Appendix: Proving MDV, MSV $\in$ NP-complete

In this appendix we prove that the novel problems considered in this chapter, MDV and MSV, belong to the NP-complete class. This implies that exponential times are (very probably) required to optimally solve them. Thus, sub-optimally solving them by means of heuristic algorithms like those considered in this chapter is an appropriate choice. The proof is structured as follows. First, we prove that both problems belong to the NP class. Next, we prove that a well-known NP-complete problem, 3-SAT, can be polynomially reduced to each considered problem, which implies that they belong to the NP-complete class. At the end of this appendix, we study variants of MDV and MSV where only graphs fulfilling $N = O$ (that is, graphs where *all* nodes are origin nodes) are considered, showing that these variants are also NP-complete.

Lemma 1. MDV $\in$ NP and MSV $\in$ NP.

Proof. We consider MDV $\in$ NP; proving MSV $\in$ NP is similar. We prove that MDV can be solved in polynomial time by a non-deterministic algorithm. Given a variable-cost graph G and a natural number $K \in \mathbb{N}$, this algorithm non-deterministically constructs a subgraph G' of G and next deterministically checks whether (a) G' is a tree of G, and (b) we have $dc(G') \leq K$. Both operations are performed in polynomial time with respect to the size of G and the size of K (measured in bits). Given a subgraph G' of G, checking whether G' is a tree of G requires polynomial time. Next, if G' is a tree of G, calculating $dc(G')$ requires traversing all paths connecting each origin node to the destination node and adding the costs of all of these paths. The length of each of these paths is polynomial, so calculating the cost of a path requires polynomial time. Since G' is a tree, for each origin node there exists a single path connecting it to the destination node. Thus, the number of paths to be considered is polynomial. Hence, we can check whether the property $dc(G') \leq K$ holds or not in polynomial time. $\square$

In order to prove the NP-completeness of MDV and MSV, we construct a polynomial reduction of a known NP-complete problem to each of these problems. In particular, we consider the well-known 3-SAT problem. Next we introduce some notions related to this problem as well as the problem itself.

Definition 1. The 3-SAT problem is stated as follows: Given a propositional logic formula φ expressed in conjunctive normal form where each disjunctive clause has at most 3 literals, is there any valuation v satisfying φ?

Let $\varphi \equiv (l_{11} \vee l_{12} \vee l_{13}) \wedge \ldots \wedge (l_{k1} \vee l_{k2} \vee l_{k3})$ be an input for 3-SAT. We denote by $\mathtt{props}(\varphi) = \{p_1, \ldots, p_n\}$ the set of propositional symbols appearing in φ. We denote the i-th disjunctive clause of φ by c_i, that is, $c_i \equiv l_{i1} \vee l_{i2} \vee l_{i3}$.

We say that c_i *holds when* p_j *is equal to* $x \in \{\top, \bot\}$, formally denoted by $\mathsf{h}(p_j, x, c_i)$, if for all valuation v fulfilling $v(p_j) = x$ we have that c_i evaluates to $\top$. That is, $\mathsf{h}(p_j, \top, c_i)$ iff $l_{im} \equiv p_j$ for some $1 \leq m \leq 3$, and $\mathsf{h}(p_j, \bot, c_i)$ iff $l_{im} \equiv \neg p_j$ for some $1 \leq m \leq 3$. $\square$

Theorem 1. 3-SAT $\in$ NP-complete. $\square$

We prove MDV, MSV $\in$ NP-complete as follows (next we consider MDV; the same arguments are given in the case of MSV). Given an input φ of 3-SAT, we show that we can construct an input (G,K) of MDV from φ in polynomial time in such a way that the solution of 3-SAT for φ is *yes* iff the solution of MDV for the variable-cost graph G and the natural number K is *yes*. By the definition of the NP-complete class, this implies MDV $\in$ NP-complete. In particular, if we were able to solve MDV in polynomial time then we could solve the NP-complete problem 3-SAT in polynomial time as well: We could just transform φ into (G,K), next call the algorithm that solves MDV in polynomial time, and finally return the answer given by it.

Before formally presenting the construction of (G,K) from φ, let us informally introduce it. Each origin node of the constructed graph G represents a *disjunctive clause* of φ. From each of these origin nodes, edges iteratively lead through some nodes representing each *proposition symbol* appearing in φ. Each of these proposition nodes is connected to the next proposition node through two edges. One of them represents valuating the corresponding proposition symbol to $\top$, while the other edge represents giving it the $\bot$ value. Depending on the origin node where we come from (that is, depending on the disjunctive clause we are considering), taking the edge that evaluates the proposition symbol to true or to false adds a different cost to the path. This cost is 1 *unless* the proposition valuation represented by the edge allows to make true the disjunctive clause *for the first time* in the path. In this case, the edge adds 0 to the overall path cost. In order to keep track of this information, the value of the *variable* of the variable-cost graph G codifies the considered clause, as well as whether this clause necessarily holds (according to the valuation represented by the path traversed so far). In particular, variable values follow the form $v_{j,w}$ where j is an index denoting a clause and $w \in \{already\top, notyet\top\}$. A value $v_{j,w}$ denotes that the current path departed at an origin node denoting the j-th clause of φ, and $w = already\top$ denotes that the j-th clause must be true regardless of the valuation of the remaining proposition symbols (because the valuation implicitly defined by the path traversed so far necessarily makes it true). Otherwise, we consider $w = notyet\top$. After the last proposition node is traversed, the destination node of G reached. The structure of G is depicted in Figure 1.

Recall that MDV seeks for a tree where the addition of costs from each origin node to the destination node is minimal. On the other hand, MSV seeks for a tree where the addition of average edge costs is minimal. Let us note that, given the variable-cost graph G, a tree of G can include only *one* of the edges that connect each proposition node to the next proposition node (otherwise, it would not be a tree). Hence, given G, trees computed by both problems represent valuations of proposition symbols. Since MDV searches the cheapest tree connecting all origin nodes to the destination node, MDV actually seeks for a tree allowing to make true as more clauses as possible. In particular, we will prove that the cost of the cheapest tree found by MDV is under a given threshold if and only if *all* clauses are true under the constructed valuation, that is, iff φ holds. Moreover, due to the specific form of G, finding a tree where the addition of costs from each origin to the destination is minimal is equivalent

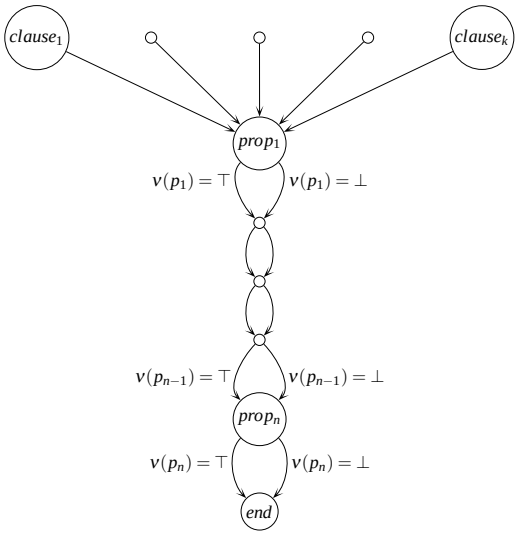

Fig. 1 Structure of the variable-cost graph G.

to finding a tree where the addition of *average* costs of edges is minimal: Due to the definition of G, for both problems the tree cost is minimized if the valuation represented by the tree makes true as more clauses as possible. Hence, we can also define a threshold such that the cost of the minimum tree for MSV is under it iff φ is satisfiable.

Theorem 2. MDV $\in$ NP-complete and MSV $\in$ NP-complete.

Proof. First, we prove MDV $\in$ NP-complete. Due to Lemma 1, if we polynomially reduce 3-SAT to MDV then MDV $\in$ NP-complete. Let φ denote a conjunctive normal form $\varphi \equiv c_1 \wedge \ldots \wedge c_k$ where $\texttt{props}(\varphi) = \{p_1, \ldots, p_n\}$. We construct a variable-cost graph $G = (N, O, d, V, A, E)$ as follows:

- $N = \{clause_1, \ldots, clause_k, prop_1, \ldots, prop_n, end\}$,
- $O = \{clause_1, \ldots, clause_k\}$,
- $d = end$,
- $V = \{v_{j,w} | 1 \leq j \leq k \wedge w \in \{already\top, notyet\top\}\}$
- For all $clause_i \in O$ we have $A(clause_i) = v_{i,notyet\top}$.

- $E = \{(clause_i, prop_1, C, T) | 1 \leq i \leq k \wedge \forall v \in V : (C(v) = 0 \wedge T(v) = v)\}$

$$\cup$$

$$\left\{ \left(\begin{array}{c} prop_i, \\ prop_{i+1}, \\ C_i^x, \\ T_i^x \end{array} \right) \left| \begin{array}{l} 1 \leq i \leq k-1 \wedge x \in \{\top, \bot\} \wedge \\ C_i^x(v_{j,notyet\top}) = \left\{ \begin{array}{l} 0 \texttt{ if } \mathsf{h}(p_i, x, c_j) \\ 1 \texttt{ otherwise} \end{array} \right\} \wedge \\ C_i^x(v_{j,already\top}) = 1 \wedge \\ T_i^x(v_{j,notyet\top}) = \left\{ \begin{array}{l} v_{j,already\top} \texttt{ if } \mathsf{h}(p_i, x, c_j) \\ v_{j,notyet\top} \texttt{ otherwise} \end{array} \right\} \wedge \\ T_i^x(v_{j,already\top}) = v_{j,already\top} \end{array} \right. \right\}$$

$$\cup$$

$$\left\{ \left(\begin{array}{c} prop_n, \\ end, \\ C_n^x, \\ T_n^x \end{array} \right) \left| \begin{array}{l} x \in \{\top, \bot\} \wedge \\ C_n^x(v_{j,notyet\top}) = \left\{ \begin{array}{l} 0 \texttt{ if } \mathsf{h}(p_n, x, c_j) \\ 1 \texttt{ otherwise} \end{array} \right\} \wedge \\ C_n^x(v_{j,already\top}) = 1 \wedge \\ T_n^x(v_{j,notyet\top}) = \left\{ \begin{array}{l} v_{j,already\top} \texttt{ if } \mathsf{h}(p_n, x, c_j) \\ v_{j,notyet\top} \texttt{ otherwise} \end{array} \right\} \wedge \\ T_n^x(v_{j,already\top}) = v_{j,already\top} \end{array} \right. \right\}$$

We show that constructing G from φ requires polynomial time. This property is a consequence of the following conditions:

(a) $|N|$ is equal to the number of clauses of φ plus the number of proposition symbols of φ plus 1 (the *end* node), which is polynomial with respect to the size of φ.

(b) $|V|$ is equal to the number of disjunctive clauses of φ multiplied by 2. Thus, for each edge in E, defining functions C and T by means of extensional arrays (relating each input value with its output value) requires polynomial size and time.

(c) $|E|$ is equal to the number of clauses plus the number of propositions multiplied by 2, which is polynomial with respect to the size of φ.

Finally, we prove that the *answer* of `MDV` for G and a given threshold is *yes* iff φ is satisfiable. In particular, we prove that φ is satisfiable iff there exists a tree G' of G such that $dc(G') \leq k * (n-1)$. We consider each implication of this statement:

$\Rightarrow$:Let us note that a tree G' of G must include all edges connecting each node $clause_i$ with $prop_1$. All of these edges have 0 cost. Besides, for each pair of edges connecting each node $prop_i$ with node $prop_{i+1}$, the tree G' must include exactly one of these edges. Let us consider a valuation v such that for all $1 \leq i \leq n$ we have $v(p_i) = \top$ if G' includes the edge $(prop_i, prop_{i+1}, C_i^\top, T_i^\top)$ and $p_i = \bot$ if G' includes $(prop_i, prop_{i+1}, C_i^\bot, T_i^\bot)$. For all $clause_i \in O$, the cost of the path from $clause_i$ to *end* in G' is $n-1$ if v makes true c_i, and n otherwise. This is because if v makes c_i true then all edges in the path but *one* add 1 cost to this path. The exception is the edge that makes true c_i for the first time, which adds 0 cost. If φ is satisfiable then there exists a valuation v' making true *all* clauses c_i. Thus, there exists a way to choose the edges connecting each $prop_i$ with $prop_{i+1}$ in such a way that, for all $clause_i$, the unique path from $clause_i$ to *end* has $n-1$ cost. In this case, $dc(G') = k * (n-1)$.

$\Leftarrow$:Let us consider a valuation v defined as in the previous case. If the cost of G' is $k*(n-1)$ then the cost from each $clause_i$ to *end* must be $n-1$. This implies that, for each $1 \leq i \leq n$, v makes true the clause c_i. Hence, φ is satisfiable.

We prove MSV $\in$ NP-complete by following very similar arguments. In particular, we can construct a polynomial reduction of 3-SAT to MSV by using the same variable-cost graph G defined before. In this case, we have that φ is satisfiable iff there exists a tree G' of G such that $sc(G') \leq n-1$. Let us recall that, in MSV, the cost of each edge e of G' is the average cost of e for all paths traversing e. Due to the structure of G, it is easy to check that $sc(G') = \frac{dc(G')}{k}$. Thus, φ is satisfiable iff $sc(G') = \frac{k*(n-1)}{k} = n-1$. $\square$

It is worth to point out that the goal of the previous construction is proving the NP-completeness of MDV and MSV, not providing a suitable graph construction to solve 3-SAT by means of RFD or ACO. In particular, if the variable-cost graph G were used to find solutions to 3-SAT by means of RFD, then we would need to introduce a *barrier node* at each edge connecting a node $prop_i$ with $prop_{i+1}$ (see details about barrier nodes in [**?**]).

Finally, we study the relation of MSV with other NP-complete problems. Let us note that MSV is a generalization of the *Minimum Steiner Tree* problem. This NP-complete problem is stated as follows: Given a (non-variable) cost-evaluated graph G and a subset S of its nodes, find the minimum spanning tree including (at least) all nodes in S. MSV generalizes this problem by considering *variable-cost* graphs instead of fix-cost graphs (in particular, the set of origin nodes O in MSV corresponds to the set S given in the previous problem definition). Thus, we may argue that the NP-completeness of MSV is a consequence of the NP-completeness of the Minimum Steiner Tree problem. However, the NP-completeness of MSV (as well as the NP-completeness of MDV) does not lie *only* in the fact that a given *subset* of nodes is required to be included in the tree. In fact, even if only trees including *all* nodes are considered, the NP-completeness of MDV and MSV is met as well – though, in this case, the Minimum Steiner Tree problem would *not* be NP-complete, because it would be equivalent to the (standard) Minimum Spanning Tree problem (which can be polynomially solved).

Let MSV$'$ and MDV$'$ be problems defined as MDV and MSV, respectively, but with the following difference: Only graphs fulfilling $N = O$ (that is, graphs where all nodes are origin nodes) are considered. We prove MSV$'$,MDV$'$ $\in$ NP-complete by using very similar arguments as before. In particular, let us consider the same construction as in the proof of Theorem 2, but now nodes $prop_1, \ldots, prop_n, end$ are also included in the set of origin nodes O. A new variable value $nullCost \in V$ is assigned, as initial value, to all of these new nodes, that is, $A(a) = nullCost$ for all $a \in \{prop_1, \ldots, prop_n, end\}$. If *nullCost* is the current variable value, then taking the next edge is costless and the value *nullCost* remains after taking any edge. Formally, for all transition $(n_1, n_2, C, T) \in E$ we have $C(nullCost) = 0$ and $T(nullCost) = nullCost$. For the rest of variable values, the behavior of edges remains exactly as defined in the proof of Theorem 2. By using the same arguments as those given in

that proof, we infer $\texttt{MSV}', \texttt{MDV}' \in \texttt{NP-complete}$. Thus, the presence of *variable-cost* edges is a sufficient condition for the NP-completeness of both problems.