CMSC 28100

Introduction to Complexity Theory

Spring 2025
Instructor: William Hoza

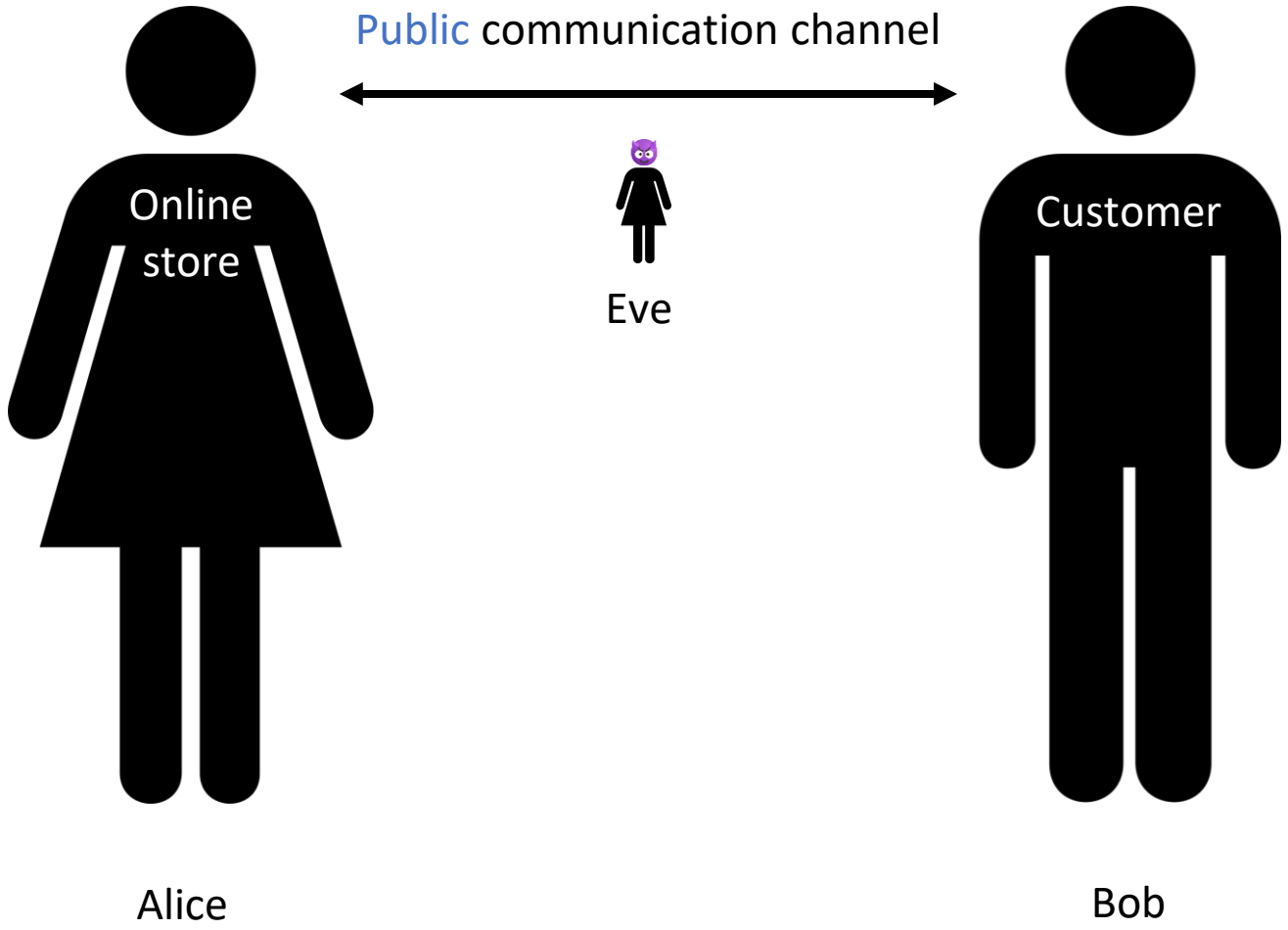
How to feel about intractability

- We have encountered several tractable problems in this course
  - PALINDROMES, PIT, 2-COLORABLE, …
  - Conventional attitude: This is "good news" 🙂
- We have also identified many problems that are probably/definitely intractable
  - HALT, BOUNDED-HALT, 3-SAT, CLIQUE, …
  - Conventional attitude: This is "bad news" 😟
- Twist: Sometimes we are hoping that certain problems are intractable! 🙃

Cryptography

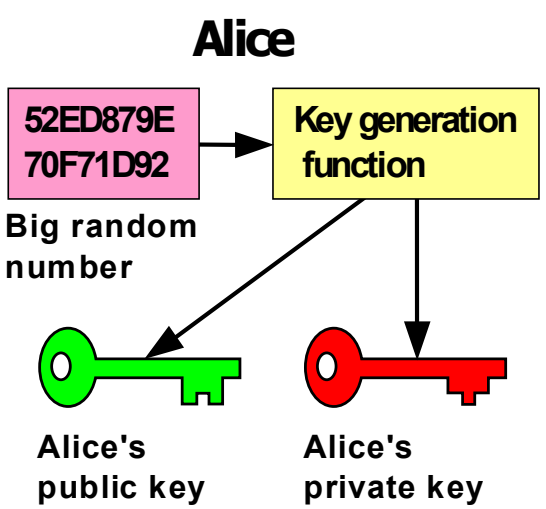
Secure communication

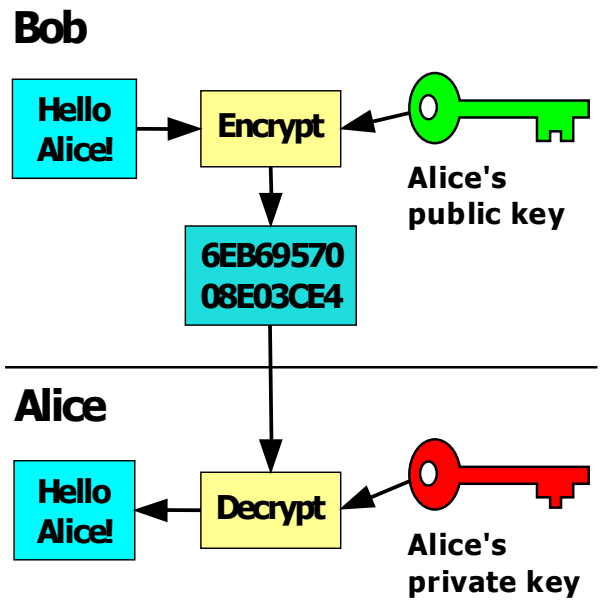
- How can Bob send a private message to Alice?
  - E.g., credit card number
- It seems impossible, because Alice and Eve receive all the same information from Bob!



- A clever approach: Try to force Eve to solve an intractable problem

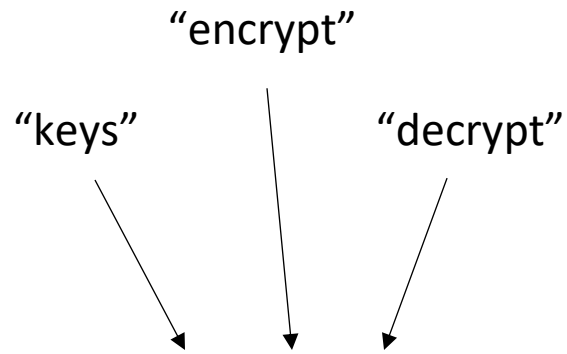
Public-key encryption





- Alice's advantage over Eve: Alice knows the private key and Eve doesn't

Public-key encryption scheme



- **Definition:** A simplified public-key encryption scheme is a triple (K, E, D), where:
  - $K \subseteq \{0,1\}^* \times \{0,1\}^*$ and $E, D: \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$
  - For every $w \in \{0,1\}^*$ and every $(k_{\text{pub}}, k_{\text{priv}}) \in K$, we have $D\left(k_{\text{priv}}, E\left(k_{\text{pub}}, w\right)\right) = w$
  - E and D can be computed in polynomial time
  - For every $(k_{\text{pub}}, k_{\text{priv}}) \in K$, we have $|k_{\text{pub}}| = |k_{\text{priv}}|$

If Eve is computationally unbounded

- Let's show that if Eve has unlimited computational power, then encryption is futile
- **Claim:** There exists a function $D_{\text{Eve}}: \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ such that for every message $w \in \{0,1\}^*$ and every pair $(k_{\text{pub}}, k_{\text{priv}}) \in K$, we have

$$D_{\text{Eve}}\left(k_{\text{pub}}, E\left(k_{\text{pub}}, w\right)\right) = w$$

- **Proof:** If $E(k_{\text{pub}}, w) = E(k_{\text{pub}}, w') = y$, then $w = D(k_{\text{priv}}, y) = w'$ ✔

What if Eve is computationally bounded?

- Amazing fact: There are known public-key encryption schemes such that decrypting without the private key seems to be intractable!
  - (*Better: There are schemes such that it is apparently intractable to "occasionally" "partially" decrypt without the private key. Making this precise is beyond the scope of our course)
- Example: "RSA"
- These amazing encryption schemes make our modern internet experience possible! Can we prove that they are secure?

Cryptography and $\mathbf{P}$ vs. $\mathbf{NP}$

- Let (K, E, D) be a simplified public-key encryption scheme
- There is a function D_{Eve} such that $D_{\mathrm{Eve}}\left(k_{\mathrm{pub}}, E\left(k_{\mathrm{pub}}, w\right)\right) = w$

Theorem: If $\mathrm{P} = \mathrm{NP}$, then D_{Eve} can be computed in polynomial time 😟

Cryptography and P vs. $\mathbf{NP}$

Theorem: If $\mathrm{P} = \mathrm{NP}$, then D_{Eve} can be computed in polynomial time 😟

- **Proof:** Let $Y = \{\langle k_{\mathrm{pub}}, y, w\rangle : \text{there exists } z \text{ such that } E(k_{\mathrm{pub}}, wz) = y\}$
- $Y \in \mathrm{NP}$: Guess z. (Since D is poly-time-computable, z is poly-size)
- We are assuming $\mathrm{P} = \mathrm{NP}$, so therefore $Y \in \mathrm{P}$
- Therefore, Eve can construct the message bit-by-bit in polynomial time

Cryptography and P vs. NP

- Disclaimer: The preceding discussion of public-key encryption is simplified
  - For example, a real encryption scheme should explain how to generate keys
- Nevertheless, the main message is accurate:
- If $\mathrm{P} = \mathrm{NP}$, then secure public-key encryption is impossible!

Cryptography and P vs. NP

- In fact, virtually all of theoretical cryptography relies on assumptions that are stronger than the assumption $\mathrm{P} \neq \mathrm{NP}$
- Maybe this makes you feel concerned about the uncertain foundations of computer security… 😟
- Or, maybe this makes you feel more confident that $\mathrm{P} \neq \mathrm{NP}$, considering how much effort people expend trying to break cryptosystems 🙂

~~Which problems~~

~~can be solved~~

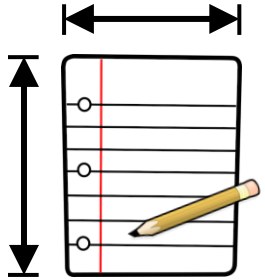
~~through computation?~~

Complexity theory:

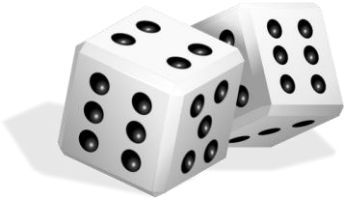
The study of computational resources

Computational resources: Fuel for algorithms

TIME

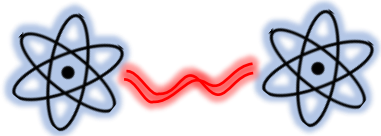


SPACE



RANDOMNESS

COMMUNICATION



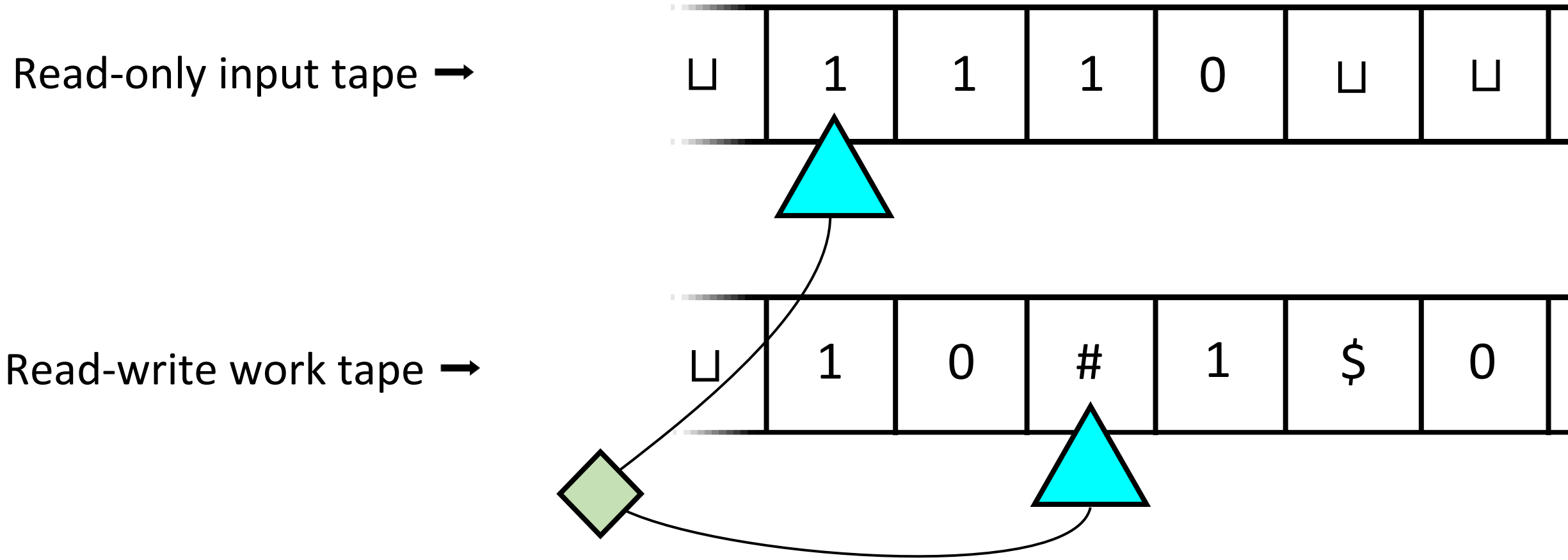
QUANTUM PHYSICS

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



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

- Let $Y \subseteq \{0,1\}^*$ and let $S\colon \mathbb{N} \to \mathbb{N}$ be a function (space bound)
- **Definition:** $Y \in \mathrm{SPACE}(S)$ if there is a two-tape Turing machine M such that:
  - M decides Y
  - M never modifies the symbols written on tape 1
  - The tape 1 head is always located within one cell of the input
  - When the input has length n, the tape 2 head visits $O\big(S(n)\big)$ cells

The complexity class 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

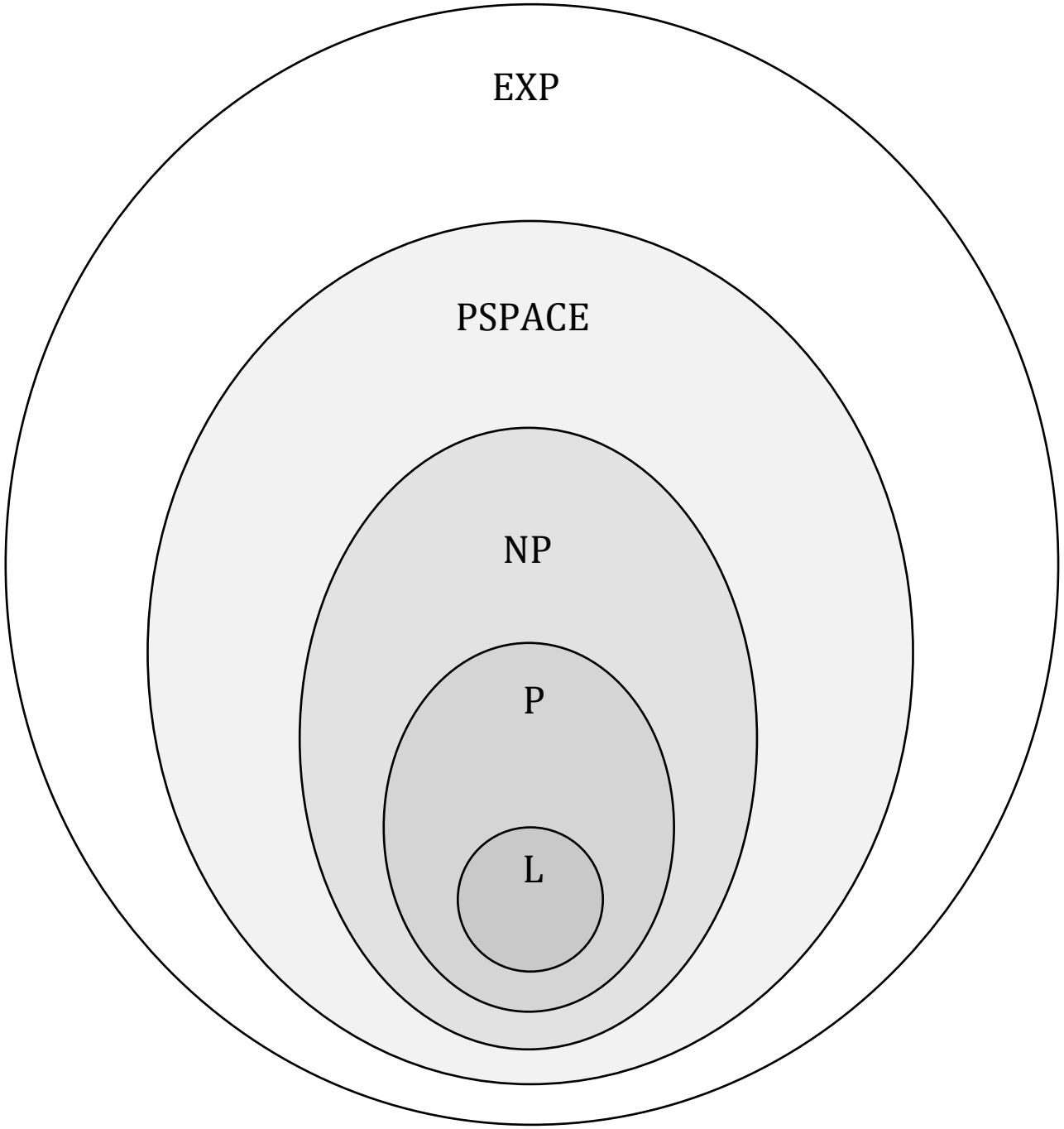
$\mathrm{BALANCED} \in \mathrm{L}$

- $\mathrm{BALANCED} = \{x \in \{0,1\}^* : x \text{ has equal numbers of zeroes and ones}\}$
- **Claim:** $\mathrm{BALANCED} \in \mathrm{L}$
- **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 L vs. 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