BU CS 332 – Theory of Computation

Lecture 23:

- Space Complexity

Reading:

Sipser Ch 8.1-8.3

Ran Canetti

December 3, 2020

Complexity measures we've studied so far

- Deterministic time TIME
- Nondeterministic time NTIME
- Classes P, NP

Complexity measures we've studied so far

- Deterministic time TIME
- Nondeterministic time NTIME
- Classes P, NP

What about space complexity?

Space complexity: The maximum amount of information that is "kept around" at any point during the computation.

Complexity measures we've studied so far

- Deterministic time TIME
- Nondeterministic time NTIME
- Classes P, NP

What about space complexity?

Space complexity: The maximum amount of information that is "kept around" at any point during the computation.

➔ A fundamental measure! (perhaps even more than time)

Space analysis

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

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

Space analysis

Space complexity of a TM (algorithm) = maximum number of tape 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 $w \in \Sigma^*$, 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 $w \in \Sigma^*$, 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 $O(f(n))$

Example: Space complexity of SAT

Theorem: $SAT \in \mathrm{SPACE}(n)$

Proof: The following deterministic TM decides SAT using linear space

On input $\langle \varphi \rangle$ where φ is a Boolean formula:

1. For each truth assignment to the variables $x_1, \dots, x_m$ of φ:
2. Evaluate φ on $x_1, \dots, x_m$
3. If any evaluation $= 1$, accept. Else, reject.

Example: NFA analysis

Theorem: Let $ALL_{NFA} = \{A \mid A \text{ is an NFA with } L(A) = \Sigma^*\}$

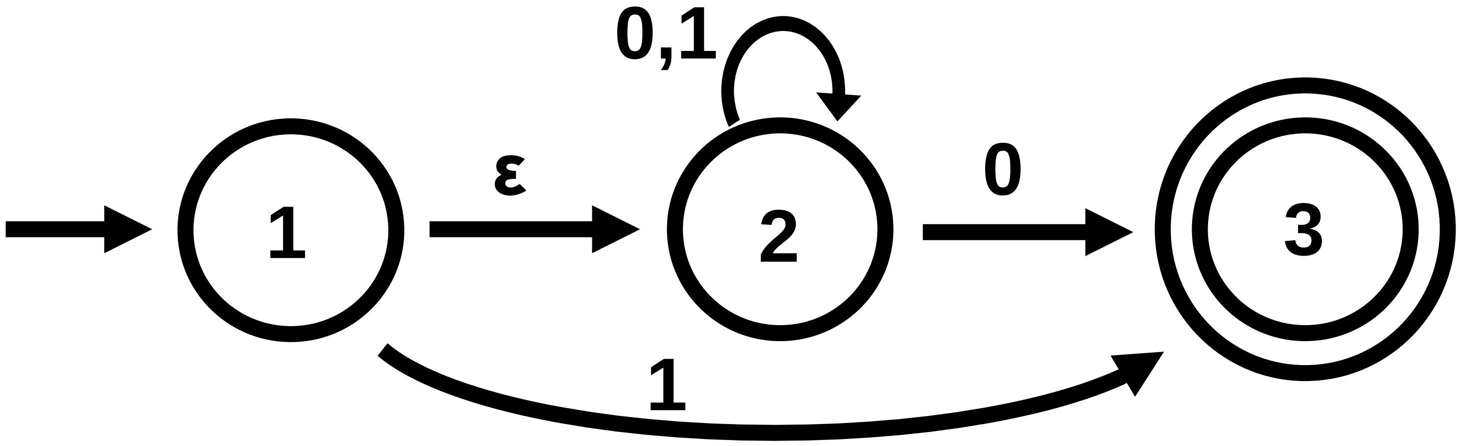
Then $\overline{ALL_{NFA}} \in \text{NSPACE}(n)$.

Proof: The following NTM decides $\overline{ALL_{NFA}}$ in linear space

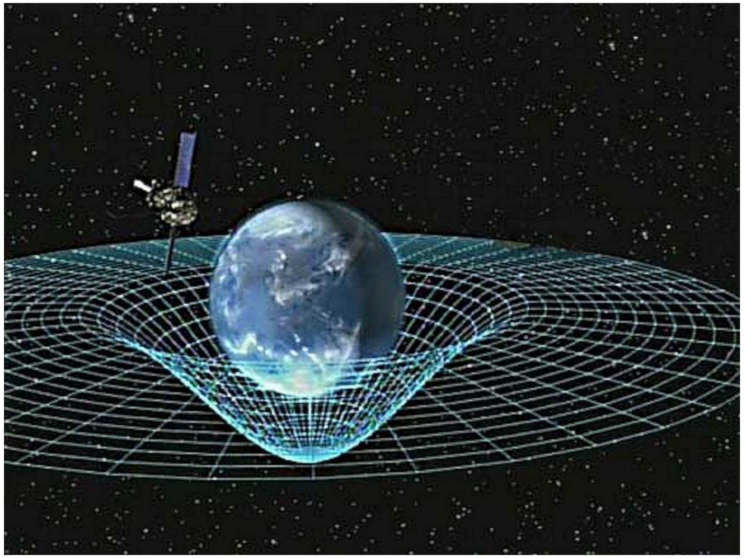
On input $\langle A \rangle$ where A is an NFA:

1. Place a marker on the start state of A.
2. Repeat 2^q times where q is the # of states of A:
3. Nondeterministically select $a \in \Sigma$.
4. Adjust the markers to simulate all ways for A to read a.
5. Accept if at any point *none* of the markers are on an accept state. Else, reject.

Example



Space vs. Time



Space vs. Time

$$TIME\big(f(n)\big) \subseteq NTIME\big(f(n)\big) \subseteq SPACE\big(f(n)\big)$$

How about the opposite direction? Can low-space algorithms be simulated by low-time algorithms?

Reminder: Configurations

A configuration is a string uqv where $q \in Q$ and $u, v \in \Gamma^*$

- Tape contents = uv (followed by blanks ⊔)
- Current state = q
- Tape head on first symbol of v

Example: $101q_5 0111$

Start configuration: $q_0 w$

Accepting configuration: q = q_{accept}

Rejecting configuration: q = q_{reject}

Reminder: Configurations

Consider a TM with

- k states
- tape alphabet $\{0, 1\}$
- space $f(n)$

How many configurations are possible when this TM is run on an input $w \in \{0,1\}^n$?

Observation: If a TM enters the same configuration twice when run on input w, it loops forever

Corollary: A TM running in space $f(n)$ also runs in time $2^{O(f(n))}$

Savitch's Theorem

Savitch's Theorem: Deterministic vs. Nondeterministic Space

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 f in space $f(n)$
- We construct a TM M deciding f 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.

  ➔ Procedure CANYIELD(c_1, c_2, t)

Savitch's Theorem

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

Proof idea:

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

M = "On input w:

1. Output the result of CANYIELD$(c_s, c_{acc}, 2^{df(n)})$"

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:

CANYIELD(c_1, c_2, t) =

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

Complexity class PSPACE

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

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

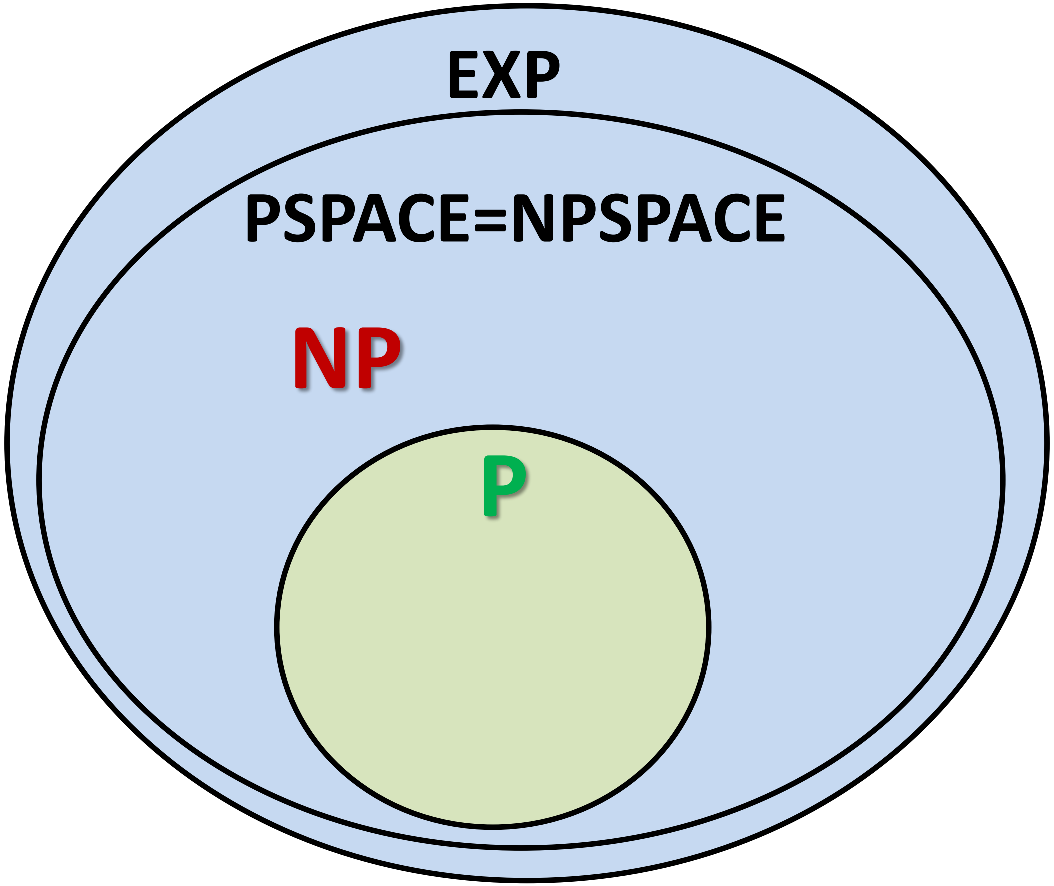
Definition: NPSPACE is the class of languages decidable in polynomial space on a single-tape (nondeterministic) TM

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

Relationships between complexity classes

1. P ⊆ NP ⊆ PSPACE ⊆ EXP

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



2. P ≠ EXP (Monday)

Which containments in (1) are proper?

Unknown!