The Polynomial Hierarchy Three Ways

We discovered a hierarchy of complexity classes between P and PSpace, with NP and coNP on the first level, and infinitely many further levels above:

**Definition by ATM:** Classes $\Sigma^P_i / \Pi^P_i$ are defined by polytime ATMs with bounded types of alternation, starting computation with existential/universal states.

**Definition by Verifier:** Classes $\Sigma^P_i / \Pi^P_i$ are given as projections of certain verifier languages in P, requiring existence/universality of polynomial witnesses.

**Definition by Oracle:** Classes $\Sigma^P_i / \Pi^P_i$ are defined as languages of NP/coNP oracle TMs with $\Sigma^P_{i-1}$ (or, equivalently, $\Pi^P_{i-1}$) oracle.

Using such oracles with deterministic TMs, we can also define classes $\Delta^P_i$.

More Classes in PH

We defined $\Sigma^P_i$ and $\Pi^P_i$ by relativising NP and coNP with oracles.

What happens if we start from P instead?

**Definition 18.1:** $\Delta^P_0 := P$ and $\Delta^P_{k+1} := \Sigma^P_k \cap \Pi^P_k$.

Some immediate observations:

- $\Delta^P_1 = P$
- $\Delta^P_2 = \text{PSPACE}$
- $\Delta^P_2 \subseteq \Sigma^P_1$ (since $P \subseteq \text{NP}$) and $\Delta^P_2 \subseteq \Pi^P_1$ (since $P \subseteq \text{coNP}$)
- $\Sigma^P_1 \subseteq \Delta^P_{k+1}$ and $\Pi^P_1 \subseteq \Delta^P_{k+1}$
Problems for $\Delta^p_k$?

$\Delta^p_k$ seems to be less common in practice, but there are some known complete problems for $P^{NP} = \Delta^p_2$:

**Uniquely Optimal TSP** [Papadimitriou, JACM 1984]
- **Input:** Undirected graph $G$ with edge weights (distances).
- **Problem:** Is there exactly one shortest travelling salesman tour on $G$?

**Divisible TSP** [Krentel, JCSS 1988]
- **Input:** Undirected graph $G$ with edge weights; number $k$.
- **Problem:** Is the shortest travelling salesman tour on $G$ divisible by $k$?

**Odd Final SAT** [Krentel, JCSS 1988]
- **Input:** Propositional formula $\phi$ with $n$ variables.
- **Problem:** Is $X_n$ true in the lexicographically last assignment satisfying $\phi$?

---

What We Know (Excerpt)

**Theorem 18.2:** If there is any $k$ such that $\Sigma^p_k = \Sigma^p_{k+1}$, then $\Sigma^p_j = \Pi^p_j = \Sigma^p_k$ for all $j > k$, and therefore $PH = \Sigma^p_k$. In this case, we say that the polynomial hierarchy collapses at level $k$.

**Proof:** Left as exercise (not too hard to get from definitions). □

**Corollary 18.3:** If $PH \neq P$ then $NP \neq P$.

Intuitively speaking: “The polynomial hierarchy is built upon the assumption that NP has some additional power over P. If this is not the case, the whole hierarchy collapses.”

---

Is the Polynomial Hierarchy Real?

**Questions:**

Are all of these classes really distinct?
Nobody knows.

Are any of these classes really distinct?
Nobody knows.

Are any of these classes distinct from $P$?
Nobody knows.

Are any of these classes distinct from $PSPACE$?
Nobody knows.

What do we know then?
What We Believe (Excerpt)

“Most experts” think that:
- The polynomial hierarchy does not collapse completely (same as $P \neq NP$)
- The polynomial hierarchy does not collapse on any level (in particular $PH \neq PSpace$ and there is no $PH$-complete problem)

But there can always be surprises . . .

Question 1: The Logarithmic Hierarchy

The Polynomial Hierarchy is based on polynomially time-bounded TMs

It would also be interesting to study the Logarithmic Hierarchy obtained by considering logarithmically space-bounded TMs instead, wouldn't it?

In detail, we can define:
- $\Sigma^L_0 = \Pi^L_0 = L$
- $\Sigma^L_{i+1} = NL^{\Sigma^L_i}$ alternatively: languages of log-space bounded $\Sigma_{i+1}$ ATMs
- $\Pi^L_{i+1} = coNL^{\Sigma^L_i}$ alternatively: languages of log-space bounded $\Pi_{i+1}$ ATMs

Therefore $\Sigma^L_i = \Pi^L_i = NL$ for all $i \geq 1$.

Historic note: In 1987, just before the Immerman-Szelepcsenyi Theorem was published, Klaus-Jörn Lange, Birgit Jenner, and Bernd Kirsig showed that the Logarithmic Hierarchy collapses on the second level [ICALP 1987].

Q1: What is the Logarithmic Hierarchy?

How do the levels of this hierarchy look?
- $\Sigma^L_0 = \Pi^L_0 = L$
- $\Sigma^L_1 = NL^L = NL$
- $\Pi^L_1 = coNL^L = coNL = NL$ (why?)
- $\Sigma^L_2 = NL^{\Sigma^L_1} = NL^{NL} = NL$ (why?)
- $\Pi^L_2 = coNL^{\Sigma^L_1} = coNL^{NL} = NL$ (why?)

The Logarithmic Hierarchy collapses on the first level.
Q2: The Hardest Problems in P

What we know about P and NP:
- We don’t know if any problem in NP is really harder than any problem in P.
- But we do know that NP is at least as challenging as P, i.e., $P \subseteq NP$.

So all problems that are hard for NP are also hard for P, aren’t they?

Q2: Is NP-hard as hard as P-hard?

Let’s first recall the definitions:

**Definition:** A problem $L$ is NP-hard if, for all problems $M \in NP$, there is a polynomial many-one reduction $M \leq_{m} L$.

**Definition:** A problem $L$ is P-hard if, for all problems $M \in P$, there is a log-space reduction $M \leq_{L} L$.

How to show “NP-hard implies P-hard”?
- Assume that $L$ is NP-hard.
- Consider any language $M \in P$.
- Then $M \in NP$.
- So there is a polynomial many-one reduction $f$ from $M$ to $L$.
- Hence, well... nothing much, really.

### Example 18.6:
We know that $L \subseteq P \subseteq NP$ but we do not know if any of these subsumptions are proper. Suppose that the truth actually looks like this: $L \subseteq P = NP$. Then all non-trivial problems in $P$ are NP-hard (why?), but not every problem would be P-hard (why?).

**Note:** This is really about the different notions of reduction used to define hardness. If we used log-space reductions for P-hardness and NP-hardness, the claim would follow.
Q3: Problems harder than P

Polynomial time is an approximation of "practically tractable" problems:
- Many practical problems are in P, including many very simple ones (e.g., ∅)
- P-hard problems are as hard as any other problem in P, and P-complete problems are therefore the hardest problems in P
- However, there are even harder problems that are no longer in P

So, clearly, problems that are not even in P must be P-hard, right?

Q3: Are problems harder than P also hard for P?

Rephrasing the question: Are there problems that are not in P, yet not hard for P?

Some observations:
- ∅ is not P-hard (why?)
- Any ExpTime-complete problem L is not in P (why?)
- We can enumerate DTMs for all languages in P (how?)
- We can enumerate DTMs for all P-hard languages in ExpTime (how?)

So, it's clear what we have to do now . . .
Q3: Are problems harder than P also hard for P?

Schöning to the rescue (see Theorem 15.2):

Corollary 18.7: Consider the classes $C_1 = \text{ExpPHard}$ (P-hard problems in ExpTime) and $C_2 = \text{P}$. Both are classes of decidable languages. We find that for either class $C_k$:

- We can effectively enumerate TMs $M_{k0}, M_{k1}, \ldots$ such that $C_k = \{L(M_{ki}) | i \geq 0\}$.
- If $L \in C_k$ and $L'$ differs from $L$ on only a finite number of words, then $L' \in C_k$.

Let $L_1 = \emptyset$, and let $L_2$ be some ExpTime-complete problem. Clearly, $L_1 \not\in \text{ExpPHard}$ and $L_2 \not\in \text{P}$ (Time Hierarchy), hence there is a decidable language $L_d \not\in \text{ExpPHard} \cup \text{P}$. Moreover, as $\emptyset \in \text{P}$ and $L_2$ is not trivial, $L_d \leq_p L_2$ and hence $L_d \in \text{ExpTime}$. Therefore $L_d \not\in \text{ExpPHard}$ implies that $L_d$ is not P-hard.

This idea of using Schöning’s Theorem has been put forward by Ryan Williams (link). Our version is a modification requiring $C_1 \subseteq \text{ExpTime}$.

Q3: Are problems harder than P also hard for P?

No, there are problems in ExpTime that are neither in P nor hard for P.

(Other arguments can even show the existence of undecidable sets that are not P-hard1)

Discussion:

- Considering Questions 2 and 3, the use of the word hard is misleading, since we interpret it as difficult
- However, the actual meaning difficult would be “not in a given class” (e.g., problems not in P are clearly more difficult than those in P)
- Our formal notion of hard also implies that a problem is difficult in some sense, but it also requires it to be universal in the sense that many other problems can be solved through it

What we have seen is that there are difficult problems that are not universal.

1Related note: the undecidable $U_H$ is not NP-hard, since it is a so-called sparse language.

Summary and Outlook

“Most experts” think that

- The polynomial hierarchy does not collapse completely (same as P ≠ NP)
- The polynomial hierarchy does not collapse on any level
  (in particular PH ≠ PSpace and there is no PH-complete problem)

But there can always be surprises . . .

We do not know if the Polynomial Hierarchy is real or collapses

Answer 1: The Logarithmic Hierarchy collapses.

Answer 2: We don’t know that NP-hard implies P-hard.

Answer 3: Being outside of P does not make a problem P-hard.

What’s next?

- Holidays
- Circuits as a model of computation
- Randomness

Your Questions
Here's wishing you
a Merry Christmas, a Happy Hanukkah,
a Joyous Yalda, a Cheerful Dōngzhì,
a Great Feast of Juul,
and a Wonderful Winter Solstice,
respectively!

Markus Krötzsch, 16th Dec 2019