XI. Computational Complexity

Yuxi Fu

BASICS, Shanghai Jiao Tong University

Mathematic proofs, like computations, are energy consuming business. There are mathematical theorems, and computational tasks, that require more resources than what are available to us.

A mathematical proof, and a program as well, must be short enough to be practically relevant.

In real world we are not only interested in if a problem is solvable but also how costly it is to solve the problem.

Synopsis

1. Complexity Measure
2. Blum's Speedup Theorem
3. Gap Theorem

1. Complexity Measure

Time appears to be the best criterion for the amount of energy necessary to execute a program.

Time Function

Given a program P, the time function $t_P^{(n)}(\widetilde{x})$ is defined by

$$t_P^{(n)}(\widetilde{x}) = \mu t.(P(\widetilde{x}) \text{ terminates in } t \text{ steps}).$$

We write $t_e^{(n)}(\widetilde{x})$ for $t_{P_e}^{(n)}(\widetilde{x})$.

Remark:

- The time function is computable since'$P(\widetilde{x})\downarrow$ in t steps' is a primitive recursive predicate.
- We shall omit the superscript (n) when $n = 1$.

Time Function

Fact.

(i) $dom(t_e^{(n)}(\widetilde{x})) = dom(\phi_e^{(n)}(\widetilde{x}))$ for all n, e.

(ii) The predicate "$t_e^{(n)}(\widetilde{x}) \leq y$" is decidable for all n.

Blum Complexity Measure

(ϕ_i, Φ_i) is a Blum complexity measure if the following hold:

- $\Phi_i(x)$ is defined iff $\phi_i(x)$ is defined.
- $\Phi_i(x) \leq n$ is decidable.

Manuel Blum

- A Machine-Independent Theory of the Complexity of Recursive Functions. J. ACM 14:322-336, 1967.

We are mainly interested in asymptotic behaviour of time function.

A predicate $M(n)$ holds almost everywhere (a.e.) if $M(n)$ holds for all but finitely many natural numbers n.

Theorem. Given a total computable function $b(x)$, there is a total computable function $f(x)$ with range $\{0, 1\}$ such that $t_e(x) > b(x)$ a.e. for every index e of $f(x)$.

There are arbitrarily complex time functions.

Define f so that it differs from every function in the sequence

$$\phi_{j_0}, \phi_{j_1}, \phi_{j_2}, \ldots, \phi_{j_k}, \ldots. \tag{1}$$

A function ϕ_i appears in (1) if $\phi_i(m) \leq b(m)$ for infinitely many m.

Suppose $f(0), f(1), \ldots, f(n-1)$ have been defined. Let i_n be

$$\mu i.(i \leq n,\ t_i(n) \leq b(n),\ i \text{ is not yet defined})$$

and let $f(n)$ be defined by

$$f(n) \;=\; \begin{cases} 1, & \text{if } i_n \text{ is defined and } \phi_{i_n}(n) = 0, \\ 0, & \text{otherwise.} \end{cases}$$

1. If $\phi_e = f$, then $e \neq i_n$ whenever i_n is defined.
2. If $t_i(m) \leq b(m)$ for infinitely many m then $i = i_n$ for some n.

2. Blum's Speedup Theorem

Is there always a best program that solves a problem?

Blum's Speedup Theorem says that the answer is negative.

Speedup Theorem

Lemma. Let r be a total computable function. There is a total computable function f such that given any program P_i for f we can construct effectively a program P_j with the following properties:

1. ϕ_j is total and $\phi_j(x) = f(x)$ a.e..
2. $r(t_j(x)) < t_i(x)$ a.e..

Proof of the Lemma

By S-m-n Theorem, there is a total computable function s st

$$\phi_{s(e,u)}(x) \simeq \phi_e^{(2)}(u,x). \tag{2}$$

We will construct some e using Recursion Theorem such that

$$\phi_e^{(2)}(u,x) \simeq g(e,u,x), \tag{3}$$

where $g(e,u,x)$ is obtained by the diagonalisation construction described on the next slide.

Proof of the Lemma

Suppose some finite sets of canceled indices $C_{e,u,0}, \ldots, C_{e,u,x-1}$ have been defined.

If $t_{s(e,i+1)}(x)$ is defined for all $i \in \{u, \ldots, x-1\}$, then let

$$C_{e,u,x} = \left\{ i \mid u \leq i < x,\ t_i(x) \leq r(t_{s(e,i+1)}(x)) \right\} \setminus \bigcup_{y<x} C_{e,u,y};$$

otherwise let $C_{e,u,x}$ be undefined.

Clearly $C_{e,u,x}$ is computable, and if $i \in C_{e,u,x}$ then $\phi_i(x)\downarrow$.

Now $g(e,u,x)$ is defined by

$$g(e,u,x) \quad = \quad \begin{cases} 1 + \max\{\phi_i(x) \mid i \in C_{e,u,x}\}, & \text{if } C_{e,u,x} \text{ is defined,} \\ \uparrow, & \text{otherwise.} \end{cases}$$

Proof of the Lemma

$$
\begin{array}{cccccccc}
C_{e,0,0}, & C_{e,0,1}, & C_{e,0,2}, & \ldots, & C_{e,0,x}, & C_{e,0,x+1}, & \ldots \\
\cup & \cup & \cup & & \cup & \cup & \\
C_{e,1,0}, & C_{e,1,1}, & C_{e,1,2}, & \ldots, & C_{e,1,x}, & C_{e,1,x+1}, & \ldots \\
\cup & \cup & \cup & & \cup & \cup & \\
\vdots & \vdots & \vdots & & \vdots & \vdots & \\
\cup & \cup & \cup & & \cup & \cup & \\
C_{e,x,0}, & C_{e,x,1}, & C_{e,x,2}, & \ldots, & C_{e,x,x}, & C_{e,x,x+1}, & \ldots \\
\cup & \cup & \cup & & \cup & \cup & \\
\vdots & \vdots & \vdots & & \vdots & \vdots &
\end{array}
$$

The unary function $g(e, u, \_)$ is defined to differ from all functions eliminated in $\bigcup_{x\in\omega} C_{e,u,x}$.

Proof of the Lemma

For each pair of e, x we show that $g(e, u, x)$ is total.

- If $u \geq x$, then $C_{e,u,x} = \emptyset$ and consequently $g(e, u, x) = 1$.
- Suppose $u < x$ and $g(e, x, x), \ldots, g(e, u+1, x)$ are defined.
  - $\phi_{s(e,x)}(x), \ldots, \phi_{s(e,u+1)}(x)$ are defined according to (3).
  - Hence $t_{s(e,x)}(x), \ldots, t_{s(e,u+1)}(x)$ are defined.
  - It follows that $C_{e,u,x}$ is defined.
  - Consequently $g(e, u, x)$ is also defined.

  This completes the downward induction.

We conclude that g is a total function, which implies that $t_{s(e,u+1)}(x)$ is always defined.

Proof of the Lemma

Fact. Some v exists such that $\phi_e^2(0,x) = \phi_e^2(u,x)$ for all $x > v$.

Proof.

By definition $C_{e,u,x} = C_{e,0,x} \cap \{u, u+1, \ldots, x-1\}$.

Let $v = \max\{x \mid C_{e,0,x} \text{ contains an index } i < u\}$.

It is clear that $C_{e,0,x} \subseteq \{u, u+1, \ldots, x-1\}$ for all $x > v$.

Hence $C_{e,0,x} = C_{e,u,x}$ for all $x > v$. $\square$

Proof of the Lemma

Fact. If $\phi_i = \phi_e^2(0, x)$, then $r(t_{s(e,i+1)}(x)) < t_i(x)$ a.e..

Proof.

Let i be an index for $\phi_e^2(0, x)$. It should be clear that for all $x > i$, the following holds:

$$r(t_{s(e,i+1)}(x)) < t_i(x).$$

If not, i would have been canceled at some stage, say $i \in C_{e,0,w}$.

But then $\phi_i(w) \neq g(e, 0, w)$ by definition.

That is $\phi_i(w) \neq \phi_e^2(0, w)$, contradicting to the assumption. $\square$

Proof of the Lemma

Let $f(x) = \phi_e^{(2)}(0, x)$.

We have proved that f satisfies the property stated in the lemma.

Theorem (Blum, 1967). Let r be a total computable function. There is a total computable function f such that given any program P_i for f there is another program P_j for f satisfying $r(t_j(x)) < t_i(x)$ a.e..

W.l.o.g. assume that r is increasing.

By a slight modification of the proof of the lemma, we get a total computable function f such that given any program P_i for f there is a program P_k for f satisfying the following:

- $\phi_k(x)$ is total and $\phi_k(x) = f(x)$ a.e..
- $r(t_k(x) + x) < t_i(x)$ a.e..

Some c exists such that $\phi_k(x) = f(x)$ whenever $x > c$. We get a program P_j from P_k by short-cutting computations at inputs $\leq c$.

If c is large enough such that the additional computation cost is less than c, then the program P_j satisfies $r(t_j(x)) < t_i(x)$ a.e..

We cannot define time complexity for problems.

We can however define time complexity for solutions.

Hartmanis and Stearns' Linear Speedup Theorem

Theorem (Hartmanis and Stearns, 1965).

If L is decidable in $T(n)$ time, then for any $\epsilon > 0$ it is decidable in $\epsilon T(n) + n + 2$ time.

Hartmanis and Stearns' Linear Speedup Theorem

Proof.

Suppose a k-tape TM $\mathbb{M} = (Q, \Gamma, \delta)$ accepts L in time $T(n)$. Design a k-tape TM $\widetilde{\mathbb{M}}$ that encodes m symbols of $\mathbb{M}$ by one symbol of $\widetilde{\mathbb{M}}$.

The alphabet of $\widetilde{\mathbb{M}}$ is $\Gamma \cup \Gamma^m$. In $n+2$ steps $\widetilde{\mathbb{M}}$ converts the input. $\widetilde{\mathbb{M}}$ then uses n/m steps to realign the head.

In state $(q, h_1, \ldots, h_k)$, where $h_1, \ldots, h_k \leq m$, $\widetilde{\mathbb{M}}$ moves right one step, left two steps, right one step to gather information. It then takes two steps to update data.

The overall time it takes is $n + 2 + \frac{n}{m} + \frac{6}{m} T(n) \leq n + 2 + \frac{7}{m} T(n)$. So let m be $7/\epsilon$. □

3. Gap Theorem

Complexity Class

Let $b(x)$ be total and computable. The time complexity class of $b(x)$, denoted by $\mathbf{TIME}(b(x))$, is the following set

$$\{\phi_e \mid \phi_e(z) \text{ is total and } t_e(z) \leq b(|z|) \text{ a.e.}\}.$$

Is it true that for every total computable function $b(x)$ the inclusion $\mathbf{TIME}(b(x)) \subseteq \mathbf{TIME}(2^{b(x)})$ is strict?

Boris Trakhtenbrot. Turing Computations with Logarithmic Delay. Algebra and Logic 3(4):33-48, 1964. (in Russian)

Allan Borodin. Computational Complexity and the Existence of Complexity Gaps. Journal of the ACM 19(1):158-174, 1972.

Gap Theorem (Trakhtenbrot, 1964; Borodin, 1972).

Let $r(x)$ be a total computable function such that $r(x) \geq x$. Then there is a total computable function $b(x)$ such that $\mathbf{TIME}(b(x)) = \mathbf{TIME}(r(b(x)))$.

Proof of Gap Theorem

Define a sequence of numbers $k_0 < k_1 < k_2 < \ldots < k_x$ by

$$\begin{aligned} k_0 &= 0, \\ k_{i+1} &= r(k_i) + 1, \quad \text{for } i < x. \end{aligned}$$

The $x + 1$ intervals $[k_0, r(k_0)]$, ..., $[k_x, r(k_x)]$ are disjoint.

Let $P(i, k)$ denote the following decidable property:

- On every input of length i, each of $\mathbb{M}_0, \ldots, \mathbb{M}_i$ either halts in k steps or does not halt in $r(k)$ steps.

Proof of Gap Theorem

Let n_i be $\sum_{j=0}^{i} |\Gamma_j|^i$, the number of input of size i in $\mathbb{M}_0, \ldots, \mathbb{M}_i$.

- The n_i input strings of size i cannot fill all the $n_i + 1$ intervals $[k_0, r(k_0)]$, ..., $[k_{n_i}, r(k_{n_i})]$.
- It follows that there is some $j \leq n_i$ such that $P(i, k_j)$ is true.
- Let $b(i)$ be the least such k_j.

Suppose $\mathbb{M}_h$ accepts $L \in \textbf{TIME}(r(b(n)))$.

For every x with $|x| \geq h$ then by definition $\mathbb{M}_h(x)$ either halts in $b(|x|)$ steps or does not halt in $r(b(|x|))$ steps.

It follows that $L \in \textbf{TIME}(b(x))$.

The growth of $b(x)$ is too fast for r to make any difference.