IAS/PCMI Summer Session 2000
Clay Mathematics Undergraduate Program
Basic Course on Computational Complexity

Lecture 14: IP = PSPACE

David Mix Barrington and Alexis Maciel

August 3, 2000

1. Overview

We have just defined interactive proof systems and the class IP of languages definable by poly-time interactive proof systems. We showed that such systems can be simulated by deterministic machines with polynomial space, so that IP $\subseteq$ PSPACE. Here we present the theorem of Shamir that in fact IP = PSPACE. Thus, surprisingly, interactive proofs with a randomized poly-time Bob have the same power to define languages as do alternating machine games with a clever Bob.

- We begin with an overview of the proof, which will center on a randomized version of the *Savitch game* from Advanced Lecture 11. Alice will begin by asserting that a certain path exists in the configuration graph of the poly-space machine. This assertion will be placed in the form of a *multilinear polynomial* in poly-many variables over the finite field $\mathbf{Z}_p$ for some prime p. The remainder of the game will consist of successive *links* from one assertion about such polynomials to another, concluding with one that can be verified. At each link we will ensure that Alice can always move to the next assertion if she is telling the truth, and has only a tiny chance of fooling Bob by moving from a false to a true assertion.

- We set up the construction of an interactive proof system to define the language of an arbitrary PSPACE machine. We represent the configurations of our machine M as strings of ℓ bits, where ℓ is polynomial in n. The key predicate $\mathrm{PATH}_i(\alpha_1, \ldots, \alpha_\ell, \beta_1, \ldots, \beta_\ell)$ will have value 1 if there is a path of length exactly 2^i from configuration α to configuration β, and have value 0 if there is no such path. We give a recursive definition of these predicates for all i from 0 to ℓ.

- We define a *multilinear polynomial* over the finite field $\mathbf{Z}_p$ for every path predicate, with 2ℓ variables ranging over $\mathbf{Z}_p$. When these variables all take values from $\{0, 1\}$, the polynomial will have a value from $\{0, 1\}$ matching the value of the boolean predicate. But the polynomial will be defined with other sets of values, and assertions about its behavior on such inputs will be crucial to our proof. We establish here that the PATH_0 polynomial can be evaluated on arbitrary inputs in $\mathbf{Z}_p$ by a poly-time deterministic machine.

- We define the general notion of an ϵ-link between assertions, an interactive protocol such that a true assertion leads to a true one (under optimal play by Alice) with probability one, and a false assertion leads to a false one (whatever Alice does) with probability $1 - \epsilon$. We describe the structure of the $O(\ell^2)$ ϵ-links we will use to connect Alice's initial assertion (equivalent to "$x \in A$") with a final assertion about the PATH_0 polynomial that can be verified in P. We summarize how this will prove that IP $\subseteq$ PSPACE.

- Finally we present and justify the series of ϵ-links used in the protocol, particularly the assertions about *hybrid polynomials* needed to simulate Alice's moves in the Savitch game.

2. Overview of the Proof

We need to construct an interactive proof system where Alice can convince Bob that $x \in A$, where A is an arbitrary PSPACE language, iff this is actually true. Sipser and Papadimitriou each present versions of Shamir's original proof, where Alice solves an instance of the PSPACE-complete problem QBF. In this presentation, which we have gratefully taken from Steven Riudich, we will have Alice and Bob play a variant of the Savitch game that we used to prove PSPACE $\subseteq$ AP in Basic Lecture 11.

Alice will begin with an assertion that a long path exists in the configuration graph of the machine M on input x. In the Savitch game, she presents Bob with an alleged midpoint of this path and Bob chooses either the first half or the second half of the alleged path to dispute. In one round we move from an assertion that a path exists to an assertion that another path exists, where the second is half the length of the first. In polynomially many rounds we reduce to the case of a single edge in the configuration graph, whose existence can be determined in P.

The new game will have the same structure as the Savitch game but deal with formally different objects. We will *arithmetize* the predicates about paths by replacing them with multilinear polynomials over some finite field $\mathbf{Z}_p$. These polynomials will have the same value as the path predicate (0 or 1) if their inputs are all 0's and 1, and will have other values on other inputs. In the new game, Alice's original polynomial will deal with paths of length 2^ℓ, and in the first round she will eventually replace her assertion about the original polynomial by an assertion about the polynomial dealing with paths of length $2^{\ell-1}$. But, crucially, this assertion will be about the value of this polynomial on some random setting of the variables to values in $\mathbf{Z}_p$, a setting which in all probability says nothing about paths at all. But after ℓ such rounds, Alice's assertion will be about the polynomial dealing with paths of length $2^0 = 1$, and we will ensure that this assertion can be verified in P as before.

At each stage, if Alice's assertion is true, she can move so that with certainty the resulting assertion is also false. If her assertion is false, we can assure that the resulting assertion is true only with probability less than some ϵ, which can be an inverse polynomial. This is called an "ϵ-link" between the two assertions. We will have $p(n)$ ϵ-links in our entire protocol, yielding a protocol where Bob's probability of being fooled by a false original assertion from Alice is at most $\epsilon p(n)$, which by choice of ϵ we can make less than any given constant. We thus are building an interactive proof system demonstrating that the language A is in IP. Since A was an arbitrary PSPACE language, we will have shown PSPACE $\subseteq$ IP and thus (with a result from Basic Lecture 13) that IP = PSPACE.

3. The Path Predicates For the Machine

We represent configurations of our machine M by strings of ℓ bits, where ℓ is a polynomial in n. For any number i and any two configurations $\alpha = (\alpha_1, \ldots, \alpha_\ell)$ and $\beta = (\beta_1, \ldots, \beta_\ell)$, we define the boolean predicate $\mathrm{PATH}_i(\alpha, \beta)$ to be true iff there is a path in the configuration graph of M of length *exactly* 2^i from α to β. Note that PATH_i refers to paths of length 2^i, a convention we adopt to reduce the level of subscripting and superscripting in our proof.

The predicate PATH_0 is true of two configurations α and β iff β is the unique successor configuration of α in M, given the specific input x. By our conventions for machines, the correct successor for α agrees with α except on $O(1)$ bits, those specifying the state, the head positions, and the contents of the $O(1)$ memory bits that are to be affected by heads. (An interesting technicality: so that we will not have to worry about carries in the numbers representing the head positions, we can store them in *unary*. We have the room, as the space needed will still be polynomial.)

The predicate PATH_0 can be represented as the OR, over the polynomially many choices of head positions, of the AND of formulas asserting the equality of the unaffected bits and a formula referring to a specific set of $O(1)$ bits. We will use this fact below in designing a polynomial to arithmetize PATH_0 so that we can evaluate it in P even when its inputs are in $\mathbf{Z}_p$.

For each i, $\mathrm{PATH}_{i+1}(\alpha, \beta)$ is true iff there is a configuration γ such that $\mathrm{PATH}_i(\alpha, \gamma)$ and $\mathrm{PATH}_i(\gamma, \alpha)$ are both true. We will use this fact in the following inductive definitions of the PATH_i predicates:

$$\mathrm{PATH}_{i+1}(\alpha, \beta) \leftrightarrow \bigvee_{\gamma_1=0,1} \dots \bigvee_{\gamma_\ell=0,1} \mathrm{PATH}(\alpha_1, \dots, \alpha_\ell, \gamma_1, \dots, \gamma_\ell) \wedge \mathrm{PATH}(\gamma_1, \dots, \gamma_\ell, \beta_1, \dots, \beta_\ell)$$

The predicate $\mathrm{PATH}_\ell(s, t)$, where s is the unique starting configuration and t is the unique accepting final configuration, will be the starting point of our proof, as Alice's initial assertion is that this equals 1. Note that we must arrange that M will halt in exactly 2^ℓ steps rather than at most 2^ℓ, because PATH_ℓ is defined in terms of paths of exactly that length. Also, we must insist that the machine M clean up its memory and put its heads in some standard places when it accepts, so that the accepting configuration will be unique.

4. Arithmetizing the Path Predicates

Along with these predicates we will define a family of multilinear polynomials over a field $\mathbf{Z}_p$, also called PATH_i and also taking 2ℓ variables $\alpha_1, \dots, \alpha_\ell, \beta_1, \dots, \beta_\ell$. When each of the 2ℓ inputs to one of these polynomials is from the set $\{0, 1\}$, we insist that the polynomial take on the value 0 or 1 according to the value of the corresponding boolean predicate. The set of multilinear polynomials over 2ℓ variables is a vector space of dimension $2^{2\ell}$ over $\mathbf{Z}_p$, so it turns out that we have just enough degrees of freedom to specify the value of the polynomial on the $2^{2\ell}$ boolean settings. (In an exercise you are asked to verify this.)

The prime p. by the way, will come from a range to be specified later. The prover Alice will be responsible for choosing it and giving Bob a proof that it is prime. (Such proofs exist by an exercise in Basic Lecture 11.)

Lemma 1 *The polynomial* $\mathrm{PATH}_0(\alpha_1, \dots, \alpha_\ell, \beta_1, \dots, \beta_\ell)$ *can be evaluated in polynomial time (in n or ℓ) even if the α_i's and β_i's take on arbitrary values in $\mathbf{Z}_p$.*

Proof Remember that the predicate PATH_0 is the OR, over the possible positions of the heads, of the AND of predicates asserting that the unaffected bits of α and β

are equal, and a specific formula saying that the affected $O(1)$ bits change according to the rules of M. Since the OR is over disjoint possibilities, we can define the polynomial PATH_0 to be the sum, over the possible head positions, of the product of an equality polynomial for each unaffected bit position i, and a polynomial returning the right values when the affected $O(1)$ bits have the right values.

The equality polynomial for α_i and β_i is just $1 - \alpha_i - \beta_i + \alpha_i\beta_i$. This takes the correct value when both inputs are boolean, and some value we don't know how to interpret in all other cases.

Let's temporarily refer to the affected $O(1)$ bits of α as $a_1, \ldots, a_k$ and the matching bits of β as $b_1, \ldots, b_k$. For each sequence of k boolean values for the a's, there is a unique proper sequence of boolean values for the b's. For each of these pairs, we can define a multilinear polynomial in these $2k$ variables that evaluates to 1 if they take the correct $2k$ values and to 0 if they take on any incorrect sequence of *all boolean* values. (This is the product of a_i or b_i when they should be 1 and $1-a_i$ or $1-b_i$ when they should be zero.) The polynomial determining whether the $2k$ affected variables are correct is the sum of these polynomials over all sequences $a_1, \ldots, a_k$.

The resulting polynomial is the polynomial sum of a polynomial product of polynomials of size $O(1)$, and is thus easy to evaluate in P over the field $\mathbf{Z}_p$ as long as p has only polynomially many bits. Note for future reference that this construction never multiplied a variable by itself, so we can be assured that PATH_0 is a multilinear polynomial. □

We now have to define the polynomials PATH_i for $i > 0$. So that these will have the correct values on boolean inputs, we must use an inductive definition matching the inductive definition of the predicates PATH_i. Since the midpoint of any path of a given length from α to β is unique, the 2^ℓ cases for different γ's are mutually exclusive, and we may calculate the OR of the different boolean values by simply adding them in $\mathbf{Z}_p$. The binary AND can be modeled by a product. Note, by the way, that this product does *not* ever multiply any variable by itself. The γ_i's are constants, not variables, so we are multiplying a polynomial containing only α_i's by one containing only β_i's. Here is our arithmetization:

$$\mathrm{PATH}_{i+1}(\alpha, \beta) \leftrightarrow \sum_{\gamma_1=0,1} \cdots \sum_{\gamma_\ell=0,1} \mathrm{PATH}(\alpha_1, \ldots, \alpha_\ell, \gamma_1, \ldots, \gamma_\ell)\mathrm{PATH}(\gamma_1, \ldots, \gamma_\ell, \beta_1, \ldots, \beta_\ell)$$

Each polynomial PATH_i is multilinear in its 2ℓ variables. However, note that in general such a polynomial has $2^{2\ell}$ possible terms, and except for PATH_0 we have no convenient representation for such a polynomial or any way to manipulate it with a poly-time machine. We will have to deal with these polynomials *implicitly*, through assertions about them. A typical single assertion is that the value of a particular PATH_i, on particular inputs α and β in $\mathbf{Z}_p^\ell$, is some particular value $r \in \mathbf{Z}_p$. We now have to see how the proof can move from the original assertion about PATH_ℓ to the final one about PATH_0 while maintaining Bob's confidence that the truth values of these assertions are equivalent.

5. Links Between Assertions

Our basic step in the proof is called an ϵ-link. This is an interactive protocol that takes an initial assertion by Alice and produces a final assertion, which may depend on random choices during the protocol. The two key properties of an ϵ-link are:

- If Alice's initial assertion is true, her final assertion will be true with probability 1.
- If Alice's initial assertion is false, either Bob rejects during the protocol or Alice's final assertion will be true with probability at most ϵ.

Our proof will consist of ℓ rounds, each consisting of $\ell+1$ ϵ-links. Thus by choosing our ϵ so that $\epsilon(\ell^2 + \ell) < 1/3$, we will limit Bob's total probability of being fooled by an initially incorrect assertion by Alice to less than 1/3, and thus we will define a valid interactive proof. The initial assertion, $\mathrm{PATH}_\ell(s, t)$, is true iff $x \in A$. The final assertion, $\mathrm{PATH}_0(\alpha, \beta)$ for some constant sequences α and β in $\mathbf{Z}_p^\ell$ that are determined during the protocol, can be tested in P by our lemma above, and we have Bob accept iff this final assertion is true. If $x \notin A$, and if none of the events of an ϵ-link failing occur, Bob eventually rejects, and this happens with probability at least 2/3.

In any given round we begin with an assertion $\mathrm{PATH}_{i+1}(\alpha, \beta) = r$ and we want to end with an assertion $\mathrm{PATH}_i(\alpha', \beta') = s$. Our first ℓ ϵ-links take us to an assertion $\mathrm{PATH}_i(\alpha, \gamma)\mathrm{PATH}_i(\gamma, \beta) = t$, where γ is a new sequence of ℓ *constants* from $\mathbf{Z}_p$. This process will be described more fully in the final section.

Let's now look at the single ϵ-link from the assertion $\mathrm{PATH}_i(\alpha, \gamma)\mathrm{PATH}_i(\gamma, \beta) = t$ to an assertion about a single PATH_i value on some α' and β'. Alice begins the process (if she is honest) by computing a polynomial

$$Q(x) = \mathrm{PATH}_i(\alpha + (\gamma - \alpha)x, \gamma + (\beta - \gamma)x)$$

and sending its coefficients to Bob. Note that despite the huge number of constants α_i, β_i, and γ_i it contains, $Q(x)$ is a *linear* polynomial in x over $\mathbf{Z}_p$. Also note that $Q(0)$ is exactly $\mathrm{PATH}_i(\alpha, \gamma)$ and that $Q(1)$ is exactly $\mathrm{PATH}_i(\gamma, \beta)$. If Alice is working with a false assertion, she will send some other linear polynomial of her own choosing.

Bob's role in the protocol is to first check that $Q(0)Q(1) = t$, that is, that $Q(x)$ as sent by Alice is consistent with Alice's current assertion. If it isn't, he rejects. Otherwise, he chooses a random number z uniformly from $\mathbf{Z}_p$ and makes the new assertion $Q(z)$, which is equal to $\mathrm{PATH}_i(\alpha + (\gamma - \alpha)z, \gamma + (\beta - \gamma)z)$. Thus he sets the new α' to be $\alpha + (\gamma - \alpha)z$ and the new β' to be $\gamma + (\beta - \gamma)z$. The final assertion is $\mathrm{PATH}_i(\alpha', \beta') = s$, where s is $Q(z)$ as computed from the polynomial $Q(x)$ supplied by Alice. If the initial assertion was true, Alice's best strategy is to give the correct $Q(x)$, and this final assertion will be true as well. (Note that Alice needs her cleverness to compute these coefficients!)

What is the chance of error in this protocol, assuming Alice starts with a false assertion and Bob doesn't reject during the protocol? Alice must have given a false $Q(x)$ to Bob, since hers had $Q(0)Q(1) = t$ and the real one, based on the real PATH_i on the given values, didn't. Two different linear polynomials in one variable over a field can agree on at most one value of their variable. The only problem for Bob is if his randomly chosen z happens to be this value, which occurs with probability at most $1/p$. By choice of p, we can make this probability less than ϵ as desired, since ϵ is only an inverse polynomial.

6. Hybrid Polynomials and Links Between Them

To finish the proof, we now have only to explain the series of ℓ ϵ-links that take us from the assertion $\mathrm{PATH}_{i+1}(\alpha, \beta) = r$ to the assertion $\mathrm{PATH}_i(\alpha, \gamma)\mathrm{PATH}_i(\gamma, \beta)$. To do this we define a sequence of *hybrid polynomials*, each a function of α and β. They range from

$$H_0(\alpha, \beta) = \mathrm{PATH}_{i+1}(\alpha, \beta) = \sum_{\gamma_1=0,1} \cdots \sum_{\gamma_\ell=0,1} \mathrm{PATH}_i(\alpha, \gamma)\mathrm{PATH}_i(\gamma, \beta)$$

through

$$H_j(\alpha, \beta) = \sum_{\gamma_{j+1}=0,1} \cdots \sum_{\gamma_\ell=0,1} \mathrm{PATH}_i(\alpha, \gamma)\mathrm{PATH}_i(\gamma, \beta),$$

where γ_1 through γ_j are replaced by constants chosen during the protocol, to

$$H_\ell = \mathrm{PATH}_i(\alpha, \gamma)\mathrm{PATH}_i(\gamma, \beta)$$

where all the γ variables have been so chosen.

Our ϵ-links will be from the initial assertion $H_0(\alpha, \beta) = r = h_0$ through intermediate assertions $H_j(\alpha, \beta) = h_j$ to the final assertion $H_\ell = h_\ell = t$. We need to establish an ϵ-link from the general assertion $H_j = h_j$ to $H_{j+1} = h_{j+1}$. This will be by a technique similar to the ϵ-link in the previous section.

Note that H_j is the sum of two polynomials, H_{j+1} with 0 in the role of γ_{j+1} and H_{j+1} with 1 in that role. Again we define a linear polynomial $R(x)$, where $R(x)$ is H_{j+1} with x in the role of γ_{j+1}, and the given values for all other constants. If she is dealing with a true assertion, Alice (with her cleverness) computes $R(x)$ and sends its coefficients to Bob. Bob checks that $R(0) + R(1) = h_j$ and rejects if it isn't. If it checks out, Bob chooses a random number γ_{j+1} uniformly from $\mathbf{Z}_p$ and sets H_{j+1} to have this value for γ_{j+1}. He sets h_{j+1} to be $R(\gamma_{j+1})$.

Once again Alice is best served by telling the truth if she starts with a true assertion. Otherwise, she must provide a false polynomial $R(x)$ to Bob (since the true polynomial will not satisfy $R(0) + R(1) = h_j$) and Bob's random choice will lead to a false assertion *unless* he chooses the one value on which Alice's linear polynomial and the real one agree. This again happens with probability at most $1/p < \epsilon$.

7. Exercises

1. It is true, of course, that NPSPACE $\subseteq$ IP from the result proved here and Savitch's Theorem. What issues arise when we try to carry out the proof above starting with a nondetermistic poly-space machine M instead of a deterministic one?

2. Let $P(x_1, \ldots, x_n)$ be a boolean predicate and let $p > 2$ be a prime. Show that there exists a unique multilinear polynomial $f(y_1, \ldots, y_n)$ over $\mathbf{Z}_p$ such that whenever the y_i's are all taken from the set $\{0, 1\}$, $f(y_1, \ldots, y_n) = P(y_1, \ldots, y_n)$. This is similar but not identical to an exercise in Basic Lecture 10.

3. Prove carefully that a sequence of assertions $P_0, \ldots, P_m$, each linked to its successor by an ϵ-link, shows the existence of an $m\epsilon$-link from P_0 to P_m.

4. Prove that for any poly-time interactive protocol, there is another such protocol defining the same language A where in the second protocol, Alice is always successful in proving that $x \in A$ when this is true. (Recall that the definition in Basic Lecture 13 did not ensure this.)

5. Using the techniques of this lecture, give a direct construction of an interactive proof system for the language COUNT-SAT. This is the set of pairs $\langle \phi, m \rangle$ where ϕ is a boolean formula with exactly m satisfying assignments. Show that this can easily be adapted to give an interactive proof system for SAT itself. Historically, Shamir's proof followed directly upon a breakthrough by Lund-Fortnow-Karloff-Nisan (that COUNT-SAT was in IP) and further work by Babai and others. The frenzy of electronic communication during the "race" to the presumed result IP = PSPACE was remarkable, ending only when Shamir won the race.