Computability and Complexity

Lecture 5, Thursday August 15, 2023

Ari Feiglin

Proposition 5.1:

The decision problem

$$\mathsf{3Color} = \{G \mid \text{There exists a three-coloring of the undirected graph } G\}$$

is **NP**-complete.

Proof:

3Color is obviously in **NP**, as the verifier accepts a graph and a coloring and verifies that the coloring is a valid three coloring. We will define a reduction from SAT to 3Color. Suppose φ is a binary formula in CNF, we define the graph $G = (V, E)$ as follows:

(1) We define three vertices T, F, N and we define edges between all of them. The significance of these vertices is that T represents all the variables which are valuated as true, and F represents those valuated as false.

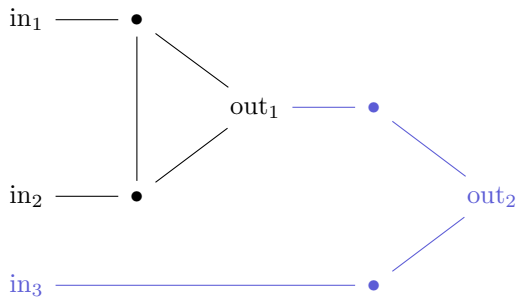
(2) For every variable x_i we define two vertices x_i and $\neg x_i$ where x_i represents the variable x_i being valuated as true, and $\neg x_i$ as x_i being valuated as false.

(3) We define edges $(x_i, \neg x_i)$, and (x_i, N) and $(\neg x_i, N)$. This ensures that each x_i is colored as either the true (the color of T), or false (the color of F).

Thus far, we can see that there is a relation between the coloring of the vertices and the valuation of variables in φ. In particular, we can valuate x_i as true if the vertex x_i has the same color as T, and false otherwise (if it has the same color as F).

(4) In the case that there is a disjunction in φ with a single literal (one variable), we connect the literal (which is either x_i or $\neg x_i$) to the vertex F. This is as the literal must be valuated as true, and so it must be colored as T, and so it is connected to F. We can also assume that no such disjunction exists, as a boolean formula in CNF can be converted to a boolean formula in CNF where every disjunction has every variable.

(5) We define an *OR Gadget* to be a subgraph of the form:



Here the black subgraph is an OR gadget with two inputs, then we take the output of this OR gadget and another input vertex and define another OR gadget (the blue subgraph). This defines an OR gadget with three inputs. This definition can be generalized to any number of input vertices: once we have defined an OR gadget with k inputs, then it has a vertex out_{k-1}. If we have another input node in_{k+1}, then we define the OR gadget with $k+1$ inputs as the OR gadget with inputs $\text{in}_1, \ldots, \text{in}_k$ composed with the OR gadget between out_{k-1} and in_{k+1}.

For every disjunction in φ, we define add the OR gadget between all the literals in the disjunction to the graph. So for example if $x_1 \vee \neg x_2 \vee x_3$ is a disjunction in φ, we define the OR gadget with inputs x_1, $\neg x_2$, and x_3 and add it to the graph.

We define edges between the outputs of all these OR gadgets of the disjunctions with the vertices F and N (to ensure that the output vertex is colored as T).

The following are properties of the OR gadgets: suppose we have an OR gadget with inputs $\text{in}_1, \ldots, \text{in}_k$, and a three coloring σ, then

(1) If $\sigma(\text{in}_1) = \cdots = \sigma(\text{in}_k) = F$, then $\sigma(\text{out}_{k-1}) = F$ as well. This can be proven inductively. It is easy to see why this is true for $k = 2$, and the induction step is simple, as the input for out_i is out_{i-1} and $\in_{i+1}$ which are both F and so by the case for $k = 2$, out_i is colored F as well.

(2) If there exist an in_i which is colored as T, then there exists a coloring of the gadget such that out_{k-1} is colored as T. Again, this reduces down to proving it for the case of $k = 2$.

Obviously G was defined in polynomial time. So now we must prove that $\varphi \in \mathsf{SAT}$ if and only if $G \in \mathsf{3Color}$. Suppose that φ is in SAT, and that τ satisfies φ. Then we color G as follows:

(1) Of course, we color $\sigma(T) = T$, $\sigma(F) = F$, and $\sigma(N) = N$.

(2) And we color $\sigma(x_i) = \tau_i$ and $\sigma(\neg x_i) = \neg\tau_i$.

(3) Since τ satisfies φ, every disjunction has a literal which satisfies it and so by the second property of OR gadgets there exists a three coloring of the OR gadget where the output vertex is colored as T.

So this is a valid three coloring of G, meaning $G \in \mathsf{3Color}$ as required.

Now suppose that $G \in \mathsf{3Color}$, so there exists a three-coloring of G, σ. Let T be the color of the vertex T, F be the color of the vertex F, and N be the color of the vertex N. We define τ_i as the color of the vertex x_i,

$$\tau_i = \sigma(x_i)$$

Since x_i is connected to N, this color is not N so this is indeed a boolean vector. We claim that τ satisfies φ.

Since σ is a three-coloring of G, the color of the output nodes of each OR gadget must be T (as they are connected to the vertices N and F). This means by the first property of OR gadgets that there exists an input node which is colored as T. This in turn means that each disjunction has a literal which is colored as T, and so each disjunction is satisfied by τ. And so φ is satisfied by τ. Thus $\varphi \in \mathsf{SAT}$ and so the map $\varphi \mapsto G$ is a Karp reduction from SAT to $\mathsf{3Color}$, as required. ■

Let us define for every $k \in \mathbb{N}$ the decision problem

$$k\mathsf{Color} = \{G \mid G \text{ is an undirected graph which can be colored with } k \text{ colors}\}$$

for $k = 2$ this is in $\mathbf{P}$, and for $k = 3$ we showed this is in $\mathbf{NP}$. Now, for $k > 3$ we can define a reduction from $\mathsf{3Color}$ to $k\mathsf{Color}$ by taking a graph G and adding a clique of size $k - 3$ and adding edges between the vertices of the rest of the graph to the vertices of this clique. Let this new graph be G'. Then G is three-colorable if and only if G' is k-colorable. This is since the three-coloring of G defines a k-coloring of G', where we color the vertices in G' which are also in G as their coloring in G, and the vertices in the clique are colored in $k - 3$ other colors. And if G' is k-colorable, the clique must use $k - 3$ of these colors, and no other vertex (which is in G) can share a color with a vertex in the clique. Thus this k-coloring of G' defines a three-coloring of G. So $G \mapsto G'$ is a Karp reduction from $\mathsf{3Color}$ to $k\mathsf{Color}$.

Thus we have proven the following.

Proposition 5.2:

The decision problem $k\mathsf{Color}$ is $\mathbf{NP}$-complete if and only if $k \geq 3$.

And the decision problem

$$\mathsf{Color} = \{(G, k) \mid G \text{ is } k\text{-colorable}\}$$

We can define a Karp reduction from $\mathsf{3Color}$ to Color by $f(G) = (G, 3)$. And thus Color is $\mathbf{NP}$-complete.

Let us define the following decision problem:

$$\mathsf{SubsetSum} = \left\{ (A, b) \;\middle|\; \begin{array}{l} A \text{ is a tuple (or finite sequence, or multiset) of natural numbers, and } b \text{ is a natural number} \\ \text{such that there exists a subsequence of } A \text{ whose sum is } b \end{array} \right\}$$

Proposition 5.3:
SubsetSum is **NP**-complete.

Proof:

SubsetSum is obviously in **NP**, as we can define a verifier $V((A,b),N)$ where N is a set of indexes of A, and V simply checks if $\sum_{n\in N} A_n = b$ (the sum of the numbers in A at indexes in N is equal to b). Now to show that SubsetSum is **NP**-complete, we define a Karp reduction from Vertex-Cover to SubsetSum.

Suppose (G,k) is an input for Vertex-Cover (so $G=(V,E)$ is an undirected graph and k is a natural number), we define $n=|V|$ and $m=|E|$. Let $V=\{v_1,\ldots,v_n\}$ and $E=\{e_1,\ldots,e_m\}$. We define a sequence A of length $n+m$ where

(1) For $1\leq i\leq n$, A_i represents the vertex v_i and we define

$$A_i = 10^m + \sum_{(v_i,v_j)=e_k\in E} 10^{k-1}$$

ie. we define a natural number where the kth digit (in base ten) denotes if v_i is in the edge e_k.

(2) For $n<i\leq m$, A_i represents the edge e_i and we define

$$A_i = 10^{i-n-1}$$

So the ith edge defines the value 10^{i-1} in A (since e_i corresponds to A_{i+n}).

Now, we define

$$b = k\cdot 10^m + \sum_{i=1}^{m} 2\cdot 10^{i-1}$$

Meaning in base ten, b is k followed by m twos.

So we must show that $(G,k)\in$ Vertex-Cover if and only if $(A,b)\in$ SubsetSum. If $(G,k)\in$ Vertex-Cover then there exists a vertex cover S of size k. So we define N to be the indexes of the vertices in S and the indexes (plus n) of the edges connected to only one vertex in S. That is

$$N = \{i \mid v_i\in S\}\cup\{i+n \mid e_i=(v_j,v_k) \text{ and only one of } v_j \text{ and } v_k \text{ is in } S\}$$

Now

$$\sum_{i\in N} A_i = \sum_{v_i\in S} A_i + \sum_{\substack{e_{i-n} \text{ is bound}\\ \text{on one side}\\ \text{by } S}} A_i = \sum_{v_i\in S}\left(10^m + \sum_{(v_i,v_j)=e_k\in E} 10^{k-1}\right) + \sum_{\substack{e_i \text{ is bound}\\ \text{on one side}\\ \text{by } S}} 10^{i-1}$$

Now,

$$\sum_{v_i\in S}\sum_{(v_i,v_j)=e_k\in E} 10^{k-1} = \sum_{\substack{e_k \text{ is bound}\\ \text{on both sides}\\ \text{by } S}} 2\cdot 10^{k-1} + \sum_{\substack{e_k \text{ is bound}\\ \text{on one side}\\ \text{by } S}} 10^{k-1}$$

This is as each edge which is bound on two sides by S is counted twice in the leftmost sum. Since every edge is bound on at least one side by S, we have

$$\sum_{i\in N} A_i = \sum_{v_i\in S} 10^m + \sum_{k=1}^{m} 2\cdot 10^{k-1} = k10^m + \sum_{k=1}^{m} 2\cdot 10^{k-1} = b$$

And so $(A,b)\in$ SubsetSum as required.

Now suppose that $(A,b)\in$ SubsetSum, so there exists a set of indexes of A, which we will denote by N, where $\sum_{i\in N} A_i = b$. So let us define

$$S = \{v_i \mid i\in N,\ i\leq n\}$$

Now, $|S| = k$ as the only way to obtain a value larger than $k \cdot 10^m$ is by taking k vertices in N, as for every digit less than m, there are only three values in A which have a one in that digit. And if we were to take $k+1$ vertices, we'd get a value larger than $(k+1)10^m$. Let

$$N_1 = \{i \in N \mid i \leq n\}, \quad N_2 = \{i \in N \mid i > n\}$$

so N_1 represents the vertices in S, and N_2 represents the edges. And so

$$\sum_{i \in N} A_i = \sum_{i \in N_1} A_i + \sum_{i \in N_2} A_i = \sum_{i \in N_1} \left(10^m + \sum_{(v_i, v_j) = e_k \in E} 10^{k-1} \right) + \sum_{k \in N_2} 10^{k-1}$$
$$= |N_1| 10^m + \sum_{i \in N_1} \sum_{(v_i, v_j) = e_k \in E} 10^{k-1} + \sum_{k \in N_2} 10^{k-1}$$

This is equal to

$$|N_1| 10^m + \sum_{i \in N_1} \sum_{(v_i, v_j) = e_k \in E} 10^{k-1} + \sum_{k \in N_2} 10^{k-1} = k 10^m + \sum_{i=1}^{m} 2 \cdot 10^{m-1}$$

so as explained before (and this gives another viewpoint) $|N_1| = k$ and so

$$\sum_{i \in N_1} \sum_{(v_i, v_j) = e_k \in E} 10^{k-1} + \sum_{k \in N_2} 10^{k-1} = \sum_{i=1}^{m} 2 \cdot 10^{i-1}$$

And since N_1 is the indexes of the vertices which are in S, and N_2 is the index

$$\sum_{i \in N_1} \sum_{(v_i, v_j) = e_k \in E} 10^{k-1} + \sum_{k \in N_2} 10^{k-1} = \sum_{\substack{e_k \text{ is bound} \\ \text{on two sides} \\ \text{by } S}} 2 \cdot 10^{k-1} + \sum_{\substack{e_k \text{ is bound} \\ \text{on one side} \\ \text{by } S}} 10^{k-1} + \sum_{k \in N_2} 10^{k-1}$$

By subtracting this from b we get zero, but it is also equal to

$$\sum_{\substack{e_k \text{ is bound} \\ \text{on one side} \\ \text{by } S}} 10^{k-1} - \sum_{k \in N_2} 10^{k-1} + \sum_{\substack{e_k \text{ is not} \\ \text{bound by } S}} 2 \cdot 10^{k-1} = 0$$

But since no number of the form $2 \cdot 10^{k-1}$ can be formed within the difference of the two sums on the left, this means that every e_k must be bound by S. So S is a vertex cover, as required. ■