COL 352 Introduction to Automata and Theory of Computation

Nikhil Balaji

Bharti 420
Indian Institute of Technology, Delhi
nbalaji@cse.iitd.ac.in

April 30, 2023

Lecture 34: Computational Complexity Theory (Part 3)

Recap

- Time complexity

Recap

- Time complexity
- TIME(n), NTIME(n)

Recap

- Time complexity
- TIME(n), NTIME(n)
- Verifier algorithms

Recap

- Time complexity
- TIME(n), NTIME(n)
- Verifier algorithms
- Examples: SAT, 3-SAT, k-Clique, Subset Sum.

Recap

- Time complexity
- TIME(n), NTIME(n)
- Verifier algorithms
- Examples: SAT, 3-SAT, k-Clique, Subset Sum.
- P, NP, coNP, EXP, NEXP

Recap

- Time complexity
- TIME(n), NTIME(n)
- Verifier algorithms
- Examples: SAT, 3-SAT, k-Clique, Subset Sum.
- P, NP, coNP, EXP, NEXP
- Hierarchy theorems

Recap

- Time complexity
- TIME(n), NTIME(n)
- Verifier algorithms
- Examples: SAT, 3-SAT, k-Clique, Subset Sum.
- P, NP, coNP, EXP, NEXP
- Hierarchy theorems
- Today: NP-completeness

Polynomial time reductions and NP-hardness

Definition

A language L is said to be NP-hard if for every language $L' \in$ NP, there is a polynomial time reduction such that $L' \leq_m L$.

Polynomial time reductions and NP-hardness

Definition

A language L is said to be NP-hard if for every language $L' \in$ NP, there is a polynomial time reduction such that $L' \leq_m L$.

Definition

A language L is said to be NP-complete if the following two conditions hold:

Polynomial time reductions and NP-hardness

Definition

A language L is said to be NP-hard if for every language $L' \in$ NP, there is a polynomial time reduction such that $L' \leq_m L$.

Definition

A language L is said to be NP-complete if the following two conditions hold:

L is in NP.

Polynomial time reductions and NP-hardness

Definition

A language L is said to be NP-hard if for every language $L' \in$ NP, there is a polynomial time reduction such that $L' \leq_m L$.

Definition

A language L is said to be NP-complete if the following two conditions hold:

- L is in NP.
- L is NP-hard.

Theorem

3-SAT is polynomial time reducible to k-Clique.

$\phi = (a_1 \vee b_1 \vee c_1) \wedge \ldots (a_k \vee b_k \vee c_k)$

Polynomial time reductions and NP-hardness

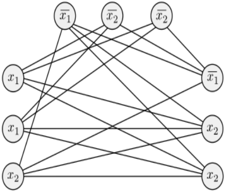
Definition

A language L is said to be NP-hard if for every language $L' \in$ NP, there is a polynomial time reduction such that $L' \leq_m L$.

Definition

A language L is said to be NP-complete if the following two conditions hold:

- L is in NP.
- L is NP-hard.



Polynomial time reductions and NP-hardness

Definition

A language L is said to be NP-hard if for every language $L' \in$ NP, there is a polynomial time reduction such that $L' \leq_m L$.

Definition

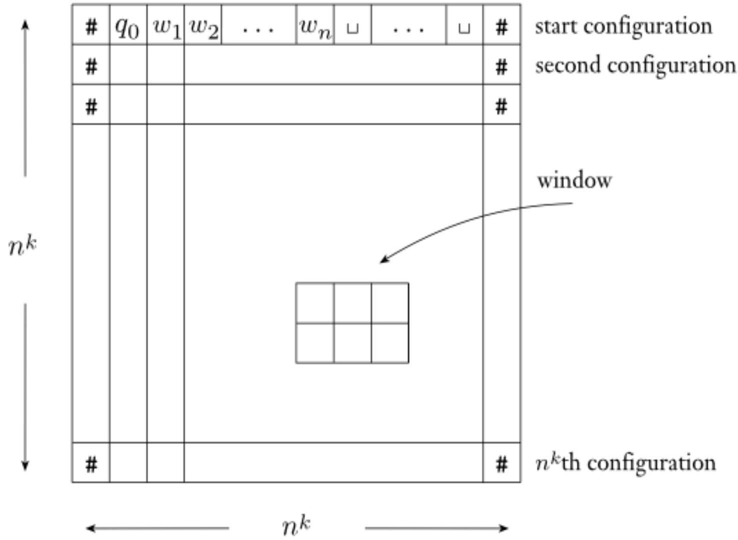
A language L is said to be NP-complete if the following two conditions hold:

- L is in NP.
- L is NP-hard.

Theorem ([Cook-Levin, 1970])

SAT is NP-complete. If L is NP-complete and $L \in P$ then, P = NP.

Proof sketch



Proof sketch

- Let $Q :=$ States, $\Gamma :=$ Tape alphabet. $C = Q \cup \Gamma \cup \{\#\}$
- Variables: $\forall i, j \in [n^k], k \in C, \quad x_{i,j,s}$

Proof sketch

- Let Q := States, Γ := Tape alphabet. $C = Q \cup \Gamma \cup \{\#\}$
- Variables: $\forall i, j \in [n^k], k \in C, \quad x_{i,j,s}$
- $cell[i,j]$: If $x_{i,j,s} = 1, cell[i,j] = s$.

Proof sketch

- Let Q := States, Γ := Tape alphabet. $C = Q \cup \Gamma \cup \{\#\}$
- Variables: $\forall i, j \in [n^k], k \in C, \quad x_{i,j,s}$
- $cell[i,j]$: If $x_{i,j,s} = 1, cell[i,j] = s$.
- $\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$

Proof sketch

- Let Q := States, Γ := Tape alphabet. $C = Q \cup \Gamma \cup \{\#\}$
- Variables: $\forall i, j \in [n^k], k \in C, \quad x_{i,j,s}$
- $cell[i,j]$: If $x_{i,j,s} = 1, cell[i,j] = s$.
- $\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$

$$\phi_{cell} = \bigwedge_{1 \le i,j \le n^k} ((\vee_{s \in C} x_{i,j,s}) \wedge (\wedge_{s,t \in C, s \neq t} \overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}))$$

Proof sketch

- Let Q := States, Γ := Tape alphabet. $C = Q \cup \Gamma \cup \{\#\}$
- Variables: $\forall i, j \in [n^k], k \in C, \quad x_{i,j,s}$
- $cell[i,j]$: If $x_{i,j,s} = 1, cell[i,j] = s$.
- $\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$

$$\phi_{cell} = \bigwedge_{1 \le i,j \le n^k} ((\vee_{s \in C} x_{i,j,s}) \wedge (\wedge_{s,t \in C, s \neq t} \overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}))$$

$$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge \ldots x_{1,3,w_1} \wedge \cdots \wedge x_{1,n^k,\#}$$

Proof sketch

- Let Q := States, Γ := Tape alphabet. $C = Q \cup \Gamma \cup \{\#\}$
- Variables: $\forall i, j \in [n^k], k \in C, \quad x_{i,j,s}$
- $cell[i,j]$: If $x_{i,j,s} = 1, cell[i,j] = s$.
- $\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$

$$\phi_{cell} = \bigwedge_{1 \le i,j \le n^k} \left((\vee_{s \in C} x_{i,j,s}) \wedge (\wedge_{s,t \in C, s \neq t} \overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right)$$

$$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge \ldots x_{1,3,w_1} \wedge \cdots \wedge x_{1,n^k,\#}$$

$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

Proof sketch

- Let $Q :=$ States, $\Gamma :=$ Tape alphabet. $C = Q \cup \Gamma \cup \{\#\}$
- Variables: $\forall i, j \in [n^k], k \in C, \quad x_{i,j,s}$
- $cell[i,j]$: If $x_{i,j,s} = 1, cell[i,j] = s$.
- $\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$

$$\phi_{cell} = \bigwedge_{1 \le i,j \le n^k} ((\vee_{s \in C} x_{i,j,s}) \wedge (\wedge_{s,t \in C, s \neq t} \overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}))$$

$$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge \ldots x_{1,3,w_1} \wedge \cdots \wedge x_{1,n^k,\#}$$

$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

$$\phi_{move} = \bigwedge_{1 \le i < n^k, 1 \le j < n^k} \text{the (i,j) window is legal}$$

Proof sketch

- Let Q := States, Γ := Tape alphabet. $C = Q \cup \Gamma \cup \{\#\}$
- Variables: $\forall i, j \in [n^k], k \in C, \quad x_{i,j,s}$
- $cell[i,j]$: If $x_{i,j,s} = 1, cell[i,j] = s$.
- $\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$

$$\phi_{cell} = \bigwedge_{1 \le i,j \le n^k} \left(\left(\vee_{s \in C} x_{i,j,s} \right) \wedge \left(\wedge_{s,t \in C, s \neq t} \overline{x_{i,j,s}} \vee \overline{x_{i,j,t}} \right) \right)$$

$$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge \ldots x_{1,3,w_1} \wedge \cdots \wedge x_{1,n^k,\#}$$

$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

$$\phi_{move} = \bigwedge_{1 \le i < n^k, 1 \le j < n^k} \text{the (i,j) window is legal}$$