W4231: Analysis of Algorithms

11/23/99

- NP-completeness of 3SAT, Minimum Vertex Cover, Maximum Independent Set,

Boolean Formulae

A Boolean formula is an expression that we can build starting from Boolean variables $x_1, \ldots, x_n, \ldots$ and then using $\wedge$ (AND) $\vee$ (OR) and $\bar{\cdot}$ (NOT).

For example

$$(x_1 \wedge \bar{x}_3) \vee ((x_4 \vee x_1) \wedge (x_2 \vee \bar{x}_3)) \qquad (1)$$

CNF Formulae

A Boolean formula is in Conjunctive-Normal-Form (CNF) if it is a AND-of-ORs. Negation can be used only as the innermost operator. Formula (1) is not in CNF. The following is in CNF

$$(x_2 \vee \bar{x}_4 \vee x_5) \wedge x_3 \wedge (x_1 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (\bar{x}_2 \vee x_5) \qquad (2)$$

A Boolean formula can be seen as a representation of a Boolean function.

Problem: Given a CNF formula, is there a setting of Boolean values to its variables that makes the formula true? (Name of the problem: SAT.)

SAT

SAT is clearly in NP. In fact it is a special case of Circuit Satisfiability. (Why?)

We want to prove that it is NP-hard. We want to reduce from Circuit Satisfiability.

Doomed approach: Given a circuit, transform it into a Boolean CNF formula that computes the same Boolean function.

Parity: consider the Boolean function that is 1 iff an odd number of inputs is 1. There is a circuit of size $O(n)$ that computes this function for inputs of length n. But the smallest CNF for this function has size more than 2^n.

So we cannot translate a circuit into a CNF formula of comparable size that computes the same function.

But we may be able to transform a circuit into a CNF formula such that the circuit is satisfiable iff the formula is satifiable.

The reduction

We will add new variables.

Suppose the circuit C has m gates, including input gates.

We use variables $g_1, \ldots, g_m$

- variable g_j corresponding to gate j.

We make a formula F which is the AND of $m+1$ sub-expression.

Sub-expressions

There is a sub-expression for every gate j, saying that the value of the variable for that gate is set in accordance to the value of the variables corresponding to inputs for gate j.

We also have a $(m+1)$-th term that says that the output gate outputs 1.

There is no sub-expression for the input gates.

For a gate j, which is a NOT applied to the output of gate i, we have the sub-expression

$$(g_i \vee g_j) \wedge (\bar{g}_i \vee \bar{g}_j)$$

For a gate j, which is a AND applied to the output of gates i and l, we have the sub-expression

$$(\bar{g}_j \vee g_i) \wedge (\bar{g}_j \vee g_l) \wedge (g_j \vee \bar{g}_i \vee \bar{g}_l)$$

Similarly for OR.

Proof of Correctness

Suppose C is satisfiable, then consider setting g_j being equal to the output of the j-th gate of C when a satisfying set of values is given in input. Such a setting for $g_1, \ldots, g_m$ satisfies F.

Suppose F is satisfiable, and give in input to C the part of the assignment to F corresponding to input gates of C. We can prove by induction that the output of gate j in C is also equal to g_j, and therefore the output gate of C outputs 1.

So C is satisfiable if and only if F is satisfiable.

3-CNF Satisfiability

SAT is a much simpler problem than Circuit Satisfiability, if we want to use it as a starting point of NP-completeness proofs. We can use an even simpler starting point: 3SAT

Definition: 3SAT is the same as SAT, except that each OR is on precisely 3 (possibly negates) variables.

E.g.

$$(x_2 \vee \bar{x}_4 \vee x_5) \wedge (x_1 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (\bar{x}_2 \vee x_3 \vee x_5) \qquad (3)$$

Certainly, 3SAT is in NP.

Terminology

Each little OR in a SAT formula is called a **clause**.

Each occurrence of a variable, complemented or not, is called a **literal**.

3SAT is NP-complete

Take a formula F of SAT. We tranform it into a formula F' of 3SAT such that F' is satisfiable if and only if F is satisfiable.

Each clause of F is transformed into a sub-expression of F'.

Clauses of length 3 are left unchanged.

A clause of length 1, such as (x) is changed as follows

$$(x \vee y_1 \vee y_2) \wedge (x \vee y_1 \vee \bar{y}_2)(x \vee \bar{y}_1 \vee y_2) \wedge (x \vee \bar{y}_1 \vee \bar{y}_2)$$

where y_1 and y_2 are two new variables added specifically for the transformation of that clause.

A clause of length 2, such as $x_1 \vee x_2$ is changed as follows

$$(x_1 \vee x_2 \vee y) \wedge (x_1 \vee x_2 \vee \bar{y})$$

where y is a new variable added specifically for the transformation of that clause.

For a clause of length $k \geq 4$, such as $(x_1 \vee \cdots \vee x_k)$, we change it as follows

$$(x_1 \vee x_2 \vee y_1) \wedge (\bar{y}_1 \vee x_3 \vee y_2) \wedge (\bar{y}_2 \vee x_4 \vee y_4) \wedge \cdots \wedge (\bar{y}_{k-3} \vee x_{k-1} \vee x_k$$

where $y_1, \cdots, y_{k-3}$ are new variables added specifically for the transformation of that clause.

Correctness

- We first argue that if F is satisfiable, then there is an assignment that satisfies F'.

  For the shorter clauses, we just set the y-variables arbitrarily. For the longer clause it is slightly more tricky.

- We then argue that if F is not satisfiable, then F' is not satisfiable.

  Fix an assignment to the x variables. Then there is a clause in F that is not satisfied. We argue that one of the derived clauses in F' is not satisfied.

Independent Set

Given an undirected non-weighted graph $G = (V, E)$, an *independent set* is a subset $I \subseteq V$ of the vertices such that no two vertices of I are adjacent.

(Similar to the notion of a *matching*, except that it involves vertices and not edges)

Given a graph, we want to find a largest independent set.

Motivations: executing conflicting tasks; related to the construction of error-correcting codes; special case of more interesting problems

Complexity

Decision version:

- Given a graph G and an integer k, does there exist an independent set in G with at least k vertices?

We are going to prove:

- (The decision version of) the Maximum Independent Set Problem is NP-complete.

It is easy to see that the problem is in NP. We have to see that it is NP-hard.

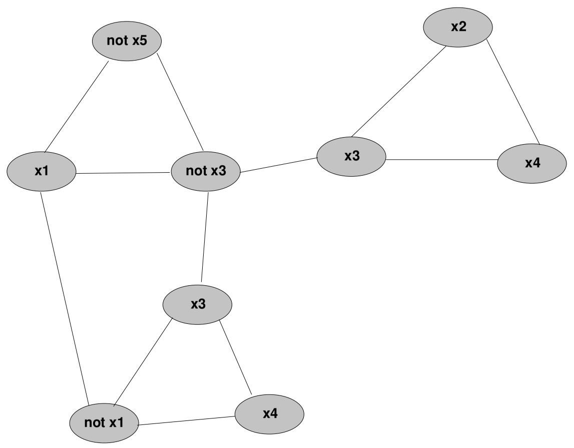
Reduction from 3SAT

We show how to reduce 3SAT to Maximum Independent Set.

Starting from a formula ϕ with n variables $x_1, \ldots, x_n$ and m clauses, we generate a graph G_ϕ with $3m$ vertices, and we show that the graph has an independent set with at least m vertices iff the formula is satisfiable.

The graph G_ϕ has a triangle for every clause in ϕ. The vertices in the triangle correspond to the three literals of the clause.

Vertices in different triangles are joined by an edge iff they correspond to two literals that are one the complement of the other.



Correctness

We have to show that

- If ϕ is satisfiable, then there is an independent set in G_ϕ with at least m vertices.
- If there is an independent set in G with at least m vertices, then ϕ is satisfiable.

From Satisfaction to Independence

Suppose we have an assignment of Boolean values to the variables $x_1, \ldots, x_n$ of ϕ such that all the clauses of ϕ are satisfied.

This means that for every clause, at least on of its literals is satisfied by the assignment.

We construct an independent set as follows: for every triangle pick a node that corresponds to a satisfied literal. Break ties arbitrarily.

It is impossible that two such nodes are adjacent.

From Independence to Satisfaction

Suppose we have an independent set I with m vertices.

We better have exactly one vertex in I for every triangle. (Two vertices in the same triangle are always adjacent.)

Let us fix an assignment that satisfies all the literals that correspond to vertices of I. Assign values to the other variables arbitrarily.

This is a consistent rule to generate an assignment (we cannot have a literal and its negation in the independent set).

Every clause is satisfied by this assignment.

Wrapping Up

We showed a reduction $\phi \to (G_\phi, m)$ that given an instance of 3SAT produces an instance of the decision version of Maximum Independen Set.

We have the property that ϕ is satisfiable (answer YES for the 3SAT problem) if and only if G_ϕ has an independent set of size at least m.

We knew 3SAT is NP-hard.

Then also Max Independent Set is NP-hard; and so also NP-complete.

Similar Problems

Maximum Clique. Given a (undirected non-weighted) graph $G = (V, E)$, a *clique* K is a set of vertices $K \subseteq V$ such that *any two* vertices in K are adjacent. We want to find the biggest clique.

Maximum Clique is NP-hard by reduction from Maximum Independent Set.

Minimum Vertex Cover. Given a (undirected non-weighted) graph $G = (V, E)$, a *vertex cover* C is a set of vertices $C \subseteq V$ such that for every edge $(u, v) \in E$, either $u \in C$ or $v \in C$ (or, possibly, both).

NP-hardness of Minimum Vertex Cover

We show a reduction from Maximum Independent Set. The reduction is based on the following observation:

If you have an independent set I in a graph $G = (V, E)$, then the set of vertices $C = V - I$ that are *not* in I is a vertex cover.

Suppose C is not a vertex cover: then there is some edge (u, v) neither of whose endpoints is in C. This means both the endpoints are in I. So I is not an independent set. Contradiction.

Reduction

Starting from an instance (G, k) of Maximum Independent set we produce an instance $(G, n - k)$ of Minimum Vertex Cover.