Time Complexity

The time complexity of a decision problem is how much time is needed to answer questions in it.

Example: Equal 0's and 1's

Consider a TM that decides the language of strings with equal 0's and 1's. One idea is to cross off first symbol; say it's 1. Then go and find the first 0 and cross it off. Then come back to the first uncrossed symbol. Etc. The TM accepts if all symbols crossed off.

If input string has length n, then time taken could be as high as about $n^2/4$ ($n/2$ iterations with average length $n/2$).

Worst-Case Analysis

We analyze the *worst-case*: what matters is whether *every* instance can be done quickly. A TM ***runs in time*** $T(n)$ if, for all inputs w, it halts within $T(|w|)$ steps.

n always denotes the input length.

Order Notation

Constants do not matter. In fact, one cannot care about constants, since exact time depends heavily on the atomic operations.

We say the above TM runs in $O(n^2)$ time, or "***order*** n^2" time, meaning there is c such that the TM runs in at most cn^2 steps for *any* input of length n.

The order or ***big-O*** notation says how the worst-case running time grows as n gets large.

Example Continued

Can one improve on above TM? Well, if there are two tapes:

Make a copy of the input. Then run the heads along both strings, matching the `0`'s and `1`'s. Such a machine accepts in ***linear*** time ($O(n)$ steps).

We Assume a Multitape TM

To measure time complexity

we assume a multitape TM.

Example. The language PALIN of all palindromes is decidable in $O(n)$ time.

$\mathcal{P}$ and Polynomial Time

The collection of all problems that can be solved in polynomial time is called $\mathcal{P}$.

That is, language L is in $\mathcal{P}$ if there is k and TM that decides L that runs in time $O(n^k)$.

The Complexity Class $\mathcal{P}$

A set of related languages is a ***(complexity) class***.

The class $\mathcal{P}$ roughly captures the collection of practically solvable problems. Or at least that is the conventional wisdom.

Polynomially Related Models

Definition. *Two models of computation are* ***polynomially related*** *if there is polynomial* p *such that: if language decidable in time* $T(n)$ *on one model, then it is decidable in time* $p(T(n))$ *on the other.*

Example. A k-tape TM and a 1-tape TM are polynomially related. (Proof by simulation.)

Example. A TM is polynomially related to a "standard computer".

The Definition of $\mathcal{P}$ is Robust

It follows that $\mathcal{P}$ is the same whether we think of TMs or of normal computer algorithms.

Example: Sorting

Sorting. Note that we are still dealing with languages, or equivalently, yes–no questions. So the question for sorting is how to check whether an input list is sorted. This is in $\mathcal{P}$.

Example: Boolean Formula

A ***boolean formula*** consists of variables and negated variables (collectively ***literals***), and the operations "and" and "or". We use $\vee$ for "or", $\wedge$ for "and", and $\bar{x}$ for "not x". E.g.

$$x \wedge (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y})$$

An ***assignment*** is setting each variable to either TRUE or FALSE. For example, if x and y are TRUE and z is FALSE, the above formula is FALSE.

True Boolean Formulas

$\texttt{TRUEBF} = \{ \langle \phi, \psi \rangle : \phi$ is boolean formula made $\texttt{TRUE}$ by assignment $\psi \}$

A boolean formula can be parsed and evaluated like a normal arithmetic expression using a single stack. Thus TRUEBF is in $\mathcal{P}$.

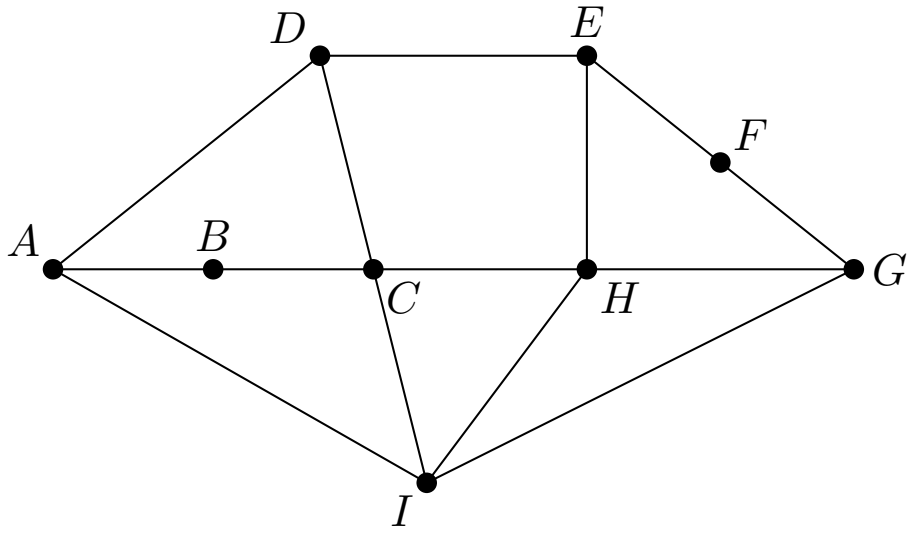
Graphs

A ***graph*** consists of a set of ***nodes*** and a set of ***edges***. Each edge joins two nodes; two nodes joined by an edge are ***adjacent***.

A ***path*** is a sequence of edges leading from one node to another. A ***hamiltonian path*** from node a to node b is a path from a to b that visits every node exactly once.

Example Graph

Here there is a path from every node to every other node. There is a hamiltonian path from A to I (visiting nodes in alphabetical order), but no hamiltonian path from A to C.



Example: PATH

$\texttt{PATH} = \{ \langle G, a, b \rangle : G$ is graph with path from a to $b \}$.

This is in $\mathcal{P}$. (E.g. use breadth-first search.)

Example: A Context-Free Language

We did not do the analysis, but the running time of the earlier CYK algorithm is $O(n^3)$. Thus every context-free language is in $\mathcal{P}$.

Practice

Show that the acceptance problem
$A_{nfa} = \{ \langle M, w \rangle : M$ is NFA accepting $w \}$ is in $\mathcal{P}$.

Solution to Practice

One cannot convert the NFA to a DFA, since the DFA can be exponentially large. Instead, use the idea that motivated the subset construction.

Simulate the NFA by keeping track of at each step what states the NFA could be in. Updating this information for a single symbol takes polynomial time.

Summary

The time complexity of a TM is the time taken as a function of the input length n in the worst case. The class $\mathcal{P}$ is the set of all languages that are decidable by a TM running in polynomial time. Examples of languages in $\mathcal{P}$ include `TRUEBF`, `PATH`, and any context-free language.