Hierarchy theorems

Evgenij Thorstensen

V18

Comparing functions

To prove results like $\mathsf{TIME}(f(n)) \subset \mathsf{TIME}(g(n))$, we need a stronger notion of one function growing faster than another.

$f \in O(g)$ just means that g bounds f. In particular, $f \in O(f)$.

We will use little-oh, which has the desired property.

Two definitions of o

Let $f, g : \mathbb{N} \to \mathbb{R}^+$. We write $f(n) = o(g(n))$, or $f \in o(g)$, if and only if

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

or, equivalently, if and only if there exists a threshold c_0 such that for every $\varepsilon \in \mathbb{R}^+$,

$$f(n) \leqslant \varepsilon \cdot g(n)$$

holds for all $n \geqslant c_0$.

The distance between f and g increases, and the increase is non-constant.

Some examples and observations

The constant can be moved: $\epsilon \cdot f(n) \leqslant g(n)$ is equivalent.

To show $f \notin o(g)$, suffices to find a constant ε violating the definition.

$\frac{1}{3x} \neq o(\frac{1}{x})$, since $\varepsilon = \frac{1}{3}$ cancels the 3.

$\log n \in o(n)$, also $n^k \in o(n^{k+1})$, also $n^k \in o(2^{\epsilon n})$

$f \in o(g)$ essentially means that $f < g$ asymptotically.

A note on functions

In principle, arbitrary functions can be used to bound time or space complexity.

However, if a TM M is time-bounded by a function $f(n)$ not computable in time $O(f(n))$, simulating M in $O(f(n))$ time given f becomes difficult.

We want well-behaved functions — they should be nondecreasing, and easy to compute.

Time and space constructability

We define a function $f(n)$ to be time (respectively, space) constructable if a DTM can output $f(n)$ in time (respectively, space) $O(f(n))$ when started on the string 1^n.

In other words, if the machine has an input of size n, it can compute the output value $f(n)$ (in binary) in time $O(f(n))$.

We can effectively diagonalise over machines with such complexity functions.

All the usual functions we have seen are time and space constructible.

Diagonalization

By diagonalization, we will prove that the space and time hierarchies do not collapse.

This will separate some classes, but only those who are asymptotically bigger.

This is the best we can do, many years later.

Thanks to Baker, Gill, and Solovay, this can't separate anything between P and PSPACE.

Space hierarchy theorem

Let f, g be space constructible functions such that $f \in o(g)$. Then there exists a language A decidable in $O(g(n))$ space but not in $O(f(n))$ space.

Corollary: If $f \in o(g)$, then $\mathsf{SPACE}(f(n)) \subset \mathsf{SPACE}(g(n))$.

We will present the deterministic proof; by invoking Immerman-Szelepcsényi the proof goes through for NSPACE too.

Space hierarchy, consequences

$\mathsf{SPACE}(o(f(n))) \subset \mathsf{SPACE}(f(n))$. Therefore:

- $\mathsf{L} \subset \mathsf{SPACE}(n) \subset \mathsf{PSPACE}$, since $\log n \in o(n)$ and $n \in o(n^2)$.
- $\mathsf{NL} \subset \mathsf{PSPACE}$, same argument.
- $\mathsf{PSPACE} \neq \mathsf{SPACE}(n^k)$ for any k — it is unbounded.
- $\mathsf{SPACE}(n^k) \subset \mathsf{SPACE}(n^{k+1})$, so PSPACE is dense.

Proof outline, space

We are going to define a language of turing machine descriptions loosely described as “DTM M did not accept itself”.

This is classic diagonalization. We will simulate a machine on itself using correct amount of space, and accept if it rejects.

The only issue of note: Small-number behaviour. M may use more than $f(n)$ space on small n.

Can be fixed by padding.

Algorithm

Given $g(n)$, on input w we do:

- Compute $g(|w|)$, mark off that much tape. If this bound is exceeded, reject.
- If w is not of the form $\langle M \rangle \# 0^*$, reject (malformed input).
- Simulate M on w. If this runs for more than $2^{g(n)}$ steps, reject.
- Accept if and only if M rejected.

Algorithm

Given $g(n)$, on input w we do:

- Compute $g(|w|)$, mark off that much tape. If this bound is exceeded, reject.
- If w is not of the form $\langle M \rangle\#0^*$, reject (malformed input).
- Simulate M on w. If this runs for more than $2^{g(n)}$ steps, reject.
- Accept if and only if M rejected.

Clearly this is decidable in $O(g(n))$ space. Need to prove that it is not decidable in $o(g(n))$ space.

Assume it is. Then there is a decider M using space $f \in o(g(n))$. Then there is a threshold c_0 after which $c \cdot f(n) \leqslant g(n)$ for any c.

This means that for some string $\langle M \rangle\#0^*$ longer than c_0, we managed to simulate M on itself. However, our language differs from the one M defines on this string.

Time hierarchy theorems

Two different ones, one for DTMs and one for NTMs.

Let f, g be time constructible functions such that $f(n) \in o(\frac{g(n)}{\log g(n)})$. Then there exists a language A decidable by DTM in $O(g(n))$ time but not in $O(f(n))$ time.

Restated, let f be time constructible. There is a language A decidable by DTM in time $O(f(n) \cdot \log f(n))$ but not in time $o(f(n))$.

Let f, g be time constructible functions such that $f(n+1) \in o(g(n))$. Then there exists a language A decidable by NTM in $O(g(n))$ time but not in $O(f(n))$ time.

NTM theorem not in Sipser, but of interest.

Time hierarchy, consequences

- $\mathsf{P} \subset \mathsf{EXP}$ and $\mathsf{NP} \subset \mathsf{NEXP}$
- $\mathsf{P} \neq \mathsf{TIME}(n^k)$ for any k
- $\mathsf{TIME}(n^k) \subset \mathsf{TIME}(n^{k+1})$, so P is also dense.

The extra log factor isn't too big of a deal.

Time hierarchy, problems

Time-bounded simulation on single-tape DTMs is very time-inefficient!

"easy" to do with cubic overhead, but that's rather high.

Will need some tricks to avoid seeking back and forth on the tape.

We will use tracks, aka striping.

Algorithm, time hierarchy

Given $g(n)$, on input w we do:

- Compute $\frac{g(|w|)}{\log g(|w|)}$, and store it. Decrement for every step of simulation, reject if we hit 0.
- If w is not of the form $\langle M \rangle \# 0^*$, reject (malformed input).
- Simulate M on w.
- Accept if and only if M rejected.

Algorithm, time hierarchy

Given $g(n)$, on input w we do:

- Compute $\frac{g(|w|)}{\log g(|w|)}$, and store it. Decrement for every step of simulation, reject if we hit 0.
- If w is not of the form $\langle M \rangle \# 0^*$, reject (malformed input).
- Simulate M on w.
- Accept if and only if M rejected.

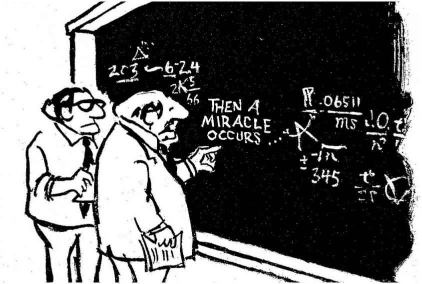

Figure: I think you need to be more explicit in this step

Simulating with logarithmic overhead

Simulating M requires us to keep track of head position, state, and tape contents.

We will stripe our tape: every third cell belongs to a track.

Track 1 has the tape of M, with head position marked. Track 2 has the current state and a copy of M's transition function.

Track 3 has the counter.

These have to be close together — can't seek too much. Every time we move the head on track 1, we move the information on track 2 to be close, and also move the counter.

Analysis

Track 2 depends on M, not on its input, so this is a constant-size overhead.

The counter on track 3 has size $O(\log g(n))$, so updating and moving it takes $O(\log g(n))$ steps.

Since the machine M we simulate runs in time $O(\frac{g(n)}{\log g(n)})$, the whole thing runs in time $O(g(n))$.

The rest of the proof is exactly like the space hierarchy theorem.

Time hierarchy for NTMs

Why can't we do what we did for DTMs?

Time hierarchy for NTMs

Why can't we do what we did for DTMs?

Right, because no closure under negation.

There is a way around it, but it involves some clever constructions that are not part of the curriculum.