CSE 347 Lecture 16

Intractability via Reductions II

1. Traveling Salesperson & Hamiltonian Cycle are NP-complete
2. 3-coloring is NP-complete

Today we'll see a couple more "famous" NP-complete problems that do not resemble any of the problems we've seen so far. This is useful since it will vastly broaden the selection of problems that are "mere variations of" problems we know to be NP-complete.

The first problem is the "Traveling Salesperson" problem. Intuitively, our salesperson would like to visit every one of n cities and return to his or her home city, while taking the shortest possible trip. Formally, we have a directed graph in which every directed edge (u,v) has a nonnegative distance $d(u,v)$. (The asymmetry could capture one-way routes, uphill vs. downhill, etc.; the problem also has much broader applications that use this flexibility.) Now, we wish to know: is there a cycle in the graph that visits every vertex and has total distance at most D?

It will be convenient to actually focus on the fol-

lowing simplified variant of the Traveling Salesperson problem, the Hamiltonian Cycle problem: in a directed graph G, is there a cycle that visits every vertex exactly once (before returning to start)?

Theorem Hamiltonian Cycle $\leq_p$ Traveling Salesperson

Proof: Consider the following algorithm. Given a directed graph G, we construct a new graph G' on the same set of vertices, in which there is a (u,v) edge with $d(u,v)=1$ if (u,v) is an edge in G, and $d(u,v)=2$ otherwise. We now ask if there is a Traveling Salesperson tour of total length n in G', and use this answer, the algorithm only takes $O(|V|^2)$ steps to fill out G'. The algorithm is correct because? A Hamiltonian Cycle in G is a cycle in G' that only uses the edges (u,v) in G, that have $d(u,v)=1$; so the total distance is n. Conversely, a tour in G' must traverse at least n edges since it visits at most one new vertex with each edge traversed. Since each edge in G' has distance at least 1, a tour of length n cannot take any of the distance 2 edges, and must arrive at a new vertex after each edge traversed. Thus, it is a cycle that only uses the edges of G, and only visits each vertex once: this is a Hamiltonian Cycle in G. ▨

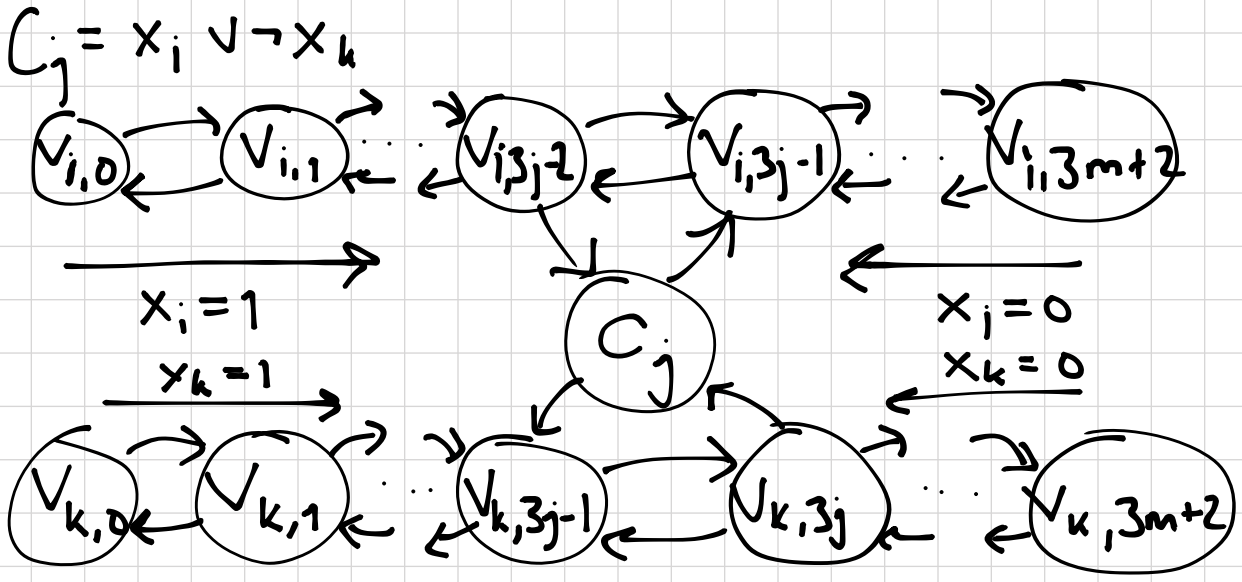
It's enough now to show that the simpler Hamilton-

ian Cycle problem is NP-complete, the Traveling Salesperson problem is in NP since, given a list of vertices, we can add up the distances and check that they total to at most D.

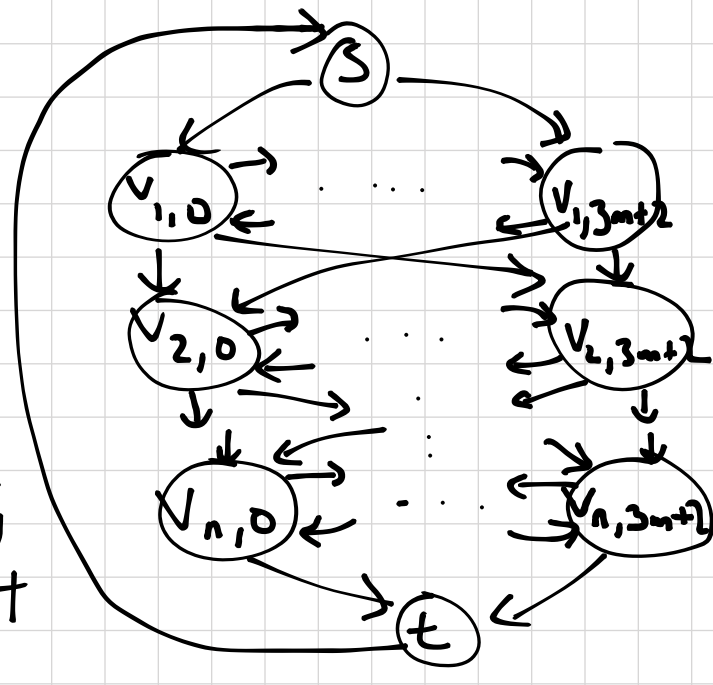
Theorem: Hamiltonian Cycle is NP complete

Proof: We first observe that Hamiltonian Cycle is also in NP since given a list of vertices we can check that the list has length $|V|$, each subsequent pair of vertices is joined by an edge, and that there are no duplicate vertices (e.g., by sorting it) in time $O(|V| \log |V|)$. Next, we'll show 3-SAT $\leq_p$ Hamiltonian Cycle. Since 3-SAT is NP-complete, we'll be finished.

Suppose we're given a 3-CNF $C_1 \wedge C_2 \wedge \cdots \wedge C_m$ on variables $x_1, \ldots, x_n$. Our basic gadget will be a chain of vertices of length $3m+3$ for each variable, that can be traversed in either direction — the two directions will correspond to setting the variable to 'true' or 'false'. Let's name the vertices for x_i $v_{i,0}, v_{i,1}, \ldots, v_{i,3m+1}, v_{i,3m+2}$. Now, we will track whether or not a clause C_j is satisfied by whether or not we (can) visit a vertex c_j: if x_i appears in C_j, we'll include a "detour" to c_j between $v_{i,3j-2}$ and $v_{i,3j-1}$ (so instead of taking the $(v_{i,3j-2}, v_{i,3j-1})$ edge as we traverse the chain from $v_{i,0}$ to $v_{i,3m+3}$, we can go via c_j instead). Similarly, if $\neg x_i$ appears in C_j we include a detour to c_j along with the $(v_{i,3j}, v_{i,3j-1})$ edge.



We add edges from both $v_{i,0}$ and $v_{i,3m+2}$ to both $v_{i+1,0}$ and $v_{i+1,3m+2}$, so that regardless of which way we traverse the ith chain, we can traverse the $i+1^{th}$ in either direction. Finally, we create vertices s and t such that s has edges to both $v_{1,0}$ and $v_{1,3m+2}$; both $v_{n,0}$ and $v_{n,3m+2}$ have edges to t; and t only has an edge to s. This completes our graph; observe that if we first create a list of the clauses that each variable appears in (in time $O(m)$) we can create each chain easily in time $O(m)$, so this takes $O(nm)$ time overall.



If the formula is satisfiable, then it's clear that by traversing the ith chain from $v_{i,0}$ to $v_{i,3m+2}$ if $x_i = 1$ in the satisfying assignment (or from $v_{i,3m+2}$ to $v_{i,0}$ if $x_i = 0$), we can take the detour to c_j in the first satisfied literal's chain (say) — this way we visit each c_j once (since C_j is satisfied) and visit all of the vertices of the chain. The cycle visits t after the nth chain before returning to s, so this is a Hamiltonian Cycle if the 3CNF is satisfiable.

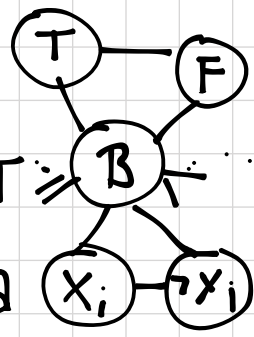
Now suppose that there is a Hamiltonian Cycle in our graph. The key claim is that in such a cycle, whenever our cycle (starting from s) traverses $v_{i,0}, \ldots, v_{i,3j-2}, c_j$, it must return to $v_{i,3j-1}$ (and similarly, if it traverses $v_{i,m+2}, \ldots, v_{i,3j}, c_j$, it must also return to $v_{i,3j-1}$). Why? The Hamiltonian Cycle must visit $v_{i,3j-1}$, but only $v_{i,3j-2}$, c_j, and $v_{i,3j}$ have edges to $v_{i,3j-1}$ — since we have already visited $v_{i,3j-2}$ (resp. $v_{i,3j}$) and c_j, we must return via $v_{i,3j}$ (resp. $v_{i,3j-2}$). But now the only edges exiting $v_{i,3j-1}$ also go to $v_{i,3j-2}$, c_j, and $v_{i,3j}$, and we have already visited all three vertices on the path, so this can't be part of a Hamiltonian Cycle. Thus, if we set x_i equal to 1 if we traverse its chain in increasing order and equal to 0 otherwise, then since the cycle can only visit c_j via a detour in the chain for some variable that satisfies C_j in the corresponding assignment (and we visit $c_1, \ldots, c_m$), the formula is satisfied by this assignment. ▨

Our next problem comes from the coloring of maps (so that no adjacent territories take the same color): given an undirected graph G, is it possible to assign one of three colors to each vertex so that no vertices u & v joined by an edge take the same color? This is the 3-coloring problem.

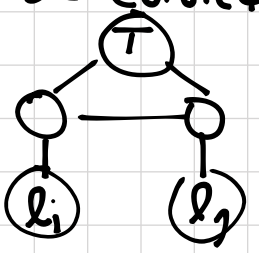
Theorem 3-coloring is NP-complete.

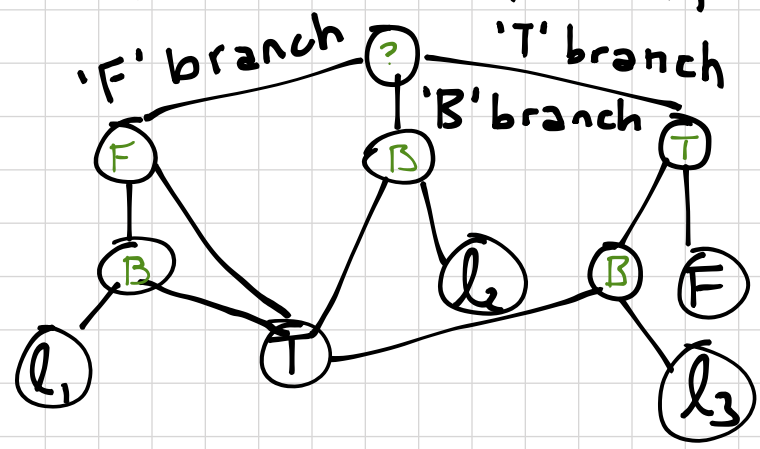
Proof First, 3-coloring is in NP since, given a list of $|V|$ colors, we can scan the edges of G and check that the endpoints received different colors in the list in linear time.

Now, we show 3-SAT $\leq_p$ 3-coloring. Consider a formula $C_1 \wedge \cdots \wedge C_m$. We'll identify the three colors with the two truth values and a third "base" value. We start with a "palette" that must take all three colors, and name them according to which vertex gets which color (we name the color of vertex T "T", the color of vertex F 'F,' etc.) Next, we create a vertex for each literal, joined to its negation and B. This ensures that one must be colored T and the other F.



Now we add gadgets for the clauses. If a single literal clause appears, we can join the literal to F so that it must be colored T; if a two-literal clause $\ell_i \vee \ell_j$ appears, we can add so that if both are F, we can't color the new nodes, and otherwise one adjacent to a satisfied literal can take F, and the other can take B. Finally, for a three-literal clause $\ell_i \vee \ell_j \vee \ell_k$, we use: Observe that if ℓ_i, ℓ_j, and ℓ_k all take color F, then the 'F' branch edge must be to color F, the 'B' branch must be to color 'B', and the 'T' branch must be to color 'T' – but then the top can't be colored. But, if one ℓ_i is colored 'T', the node that was forced to color 'B' above it can be switched to 'F', allowing the node at the top of the branch to switch so that we repeat a color, freeing up that branch's color for the top node. Thus we've shown how to color each structure using a satisfying assignment, and we've argued that if the structure





is colorable, then the corresponding clause is satisfied if we set each x_i to the truth value matching its color. That is, setting $x_i = 1$ when x_i is colored 'T' (and $x_i = 0$ when x_i is colored 'F') gives a satisfying assignment. ▨