on IP=PSPACE in communication complexity

immediate

October 15, 2020

1 Bilbo – Gollum games and computations models

There are 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 questions 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 d such that there is a way of finding $f(x)$ by asking at most d question (whatever $x \in \{0,1\}^n$ Gollum has). The minimal d 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, $a = x_1x_2\dots x_{n/2}$ and $b = x_{n/2+1}x_{n/2+2}\dots x_n$. There are Alice who gets a and Bob who gets b. They want to find $f(x) = f(ab)$. The deterministic communication complexity of f is 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[1]. 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 $d \in \mathbb{N}$ such that there is a Bilbo's algorithm that finds $f(x)$ in at most d questions from Q_n, whatever $x \in \{0,1\}^n$ Gollum has.

We will use the notation $\mathcal{M} = dt$ for the decision tree complexity and the notation $\mathcal{M} = cc$ for the communication complexity.

There is a standard way of representing Bilbo's algorithms in some computational model $\mathcal{M}$ as binary trees. Nodes of a tree will represent possible positions of the game (and also states of an algorithm). 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 $\mathrm{depth}(T)$. Note that $\mathrm{depth}(T)$ 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 corresponding Bilbo's algorithm 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

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

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

[1] That's, of course, a much more general term than what we consider here. Maybe better to say "tree-like models" or "top-to-bottom models".

- 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 binary AND-OR tree, attach $T_{i,j}$ to the leaves, and take the depth of the result.

Alternating trees (*that is what Babai et al.'86 call PSPACE for cc).* By an alternating $\mathcal{M}$-tree A we just mean a binary 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 computed by an alternating tree A 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 then 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 *(appears in Aaronson–Wigderson for cc).* Let us define a notion of $\max \frac{1}{2}$*-trees*. This is a rooted binary tree, where the non-leaf nodes are partitioned into the max-nodes and the *random* nodes. Now, take an $\max \frac{1}{2}$-tree R and consider an assignment of the leaves of R to Boolean values. Let us define *the output* of R on such an assignment (this will be a rational number from $[0,1]$). We define it recursively. For a leaf, its output is just a Boolean value assigned to it. For a max-node, its output is just the maximum of the outputs of its children. For a random node, its output is the arithmetic mean of the outputs of its children.

Now, an *interactive proof $\mathcal{M}$-tree* (IP $\mathcal{M}$-tree for short) I is a $\max \frac{1}{2}$-tree, equipped with an assignment of its leaves to deterministic $\mathcal{M}$-trees. We let $\mathrm{depth}(I)$ be the depth of a tree obtained by attaching these deterministic $\mathcal{M}$-trees to the underlying $\max \frac{1}{2}$-tree.

For $x \in \{0,1\}^n$, we define the output of I on x as follows. First, we obtain an assignment of the leaves of the underlying $\max \frac{1}{2}$-tree to Boolean values. We do so by evaluating the bottom deterministic $\mathcal{M}$-trees on x. Then we just let $I(x)$ to be the output of our $\max \frac{1}{2}$-tree on this assignment.

We say that an IP $\mathcal{M}$-tree I computes $f\colon \{0,1\}^n \to \{0,1\}$ if for any $x \in \{0,1\}^n$ we have $|f(x) - I(x)| \leqslant 1/3$.

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

Remark. *Here Prover and Verifies are communicating by descending in the tree. Prover communicates in the random nodes, where he descends to a random child (so he just sends random bits to Verifier). Verifier communicates in the* max*-nodes, where he is allowed to choose any child to descend. Finally, they reach some leaf. The verdict is made by evaluating a deterministic tree, attached to this leaf, on x. If the verdict is* 1*, Prover is convinced that $f(x) = 1$. If the verdict is* 0*, he is not convinced.*

Clearly, $I(x)$ equals the maximum over all Verifier's strategies of the probability that Verifier convinces Prover that $f(x) = 1$.

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 subscript indicates 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 $\delta\colon S \setminus \{s_0, s_1\} \times \{YES, NO\} \to S$ that takes a current state of Bilbo and a Gollum's answer, and outputs the 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 $STATE^{\mathcal{M}}(f)$.*

We also denote $SPACE^{\mathcal{M}}(f) = \log_2\left(STATE^{\mathcal{M}}(f)\right)$ (that many bits we need to store a state of our automaton).

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 v there is an ordering on its two out-going edges. If the answer is YES, we transit by the first out-going edge of our current node, 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 $STATE^{\mathcal{M}}(f)$ 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*

$$SPACE^{\mathcal{M}}(f) \leqslant D^{\mathcal{M}}(f) \leqslant 2^{SPACE^{\mathcal{M}}(f)} = STATE^{\mathcal{M}}(f).$$

An exponential gap is possible, as the following simple example shows.

Proposition 2. *We have that $SPACE^{dt}(x_1 \wedge x_2 \ldots \wedge x_n) \leqslant \log_2(n) + O(1)$, while it is well-known that $D^{dt}(x_1 \wedge x_2 \ldots \wedge x_n) = n$.*

Proof. Bilbo just asks the bits of $x = x_1 x_2 \ldots x_n$ one by one, waiting for a zero. For that he only need $\log_2(n) + O(1)$ bits of memory to remember the index of the current bit. □

The same separation can be obtained in communication complexity for, say, the disjointness predicate.

4 Results

Proposition 3. *For every computational model $\mathcal{M}$ we have*

$$SPACE^{\mathcal{M}}(f) = \left(A^{\mathcal{M}}(f)\right)^{O(1)}, \qquad SPACE^{\mathcal{M}}(f) = \left(IP^{\mathcal{M}}(f)\right)^{O(1)}.$$

This follows from the standard proofs of AP⊆PSPACE and IP⊆PSPACE. For instance, take an interactive proof tree of depth d. Bilbo will just recursively compute the output of all the nodes of the underlying max $\frac{1}{2}$-tree on x. For the leaves he will just need to run a deterministic $\mathcal{M}$-tree, for that he need at most as many bits of memory as the depth of this tree. Now, assume that it is possible to compute the output of the nodes of depth k in s_k space. Then for the nodes of depth $k-1$ Bilbo can compute the output in $O(d)+s_k$ space. Indeed, to compute the output in the first and in the second child he can just use the same s_k bits of memory. He will only need additional $O(d)$ bits of memory to remember the answers (it is not hard to see that the outputs will be dyadic rationals with at most d bits).

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

Proposition 4. *For every computational model $\mathcal{M}$ we have that $A^{\mathcal{M}}(f)$ and $SPACE^{\mathcal{M}}(f)$ are polynomially equivalent.*

Question. *Is this just a formal consequence of the fact that these inclusions relativize?*

For that we define an analog of TQBF for $SPACE^{\mathcal{M}}(f)$. We also use this analog to show that IP=PSPACE in communication complexity.

Proposition 5. *It holds that $IP^{cc}(f)$ and $SPACE^{cc}(f)$ are polynomially equivalent.*

Using an observation from [1], we show that IP≠PSPACE in the decision tree complexity.

Proposition 6. *We have $IP^{dt}(x_1 \wedge x_2 \wedge \ldots \wedge x_n) = \Omega(n)$, while by Observation 2 we have $SPACE^{dt}(x_1 \wedge x_2 \ldots \wedge x_n) \leqslant \log_2(n) + O(1)$.*

5 "TQBF" and "PSPACE⊆AP"

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

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 to a predicate symbol $P \in \mathcal{P}_{usual}$ assigns a predicate $P^{\mathcal{I}}$ over U of the same arity;

- a mapping that for $P \in \mathcal{P}_{special}$ and $x \in \{0,1\}^n$ assigns to a pair (P, x) a predicate $P_x^{\mathcal{I}}$ over U of the same arity as P.

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 $P_x^{\mathcal{I}}$ 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 $\phi_x^{\mathcal{I}}(a_1, \ldots, a_k)$ 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:*

$$g(x) = P_x^{\mathcal{I}}(a, a_2, \ldots, a_k)$$

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

Proposition 7. *There exists a finite X-signature $\mathcal{P}$ with just one univariate special symbol $P_{special}(\cdot)$ 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 $STATE^{\mathcal{M}}(f)$;*
- *a closed PNF formula φ over $\mathcal{P}$ of length $(SPACE^{\mathcal{M}}(f))^{O(1)}$ 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 $m = STATE^{\mathcal{M}}(f)$, 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 any non-terminal node there is an ordering on its out-going edges. The predicate symbol FirstSuccessor$(\cdot,\cdot)$ will be a usual predicate symbol. In $\mathcal{I}$ we will have 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, $P_{special}$ = Gollum will be our only special predicate symbol. We let $\text{Gollum}_x^{\mathcal{I}}(s)$ 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 $\psi_{\lceil l/2 \rceil}$:

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

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

This proposition implies that $A^{\mathcal{M}}(f) = \left(SPACE^{\mathcal{M}}(f)\right)^{O(1)}$, and proves Proposition 4. Indeed, we take an X-interpretation $\mathcal{I}$ and a formula

$$\varphi = Q_1 z_1\ Q_2 z_2 \ldots\ Q_m z_m\ \psi(z_1, \ldots, z_m)$$

from Proposition 7. We then construct an alternating $\mathcal{M}$-tree for f by emulating existential quantifiers with OR's and universal quantifiers with AND's. The quantifiers are over a set of size $STATE^{\mathcal{M}}(f)$, so each needs $\log_2 STATE^{\mathcal{M}}(f) = SPACE^{\mathcal{M}}(f)$ depth. The length of the formula is $\left(SPACE^{\mathcal{M}}(f)\right)^{O(1)}$, so there will be at most that many alternations between layers of OR's and AND's. Now, given $x \in \{0,1\}^n$, for any leaf we would have to evaluate $\psi_x^{\mathcal{I}}(a_1, \ldots, a_m)$ for some $a_1, \ldots, a_m$ from the support of $\mathcal{I}$. In ψ there are only two occurrences of the special symbol. So by $\mathcal{M}$-correctness we will need just two Bilbo's questions.

6 IP=PSPACE in communication complexity

We show that

$$IP^{cc}(f) = \left(SPACE^{cc}(f)\right)^{O(1)}.$$

By Proposition 7 we only have to do the following. Let $\mathcal{I}$ be a *cc*-correct X-interpretation with the support U of size 2^s, and let

$$\varphi = Q_1 z_1\ Q_2 z_2 \ldots\ Q_m z_m\ \psi(z_1, \ldots, z_m)$$

be a closed PNF formula of length $s^{O(1)}$ with just two occurrences of the $\mathcal{I}$'s unique special predicate symbol $P_{special}(\cdot)$. We will show that there is a $s^{O(1)}$-depth interactive proof *cc*-tree for the function

$$f\colon \{0,1\}^n \to \{0,1\}, \qquad f(x) = \varphi^{\mathcal{I}}_x, .$$

All we do is just adapting the Shamir's proof [2].

First, let us make the following technical modification. The special symbol $P_{special}$ is interpreted *cc*-correctly in $\mathcal{I}$. This means that for any $u \in U$ the function $g_x = (P_{special})^{\mathcal{I}}_x(u)$ either depends only on the first half of x or only on the second half. In the first case we will call u *an Alice's element*, and in the second case we will call u *a Bob's element*. Now, let as introduce two new special predicate symbols $P^A(\cdot)$ and $P^B(\cdot)$. By definition, we set

$$(P^A)^{\mathcal{I}}_x(u) = \begin{cases} (P_{special})^{\mathcal{I}}_x(u) & u \text{ is an Alice's element,} \\ 1 & u \text{ is a Bob's element,} \end{cases}$$

$$(P^B)^{\mathcal{I}}_x(u) = \begin{cases} 1 & u \text{ is an Alice's element} \\ (P_{special})^{\mathcal{I}}_x(u) & u \text{ is a Bob's element.} \end{cases}$$

So now $(P^A)^{\mathcal{I}}_x(u)$ always depends only on the first half of x, and $(P^B)^{\mathcal{I}}_x(u)$ always depends only on the second half of x.

Observe that $(P_{special})^{\mathcal{I}}_x(u) = (P^A)^{\mathcal{I}}_x(u) \wedge (P^B)^{\mathcal{I}}_x(u)$. We replace by this conjuction every occurrence of $P_{special}$ in φ.

Now we "booleanize" φ. I.e., we represent elements of the support as s-bit Boolean vectors. Next, in φ we replace each variable by a vector of s Boolean variables. After that every predicate symbol will depend on at most $2s$ Boolean variables (as before it was at most 2-variate).

Then we "algebrize" φ. For each usual predicate symbol $P(v_1, v_2, \ldots)$, where $v_1, v_2, \ldots$ are Boolean variables on which P depends, we consider a multi-linear polynomial $h_P \in \mathbb{F}_2[v_1, v_2, \ldots]$ such that:

$$P^{\mathcal{I}}(v_1, v_2, \ldots) = h_P(v_1, v_2, \ldots).$$

Such h_P exists because multi-linear polynomials can realize any Boolean function. Moreover, as P depends on at most $2s$ variables, the total degree of h_P is at most $2s$.

We do almost the same for P^A and P^B. For example, for P^A we take a family of multi-linear polynomials $\{h^{Alice}_a \in \mathbb{F}_2[v_1, \ldots, v_s] \mid a \in \{0,1\}^{n/2}\}$ such that:

$$(P^A)^{\mathcal{I}}_{a\ldots}(v_1, \ldots, v_s) = h^{Alice}_a(v_1, v_2, \ldots, v_s)$$

(what goes on after a in the subscript is irrelevant for P^A). Similarly, we take a family of multi-linear polynomials $\{h^{Bob}_b \in \mathbb{F}_2[v_1, \ldots, v_s] \mid b \in \{0,1\}^{n/2}\}$ such that:

$$(P^B)^{\mathcal{I}}_{\ldots b}(v_1, \ldots, v_s) = h^{Bob}_b(v_1, v_2, \ldots, v_s).$$

Then we compose the polynomials $h_P, P \in \mathcal{P}_{usual}$, h_a^{Alice}, h^{Bob} into a polynomial representing in a natural way the formula ψ (the part of φ after the quantifiers). More precisely, this will be a family of polynomials $h_{a,b}$, indexed by vectors $a, b \in \{0,1\}^{n/2}$, such that

$$\psi_{ab}^{\mathcal{I}}(v_1, v_2, \ldots) = h_{a,b}(v_1, v_2, \ldots).$$

The total degree of $h_{a,b}$ will be $s^{O(1)}$, because it is composed from polynomials of total degree $O(s)$, and because the formula ψ has length $s^{O(1)}$.

Finally, we just let Prover and Verifier to perform exactly the Shamir's protocol. The difference will be only when the communication of Prover and Verifier is finished. In this moment Alice and Bob will have to do the following. From the Shamir's protocol we will be left with some vector $(r_1, r_2, \ldots)$ of elements of some $s^{O(1)}$-size finite field $\mathbb{F} \supset \mathbb{F}_2$, and with some polynomial $h' \in \mathbb{F}[v_1, v_2, \ldots]$. Alice and Bob will have to check, whether

$$h'(r_1, r_2, \ldots) = h_{a,b}(r_1, r_2, \ldots).$$

For that they, in fact, need only $O(\log s)$ bits of communication. Namely, all they need to do is to compute each of h_a^{Alice}, h_b^{Bob} on at most two different inputs (because there are only two occurrences of P^A and of P^B in ψ). The values of these polynomials will belong to the field $\mathbb{F}$, so it we will only take $O(\log s)$ bits to send each other these values.

Remark. *The proof shows that WLOG Alice and Bob communicate at most $O(\log d)$ bits in d-depth interactive proof trees. Compare it to alternating trees, where WLOG in the bottom there's always just one bit sent.*

7 IP≠PSPACE in decision tree complexity

Lemma 8. *Let $f\colon \{0,1\}^n \to \{0,1\}$ be a Boolean function and let $x \in f^{-1}(1)$ be a 1-input to f with block sensitivity k with respect to f. Next, let T be a randomized decision tree that always queries at most d bits. Assume that $\mathbf{Pr}[T(x) = 1] \geqslant 2/3$ but $\mathbf{Pr}[T(y) = 1] \leqslant 1/3$ for any $y \in f^{-1}(0)$. Then $d \geqslant k/100$.*

Proof. Assume for contradiction that $d < k/100$. Consider k disjoint blocks realizing the block sensitivity of x with respect to f. The tree T always queries less than $k/100$ blocks. So there is a block which is queried by T with probability at most $1/100$. Flip x inside this block, and let x' be the result. Note that $f(x') = 0$. But with probability at least $99/100$ over the random bits of T its output on x and on x' coincide. So it is impossible that we have simultaneously $\mathbf{Pr}[T(x) = 1] \geqslant 2/3$ and $\mathbf{Pr}[T(x') = 1] \leqslant 1/3$. □

We use this lemma to prove Proposition 6. Assume that there is a d-depth IP tree for $x_1 \wedge x_2 \wedge \ldots \wedge x_n$. We take a Verifier's strategy that convinces Prover that $1 \wedge 1 \wedge \ldots \wedge 1 = 1$. We plug-in this strategy into our interactive proof tree. We obtain a randomized decision tree T, querying at most d bits, such that $\mathbf{Pr}[T(11\ldots1) = 1] \geqslant 2/3$ but $\mathbf{Pr}[T(y) = 1] \leqslant 1/3$ for every $y \neq 11\ldots1$. Clearly, the block sensitivity (and also the sensitivity) of $11\ldots1$ with respect to $x_1 \wedge x_2 \wedge \ldots \wedge x_n$ is n. So by our lemma $d = \Omega(n)$.

References

[1] Raz, R., Tardos, G., Verbitsky, O., and Vereshchagin, N. Arthur–merlin games in boolean decision trees. *Journal of Computer and System Sciences 59*, 2 (1999), 346–372.

[2] Shamir, A. Ip= pspace. *Journal of the ACM (JACM) 39*, 4 (1992), 869–877.