Notes taken on April 18th and 20th

1 P/poly and the Polynomial Hierarchy.

Definition. $P/Poly = \{A | \exists \text{poly-sized circuit family} \{C_n | n \in naturals\}\, x \in C_n \Leftrightarrow |x| = n \vee x \in A\}$ For example, the Halting problem: $\{0^n | n \in K\} \in P/poly$

Theorem. *If* $NP \subseteq P/Poly \Rightarrow PH = \sum_2^P$

Proof. To prove this, we will construct a $\sum_2^P$ algorithm for a $\text{CO-NP}^{\text{SAT}}$ machine, using a polysized circuit for the $\overline{\text{SAT}}$ oracle.

The algorithm is as follows:

On input x
$\exists$ guess a small circuit C
$\forall$ simulate $\text{CO-NP}^{\text{SAT}}$ using the circuit C instead of the oracle.
$\forall$ check for all x where $|x| < n^k\ c(x) = 1 \Leftrightarrow x \in \text{SAT}$

To perform this last universal step, we need to check if the circuit is consistent with SAT. And we need to check this over all of the possible inputs for the circuit. This is when we use the self-reducible property of SAT.

The algorithm for the last step is then:

for all x **do**
  if x has no variables **then**
    evaluate x and check that $c(x)$ is correct
  else
    $v \leftarrow$ first variable
    $x_0 \leftarrow x\{v \mapsto 0\}$
    $x_1 \leftarrow x\{v \mapsto 1\}$
    verify that $c(x) = c(x_1) \vee c(x_0)$
  end if
end for

With this algorithm in place, we have shown that $\prod_2^P \subseteq \sum_2^P$. In previous notes, we have demonstrated that if $\sum_2^P \subseteq \prod_2^P$, then the polynomial hierarchy collapses, and from these two we get that $\prod_2^P = \sum_2^P$, which shows that with these premises, the polynomial hierarchy collapses. ☐

2 Interactive Proof System

The best introduction to the interactive proof system is to start with an example:

For this example, we'll use the graph isomorphism problem. In this problem, we are given two graphs and we want to decide if one is an isomorphism of the other.

Definition. $\text{GI} = \{(G, H) | G \equiv H\}$

In this case, it should be obvious that GI $\in$ NP.

What we will show is that $\overline{\text{GI}}$ has an interactive proof.

We will use an Arthur-Merlin proof for our interactive proof. In this example, Arthur has two graphs that he wants to answer the $\overline{\text{GI}}$ problem for. Merlin knows the correct answer, but will always attempt to convince Arthur that the two graphs are not isomorphic to each other.

For this first example, we will use the concept of "private coins": Arthur will flip a coin n times, and only he will know what the coins decided.

For each coin flip, Arthur will create an isomorphism of either G or H, depending on the coin flip. Then Arthur will give all of these permuted graphs, along with the original graphs to Merlin. Let these be denoted by K_i. If G and H are not isomorphisms of each other, then the all-knowing Merlin will be able for each K_i to determine if Arthur made the given K_i from either G or H. If they are isomorphic to each other, then Merlin can't know which graph was used by Arthur to create the given K_i, unless he can determine the random sequence that was used by Arthur. Any mechanism the Merlin uses, aside from cheating and determining the coin sequence, can do no better than guessing (by definition of "random"), so he'll just guess that they all came from G. Notice that this gives Merlin a $\frac{1}{2^n}$ probability of being correct.

Arthur will get the answers back from Merlin and compare them to his coin flips. In the case that he sees Merlin handed back the same answers as his coin flips, he can assume that the two graphs are not isomorphic to each other. His error margin is the same as the chance that Merlin will guess correctly: $\frac{1}{2^n}$ on an isomorphic pair of graphs, and zero if they are not isomorphic.

2.1 The Interactive Proof System

Now we can define what an interactive proof system is.

Definition. An **Interactive Proof** system with "k-rounds" for a language A consists of a verifier V, which is a probabilistic poly-time TM, and a prover, P. The algorithm for the interactive proof

On input x

for $i = 1$ to k **do**

V flips a random coin sequence r_i and sends r_i to P.

receive message p_i

end for

accept or reject depending on messages received.

An interactive proof also has the following condition:

$x \in A\, \exists P_x\, s.t.\, Prob(V \text{accepts } x \text{ with answers given by } P_x) \geq \frac{3}{4}$

and

$x \notin A \Rightarrow \forall P_x\, Prob(V \text{accepts } x \text{ with answers given by } P) < \frac{1}{4}$

Also, P_x is a function that maps $(x, r_1, p_1, r_2, p_2, ..., r_i) \mapsto p_i$.

2.2 The Optimal Prover

For each verifier, V, and each input x, there is an "optimal prover".

Definition. An **Optimal Prover** will return the message that optimizes the accepting probability.

For example, if Merlin is given two graphs that are isomorphic, he just guesses that all of the isomorphic graphs came from the same input graph. This maximizes the probability of being correct. (In this case, any guess of the coin flip sequence would result in an optimal prover).

2.3 Main Theorems

Theorem. IP=PSPACE - *even with public coins.*

In general, public coins with $k+2$ rounds can simulate private k round protocols.

Theorem. IP$[O(1)]$ =IP[1 *round public coins*]=AM.

Neither of these will be proven here. Instead, we will look at the $\overline{\text{GI}}$ problem with public coins.

2.4 $\overline{\text{GI}}$ with public coins

In order to create and analyze this algorithm, we'll need to use some new concepts:

Definition. A **random linear function** $\{0,1\}^n \to \{0,1\}^k$ is a $k \times n$ $0-1$ matrix M. The output of Mx is in $\mathcal{F}_2$.

We'll also define the following:

Let: $X \subseteq \{0,1\}^n$, where $X \neq \emptyset$

and

Let: M be chosen at random, and define S to be $|\{x \in X | Mx = 0^k\}|$.

Given these two definitions, it can be shown that
$E(S) = \frac{|X|}{2^k}$ and $Var(S) = \frac{|X|}{2^k}\left(1 - \frac{1}{2^k}\right)$
but we will not prove this here.

In essence, Arthur will give this random linear function to Merlin, along with the two graphs, G_1 and G_2. Merlin will try to find 5 isomorphism/automorphism pairs that once encoded, will cause this function to yield zero. If Merlin has a larger field of pairs to pick from, intuitively he would have a better chance of completing his task. That's what we'll show happens with this algorithm.

At this point we can present the public-coin $\overline{\text{GI}}$ protocol.

Let us define $p(n)$ to be enough bits to encode 5 pairs of graph(H) and permutation(ϕ) combinations as $\{H, \phi\}$.

1: Pick M of size $p(n) \times \lceil \log(4(n!)^5) \rceil$ at random
2: Send M to prover.
3: Prover responds with $y = \{(H_1, \phi_1), (H_2, \phi_2), (H_3, \phi_3), (H_4, \phi_4), (H_5, \phi_5)\}$ and 5 isomorphisms showing that each $H_i \equiv G_{1\,or\,2}$
4: Verifier accepts if $My = 0^k$, each isomorphism is correct, and each ϕ_i is an automorphism of H_i.

Let:

$$\begin{aligned} N(G_1, G_2) &= \{(H, \phi) | (H \equiv G_1 \vee H \equiv G_2) \, and \, \phi \in \text{AUTO}(H)\} \\ &= \{(H, \phi) | H \equiv G_1 \wedge \phi \in \text{AUTO}(H)\} \\ &\cup \{(H, \phi) | H \equiv G_2 \wedge \phi \in \text{AUTO}(H)\} \end{aligned}$$

There are $n!$ members in the set of all isomorphisms and auto-morphisms of G. Therefore, if $G_1 \equiv G_2$, then $|N(G_1, G_2)| = n!$, but if they are not isomorphic, then $|N(G_1, G_2)| = 2n!$, since each graph would have its own set of isomorphisms and auto-morphisms that are necessarily independent of the other.

Let: $X = (N(G_1, G_2)^5)$

$$\begin{aligned} G_1 \equiv G_2 &\Rightarrow |X| = (n!)^5 \\ G_1 \not\equiv G_2 &\Rightarrow |X| = 32(n!)^5 \end{aligned}$$

At this point, we will show how likely the prover will be able to convince the verifier in either case. First, in the case where the two graphs are isomorphic:

$$\begin{aligned} G_1 \equiv G_2 \Rightarrow Prob(\exists y \, s.t. My = 0^k) &= Prob(S \geq 1) \\ &\leq E(S) = \frac{|X|}{4(n!)^5} = \frac{n!}{4(n!)^5} = \frac{1}{4} \end{aligned}$$

This is within the error that the verifier can have. Now in the case that the two graphs are not equivalent:

$$\begin{aligned} G_1 \not\equiv G_2 \Rightarrow Prob(\exists y \, s.t. My = 0^k) &= 1 - Prob(S = 0) && (1) \\ &\geq 1 - Prob(|S - E(S)| \geq E(S)) && (2) \\ &\geq 1 - \frac{var(S)}{E(S)^2} && (3) \\ &= 1 - \frac{|X|}{2^k} \frac{(1 - \frac{1}{2^k}) 2^{2k}}{|X|^2} && (4) \\ &= 1 - \frac{1 - \frac{1}{2^k}}{8} \geq \frac{3}{4} && (5) \end{aligned}$$

Line 3 of this proof relies on the fact that $Prob(|S - E(S)| \geq t) \leq \frac{1}{t^2} Var(S)$. This is the well known Chebyshev's inequality, and will not be shown here.

And this completes the algorithm for the public coin $\overline{\text{GI}}$ interactive proof.