In Pursuit of a Randomized Time Hierarchy Theorem

Karthik Gajulapalli (2018) [Survey]

June 4, 2022

Big Picture: What are Hierarchy Theorems and why study them?

- In complexity theory we are interested in the following question how does a resource affect our ability to recognize languages?
- How much power does increasing some amount of a resource have in recognizing more languages
- We have Hierarchy theorems for Time, Space, Even Non-uniform advice for Circuits

Deterministic Time Hierarchy Theorem

- Theorem: HS[65]
  If f,g are time-constructible functions satisfying
  $f(n) \log(f(n)) = o(g(n))$ then
  $\mathbf{DTIME}(f(n)) \subsetneq \mathbf{DTIME}(g(n))$
- Proof:
  - Idea: Use Diagnolization
  - Define D: simulate M_i on input x_i for g(n) steps.
    Then flip the answer; if doesn't halt set to 0
  - Imagine $\exists$ TM M such that M can solve D in TIME(f(n)), then
    $M(x) = D(x)\ \forall\ x$ and M(x) runs in time $f(|x|)$
    So Now $M(M) = b$ in Time f(n) by assumption
    But $D(M) = 1\text{-}b$ which means $M(x) \neq D(x)$ thus causing a contradiction
  - Hence Proved !!

Non-Deterministic Time Hierarchy Theorem

- Theorem: Cook[71]
  If f,g are time-constructible functions satisfying
  $f(n+1) = o(g(n))$ then
  $\mathbf{NTIME}(f(n)) \subsetneq \mathbf{NTIME}(g(n))$
- Proof:
  - Problem: Cannot use same approach (Why??)
  - Idea: Use Lazy Diagonalization

Quick Review of Probabilistic Polynomial Time

- Difference between PTM vs NDTM is that in a PTM I am interested in the fraction of branches that accept, while in NDTM I am interested in whethere a single branch accepts
- **BPTIME(Bounded Error Probabilistic Time)**:
  - BPTIME(f(n)) if RT(f(n), f(n))
    Pr [M(x) = L(x)] $\geq 2/3$
  - BPP = BPTIME(n^c)
- **RPTIME(Randomized Time)**:
  - RTIME(f(n)) if RT(f(n), f(n))
    if x $\in$ L, Pr [M(x) = 1] $\geq 2/3$
    if x $\notin$ L, Pr [M(x) = 1] = 0
  - RP = RTIME(n^c)

Hierarchy Theorems on PTMs

Question: Can I use some kind of diagonalization hack on PTMs to achieve hierarchy results?

Syntactical vs Semantic TMs

- **Syntactic:** Can exactly enumerate all the TM's (DTM, NDTM)
- **Semantic:** Cannot exactly enumerate each TM (PTM's and Quantum Machines with one,two and zero sided error)
  - For example in BPP there is a special property $\forall$ x $\in \{0, 1\}^*$ either $\Pr[M(x) = 1] \geq 2/3$ or $\Pr[M(x) = 1] \leq 1/3$.
    It is undecidable to test whether a machine can satisfy this property
  - It is also unknown whether we can test if a machine M satisfies this for a given input in less than 2^n steps

Goal

Main Goal of Today:

Try achieving Hierarchy for BPP to something like this:

$\textbf{BPTIME}(n^d) \subsetneq \textbf{BPTIME}(n^{d+1})\ \forall\ \mathrm{d} \geq 1$

Always Start with Brute Force

$$\mathbf{RT}\ (\mathrm{p(n)},\ p^*(\mathrm{n})) \subsetneq \mathbf{RT}(2^{p^*(n)}\mathrm{p(n)}log^2 p(n),\ p^*(\mathrm{n}))$$

But this is terrible we shouldn't have to expect an exponential blow up between two slices

Idea 2: There exists a Hierarchy if BPP has a complete problem

BPTime-hard We say that L is BPTIME-hard if $\exists$ constant c such that for any time constructible function t and any language L' BPTIME(t) there exists a deterministic $t(|x|)^c$ time computable function f such that $\forall$ x, x *in* L' $\iff$ f(x) $\in$ L

BPP-complete if L $\in$ BPP and BPTIME-hard.

Promise Problems

DEF 1: A promise problem π is a pair of sets (π_Y, π_N) where π_Y, π_N are disjoint.

DEF 2: Let t(n) be a function on the Naturals, We say that $\pi = (\pi_Y, \pi_N)$ is in PromiseBPTime(t(n)) if there exists a probabilistic t(n)-time machine M such that x $\in \pi_Y \implies$ Pr[M(x) = 1] $> 2/3$ and x $\in \pi_N \implies$ Pr[M(x) =1] $< 1/3$.
We now define PromiseBPP = U_c PromiseBPTime(n^c)

Promise BPP has A hierarchy

PromiseBPTime has a PromiseBPTime-complete language so we get a hierarchy of the form PromiseBPTime(n^d) $\subsetneq$ PromiseBPTIME(n^{d+1}) $\forall$ d.

CAP:
The promise problem Circuit Acceptance Probability is the pair (CAP_Y, CAP_N) where CAP_Y contains all circuits C such that Pr_x[C(x) = 1] $> 2/3$ and CAP_N contains all circuits C such that Pr_x[C(x) = 1] $< 1/3$.
CAP $\in$ PromiseBPP
Consistent: We say that a language L is consist with a promise problem $\pi = (\pi_Y, \pi_N)$ if $\forall$ x $\in \{0,1\}^*$ it holds that x $\in \pi_Y \implies$ x $\in$ L and x $\in \pi_N \implies$ x $\notin$ L.

Lemma : Let L be a a language consistent with the promise problem CAP. Then L is BPtime-hard.
Proof : Any Language L' can be reduced to CAP and there fore to L in t^2 steps using a Cook-Levin Reduction.

Corollary 1: If there exists a language L such that:
1. L is consistent with the promise problem CAP.
2. L $\in$ BPP

Then there exists a BPP-complete language.

Some Helpful Lemmas

All following scaling up lemmas should follow from relatively straightforward padding.

Lemma 1: $\forall$ constant d ≥ 1, if BPTIME(n^d) = BPTIME(n^{d+1}) then BPTIME(n^d) = BPP.

Lemma 2: $\forall$ constant d ≥ 1, if BPTIME(n^d) = BPP then BPTIME (t(n)) = BPTIME($t(n)^c$) for every constant c ≥ 1 and time-constructible function t that satisfies t(n) $\geq n^d$

Corollary 2 from above: For every constant d ≥ 1, if there exists a time constructible function t and a constant c > 1 such that t(n) $\geq n^d$ and BPTIME (t(n)) $\subsetneq$ BPTIME($t(n)^c$) then BPTIME(n^d) $\subsetneq$ BPTIME(n^{d+1})

Finally the Hierarchy Theorem

Theorem: Suppose that BPP has a complete problem. Then there exists a constant c such that for every time-constructible t it holds that BPTIME(t) $\subsetneq$ BPTIME(t^c).
And from Corollary 2, this proves that BPTIME(n^d $\subsetneq$ BPTIME(n^{d+1}) $\forall$ d $\geq$ 1.

Proof

1. Let L be a BPP-complete problem and let M_L be its accepting TM that runs in time n^a for some constant a.
2. We know that there exists a constant b such that for every time-constructible function t, every language in BPTime(t) is reducible to L using a t^b-time deterministic reduction.
3. For a string i, let M_i be the i-th deterministic TM. Define the language K such that x $\in$ K $\iff$ $M_x^{t(x)^b}$(x) $\notin$ L. We get:
(a) K $\in$ BPTIME($t^{O(ab)}$).

(b) K $\notin$ BPTIME(t).
Item(a) is true since we can decide K by negating $M_L(M_x(x))$,

and it takes $t(\|x\|)^{O(ab)}$ time. To prove item(b) let us assume for sake of contradiction that K $\in$ BPTIME(t). L is complete for BPP. So there exists an i such that i $\in$ K $\iff$ M_i (i) running in time $t(i)^b \in$ L. But by definition of K this happens $\iff$ i$\notin$ K and we get a contradiction.

Other Work

- A slightly non-uniform probabistic time hierarchy theorem [Barak 02]
- Hierarchy Theorems for PPT [Fortnow, Santhanam 04]
- From Log Bit to 1 Bit [Goldreich, Sudhan, Trevisan 05]
- Circuit Lower bounds for MA/1 [Santhanam 07]

Thank you !!!!