Hamiltonian Cycles

The next problem will build on the prior proof by using it in another reduction. Once again, we need to start with some definitions.

Definition: Given an undirected graph $G = (V, E)$, a Hamiltonian cycle is a simple cycle that contains each vertex in V.

Definition: The Hamiltonian cycle problem asks "Does graph G contain a Hamiltonian cycle?"

Thus, for a particular graph, the question relates to finding a simple cycle in the graph containing all of the vertices of the graph. For some graphs, this is a simple problem to solve (e.g., graphs consisting only of one simple cycle, or complete graphs). Unfortunately, this problem is difficult in general.

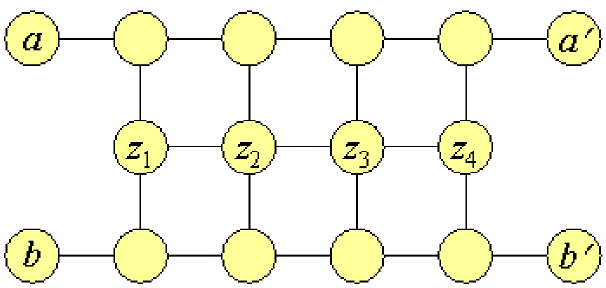
Theorem (HAM-CYCLE): The Hamiltonian cycle problem is NP-complete.

Proof:

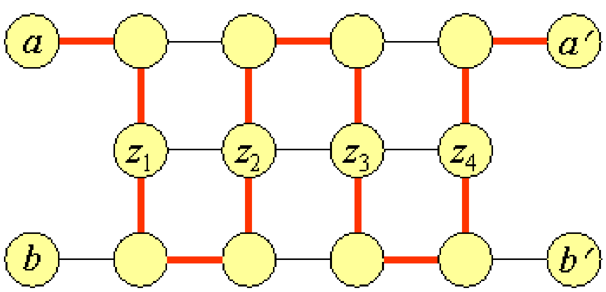
Given a graph G, we define a "certificate" (i.e., a potential solution) for the problem to be a sequence of $|V|$ vertices that make up a Hamiltonian cycle. The verification algorithm checks that each vertex in V occurs exactly once in this sequence and that, with the first vertex appended to the end of the sequence, a cycle forms in G. To do the latter, the verification algorithm makes sure that all adjacent vertices in the sequence have an edge connecting them in G. The verification can be completed in polynomial time, so the Hamiltonian cycle problem is in NP.

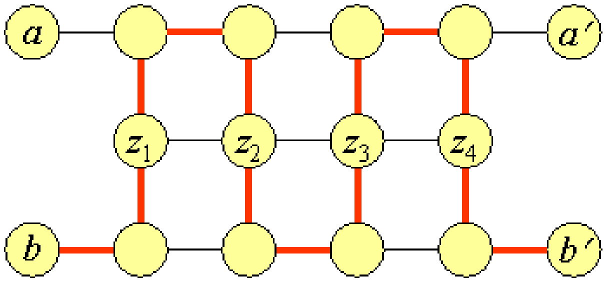
We will now show that 3-CNF-SAT $\leq_P$ HAM-CYCLE. To do this, we start with some 3-CNF Boolean formula ϕ defined over variables $x_1, \ldots, x_n$ in clauses $C_1, \ldots, C_k$, each containing exactly three distinct literals. Then we will construct a graph G in polynomial time such that G has a Hamiltonian cycle if and only if is satisfiable. This construction will make use of a technique called widgets (or sometimes gadgets). These widgets will correspond to pieces of graphs that will need to be pieced together.

The first widget (widget A) is as follows:

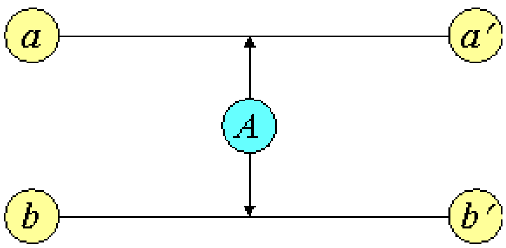


This widget will be used in such a way that it connects to the rest of the graph only via the nodes labeled a, b, a', and b'. In addition, we are going to restrict the widget such that any Hamiltonian cycle with this structure must pass through all four nodes, z_1, z_2, z_3, and z_4. It can do this in one of two ways, illustrated as follows:

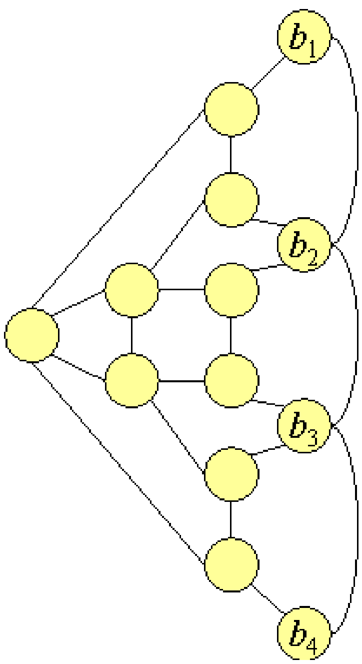




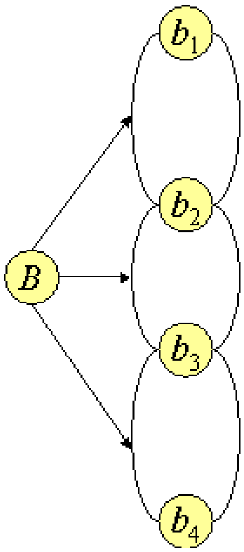
Given these paths, we may treat the widget as if it was simply a pair of edges (a, a') and (b, b') with the restriction that any Hamiltonian cycle must include exactly one of these edges.



Widget B is more complicated and takes the following form:



As with the A widget, we connect this widget into the graph via the b nodes. Simplifying, we can represent this widget as follows:



Admittedly, these widgets look a bit odd, but once we construct our graph, we will be able to see how the graph relates to the satisfiability problem.

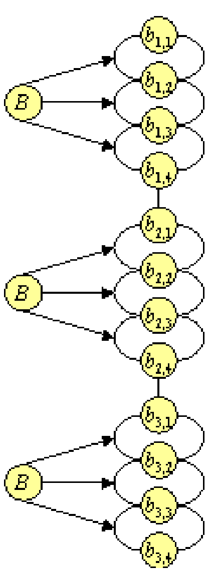
Suppose this B widget is also a subgraph of a graph with a Hamiltonian cycle. Suppose further that the only connection to the rest of the graph is via nodes b_1, b_2, b_3, and b_4. The Hamiltonian cycle cannot traverse all of the edges of (b_1, b_2), (b_2, b_3), and (b_3, b_4) because of the way the nodes are connected. However, a Hamiltonian cycle can traverse any proper subset of these edges.

Now that we have our widgets, we construct the final graph as follows. For each of the k clauses $(C_1, \ldots, C_k)$, we will create a copy of widget B. The widgets will be joined in series. To do this, let b_{ij} be the copy of vertex b_j for clause C_i (i.e., in the ith widget). We then connect $b_{i,4}$ to $b_{i+1,1}$ for $i = 1, \ldots, k-1$.

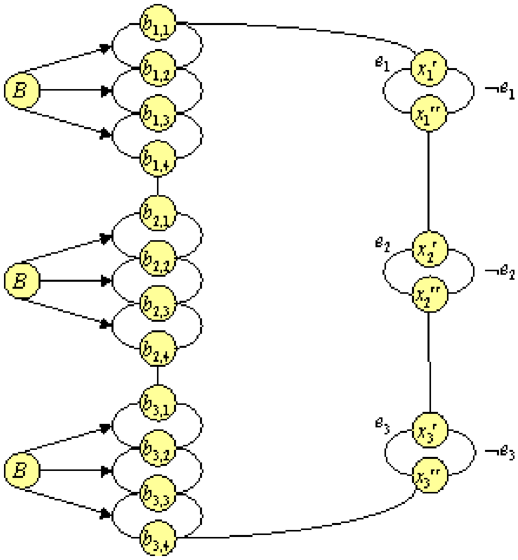
As an example, suppose we want to generate a "satisfiability" graph for the formula

$$\phi = (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3).$$

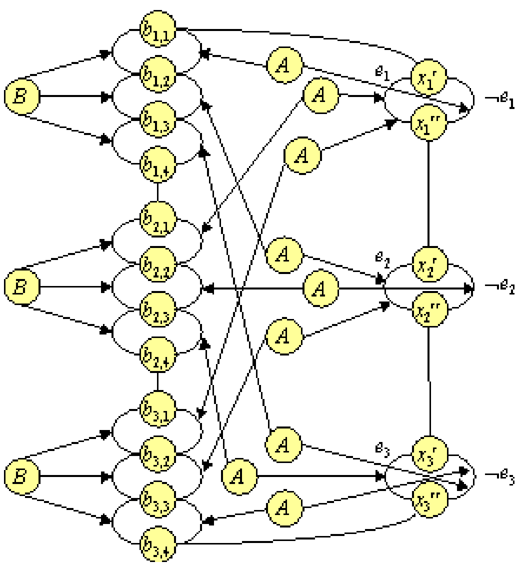
This means we need three B widgets, organized as follows:



Then for each variable x_m in the formula, we will add two vertices to the graph, denotedx'_m and x''_m. These vertices will be connected with two copies of an edge (x'_m, x''_m), which we denote e_m and $\neg e_m$. If the Hamiltonian cycle takes e_m, it is like assigning a value of 1 in the corresponding 3-CNF formula. If it takes $\neg e_m$, it is like assigning a value of 0. All of these loops are connected in series, and $(b_{1,1}, x'_m)$ is connected to $(b_{k,4}, x''_m)$.



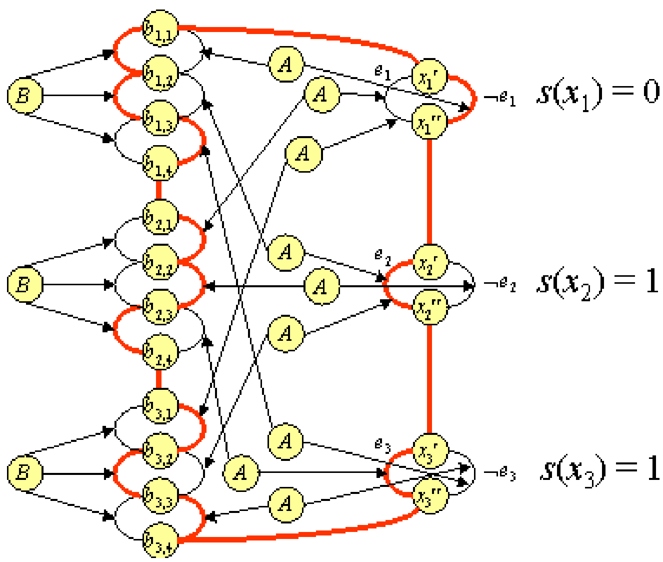
Finally, we need to relate the variables on the right to the clauses on the left. These relationships are established via A widgets. Specifically, if the jth literal of clause C_i is x_m, we will use an A widget to connect $(b_{ij}, b_{i,j+1})$ to e_m. On the other hand, if the jth literal of clause C_i is $\neg x_m$, we will use an A widget to connect $(b_{ij}, b_{i,j+1})$ to $\neg e_m$.



Note that a given literal ℓ_m may appear in several clauses, and thus an edge e_m (or $\neg e_{m,}$) may be influenced by several A widgets. These A widgets would be connected in serial in this case.

To interpret this graph, we note that formula ϕ is satisfiable if and only if graph G has a Hamiltonian cycle. Furthermore, this graph must take a particular form. The interpretation then follows a walk of certain edges of the graph. First, edge $(b_{1,1}, x'_1)$ is traversed. Then we follow all of the x'_m and x''_m vertices from the top of the graph to the bottom, selecting one of either e_m or $\neg e_m$, but not both. At the bottom of the graph, we then traverse back via the edge $(b_{k,4}, x'_n)$. Finally, the B widgets are traversed from the bottom back to the top.

Given a particular Hamiltonian cycle, we define a truth value assignment as follows. If edge e_m belongs to the cycle, assign value 1 to variable x_m. Otherwise, the Hamiltonian cycle must cross $\neg e_m$, so we assign a value of 0 to x_m. The resulting truth value assignment will satisfy the original formula ϕ. If the formula is unsatisfiable, then no Hamiltonian cycle exists.



To analyze the construction process, note that each widget introduces a constant number of nodes, and there is one A widget per literal instance and one B widget per clause. Thus the graph can be constructed in polynomial time. QED