CS 341 Algorithms. Spring 2025. University of Waterloo

Lecture 19: Hard graph problems

We show that two classical graph problems, Hamiltonian cycle and graph coloring, are NP-complete. Both reductions are from 3-SAT and require clever gadget design.

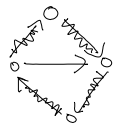
Hamiltonian Cycle [KT 8.5]

We will introduce an intermediate problem for the reduction.

Directed Hamiltonian Cycle (DHC)

Input: A directed graph $G=(V,E)$

Output: Does G have a directed cycle that touches every vertex exactly once?



We will show that 3-SAT $\leq_p$ DHC $\leq_p$ HC.

3-SAT and DHC are very different, so it requires some creativity to connect the two problems.

Theorem Directed Hamiltonian Cycle is NP-complete.

Proof It is easy to see that DHC is in NP. To prove it is NP-complete, we will prove 3-SAT $\leq_p$ DHC.

Given a 3-SAT instances with n variables $x_1, x_2, \ldots, x_n$ and m clauses $C_1, C_2, \ldots, C_m$, we would like to construct a directed graph G so that the formula is satisfiable iff G has a Hamiltonian cycle.

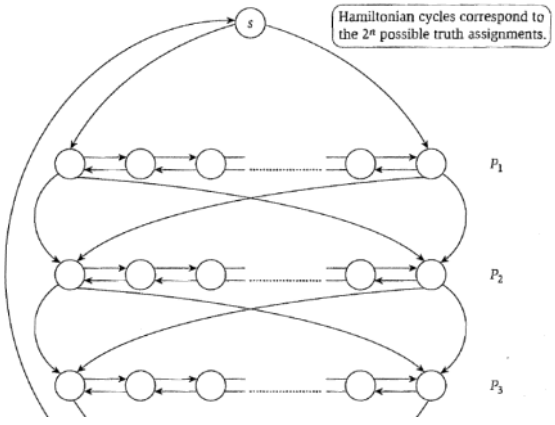
We need to create some graph structures for the variables and the truth assignment.

The idea is to associate a long "two-way" path to a variable and going the path in one way corresponds to setting the variable True, while going the other way corresponds to setting it to False.

The intention is that we go from left-to-right iff x_i is set to be True (i.e. right-to-left iff False).

In the following graph, there is a one-to-one correspondence between the 2^n truth assignment of the n variables and the directed Hamiltonian cycles in the graph.

[KT]

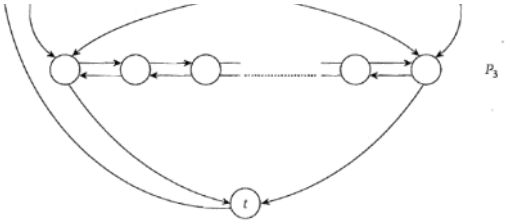


There are totally n two-way paths, one for each variable.

Each path is of length $3m$.

The two endpoints of the path P_i connect to the two endpoints of P_{i+1}.

There is a source s that connects to the two ends



There is a source s that connects to the two ends of P_1 and the two ends of P_n connects to the sink t, and t connects to s.

It should be clear that there are only 2^n possible Hamiltonian cycles in this directed graph, since we must use each path P_i either from left-to-right or from right-to-left. as the intermediate of the paths are not connected to anything else.

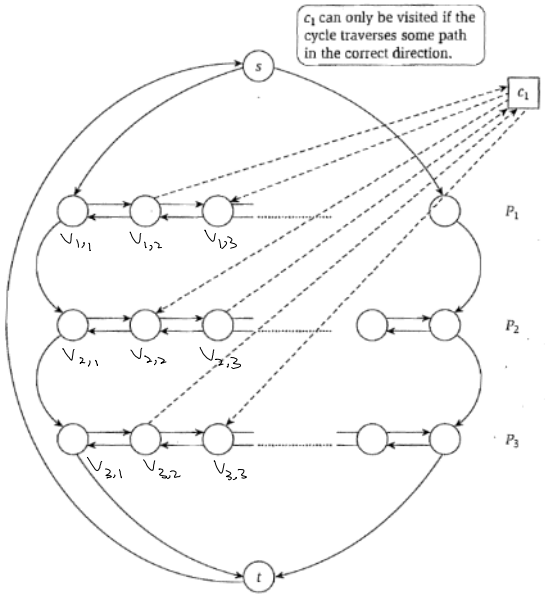
That's a good start, with a one-to-one correspondence between truth assignments and Hamiltonian cycles.

Next, we would like to add some clause structures to "kill" all the Hamiltonian cycles that do not correspond to satisfying assignments.

Recall that the two-way paths are of length 3m. The first three edges belong to the first clause, and in general the three edges between v_{3i+1}, v_{3i+2}, v_{3i+3}, $v_{3(i+1)+1}$ belong to the i-th clause.

Suppose there is a clause, say $x_1 \vee \bar{x}_2 \vee x_3$, then we want the Hamiltonian cycles to either (1) go from left-to-right in P_1, or (2) go from right-to-left in P_2, or (3) go from left-to-right in P_3 to satisfy the clause.

[kT]



To this end, we create one vertex c_j for each clause C_j.

Call the vertices on P_i be $v_{i,1}, v_{i,2}, \ldots, v_{i,3m+1}$.

If literal x_i appears in C_j, then we add the directed edges $v_{i,3j-1}\, c_j$ and $c_j\, v_{i,3j}$

Otherwise, if literal $\bar{x}_i$ appears in C_j, then we add directed edges $c_j\, v_{i,3j-1}$ and $v_{i,3j}\, c_j$.

We do this for every clause C_j.

Note that the edges for C_j and C_k don't share vertices, and that's why we created long paths for.

That's the whole construction. Clearly, it can be done in polynomial time.

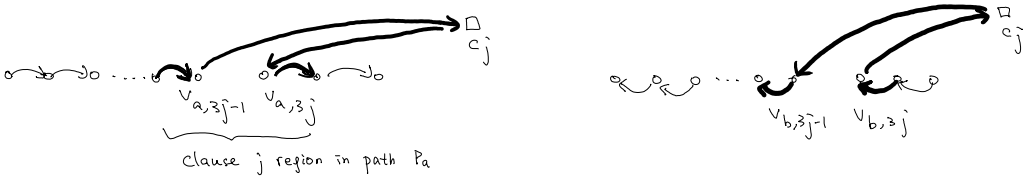
It remains to prove that the formula is satisfiable iff there is a Hamiltonian cycle in this graph.

⇒) Suppose there is a satisfying assignment.

If $x_i = T$, we visit P_i from left-to-right; otherwise if $x_i = F$ we visit P_i from right-to-left.

For a clause say $C_j = (x_a \vee \bar{x}_b \vee x_c)$, at least one literal is true in the satisfying assignment, say x_a.

Then, when we visit P_a from left-to-right, we "detour" to visit vertex c_j during the clause j region in the path P_a.



Similarly, if $x_b = F$, then we can also detour to visit c_j when we visit P_b from right-to-left.

Since every clause is satisfied in a satisfying assignment, we can visit all clause vertices following these directions and detours, and hence form a Hamiltonian cycle in the graph.

$\Leftarrow$) Suppose there is a Hamiltonian cycle. We would like to argue that it must look like those Hamiltonian cycles above that correspond to a satisfying assignment, but why must it be the case?

The crucial observation is that if we use the directed edge $v_{a,3j-1}\, c_j$, then we must use the edge $c_j\, v_{a,3j}$ immediately after it, as otherwise the vertex $v_{a,3j}$ will become a "dead-end" and there is no way to complete the cycle.

Since each clause vertex is visited in a Hamiltonian cycle, at least one variable path is going in the correct direction, and all the paths must go from left-to-right or right-to-left (as it must come back immediately after the "detour" to clause vertices), and so this corresponds to a satisfying assignment as we intended. □

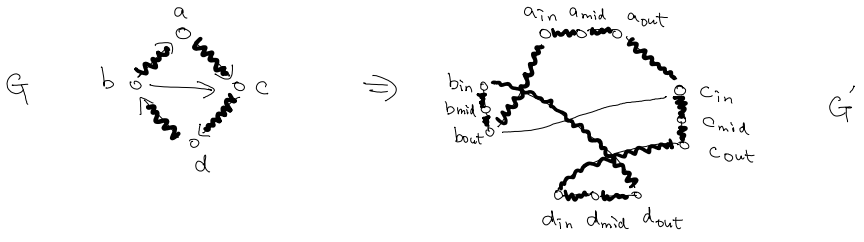
To prove DHC $\leq_p$ HC, the idea is to use a length 3 path to simulate a directed edge.

Proposition DHC $\leq_p$ HC.

Proof Given a directed graph $G = (V, E)$ for DHC, we construct an undirected graph G' in which we create three vertices v_{in}, v_{mid}, v_{out} for each vertex $v \in V(G)$.

Then, we create the edges in G' as follows: for every $v \in V(G)$, we add the edges (v_{in}, v_{mid}) and (v_{mid}, v_{out}).

Also, for each directed edge $uv \in E(G)$, we add an undirected edge (u_{out}, v_{in}) in G'



That's the whole construction, which can clearly be done in polynomial time.

Now we prove that G has a directed Hamiltonian cycle iff G' has an undirected Hamiltonian cycle.

$\Rightarrow$) One direction is immediate: if G has a Hamiltonian cycle, by following the cycle and replacing each directed edge uv by $u_{out}v_{in}$ and using the paths v_{in}-v_{mid}-v_{out} for all $v \in V(G)$, it is a Hamiltonian cycle in G'.

$\Leftarrow$) The other direction is slightly more interesting: in an undirected Hamiltonian cycle, start with a vertex v_{in}, then v_{mid} must be a neighbor of v_{in} in the Hamiltonian cycle, as otherwise v_{mid} will be a "dead-end" since it is of degree two, and then v_{out} must be a neighbor of v_{mid} in the Hamiltonian cycle.. Then, from v_{out}, by construction, the cycle must go to w_{in} for some w, and then w_{mid} and w_{out} as argued above. So, following the undirected Hamiltonian cycle of G', it must be of the form as described in the previous direction, and hence it corresponds to a directed Hamiltonian cycle in G. □

Corollary TSP is NP-complete.

Graph Coloring [KT 8.7]

Graph coloring is one of the most classical problem in graph theory, e.g. 4-color theorem.

Input: An undirected graph $G=(V,E)$, and a positive integer k.

Output: Is it possible to use k colors to color all the vertices so that every vertex receives one color and any two adjacent vertices receive different colors?

When $k=1$, it is possible iff the graph has no edges.

When $k=2$, it is possible iff the graph is bipartite (why?).

When $k=3$, the problem becomes NP-complete.

The graph coloring problem is very useful in modeling resource allocation problems, e.g. interval coloring problem.

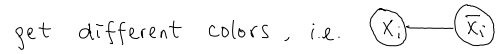
Theorem 3-coloring is NP-Complete.

Proof It is clear that 3-coloring is in NP (easy exercise). We prove 3-SAT $\leq_p$ 3-coloring.

Given a 3-SAT instances with n variables $x_1, x_2, \ldots, x_n$ and m clauses $C_1, C_2, \ldots, C_m$, we would like to construct a graph G so that the formula is satisfiable iff G is 3-colorable.

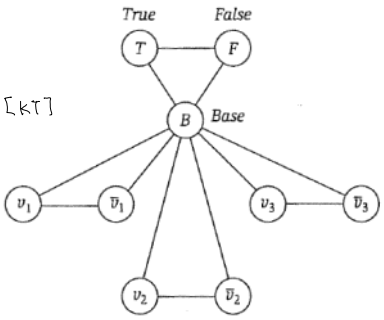
Quite naturally, we would like to associate one color to True and one color to False.

And we would like to ensure that the literals are colored consistently.

This is not difficult to do: we just create two vertices x_i and $\bar{x}_i$ for each variable and add an edge between them so that they will get different colors, i.e. 

Since there are three colors, to enforce that the two literals x_i and $\bar{x}_i$ get T/F colors,

we connect every literal vertex to a common vertex, called the base vertex.



[KT]

Call the three colors T, F, B.

We create two vertices v_i and $\bar{v}_i$ for each variable x_i.

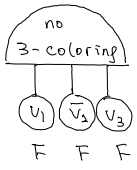
These edges enforce that either (v_i=T and $\bar{v}_i$=F) or (v_i=F and $\bar{v}_i$=T).

So, there is a one-to-one correspondence between truth assignments and the 3-colorings.

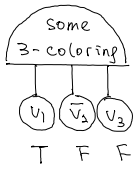
As in the NP-completeness proof of DHC, we would like to add some clause structures to "kill" the 3-colorings that don't correspond to satisfying assignments.

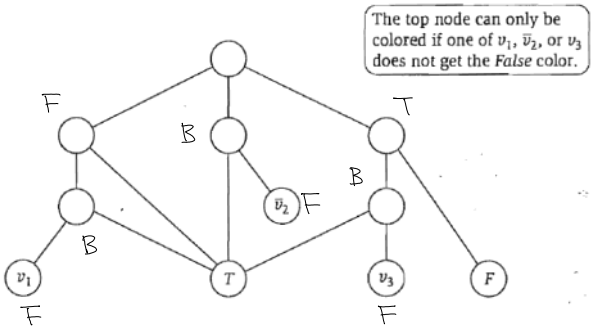
Say we have a clause $(x_1 \vee \bar{x}_2 \vee x_3)$, it would be great if there is a "gadget" to connect to the three variables $v_1, \bar{v}_2, v_3$ so that there is a 3-coloring for the gadget if and only if at least one of $v_1, \bar{v}_2, v_3$ gets the color T, e.g.

With some trial-and-error, it is possible to construct a gadget with exactly this "functionality".



and





⇐ We need to verify this claim.

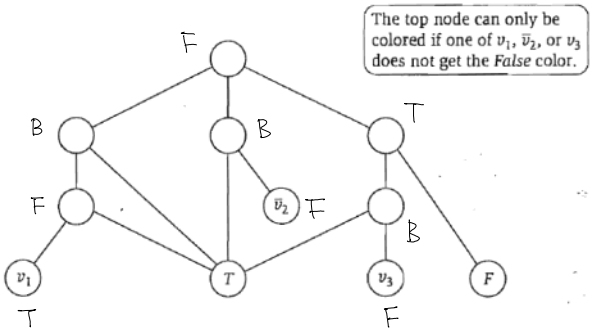
We create one such gadget for each clause.

All unlabeled vertices are new vertices created only for this clause.

First, suppose all $v_1, \bar{v}_2, v_3$ are colored F, then the top node in the gadget cannot be colored.

See the picture above; all the colors are forced, and there is no way to color the top node.

On the other hand, if at least one of $v_1, \bar{v}_2, v_3$ is colored T, then it is possible to color all nodes in the gadget.



If v_1 is colored T, then we can color all the nodes in the gadget using 3 colors.

We leave checking all other cases as coloring exercises.

It is clear that the reduction can be done in polynomial time.

With the claim about the gadget checked, it is not difficult to finish the proof by showing that

there is a satisfying assignment in the formula iff there is a 3-coloring in the graph.

⇒) If there is a satisfying assignment, for each variable x_i, if x_i = True, then we color $v_i = T$ and $\bar{v}_i = F$; otherwise if x_i = False, then we color $v_i = F$ and $\bar{v}_i = T$.

Clearly, it is a valid coloring for the initial base graph (without gadgets yet).

Since it is a satisfying assignment, each clause gadget has at least one of the three "inputs" set to be T, and thus it can be extended to a 3-coloring in the gadget by the gadget claim.

⇐) If there is a 3-coloring of the graph, then by the claim for each gadget, there is at least one "input" of the gadget with color T.

By the initial base graph, the coloring on $v_1, \bar{v}_1, v_2, \bar{v}_2, \ldots, v_n, \bar{v}_n$ must be consistent with a truth assignment.

So, following the coloring, we can define a truth assignment that satisfies all the clauses. □
