travelling salesman problem is $\mathcal{NP}$-hard. For a wealth of information on $\mathcal{NP}$-hard optimization problems and their approximability properties, see the book [33] by Ausiello, Crescenzi, Gambosi, Kann, Marchetti-Spaccamela and Protasi.

From a complexity point of view, there is no significant difference between a decision problem and its optimization analogue (if it exists). To illustrate this statement, let us consider the problem of deciding whether a strong digraph has a cycle of length at least k (here k is part of the input). The optimization analogue is the problem of finding a cycle of maximum length in a strong digraph. If we solve the optimization problem, we easily obtain a solution to the decision problem: just check whether the length of the longest cycle is at least k. On the other hand, using binary search one can find an answer to the optimization problem by solving a number of decision problems. In our example, we first check whether or not the digraph under consideration has a cycle of length at least $n/2$. Then, solve the analogous problem with $n/4$ (if D has no cycle of length at least $n/2$) or $3n/4$ (if D has a cycle of length at least $n/2$) instead of $n/2$, etc. So, we would need to solve $O(\log n)$ decision problems, in order to obtain an answer to the optimization problem.

1.10 Application: Solving the 2-Satisfiability Problem

In this section we deal with a problem that is not a problem on digraphs, but it has applications to several problems on graphs, in particular when we want to decide whether a given undirected graph has an orientation with certain properties. In Chapter 8 we will give examples of this. We will show how to solve this problem efficiently using the algorithm for strong components of digraphs from Chapter 4.

A **boolean variable** x is a variable that can assume only two values 0 and 1. The **sum** of boolean variables $x_1 + x_2 + \ldots + x_k$ is defined to be 1 if at least one of the x_i's is 1 and 0 otherwise. The **negation** $\overline{x}$ of a boolean variable x is the variable that assumes the value $1 - x$. Hence $\overline{\overline{x}} = x$. Let X be a set of boolean variables. For every $x \in X$ there are two **literals**, over x, namely x itself and $\overline{x}$. A **clause** C over a set of boolean variables X is a sum of literals over the variables from X. The **size** of a clause is the number of literals it contains. For example, if u, v, w are boolean variables with values $u = 0, v = 0$ and $w = 1$, then $C = (u + \overline{v} + \overline{w})$ is a clause of size 3, its value is 1 and the literals in C are u, $\overline{v}$ and $\overline{w}$. An assignment of values to the set of variables X of a boolean expression is called a **truth assignment**. If the variables are $x_1, \ldots, x_k$, then we denote a truth assignment by $t = (t_1, \ldots, t_k)$. Here it is understood that x_i will be assigned the value t_i for $i = 1, \ldots, k$.

The **2-satisfiability problem**, also called **2-SAT**, is the following problem. Let $X = \{x_1, \ldots, x_k\}$ be a set of boolean variables and let $C_1, \ldots, C_r$ be a collection of clauses, all of size 2, for which every literal is over X. Decide if there exists a truth assignment $t = (t_1, \ldots, t_k)$ to the variables in X such that the value of every clause will be 1. This is equivalent to asking whether or not the boolean expression $\mathcal{F} = C_1 * \ldots * C_p$ can take the value 1. Depending on whether this is possible or not, we say that $\mathcal{F}$ is **satisfiable** or **unsatisfiable**. Here '$*$' stands for **boolean multiplication**, that is, $1 * 1 = 1$, $1 * 0 = 0 * 1 = 0 * 0 = 0$. For a given truth assignment $t = (t_1, \ldots, t_k)$ and literal q we denote by $q(t)$ the value of q when we use the truth assignment t (i.e. if $q = \overline{x_3}$ and $t_3 = 1$, then $q(t) = 1 - 1 = 0$)

To illustrate the definitions, let $X = \{x_1, x_2, x_3\}$ and let $C_1 = (\overline{x_1} + \overline{x_3})$, $C_2 = (x_2 + \overline{x_3})$, $C_3 = (\overline{x_1} + x_3)$ and $C_4 = (x_2 + x_3)$. Then it is not difficult to check that $\mathcal{F} = C_1 * C_2 * C_3 * C_4$ is satisfiable and that taking $x_1 = 0, x_2 = 1, x_3 = 1$ we obtain $\mathcal{F} = 1$.

If we allow more than 2 literals per clause then we obtain the more general problem **Satisfiability** (also called **SAT**) which is $\mathcal{NP}$-complete, even if all clauses have size 3, in which case it is also called **3-SAT** (see e.g. page 359 in the book [600] by Papadimitriou and Steiglitz). (In his proof of the existence of an $\mathcal{NP}$-complete problem, Cook used the satisfiability problem and showed how every other problem in $\mathcal{NP}$ can be reduced to this problem.) Below we will show how to reduce 2-SAT to the problem of finding the strong components in a certain digraph. We shall also show how to find a satisfying truth assignment if one exists. This step is important in applications, such as those in Chapter 8.

Let $C_1, \ldots, C_r$ be clauses of size 2 such that the literals are taken among the variables $x_1, \ldots, x_k$ and their negations and let $\mathcal{F} = C_1 * \ldots * C_r$ be an instance of 2-SAT. Construct a digraph $D_{\mathcal{F}}$ as follows. Let $V(D_{\mathcal{F}}) = \{x_1, \ldots, x_k, \overline{x_1}, \ldots, \overline{x_k}\}$ (i.e. $D_{\mathcal{F}}$ has two vertices for each variable, one for the variable and one for its negation). For every choice of $p, q \in V(D_{\mathcal{F}})$ such that some C_i has the form $C_i = (p + q)$, $A(D_{\mathcal{F}})$ contains an arc from $\overline{p}$ to q and an arc from $\overline{q}$ to p (recall that $\overline{\overline{x}} = x$). See Figure 1.20 for examples of a 2-SAT expressions and the corresponding digraphs. The first expression is satisfiable, the second is not.

Lemma 1.10.1 *If $D_{\mathcal{F}}$ has a (p, q)-path, then it also has a $(\overline{q}, \overline{p})$-path. In particular, if p, q belong to the same strong component in $D_{\mathcal{F}}$, then $\overline{p}, \overline{q}$ belong to the same strong component in $D_{\mathcal{F}}$.*

Proof: This follows easily by induction on the length of a shortest (p, q)-path, using the fact that $(x, y) \in A(D_{\mathcal{F}})$ if and only if $(\overline{y}, \overline{x}) \in A(D_{\mathcal{F}})$. $\square$

Lemma 1.10.2 *If $D_{\mathcal{F}}$ contains a path from p to q, then, for every satisfying truth assignment t, $p(t) = 1$ implies $q(t) = 1$.*

Proof: Observe that $\mathcal{F}$ contains a clause of the form $(\overline{a} + b)$ and every clause takes the value 1 under any satisfying truth assignment. Thus, by the fact that t is a satisfying truth assignment and by the definition of $D_{\mathcal{F}}$, we have that for every arc $(a, b) \in A(D_{\mathcal{F}})$, $a(t) = 1$ implies $b(t) = 1$. Now the claim follows easily by induction on the length of the shortest (p, q)-path in $D_{\mathcal{F}}$. $\square$

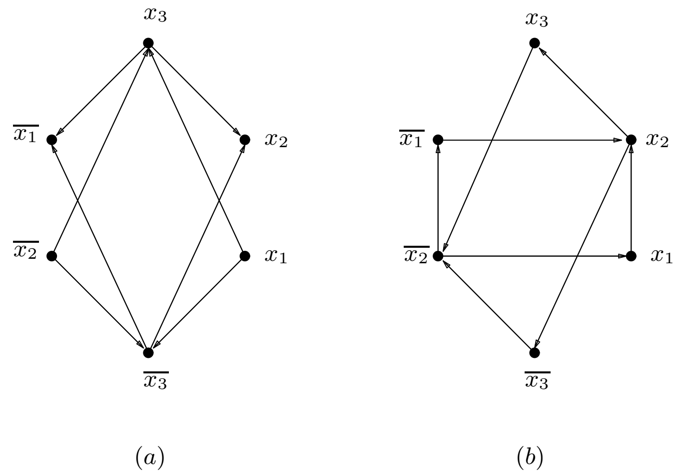

Figure 1.20 The digraph $D_{\mathcal{F}}$ is shown for two instances of 2-SAT. In (a) $\mathcal{F} = (\overline{x_1} + \overline{x_3}) * (x_2 + \overline{x_3}) * (\overline{x_1} + x_3) * (x_2 + x_3)$ and in (b) $\mathcal{F} = (x_1 + x_2) * (\overline{x_1} + x_2) * (\overline{x_2} + x_3) * (\overline{x_2} + \overline{x_3})$

The following is an easy corollary of Lemma 1.10.1 and Lemma 1.10.2.

Corollary 1.10.3 *If t is a satisfying truth assignment, then for every strong component D' of $D_{\mathcal{F}}$ and every choice of distinct vertices $p, q \in V(D')$ we have $p(t) = q(t)$. Furthermore we also have $\overline{p}(t) = \overline{q}(t)$.* $\square$

Lemma 1.10.4 *$\mathcal{F}$ is satisfiable if and only if for every $i = 1, 2, \ldots, k$, no strong component of $D_{\mathcal{F}}$ contains both the variable x_i and its negation $\overline{x_i}$.*

Proof: Suppose t is a satisfying truth assignment for $\mathcal{F}$ and that there is some variable x_i such that x_i and $\overline{x_i}$ are in the same strong component in $D_{\mathcal{F}}$. Without loss of generality $x_i(t) = 1$ and now it follows from Lemma 1.10.2 and the fact that $D_{\mathcal{F}}$ contains a path from x_i to $\overline{x_i}$ that we also have $\overline{x_i}(t) = 1$ which is impossible. Hence if $\mathcal{F}$ is satisfiable, then for every $i = 1, 2, \ldots, k$, no strong component of $D_{\mathcal{F}}$ contains both the variable x_i and its negation $\overline{x_i}$.

Now suppose that for every $i = 1, 2, \ldots, k$, no strong component of $D_{\mathcal{F}}$ contains both the variable x_i and its negation $\overline{x_i}$. We will show that $\mathcal{F}$ is satisfiable by constructing a satisfying truth assignment for $\mathcal{F}$.

Let $D_1, \ldots, D_s$ denote some acyclic ordering of the strong components of $D_{\mathcal{F}}$ (i.e. there is no arc from D_j to D_i if $i < j$). Recall that by Proposition 1.4.3, such an ordering exists. We claim that the following algorithm will determine a satisfying truth assignment for $\mathcal{F}$: first mark all vertices 'unassigned' (meaning truth value still pending). Then going backwards starting from D_s and ending with D_1 we proceed as follows. If there is any vertex $v \in V(D_i)$ such that $\overline{v}$ has already been assigned a value, then assign all vertices in D_i the value 0 and otherwise assign all vertices in D_i the value 1. Observe that this means that, for every variable x_i, we will always assign the value 1 to whichever of x_i, $\overline{x_i}$ belongs to the strong component with the highest index. To see this, let p denote whichever of $x_i, \overline{x_i}$ belongs to the strong component of highest index j. Let $i < j$ be chosen such that $\overline{p} \in D_i$. Suppose we assign the value 0 to p. This means that at the time we considered p, there was some $q \in D_j$ such that $\overline{q} \in D_f$ for some $f > j$. But then $\overline{p} \in D_f$, by Lemma 1.10.1, contradicting the fact that $i < f$.

Clearly all vertices in $V(\mathcal{F})$ will be assigned a value, and by our previous argument this is consistent with a truth assignment for the variables of $\mathcal{F}$. Hence it suffices to prove that each clause has value 1 under the assignment. Suppose some clause $C_f = (p + q)$ attains the value 0 under our assignment. By our observation above, the reason we did not assign the value 1 to p was that at the time we considered p we had already given the value 1 to $\overline{p}$ and $\overline{p}$ belonged to a component D_j with a higher index than the component D_i containing p. Thus the existence of the arc $(\overline{p}, q) \in A(D_{\mathcal{F}})$ implies that $q \in D_h$ for some $h \geq j$. Applying the analogous argument to q we conclude that $\overline{q}$ is in some component D_g with $g > h$. But then, using the existence of the arc $(\overline{q}, p)$, we get that $i \geq g > h \geq j > i$, a contradiction. This shows that we have indeed found a correct truth assignment for $\mathcal{F}$ and hence the proof is complete. $\square$

In Chapter 4 we will see that for any digraph D one can find the strong components of D and an acyclic ordering of these in $O(n+m)$ time. Since the number of arcs in $D_{\mathcal{F}}$ is twice the number of clauses in $D_{\mathcal{F}}$ and the number of vertices in $D_{\mathcal{F}}$ is twice the number of variables in $D_{\mathcal{F}}$, it is not difficult to see that the algorithm outlined above can be performed in time $O(k + r)$ and hence we have the following result.

Theorem 1.10.5 *The problem 2-SAT is solvable in linear time with respect to the number of clauses.* $\square$

The approach we adopted is outlined briefly in Exercise 15.6 of the book [600] by Papadimitriou and Steiglitz, see also the paper [230] by Even, Itai and Shamir.

It is interesting to note that if, instead of asking whether $\mathcal{F}$ is satisfiable, we ask whether there exists some truth assignment such that at least ℓ clauses will get the value 1, then this problem, which is called **MAX-2-SAT**, is $\mathcal{NP}$-complete as shown by Garey, Johnson and Stockmeyer [304] (here ℓ is part of the input for the problem).

1.11 Exercises

1.1. Let X and Y be finite sets. Show that $|X \cup Y| + |X \cap Y| = |X| + |Y|$.

1.2. Let X and Y be finite sets. Show that $|X \cup Y|^2 + |X \cap Y|^2 \geq |X|^2 + |Y|^2$.