POLYNOMIAL TIME ALGORITHMS FOR COMPUTING A MINIMUM HULL SET IN DISTANCE-HEREDITARY AND CHORDAL GRAPHS

MAMADOU MOUSTAPHA KANTÉ AND LHOUARI NOURINE

Abstract. We give linear and polynomial time algorithms for computing a minimum hull-set in distance-hereditary and chordal graphs respectively. Prior to our result a polynomial time algorithm was only known for sub-classes of considered graph classes. Our techniques allow us to give at the same time a linear time algorithm for computing a minimum geodetic set in distance-hereditary graphs.

1. Introduction

In this paper we consider convexity related to shortest paths in graphs. For an undirected graph G and a subset X of the vertex-set of G, let $I[X]$ be the set of vertices that lie in a shortest path between two vertices of X. The *closure* of X is the smallest $X' \supseteq X$ such that $I[X'] = X'$. The *hull number* [19] of a graph G is defined as the minimum k such that there exists a set X of size k whose closure is the set of vertices of G. The notion of hull number has been introduced in [19] and has attracted many interest in the years [1, 3, 15, 16]. Computing the minimum hull number of bipartite graphs is NP-complete [1] and polynomial time algorithms have been proposed for some graph classes: proper interval graphs, cographs, split graphs [15], cobipartite graphs, $(q, q-4)$-graphs [1].

In this paper we give polynomial time algorithms for computing a minimum hull set in distance-hereditary and chordal graphs. The computational complexity of the hull number of chordal graphs has been left open since [15] and to our knowledge no exact algorithm for the computation of the hull number of distance-hereditary graphs is known. Surprisingly, the related notion *geodetic number* has been proven to be NP-complete in chordal graphs [17] and a polynomial time algorithm for interval graphs is open (polynomial time algorithms for split graphs and proper interval graphs are given in [17] and [18]).

Our linear time algorithm for distance-hereditary graphs can be summarised as follows. We will first give a *monadic second-order* formula $HullSet(X)$ that holds in a connected distance-hereditary graph G if and only if X is a hull set of G. We will then use Courcelle et al.'s theorem [9] stating that monadic second-order properties can be solved in linear time in distance-hereditary graphs. In order to write the formula $HullSet(X)$ we will express in monadic second-order logic the property "z is in a shortest path between x and y" and we will use for that the well-known notion of *split-decomposition* [11] and a characterisation of the property "being in a shortest path" by means of split-decomposition.

The algorithm for chordal graphs is based upon the notion of *functional dependencies* which are constraints used in relational database design and specially in

Date: June 15, 2012.

M.M. Kanté is supported by the DORSO project, and L. Nourine by the DAG project, both of "Agence Nationale Pour la Recherche".

normalization process [5].

Summary. Section 2 is devoted to preliminaries and basic facts. A linear time algorithm for distance-hereditary graphs, based on logical tools, is given in Section 3. Section 4 is devoted to the polynomial time algorithm for chordal graphs. We finish by some concluding remarks and open questions in Section 5.

2. Preliminaries

2.1. **Graphs.** If A and B are two sets, $A \backslash B$ denotes the set $\{x \in A \mid x \notin B\}$. The power set of a set V is denoted by 2^V. The size of a set A is denoted by $|A|$.

We refer to [13] for our graph terminology. A *graph* G is a pair (V_G, E_G) where V_G is the set of vertices and $E_G \subseteq (V_G \times V_G) \setminus (\{(x,x) \mid x \in V_G\})$ is the set of edges. A graph G is said to be *undirected* if $(x,y) \in E_G$ implies $(y,x) \in E_G$; an edge (x,y) is hence written xy (equivalently yx). For a graph G, we denote by $G[X]$, called the subgraph of G induced by $X \subseteq V_G$, the graph $(X, E_G \cap (X \times X))$. The *size* of a graph G, denoted by $\|G\|$, is defined as $|V_G| + |E_G|$. For a vertex x of a graph G, we let $N_G(x)$ be the set $\{y \mid (x,y) \in E_G\}$.

A *path of length* k in a graph G from the vertex x to the vertex y is a sequence $(x = x_0, x_1, \ldots, x_k = y)$ such that the set $\{(x_i, x_{i+1}) \mid 0 \leq i \leq k-1\}$ is a subset of E_G; this path is said *chordless* if $(x_i, x_j) \notin E_G$ whenever $j > i+1$ or $j < i-1$. It is worth noticing that whenever G is undirected, if $P := (x_0, \ldots, x_k)$ is a path from x to y, $(x_k, \ldots, x_0)$ is also a path from y to x and we will say in this case that P is between x and y. A graph G is said *strongly connected* if for every pair (x,y) of vertices there exists a path from x to y and also a path from y to x. A *strongly connected component* of a graph G is a maximal strongly connected induced subgraph of G. Strongly connected undirected graphs are simply said to be *connected* and their strongly connected components are called *connected components*.

The *distance* between two vertices x and y in an undirected graph G, denoted by $d_G(x,y)$, is the minimum k such that there exists a path of length k between x and y; if no such path exists then $d_G(x,y) = \infty$ (this does happen if G is not connected). Any path between two vertices x and y of length $d_G(x,y)$ is called a *shortest path* and is by definition a chordless path.

An undirected graph G is said *complete* if $E_G = (V_G \times V_G) \setminus (\{(x,x) \mid x \in V_G\})$ (it is denoted K_n if it has n vertices), and it is called a *cycle of length* n if its edge-set is the set $\{x_i x_{i+1} \mid 1 \leq i \leq n-1\} \cup \{x_1 x_n\}$ with $(x_1, \ldots, x_n)$ an ordering of its vertex-set. An undirected graph G is called *distance-hereditary* if for every two vertices x and y of G all the chordless paths between x and y have the same length, and it is called *chordal* if it has no induced cycle of length greater than or equal to 4. A vertex x of an undirected graph G is called *simplicial* if $N_G(x)$ is a complete graph. A *tree* is an acyclic connected undirected graph and a *star* is a tree with a distinguished vertex adjacent to the other vertices (it is denoted S_n if it has n vertices).

2.2. **Betweenness Relations.** Let V be a finite set. A *betweenness relation on* V is a ternary relation $B \subseteq V^3$ such that for every $x, y, z \in V$, we have $B(x,z,y)$ holds if and only if $B(y,z,x)$ holds; z is said to be *between* x and y. Several betweenness relations, particulary in graphs, are studied in the literature (see [4, 25]).

Let B be a betweenness relation on a finite set V. For every subset X of V, we let $B^o(X)$ be $\bigcup_{x,y \in X} \{z \in V \mid B(x,z,y)\}$. A subset X of V is said *B-convex* if $B^o(X) = X$ and its *B-convex hull*, denoted $B^+(X)$, is the smallest B-convex set that contains X. A subset X of V is a *B-hull set* (resp. *B-geodetic set*) if $B^+(X) = V$ (resp. $B^o(X) = V$).

In this paper, we deal with the following betweenness relation. For every undirected graph G, we define the betweenness relation $\mathcal{SP}_G$ on V_G where $\mathcal{SP}_G(x,z,y)$ holds if and only if z is in a shortest path between x and y. We are interested in computing the *hull number* of an undirected graph G defined as the size of a minimum $\mathcal{SP}_G$-hull set of V_G [19]. The computation of the hull number of a graph is NP-complete [15] and polynomial time algorithms exist for some graph classes (see for instance [1, 15, 16]). We give polynomial time algorithms for distance-hereditary and chordal graphs. Our techniques will allow us to derive a linear time algorithm for computing the *geodetic number* of a distance-hereditary graph G, defined as the $\mathcal{SP}_G$-geodetic set of V_G [17].

Fact 1. *A $\mathcal{SP}_G$-hull set of a non connected undirected graph G is the union of $\mathcal{SP}_G$-hull sets of its connected components. Similarly for $\mathcal{SP}_G$-geodetic sets.*

Proof. This follows from the fact that for every triple x, y, z of V_G, z is in a shortest path between x and y if and only if $d_G(x,y) = d_G(x,z) + d_G(z,y)$. □

Fact 2. *Any $\mathcal{SP}_G$-hull set (or $\mathcal{SP}_G$-geodetic set) of an undirected graph G must contain the set of simplicial vertices of G.*

We conclude this section by some facts about shortest paths in a connected undirected graph. Let G be a connected undirected graph. A *BFS* (*Breadth-First Search*) tree at x in G is a tree constructed by a BFS search starting at x [6]. For each $i > 1$, let L_i, called the *level* i, be the set $\{y \in V_G \mid d_G(x,y) = i\}$. It is well-known that a BFS tree T partitions the edges of G into 3 parts: the edges of the BFS tree that we call *white* edges; the edges between two vertices in the same level, called *red* edges; and those that are between two vertices in consecutive levels, called *blue* edges.

Proposition 1. *Let x be a vertex in a connected undirected graph G and let T be a BFS tree at x in G. Then, a sequence $P := (x = x_0, x_1, \ldots, x_k)$ is a shortest path if and only if $x_i x_{i+1}$ is either white or blue with $x_i \in L_i$ and $x_{i+1} \in L_{i+1}$ for all $0 \leq i \leq k-1$.*

Proof. By induction on the length k of P. The statement is clear for $k = 1$. Assume it is true for some $k \geq 1$ and let prove it for $k+1$.

Let $P := (x = x_0, x_1, \ldots, x_{k+1})$ be a path where $x_i x_{i+1}$ is either white or blue, and $x_i \in L_i$ and $x_{i+1} \in L_{i+1}$ for all $0 \leq i \leq k$. By induction, $(x = x_0, x_1, \ldots, x_k)$ is a shortest path between x and x_k. Since x_{k+1} is in level L_{k+1}, then $d_G(x, x_{k+1}) = k+1$, *i.e.*, P is a shortest path between x and x_{k+1}.

Assume now that $P := (x = x_0, x_1, \ldots, x_{k+1})$ is a shortest path. By definition, $(x_0 = x, \ldots, x_k)$ is a shortest path between x and x_k and by inductive hypothesis $x_i x_{i+1}$ is white or blue for all $0 \leq i \leq k-1$. Since x_{k+1} is in level $k+1$ and x_k is in level k ($d_G(x, x_k) = k$ and $d_G(x, x_{k+1}) = k+1$), the edge $x_k x_{k+1}$ can only be blue or white. This finishes the proof. □

As a corollary, we get the following.

Proposition 2. *Let G be a connected undirected graph and let x be a vertex of G. Then, we can compute all the sets $\mathcal{SP}^o_G(\{x,y\})$, for all vertices $y \in V_G \setminus \{x\}$, in time $O(\|G\| + \sum_{y \in V_G \setminus \{x\}} \left|\mathcal{SP}^o_G(\{x,y\})\right|)$.*

Proof. In the algorithm that computes a BFS tree T at x, we initialise $\mathcal{SP}^o_G(\{x\})$ to $\emptyset$ and when adding a vertex y in level i of T we let $\mathcal{SP}^o_G(\{x,y\})$ be the set

$\bigcup_{\substack{zy\in E_G\\ z\in L_{i-1}}} (\mathcal{SP}^o_G\{x,z\}\cup\{z\})$. It is a straightforward induction to prove that this algorithm computes $\mathcal{SP}^o_G(\{x,y\})$ for all $y\in V_G\backslash x$ in time $O\left(\|G\| + \sum_{y\in V_G\backslash\{x\}} \left|\mathcal{SP}^o_G(\{x,y\})\right|\right)$. □

2.3. **Monadic Second-Order Logic.** We refer to [8] for more information. A *relational signature* is a finite set $\mathcal{R} := \{R, S, T, \ldots\}$ of relation symbols, each of which given with an arity $ar(R) \geq 1$. A *relational $\mathcal{R}$-structure* $\mathfrak{A}$ is a tuple $(A, (R_\mathfrak{A})_{R\in\mathcal{R}})$ with $R_\mathfrak{A} \subseteq A^{ar(R)}$ for every $R \in \mathcal{R}$ and A is called its domain. Examples of relational structures are graphs that can be seen as relational $\{edg\}$-structures with $ar(edg) = 2$, *i.e.*, a graph G is seen as the relational $\{edg\}$-structure (V_G, edg_G) with V_G its set of vertices and $edg_G(x, y)$ holds if and only if $(x, y) \in E_G$.

We will use lower case variables $x, y, z, \ldots$ (resp. upper case variables $X, Y, Z, \ldots$) to denote elements of domains (resp. subsets of domains) of relational structures. Let $\mathcal{R}$ be a relational signature. The *atomic formulas over* $\mathcal{R}$ are $x = y$, $x \in X$ and $R(x_1, \ldots, x_{ar(R)})$ for $R \in \mathcal{R}$. The set $MS_\mathcal{R}$ of *monadic second-order formulas over* $\mathcal{R}$ is the set of formulas formed from atomic formulas over $\mathcal{R}$ with Boolean connectives $\wedge$, $\vee$, $\neg$, $\Longrightarrow$, $\Longleftrightarrow$, *element quantifications* $\exists x$ and $\forall x$, and *set quantifications* $\exists X$ and $\forall X$. An occurrence of a variable which is not under the scope of a quantifier is called a *free variable*. We will write $\varphi(x_1, \ldots, x_m, Y_1, \ldots, Y_q)$ to express that the formula φ has $x_1, \ldots, x_m, Y_1, \ldots, Y_q$ as free variables and $\mathfrak{A} \models \varphi(a_1, \ldots, a_m, Z_1, \ldots, Z_q)$ to say that $\varphi(a_1, \ldots, a_m, Z_1, \ldots, Z_q)$ holds in $\mathfrak{A}$ when substituting $(a_1, \ldots, a_m) \in A^m$ to element variables $(x_1, \ldots, x_m)$ and $(Z_1, \ldots, Z_q) \in (2^A)^q$ to set variables $(Y_1, \ldots, Y_q)$ in $\varphi(x_1, \ldots, x_m, Y_1, \ldots, Y_q)$. The following is an example of a formula expressing that two vertices x and y are connected by a path

$$\forall X(x \in X \ \wedge\ \forall z, t(z \in X \ \wedge\ edg(z, t) \implies t \in X) \Longrightarrow y \in X).$$

If $\varphi(x_1, \ldots, x_m, Y_1, \ldots, Y_q)$ is a formula in $MS_\mathcal{R}$, we let opt_φ, with $opt \in \{\min, \max\}$, be the problem that consists in, given a relational $\mathcal{R}$-structure $\mathfrak{A}$, to finding a tuple $(Z_1, \ldots, Z_q)$ of $(2^A)^q$ such that

$$\sum_{1\leq i\leq q} |Z_i| = opt\left\{\sum_{1\leq i\leq q} \left|W_j\right| \mid \mathfrak{A} \models \varphi(a_1, \ldots, a_m, W_1, \ldots, W_q)\right\}.$$

Many optimisation graph problems, *e.g.*, minimum dominating set, maximum clique, ..., correspond to opt_φ for some $MS_{\{edg\}}$ formula φ.

Let A be a finite set. An *A-coloured graph* is a graph with its edges and vertices labelled with elements in A. Let $\mathcal{R}_A$ be the relational signature $\{(edg_a)_{a\in A}, (nlab_a)_{a\in A}\}$ with $ar(edg_a) = 2$ and $ar(nlab_a) = 1$ for every $a \in A$. Every A-coloured graph G can be represented by the relational $\mathcal{R}_A$-structure $(V_G, (edg_{aG})_{a\in A}, (nlab_{aG})_{a\in A})$ where $edg_{aG}(x, y)$ holds if and only if $xy \in E_G$ is labelled with $a \in A$ and $nlab_{aG}(x)$ holds if and only if $x \in V_G$ is labelled with $a \in A$.

Clique-width is a graph complexity measure introduced by Courcelle et al. [9, 10] and that is important in complexity theory. We refer to [8] for more information on clique-width. We recall the following important theorem that is the base of our algorithm for distance-hereditary graphs.

Theorem 1. [8] *Let A be a fixed finite set and let k be a fixed constant. For every $MS_{\mathcal{R}_A}$ formula $\varphi(x_1, \ldots, x_m, Y_1, \ldots, Y_q)$, opt_φ, for $opt \in \{\min, \max\}$, can be solved in linear time in any A-coloured graph of clique-width at most k, provided the clique-width expression is given.*

3. Distance-Hereditary Graphs

Our algorithm will use Theorem 1. We will first associate with every distance-hereditary graph G an A-coloured graph $\mathcal{S}_G$ for some finite set A. For doing so, we will use the well-known notion of *split-decomposition* defined by Cunningham and Edmonds [11]. In a second step, we will prove that there exists a formula $Bet_{\mathcal{SP}}(x,z,y)$ in $MS_{\mathcal{R}_A}$ that holds in $\mathcal{S}_G$ if and only if z is in a shortest path in G between x and y. These two constructions combined with Theorem 1 and the next proposition will give rise to the linear time algorithm.

Proposition 3. *Let B be a betweenness relation on a finite set V and assume there exists a relational $\mathcal{R}$-structure $\mathfrak{A}$ with $V \subseteq A$, for some relational signature $\mathcal{R}$ that contains a relation $nlab_{\mathbf{V}}$ representing V. Assume also that there exists an $MS_{\mathcal{R}}$ formula $Bet(x,z,y)$ that holds in $\mathfrak{A}$ if and only if $B(x,z,y)$ holds. Then, there exist $MS_{\mathcal{R}}$ formulas $HullSet(X)$ and $GeodeticSet(X)$ expressing that X is a B-hull set and a B-geodetic set respectively.*

Proof. We write "$X \subseteq Y$" and "$X \subsetneq Y$" as shortcuts for "$\forall x(x \in X \implies x \in Y)$" and "$X \subseteq Y \wedge \exists y(y \in Y \wedge y \notin X)$" respectively. The following formula $Closed(X)$, with free set variable X, says that X is a B-convex set

$$X \subseteq nlab_{\mathbf{V}} \ \wedge \ \forall x,y(x \in X \ \wedge \ y \in X \implies \neg\exists z(Bet(x,z,y))).$$

The following formula, $ConvexHull(X,Y)$, with free set variables X and Y, says that Y is the B-convex hull of X

$$Closed(Y) \wedge X \subseteq Y \wedge \forall Z(X \subseteq Z \wedge Z \subseteq Y \implies \neg Closed(Z)).$$

The next formulas are $HullSet(X)$ and $GeodeticSet(X)$ respectively (in this order).

$$\forall Z\big(Z \subsetneq nlab_{\mathbf{V}} \implies \neg ConvexHull(X,Z)\big).$$

$$X \subseteq nlab_{\mathbf{V}} \wedge \forall z\Big(nlab_{\mathbf{V}}(z) \implies \exists x,y\big(x \in X \wedge y \in X \wedge Bet(x,z,y)\big)\Big).$$

It is straightforward to check the validity of each formula. □

3.1. **Split Decomposition.** We will follow [7] (see also [22]) for our definitions. Two bipartitions $\{X_1,X_2\}$ and $\{Y_1,Y_2\}$ of a set V *overlap* if $X_i \cap Y_j \neq \emptyset$ for all $i,j \in \{1,2\}$. A *split* in a connected undirected graph G is a bipartition $\{X,Y\}$ of the vertex set V_G such that $|X|,|Y| \geq 2$, and there exist $X_1 \subseteq X$ and $Y_1 \subseteq Y$ such that $E_G = E_{G[X]} \cup E_{G[Y]} \cup X_1 \times Y_1$ (see Figure 1). A split $\{X,Y\}$ is *strong* if there is no other split $\{X',Y'\}$ such that $\{X,Y\}$ and $\{X',Y'\}$ overlap. Notice that not all graphs have a split. Those that do not have a split are called *prime*.

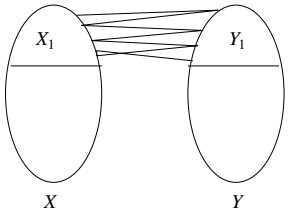

Figure 1. A schematic view of a split.

If $\{X,Y\}$ is a split of an undirected graph G, then we let $\mathcal{G}[X]$ and $\mathcal{G}[Y]$ be respectively $(X \cup \{h_X\}, E_{G[X]} \cup \{xh_X \mid x \in X \text{ and } \exists y \in Y,\ xy \in E_G\})$ and $(Y \cup \{h_Y\}, E_{G[Y]} \cup \{yh_Y \mid y \in Y \text{ and } \exists x \in X,\ xy \in E_G\})$ where h_X and h_Y are new vertices and are called *neighbour markers* of $\mathcal{G}[X]$ and $\mathcal{G}[Y]$. Notice that

given $\mathcal{G}[X]$ and $\mathcal{G}[Y]$ with neighbour markers h_X and h_Y distinguished, we can reconstruct G as follows

$$V_G = (V_{\mathcal{G}[X]} \cup V_{\mathcal{G}[Y]}) \setminus \{h_X, h_Y\},$$
$$E_G = (E_{\mathcal{G}[X]} \cup E_{\mathcal{G}[Y]}) \setminus (\{xh_X \in E_{\mathcal{G}[X]}\} \cup \{xh_Y \in E_{\mathcal{G}[Y]}) \ \cup$$
$$\{xy \mid x \in V_{\mathcal{G}[X]},\ y \in V_{\mathcal{G}[Y]} \text{ and } xh_X \in E_{\mathcal{G}[X]},\ yh_Y \in E_{\mathcal{G}[Y]}\}.$$

Fact 3. *Let $\{X, Y\}$ be a split in a connected undirected graph G and let $P := (x_0, \ldots, x_k)$ be a path in G.*

1. *If $x_0 \in X$ and $x_k \in Y$, then P is a shortest path if and only if there exists $x_i \in P$ such that $(x_0, \ldots, x_i, h_X)$ and $(h_Y, x_{i+1}, \ldots, x_k)$ are shortest paths in $\mathcal{G}[X]$ and $\mathcal{G}[Y]$ respectively.*
2. *If $x_0, x_k \in X$, then P is a shortest path if and only if either P is a shortest path in $\mathcal{G}[X]$ or $(x_0, \ldots, x_{i-1}, h_X, x_{i+1}, \ldots, x_k)$ is a shortest path in $\mathcal{G}[X]$with $x_i \in Y$. Similarly, for $x_0, x_k \in Y$.*

A *decomposition* of a connected undirected graph G is defined inductively as follows: $\{G\}$ is the only decomposition of size 1. If $\{G_1, \ldots, G_n\}$ is a decomposition of size n of G and G_i has a split $\{X, Y\}$, then $\{G_1, \ldots, G_{i-1}, \mathcal{G}_i[X], \mathcal{G}_i[Y], G_{i+1}, \ldots, G_n\}$ is a decomposition of size $n + 1$. Notice that the decomposition process must terminate because the new graphs $\mathcal{G}_i[X]$ and $\mathcal{G}_i[Y]$ are smaller than G_i. The graphs G_i of a decomposition are called *blocks*. If two blocks have neighbour markers, we call them *neighbour blocks*.

A decomposition is *canonical* if and only if: (i) each block is either prime (called *prime block*), or is isomorphic to K_n (called *clique block*) or to a S_n (called *star block*) for $n \geq 3$, (ii) no two clique blocks are neighbour, and (iii) if two star blocks are neighbour, then either their markers are both centers or both not centers.

Theorem 2 ([11, 12])**.** *Every connected undirected graph has a unique canonical decomposition, up to isomorphism. It can be obtained by iterated splitting relative to strong splits. This canonical decomposition can be computed in time $O(\|G\|)$ for every undirected graph G.*

The canonical decomposition of a connected undirected graph G constructed in Theorem 2 is called *split-decomposition* and we will denote it by $\mathcal{D}_G$.

3.2. **Definition of $Bet_{\mathcal{SP}}(x, z, y)$ and The Linear Time Algorithm.** We let A be the set $\{s, \epsilon, \mathbf{V}, \mathbf{M}\}$. For every connected undirected graph G we associate the A-coloured graph $\mathcal{S}_G$ where $V_{\mathcal{S}_G} = \bigcup_{G_i \in \mathcal{D}_G} V_{G_i}$, $E_{\mathcal{S}_G} = \left(\bigcup_{G_i \in \mathcal{D}_G} E_{G_i}\right) \cup \{xy \mid x, y$ are neighbour markers$\}$, a vertex x is labelled $\mathbf{V}$ if and only if $x \in V_G$, otherwise it is labelled $\mathbf{M}$, and an edge xy is labelled s if and only if $xy \in E_{G_i}$ for some $G_i \in \mathcal{D}_G$, otherwise it is labelled ϵ. Figure 2 gives an example of the graph $\mathcal{S}_G$.

A path $(x_0, x_1, \ldots, x_k)$ in $\mathcal{S}_G$ is said *alternating* if, for every $1 \leq i \leq k - 1$, the labels of the edges $x_{i-1}x_i$ and x_ix_{i+1} are different.

Lemma 1. [7] *Let G be a connected undirected graph. Then, $xy \in E_G$ if and only if there exists an alternating path between x and y in $\mathcal{S}_G$. This alternating path is moreover unique.*

Fact 4. *Let $\{X, Y\}$ be a strong split in a connected undirected graph G and let $P := (x_0, \ldots, x_k)$ be a path in $\mathcal{S}_G$.*

1. *If $x_0 \in V_{\mathcal{S}_{\mathcal{G}[X]}}$ and $x_k \in V_{\mathcal{S}_{\mathcal{G}[Y]}}$, then P is a shortest path if and only if there exists $x_i \in P$ such that $(x_0, \ldots, x_i = h_X)$ and $(h_Y = x_{i+1}, \ldots, x_k)$ are shortest paths in $\mathcal{S}_{\mathcal{G}[X]}$ and $\mathcal{S}_{\mathcal{G}[Y]}$ respectively.*
2. *If $x_0, x_k \in V_{\mathcal{S}_{\mathcal{G}[X]}}$ (resp. $x_0, x_k \in V_{\mathcal{S}_{\mathcal{G}[Y]}}$), then P is a shortest path in $\mathcal{S}_G$ if and only if P is a shortest path in $\mathcal{S}_{\mathcal{G}[X]}$ (resp. $\mathcal{S}_{\mathcal{G}[Y]}$).*

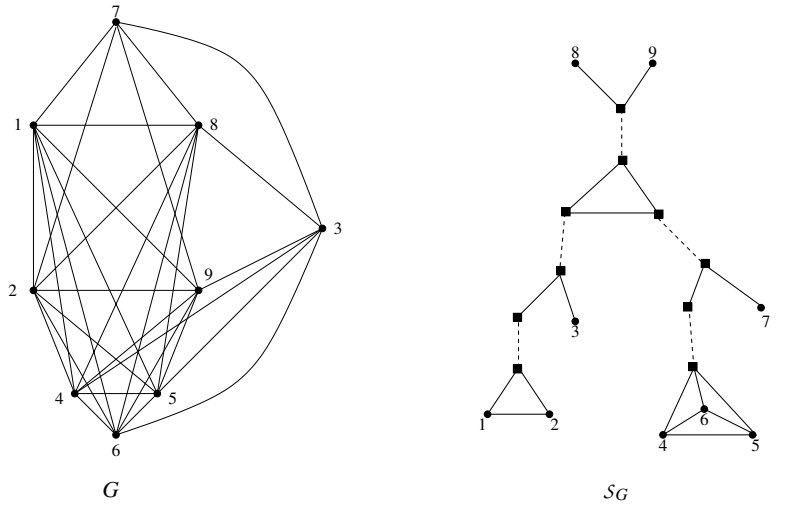

FIGURE 2. Solid lines are edges labelled by s and the dashed ones are labelled by ϵ. The vertices labelled by **V** are those with a number and those labelled by **M** are the rest.

The following gives a characterisation of the property "z is in a shortest path in G between x and y" with respect to shortest paths in $\mathcal{S}_G$.

Proposition 4. *Let G be a connected undirected graph and let x, y, z be vertices of G. Then, $\mathcal{SP}_G(x, z, y)$ holds if and only if*

(i) *$\mathcal{SP}_{\mathcal{S}_G}(x, z, y)$ holds, or*
(ii) *there exists a marker h such that $\mathcal{SP}_{\mathcal{S}_G}(x, h, y)$ holds and there exists an alternating path between h and z in $\mathcal{S}_G$ starting with an edge labelled by ϵ.*

Proof. By induction on the size of $\mathcal{D}_G$. If $\mathcal{D}_G = \{G\}$, then $\mathcal{S}_G = G$ and hence z is in a shortest path between x and y in G if and only if it is in a shortest path between x and y in $\mathcal{S}_G$. Assume now that the statement is true for all graphs with split decomposition of size n and let us prove it for graphs with split decomposition of size $n+1$. Suppose $|\mathcal{D}_G| = n+1$, $n \geq 1$ and let $\{X, Y\}$ be a strong split in G. Then, $\mathcal{D}_G = \mathcal{D}_{\mathcal{G}[X]} \cup \mathcal{D}_{\mathcal{G}[Y]}$, and hence $|\mathcal{D}_G| > |\mathcal{D}_{\mathcal{G}[X]}|, |\mathcal{D}_{\mathcal{G}[Y]}|$.

Case 1. $x \in X$ and $y \in Y$. By Fact 3 z is in a shortest path between x and y in G if and only if z is in a shortest path between x and h_X in $\mathcal{G}[X]$ or in a shortest path between y and h_Y in $\mathcal{G}[Y]$. By inductive hypothesis, z is in a shortest path between x and h_X in $\mathcal{G}[X]$ if and only if one of (i) or (ii) is satisfied. Similarly for z being in a shortest path between h_Y and y in $\mathcal{G}[Y]$. By Lemma 1 and Fact 4 we can conclude the statement of the proposition.

Case 2. $x, y \in X$. We have that z is in a shortest path between x and y in G if and only if z is in X and is in a shortest path between x and y in $\mathcal{G}[X]$, or z is in Y, is adjacent to h_Y and there exists a shortest path in $\mathcal{G}[X]$ between x and y that contains h_X. By inductive hypothesis, $z \in X$ is in a shortest path between x and y if and only if one of (i) or (ii) is satisfied, and since shortest paths in $\mathcal{S}_{\mathcal{G}[X]}$ are shortest paths in $\mathcal{S}_G$ we are done. For $z \in Y$, again by inductive hypothesis h_X is in a shortest path between between x and y in $\mathcal{G}[X]$ if and only if one of (i) or (ii) is satisfied. If (i) is satisfied, then we can clearly conclude. If there exists a marker h in a shortest path between x and y and an alternating path $(h, h_1, \ldots, h_k = h_X)$ with hh_1 labelled by ϵ, then $(h, h_1, \ldots, h_k = h_X, h'_0 = h_Y, \ldots, h'_\ell = z)$ is an alternating

path between h and z if $(h'_0 = h_Y, \ldots, h'_\ell = z)$ is an alternating path. We can therefore conclude the statement.

Case 3. $x, y \in Y$. This case is similar to Case 2 (replace X by Y).

□

We now return to distance-hereditary graphs. We will use the following theorem that characterises distance-hereditary graphs (among the several ones).

Theorem 3 ([2])**.** *A connected undirected graph is a distance-hereditary graph if and only if each block of its split decomposition is either a clique or a star block.*

As a corollary we get the following (the proof is an easy induction on the size of the split decomposition).

Corollary 1. *Let G be a connected distance-hereditary graph. Then, a sequence $P := (x_0, \ldots, x_k)$ is a shortest path in $\mathcal{S}_G$ if and only if P is a chordless path in $\mathcal{S}_G$. Moreover, a shortest path between two vertices in $\mathcal{S}_G$ is unique.*

Proposition 5. *There exists an $MS_{\mathcal{R}_A}$ formula $Bet_{\mathcal{SP}}(x, z, y)$ that expresses that z is in a shortest path between x and y in connected distance-hereditary graphs.*

Proof. Let G be a connected distance-hereditary graph. By Proposition 4 and Corollary 1, we have $\mathcal{SP}_G(x, z, y)$ holds if and only if

(a) z is in a chordless path between x and y in $\mathcal{S}_G$, or
(b) there exists a marker h in a chordless path between x and y in $\mathcal{S}_G$ and there is an alternating path between h and z in $\mathcal{S}_G$ starting with an edge labelled with ϵ.

We now write formulas that describe conditions (a) and (b). Let $\varphi(x, y)$ be the formula $edg_s(x, y) \vee edg_\epsilon(x, y)$. Let the following formulas.

$$\begin{aligned}
\varphi_1(x, X) \Leftrightarrow\ & \text{in the subgraph induced by } X \text{ the vertex } x \text{ has degree 1}\\
\equiv\ & \exists z\Big(z \in X \wedge \varphi(x, z) \wedge \forall t(t \in X \wedge \varphi(x, t) \Longrightarrow z = t)\Big),\\
\varphi_2(x, X) \Leftrightarrow\ & \text{in the subgraph induced by } X \text{ the vertex } x \text{ has degree 2}\\
\equiv\ & \exists z, t \in X\Big(z \neq t \wedge \varphi(x, z) \wedge \varphi(x, t) \wedge \forall z'(z' \in X \wedge \varphi(x, z') \Longrightarrow z' = z \vee z' = t)\Big),\\
\varphi_3(Y, Z, X) \Leftrightarrow\ & \{Y, Z\} \text{ is a partition of } X\\
\equiv\ & \forall x\Big((x \in X \Longleftrightarrow x \in Y \vee x \in Z) \wedge (x \in Y \Longrightarrow x \notin Z) \wedge (x \in Z \Longrightarrow x \notin Y)\Big),\\
\varphi_4(X) \Leftrightarrow\ & \text{the subgraph induced by } X \text{ is connected}\\
\equiv\ & \neg\Big(\exists Y, Z(\varphi_3(Y, Z, X) \wedge \forall y, z(y \in Y \wedge z \in Z \Longrightarrow \neg\varphi(y, z)))\Big),\\
\varphi_5(x, y, X) \Leftrightarrow\ & X \text{ is the set of vertices of a chordless path between } x \text{ and } y\\
\equiv\ & x \in X \wedge y \in X \wedge \varphi_4(X) \wedge \varphi_1(x, X) \wedge \varphi_1(y, X) \wedge\\
& \qquad \forall z\big(z \in X \wedge z \neq x \wedge z \neq y \Longrightarrow \varphi_2(z, X)\big),\\
\varphi_6(x, X) \Leftrightarrow\ & \text{there does not exist } z \text{ and } t \text{ in } X \text{ such that } xz \text{ and } xt \text{ have same labels}\\
\equiv\ & \forall z, t\Big(z \neq t \wedge z \in X \wedge t \in X \wedge \varphi(x, z) \wedge \varphi(x, t) \Longrightarrow\\
& \qquad \neg(edg_s(x, z) \wedge edg_s(x, t)) \wedge \neg(edg_\epsilon(x, z) \wedge edg_\epsilon(x, t)))\Big),\\
\varphi_7(x, y) \Leftrightarrow\ & \text{there exists an alternating chordless path between } x \text{ a marker and } y \text{ a vertex,}\\
& \qquad \text{a path starting with an edge labelled } \epsilon\\
\equiv\ & \exists X\Big(\varphi_5(x, y, X) \wedge \forall z\big(z \in X \wedge z \neq x \wedge z \neq y \Longrightarrow \varphi_6(z, X)\big)\ \wedge\ \exists z\big(z \in X \wedge edg_\epsilon(x, z)\big)\Big).
\end{aligned}$$

The formula $Bet_{\mathcal{SP}}(x,z,y)$ is then the following.

$$nlab_{\mathbf{V}}(x) \wedge nlab_{\mathbf{V}}(y) \wedge nlab_{\mathbf{V}}(z) \wedge$$
$$\exists X \Big(\varphi_5(x,y,X) \wedge \big(z \in X \ \vee \big(\exists h (nlab_{\mathbf{M}}(h) \wedge h \in X \wedge \varphi_7(h,z)) \big) \big) \Big).$$

□

By Corollary 1 the graph obtained from $\mathcal{S}_G$ by forgetting the labels of the vertices and of the edges is a distance-hereditary graph, and then has clique-width at most 3 (see [10]). We can in fact prove that the A-coloured graph $\mathcal{S}_G$ has clique-width at most 3 as stated below (its proof is an easy induction).

Proposition 6. *For every connected distance-hereditary graph G, the A-coloured graph $\mathcal{S}_G$ has clique-width at most 3. Moreover, a clique-width expression of $\mathcal{S}_G$ can be computed in time $O(\|\mathcal{S}_G\|)$.*

Theorem 4. *For every distance-hereditary graph G one can compute in time $O(\|G\|)$ a minimum $\mathcal{SP}_G$-hull set and a minimum $\mathcal{SP}_G$-geodetic set of G.*

Proof. From Fact 1 it is enough to prove the theorem for connected distance-hereditary graphs. So assume that G is connected. By Theorem 3 one can compute in time $O(\|G\|)$ the split decomposition $\mathcal{D}_G$ of G. By [22, Lemma 2.2] we have $\|\mathcal{S}_G\| = O(\|G\|)$, and therefore one can compute $\mathcal{S}_G$ in time $O(\|G\|)$. By Theorem 1 and Propositions 3, 5 and 6, one can compute a minimum $\mathcal{SP}_G$-hull set and a minimum $\mathcal{SP}_G$-geodetic set of G in time $O(\|\mathcal{S}_G\|) = O(\|G\|)$. □

4. Chordal Graphs

In this section we give a polynomial time algorithm for computing a minimum $\mathcal{SP}_G$-hull set in a chordal graph G. We recall that computing a minimum $\mathcal{SP}_G$-geodetic set is NP-complete in chordal graphs. The algorithm for chordal graphs is based on the notion of *functional dependencies*, borrowed from database community. Before introducing them, let us recall some properties of chordal graphs.

Lemma 2. [14] *Every chordal graph has at least two simplicial vertices.*

A *perfect elimination ordering* of a graph G is an ordering $\sigma := (x_1, \ldots, x_n)$ of V_G such that for every $1 \leq i \leq n$, the vertex x_i is simplicial in $G_i^\sigma := G[\{x_i, \ldots, x_n\}]$. The following is a well-known result.

Theorem 5. [21, 23] *A graph G is chordal if and only if it has a perfect elimination ordering. Moreover, a perfect elimination ordering can be computed in time $O(\|G\|)$.*

4.1. **Functional Dependencies and Dependence Graphs.** Let V be a finite set. A *functional dependency* on V is a pair (X, y), often written $X \to y$, where $X \subseteq V$ is called the *premise*, and $y \in V$ is called the *conclusion*. If $X \to z$ is a functional dependency with $X = \{x, y\}$ (or $X = \{x\}$), we will write $xy \to z$ (or $x \to z$) instead of $\{x, y\} \to z$ (or $\{x\} \to z$). A *model* for $X \to y$ is a subset F of V such that $y \in F$ whenever $X \subseteq F$. If $r := X \to y$ is a functional dependency on V, we define the function $\varphi_r : 2^V \to V$ where for every $Z \subseteq V$,

$$\varphi_r(Z) := \begin{cases} Z & \text{if } X \not\subseteq Z, \\ Z \cup \{y\} & \text{otherwise.} \end{cases}$$

A set Σ of functional dependencies on V is called an *implicational system on* V. If an implicational system is composed only of functional dependencies with premises of size 1 we call it a *unit-premise implicational system*. A set $X \subseteq V$ is said Σ-*closed* if it is a model for each functional dependency in Σ. The Σ-*closure* of a set X, noted $\Sigma(X)$, is the smallest Σ-closed set that contains X. A *superkey* for

Σ is a subset K of V such that $\Sigma(K) = V$ and a *key* is an inclusionwise minimal superkey.

Fact 5. *A set $K \subseteq V$ is a superkey for an implicational system Σ on V if and only if there exist functional dependencies $r_1, \ldots, r_p$ in Σ such that $V = \varphi_{r_p}(\ldots(\varphi_{r_1}(K)\ldots))$.*

If Σ is an implicational system on a finite set V, we let $\mathscr{G}(\Sigma)$, called the *dependence graph* of Σ, be the directed graph (V', E') with

$$V' := V \cup \{P_X \mid X \text{ is a premise in } \Sigma\},$$
$$E' := \bigcup_{X \to y \in \Sigma} \Big(\big(\{(x, P_X) \mid x \in X\}\big) \cup \{(P_X, y)\} \Big).$$

One observes that $\mathscr{G}(\Sigma)$ has size $O\left(|V| + \sum_{X \to y \in \Sigma} \big(|X| + 1\big)\right)$ and can be computed in time $O\left(|V| + \sum_{X \to y \in \Sigma} \big(|X| + 1\big)\right)$.

Proposition 7. *Let Σ be a unit-premise implicational system on a finite set V. Then a minimum key of Σ can be computed in time $O(|V| + |\Sigma|)$.*

Proof. Let $\equiv_\Sigma$ be the relation where $x \equiv_\Sigma y$ if $\Sigma(x) = \Sigma(y)$. The relation $\equiv_\Sigma$ is clearly an equivalence relation. Let $P := (V/\equiv_\Sigma, \preceq)$ where $[x]_{\approx_\Sigma} \preceq [y]_{\approx_\Sigma}$ whenever there exist $x' \in [x]_{\approx_\Sigma}$ and $y' \in [y]_{\approx_\Sigma}$ such that $x' \to y' \in \Sigma$. Let M be the minimal elements of $V/\equiv_\Sigma$. Then one easily verifies that the set K obtained by choosing an element in each equivalence class $[x]_{\approx_\Sigma}$ in M is a minimum key.

We now show how to compute the equivalent classes. It is easy to check that $x \equiv_\Sigma y$ if and only if x and y are in the same strongly connected component of $\mathscr{G}(\Sigma)$. The graph $\mathscr{G}(\Sigma)$ can be computed in time $O(|V| + |\Sigma|)$ and has size $O(|V| + |\Sigma|)$. And since we can compute the set of strongly connected components in time $O(\|\mathscr{G}(\Sigma)\|)$ (see [6] for instance), we can then compute the equivalence classes of $\equiv_\Sigma$ in time $O(\|\mathcal{G}(\Sigma)\|) = O(|V| + |\Sigma|)$. And since the minimal elements of $\preceq$ can be computed in $O(|V/\equiv_\Sigma|)$, we are done. □

We now borrow some ideas from [26]. Let Σ be an implicational system on a finite set V. A strongly connected component S of $\mathscr{G}(\Sigma)$ is called a *source component* if for all $x, y \in V_{\mathscr{G}(\Sigma)}$, if $(x, y) \in E_{\mathscr{G}(\Sigma)}$ and $y \in V_S$, then $x \in V_S$. That means that all edges between a vertex in S and a vertex in $V_{\mathscr{G}(\Sigma)} \setminus S$ is always from S.

Let R be a subset of V. We let Σ^R, called the *restriction of Σ to R*, be the implicational system on R defined as $\{X \to y \in \Sigma \mid X \cup \{y\} \subseteq R\}$. A *contraction* of Σ to R is the implicational system $\overline{\Sigma^R}$ on $V \setminus \Sigma(R)$ defined as

$$\big\{X \to y \mid X \cup S \to y \in \Sigma, \text{ and } S \subseteq \Sigma(R) \text{ and } X \cup \{y\} \subseteq V \setminus \Sigma(R)\big\} \bigcup \big\{X \to y \in \Sigma \mid X \cup \{y\} \subseteq V \setminus \Sigma(R)\big\}.$$

Proposition 8. [26] *Let Σ be an implicational system on a finite set V. Let S be a source component of $\mathscr{G}(\Sigma)$. Then each minimum key of Σ is a union of a minimum key of Σ^{V_S} and of a minimum key of $\overline{\Sigma^{V_S}}$. Moreover, all such unions are minimum keys of Σ.*

If Σ is an implicational system on V, then $x \in V$ is called an *extreme point* if x is not the conclusion of any functional dependency in Σ. It is straightforward to verify that any extreme point is a source component in $\mathscr{G}(\Sigma)$ and then is in any minimum key of Σ.

Lemma 3. *Let $\Sigma := \Sigma_1 \cup \Sigma_2$ be an implicational system on a finite set V with $\Sigma_1 := \{z \to t \in \Sigma\}$ and $\Sigma_2 := \{zt \to y \in \Sigma\}$. Let x in V be an extreme point in Σ_2 and not in Σ_1 and let $\Sigma' := \big(\Sigma \setminus \{xz \to y \in \Sigma\}\big) \cup \big(\{tz \to y \mid xz \to y, t \to x \in \Sigma\}\big)$. Then, any minimum key K' of Σ' is a minimum key of Σ. Conversely, to any minimum key K of Σ, one can associate a minimum key K' of Σ'.*

Proof. It is enough to prove that any superkey K' of Σ' is a superkey of Σ, and to any superkey K of Σ, one can associate a superkey K' of Σ' of same size.

Let K' be a key for Σ'. Then, there exist functional dependencies $r'_1, \ldots, r'_p$ in Σ' such that $V = \varphi_{r'_p}(\ldots(\varphi_{r'_1}(K')\ldots))$. For each i let r_i be in Σ where

$$r_i := \begin{cases} r'_i & \text{if } r'_i \in \Sigma, \\ xz \to y & \text{if } r'_i := tz \to y \in \Sigma' \setminus \Sigma. \end{cases}$$

If x is in K', then $V = \varphi_{r_p}(\ldots(\varphi_{r_1}(K')\ldots))$. If x is not in K', then let i be the smallest integer such that $r'_i \in \Sigma' \setminus \Sigma$. Then, $r'_i := tz \to y$ where $t \to x \in \Sigma$. Let $r := t \to x$. Therefore, $V = \varphi_{r_p}(\ldots(\varphi_{r_i}(\varphi_r(\varphi_{r_{i-1}}(\ldots(\varphi_{r_1}(K')\ldots)))))\ldots)$.

Conversely, let K be a key for Σ. Then, there exist functional dependencies $r_1, \ldots, r_p$ in Σ such that $V = \varphi_{r_p}(\ldots(\varphi_{r_1}(K')\ldots))$. For each i let r'_i be in Σ where

$$r'_i := \begin{cases} r_i & \text{if } r_i \in \Sigma', \\ tz \to y & \text{if } r_i := xz \to y \in \Sigma \setminus \Sigma. \end{cases}$$

If x is not in K, then we have clearly $V = \varphi_{r'_p}(\ldots(\varphi_{r'_1}(K')\ldots))$ because there exists some functional dependency $r_i := t \to x$ and if $r_j := xz \to y$, we have $j > i$. We now assume that x is in K. If there exists t in V such that $r' := t \to x$ is in Σ and t is in K, then we have clearly $V = \varphi_{r'_p}(\ldots(\varphi_{r'_1}(K')\ldots))$. Otherwise, let $K' := K \setminus \{x\} \cup \{t\}$ where $r' := t \to x$ is in Σ. Then, we have $V = \varphi_{r'_p}(\ldots(\varphi_{r'_1}(\varphi_{r'}(K'))\ldots))$. This concludes the proof. □

4.2. **The Algorithm for Chordal Graphs.** We will associate with every graph G the implicational system on V_G

$$\Sigma_G := \bigcup_{x,y \in V} \{xy \to z \mid z \in \mathcal{SP}^o_G(\{x,y\}) \setminus \{x,y\}\}.$$

One notices that a vertex x of a graph G is simplicial if and only if x is an extreme point in Σ_G.

Fact 6. *Let G be an undirected graph. Then, $\mathcal{SP}^+_G(X) = \Sigma_G(X)$ for every $X \subseteq V_G$. Then, K is a minimum $\mathcal{SP}_G$-hull set of G if and only if K is a minimum key of Σ_G.*

Theorem 6. *For any chordal graph G, one can compute a minimum $\mathcal{SP}_G$-hull set of G in time $O\left(\|G\| + \sum_{x,y \in V_G} \left|\mathcal{SP}^o_G(\{x,y\})\right|\right)$, which is less than $O(|V_G|^3)$.*

Proof. Let G be a chordal graph. By Fact 1 it is enough to prove that we can compute a minimum $\mathcal{SP}_G$-hull set in each connected component of G. For that, we will show that Algorithm 1 computes a minimum $\mathcal{SP}_G$-hull set for any connected chordal graph G in time $O\left(\|G\| + \sum_{x,y \in V_G} \left|\mathcal{SP}^o_G(\{x,y\})\right|\right)$.

The important variables of the algorithm are K, K^+, S and Σ with $K^+ = \Sigma_G(K)$, and S is a minimum key of Σ. Since the algorithm returns $K \cup S$, we need to prove that $K \cup S$ is a minimum key for Σ_G. This claim is trivially true before entering the for loop Σ is initialised to Σ_G and the other variables to $\emptyset$. We will

assume that it is true for some iteration i and will prove that it is still true for the iteration $i+1$.

One first notices that at any iteration i of the for loop, only vertices in $V_{G_i^\sigma} \setminus K^+$ can appear as premises in the functional dependencies $\{xy \to z \in \Sigma\}$. One also easily proves that at any time K^+ is equal to $\Sigma_G(K)$. We now prove that after iteration $i+1$, the set $K \cup S$ is a minimum key for Σ_G.

If x_{i+1} is in the closure of K, then it cannot be in any functional dependency and then cannot be in any key of Σ. And then we are done. So, we will assume that x_{i+1} is not in $K^+ = \Sigma_G(K)$.

At iteration $i+1$, the vertex x_{i+1} is a simplicial vertex in G_{i+1}^σ. We will first prove that there exists no functional dependency of the form $zt \to x_{i+1}$ in Σ. Assume there exists $z, t \in V_G \setminus \Sigma_G(K)$ such that x_{i+1} is in a shortest path between them and let P be such a path. P cannot be included in G_{i+1}, otherwise x_{i+1} would not be a simplicial vertex in G_{i+1}. Let $j < i+1$ be the least number such that x_j is in P. Then, P is included in G_j. But, this should contradict the fact that x_j is simplicial in G_j. So, there is no functional dependency of the form $zt \to x_{i+1}$ in Σ. Now, if x_{i+1} is not the conclusion of any functional dependency in Σ, that means that it is an extreme point in Σ and then must be in any key of Σ. Then by Proposition 8, $\{x_{i+1}\} \cup S$ where S is a key of $\overline{\Sigma^{x_{i+1}}}$ is key for Σ. By induction, $K \cup \{x_{i+1}\} \cup S$ is a key for Σ_G and then we are done. Assume now that there exists a functional dependency $t \to x_{i+1}$ in Σ and let $\Sigma' := \big(\Sigma \setminus \{x_i z \to t\}\big) \cup \big(\{tz \to y \mid x_i z \to y, t \to x_i \in \Sigma\}\big)$. By Lemma 3 any minimum key of Σ' is a minimum key of Σ and to any minimum key of Σ one can associate a minimum key of Σ'. Therefore, after iteration $i+1$ the set $K \cup S$ is a minimum key for Σ_G where S is a minimum key for Σ.

The fact that after the for loop, Σ is a unit-premise implicational system follows from the fact that the functional dependencies in Σ with premises of size 2 have their premises included in G_i^σ.

We now discuss the time complexity. By Proposition 2 The implicational system Σ_G can be constructed in time $O\left(\|G\| + \sum_{x,y \in V_G} \left|\mathcal{SP}_G^o(\{x,y\})\right|\right)$. The test in Line 6 can be done in time $O(\log(|V_G|))$ by implementing K^+ as a binary search tree (or any other data structure that allows logarithmic search and insertion operations). Lines 7, 9, 10 and 11 are done in time $O(|\Sigma_G|)$. Line 13 is done in time $O(|V_G| + |\Sigma_G|)$. Since $|\Sigma_G| = O\left(\sum_{x,y \in V_G} \left|\mathcal{SP}_G^o(\{x,y\})\right|\right)$, then we can conclude that the algorithm computes a minimum $\mathcal{SP}_G$-hull set in time $O\left(\|G\| + \sum_{x,y \in V_G} \left|\mathcal{SP}_G^o(\{x,y\})\right|\right)$. □

5. Concluding Remarks

In this paper we have given a linear time algorithm and a cubic-time algorithm for computing a minimum $\mathcal{SP}_G$-hull set in distance-hereditary and chordal graphs respectively. The techniques used to get these two algorithms are different from the ones known in the literature, specially those techniques based on functional dependencies borrowed from database community. We hope these techniques will be fruitful to obtain new algorithms for graph classes where the complexity of computing a minimum $\mathcal{SP}_G$-hull set is open. We can cite among them graph classes of bounded tree-width and more generally those of bounded clique-width, weakly chordal graphs, degenerate graphs, ...

Algorithm 1: ChordalMinimumHullSet(G)

Data : A connected chordal graph G
Result: A $\mathcal{SP}_G$-hull set of minimum size of G
begin

Let $\sigma := (x_1, \ldots, x_n)$ be a perfect elimination scheme of G
$K := \emptyset$
$K^+ := \emptyset$
$\Sigma := \Sigma_G$
for $i \leftarrow 1$ ***to*** n **do**
  if $\big(x_i \notin K^+\big)$ **then**
    if $\big(x_i \notin \{x \mid z \to x \in \Sigma\}\big)$ **then**
      $K := K \cup \{x_i\}$
      $K^+ := K^+ \cup \Sigma(\{x_i\})$
      $\Sigma := \overline{\Sigma^{x_i}}$
    else
      $\Sigma := \Sigma \setminus \{x_i z \to t\} \cup \{tz \to y \mid x_i z \to y, t \to x_i \in \Sigma\}$
  //`At this step` Σ `is a unit-premise implicational system`
  if $\Sigma \neq \emptyset$ **then**
    compute a minimum key S of Σ
  else
    $S := \emptyset$
  return $K \cup S$
end

Our techniques for distance-hereditary graphs allowed at the same time to derive a linear time algorithm for computing a minimum $\mathcal{SP}_G$-geodetic set of a distance-hereditary graph. In fact the whole ingredient in Theorem 4 was the possibility to encode the betweenness relation $\mathcal{SP}_G$ in a distance-hereditary graph by a monadic second-order formula and then use Courcelle et al.'s theorem [8, 9]. A natural question that arises is whether we can extend this encoding to other graph classes. Indeed, we can for all graphs by using the dependence graphs as stated below.

Proposition 9. *There exists a formula $Bet_{\mathcal{SP}}(x, z, y)$ that is true in the dependence graph of $\mathcal{G}(\Sigma_G)$ if and only if $\mathcal{SP}_G(x, z, y)$ holds.*

Proof. Let A be the set $\{\mathbf{V}, \mathbf{M}\}$. For every undirected graph G, let $\mathcal{G}'(\Sigma_G)$ be the A-coloured graph obtained from $\mathcal{G}(\Sigma_G)$ by labelling vertices of V_G by $\mathbf{V}$ and the others by $\mathbf{M}$. One easily checks that the following formula $Bet_{\mathcal{SP}}(x, z, y)$ is true in $\mathcal{G}'(\Sigma_G)$ if and only if $\mathcal{SP}_G(x, z, y)$ holds

$$nlab_{\mathbf{V}}(x) \wedge nlab_{\mathbf{V}}(y) \wedge nlab_{\mathbf{V}}(z) \wedge \exists t \Big(nlab_{\mathbf{M}}(t) \wedge edg(x, t) \wedge edg(y, t) \wedge edg(t, z) \Big).$$

□

A consequence of Theorem 1, and of Propositions 3 and 9 combined with results in [24] gives the following.

Proposition 10. *Let k be a fixed integer. Let G be an undirected graph such that $\mathcal{G}(\Sigma_G)$ has clique-width at most k. Then, one can compute in time $O(|V_G|^6)$ a minimum $\mathcal{SP}_G$-hull set and a minimum $\mathcal{SP}_G$-geodetic set of G.*

A question that arises then is which graphs have dependence graphs of bounded clique-width?

We conclude by pointing out that the logical tools can be used for computing minimum B-hull sets and B-geodetic sets for other betweenness relations B. For instance, for an undirected graph G, we let $\mathcal{P}_G$ be the betweenness relation where $\mathcal{P}_G(x, z, y)$ holds if and only if z is in a chordless path between x and y. It is known that a minimum $\mathcal{P}_G$-hull set and a minimum $\mathcal{P}_G$-geodetic set of a chordal graph can be computed in polynomial time (see [20]). The following proves that it is the case for graphs of bounded clique-width.

Proposition 11. *Let k be a fixed positive integer. Let G be an undirected graph of clique-width at most k. Then, one can compute in time $O(|V_G|^3)$ a minimum $\mathcal{P}_G$-hull set and a minimum $\mathcal{P}_G$-geodetic set of G.*

Proof. In the proof of Proposition 5 we have given the formula $\varphi_5(x, y, X)$ that holds if and only if X is the set of vertices of a chordless path between x and y. So, the following formula holds if and only if z is in a chordless path between x and y

$$Bet_{\mathcal{P}}(x, z, y) \equiv \ \exists X \Big(\varphi_5(x, y, X) \ \wedge \ z \in X \Big).$$

Proposition 3 and Theorem 1 combined with results in [24] conclude the statement. □

References

[1] Julio Araújo, Victor Campos, Frédéric Giroire, Nicolas Nisse, Leonardo Sampaio, and Ronan Pardo Soares, *On the hull number of some graph classes*, Tech. report, 2011.

[2] André Bouchet, *Transforming trees by successive local complementations*, J. Graph Theory **12** (1988), no. 2, 195–207.

[3] Gary Chartrand, John Frederick Fink, and Ping Zhang, *The hull number of an oriented graph*, International Journal of Mathematics and Mathematical Sciences **2003** (2003), no. 36, 2265–2275.

[4] Vašk Chvátal, *Antimatroids, betweenness, convexity*, Research Trends in Combinatorial Optimization (William J. Cook, László Lovász, and Jens Wygen, eds.), Springer, 2009, pp. 57–64.

[5] E. F. Codd, *Further normalization of the data base relational model*, Data Base Systems (R. Rustin, ed.), Prentice Hall, Englewood cliffs, NJ, 1972, pp. 33–64.

[6] T.H. Cormen, C.E Leiserson, R.L. Rivest, and C. Stein, *Introduction to algorithms*, 2nd ed., the MIT Press, 2002.

[7] Bruno Courcelle, *The monadic second-order logic of graphs XVI : Canonical graph decompositions*, Logical Methods in Computer Science **2** (2006), no. 2.

[8] Bruno Courcelle and Joost Engelfriet, *Graph structure and monadic second-order logic: a language theoretic approach*, Encyclopedia of Mathematics and its Applications, vol. 138, Cambridge University Press, 2012.

[9] Bruno Courcelle, Johann A. Makowsky, and Udi Rotics, *Linear time solvable optimization problems on graphs of bounded clique-width*, Theory Comput. Syst. **33** (2000), no. 2, 125–150.

[10] Bruno Courcelle and Stephan Olariu, *Upper bounds to the clique width of graphs*, Discrete Applied Mathematics **101** (2000), no. 1-3, 77–114.

[11] William H. Cunnigham and Jack Edmonds, *A combinatorial decomposition theory*, Canadian Journal of Mathematics **32** (1980), 734–765.

[12] Elias Dahlhaus, *Parallel algorithms for hierarchical clustering and applications to split decomposition and parity graph recognition*, J. Algorithms **36** (2000), no. 2, 205–240.

[13] Reinhardt Diestel, *Graph theory*, 3rd ed., Springer-Verlag, 2005.

[14] G.A. Dirac, *On rigid circuit graphs*, Abhandlungen Aus Dem Mathematischen Seminare der Universität Hamburg **25** (1961), no. 1-2, 71–76.

[15] Mitre Costa Dourado, John G. Gimbel, Jan Kratochvíl, Fábio Protti, and Jayme Luiz Szwarcfiter, *On the computation of the hull number of a graph*, Discrete Mathematics **309** (2009), no. 18, 5668–5674.

[16] Mitre Costa Dourado, Fábio Protti, Dieter Rautenbach, and Jayme Luiz Szwarcfiter, *On the hull number of triangle-free graphs*, SIAM J. Discrete Math. **23** (2010), no. 4, 2163–2172.
[17] ———, *Some remarks on the geodetic number of a graph*, Discrete Mathematics **310** (2010), no. 4, 832–837.
[18] Tinaz Ekim, Aysel Erey, Pinar Heggernes, Pim van 't Hof, and Daniel Meister, *Computing minimum geodetic sets of proper interval graphs*, LATIN (David Fernández-Baca, ed.), Lecture Notes in Computer Science, vol. 7256, Springer, 2012, pp. 279–290.
[19] Martin G. Everett and Stephen B. Seidman, *The hull number of a graph*, Discrete Mathematics **57** (1985), no. 3, 217–223.
[20] Martin Farber and Robert E. Jamison, *Convexity in graphs and hypergraphs*, SIAM Journal on Algebraic and Discrete Methods **7** (1986), 433–444.
[21] D.R. Fulkerson and O.A. Gross, *Incidence Matrices and Interval Graphs*, Pacific J. Math. **15** (1965), no. 3, 835–855.
[22] Cyril Gavoille and Christophe Paul, *Distance labeling scheme and split decomposition*, Discrete Mathematics **273** (2003), no. 1-3, 115–130.
[23] Michel Habib, Ross M. McConnell, Christophe Paul, and Laurent Viennot, *Lex-bfs and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing*, Theor. Comput. Sci. **234** (2000), no. 1-2, 59–84.
[24] Petr Hlinený and Sang il Oum, *Finding branch-decompositions and rank-decompositions*, SIAM J. Comput. **38** (2008), no. 3, 1012–1032.
[25] Ignacio M. Pelayo, *On convexity in graphs*, Tech. report, 2004.
[26] Hossein Saiedian and Thomas Spencer, *An efficient algorithm to compute the candidate keys of a relational database schema*, Comput. J. **39** (1996), no. 2, 124–132.

Clermont-Université, Université Blaise Pascal, LIMOS, CNRS, France