THE HAMILTONIAN PATH PROBLEM

Recall that the Hamiltonian path problem asks whether the input graph contains a path from \( s \) to \( t \) that goes through every node exactly once.

**Theorem 7.46**

\( \text{HAMPATH} \) is NP-complete.

**Proof Idea** We showed that \( \text{HAMPATH} \) is in NP in Section 7.3. To show that every NP-problem is polynomial time reducible to \( \text{HAMPATH} \), we show that \( 3\text{SAT} \) is polynomial time reducible to \( \text{HAMPATH} \). We give a way to convert 3cnf-formulas to graphs in which Hamiltonian paths correspond to satisfying assignments of the formula. The graphs contain gadgets that mimic variables and clauses. The variable gadget is a diamond structure that can be traversed in either of two ways, corresponding to the two truth settings. The clause gadget is a node. Ensuring that the path goes through each clause gadget corresponds to ensuring that each clause is satisfied in the satisfying assignment.

**Proof** We previously demonstrated that \( \text{HAMPATH} \) is in NP, so all that remains to be done is to show \( 3\text{SAT} \) \( \leq_P \) \( \text{HAMPATH} \). For each 3cnf-formula \( \phi \) we show how to construct a directed graph \( G \) with two nodes, \( s \) and \( t \), where a Hamiltonian path exists between \( s \) and \( t \) iff \( \phi \) is satisfiable.

We start the construction with a 3cnf-formula \( \phi \) containing \( k \) clauses:

\[
\phi = (a_1 \lor b_1 \lor c_1) \land (a_2 \lor b_2 \lor c_2) \land \cdots \land (a_k \lor b_k \lor c_k),
\]

where each \( a, b, \) and \( c \) is a literal \( x_i \) or \( \overline{x_i} \). Let \( x_1, \ldots, x_l \) be the \( l \) variables of \( \phi \).

Now we show how to convert \( \phi \) to a graph \( G \). The graph \( G \) that we construct has various parts to represent the variables and clauses that appear in \( \phi \).

We represent each variable \( x_i \) with a diamond-shaped structure that contains a horizontal row of nodes, as shown in the following figure. Later we specify the number of nodes that appear in the horizontal row.

![Diagram of a diamond-shaped structure representing a variable \( x_i \).](image)

**Figure 7.47**

Representing the variable \( x_i \) as a diamond structure
We represent each clause of $\phi$ as a single node, as follows.

\[ \circ \quad c_j \]

**Figure 7.48**
Representing the clause $c_j$ as a node

The following figure depicts the global structure of $G$. It shows all the elements of $G$ and their relationships, except the edges that represent the relationship of the variables to the clauses that contain them.

**Figure 7.49**
The high-level structure of $G$
Next we show how to connect the diamonds representing the variables to the nodes representing the clauses. Each diamond structure contains a horizontal row of nodes connected by edges running in both directions. The horizontal row contains $3k + 1$ nodes in addition to the two nodes on the ends belonging to the diamond. These nodes are grouped into adjacent pairs, one for each clause, with extra separator nodes next to the pairs, as shown in the following figure.

![Diagram 7.50](image)

**Figure 7.50**
The horizontal nodes in a diamond structure

If variable $x_i$ appears in clause $c_j$, we add the following two edges from the $j$th pair in the $i$th diamond to the $j$th clause node.

![Diagram 7.51](image)

**Figure 7.51**
The additional edges when clause $c_j$ contains $x_i$

If $\overline{x_i}$ appears in clause $c_j$, we add two edges from the $j$th pair in the $i$th diamond to the $j$th clause node, as shown in Figure 7.52.

After we add all the edges corresponding to each occurrence of $x_i$ or $\overline{x_i}$ in each clause, the construction of $G$ is complete. To show that this construction works, we argue that, if $\phi$ is satisfiable, a Hamiltonian path exists from $s$ to $t$ and, conversely, if such a path exists, $\phi$ is satisfiable.
FIGURE 7.52
The additional edges when clause $c_j$ contains $\overline{x_i}$

Suppose that $\phi$ is satisfiable. To demonstrate a Hamiltonian path from $s$ to $t$, we first ignore the clause nodes. The path begins at $s$, goes through each diamond in turn, and ends up at $t$. To hit the horizontal nodes in a diamond, the path either zig-zags from left to right or zag-zigs from right to left, the satisfying assignment to $\phi$ determines which. If $x_i$ is assigned TRUE, the path zig-zags through the corresponding diamond. If $x_i$ is assigned FALSE, the path zag-zigs. We show both possibilities in the following figure.

FIGURE 7.53
Zig-zagging and zag-zigging through a diamond, as determined by the satisfying assignment

So far this path covers all the nodes in $G$ except the clause nodes. We can easily include them by adding detours at the horizontal nodes. In each clause, we select one of the literals assigned TRUE by the satisfying assignment.

If we selected $x_i$ in clause $c_j$, we can detour at the $j$th pair in the $i$th diamond. Doing so is possible because $x_i$ must be TRUE, so the path zig-zags from left to right through the corresponding diamond. Hence the edges to the $c_j$ node are in the correct order to allow a detour and return.

Similarly, if we selected $\overline{x_i}$ in clause $c_j$, we can detour at the $j$th pair in the $i$th diamond because $x_i$ must be FALSE, so the path zag-zigs from right to left through the corresponding diamond. Hence the edges to the $c_j$ node again are
in the correct order to allow a detour and return. (Note that each true literal in a clause provides an option of a detour to hit the clause node. As a result, if several literals in a clause are true, only one detour is taken.) Thus we have constructed the desired Hamiltonian path.

For the reverse direction, if $G$ has a Hamiltonian path from $s$ to $t$, we demonstrate a satisfying assignment for $\phi$. If the Hamiltonian path is normal—it goes through the diamonds in order from the top one to the bottom one, except for the detours to the clause nodes—we can easily obtain the satisfying assignment. If the path zig-zags through the diamond, we assign the corresponding variable $\text{TRUE}$, and if it zag-zigs, we assign $\text{FALSE}$. Because each clause node appears on the path, by observing how the detour to it is taken, we may determine which of the literals in the corresponding clause is $\text{TRUE}$.

All that remains to be shown is that a Hamiltonian path must be normal. Normality may fail only if the path enters a clause from one diamond but returns to another, as in the following figure.

![Diagram](image-url)

**FIGURE 7.54**
This situation cannot occur.

The path goes from node $a_1$ to $c$, but instead of returning to $a_2$ in the same diamond, it returns to $b_2$ in a different diamond. If that occurs, either $a_2$ or $a_3$ must be a separator node. If $a_2$ were a separator node, the only edges entering $a_2$ would be from $a_1$ and $a_3$. If $a_3$ were a separator node, $a_1$ and $a_2$ would be in the same clause pair, and hence the only edges entering $a_2$ would be from $a_1$, $a_3$, and $c$. In either case, the path could not contain node $a_2$. The path cannot enter $a_2$ from $c$ or $a_1$ because the path goes elsewhere from these nodes. The path cannot enter $a_2$ from $a_3$, because $a_3$ is the only available node that $a_2$ points at, so the path must exit $a_2$ via $a_3$. Hence a Hamiltonian path must be normal. This reduction obviously operates in polynomial time and the proof is complete.
Next we consider an undirected version of the Hamiltonian path problem, called \textsc{UHampath}. To show that \textsc{UHampath} is NP-complete we give a polynomial time reduction from the directed version of the problem.

**Theorem 7.55**

\textsc{UHampath} is NP-complete.

**Proof** The reduction takes a directed graph $G$ with nodes $s$ and $t$ and constructs an undirected graph $G'$ with nodes $s'$ and $t'$. Graph $G$ has a Hamiltonian path from $s$ to $t$ iff $G'$ has a Hamiltonian path from $s'$ to $t'$. We describe $G'$ as follows.

Each node $u$ of $G$, except for $s$ and $t$, is replaced by a triple of nodes $u^{\text{in}}$, $u^{\text{mid}}$, and $u^{\text{out}}$ in $G'$. Nodes $s$ and $t$ in $G$ are replaced by nodes $s^{\text{out}}$ and $t^{\text{in}}$ in $G'$. Edges of two types appear in $G'$. First, edges connect $u^{\text{mid}}$ with $u^{\text{in}}$ and $u^{\text{out}}$. Second, an edge connects $u^{\text{out}}$ with $v^{\text{in}}$ if an edge goes from $u$ to $v$ in $G$. That completes the construction of $G'$.

We can demonstrate that this construction works by showing that $G$ has a Hamiltonian path from $s$ to $t$ iff $G'$ has a Hamiltonian path from $s^{\text{out}}$ to $t^{\text{in}}$. To show one direction, we observe that a Hamiltonian path $P$ in $G$,

$$s, u_1, u_2, \ldots, u_k, t,$$

has a corresponding Hamiltonian path $P'$ in $G'$,

$$s^{\text{out}}, u_1^{\text{in}}, u_1^{\text{mid}}, u_1^{\text{out}}, u_2^{\text{in}}, u_2^{\text{mid}}, u_2^{\text{out}}, \ldots, t^{\text{in}}.$$

To show the other direction, we claim that any Hamiltonian path in $G'$ from $s^{\text{out}}$ to $t^{\text{in}}$ in $G'$ must go from a triple of nodes to a triple of nodes, except for the start and finish, as does the path $P'$ we just described. That would complete the proof because any such path has a corresponding Hamiltonian path in $G$. We prove the claim by following the path starting at node $s^{\text{out}}$. Observe that the next node in the path must be $u_i^{\text{in}}$ for some $i$ because only those nodes are connected to $s^{\text{out}}$. The next node must be $u_i^{\text{mid}}$, because no other way is available to include $u_i^{\text{mid}}$ in the Hamiltonian path. After $u_i^{\text{mid}}$ comes $u_i^{\text{out}}$ because that is the only other one to which $u_i^{\text{mid}}$ is connected. The next node must be $u_j^{\text{in}}$ for some $j$ because no other available node is connected to $u_i^{\text{out}}$. The argument then repeats until $t^{\text{in}}$ is reached.

**The Subset Sum Problem**

Recall the \textsc{Subset-Sum} problem defined on page 273. In that problem, we were given a collection of numbers $x_1, \ldots, x_k$ together with a target number $t$, and were to determine whether the collection contains a subcollection that adds up to $t$. We now show that this problem is NP-complete.