CMSC 28100

Introduction to Complexity Theory

Spring 2024
Instructor: William Hoza

Which problems can be solved through computation?

CLASSICAL

Which languages are in $\mathbf{P}$?

Which languages are not in $\mathbf{P}$?

The Time Hierarchy Theorem

- Let $T: \mathbb{N} \to \mathbb{N}$ be a "reasonable" time complexity bound (we will come back to this)

Theorem: $\mathrm{TIME}\big(o(T)\big) \neq \mathrm{TIME}(T^3)$

- Given a little more time, we can solve more problems

Proof of the Time Hierarchy Theorem

Let $L = \{\langle M \rangle : M \text{ rejects } \langle M \rangle \text{ within } T(|\langle M \rangle|) \text{ steps}\}$

- **Claim 1:** $L \notin \mathrm{TIME}\big(o(T)\big)$
- Proof: Last class

Proof of the Time Hierarchy Theorem

Let $L = \{\langle M \rangle : M \text{ rejects } \langle M \rangle \text{ within } T(|\langle M \rangle|) \text{ steps}\}$

- **Claim 2:** $L \in \mathrm{TIME}(T^3)$

Subtle point: How do we know when we're done?

- **Proof:** Given $\langle M \rangle$, we simulate M on $\langle M \rangle$ for $T(|\langle M \rangle|)$ steps and check whether it rejects
- Exercise: Verify that we can simulate a single step of M using $O(T^2)$ steps
- Total time complexity: $O(T \cdot T^2) = O(T^3)$

Time-constructible functions

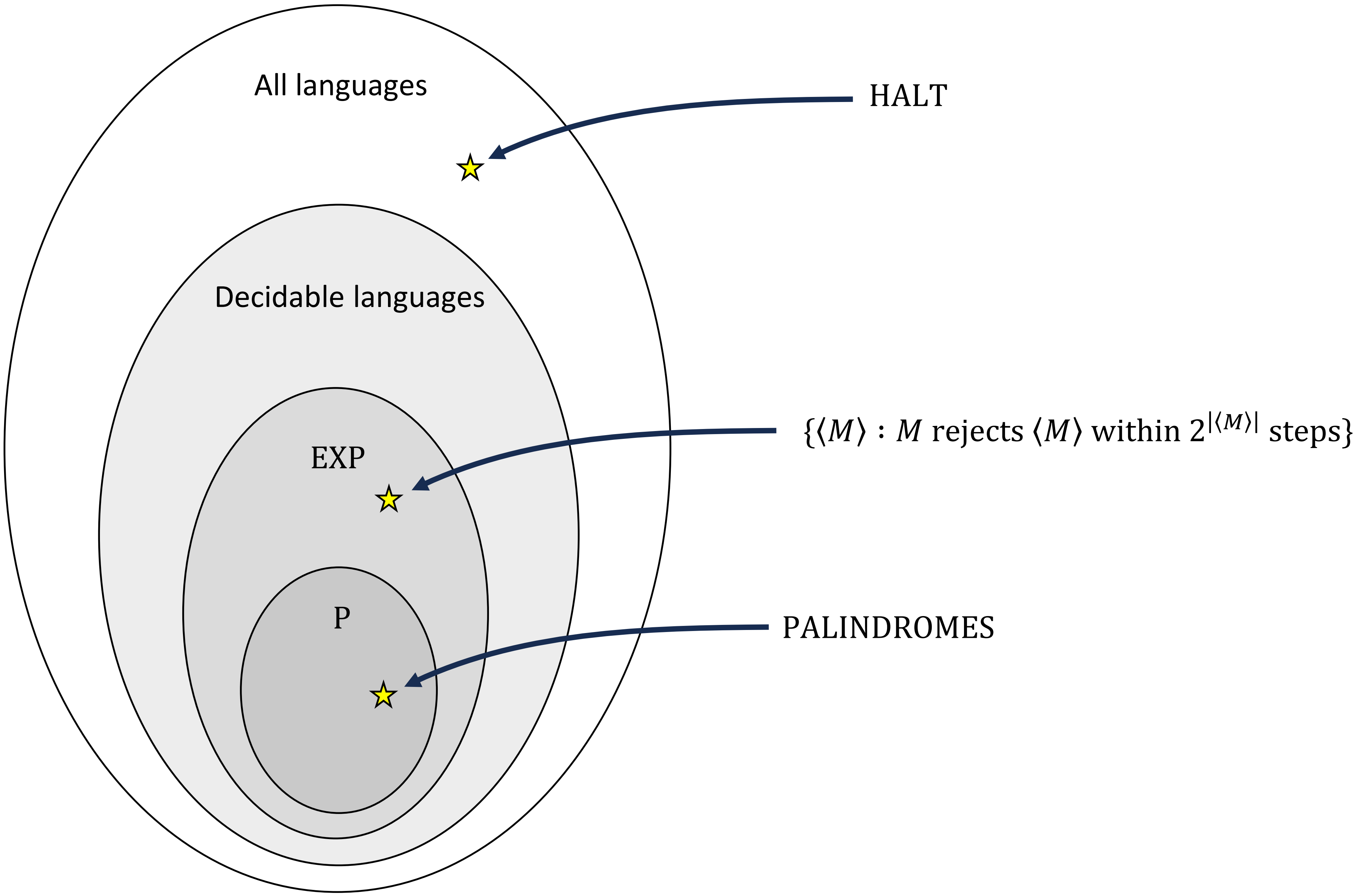
- We say that a function $T: \mathbb{N} \to \mathbb{N}$ is time-constructible if there is a multi-tape Turing machine M such that
  - Given input 1^n, M halts with $1^{T(n)}$ written on tape 2
  - M has time complexity $O(T(n))$
- The time hierarchy theorem applies to any time-constructible T
- All “reasonable” time complexity bounds (e.g., $5n$, n^2, 2^n, etc.) are time-constructible

Corollary: $\mathrm{P} \neq \mathrm{EXP}$

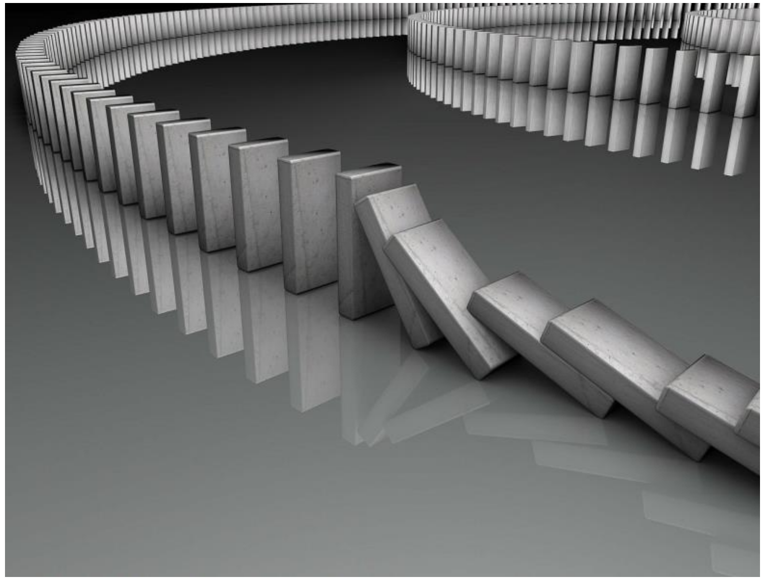
- **Proof:**

$$\mathrm{P} = \bigcup_{k=1}^{\infty} \mathrm{TIME}\left(n^k\right) \subseteq \mathrm{TIME}\left(o(2^n)\right) \subsetneq \mathrm{TIME}(2^{3n}) \subseteq \mathrm{EXP}$$

- Interpretation: There are some exponential-time algorithms that cannot be converted into polynomial-time algorithms


All languages
Decidable languages
EXP
P
HALT
$\{\langle M \rangle : M \text{ rejects } \langle M \rangle \text{ within } 2^{|\langle M \rangle|} \text{ steps}\}$
PALINDROMES

"Natural" languages outside P



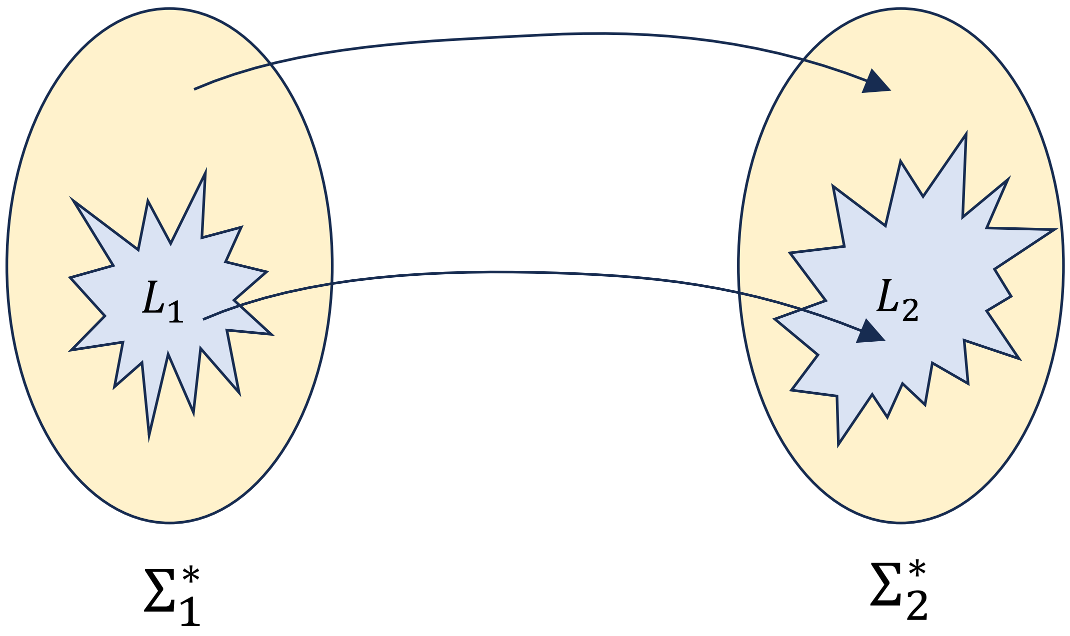
- Now we know that there exists a decidable language outside P
- However, the language seems a bit "artificial" / "contrived"
- What else is outside P?
- Can we prove that some "natural" decidable languages are outside P?

Polynomial-time reductions

- Let L_1 and L_2 be languages over the alphabets Σ_1 and Σ_2 respectively
- **Definition:** A poly-time mapping reduction from L_1 to L_2 is a function $f: \Sigma_1^* \to \Sigma_2^*$ such that
  - For every $w \in L_1$, we have $f(w) \in L_2$ (“YES maps to YES”)
  - For every $w \in \Sigma_1^* \setminus L_1$, we have $f(w) \notin L_2$ (“NO maps to NO”)
  - The function f is poly-time computable, i.e., there exists a TM M such that for every $w \in \Sigma_1^*$, M halts on w within poly($|w|$) steps with $\Diamond f(w)$ written on its tape

Polynomial-time reductions

- A poly-time mapping reduction from L_1 to L_2 is a way of efficiently converting instances of L_1 into equivalent instances of L_2



Reductions: Proving that a language is in P

- Suppose there exists a poly-time mapping reduction f from L_1 to L_2
- **Claim:** If $L_2 \in \mathrm{P}$, then $L_1 \in \mathrm{P}$
- **Proof:** Given $w \in \Sigma_1^*$:

Let $n = |w|$ and $m = |f(w)|$. What can we say about the relationship between n and m?

A: $m \leq \mathrm{poly}(n)$

B: $n \leq \mathrm{poly}(m)$

C: $n = m$

D: We cannot say anything

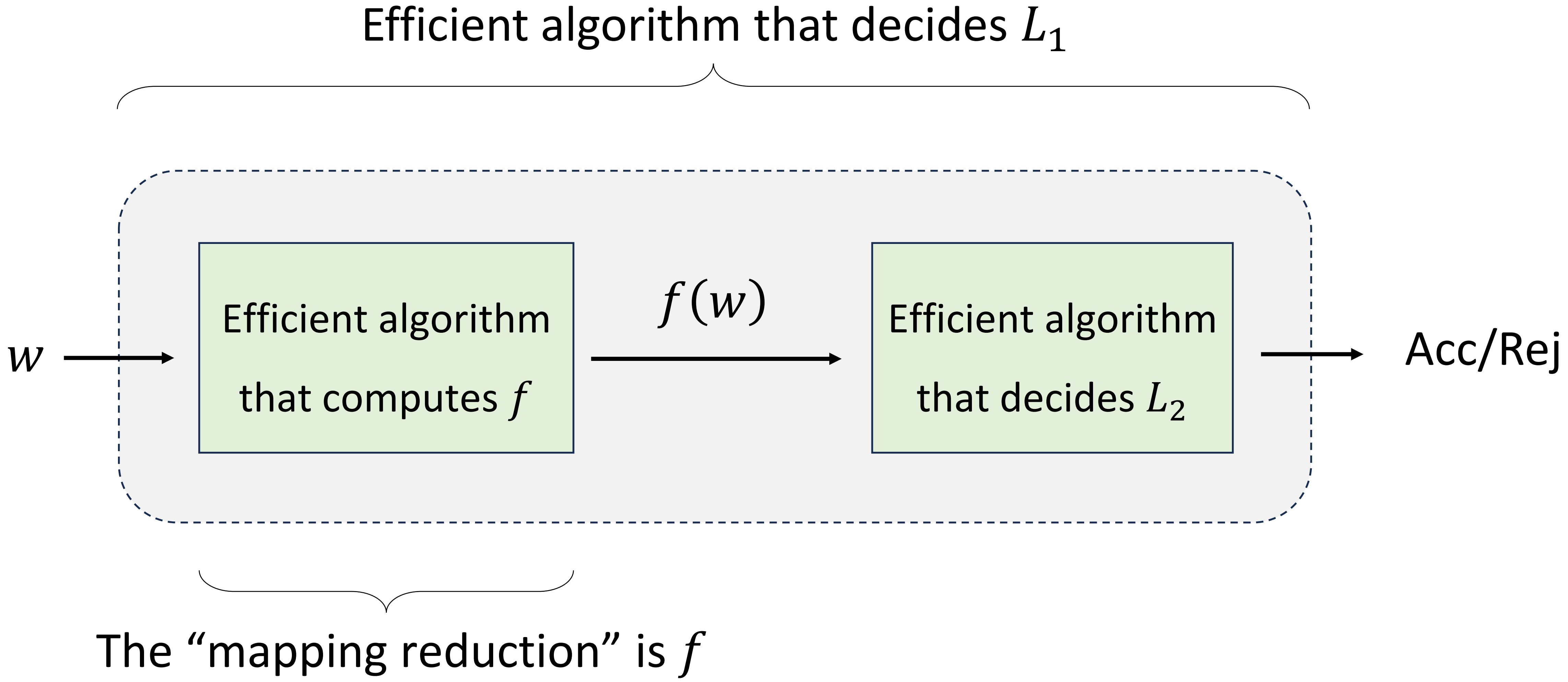
Respond at PollEv.com/whoza or text “whoza” to 22333

s $O(n^{k_1})$ time)

s $O(m^{k_2})$ time where $m = |f(w)|$)

$k_1 \cdot k_2) = \mathrm{poly}(n)$

Reductions: Proving that a language is in P



Reductions: Proving that a language is not in P

- Suppose there exists a poly-time mapping reduction f from L_1 to L_2
- **Claim:** If $L_1 \notin \mathrm{P}$, then $L_2 \notin \mathrm{P}$
- **Proof:** If L_2 were in P, then L_1 would be in P

Using reductions to prove intractability

- Strategy for proving that some language L is not in P:
  - Identify a suitable language L_{HARD} that we previously proved is not in P
  - Design a poly-time mapping reduction f from L_{HARD} to L
  - ⚠️ *Make sure you do the reduction in the correct direction!*
- The amazing thing about this strategy is that the existence of one efficient algorithm implies the nonexistence of another!

The time-bounded halting problem

- Let $\mathrm{BOUNDED\text{-}HALT} = \{\langle M, w, T\rangle : M \text{ halts on } w \text{ within } T \text{ steps}\}$
- Exercise: By simulating M on w, one can decide $\mathrm{BOUNDED\text{-}HALT}$ in time $O(|\langle M\rangle| \cdot T^2)$
- Does this mean $\mathrm{BOUNDED\text{-}HALT} \in \mathrm{P}$?
- No! If we wanted a polynomial-time algorithm, then our time budget would be $\mathrm{poly}(n)$, where $n = |\langle M, w, T\rangle| \approx |\langle M\rangle| + |\langle w\rangle| + \log T$

Time-bounded halting problem

- Let $\text{BOUNDED-HALT} = \{\langle M, w, T\rangle : M \text{ halts on } w \text{ within } T \text{ steps}\}$
- **Claim:** $\text{BOUNDED-HALT} \notin \text{P}$
- **Proof:** Let $L_{\text{HARD}} = \{\langle M\rangle : M \text{ rejects } \langle M\rangle \text{ within } 2^{|\langle M\rangle|} \text{ steps}\}$
- Mapping reduction: $f(\langle M\rangle) = \langle M', w, T\rangle$, where $w = \langle M\rangle, T = 2^{|\langle M\rangle|}$, and M' is a modified version of M in which q_{accept} has been replaced with looping

> - YES maps to YES: If M rejects $\langle M\rangle$ within $2^{|\langle M\rangle|}$ steps, then M' halts on w within T steps ✓
> - NO maps to NO: If M does not reject $\langle M\rangle$ within $2^{|\langle M\rangle|}$ steps, then M' does not halt on w within T steps ✓
> - f is poly-time computable ✓ Note that $\langle T\rangle = 10^{|\langle M\rangle|}$.

How hard is hard?

- We can say more than merely “$\mathrm{BOUNDED\text{-}HALT} \notin \mathrm{P}$”
- We can more precisely characterize the complexity of $\mathrm{BOUNDED\text{-}HALT}$

EXP-hardness

- **Definition:** Let L be a language. Suppose that for every $L' \in \text{EXP}$, there is a poly-time mapping reduction from L' to L. In this case, we say that L is EXP-hard
- "L is EXP-hard" means "L is at least as hard as any language in EXP"

EXP-completeness

- **Definition:** Let L be a language. We say that L is EXP-complete if L is EXP-hard and $L \in \mathrm{EXP}$
- The EXP-complete languages are the hardest languages in EXP
- If L is EXP-complete, then the language L can be said to "capture" / "express" the entire complexity class EXP

$\mathrm{BOUNDED\text{-}HALT}$ is EXP-complete

- **Claim:** $\mathrm{BOUNDED\text{-}HALT}$ is EXP-complete
- **Proof:** First, let's show that $\mathrm{BOUNDED\text{-}HALT} \in \mathrm{EXP}$
- Algorithm: Given $\langle M, w, T \rangle$, we simulate M on w for T steps
- Exercise: This algorithm has time complexity

$$O(|\langle M \rangle| \cdot T^2) = O(n \cdot (2^n)^2) = 2^{O(n)}$$