodes(G_i)$

$return\ (i == j)$

Interactive proofs

Example: graph nonisomorphism (are two given graph different?)
We do not know whether this problem is in **NP**, but it has an easy interactive proof:

- Two players: Verifier & Prover
- Verifier picks randomly one of the two given graphs, permutes randomly its nodes, and shows it to Prover
- Prover has to say, which graph he has received
- if the graphs differ, he can always answer correctly
- if the graphs are isomorophic, Prover has no idea which graph was chosen by Verifier (he answers correctly with probability 1/2)
- the error probability can be decreased arbitrarily, by repeating the experiment
- Verifier works in polynomial time, using random bits
- Prover has a complicated task (we do not require polynomial time – this is similar to **NP**, where we do not require that a witness can be found in polynomial time)

Interactive proofs

<u>Formal definition</u> (deterministic Verifier):

- consider two functions $V, P: \{0,1\}^* \rightarrow \{0,1\}^*$
- V – Verifier, P – Prover
- a k-round interaction between them on an input word w:

$$
\begin{array}{ll}
q_1=V(w) & a_1=P(w,q_1) \\
q_2=V(w,q_1,a_1) & a_2=P(w,q_1,a_1,q_2) \\
\dots & \\
q_k=V(w,q_1,a_1,\dots,q_{k-1},a_{k-1}) & a_k=P(w,q_1,a_1,\dots,q_{k-1},a_{k-1},q_k)
\end{array}
$$

$$out(V,P)(w)=V(w,q_1,a_1,\dots,q_k,a_k)$$

[we assume here some encoding of tuples in words]

- A language L has a deterministic interactive protocol (i.e., $L\in$**dIP**) if there is a deterministic machine V working in polynomial time, and a polynomial number of rounds $k(n)$, such that

$$w\in L \Leftrightarrow \exists P.\ out(V,P)(w)=1$$

- We do not put any computability restrictions on the function P

Interactive proofs

It turns out that a deterministic interaction does not increase the computational power: **dIP**=**NP**

<u>Proof</u>

- **NP**⊆**dIP** – one round is enough: Prover presents a witness, Verifier checks this witness
- **dIP**⊆**NP** – the record of the conversation can serve as a witness Clearly one can check in polynomial time that the conversation record is correct (i.e. whether the algorithm of Verifier is respected) Because Verifier uses no randomness, a correct conversation record witnesses that Prover has convinced Verifier.

<u>Remark</u>

The protocol for graph nonisomorphism does not fit to this setting; this protocol is randomized

Interactive proofs

<u>Formal definition</u> (randomized Verifier):

- consider two functions $V, P: \{0,1\}^* \rightarrow \{0,1\}^*$
- V – Verifier, P – Prover
- a k-round interaction between them on an input word w, with access to a random string r of polynomial length:

$$\begin{array}{ll} q_1=V(w,r) & a_1=P(w,q_1) \\ q_2=V(w,r,q_1,a_1) & a_2=P(w,q_1,a_1,q_2) \\ \ldots & \\ q_k=V(w,r,q_1,a_1,\ldots,q_{k-1},a_{k-1}) & a_k=P(w,q_1,a_1,\ldots,q_{k-1},a_{k-1},q_k) \end{array}$$

$out(V,P)(w,r)=V(w,r,q_1,a_1,\ldots,q_k,a_k)$

- Notice that P has no access to the random string r.
- A language L has an interactive protocol (i.e., $L\in$**IP**) if there is a deterministic machine V working in polynomial time, a polynomial number of rounds $k(n)$, and a polynomial length of random strings, such that

$$w\in L \Leftrightarrow \exists P.\, Pr_r[out(V,P)(w,r)=1] \geq 3/4$$
$$w\notin L \Leftrightarrow \forall P.\, Pr_r[out(V,P)(w,r)=1] \leq 1/4$$

- We do not put any computability restrictions on the function P

Interactive proofs

$$w \in L \Leftrightarrow \exists P.\ Pr_r[out(V,P)(w,r)=1] \geq 3/4$$

$$w \notin L \Leftrightarrow \forall P.\ Pr_r[out(V,P)(w,r)=1] \leq 1/4$$

Thus, we have a protocol such that:

- for every word in L one can prove that the word is in L with high probability
- for words not in L, one can cheat and prove that such a word is in L only with a small probability

Interactive proofs

$$w \in L \Leftrightarrow \exists P.\, Pr_r[out(V,P)(w,r)=1] \geq 3/4$$

$$w \notin L \Leftrightarrow \forall P.\, Pr_r[out(V,P)(w,r)=1] \leq 1/4$$

Th. The error $1/4$ in the definition can be made arbitrarily small

Proof:

- Amplification as for **BPP** – we repeat the verification process several times, and we take majority voting
- For words in L OK – Prover repeats the same multiple times
- For words not in L this is more complicated – there can be Provers that make use of questions asked in previous experiments

Interactive proofs

$$w \in L \Leftrightarrow \exists P.\, Pr_r[out(V,P)(w,r)=1] \geq 3/4$$
$$w \notin L \Leftrightarrow \forall P.\, Pr_r[out(V,P)(w,r)=1] \leq 1/4$$

Th. The error $1/4$ in the definition can be made arbitrarily small

Proof:

- Amplification as for **BPP** – we repeat the verification process several times, and we take majority voting
- For words in L OK – Prover repeats the same multiple times
- For words not in L this is more complicated – there can be Provers that make use of questions asked in previous experiments
- But: V does not remember which questions he asked in previous experiments. If there is Prover, which in the m-th experiment has high probability of incorrect acceptance (conditional probability, under the condition of having particular questions in previous experiments), then there is also Prover, which from the beginning assumes that he has he has seen such a questions in m-1 previous experiments, and from the beginning has high probability of incorrect acceptance.

Interactive proofs

$$w \in L \Leftrightarrow \exists P.\ Pr_r[out(V,P)(w,r)=1] \geq 3/4$$

$$w \notin L \Leftrightarrow \forall P.\ Pr_r[out(V,P)(w,r)=1] \leq 1/4$$

Th. The error $1/4$ in the definition can be made arbitrarily small

Proof:

- ...
- The same can be done without increasing the number of rounds: we perform the experiments in parallel

Interactive proofs

$$w \in L \Leftrightarrow \exists P.\ Pr_r[out(V,P)(w,r)=1] \geq 3/4$$

$$w \notin L \Leftrightarrow \forall P.\ Pr_r[out(V,P)(w,r)=1] \leq 1/4$$

- In the protocol for graph nonisomorphism, if the graphs are non-isomorphic (i.e., $w \in L$), then there is Prover that always presents a correct proof (in the above definition $3/4$ can be changed to 1)
- We will prove soon that the same can be done for every language (changing $3/4$ to 1 does not change the class **IP**)

Interactive proofs

$$w \in L \Leftrightarrow \exists P.\ Pr_r[out(V,P)(w,r)=1] \geq 3/4$$
$$w \notin L \Leftrightarrow \forall P.\ Pr_r[out(V,P)(w,r)=1] \leq 1/4$$

- In the protocol for graph nonisomorphism, if the graphs are non-isomorphic (i.e., $w \in L$), then there is Prover that always presents a correct proof (in the above definition $3/4$ can be changed to 1)
- We will prove soon that the same can be done for every language (changing $3/4$ to 1 does not change the class **IP**)
- On the other hand, changing $1/4$ to 0 decreases the class **IP** to **NP**

  Proof:

  As a witness for a word w we can take a record of a single interaction, together with the random string that was used.
  Such a record exists only for words in L. One can check in polynomial time whether such a record is correct.

Interactive proofs

- In the class **IP** Prover does not see random bits used by Verifier.
- For the protocol for graph nonisomorphism this is essential: (if P knows random bits, he can always answer correctly)
- One also considers protocols in which P can see the random bits used by V (but only those already used by V, not those from the future). This is called Arthur-Merlin protocol – class **AM**[poly] (**AM** itself denotes such a single-round protocol)

Interactive proofs

- In the class **IP** Prover does not see random bits used by Verifier.
- For the protocol for graph nonisomorphism this is essential:
  (if P knows random bits, he can always answer correctly)
- One also considers protocols in which P can see the random bits used by V (but only those already used by V, not those from the future). This is called Arthur-Merlin protocol – class **AM**[poly]
  (**AM** itself denotes such a single-round protocol)
- <u>Fact</u>: **AM**[poly]$\subseteq$**IP** (and the number of rounds remains unchanged)
  <u>Proof</u>: We change the protocol, so that V sends to P his random bits. Then it does not matter whether P can see the random bits directly, or not.

Interactive proofs

- In the class **IP** Prover does not see random bits used by Verifier.
- For the protocol for graph nonisomorphism this is essential:
  (if P knows random bits, he can always answer correctly)
- One also considers protocols in which P can see the random bits used by V (but only those already used by V, not those from the future). This is called Arthur-Merlin protocol – class **AM**[poly]
  (**AM** itself denotes such a single-round protocol)
- Fact: **AM**[poly]⊆**IP** (and the number of rounds remains unchanged)
  Proof: We change the protocol, so that V sends to P his random bits. Then it does not matter whether P can see the random bits directly, or not.
- Theorem (Goldwasser-Sipser 1987)
  IP⊆**AM**[poly] (and the number of rounds increases only by 1)
  Nonobvious (and surprising) – we will prove this soon, without preserving the number of round

Interactive proofs

How large is the **IP** class?

- we know that **dIP**=**NP**
- simultaneously, we believe that **BPP**=**P**, i.e., that randomization is meaningless
- on the other hand: graph nonisomorphism is in **IP**, but we do not know whether it is in **NP**
- initially, people suspected that even if **IP** is larger than **NP**, it rather does not contain **coNP**

Interactive proofs

How large is the **IP** class?

- we know that **dIP**=**NP**
- simultaneously, we believe that **BPP**=**P**, i.e., that randomization is meaningless
- on the other hand: graph nonisomorphism is in **IP**, but we do not know whether it is in **NP**
- initially, people suspected that even if **IP** is larger than **NP**, it rather does not contain **coNP**
- a surprising theorem: **IP**=**PSPACE** (Lund, Fortnow, Karloff, Nisan, Shamir 1990)

Interactive proofs

Theorem (Lund, Fortnow, Karloff, Nisan, Shamir 1990)

IP=PSPACE

Proof

It is more-or-less clear that **IP⊆PSPACE**:

- we simulate Verifier on all possible sequences of random bits, and we compute the probability of acceptance
- we browse all possible answers that could be given by Prover, and we choose the one that maximizes the acceptance probability

Interactive proofs

Theorem (Lund, Fortnow, Karloff, Nisan, Shamir 1990)

IP=PSPACE

Proof

It is more-or-less clear that **IP**$\subseteq$**PSPACE**:

- we simulate Verifier on all possible sequences of random bits, and we compute the probability of acceptance
- we browse all possible answers that could be given by Prover, and we choose the one that maximizes the acceptance probability

It remains to prove that **PSPACE**$\subseteq$**IP**:

- We first prove an easier inclusion **coNP**$\subseteq$**IP**
- For that, it is enough to prove that 3CNF-NSAT$\in$**IP**
- We will prove an even stronger fact: the following language is in **IP**:

  $\{(\phi,K) \mid \phi$ is 3CNF and has precisely K satisfying valuations$\}$

  If this problem is in **IP**, then 3CNF-NSAT as well: it is enough to follow the protocol for $K=0$

Interactive proofs

We aim in proving that the following language is in **IP**:

$\{(\phi, K) \mid \phi$ is 3CNF and has precisely K satisfying valuations$\}$

- We treat logical formulas as polynomials:

$$\begin{aligned} x \wedge y &\leftrightarrow x \cdot y \\ \neg x &\leftrightarrow 1 - x \\ x \vee y &\leftrightarrow 1-(1-x)(1-y) \\ x \vee \neg y \vee z &\leftrightarrow 1-(1-x)y(1-z) \end{aligned}$$

- If ϕ is 3CNF and has k clauses, then the corresponding polynomial $P_\phi(x_1,\ldots,x_n)$ is of degree $d=3k$; it evaluates (in every field) to 1 for valuations satisfying ϕ, and to 0 for valuations not satisfying ϕ
- Observe that $d=O(n^3)$, as this is the upper bound for the number of different clauses in ϕ.
- We want to check whether

$$K = \sum_{v_1 \in \{0,1\}} \sum_{v_2 \in \{0,1\}} \cdots \sum_{v_n \in \{0,1\}} P_\phi(v_1,\ldots,v_n)$$

Interactive proofs

We want to check whether

$$K = \sum_{v_1 \in \{0,1\}} \sum_{v_2 \in \{0,1\}} \dots \sum_{v_n \in \{0,1\}} P_\phi(v_1,\dots,v_n)$$

- We prefer to compute everything over some finite field. We know that the sum on the right has value $\leq 2^n$, thus P and V have to fix some prime number $p > 2^n$, and then they can compute modulo p.
- How p should be selected? Preferably, it should be chosen by P (he has an unlimited power). Thus step 1 of the interaction is:
  - ➔ P presents a number $2^n < p \leq 4^n$ (it should not be too large)
  - ➔ V checks, that p is prime (using the AKS test, or any randomized test)

Interactive proofs

We want to check whether

$$K=\sum_{v_1\in\{0,1\}}\sum_{v_2\in\{0,1\}}\dots\sum_{v_n\in\{0,1\}}P_\phi(v_1,\dots,v_n)$$

- We prefer to compute everything over some finite field. We know that the sum on the right has value $\leq 2^n$, thus P and V have to fix some prime number $p>2^n$, and then they can compute modulo p.
- How p should be selected? Preferably, it should be chosen by P (he has an unlimited power). Thus step 1 of the interaction is:
  - P presents a number $2^n<p\leq 4^n$ (it should not be too large)
  - V checks, that p is prime (using the AKS test, or any randomized test)
- Thus our task is: we are given a polynomial $g(x_1,\dots,x_n)$ of degree d, a number K, and a prime p; check interactively whether

  $$K=\sum_{v_1\in\{0,1\}}\sum_{v_2\in\{0,1\}}\dots\sum_{v_n\in\{0,1\}}g(v_1,\dots,v_n)$$

  modulo p. The polynomial g is given so that V can easily compute values of g for given arguments.

Interactive proofs

We want to check whether

$$K = \sum_{v_1 \in \{0,1\}} \sum_{v_2 \in \{0,1\}} \dots \sum_{v_n \in \{0,1\}} g(v_1,\dots,v_n) \quad (\mathrm{mod}\ p) \qquad (1)$$

- V asks for a polynomial of a single variable:

  $$h(x_1) = \sum_{v_2 \in \{0,1\}} \dots \sum_{v_n \in \{0,1\}} g(x_1,v_2,\dots,v_n)$$

  This polynomial has degree $\leq d = O(n^3)$, and is considered modulo p, so it can be written succinctly.

  ➔ P sends some polynomial $s(x_1)$ (allegedly $s(x_1)=h(x_1)$)

  ➔ V check whether $s(0)+s(1)=K$ – if not, he rejects

Interactive proofs

We want to check whether

$$K = \sum_{v_1\in\{0,1\}} \sum_{v_2\in\{0,1\}} \dots \sum_{v_n\in\{0,1\}} g(v_1,\dots,v_n) \quad (\mathrm{mod}\ p) \qquad (1)$$

- V asks for a polynomial of a single variable:

  $$h(x_1) = \sum_{v_2\in\{0,1\}} \dots \sum_{v_n\in\{0,1\}} g(x_1,v_2,\dots,v_n)$$

  This polynomial has degree $\leq d=O(n^3)$, and is considered modulo p, so it can be written succinctly.
  - ➔ P sends some polynomial $s(x_1)$ (allegedly $s(x_1)=h(x_1)$)
  - ➔ V check whether $s(0)+s(1)=K$ – if not, he rejects

How P copes with the situation that (1) is false?

- If P sends the correct h as s, then we have $s(0)+s(1)\neq K$ and V rejects. Thus P has to send s differing from h.
- Since the polynomial $s(x_1)$-$h(x_1)$ has degree $\leq d$, it has $\leq d$ roots.

  Thus for a random $a\in\{0,\dots,p\text{-}1\}$ the probability that $s(a)=h(a)$ is $\leq d/p$

Interactive proofs

We want to check whether

$$K=\sum_{v_1\in\{0,1\}}\sum_{v_2\in\{0,1\}}\dots\sum_{v_n\in\{0,1\}} g(v_1,\dots,v_n) \quad (\mathrm{mod}\ p) \qquad (1)$$

- We proceed as follows: V draws $a\in\{0,\dots,p-1\}$, and then he solves the same problem as (1), but with one variable less:

$$s(a)=\sum_{v_2\in\{0,1\}}\dots\sum_{v_n\in\{0,1\}} g(a,v_2,\dots,v_n) \quad (\mathrm{mod}\ p)$$

- At the very end, when there are no variables ($n=0$), V simply checks whether $K=g()$

Interactive proofs

We want to check whether

$$K = \sum_{v_1\in\{0,1\}} \sum_{v_2\in\{0,1\}} \dots \sum_{v_n\in\{0,1\}} g(v_1,\dots,v_n) \pmod{p} \qquad (1)$$

- We proceed as follows: V draws $a\in\{0,\dots,p\text{-}1\}$, and then he solves the same problem as (1), but with one variable less:

$$s(a) = \sum_{v_2\in\{0,1\}} \dots \sum_{v_n\in\{0,1\}} g(a,v_2,\dots,v_n) \pmod{p}$$

- At the very end, when there are no variables ($n=0$), V simply checks whether $K=g()$
- If (1) holds, the „truthful" Prover always convinces V
- If (1) does not hold, every Prover gets caught on cheating with probability $\geq (1\text{-}d/p)^n \geq 1\text{-}dn/p \geq 3/4$

Bernoulli's inequality

holds for large n, since p is exponential in n, while $d=O(n^3)$)

<u>We have shown</u> that 3CNF-NSAT$\in$**IP**, i.e., that **coNP**$\subseteq$**IP**

Interactive proofs

Now the actual theorem: QBF∈**IP**, i.e., **PSPACE**⊆**IP**

This time we have a formula with quantifiers:

$$\Phi=\exists x_1 \forall x_2 \exists x_3 \ldots \forall x_n \, \phi(x_1,\ldots,x_n)$$

After translating ϕ into a polynomial, we want to check whether:

$$0 < \sum_{v_1\in\{0,1\}} \prod_{v_2\in\{0,1\}} \sum_{v_3\in\{0,1\}} \ldots \prod_{v_n\in\{0,1\}} g(v_1,\ldots,v_n)$$

Interactive proofs

Now the actual theorem: QBF∈**IP**, i.e., **PSPACE**⊆**IP**

This time we have a formula with quantifiers:

$$\Phi=\exists x_1 \forall x_2 \exists x_3 \ldots \forall x_n\ \phi(x_1,\ldots,x_n)$$

After translating ϕ into a polynomial, we want to check whether:

$$0 < \sum_{v_1\in\{0,1\}} \prod_{v_2\in\{0,1\}} \sum_{v_3\in\{0,1\}} \ldots \prod_{v_n\in\{0,1\}} g(v_1,\ldots,v_n)$$

- We would like to use the previous protocol (for a product we proceed similarly as for a sum, but instead of checking $s(0)+s(1)=K$ we have to check whether $s(0)\cdot s(1)=K$)
- A problem: resulting single-variable polynomials may have an exponential degree; they are too long to be sent in polynomial time during the interaction

Interactive proofs

Now the actual theorem: QBF$\in$**IP**, i.e., **PSPACE$\subseteq$IP**

This time we have a formula with quantifiers:

$$\Phi=\exists x_1\forall x_2\exists x_3 \ldots\forall x_n\, \phi(x_1,\ldots,x_n)$$

After translating ϕ into a polynomial, we want to check whether:

$$0 < \sum_{v_1\in\{0,1\}} \prod_{v_2\in\{0,1\}} \sum_{v_3\in\{0,1\}} \ldots \prod_{v_n\in\{0,1\}} g(v_1,\ldots,v_n)$$

- We would like to use the previous protocol (for a product we proceed similarly as for a sum, but instead of checking $s(0)+s(1)=K$ we have to check whether $s(0)\cdot s(1)=K$)
- A problem: resulting single-variable polynomials may have an exponential degree; they are too long to be sent in polynomial time during the interaction
- Solution: transform Φ to an equivalent (in the sense of the QBF problem) formula such that the resulting polynomials are of low degree.

Interactive proofs

We have a formula $\Phi=\exists x_1 \forall x_2 \exists x_3 \ldots \forall x_n\, \phi(x_1,\ldots,x_n)$

We transform Φ to an equivalent (in the sense of the QBF problem) formula such that the resulting polynomials are of low degree:

- We proceed from right to left (i.e., inside-out)
- Every subformula of the form $\forall x_i\, \theta(x_1,\ldots,x_i)$ (where θ can contain more quantifiers) is replaced with:
  $\forall x_i \exists y_1 \ldots \exists y_i\, (x_1=y_1) \wedge \ldots \wedge (x_i=y_i) \wedge \theta(y_1,\ldots,y_i)$
- We obtain an equivalent formula of n^2 variables, size = $O(n^2)$

Interactive proofs

We have a formula $\Phi=\exists x_1 \forall x_2 \exists x_3 \dots \forall x_n\, \phi(x_1,\dots,x_n)$

We transform Φ to an equivalent (in the sense of the QBF problem) formula such that the resulting polynomials are of low degree:

- We proceed from right to left (i.e., inside-out)
- Every subformula of the form $\forall x_i\, \theta(x_1,\dots,x_i)$ (where θ can contain more quantifiers) is replaced with:
  $\forall x_i \exists y_1 \dots \exists y_i\, (x_1=y_1) \wedge \dots \wedge (x_i=y_i) \wedge \theta(y_1,\dots,y_i)$
- We obtain an equivalent formula of n^2 variables, size = $O(n^2)$
- We change it into a polynomial; $x=y$ results in $x \cdot y+(1-x) \cdot (1-y)$
- What are the degrees of polynomials for subformulas:
  - ➔ the degree of ϕ is bounded by the size of ϕ
  - ➔ appending "$\exists x$" does not change the degree (we append a sum)
  - ➔ appending "$(x=y)\wedge$" increases the degree by ≤ 2
  - ➔ The polynomial for $\forall x_i \exists y_1 \dots \exists y_i\, (x_1=y_1) \wedge \dots \wedge (x_i=y_i) \wedge \theta(y_1,\dots,y_i)$ has degree $\leq 2(i-1)$ (because ≤ 2 with respect to every variable x_j)

Interactive proofs

We can apply the previous interactive algorithm for such a modified formula.

- Additional problem: previously, we were computing modulo a small prime number p, since we knew that the result is $\leq 2^n$
- Now the result can be doubly exponential, so in order to compute precisely, we would need to take p of exponential length
- Solution: P proposes a small ($\leq 2^n$) prime number p such that $(K \bmod p) \neq 0$ – it surely exists, since K cannot be divisible by all prime numbers $\leq 2^n$ (see one of the previous lectures)

Interactive proofs

We can apply the previous interactive algorithm for such a modified formula.

- Additional problem: previously, we were computing modulo a small prime number p, since we knew that the result is $\leq 2^n$
- Now the result can be doubly exponential, so in order to compute precisely, we would need to take p of exponential length
- Solution: P proposes a small ($\leq 2^n$) prime number p such that $(K \bmod p) \neq 0$ – it surely exists, since K cannot be divisible by all prime numbers $\leq 2^n$ (see one of the previous lectures)

<u>Remark 1</u>

In these protocols, random bits can be visible to P.
We have thus also shown that **AM**[poly]=**IP**.

Interactive proofs

We can apply the previous interactive algorithm for such a modified formula.

- Additional problem: previously, we were computing modulo a small prime number p, since we knew that the result is $\leq 2^n$
- Now the result can be doubly exponential, so in order to compute precisely, we would need to take p of exponential length
- Solution: P proposes a small ($\leq 2^n$) prime number p such that $(K \mathit{mod}\ p) \neq 0$ – it surely exists, since K cannot be divisible by all prime numbers $\leq 2^n$ (see one of the previous lectures)

Remark 1

In these protocols, random bits can be visible to P.
We have thus also shown that **AM**[poly]=**IP**.

Remark 2

For words in L there exists a Prover, which always proves this. Thus if in the first point of the definition of **IP** we have probability 1 instead of 3/4, we do not change the class **IP**.

Interactive proofs vs alternating computations

We have **IP**=**PSPACE**, but also **AP**=**PSPACE**

alternating polynomial time

Differences between **AP** and **IP**:

- player ∃ ↔ Prover, player ∀ ↔ Verifier
- in **AP** player ∀ can be nondeterministic, in **IP** Verifier is deterministic (with access to random bits)
- in **AP** player ∃ works in (nondeterministic) polynomial time, in **IP** Prover has unlimited computational power