Hierarchy Theorems

Peter Mayr

Computability Theory, November 17, 2023

So far we proved the following inclusions:

$$\mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXPTIME} \subseteq \mathrm{EXPSPACE}$$

Question

Which are proper?

An efficient Universal Turing Machine

Theorem

There exists a DTM U such that $U(e, x) = M_e(x)$ for all e, x. Further, if M_e halts on x in t steps, then U halts on (e, x) in $O(t \log t)$ steps.

Proof.

U uses the tape alphabet $(0, 1, \sqcup)$ and multiple tapes:

1. input
2. work tape 1: simulating M_e's k work tapes in k parallel tracks (cells with numbers $\equiv_k 0, \ldots, k-1$, respectively).
3. work tape 2: description of M_e
4. work tape 3: current state of M_e
5. scratch
6. output

Problem: M_e's k tape heads can move independently left/right but U's head on tape 1 moves over all tracks in parallel.

1. **Idea:** U scans all of tape 1 to reach M_e's current tape positions. A single step of M_e needs $O(t)$ steps of U.

   track 1 machines
   track 2 rise

   running time of U is $O(t^2)$

2. **Idea:** Shift the contents of the k tracks under the head of tape 1.

   ← machines
   rise →

   To avoid the same factor $O(t)$ as in 1., add **buffer zones** on the tracks (amortized analysis).

   - Let □ denote an extra blank symbol.
   - Split U's bi-infinite* tracks into zones

   * same running as one-sided infinite

before L_2 . . . L_1 L_0 0 R_0 R_1 . . . R_2

- Initially each R_i, L_i is half-full with $\sqcup$s and $\boxtimes$s.
- Position 0 always holds a non-$\Box$.
- During the computation R_i, L_i each contain either $0, 2^i$ or 2^{i+1} non-$\Box$s, together 2^{i+1}.

How to shift a track left:

- Let i be smallest such that R_i is not empty (all $\Box$).
- Put the leftmost non $\Box$s of R_i at 0.
- Use the next $2^i - 1$ non-$\Box$s of R_i to fill half of each $R_0, \ldots, R_{i-1}$.
- Do the dual on the left: for $j = i-1, \ldots, 0$, move the 2^{j+1} symbols from L_j to fill half of L_{j+1}.
- Put the update of the original symbol at 0 into L_1.

Runtime for one shift at index i:

- Linear the size of the involved zones $L_i, \ldots, R_i$, i.e. $O(\sum_{j=0}^{i} 2^{j+1}) = O(2^i)$.

After a shift at index i:

- Zones $L_{i-1}, \ldots, R_{i-1}$ are all half full.
- Needs $\geq 2^{i-1}$ left shifts until $R_0, \ldots, R_{i-1}$ are empty again ($\geq 2^{i-1}$ right shifts until $L_0, \ldots, L_{i-1}$ are empty).
- Hence $\leq 1/2^i$ of the shifts have index i.
- Since U does $\leq t$ shifts on a track, the highest possible index is $\log_2 t$ (only zones up to $R_{\log t}, L_{\log t}$ are used).
  The total run time for shifting is

$$O(k \sum_{i=0}^{\log t} \frac{t}{2^i} 2^i) = O(t \log t).$$

The other tapes can be updated in constant time. Hence U on input (e, x) runs in time $O(t \log t)$. □

Universal TM with time bound

By adding a **time counter** to U above, we obtain a universal TM that on input e, x, T runs $O(T \log T)$ steps and outputs $M_e(x)$ iff M_e halts on x in $\leq T$ steps. (HW)

Time constructible functions

In simulations we often want to determine whether a TM has run for $t(n)$ steps without accruing any overhead in time complexity (i.e. in $O(t(n))$ time).

Definition

$t\colon \mathbb{N} \to \mathbb{N}$ is **time constructible** if $t(n) \geq n$ and there exists a DTM with input & output tape that on input $x \in \{0,1\}^*$ outputs $t(|x|)$ in binary in time $O(t(|x|))$.

Example

$n, n \log n, n^k, 2^n \ldots$ are time constructible.

DTM to count the length of the input x in binary:

- The counter increments by 1 (requiring $O(\log n)$ steps) for every input position.
- Running time is $O(n \log n)$.

It follows that $n \log n$ can be computed in binary in $O(n \log n)$ time.

Little o-notation

Definition

For $f, g\colon \mathbb{N} \to \mathbb{R}^+$ we say $f = o(g)$ (read f is little-o of g) if

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

Intuitively: g grows much faster than f.

Note

- $f = o(g) \Rightarrow f = O(g)$, but not conversely.
- $f \neq o(f)$

Separating time complexity classes

Time Hierarchy Theorem

Let f, g be time constructible and $f(n) \log f(n) = o(g(n))$.
Then $\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$.

Proof by diagonalization.

Define a DTM D such that on input x of length n:

1. Compute $g(n)$ (possible in time $O(g(n))$).
2. Run the Universal TM U from the previous Theorem to simulate M_x on x for $g(n)$ steps.
3. If M_x accepts x in $g(n)$ steps, D rejects; else D accepts.
4. Then $L(\mathrm{D})$ can be decided in $O(g(n))$.

Claim: $L(\mathrm{D})$ is not decidable in $O(f(n)$ time.

- Suppose otherwise, that there exists M that decides $L(\mathrm{D})$ in $O(f(n))$ time.
- By the previous Theorem, there exists a constant c such that U can simulate M on any input x in time $cf(n)\log f(n)$.
- $\exists N\ \forall n \geq N: \ cf(n)\log f(n) < g(n)$ and D's simulation runs to completion if M's input has length $\geq N$.
- Let x be an encoding of M with $|x| \geq N$ (exists since M has infinitely many encodings).
  Then x is accepted by M iff x is rejected by D.
- Hence $L(\mathrm{D}) \neq L(\mathrm{M})$.

$\square$

Consequences of the Time Hierarchy Theorem

Corollary

$\mathrm{DTIME}(n^k) \neq \mathrm{DTIME}(n^\ell)$ if $0 \leq k < \ell$.

Corollary

$\mathrm{P} \neq \mathrm{EXPTIME}$

Proof.

HW $\square$

Space constructible functions

In simulations we often want to fix $s(n)$ space on the working tape before starting the actual computation without accruing any overhead in space complexity.

Definition

$s\colon \mathbb{N} \to \mathbb{N}$ is **space constructible** if $s(n) \geq \log n$ and there exists a DTM with input & output tape that on input $x \in \{0,1\}^*$ outputs $s(|x|)$ in binary in space $O(s(|x|))$.

Note

$s(n)$ is space constructible iff there exists $N \in \mathbb{N}$ and a DTM with input tape that on any input of length $n \geq N$ uses and marks off $s(n)$ cells on its working tape and halts

Example

$\log n, n^k, 2^n$ are space constructible.

E.g. $\log_2 n$ space is constructed by counting the length n of the input in binary.

Separating space complexity classes

Space Hierarchy Theorem

Let r, s be space constructible and $r(n) = o(s(n))$.
Then $\mathrm{SPACE}(r(n)) \subsetneq \mathrm{SPACE}(s(n))$.

Proof.

Use a Universal TM for space bounded computation.
(Like Time Bounded Universal TM but easier since multiple tapes are not an issue) $\square$

Consequences of the Space Hierarchy Theorem

Corollary

$\text{SPACE}(n^k) \neq \text{SPACE}(n^\ell)$ if $0 \leq k < \ell$.

Corollary

$\text{NL} \neq \text{PSPACE}$

Corollary

$\text{PSPACE} \neq \text{EXPSPACE}$

Proof.

HW □