$$L \subseteq NL \subseteq P \subseteq NP \cap coNP \subseteq \{NP, coNP\} \subseteq \Delta_2 \subseteq \{\Sigma_2, \Pi_2\} \subseteq PH \subseteq PSPACE = IP \subseteq EXP$$

$1^η$ Σειρά $2^η$ Ασκηση

In order to show that $2^{\mathcal{O}(n)} = 3^{\mathcal{O}(n)}$, we must show that for any function $h(n)$, $[h(n) \in 2^{\mathcal{O}(n)} \Rightarrow h(n) \in 3^{\mathcal{O}(n)}] \wedge [h(n) \in 3^{\mathcal{O}(n)} \Rightarrow h(n) \in 2^{\mathcal{O}(n)}]$. In order to prove that $h(n) \in 2^{\mathcal{O}(n)} \Rightarrow h(n) \in 3^{\mathcal{O}(n)}$, it is sufficient to simply note that $2^x \leq 3^x$ for all $x \geq 0$ (remember that in dealing with $\mathcal{O}(f(n))$-notation, we concern ourselves only with positive values of $f(n)$, since we do not need to consider negative resources).

Now, in order to prove the latter claim, we may note that for $h(n) \in 3^{\mathcal{O}(n)}$ to be true, there must exist some function $g(n)$ in $\mathcal{O}(n)$ such that:

$$\begin{aligned} h(n) &\leq 3^{g(n)} \\ &= 3^{log_3(2) * \frac{g(n)}{log3(2)}} \\ &= 2^{\frac{g(n)}{log_3(2)}} \end{aligned}$$

Since $g(n) \in \mathcal{O}(n)$, then $\frac{g(n)}{log_3(2)} \in \mathcal{O}(n)$ as well. This proves that $h(n) \in 3^{\mathcal{O}(n)} \Rightarrow h(n) \in 2^{\mathcal{O}(n)}$, and that $2^{\mathcal{O}(n)} = 3^{\mathcal{O}(n)}$.

Now, we will show that $2^{\mathcal{O}(n)} = \bigcup_c c^n$.

For any function $h(n) \in 2^{\mathcal{O}(n)}$ there is a number n_0 and a constant $c_1 > 0$ such that the following hold for all $n \geq n_0$:

$$\begin{aligned} h(n) &\leq 2^{c_1 n} \\ &= (2^{c_1})^n \\ &= c_2^n \text{ where } c_2 = 2^{c_1} \\ &\in \bigcup_c c^n \end{aligned}$$

Furthermore, for any function $h(n) \in \bigcup_c c^n$ there is a number n_0 and constants $c_1, c_2 > 1$ such that the following hold for all $n \geq n_0$:

$$\begin{aligned} h(n) &\leq c_1 c_2^n \\ &= 2^{log_2(c_1)}(2^{log_2(c_2)})^n \\ &= 2^{log_2(c_2)n + log_2(c_1)} \\ &= 2^{c_3 n + c_4} \\ &\in 2^{\mathcal{O}(n)} \end{aligned}$$

b.) It is the same as P, the class of (positive) polynomially bounded functions, i.e. functions that are $\mathcal{O}(n^c)$ for some constant c. For any function $h(n) \in 2^{\mathcal{O}(log(n))}$ there is a number n_0 and a constant $c_1 > 0$ such that the following hold for all $n \geq n_0$:

$$\begin{aligned} h(n) &\leq 2^{c_1 log(n)} \\ &= (2^{log(n)})^{c_1} \\ &= n^{c_1} \\ &\in P \end{aligned}$$

Conversely, if $h(n) \leq c_1 n^{c_2}$ for $n \geq n_0$, then $h(n) \leq 2^{c_2 log_2(n) + log_2(c_1)}$, hence $h(n) \in 2^{\mathcal{O}(log(n))}$.

To show that **NP** is closed under union and intersection, we will argue that for arbitrary languages L_1 and L_2 in **NP** decided by nondeterministic Turing Machines (NTM) M_1 and M_2, respectively, we can create a NTM $M_\cup$ that decides $L_1 \cup L_2$ and a NTM $M_\cap$ that decides $L_1 \cap L_2$. We will first detail $M_\cup$, which functions accordingly on input x:

- Simulate x on M_1
- Simulate x on M_2
- If either M_1 or M_2 accepts, $M_\cup$ accepts. Otherwise, $M_\cup$ rejects.

$M_\cap$ runs similarly on input x:

- Simulate x on M_1
- Simulate x on M_2
- If both M_1 and M_2 accept, $M_\cap$ accepts. Otherwise, $M_\cap$ rejects.

Since both M_1 and M_2 run nondeterministically in polynomial time, $M_\cup$ can run nondeterministically in time equal to the sum of two polynomials (for the two simulations) plus linear time (for the overhead), which results in nondeterministic polynomial time. Not surprisingly, $M_\cap$ shares this complexity result. It should also be fairly obvious that the decisions formulated by these machines are the union and intersection of L_1 and L_2.

2η Σειρά 2η Ασκηση

a.) First we note that $\mathbf{TIME}(2^n) \subseteq \mathbf{TIME}(2^{2n})$ is easy to show: let $L \in \mathbf{TIME}(2^n)$, i.e. there is a TM M with $L(M) = L$, and there are constants $c > 0$ and $n_0 \geq 0$ such that $T_M(n) \leq c2^n$ for all $n \geq n_0$. Then $M \in \mathbf{TIME}(2^{2n})$, since $c2^n \leq c2^{2n}$ for all $n \geq 0$.

In class we showed the time-hierarchy theorem:

For any proper function $f(n) \geq n \log n$ and any function $g(n) = o(f(n)/\log n)$, there is a language in $\mathbf{TIME}(f(n)) - \mathbf{TIME}(g(n))$.

Obviously $2^{2n} \geq n \log n$ and $2^n = o(2^{2n}/\log n)$, since

$$\lim_{n\to\infty} \frac{2^n}{2^{2n}/\log n} = \lim_{n\to\infty} \frac{\log n}{2^n} = 0.$$

Therefore $\mathbf{TIME}(2^n) \subsetneq \mathbf{TIME}(2^{2n})$

b.) First note that $\mathbf{NTIME}(n^2) \subseteq \mathbf{SPACE}(n^2)$. By definition $\mathbf{PSPACE} = \bigcup_k \mathbf{SPACE}(n^k)$ and therefore $\mathbf{NTIME}(n^2) \subseteq \mathbf{SPACE}(n^2) \subseteq \mathbf{PSPACE}$.

To show strict containment we use the space-hierarchy theorem:

For any proper function $f(n) \geq \log n$ and any function $g(n) = o(f(n))$, there is a language in $\mathbf{SPACE}(f(n)) - \mathbf{SPACE}(g(n))$.

Of course $n^3 \geq \log n$ and $n^2 = o(n^3)$. Thus $\mathbf{SPACE}(n^3) - \mathbf{SPACE}(n^2) \neq \emptyset$. But since $\mathbf{SPACE}(n^3) \subseteq \mathbf{PSPACE}$ we conclude that $\mathbf{SPACE}(n^2) \subsetneq \mathbf{PSPACE}$.

In summary: $\mathbf{NTIME}(n^2) \subseteq \mathbf{SPACE}(n^2) \subsetneq \mathbf{PSPACE}$.

2η Σειρά 3η Ασκηση

A DNF formula is satisfiable if and only if at least one of its terms is satisfiable. Since each term is a conjunction, each one is satisfiable if and only if there exists no contradiction among its literals – that is to say that for every literal x_i in the term, the literal $\overline{x_i}$ is not present in that term. To verify this we need only iterate through each term and keep a list of the literals in the term. Given a formula of n literals over m variables, this can be accomplished in $\mathcal{O}(nm) \subseteq \mathcal{O}(n^2)$ time, demonstrating that DNF satisfiability is in P.

By definition, DNF-TAUT will be co-NP complete if its complement is NP-complete. The complement of DNF-TAUT is the language of all DNF formulas that have an unsatisfying assignment, which we will refer to as DNF-UNSAT. Clearly, DNF-UNSAT is in NP, because we can guess an assignment and verify in polynomial time that it does not satisfy the formula. For the NP-hardness, we note that a reduction of trivial complexity exists from 3-SAT to DNF-UNSAT: A 3CNF formula ϕ is satisfiable if and only if its inverse $\overline{\phi}$ has an unsatisfying assignment. By de Morgan's law, the inverse of a CNF formula is a DNF formula; specifically, for every clause $C = a \vee b \vee c$ of ϕ (where a, b, c are literals), the inverse $\overline{\phi}$ contains a term $\overline{C} = \overline{a} \wedge \overline{b} \wedge \overline{c}$. Thus, the 3CNF formula ϕ is in 3-SAT if and only if the DNF formula $\overline{\phi}$ is in DNF-UNSAT. Since 3-SAT is NP-complete, it follows that DNF-UNSAT is also NP-complete. Therefore, DNF-TAUT is coNP-complete.

3η Σειρά 2η Ασκηση

$3^η$ Σειρά $3^η$ Ασκηση

Input: Graph G, number k
Question: Does the maximum clique of G have at most k nodes?

Class: coNP

Maximum clique of G has at most k nodes ⇔ there is no clique with more than k nodes

Input: Graph G
Question: Does G contain a node cover with 7 nodes ?

Class: L . Partial credit for NL, P.

Systematically generate every 7-tuple of nodes (reusing the same space of 7logn bits), and check if it is a node cover.

Input: Number N in binary
Question: Is N the product of two primes?

Class: NP∩coNP

We saw in class that factoring is in NP∩coNP

Input: Three numbers a, b, c in binary
Question: Is a = b+c ?

Class: L

A problem in [illegible] 1 showed that addition can be done in log space.

. Input: Boolean formula ψ, number k in binary.
Question: Does ψ have at least k satisfying truth assignments?

Class: PSPACE

Input: Directed graph G
Question: Is G strongly connected, i.e. can every node of G reach every other node?

Class: NL. Partial credit for P.

We know that Reachability between two nodes is in NL. Check Reachability between every pair of nodes, using the same O(logn) space.

1. If A, B are languages in NP, then their difference $A - B = \{x \mid x \in A \text{ and } x \notin B\}$ is also in NP.

Open

3η Σειρά 4η Ασκηση

This is equivalent to NP=coNP.
If NP=coNP, then the following polynomial time NTM accepts A-B: on input x, simulate (nondeterministically) a computation of the poly-time NTM for A and then a computation of the poly-time NTM for the complement of B, and accept if both computations accept, else reject.

Conversely, suppose the claim is true. Take $A=\Sigma^*$ and B=3SAT. Then A - B is the complement of 3SAT, and we know that if this is in NP then NP=coNP.

2. If A, B are languages in NSPACE(n), then their difference A-B is also in NSPACE(n).

True

NSPACE(n) is closed under complementation by the Immerman-Selepzenyi theorem.
The claim follows from this theorem by the argument of part 1.

3. $\mathrm{TIME}(2^n) = \mathrm{TIME}(2^{2n})$

False

We showed in HW2 that $\mathrm{TIME}(2^n)$ is a proper subset of $\mathrm{TIME}(2^{2n})$ because of the Time Hierarchy Theorem.

4. P = SPACE(n)

False

We showed in HW3 that SPACE(n) is not closed under polynomial time reductions, whereas P is obviously closed. It follows that $\mathrm{P} \neq \mathrm{SPACE}(n)$.
Note that neither containment ($\mathrm{P} \subseteq \mathrm{SPACE}(n)$ or $\mathrm{SPACE}(n) \subseteq \mathrm{P}$) is known to be either true or false. However, we know that they cannot both be true.