The results in this lecture are mostly taken from [1].

1 Preliminaries

Definition 1 (Infinitely-often). For an arbitrary complexity class $C$ over $\Sigma$, we define

$$\text{io-}C = \{ L' \subseteq \{0,1\}^n \mid \exists L \in C \exists \text{ an infinite } I \subseteq \mathbb{N} \forall n \in I . L \cap \Sigma^n = L' \cap \Sigma^n \}$$

2 EXP is not contained in fixed polynomial-sized circuits

Theorem 2.

• (easy) Every function $f : \{0,1\}^n \rightarrow \{0,1\}$ can be computed by a circuit of size $O(n2^n)$.
• Every function $f : \{0,1\}^n \rightarrow \{0,1\}$ can be computed by a circuit of size $(1 + o(1))\frac{2^n}{n}$.
• There exists a function $f : \{0,1\}^n \rightarrow \{0,1\}$ that cannot be computed by a circuit of size $(1 - o(1))\frac{2^n}{n}$.

Proof. (1) is trivial, e.g., by CNF or DNF. For (2) see [2]. For (3) count the number of size $S$ circuits (about $S2^S$) and functions (about $2^{2^n}$).

Lemma 3. Suppose $s(n)$ is such that $n \leq s(n) \leq \frac{2^n}{4n}$. Then there exists some $n_0$ such that for every $n \geq n_0$, $\text{SIZE}(s(n)) \leq \text{SIZE}(4s(n))$.

Proof. Exercise. Hint: by the above, when restricting the the right number of bits.

Theorem 4. For any fixed $a$, $\text{EXP} \nsubseteq \text{io-SIZE}(n^a)$.

Proof. There are about $S2^S$ circuits of size $S$ and we can efficiently (and brute force) enumerate them in about $S2^S$ space and $H = 2^{(S2^S)}$ time. Given two size $S$ circuits on $n$ bits we can brute force check whether they encode the same functionality in about $2^n \cdot S$ time. In particular we can find in $H2^nS$ time the lexicographically first circuit that can be solved with $4n^a$ size and not $n^a$ size guaranteed by Lemma 3.

We define a language $L$ as follows. Given $x \in \{0,1\}^n$ we find the circuit $C_n$ on $n$ inputs described above. $C_n$ has size $4n^a$ and no size $n^a$ circuit agrees with him on inputs of length $n$. We output $C_n(x)$. Clearly, $L \in \text{EXP}$ and $L \notin \text{io-SIZE}(n^a)$.
3 Diagonalizing Deterministic Time

We are all familiar with diagonalization and the time hierarchy. In words: having “more” time enables computing more. In particular there is no fixed $a$ such that $E \subseteq \text{DTIME}(2^n^a)$.

We also recall the proof method. We diagonalize over all small time machines $t$: For every $x$ we simulate the $x$’th Turing Machine (TM) $M_x$ for $t$ steps and answer the opposite. The language is in time $T$ (assuming $T$ time suffices to simulate $t$ steps) but not in time $t$.

We now extend this argument in two ways: first we want to define a language $L$ that differs with every TM $M$ in $\text{DTIME}(2^n^a)$ on every input length large enough (and not only once). Also we allow the small-time TM a short non-uniform advice.

**Theorem 5.** For every fixed $a \in \mathbb{N}$ it holds that $\text{EXP} \nsubseteq \text{io-DTIME}(2^n^a)/n^a$.

**Proof.** Fix $a$. There are at most $2^n$ TM with description size at most $n$ that use an advice string of size at most $n^a$. There are $2^{n^c}$ advice strings. Any TM $M$ (with description size at most $n$) and advice string $adv$ (of size $n^a$) determine a string (or a ”truth table”) of length $2^n$, that in place $x \in \{0, 1\}^n$ has the bit $M(x, adv)$.

We define a language $L$ as follows. On input $x \in \{0, 1\}^n$, $L$ does the following: If first computes a set $S$ of all TM with description size at most $n$ and advice strings of size at most $n^a$. There are $2^n$ TM that use an advice string $adv$ of size $n^a$. If $S$ becomes empty (which happens after at most $n^c + n$ steps), we choose an arbitrary answer (say, 0) for $w$ and all following length $n$ strings. Finally, we look at $x$ and let $L(x)$ be the value output on $x$ in the above process.

Clearly:

- $L \in \text{DTIME}(2^{O(n^a)})$ and therefore $L \in \text{EXP}$, and,
- $L \not\in \text{io-DTIME}(2^{n^a})/n^a$.

4 If $\text{NEXP} \subseteq \text{P/poly}$

**Theorem 6.** If $\text{NEXP} \subseteq \text{P/poly}$ then there exists a constant $d_0$ such that $\text{NTIME}(2^n)/n \subseteq \text{SIZE}(n^{d_0})$.

**Proof.** We want one language $U$ in $\text{NEXP}$ that capture them all (i.e., all languages in $\text{NTIME}(2^n)$).

Since $U$ is in $\text{NEXP}$ by our assumption it is also in $\text{P/poly}$, hence solvable by some fixed-polynomial size circuit. This implies a the same fixed-polynomial size circuit for all languages in $\text{NTIME}(2^n)/n$.

Specifically, define the following non-deterministic machine $U$. On input $(i, x)$ it simulates the $i$’th non-deterministic TM $M_i$ on input $x$ for $2^n$ steps, and accepts on a path iff $M_i$ accepts on that path. Then $U \in \text{NTIME}(2^n)$. Hence $U \in \text{SIZE}(n^d)$ for some constant $d$. 


Now, let $L \in \text{NTIME}(2^n)/n$. Then, there is a non-deterministic TM $M(x, a)$ running in time $2^n$, and an advice sequence $\{a_n\}$ where $|a_n| = n$ such that $x \in L \cap \{0, 1\}^n$ iff $M(x, a_{|x|}) = 1$. Say $M = M_i$. Then, $x \in L$ iff $U(i, x, a_{|x|}) = 1$. Hence, $L \in \text{SIZE}(O(2n^d))$. 

**Corollary 7.** If $\text{NEXP} \subseteq \text{P/poly}$ then for every fixed $a \in \mathbb{N}$ it holds that $\text{EXP} \not\subseteq \text{io-NTIME}(2^{n^a})/n$.

**Proof.** Suppose $\text{EXP} \not\subseteq \text{io-NTIME}(2^{n^a})/n$. Since $\text{NEXP} \subseteq \text{P/poly}$, by the previous claim, there exists some constant $d_0$ such that $\text{EXP} \not\subseteq \text{io-SIZE}(n^{d_0})$. But this contradicts Theorem 5. \hfill $\square$

**References**
