CMSC 451: Lecture 19
NP-Completeness: Hamiltonian Cycle

Hamiltonian Cycle: Today we consider the Hamiltonian cycle problem in directed graphs. Given a digraph $G = (V, E)$, the question is whether there exists a simple cycle that visits all the vertices. The fact that the cycle is simple, implies that every vertex is visited exactly once (except for the first and last vertex.) There are three natural variants: Hamiltonian cycle in undirected graphs, and Hamiltonian paths in both directed and undirected graphs. A Hamiltonian path is a simple path that visits every vertex (exactly once). All four problems are NP-complete. We will present a proof of the Hamiltonian cycle problem for directed graphs and leave the others as exercises. Formally, let us define

$$\text{DHC} = \{G : G \text{ is a directed graph with a Hamiltonian cycle}\}.$$

An important related problem is the traveling salesman problem (TSP). Given a complete graph (or digraph) with nonnegative edge weights, compute the cycle of minimum weight that visits all the vertices. Since the graph is complete, such a cycle will always exist. We can formulate as a decision problem as follows: given a complete weighted graph G, and integer w, does there exist a Hamiltonian cycle of total weight at most w? TSP is also NP-complete, which we will leave as an exercise.

Another related problem is called the *Eulerian circuit problem*. In this problem, we want to compute a cycle that visits every *edge* exactly once. This is related to a famous math problem, called the Seven Bridges of Königsburg. This problem can be solved in polynomial time by depth-first search. (Again, we'll leave this as an exercise.)

Component Design: Up to now, most of the reductions that we have seen (e.g., for clique, vertex cover, and dominating set) are of a relatively simple variety. They are sometimes called *local replacement reductions*, because they operate by making some local change throughout the graph.

The reduction for Hamiltonian cycle is more complicated, involving a technique called a *component design*. This method involves designing special subgraphs, sometimes called *components* or *gadgets* whose job it is to enforce a particular constraint of the problem. Very complex reductions may involve the creation of many gadgets. This one involves the construction of only one.

DHC Gadget: The main part of the proof is a reduction showing that 3SAT $\leq_P$ DHC. Our reduction will take a 3SAT formula F as input. Recall that such a formula is a conjunction (logical "and") of clauses, each of which is a disjunction (logical "or") of three literals, each of which is either a variable x_j or its complement $\overline{x}_i$.

Our reduction will map each clause to a special subgraph, called a *DHC gadget* (see Fig. 1). This subgraph has three incoming edges and three outgoing edges, one for each of the clause's literals. Let's call the incoming edges i_1, i_2 and i_3, and let's call the outgoing edges o_1, o_2 and o_3.

Our overall plan will be to string these gadgets together in chains, one per literal. We will connect the output from one gadget to the input of the next such that the literals agree (that

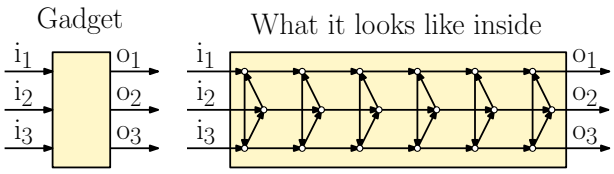

Fig. 1: The DHC gadget.

is, connecting the x_j output of one gadget to the x_j input of the next and connect the $\overline{x}_j$ output of one gadget to the $\overline{x}_j$ input of the next). The strategy is that for each true literal in a clause, the Hamiltonian cycle will travel in along the corresponding input and out along the corresponding output. The DHC gadget is carefully designed to satisfy the following property.

Lemma: (DHC Gadget Properties)

(i) Given any nonempty subset of k input edges, there exists a set of k vertex-disjoint paths that together visit all the vertices of the gadget exactly once and connect each input edge, i_j, with its corresponding output edge, o_j.

(ii) Given any nonempty set of k vertex-disjoint simple paths from entry to exit that hit all the vertices, each path that enters along i_j it must exist on the corresponding output o_j.

Proof: For (i), this is literally a "proof by picture" (see Fig. 2). We show the case of one, two, and three entry edges. (There are a total of $2^3 - 1 = 7$ nonempty subsets of inputs, but due to the symmetry of the gadget, it is easy to see how the omitted cases work.)

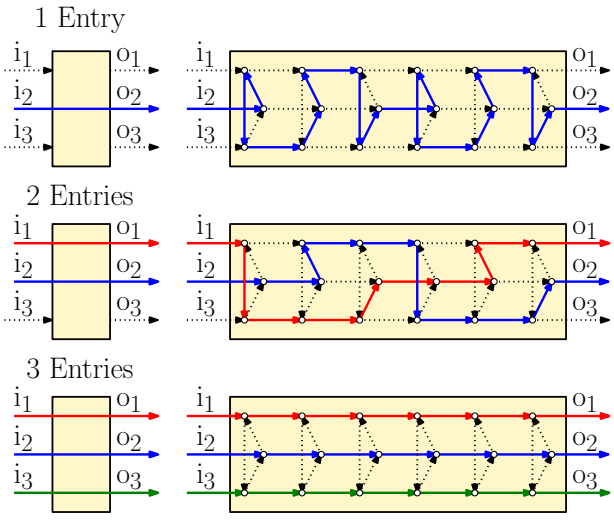

Fig. 2: DHC gadget and examples of path traversals.

The proof of (ii) is a bit harder to see.

- In the case of one entry, in order to hit all the vertices, it is necessary to exit each three-vertex cluster by cycling one turn around. Since there are six clusters of three

vertices, on exit the path returns to the same output edge.

- In the case of three entries, it is also easy to see that the only way to hit all the vertices exactly once is for the paths to travel straight through, implying that each input is routed to the corresponding output.
- In the case of two inputs, in order to hit every vertex once, one of the two paths needs to hit two vertices in each of the cluster and the other hits one. If you play around with the gadget, you will see that the choice of which path does which is fixed. This causes the two paths to cycle around. After three cycles they return to the same positions, but their order is switched. After six cycles, they return both to the same position and in the same order.

To see whether you understand the gadget, you might ask the question of why we used exactly 6 three-vertex cycles? Why a multiple of three? Would 3 work? Would 9? Would 12?

The Reduction: Our objective is to show that 3SAT $\leq_P$ DHC. That is, we need to present a polynomial-time function f, which given an boolean formula F for 3SAT, outputs a directed graph G such that F is satisfiable if and only if G has a Hamiltonian cycle. Recall that a satisfying a 3-CNF formula corresponds to assigning truth values to each variable so that every clause has at least one true literal.

Intuitively, we can think of the DHC gadgets as forming a string of holiday light bulbs along a wire. There will be one DHC gadget for each clause in the formula. Each gadget has three entry edges and three exits edges. Think of this like three wires entering and exiting each of these bulbs. Each variable x_j will have two paths which we can choose from. One corresponds to setting $x_j = \mathsf{T}$ and taking the other path corresponds to setting $x_j = \mathsf{F}$ (or equivalently to setting $\overline{x}_j = \mathsf{T}$). We think of the wire that carries the actual truth value for each variable to be the "live wire" that carries electrical current. Ultimately, we want every one of these light bulbs to light up. This will happen if at least one of the wires that is carrying current travels through the light bulb. This is equivalent to saying that at least one of the literals in this clause is true, which is exactly what we need for the formula to be satisfiable. Of course, we don't know whether each variable will be true or false. The idea is that the string of light bulbs will be illuminated if and only if we can assign truth values to all the variables so that the resulting current paths hit every bulb.

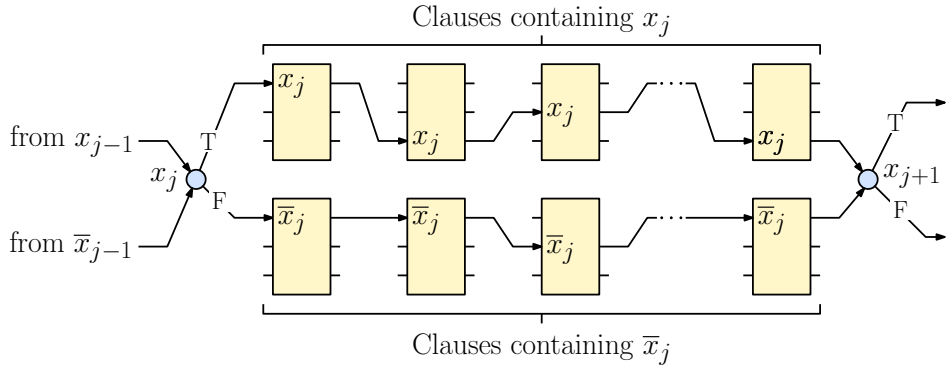

Fig. 3: General structure of reduction from 3SAT to DHC.

For each variable x_j, we will create a special *variable vertex*, named x_j. Each vertex x_j will have two outgoing edges, one labeled T and the other labeled F. Intuitively, any Hamiltonian cycle must visit each variable vertex x_j. If it chooses to take the T edge, we interpret this as setting $x_j = \mathsf{T}$ and taking the other edge corresponds to setting $x_j = \mathsf{F}$. The path on the T edge will be strung through all the clause gadgets that contain the literal x_j, and the path along the F edge will be strung through all the clause gadgets that contain the literal $\overline{x}_j$ (see Fig. 3).

To connect all of these paths together, we create a start vertex s and add an edge (s, x_1). The two paths leading out from x_1 will converge at x_2, where again two paths will diverge, joining at x_3. We continue in this manner until reaching the final vertex x_n. Its two paths will converge at a final vertex t. To complete the cycle, we generate an edge (t, s). An example of the reduction is shown in Fig. 4.

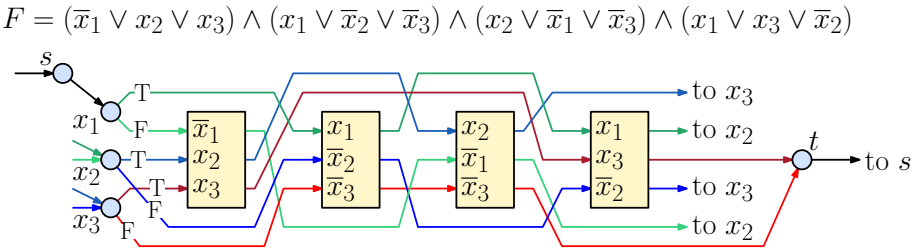

Fig. 4: Example of the 3SAT to DHC reduction. Wires are color-coded (x_1 is green, x_2 is blue, and x_3 is red).

The following lemma establishes the correctness of this reduction.

Lemma: The boolean formula F is satisfiable if and only if the digraph G produced by the above reduction has a Hamiltonian cycle.

Proof:

($\Rightarrow$) : Suppose that F has a satisfying assignment. We claim that G has a Hamiltonian cycle. This path starts at s and goes immediately to the variable vertex x_1. From now on, whenever it reaches a variable vertex, it follows the outgoing T edge if this variable is assigned true and the F edge otherwise. Depending on which path is taken, it travels through all the gadgets involving this literal. When the final clause containing this literal is visited, it goes to the next variable vertex, x_{j+1}, and the process continues from there.

Because this is a satisfying assignment, every clause will have at least one true literal, and hence each clause gadget will be visited by at least one path. By DHC Gadget Property (i), whenever a path enters a gadget on some entry edge, it exits on the corresponding exit edge (see Fig. 5). (That is, wires never get crossed.)

On finishing with the path for the last variable x_n, we go to the end vertex t, and from there we go back to s, thus completing the Hamiltonian cycle. Since every clause is visited, every vertex is visited exactly once, and therefore, G has a Hamiltonian cycle.

($\Leftarrow$) : Suppose that G has a Hamiltonian cycle. Since the cycle must visit all the vertices, we may assume it starts with s. Since this is a Hamiltonian cycle, every vertex is visited, and hence, every clause gadget is visited by some number of paths (either 1, 2, or 3).

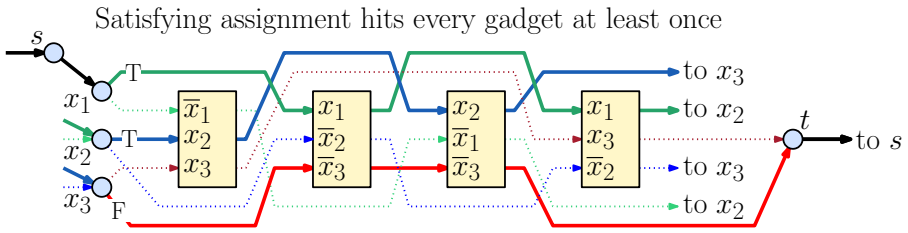

Fig. 5: A satisfying assignment ($x_1 = \mathsf{T}$, $x_2 = \mathsf{T}$, $x_3 = \mathsf{F}$), yields a Hamiltonian cycle.

From s the path visits x_1. Whenever it visits a variable vertex, the path takes either the outgoing edge labeled T or the edge labeled F. If it takes the former, we assign this variable true, and otherwise we assign it the value false.

We assert that this is a satisfying assignment for the formula. By the DHC Gadget Properties (ii), whenever a Hamiltonian cycle enters a gadget, it must exit along the exit edge corresponding to its entry edge. Therefore, each of the paths from the variable vertices behaves like a wire going through all gadgets in which this literal appears, and then going on to the next variable. When it visits the last variable, its only option is to go to t and then back to s.

Since this is a Hamiltonian cycle, every gadget must be visited by at least one path, and hence every clause must have at least one literal whose assigned value is true. (To illustrate this, consider the non-satisfying assignment in Fig. 6. At least one clause has all literals evaluate to false, and hence the cycle misses the associated gadget. However, this contradicts the hypothesis that G has a Hamiltonian cycle.) Since every clause has at least one true literal, this is a satisfying assignment.

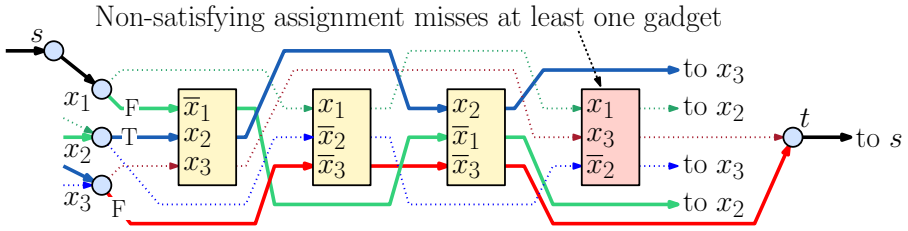

Fig. 6: A non-satisfying assignment ($x_1 = \mathsf{F}$, $x_2 = \mathsf{T}$, $x_3 = \mathsf{F}$), does not yield a Hamiltonian cycle.

Final Conclusion: We can now present the proof that DHC is NP-complete. Recall that we need to show that (i) DHC $\in$ NP and (ii) some known NP-complete problem is reducible to it.

Theorem: Directed Hamiltonian Cycle (DHC) is NP-complete.

Proof:

- (DHC $\in$ NP) The certificate consists of the sequence of vertices in the cycle. In $O(n)$ time, we can check each consecutive pair to see that it is an edge in G. If so, the verification accepts, and if not, the verification rejects.
- (3SAT $\leq_P$ DHC) This follows from correctness of the reduction presented earlier.