CS310 Automata Theory – 2016-2017

Nutan Limaye

Indian Institute of Technology, Bombay
nutan@cse.iitb.ac.in

Lecture 36: Effective computation
April 13, 2017

Space bounded Turing Machines

The Turing Machine model with space bounds

Space bounded Turing Machines

The Turing Machine model with space bounds

The input tape is assumed to be read-only.

Space bounded Turing Machines

The Turing Machine model with space bounds

- The input tape is assumed to be read-only.

- The space required to write down the input is not counted towards the space of the machine.

Space bounded Turing Machines

The Turing Machine model with space bounds

- The input tape is assumed to be read-only.

- The space required to write down the input is not counted towards the space of the machine.

- The output tape assumed to be write-only.

Space bounded Turing Machines

The Turing Machine model with space bounds

- The input tape is assumed to be read-only.
- The space required to write down the input is not counted towards the space of the machine.
- The output tape assumed to be write-only.
- The space required to write down the output is not counted towards the space of the machine.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class SPACE$(s(n))$

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class SPACE($s(n)$) if there exists a deterministic Turing machine M such that $\forall x \in \Sigma^*$,

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class $\mathrm{SPACE}(s(n))$ if there exists a deterministic Turing machine M such that $\forall x \in \Sigma^*$,

M halts on x using at most space $O(s(|x|))$

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class SPACE($s(n)$) if there exists a deterministic Turing machine M such that $\forall x \in \Sigma^*$,

M halts on x using at most space $O(s(|x|))$,

where $|x|$ indicates the length of x.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class SPACE$(s(n))$ if there exists a deterministic Turing machine M such that $\forall x \in \Sigma^*$,

M halts on x using at most space $O(s(|x|))$,

where $|x|$ indicates the length of x.

if $x \in L$ then M accepts x.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class SPACE($s(n)$) if there exists a deterministic Turing machine M such that $\forall x \in \Sigma^*$,

M halts on x using at most space $O(s(|x|))$,

where $|x|$ indicates the length of x.

if $x \in L$ then M accepts x.

if $x \notin L$ then M rejects x.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class $\mathrm{SPACE}(s(n))$ if there exists a deterministic Turing machine M such that $\forall x \in \Sigma^*$,

- M halts on x using at most space $O(s(|x|))$, where $|x|$ indicates the length of x.
- if $x \in L$ then M accepts x.
- if $x \notin L$ then M rejects x.

$$L = \mathrm{SPACE}(\log n)$$

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class $\mathrm{SPACE}(s(n))$ if there exists a deterministic Turing machine M such that $\forall x \in \Sigma^*$,

- M halts on x using at most space $O(s(|x|))$,
- where $|x|$ indicates the length of x.
- if $x \in L$ then M accepts x.
- if $x \notin L$ then M rejects x.

$$L = \mathrm{SPACE}(\log n)$$

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

Examples of languages in Log

Min $= \{(w_1, w_2, \ldots, w_n, i) \mid w_i$ is the minimum among $w_1 \ldots w_n\}$.

Examples of languages in Log

Min = $\{(w_1, w_2, \ldots, w_n, i) \mid w_i$ is the minimum among $w_1 \ldots w_n\}$.

Deg = $\{(G = (V, E), d, i) \mid v_i$ has degree $d\}$.

Examples of languages in Log

Min $= \{(w_1, w_2, \ldots, w_n, i) \mid w_i$ is the minimum among $w_1 \ldots w_n\}$.

Deg $= \{(G = (V, E), d, i) \mid v_i$ has degree $d\}$.

ADD $= \{(u, v, i) \mid i$th bit of $u + v$ is $1\}$.

Examples of languages in Log

Min $= \{(w_1, w_2, \ldots, w_n, i) \mid w_i$ is the minimum among $w_1 \ldots w_n\}$.

Deg $= \{(G = (V, E), d, i) \mid v_i$ has degree $d\}$.

ADD $= \{(u, v, i) \mid i$th bit of $u + v$ is $1\}$.

Verify-SAT

$= \{(\phi, a) \mid a = a_1, a_2, \ldots, a_n$ is an assignment satisfying $\phi\}$.

Examples of languages in Log

Min $= \{(w_1, w_2, \ldots, w_n, i) \mid w_i$ is the minimum among $w_1 \ldots w_n\}$.

Deg $= \{(G = (V, E), d, i) \mid v_i$ has degree $d\}$.

ADD $= \{(u, v, i) \mid i$th bit of $u + v$ is $1\}$.

Verify-SAT

$= \{(\phi, a) \mid a = a_1, a_2, \ldots, a_n$ is an assignment satisfying $\phi\}$.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class $\mathrm{NSPACE}(s(n))$

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class NSPACE$(s(n))$ if there exists a non-deterministic Turing machine M such that $\forall x \in \Sigma^*$,

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class NSPACE($s(n)$) if there exists a non-deterministic Turing machine M such that $\forall x \in \Sigma^*$,

> M halts on x using at most space $O(s(|x|))$ on any run of the machine

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class NSPACE$(s(n))$ if there exists a non-deterministic Turing machine M such that $\forall x \in \Sigma^*$,

> M halts on x using at most space $O(s(|x|))$ on any run of the machine,
>
> where $|x|$ indicates the length of x.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class NSPACE($s(n)$) if there exists a non-deterministic Turing machine M such that $\forall x \in \Sigma^*$,

- M halts on x using at most space $O(s(|x|))$ on any run of the machine,
- where $|x|$ indicates the length of x.
- if $x \in L$ then there exists an accepting run of M on x.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class NSPACE$(s(n))$ if there exists a non-deterministic Turing machine M such that $\forall x \in \Sigma^*$,

- M halts on x using at most space $O(s(|x|))$ on any run of the machine,
- where $|x|$ indicates the length of x.
- if $x \in L$ then there exists an accepting run of M on x.
- if $x \notin L$ then M rejects x on all the runs.

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class $\mathrm{NSPACE}(s(n))$ if there exists a non-deterministic Turing machine M such that $\forall x \in \Sigma^*$,

- M halts on x using at most space $O(s(|x|))$ on any run of the machine,
- where $|x|$ indicates the length of x.
- if $x \in L$ then there exists an accepting run of M on x.
- if $x \notin L$ then M rejects x on all the runs.

$$\mathrm{NL} = \mathrm{NSPACE}(\log n)$$

Space complexity and complexity classes

Let $s : \mathbb{N} \to \mathbb{N}$.

Definition

A language $L \subseteq \Sigma^*$ is said to be in class NSPACE$(s(n))$ if there exists a non-deterministic Turing machine M such that $\forall x \in \Sigma^*$,

> M halts on x using at most space $O(s(|x|))$ on any run of the machine,
>
> where $|x|$ indicates the length of x.
>
> if $x \in L$ then there exists an accepting run of M on x.
>
> if $x \notin L$ then M rejects x on all the runs.

$$\mathrm{NL} = \mathrm{NSPACE}(\log n)$$

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

Example of a language in NL

$$\text{Reach} = \{(G = (V, E), s, t) \mid \text{there is a path in } G \text{ from } s \text{ to } t\}$$

Example of a language in NL

$$\text{Reach} = \{(G = (V, E), s, t) \mid \text{there is a path in } G \text{ from } s \text{ to } t\}$$

current $\leftarrow s$; count $\leftarrow 0$;

Example of a language in NL

$$\text{Reach} = \{(G = (V, E), s, t) \mid \text{there is a path in } G \text{ from } s \text{ to } t\}$$

current $\leftarrow s$; count $\leftarrow 0$;

while count $< n + 1$ or current $\neq t$;

Example of a language in NL

$$\text{Reach} = \{(G = (V, E), s, t) \mid \text{there is a path in } G \text{ from } s \text{ to } t\}$$

current $\leftarrow s$; count $\leftarrow 0$;

while count $< n + 1$ or current $\neq t$;

{

next $\leftarrow$ non-det. guess a vertex from neighbors of current;

Example of a language in NL

$$\text{Reach} = \{(G = (V, E), s, t) \mid \text{there is a path in } G \text{ from } s \text{ to } t\}$$

current $\leftarrow s$; count $\leftarrow 0$;

while count $< n + 1$ or current $\neq t$;

{

next $\leftarrow$ non-det. guess a vertex from neighbors of current;

current $\leftarrow$ next;

Example of a language in NL

$$\text{Reach} = \{(G = (V, E), s, t) \mid \text{there is a path in } G \text{ from } s \text{ to } t\}$$

current $\leftarrow s$; count $\leftarrow 0$;

while count $< n + 1$ or current $\neq t$;

{

next $\leftarrow$ non-det. guess a vertex from neighbors of current;

current $\leftarrow$ next;

count $++$;

Example of a language in NL

$$\text{Reach} = \{(G = (V, E), s, t) \mid \text{there is a path in } G \text{ from } s \text{ to } t\}$$

current $\leftarrow s$; count $\leftarrow 0$;

while count $< n + 1$ or current $\neq t$;

{

  next $\leftarrow$ non-det. guess a vertex from neighbors of current;
  current $\leftarrow$ next;
  count $++$;

}

if current $= t$ then accept;

else reject;

Example of a language in NL

$$\text{Reach} = \{(G = (V, E), s, t) \mid \text{there is a path in } G \text{ from } s \text{ to } t\}$$

current $\leftarrow s$; count $\leftarrow 0$;

while count $< n + 1$ or current $\neq t$;

{

  next $\leftarrow$ non-det. guess a vertex from neighbors of current;
  current $\leftarrow$ next;
  count $++$;

}

if current $= t$ then accept;

else reject;

NL is contained in P

Configurations of a non-deterministic space bounded machine.

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

data: the working space bits

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

data: the working space bits (uses $O(s(n))$ bits)

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

data: the working space bits (uses $O(s(n))$ bits)

S_M: machine related information

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

data: the working space bits (uses $O(s(n))$ bits)

S_M: machine related information (Q, δ)

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

data: the working space bits (uses $O(s(n))$ bits)

S_M: machine related information (Q, δ) (uses $O(1)$ bits)

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

data: the working space bits (uses $O(s(n))$ bits)

S_M: machine related information (Q, δ) (uses $O(1)$ bits)

A typical configuration $\langle \text{index, data}, S_M \rangle$

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

data: the working space bits (uses $O(s(n))$ bits)

S_M: machine related information (Q, δ) (uses $O(1)$ bits)

A typical configuration $\langle \text{index, data}, S_M \rangle$

Let $\mathcal{C}_M$ be the set of all possible configuration of M.

Let C_0 be the initial configuration.

NL is contained in P

Configurations of a non-deterministic space bounded machine.

Configuration of a space bounded Turing machine M

index: input head position (uses $O(\log n)$ bits)

data: the working space bits (uses $O(s(n))$ bits)

S_M: machine related information (Q, δ) (uses $O(1)$ bits)

A typical configuration $\langle \text{index, data}, S_M \rangle$

Let $\mathcal{C}_M$ be the set of all possible configuration of M.

Let C_0 be the initial configuration.

Let C_{acc} be the accepting configuration.

NL is contained in P

Definition

Let L be a language in $NSPACE(\mathrm{s(n)})$ with TM M.

NL is contained in P

Definition

Let L be a language in $NSPACE(\text{s(n)})$ with TM M. Let C, C' be two configurations in $\mathcal{C}_M$.

NL is contained in P

Definition

Let L be a language in $NSPACE(\mathrm{s(n)})$ with TM M. Let C, C' be two configurations in $\mathcal{C}_M$. We say that a configuration C yields C' on input w

NL is contained in P

Definition

Let L be a language in $NSPACE$(s(n)) with TM M. Let C, C' be two configurations in $\mathcal{C}_M$. We say that a configuration C yields C' on input w if the machine M in one step goes from C to C' on input w.

Configurations Graph of M on input w.

NL is contained in P

Definition

Let L be a language in $NSPACE(\mathrm{s(n)})$ with TM M. Let C, C' be two configurations in $\mathcal{C}_M$. We say that a configuration C yields C' on input w if the machine M in one step goes from C to C' on input w.

Configurations Graph of M on input w.

$$\text{Let } \mathcal{E}_{M,w} = \{(C, C') \mid C, C' \in \mathcal{C}_{\mathcal{M}} \text{ and } C \text{ yields } C' \text{ on input } w\}$$

NL is contained in P

Definition

Let L be a language in $NSPACE$(s(n)) with TM M. Let C, C' be two configurations in $\mathcal{C}_M$. We say that a configuration C yields C' on input w if the machine M in one step goes from C to C' on input w.

Configurations Graph of M on input w.

Let $\mathcal{E}_{M,w} = \{(C, C') \mid C, C' \in \mathcal{C}_{\mathcal{M}} \text{ and } C \text{ yields } C' \text{ on input } w\}$

Let $\mathcal{G}_{M,w} = (\mathcal{C}_M, \mathcal{E}_{M,w})$

NL is contained in P

Definition

Let L be a language in $NSPACE$(s(n)) with TM M. Let C, C' be two configurations in $\mathcal{C}_M$. We say that a configuration C yields C' on input w if the machine M in one step goes from C to C' on input w.

Configurations Graph of M on input w.

Let $\mathcal{E}_{M,w} = \{(C, C') \mid C, C' \in \mathcal{C}_{\mathcal{M}}$ and C yields C' on input $w\}$

Let $\mathcal{G}_{M,w} = (\mathcal{C}_M, \mathcal{E}_{M,w})$

Let $\mathcal{G}_{M,w}$ be the configuration graph of M on w.

NL is contained in P

Definition

Let L be a language in $NSPACE$(s(n)) with TM M. Let C, C' be two configurations in $\mathcal{C}_M$. We say that a configuration C yields C' on input w if the machine M in one step goes from C to C' on input w.

Configurations Graph of M on input w.

Let $\mathcal{E}_{M,w} = \{(C, C') \mid C, C' \in \mathcal{C}_{\mathcal{M}}$ and C yields C' on input $w\}$

Let $\mathcal{G}_{M,w} = (\mathcal{C}_M, \mathcal{E}_{M,w})$

Let $\mathcal{G}_{M,w}$ be the configuration graph of M on w.

NL is contained in P

Theorem

If L is in $NSPACE(s(n))$ then L is in $TIME(2^{O(s(n))})$.

NL is contained in P

Theorem

If L is in $NSPACE(s(n))$ then L is in $TIME(2^{O(s(n))})$.

We know that $L \in NSPACE(\mathrm{s(n)})$. Let M be the machine.

NL is contained in P

Theorem

If L is in $NSPACE(s(n))$ then L is in $TIME(2^{O(s(n))})$.

We know that $L \in NSPACE(\mathrm{s(n)})$. Let M be the machine.

First note that, $w \in L$ if and only if C_{acc} is reachable from C_0 in $\mathcal{G}_{M,w}$.

NL is contained in P

Theorem

If L is in $NSPACE(s(n))$ then L is in $TIME(2^{O(s(n))})$.

We know that $L \in NSPACE(\mathrm{s(n)})$. Let M be the machine.

First note that, $w \in L$ if and only if C_{acc} is reachable from C_0 in $\mathcal{G}_{M,w}$.

On any input w, the graph $\mathcal{G}_{M,w}$ can be computed in time $TIME(2^{O(s(n))})$.

NL is contained in P

Theorem

If L is in NSPACE(s(n)) then L is in $TIME(2^{O(s(n))})$.

We know that $L \in NSPACE(\mathrm{s(n)})$. Let M be the machine.

First note that, $w \in L$ if and only if C_{acc} is reachable from C_0 in $\mathcal{G}_{M,w}$.

On any input w, the graph $\mathcal{G}_{M,w}$ can be computed in time $TIME(2^{O(s(n))})$.

$|\mathcal{C}_M| = 2^{O(s(n))}$.

NL is contained in P

Theorem

If L is in NSPACE(s(n)) then L is in $TIME(2^{O(s(n))})$.

We know that $L \in NSPACE(\mathrm{s(n)})$. Let M be the machine.

First note that, $w \in L$ if and only if C_{acc} is reachable from C_0 in $\mathcal{G}_{M,w}$.

On any input w, the graph $\mathcal{G}_{M,w}$ can be computed in time $TIME(2^{O(s(n))})$.

- $|\mathcal{C}_M| = 2^{O(s(n))}$.
- Given C, C', checking whether $(C, C') \in \mathcal{E}_{M,w}$ or not is checkable in time $2^{O(s(n))}$.

NL is contained in P

Theorem

If L is in $NSPACE(s(n))$ then L is in $TIME(2^{O(s(n))})$.

We know that $L \in NSPACE(\mathrm{s(n)})$. Let M be the machine.

First note that, $w \in L$ if and only if C_{acc} is reachable from C_0 in $\mathcal{G}_{M,w}$.

On any input w, the graph $\mathcal{G}_{M,w}$ can be computed in time $TIME(2^{O(s(n))})$.
Checking whether C_{acc} is reachable from C_0 can be checked in time $2^{O(s(n))}$.

NL is contained in P

Theorem

If L is in $NSPACE(s(n))$ then L is in $TIME(2^{O(s(n))})$.

We know that $L \in NSPACE(\mathrm{s(n)})$. Let M be the machine.

First note that, $w \in L$ if and only if C_{acc} is reachable from C_0 in $\mathcal{G}_{M,w}$.

On any input w, the graph $\mathcal{G}_{M,w}$ can be computed in time $TIME(2^{O(s(n))})$.
Checking whether C_{acc} is reachable from C_0 can be checked in time $2^{O(s(n))}$.

Reachability in a graph of size $2^{O(s(n))}$.

NL is contained in P

Theorem

If L is in $NSPACE(s(n))$ then L is in $TIME(2^{O(s(n))})$.

We know that $L \in NSPACE(\mathrm{s(n)})$. Let M be the machine.

First note that, $w \in L$ if and only if C_{acc} is reachable from C_0 in $\mathcal{G}_{M,w}$.

On any input w, the graph $\mathcal{G}_{M,w}$ can be computed in time $TIME(2^{O(s(n))})$.
Checking whether C_{acc} is reachable from C_0 can be checked in time $2^{O(s(n))}$.

Reachability in a graph of size $2^{O(s(n))}$.

Corollary

NL is contained in P.