21.6.2
Reducing SAT to 3SAT

SAT $\leq_P$ 3SAT

How SAT is different from 3SAT?

In **SAT** clauses might have arbitrary length: $1, 2, 3, \ldots$ variables:

$$\left(x \vee y \vee z \vee w \vee u\right) \wedge \left(\neg x \vee \neg y \vee \neg z \vee w \vee u\right) \wedge \left(\neg x\right)$$

In **3SAT** every clause must have exactly 3 different literals.

To reduce from an instance of **SAT** to an instance of **3SAT**, we must make all clauses to have exactly 3 variables...

Basic idea

1. Pad short clauses so they have 3 literals.
2. Break long clauses into shorter clauses.
3. Repeat the above till we have a 3CNF.

SAT $\leq_P$ 3SAT

How **SAT** is different from **3SAT**?

In **SAT** clauses might have arbitrary length: $1, 2, 3, \ldots$ variables:

$$\left(x \vee y \vee z \vee w \vee u\right) \wedge \left(\neg x \vee \neg y \vee \neg z \vee w \vee u\right) \wedge \left(\neg x\right)$$

In **3SAT** every clause must have **exactly** 3 different literals.

To reduce from an instance of **SAT** to an instance of **3SAT**, we must make all clauses to have exactly 3 variables...

Basic idea

1. Pad short clauses so they have 3 literals.
2. Break long clauses into shorter clauses.
3. Repeat the above till we have a $3\mathrm{CNF}$.

3SAT $\leq_P$ SAT

1. **3SAT** $\leq_P$ **SAT**.
2. Because...
   A **3SAT** instance is also an instance of **SAT**.

SAT $\leq_P$ 3SAT

Claim 21.3.

SAT $\leq_P$ 3SAT.

Given φ a SAT formula we create a 3SAT formula φ' such that

1. φ is satisfiable $\iff$ φ' is satisfiable.
2. φ' can be constructed from φ in time polynomial in $|\varphi|$.

Idea: if a clause of φ is not of length 3, replace it with several clauses of length exactly 3.

SAT $\leq_P$ 3SAT

Claim 21.3.

SAT $\leq_P$ 3SAT.

Given φ a **SAT** formula we create a **3SAT** formula φ' such that

1. φ is satisfiable $\iff$ φ' is satisfiable.
2. φ' can be constructed from φ in time polynomial in $|\varphi|$.

Idea: if a clause of φ is not of length 3, replace it with several clauses of length exactly 3.

SAT $\leq_P$ 3SAT

Claim 21.3.

SAT $\leq_P$ 3SAT.

Given φ a **SAT** formula we create a **3SAT** formula φ' such that

1. φ is satisfiable $\iff$ φ' is satisfiable.
2. φ' can be constructed from φ in time polynomial in $|\varphi|$.

Idea: if a clause of φ is not of length $\mathbf{3}$, replace it with several clauses of length exactly $\mathbf{3}$.

SAT $\leq_P$ 3SAT

A clause with two literals

Reduction Ideas: clause with 2 literals

1. Case clause with 2 literals: Let $c = \ell_1 \vee \ell_2$. Let u be a new variable. Consider

$$c' = \left(\ell_1 \vee \ell_2 \vee u\right) \wedge \left(\ell_1 \vee \ell_2 \vee \neg u\right).$$

2. Suppose $\varphi = \psi \wedge c$. Then $\varphi' = \psi \wedge c'$ is satisfiable $\iff$ φ is satisfiable.

SAT $\leq_P$ 3SAT

A clause with a single literal

Reduction Ideas: clause with 1 literal

1. Case clause with one literal: Let c be a clause with a single literal (i.e., $c = \ell$). Let u, v be new variables. Consider

$$c' = \left(\ell \vee u \vee v\right) \wedge \left(\ell \vee u \vee \neg v\right) \wedge \left(\ell \vee \neg u \vee v\right) \wedge \left(\ell \vee \neg u \vee \neg v\right).$$

2. Suppose $\varphi = \psi \wedge c$. Then $\varphi' = \psi \wedge c'$ is satisfiable $\iff$ φ is satisfiable.

SAT $\leq_P$ 3SAT

A clause with more than 3 literals

Reduction Ideas: clause with more than 3 literals

1. Case clause with five literals: Let $c = \ell_1 \vee \ell_2 \vee \ell_3 \vee \ell_4 \vee \ell_5$. Let u be a new variable. Consider

$$c' = \left(\ell_1 \vee \ell_2 \vee \ell_3 \vee u\right) \wedge \left(\ell_4 \vee \ell_5 \vee \neg u\right).$$

2. Suppose $\varphi = \psi \wedge c$. Then $\varphi' = \psi \wedge c'$ is satisfiable $\iff$ φ is satisfiable.

SAT $\leq_P$ 3SAT

A clause with more than 3 literals

Reduction Ideas: clause with more than 3 literals

1. Case clause with $k > 3$ literals: Let $c = \ell_1 \vee \ell_2 \vee \ldots \vee \ell_k$. Let u be a new variable. Consider

$O(k)$

$$c' = (\ell_1 \vee \ell_2 \ldots \ell_{k-2} \vee u) \wedge (\ell_{k-1} \vee \ell_k \vee \neg u).$$

2. Suppose $\varphi = \psi \wedge c$. Then $\varphi' = \psi \wedge c'$ is satisfiable $\iff$ φ is satisfiable.

Breaking a clause

Lemma 21.4.

For any boolean formulas $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *and* $\boldsymbol{z}$ *a new boolean variable. Then*

$$\boldsymbol{X} \vee \boldsymbol{Y} \text{ is satisfiable}$$

if and only if, $\boldsymbol{z}$ *can be assigned a value such that*

$$\left(\boldsymbol{X} \vee \boldsymbol{z}\right) \wedge \left(\boldsymbol{Y} \vee \neg \boldsymbol{z}\right) \text{ is satisfiable}$$

(with the same assignment to the variables appearing in $\boldsymbol{X}$ *and* $\boldsymbol{Y}$*).*

SAT $\leq_P$ 3SAT (contd)

Clauses with more than 3 literals

Let $c = \ell_1 \vee \cdots \vee \ell_k$. Let $u_1, \ldots u_{k-3}$ be new variables. Consider

$$c' = \Big(\ell_1 \vee \ell_2 \vee u_1\Big) \wedge \Big(\ell_3 \vee \neg u_1 \vee u_2\Big)$$
$$\wedge \Big(\ell_4 \vee \neg u_2 \vee u_3\Big) \wedge$$
$$\cdots \wedge \Big(\ell_{k-2} \vee \neg u_{k-4} \vee u_{k-3}\Big) \wedge \Big(\ell_{k-1} \vee \ell_k \vee \neg u_{k-3}\Big).$$

Claim 21.5.

$\varphi = \psi \wedge c$ is satisfiable $\iff$ $\varphi' = \psi \wedge c'$ is satisfiable.

Another way to see it — reduce size of clause by one:

$$c' = \Big(\ell_1 \vee \ell_2 \ldots \vee \ell_{k-2} \vee u_{k-3}\Big) \wedge \Big(\ell_{k-1} \vee \ell_k \vee \neg u_{k-3}\Big).$$

An Example

Example 21.6.

$$\varphi = \left(\neg x_1 \vee \neg x_4\right) \wedge \left(x_1 \vee \neg x_2 \vee \neg x_3\right)$$
$$\wedge \left(\neg x_2 \vee \neg x_3 \vee x_4 \vee x_1\right) \wedge \left(x_1\right).$$

Equivalent form:

$$\psi = (\neg x_1 \vee \neg x_4 \vee z) \wedge (\neg x_1 \vee \neg x_4 \vee \neg z)$$
$$\wedge\ (x_1 \vee \neg x_2 \vee \neg x_3)$$
$$\wedge\ (\neg x_2 \vee \neg x_3 \vee y_1) \wedge (x_4 \vee x_1 \vee \neg y_1)$$
$$\wedge\ (x_1 \vee u \vee v) \wedge (x_1 \vee u \vee \neg v)$$
$$\wedge\ (x_1 \vee \neg u \vee v) \wedge (x_1 \vee \neg u \vee \neg v).$$

An Example

Example 21.6.

$$\varphi = \left(\neg x_1 \vee \neg x_4\right) \wedge \left(x_1 \vee \neg x_2 \vee \neg x_3\right) \wedge \left(\neg x_2 \vee \neg x_3 \vee x_4 \vee x_1\right) \wedge \left(x_1\right).$$

Equivalent form:

$$\begin{aligned}\psi = {} & (\neg x_1 \vee \neg x_4 \vee z) \wedge (\neg x_1 \vee \neg x_4 \vee \neg z) \\ & \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \\ & \wedge (\neg x_2 \vee \neg x_3 \vee y_1) \wedge (x_4 \vee x_1 \vee \neg y_1) \\ & \wedge (x_1 \vee u \vee v) \wedge (x_1 \vee u \vee \neg v) \\ & \wedge (x_1 \vee \neg u \vee v) \wedge (x_1 \vee \neg u \vee \neg v).\end{aligned}$$

An Example

Example 21.6.

$$\varphi = \left(\neg x_1 \lor \neg x_4\right) \land \left(x_1 \lor \neg x_2 \lor \neg x_3\right)$$
$$\land \left(\neg x_2 \lor \neg x_3 \lor x_4 \lor x_1\right) \land \left(x_1\right).$$

Equivalent form:

$$\begin{aligned}\psi = {} & (\neg x_1 \lor \neg x_4 \lor z) \land (\neg x_1 \lor \neg x_4 \lor \neg z) \\ & \land (x_1 \lor \neg x_2 \lor \neg x_3) \\ & \land (\neg x_2 \lor \neg x_3 \lor y_1) \land (x_4 \lor x_1 \lor \neg y_1) \\ & \land (x_1 \lor u \lor v) \land (x_1 \lor u \lor \neg v) \\ & \land (x_1 \lor \neg u \lor v) \land (x_1 \lor \neg u \lor \neg v).\end{aligned}$$

An Example

Example 21.6.

$$\varphi = \left(\neg x_1 \vee \neg x_4\right) \wedge \left(x_1 \vee \neg x_2 \vee \neg x_3\right)$$
$$\wedge \left(\neg x_2 \vee \neg x_3 \vee x_4 \vee x_1\right) \wedge \left(x_1\right).$$

Equivalent form:

3SAT

$$\begin{aligned}\psi = {} & (\neg x_1 \vee \neg x_4 \vee z) \wedge (\neg x_1 \vee \neg x_4 \vee \neg z) \\ & \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \\ & \wedge (\neg x_2 \vee \neg x_3 \vee y_1) \wedge (x_4 \vee x_1 \vee \neg y_1) \\ & \wedge (x_1 \vee u \vee v) \wedge (x_1 \vee u \vee \neg v) \\ & \wedge (x_1 \vee \neg u \vee v) \wedge (x_1 \vee \neg u \vee \neg v).\end{aligned}$$

Overall Reduction Algorithm

Reduction from **SAT** to **3SAT**

```
ReduceSATTo3SAT(φ):
    // φ: CNF formula.
    for each clause c of φ do
        if c does not have exactly 3 literals then
            construct c′ as before
        else
            c′ = c
    ψ is conjunction of all c′ constructed in loop
    return Solver3SAT(ψ)
```

Correctness (informal)

φ is satisfiable $\iff$ ψ is satisfiable because for each clause c, the new $3\mathrm{CNF}$ formula c' is logically equivalent to c.

THE END

...

(for now)