5CCS2FC2: Foundations of Computing II

Introduction to SAT Solving

Week 5

Dr Christopher Hampson

Department of Informatics

King's College London

Why is SAT important?

Why is SAT important?

- **Why SAT?**
  - The first example of a problem shown to be **NP-complete**
    - All NP problems can be **reduced to SAT**
    - Finding a **polynomial time** algorithm for SAT proves that **P = NP**

Why is SAT important?

- **Why SAT?**
  - Logic acts as a **specification language** to describe real-world problems.
    - Reductions *to* SAT are **often straightforward**
    - Even if **P** $\neq$ **NP** we still want to be able to answer **real-world problems** as quickly as possible

Given the importance of SAT, a great deal of research has been invested in finding **efficient algorithms** for checking satisfiability.

Why is SAT important?

- **SAT Solvers**
  - **SAT Solvers** are used in industrial application to solve instances of SAT,
  - Modern solvers can cope with formulas with **millions of literals**

A Greedy Approach

A Greedy SAT Algorithm

GREEDY-SAT (C):

1. Randomly select a variable assignment V
2. **While** there is some unsatisifed clause in C
    2. Count the number of *unsatisifed* clauses
    3. **For each** variable P:
        4. Count the number of clauses satisfied by *flipping* the assignment of P
    5. **If** no improvement: **Return** False
    3. Update V to the best improvement
7. **Return** True

- **A Greedy SAT Algorithm**

Step 1) Guess a variable assignment at random!

$P :=$ True, $Q :=$ False, $R :=$ False

Step 2) Evaluate you guess by counting the number of *satisfied clauses*

$(\neg P \vee Q \vee R)$ ✗ $(P \vee Q)$ ✓ 1 $(Q \vee R)$ ✗ $(\neg P \vee Q)$ ✗

$(\neg P \vee \neg Q \vee R)$ ✓ 2 $(\neg P \vee \neg Q)$ ✓ 3 $(P \vee \neg Q \vee R)$ ✓ 4

- **A Greedy SAT Algorithm**

Step 3) Consider the effect of **swapping** a single variable from **True** to **False** (or vice versa)

Step 4) Update the assignment to the assignment that leads the the **biggest increase** in the number of satisfied clauses

Step 5) Repeat until no further improvements are possible.

$P :=$ **True** $P :=$ **False**, $Q :=$ **True** $Q :=$ **False**, $R :=$ **True** $R :=$ **False**,

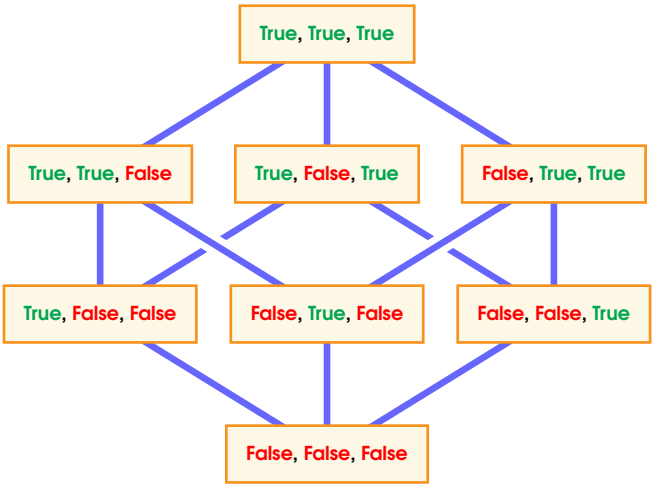
A Greedy SAT Algorithm

Theorem The Greedy SAT Algorithm runs in polynomial time.

Proof:

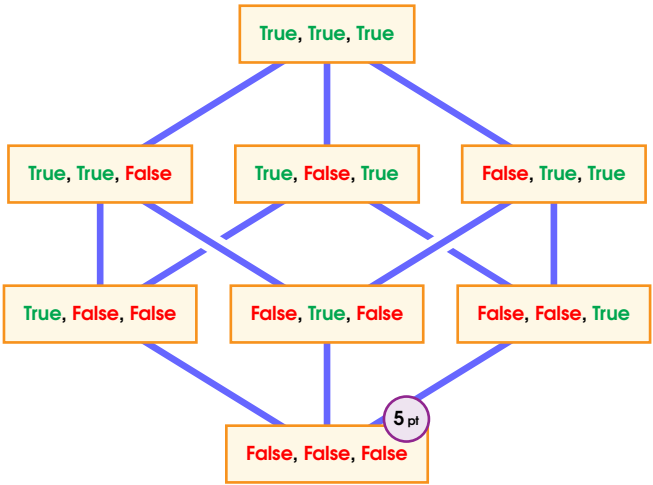
- Selecting a **random assignment** and **evaluating** the 'score' of the assignment can both be done in **linear time**.
- Each **iteration** involves evaluating at most n assignments that different in a single variable, requring at most a **polynomial** number of steps
- Since we never **decrease** our 'score', we iterate at most a **polynomial** number of times.

A Greedy SAT Algorithm



$$(P \vee Q)$$
$$(P \vee R)$$
$$(P \vee \neg Q \vee R)$$
$$(\neg P \vee \neg Q)$$
$$(\neg P \vee Q)$$
$$(\neg P \vee \neg R)$$
$$(P \vee Q \vee \neg R)$$

A Greedy SAT Algorithm



$(P \vee Q)$ ✗

$(P \vee R)$ ✗

$(P \vee \neg Q \vee R)$ ✓

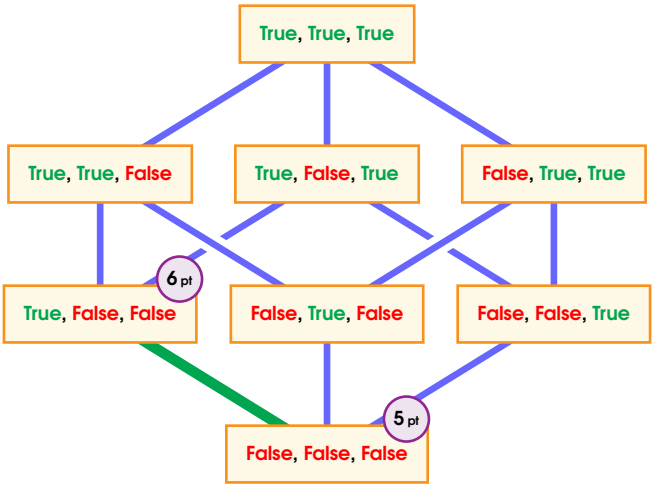
$(\neg P \vee \neg Q)$ ✓

$(\neg P \vee Q)$ ✓

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

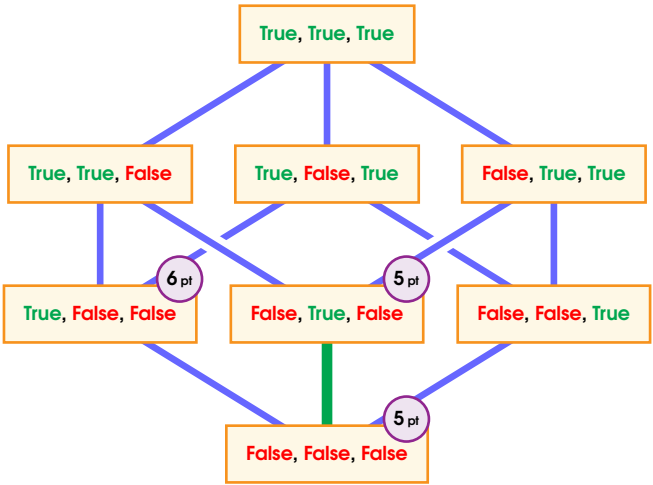
$(\neg P \vee \neg Q)$ ✓

$(\neg P \vee Q)$ ✗

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✗

$(P \vee \neg Q \vee R)$ ✗

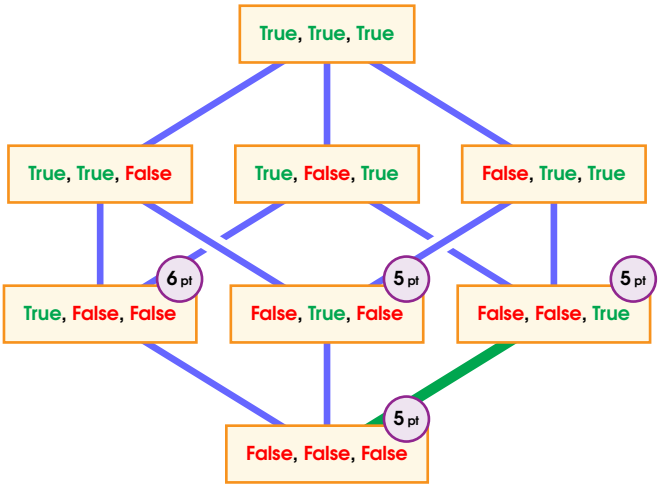
$(\neg P \vee \neg Q)$ ✓

$(\neg P \vee Q)$ ✓

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✗

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

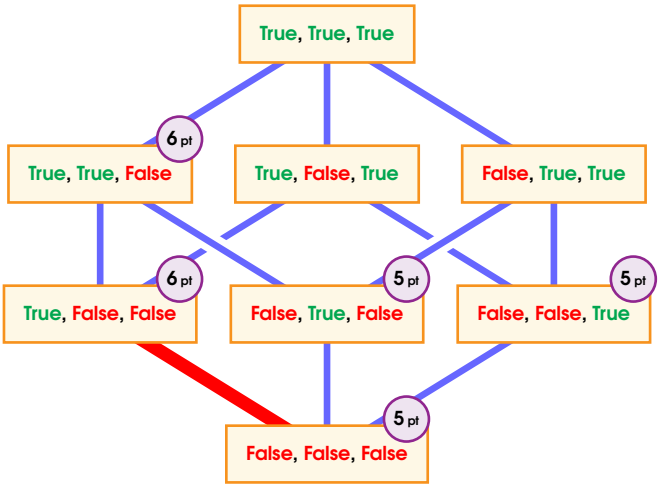
$(\neg P \vee \neg Q)$ ✓

$(\neg P \vee Q)$ ✓

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✗

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

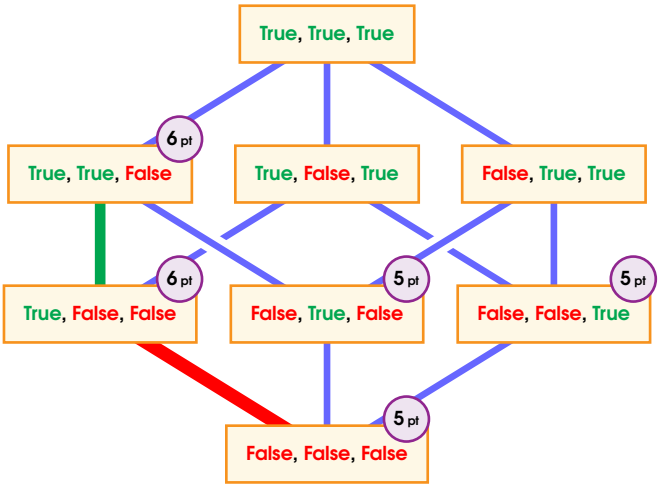
$(\neg P \vee \neg Q)$ ✓

$(\neg P \vee Q)$ ✗

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

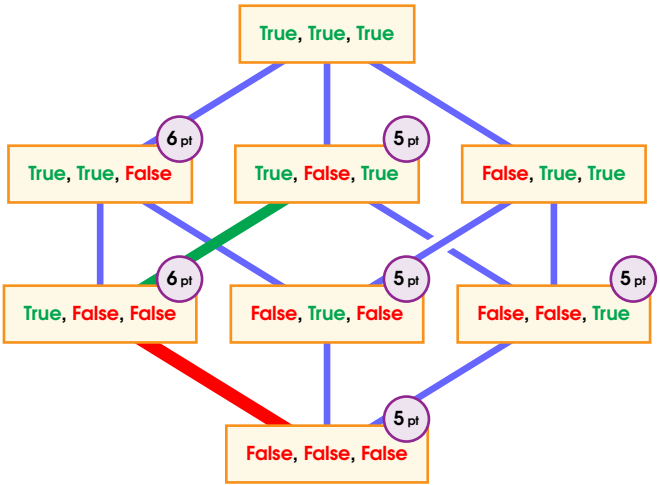
$(\neg P \vee \neg Q)$ ✗

$(\neg P \vee Q)$ ✓

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

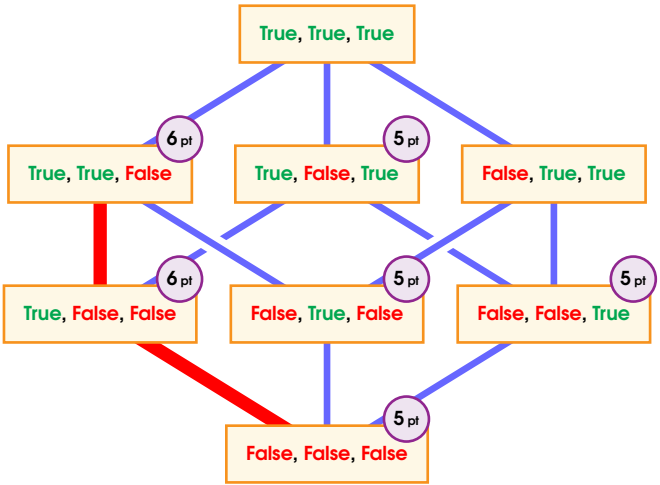
$(\neg P \vee \neg Q)$ ✓

$(\neg P \vee Q)$ ✗

$(\neg P \vee \neg R)$ ✗

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

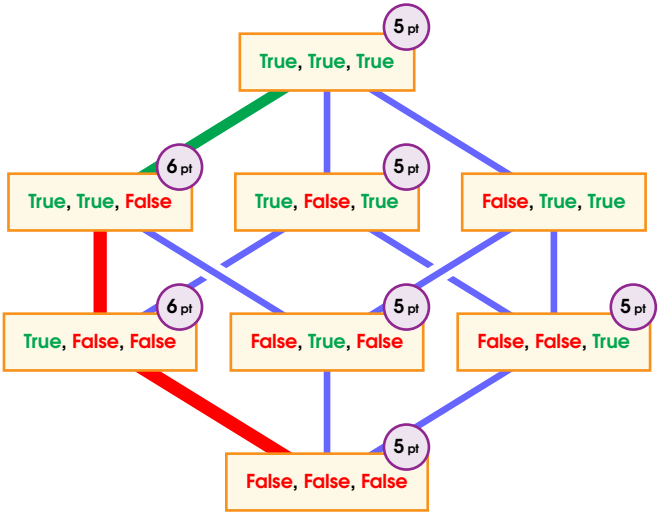
$(\neg P \vee \neg Q)$ ✗

$(\neg P \vee Q)$ ✓

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

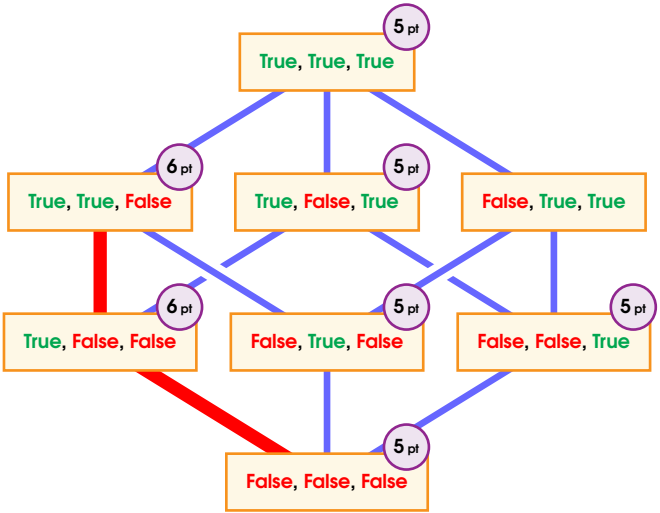
$(\neg P \vee \neg Q)$ ✗

$(\neg P \vee Q)$ ✓

$(\neg P \vee \neg R)$ ✗

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

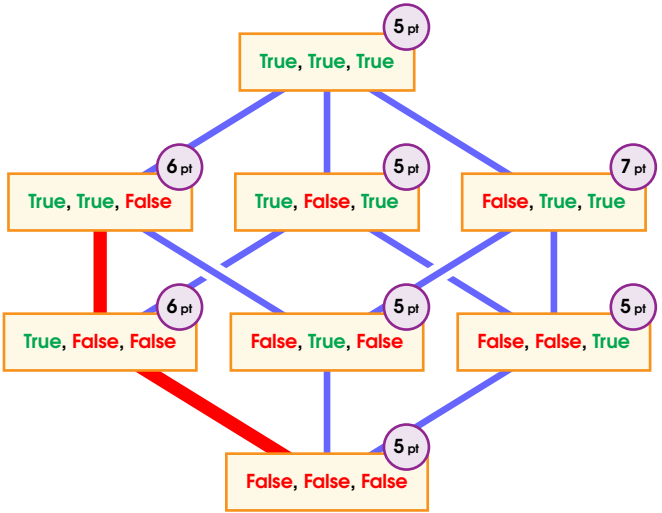
$(\neg P \vee \neg Q)$ ✗

$(\neg P \vee Q)$ ✓

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✓

A Greedy SAT Algorithm



$(P \vee Q)$ ✓

$(P \vee R)$ ✓

$(P \vee \neg Q \vee R)$ ✓

$(\neg P \vee \neg Q)$ ✓

$(\neg P \vee Q)$ ✓

$(\neg P \vee \neg R)$ ✓

$(P \vee Q \vee \neg R)$ ✓

End of Slides!