A structural decomposition for hypergraphs

Cohen, David A.; Jeavons, Peter G. & GYSSENS, Marc (1994) A structural decomposition for hypergraphs. In: Barcelo, Hélène; Kalai, Gil (Ed.). Jerusalem Combinatorics '93, p. 161-177.

DOI: 10.1090/conm/178
Handle: http://hdl.handle.net/1942/13464

Contemporary Mathematics
Volume 00, 0000

A Structural Decomposition for Hypergraphs

PETER JEAVONS, DAVID COHEN, AND MARC GYSSENS

ABSTRACT. Hypergraphs arise in a variety of applications and are commonly classified as cyclic or acyclic. In this paper we develop a more refined classification scheme for cyclic hypergraphs based on a natural decomposition strategy. The fundamental building blocks in our decompositions are subsets of edges known as k-hinges. For any hypergraph, a set of more than k of its edges is defined to be a k-hinge if all connected components of the hypergraph with respect to the set of edges meet the latter within at most k of its edges. A k-hinge tree is a set of minimal k-hinges that cover all edges of H, and form a tree with respect to intersection.

The size of the largest node in any 1-hinge tree is shown to be an invariant of the hypergraph, which we call the degree of cyclicity. Acyclic hypergraphs are hypergraphs with degree of cyclicity 2. The concept of degree of cyclicity was first presented by Gyssens in the context of relational database design, but is presented here for arbitrary hypergraphs with a greatly simplified proof. For more general k-hinges we show that it is possible to obtain more powerful decompositions. However, in this case there may be several possible decompositions which do not share any structural invariant. We therefore consider restrictions on k-hinges which are necessary in order to guarantee this structural invariant.

1. Introduction

The complexity of any problem associated with a graph may depend very strongly on the structure of that graph. Examples of such problems are finding a minimal triangulation of a graph [**11**], coloring problems, and more general constraint satisfaction problems [**6**].

Problems which are computationally hard for general *graphs* may be tractable for particular classes of graphs with some restriction on the structure (for a survey, see [**1**]). In particular, if the graph may be decomposed into suitable subgraphs (e.g., the biconnected components) in such a way that the problem can be solved for each subgraph separately, and the solutions to these subproblems can

1991 *Mathematics Subject Classification.* 05C65, 05C85; Secondary 68P15, 68R15, 68T20.
This research was supported by the "British-Flemish Academic Research Collaboration Programme" of the British Council and the Belgian National Fund of Scientific Research.

be efficiently combined to obtain an overall solution, then significant reductions in complexity may be achieved.

The corresponding theory for general *hypergraphs* is much less well-developed. It has been shown that many problems possess efficient solution techniques when the associated hypergraph is *acyclic* [**2, 3**]. However, the classification of hypergraphs into acyclic and cyclic ones is rather crude.

A more powerful decomposition strategy for hypergraphs was developed in the context of a particular application in database theory. This approach led to a hierarchy of classes of hypergraphs in which the acyclic hypergraphs are merely the smallest non-trivial class [**9, 10, 7**]. It has been shown that in the special case of graphs this approach gives rise to a finer decomposition than the decomposition into biconnected components, and therefore provides a more informative bound on the complexity of many problems [**7, 8**].

In this paper, we develop a very general decomposition strategy for hypergraphs in the context of general hypergraph theory and discuss its properties. The paper is organized as follows. In Section 2, we introduce some terminology and define hypergraph decompositions in terms of *k-hinges* and *k-hinge-trees*. In Section 3, we review the desirable properties of 1-hinges and 1-hinge-trees that motivated this study and propose a new, short, and more insightful proof of the main property. The remaining sections are devoted to general k-hinges. In Section 4, we demonstrate the usefulness of general k-hinge-tree decompositions. In Section 5 we discuss the difficulties that arise when trying to build a general theory for constructing arbitrary k-hinge-tree decompositions, and the open problems resulting from them. Most of the difficulties arise from the fact that the desirable properties for 1-hinges exhibited in Section 3 do not carry over to arbitrary k-hinges, even for $k = 2$.

2. Definitions

The building blocks of the decomposition we intend to propose are so-called *k-hinges*, which are defined in this section. Our notion of hypergraph decomposition is then formalized by the concept of *k-hinge-trees*.

First though, we briefly review some of the relevant terminology concerning hypergraphs.

DEFINITION 2.1. [**4**] A *hypergraph* is an ordered pair (V, E) where V is a finite set of vertices and E is a set of edges, each of which is a subset of V.

Undirected graphs can be seen as special cases of hypergraphs, where each edge contains exactly two vertices.

DEFINITION 2.2. Let (V, E) be a hypergraph, let $H \subseteq E$, and let $F \subseteq E - H$. F is called *connected with respect to* H if, for any two edges $e, f \in F$, there exists a sequence $e_1, \ldots, e_n$ of edges in F such that *(i)* $e_1 = e$; *(ii)* for $i = 1, \ldots, n-1$, $e_i \cap e_{i+1} \not\subseteq \bigcup H$; and *(iii)* $e_n = f$. Such a sequence is called a *path with respect to* H connecting e and f in F.

The maximal connected subsets of $E - H$ with respect to H are called the *connected components of $E - H$ with respect to H*. They obviously form a partition of $E - H$. In the case that H is the empty set of edges, we usually drop the phrase "with respect to H" and simply speak about connected subsets and connected components of E. If E itself is connected, then (V, E) is said to be a *connected hypergraph*. In this paper, we shall assume that all hypergraphs to be decomposed are connected, since in a disconnected hypergraph each connected component may be decomposed individually.

Using this notion of connectivity we now define the sets of edges which will be called *k-hinges*:

DEFINITION 2.3. Let (V, E) be a hypergraph, and let H be either E or a proper subset of E containing at least $k + 1$ edges. Let $H_1, \ldots, H_m$ be the connected components of $E - H$ with respect to H. Then H is called a *k-hinge* if, for $i = 1, \ldots, m$, there exists a set $h_i \subseteq H$ consisting of k edges such that

$$\left(\bigcup H_i\right) \cap \left(\bigcup H\right) \subseteq \bigcup h_i.$$

The set of edges h_i is called a *separating edge set* for H_i.

In words, a k-hinge of a hypergraph is either the entire hypergraph or a subset H of its edges (containing at least $k + 1$ edges) with the property that the set of nodes in each connected component with respect to that subset intersects the set of nodes in H within at most k edges.

In general, k-hinges can in turn contain other k-hinges. We are most interested in k-hinges that do *not* contain other k-hinges; these are called *minimal k-hinges*.

Our aim is to decompose any hypergraph into a "tree" of minimal k-hinges, with the following properties:

DEFINITION 2.4. Let (V, E) be any hypergraph. A *k-hinge-tree* of (V, E) is a tree[1], (N, A), with nodes N and labeled arcs[2] A, such that

(i) each node in N is a minimal k-hinge of (V, E), and each label of an arc in A is a subset of E;

(ii) $\bigcup N = E$;

(iii) for each labeled arc $(\{n_i, n_j\}, E') \in A$, $E' = n_i \cap n_j$ and satisfies $(\bigcup n_i) \cap (\bigcup n_j) = \bigcup E'$ and $1 \leq |E'| \leq k$; and

(iv) the vertices of V shared by two tree nodes are entirely contained within each tree node on their connecting path.

Property 3 states that adjacent tree nodes share at most k edges of E, and this set of edges is also the label of their connecting tree-arc; moreover, the vertices shared by the adjacent tree nodes are precisely the vertices of this edge set.

Note that a hypergraph may, in general, have many different k-hinge-trees, containing different sets of minimal k-hinges.

Below, we give an example of a k-hinge-tree of a hypergraph for $k = 1$.

---

[1] By "tree", we understand an unrooted tree, i.e., an undirected acyclic graph.

[2] A labeled arc is formally defined as an ordered pair $(\{n_i, n_j\}, a)$ with n_i and n_j different tree nodes and a the arc-label.

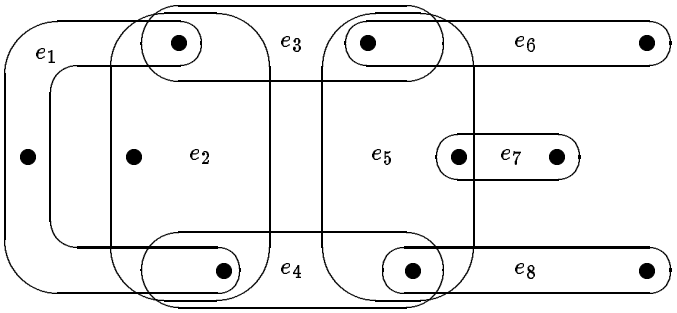

FIGURE 1. A hypergraph (Example 2.1).

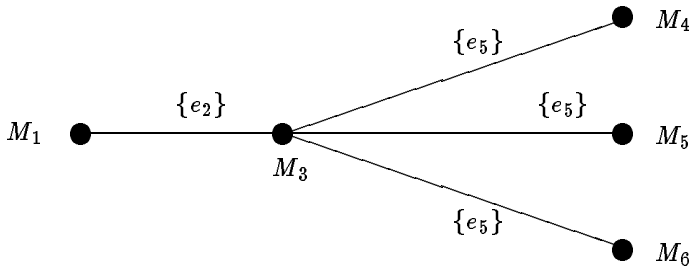

FIGURE 2. A 1-hinge-tree of the hypergraph in Figure 1.

EXAMPLE 2.1. Consider the hypergraph illustrated in Figure 1. The minimal 1-hinges of the hypergraph in Figure 1 are the following:

$$M_1 = \{e_1, e_2\}; \qquad M_4 = \{e_5, e_6\};$$
$$M_2 = \{e_1, e_3, e_4, e_5\}; \qquad M_5 = \{e_5, e_7\};$$
$$M_3 = \{e_2, e_3, e_4, e_5\}; \qquad M_6 = \{e_5, e_8\}.$$

Figure 2 illustrates one possible 1-hinge-tree of the hypergraph in Figure 1.

An alternative 1-hinge-tree of (V, E) may be obtained by replacing M_3 with M_2 and labeling the arc connecting it to M_1 with $\{e_1\}$.

Definition 2.4 is partly inspired by the notion of tree developed by Courcelle [**5**] in the context of (binary) graphs, and is a generalization of the notion of "hinge-tree" in [**7, 8**] (which correspond to 1-hinge-trees in this paper). The introduction of general k-hinge-trees was motivated by the observation that, while 1-hinge-tree decompositions have many desirable properties with respect to solving problems defined on a hypergraph, many hypergraphs do not possess non-trivial 1-hinge-tree decompositions but do possess interesting k-hinge tree decompositions for higher values of k.

3. Decompositions using 1-hinges

The concept of 1-hinge was introduced in [**9**] under the name "hinge" in the context of relational database theory. It was subsequently found that every hypergraph has at least one 1-hinge-tree, and that the size of the largest node in any 1-hinge-tree decomposition is an invariant of the hypergraph, i.e., independent of the particular 1-hinge-tree chosen [**7**]. This result has subsequently been applied to the analysis of constraint satisfaction problems [**8**].

Unfortunately, the initial proof in [**7**] heavily depends on properties of certain database concepts and therefore is not suitable for possible generalization to decomposition into arbitrary k-hinge-trees. Therefore, we exhibit a simpler proof stated entirely within the context of general hypergraph theory.

In Section 3.1, we describe two key properties of 1-hinges. Using these key properties, we then prove the actual invariance result in Section 3.2. For completeness, an algorithm to find a particular 1-hinge-tree for any hypergraph is given in Section 3.3. Because of the invariance result, the particular 1-hinge-tree found by this algorithm is optimal.

3.1. Properties of 1-hinges. The first of the two key properties of 1-hinges we shall exhibit in this section was originally proved in [**9**] by a straightforward consideration of the possible connected components with respect to H' and will here be referred to as the "Inheritance Property:"

INHERITANCE PROPERTY. [**9**] *Let (V, E) be a hypergraph, let H be a 1-hinge of (V, E), and let $H' \subseteq H$. Then H' is a 1-hinge of (V, E) if and only if it is a 1-hinge of the hypergraph $(\bigcup H, H)$.*

To state the second property, which we shall call the "Decomposition Property," we first propose a notion of decomposition in Definition 3.1. With a possible generalization in mind, we state Definition 3.1 in a slightly more general fashion than strictly needed in this section.

DEFINITION 3.1. Let (V, E) be a hypergraph, let H be a subset of edges, and let h be a subset of E. An *h-decomposition* of H is any set, $\mathcal{D}$, of subsets of H, such that $H = \bigcup \mathcal{D}$ and, for all $S, T \in \mathcal{D}$, $S \neq T$, $(\bigcup S) \cap (\bigcup T) \subseteq \bigcup h$.

DECOMPOSITION PROPERTY. *Let (V, E) be a hypergraph, and let H be a 1-hinge of (V, E). Let e be an arbitrary edge in E, and let $\mathcal{D}$ be an $\{e\}$-decomposition of H. There exists $S \in \mathcal{D}$ such that, for every $\mathcal{D}' \subseteq \mathcal{D}$ with $S \in \mathcal{D}'$, $\bigcup \mathcal{D}'$ is either a 1-hinge of (V, E) or a single edge in E.*

PROOF. If $e \notin H$, let C_e be the connected component of $E - H$ with respect to H which contains e, and let $\{f\}$ be a corresponding singleton separating edge set in H. If $e \in H$, let $C_e = \emptyset$, and let $f = e$. In either case, let S be an element of $\mathcal{D}$ containing f. Let $H' = H \cup C_e$. Note that, as H' is a 1-hinge of (V, E), every 1-hinge of $(\bigcup H', H')$ is also a 1-hinge of (V, E), by the Inheritance Property.

We have

$$\begin{aligned}
&\left(\bigcup \mathcal{D}'\right) \cap \left(\bigcup (H' - (\bigcup \mathcal{D}'))\right) \\
&\qquad = \left[\left(\bigcup \mathcal{D}'\right) \cap \left(\bigcup (H' - H)\right)\right] \cup \left[\left(\bigcup \mathcal{D}'\right) \cap \left(\bigcup (H - (\bigcup \mathcal{D}'))\right)\right] \\
&\qquad \subseteq \left[\left(\bigcup H\right) \cap \left(\bigcup C_e\right)\right] \cup \left[\left(\bigcup \mathcal{D}'\right) \cap \left(\bigcup (H - (\bigcup \mathcal{D}'))\right) \cap \left(\bigcup H\right)\right] \\
&\qquad \subseteq f \cup \left(e \cap \left(\bigcup H\right)\right) \\
&\qquad \subseteq f
\end{aligned}$$

Hence if $S \in \mathcal{D}'$, then $\bigcup \mathcal{D}'$ is either $\{f\}$ or a 1-hinge of $(\bigcup H', H')$, in which case it is a 1-hinge of (V, E). □

The Decomposition Property has the following immediate corollary:

INSEPARABILITY PROPERTY. *Let (V, E) be a hypergraph and let H be a minimal 1-hinge of (V, E). Let e be an arbitrary edge in E and let $\mathcal{D}$ be an $\{e\}$-decomposition of H. Then $\mathcal{D}$ has at most two nonempty elements, and at most one element containing more than one edge.*

3.2. An invariant of 1-hinge-trees. Using the properties described above, we can now prove the following:

PROPOSITION 3.1. *Let (V, E) be a hypergraph and let $T = (N, A)$ be a 1-hinge-tree for (V, E). For any minimal 1-hinge H of (V, E), $|H| \leq max\{|n| \mid n \in N\}$.*

PROOF. Every arc of T divides T into two subtrees, and by Properties 3 and 4 in Definition 2.4, the partition of H defined by these subtrees is an $\{e\}$-decomposition of H. By the minimality of H and the Inseparability Property, one of the two subsets in this partition contains at most one edge of H. Hence we may orient each arc of T towards a subtree containing at most one edge of H.[3]

As T is now a directed tree, there is some node n_c with in-degree zero. By the choice of orientation, each branch out of n_c contains at most one edge of H.[4]

If there are two branches out of n_c both labeled with the same edge $\{e\}$ and each containing a distinct single edge of H, then we have an $\{e\}$-decomposition of H containing two singleton sets, so the Inseparability Property implies that $|H| = 2$, whence the result holds trivially.

Otherwise, all branches out of n_c labeled with the same edge contain at most one edge of H between them. By Property 3 of Definition 2.4, if $\{e\}$ is the label of the arc connecting a branch out of n_c to n_c, then e is the only edge contained in both n_c and the branch. Hence, if $e \notin H$ and if the branches connected to n_c by arcs labeled $\{e\}$ do contain an edge of H between them, then that edge cannot be contained in n_c. Let L denote the set of edges that is the union of the labels of the arcs connecting branches to n_c. By the preceding argument, it

[3] When we say that a subtree contains an edge we mean that that subtree has a node containing that edge.

[4] The branches are defined not to contain the central node.

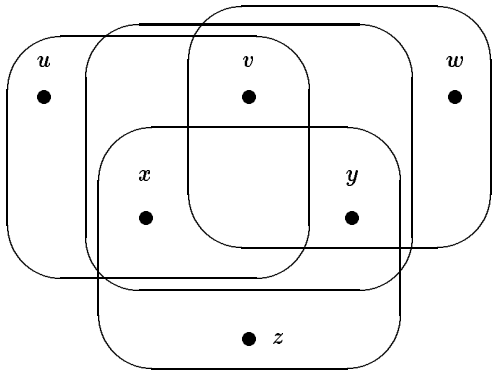

FIGURE 3. The hypergraph (V, E') (Example 3.1).

follows that $|L - H| \geq |H - n_c|$. Since $L \subseteq n_c$, $|n_c - L| \geq |H - n_c|$, whence $|n_c - H| \geq |H - n_c|$, whence $|n_c| \geq |H|$, whence the theorem. □

INVARIANCE THEOREM. *For any hypergraph, (V, E), the size of the largest node in a 1-hinge-tree of (V, E) is an invariant.*

PROOF. Let (V, E) be a hypergraph, and let T be any 1-hinge-tree of (V, E). By Proposition 3.1, the size of the largest node in T is equal to the size of the largest minimal 1-hinge of (V, E). □

The size of the largest minimal 1-hinge of a hypergraph will be referred to as the *degree of cyclicity*. A hypergraph will be called *n-cyclic* if its degree of cyclicity is less than or equal to n. Acyclicity [**2**] is equivalent to 2-cyclicity [**7**].

EXAMPLE 3.1. First consider the hypergraph in Figure 1, Example 2.1. As the largest node in the 1-hinge-tree illustrated in Figure 2 has size 4, the hypergraph is 4-cyclic. Notice that the largest node of the other 1-hinge-tree described in Example 1 also has size 4, in accordance with the Invariance Theorem.

Next, consider the hypergraph (V, E) with 6 vertices, u, v, w, x, y, and z, and 3 edges, $\{x, y, z\}$, $\{v, w, y\}$, and $\{u, v, x\}$. The only 1-hinge of (V, E) is the set of all 3 edges, so (V, E) is 3-cyclic (and n-cyclic for all $n \geq 3$). If we add an additional edge, $\{v, x, y\}$, we obtain a new hypergraph, (V, E'), as shown in Figure 3.

The hypergraph (V, E') contains three 1-hinges of size 2:

$$\{\{v, x, y\}, \{u, v, x\}\}; \{\{v, x, y\}, \{v, w, y\}\}; \text{ and } \{\{v, x, y\}, \{x, y, z\}\}.$$

Any tree with these three nodes and with arcs labeled $\{v, x, y\}$, is a 1-hinge-tree of (V, E'), showing that (V, E') is 2-cyclic (i.e., acyclic).

This example illustrates the surprising fact that the *addition* of an edge may *reduce* the degree of cyclicity, and may even turn a cyclic hypergraph into an acyclic hypergraph.

3.3. An algorithm for 1-hinge-trees. Given a hypergraph (V, E), a 1-hinge-tree may be obtained in polynomial time by the following algorithm [**8**]:

HINGE-TREE DECOMPOSITION ALGORITHM.

Input: A hypergraph (V, E).

Output: A 1-hinge-tree T for (V, E).

Method:

(i) Mark each edge in E as *unused*. Set $i = 0, N_0 = \{E\}$ and $A_0 = \emptyset$, and mark the node E in N_0 as *non-minimal*.

(ii) If all nodes of N_i are marked *minimal*, then set $T = (N_i, A_i)$ and stop. Else, choose a *non-minimal* node F in N_i.

(iii) If all edges in F are marked *used*, then mark F as *minimal* and return to 2. Else, choose an *unused* edge $e \in F$ and mark e as *used*.

(iv) Let $\Gamma = \{G \cup \{e\} \mid G \text{ is a connected component of } F - \{e\} \text{ w.r.t. } e\}$, and let $\gamma : F \to \Gamma$ be any function such that for all $f \in F$, $f \in \gamma(f)$. If $|\Gamma| = 1$, then return to 3.

(v) Set $N_{i+1} = (N_i - \{F\}) \cup \Gamma$ and

$$\begin{aligned} A_{i+1} = (A_i - \{(\{F, F'\}, \{f\}) \mid (\{F, F'\}, \{f\}) \in A_i\}) \\ \cup \{(\{\gamma(f), F'\}, \{f\}) \mid (\{F, F'\}, \{f\}) \in A_i\} \\ \cup \{(\{\gamma(f), \gamma(e)\}, \{e\}) \mid f \in F,\ \gamma(f) \neq \gamma(e)\}, \end{aligned}$$

and mark all the new nodes added to N_{i+1} as *non-minimal*.

(vi) Increment i and return to 2.

The correctness of the algorithm relies on the Inheritance Property, the Decomposition Property, and the Inseparability Property.

At Step 3, we always have a set of nodes N_i which form a tree of 1-hinges, but which are not always minimal 1-hinges. At Step 4, we examine each edge e of F to see if it gives rise to an $\{e\}$-decomposition of $F - \{e\}$ with two or more non-empty elements. By the Decomposition Property, the existence of such a decomposition means that the node is non-minimal, so it is decomposed into smaller nodes. The fact that these smaller nodes are 1-hinges of the original hypergraph follows from the Inheritance Property. If no edge separates a node in this way then each node of N_i is minimal so we have a valid 1-hinge-tree and the algorithm stops.

4. Decompositions using arbitrary k-hinges

In the previous section, we have shown that there is a straightforward decomposition strategy using 1-hinges, which provides a useful measure for the structural complexity of hypergraphs. However, the minimal 1-hinges may be quite large in many cases, so it is natural to look for more powerful decompo-

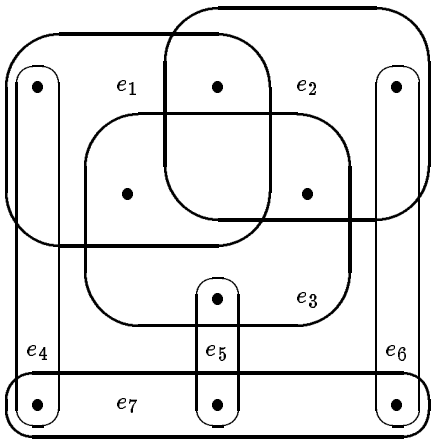

FIGURE 4. A hypergraph with a 2-hinge-tree decomposition.

sition strategies which can break down hypergraphs into smaller units. In this section, we show that more powerful decompositions are possible in certain cases if we are not restricted to 1-hinges.

The following example shows that a hypergraph may have a useful 2-hinge-tree decomposition even when it contains no proper 1-hinges:

EXAMPLE 4.1. Figure 4 shows a hypergraph with 7 edges, $e_1, \ldots, e_7$, containing no smaller 1-hinges. It therefore has degree of cyclicity 7. However, this hypergraph has a 2-hinge-tree in which the size of the largest node is 4. The largest 2-hinge in this tree, $\{e_1, e_2, e_3, e_7\}$, is indicated by the heavy lines.

Example 4.1 also shows that minimal 2-hinges need not be connected, in contrast to minimal 1-hinges which are always connected [**9**].

The next result demonstrates the power of 2-hinges for the decomposition of series-parallel graphs:

PROPOSITION 4.1. *Any series-parallel graph has a 2-hinge-tree decomposition in which every tree node contains at most three edges.*

PROOF. Let $G = (V, E)$ be a series-parallel graph with terminals u and v. For any such graph we may write $G = G_1 * G_2$, where $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, and "$*$" represents either series combination or parallel combination. Choose any edges $e_1 \in E_1$ and $e_2 \in E_2$ such that $u \in e_1$ and $v \in e_2$. We will prove, by induction on the number of edges, that G can be decomposed into a 2-hinge-tree in which the largest node has size 3, and some node contains both e_1 and e_2.

If $|E| \leq 3$, then E itself, obviously containing e_1 and e_2, is a minimal 2-hinge, yielding a single-node 2-hinge-tree, so the result holds trivially. Now assume that $|E| > 3$ and that the induction hypothesis holds for all series-parallel graphs with fewer than $|E|$ edges. The graph G may be written as $G_1 * G_2$, as above, with $|E_1| < |E|$ and $|E_2| < |E|$. Let the terminals of G_1 be u_1 and v_1, and let the terminals of G_2 be u_2 and v_2. Choose $f_1 \in E_1$ and $f_2 \in E_2$ such that $u_1 \in e_1$

and $v_1 \in f_1$ or vice versa and $u_2 \in e_2$ and $v_2 \in f_2$ or vice versa. If possible, f_1 should be chosen different from e_1, and f_2 different from e_2. By the induction hypothesis, there exists a 2-hinge-tree T_1' of G_1 with a node n_1' containing both e_1 and f_1, and a 2-hinge-tree of G_2 with a node n_2' containing both e_2 and f_2. Let $n_1 = \{e_1, e_2, f_1\}$, and $n_2 = \{e_1, e_2, f_2\}$. Then n_1 (respectively n_2) is a 2-hinge of G if its size is 3. There are three cases to consider:

Case 1. $|E_1| = 1$ *or* $|E_2| = 1$. We only consider the first possibility. From $|E_1| = 1$ and $|E| > 3$ it follows that $|E_2| > 2$. By the choice of f_2, it then follows that $|n_2| = 3$, whence n_2 is a 2-hinge of G. It can also be seen straightforwardly that all nodes of T_2' are also minimal 2-hinges of G. A 2-hinge-tree of G with a node containing both e_1 and e_2 is now obtained from T_2' and n_2 by connecting n_2 to n_2', and labeling the arc $\{e_2, f_2\}$.

Case 2. $|E_1| = 2$ *or* $|E_2| = 2$. Again, we only consider the first possibility. From $|E_1| = 2$, it follows that $|E_2| \geq 2$, whence both $|n_1| = 3$ and $|n_2| = 3$, and both n_1 and n_2 are 2-hinges of G. Now if $|E_2| = 2$, a 2-hinge-tree of G with a node containing both e_1 and e_2 is obtained from n_1 and n_2 by simply connecting them, and labeling the arc $\{e_1, e_2\}$. If $|E_2| > 2$, all nodes of T_2' are also minimal 2-hinges of G, as in Case 1. A 2-hinge-tree of G with a node containing both e_1 and e_2 is now obtained from T_2', n_1, and n_2 by connecting n_1 to n_2, n_2 to n_2', and labeling the arcs appropriately.

Case 3. $|E_1| > 2$ *and* $|E_2| > 2$. Then both $|n_1| = 3$ and $|n_2| = 3$, both n_1 and n_2 are 2-hinges of G, and both the nodes of T_1' and T_2' are minimal 2-hinges of G. A 2-hinge-tree for G with a node containing both e_1 and e_2 is now obtained from T_1', T_2', n_1, and n_2 by connecting n_1 to n_2, n_1 to n_1', and n_2 to n_2', and labeling the arcs appropriately. □

We can apply Proposition 4.1 to obtain a decomposition for the cycle graph. (Note that in the cycle graph *every* set of three edges is a 2-hinge.)

EXAMPLE 4.2. The cycle graph C_n with n vertices and edges, $n \geq 3$, has a 2-hinge-tree where every node contains exactly three edges. This tree is in fact a linear list. We construct it by consecutively labeling the edges as $0, 1, \ldots n-1$ and putting $N_i = \{\{0, i, i+1\} \mid i = 1, \ldots, n-2\}$. The required 2-hinge-tree is now obtained by connecting sets of edges with consecutive indices.

The next result shows that most graphs contain many small k-hinges:

PROPOSITION 4.2. *In any graph, every set of* $k+1$ *edges containing a path of length* 3 *is a* k*-hinge.*

PROOF. Let $G = (V, E)$ be a graph, and let $E' = \{e_1, e_2, \ldots, e_{k+1}\}$ be a subset of edges containing a path of length 3. Without loss of generality, assume that e_1, e_2, e_3 form a path. Then $\bigcup E' = \bigcup (E' - e_2)$. Hence $E' - e_2$ is a separating set of k edges for each connected component of $E - E'$ with respect to E', whence E' is a k-hinge. □

Since there are so many k-hinges with $k+1$ edges in simple graphs, it is natural to ask whether a minimal k-hinge in a graph can be very large. We therefore list some examples of large minimal 2-hinges in graphs.

EXAMPLE 4.3.

(i) Any 1-factor of K_{2n} is a minimal 2-hinge with n edges.[5]

(ii) Any maximal star, $K_{1,n-1}$, in K_n, $n \geq 4$, is a connected minimal 2-hinge with $n-1$ edges.[6]

(iii) Consider the previous example. If each edge of K_n not in $K_{1,n-1}$ is replaced by a path of length 2, $K_{1,n-1}$ remains a connected minimal 2-hinge with $n-1$ edges which is moreover vertex-generated.

In order to decompose an arbitrary graph into a k-hinge-tree we are sometimes forced to use k-hinges which contain more than $k+1$ edges, as the following proposition shows:

PROPOSITION 4.3. *There is no k-hinge-tree of K_{2k+3} in which every node has size $k+1$.*

PROOF. Assume, for contradiction, that there exists a k-hinge-tree T of K_{2k+3} in which every node has size $k+1$. Thus each node of T contains at most $2k+2$ vertices. Let n_1 be a node in T of which the number of vertices is maximal. Let v_2 be an arbitrary vertex of K_{2k+3} not in n_1 (such a vertex exists), and let n_2 be a node of T containing v_2 such that the distance in T between n_1 and n_2 is minimal. By the choice of n_1 and n_2, n_1 contains a vertex, say v_1, not in n_2. Let n_3 be an arbitrary node of T containing the edge $\{v_1, v_2\}$. Clearly, $n_3 \neq n_1$ and $n_3 \neq n_2$. By Property 4 of Definition 2.4, the node n_2 is not on the path in T connecting n_1 and n_3, since n_1 and n_3 share the vertex v_1, which is not in n_2. Similarly, n_1 cannot be on the path connecting n_2 and n_3. Hence the path connecting n_1 to n_2 contains an internal node n_4 which is both on the path connecting n_1 to n_3 and the path connecting n_2 to n_3.[7] By Property 4 of Definition 2.4, the node n_4 contains the vertex v_2 shared by the nodes n_1 and n_3. However n_4 is closer to n_1 than is n_2, a contradiction. Hence K_{2k+3} has no k-tree in which every node has size $k+1$. □

5. Difficulties and open problems for arbitrary k-hinges

Since arbitrary k-hinges allow more powerful decompositions than 1-hinges, it is natural to ask whether the results of Section 3 can be generalized to k-hinges. In other words, is it also true that, for $k > 1$, every hypergraph has a k-hinge-tree decomposition, and if so, is there an associated invariant complexity measureΓ

[5] A 1-factor of a graph is a set of edges that are pairwise non-incident and cover the set of nodes. For any number n, K_n denotes the complete graph on n vertices.

[6] A star is a graph the edges of which are incident with a common vertex.

[7] Observe that n_3 and n_4 can be equal.

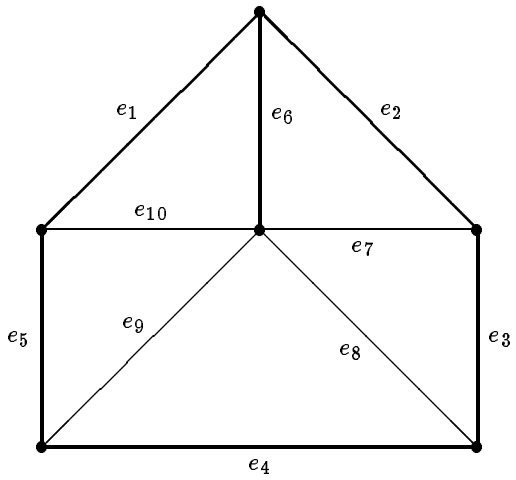

FIGURE 5. A hypergraph illustrating that 2-hinges do not have the Inheritance Property.

In order to establish the results for 1-hinges we made use of the Inheritance Property, the Decomposition Property, and the Inseparability Property. The following results indicate that these properties do not directly generalize to arbitrary k-hinges.

First of all unlike 1-hinges, 2-hinges do not have the Inheritance Property.

EXAMPLE 5.1. Figure 5 shows a graph, G, with edges $e_1, \ldots, e_{10}$, which contains a 2-hinge, $H = \{e_1, \ldots, e_5, e_6\}$, indicated by heavy lines. The subgraph $(\bigcup H, H)$ contains a 2-hinge, $H' = \{e_1, \ldots, e_5\}$, which is not a 2-hinge of G.

Furthermore, 2-hinges do not have the Decomposition Property, the obvious generalization of which to arbitrary k-hinges is as follows:

DECOMPOSITION PROPERTY FOR k-HINGES. Let (V, E) be a hypergraph and let H be a k-hinge of (V, E). Then H satisfies the *decomposition property* if for every subset $h \subseteq E$ containing k edges and for every h-decomposition $\mathcal{D}$ of H, there exists $S \in \mathcal{D}$ such that, for every $\mathcal{D}' \subseteq \mathcal{D}$ with $S \in \mathcal{D}'$, $\bigcup \mathcal{D}'$ is either a k-hinge of (V, E) or a subset of E containing at most k edges.

EXAMPLE 5.2. Consider a family of hypergraphs, $R_n, n \geq 2$. For every n, R_n consists of $3n$ points and $2n + 2$ edges. The points are labeled x_i, y_i, and z_i, $i = 1, \ldots, n$, and the edges are as follows:

$$
\begin{aligned}
e_x &= \{x_1, \ldots, x_n\};\\
e_y &= \{y_1, \ldots, y_n\};\\
f_i &= \{x_i, y_i, z_i\},\ i = 1, \ldots, n;\\
g_i &= \{z_i, x_{(i+1) \bmod n}, y_{(i+1) \bmod n}\},\ i = 1, \ldots, n.
\end{aligned}
$$

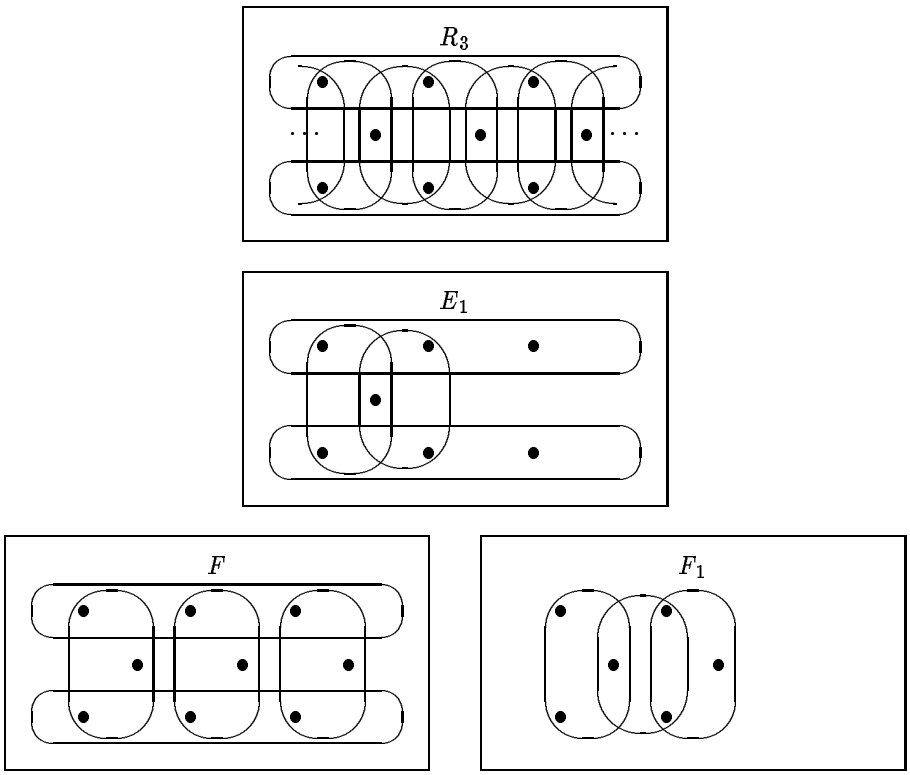

FIGURE 6. The hypergraph R_3 and its minimal 2-hinges E_1, F, and F_1 (Example 5.2).

The minimal 2-hinges of R_n are as follows:

$$\begin{aligned}
E_i &= \{e_x, e_y, f_i, g_i\},\ i = 1, \ldots, n;\\
F &= \{e_x, e_y, f_1, \ldots, f_n\};\\
G &= \{e_x, e_y, g_1, \ldots, g_n\};\\
F_i &= \{f_i, g_i, f_{(i+1) \bmod n}\},\ i = 1, \ldots, n;\\
G_i &= \{g_i, f_{(i+1) \bmod n}, g_{(i+1) \bmod n}\},\ i = 1, \ldots, n.
\end{aligned}$$

The hypergraph R_3 and its minimal 2-hinges E_1, F, and F_1 are illustrated in Figure 6.

To see that 2-hinges do not have the Decomposition Property, consider the minimal 2-hinge F. Let $h = \{e_x, e_y\}$, and let $\mathcal{D} = \{\{e_x, e_y\}, \{f_1\}, \ldots, \{f_n\}\}$. Clearly, $\mathcal{D}$ is an h-decomposition of F. Now notice that, for $i = 1, \ldots, n$, $\{e_x, e_y, f_i\}$ is *not* a 2-hinge of R_n. Hence there is no $S \in \mathcal{D}$ such that every subset of $\mathcal{D}$ encompassing at least 3 edges including those of S generates a 2-hinge of (V, E).

As Proposition 3.1 and the Invariance Theorem rely on the Decomposition Property, we cannot hope to generalize their proofs to show the equivalent results

for arbitrary k-hinges. As a matter of fact, a hypergraph can indeed have two different 2-hinge-trees where the size of the largest node is not the same, as the following example shows:

EXAMPLE 5.3. Consider the family of hypergraphs R_n, $n \geq 2$, described in Example 5.2. A first possible 2-hinge-tree, T_E, is the linear list $E_1, \ldots, E_n$. A second possible 2-hinge-tree, T_F, is the star with central node F and leaf nodes $F_1, \ldots F_n$. A third possible 2-hinge-tree, T_G, is the star with central node G and leaf nodes $G_1, \ldots G_n$. The size of the largest node in T_E is 4, whereas the size of the largest node in both T_F and T_G is $n+2$.

A closer examination of the proof of Proposition 3.1 reveals that it does not rely on the full Decomposition Property, but rather on the consequent Inseparability Property. We might therefore attempt to remedy the situation by requiring that the nodes of a k-hinge-tree satisfy the Inseparability Property below, which is a generalization of the Inseparability Property for 1-hinges.

INSEPARABILITY PROPERTY FOR k-HINGES. Let (V, E) be a hypergraph and let H be a k-hinge of (V, E). Then H is *inseparable* if for every subset $h \subseteq E$ containing k edges, and for every h-decomposition $\mathcal{D}$ of H, at most one element of $\mathcal{D}$ contains more than k edges, and all elements of $\mathcal{D}$ except for a largest one contain at most k edges in total.

Notice that, for general k-hinges, the Inseparability Property follows from the Decomposition Property in the same way as it does for 1-hinges. Furthermore notice that inseparability implies minimality:

PROPOSITION 5.1. *An inseparable k-hinge of a hypergraph is also minimal.*

PROOF. We prove the contrapositive. Thereto, let (V, E) be a hypergraph and let H be a k-hinge of (V, E) that is *not* minimal. Thus H contains another k-hinge of (V, E), say H'. Let e be any edge of $H - H'$, let C_e be the connected component of $E - H'$ with respect H' containing e and let $h' = \{e'_1, \ldots, e'_k\}$ be a corresponding separating edge set. Let $\mathcal{D} = \{\{e'_1\}, \ldots, \{e'_k\}, H \cap C_e, H - (C_e \cup h')\}$. Clearly, $\mathcal{D}$ is an h'-decomposition of H which contains $k+2$ nonempty elements, whence it necessarily violates the second condition of the Inseparability Property for k-hinges. Hence H is not inseparable. □

On the other hand, not every minimal k-hinge is inseparable, as is shown by Example 5.4, below.

EXAMPLE 5.4. Consider the following variation on Example 5.1. Let $\bar{G}$ be the *hyper*graph constructed from the graph G in Figure 5 by adding a unique extra node to each of the edges $e_1, \ldots, e_5$, and by adding all binary edges connecting two extra nodes. Let $\bar{H}$ and $\bar{H}'$ be the sets of edges of $\bar{G}$ corresponding to H and H' in G, respectively. An exhaustive but straightforward examination of all subsets of $\bar{H}$ shows that $\bar{H}$ is a minimal 2-hinge. However, $\bar{H}$ is *not* inseparable, since $\{\bar{H}', \bar{H} - \bar{H}'\}$ is an $\{e_1, e_2\}$-decomposition of $\bar{H}$ which does not satisfy the Inseparability Property for 2-hinges.

Hence imposing the Inseparability Property on the nodes of a k-hinge-tree yields a genuine restriction.

We might hope that, with this restriction, it would be possible to derive an invariant for k-hinge-trees. Notice that Example 5.3 is not a counterexample to this proposal, as the nodes F in T_F and G in T_G are not inseparable, as can be easily verified. We conjecture that, using similar arguments as in the proof of Proposition 3.1, it is possible to find a bound for the largest inseparable k-hinge in terms of the size of the largest node in an arbitrary k-hinge-tree of which all the nodes are inseparable. Unfortunately, the bound will be weaker than in Proposition 3.1 and will not allow us to derive an equality as in the proof of the Invariance Theorem. In fact, Example 5.5 below *demonstrates* that the size of the largest node in a k-hinge-tree of which all the nodes are inseparable is not an invariant.

EXAMPLE 5.5. Consider the hypergraph G with 11 vertices, $v_1, \ldots, v_{11}$, and 7 edges, $e_1, e_2, e_3, f_1, \ldots, f_4$, defined as follows:

$$\begin{aligned} e_1 &= \{v_1, v_2, v_3, v_4\}; & f_1 &= \{v_1, v_3, v_4, v_8, v_9\}; \\ e_2 &= \{v_8, v_9, v_{10}, v_{11}\}; & f_2 &= \{v_2, v_3, v_4, v_{10}, v_{11}\}; \\ e_3 &= \{v_4, v_5, v_6, v_7\}; & f_3 &= \{v_4, v_5, v_7, v_8, v_{10}\}; \\ & & f_4 &= \{v_4, v_5, v_6, v_9, v_{11}\}. \end{aligned}$$

By an exhaustive search we find that the inseparable 2-hinges of G are

$$\begin{array}{lll} \{e_1, e_2, e_3\}; & \{e_2, e_3, f_3\}; & \{e_2, f_1, f_2\}; \\ \{e_1, e_2, f_1\}; & \{e_2, e_3, f_4\}; & \{e_3, f_3, f_4\}; \\ \{e_1, e_2, f_2\}; & \{e_1, f_1, f_2\}; & \{f_1, f_2, f_3, f_4\}. \end{array}$$

Two possible 2-hinge-trees of G built from these inseparable 2-hinges are shown in Figure 5.5. The size of the largest node in T_e is 3, whereas the size of the largest node in T_f is 4.

The Hinge-Tree Decomposition Algorithm for 1-hinge-tree decompositions also relies on the Inheritance, Decomposition, and Inseparability Properties, so it cannot be directly generalized to k-hinges either. For example, the algorithm uses the fact that a 1-hinge-tree can be obtained by first decomposing the hypergraph into a tree of 1-hinges that are not necessarily minimal and then further decomposing these 1-hinges. As is shown in Example 5.1, a similar strategy for 2-hinge-trees is not guaranteed to yield a correct result. Another property the Hinge-Tree Decomposition Algorithm relies on is that a singleton separating edge set is always contained in some minimal 1-hinge. Unfortunately, this property does not carry over to 2-hinges, either. The following example exhibits a hypergraph having a separating edge pair not contained in *any* minimal 2-hinge of that hypergraph.

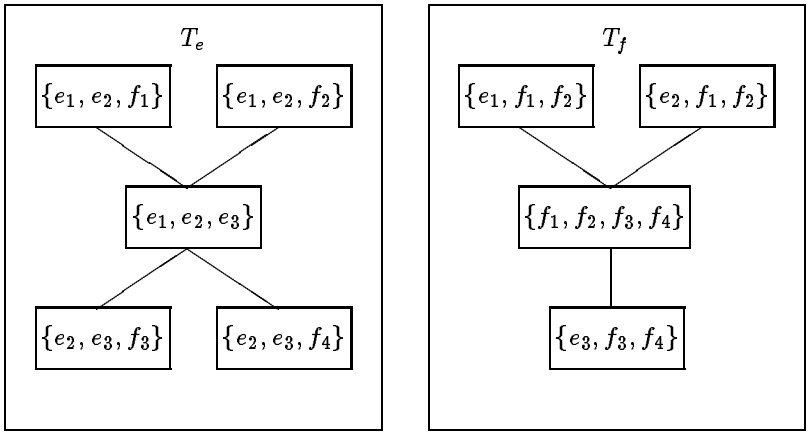

FIGURE 7. Two possible 2-hinge-trees for G (Example 5.5).

EXAMPLE 5.6. Consider the hypergraph with nodes

$$\{x_1, x_2, y_1, y_2, z_1, z_2, u_1, u_2\}$$

and edges

$$\begin{array}{lll} e_1 = \{x_1, x_2, z_1, u_1\}; & f_1 = \{y_1, y_2, z_1, u_2\}; & g_1 = \{x_1, y_1\}; \\ e_2 = \{x_1, x_2, z_2, u_2\}; & f_2 = \{y_1, y_2, z_2, u_1\}; & g_2 = \{x_2, y_2\}. \end{array}$$

Clearly, $\{g_1, g_2\}$ is a separating edge pair separating off $\{e_1, f_1\}$ from $\{e_2, f_2\}$. Now, the smallest 2-hinges (with respect to inclusion) containing $\{g_1, g_2\}$ are the edge sets $\{e_1, f_1, g_1, g_2\}$, $\{e_2, f_2, g_1, g_2\}$, $\{e_1, f_2, g_1, g_2\}$, and $\{e_2, f_1, g_1, g_2\}$, as can easily be verified by a simple, exhaustive search. None of these four 2-hinges are minimal however, as all eight sets obtained from them by either removing g_1 or g_2 are 2-hinges, too.

The above examples clearly show that an algorithm for generating k-hinge-trees cannot simply be obtained by a straightforward generalization of the Hinge-Tree Decomposition Algorithm. As a matter of fact, it is an open problem whether or not there exists an efficient and effective algorithm for finding k-hinge-trees.

6. Conclusions

We have shown that any hypergraph has a decomposition into a tree of 1-hinges, and the size of the largest minimal hinge in such a tree is an invariant of the hypergraph. This provides a classification scheme for hypergraphs giving a useful measure of their structural complexity.

The extension of the notion of a 1-hinge to k-hinges allows more refined decompositions in many cases, but k-hinges fail to have the pertinent properties of inheritance and inseparability. Without inheritance it is difficult to guarantee

the existence of a tree decomposition into *minimal* k-hinges, whence the existence of an invariant on the structure of k-hinges is unlikely. Even when trees do exist, it is not clear whether there is a general algorithm to construct them.

One approach to these problems is to strengthen the definition of a minimal k-hinge in order to guarantee the required properties. Alternatively, it may be necessary to formulate a more sophisticated definition of the invariant.

In the light of these results, it is clear that the degree of cyclicity is a strikingly simple and important measure for the structural complexity of a hypergraph.

References

1. S. Arnborg, *Efficient algorithms for combinatorial problems on graphs with bounded decomposability—a survey.* BIT **25** (1985), 2–23.
2. C. Beeri et al., *Properties of acyclic database schemes.* Proc. 13th Annual ACM Symposium on the Theory of Computing (Milwaukee, 1981), ACM Press, New York, pp. 355–362.
3. C. Beeri et al., *On the desirability of acyclic database schemes.* J. Assoc. Comput. Mach. **30** (1983), 479–513.
4. C. Bergé, *Graphs and Hypergraphs*, North-Holland, New York, 1976.
5. B. Courcelle, *Graph rewriting: an algebraic and logic approach.* Handbook of Theoretical Computer Science (J. van Leeuwen, ed.), Elsevier Science Publishers, Amsterdam, 1990, p. 224.
6. E. C. Freuder, *A sufficient condition for backtrack-bounded search.* J. Assoc. Comput. Mach. **32** (1985), 755–761.
7. M. Gyssens, *On the complexity of join dependencies.* ACM Trans. Database Systems **11** (1986), 81–108.
8. M. Gyssens, P. Jeavons, and D. Cohen, *Decomposing constraint satisfaction problems using database techniques.* Artificial Intelligence **66** (1994), 57–89.
9. M. Gyssens and J. Paredaens, *A decomposition methodology for cyclic databases.* Advances in Database Theory, vol. 2 (Toulouse, 1983, H. Gallaire, J. Minker, and J. M. Nicolas, eds.), Plenum Press, New York, 1984, pp. 85–122.
10. _____, *On the decomposition of join dependencies.* Proc. 3rd ACM SIGACT-SIGMOD Symposium on Principles of Database Systems (Waterloo, Ont., 1984), ACM Press, New York, 1984, pp. 143–152.
11. Yannakakis, M., *Computing the minimum fill-in is NP-complete.* SIAM J. Algebraic Discrete Methods **2** (1981), 77–79.

Department of Computer Science, Royal Holloway, University of London, Egham, Surrey TW20 0EX, England
E-mail address: pete@dcs.rhbnc.ac.uk

Department of Computer Science, Royal Holloway, University of London, Egham, Surrey TW20 0EX, England
E-mail address: pete@dcs.rhbnc.ac.uk

Department WNI, University of Limburg (LUC), B-3590 Diepenbeek, Belgium
E-mail address: gyssens@charlie.luc.ac.be