COSC 544 Probabilistic Proof Systems 10/12/17

Introduction to MIPs

Lecturer: Justin Thaler

This lecture introduces the notions of multi-prover interactive proofs (MIPs). In the next lecture, we will see a state of the art MIP.

While MIPs are of some interest in their own right, we will see later that they can be building blocks for constructing succinct arguments. In particular, at the end of the next lecture, we briefly outlines how to get a succinct argument from a state of the art MIP.[1] We also cover them because of their historical importance, and because the state of the art MIP in the next lecture exhibits several ideas that will recur in more complicated forms later in the course.

1 MIPs: Definitions and Basic Results

Definition 1.1. A k-prover interactive proof protocol for a language $\mathcal{L} \subseteq \{0,1\}^*$ involves $k+1$ parties: a probabilistic polynomial time verifier, and k provers. The verifier exchanges a sequence of messages with each prover; each prover's message is a function of the input and the messages from $\mathcal{V}$ that it has seen so far. The interaction produces a transcript $\mathbf{t} = (\mathcal{V}(\mathbf{r}), \mathcal{P}_1, \ldots, \mathcal{P}_k)(\mathbf{x})$, where $\mathbf{r}$ denotes $\mathcal{V}$'s internal randomness. After the transcript $\mathbf{t}$ is produced, $\mathcal{V}$ decides whether to output accept or reject based on $\mathbf{r}$, $\mathbf{t}$, and $\mathbf{x}$. Denote by $\text{out}(\mathcal{V}, \mathbf{x}, \mathbf{r}, \mathcal{P}_1, \ldots, \mathcal{P}_k)$ the output of verifier $\mathcal{V}$ on input $\mathbf{x}$ given prover strategies $(\mathcal{P}_1, \ldots, \mathcal{P}_k)$ and that $\mathcal{V}$'s internal randomness is equal to $\mathbf{r}$.

The interactive proof system has completeness error δ_c and soundness error δ_s if the following two properties hold.

1. *(Completeness)* There exists a tuple of prover strategies $(\mathcal{P}_1, \ldots, \mathcal{P}_k)$ such that for every $\mathbf{x} \in \mathcal{L}$,

$$\Pr[\text{out}(\mathcal{V}, \mathbf{x}, \mathbf{r}, \mathcal{P}_1, \ldots, \mathcal{P}_k) = \text{accept}] \geq 1 - \delta_c.$$

2. *(Soundness)* For every $\mathbf{x} \notin \mathcal{L}$ and every tuple of prover strategy $(\mathcal{P}'_1, \ldots, \mathcal{P}'_k)$,

$$\Pr[\text{out}(\mathcal{V}, \mathbf{x}, \mathbf{r}, \mathcal{P}'_1, \ldots, \mathcal{P}'_k) = \text{accept}] \leq \delta_s.$$

Say that a k-prover interactive proof system is valid if $\delta_c, \delta_s \leq 1/3$. The complexity class **MIP** is the class of all languages possessing valid k-prover interactive proof systems, for some $k = \text{poly}(n)$.

The MIP model was introduced by Ben-Or, Goldwasser, Kilian, and Wigderson [BGKW88]. It is crucial in Definition 1.1 that each prover's message is a function only of the input and the messages from $\mathcal{V}$ that it has seen so far. In particular, $\mathcal{P}_i$ cannot tell $\mathcal{P}_j$ what messages $\mathcal{V}$ has sent it, or vice versa, for any $i \neq j$. If such "cross-talk" between $\mathcal{P}_i$ and $\mathcal{P}_j$ were allowed, then it would be possible to simulate any MIP by a single-prover interactive proof, and the classes **MIP** and **IP** would become equal.

It can be helpful to think of MIP as follows. The provers are like prisoners who are about to be interrogated. The prisoners get placed in separate interrogation rooms. Prior to going into these rooms, the

[1] The approach to obtaining succinct arguments described in the next lecture has not been previously published; the only published approach to turning MIPs into succinct arguments [BC12] makes use of a cryptographic primitive known as fully homomorphic encryption, which is currently impractical.

prisoners can talk amongst themselves, plotting a strategy for answering questions. But once they are placed in the rooms, they can no longer talk to each other, and in particular prover i cannot tell the other provers what questions the verifier is asking it. The verifier is like the interrogator, trying to determine if the prover's stories are consistent with each other, and with the claim being asserted.

The next section shows that, up to polynomial blowups in $\mathcal{V}$'s runtime, 2-prover MIPs are just as expressive as k-prover MIPs, for any $k = \text{poly}(n)$.

1.1 What Does a Second Prover Buy?

Non-Adaptivity. In a single-prover interactive proof, the prover $\mathcal{P}$ is allowed to act adaptively, in the sense that $\mathcal{P}$'s response to the ith message m_i sent from $\mathcal{V}$ is allowed to depend on the preceding $i-1$ messages. Intuitively, the reason that MIPs are more expressive than IPs is that the presence of a second prover (who does not know $\mathcal{V}$'s messages to the first prover) prevents the first prover from behaving in this adaptive manner. This can be formalized via the following easy lemma showing that the complexity class **MIP** is equivalent to the class of languages satisfied by polynomial time randomized *oracle machines*. Here, an oracle machine is essentially a computer that has query access to a giant string $\mathcal{O}$ that is fixed at the start of the computer's execution. The string $\mathcal{O}$ may be enormous, but the computer is allowed to look at any desired symbol $\mathcal{O}_i$ (i.e., the ith symbol of $\mathcal{O}$) in unit time. One can think of any query that the computer makes to $\mathcal{O}$ as a question, and $\mathcal{O}_i$ as the answer. Because $\mathcal{O}$ is fixed at the start of the computer execution, the answers are returned by $\mathcal{O}$ are non-adaptivem in the sense that the answer to the computer's jth question *does not depend on which questions the computer asked previously*.

Lemma 1.2 ([FRS88])**.** *Let $\mathcal{L}$ be a language, and M a probabilistic Turing Machine such that $\mathbf{x} \in \mathcal{L} \Longrightarrow \exists$ an oracle $\mathcal{O}$ such that $M^{\mathcal{O}}$ accepts $\mathbf{x}$ with probability 1, and $\mathbf{x} \notin \mathcal{L} \Longrightarrow \forall$ oracles $\mathcal{O}$, $M^{\mathcal{O}}$ rejects $\mathbf{x}$ with probability at least $2/3$. Then there is a (2-prover) MIP for $\mathcal{L}$.*

Remark 1. In Lemma 1.2, one can think of $\mathcal{O}$ as a giant purported proof that $x \in \mathcal{L}$, and machine M only looks at a small (i.e., polynomial) number of symbols of the proof. This is the same notion as a *probabilistically checkable proof*, which we will introduce in a couple of lectures. In this terminology, Lemma 1.2 states that any PCP with a polynomial time verifier can be turned into a 2-prover MIP with a polynomial time verifier.

Proof. $\mathcal{V}$ simulates M, and every time M poses a query q to the oracle, $\mathcal{V}$ asks the query to $\mathcal{P}_1$, treating $\mathcal{P}_1$'s response as $\mathcal{O}(q)$. At the end of the protocol, $\mathcal{V}$ picks a query q uniformly at random from all queries that were posed to $\mathcal{P}_1$, and poses it to $\mathcal{P}_2$, rejecting if $\mathcal{P}_2$'s response to q does not equal $\mathcal{P}_1$'s. Finally, the protocol is repeatedly independently $3m$ times, where m is (an upper bound on) the number of queries M poses to the oracle on any input $\mathbf{x} \in \{0,1\}^n$. $\mathcal{V}$ accepts only if all instances accept.

Completeness is clear: if $\mathbf{x} \in \mathcal{L}$, there is some oracle $\mathcal{O}^*$ causing M to accept $\mathbf{x}$ with probability 1. If $\mathcal{P}_1$ and $\mathcal{P}_2$ respond to any query q with $\mathcal{O}^*(q)$, then $\mathcal{V}$ will accept $\mathbf{x}$ on each of the runs of the protocol with probability 1.

For soundness, observe that since $\mathcal{P}_2$ is only asked a single query, we can treat $\mathcal{P}_2$ as an oracle $\mathcal{O}$. That is, $\mathcal{P}_2$'s answer on query q is a function only of q. On any run of the protocol, let $q_1, \ldots, q_m$ denote the queries that $\mathcal{V}$ poses to $\mathcal{P}_1$ on input $\mathbf{x}$. On the one hand, if $\mathcal{P}_1$ ever answers a query q_i differently than $\mathcal{O}(q_i)$, the verifier will pick that query to pose to $\mathcal{P}_2$ and reject, with probability at least $1/m$. On the other hand, if $\mathcal{P}_1$ answers every query q_i with $\mathcal{O}(q_i)$, then $\mathcal{V}$ will reject with probability at least $2/3$ because $M^{\mathcal{O}}$ rejects with that probability. Therefore, $\mathcal{V}$ rejects on each run of the protocol with probability at least $1/m$, and hence $\mathcal{V}$ rejects on at least one run of the protocol with probability at least $1-(1-1/m)^{3m} > 2/3$. □

The same argument implies that any k-prover MIP (with completeness error at most $\delta_c \leq 1/(9m)$, where m is the total number of queries asked) can be simulated by a 2-prover MIP [BGKW88]: in the simulation, $\mathcal{V}$ poses all of the questions from the k-prover MIP to $\mathcal{P}_1$, then picks a question at random and pose it $\mathcal{P}_2$, rejecting if the answers do not agree. $\mathcal{P}_2$ can be treated as an oracle, and if $\mathcal{P}_1$ answers even a single query q_i "non-adaptively" (i.e. different than how $\mathcal{P}_2$ would answer), the probability this is detected is at least $1/m$. The whole 2-prover protocol must be repeated $\Omega(m)$ times to drive the soundness error from $1/m$ down to $1/3$.

In summary, one can both force non-adaptivity and reduce the number of provers to 2 by posing all queries to $\mathcal{P}_1$ and choosing one of the queries at random to pose to $\mathcal{P}_2$. While this conveys much of the intuition for why MIPs are more expressive than IPs, the technique is very expensive in practice, due to the need for $\Omega(m)$ repetitions (typically, m is on the order of $\log n$, and can easily be in the hundreds in implementations). Fortunately, the MIP that we describe in the next lecture requires only two provers without the need for repetition to force non-adaptivity or reduce the number of provers to 2.

But What Does Non-Adaptivity Buy? We will see next lecture that non-adaptivity buys *succinctness*. That is, we will give a MIP for arithmetic circuit satisfiability in which the total communication and verifier runtime is sublinear in the size of the witness w.

This should not be surprising, as we saw essentially the same phenemenon in the previous lecture. There, we used a polynomial commitment scheme to *bind* the prover to a multilinear polynomial $\tilde{w}$ that was fixed at the start of the interaction with the verifier. In particular, the polynomial commitment scheme forced the prover to tell the verifier $\tilde{w}(r)$, without allowing the prover to "change its answer" based on the interaction with the verifier.

References

[BC12] Nir Bitansky and Alessandro Chiesa. Succinct arguments from multi-prover interactive proofs and their efficiency benefits. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 255–272. Springer, 2012.

[BGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 113–131. ACM, 1988.

[FRS88] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-power interactive protocols. In *Structure in Complexity Theory Conference, 1988. Proceedings., Third Annual*, pages 156–161. IEEE, 1988.