University of California, Los Angeles
CS 281 Computability and Complexity

Instructor: Alexander Sherstov
Scribe: Junkai Zhang
Date: February 7, 2024

LECTURE
9

Space Complexity

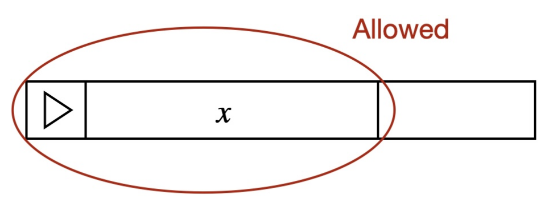
In modern computers, each machine employs a hierarchical memory architecture, including components such as RAM and flash disk, to augment available virtual memory. However, memory capacity remains finite, leading to potential out-of-memory errors when executing certain programs. Thus, alongside time, space represents a critical dimension for any computational framework. In this class, we focus on understanding the role of space, i.e., memory in computation. We started from space alone and then investigated the relationship between space and time.

9.1 Basic Notions

We first introduce the concept of *space-bounded computation*, which restricts the number of tape cells a TM can use during its computation. The following definition characterizes the space-bounded computation of a TM/NTM.

Definition 9.1. Let $S : \mathbb{N} \to \mathbb{N}$. TM/NTM runs in space $S(n)$ if for calls on any input x,

- input tape head stays on input, and



- other heads reach at most $O(S(|x|))$ cells.

Equipped with Definition 9.1, we can formally define the space complexity of languages.

Definition 9.2. Let $S : \mathbb{N} \to \mathbb{N}$ and $L \subseteq \{0,1\}^*$. We say that $L \in \mathrm{SPACE}(S(n))$ (or $L \in \mathrm{NSPACE}(S(n))$) if L is decided by a TM (or NTM) in space $c \cdot S(n)$ for some $c > 0$.

Analogous to time complexity, we will restrict our attention to space bounds S that are space-constructible:

Definition 9.3. $S : \mathbb{N} \to \mathbb{N}$ is space constructible if

- $S(n) \geq \log n$
- some TM maps $\underbrace{11 \cdots 111}_{n} \to \underbrace{11 \cdots 111}_{S(n)}$ in space $O(S(n))$.

Intuitively, if S is space-constructible, then the machine can know the space restriction of its computation. Definition 9.3 is quite a mild requirement since most functions of interest, like functions in Example 9.4, satisfy this requirement.

Example 9.4. Work through these examples

$$\lceil \log n \rceil, n, n^2, 2^n, \cdots$$

The reason it makes sense to consider functions $S(n) < n$, such as $S(n) = \lceil \log n \rceil$, is because the input tape is separated from the work tape. This is in contrast to time-bounded computation as the machine needs at least time greater than n to read through the input. However, Definition 9.3 do require $S(n) > \log n$ since we need the machine can remember the position of its input tape head.

We can develop the spatial analogs to the time complexity classes P and NP, known as PSPACE and NPSPACE.

Definition 9.5. For TM, the PSPACE is defined as

$$\text{PSPACE} = \text{SPACE}(n) \cup \text{SPACE}(n^2) \cup \text{SPACE}(n^3) \cup \cdots .$$

Similarly, for NTM, we can define NPSPACE:

$$\text{NPSPACE} = \text{NSPACE}(n) \cup \text{NSPACE}(n^2) \cup \text{NSPACE}(n^3) \cup \cdots .$$

The PSPACE is in fact a quite large complexity class, for example, we have NP $\subseteq$ PSPACE. We can show that SAT $\in$ PSPACE by describing a TM that decides SAT in linear space. The space uses the linear space to cycle through all 2^n possible assignments in order. Once an assignment has been checked it can be erased from the work tape.

Intuitively, we have $\text{SPACE}(n) \subsetneq \text{SPACE}(n^2) \subsetneq \text{SPACE}(n^3) \subsetneq \cdots$ since $n^{k-1}/n^k \to 0$. The next theorem reveals this kind of inner hierarchy of space complexity.

Theorem 9.6. *Fix space constructible function $S, s : \mathbb{N} \to \mathbb{N}$ where*

$$\frac{s(n)}{S(n)} \to 0 \text{ as } n \to \infty.$$

Then $\text{SPACE}(s(n)) \subsetneq \text{SPACE}(S(n))$.

Proof. Let's consider the **Turing Machine D**:

Input: x
Until $S(|x|)$ space exceeded or time exceeded

Simulate M_x on x
If M_x has halted
  Output $\neg M_x(x)$
Else
  Output 0

Then by definition, D halts within $S(n)$ steps and hence

$$L(D) \in \text{SPACE}(S(n)).$$

We prove that $L(D) \notin \text{SPACE}(s(n))$ by contradiction. Assume that there is some TM M such that TM M, given any input $x \in \{0,1\}^*$, works within $c|x|$ space for some constant c, and outputs $D(x)$.

Easy to check universal TM $\mathcal{U}$ simulate TM M on x with $c'c|x|$ space. There is some number n_0 such that $S(n) > c'cs(n)$ for every $n \geq n_0$. Let x be a string representing the machine M whose length is at least n_0. Then, $M(x)$ will obtain the output $D(x)$ within $S(x)$ space, but by definition of D, we have $D(x) = \neg M(x)$. This leads to the contradiction. □

9.2 Configurations

In this section, we introduce the notion of the configuration graph of a TM. This concept is quite useful to characterize the relationship between time and space. The following is the formal definition of a configuration of TM.

DEFINITION 9.7. A configuration of Turing machine or non-deterministic Turing machine M consists of:

- current state of M,
- current contents of each tape,
- head locations.

Then, all configurations consist of a configuration Graph G_M.

DEFINITION 9.8. Configuration Graph G_M of TM/NTM M is

- infinite directed graph,
- vertices = configurations,
- edge (C, C') if and only if C leads to C' in one step.

In addition, G_M has degree 1 for TMs M, and degree ≤ 2 for NTMs M.

Let $C_{M,x}^{\text{START}}$ be the starting configuration of M on input x.

DEFINITION 9.9. $G_{M,x}$ = subgraph of G_M reachable from $C_{M,x}^{\text{START}}$. Then we have $G_M = \bigcup_x G_{M,x}$.

Convention. Unique accepting configuration in $G_{M,x}$ is denoted by $C_{M,x}^{\text{ACCEPT}}$. $C_{M,x}^{\text{ACCEPT}}$ must satisfy the following conditions:

- All heads are on $\triangleright$.
- Work tape blank.
- Output tape:

$\triangleright$	1	

By definition, M accepts x iff there exists a directed path in $C_{M,x}$ from $C_{M,x}^{\text{START}}$ to $C_{M,x}^{\text{ACCEPT}}$. The following definition gives a way to represent the vertex.

Definition 9.10. Vertex of $G_{M,x}$ encoded by

- Current contents + head locations for each r/w tape. $(k-1)S(|x|)\log|\Gamma| = O(S(|x|))$.
- State of M. $\log|Q| = O(1)$.
- Input head location. $\log(|x|+1) \leq S(|x|+1)$.

In addition, we have the following immediate properties of $G_{M,x}$.

Proposition 9.11. *Fix a TM/NTM M that runs in SPACE S(n). Then,*

- *Every vertex of TM $G_{M,x}$ can be represent by length $O(S(n))$ Boolean strings. In particular, $G_{M,x}$ has at most $2^{c \cdot S(x)}$ nodes.*
- *The $C_{M,x}^{START}$ and $C_{M,x}^{ACCEPT}$ can be constructed in time $O(S(n))$ on input x*
- *Can check if $(C, C') \in G_{M,x}$ in time $O(S(|x|) + |x|)$ on input x, C, C'.*

9.3 Time vs Space

In this section, we characterize the relationship between space and time by Theorems 9.12 and 9.14. The next theorem shows that NP is indeed included in PSPACE.

Theorem 9.12. *For every space constructible $S : \mathbb{N} \to \mathbb{N}$*

$$DTIME(S(n)) \subseteq NTIME(S(n)) \subseteq SPACE(S(n))$$

Proof. The first DTIME($S(n)$) $\subseteq$ NTIME($S(n)$) is trivial, For the later part, the machine can just try out every non-deterministic guess and reuse space across guess. $\square$

Theorem 9.12 also indicates the following corollary.

Corollary 9.13.

$$SAT \in PSPACE$$

After obtaining the lower bound of SPACE($S(n)$), we shows that SPACE($S(n)$) (and even NSPACE(S(n))) is in DTIME($2^{O(S(n))}$).

THEOREM 9.14. *For any space-constructible* $S : \mathbb{N} \to \mathbb{N}$,

$$\mathit{SPACE}(S(n)) \subseteq \mathit{NSPACE}(S(n)) \subseteq \mathit{DTIME}(2^{O(S(n))})$$

Proof. We just need to show NSPACE$(S(n)) \subseteq$ DTIME$(2^{O(S(n))})$. We can construct $G_{M,x}$ in time $2^{O(S(|x|))}$ and check if $C_{M,x}^{\text{ACCEPT}}$ is reachable from $C_{M,x}^{\text{START}}$ using the standard breadth-first search (linear in the size of the graph) algorithm for connectivity. ∎

Theorems 9.12 and 9.14 gives the following inclusion sequence:

$$\text{P} \subseteq \text{NP} \subseteq \text{PSPACE} \subseteq \text{NPSPACE} \subseteq \text{EXP}.$$

This relationship is illustrated in Figure 9.3.

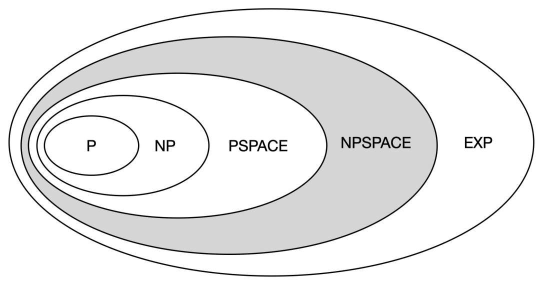


FIGURE 9.1: The relationship of time and space complexity classes.

Later we will see that indeed PSPACE=NPSPACE. However, whether P = PSPACE is still open.

9.4 Deterministic vs Non-deterministic Space

Unlike P=NP is a still problem, the space complexity classes for deterministic TM and non-deterministic TM are not significantly different. Savitch's Theorem reveals that non-deterministic TM can be indeed approximated by deterministic TM with only a square of space utilization. This can further indicate PSPACE = NPSPACE.

THEOREM 9.15 (Savitch's Theorem). *For every space-constructible* $S : \mathbb{N} \to \mathbb{N}$,

$$\mathit{NSPACE}(S(n)) \subseteq \mathit{SPACE}(S(n)^2).$$

Proof. Fix space constructible function $S(n)$, NTM M $G_{M,x}$ has $2^{c \cdot S(|x|)}$ vertices for every $x \in \{0,1\}^*$. $M(x) = 1$ is equivalent to that $C_{m,x}^{\text{ACCEPT}}$ is reachable from $C_{M,x}^{\text{START}}$ in $2^{c \cdot S(|x|)}$ steps. Consider the algorithm `LEADSTO`, which output YES iff C leads to C' within l steps in $G_{M,x}$. The intuitive idea is illustrated in Figure 9.4.

Input: configurations C, C', integer l
If $l = 1$
  If $(C, C') \in G_{M,x}$ (this is checkable in time $O(S(|x|))$, so also in space $O(S(|x|))$).
    Output "YES"

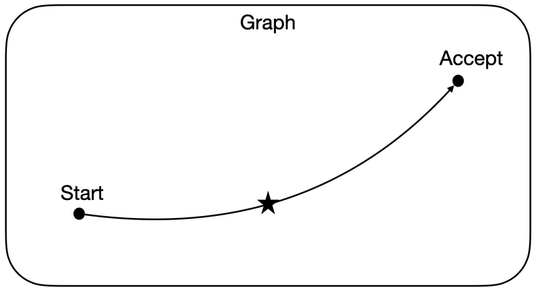


FIGURE 9.2: The illustration of LEADSTO algorithm.

Else
Output "No"
Else
For each C''
If `LEADSTO`(C, C'', $\lceil \frac{l}{2} \rceil$) and `LEADSTO`($C''$, C', $\lfloor \frac{l}{2} \rfloor$)
Output "YES"
Else
Output "NO"

Therefore, $M(x) = \texttt{LEADSTO}(C_{M,x}^{\text{START}}, C_{M,x}^{\text{ACCEPT}}, 2^{c \cdot S(|x|)})$. For the space utilization of `LEADSTO`, The height of the recursion stack is $c \cdot S(|x|)$ and the space per recursion call is $O(S(|x|))$ space. Thus `LEADSTO` uses $O(S(|x|)^2)$ space in total. ∎

Savitch's Theorem indicates the following corollary.

COROLLARY 9.16. *The PSPACE and NPSPACE are indeed equivalent.*

$$PSPACE = NPSPACE.$$

Challenge Problem

Let's prove that there is an oracle O that can make P and NP to be identical.

THEOREM 9.17. *There is $O \subseteq \{0,1\}^*$ s.t.*

$$P^O = NP^O.$$

Intuitively, we would like to make the oracle O strong enough so that it can bridge the potential gap between P and NP.

Proof. Consider $O = \{(M, x, \underbrace{11 \cdots 111}_{m}) : \text{M accepts x in time} \leq 2^m\}$. O is EXP-complete, which is somehow the strongest oracle we can imagine. We want to show $P^O = NP^O$ via the following sequence:

$$\underbrace{EXP \subseteq P^O \subseteq NP^O}_{\text{trivial}} \subseteq EXP.$$

We just need to prove the last inclusion. The exponential time is enough to try out all non-deterministic guesses and to answer all queries. ∎