Lecture 8

Gidon Rosalki

June 17, 2025

1 Reminders

Hear me and rejoice mortal, for here are some reminders of definitions of complexity sets:

Definition 1.1 (TIME)**.**

$$TIME\left(f\left(n\right)\right)=\{L\subseteq\Sigma^{*}:\textit{There exists a deterministic TM that runs in }O\left(f\left(n\right)\right)\textit{ and decides the language }L\}$$

Definition 1.2 (NTIME)**.**

$$NTIME\left(f\left(n\right)\right)=\{L\subseteq\Sigma^{*}:\textit{There exists a non deterministic TM that runs in }O\left(f\left(n\right)\right)\textit{ and decides the language }L\}$$

For a non deterministic TM, for some word $w\in\Sigma^{*}$, there are different runs with different outcomes and runtimes. We will say that $w\in L\left(M\right)$ **if and only if** there exists a run that ends in q_{acc}. The runtime is the worst case between all the possible runs.

Definition 1.3 (SPACE)**.**

$$SPACE\left(f\left(n\right)\right)=\{L\subseteq\Sigma^{*}:\textit{There exists a deterministic TM that decides }L\textit{ while using the space }O\left(f\left(n\right)\right)\}$$

Definition 1.4 (NSPACE)**.**

$$NSPACE\left(f\left(n\right)\right)=\{L\subseteq\Sigma^{*}:\textit{There exists a non deterministic TM that decides }L\textit{ while using the space }O\left(f\left(n\right)\right)\}$$

$w\in L$ is defined similarly to $NTIME$, and space complexity according to the worst case of all the runs.

Definition 1.5 (P)**.**

$$P=\bigcup_{k=1}^{\infty}TIME\left(n^{k}\right)$$

All the languages that can be decided in deterministic polynomial time.

Definition 1.6 (NP)**.**

$$NP=\bigcup_{k=1}^{\infty}NTIME\left(n^{k}\right)$$

All the languages that can be decided in nondeterministic polynomial time.

"If you want to make a million dollars, there are easier ways than to prove equivalence of P and NP"-Oded Schwartz, May 2025.

Definition 1.7 (EXP)**.**

$$EXP=\bigcup_{k=1}^{\infty}TIME\left(2^{O\left(n^{k}\right)}\right)$$

It is true that

$$P\subseteq NP\subseteq EXP\subseteq R$$

The first containment is because the DTM is a specific case of an NDTM. The second containment is left as an exercise for the reader, since it is homework, and the third is because there is a deciding machine (happens to be also of exponential runtime).

Reminder: We will say that there is a polynomial reduction from A to B, and write $A\leq_{p}B$, if there exists a procedure f, and TM M_{f}, such that M_{f} computes f in polynomial time, and also

$$w\in A\Leftrightarrow f\left(w\right)\in B$$

(Polynomial time by $|w|$).

Theorem 1 (Reduction theorem)**.** *The reduction theorem for polynomial reductions:*

- $A\leq_{p}B\wedge B\in P\implies A\in P$

2 Extension of reduction

Theorem 2.

$$A \leq_p B \wedge B \leq_p C \implies A \leq_p C$$

Proof . Since $A \leq_p B$, there exists a function f, and there exists a polynomial DTM M_f which computes f, and

$$w \in A \Leftrightarrow f(w) \in B$$

Similarly, since $B \leq_p C$, there exists a function g, and there exists a polynomial DTM M_g, which computes g, and

$$w \in B \Leftrightarrow g(w) \in C$$

- Construction: We will construct the machine M_h, which computes h, a reduction from A to C, such that M_h is a polynomial DTM.
  For an input $x \in A$, M_h will compute $y = f(x)$, like M_f, and then compute $z = g(y)$, like M_g, and then return z.
  We need to show correctness, and polynomial runtime.
- Correctness:

  $$\begin{aligned} x \in A &\Leftrightarrow y = f(x) \in B \\ &\Leftrightarrow z = g(y) \in C \end{aligned}$$

  Where the first follows from the correctness of M_f, and the second from the correctness of M_g.
- Runtime: The first step is running M_f, and it is known that M_f is a polynomial DMT, which is to say it runs in $O\left(|x|^{k_1}\right)$, for some constant k_1. The second step is running the machine M_g, which is known to be a polynomial DTM, and therefore runs in $O\left(|y|^{k_2}\right)$, for some constant k_2. This is not good enough, since we need the runtime to be polynomial in the length of the input x, but we are also dependent on y, an internal variable. We need to find some bound for y, associated with $|x|$. We will note that the machine M_f can only create an output y of length that is at most the number of steps in which it runs. Therefore $|y| = O\left(|x|^{k_1}\right)$, which is to say that the runtime of the second step is $O\left(|x|^{k_1 \cdot k_2}\right)$. Therefore, the runtime of both steps is $O\left(|x|^{k_1 \cdot k_2}\right)$, where k_1 and k_2 are both constants.

□

Let us recall the definitions of $\mathcal{C}$-hard, and $\mathcal{C}$-complete. $L \in NP - hard$ **if and only if** for every $L' \in NP$, $L' \leq_p L$. $L \in NP - complete$ **if and only if** $L \in NP$ and $L \in NP - hard$.

Now, $P = NP$? Worth a million dollars, and 100% in the course (which is frankly the important thing here), but there are better ways to earn a million dollars, and easier ways to get 100% in this course.
If $L \in NPC$ (NP-complete), and also $L \in P$, then $P = NP$, and if $L \in NPC$, and also $L \notin P$ then $P \neq NP$.
Explanation: If $L \in NPC$, then $L \in NP - hard$, which is to say for every $L' \in NP$, then $L' \leq_p L$. If in addition $L \in P$, then according to the reduction theorem, $L' \in P$, which is to say that $NP \subseteq P$, and therefore $P = NP$.

3 NPC languages

3.1 3SAT

A first example of an NPC language:

$$3SAT = \{\varphi : \varphi \text{ is a 3CNF expression, and also } \varphi \text{ satisfies}\}$$

Reminder: A boolean expression will be called CNF if it is of the following structure: Constructed from statements, and within each statement is boolean variables separated by logical ors. Between each statement, are logical ands. An expression will be called 3CNF if in each statement there are exactly 3 literals, which is to say 3 variables. An expression φ will be called satisfying **if and only if** there is a placement that satisfies the variables, which is to say, a placement of the values True or false on teh variables such that the overall result of φ is true.

Theorem 3.

$$3SAT \in NPC$$

Proof . We need to show

1. $3SAT \in NP$:
   We will show that $3SAT \in NP$, which is to say, to show that there exists a polynomial DTM that decides 3SAT. The machine M:

   - Will produce an expression of the structure 3CNF (easy)
   - Will guess a placement, and verify that the placement satisfies φ

   The runtime is polynomial, for obvious reasons. The correctness follows from if the placement satisfies, then there is a run that accepts, and if not, then there is not.

2. $3SAT \in NPH$ - next week.

□

3.2 CLIQUE

Second example of an NCP language:

$$CLIQUE = \{(G,k) : G \text{ is an undirected graph, } k \text{ is a natural number, and there exists in } G \text{ a clique of size } k\}$$

This is to say, that there exists a subset of edges of size k, such that for every 2 nodes in the set, they are connected by an edge.

Theorem 4. *CLIQUE is NPC*

Proof .

- We need to show that $CLIQUE \in NP$.
  We will show a polynomial NTM that decides CLIQUE. The machine will take an input (G,k), and will non deterministically guess a subset S of nodes, and then will check deterministically the following:

  1. Count how many nodes are in S, and if it is different from k, stop on q_{rej}
  2. For every pair of nodes in S, check if there is an edge between them. If we find a pair in S without an edge between them, then stop on q_{rej}. If there is an edge between every pair of nodes in S, then stop on q_{acc}.

  Correctness: If $(G,k) \in CLIQUE$, then there exists a subset S of edges in G, such that $k = |S|$, and also between every pair of edges in S, there is an edge. Therefore, if the machine M will guess S, then it will stop on q_{acc}. If $(G,k) \notin CLIQUE$, then for every set S, it is either not of size k, or there is a pair of nodes that are not connected by an edge, and therefore in all cases, M will stop on q_{rej}.
  Runtime: A guess S is bound by the length of the input. Counting the size of S takes place in $O(n)$. Checking the edges: $O\left(k^2\right)$ pairs, and for every pair, $O(n)$, so overall $O\left(n^3\right)$, which is to say a polynomial runtime.

- We need to show that $CLIQUE$ is NP hard.
  This is to say that $\forall L' \in NP$, there is a reduction $L' \leq_p CLIQUE$. We will show "only" a reduction from $3SAT \leq_p CLIQUE$. Since $3SAT \in NPH$, it is true that $\forall L' \in NP,\ L' \leq_p 3SAT$. We have shown that polynomial reductions are closed to compounding, and therefore we can conclude that $\forall L' \in NP,\ L' \leq_p CLIQUE$, which is to say $CLIQUE \in NPH$. We need to show that $3SAT \leq_p CLIQUE$: This is to say that there exists a function f, and a TM M_f, such that M_f computes f, in polynomial time, and also f is a reduction function, which is to say

  $$\varphi \in 3SAT \Leftrightarrow f(\varphi) \in CLIQUE$$

  where $f(\varphi) = (G,k)$. So

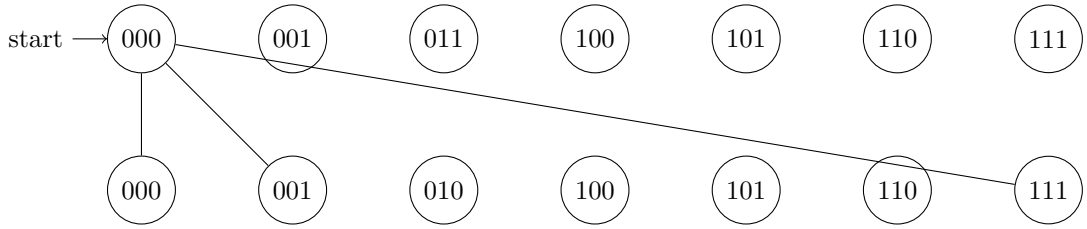
  $$\varphi \text{ is 3CNF satisfiable} \Leftrightarrow G \text{ has a CLIQUE of size } k$$

  Construction: Given a statement φ, the machine M_f will run as follows:

  1. Check if φ is of the structure $3CNF$ (easy). If not, return $(G,k) \notin CLIQUE$, for example, G has 3 nodes, $k = 100$.
  2. W.l.o.g. φ is of the structure $3CNF$. M_f will run as follows: For all statements in φ, we will define 7 nodes, one for every placement that satisfies the statement. Between every pair of nodes, M_f will make a connection, aside from when they are related to a shared variable, and do not agree on its placement. k will be the number of statements in φ.

  Example:

  $$\varphi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_3} \vee \overline{x_4})$$



Filling out the rest of the edges is left as an exercise for the reader. We create a graph, 7 wide, and m, the number of statements in φ, deep.

Runtime: Creating edges: This is polynomial in φ, since there are $7m$ edges, and $m < |\varphi|$. Connecting the edges: This needs to pass over every pair of edges $O\left((7m)^2\right)$, which is to say $O\left(m^2\right)$, and for every pair we need to check: If we do not have shared variables, we make an edge. If we do have shared variables, and we give the shared variables equivalent values, we will connect an edge, and otherwise will not. This takes $O\left(|\varphi|\right)$ for every pair, and we thus have an overall runtime of $O\left(|\varphi|^3\right)$.

Correctness: We need to show that if $\varphi \in 3SAT$, then $(G,k) \in CLIQUE$. This is to say if φ is a satisfying $3SAT$ statement, then G contains a CLIQUE of size k.. We will look at a placement A that satisfies φ. For every statement, we will choose a node (from the 7-tuple), that is suitable to the placement that agrees with A. We will note that A is a satisfying placement, and therefore is relevant to precisely 1 node of all the 7-tuple. We shall theorise that the set S of nodes that we chose is a clique in G, of size k. It is of size k since in S there is exactly one node for every layer / statement. It is a clique since every pair of nodes in S are connected by an edge. We will look at a pair of nodes in S. If they are not matched to a shared variable, according to the construction, they must have an edge between them. If they do have a shared variable, since the placement for every statement is related to the general placement A, then they satisfy the placement for shared variables, which is to say, according to the construction, there is an edge between the pair of nodes.

Let us consider the other direction: If $(G,k) \in CLIQUE$, then $\varphi \in 3SAT$. We will look at the subset S of nodes, which is a clique in G of size k. We will note that S necessarily contains a single node from each layer. It cannot be more than 1 since in each layer there are no edges, and S is a clique. It cannot be less than 1, since the size of the clique is the number of layers. We will define a placement A through S. For every statement, the node in S from the relevant layer defines a placement for the variables of the statement. We will note that a pair of nodes that handle a shared variable agree on its placement, since S is a clique, which is to say they are connected by an edge, and therefore we have received a a placement for all the variables. Additionally, every node is relevant to a placement according to the statement that satisfies it, and since there is in S a node in every layer, the placement A satisfies all the statements, which is to say, it satisfies φ, which is to say that $\varphi \in 3SAT$.

□

4 Alternative definition of NP

A DTM M is a **polynomial verifier** for the language L. If M accepts a pair of inputs (w,c), and runs in polynomial over $|w|$, and it is true that

- For all $w \in L$, there exists c such that $M\left(w,c\right) = q_{\text{acc}}$.
- Fir all $w \notin L$, and for all c $M\left(w,c\right) = q_{\text{rej}}$

We will define NP' to be the set of all languages L which have a polynomial verifier.

Theorem 5 (NP = NP')**.** *Example: We want to show that there is a polynomial verifier for the language CLIQUE. w will be a pair (G,k), which is a candidate for being in $CLIQUE$. c will be a witness to the presence, and in this case is meant to represent the clique in G, of size k. The deterministic machine M will check if c is a set of k nodes, and if between every pair of nodes there is an edge. If so, M will return q_{acc}, and otherwise, M returns q_{rej}.*
Correctness: If $(G,k) \in CLIQUE$, then the placement of c to be a clique of size k will enable M to return q_{acc}. If $(G,k) \notin CLIQUE$, then for every placement of c, running M will result in q_{rej}.
Runtime: Easy to show that it is polynomial in $|(G,k)|$.