BU CS 332 – Theory of Computation

Lecture 23:

- Savitch's Theorem
- PSPACE-Completeness
- Unconditional Hardness
- Course Evaluations

Final: 48-hour take-home

Out 5:00 Tu 5/5, Due 5:00 Th 5/7

Later today: Practice final

Tomorrow/Wed: reviewing

Reading:

Sipser Ch 8.1-8.3, 9.1

Mark Bun

April 27, 2020

Space analysis

Space complexity of a TM (algorithm) = maximum number of tapes cell it uses on a worst-case input

Formally: Let $f : \mathbb{N} \rightarrow \mathbb{N}$. A TM M runs in space $f(n)$ if on every input $w \in \Sigma^{\cancel{*}\,n}$, M halts on w using at most $f(n)$ cells

For nondeterministic machines: Let $f : \mathbb{N} \rightarrow \mathbb{N}$. An NTM N runs in space $f(n)$ if on every input $w \in \Sigma^{\cancel{*}\,n}$, N halts on w using at most $f(n)$ cells on every computational branch

Space complexity classes

Let $f : \mathbb{N} \rightarrow \mathbb{N}$

A language $A \in \mathrm{SPACE}(f(n))$ if there exists a basic single-tape (deterministic) TM M that

1) Decides A, and
2) Runs in space $O(f(n))$

A language $A \in \mathrm{NSPACE}(f(n))$ if there exists a single-tape nondeterministic TM N that

1) Decides A, and
2) Runs in space $O(f(n))$

Savitch's Theorem

Savitch's Theorem: Deterministic vs. Nondeterministic Space

Contrast to time $NTIME(f(n)) \subseteq TIME(2^{O(f(n))})$

Theorem: Let f be a function with $f(n) \geq n$. Then $NSPACE(f(n)) \subseteq SPACE\left((f(n))^2\right)$.

Nondeterministic space-bounded TMs can be simulated by deterministic space-bounded TMs w/ quadratic overhead

Proof idea:

- Let N be an NTM deciding $\cancel{f}\,A$ in space $f(n)$
- We construct a TM M deciding $\cancel{f}\,A$ in space $O\left((f(n))^2\right)$
- Actually solve a more general problem:
  - Given configurations c_1, c_2 of N and natural number t, decide whether N can go from c_1 to c_2 in $\leq t$ steps on some nondeterministic path.
  - Design procedure CANYIELD(c_1, c_2, t)

Deciding whether $w \in A$ $\Longleftrightarrow$ CANYIELD(c_0, c_{accept}, t) where t is a bound on N's run time

How do we bound t? $t = 2^{O(f(n))}$

Savitch's Theorem

Theorem: Let f be a function with $f(n) \geq n$. Then $NSPACE(f(n)) \subseteq SPACE\left((f(n))^2\right)$.

Proof idea:

- Let N be an NTM deciding A in space $f(n)$

M = "On input w:

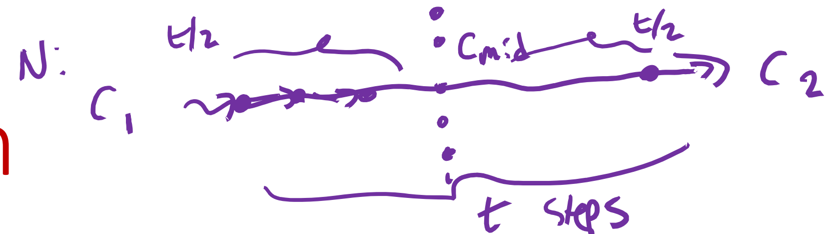
c_1 = start configuration
c_2 = accept config.

1. Output the result of CANYIELD$(c_1, c_2, 2^{df(n)})$"

d constant chosen s.t. N halts in time $2^{df(n)}$

where CANYIELD(c_1, c_2, t) decides whether N can go from configuration c_1 to c_2 in $\leq t$ steps on some nondeterministic path

Savitch's Theorem



CANYIELD(c_1, c_2, t) decides whether N can go from configuration c_1 to c_2 in $\leq t$ steps on some nondeterministic path:

Idea: Divide and conquer

Final space bound: $S(2^{d f(n)}) = O(f(n) \cdot \log(2^{d f(n)}))$ $= O(f(n)^2)$

CANYIELD(c_1, c_2, t) =

Base case

1. If $t = 1$, accept if $c_1 = c_2$ or c_1 yields c_2 in one transition.

   Else, reject. Exp. time, but only $O(f(n))$ space

Inductive case

2. If $t > 1$, then for each config c_{mid} of N with $\leq f(n)$ cells:
3. Run CANYIELD$(\langle c_1, c_{mid}, t/2 \rangle)$.
4. Run CANYIELD$(\langle c_{mid}, c_2, t/2 \rangle)$.
5. If both runs accept, accept.
6. Reject.

Correctness: CANYIELD(c_1, c_2, t) $\iff$ $\exists\, c_{mid}$ s.t. CANYIELD$(c_1, c_{mid}, t/2)$ and CANYIELD$(c_{mid}, c_2, t/2)$

Space complexity: $S(t)$ = space required for CANYIELD(c_1, c_2, t)

Base case: $S(1) = n$

$S(t) = O(f(n)) + S(t/2)$

$\Rightarrow S(t) = O(f(n) \cdot \log(t))$

Complexity class PSPACE

Definition: PSPACE is the class of languages decidable in polynomial space on a basic single-tape (deterministic) TM

$$\text{PSPACE} = \bigcup_{k=1}^{\infty} \text{SPACE}(n^k)$$

Definition: NPSPACE is the class of languages decidable in polynomial space on a single-tape (nondeterministic) TM

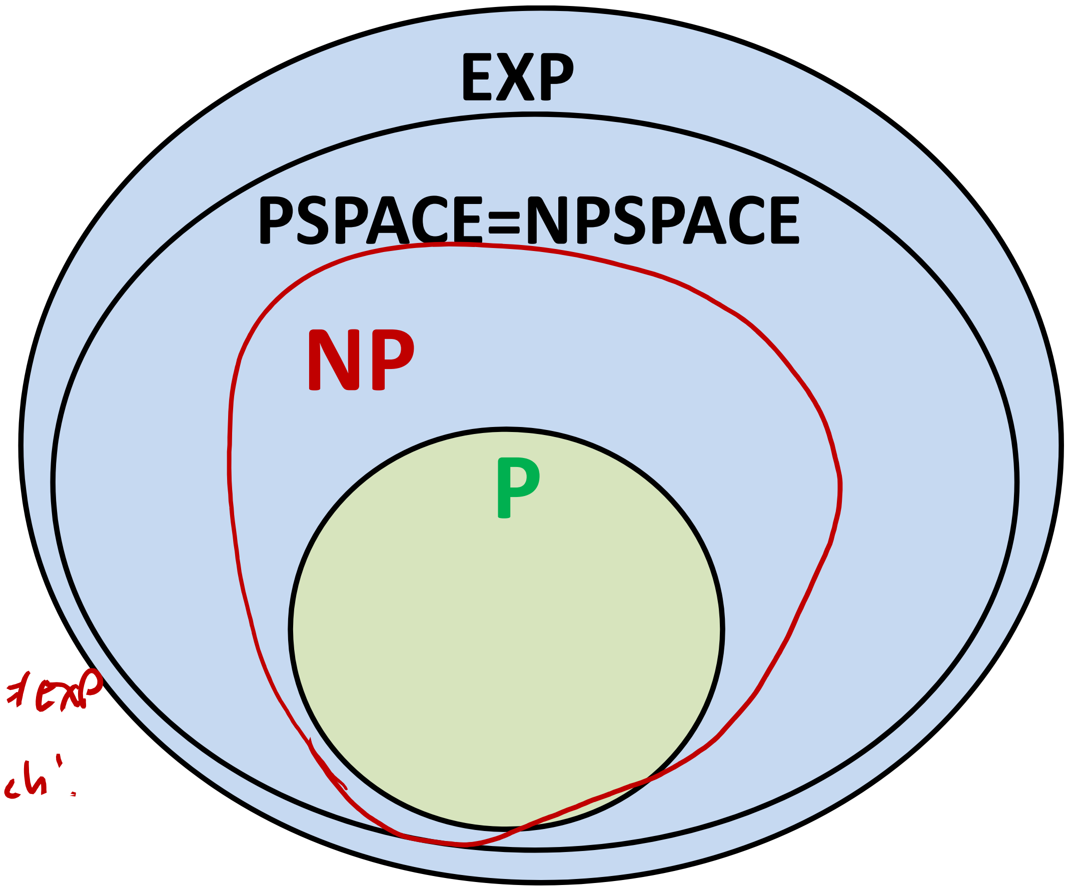
$$\text{NPSPACE} = \bigcup_{k=1}^{\infty} \text{NSPACE}(n^k)$$

PSPACE = NPSPACE $\subsetneq$ EXPSPACE

Relationships between complexity classes

1. P ⊆ NP ⊆ PSPACE ⊆ EXP

since $SPACE\big(f(n)\big) \subseteq TIME(2^{O(f(n))})$



2. P ≠ EXP (~~Monday~~ Today)

Which containments in (1) are proper?

Unknown!

At least one of P ≠ NP, NP ≠ PSPACE, or PSPACE ≠ EXP

but we don't know which!

PSPACE-Completeness

What happens in a world where $\mathrm{P} \neq \mathrm{PSPACE}$?

Even more believable than $\mathrm{P} \neq \mathrm{NP}$, but still(!) very far from proving it

Question: What would $\mathrm{P} \neq \mathrm{PSPACE}$ allow us to conclude about problems we care about?

PSPACE-completeness: Find the "hardest" problems in PSPACE

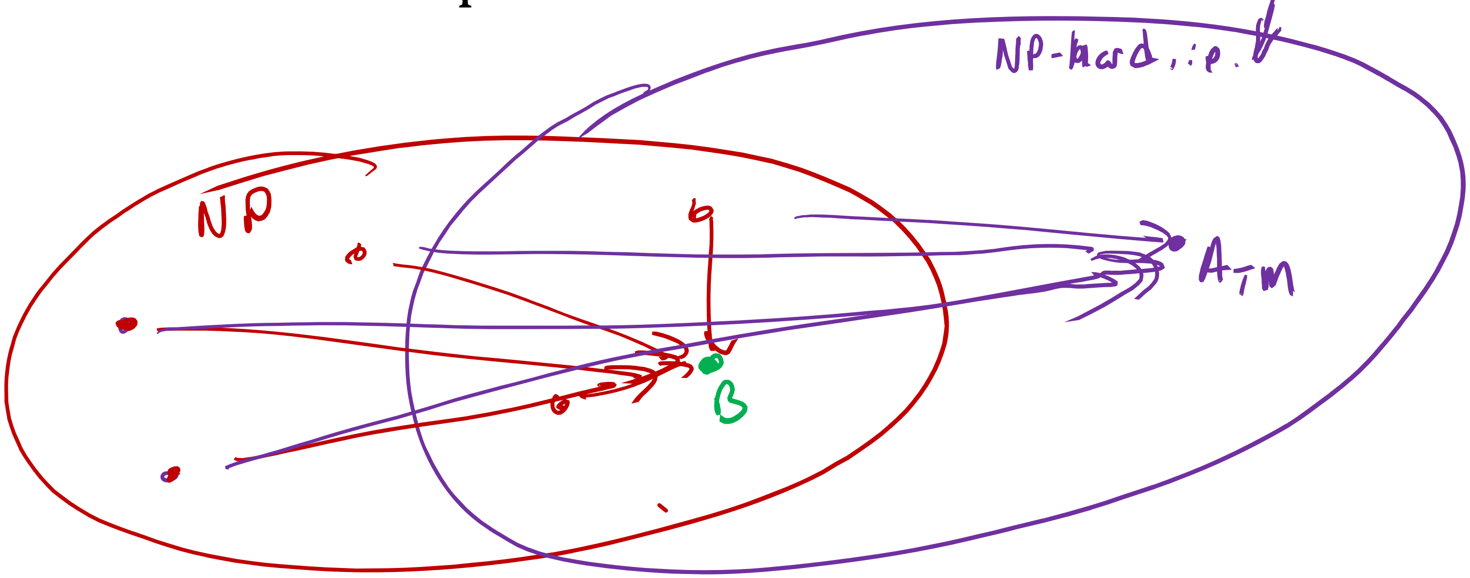
Find $L \in \mathrm{PSPACE}$ such that $L \in \mathrm{P}$ iff $\mathrm{P} = \mathrm{PSPACE}$

Poly-time reductions

Reminder: NP-completeness

Definition: A language B is NP-complete if

1) $B \in \mathrm{NP}$, and
2) **Every** language $A \in \mathrm{NP}$ is poly-time reducible to B, i.e., $A \leq_{\mathrm{p}} B$ ("B is NP-hard")



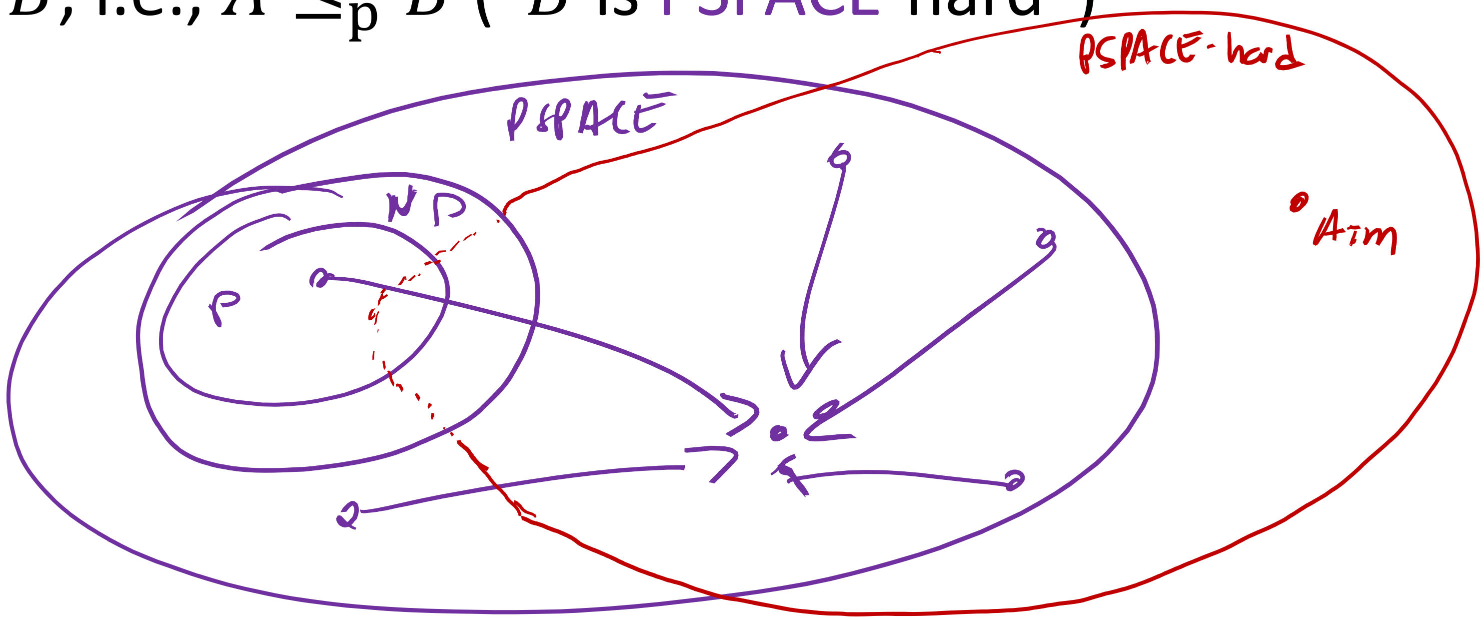
PSPACE-completeness

Definition: A language B is PSPACE-complete if

1) $B \in \text{PSPACE}$, and

$P = PSPACE \iff B \in P$

2) Every language $A \in \text{PSPACE}$ is poly-time reducible to B, i.e., $A \leq_{\mathrm{p}} B$ ("B is PSPACE-hard")



A PSPACE-complete problem: TQBF

"True quantified Boolean formula"

"Is a fully quantified logical formula true?"

- Boolean variable: Variable that can take on the value true/false (encoded as 0/1)
- Boolean operations: $\wedge$ (AND), $\vee$ (OR), $\neg$ (NOT)
- Boolean formula: Expression made of Boolean variables and operations. Ex: $(x_1 \vee \overline{x_2}) \wedge x_3$

Player 2

- Fully quantified Boolean formula: Boolean formula with all variables quantified ($\forall, \exists$) Ex: $\forall x_1 \exists x_3 \forall x_2 \boxed{(x_1 \vee \overline{x_2}) \wedge x_3}$

Player 1

- Every fully quantified Boolean formula is either true or false

SAT = $\{\langle \varphi \rangle \mid \varphi$ is a TQBF, and all quantifiers are $\exists\}$

- $TQBF = \{\langle \varphi \rangle | \varphi \text{ is a true fully quantified formula}\}$

Theorem: TQBF is PSPACE-complete

Need to prove two things…

1) $TQBF \in \text{PSPACE}$

2) Every problem in PSPACE is poly-time reducible to $TQBF$ ($TQBF$ is PSPACE-hard)

1) TQBF is in PSPACE

ψ

$\varphi = \exists x_1 \forall x_2 \exists x_3 \forall x_4 \dots (x_1 \vee x_2) \wedge x_3 \vee$

T = "On input $\langle \varphi \rangle$,
where φ is a fully quantified Boolean formula:

1. If φ has no quantifiers, it has only constants (and no variables). Evaluate φ. If true, accept; else, reject.

   e.g. $(1 \vee 0) \wedge \overline{(0 \wedge 1)}$

2. If φ is of the form $\exists x\, \psi$, recursively call T on ψ with $x = 0$ and then on ψ with $x = 1$. If either call accepts, accept; else, reject.
3. If φ is of the form $\forall x\, \psi$, recursively call T on ψ with $x = 0$ and then on ψ with $x = 1$. If both calls accept, accept; else, reject."

- If n is the input length, T uses space $O(n)$.

$S(k)$ = space bound for TQBF on k variables

$S(0) = O(n)$

$S(k) = T(k-1) + O(1)$

$S(k) = O(k) + O(n)$

$\Rightarrow S(n) = O(n)$

2) TQBF is PSPACE-hard

Theorem: Every language $A \in \mathrm{PSPACE}$ is poly-time reducible to $TQBF$

Proof idea:

Let $A \in \mathrm{PSPACE}$ be decided by a poly-space deterministic TM M. Using proof of Cook-Levin Theorem,

$$M \text{ accepts input } w \iff \underbrace{\text{formula } \varphi_{M,w} \text{ is true}}$$

M space-bounded $\Rightarrow$ $\varphi_{M,w}$ might have exponential size

Using idea of Savitch's Theorem, replace $\varphi_{M,w}$ with a quantified formula of poly-size that can be computed in poly-time

Unconditional Hardness

Hardness results so far

- If $\mathrm{P} \neq \mathrm{NP}$, then $3SAT \notin P$ [3SAT NP-complete]
- If $\mathrm{P} \neq \mathrm{PSPACE}$, then $TQBF \notin P$ {TQBF PSPACE-complete]

Question: Are there decidable languages that we can show are not in P? w/o using unproven assumptions

Diagonalization redux

(used before to show $\overline{SA_{TM}}$ undecidable)

TM M						
M_1						
M_2						
M_3						
M_4						
$\vdots$						

Diagonalization redux

TM M	$M(\langle M_1 \rangle)$?	$M(\langle M_2 \rangle)$?	$M(\langle M_3 \rangle)$?	$M(\langle M_4 \rangle)$?		$D(\langle D \rangle)$?
M_1	~~Y~~ N	N	Y	Y	…	
M_2	N	~~N~~ Y	Y	Y		
M_3	Y	Y	~~Y~~ N	N		
M_4	N	N	Y	~~N~~ Y		
⋮					⋱	
D						~~Y~~ N ~~N~~ Y

undecidable

$\overline{SA_{\text{TM}}} = \{\langle M \rangle \mid M$ is a TM that does not accept input $\langle M \rangle\}$

$\overline{SA_{\text{TM},EXP}} = \{\langle M \rangle \mid M$ is a TM that does not accept input $\langle M \rangle$ within $2^{|\langle M \rangle|}$ steps$\}$

suppose D decides $\overline{SA_{\text{TM},EXP}}$ in poly time

An explicit undecidable language

- Theorem: $L = \overline{SA_{\mathrm{TM},EXP}} = \{\langle M\rangle \mid M$ is a TM that does not accept input $\langle M\rangle$ within $2^{|\langle M\rangle|}$ steps$\}$

is in EXP, but not in P

P ≠ EXP

Proof:

- In EXP: Simulate M on input $\langle M\rangle$ for $2^{|\langle M\rangle|}$ steps and flip its decision

  (2^n steps)

- Not in P: Suppose for contradiction that D decides L in time n^k

  $\langle D\rangle \in L \Rightarrow D(\langle D\rangle)$ rejects in time $|\langle D\rangle|^k \leq 2^{|\langle D\rangle|}$ $\times$

  $\langle D\rangle \notin L \Rightarrow D(\langle D\rangle)$ accepts in time " $\times$

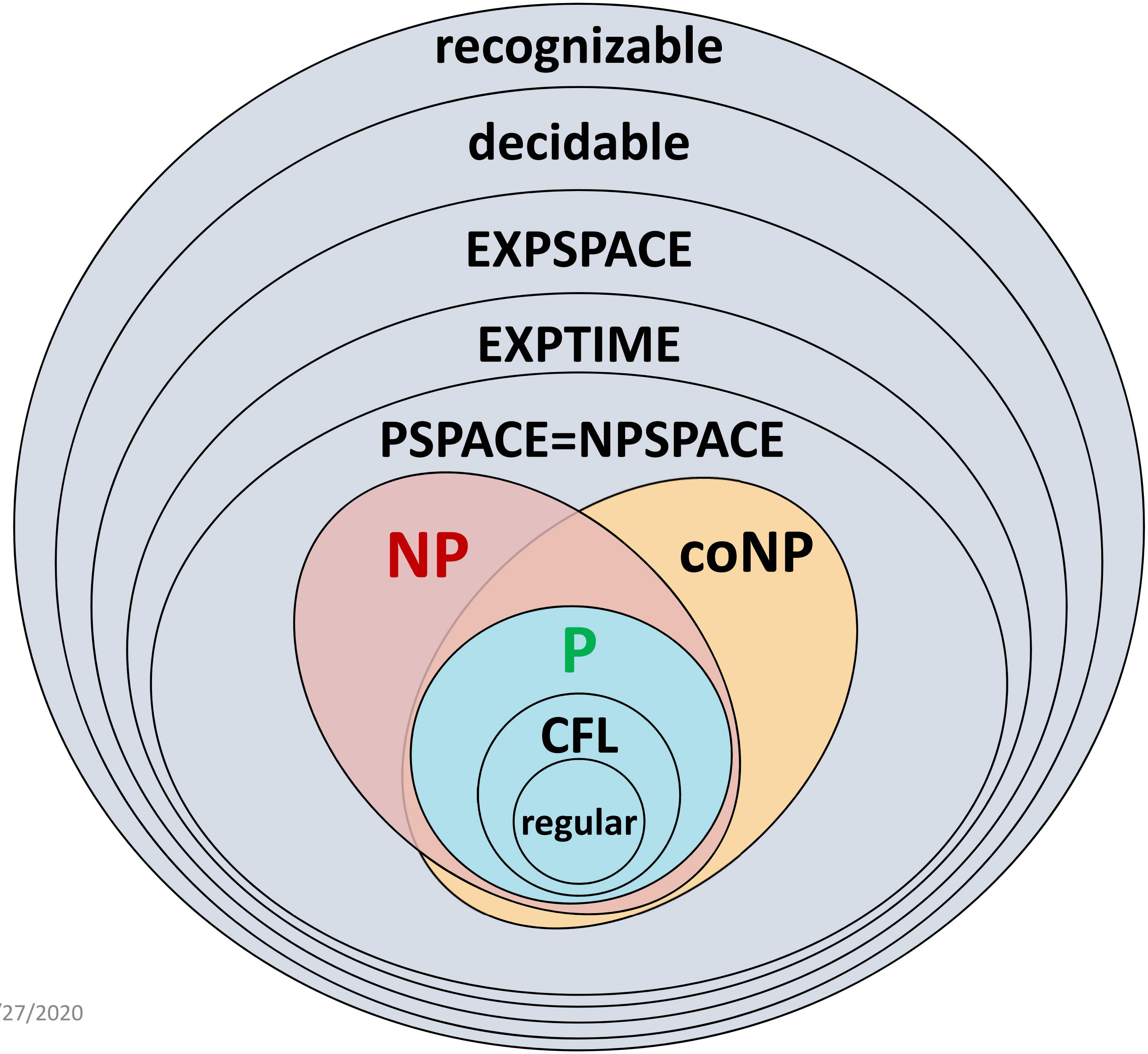
Time and space hierarchy theorems

- For any* function $f(n) \geq n \log n$, a language exists that is decidable in $f(n)$ time, but not in $o\left(\frac{f(n)}{\log f(n)}\right)$ time.

- For any* function $f(n) \geq n \log n$, a language exists that is decidable in $f(n)$ space, but not in $o(f(n))$ space.

*time constructible and space constructible, respectively

$\exists$ TM that on input 1^n outputs $1^{f(n)}$ in $O(f(n))$ steps


recognizable
decidable
EXPSPACE
EXPTIME
PSPACE=NPSPACE
NP
coNP
P
CFL
regular

Course evaluations

535 - Grad Complexity Fall 2020

https://bu.campuslabs.com/courseeval

SNARG = Succinct non-interactive argument

- Verifier only needs polylog n communication from prover
- Sacrifice "soundness" for "computational soundness"

$2^{\log^2 n} = n^{\log n}$

NP-intermediate: Neither in P nor NP-complete

Candidates: - Variant of integer factoring
- Graph isomorphism [Babai 2016]

GI ∈ quasi P GI ∈ TIME$(2^{\log^c n})$ for some const c

Logarithmic?

Probably GI ∈ P?

How powerful are "simple" circuit models?

constant depth, polynomial size

$AC^0 \not\ni$ PARITY [80's]

$ACC^0 = AC^0$ + "mod p" gates

very recent: 2011 - ongoing

$NQP \not\subseteq ACC^0$

NQP = nondet. quasi-P time