

CSE200: Complexity theory

Time and space hierarchies

Shachar Lovett

January 16, 2020

1 Time hierarchy

We next look more closely at the resources used by a TM. For convenience, we will assume that any string $x \in \{0, 1\}^*$ corresponds to a description of some Turing machine $x = \langle M \rangle$. Say, if x doesn't compile, then M halts on the first step and returns 0. The important thing is that for every TM M there exists at least one description $\langle M \rangle$.

Definition 1.1 (Time complexity). *Let $T : \mathbb{N} \rightarrow \mathbb{N}$ be a time bound. We say a TM has time complexity T , if for any $n \in \mathbb{N}$ and any input $x \in \{0, 1\}^n$, the TM halts after making at most $T(n)$ many steps. A function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ is computed in time T if there is a TM computing it running in time T . We define*

$$\text{TIME}(T(n)) = \{f : \{0, 1\}^* \rightarrow \{0, 1\}^* \text{ computable in time } cT(n) \text{ for some } c \geq 1\}.$$

We will always assume that the function T is computable, otherwise we will run into funky situations. Some examples are:

- Polynomial time: $P = \cup_{c \geq 1} \text{TIME}(n^c)$.
- Single exponential time: $E = \cup_{c \geq 1} \text{TIME}(2^{cn})$.
- Exponential time: $\text{EXP} = \cup_{c \geq 1} \text{TIME}(2^{n^c})$.

Some examples of known algorithms and where they lie:

- Graph connectivity is in P . Given a graph on n vertices, we can run BFS to find whether two vertices are connected. This takes time $O(n^2)$, which is polynomial.
- Satisfiability is in E . Given a boolean formula on n variables, we can try all 2^n assignments to see if it is satisfiable.
- Factoring is in E . The best algorithms to factor an n -bit number run in time about $2^{n^{1/3}}$.

Claim 1.2 (Robustness of P). *P is invariant to changing the TM model. For example, for the model variants we have discussed:*

- *Changing the alphabet replace time $T(n)$ with $O(T(n))$.*
- *Reducing the number of tapes increases the time from $T(n)$ to $O(T(n)^2)$ (and this can even be improved to $O(T \log T)$).*
- *Replacing TM with RAM with standard TM increases time from $T(n)$ to $O(T(n)^2)$ (and this can even be improved to $O(T \log T)$).*

We will consider from now on TMs with a single work tape and some fixed alphabet. The first question that we aim to tackle is whether giving TM more time make them more powerful. For example, is EXP strictly more powerful than P? The next theorem shows that this is indeed the case.

Theorem 1.3 (Time hierarchy). *Let $T : \mathbb{N} \rightarrow \mathbb{N}$ be a function with $T(n) \geq n$, and let $g : \mathbb{N} \rightarrow \mathbb{R}$ be any function such that $g(n) = \omega(1)$ (e.g. $\log \log \log(n)$). Assume furthermore that the function T can be computed in time at most $T(n)$. Then*

$$\text{TIME}(T(n)) \subsetneq \text{TIME}(T^2(n)g(n)).$$

We will not prove this, but in fact this can be strengthened to $\text{TIME}(T(n)) \subsetneq \text{TIME}(T(n) \log T(n)g(n))$.

Proof. We define a function $f : \{0, 1\}^* \rightarrow \{0, 1\}$ that will be in $\text{TIME}(T^2g)$ but not in $\text{TIME}(T)$. To define it, fix the time function $\tilde{T}(n)$ to be $T(n)\sqrt{g(n)}$, which grows faster than $T(n)$. On input x of length $n = |x|$, we define $f(x)$ as follows:

- Interpret x as a description of a TM $x = \langle M \rangle$.
- Simulate the run $M(\langle M \rangle)$ for at most $\tilde{T}(n)$ steps.
- If M halts and outputs 0 then set $f(x) = 1$. Otherwise set $f(x) = 0$.
- If M doesn't halt within $\tilde{T}(n)$ steps then set $f(x) = 0$.

We first argue that $f \notin \text{TIME}(T(n))$. Assume it was. Then there exists a TM M for which $f(x) = M(x)$ for all $x \in \{0, 1\}^*$ and such that M halts after at most $cT(|x|)$ steps for some constant c . As $T = o(\tilde{T})$, we have that $cT(n) \leq \tilde{T}(n)$ for any large enough n , namely $n \geq n_0$. We may assume that $|\langle M \rangle| \geq n_0$ by padding the TM description. We now do a case analysis:

- If $f(\langle M \rangle) = 0$ then $M(\langle M \rangle) = 1$. Note that $M(\langle M \rangle)$ halts in at most $\tilde{T}(|\langle M \rangle|)$ steps by our assumption.
- If $f(\langle M \rangle) = 1$ then $M(\langle M \rangle) = 0$.

Thus $\langle M \rangle \in \{0, 1\}^*$ satisfies $f(\langle M \rangle) \neq M(\langle M \rangle)$, contradicting our assumption.

We next argue that $f \in \text{TIME}(T^2(n)g(n))$ (and in fact $f \in \text{TIME}(T \log Tg)$). To show this, we need to build a TM computing f which runs in time $O(T(n)^2g(n))$. Let U be a universal Turing machine with a step counter. That is, U takes inputs $\langle M \rangle, x$ and has on its work tape:

- Counter for the number of steps.
- Current state of M .
- Current work memory of M (with a special symbol for the location of the reading head of M).

Consider a simulation of running M on input x . First, we initialize the counter with $\tilde{T}(|x|)$. Then, to simulate a single step of M , U needs to

- Read the symbol under M 's simulated reading head.
- Scan the input tape for the transition step corresponding to the current state.
- Update the current state.
- Decrease the counter.

When the counter reaches zero U halts. To simulate a single step of M , if the tape of U has k symbols then it takes $O(k)$ steps for U to simulate M . Consider now simulating $M(\langle M \rangle)$ with $n = |\langle M \rangle|$. The length of the work tape of U is at most $O(\tilde{T}(n))$, so U halts after at most $O(\tilde{T}^2(n))$ steps. Hence $f \in \text{TIME}(T(n)^2g(n))$. \square

Corollary 1.4. $P \subsetneq EXP$.

2 Space hierarchy

Similar to time, we can measure the space (in other words, memory) that Turing machines use.

Definition 2.1 (Space complexity). *Let $S : \mathbb{N} \rightarrow \mathbb{N}$ be a space bound. We say a TM has space complexity S , if for any $n \in \mathbb{N}$ and any input $x \in \{0, 1\}^n$, the TM halts while using only at most the first $S(n)$ symbols of the work tape. A function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ is computed in space S if there is a TM computing it running in space S . We define*

$$\text{SPACE}(S(n)) = \{f : \{0, 1\}^* \rightarrow \{0, 1\}^* \text{ computable in space } cS(n) \text{ for some } c \geq 1\}.$$

As before, we will always assume that the function S is computable. Some examples are:

- Polynomial space: $\text{PSPACE} = \cup_{c \geq 1} \text{SPACE}(n^c)$.

- Logarithmic space: $\text{LOGSPACE} = \text{SPACE}(\log n)$.

Some examples of well-known algorithms and where they lie:

- BFS or DFS use linear space, and hence are in PSPACE .
- Satisfiability needs to store the assignment, which uses linear space, and hence is in PSPACE .
- Adding two numbers can be done in LOGSPACE . That is, adding two n -bit numbers, where the output is an $(n + 1)$ -bit number, can be done using $O(\log n)$ work memory. We would dig deeper into log-space later in the class to understand the intricacies involved.

Theorem 2.2 (Space hierarchy). *Let $S : \mathbb{N} \rightarrow \mathbb{N}$ be a function where the function S can be computed in space at most $O(S(n))$. Assume that $S(n) = \Omega(\log n)$. Then*

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

To recall, a function $S'(n) = \omega(S(n))$ is $S'(n)$ grows strictly faster than $S(n)$. That is, $\lim_{n \rightarrow \infty} \frac{S(n)}{S'(n)} = 0$.

Proof. Let $S' : \mathbb{N} \rightarrow \mathbb{N}$ be any function which can be computed in space at most $O(S'(n))$ and satisfies $S'(n) = \omega(S(n))$. Let $f : \{0, 1\}^* \rightarrow \{0, 1\}$ be the following function. On input x of length $n = |x|$,

- Interpret x as a description of a TM $x = \langle M \rangle$.
- Simulate the run $M(\langle M \rangle)$ using at most $S'(n)$ space in the work and output.
- If the reading head in either work or output takes moves beyond the first $S'(n)$ cells, halt and output 0.
- If the TM makes more than $T'(n) = 2^{S'(n)}$ steps, halt and return 0.
- If M halts and outputs 0 then $f(x) = 1$. Otherwise $f(x) = 0$.

We first argue that $f \notin \text{SPACE}(S(n))$. Assume it was. Then there exists a TM M for which $f(x) = M(x)$ for all $x \in \{0, 1\}^*$ and such that M runs in space $cS(n)$ and time $cT(n)$ where $T(n) = 2^{cS(n)}$, for some constant $c \geq 1$, and halts. As in the Time Hierarchy Theorem, assume we have padded the description of the TM M such that $S'(|\langle M \rangle|) > cS(n)$. So, the simulation lets M run its course. Then

- If $f(\langle M \rangle) = 0$ then $M(\langle M \rangle) = 1$.
- If $f(\langle M \rangle) = 1$ then $M(\langle M \rangle) = 0$.

We next argue that $f \in \text{SPACE}(S'(n))$. To show this, we need to build a TM computing f which runs in space $O(S'(n))$. Let U be a universal Turing machine with a protection against memory overflow. That is, U takes inputs $\langle M \rangle, x$; it first computes $S'(|x|)$ and marks the first disallowed cell in the work and output tape with a special symbol. After that symbol it stores the current state of M . In addition, it has a counter for the number of steps it makes. It then simulates $M(x)$. The space that U uses is $O(S'(n))$ in the work tape, plus additionally $O(\log n)$ to remember the state of M and other points. \square

Corollary 2.3. $\text{SPACE}(\log n) \subsetneq \text{SPACE}(\log^2(n))$.

3 Time vs space

There are some connections between time complexity and space complexity.

Claim 3.1. $\text{TIME}(T(n)) \subset \text{SPACE}(T(n))$.

Proof. In $T(n)$ steps you can move the head at most $T(n)$ times. \square

Claim 3.2. $\text{SPACE}(S(n)) \subset \text{TIME}(n \cdot 2^{O(S(n))})$.

Proof. Let M be a TM with states Q and tape alphabet Γ . The total number of configurations it has, if at most s cells of work tape has been used, is $Q \cdot n \cdot s \cdot |\Gamma|^s = O(n2^{O(s)})$ for $|Q|, |\Gamma| = O(1)$. If a TM terminates, it can reach each configuration at most once. Thus this puts a limit on its runtime of $O(n \cdot 2^{O(S(n))})$. \square