Computability and Complexity

Lecture 13

Space Complexity
Savitch's Theorem
Space Complexity Class PSPACE

given by Jiri Srba

Time and Space Complexity

For practical solutions to computational problems

- available time, and
- available memory

are two main considerations.

We have already studied time complexity, now we will focus on space (memory) complexity.

Question:

How do we measure space complexity of a Turing machine?

Time and Space Complexity

For practical solutions to computational problems

- available time, and
- available memory

are two main considerations.

We have already studied time complexity, now we will focus on space (memory) complexity.

Question:

How do we measure space complexity of a Turing machine?

Answer:

The largest number of tape cells a Turing machine visits on all inputs of a given length n.

Definition of Space Complexity

Definition (Space Complexity of a TM)

Let M be a deterministic decider. The space complexity of M is a function

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

where $f(n)$ is the maximum number of tape cells that M scans on any input of length n. Then we say that M runs in space $f(n)$.

Definition of Space Complexity

Definition (Space Complexity of a TM)

Let M be a deterministic decider. The space complexity of M is a function

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

where $f(n)$ is the maximum number of tape cells that M scans on any input of length n. Then we say that M runs in space $f(n)$.

Definition (Space Complexity of a Nondeterministic TM)

Let M be a nondeterministic decider. The space complexity of M is a function

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

where $f(n)$ is the maximum number of tape cells that M scans on any branch of its computation on any input of length n. Then we say that M runs in space $f(n)$.

The Complexity Classes SPACE and NSPACE

Definition (Complexity Class SPACE($t(n)$))

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function.

$$\mathrm{SPACE}(t(n)) \stackrel{\mathrm{def}}{=} \{L(M) \mid M \text{ is a decider running in space } O(t(n))\}$$

The Complexity Classes SPACE and NSPACE

Definition (Complexity Class SPACE($t(n)$))

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function.

$$\mathrm{SPACE}(t(n)) \stackrel{\mathrm{def}}{=} \{L(M) \mid M \text{ is a decider running in space } O(t(n))\}$$

Definition (Complexity Class NSPACE($t(n)$))

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function.

$$\mathrm{NSPACE}(t(n)) \stackrel{\mathrm{def}}{=}$$
$$\{L(M) \mid M \text{ is a nondetermin. decider running in space } O(t(n))\}$$

The Complexity Classes SPACE and NSPACE

Definition (Complexity Class SPACE($t(n)$))

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function.

$\text{SPACE}(t(n)) \stackrel{\text{def}}{=} \{L(M) \mid M \text{ is a decider running in space } O(t(n))\}$

Definition (Complexity Class NSPACE($t(n)$))

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function.

$\text{NSPACE}(t(n)) \stackrel{\text{def}}{=}$
$\{L(M) \mid M \text{ is a nondetermin. decider running in space } O(t(n))\}$

In other words:

- SPACE($t(n)$) is the class (collection) of languages that are decidable by TMs running in space $O(t(n))$, and
- NSPACE($t(n)$) is the class (collection) of languages that are decidable by nondeterministic TMs running in space $O(t(n))$.

Example: *SAT* is Decidable in Linear Deterministic Space

Theorem

$SAT \in \text{SPACE}(n)$

Proof: Here is a deterministic decider M for SAT:

$M =$ "On input $\langle \phi \rangle$ where ϕ is a Boolean formula:

1. For every truth assignment to the variables in ϕ, evaluate ϕ on that assignment.
2. If ϕ gets ever evaluated to 1 then <u>accept</u>, else <u>reject</u>."

Example: SAT is Decidable in Linear Deterministic Space

Theorem

$SAT \in \text{SPACE}(n)$

Proof: Here is a deterministic decider M for SAT:

$M =$ "On input $\langle \phi \rangle$ where ϕ is a Boolean formula:

1. For every truth assignment to the variables in ϕ, evaluate ϕ on that assignment.
2. If ϕ gets ever evaluated to 1 then accept, else reject."

- Observe that M can reuse the space for the different truth assignments.
- M runs in $O(n)$ space (number of variables is bounded by n).
- Btw. the time complexity of the machine M is exponential!

$\square$

Example: $\overline{ALL_{NFA}}$ is Decidable in Linear Nondeterm. Space

$$ALL_{NFA} \stackrel{\text{def}}{=} \{\langle A \rangle \mid \ A \text{ is an NFA such that } L(A) = \Sigma^*\}$$

Theorem

$\overline{ALL_{NFA}} \in \mathrm{NSPACE}(n)$

Example: $\overline{ALL_{NFA}}$ is Decidable in Linear Nondeterm. Space

$$ALL_{NFA} \stackrel{\text{def}}{=} \{\langle A\rangle \mid A \text{ is an NFA such that } L(A) = \Sigma^*\}$$

Theorem

$\overline{ALL_{NFA}} \in \mathrm{NSPACE}(n)$

Proof: We want to find a nondeterministic decider M running in linear space such that

M accepts $\langle A\rangle$ where A is NFA if and only if $L(A) \neq \Sigma^*$.

Example: $\overline{ALL_{NFA}}$ is Decidable in Linear Nondeterm. Space

$$ALL_{NFA} \stackrel{\text{def}}{=} \{\langle A \rangle \mid A \text{ is an NFA such that } L(A) = \Sigma^*\}$$

Theorem

$\overline{ALL_{NFA}} \in \mathrm{NSPACE}(n)$

Proof: We want to find a nondeterministic decider M running in linear space such that

M accepts $\langle A \rangle$ where A is NFA if and only if $L(A) \neq \Sigma^*$.

Idea: Use the power-set construction to determinize A into DFA A_{det} and accept iff A_{det} has some rejecting computation.

Problem: A_{det} might be exponentially larger than A.

Solution: Find nondeterministically a rejecting path in A_{det} while reusing the space (on-the-fly construction of a rejecting path).

A Nondeterministic Decider for $\overline{ALL_{NFA}}$ Using $O(n)$ Space

We want to construct M such that

M accepts $\langle A \rangle$ if and only if $L(A) \neq \Sigma^*$

$M =$ "On input $\langle A \rangle$ where A is an NFA with control states Q:

1. Let $s \subseteq Q$ be the initial state in A_{det}.
2. Repeat 2^m times where $m = |Q|$:
   a) If s consists only of rejecting states of A then accept.
   b) Nondeterministically select a transition $s \xrightarrow{a} s'$ in A_{det}.
   c) $s := s'$
3. Reject."

A Nondeterministic Decider for $\overline{ALL_{NFA}}$ Using $O(n)$ Space

We want to construct M such that

M accepts $\langle A \rangle$ if and only if $L(A) \neq \Sigma^*$

M = "On input $\langle A \rangle$ where A is an NFA with control states Q:

1. Let $s \subseteq Q$ be the initial state in A_{det}.
2. Repeat 2^m times where $m = |Q|$:
   a) If s consists only of rejecting states of A then <u>accept</u>.
   b) Nondeterministically select a transition $s \xrightarrow{a} s'$ in A_{det}.
   c) $s := s'$
3. <u>Reject.</u>"

Space requirements:

- To store $s, s' \subseteq Q$ we need $O(n)$ space.
- To store a counter value in the loop we need $O(n)$ space.
- Total: $O(n)$ of nondeterministic space.

Discussion

So now we know that

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

What can we say about ALL_{NFA}?

Discussion

So now we know that

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

What can we say about ALL_{NFA}?

- In order to complement the nondeterministic machine M from the proof above, we first need to determinize it.
- Determinization of TM means exponential slow-down in the running time.
- Does determinization imply that we will need also exponentially more memory?

Discussion

So now we know that

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

What can we say about ALL_{NFA}?

- In order to complement the nondeterministic machine M from the proof above, we first need to determinize it.
- Determinization of TM means exponential slow-down in the running time.
- Does determinization imply that we will need also exponentially more memory?

Answer (Savitch's Theorem):

For every nondeterministic TM there is an equivalent deterministic TM that uses only quadratically more space!

Savitch's Theorem

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function such that $t(n) \geq n$. Then

$$\mathsf{NSPACE}(t(n)) \subseteq \mathsf{SPACE}(t^2(n)) .$$

Savitch's Theorem

Savitch's Theorem

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function such that $t(n) \geq n$. Then

$$\mathsf{NSPACE}(t(n)) \subseteq \mathsf{SPACE}(t^2(n)) \ .$$

Proof:

- Let N be a nondeterministic TM using space $O(t(n))$.
- We want to find an equivalent deterministic TM M with space complexity $O(t^2(n))$.

Savitch's Theorem

Savitch's Theorem

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function such that $t(n) \geq n$. Then

$$\mathrm{NSPACE}(t(n)) \subseteq \mathrm{SPACE}(t^2(n)) \ .$$

Proof:

- Let N be a nondeterministic TM using space $O(t(n))$.
- We want to find an equivalent deterministic TM M with space complexity $O(t^2(n))$.

Algorithm CANYIELD(c_1, c_2, t) accepts iff configuration c_2 is reachable (in N) from c_1 in at most t steps.

N is using $O(t(n))$ space which means that it has at most $2^{dt(n)}$ different configurations for some constant d.

Savitch's Theorem

Savitch's Theorem

Let $t : \mathbb{N} \to \mathbb{R}^{>0}$ be a function such that $t(n) \geq n$. Then

$$\mathsf{NSPACE}(t(n)) \subseteq \mathsf{SPACE}(t^2(n)) \ .$$

Proof:

- Let N be a nondeterministic TM using space $O(t(n))$.
- We want to find an equivalent deterministic TM M with space complexity $O(t^2(n))$.

Algorithm CANYIELD(c_1, c_2, t) accepts iff configuration c_2 is reachable (in N) from c_1 in at most t steps.

N is using $O(t(n))$ space which means that it has at most $2^{dt(n)}$ different configurations for some constant d.

Now N accepts w iff CANYIELD$(c_{start}, c_{accept}, 2^{dt(n)})$ accepts.

Computing CANYIELD(c_1, c_2, t) Deterministically

CANYIELD $=$

"On input $\langle c_1, c_2, t\rangle$:

1. Case $t = 1$:

   If $c_1 = c_2$ or $c_1 \to c_2$ (in N) then accept, else reject.

2. Case $t > 1$:

   For all configurations c_m (in N) of length at most $t(n)$:

   Call CANYIELD($\langle c_1, c_m, t/2\rangle$).

   Call CANYIELD($\langle c_m, c_2, t/2\rangle$).

   If both calls accepted then accept.

3. Reject."

Computing CANYIELD(c_1, c_2, t) Deterministically

CANYIELD =
"On input $\langle c_1, c_2, t\rangle$:

1. Case $t = 1$:
   If $c_1 = c_2$ or $c_1 \to c_2$ (in N) then accept, else reject.
2. Case $t > 1$:
   For all configurations c_m (in N) of length at most $t(n)$:
   Call CANYIELD($\langle c_1, c_m, t/2\rangle$).
   Call CANYIELD($\langle c_m, c_2, t/2\rangle$).
   If both calls accepted then accept.
3. Reject."

- CANYIELD is implementable on a deter. TM using a stack.
- The initial call CANYIELD($c_{start}, c_{accept}, 2^{dt(n)}$) uses a stack of height $O(t(n))$.
- Every stack entry stores a configuration of size $O(t(n))$.

Total space used: $O(t(n)) * O(t(n)) = O(t^2(n))$. □

Definition

The class PSPACE is the class of languages decidable in polynomial space on deterministic Turing machine, i.e.,

$$\mathsf{PSPACE} \stackrel{\mathrm{def}}{=} \bigcup_{k \geq 0} \mathsf{SPACE}(n^k) \ .$$

Complexity Class PSPACE

Definition

The class PSPACE is the class of languages decidable in polynomial space on deterministic Turing machine, i.e.,

$$\mathsf{PSPACE} \stackrel{\text{def}}{=} \bigcup_{k \geq 0} \mathsf{SPACE}(n^k) \ .$$

Discussion:

- The class PSPACE is robust with respect to nondeterminism (the class remains the same even if we use nondeterministic TM in its definition). Hence PSPACE=NPSPACE.
- Because every language $L \in \mathsf{PSPACE}$ has a deterministic poly-space TM M, we can swap the accept and reject states in M and get a poly-space decider for $\overline{L}$, hence $\overline{L} \in \mathsf{PSPACE}$.

Complexity Class PSPACE

Definition

The class PSPACE is the class of languages decidable in polynomial space on deterministic Turing machine, i.e.,

$$\mathrm{PSPACE} \stackrel{\mathrm{def}}{=} \bigcup_{k \geq 0} \mathrm{SPACE}(n^k) \ .$$

Discussion:

- The class PSPACE is robust with respect to nondeterminism (the class remains the same even if we use nondeterministic TM in its definition). Hence PSPACE=NPSPACE.
- Because every language $L \in \mathrm{PSPACE}$ has a deterministic poly-space TM M, we can swap the accept and reject states in M and get a poly-space decider for $\overline{L}$, hence $\overline{L} \in \mathrm{PSPACE}$.

Example: $SAT, \overline{SAT}, ALL_{NFA}, \overline{ALL_{NFA}} \in \mathrm{PSPACE}$

Theorem

NP $\subseteq$ PSPACE

Proof: Let $L \in$ NP.

- Then there is nondeterministic decider M running in time $O(n^k)$ such that $L(M) = L$.
- Note that M can scan at most $O(n^k)$ tape cells and so $L \in \text{NSPACE}(n^k)$.
- Savitch's Theorem gives that $L \in \text{SPACE}(n^{2k})$.
- Hence $L \in$ PSPACE. $\square$

Properties of the Class PSPACE

Theorem

NP $\subseteq$ PSPACE

Proof: Let $L \in$ NP.

- Then there is nondeterministic decider M running in time $O(n^k)$ such that $L(M) = L$.
- Note that M can scan at most $O(n^k)$ tape cells and so $L \in \text{NSPACE}(n^k)$.
- Savitch's Theorem gives that $L \in \text{SPACE}(n^{2k})$.
- Hence $L \in$ PSPACE. $\square$

Theorem

co-NP $\subseteq$ PSPACE

Proof: Next tutorial. $\square$

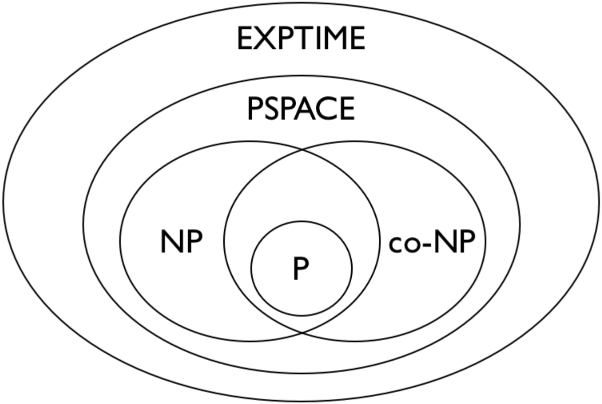
Theorem

PSPACE $\subseteq$ EXPTIME

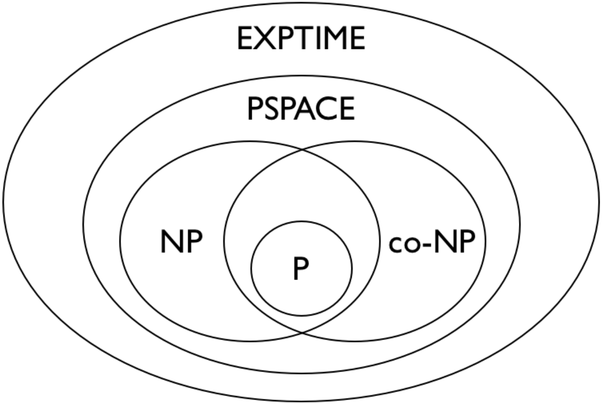
Proof: Let $L \in$ PSPACE.

- Then there is a deterministic decider M running in space $O(n^k)$ such that $L(M) = L$.
- Note that M has at most $2^{O(n^k)}$ different configurations.
- In any computation of the decider M none of the configurations can ever repeat (otherwise the machine loops).
- Hence the running time of M is bounded by $2^{O(n^k)}$ and so $L \in$ EXPTIME. $\square$

Overview of Time and Space Complexity Classes



Overview of Time and Space Complexity Classes



Remarks:

- We know that P $\neq$ EXPTIME, but
- the strictness of all the other inclusions is still open!

Exam Questions

- Definition of space complexity of deterministic and nondeterministic TMs.
- Definitions of SPACE, NSPACE and PSPACE.
- Savitch's Theorem.
- Basic properties of the class PSPACE (hierarchy of complexity classes).