What is an oracle for a language A?
- a device that can determine whether or not any string w is a member of A.
- imagine the TM has a special “oracle tape”, writes a string on it and then gets answer in one step.

M^A = a TM with an oracle for A

P^A = languages decidable by a polynomial time TM with an oracle for A
NP^A = languages decidable by a polynomial time NTM with an oracle for A

Claim:
$NP \subseteq P^{SAT}$
Why? Since SAT is NP-complete, any problem in NP can be reduced to SAT in polynomial time. Suppose B is a language in NP. We create a machine M in P^{SAT} that is programmed as follows:
1) Input instance w of B
2) Perform reduction to a boolean formula F
3) Call SAT oracle on F
4) return accept or reject depending upon whether F is SAT or not

$coNP \subseteq P^{SAT}$
Why? Because P^{SAT} is a deterministic complexity class, it is closed under complement (can swap accept/reject states for any TM in it).

Using oracles, we can construct even larger (harder) problem classes, such as NP^{SAT}.

Alternating Machines

What are these?

TM where each state (except q_accept and q_reject) are either Universal or Existential states. When run on a input w, consider its ND computation tree.

A node is accepting if it is a AND and all of its children are accepting,
or OR and at least one of its children is accepting.

String is accepted if root in tree is accepting.

ATIME(t(n)) = {L | L is decided by an O(t(n)) time alternating TM }

AP = $\bigcup_k ATIME(n^k)$ = alternating polynomial time

AP is very powerful! Includes:
SAT - why?
UNSAT - why?
TAUTOLOGY? - why? (almost same as UNSAT)

So, $NP \subseteq AP$ as well as $co-NP \subseteq AP$

The similarity between the AND and OR nodes in an alternating machine and the quantifiers in QBF would lead you to think that AP is closely connected to PSPACE. In fact:

$ATIME(f(n)) \subseteq SPACE(f(n)) \subseteq ATIME(f^2(n))$

Thus $AP = PSPACE$

Thursday 26 April

Recall:

ATIME(t(n)) = {L | L is decided by an O(t(n)) time alternating TM }

$AP = \bigcup_k ATIME(n^k)$ = alternating polynomial time

So, $NP \subseteq AP$
$co - NP \subseteq AP$
$ATIME(f(n)) \subseteq SPACE(f(n)) \subseteq ATIME(f^2(n))$
AP $=$ PSPACE

Alternating machines are so powerful that we wonder if we can "dial back" the power. We can do this by **constraining the pattern** of alternations.

A Σ_i - alternating TM is an alternating TM that contains at most i runs of universal or existential steps, starting with existential steps.

$\Sigma_i TIME(f(n))$ = class of languages decided by a Σ_i - alternating TM in f(n) time

$\Sigma_i P = \bigcup_k \Sigma_i TIME(n^k)$ = languages accepted in polynomial time by Σ_i - alternating TM

Polynomial Hierarchy PH = union over all alternations

PH = $\bigcup_i \Sigma_i P$

What is $\Sigma_1 P$ equivalent to? $\Sigma_1 P$ = NP

We can also define classes of alternating machines where we begin with universal (rather than existential) quantifiers.

A Π_i - alternating TM is an alternating TM that contains at most i runs of universal or existential steps, starting with universal steps.

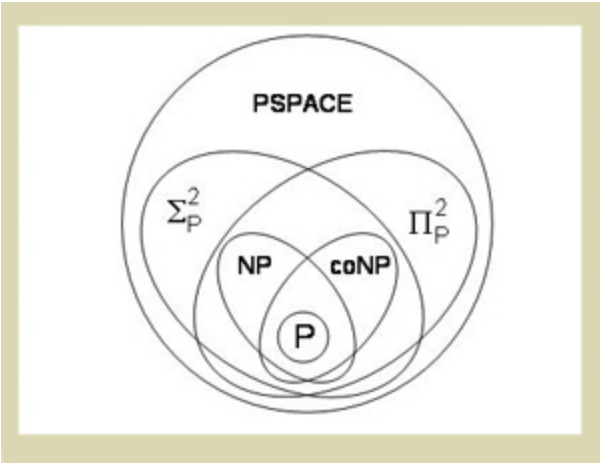
$\Pi_i P = \bigcup_k \Pi_i TIME(n^k)$ = languages accepted in polynomial time by Π_i -alternating TM

What is $\Pi_1 P$ equal to? co-NP

What is $\bigcup_i \Pi_i \ P$ equal to? AP = PSPACE

Call someone to board to draw Venn diagram of P, $\Sigma_1 P$, $\Pi_1 P$, PSPACE

Now: discuss where $\Sigma_2 P$ would fit. Note:
Must contain $\Sigma_1 P$.
Also, must contain $\Pi_1 P$ - why?
Now, add $\Pi_1 P$



note: move upper subscript to bottom.

It is handy to define a new class, that is simply the union of the sigma and pi classes at the same level of alternations:

We can also define polynomial hierarchy using oracles!

Recall:

P^A = languages decidable by a polynomial time TM with an oracle for A
NP^A = languages decidable by a polynomial time NTM with an oracle for A

Suppose A is a "complete" problem for some *class* of problems C. Then, instead of writing P^A we can write P^C .
For example, instead of P^{SAT} we will write P^{NP}.
Similarly, instead of NP^{SAT} we will write NP^{NP}.

Now, we're going to recursively define a series of oracles.

$$\Delta_0^{\mathrm{P}} := \Sigma_0^{\mathrm{P}} := \Pi_0^{\mathrm{P}} := \mathrm{P},$$

Then for i ≥ 0 define

$$\Delta_{i+1}^{\mathrm{P}} := \mathrm{P}^{\Sigma_i^{\mathrm{P}}}$$
$$\Sigma_{i+1}^{\mathrm{P}} := \mathrm{NP}^{\Sigma_i^{\mathrm{P}}}$$
$$\Pi_{i+1}^{\mathrm{P}} := \mathrm{coNP}^{\Sigma_i^{\mathrm{P}}}$$

Claim: these definitions for Sigma and Pi yield the same classes as the definitions involving alternating Turing Machines. For example,

$\Sigma_1 P = \bigcup_k \Sigma_1 TIME(n^k)$ = problems solved in polynomial time by an alternating turing machine whose pattern of alternations is $\exists\exists\exists$ (only existentials).

The new definitions tell us that

$\Sigma_1 P = NP^{\Sigma_0^P} = NP^P = NP$

Let's go up a level.

$\Sigma_2 P = \bigcup_k \Sigma_2 TIME(n^k)$ = problems solved in polynomial time by an alternating turing machine whose pattern of alternation is $\exists\exists\exists\forall\forall\forall$.

The new definitions tell us that

$\Sigma_2 P = NP^{\Sigma_1^P} = NP^{NP}$

Why does this make sense? Intuitively, the "base" NP means that we have a non-deterministic TM that can make a series of lucky guesses -- this is the sequence of existential quantifiers. The oracle for an NP-complete problem (e.g. SAT) means that the TM can decide (both positively and negatively) that problem in unit time - for example, it can determine that a formula is **not** satisfiable in unit time. The ability to use the oracle to determine "reject" answers give the TM the power of the sequence of universal quantifiers.

What about this new series of classes, Delta? We see

$\Delta_1 P = P^{\Sigma_0^P} = P^P = P$

not a very interesting class. But see what happens when we go up a level:

$\Delta_2 P = P^{\Sigma_1^P} = P^{NP}$

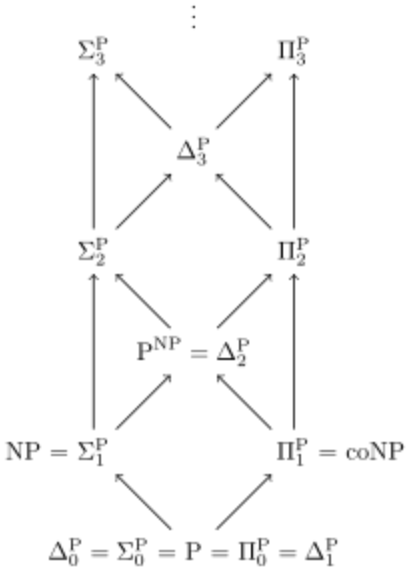
This is something new. Because it has an NP oracle, it must contain both NP and co-NP:

$NP \subseteq P^{NP}$

$co-NP \subseteq P^{NP}$

However, it is not as powerful as $\Sigma_2 P = NP^{NP}$.

So let's summarize the containment relations:



Suppose we take the union over all of these classes for all i. This gives us the Polynomial Hiearchy, PH.

Is PH the same as PSPACE? We don't know, unlike for the case of AP, where we know that AP = PSPACE. There might be problems that are harder than any in PH and yet still be in PSPACE and AP.

Most computer scientist believe all of these levels are distinct. If P=NP, then, of course, everything just collapses into P. But, as far as we know, the hierarchy might only partially collapse, above a certain level.

If particular, if $\Sigma_k^{\mathrm{P}} = \Pi_k^{\mathrm{P}}$, then the hierarchy *collapses to level k*.

Homework: this will be posted by tomorrow (Friday) afternoon and will be due in class (physical paper) in class Tuesday. It will count the same as a quiz. In it, you will play with some problems that fall at different points in the polynomial hierarchy.