Space Complexity

Chanathip Namprempre

Computer Science
Reed College

Outline

Outline

1. Definitions
2. Examples
3. Relations among Classes

Space Complexity

Let $f : \mathrm{N} \to \mathrm{N}$.

Definition

Let M be a deterministic TM. We say that M runs in space $f(n)$ iff

$$f(n) = \text{the maximum number of tape cells that } M \text{ scans on any input of length } n\,.$$

Definition

Let N be a nondeterministic TM. We say that N runs in space $f(n)$ iff

$$f(n) = \text{the maximum number of tape cells that } M \text{ scans on any branch of its computation for any input of length } n\,.$$

Space Complexity Classes

Let $f : \mathrm{N} \to \mathrm{R}^+$.

Definition

$$\mathrm{SPACE}(f(n)) = \{L \mid L \text{ is a language decided by an } O(f(n)) \text{ space deterministic TM } \}.$$

$$\mathrm{NSPACE}(f(n)) = \{L \mid L \text{ is a language decided by an } O(f(n)) \text{ space nondeterministic TM } \}.$$

Space Complexity Classes

Let $n \in \mathrm{Z}^{+}$.

Definition

$$\mathrm{PSPACE} = \bigcup_{k} \mathrm{SPACE}(n^k)$$

$$\mathrm{NPSPACE} = \bigcup_{k} \mathrm{NSPACE}(n^k)$$

Outline

SAT $\in$ SPACE(n)

Theorem

SAT $\in$ SPACE(n)

Proof.

Define M as follows:

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

1. For each truth assignment to the variables $x_1, \ldots, x_m$ of ϕ:
2. Evaluate ϕ on that truth assignment.
3. If ϕ ever evaluated to 1, accept. Otherwise, reject."

□

So,

SAT $\in$ PSPACE .

SAT $\in$ SPACE(n)

Theorem

SAT $\in$ SPACE(n)

Proof.

Define M as follows:

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

1. For each truth assignment to the variables $x_1, \ldots, x_m$ of ϕ:
2. Evaluate ϕ on that truth assignment.
3. If ϕ ever evaluated to 1, accept. Otherwise, reject."

$\square$

So,

$$\text{SAT} \in \text{PSPACE} .$$

$\text{All}_{\text{NFA}} \in \text{co-NSPACE}(n)$

We define the following language:

$$\text{All}_{\text{NFA}} = \{\langle A \rangle \mid A \text{ is an NFA such that } L(A) = \Sigma^*\} .$$

Theorem

$\text{All}_{\text{NFA}} \in \text{co-NSPACE}(n)$

Proof Idea:

- Construct an NTM N that, on input an NFA A, non-deterministically guesses bit-by-bit an input w that A rejects.
- If q is the number of states of A, then N uses a q-bit array s to keep track of the subsets that A can be in upon processing the each bit of w; that is,

position s_i is 1 iff A lands on s_i upon processing the current input symbol.

$\mathrm{All}_{\mathrm{NFA}} \in \text{co-NSPACE}(n)$

We define the following language:

$$\mathrm{All}_{\mathrm{NFA}} = \{\langle A \rangle \mid A \text{ is an NFA such that } L(A) = \Sigma^*\} .$$

Theorem

$\mathrm{All}_{\mathrm{NFA}} \in \text{co-NSPACE}(n)$

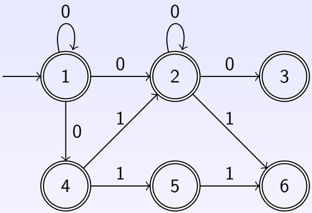
Proof Idea:

- Construct an NTM N that, on input an NFA A, non-deterministically guesses bit-by-bit an input w that A rejects.
- If q is the number of states of A, then N uses a q-bit array s to keep track of the subsets that A can be in upon processing the each bit of w; that is,

position s_i is 1 iff A lands on s_i upon processing the current input symbol.

$\mathrm{All}_{\mathrm{NFA}} \in$ co-NSPACE(n) (cont.)

Example: Suppose A is the following NFA.



Try simulating A on, say, 00. What does the tree look like?

$\mathrm{All}_{\mathrm{NFA}} \in$ co-NSPACE(n) (cont.)

Proof.

We define N for $\overline{\mathrm{All}_{\mathrm{NFA}}}$ so that N takes linear space as follows.
$N =$ "On input $\langle A \rangle$ where A is an NFA:

1. Let q be the number of states in A.
2. Let s be a q-bit array.
3. Repeat 2^q times.
    1. Keep track of the set of states A is in using s.
    2. Guess w one bit at a time.
    3. Simulate A on the current bit of w.
    4. If no states in s are accepting, then accept A.

□

N takes $O(n)$ space where $n = |A|$.

Outline

Relating time and space complexity

Theorem

For $t(n) \geq n$,

$$\mathrm{TIME}(t(n)) \subseteq \mathrm{SPACE}(t(n)) \,.$$

So,

$$\mathrm{P} \subseteq \mathrm{PSPACE}$$

Theorem

For $t(n) \geq n$,

$$\mathrm{SPACE}(t(n)) \subseteq \mathrm{TIME}(2^{O(t(n))}) = \bigcup_{c} \mathrm{TIME}(c^{t(n)}) \,.$$

Relating the class NP to space complexity

Theorem

$$\text{NP} \subseteq \text{PSPACE}$$

This is true because

1. SAT $\in$ PSPACE,
2. SAT is NP-Complete, and
3. If $A \leq_p B$ and $B \in$ PSPACE, then $A \in$ PSPACE.

$$\text{coNP} \subseteq \text{PSPACE}$$

because PSPACE $=$ coPSPACE.

Relating deterministic space to nondeterministic space

Clearly, for any function $f : \mathrm{N} \to \mathrm{R}^+$,

Lemma

$$\mathrm{SPACE}(f(n)) \subseteq \mathrm{NSPACE}(f(n))$$

Savitch's Theorem

Theorem

For any function $f : \mathrm{N} \to \mathrm{R}^+$ *where* $f(n) \geq n$*,*

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

Savitch's Theorem: Proof idea

Given an NTM N, we construct a deterministic TM M as follows:
$M =$ "On input w,

1. Let q be the number of states of N.
2. Let $d = |\Gamma|$ be the number of tape symbols for N.
3. Let $n = |w|$.
4. Let $t = q \times f(n) \times d^{f(n)}$
5. Output the result of $CY(c_{\text{start}}, c_{\text{accept}}, 2^t)$"

Savitch's Theorem: Proof (cont.)

CanYield = "On input c_1, c_2, g,

1. If $g = 1$, then test directly whether c_1 can yield c_2 or whether $c_1 = c_2$. Accept if either succeeds. Reject otherwise.
2. If $g > 1$, then for each configuration c_m
   1. Run CanYield$(c_1, c_m, g/2)$
   2. Run CanYield$(c_m, c_2, g/2)$
   3. If both of these accept, then accept.
3. Reject.

Analysis

- The total number of possible configurations is t.
- Each recursion level stores one config $= O(f(n))$ space.
- Number of levels $= \log t = O(f(n))$.
- Total space is $O(f^2(n))$.

Relations among classes

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} = \mathrm{NPSPACE} \subseteq \mathrm{EXP}$$

There is much we don't know. For example, we don't know whether $\mathrm{P} = \mathrm{PSPACE}$.