PSPACE, AM and IP in non-uniform computational models

immediate

October 13, 2020

1 Bilbo – Gollum games and computations models

There is Gollum, Bilbo and a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$. Gollum has $x \in \{0,1\}^n$. Bilbo wants to find $f(x)$. In order to do that, Bilbo can ask Gollum some YES/NO question about x. Bilbo can act adaptively (his questions may depend on the Gollum's previous answers). We assume that Gollum answers honestly.

Simultaneously, Bilbo wants to find $f(x)$ as fast as possible. I.e., he wants to minimize q such that there is a way of finding $f(x)$ by asking at most q question (whatever $x \in \{0,1\}^n$ Gollum has). The minimal q for which it is possible will be called *complexity* of f.

Of course, this is not interesting if Bilbo is allowed to ask *any* YES/NO questions (he can just ask $f(x)$ in one question). By restricting the set of questions Bilbo can ask in one time we obtain different *computational models*. For example, in the *decision tree complexity* Bilbo can only ask the value of some coordinate of x. Next, there are a lot of generalizations of the decision tree complexity, and these generalizations can be also defined in this language. For example, one can obtain *parity decision tree complexity* by allowing Bilbo to ask the value of any *linear* function (over $\mathbb{F}_2$) of x.

Moreover, we can also define the *communication complexity* in this language. Recall that the deterministic communication complexity of f is defined as follows. We split x into two halves, $a = x_1 x_2 \ldots x_{n/2}$ and $b = x_{n/2} x_{n/2+1} \ldots x_n$. There is Alice who gets a and Bob who gets b. They want to find $f(x) = f(ab)$. The deterministic communication complexity of f is just the minimal number of bits Alice and Bob have to send each other to do that (in a worst case over a, b).

Here is how to define this quantity in terms of a certain Bilbo-Gollum game. Namely, allow Bilbo to ask the following two kinds of questions:

- any YES/NO question about the first half of x, i.e., about a;
- any YES/NO question about the second half of x, i.e., about b.

It is not hard to see that the minimal number of questions Bilbo needs here to compute $f(x)$ just equals the deterministic communication complexity of f.

These examples motivate a general notion of a computational model. Namely, by a computational model we mean an infinite sequence $\mathcal{M} = \{Q_n\}_{n\in\mathbb{N}}$, where $Q_n \subseteq \mathbb{P}_2(n)$. Here $\mathbb{P}_2(n)$ denotes the set of all Boolean functions with n input bits. By Q_n we mean the set of questions Bilbo can ask about $x \in \{0,1\}^n$ in the computational model $\mathcal{M}$.

In this formalism we obtain

- the decision tree complexity by setting Q_n to be the set of all dictator functions;
- the parity decision tree complexity by setting $Q_n = \{\bigoplus_{i\in S} x_i \mid S \subseteq \{1,2,\ldots,n\}$;
- the communication complexity by setting Q_n to be the set of functions that either depend only on the first half of x or only on the second half of x.

The deterministic complexity of $f\colon \{0,1\}^n \to \{0,1\}$ in the computational model $\mathcal{M}$ (denoted by $D_{\mathcal{M}}(f)$) is the minimal $q \in \mathbb{N}$ such that there is a Bilbo's algorithm that finds $f(x)$ in at most q questions from Q_n, whatever $x \in \{0,1\}^n$ Gollum has.

Let us mention that there is a standard way of representing Bilbo's algorithm in a computational model $\mathcal{M}$ as binary trees. Nodes of a tree will represent possible positions of the game. Namely, we always start in the root of the tree. If in the current node Gollum answers YES, we go to the left child, if NO – to the right child. For every non-leaf v there is a question $q_v \in Q_n$, associated with this node. Bilbo will ask q_v if the current node is v. In the leaves of the tree are outputs of Bilbo.

We will call such trees (deterministic) $\mathcal{M}$-trees (or simply trees if $\mathcal{M}$ is clear from the context). For an $\mathcal{M}$-tree T we will denote its depth by $\text{depth}(T)$. Note that $\text{depth}(T)$ is just the complexity of the corresponding Bilbo's algorithm, i.e., the maximal number of questions Bilbo asks in T. We will also view an $\mathcal{M}$-tree T as a Boolean function $T\colon \{0,1\}^n \to \{0,1\}$. Namely, $T(x)$ will be the output of the Bilbo's algorithm, depicted by T, on x.

2 "Complexity classes"

Disclaimer: we will not define any complexity classes but rather we will talk about different "complexities" of f, e.g. non-deterministic complexity, interactive proof complexity and so on.

Let $\mathcal{M}$ be a computational model.

NP and the polynomial time hierarchy. By a non-deterministic $\mathcal{M}$-tree N we mean a set $\{T_1, T_2, \ldots, T_m\}$ of deterministic $\mathcal{M}$-trees. By the depth of N we mean

$$\text{depth}(N) = \lceil \log_2(m) \rceil + \max_{i=1,\ldots,m} \text{depth}(T_i).$$

We say that N computes a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ if the following holds

- for any x with $f(x) = 1$ there exists i such that $T_i(x) = 1$;
- for any x with $f(x) = 0$ there is no i such that $T_i(x) = 0$.

In other words, a non-deterministic $\mathcal{M}$-tree N computes a function

$$N(x) = \bigvee_{i=1}^{m} T_i(x).$$

The last formula also motives the definition of $\mathrm{depth}(N)$. Represent an $\bigvee$-gate of fan-in m as a $\lceil \log_2 m \rceil$-depth tree of the binary $\vee$-gates. Attach $T_1, T_2, \ldots, T_m$ to the leaves of this tree. The result will have depth $\mathrm{depth}(N)$.

Definition 1. *The non-deterministic communication complexity of f in a computational model $\mathcal{M}$ is the minimal d for which there exists a non-deterministic $\mathcal{M}$-tree N such that* $\mathrm{depth}(N) \leqslant d$ *and $N(x) = f(x)$ for all $x \in \{0,1\}^n$. This quantity will be denoted by $NP_{\mathcal{M}}(f)$.*

Similarly we can define co-non-deterministic trees, Σ_2-trees and so on (and the corresponding complexities of f). For example, Π_2-$\mathcal{M}$-trees can be represented as formulas of the form:

$$\Pi = \bigwedge_{i=1}^{a} \bigvee_{i=1}^{b} T_{i,j}.$$

Here $T_{i,j}$ are deterministic $\mathcal{M}$-trees. Again, to define $\mathrm{depth}(\Pi)$ we represent this formula in a natural way as a fan-in 2 AND-OR tree, attach $T_{i,j}$ to the leaves, and take the depth of the result.

Alternating trees. By an alternating $\mathcal{M}$-tree A we just mean an arbitrary fan-in 2 AND-OR tree with some deterministic $\mathcal{M}$-trees attached to its leaves. We define $\mathrm{depth}(A)$ as before – we draw the underlying AND-OR tree, then draw the deterministic protocols attached to it, and then take the depth of a tree on the resulting picture.

The function A computes is defined as follows. We take $x \in \{0,1\}^n$, plug-in x into deterministic trees attached to the leaves of A, compute the outputs of these trees on x, put the results into the leaves, and then evaluate the underling AND-OR tree.

Definition 2. *The alternating complexity of f in a computation model $\mathcal{M}$ is a minimal d such that there exists an alternating $\mathcal{M}$-tree of depth at most d that computes f. This quantity is denoted by $A_{\mathcal{M}}(f)$.*

Non-deterministic trees, Π_2-trees and so on can be seen as special cases of alternating trees. For example, a non-deterministic tree is an alternating tree where we just have a single layer of $\vee$'s. A Π_2-tree is an alternating tree where we first have a layer of $\wedge$'s, and the a layer of $\vee$'s. Of course, in general alternating trees the number of such layers is not bounded.

Remark. *WLOG we may assume that the deterministic trees in the leaves of an alternating tree are trivial. By "trivial " we mean that Bilbo asks only one question in them. This is because we can represent a deterministic tree is an OR over its leaves with output* 1*, and then each such leaf as an AND over questions Bilbo asks on the path to the leaf. This construction can only increase the depth of our initial alternating tree by a constant factor.*

E.g., this means that the alternating trees in case of the decision tree complexity are essentially DeMorgan formulas.

Interactive proof trees.

An IP $\mathcal{M}$-tree I is a binary rooted tree, where

- all the internal nodes of I are partitioned into two sets V and P;
- for every leaf l of I there is a deterministic $\mathcal{M}$-tree T_l, attached to l.

We define depth(I) exactly as before.

A Prover's strategy is a mapping S that to every node $a \in P$ assigns a child of a in the tree I.

For a Prover's strategy S and for $x \in \{0,1\}^n$ we define a random variable $I_S(x)$, taking values in the set $\{0,1\}$. The random variable $I_S(x)$ is defined in two steps:

- first we define a random variable L_S taking values in the set of leaves of I;
- then we set $I_S(x) = T_{L_S}(x)$.

Distribution of L_Z depends only on S but not on x. We may define L_Z as a result of a "semi-random" walk descending from the root of I to one of its leaves. More precisely, place a pebble in the root of I. If currently the pebble is in a node $a \in P$, then descend the pebble into $S(a)$. If currently the pebble is in the node $b \in V$, choose a child of a uniformly at random and descend the pebble there. Each time we are in the new node from V we use a coin toss independent from all the previous ones. After a finite number of descents the pebble will reach a leaf of I. Define the value of L_S to be this leaf.

Remark. *I.e., we assume that Verifier only sends random bits to Prover when communicating with him.*

We say that an IP $\mathcal{M}$-tree I computes $f\colon \{0,1\}^n \to \{0,1\}$ if the following holds:

- for every x with $f(x) = 1$ there exists a Prover's strategy S such that $\mathbf{Pr}[I_S(x) = 1] \geqslant 2/3$.
- for every x with $f(x) = 0$ and for every Prover' strategy S with have $\mathbf{Pr}[I_S(x) = 1] \leqslant 1/3$.

Definition 3. *The interactive proof complexity of f in a computation model $\mathcal{M}$ is a minimal d such that there exists an IP $\mathcal{M}$-tree of depth at most d that computes f. This quantity is denoted by $IP_{\mathcal{M}}(f)$.*

3 “PSPACE”?

We now define “space complexity” of f in the computational model $\mathcal{M} = \{Q_n\}_{n\in\mathbb{N}}$. For that we turn back to the Bilbo-Gollum games. Recall that Gollum gets $x \in \{0,1\}^n$, and Bilbo’s goal is to find $f(x)$. Before Bilbo wanted to minimize the number of questions he asks to Gollum. Now he will want to minimize the amount of *memory* he uses.

To formalize it we assume that Bilbo’s actions are governed by a finite automaton $\mathcal{A}$. The automaton $\mathcal{A}$ has a finite set of states S with 3 designated stated s_{init}, s_0 and s_1. The state s_{init} will be the initial state, the states s_0 and s_1 will be two terminating states (the subscriptindicates the output of Bilbo). For each $s \in S \setminus \{s_0, s_1\}$ there is a question $q_s \in Q_n$ that Bilbo asks to Gollum when in the state s. Finally, there is a transition function $\delta \colon S \setminus \{s_0, s_1\} \times \{0,1\} \to S$ that takes a current state of Bilbo, a Gollums answers, and outputs the next state of Bilbo upon receiving this answer.

We say that such an automaton computes f if whenever Bilbo plays with Gollum according to this automaton, the automaton comes into the state $s_{f(x)}$ after a finite number of question. Note that in principle we can come into a loop without ever reaching a terminal state.

Definition 4. *The state complexity of f in a computational model $\mathcal{M}$ is the minimal s such that there exists an s-state Bilbo’s automaton that computes f. This quantity is denoted by* $\mathrm{state}_{\mathcal{M}}(f)$*.*

We also denote $\mathrm{space}_{\mathcal{M}}(f) = \log_2(\mathrm{state}_{\mathcal{M}}(f))$.

Equivalently, we can represent such automata as *transition* graphs. In these graphs each node will have out-degree 2, except for the two terminal nodes (one labeled by 0 and the other labeled by 1). There will also be one initial node. With each non-terminal node there is a question from Q_n associated with this node. This will be a question Bilbo asks Gollum in this node. For every non-terminal node there is an ordering on its two out-going edges. If the answer is YES, we transit by the first out-going edge, if NO – by the second.

Such a transition graph computes f if we come into a terminal, labeled by $f(x)$, whenever we play with Gollum having $x \in \{0,1\}^n$. The quantity $\mathrm{state}_{\mathcal{M}}(f)$ corresponds to the minimal size (the number of nodes) of such a graph.

In a special case when these transition graphs are trees we obtain just the standard deterministic complexity of f. In fact, it is not hard to see that

Proposition 1. *For any computational model $\mathcal{M}$ we have*

$$\mathrm{space}_{\mathcal{M}}(f) \leqslant D_{\mathcal{M}}(f) \leqslant 2^{\mathrm{space}_{\mathcal{M}}(f)} = \mathrm{state}_{\mathcal{M}}(f).$$

4 Results

From the standard proofs of AP⊆PSPACE and IP⊆PSPACE one can get

Proposition 2. *For every computational model $\mathcal{M}$ we have*

$$A_{\mathcal{M}}(f) = (\mathrm{space}_{\mathcal{M}}(f))^{O(1)}, \qquad IP_{\mathcal{M}}(f) = (\mathrm{space}_{\mathcal{M}}(f))^{O(1)}.$$

In fact, for every computational model $\mathcal{M}$ the analog of AP=PSPACE holds.

Proposition 3. *For every computational model $\mathcal{M}$ we have that $A_{\mathcal{M}}(f)$ and $\mathrm{space}_{\mathcal{M}}(f)$ are polynomially equivalent.*

5 "TQBF"

It is instructive to define "TQBF" for $\mathrm{space}_{\mathcal{M}}(f)$ not in terms of just Boolean formulas with quantifiers, but in terms of first-order formulas with the predicates.

By an X-signature we mean a set $\mathcal{P}$ of 0, 1 and 2-variate predicate symbols, partitioned into two subsets $\mathcal{P}_{usual}$ and $\mathcal{P}_{special}$.

An X-interpretation $\mathcal{I}$ of an X-signature $\mathcal{P} = \mathcal{P}_{usual} \sqcup \mathcal{P}_{special}$ consists of

- a set U (the support of $\mathcal{I}$)
- a mapping that assigns $P \in \mathcal{P}_{usual}$ assigns a predicate $P^{\mathcal{I}}$ of the same arity over U;
- a mapping that assigns to every pair (P, x), where $P \in \mathcal{P}_{special}$ and $x \in \{01\}^n$, a predicate $P^{\mathcal{I}}_x$ of the same arity as P over U.

For any $x \in \mathcal{X}$ an X-interpretation $\mathcal{I}$ produces an interpretation $\mathcal{I}_x$ (in the ordinary model-theoretic sense) of the signature $\mathcal{P}$. Namely, we just assign a predicate $P^{\mathcal{I}}_x$ to a predicate symbol P for every $P \in \mathcal{P}_{special}$.

For any first order formula $\phi(z_1, \ldots, z_k)$ over signature $\mathcal{P}$ with free variables $z_1, \ldots, z_k$ and for every $a_1, \ldots, a_k \in U$ one can define $\phi^{\mathcal{I}}_x(a_1, \ldots, a_k)$ as the value of $\phi(a_1, \ldots, a_k)$ in the interpretation $\mathcal{I}_x$.

Definition 5. *Let $\mathcal{M} = \{Q_n\}_{n\in\mathbb{N}}$ be a computational model. We say that an X-interpretation $\mathcal{I}$ is $\mathcal{M}$-correct if for any k, for any k-variate predicate symbol $P \in \mathcal{P}_{special}$ and for any $a_1, a_2, \ldots, a_k \in U$ it holds that the function:*

$$g(x) = P^{\mathcal{I}}_x(a_, a_2, \ldots, a_k)$$

belongs to Q_n (i.e., Bilbo can ask a question $g(x)$ in the computational model $\mathcal{M}$).

Theorem 4. *There exists a finite X-signature $\mathcal{P}$ with one special symbol such that the following holds. Let $\mathcal{M}$ be a computational model and let $f\colon \{0,1\}^n \to \{0,1\}$ be a Boolean function. Then there exist*

- *an $\mathcal{M}$-correct X-interpretation $\mathcal{I}$ of $\mathcal{P}$ with the support of size $\mathrm{state}_{\mathcal{M}}(f)$;*

- *a closed PNF formula φ over $\mathcal{P}$ of length* $(\mathrm{space}_{\mathcal{M}}(f))^{O(1)}$ *with only two occurrences of the special predicate symbol;*

such that $f(x) = \varphi_x^{\mathcal{I}}$ *for every* $x \in \{0,1\}^n$.

Proof. We take a transition graph $\mathcal{A}$ of size $m = \mathrm{state}_{\mathcal{M}}(f)$, computing f. The support of our X-interpretation $\mathcal{I}$ will be the set S of nodes (states) of $\mathcal{A}$.

For every l we define a formula $\psi_l(\cdot,\cdot)$ with two free variables such that the following holds. For every $x \in \{0,1\}^n$ and for every $s,t \in S$ the transition graph $\mathcal{A}$ transits from s to t on input x in at most l steps if and only if $(\psi_l)_x^{\mathcal{I}} = 1$.

Assuming we have constructed such ψ_l, we can just set

$$\varphi = \psi_m(s_{init}, s_1),$$

where s_{init}, s_1 are constants that are interpreted in $\mathcal{I}$ as, respectively, the initial and the 1-terminal node.

It is enough to show that ψ_l can be realized by a PNF formula of length $O(\log l)$ with just two occurrences of the special predicate symbol.

Let us now accurately realize ψ_1:

$$\psi_1(s,t) = (s = t) \vee (\mathrm{FirstSuccessor}(s,t) \wedge \mathrm{Gollum}(s)) \\ \vee (\mathrm{SecondSuccessor}(s,t) \wedge \neg\mathrm{Gollum}(s))$$

First we write $s = t$ to capture a possibility that s and t are equal so that there is a 0-length path from s to t. Now, recall that for non-terminal node there is an ordering on its out-going edges. The predicate symbol $\mathrm{FirstSuccessor}(\cdot,\cdot)$ will be a usual predicate symbol. In $\mathcal{I}$ we will have $\mathrm{FirstSuccessor}(s,t) = 1$ if and only if s is a non-terminal node and t is the node where the first out-going edges of s leads. We define SecondSuccessor similarly. Now, Gollum will be the only special predicate symbol. We let $\mathrm{Gollum}_x^{\mathcal{I}}(s)$ to be the Gollum's answer to the question Bilbo asks in the node s, assuming Gollum has x on input. Clearly, this ensures that $\mathcal{I}$ is $\mathcal{M}$-correct. Moreover, it is not hard to see that ψ_1, defined as above, indeed realizes the reachability by paths of length at most 1.

Next, we can define ψ_l recursively. Of course, we can write:

$$\psi_l(s,t) = \exists u\ \psi_{\lceil l/2\rceil}(s,u) \wedge \psi_{\lceil l/2\rceil}(u,t).$$

But then the length of ψ_l will be linear in l. There is a standard trick of exponentially improving it, by using just one recursive call to $\psi_{\lceil l/2\rceil}$:

$$\psi_l(s,t) = \exists u \forall a \forall b\ \ ((a = s \wedge b = u) \vee (a = u \wedge b = t)) \leftarrow \psi_{\lceil l/2\rceil}(a,b).$$

This also preserves the number of occurrences of the Gollum symbol. □