17 The First NP-Complete Problem

NP. NP is the set of all decision problems for which the instances where the answer is "yes" have efficiently verifiable proofs. More precisely, these proofs have to be verifiable by deterministic computations that can be performed in polynomial time. Obviously, we have $P \subseteq NP$.

NP-Complete (NPC). The set of hardest problems in the class NP. More precisely, X is NPC if (1) $X \in$ NP; and (2) for all $Y \in$ NP, $Y \leq_P X$.

NP-Hard. The set of problems X such that for all $Y \in$ NP, $Y \leq_P X$.

The First NP-Complete problem: Circuit Satisfiability. To show some problem is NP-Complete, one has to show that it could encode *any* problem in NP.

Circuit: we define a circuit K to be a labeled, directed acyclic graph such as the one shown in the example of Figure 8.4. The sources in K (the nodes with no incoming edges) are labeled either with one of the constants 0 or 1, or with the name of a distinct variable. The nodes of the latter type will be referred to as the inputs to the circuit. Every other node is labeled with one of the Boolean operators $\wedge$, $\vee$, or $\neg$; nodes labeled with $\wedge$ or $\vee$ will have two incoming edges, and nodes labeled with $\neg$ will have one incoming edge. There is a single node with no outgoing edges, and it will represent the output: the result that is computed by the circuit.

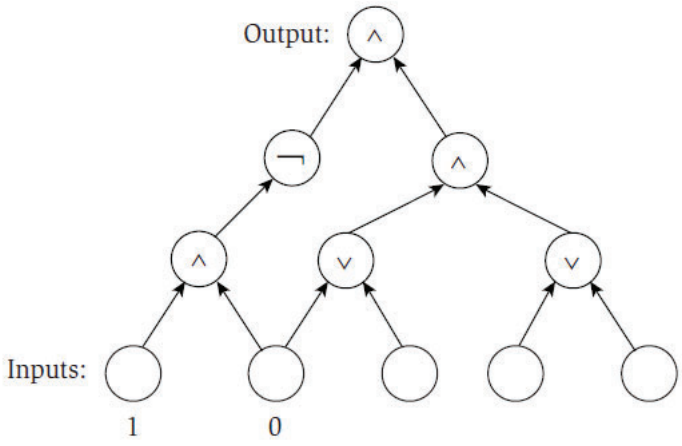

Figure 8.4 A circuit with three inputs, two additional sources that have assigned truth values, and one output.

Circuit Satisfiability: We are given a circuit as input, and we need to decide whether there is an assignment of values to the inputs that causes the output to take the value 1.

An example demonstrating NPC. Consider the following NP problem "Given a graph G, does it contain a two-node independent set"?

The input s can be specified by $\binom{n}{2}$ bits (representing the existence of edges). Proof t can be specified by n bits: for each node we use a bit saying whether this node belongs to the proposed independent set. The verifier needs to check whether at least two of the t bits are set to be 1, and no two '1' bits in t correspond to nodes that are connected. Show a figure on $n = 3$.

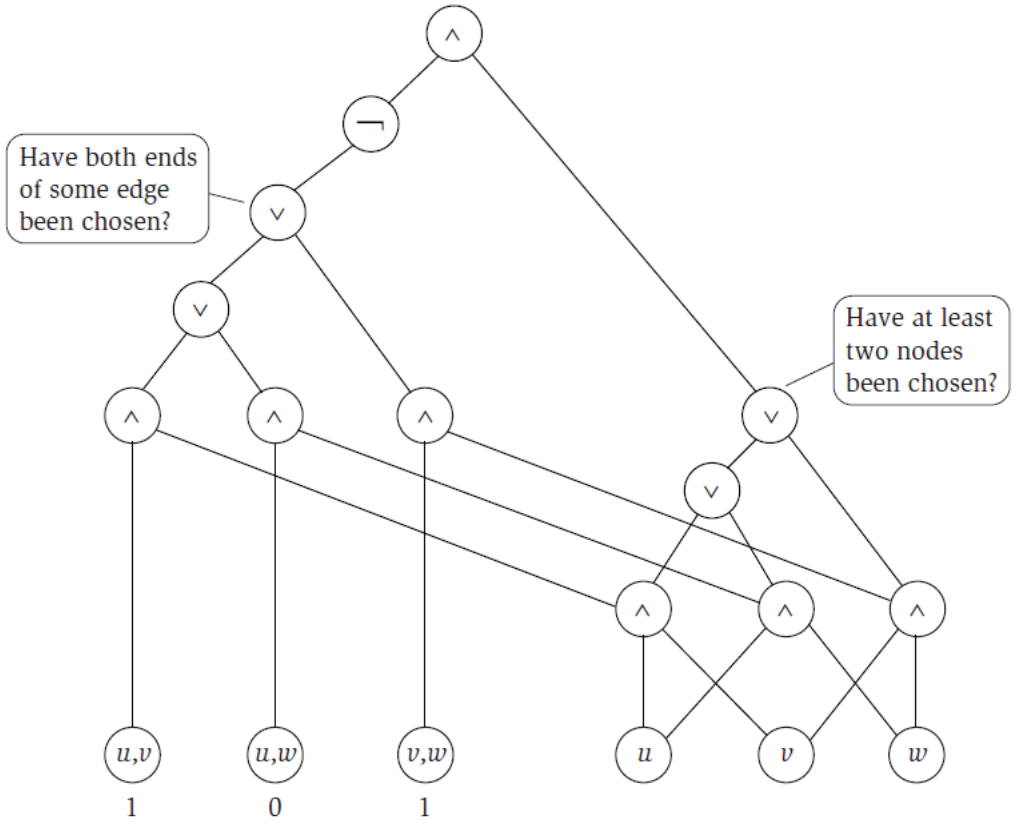

Figure 8.5 A circuit to verify whether a 3-node graph contains a 2-node independent set.