Lecture 9

September 16, 2005

1 Satisfiability

1.1 Classification of Formulas

Remember the 2 classifications of problems we have discussed in the past: Satisfiable and Valid. The Classification Problem is to determine whether a given formula φ is valid and/or satisfiable.

Lemma 1 (Validity vs. Satisfiability) *There is a duality between validity and satisfiability:*

1. *φ is valid iff $(\neg\varphi)$ is unsatisfiable.*
2. *φ is satisfiable iff $(\neg\varphi)$ is not valid.*

Thus, the satisfiability problem and the validity problem are related, but as we will show in lecture 11, the two problems are not polynomially equivalent. For the time being we focus on the satisfiability problem.

1.2 Variety of Satisfiability Problems

Satisfiability Problem: Given $\varphi \in Form$, is φ satisfiable?

Satisfiability is something that turns up in many contexts. Here are some examples. For each formula, we ask "is it satisfiable?".

1. Boolean Equation: $(((p \to q) \wedge (q \to r)) \wedge p) \wedge (\neg r))$
2. System of Linear Equations: $x + y = 5; x - y = 3$.
3. System of Linear Inequalities: $x + y < 5; x - y > 3$.
4. Quadratic Equations: $x^2 + 5x - 7 = 0$.
5. Cubic Equations: $x^3 + 3x^2 - x + 4 = 0$.
6. Quartic Equations: $3x^4 + 4x^2 - 12 = 0$.
7. Quintic Equations: $7x^5 - x^3 + 3 - 0$.

8. Polynomial Equations: $x^2y + yx^2 - 12 = 0$.

Each one of these examples either may or may not be solvable. In each case, satisfiability is defined (informally) as the existence of some instantiation of the variables such that the formula may be reduced to *true*. Note that the variables in each case range over some domain, and there are constraints over the possible assignments of values to the variables. Given these constraints, we are asked to find a satisfying assignment. This sort of satisfiability problem is known as a Constrained Satisfiability Problem.

1.3 Solutions to Satisfiability Problems

One might hope, given that these problems have a lot in common, that there is some general solution for solving all of them. Unfortunately (or fortunately, if you are a mathematician), this is not the case. Various methods have been developed for solving each class of problems, as outlined below:

- Linear Equalities: These can be solved using Gaussian elimination in PTime.
- Linear Inequalities: In high school, we learned to solve these graphically. This implies that there is no easy algorithm. A solution was developed a bit before World War II as a way of solving optimization problems when allocating bombers. This was followed by the Simplex Method, discovered by George Dantzig in the 1940s, and which was the method of choice for many years. Then, in 1974, someone analyzed this algorithm and showed that it was exponential. In 1979, the Russian mathematician Kachian developed the Interior Method for solving this problem. This method is in PTime.
- Quadratic Equations, Cubic Equations, Quartic Equations: A closed form solution can be found for these in PTime.
- Quintic Equations and Polynomial Equations: It can be shown that no closed form solution exists for general quintic or higher degree equation exists. (Some special cases may have one, though).

These observations assume that we are dealing with real numbers. The problems become even more difficult if we are dealing with integers, because many optimization methods such as Steepest Descent, Newton's Method, etc, do not work with integers.

An example of this is Diophantime Equations, also known as Hilbert's 10th problem. Yuri Matiyaserich, a Russian mathematician, proved this problem is undecidable in the 1970s. From this, we can see that even if problems appear to have the same structure, small changes to a problem can significantly change its difficulty.

1.4 Algorithm

The most simple algorithm for determining satisfiability goes as follows:

```
SAT(φ)
        B := False
        for all τ ∈ 2^{AP(φ)} do
           B = B ∨ Eval(φ,τ)
        od
        return B
```

This algorithm is clearly correct, giving rise to the following lemma:

Lemma 2 *SAT(φ)=1 iff φ is satisfiable.*

So the problem is in its running time. It takes $2^{AP(\varphi)}$ times the cost of doing an Eval. We could lower the average running time a bit by terminating the loop when Eval returns true, but in the worst case, the running time remains the same.

These kinds of algorithm are called "Exhaustive Search" or "Brute-force Search" algorithms, since they look through every possible element in the search space, and try it to see if it fulfills the requirement.

1.5 Graph Colorability

Now let's consider a related problem. Is there a way to assign students to colleges based on several constraints (of the form student x cannot share a college with student y)? Since this problem involves a number of elements (students) with some kind of relationship between them, a good way to model this is with a graph. So, we can represent this problem as follows:

Given a graph $G = (V, E)$, we want to find a coloring $c : v \rightarrow 1, ..., k$ such that if $(u, v) \in$ E then $c(u) \neq c(v)$. This general representation is known as the k-colorability problem.

A specific subset of the k-colorability problem is the map colorability problem. Are all maps 4-colorable? Clearly, not all graphs are 4-colorable, so the map problem restricts itself to planar graphs. This leads to the following theorem:

Theorem 1 (4-Color Theorem) *Every planar graph is 4-colorable.*

Although no one can find a counter-example, an elegant proof for this has been difficult to find. Two mathematicians in the 1970s used case analysis to show that all maps reduce to 1800 cases. They then wrote a program to verify all 1800 cases. In the 1990s, different mathematicians used case analysis to reduce the number of cases to 700 and again verified them with a computer. These are the best proofs that have been found.

Graph coloring is related to satisfiability because exhaustive search works for both, and if someone gives you a solution it is easy to verify. This leads to 2 important properties of this class of problems: 1. Verifying a solution is easy. 2. Finding a solution is hard. By analogy, it is easier to critique than to create.

2 Complexity Theory

2.1 A Bit of History

This difference between verifying a solution and finding a solution has always been a fundamental problem in mathematics. A basic example of this is the difference between verifying a proof and coming up with a proof on your own. However, the difference did not begin to be formally recognized until the 1950s. Complexity theory itself was not developed until the 1960s.

Then, someone came up with the idea of writing an algorithm that guesses answers and then checks them. If the algorithm is good at guessing, the problem becomes easy. This idea is known as Nondeterminism.

Complexity theory has arisen around the decision problem. Finding whether or not a solution exists seems cleaner than finding a particular solution.

2.2 Formalization

Assume some finite alphabet Σ, with Σ^* the set of all strings over Σ.

Definition 1 *: A Decision Problem is some $L \subseteq \Sigma^*$, about which we ask, given $x \in \Sigma^*$, is $x \in L$?*

An example of this is the question, is "antidisetalishmentariarism" a word in English.

Convention: $\sigma^{\leq i} = \mathrm{u} \in \Sigma^*, |u| \leq i$. In other words, $\sigma^{\leq i}$ means that σ is at most i characters long.

Definition 2 *: A Decision Problem, $L \subseteq \Sigma^*$ is in NP if there is a polynomial P and PTime algorithm A such that: $x \in L$ iff there exists $y \in \Sigma^{\leq P(|x|)}$ such that A(x,y) = 1.*

So, the basic idea is that you guess y, which won't be too long. Then you run the verification algorithm with x and y. There are clearly many possible candidates for y.

y is called 'the witness.' But, like many witnesses, y is usually in the witness protection program and hiding somewhere. This is what makes the problem so difficult—we must first find y.

The witness for satisfiability is $\tau \in 2^{AP(\varphi)}$, a truth assignment. The witness for k-colorability is some $c : v \to 1, ...k$, a mapping of coloring assignments. Both of these are linear witnesses, but there are still many, many of them to choose from.