24.4
Proof of Cook-Levin Theorem

24.4.1
Statement and sketch of idea for the proof

Cook-Levin Theorem

Theorem 24.1 (Cook-Levin).

SAT *is* **NP-Complete***.*

We have already seen that **SAT** is in **NP**.

Need to prove that <u>every</u> language $\boldsymbol{L} \in$ **NP**, $\boldsymbol{L} \leq_{\boldsymbol{P}}$ **SAT**

Difficulty: Infinite number of languages in **NP**. Must <u>simultaneously</u> show a <u>generic</u> reduction strategy.

Cook-Levin Theorem

Theorem 24.1 (Cook-Levin).

SAT *is* **NP-Complete***.*

We have already seen that **SAT** is in **NP**.

Need to prove that <u>every</u> language $\boldsymbol{L} \in$ **NP**, $\boldsymbol{L} \leq_{\boldsymbol{P}}$ **SAT**

Difficulty: Infinite number of languages in **NP**. Must <u>simultaneously</u> show a <u>generic</u> reduction strategy.

The plot against SAT

High-level plan to proving the Cook-Levin theorem

What does it mean that $L \in$ **NP**?

$L \in NP$ implies that there is a non-deterministic TM M and polynomial $p()$ such that

$$L = \{x \in \Sigma^* \mid M \text{ accepts } x \text{ in at most } p(|x|) \text{ steps}\}$$

Input: M, x, p.
Question: Does M stops on input x after $p(|x|)$ steps?

Describe a reduction R that computes from M, x, p a **SAT** formula φ.

- R takes as input a string x and outputs a SAT formula φ
- R runs in time polynomial in $|x|, |M|$
- $x \in L$ if and only if φ is satisfiable

The plot against SAT

High-level plan to proving the Cook-Levin theorem

What does it mean that $L \in \mathbf{NP}$?

$L \in NP$ implies that there is a non-deterministic TM M and polynomial $p()$ such that

$$L = \{x \in \Sigma^* \mid M \text{ accepts } x \text{ in at most } p(|x|) \text{ steps}\}$$

Input: M, x, p.
Question: Does M stops on input x after $p(|x|)$ steps?

Describe a reduction R that computes from M, x, p a **SAT** formula φ.

- R takes as input a string x and outputs a SAT formula φ
- R runs in time polynomial in $|x|, |M|$
- $x \in L$ if and only if φ is satisfiable

The plot against SAT

High-level plan to proving the Cook-Levin theorem

What does it mean that $L \in \mathbf{NP}$?

$L \in NP$ implies that there is a non-deterministic TM M and polynomial $p()$ such that

$$L = \{x \in \Sigma^* \mid M \text{ accepts } x \text{ in at most } p(|x|) \text{ steps}\}$$

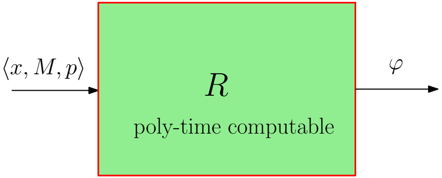
Input: M, x, p.

Question: Does M stops on input x after $p(|x|)$ steps?

Describe a reduction R that computes from M, x, p a **SAT** formula φ.

- R takes as input a string x and outputs a SAT formula φ
- R runs in time polynomial in $|x|, |M|$
- $x \in L$ if and only if φ is satisfiable

The plot against SAT continued



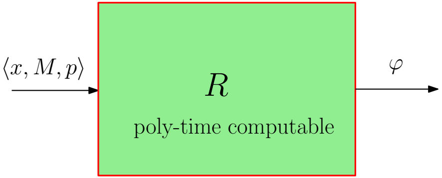
φ is satisfiable if and only if $x \in L$
φ is satisfiable if and only if nondeterministic M accepts x in $p(|x|)$ steps

BIG IDEA

- φ will express "M on input x accepts in $p(|x|)$ steps"
- φ will encode a computation history of M on x

φ: CNF formula s.t if we have a satisfying assignment to it $\implies$ accepting computation of M on x down to the last details (where the head is, what transition is chosen, what the tape contents are, at each step, etc).

The plot against SAT continued



φ is satisfiable if and only if $\boldsymbol{x} \in \boldsymbol{L}$

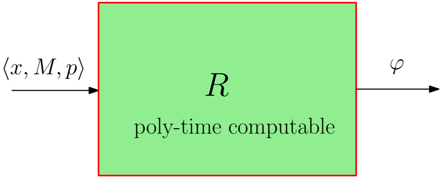
φ is satisfiable if and only if nondeterministic $\boldsymbol{M}$ accepts $\boldsymbol{x}$ in $\boldsymbol{p}(|\boldsymbol{x}|)$ steps

BIG IDEA

- φ will express "$\boldsymbol{M}$ on input $\boldsymbol{x}$ accepts in $\boldsymbol{p}(|\boldsymbol{x}|)$ steps"
- φ will encode a computation history of $\boldsymbol{M}$ on $\boldsymbol{x}$

φ: CNF formula s.t if we have a satisfying assignment to it $\implies$ accepting computation of $\boldsymbol{M}$ on $\boldsymbol{x}$ down to the last details (where the head is, what transition is chosen, what the tape contents are, at each step, etc).

The plot against SAT continued



φ is satisfiable if and only if $x \in L$

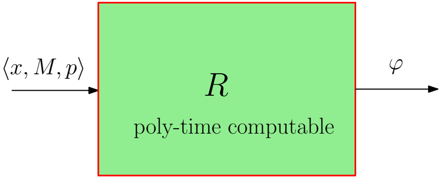
φ is satisfiable if and only if nondeterministic M accepts x in $p(|x|)$ steps

BIG IDEA

- φ will express "M on input x accepts in $p(|x|)$ steps"
- φ will encode a computation history of M on x

φ: CNF formula s.t if we have a satisfying assignment to it $\implies$ accepting computation of M on x down to the last details (where the head is, what transition is chosen, what the tape contents are, at each step, etc).

The plot against SAT continued



φ is satisfiable if and only if $x \in L$
φ is satisfiable if and only if nondeterministic M accepts x in $p(|x|)$ steps

BIG IDEA

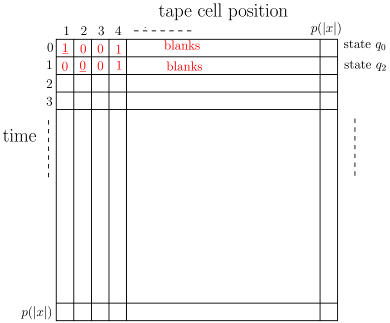
- φ will express "M on input x accepts in $p(|x|)$ steps"
- φ will encode a computation history of M on x

φ: CNF formula s.t if we have a satisfying assignment to it $\implies$ accepting computation of M on x <u>down to the last details</u> (where the head is, what transition is chosen, what the tape contents are, at each step, etc).

The Matrix Executions

Tableau of Computation

M runs in time $p(|x|)$ on x. Entire computation of M on x can be represented by a "tableau"



Row i gives contents of all cells at time i

At time $\mathbf{0}$ tape has input x followed by blanks

Each row long enough to hold all cells M might ever have scanned.

Variables of φ

Four types of variables to describe computation of M on x

- $T(b, h, i)$: tape cell at position h holds symbol b at time i.
  For $h = 1, \ldots, p(|x|)$, $b \in \Gamma$, $i = 0, \ldots, p(|x|)$.
- $H(h, i)$: read/write head is at position h at time i.
  Fir $h = 1, \ldots, p(|x|)$, and $i = 0, \ldots, p(|x|)$
- $S(q, i)$ state of M is q at time i.
  For all $q \in Q$ and $i = 0, \ldots, p(|x|)$.
- $I(j, i)$ instruction number j is executed at time i
  M is non-deterministic, need to specify transitions in some way. Number transitions as $1, 2, \ldots, \ell$ where jth transition is $< q_j, b_j, q'_j, b'_j, d_j >$ indication $(q'_j, b'_j, d_j) \in \delta(q_j, b_j)$, direction $d_j \in \{-1, 0, 1\}$.

Number of variables is $O(p(|x|)^2|M|^2)$

Notation

Some abbreviations for ease of notation

$\bigwedge_{k=1}^{m} x_k$ means $x_1 \wedge x_2 \wedge \ldots \wedge x_m$

$\bigvee_{k=1}^{m} x_k$ means $x_1 \vee x_2 \vee \ldots \vee x_m$

$\bigoplus(x_1, x_2, \ldots, x_k)$ is a formula that means **exactly one** of $x_1, x_2, \ldots, x_m$ is true. Can be converted to CNF form

---

CNF formula showing making sure that at most one variable is assigned value $\mathbf{1}$:

$$\bigwedge_{1 \le i < j \le k} (\overline{x_i} \vee \overline{x_j})$$

Making sure that one of the variables is true: $\bigvee_{i=1}^{k} x_i$.

$$\bigoplus(x_1, x_2, \ldots, x_k) = \bigwedge_{1 \le i < j \le k} (\overline{x_i} \vee \overline{x_j}) \bigwedge (x_1 \vee x_2 \vee \cdots \vee x_k).$$

Notation

Some abbreviations for ease of notation

$\bigwedge_{k=1}^{m} x_k$ means $x_1 \wedge x_2 \wedge \ldots \wedge x_m$

$\bigvee_{k=1}^{m} x_k$ means $x_1 \vee x_2 \vee \ldots \vee x_m$

$\bigoplus(x_1, x_2, \ldots, x_k)$ is a formula that means **exactly one** of $x_1, x_2, \ldots, x_m$ is true. Can be converted to CNF form

---

CNF formula showing making sure that at most one variable is assigned value $\mathbf{1}$:

$$\bigwedge_{1 \le i < j \le k} (\overline{x_i} \vee \overline{x_j})$$

Making sure that one of the variables is true: $\bigvee_{i=1}^{k} x_i$.

$$\bigoplus(x_1, x_2, \ldots, x_k) = \bigwedge_{1 \le i < j \le k} (\overline{x_i} \vee \overline{x_j}) \bigwedge (x_1 \vee x_2 \vee \cdots \vee x_k).$$

Notation

Some abbreviations for ease of notation

$\bigwedge_{k=1}^{m} x_k$ means $x_1 \wedge x_2 \wedge \ldots \wedge x_m$

$\bigvee_{k=1}^{m} x_k$ means $x_1 \vee x_2 \vee \ldots \vee x_m$

$\bigoplus(x_1, x_2, \ldots, x_k)$ is a formula that means **exactly one** of $x_1, x_2, \ldots, x_m$ is true. Can be converted to CNF form

---

CNF formula showing making sure that at most one variable is assigned value $\mathbf{1}$:

$$\bigwedge_{1 \le i < j \le k} (\overline{x_i} \vee \overline{x_j})$$

Making sure that one of the variables is true: $\bigvee_{i=1}^{k} x_i$.

$$\bigoplus(x_1, x_2, \ldots, x_k) = \bigwedge_{1 \le i < j \le k} (\overline{x_i} \vee \overline{x_j}) \bigwedge (x_1 \vee x_2 \vee \cdots \vee x_k).$$

Notation

Some abbreviations for ease of notation
$\bigwedge_{k=1}^{m} x_k$ means $x_1 \wedge x_2 \wedge \ldots \wedge x_m$

$\bigvee_{k=1}^{m} x_k$ means $x_1 \vee x_2 \vee \ldots \vee x_m$

$\bigoplus(x_1, x_2, \ldots, x_k)$ is a formula that means **exactly one** of $x_1, x_2, \ldots, x_m$ is true. Can be converted to CNF form

---

CNF formula showing making sure that at most one variable is assigned value $\mathbf{1}$:

$$\bigwedge_{1 \le i < j \le k} (\overline{x_i} \vee \overline{x_j})$$

Making sure that one of the variables is true: $\bigvee_{i=1}^{k} x_i$.

$$\bigoplus(x_1, x_2, \ldots, x_k) = \bigwedge_{1 \le i < j \le k} (\overline{x_i} \vee \overline{x_j}) \bigwedge (x_1 \vee x_2 \vee \cdots \vee x_k).$$

Clauses of φ

φ is the conjunction of **8** clause groups:

$$\varphi = \bigwedge_{i=1}^{12} \varphi_i$$

where each φ_i is a CNF formula. Described in subsequent slides.

Property: φ is satisfied $\iff$ there is an execution of M on x that accepts the language in $p(|x|)$ time.

THE END

…

(for now)