CMSC 28100

Introduction to Complexity Theory

Spring 2024
Instructor: William Hoza

Quantum complexity theory

- One can define a complexity class, BQP, consisting of all languages that could be decided in polynomial time by a fully-functional quantum computer
- The mathematical definition of BQP is beyond the scope of this course
- One can prove that $\mathrm{BPP} \subseteq \mathrm{BQP} \subseteq \mathrm{PSPACE}$

Shor's algorithm

- Recall $\text{FACTOR} = \{\langle N, K\rangle : N \text{ has a prime factor } p \leq K\}$
- **Conjecture:** $\text{FACTOR} \notin \text{P}$

Theorem (Shor's algorithm): $\text{FACTOR} \in \text{BQP}$

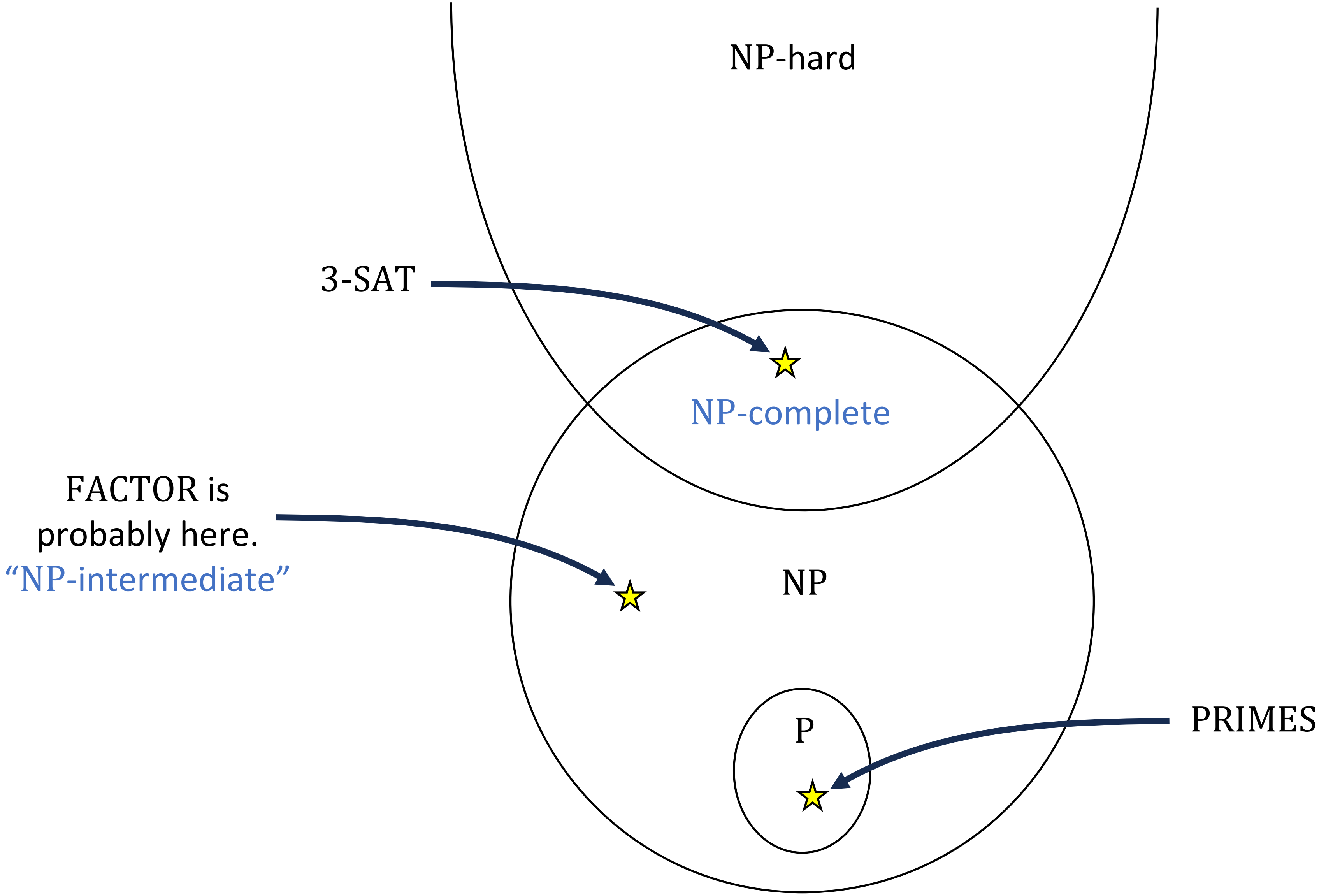
- FACTOR is a likely counterexample to the extended Church-Turing thesis!

Quantum computing and NP-completeness

- Recall: $\mathrm{FACTOR} \in \mathrm{NP}$ (guess the factor)
- Shor's algorithm raises the question: Is FACTOR NP-complete?
- If yes, then $\mathrm{NP} \subseteq \mathrm{BQP}$, meaning that all NP-complete problems could be solved in polynomial time on a fully-functional quantum computer! 😲

Complexity of factoring integers

- In most cases, if a language L is in NP, then we can either prove $L \in \mathrm{P}$ or we can prove that L is NP-complete
- FACTOR is one of the rare exceptions to this rule
- **Conjecture:** FACTOR is neither in P nor NP-complete!


NP-hard
3-SAT
NP-complete
FACTOR is
probably here.
"NP-intermediate"
NP
P
PRIMES

Complexity of factoring integers

- To explain why we expect that FACTOR is not NP-complete, we now introduce another complexity class, called coNP
- The definition of coNP is the same as the definition of NP, except that we swap the roles of “yes” and “no”

The complexity class coNP

- Let $L \subseteq \Sigma^*$ be a language
- **Definition:** $L \in \mathrm{coNP}$ if there exists a randomized polynomial-time Turing machine M such that for every $w \in \Sigma^*$:
  - If $w \in L$, then $\Pr[M \text{ accepts } w] = 1$
  - If $w \notin L$, then $\Pr[M \text{ accepts } w] \neq 1$

The complexity class $\mathbf{coNP}$

- Let L be a language, $L \subseteq \Sigma^*$, and let $\bar{L} = \Sigma^* \setminus L$
- **Fact:** $L \in \mathrm{NP}$ if and only if $\bar{L} \in \mathrm{coNP}$
- coNP is the set of complements of languages in NP
- (This is why it is called "coNP")

The complexity class $\mathbf{coNP}$

- Example: We say that a Boolean formula is unsatisfiable if it is not satisfiable
- Let $3\text{-UNSAT} = \{\langle\phi\rangle : \phi \text{ is an unsatisfiable 3-CNF formula}\}$
- Then $3\text{-UNSAT} \in \text{coNP}$, because a satisfying assignment is a certificate showing that $\phi \notin 3\text{-UNSAT}$

FACTOR $\in$ coNP

- $\text{FACTOR} = \{\langle N, K\rangle : N$ has a prime factor p such that $p \leq K\}$
- **Claim:** $\text{FACTOR} \in \text{coNP}$
- **Proof:** The certificate for non-membership is the full prime factorization of N, i.e., $\langle p_1, \dots, p_k, e_1, \dots, e_k\rangle$ where $N = p_1^{e_1} \cdot \dots \cdot p_k^{e_k}$ and p_i's are distinct primes
- Since $p_i \geq 2$, we have $k \leq \log N$, so the certificate has poly size
- Verification: Confirm that each p_i is prime ($\text{PRIMES} \in \text{P}$); confirm that N really is equal to $\prod_i p_i^{e_i}$; and confirm that the smallest p_i is bigger than K

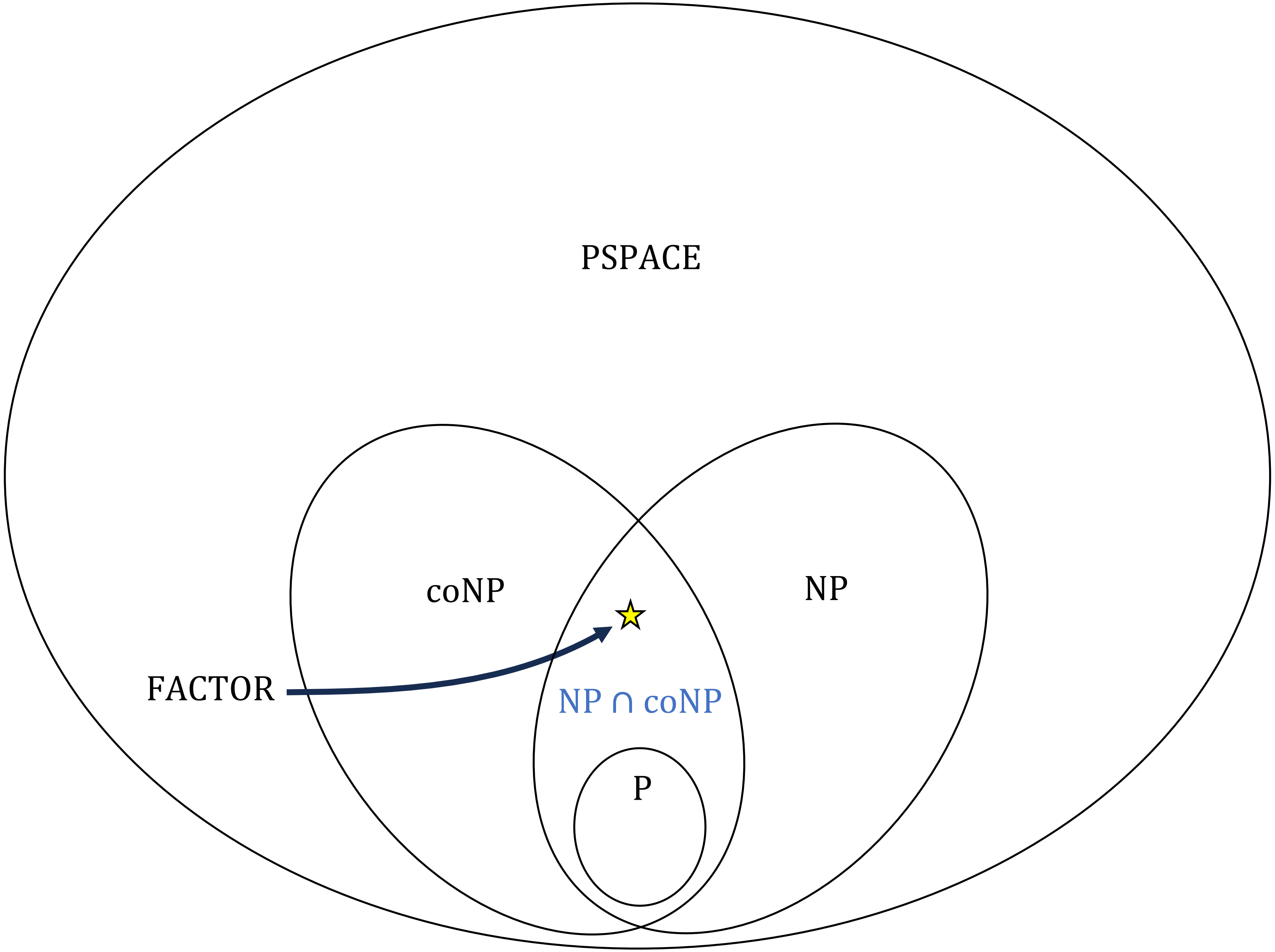
The complexity class $\mathrm{NP} \cap \mathrm{coNP}$

- We have shown that $\mathrm{FACTOR} \in \mathrm{NP}$ and $\mathrm{FACTOR} \in \mathrm{coNP}$
- $\mathrm{FACTOR} \in \mathrm{NP} \cap \mathrm{coNP}$
- $L \in \mathrm{NP} \cap \mathrm{coNP}$ means that for every instance, there is a certificate: a certificate of membership for YES instances and a certificate of non-membership for NO instances

The NP vs. coNP problem

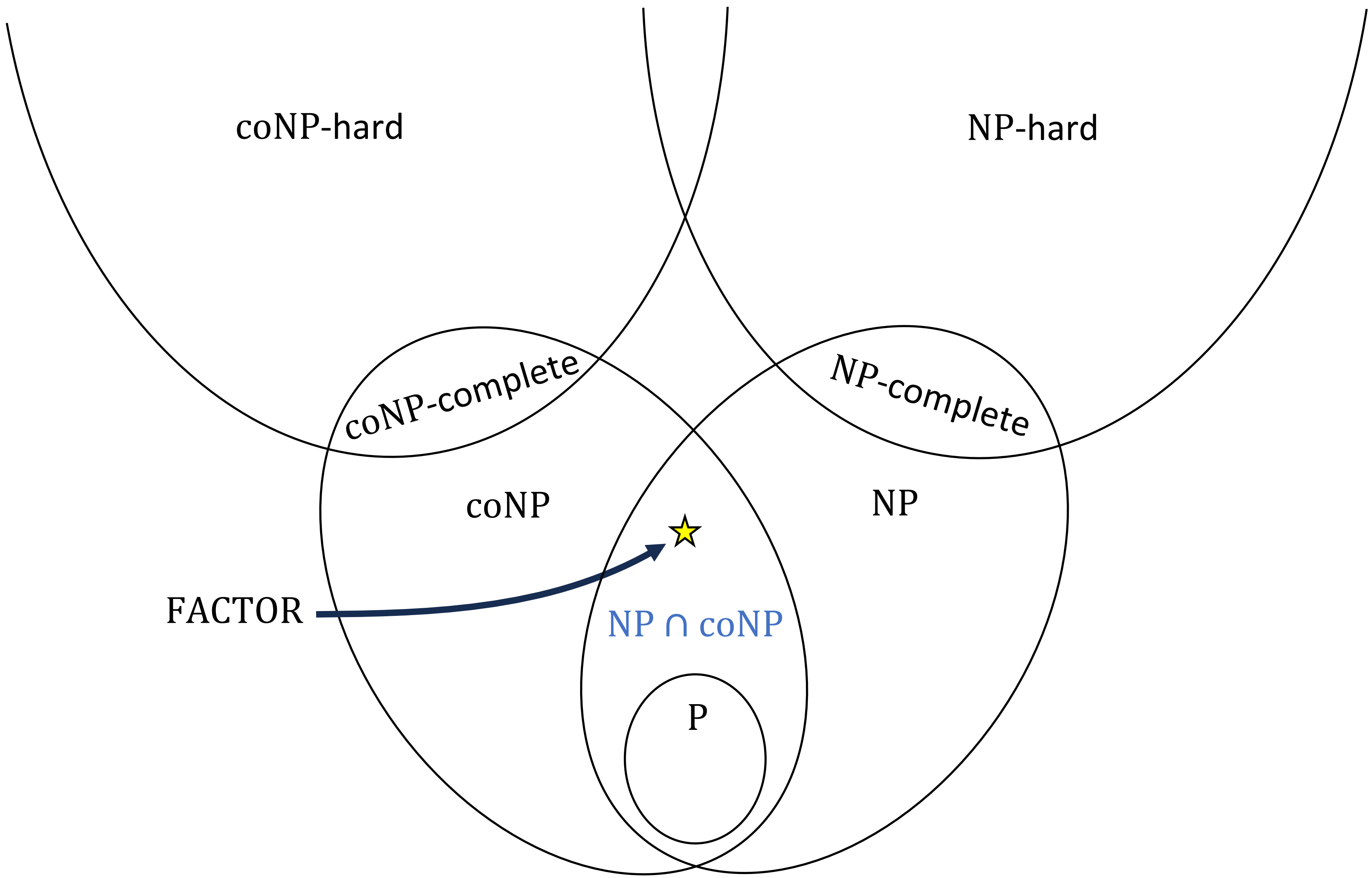
Conjecture: $\mathrm{NP} \neq \mathrm{coNP}$

- The statement $\mathrm{NP} = \mathrm{coNP}$ would mean that for every unsatisfiable circuit, there is some short certificate I could present to prove to you that a circuit is unsatisfiable
- That sounds counterintuitive! But we don't really know


PSPACE
coNP
NP
FACTOR
NP ∩ coNP
P

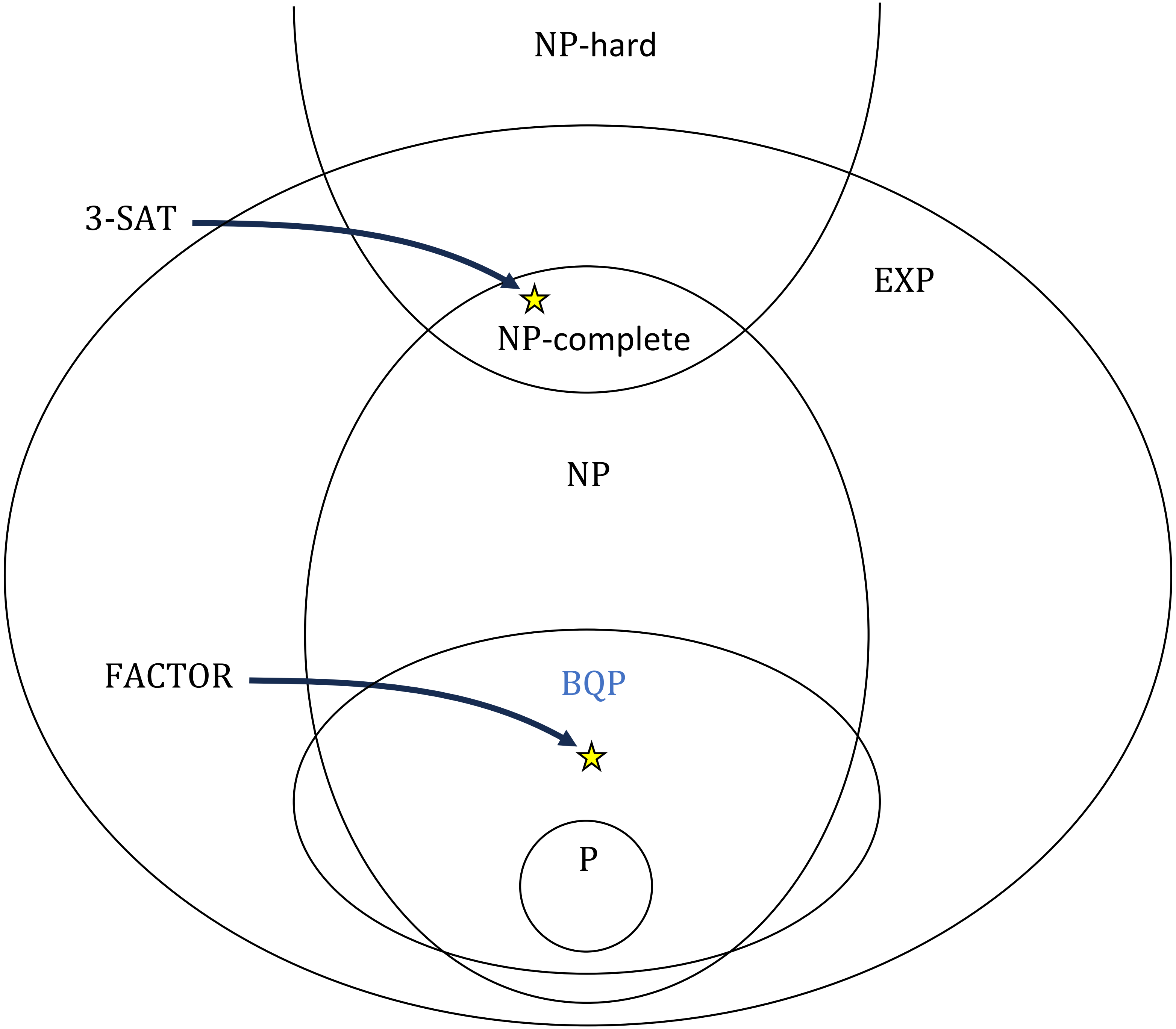
NP-completeness and $\mathbf{NP} \cap \mathbf{coNP}$

- **Fact:** Assuming $\mathrm{NP} \neq \mathrm{coNP}$, there are no NP-complete languages in $\mathrm{NP} \cap \mathrm{coNP}$
- (Proof: Exercise)
- This gives us evidence that FACTOR is not NP-complete


coNP-hard
NP-hard
coNP-complete
NP-complete
coNP
NP
FACTOR
NP ∩ coNP
P

Quantum computing is not a panacea

- $\mathrm{FACTOR} \in \mathrm{BQP}$, but FACTOR is probably not NP-complete
- In fact, it is conjectured that $\mathrm{NP} \nsubseteq \mathrm{BQP}$
- In this case, even a fully-functional quantum computer would not be able to solve NP-complete problems in polynomial time
- Even quantum computers have limitations


NP-hard
3-SAT
EXP
NP-complete
NP
FACTOR
BQP
P

Which problems

can be solved

through computation?

~~CLASSICAL~~

Limitations of quantum computers

- We have developed several techniques for identifying hardness
  - Undecidability
  - EXP-completeness
  - NP-completeness
- Those techniques are all still applicable even in a world with fully-functional quantum computers!
- Complexity theory is intended to be “future-proof” / “timeless”

Which problems

can be solved

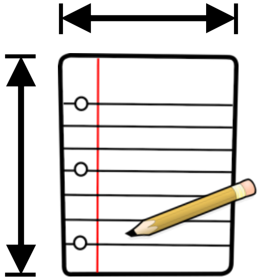
through computation?

Complexity theory:

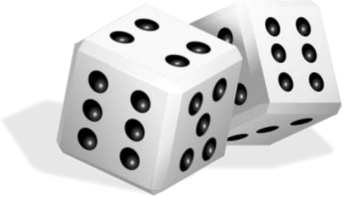
The study of computational resources

Computational resources: Fuel for algorithms

TIME

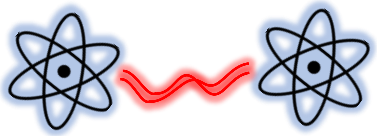


SPACE



RANDOMNESS

COMMUNICATION



QUANTUM PHYSICS

PARALLELISM

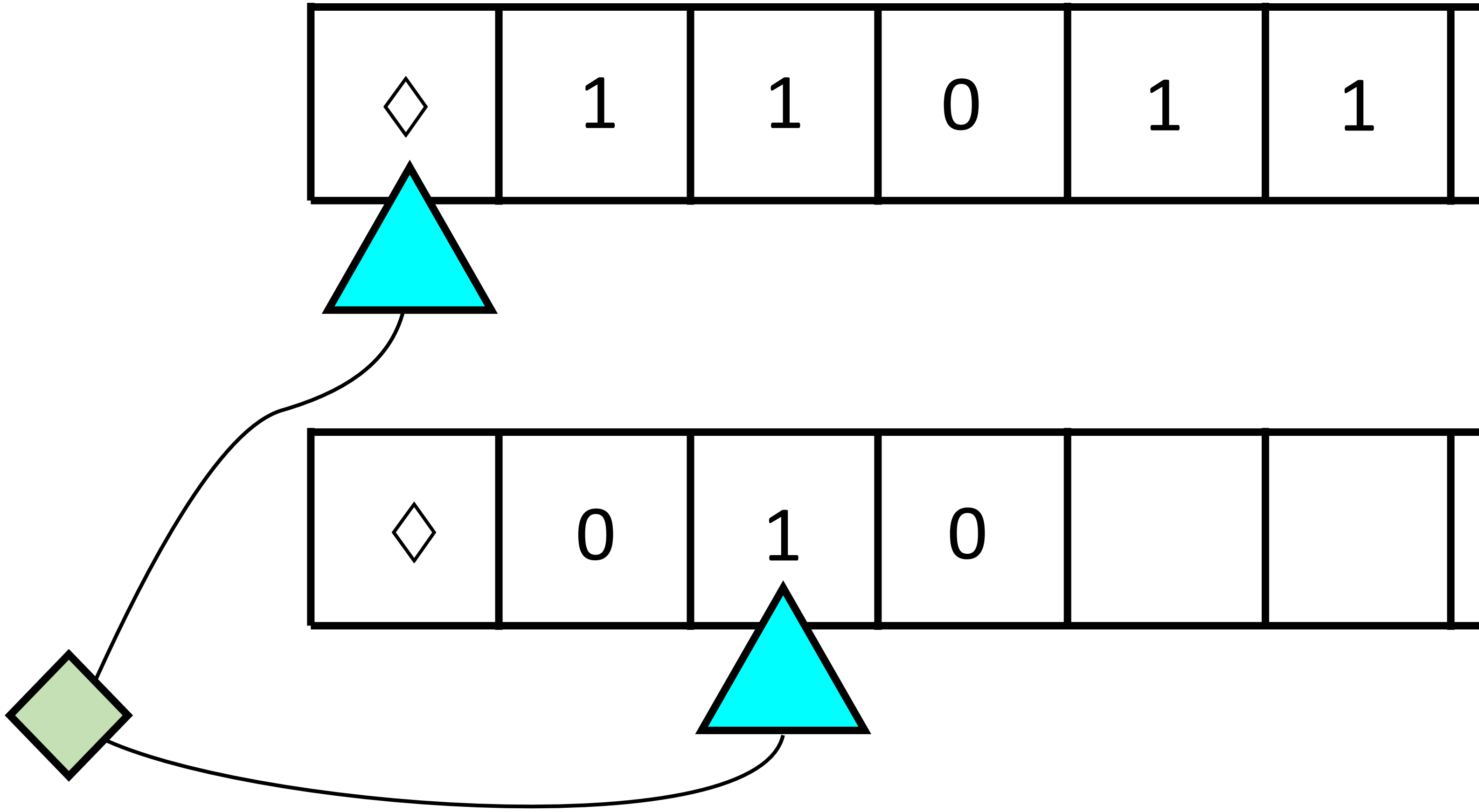
Sublinear-space computation

- Can we solve any interesting problems using $o(n)$ space?
- The one-tape Turing machine is the not the right model of computation for studying sublinear-space algorithms

Sublinear-space computation

Read-only input tape ➞

Read-write work tape ➞



The complexity class $\mathrm{SPACE}(S)$

- Let L be a language and let $S: \mathbb{N} \to \mathbb{N}$ be a function (space bound)
- **Definition:** $L \in \mathrm{SPACE}(S)$ if there is a two-tape Turing machine M such that:
  - M decides L
  - M never modifies the symbols written on tape 1
  - Whenever M reads a blank symbol $\sqcup$ on tape 1, the tape 1 head moves to the left
  - We have $S_M(n) = O(S(n))$, where $S_M(n)$ is the maximum i such that the tape 2 head visits cell i during the computation of M on w for some $w \in \Sigma^n$

The complexity class $\mathbf{L}$

- Exercise: $\mathrm{PSPACE} = \bigcup_k \mathrm{SPACE}(n^k)$
- **Definition:** $\mathrm{L} = \mathrm{SPACE}(\log n)$
- L is the set of languages that can be decided in logarithmic space

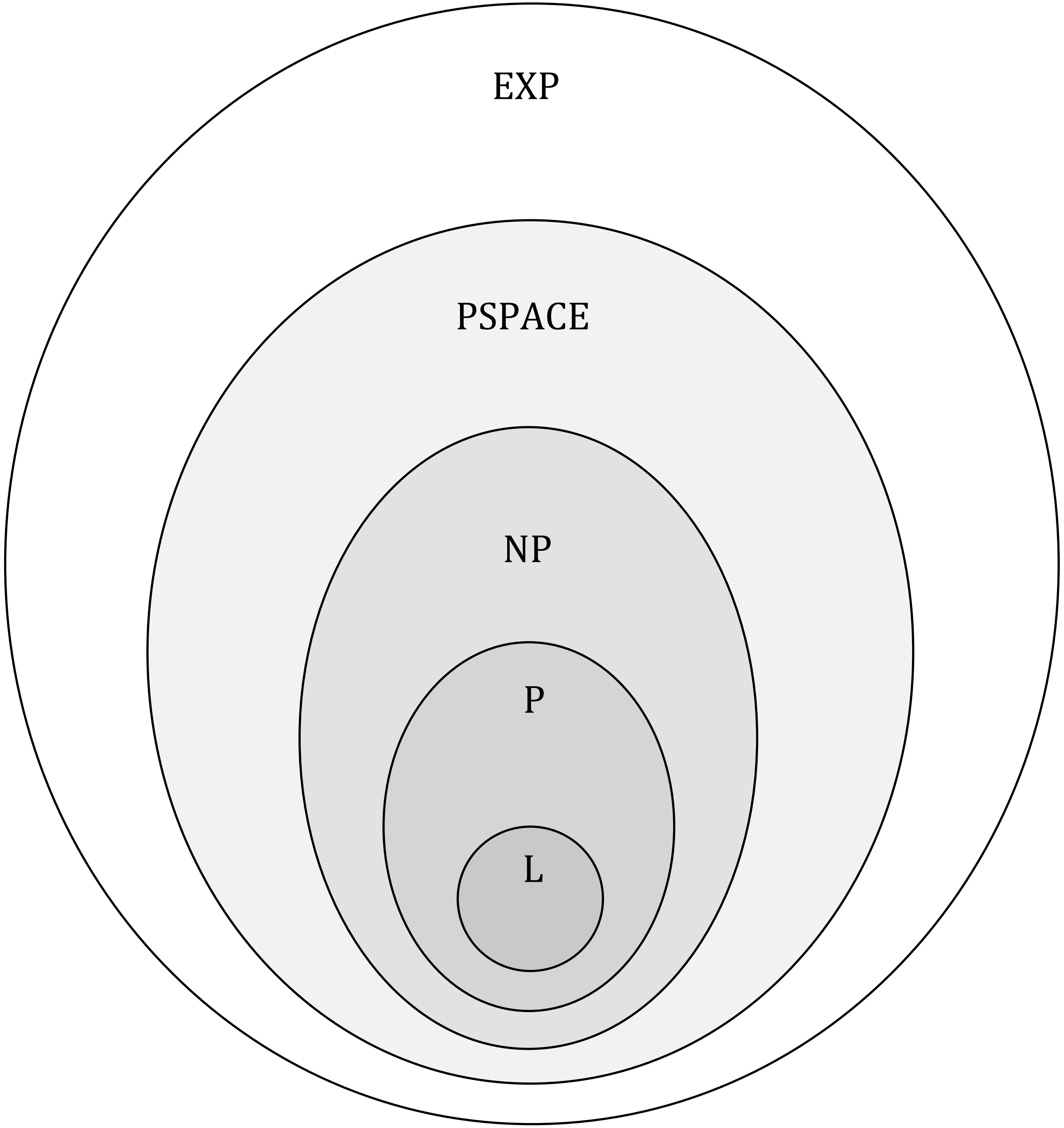
$\text{BALANCED} \in \text{L}$

- $\text{BALANCED} = \{x \in \{0,1\}^* : x \text{ has equal numbers of zeroes and ones}\}$
- **Claim:** $\text{BALANCED} \in \text{L}$
- **Proof sketch:** Given $x \in \{0,1\}^n$:
  - Count the number of ones in x
  - Count the number of zeroes in x
  - Check whether the two counts are equal

These counters are only $\log n$ bits each!

$\mathrm{L} \subseteq \mathrm{P}$

- Exercise: Show that $\mathrm{L} \subseteq \mathrm{P}$
- (Similar to the proof that $\mathrm{PSPACE} \subseteq \mathrm{EXP}$)


EXP
PSPACE
NP
P
L

The $\mathbf{L}$ vs. $\mathbf{P}$ problem

- We expect that $\mathrm{L} \neq \mathrm{P}$, but we don't know how to prove it
- $\mathrm{L} = \mathrm{P}$ would mean that every efficient algorithm can be modified so that it only uses a tiny amount of work space

L vs. P vs. NP vs. PSPACE

- $\mathrm{L} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE}$
- What we expect: All of these containments are strict
- What we can prove: At least one of these containments is strict:

Theorem: $\mathrm{L} \neq \mathrm{PSPACE}$

Nondeterministic log space computation

- We define NL to be the class of languages that can be decided by a nondeterministic log-space Turing machine
- Equivalently: NL is the class of languages for which membership can be verified in logarithmic space – with the extra requirement that the verifier can only read the certificate one time from left to right

Two surprises about $\mathbf{NL}$

- We expect that $\mathrm{P} \neq \mathrm{NP}$. However, in the space complexity world…

Savitch's Theorem: $\mathrm{NL} \subseteq \mathrm{SPACE}(\log^2 n)$

- We expect that $\mathrm{NP} \neq \mathrm{coNP}$. However, in the space complexity world…

Immerman-Szelepcsényi Theorem: $\mathrm{NL} = \mathrm{coNL}$