 Handout N11
10/1/2002 (first draft)

Notes for Lecture 11

Today we show that **PSPACE** and **EXP** contain problems that are as hard to solve in the worst case as they are to solve on average. We first prove such a statement for average-case algorithms that succeed on about a 7/8 fraction of inputs. Then we show one way to prove the statement even for algorithms that suceed on about a 1/2 fraction of inputs.

1 Average-case Complexity in PSPACE and EXP for 7/8 Fraction of Inputs

Lemma 1 (Multivariate Interpolation) *Let $\mathcal{F}$ be a field, $H \subseteq \mathcal{F}$ a subset and $f : H^m \to \mathcal{F}$ an arbitrary function. Then there is a m-variate polynomial p of degree at most $m|H|$ such that $p(x) = f(x)$ for every $x \in H^m$. Furthermore, the value $p(x)$ can be computed using time polynomial in $|\mathcal{F}|^m$ and space logarithmic in $|\mathcal{F}|^m$ given oracle access to f.*

Let L be an **EXP**-complete problem (the same construction and reasoning can be repeated with a **PSPACE**-complete problem), and for every n let $L_n : \{0,1\}^n \to \{0,1\}$ the characteristic function of L on inputs of length n. We will construct another **EXP**-complete problem L' such that for every n there is a $n' = O(n)$ such that the truth-table of $L'_{n'}$ is the encoding of the truth-table of L_n using a concatenation of Reed-Muller and Hadamard codes. Then we will be able to show that L' is as hard in the worst case as it is hard on average.

Let us fix an input length n.

Let $k = 3\lceil \log n \rceil$, and let $\mathcal{F}$ be a field of size $2^k = O(n^3)$. We will identify elements of $\mathcal{F}$ with vectors of $\{0,1\}^k$ and, in particular, the zero element of $\mathcal{F}$ is mapped into $(0,\ldots,0)$ and the one element of $\mathcal{F}$ is mapped into $(1,0,\ldots,0)$. Let $H \subseteq \mathcal{F}$ be a subset of size $2^{k/3}$. Let $m = 3n/k$, then H^m contains at least 2^n elements and there is some efficiently computable injective function $h : \{0,1\}^n \to H^{k/2}$. From Lemma 1 it follows that there is a multivariate polynomial p of total degree at most $O(n^2/\log n)$ such that $p(h(x)) = L_n(x)$ for all $x \in \{0,1\}^n$. Let $f : \mathcal{F}^m \times \{0,1\}^k \to \{0,1\}$ the function such that $f(x_1,\ldots,x_m,a) = p(x_1,\ldots,x_m) \cdot a$, where we abused notation a little by viewing $p(x_1,\ldots,x_m)$ as an element of $\{0,1\}^k$. With similar abuse, we can view f as a function $f : \{0,1\}^{3n+k} \to \{0,1\}$; we define the language L' such that f is the characteristic function of L' on inputs of length $3n + 3\lceil \log n \rceil$. Clearly, L' is **EXP**-complete, because for every $x \in B^n$ we have $x \in L$ if and only if $p(h(x)) = 1$, which holds if and only if $(h(x), \mathbf{1}) \in L'$, where $\mathbf{1} = (1,1,\ldots,1)$.

Theorem 2 *Let $p : \mathcal{F}^m \to \mathcal{F}$ be a degree d polynomial, let us identify elements of $\mathcal{F}$ with elements of $\{0,1\}^k$, and let $f : \{0,1\}^{km+k} \to \{0,1\}$ be such that $f(x,a) = p(x) \cdot a$. Suppose we are given oracle access to $g : \{0,1\}^{km+k} \to \{0,1\}$ such that f and g agree on at least a fraction $7/8 + \epsilon$ of the inputs, where $|\mathcal{F}| \geq 1 + \max\{4/\epsilon^2, d/4\epsilon\}$. Then we can compute $p(x)$ for every x in time polynomial in m, $|F|$ and 2^k.*

PROOF: For every x, define $c(x)$ to be the value that maximizes $\mathbf{Pr}_a[c(x) \cdot a = f(x,a)]$, and break ties arbitrarily if the maximum is not unique. Notice that $c(x)$ can be computed in time polynomial in 2^k. For random (x,a), $f(x,a) \neq g(x,a)$ with probability at most $1/8 - \epsilon$, so it follows that

$$\mathbf{Pr}_x[\mathbf{Pr}_a[f(x,a) \neq g(x,a)] \geq 1/4] \leq 1/2 - 4\epsilon$$

So for at least a $1/2 + 4\epsilon$ fraction of the xs, we have $g(x,a) = f(x,a) = p(x) \cdot a$ for more than $3/4$ of the as. For each such x, then, $p(x) = c(x)$. The function $c()$ has then agreement $1/2 + \epsilon$ with $p()$, which means that, by previous results, we can compute $p()$ everywhere using $c()$. □

Note that the above theorem is essentially giving a poly-logarithmic time decoding procedure for the concatenation of the Reed-Muller code with the Hadamard code. The procedure first decodes the Hadamard code locally, and then simulates the procedure for decoding the Reed-Muller code using the results of the Hadamard decoding. The Reed-Muller decoding needs the number of errors to be less than 1/2, and the Hadamard decoding needs the number of errors to be less than 1/4, so the overall procedure needs the number of errors to be less than 1/8.

In order to state the application of the previous theorem to the average-case hardness of problems in **PSPACE** and **EXP**, let us define $avg\mathbf{BPTIME}_{p(n)}(t(n))$ to be the class of decision problems L for which there is a probabilistic algorithm A running in time $t(n)$ such that for every input length n

$$\mathbf{Pr}_{x \in \{0,1\}^n}[\mathbf{Pr}_r[A(x,r) = L_n(x)] \geq 3/4] \geq p(n)$$

In other words, $avg\mathbf{BPP}_{p(n)}(t(n))$ contains all problems that can be solved on at least a $p(n)$ fraction of the inputs by BPP-type algorithms in time $t(n)$. Notice that $avg\mathbf{BPP}_1(t(n)) = \mathbf{BPP}(t(n))$.

Corollary 3 *Suppose that* $\mathbf{EXP} \subseteq avg\mathbf{BPP}_{7/8+1/n}\left(n^{O(1)}\right)$*, then* $\mathbf{EXP} = \mathbf{BPP}$*. Suppose that* $\mathbf{PSPACE} \subseteq avg\mathbf{BPP}_{7/8+1/n}\left(n^{O(1)}\right)$*, then* $\mathbf{PSPACE} = \mathbf{BPP}$*.*

2 Coding-Theoretic Perspective

The construction of the previous section can be abstracted as follows: starting from a language L and an input length n, we encode the characteristic function $L_n : \{0,1\}^n \rightarrow \{0,1\}$ of L on inputs of length n as a function $f_L : \{0,1\}^{O(n)} \rightarrow \{0,1\}$, and then we argue that given a procedure that computes g on a $1-\delta$ fraction of inputs, with $\delta < 1/8$ we can reconstruct L_n everywhere.

The construction can be carried over for every L_n, and we can see that, in order for the theorem to hold, it must be the case that if we have to different functions $L_n, L'_n : \{0,1\}^n \rightarrow \{0,1\}$ then the functions f_L and $f_{L'}$ must differ in at least a 2δ fraction of inputs, otherwise it would not be possible to reconstruct f_L given only a function g that has agreement $1-\delta$ with f_L. In fact, the encoding of L_n into f_L is a concatenation of a Reed-Muller code with a Hadamard code, and its minimum relative distance is almost $1/2$.

Now suppose we are trying to prove that **EXP** contains languages that hard to compute even on a $1/2+\epsilon$ fraction of inputs. By the above argument, we need an encoding that will map two different languages into two boolean functions that differ in at least $1-2\epsilon$ fraction of points. Unfortunately, this is impossible: there is no error-correcting code with relative minimum distance bigger than 1/2 (except for trivial codes containing a small constant number of codewords), which means that the unique reconstruction problem is possible only if there are less than a 1/4 fraction of errors.[1]

In order to prove stronger hardness results, we need to stop looking for unique decoding procedures, and start looking for *list decoding* procedures instead. In the list decoding problem for a code $C : \{0,1\}^n \to \{0,1\}^m$, we are given a received string $z \in \{0,1\}^m$, and we believe that less than a δ fraction of errors occurred, and we want to compute the list of all possible messages $x \in \{0,1\}^n$ such that $C(x)$ and z are at distance less than δ. If the list is small and it can be produced efficiently, it gives a lot of information about the original message.

We will state without proof a list-decoding result about the Hadamard code and one about the Reed-Muller code.

Lemma 4 *Given $g : \{0,1\}^k \to \{0,1\}$ and $\epsilon > 0$, there are at most $1/4\epsilon^2$ strings $a \in \{0,1\}^k$ such that $\mathbf{Pr}_x[g(x) = a \cdot x] \geq 1/2 + \epsilon$.*

For the Hadamard code, the list can be find by brute force in time polynomial in the length of the encoding. Goldreich and Levin show that, given oracle access to f, the list can even be found in time polynomial in k and $1/\epsilon$ (by a randomized algorithm, with high probability), although we will not need such a stronger result in our applications.

Theorem 5 (Sudan) *Let $\mathcal{F}$ be a field, d a degree parameter, and $1 > \alpha > \sqrt{2d/m}$ be an accuracy parameter. Then there is a polynomial time algorithm that on input pairs of field elements $(a_1, b_1), \ldots, (a_m, b_m)$ finds a list of all univariate polynomials $q()$ of degree $\leq d$ such that $q(a_i) = b_i$ for at least αm pairs. Furthermore, the list contains at most $2/\alpha$ polynomials.*

In the next section we will see how to use Sudan's algorithm to achieve a form of list-decoding of Reed-Muller code even given a $1-\epsilon$ fraction of errors. Since we can do list-decoding of the Hadamard code with up to $1/2-\epsilon$ errors, it should be intuitive that the combination of the two procedures will give a decoding for the concatenation of the two codes that will work with $1/2 - epsilon'$ fraction of errors. Using non-uniformity, we will be able pick the right decoding from the list of possible decodings by hard-wiring the choice in the circuit performing the reconstruction. This will show that if every problem in **EXP** can be solved by small circuits on $1/2+\epsilon$ fraction of inputs, then all problems in **EXP** can be solved by small circuits on all inputs.

[1] Note that there is still some margin of improvement, because our reconstruction procedure works only with less than a 1/8 fraction of errors.

3 Average-case Complexity in PSPACE and EXP for 1/2 Fraction of Inputs

Let us recall a (weaker version of a) result from a past lecture. It was stated in terms of probabilistic procedures, but it holds for a stronger reason in terms of small circuits. (Because we have seen that probabilistic procedures can always be simulated by circuits.)

Theorem 6 *Let $p : \mathcal{F}^m \to \mathcal{F}$ be a polynomial of degree d, $|\mathcal{F}| > 16d$, and let $f : \mathcal{F}^m \to \mathcal{F}$ be a function that agrees with p on a $7/8$ fraction of inputs. Then there is an oracle circuit C of size polynomial in m and $|\mathcal{F}|$ such that $C^f(x) = p(x)$ for every x.*

Here is the multivariate list-decoding result. We state it as a non-uniform unique reconstruction procedure.

Theorem 7 *Let $p : \mathcal{F}^m \to \mathcal{F}$ be a polynomial of degree d, $|\mathcal{F}| \geq 1 + 8d/\epsilon^2$, and let $f : \mathcal{F}^m \to \mathcal{F}$ be a function that agrees with p on at least an ϵ fraction of inputs, with $|\mathcal{F}| \geq 1 + 8d/\epsilon^2$. Then there is an oracle circuit C of size polynomial in m and $|\mathcal{F}|$ such that $C^f(x) = p(x)$ for every x.*

Proof: We will just show the existence of a polynomial size circuit that computes $p()$ on a $7/8$ fraction of inputs, and then invoke Theorem 6.

Let us pick x and y at random in $\mathcal{F}^m$, we want to argue that knowing $p(y)$ will help us compute $p(x)$. The following procedure does that.

input: x, y, $p(y)$ consider line $l(t) = tx + (1-t)y$ and find set Q of all degree-d univariate polynomials q such that $\lvert\{t \in \mathcal{F} : q(t) = f(tx + (1-t)y)\}\rvert > \epsilon\lvert\mathcal{F}\rvert/2$ if there is unique $q^* \in Q$ such that $q^*(0) = p(y)$ return $q^*(1)$ otherwise fail

By our assumption on $\mathcal{F}$, d, and ϵ, we have $\epsilon/2 > \sqrt{2d/|\mathcal{F}|}$, so the above algorithm can be implemented by a circuit C of size polynomial in m and $|\mathcal{F}|$. Now we want to argue that the algorithm is quite effective.

Claim 8 $\mathbf{Pr}_{x,y}[C^f(x, y, p(y)) = p(x)] \geq 7/8$.

Before proving the claim, let us see that, once the claim is proved, we are done. By an averaging argument, there is a y such that

$$\mathbf{Pr}_x[C^f(x, y, p(y)) = p(x)] \geq 7/8.$$

Then we just have to hard-wire y and $p(y)$ into C, and we have a circuit that computes $p()$ on a $7/8$ fraction of inputs, and we can invoke Theorem 6 and we are done. Now we prove the claim.

Proof: [Of Claim] First of all, note that when we pick x and y at random, the points on the line $l(t) = tx + (1-t)y$ are pairwise independent. Therefore, there is a probability at least 15/16 then there are at least $|\mathcal{F}|\epsilon - 4\sqrt{|\mathcal{F}|}$ points on the line where f agrees with p, and this number is at least $\epsilon/2$ by our assumptions on $\mathcal{F}$, d and ϵ.

So, except with probability 1/16, the polynomial q^* such that $q^*(t) = p(l(t))$ is part of the list Q constructed by the algorithm. It remains to see that it is the unique one such that $q(0) = p(y)$. Notice that each other polynomial $q \in Q$ agrees with q^* in at most d points of the line, so, on a random point of the line, q^* has a probability at least $1-(|Q|-1)d/|\mathcal{F}|$ of having a value different from the one of all other polynomials in Q on that random point. Well, y was a random point, but it is not exactly a random point on $l()$: by definition, indeed, $y = l(0)$. Can we still argue that y is as good as a random point of the line in distinguishing $q^*()$ from the other polynomials in Q?

To see that this is the case, let us pick the points x, y in a different (but equivalent) way: let us first pick two random points $x', y' \in \mathcal{F}^m$, then consider the line $l'(t) = tx' + (1-t)y'$, and finally pick two random points $a, b \in \mathcal{F}$ on the line and call $x = l'(a)$ and $y = l'(b)$. Surely, x and y picked in this way are uniformly distributed. What about the line $l(t) = tx + (1-t)y$? It is the same line as $l'()$, just with a different parameterization: we have $l(t) = l'(t(a-b)+b)$. So for each polynomial $q \in Q$ such that $q(t)$ has agreement $\epsilon/2$ with $p(l(t))$ there is a corresponding polynomial $q'(t) = q(t(a-b)+b))$ such that $q'(t)$ has agreement $\epsilon/2$ with $p(l'(t))$ and we can call Q' the list of such polynomials. Let $q'^*(t) = q^*(t(a-b)+b))$, then we have that q^* is the unique polynomial in Q such that $q(0) = p(y)$ if and only if q'^* is the unique polynomial in Q' such that $q'^*(b) = p(y)$. In the latter case, it is clear that we are looking at random point, and so the probability that q^* is isolated is at least $1-(|Q|-1)d/|\mathcal{F}|$ which is at least 15/16.

Overall, the success probability of the algorithm is at least 7/8 and the claim is proved. $\square$

$\square$

We can now move to the concatenated version.

Theorem 9 *Let $p : \mathcal{F}^m \to \mathcal{F}$ be a degree d polynomial, let us identify elements of $\mathcal{F}$ with elements of $\{0,1\}^k$, and let $f : \{0,1\}^{km+k} \to \{0,1\}$ be such that $f(x,a) = p(x) \cdot a$. Suppose we are given oracle access to $g : \{0,1\}^{km+k} \to \{0,1\}$ such that f and g agree on at least a fraction $1/2 + \epsilon$ of the inputs, where $|\mathcal{F}| \geq 8d/\epsilon^6$. Then there is an oracle circuit C of size polynomial in m, $|F|$ and 2^k such that C^g computes $p()$ everywhere.*

PROOF: It will be enough to show how to compute p on a fraction ϵ^3 of the inputs.

For every x, let $S(x) \subseteq \{0,1\}^k$ be the list of all strings c such that $g(x,a)$ and $c\dot{a}$ agree on a fraction at least $1/2 + \epsilon/2$ of the values $a \in \{0,1\}^k$. Notice that $S(x)$ is computable in time polynomial in 2^k and that it contains at most $1/\epsilon^2$ elements. A Markov argument shows that

$$\mathbf{Pr}_x[\mathbf{Pr}_a[g(x,a) = p(x) \cdot a] \geq 1/2 + \epsilon/2] \geq \epsilon$$

Now, for each x such that $\mathbf{Pr}_a[g(x,a) = p(x) \cdot a] \geq 1/2 + \epsilon/2$ we have $p(x) \in S(x)$ by definition. Let $c_i(x)$, for $i = 1, \ldots, 1/\epsilon^2$ be the i-th element of $S(x)$ (say, in lexicographic order, and if $S(x)$ has less than i elements let $c_i(x)$ be $\mathbf{0}$). Again, we have that $c_i(x)$ is computable in polynomial in 2^k and, by an averaging argument, there must be an i^* such that for at least an ϵ^3 fraction of the x we have $c_{i^*}(x) = p(x)$. We then let $C^f(x) = c_{i^*}(x)$, and apply Theorem 7. $\square$

At this point, we are ready to state our main result, except that we need some more notation. Let us define $avg\mathbf{SIZE}_{p(n)}(t(n))$ to be the class of decision problems L such that for every input length n there is a circit C of size $t(n)$ such that

$$\mathbf{Pr}_{x\in\{0,1\}^n}[C(x) = L_n(x)] \geq p(n)$$

Corollary 10 *Suppose that* $\mathbf{EXP} \subseteq avg\mathbf{SIZE}_{1/2+1/n}\left(n^{O(1)}\right)$, *then* $\mathbf{EXP} \subseteq \mathbf{SIZE}(n^{O(1)})$. *Suppose that* $\mathbf{PSPACE} \subseteq avg\mathbf{SIZE}_{1/2+1/n}\left(n^{O(1)}\right)$, *then* $\mathbf{PSPACE} \subseteq \mathbf{SIZE}(n^{O(1)})$.

Exercises

- Let $C : \{0,1\}^n \to \{0,1\}^m$ be an error-correcting code of minimum relative distance 3/4. Show that $n = 1$.
- Show that $avg\mathbf{BPP}_{1-1/n}(n^2)$ contains undecidable languages.
- This may be somewhat complicated: try to get a version of Theorem 7 that is more explicitely in terms of a list-decoding algorithm. The proof already shows that one can (withouth non-uniformity) define a list of $|\mathcal{F}|^{m+1}$ decoding algorithms (one for each choice of y and $p(y)$) such that one of them is correct.
  - Show that (with a small loss in the parameters) y can be chosen at random, so that we only need to enumerate over the $|\mathcal{F}|$ possible values for $p(y)$.
  - Can you modify the algorithm so that the list contains only $poly(1/\epsilon)$ elements instead of $|\mathcal{F}|$?.