Lecture 13

October 3, 2005

In the previous lecture we showed that 3-COLOR is an NP-Complete problem by reducing it to 3-SAT. Recall that 3-SAT is a special case of SAT where the formulas have a special structure:

$$\varphi = \bigwedge_i c_i \text{ where } c_i = (a \vee b \vee c)$$

where a,b and c are *literals*, i.e. atomic propositions or negations of atomic propositions.

1 3-SAT $\leq_p$ 3-COLOR

In this lecture we show a reduction in the other direction: 3-SAT $\leq_p$ 3-COLOR. We introduce a correspondence between the formula and a graph, so that the formula is satisfiable if and only if the graph is 3-colorable.

First we define some auxiliary nodes which correspond to true (T), false (F) and a third color, say, R. We also add a node for each proposition and its negation. Figure 1 illustrates how we can enforce the rule that a proposition p and its negation are assigned different colors, and also that each one is either assigned the color T or F.

For each clause we add three nodes (corresponding to the literals in the clause). Figure 1 shows how to enforce the rule that in each clause at least one of the literals is colored T. In the figure the nodes a, b and c correspond to the literals in a clause. The "widget" will not be 3-colorable if all three are colored F.

Formally, the graph is defined as follows:

$G_\varphi(V, E)$:

•

$$\begin{aligned} V &= \{F, T, R\} \\ &\cup \{p : p \in AP(\varphi)\} \\ &\cup \{\neg p : p \in AP(\varphi)\} \\ &\cup \{\overline{a}, \overline{b}, \overline{c} : \{(a \vee b \vee c) \in \varphi\} \end{aligned}$$

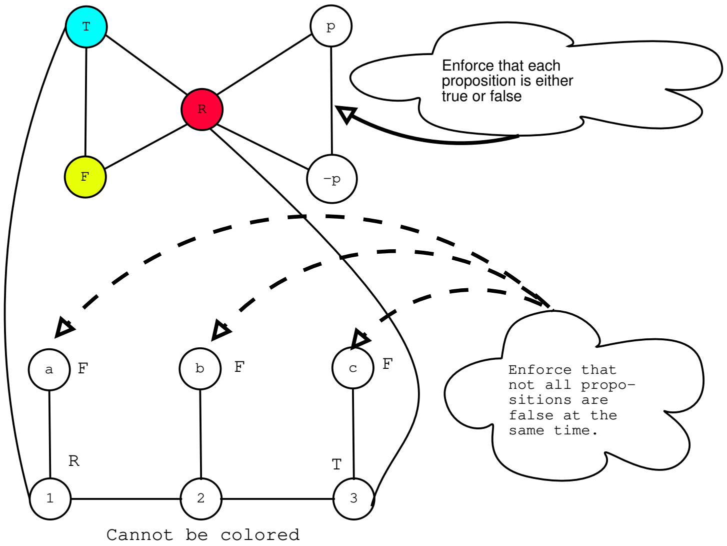

Figure 1: The "widget"

- $$\begin{aligned} E &= \{(R,T),(R,F),(F,T)\} \\ &\cup \{(p,\neg p), p \in AP(\varphi)\} \\ &\cup \{(R,p), p \in AP(\varphi)\} \\ &\cup \{(R,\neg p), p \in AP(\varphi)\} \\ &\cup \{(\overline{a},a),(\overline{b},b),(\overline{c},c) : (a \vee b \vee c) \in \varphi\} \\ &\cup \{(\overline{a},\overline{b}),(\overline{b},\overline{c}),(\overline{a},T),(\overline{c},R)\} \end{aligned}$$

For a single clause $(a \vee b \vee c)$ there are 8 possible truth assignments. We need to check that for each one of them the corresponding "widget" is 3-colorable. In Figure 2 we show one of them.

To prove the correctness of our construction, we consider the following lemma.

Lemma 1. *Let φ be a 3-CNF formula, and let G_φ be defined as above. Then φ is satisfiable iff G_φ is 3-colorable*

Proof.

1. $\boxed{\Rightarrow}$ Suppose that $\tau \models \varphi$. Then there is a 3-coloring of G_φ defined as following: $c_\tau(p) = c(T)$ iff $\tau(p) = 1$ and $c_\tau(\neg p) = c(F)$ iff $\tau(p) = 1$. Each one of the clauses is satisfiable by τ, and we already know that each satisfiable clause is 3-colorable, therefore a coloring for the whole graph exists.

2. $\boxed{\Leftarrow}$ Suppose that $C : V \to \{1, 2, 2\}$ is a 3-coloring. Then there is a corresponding satisfying truth assignment τ_c defined as following: $\tau_c(p) =$

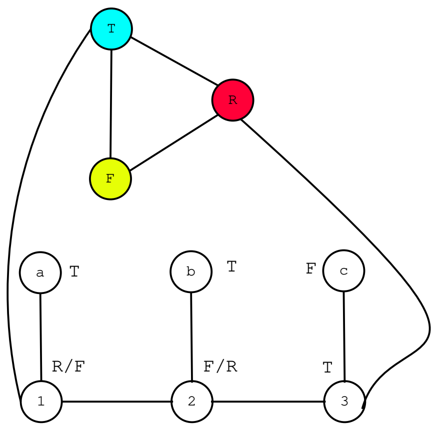

Figure 2: A "satisfiable" clause is 3-colorable

1 iff $c(p) = c(T)$. By construction we know that if each "widget" is colorable if and only if the corresponding clause is satisfiable, therefore φ is satisfiable.

□

2 Developing an algorithm for solving SAT

Since SAT is a very important problem, we would like to develop efficient algorithms for its solution. In the remainder of this lecture we will prepare the ground, and in the next lecture we will introduce the algorithm itself.

Since the problem is NP-Complete, the algorithm to solve it is going to be exponential (to the best of our knowledge so far). Yet, given two exponential algorithms, one can be exponentially better than the other! This means that we should look for possible optimizations. In particular, below we give two examples where the problem could be solved really easily because the formula has some porperty that we can detect.

- Consider $\varphi = (p \vee q \vee \neg r) \wedge (q \vee \neg r \vee s) \wedge \ldots \wedge s \wedge (\neg s)$. The formula can be arbitrarily long, but still, because it contains a contradiction $s \wedge (\neg s)$ it is unsatisfiable no matter what the truth assignment is to the other variables!

- Another example is a formula like $\varphi = p \wedge \psi$, where ψ is also a formula. Although we cannot immediatly say anything about the final result, we can reduce the problem in size. Specifically, p **must** be true, otherwise the formula is obviously not satisfiable. So we can assign to proposition p its only possible truth value, and propagate this assignment to ψ. Thus, p becomes a constant and the original formula that we had to check is transformed to an equivalent formula which is simpler since it has one less variable.

The first bullet could be characterized as a heuristic method. We try to detect shortcuts that might give as a faster solution. The second is more systematic as we could progressivlely substitute variables with constants and thus simplify the formula a little bit in each step. One has to be careful, though. We have not yet defined formulas containing symbols such as **0** (false) or **1** (true). To be able to use such formulas we must first extend our definition of Form:

$$FORM' = FORM \cup \{0, 1\}.$$

This practically means that we now consider the constants 0 and 1 as formulas. To do this we need to define what our standard connectives mean in this new context:

- $\neg : FORM' \to FORM'$.
- $\circ : FORM' \times FORM' \to FORM'$

are defined as follows:

$$\neg(\varphi) = \begin{cases} 1, & \varphi = 0 \\ 0, & \varphi = 1 \\ (\neg\varphi), & \varphi \in FORM \end{cases},$$

$$\vee(\varphi, \psi) = \begin{cases} 1 & \varphi = 1 \\ 1 & \psi = 1 \\ 0 & \varphi = 0 \text{ and } \psi = 0 \\ (\varphi \vee \psi) & \varphi, \psi \in FORM \\ \psi & \varphi = 0 \text{ and } \psi \in FORM \\ \varphi & \psi = 0 \text{ and } \varphi \in FORM \end{cases},$$

$$\wedge(\varphi, \psi) = \begin{cases} 0 & \varphi = 0 \\ 0 & \psi = 0 \\ 1 & \varphi = 1 \text{ and } \psi = 1 \\ (\varphi \wedge \psi) & \varphi, \psi \in FORM \\ \psi & \varphi = 1 \text{ and } \psi \in FORM \\ \varphi & \psi = 1 \text{ and } \varphi \in FORM \end{cases},$$

$$\to(\varphi, \psi) = \begin{cases} 0 & \varphi = 1 \text{ and } \psi = 0 \\ 1 & \varphi = 0 \text{ or } \psi = 1 \\ (\varphi \to \psi) & \varphi, \psi \in FORM \\ \psi & \varphi = 1 \text{ and } \psi \in FORM \\ \neg\varphi & \psi = 0 \text{ and } \varphi \in FORM \end{cases},$$

$$\leftrightarrow(\varphi,\psi)=\begin{cases}0 & \varphi=1 \text{ and } \psi=0\\ 0 & \varphi=0 \text{ and } \psi=1\\ 1 & \varphi=1 \text{ and } \psi=1\\ 1 & \varphi=0 \text{ and } \psi=0\\ (\varphi\leftrightarrow\psi) & \varphi,\psi\in FORM\\ \psi & \varphi=1 \text{ and } \psi\in FORM\\ \varphi & \psi=1 \text{ and } \varphi\in FORM\\ \neg\varphi & \psi=0 \text{ and } \varphi\in FORM\\ \neg\psi & \varphi=0 \text{ and } \psi\in FORM\end{cases}.$$

Finally, we define *simplification*.
Let $\varphi \in Form$. Let $c \in \{0,1\}$. Let $p \in AP$.

$$\varphi[p\mapsto c]=\begin{cases}c & \text{if} \quad \varphi=p\\ q & \text{if} \quad \text{for some } q\neq p,\ \varphi=q\\ \neg(\varphi'[p\mapsto c]) & \text{if} \quad \varphi=(\neg\varphi')\\ \circ(\varphi_1[p\mapsto c],\varphi_2[p\mapsto c]) & \text{if} \quad \varphi=(\varphi_1 o \varphi_2)\end{cases}$$

The following lemma, which we will prove in the next lecture, states that by substituting a proposition with its truth assignment, we don't change the satisfiability of the formula.

Lemma 2. *Let $\varphi \in Form$ and $\tau \in 2^{AP(\varphi)}$, $p \in AP(\varphi)$. Then $\tau \models \varphi$ iff $\tau\ |_{AP(\varphi)-\{p\}} \models \varphi[p \mapsto \tau(p)]$.*