6.841 Advanced Complexity Theory March 28, 2005

Lecture 14

Lecturer: Madhu Sudan *Scribe: Alissa Reyzin*

1 Today

1.1 Announcements

- Have you scribed twice? Please sign up to scribe if you haven't.
- Pick up graded problem set 2 from Sergey

1.2 Toda's Theorem

- Recall operators $\exists$, $\forall$, $\bigoplus$, BP
- Operator Calculus $PH \subseteq BP\cdot \bigoplus \cdot P$
- Arithmetic $BP\cdot \bigoplus \cdot P \subseteq P^{\#P}$

Although Toda's Theorem is very powerful, we still do not know whether #P is different from PSPACE. We also do not know whether it is different from LOGSPACE.

2 Applying Operators to Complexity Classes

Given a complexity class C and an operator O we can construct a new complexity class $O\cdot C$. For example, we have already seen that:

- $\exists \cdot P = NP$
- $\exists \cdot co\text{-}NP = \Sigma_2^P$
- $\forall \cdot \Sigma_2^P = \Pi_3^P$

Remember that the operator $\bigoplus$ gives us an exact count on the least significant digit and the operator BP gives an approximate count on the total. The precise definitions of the operators are as follows.

- A language $L \in \exists \cdot C$ if there exists a machine M such that $L(M) \in C$ and $L = \{x | \exists y$ such that $M(x,y) = 1\}$.

  Keep in mind that the running time complexity for all classes created by operator applications must be measured as a function of the first input, since the later inputs can be extremely large.

- A language $L \in \bigoplus \cdot C$ if there exists a machine M such that $L(M) \in C$ and $L = \{x|$ the number of y's such that $M(x,y) = 1$ is odd $\}$.

  Consider what happens if we replace the word "odd" with "even". This gives us the class $co\bigoplus \cdot P$. We can easily see that this class is the same as $\bigoplus \cdot P$. Consider $M'(x, by) =$ accept if $by = \bar{0}$ or if $b = 1$ and $M(x,y)$ accepts. This has exactly one more accept than $M(x,y)$.

- We use the strong definition of BP when defining the operator. The distinction is important, since we are not guaranteed to be able to amplify when the operator is attached to a given complexity class.

  A language $L \in BP\cdot P$ if $\forall$ polynomials q there exists a machine M such that $L(M) \in C$ and $\Pr_y[M(x,y) = \text{“}x \in L''] \geq 1 - \frac{1}{2^{q|x|}}$

3 Proof of $PH \subseteq BP\cdot \oplus\cdot P$

To show that $PH \subseteq BP\cdot \bigoplus\cdot P$ we need the following steps.

$$
\begin{aligned}
\text{(we can also show this for the } \forall \text{ operator) } \exists\cdot BP\cdot \oplus\cdot P &\subseteq BP\cdot \oplus\cdot BP\cdot \oplus\cdot P && (1)\\
&\subseteq BP\cdot BP\cdot \oplus\cdot \oplus\cdot P && (2)\\
&\subseteq BP\cdot \oplus\cdot \oplus\cdot P && (3)\\
&\subseteq BP\cdot \oplus\cdot P && (4)
\end{aligned}
$$

After this has been shown, we can use induction to demonstrate that any constant length sequence of $\forall$ and $\exists$ quantifiers can be absorbed into the class $BP\cdot \bigoplus\cdot P$.

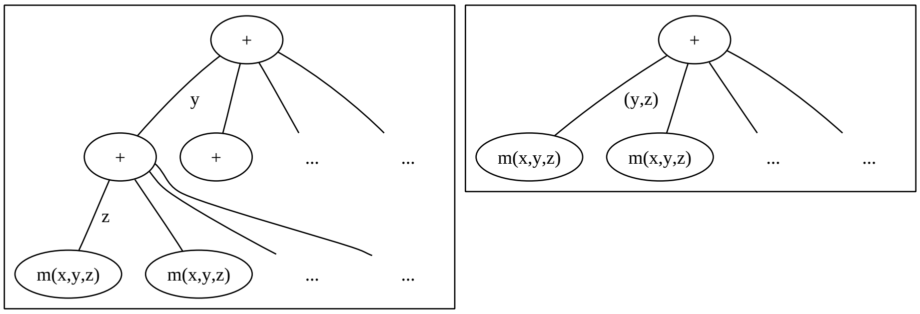
3.1 Step 4

We need to show that $\bigoplus\cdot \bigoplus\cdot C \subseteq \bigoplus\cdot C$

Let $L(M) \in C$

Let $L_1 = \{(x,y)|$the number of z such that $M((x,y),z)$ accepts is odd $\}$. Then $L_1 \in \bigoplus\cdot C$.

Let $L_2 = \{x|$ the number of y such that $(x,y) \in L_1$ is odd $\}$



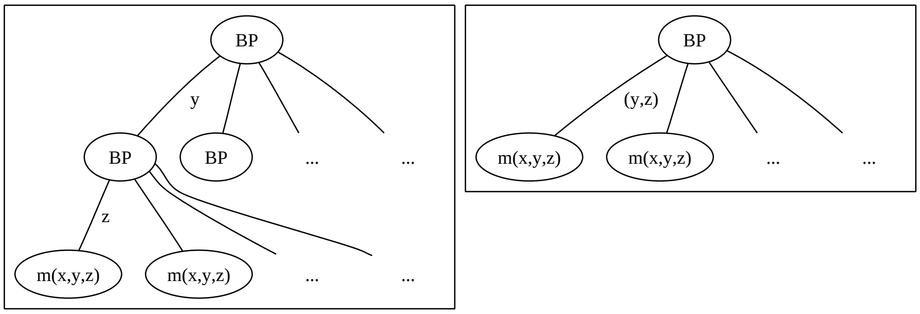
We can draw the circuits using parity operators for the above languages. We see that all we need to show is that $\sum_y \sum_z M(x,y,z) = \sum_{(y,z)} M(x,y,z)$. But this is obvious.

The only small caveat is that we need to have associativity. That is $M((x,y)z) = M(x,(y,z))$. We also need to have available linear time in order to reparenthasize.

3.2 Step 3

We need to show that $BP\cdot BP\cdot P \subseteq BP\cdot P$.

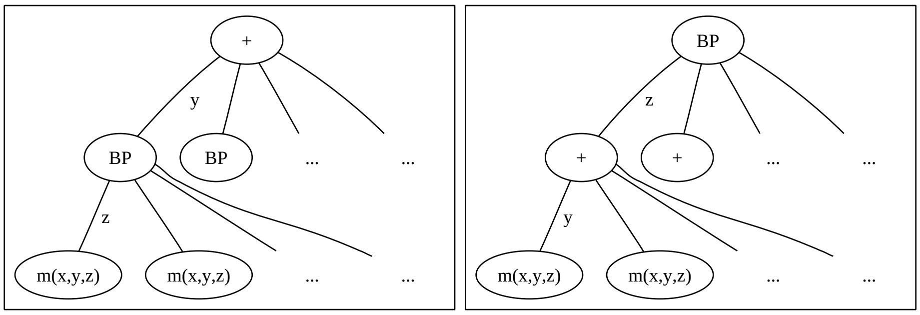
The main idea is that instead of calculating approximate majority first over all the y's and then over all the z's we do it over all pairs (y, z) which will at most sum the two error probabilities.



That is, we draw a circuit with BP gates. Assume that the error we want in our final circuit is $2^{-q(n)}$. Then we simply pick our BP gates in the original circuit to have error $2^{-q(n)+1}$. At worst, the probabilities of error will be added together, which gives us the desired probability.

3.3 Step 2

We need to show that $\bigoplus \cdot BP \cdot C \subseteq BP \cdot \bigoplus \cdot C$.



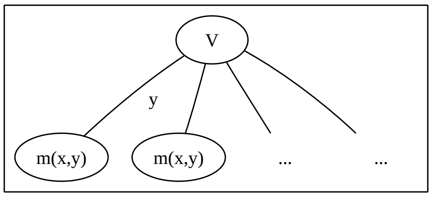
Let us call the fan out of the parity gate 2^l and the fanout of the BP gate 2^m. Suppose for a moment that the BP gate made no errors. In that case, clearly, the switch is a valid one. Every error made can give the wrong output of the parity gate. How many errors do we have? $2^l * 2^m * 2^- q = 2^{l+m-q}$.

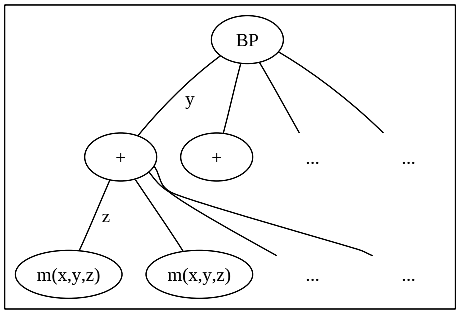
Now consider the number of parity gates computing the wrong value after the switch. This is also at most 2^{l+m-q}. Therefore, the probability of error is $\frac{2^{l+m-q}}{2^m} \leq 2^{l-q}$. So, if we want $q'(n)$ to be the probability of error, we simply choose $q(n) = q'(n) + l$.

Note that this does not work in the reverse direction, since each parity gate may have errors.

3.4 Step 1

We need to show that $\exists \cdot C \subseteq BP \cdot \bigoplus \cdot P$ where $C = BP \cdot \bigoplus P$. (This does not hold true for all complexity classes, since we need to be able to use amplification.)





Where have we seen a similar picture before? This looks a great deal like the Razborov-Smolensky proof. Unfortunately, picking a random subset of an exponential number of things costs us too much time. It is possible to use pseudo-randomness instead of true randomness, but this requires techniques that we haven't seen. Instead, we will try to use Valiant-Vazirani (Unique SAT in Parity SAT).

The problem with using Valiant-Vazirani is that the error is too high. On the other hand, the error is one sided, which makes amplification easier.

Valiant-Vazirani gives us:

- ϕ satisfiable $\rightarrow \phi'$ odd with probability $1/n$
- ϕ unsatisfiable $\rightarrow \phi'$ even with probability 1.

What we want is:

- ϕ satisfiable $\rightarrow \phi'$ odd with probability $1 - 2^{-n}$
- ϕ unsatisfiable $\rightarrow \phi'$ even with probability $1 - 2^{-n}$.

Let's complement everywhere. That is, let's have a satisfiable ϕ go to an even ϕ' and an unsatisfiable ϕ go to an odd ϕ'. Then, we can calculate $\phi'_1, \phi'_2, \phi'_3$... which will be completely independent variables. Since #SAT = product of #SAT, we know that if all of the ϕ'_i are odd, the product will be odd, otherwise it will be even. This gives us exactly the amplification we were looking for.

Since all that we used to do the amplification was complementation and polynomial time manipulations, this shows that $\exists$ can be replaced as stated above.

4 Proof of $BP\cdot \oplus\cdot P \subseteq P^{\#P}$

We need to show that $BP\cdot \oplus\cdot P \subseteq P^{\#P}$

$\bigoplus\cdot P \subseteq \{(\phi, t, N)|$ the number of accepting configurations$\in \{0,1\}(\text{mod} N)\}$. We accept if the number of accepting configurations is 0 mod 2 and reject if it is 1 mod 2. What happens if instead of 2 we use 2^N where N is large? If x is in the language L, then the number of accepting configurations is $\leq 2^{m-q} \pmod{2}^N$. If x is not in L then the number of accepting configurations is between $2^m - 2^{m-q}$ and $2^m \pmod{2}^N$.

Unfortunately, we can't actually do this. However, we can accept if the number of accepting configurations is 0 mod 2 and reject if it is -1 mod 2. This will leave the intervals disjoint, which is sufficient for us.

4.1 Arithmetic with Counting Functions

$f : \Sigma^* \to \mathbb{Z}$ is a counting function if $\exists M \in P$ such that $f(x) = \#$ y's such that $M(x, y) = 1$.

We show that if f_1 and f_2 are counting functions, then so are $f_1 \cdot f_2$ and $f_1 + f_2$. This is easy to see. To get a TM that that accepts $f_1 \cdot f_2$ things given a TM M_1 that accepts f_1 things and a TM M_2 that accepts f_2 things we simply put the two "in series". That is, for branch of M_1 that accepts, we then run the the machine M_2 on x. To get a TM that accepts $f_1 + f_2$ things, we flip a coin. depending on the result of the coin flip, we either run M_1 or M_2.

$h : \mathbb{Z} \to \mathbb{Z}$ is a polynomial with non-negative coefficients. If f is a counting function, clearly so is $h(f)$ by the above. We come up with a clever polynomial $h = 3f^4 + 4f^3$. If f has 0 accepting configurations modulo 2^k then $h(f)$ will have 0 accepting configurations modulo 2^{2k}. Moreover, if f has 1 accepting configuration modulo 2^k then h will have -1 accepting configurations modulo 2^{2k}.

We then need to apply the process log n times in order to actually get something with the correct properties modulo 2^N (which we can actually do in the correct time bound).

Reference the slides to see how to come up with the polynomial $h(f)$ that works.