CS 390 HW 4 Q2

Aaryan Wadhwani

A polynomial time algorithm for (3) k-occurrence-SAT implies a polynomial time algorithm for (CNF-)SAT.

Reduction from CNF-SAT to 3-Occurrence-SAT

Transformation Steps

Replace Repeated Variables
For each variable x_i that appears t times in ϕ, create t new variables $x_{i,1}, x_{i,2}, \dots, x_{i,t}$. Replace the j-th occurrence of x_i (or $\neg x_i$) in ϕ with the corresponding copy $x_{i,j}$. After this step:

- Each new variable $x_{i,j}$ appears in **exactly one clause** of the original formula ϕ.
- No variable appears more than once in the transformed formula (so far).

Add Consistency Clauses
To ensure all copies of x_i have the same truth value, add a cycle of implication clauses. For example, if x_i appears $t = 3$ times, add the following implications:

$$x_{i,1} \to x_{i,2}, \quad x_{i,2} \to x_{i,3}, \quad x_{i,3} \to x_{i,1}.$$

Each implication $A \to B$ is logically equivalent to the clause $\neg A \lor B$. Thus, the above implications become:

$$(\neg x_{i,1} \lor x_{i,2}), \quad (\neg x_{i,2} \lor x_{i,3}), \quad (\neg x_{i,3} \lor x_{i,1}).$$

For a general t, add the clauses:

$$(\neg x_{i,1} \lor x_{i,2}),\ (\neg x_{i,2} \lor x_{i,3}),\ \dots,\ (\neg x_{i,t-1} \lor x_{i,t}),\ (\neg x_{i,t} \lor x_{i,1}).$$

These clauses enforce that all copies $x_{i,1}, \dots, x_{i,t}$ must be logically equivalent.
Count Variable Occurrences
After Step 2:

- Each $x_{i,j}$ appears in **1 original clause** (from Step 1).
- Each $x_{i,j}$ appears in **2 implication clauses** (from Step 2: one as $\neg x_{i,j}$, one as $x_{i,j}$).

Thus, every variable in the new formula ϕ' has **exactly 3 occurrences**.

Proof of Correctness

Forward Direction (ϕ is satisfiable $\Rightarrow$ ϕ' is satisfiable)
Suppose ϕ has a satisfying assignment. For each variable x_i in ϕ, assign all its copies $x_{i,1}, x_{i,2}, \ldots, x_{i,t}$ the same value as x_i.

- *Original Clauses:* Satisfied because we replaced x_i with copies that match x_i's value.
- *Consistency Clauses:* Satisfied because all copies of x_i are equal. For example:
  - If $x_{i,1} = \text{True}$, then $x_{i,2} = \text{True}$ (from $x_{i,1} \to x_{i,2}$).
  - If $x_{i,1} = \text{False}$, then $\neg x_{i,1} = \text{True}$, so $(\neg x_{i,1} \vee x_{i,2})$ is satisfied regardless of $x_{i,2}$.

Reverse Direction (ϕ' is satisfiable $\Rightarrow$ ϕ is satisfiable)
Suppose ϕ' has a satisfying assignment. The consistency clauses force all copies $x_{i,1}, \ldots, x_{i,t}$ to have the same value. To see why, consider the implications:

- $x_{i,1} \to x_{i,2}$ (i.e., $\neg x_{i,1} \vee x_{i,2}$): If $x_{i,1} = \text{True}$, then $x_{i,2} = \text{True}$.
- $x_{i,2} \to x_{i,3}$ (i.e., $\neg x_{i,2} \vee x_{i,3}$): If $x_{i,2} = \text{True}$, then $x_{i,3} = \text{True}$.
- $\ldots$
- $x_{i,t} \to x_{i,1}$ (i.e., $\neg x_{i,t} \vee x_{i,1}$): If $x_{i,t} = \text{True}$, then $x_{i,1} = \text{True}$.

This creates a cycle where all $x_{i,j}$ must be equal. Now, assign each original x_i the value of its copies. This assignment satisfies ϕ because every original clause in ϕ was satisfied by at least one copy in ϕ'.

Why the Cycle Ensures Equality
The cyclic implications ensure that all copies $x_{i,1}, x_{i,2}, \ldots, x_{i,t}$ are equal. Here's why:

- Suppose $x_{i,1} = \text{True}$. Then:
  - $x_{i,1} \to x_{i,2}$ forces $x_{i,2} = \text{True}$.
  - $x_{i,2} \to x_{i,3}$ forces $x_{i,3} = \text{True}$.
  - This continues until $x_{i,t} \to x_{i,1}$ forces $x_{i,1} = \text{True}$, which is consistent.
- Suppose $x_{i,1} = \text{False}$. Then:
  - $x_{i,1} \to x_{i,2}$ simplifies to $\neg x_{i,1} \vee x_{i,2}$. Since $\neg x_{i,1} = \text{True}$, this clause is satisfied regardless of $x_{i,2}$.
  - However, $x_{i,2} \to x_{i,3}$ must also hold. If $x_{i,2} = \text{True}$, then $x_{i,3} = \text{True}$, and so on, until $x_{i,t} \to x_{i,1}$ forces $x_{i,1} = \text{True}$, which contradicts $x_{i,1} = \text{False}$.
  - Therefore, the only consistent assignment is $x_{i,2} = \text{False}$, $x_{i,3} = \text{False}$, and so on, until $x_{i,t} = \text{False}$.

Thus, the cyclic implications force all copies $x_{i,j}$ to have the same value as $x_{i,1}$.

Polynomial-Time Argument

The transformation takes time proportional to the size of ϕ:

- **Variables:** If ϕ has m clauses, each variable x_i with t occurrences generates t new variables. Total new variables = $O(m)$.
- **Clauses:** For each variable x_i with t occurrences, we add t new clauses. Total new clauses = $O(m)$.

Thus, the reduction runs in linear time relative to the size of ϕ.

If 3-occurrence-SAT could be solved in polynomial time, this reduction would allow solving CNF-SAT in polynomial time.