Time Hierarchy theorem.

Question: How much more time results in a strict measure of computational power?

Theorem: Let $t_1(n), t_2(n)$ be "time constructible" functions

such that $t_1(n) \log t_1(n) \in o(t_2(n))$.

Then $DTIME(t_1(n)) \subsetneq DTIME(t_2(n))$.

Time constructible functions

↳ should be able to ~~constr~~ compute $t_2(n)$ in $t_2(n)$ time

⇒ A fn $t(n): \mathbb{N} \to \mathbb{N}$ is called time constructable if there ∃ a TM such that on i/p 1^n, it o/p a binary of $t(n)$ in $O(t(n))$ steps.

Space constructible function

A fn $f: \mathbb{N} \to \mathbb{N}$, where $f(n) \geq \log n$. is space constructible if ∃ a DTM running in $O(f(n))$ space that when given 1^n as input write the binary representation of $f(n)$ and halt.

Space Hierarchy theorem

For any space constructible function $f: \mathbb{N} \to \mathbb{N}$, and a function $g(n)$ such that

$$g(n) = o(f(n)),$$

there is a language $A \in DSPACE(f(n))$ but $A \notin DSPACE(g(n))$.

Note: (Related to Time Hierarchy).

A Turing M/C that runs in time $t(n)$ takes $O(t(n) \log t(n))$ time to be simulated by a universal turing M/C

(Stearns 1966)