1 Oracle TMs and Relativization (Continued)

**Theorem 1.1.** *(Baker-Gill-Solovay Theorem)* There exist oracles \(A, B\) such that \(P^A = NP^A\), and \(P^B \neq NP^B\).

**Moral:** Cannot settle P vs NP by a proof that relativizes.

**Proof.** Let \(A = \{([M], x, 1^n) | M \text{ accepts } x \text{ in } \leq 2^n \text{ steps}\}\). We'll prove that \(EXP = P^A = NP^A\), where \(EXP = \bigcup_{c \in N} DTIME(2^{cn})\), by showing that \(EXP \subseteq P^A \subseteq NP^A \subseteq EXP\).

By definition, \(P^A \subseteq NP^A\). To show that \(EXP \subseteq P^A\), suppose \(L \in EXP\). This implies that \(\exists M_L\) such that \(M_L\) computes \(L\) in \(2^{nc}\) time for some constant \(c\). We construct a TM \(N^A\) that computes \(L\) in polynomial time by the following. We'll hardcode \(c\) and \([M_L]\) in \(N^A\), and for any input \(x\), \(N^A\) checks if \([M_L], x, 1^{nc}\) \(\in A\), which takes polynomial time, by writing it on the oracle tape and transitions to \(q_{\text{query}}\). The output of \(N^A\) will just be the answer of oracle. If the oracle answers YES, it implies that \(M_L\) accepts \(x\) in \(2^{nc}\) time, and \(x \in L\). Similarly, if oracle answers NO, it implies that \(M_L\) rejects \(x\), and \(x \notin L\). As a result, \(N^A\) computes \(L\) in polynomial time, so \(EXP \subseteq P^A\).

Additionally, to show that \(NP^A \subseteq EXP\), suppose \(L \in NP^A\). Then there exists a NDTM \(N\) such that \(N^A\) computes \(L\) in polynomial time, i.e., there are at most \(2^{\text{poly}(n)}\) possible paths that the machine executes. In each path it makes at most \(\text{poly}(n)\) oracle calls, and each oracle call will take at most \(2^{\text{poly}(n)}\) steps. So on a deterministic Turing machine, it takes \(O(2^{\text{poly}(n)} \cdot \text{poly}(n) \cdot 2^{\text{poly}(n)})\) time to compute \(L\), which proves that \(NP^A \subseteq EXP\). This finishes our proof that \(P^A = NP^A\).

To find \(B\) such that \(P^B \neq NP^B\), we first define \(U_B = \{1^n | \exists y \in B, |y| = n\}\) for any \(B \subseteq \{0, 1\}^\ast\). Notice that \(U_B \in NP^B\), since for an input \(1^n\) an NDTM can simply guess non-deterministically \(x \in \{0, 1\}^n\) with a path of \(n\) steps and check if \(x \in B\) with the oracle.

To finish the proof, we want to find \(B\) with \(U_B \notin P^B\). Let \(\{M_k\}\) be an enumeration of TMs. In state 0, set \(B = \emptyset\). In state \(i\), only finitely many strings \(y\) have been decided if \(y \in B\). Let \(n_i\) be the smallest integer such that no string \(\in \{0, 1\}^n\) has been decided. Consider the algorithm below:

- Run \(M^B\) on \(1^n\), and simulate \(M_i\) for \(2^n/10\) steps
- if \(M_i\) makes query of \(y \in B\) then
  - if \(y \in B\) or not is already decided then
    - Answer truthfully
  - else
    - Answer No
- end if
- end if
- if \(M_i\) answers YES then
  - set \(y \notin B\) for any unqueried \(y\)
- else
  - set \(z \in B\) for some unqueried \(z\), since \(M_i\) cannot ask every \(z\)
- end if
To prove $U_B \notin P^B$, suppose $U_B \in P^B$. Let $L(M_i) = U_B$ where $M_i$ runs in $c \cdot n^c$ time. Consider $M_i$ on input $1^n$. By construction of $B$, $M_i$ cannot query every $y$ of length $n_i$ since it only runs for $2^{n_i}/10$ steps. If $M_i$ outputs YES, since any $y$ of length $n$ will be set not in $B$, we know that $1^n \notin U_B$. If $M_i$ outputs NO, since we make some unqueried $z \in B$, we have $1^n \in B$. Either way, $M_i$ makes a mistake on $1^n$, which is a contradiction. Therefore, $U_B \notin P^B$, and $P^B \neq NP^B$.

\[\square\]

## 2 Space complexity

**Definition 2.1.** A Turing machine $M$ on input $x$ uses space $S$ if at most $S$ work and output tape cells are accessed.

**Remark 2.2.** Notice that it takes at least $o(\log(n))$ space for input of length $n$. If $S(n) = o(\log(\log(n)))$ (or smaller), it’s regular language.

**Definition 2.3.** Define $\text{DSPACE}$ similarly to $\text{DTIME}$, that is, $\text{DSPACE}(S(n)) := \{L \in \text{DTIME}(S(n)) : \exists M \text{ such that } L = L(M) \}$, all languages such that there is a TM that computes it in $S(n)$ space. Similarly, $\text{NSPACE}(S(n)) := \{L \in \text{NTIME}(S(n)) : \exists M \text{ such that } L = L(M) \}$, all languages such that there is a NDTM that computes it in $O(S(n))$ space.

**Definition 2.4.** Define $\text{PSPACE} = \bigcup_{c \in \mathbb{N}} \text{DSPACE}(n^c)$, $\text{NPSPACE} = \bigcup_{c \in \mathbb{N}} \text{NSPACE}(n^c)$. Let $L = \text{DSPACE}(\log(n))$, $NL = \text{NSPACE}(\log(n))$.

**Question 1.** What is the relation between the complexity classes $L$, $NL$ and $P$? Similarly, can we say anything about the relationship between the complexity classes $PSPACE$, $P$ and $NP$?

**Claim 2.5.** $P \subseteq \text{PSPACE}$, and $NP \subseteq \text{PSPACE}$.

*Proof. Since $P \subseteq NP$, we just need to prove that $NP \subseteq \text{PSPACE}$. It’s because we can brute-force each path of the NDTM while reusing the same tape cells for each step of the path.*

**Claim 2.6.** $\text{NSPACE} \subseteq EXP$.

The proof of the above claim is based on the notion of **configuration graphs** that we now define.

**Configuration graph of a TM:** A configuration of TM contains its current state, the contents of non-input tapes and the tape heads. The number of possible configurations at a certain point is thus $|Q| \times 2^{O(S(n))} \times S(n)$. For a TM $M$ using space $O(S(n))$ on input $x$, the configuration graph $G_{M,x}$ is defined as follows: The nodes of $G_{M,x}$ will be all the configurations, which is at most $2^{CS(n)}$. An edges $u \rightarrow v$ exist in the graph if $M$ can go from configuration $u$ to $v$ in 1 step.

We are now ready to prove Claim 2.6. Suppose $L \in \text{NSPACE}(S(n))$, there exists NDTM $M$ using $S(n)$ space that computes $L$. Then we can simply construct the configuration graph, and use BFS to check the connectivity from the start configuration to accept configuration. This process takes $2^{CS(n)}$ time, so $L \in EXP$.

**Corollary 2.7.** For any space constructible function $S(n)$, we have $\text{DSPACE}(S(n)) \subseteq \text{NSPACE}(S(n)) \subseteq \text{DTIME}(2^{O(S(n))})$. Thus, $L \subseteq NL \subseteq P$.

**Remark 2.8.** It is conjectured that $L = NL$ and $NL \neq P$. 

Lecture 6: Sep 7, 2023