

# Dynamic Distance Hereditary Graphs Using Split Decomposition\*

Emeric Gioan

Christophe Paul †

LIRMM Research Report 07007

March 19, 2007

## Abstract

The problem of maintaining a representation of a dynamic graph as long as a certain property is satisfied, has recently been considered for a number of properties. This paper presents an optimal algorithm for this problem on vertex-dynamic connected distance hereditary graphs: both vertex insertion and deletion have complexity  $O(d)$ , where  $d$  is the degree of the vertex involved in the modification. Our vertex-dynamic algorithm is competitive with the existing linear time recognition algorithms of distance hereditary graphs, and is also simpler. To achieve this, we revisit the split decomposition by which distance hereditary graphs are known to be completely decomposable. We propose a formulation of this decomposition in terms of graph-labelled trees. Doing so, we are also able to derive an intersection model for distance hereditary graphs, which answers an open problem.

---

\*Research supported by the French ANR project “*Graph Decompositions and Algorithms (GRAAL)*”

†Research conducted while C. Paul was on Sabbatical at School of Computer Science, McGill University, Montréal, Canada

# 1 Introduction

A graph is *dynamic* if it is submitted to a series of (edge or vertex) modifications. The area of dynamic graphs has been shown of particular interest for practical settings (e.g. networking, peer-to-peer) as well as for its theoretical aspects. The *dynamic representation problem* asks for the maintenance of a representation of a dynamic graph under a series of modification as long as certain property  $\Pi$  is satisfied. Whenever the property  $\Pi$  fails, a certificate may be returned by the algorithm. In the last few years, dynamic representation algorithms have been obtained for  $\Pi$  being the membership to some restricted graph families such as chordal graphs [16], proper interval graphs [14], cographs [18], directed cographs [4], permutation graphs [5]. In this context the only known result on distance hereditary graphs is a recent  $O(1)$  time edge-only dynamic algorithm [3]. However the method of [3] does not allow vertex insertion or deletion.

In this paper, we propose a  $O(d)$  algorithm for the vertex insertion and vertex deletion (with  $d$  being the degree of the vertex involved). Our technique is based on the split decomposition, also called 1-join decomposition, which was introduced by Cunningham [6]. The family of completely decomposable graphs for the split decomposition is known to be the distance hereditary graphs [1, 13] which thereby leads to a canonical decomposition. However most of the known algorithms operating on distance hereditary graphs are either based on a heavy breadth first search layering characterization [1], or on some ad-hoc (rooted) tree-decompositions [2, 15, 2, 20].

We represent the split decomposition of a graph  $G$  by a (unrooted) graph-labelled tree, called *split tree*, the nodes of which are in bijection with a family  $\mathcal{F}$  of graphs. The graph  $G$  can be retrieved from as an *accessibility graph* of the graph-labelled tree. Using this formalism, we can derive two new characterizations of distance hereditary graphs. The first one is an intersection model [17] for the family of distance hereditary graphs. The existence of such a model was known, but up to our knowledge, the model itself was still not discovered [19]. The second characterization is an incremental one that specifies the condition under which a distance hereditary graph augmented by a new vertex remains distance hereditary. Our dynamic algorithm relies on this latter characterization and simply maintains the split tree under vertex modifications. As a by-product, let us notice that our  $O(d)$  vertex insertion algorithm is not only competitive with the already known static linear time distance hereditary graph recognition algorithms [13, 9, 2], but is also simpler.

The paper is organized as follows. We first present the split decomposition and formalize the split tree decomposition. Section 3 deals with theoretical characterizations of distance hereditary graphs. And Section 4 is devoted to the vertex fully-dynamic algorithm.

## 2 Split decomposition and graph-labelled trees

In this preliminary section, we recall the definition and main result on split decomposition, while introducing a new formal representation.

Any graph  $G$  we consider is simple and loopless. Its vertex set is denoted  $V(G)$  and its edge set  $E(G)$ . For a subset  $S \subseteq V(G)$ ,  $G[S]$  is the subgraph of  $G$  induced by  $S$ . If  $T$  is a tree and  $S$  a subset of leaves of  $T$ , then  $T(S)$  is the smallest subtree of  $T$  spanning the leaves of  $S$ . If  $x$  is a vertex of  $G$  then  $G-x = G[V(G) - \{x\}]$ . Similarly if  $x \notin V(G)$ ,  $G+(x, S)$  is the graph  $G$  augmented by the new vertex  $x$  adjacent to some specified neighborhood  $S \subseteq V(G)$ . We denote  $N(x)$  the neighbourhood of a vertex  $x$ . The neighborhood of a set  $S \subseteq V(G)$  is  $N(S) = \{x \notin S \mid \exists y \in S, xy \in E(G)\}$ . The *clique* is the complete graph and the star is the complete bipartite graph  $K_{1,n}$ . The universal

vertex of the star is called its *centre* and the degree one vertices its *extremities*.

**Definition 2.1** [6] *A split of a graph  $G$  is a bipartition  $(V_1, V_2)$  of  $V(G)$  such that*

1.  $|V_1| \geq 2$  and  $|V_2| \geq 2$ ; and
2. every vertex of  $N(V_1)$  is adjacent to every vertex of  $N(V_2)$ .

The clique  $K_n$  ( $n \geq 4$ ) and the star  $K_{1,n}$  ( $n \geq 3$ ) are called *degenerate*, since any bipartition respecting the first criteria of Definition 2.1 is a split. A graph with no split that is not degenerate is called *prime*.

The split decomposition of a graph, as it is studied in [6], consists in finding a split  $(V_1, V_2)$ , decompose the graph  $G$  into two graphs  $G_1 = G[V_1 \cup \{x_1\}]$ , with  $x_1 \in N(V_1)$  and  $G_2 = G[V_2 \cup \{x_2\}]$  with  $x_2 \in N(V_2)$ ,  $x_1$  and  $x_2$  being called the *marker vertices*, and then continue recursively on graphs  $G_1$  and  $G_2$  as long as these resulting graphs are not prime nor degenerate.

**Theorem 2.2** [6] *Each connected graph has a unique split decomposition into prime, clique or star components with a minimum number of components.*

In the seminal paper [6], Cunningham presents the idea of a tree decomposition. But its main result stating the uniqueness of lastly resulting graphs in a split decomposition focuses on the set of resulting graphs more than on the structure linking them together. It will be reformulated later. At first, we introduce a more efficient formalism for our purpose.

**Definition 2.3** *A graph-labelled tree  $(T, \mathcal{F})$  is a tree in which any node  $v$  of degree  $k$  is labelled by a graph  $G_v \in \mathcal{F}$  on  $k$  vertices such that there is a bijection  $\rho_v$  from the tree-edges incident to  $v$  to the vertices of  $G_v$ .*

We call *nodes* the internal vertices of a tree, and *leaves* the other ones. Let  $(T, \mathcal{F})$  be a graph-labelled tree and  $l$  be a leaf of  $T$ . A node or a leaf  $u$  different from  $l$  is  *$l$ -accessible* if for any tree-edges  $e = uv$  and  $e' = vw'$  on the  $l, u$ -path in  $T$ , we have  $\rho_v(e)\rho_v(e') \in E(G_v)$ . By convention, the unique neighbor of  $l$  in  $T$  is also  $l$ -accessible. See Figure 1 for an example.

**Definition 2.4** *The accessibility graph  $G(T, \mathcal{F})$  of a graph-labelled tree  $(T, \mathcal{F})$  has the leaves of  $T$  as vertices, and there is an edge between  $x$  and  $y$  if and only if  $y$  is  $x$ -accessible.*

Figure 1 shows a graph-labelled tree and its accessibility graph.

**Lemma 2.5** *Let  $(T, \mathcal{F})$  be a graph-labelled tree and  $T_1, T_2$  be the subtrees of  $T - e$  where  $e$  is a tree-edge non-incident to a leaf. Then the bipartition  $(L_1, L_2)$  of the leaves of  $T$ , with  $L_i$  being the leaf set of  $T_i$ , defines a split in the graph  $G(T, \mathcal{F})$ .*

*Proof:* Let  $t_1$  and  $t_2$  be the extremities of  $e$  and let  $l_1$  and  $l_2$  be leaves of  $L_1$  and  $L_2$  respectively. By definition of  $G(T, \mathcal{F})$ ,  $l_1$  and  $l_2$  are adjacent if and only if  $t_2$  is  $l_1$ -accessible and  $t_1$  is  $l_2$ -accessible. It follows that  $(L_1, L_2)$  defines a split of  $G(T, \mathcal{F})$ .  $\square$

We can naturally define the *split* and the converse *join* operations on a graph-labelled tree  $(T, \mathcal{F})$  as follows (see Figure 2):

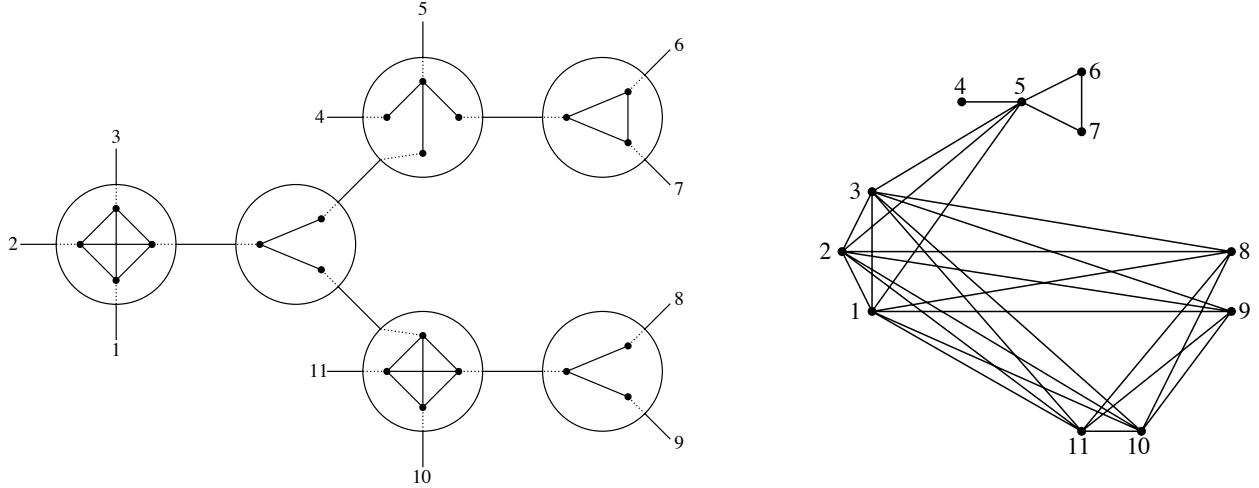


Figure 1: A graph-labelled tree and its accessibility graph.

- *Split of  $(T, \mathcal{F})$* : Let  $v$  be a node of  $T$  whose graph  $G_v$  has a split  $(A, B)$ . Let  $G_A$  and  $G_B$  be the subgraphs resulting from the split  $(A, B)$  and  $a, b$  be the respective marker vertices. Splitting the node  $v$  consists in substituting  $v$  by two adjacent nodes  $v_A$  and  $v_B$  respectively labelled by  $G_A$  and  $G_B$  such that for any  $x \in V(G_A)$  different from  $a$ ,  $\rho_{v_A}(x) = \rho_v(x)$  and  $\rho_{v_A}(a) = v_A v_B$  (similarly for any  $x \in V(G_B)$  different from  $b$ ,  $\rho_{v_B}(x) = \rho_v(x)$  and  $\rho_{v_B}(b) = v_A v_B$ ).
- *Join of  $(T, \mathcal{F})$* : Let  $uv$  be a tree-edge of  $T$ . Then joining the nodes  $u$  and  $v$  consists in substituting them by a single node  $w$  labelled by  $G_w$  the accessibility graph of the tree with only edge  $u, v$  and respective labels  $G_u$  and  $G_v$ . For any vertex  $x$  of  $G_w$ ,  $\rho_w(x) = \rho_v(x)$  or  $\rho_w(x) = \rho_u(x)$  depending on which graph  $x$  belonged.

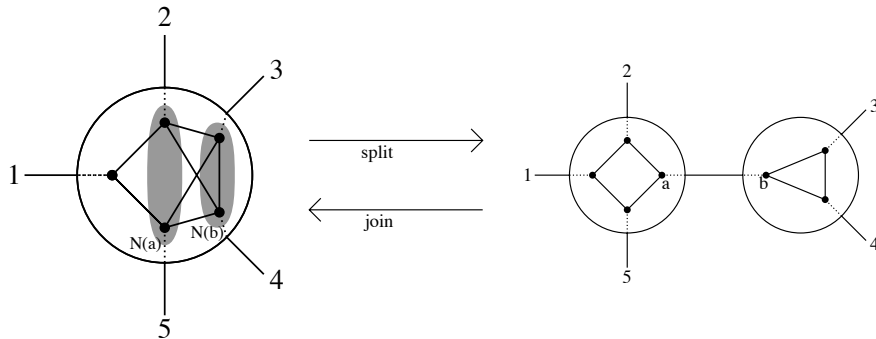


Figure 2: The split and the join operations on a graph-labelled tree.

Observe that if  $(T, \mathcal{F})$  is obtained from  $(T', \mathcal{F}')$  by a join or a split operation, then it follows from the definitions that  $G(T, \mathcal{F}) = G(T', \mathcal{F}')$ .

Among the join operations, let us distinguish: the *clique-join* which operating on two neighboring nodes labelled by cliques and the *star-join* concerning star-labelled neighboring nodes  $u, v$  such that the tree-edge  $uv$  links the root of one star to a leaf of the other.

**Definition 2.6** A graph-labelled tree  $(T, \mathcal{F})$  is reduced if neither clique-join nor star-join can be applied.

With the above formalism, the main theorem in [6] stating the uniqueness of resulting graphs in a split decomposition can now be reformulated the following way.

**Theorem 2.2 (Cunningham's Theorem reformulated)** For any connected graph  $G$ , there exists a unique reduced graph-labelled tree  $(T, \mathcal{F})$  such that  $G = G(T, \mathcal{F})$  and any graph of  $\mathcal{F}$  is prime or degenerate.

For a connected graph  $G$ , the *split tree*  $ST(G)$  of  $G$  is the unique reduced graph-labelled tree  $(T, \mathcal{F})$  in the above Theorem 2.2. Notice that in a split-tree  $ST(G)$ , the clique nodes are pairwise non-adjacent and two star nodes  $u$  and  $v$  can be adjacent only if  $\rho_u(uv)$  and  $\rho_v(uv)$  are both centres or both extremities of their respective stars  $G_u$  and  $G_v$ . For example, Figure 1 shows the split tree of the corresponding graph.

The next two lemmas will be central to various proofs.

**Lemma 2.5** Let  $ST(G) = (T, \mathcal{F})$  be the split tree of a graph  $G$ . Let  $l$  be a leaf of  $T$ , and  $e = uv$ ,  $e' = uv'$  be distinct tree edges such that  $u$  is a  $l$ -accessible and  $e$  is on the  $u, l$  path in  $T$ . Then  $\rho_u(e)\rho_u(e') \in E(G_u)$  if and only if there exists a  $l$ -accessible leaf  $l'$  in the subtree of  $T - e'$  containing  $v$ .

*Proof:* Assume  $\rho_u(e)\rho_u(e') \in E(G_u)$ . Let  $L_u$  and  $L_{v'}$  be the subset of leaves of the maximal subtrees resulting from the removal of  $e'$  and containing respectively  $u$  and  $v'$ . By Lemma 2.5  $(L_u, L_{v'})$  defines a split of  $G$ . It follows that there exists  $l' \in L_{v'}$  such that  $u$  is  $l'$ -accessible. Since  $v'$  is  $l$ -accessible,  $l'$  is also  $l$ -accessible. The converse follows from Definition 2.4 and from the fact that  $ST(G) = (T, \mathcal{F})$ .  $\square$

The above lemma can be rephrased as follows: if  $u$  and  $v$  are two adjacent  $l$ -accessible nodes, then there exists a  $l$ -accessible leaf  $l'$  such that the  $l, l'$ -path contains the tree edge  $u, v$ . A consequence is the following.

**Lemma 2.6** Let  $ST(G) = (T, \mathcal{F})$  be the split tree of a graph  $G$ . For any vertex  $x \in V(G)$ , the subtree  $T(N(x))$  has at most  $2 \cdot |N(x)|$  nodes.

*Proof:* Let  $u$  and  $v$  be two adjacent nodes in  $T(N(x))$  such that  $v$  has degree 2 in  $T(N(x))$  and  $u$  is on the  $x, v$ -path. Let  $a$  be the vertex of  $G_v$  such that  $\rho_v(a) = uv$ . Then  $a$  has degree 1 in  $G_v$  otherwise by Lemma 2.5  $v$  would have degree  $> 2$ . Hence  $G_v$  is not prime (the minimum degree of a prime graph is at least 2), hence it is a star with center  $b$  such that  $ab$  is an edge of  $G_v$ . Let  $w$  be the node neighbor of  $v$  such that  $\rho_v(b) = vw$ . If  $w$  is not a leaf, then  $w$  has degree  $> 2$  in  $T(N(x))$  otherwise it would be a star with center corresponding to  $vw$  and the tree would not be reduced.

So if  $T(N(x))$  is directed from  $x$ , any node of degree 2 is followed by a node with degree  $> 2$ . Hence the result.  $\square$

### 3 Characterizations of distance hereditary graphs

A graph  $G$  is *distance hereditary* if for any connected induced subgraph  $H$  of  $G$  and for any vertices  $x, y \in V(H)$ , we have  $d_H(x, y) = d_G(x, y)$ .

By a theorem of [13], a graph is distance hereditary if and only if it is completely decomposable under the split decomposition, i.e. its split tree is labelled by cliques and stars only. Hence distance hereditary graphs are exactly accessibility graphs of graph-labelled trees whose labels are either stars or cliques, which we call *clique-star trees*. We mention that ternary clique-star trees were used in [10] under the name of  $\Delta$ -confluent trees for drawing distance hereditary graphs. Among several possible clique-star trees, the split tree is the unique reduced one. For example the graph of Figure 1 is distance hereditary. For another general example, cographs are distance hereditary graphs whose stars in the split tree are all directed towards a root of the tree.

#### 3.1 An intersection model

Given a family  $S$  of sets, one can define the intersection graph  $\mathcal{I}(S)$  as the graph whose vertices are the elements of  $S$  and there is an edge between two elements if and only if they intersect (see [17]). Many restricted graph families are defined or characterized as the intersection graphs of certain types of sets. For example, sets of intervals on the real line give interval graphs [11], subtrees of trees give chordal graphs [12]. It is possible to characterize the graph families that support an intersection model without even specifying that model [17]. This result applies to distance hereditary graphs, however no model was known [19]. Based on clique-star trees (not necessarily reduced), an intersection model can be easily derived. We call *accessibility set* of a leaf  $l$  in a graph labelled tree the set of pairs  $\{l, l'\}$  with  $l'$  a  $l$ -accessible leaf.

**Theorem 3.1 (Intersection model)** *A graph is distance hereditary if and only if it is the intersection graph of a family of accessibility sets of leaves in a set of clique-star trees.*

*Proof:* Follows directly from the representation of distance hereditary graphs as accessibility graphs of clique-star trees.  $\square$

In the case of non-connected graphs, it suffices to consider a forest of clique-star trees. It should be noticed that the above result can be formulated for arbitrary clique-star trees as well as for ternary or reduced ones.

#### 3.2 Incremental characterization

Let  $G$  be a distance hereditary graph and let  $ST(G) = (T, \mathcal{F})$  be its split tree. Let  $S$  be a subset of vertices of  $G$  and  $x \notin V(G)$ . We consider the problem of whether the graph  $G + (x, S)$  is distance hereditary or not.

**Definition 3.2** *Let  $T(S)$  be the smallest subtree of  $T$  with leaves  $S$ . Let  $u$  be a node of  $T(S)$ .*

1.  $u$  is fully-accessible if any subtree of  $T - u$  contains a leaf  $l \in S$ ;
2.  $u$  is singly-accessible if it is a star-node and exactly two subtrees of  $T - u$  contain a leaf  $l \in S$  among which the subtree containing the neighbor  $v$  of  $u$  such that  $\rho_u(uv)$  is the centre of  $G_u$ ;
3.  $u$  is partially-accessible otherwise.

We say that a star node is *oriented towards* an edge (or a node) of  $T$  if the tree-edge corresponding to the center of the star is on the path between the edge (or node) and the star.

**Theorem 3.3 (Incremental characterization)** *Let  $G$  be a connected distance hereditary graph and  $ST(G) = (T, \mathcal{F})$  be its split tree. Then  $G + (x, S)$  is distance hereditary if and only if:*

1. *at most one node of  $T(S)$  is partially-accessible;*
2. *any clique node of  $T(S)$  is either fully or partially-accessible.*
3. *if there exists a partially-accessible node  $u$ , then any star node  $v \neq u$  of  $T(S)$  is oriented towards  $u$  if and only if it is fully-accessible. Otherwise, there exists a tree-edge  $e$  of  $T(S)$  towards which any star node of  $T(S)$  is oriented if and only if it is fully-accessible;*

*Proof:*

$\Rightarrow$  Since  $G + (x, S)$  is a distance hereditary graph, it has a ternary clique-star tree  $(\tilde{T}, \tilde{\mathcal{F}})$ . Let  $u$  be the node of  $T'$  to which  $x$  is attached and let  $v, w$  be its neighbors. Now consider the clique-star tree  $(T', \mathcal{F}')$  obtained by applying any possible clique-join or a star-join to tree-edges different from  $uv$  and  $uw$ . Notice that  $ST(G)$  is obtained by 1) removing  $x$  and  $u$  in  $T'$ , 2) making  $vw$  adjacent, and 3) if needed apply a join on the new tree-edge  $vw$ .

Assume the join on  $vw$  is not required to obtain  $ST(G)$ . Then any node of  $T(S)$  is a node of  $T'$  and as any leaf of  $S$  was  $x$ -accessible in  $T'$ , the three conditions are a consequence of Lemma 2.5. In that case, all the fully accessible star-nodes are oriented toward the tree-edge  $vw$ .

Assume  $ST(G)$  is obtained after a join on  $vw$  which result on a new node  $u'$ . Then any node of  $T(S)$  but  $u'$  of  $ST(G)$  corresponds to a node of  $T'$ . Again by Lemma 2.5 the nodes of  $ST(G)$  different than  $u'$  are all singly of fully-accessible. It is straightforward to check that  $u'$  is partially accessible and that the third condition holds (as a consequence of Lemma 2.5).

$\Leftarrow$  Assume there is no partially accessible node, so there exists a tree-edge  $e$  toward which only the fully-accessible star nodes are oriented. Any node of  $T(S)$  is either fully-accessible or singly-accessible, a degree 2 node in  $T(S)$  is singly-accessible. Let  $w$  be a node on the tree-path between any  $y \in S$  and  $e$  and let  $e_y, e_x$  be the two tree-edges on that path incident to  $w$ . By Definition 3.2, we have that  $\rho_w(e_x)\rho_w(e_y) \in G_w$ . It follows that any  $y \in S$  is a neighbor of  $x$  in  $G(T', \mathcal{F}')$ . Let us now prove that any  $z \notin S$  is not a neighbor of  $x$  in  $G(T', \mathcal{F}')$ , thereby proving that  $G(T', \mathcal{F}') = G + (x, S)$ . Let  $w$  the node of  $T(S)$  which is the closest to the leaf corresponding to  $z$  and  $e_z, e_f$  be the two tree-edges on the tree-path from  $z$  to  $e$  incident to  $w$ . If  $\rho_w(e_z)\rho_w(e_f) \notin E(G_w)$  then  $z$  is not a neighbor of  $x$  in  $G(T', \mathcal{F}') = G + (x, S)$ . As assume by contradiction  $\rho_w(e_z)\rho_w(e_f) \in E(G_w)$ . It follows that  $w$  is a fully-marked node, meaning that each subtree of  $ST(G) - w$  (but possibly the one containing  $e$ ) contains a leaf of  $S$ , which contradict the fact that  $w$  is the closest node to  $z$  belonging to  $T(S)$ .

Assume there is a partially accessible node  $u$ . Then it suffices to split the node  $u$  into two new nodes  $v$  and  $w$ , such that  $v$  is adjacent to the neighbors of  $u$  not belonging to  $T(S)$  and  $w$  to those belonging to  $T(S)$ . Now star-nodes of  $T(S)$  are oriented toward the new tree-edge  $e = vw$ , and the same arguments than above apply.

□

## 4 A vertex-only fully-dynamic recognition algorithm

In this section we mainly compute the characterization given by Theorem 3.3 to obtain the following main result.

**Theorem 4.1** *There exists a fully dynamic recognition algorithm for connected distance hereditary graphs with complexity  $O(d)$  per vertex insertion or deletion operation of  $d$  edges.*

The vertex insertion algorithm can be used to obtain a linear time recognition algorithm of (static) distance hereditary graphs, thereby achieving the best known bound. We claim that this incremental algorithm turns out to be simpler than the previous non-incremental ones [13, 9, 2].

**Corollary 4.2 (Static recognition)** *Given a graph  $G$ , the vertex insertion routine enables to test in linear time whether  $G$  is distance hereditary or not.*

*Proof:* Assume the graph  $G$  is connected. Notice that our insertion algorithm applied to connected graphs. So we have to preprocess the input graph in order to obtain an ordering on its vertices such that the subgraph induced by the first  $i$  vertices (for  $1 \leq i \leq n$ ) is connected. Such a preprocessing can be done by any search in linear time. Then the vertex insertion algorithm can be used by adding vertices one by one. The cost is linear in the sum of degrees of vertices, hence linear in the number of edges of the graph. And if the graph is not connected, this can be applied on each connected component separately.  $\square$

The algorithm for vertex insertion is described in Subsection 4.1 and the vertex deletion is in Subsection 4.2. The following data-structure is used for the clique-star tree  $ST(G) = (T, \mathcal{F})$  of the given distance hereditary graph  $G$ : a (rooted) representation of the tree  $T$ ; a single *clique-star marker* distinguishing the type of each node; a *center-marker* distinguishing the center of each star node; and the degree of each node. Let us notice that this data-structure is an  $O(n)$  space representation of the distance hereditary graphs. By our algorithm, or other known algorithms [7, 8], given a distance hereditary graph  $G$  on  $n$  vertices and  $m$  edge, this representation can be computed in time  $O(n + m)$ . So the input of our updating routines (insertion and deletion) is this clique-star tree data-structure, a vertex  $x$  and a set  $S$  of vertices (or leaves of the clique-star tree).

### 4.1 Vertex insertion

The insertion algorithm consists in three different steps: we first identify the subtree  $T(S)$ , then check for the conditions of Theorem 3.3 and finally if the augmented graph is distance hereditary, we update the data-structure of the split tree.

**Computing the smallest subtree spanning a set of leaves.** First, given a set  $S$  of leaves of a tree  $T$ , we need to identify the smallest subtree  $T(S)$  spanning  $S$ , and store also the degrees of its nodes. This problem is easy to solve on rooted (or directed) trees. The algorithm is a simple bottom-up marking process with complexity  $O(|T(S)|)$ :

1. Mark each leaf of  $S$ . A node is *active* if its father is not marked.
2. Each marked node marked its father as long as: 1) the root is not marked and there is more than one active node, or 2) the root is marked and there is at least one active node.

3. As long as the root of the subtree induced by the marked nodes is a leaf not in  $S$ , remove this node and check again.

We point out that the representation of our split tree  $T$  has to be (artificially) rooted to enable this efficient computation of  $T(S)$ . It is this constraint only which imposes a restriction to connected graphs in Theorem 4.1.

**Testing conditions of Theorem 3.3.** The first two conditions of Theorem 3.3 are fairly easy to check by following Definition 3.2: a node  $u$  is fully-accessible if its degrees in  $T(S)$  and  $T$  are the same;  $u$  is singly-accessible if it is a star, if it has degree 2 in  $T(S)$  and if the centre neighbor belongs to  $T(S)$ ; and  $u$  is partially accessible otherwise, such a node having to be unique if it exists. This test costs  $O(|T(S)|)$ .

We now assume that the first two conditions of Theorem 3.3 are fulfilled. At first, the case is trivial if  $|S| = 1$ . We therefore assume that  $|S| > 1$ .

We define *local orientations* on vertices of a tree as the choice, for every vertex  $u$ , of a vertex  $f(u)$  such that either  $f(u) = u$  or  $f(u)$  is a neighbor of  $u$ . Local orientations are called *compatible* if 1)  $f(u) = u$  implies  $f(v) = u$  for every neighbor  $v$  of  $u$ , and 2)  $f(u) = v$  implies  $f(w) = u$  for every neighbor  $w \neq v$  of  $u$ .

It is an easy exercise to see that if local orientations are compatible then exactly one of the two following properties is true: either there exist a unique vertex  $u$  with  $f(u) = u$ , in this case  $u$  is called *node-root*, or there exist a unique tree-edge  $uv$  with  $f(u) = v$  and  $f(v) = u$ , in this case  $uv$  is called *edge-root*. The test for the third condition of Theorem 3.3 consists in building, if possible, suitable compatible local orientations in the tree  $T(S)$ :

1. Let  $u$  be a leaf of  $T(S)$ . Then  $f(u)$  is the unique neighbor of  $u$ .
2. Let  $u$  be a star node of  $T(S)$ . If  $u$  is partially-accessible, then  $f(u) = u$ . If  $u$  is singly-accessible, then  $f(u) = v$  with  $v$  the unique neighbor  $v$  of  $u$  belonging to  $T(S)$  such that  $\rho_u(uv)$  is an extremity of the star. If  $u$  is fully-accessible, then  $f(u) = v$  with  $v$  the neighbor of  $u$  such that  $\rho_u(uv)$  is the centre of the star.
3. Let  $u$  be a clique node of  $T(S)$ . If  $u$  is partially-accessible, then  $f(u) = u$ . Otherwise,  $u$  is fully-accessible and its neighbors are leaves or star nodes. If  $f(v) = u$  for every neighbor  $v$  of  $u$  then  $f(u) = u$ . If  $f(v) = u$  for every neighbor  $v$  of  $u$  but one, say  $w$ , then  $f(u) = w$ . Otherwise  $u$  is an *obstruction*.

The third condition of Theorem 3.3 is satisfied if and only if 1) there is no obstruction and 2) local orientations of  $T(S)$  are compatible. This test can be done in time  $O(|T(S)|)$  by a search of  $T(S)$ . Hence, the conditions of Theorem 3.3 can be tested in  $O(|T(S)|)$  time.

**Updating the split-tree.** We now assume that conditions of Theorem 3.3 are satisfied, i.e. the graph  $G + (x, S)$  is distance hereditary. As already seen, the tree either has a unique node-root or a unique edge-root. There are three cases (see Figure 3).

1. There is a node-root  $u$  being partially-accessible, or  $S$  is reduced to a unique vertex  $u$ . We may have to make a first update of  $T$  by splitting the node  $u$  under some conditions on the degrees. Let  $U$ , resp.  $A$ , be the set of tree-edges adjacent to  $u$  in  $T$ , resp. in  $T(S)$ .

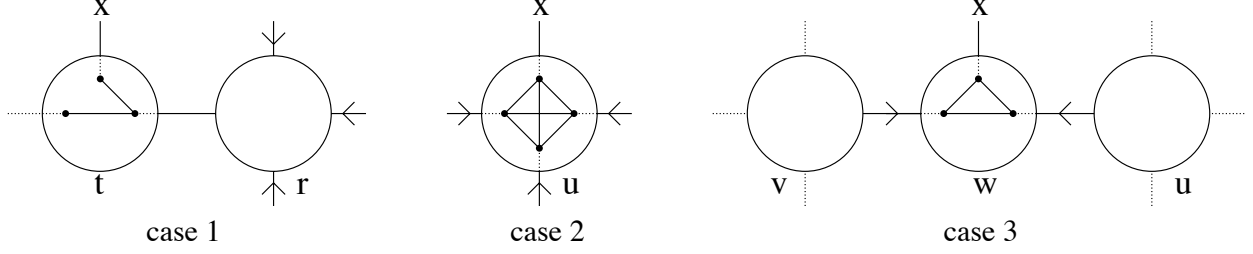


Figure 3: Vertex insertion

- (a) If  $u$  is a clique node with  $|U \setminus A| \geq 2$ , then  $u$  is replaced with an edge  $vw$  in  $T$ . Then  $v$ , resp.  $w$ , is labelled by a clique whose vertices correspond to  $A$ , resp.  $U \setminus A$ , except one more which corresponds to  $vw$ . In this case,  $v$  is now the node-root.
- (b) If  $u$  is a star node with center corresponding to the tree edge  $e$  and  $|(U \setminus A) \setminus e| \geq 1$ , then  $u$  is replaced with an edge  $vw$  in  $T$ . Then  $v$  is labelled by a star whose extremities correspond to  $A \setminus \{e\}$  and center to  $vw$  (we have  $|A \setminus \{e\}| > 1$  since  $u$  is not singly accessible), and  $w$  is labelled by a star whose extremities correspond to  $(U \setminus A) \cup \{vw\}$  and center to  $e$ . In the case where  $e \in A$ , the edge  $vw$  is now the edge-root. In the case where  $e \notin A$ , the node  $v$  is now the node-root.

If the tree still has a node-root  $r = u$  or  $r = v$ , then let  $s$  be its neighbor in  $T$  that does not belong to  $T(S)$ . Then the insertion edge is  $e = rs$ , and  $ST(G + (x, S))$  is obtained by subdividing  $rs$  with a star node  $t$  of degree 3 whose centre is  $\rho_w(rt)$  and making the leaf  $x$  adjacent to  $t$ . Then, in the case where  $s$  is a star with centre  $\rho_v(st)$ , we proceed a join operation on the tree-edge  $st$ .

2. There is a node-root  $u$  not being partially-accessible, Then  $u$  is a clique node, and  $ST(G + (x, S))$  is obtained by adding the new leaf  $x$  adjacent to  $u$  whose degree thereby increases by one.
3. There is an edge-root  $uv$ . Then  $ST(G + (x, S))$  is obtained by subdividing  $uv$  with a clique node  $w$  of degree 3 and making the leaf  $x$  adjacent to  $w$ .

Each above update operation can be done in  $O(1)$  time, except the splitting in case 1 which can be done in time  $O(|T(S)|)$  (by deleting  $A$  from  $u$  to get  $w$ , and adding  $A$  to a new empty node  $v$ ). It is straightforward to see that all other update operations required to maintain the data structure of the split tree (artificial root, degrees...) can be done in  $O(1)$  time. Finally, we obtain the main result for vertex insertion. The complexity for the whole algorithm derives from previous steps and the fact that  $O(|T(S)|) = O(|S|)$  by Lemma 2.6.

**Theorem 4.3 (Vertex insertion)** *Let  $G + (x, S)$  be a graph such that  $G$  is a connected distance hereditary graph. Given the data structure of split tree  $ST(G)$ , testing whether  $G + (x, S)$  is a distance hereditary graph and if so computing the data structure of  $ST(G + (x, S))$  can be done in  $O(|S|)$  time.*

*Proof:* Follows from the discussion above. □

## 4.2 Vertex deletion

Removing a vertex  $x$  from a distance hereditary graph  $G$  always yields a distance hereditary graph  $G - x$ . Let  $(T, \mathcal{F})$  be the clique-star tree of  $G$ . Updating the data-structure of the clique-star tree can be done as follows.

1. Remove the leaf  $x$  and update the degree of its neighbor  $v$ .
2. If  $v$  now has degree 2, then replace  $v$  with an edge between its neighbors, and, if needed, reduce the clique-star tree on this edge.
3. If  $v$  is a star node whose centre neighbor was  $x$ , then  $G - x$  is no longer connected, and the split-trees of each connected component are the components of  $T - \{v, x\}$ .

**Lemma 4.4 (Vertex deletion)** *Let  $G$  be a connected distance hereditary graph and  $x$  be a degree  $d$  vertex of  $G$ . Given the data structure of split tree  $ST(G)$ , testing whether  $G - x$  is a connected distance hereditary graph and if so computing the data structure of  $ST(G - x)$  can be done in  $O(d)$  time.*

*Proof:* It is easy to see that any operation costs  $O(1)$  except the join operation which costs  $\min(d', d'')$  where  $d', d''$  are degree of the concerned nodes. Since at least one of these nodes is fully accessible, this minimum degree is lower than  $d$ . Hence this join operation costs  $O(d)$ .  $\square$

## 5 Concluding remark

As already noticed, our algorithm is optimal even with respect to the static recognition problem of distance hereditary graphs. Let us also mention that its principle can be generalized to compute the split tree of arbitrary graphs, but then the complexity is no longer linear in the number of edges. Actually even for circle graphs a vertex insertion (or deletion) implies  $\Omega(n)$  changes in the split-tree representation. This complexity is reached for example for the insertion of a vertex  $x$  adjacent to the two extremities of a path  $P_n$  on  $n$  vertices. The graph  $P_n$  is distance hereditary (its split tree is a path of ternary stars) and the cycle on  $n + 1$  vertices is a circle graph, which is prime and thus has a unique node in its split tree.

## References

- [1] H.-J. Bandelt and H.M. Mulder. Distance hereditary graphs. *Journal of Combinatorial Theory Series B*, 41:182–208, 1986.
- [2] A. Bretscher. *LexBFS based recognition algorithms for cographs and related families*. PhD thesis, Department of Computer Science, University of Toronto, 2005.
- [3] D. Corneil and M. Tedder. An optimal, edges-only fully dynamic algorithm for distance-hereditary graphs. In *24th International Symposium on Theoretical Aspects of Computer Science (STACS)*, Lecture Notes in Computer Science, 2007.

- [4] C. Crespelle and C. Paul. Fully-dynamic recognition algorithm and certificate for directed cographs. In *30th Int. Workshop on Graph Theoretical Concepts in Computer Science (WG)*, number 3353 in Lecture Notes in Computer Science, pages 93–104, 2004. Full version in *Discrete Applied Mathematics* 154(12):1722–1741, 2006.
- [5] C. Crespelle and C. Paul. Fully dynamic algorithm for recognition and modular decomposition of permutation graphs. In *31th Int. Workshop on Graph Theoretical Concepts in Computer Science (WG)*, number 3787 in Lecture Notes in Computer Science, 2005.
- [6] W.H. Cunningham. Decomposition of directed graphs. *SIAM Journal on Algebraic Discrete Methods*, 3:214–228, 1982.
- [7] Elias Dahlhaus. Efficient parallel and linear time sequential split decomposition (extended abstract). In *Foundations of Software Technology and Theoretical Computer Science - FSTTCS*, volume 880 of *Lecture Notes in Computer Science*, pages 171–180, 1994.
- [8] Elias Dahlhaus. Parallel algorithms for hierarchical clustering and applications to split decomposition and parity graph recognition. *J. Algorithms*, 36(2):205–240, 2000.
- [9] G. Damiand, M. Habib, and C. Paul. A simple paradigm for graph recognition: application to cographs and distance hereditary graphs. *Theoretical Computer Science*, 263:99–111, 2001.
- [10] David Eppstein, Michael T. Goodrich, and Jeremy Yu Meng. Delta-confluent drawings. In *13th Int. Symp. Graph Drawing (GD)*, number 3843 in Lecture Notes in Computer Science, pages 165–176, 2005.
- [11] D.R. Fulkerson and O.A. Gross. Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15:835–855, 1965.
- [12] F. Gavril. The intersection graphs of a path in a tree are exactly the chordal graphs. *Journ. Comb. Theory*, 16:47–56, 1974.
- [13] P. Hammer and F. Maffray. Completely separable graphs. *Discrete Applied Mathematics*, 27:85–99, 1990.
- [14] P. Hell, R. Shamir, and R. Sharan. A fully dynamic algorithm for recognizing and representing proper interval graphs. *SIAM Journal on Computing*, 31(1):289–305, 2002.
- [15] S.-Y. Hsieh, C.-W. Ho, T.-S. Hsu, and M.-T. Ho. Efficient algorithms for the hamiltonian problem on distance hereditary graphs. In *Annual International Computing and Combinatorics Conference (COCOON)*, volume 2387 of *Lecture Notes in Computer Science*, pages 77–86, 2002.
- [16] L. Ibarra. Fully dynamic algorithms for chordal graphs. In *10th ACM-SIAM Annual Symposium on Discrete Algorithm (SODA)*, pages 923–924, 1999.
- [17] T. McKee and F.R. McMorris. *Topics in intersection graph theory*. Number 2 in SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics (SIAM), 1999.

- [18] R. Shamir and R. Sharan. A fully dynamic algorithm for modular decomposition and recognition of cographs. *Discrete Applied Mathematics*, 136(2-3):329–340, 2004.
- [19] J. Spinrad. List of open problems. <http://www.vuse.vanderbilt.edu/spin/open.html>.
- [20] R. Uehara and Y. Uno. Canonical tree representation of distance hereditary graphs and its applications. Technical Report COMP2005-61, IEICE Technical Report, 2006.