| | |
|---|---|
| **CS221: Computational Complexity** | Prof. Salil Vadhan |

Lecture 14: More on $\mathbf{NTIME}(n) \neq \mathbf{TIME}(n)$ and Provably Intractable Problems

*10/23* *Scribe: Dave Fetterman*

# 1 Announcements

- PS 2 due 10/24
- Monday 10/28 4PM MD G-125 CS colloqium with Joan Feigenbaum
- Friday 11/1 11AM - CS colloqium:Primes in P, MD G-125, no lecture
- The relevant parts of Papadimitriou for this lecture are Problems 7.4.17, 14.5.12, Chapter 20, and Problem 20.2.13.

# 2 More intuition for PPST

**Theorem 1 (Paul, Pippenger, Szemeredi, Trotter)** *:* $\mathbf{TIME}(f(n)) \subseteq \mathbf{\Sigma_4 TIME}(o(f(n)))$.

**Corollary 2** $\mathbf{NTIME}(n) \neq \mathbf{TIME}(n)$

Main idea: Break time $f(n)$ computation into subcomputations of "size" $o(f(n))$, use alternation to check all of these at price $o(f(n))$.

Our first attempt:

- Split $f(n)$ into $a(n)$, $b(n)$ such that $f(n) = a(n) \cdot b(n)$.
- Delimit computation time intervals of length $b(n)$ by $t_0, t_1, \ldots, t_{a(n)}$.
- $\exists$ : guess existentially configurations of TM at times $t_0, t_1, \ldots, t_{a(n)}$; call them $c_0, c_1, \ldots, c_{a(n)}$
- $\forall$ : verify that $c_i \rightarrow c_{i+1}$ in b(n) steps.

The problem with this is that each configuration is of size $f(n)$, so even the guessing takes time $a(n) \cdot f(n) > f(n)$, and we've gained nothing.

*Observation:* In one time segment, the TM touches only $O(b(n))$ tape cells. Maybe we could guess $O(b(n))$ "relevant" cells for each $t_i$. However, this creates the difficulty that to verify consistency, it no longer suffices to look only at adjacent pairs!

There exists then, a graph-theoretic lemma where the real magic occurs. We identify $o(a(n))$ times $t_{i_1}, t_{i_2}, \ldots, t_{i_{o(a(n))}}$ such that configurations at each $t_{i_j}$ only "depends" on only $o(a(n))$ of the others. (This exploits some structural properties of the computation graph of a block-respecting TM.)

- Guess $c_1, c_2, \ldots, c_{o(a(n))}$ (Takes time $o(a(n)) \cdot b(n) = o(f(n))$.)
- $\forall j$ will verify if $c_{i_j}$ consistent with those configurations on which it depends (we use another two quantifiers for this).

Remarks:

- Ideas originated in valiant, Hopcroft-Paul-Valiant: $\mathbf{TIME}(f(n)) \subseteq \mathbf{SPACE}(\frac{f(n)}{\log f(n)})$.
- These results depend heavily on the Turing Machine model.
- Can actually prove something like $\mathbf{NTIME}(n) \subsetneq \mathbf{TIME}(n(\log^* n)^{1/4})$ with this approach.

# 3 Provably intractable problems

For natural problems, so far our best lower bounds are showing that GGEO, GO, QBF, etc. are not in **L** by their **PSPACE**-completeness and the space hierarchy theorem. To get problems not in **P**, it is sufficient to find problems complete for **EXP** (or higher!).

## 3.1 EXP and NEXP-complete problems

One general method is to find *succinct* versions of **P**, **NP**-complete problems. But what is a succinct representation of a graph?

**Definition 3** *A* succinct representation of a graph *is a circuit* $C : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ *defining* $G_C = (V, E)$, $V = \{0,1\}^n$, $E = \{(u,v) : C(u,v) = 1\}$

We then have a potentially succinct representation of the graph, since it can represent something exponentially larger than its description length (the circuit).

This yields succinct versions of familiar graph problems.

**Definition 4** SUCCINCT HAMILTONIAN PATH*:* SHP $= \{C : G_C$ *has a Hamiltonian path*$\}$.

**Proposition 5** SHP $\in$ **NEXP**

**Proof:** Given $C$, can construct $G_C$ in exponential time. We then can run our **NP** algorithm on $G_C$. ■

We can also look at problems like SUCCINCT CIRCUIT VALUE, SUCCINCT CIRCUIT SAT, SUCCINCT 3-SAT, etc. In these, the succinct representation of a circuit $C$ describes an exponentially bigger circuit $\tilde{C}$.

**Theorem 6** SUCCINCT CIRCUIT VALUE *is* **EXP***-complete. Also,* SUCCINCT CIRCUIT SAT *is* **NEXP***- complete.*

**Proof:** This is a use of the Cook-Levin reduction, on a $2^{n^k} \times 2^{n^k}$ tableau. The key point is that the circuit produced by Cook-Levin has a very regular structure (mostly the same constant-size circuit repeated many times) so can be succinctly described by a smaller circuit. See Papadimitriou, Thm 20.2. ∎

In fact, succinct versions of most familiar **P**-complete or **NP**-complete problems are **EXP**-complete or **NEXP**-complete, respectively, including SUCCINCT HAMILTONIAN PATH. But these succinct problems are not totally "natural" in that they still refer explicitly to computation (namely circuits). Next we'll see an even more natural intractable problem.

### 3.2 Regular Expressions with Exponentiation (r.e.e.'s)

$R(\Sigma, \circ, \cup, *, \uparrow)$ is the set of regular expressions with the exponentiation operation $R \uparrow n$ ($n$ given in binary), whose semantics are given by $L(R \uparrow n) \equiv L(RR \cdots R)$, there being n occurrences of $R$. These are also equivalent to regular expressions with squaring (i.e. $L(R^2) = L(RR)$).

Thus, a natural language/problem spawns from this: REE $= \{r.e.e.R | L(R) = \Sigma^*\}$.

**Theorem 7 (Meyer–Stockmeyer)** REE *is* **EXPSPACE** *complete*

**Corollary 8** *Any deterministic algorithm deciding* REE *takes space (and time, as a consequence)* $\geq 2^{n^{\Omega(1)}}$ *(for infinitely many $n$), by Space Hierarchy Theorem*