Algorithm for the Minimum Caterpillar Problem with Terminals*

Taku Okada[1,a)] Akira Suzuki[1,b)] Takehiro Ito[1,c)] Xiao Zhou[1,d)]

Abstract: Suppose that each arc in a digraph $D = (V, A)$ has two costs of non-negative integers, called a spine cost and a leaf cost. A caterpillar is a directed tree consisting of a single directed path (of spine arcs) and leaf vertices each of which is incident to the directed path by exactly one incoming arc (leaf arc). For a given terminal set $K \subseteq V$, we study the problem of finding a caterpillar in D such that it contains all terminals in K and its total cost is minimized, where the cost of each arc in the caterpillar depends on whether it is used as a spine arc or a leaf arc. In this paper, we first show that the problem is NP-hard even for three terminals. We then give a linear-time algorithm to solve the problem for digraphs with bounded treewidth, where the treewidth for a digraph D is defined as the one for the underlying graph of D. Our algorithm runs in linear time even if $|K| = O(|V|)$.

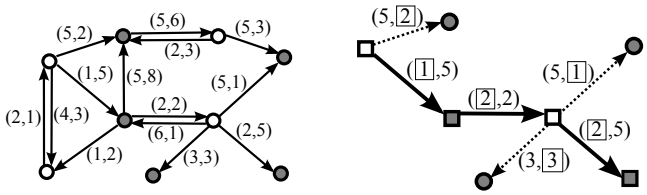
1. Introduction

Let $D = (V, A)$ be a digraph whose vertex set is V and arc set is A; we sometimes denote by $V(D)$ the vertex set of D and by $A(D)$ the arc set of D. A digraph F with a subset $S \subseteq V(F)$ is called a *caterpillar*, denoted by $\langle F, S \rangle$, if S induces a directed path in F and every vertex $V(F) \setminus S$ has no outgoing arc and has exactly one incoming arc; the directed path induced by S is called the *spine* of $\langle F, S \rangle$. We denote by $A_S(F, S)$ the set of all arcs on the spine of $\langle F, S \rangle$; each arc in $A_S(F, S)$ is called a *spine arc*, and each arc in $A_L(F, S) = A(F) \setminus A_S(F, S)$ is called a *leaf arc*. Figure 1(b) illustrates a caterpillar $\langle F, S \rangle$, where each vertex in S is depicted by a square, each spine arc by a thick arrow, and each leaf arc by a dotted arrow.

Suppose that we are given a digraph $D = (V, A)$ together with two cost functions $c_S : A \to \mathbb{Z}^+$ and $c_L : A \to \mathbb{Z}^+$, where $\mathbb{Z}^+$ is the set of all non-negative integers. Then, for a caterpillar $\langle F, S \rangle$ as a subgraph of D, the *cost* $c(F, S)$ of $\langle F, S \rangle$ is defined as follows:

$$c(F, S) = \sum_{e \in A_S(F,S)} c_S(e) + \sum_{e \in A_L(F,S)} c_L(e).$$

Let $K \subseteq V$ be a given set of vertices, called *terminals*. Then, a caterpillar $\langle F, S \rangle$ is called a *K-caterpillar* if $K \subseteq V(F)$. The *minimum caterpillar problem* is to find a K-caterpillar $\langle F, S \rangle$ as a subgraph of D whose cost $c(F, S)$ is minimized. Note that a digraph does not always have a K-caterpillar for a given set $K \subseteq V(D)$. In the instance of Fig. 1(a), there are five terminals, each of which is shaded, and the two costs for each arc $e \in A$ are depicted by a pair $(c_S(e), c_L(e))$. Then, the K-caterpillar $\langle F, S \rangle$ in Fig. 1(b) is an optimal solution for the instance of Fig. 1(a), whose cost is $c(F, S) = (1 + 2 + 2) + (2 + 1 + 3) = 11$.



(a) $D = (V, A)$ (b) $\langle F, S \rangle$

Fig. 1 (a) An instance of the minimum caterpillar problem, and (b) its optimal solution.

The minimum caterpillar problem in digraphs is a generalization of the minimum *spanning* caterpillar problem in undirected graphs, defined as follows [3], [4], [8]: the *minimum spanning caterpillar problem* is the minimum caterpillar problem in which all vertices in a given digraph D are terminals, that is, $K = V(D)$, and there always exists an arc (u, v) if there is an arc (v, u) such that $c_S((u, v)) = c_S((v, u))$ and $c_L((u, v)) = c_L((v, u))$. The minimum spanning caterpillar problem (and hence the minimum caterpillar problem) has some applications to the network design problem, the facility transportation problem, etc [4], [8]. However, the minimum spanning caterpillar problem is known to be NP-hard [4], and hence the minimum caterpillar problem is also NP-hard in general. For the minimum spanning caterpillar problem on general graphs, Simonetti *et al.* [8] gave a non-polynomial-time exact algorithm, and Dinneen and Khosravani [4] studied the problem from the viewpoint of approximation. Dinneen and Khosravani [3] also gave a linear-time (exact) algorithm for (undirected) graphs with bounded treewidth.

In this paper, we give two results for the minimum caterpillar problem. We first show that the problem remains NP-hard even for digraphs with three terminals. Note that the known re-

*This work is partially supported by JSPS Grant-in-Aid for Scientific Research, Grant Numbers 24.3660 (A. Suzuki), 22700001 (T. Ito) and 23500001 (X. Zhou).

[1] Graduate School of Information Sciences, Tohoku University.
[a)] okada@ecei.tohoku.ac.jp
[b)] a.suzuki@ecei.tohoku.ac.jp
[c)] takehiro@ecei.tohoku.ac.jp
[d)] zhou@ecei.tohoku.ac.jp

sult of [4] does not imply the NP-hardness for a constant number of terminals. We then give a linear-time algorithm to solve the problem for digraphs with bounded treewidth. Note that, in this paper, the treewidth of a digraph D is defined simply as the one of the "underlying graph" of D, and hence it is different from [6]. (The formal definition will be given in Section 3.1.) We remark that our algorithm runs in linear time even if $|K| = O(n)$, where n is the number of vertices in a digraph. Therefore, our algorithm improves the known one [3] in the sense that our algorithm also solves the minimum spanning caterpillar problem in linear time for (undirected) graphs with bounded treewidth.

It is known that any optimization problem that can be expressed by Extended Monadic Second Order Logic (EMSOL) can be solved in linear time for graphs with bounded treewidth [2]. However, the algorithm obtained by this method is hard to implement, and is very slow since the hidden constant factor of the running time is a tower of exponentials of unbounded height with respect to the treewidth [7]. On the other hand, our algorithm is simple, and the hidden constant factor is just a single exponential of the treewidth.

2. NP-hardness

The main result of this section is the following theorem.

Theorem 1 The minimum caterpillar problem is NP-hard even for digraphs with three terminals.

proof. We give a polynomial-time reduction from the directed vertex-disjoint paths problem [5] to the minimum caterpillar problem for digraphs with three terminals. In the directed vertex-disjoint paths problem, we are given a digraph D' and k pairs $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$ of vertices $s_i, t_i \in V(D')$, $1 \le i \le k$, and we are asked to determine whether all the k pairs have directed s_it_i-paths in D' such that they do not share any vertex, where a *directed s_it_i-path* is a directed path which starts from s_i and ends in t_i. This problem is known to be NP-complete even for two pairs, that is, $k = 2$ [5].

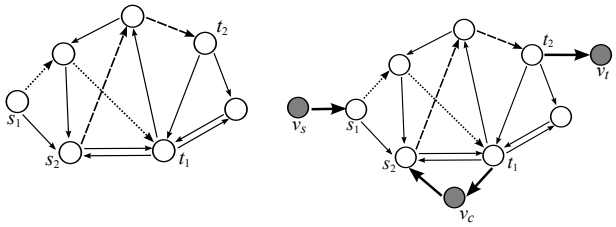
[Construction of the corresponding instance]

Suppose that we are given a digraph $D' = (V', A')$ and two pairs (s_1, t_1) and (s_2, t_2), as an instance of the directed vertex-disjoint paths problem for $k = 2$. Note that the four vertices s_1, t_1, s_2, t_2 are all distinct; otherwise the answer is clearly "No." In the following, we construct a digraph $D = (V, A)$, two cost functions $c_S : A \to \mathbb{Z}^+$ and $c_L : A \to \mathbb{Z}^+$, and a set $K \subseteq V$ of terminals with $|K| = 3$, as the corresponding instance of the minimum caterpillar problem.

The corresponding digraph D is made from a copy of D' together with three new vertices v_s, v_t, v_c and four new arcs (v_s, s_1), (t_1, v_c), (v_c, s_2), (t_2, v_t), that is, $V = V' \cup \{v_s, v_t, v_c\}$ and $A = A' \cup \{(v_s, s_1), (t_1, v_c), (v_c, s_2), (t_2, v_t)\}$. (See Fig. 2.) For each arc $e \in A$, let $c_S(e) = 0$ and $c_L(e) = 1$. Finally, let $K = \{v_s, v_t, v_c\}$, and hence $|K| = 3$. This completes the construction of the corresponding instance. The construction can be clearly done in polynomial time.

[Correctness]

To complete the proof, we now prove that D' has two directed vertex-disjoint paths for the two pairs (s_1, t_1) and (s_2, t_2) if and only if D has a K-caterpillar $\langle F, S \rangle$ such that $c(F, S) = 0$. Notice that $c(F, S) = 0$ if and only if $\langle F, S \rangle$ contains no leaf arc and hence is simply a directed path (spine).



(a) $D' = (V', A')$ (b) $D = (V, A)$

Fig. 2 (a) An instance of the directed vertex-disjoint paths problem for $k = 2$, and (b) its corresponding instance of the minimum caterpillar problem.

We first prove the necessity. Suppose that D' has a directed s_1t_1-path P_1 and a directed s_2t_2-path P_2 such that they are vertex-disjoint. Then, $A(P_1) \cup A(P_2) \cup \{(v_s, s_1), (t_1, v_c), (v_c, s_2), (t_2, v_t)\}$ clearly induces a directed path in D from v_s to v_t. The directed v_sv_t-path is a K-caterpillar $\langle F, S \rangle$ which consists only of spine arcs. Since $c_S(e) = 0$ for all arcs $e \in A$, we have $c(F, S) = 0$. Therefore, D has a K-caterpillar $\langle F, S \rangle$ such that $c(F, S) = 0$.

We then prove the sufficiency. Suppose that D has a K-caterpillar $\langle F, S \rangle$ such that $c(F, S) = 0$. Then, $\langle F, S \rangle$ must consist only of spine arcs since $c_L(e) = 1$ for all arcs $e \in A$. Therefore, $\langle F, S \rangle$ is a directed path containing all the three terminals v_s, v_t and v_c. Since v_s has no incoming arc and has exactly one outgoing arc (v_s, s_1), the spine of $\langle F, S \rangle$ must start from the terminal v_s. On the other hand, since v_t has no outgoing arc and has exactly one incoming arc (t_2, v_t), the spine of $\langle F, S \rangle$ must end in the terminal v_t. Therefore, $\langle F, S \rangle$ must be a directed v_sv_t-path that contains the terminal v_c as an internal vertex. Since v_c has only one incoming arc (t_1, v_c) and only one outgoing arc (v_c, s_2), the vertices v_s, s_1, t_1, v_c, s_2, t_2, v_4 lie on $\langle F, S \rangle$ in this order. Therefore, the K-caterpillar $\langle F, S \rangle$ can be partitioned into the following five directed paths (a)–(e):

(a) a directed v_ss_1-path consisting of a single arc (v_s, s_1);
(b) a directed s_1t_1-path P_1;
(c) a directed t_1s_2-path consisting of two arcs (t_1, v_c) and (v_c, s_2);
(d) a directed s_2t_2-path P_2; and
(e) a directed t_2v_t-path consisting of a single arc (t_2, v_t).

Notice that P_1 and P_2 are contained in D', and hence D' has a directed s_1t_1-path P_1 and a directed s_2t_2-path P_2 such that they are vertex-disjoint. □

3. Algorithm for Digraphs with Bounded Treewidth

The main result of this section is the following theorem.

Theorem 2 The minimum caterpillar problem can be solved in linear time for digraphs with bounded treewidth.

In this section, we give such an algorithm as a proof of Theorem 2. Indeed, for a given digraph D and a given terminal set K, we give a linear-time algorithm which computes the minimum cost of a K-caterpillar in D; it is easy to modify our algorithm so that

it actually finds a K-caterpillar with the minimum cost.

The rest of this section is organized as follows. In Section 3.1, we formally define the notion of treewidth for a digraph and introduce its tree-decomposition. We then explain main ideas of our algorithm in Section 3.2. Finally, we give our algorithm together with its analyses in Section 3.3.

3.1 Treewidth for digraphs

We first define the notion of treewidth for an undirected graph, together with its (nice) tree-decomposition. In this paper, the treewidth for a digraph D is defined as the one for the underlying graph of D, where the *underlying graph* $U(D)$ of a digraph D is an undirected graph whose vertex set is $V(D)$ and edge set is $\{\{x, y\} \mid (x, y) \in A(D) \text{ or } (y, x) \in A(D)\}$. For example, Fig. 3(b) is the underlaying graph $U(D)$ of the digraph D in Fig. 3(a).

Let G be an undirected graph with n vertices. We denote by $V(G)$ and $E(G)$ the vertex set and edge set of G, respectively. A *tree-decomposition* of G is a pair $\langle\{X_i \mid i \in V_T\}, T\rangle$, where $T = (V_T, E_T)$ is a rooted tree such that the following four conditions (1)–(4) hold [1]:

(1) each X_i is a subset of $V(G)$;
(2) $\bigcup_{i \in V_T} X_i = V(G)$;
(3) for each edge $\{u, v\} \in E(G)$, there is at least one node $i \in V_T$ such that $u, v \in X_i$; and
(4) for any three nodes $p, q, r \in V_T$, if node q lies on the path between p and r in T, then $X_p \cap X_r \subseteq X_q$.

In particular, a tree-decomposition $\langle\{X_i \mid i \in V_T\}, T\rangle$ of G is called a *nice tree-decomposition* if the following four conditions (5)–(8) hold [1]:

(5) $|V_T| = O(n)$;
(6) every node in V_T has at most two children in T;
(7) if a node $i \in V_T$ has two children l and r, then $X_i = X_l = X_r$; and
(8) if a node $i \in V_T$ has only one child j, then one of the following two conditions (a) and (b) holds:
   (a) $|X_i| = |X_j| + 1$ and $X_i \supset X_j$ (such a node i is called an *introduce* node); and
   (b) $|X_i| = |X_j| - 1$ and $X_i \subset X_j$ (such a node i is called a *forget* node.)

The *width* of $\langle\{X_i \mid i \in V_T\}, T\rangle$ is defined as $\max\{|X_i| - 1 : i \in V_T\}$, and the *treewidth* of G is the minimum k such that G has a tree-decomposition of width k. Figure 3(c) illustrates a nice tree-decomposition $\langle\{X_i \mid i \in V_T\}, T\rangle$ of the graph $U(D)$ in Fig. 3(b) whose treewidth is 3.

In this paper, we say that a digraph $D = (V, A)$ has treewidth k if $U(D)$ is of treewidth k. Since a nice tree-decomposition $\langle\{X_i \mid i \in V_T\}, T\rangle$ of an undirected graph $U(D)$ with bounded treewidth can be found in linear time [1], we may assume without loss of generality that a digraph D and the nice tree-decomposition $\langle\{X_i \mid i \in V_T\}, T\rangle$ of $U(D)$ are both given.

Let D be a digraph, and let $\langle\{X_i \mid i \in V_T\}, T\rangle$ be a nice tree-decomposition of $U(D)$. Each node $i \in V_T$ corresponds to a (directed) subgraph $D_i = (V_i, A_i)$ of D which is induced by the vertices that are contained in X_i and all descendants of i in T. Therefore, if a node $i \in V_T$ has two children l and r in T, then D_i is the union of D_l and D_r which are the subgraphs corresponding to nodes l and r, respectively. Clearly, $D = D_0$ for the root 0 of T. For example, the digraph D_i in Fig. 3(d) is the subgraph of the digraph D in Fig. 3(a) which corresponds to the node $i \in V_T$ in Fig. 3(c).

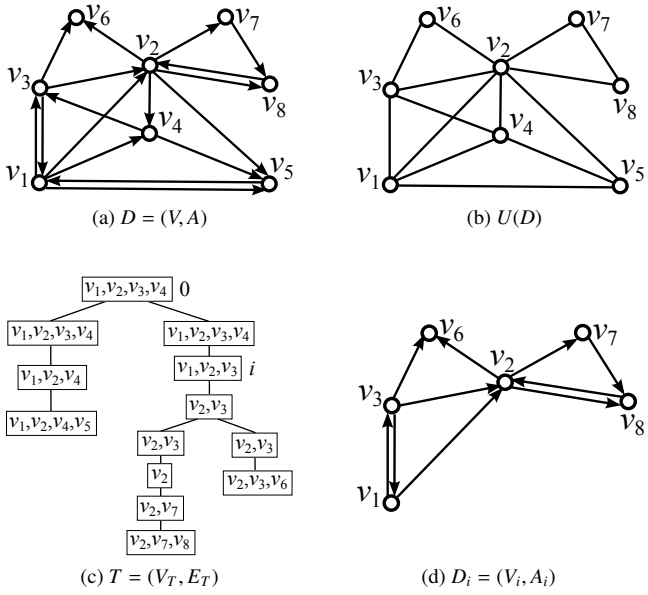


Fig. 3 (a) A digraph D, (b) the underlaying graph $U(D)$ of D, (c) a nice tree-decomposition $\langle\{X_i \mid i \in V_T\}, T\rangle$ of $U(D)$, and (d) the subgraph D_i of D for the node $i \in V_T$.

3.2 Main ideas and definitions

We first introduce some terms. Let $\langle F, S\rangle$ be a caterpillar. Then, each vertex in S is called a *spine vertex*, while each vertex in $V_{\mathrm{L}}(F, S) = V(F) \setminus S$ is called a *leaf vertex*. Therefore, each spine arc in $A_{\mathrm{S}}(F, S)$ joins two spine vertices, and each leaf arc (v, w) in $A_{\mathrm{L}}(F, S)$ joins a spine vertex $v \in S$ and a leaf vertex $w \in V_{\mathrm{L}}(F, S)$; we say that the leaf vertex w is *covered* by the spine vertex v. A spine vertex $v \in S$ is called the *tail* of $\langle F, S\rangle$ if the spine of $\langle F, S\rangle$ starts from v, while a spine vertex $w \in S$ is called the *head* of $\langle F, S\rangle$ if the spine of $\langle F, S\rangle$ ends in w. The head and tail of $\langle F, S\rangle$ are also called the *end-vertices* of $\langle F, S\rangle$.

We now give our main ideas. Let D be a digraph, and let $\langle\{X_i \mid i \in V_T\}, T\rangle$ be a nice tree-decomposition of $U(D)$. Since we wish to find a K-caterpillar with the minimum cost, it suffices to consider K-caterpillars such that all leaf vertices are terminals in K. Consider a K-caterpillar $\langle F, S\rangle$ as a subgraph of D, and consider the subgraph F_i of F which is induced by the vertices in $V(F) \cap V(D_i)$ for a node $i \in V_T$. Then, there are the following three cases to consider, as illustrated in Figs. 4–6 where each terminal is shaded and each spine vertex is depicted by a square.

Case (a): $S \subseteq V(D_i) \setminus X_i$. (See Fig. 4.)

In this case, we claim that D_i contains the whole K-caterpillar $\langle F, S\rangle$, that is, $F_i = F$, as follows. By the definition (4) of tree-decomposition, there is no arc joining a vertex in $V(D_i) \setminus X_i$ and a vertex in $V(D) \setminus V(D_i)$. Then, no spine vertex in $S \subseteq V(D_i) \setminus X_i$ has an arc to a vertex in $V(D) \setminus V(D_i)$. We thus have $V_{\mathrm{L}}(F, S) \subseteq V(D_i)$, and hence $V(F) \subseteq V(D_i)$. Therefore, D_i contains the whole K-caterpillar $\langle F, S\rangle$, as we claimed.

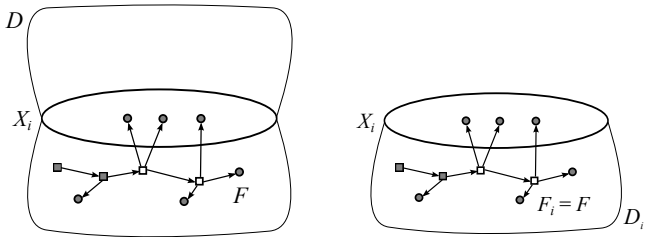


Fig. 4 (a) A K-caterpillar $\langle F, S\rangle$ in D for the case where $S \subseteq V(D_i) \setminus X_i$, and (b) a caterpillar $(\emptyset, \boldsymbol{S})$-forest of D_i, where $\boldsymbol{S} = (1)$.

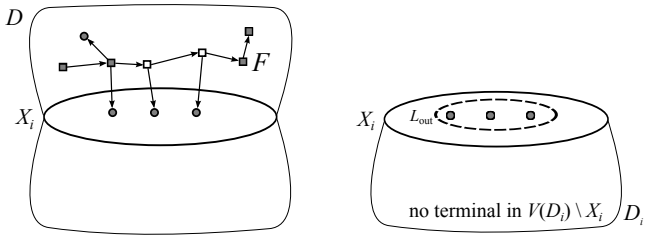


Fig. 5 (a) A K-caterpillar $\langle F, S\rangle$ in D for the case where $S \subseteq V(D) \setminus V(D_i)$, and (b) a caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forest of D_i, where $L_{\text{out}} = K \cap V(D_i)$ and $\boldsymbol{S} = (0)$.

Case (b): $S \subseteq V(D) \setminus V(D_i)$. (See Fig. 5.)

In this case, D_i contains no spine vertex in S, but may contain leaf vertices (terminals) in $V_{\text{L}}(F, S)$ which will be covered by spine vertices in $S \subseteq V(D) \setminus V(D_i)$. Since no spine vertex in $S \subseteq V(D) \setminus V(D_i)$ has an arc to a vertex in $V(D_i) \setminus X_i$, such terminals must be in X_i. (See the three terminals surrounded by the oval L_{out} in Fig. 5(b).)

Case (c): $S \cap X_i \neq \emptyset$. (See Fig. 6.)

In this case, both D_i and $D \setminus D_i$ may contain spine vertices, and hence F_i is not always a (single) caterpillar. However, F_i forms a *caterpillar forest* $\langle F_i, S_i\rangle$, where $S_i = S \cap V(D_i)$; each (weakly) connected component in it is either a caterpillar or a single vertex that was a leaf vertex in $\langle F, S\rangle$. (Note that a single spine vertex is regarded as a caterpillar.) Consider any single leaf vertex in $V_{\text{L}}(F, S) \cap V(D_i)$. Then, similarly as in Case (b) above, it will be covered by some spine vertex in $S \setminus V(D_i)$ and hence it must be in X_i. On the other hand, consider all caterpillars in $\langle F_i, S_i\rangle$. Then, we can naturally order the spine vertices in $S_i = S \cap V(D_i)$ according to the order of the spine vertices of $\langle F, S\rangle$. It is easy to observe that every end-vertex of caterpillars in $\langle F_i, S_i\rangle$ must be in X_i unless it is the end-vertex of $\langle F, S\rangle$. (See the end-vertices v_1, v_2, v_4, v_5, v_6 in Fig. 6(b).)

Motivated by the three Cases (a)–(c) above, we classify caterpillar forests $\langle F', S'\rangle$ in D_i into "caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forests" with respect to the vertices in X_i. A terminal subset $L_{\text{out}} \subseteq K \cap X_i$ represents the terminals that are neither spine vertices in S' nor leaf vertices covered by spine vertices in $S' \subseteq V(D_i)$; and hence every vertex in L_{out} will be a leaf vertex which is covered by some spine vertex outside D_i. A "spine vector" $\boldsymbol{S}$ for X_i represents the spine vertices in $S' \cap X_i$ together with their order and connectivity: a vector $\boldsymbol{S} = (a_0, v_1, a_1, v_2, a_2, \ldots, v_t, a_t)$, $t \geq 0$, is called a

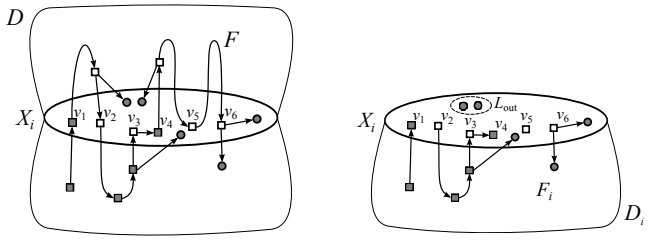


Fig. 6 (a) A K-caterpillar $\langle F, S\rangle$ in D for the case where $S \cap X_i \neq \emptyset$, and (b) a caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forest of D_i, where $\boldsymbol{S} = (1, v_1, 0, v_2, 1, v_3, 1, v_4, 0, v_5, 0, v_6, 0)$.

spine vector for X_i if $a_x \in \{0, 1\}$ for each index x, $0 \leq x \leq t$, and $v_x \in X_i$ for each index x, $1 \leq x \leq t$. We sometimes denote by $V(\boldsymbol{S})$ the set of all vertices in $\boldsymbol{S}$; note that $V(\boldsymbol{S}) = \emptyset$ if $t = 0$. Then, a caterpillar forest $\langle F', S'\rangle$ as a subgraph of D_i is called a *caterpillar* $(L_{\text{out}}, \boldsymbol{S})$*-forest of* D_i if the following three conditions (a)–(c) hold:

(a) if $\boldsymbol{S} = (1)$, then $\langle F', S'\rangle$ is a K-caterpillar such that $S' \cap X_i = \emptyset$;
(b) if $\boldsymbol{S} = (0)$, then $V(F') = L_{\text{out}}$ and F' forms an independent set; and
(c) if $t \geq 1$, then the following six conditions (i)–(vi) hold:
  (i) all terminals in $K \cap V(D_i)$ are contained in $V(F')$;
  (ii) L_{out} forms an independent set in F';
  (iii) if we remove all vertices in L_{out} from F' and add to F' a (dummy) arc from v_x to v_{x+1} for every two vertices $v_x, v_{x+1} \in V(\boldsymbol{S})$ such that $a_x = 0$, $1 \leq x \leq t-1$, then the resulting digraph F'' is a caterpillar $\langle F'', S'\rangle$;
  (iv) $S' \cap X_i = V(\boldsymbol{S})$, and $v_1, v_2, \ldots, v_t$ appear on the spine of $\langle F'', S'\rangle$ in this order;
  (v) if $a_0 = 0$, then v_1 is the tail of $\langle F'', S'\rangle$, otherwise the tail of $\langle F'', S'\rangle$ is in $V(D_i) \setminus X_i$; and
  (vi) if $a_t = 0$, then v_t is the head of $\langle F'', S'\rangle$, otherwise the head of $\langle F'', S'\rangle$ is in $V(D_i) \setminus X_i$.

For example, the caterpillar forest in Fig. 6(b) is a caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forest of D_i for $\boldsymbol{S} = (1, v_1, 0, v_2, 1, v_3, 1, v_4, 0, v_5, 0, v_6, 0)$. We call the head (or the tail) of $\langle F'', S'\rangle$ the *head* (resp., *tail*) of the caterpillar forest $\langle F', S'\rangle$.

Let $\langle F', S'\rangle$ be a caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forest of D_i for some pair $(L_{\text{out}}, \boldsymbol{S})$. If $\boldsymbol{S} = (1)$, then the spine vertices of $\langle F', S'\rangle$ is in $V(D_i) \setminus X_i$ and hence it cannot be extended to the outside of D_i; we thus know that L_{out} must be the empty set and D_i must contains all terminals in K. On the other hand, if $\boldsymbol{S} = (0)$, then $\langle F', S'\rangle$ has no spine vertex and hence we know that all terminals in D_i must be covered by spine vertices outside D_i. Therefore, we say that a pair $(L_{\text{out}}, \boldsymbol{S})$ is *feasible for* X_i if it satisfies the following three conditions (a)–(c):

(a) if $\boldsymbol{S} = (1)$, then $L_{\text{out}} = \emptyset$ and $K \subseteq V(D_i)$;
(b) if $\boldsymbol{S} = (0)$, then $L_{\text{out}} = K \cap V(D_i)$; and
(c) $L_{\text{out}} \cap V(\boldsymbol{S}) = \emptyset$, and each vertex in $V(\boldsymbol{S})$ appears exactly once in $\boldsymbol{S}$.

Then, it suffices to consider caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forests of D_i only for feasible pairs $(L_{\text{out}}, \boldsymbol{S})$ for X_i.

We finally define a value $f(i; L_{\text{out}}, \boldsymbol{S})$ for a node $i \in V_T$ and a pair $(L_{\text{out}}, \boldsymbol{S})$, which will be computed by our algorithm, as follows:

$$f(i; L_{\text{out}}, \boldsymbol{S}) = \min\{c(F', S') \mid \langle F', S' \rangle \text{ is a caterpillar } (L_{\text{out}}, \boldsymbol{S})\text{-forest of } D_i\},$$

where $c(F', S')$ is the cost of a caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forest $\langle F', S' \rangle$, that is, the total cost of all caterpillars in $\langle F', S' \rangle$. Let $f(i; L_{\text{out}}, \boldsymbol{S}) = +\infty$ if D_i has no caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forest or $(L_{\text{out}}, \boldsymbol{S})$ is not feasible for X_i.

Our algorithm computes $f(i; L_{\text{out}}, \boldsymbol{S})$ for each node $i \in V_T$ and all feasible pairs $(L_{\text{out}}, \boldsymbol{S})$ for X_i, from the leaves of T to the root of T, by means of dynamic programming. Then, since $D_0 = D$ for the root 0 of T, one can compute the minimum cost $c(D, K)$ of a K-caterpillar in a given digraph D, as follows:

$$c(D, K) = \min f(0; \emptyset, \boldsymbol{S}), \tag{1}$$

where the minimum above is taken over all spine vectors $\boldsymbol{S} = (a_0, v_1, a_1, \ldots, v_t, a_t)$, $0 \le t \le |X_0|$, for X_0 such that $a_x = 1$ for all x, $1 \le x \le t-1$. Note that $c(D, K) = +\infty$ if D has no K-caterpillar.

3.3 Algorithm

In this subsection, we explain how to compute $f(i; L_{\text{out}}, \boldsymbol{S})$ for each node $i \in V_T$ and all feasible pairs $(L_{\text{out}}, \boldsymbol{S})$ for X_i, from the leaves of T to the root of T.

We first compute $f(i; L_{\text{out}}, \boldsymbol{S})$ for each leaf i of T and each feasible pair $(L_{\text{out}}, \boldsymbol{S})$ for X_i, as follows: we enumerate all possible caterpillar forests in D_i, and check whether each of them is a caterpillar $(L_{\text{out}}, \boldsymbol{S})$-forest of D_i. (The running time will be estimated later.)

We then compute $f(i; L_{\text{out}}, \boldsymbol{S})$ for each internal node i in T and each feasible pair $(L_{\text{out}}, \boldsymbol{S})$ for X_i. For notational convenience, let $L^i_{\text{out}} = L_{\text{out}}$ and $\boldsymbol{S}^i = \boldsymbol{S} = (a^i_0, v^i_1, a^i_1, \ldots, v^i_{t_i}, a^i_{t_i})$. Since $\langle \{X_i \mid i \in V_T\}, T \rangle$ is a nice tree-decomposition of $U(D)$, there are three cases to consider, that is, i has two children, is a forget node, and is an introduce node.

[The node i has two children l and r]

In this case, a caterpillar $(L^i_{\text{out}}, \boldsymbol{S}^i)$-forest $\langle F_i, S_i \rangle$ of D_i can be obtained by merging a caterpillar $(L^l_{\text{out}}, \boldsymbol{S}^l)$-forest $\langle F_l, S_l \rangle$ of D_l with a caterpillar $(L^r_{\text{out}}, \boldsymbol{S}^r)$-forest $\langle F_r, S_r \rangle$ of D_r, where $\boldsymbol{S}^l = (a^l_0, v^l_1, a^l_1, \ldots, v^l_{t_l}, a^l_{t_l})$ and $\boldsymbol{S}^r = (a^r_0, v^r_1, a^r_1, \ldots, v^r_{t_r}, a^r_{t_r})$, such that

(1) the union of the spines of $\langle F_l, S_l \rangle$ and $\langle F_r, S_r \rangle$ forms directed paths, each of which is the spine of a caterpillar in $\langle F_i, S_i \rangle$; and
(2) each terminal in $(K \cap V(D_i)) \setminus L^i_{\text{out}}$ is covered by exactly one of $\langle F_l, S_l \rangle$ and $\langle F_r, S_r \rangle$.

Since $X_l = X_r = X_i$ and there is no arc joining a vertex in $V(D_l) \setminus X_l$ and a vertex $V(D_r) \setminus X_r$, the condition (1) above can be rephrased as the following three conditions (a)–(c):

(a) $t_l = t_r = t_i$;
(b) $v^l_x = v^r_x = v^i_x$ for all indices x, $1 \le x \le t_i$; and
(c) $a^l_x + a^r_x = a^i_x$ for all indices x, $0 \le x \le t_i$.

The conditions (a) and (b) above ensure that both $\langle F_l, S_l \rangle$ and $\langle F_r, S_r \rangle$ have the same spine vertices as $\langle F_i, S_i \rangle$ in $X_i = X_l = X_r$; furthermore, they appear in the same order as in $\boldsymbol{S}^i$. The condition (c) above ensures that the union of the spines of $\langle F_l, S_l \rangle$ and $\langle F_r, S_r \rangle$ forms directed paths: notice that, if there is an index x, $1 \le x \le t_i - 1$, such that $a^l_x = a^r_x = 1$, then both $\langle F_l, S_l \rangle$ and $\langle F_r, S_r \rangle$ have a directed path from v_x to v_{x+1} and hence $\langle F_i, S_i \rangle$ would contain a cycle or the cost of the same arc (v_x, v_{x+1}) would be counted twice; furthermore, if both $a^l_0 = a^r_0 = 1$ or $a^l_{t_l} = a^r_{t_r} = 1$, then the spine of $\langle F_i, S_i \rangle$ would be "two-forked."

Similarly, the condition (2) above can be rephrased as the following two conditions (d) and (e):

(d) $L^l_{\text{out}} \cap L^r_{\text{out}} = L^i_{\text{out}}$; and
(e) $v \in L^l_{\text{out}}$ or $v \in L^r_{\text{out}}$ hold for each terminal $v \in (K \cap X_i) \setminus V(\boldsymbol{S}^i)$.

The condition (d) above ensures that any vertex in L^i_{out} is covered by neither a spine vertex in S_l nor a spine vertex in S_r. The condition (e) above ensures that each terminal in $(K \cap V(D_i)) \setminus L^i_{\text{out}}$ is covered by exactly one of $\langle F_l, S_l \rangle$ and $\langle F_r, S_r \rangle$; note that, if $v \notin L^l_{\text{out}}$ and $v \notin L^r_{\text{out}}$ hold, then the vertex v is covered by some spaine vertex in S_l and S_r at the same time.

In this way, $f(i; L^i_{\text{out}}, \boldsymbol{S}^i)$ can be computed, as follows:

$$f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = \min\{f(l; L^l_{\text{out}}, \boldsymbol{S}^l) + f(r; L^r_{\text{out}}, \boldsymbol{S}^r)\},$$

where the minimum above is taken over all feasible pairs $(L^l_{\text{out}}, \boldsymbol{S}^l)$ for X_l and $(L^r_{\text{out}}, \boldsymbol{S}^r)$ for X_r satisfying the following five conditions (a)–(e):

(a) $t_l = t_r = t_i$;
(b) $v^l_x = v^r_x = v^i_x$ for all indices x, $1 \le x \le t_i$;
(c) $a^l_x + a^r_x = a^i_x$ for all indices x, $0 \le x \le t_i$;
(d) $L^l_{\text{out}} \cap L^r_{\text{out}} = L^i_{\text{out}}$; and
(e) $v \in L^l_{\text{out}}$ or $v \in L^r_{\text{out}}$ hold for each terminal $v \in (K \cap X_i) \setminus V(\boldsymbol{S}^i)$.

It should be noted that the update formula above correctly computes the cases where $\boldsymbol{S}^i = (0)$ or $\boldsymbol{S}^i = (1)$, too.

[The node i is a forget node]

In this case, the node i has exactly one child j in T such that $|X_i| = |X_j| - 1$ and $X_i \subset X_j$. Let v' be the vertex in $X_j \setminus X_i$. Since $D_j = D_i$, a caterpillar $(L^i_{\text{out}}, \boldsymbol{S}^i)$-forest $\langle F_i, S_i \rangle$ of D_i is a caterpillar $(L^j_{\text{out}}, \boldsymbol{S}^j)$-forest $\langle F_j, S_j \rangle$ of D_j for some feasible pair $(L^j_{\text{out}}, \boldsymbol{S}^j)$ for X_j. Then, there are the following three cases (1)–(3) to consider:

(1) v' is not a terminal, and is not contained in $V(F_j)$;
(2) v' is a terminal, and is contained in $\langle F_j, S_j \rangle$ as a leaf vertex; and
(3) v' is a spine vertex in $\langle F_j, S_j \rangle$, that is, $v' \in S_j$.

We thus define the three values $f^1(i; L^i_{\text{out}}, \boldsymbol{S}^i)$, $f^2(i; L^i_{\text{out}}, \boldsymbol{S}^i)$ and $f^3(i; L^i_{\text{out}}, \boldsymbol{S}^i)$ for the three cases above, and take the minimum one among them. Notice that $v' \notin L^j_{\text{out}}$, because $v' \notin X_i$ and hence v' cannot be covered by any vertex in $V(D) \setminus V(D_j)$.

(1) v' is not a terminal, and is not contained in $V(F_j)$.

In this sub-case, since $X_i = X_j \setminus \{v'\}$, we have $(L^j_{\text{out}}, \boldsymbol{S}^j) = (L^i_{\text{out}}, \boldsymbol{S}^i)$. Therefore, we let $f^1(i; L^i_{\text{out}}, \boldsymbol{S}^i) = f(j; L^i_{\text{out}}, \boldsymbol{S}^i)$.

(2) v' is a terminal, and is contained in $\langle F_j, S_j \rangle$ as a leaf vertex.

In this sub-case, since $v' \notin L^j_{\text{out}} \cup V(\boldsymbol{S}^j)$, we have $(L^j_{\text{out}}, \boldsymbol{S}^j) = (L^i_{\text{out}}, \boldsymbol{S}^i)$. Therefore, we let $f^2(i; L^i_{\text{out}}, \boldsymbol{S}^i) = f(j; L^i_{\text{out}}, \boldsymbol{S}^i)$.

(3) v' is a spine vertex in $\langle F_j, S_j\rangle$.

In this sub-case, since $v' \notin L^j_{\text{out}}$, we have $L^j_{\text{out}} = L^i_{\text{out}}$. Since $v' \in S_j$ and $v' \in X_j$, we have $t_j = t_i + 1$. Therefore, we let

$$f^3(i; L^i_{\text{out}}, \boldsymbol{S}^i) = \min f(j; L^i_{\text{out}}, \boldsymbol{S}^j),$$

where the minimum is taken over all spine vectors $\boldsymbol{S}^j$ for X_j such that

(a) if $a^i_x = 1$ for some index x, $1 \le x \le t_i - 1$, then $\boldsymbol{S}^j = (a^i_0, v^i_1, \ldots, v^i_x, 1, v', 1, v^i_{x+1}, \ldots, v^i_{t_i}, a^i_{t_i})$;
(b) if $a^i_{t_i} = 1$, then $\boldsymbol{S}^j = (a^i_0, v^i_1, \ldots, v^i_{t_i}, a^i_{t_i}, v', a^j_{t_j})$ with $a^j_{t_j} \in \{0, 1\}$; and
(c) if $a^i_0 = 1$, then $\boldsymbol{S}^j = (a^j_0, v', a^i_0, v^i_1, \ldots, v^i_{t_i}, a^i_{t_i})$ with $a^j_0 \in \{0, 1\}$.

Therefore, $f(i; L^i_{\text{out}}, \boldsymbol{S}^i)$ can be computed, as follows:

$$f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = \min\{f^1(i; L^i_{\text{out}}, \boldsymbol{S}^i),\ f^2(i; L^i_{\text{out}}, \boldsymbol{S}^i),\ f^3(i; L^i_{\text{out}}, \boldsymbol{S}^i)\}.$$

[The node i is an introduce node]

In this case, the node i has exactly one child j in T such that $|X_i| = |X_j| + 1$ and $X_i \supset X_j$. Let v' be the vertex in $X_i \setminus X_j$. Since v' is introduced by X_i, every arc in D_i from/to v' is to/from a vertex in X_i. For notational convenience, we denote simply by $c_{\text{S}}(v, w)$ and $c_{\text{L}}(v, w)$ the two costs of (v, w) with $v, w \in V$, and let $c_{\text{S}}(u, v) = c_{\text{L}}(u, v) = +\infty$ if $(u, v) \notin A$. According to v' and $(L^i_{\text{out}}, \boldsymbol{S}^i)$, we classify caterpillar $(L^i_{\text{out}}, \boldsymbol{S}^i)$-forests $\langle F_i, S_i\rangle$ of D_i into the following six types:

(1) v' is not contained in $V(F_i)$;
(2) v' is contained in $V(F_i) \setminus S_i$;
(3) v' is contained in S_i which is neither the head nor the tail of $\langle F_i, S_i\rangle$;
(4) v' is the head of $\langle F_i, S_i\rangle$;
(5) v' is the tail of $\langle F_i, S_i\rangle$; and
(6) v' is both head and tail of $\langle F_i, S_i\rangle$, that is, $S_i = \{v'\}$.

Notice that, from v' and $(L^i_{\text{out}}, \boldsymbol{S}^i)$, we can distinguish which type of caterpillar $(L^i_{\text{out}}, \boldsymbol{S}^i)$-forests are considered, and hence $f(i; L^i_{\text{out}}, \boldsymbol{S}^i)$ can be computed, as follows.

(1) if $v' \notin V(\boldsymbol{S}^i)$ and $v' \notin K$

This sub-case corresponds to the type (1): v' is not contained in $V(F_i)$. Therefore, $\langle F_i, S_i\rangle$ is a caterpillar $(L^i_{\text{out}}, \boldsymbol{S}^i)$-forest of D_j. We thus have

$$f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = f(j; L^i_{\text{out}}, \boldsymbol{S}^i).$$

(2) if $v' \notin V(\boldsymbol{S}^i)$ and $v' \in K$

This sub-case corresponds to the type (2): v' is contained in $V(F_i) \setminus S_i$. If $v' \in L^i_{\text{out}}$, then v' will be covered by a spine vertex in $D \setminus D_i$; and hence $V(F_i) \cap V(D_j)$ induces a caterpillar $(L^i_{\text{out}} \setminus \{v'\}, \boldsymbol{S}^i)$-forest of D_j. If $v' \notin L^i_{\text{out}}$, then v' must be covered by a spine vertex in X_i; and hence $V(F_i) \cap V(D_j)$ induces a caterpillar $(L^i_{\text{out}}, \boldsymbol{S}^i)$-forest of D_j. Note that, if $\boldsymbol{S}^i = (1)$, then X_i contains no spine vertex and hence v' cannot be covered by any spine vertex. We thus have

$$f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = \begin{cases} +\infty & \text{if } \boldsymbol{S}^i = (1); \\ f(j; L^i_{\text{out}} \setminus \{v'\}, \boldsymbol{S}^i) & \text{if } v' \in L^i_{\text{out}}; \\ f(j; L^i_{\text{out}}, \boldsymbol{S}^i) & \\ \quad + \min\{c_{\text{L}}(v^i_x, v') \mid 1 \le x \le t_i\} & \text{otherwise.} \end{cases}$$

(3) if $v' \in V(\boldsymbol{S}^i)$, $v^i_1 \neq v'$ and $v^i_{t_i} \neq v'$

This sub-case corresponds to the type (3): v' is contained in S_i which is neither the head nor the tail of $\langle F_i, S_i\rangle$. Let $v' = v^i_{x+1}$ in $\boldsymbol{S}^i$. Then, $V(F_i) \cap V(D_j)$ induces a caterpillar $(L^j_{\text{out}}, \boldsymbol{S}^j)$-forest of D_j, where $\boldsymbol{S}^j = (a^i_0, v^i_1, \ldots, v^i_x, 0, v^i_{x+2}, \ldots, v^i_{t_i}, a^i_{t_i})$ and L^j_{out} is some subset of $K \cap X_j$ such that $L^i_{\text{out}} \subseteq L^j_{\text{out}}$. We thus have

$$f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = \min\Big\{f(j; L^j_{\text{out}}, \boldsymbol{S}^j) + a^i_x \cdot c_{\text{S}}(v^i_x, v') + a^i_{x+1} \cdot c_{\text{S}}(v', v^i_{x+2}) + \sum_{w \in L^j_{\text{out}} \setminus L^i_{\text{out}}} c_{\text{L}}(v', w)\Big\},$$

where the minimum above is taken over all subsets $L^j_{\text{out}} \subseteq (K \cap X_j) \setminus V(\boldsymbol{S}^j)$ such that $L^i_{\text{out}} \subseteq L^j_{\text{out}}$.

(4) if $v' \in V(\boldsymbol{S}^i)$, $v^i_1 \neq v'$ and $v^i_{t_i} = v'$

This sub-case corresponds to the type (4): v' is the head of $\langle F_i, S_i\rangle$. Since $v' \in X_i$, we have $a^i_{t_i} = 0$ and hence let $f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = +\infty$ if $a^i_{t_i} = 1$. If $a^i_{t_i} = 0$, then $V(F_i) \cap V(D_j)$ induces a caterpillar $(L^j_{\text{out}}, \boldsymbol{S}^j)$-forest of D_j, where $\boldsymbol{S}^j = (a^i_0, v^i_1, a^i_1, \ldots, a^i_{t_i-2}, v^i_{t_i-1}, 0)$ and L^j_{out} is some subset of $K \cap X_j$ such that $L^i_{\text{out}} \subseteq L^j_{\text{out}}$. We thus have

$$f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = \min\Big\{f(j; L^j_{\text{out}}, \boldsymbol{S}^j) + a^i_{t_i-1} \cdot c_{\text{S}}(v^i_{t_i-1}, v') + \sum_{w \in L^j_{\text{out}} \setminus L^i_{\text{out}}} c_{\text{L}}(v', w)\Big\},$$

where the minimum above is taken over all subsets $L^j_{\text{out}} \subseteq (K \cap X_j) \setminus V(\boldsymbol{S}^j)$ such that $L^i_{\text{out}} \subseteq L^j_{\text{out}}$.

(5) if $v' \in V(\boldsymbol{S}^i)$, $v^i_1 = v'$ and $v^i_{t_i} \neq v'$

This sub-case corresponds to the type (5): v' is the tail of $\langle F_i, S_i\rangle$. Since $v' \in X_i$, we have $a^i_0 = 0$ and hence let $f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = +\infty$ if $a^i_0 = 1$. If $a^i_0 = 0$, then $V(F_i) \cap V(D_j)$ induces a caterpillar $(L^j_{\text{out}}, \boldsymbol{S}^j)$-forest of D_j, where $\boldsymbol{S}^j = (0, v^i_2, a^i_2, \ldots, v^i_{t_i}, a^i_{t_i})$ and L^j_{out} is some subset of $K \cap X_j$ such that $L^i_{\text{out}} \subseteq L^j_{\text{out}}$. We thus have

$$f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = \min\Big\{f(j; L^j_{\text{out}}, \boldsymbol{S}^j) + a^i_1 \cdot c_{\text{S}}(v', v^i_2) + \sum_{w \in L^j_{\text{out}} \setminus L^i_{\text{out}}} c_{\text{L}}(v', w)\Big\},$$

where the minimum above is taken over all subsets $L^j_{\text{out}} \subseteq (K \cap X_j) \setminus V(\boldsymbol{S}^j)$ such that $L^i_{\text{out}} \subseteq L^j_{\text{out}}$.

(6) if $V(\boldsymbol{S}^i) = \{v'\}$, that is, $v^i_1 = v^i_{t_i} = v'$

This sub-case corresponds to the type (6): v' is both head and tail of $\langle F_i, S_i\rangle$, that is, $S_i = \{v'\}$. Since $v' \in X_i$, let $f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = +\infty$ if $a^i_0 = 1$ or $a^i_1 = 1$. If $a^i_0 = a^i_1 = 0$ and hence $\boldsymbol{S}^i = (0, v', 0)$, then $V(F_i) \cap V(D_j)$ induces a caterpillar $(L^j_{\text{out}}, (0))$-forest of D_j for $L^j_{\text{out}} = K \cap V(D_j)$. Note that, since $v' \in X_i$ is only the spine vertex in $\langle F_i, S_i\rangle$, the subgraph D_j does not contain any spine vertex. We thus have

$$f(i; L^i_{\text{out}}, \boldsymbol{S}^i) = f(j; K \cap V(D_j), (0)) + \sum_{w \in (K \cap V(D_j)) \setminus L^i_{\text{out}}} c_{\text{L}}(v', w).$$

Running time.

Remember that $|X_i| \leq k+1$ for each node $i \in V_T$, where k is the treewidth of D. Then, there are at most $\binom{k+1}{t} \cdot t! \cdot 2^{t+1}$ spine vectors $\boldsymbol{S} = (a_0, v_1, a_1, \ldots, v_t, a_t)$ for each $t \geq 0$. Thus, the number of all feasible pairs $(L_{\text{out}}, \boldsymbol{S})$ for X_i can be bounded by

$$\sum_{t=0}^{k+1} \binom{k+1}{t} \cdot t! \cdot 2^{t+1} \cdot 2^{k+1-t} \leq 2^{2k+3}(k+1)^{k+1} = O(1).$$

Since the subgraph D_i corresponding to a leaf i of T contains at most $k+1$ vertices, our brute-force algorithm for a leaf can be done in $O(1)$ time. Therefore, we can compute $f(i; L_{\text{out}}, \boldsymbol{S})$ in $O(1)$ time for each leaf i and all feasible pairs $(L_{\text{out}}, \boldsymbol{S})$ for X_i. By the definition (5) of a nice tree-decomposition, T has at most $O(n)$ leaves, and hence $f(i; L_{\text{out}}, \boldsymbol{S})$ can be computed in linear time for all leaves i.

Similarly, for each internal node i of T, each of the update formulas above can be computed in $O(1)$ time. Since there are $O(n)$ nodes in T, for the root 0 of T, we can compute $f(0; L_{\text{out}}, \boldsymbol{S})$ for all feasible pairs $(L_{\text{out}}, \boldsymbol{S})$ for X_0 in $O(n)$ time. Then, by Eq. (1) we can compute the minimum cost $c(D, K)$ in $O(1)$ time.

In this way, our algorithm runs in linear time in total.

This completes the proof of Theorem 2. □

4. Conclusion

In this paper, we first showed that the minimum caterpillar problem is NP-hard even for digraphs with three terminals. We then gave a linear-time algorithm to solve the problem for digraphs with bounded treewidth.

References

[1] N. Betzler, R. Niedermeier, and J. Uhlmann.: *Tree decompositions of graphs: saving memory in dynamic programming*, Discrete Optimization, vol. 3, pp. 220–229, 2006.

[2] B. Courcelle.: *Graph rewriting: an algebraic and logic approach*, Handbook of Theoretical Computer Science (vol. B), pp. 193–242, MIT Press, 1990.

[3] M. J. Dinneen and M. Khosravani.: *A linear time algorithm for the minimum spanning caterpillar problem for bounded treewidth graphs*, In Proc. of SIROCCO 2010, pp. 237–246, 2010.

[4] M. J. Dinneen and M. Khosravani.: *Hardness of approximation and integer programming frameworks for searching for caterpillar trees*, In Proc. of CATS 2011, pp. 145–150, 2011.

[5] S. Forutne, J. Hopcroft, and J. Wyllie.: *The directed subgraph homeomorphism problem*, Theoretical Computer Science, vol. 10, pp. 111–121, 1980.

[6] T. Johnson, N. Robertson, P. D. Seymour, and R. Thomas.: *Directed tree-width*, Journal of Combinatorial Theory, Series B, vol. 82, pp. 138–154, 2001.

[7] M. Lampis.: *Algorithmic meta-theorems for restrictions of treewidth*, Algorithmica, vol. 64, pp. 19–37, 2012.

[8] L. Simonetti, Y. Frota, and C. C. de Souza.: *An exact method for the minimum caterpillar spanning problem*, In Proc. of CTW 2009, pp. 48–51, 2009.