Fundamental Computer Science
Lecture 4: Complexity
The Cook-Levin theorem

Denis Trystram
MoSIG1 and M1Info – University Grenoble-Alpes

March, 2021

Objective

- Exhibit a problem that belongs to NP-COMPLETE

The satisfiability problem

- $X = \{x_1, x_2, \ldots, x_n\}$: set of variables
- $C = \{C_1, C_2, \ldots, C_m\}$: set of clauses
- $\mathcal{F} = C_1 \wedge C_2 \wedge \ldots \wedge C_m$

$$\mathrm{SAT} = \{\langle \mathcal{F} \rangle \mid \mathcal{F} \text{ is a satisfiable Boolean formula } \}$$

Cook-Levin theorem

Theorem

(original formulation) $\mathrm{SAT} \in \mathcal{P}$ *if and only if* $\mathcal{P} = \mathcal{NP}$.

equivalently: SAT is NP-$\textsc{complete}$.

SAT $\in$ NP-complete

SAT $\in \mathcal{NP}$

Informally, given an assignment of variables, we scan all clauses to check if they evaluate to TRUE

The verifier:

1. Generate **non-deterministically** an interpretation function
2. Evaluate this function and if it is TRUE then *accept*

The cost is in $\Theta(n)$

The idea behind the reduction

Coding the execution of a NDTM by means of a boolean expression.

More precisely, we transform any language of $\mathcal{NP}$ to the encoding of the positive instances of SAT.
We extend here the transformation to a pair (word, language)[1]

- We exhibit such a transformation (reduction) that associates to each word ω and to each language L of $\mathcal{NP}$ an instance of SAT that is positive iff $\omega \in L$.

The technical point is to show how to code the data related to L: all the informations on the tape and also all the states and transitions.

Then, we will verify that it is polynomial in $|\omega|$ for any fixed language.

[1] usually, it is restricted to a word

How to proceed?

The only characterization of the languages in $\mathcal{NP}$ is to be accepted by a non-deterministic Turing Machine.

Thus, we will build such a transformation:

- given a NDTM M $(K, \Sigma, \Gamma, \Delta, start, halt)$ and an input word ω, it produces a positive instance of SAT iff M accepts ω

Express the execution as a SAT formula

$A \leq_{\mathcal{P}} \mathrm{SAT}$ for every language $A \in \mathcal{NP}$

- M: a Non-Deterministic Turing Machine that *decides* A in polynomial time, say n^k
- Each configuration is described by a **state**, the **position of the header** and the **content of the tape**.
- An execution is thus fully described by three vectors/table.

Express the execution as a SAT formula

- **Tape:** create a table $T[i][j]$ of size $n^k \times n^k$, we donot count the initial state of the tape at row "0"
  - each row $T[i]$ corresponds to a configuration of the tape
- the head is recorded into a vector called P
- the current state is on vector Q
- An extra vector is introduced for the (non-deterministic) choice of the transition
- a table is **accepting** if any row is an accepting configuration

We introduce the variable x_{ijs} that means that the symbol s is in $T[i][j]$

SAT $\in$ NP-complete

For each i, j, s, where $1 \leq i, j \leq n^k$ and $s \in \Gamma \cup \Sigma$, define a variable

$$x_{i,j,s} = \begin{cases} \text{TRUE} & \text{if the cell in row } i \text{ and column } j \text{ contains the symbol } s \\ \text{FALSE} & \text{otherwise} \end{cases}$$

Define clauses to guarantee the calculation of M

SAT $\in$ NP-COMPLETE

For each i, j, s, where $1 \le i, j \le n^k$ and $s \in \Gamma \cup \Sigma$, define a variable

$$x_{i,j,s} = \begin{cases} \text{TRUE} & \text{if the cell in row } i \text{ and column } j \text{ contains the symbol } s \\ \text{FALSE} & \text{otherwise} \end{cases}$$

Define clauses to guarantee the calculation of M

- there is exactly one symbol in each cell

$$\phi_{\mathsf{cell}} = \bigwedge_{0 \le i,j \le n^k} \left[\left(\bigvee_{s \in \Gamma \cup \Sigma} x_{i,j,s} \right) \wedge \left(\bigwedge_{\substack{s,t \in \Gamma \cup \Sigma \\ s \ne t}} (\bar{x}_{i,j,s} \vee \bar{x}_{i,j,t}) \right) \right]$$

We verify similarly that:

- there is a unique configuration
- the header is pointing only a unique cell
- there is a only one choice for a transition

These conditions are expressed as boolean CNF expressions.

Other conditions

- the first row corresponds to the starting configuration
- each configuration is obtained from the previous one by a transition $c_i \vdash_M c_{i+1}$
- there is an accepting state before n^k steps

Can you write these conditions as CNF expressions?

The first row corresponds to the starting configuration

The word ω is on the tape and all other cells are filled by $\sqcup$.
The header should be at the left of this input word.
The process starts at the first state.

$$\phi_{\mathsf{init}} = \left[\left(\bigwedge_{0 \le j \le n-1} x_{0,j,\omega_{n+1}} \right) \wedge \left(\bigwedge_{n \le j \le n^k} (x_{0,j,\sqcup}) \right) \right] \wedge p_{0,0} \wedge q_{0,start}$$

- This boolean expression is in CNF and its length is in $O(n^k)$

There is an accepting state

- there is an accepting state

$$\phi_{\mathsf{accept}} = \bigvee_{1 \leq i \leq n^k} q_{i,halt}$$

The hardest part

Proving that the successive configurations are valid (it is in accordance with the transition table).

This is done through two conditions

- All the cells of the tape that are not concerned by the header are not modified.
- The transformation from a step to the next is valid.

The not concerned cells are unchanged

$$\phi_{\mathsf{move}} = \bigwedge_{0 \leq i,j < n^k, s \in \Gamma \cup \Sigma} \left[(x_{i,j,s} \wedge \bar{p}_{i,j}) \Rightarrow x_{(i+1),j,s} \right]$$

that can be written as a CNF as follows

$$\phi_{\mathsf{move}} = \bigwedge_{0 \leq i,j \leq n^k, s \in \Gamma \cup \Sigma} \left[\bar{x}_{i,j,s} \vee p_{i,j} \vee x_{(i+1),j,s} \right]$$

The not concerned cells are unchanged

$$\phi_{\mathsf{move}} = \bigwedge_{0 \leq i,j < n^k, s \in \Gamma \cup \Sigma} \left[(x_{i,j,s} \wedge \bar{p}_{i,j}) \Rightarrow x_{(i+1),j,s} \right]$$

that can be written as a CNF as follows

$$\phi_{\mathsf{move}} = \bigwedge_{0 \leq i,j \leq n^k, s \in \Gamma \cup \Sigma} \left[\bar{x}_{i,j,s} \vee p_{i,j} \vee x_{(i+1),j,s} \right]$$

The last condition is left to the reader.

SAT $\in$ NP-COMPLETE

Construct $\mathcal{F} = \phi_{\text{cell}} \wedge \phi_{\text{start}} \wedge \phi_{\text{accept}} \wedge \phi_{\text{move}}$

- $\mathcal{F}$ has $O(n^k)$ variables and clauses

SAT $\in$ NP-complete

Construct $\mathcal{F} = \phi_{\text{cell}} \wedge \phi_{\text{start}} \wedge \phi_{\text{accept}} \wedge \phi_{\text{move}}$

- $\mathcal{F}$ has $O(n^k)$ variables and clauses

Theorem: $\mathcal{F}$ is satisfiable if and only if A is decided by M