Retractions of chordal and related graphs

by

Cynthia Loten

B.Sc. Honours, University of Regina, May 1995.

M.Math., Univeristy of Waterloo, Oct. 1996.

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
in the Department
of
Mathematics

SIMON FRASER UNIVERSITY
December 2003

APPROVAL

Name:	Cynthia Loten
Degree:	Ph.D.
Title of thesis:	Retractions of chordal and related graphs

Examining Committee: Dr. P. Lisonek
Chair

Dr. P. Hell
Senior Supervisor

Dr. T. Brown
Supervisory Committee

Dr. L. Goddyn
Supervisory Committee

Dr. L. Stacho

Dr. G. MacGillivray
External Examiner

Date Approved:

Abstract

A graph H is a retract of a graph G if H is a subgraph of G and there exists an edge preserving function of the vertex set of G to the vertex set of H that fixes each vertex of H. There are no known necessary and sufficient conditions for H to be a retract of G. We can, however, choose a particular necessary condition, call it N, and study the graphs for which that particular necessary condition is also sufficient. Such graphs are called absolute retracts with respect to N.

A simple necessary condition is preserving distances; this generates the class of absolute retracts with respect to isometry, which has been well studied.

Another necessary condition is the following: if there is no vertex in H that is within prescribed distances to a fixed set of vertices of H and H is a retract of G, then there is no such vertex in G either. Thus there is a hole in H that can't be filled by a vertex of G. This is the first necessary condition we explore.

There are two other necessary conditions that we study. The former is concerns partial mappings of trees and the latter is based on rephrasing the retraction problem as a list homomorphism problem.

These three necessary conditions generate the class of absolute retracts with respect to holes, the class of absolute retracts with respect to tree obstructions, and the class of absolute retracts with respect to arc consistency.

We investigate chordal graphs with regard to all three classes of absolute retracts listed above. This leads to the introduction of three classes of graphs that generalize chordal graphs: stretched graphs, strongly stretched graphs, and wheeled graphs.

Stretched graphs and strongly stretched graphs are used to characterize the variety generated by chordal graphs, and we prove that wheeled graphs are absolute retracts with respect to arc consistency.

We also compare these three classes of absolute retracts with the graphs that admit near unanimity functions and the dismantlable graphs.

Acknowledgments

There's no way to rank thank you's, so these are in no particular order.

I would like to thank my supervisor, Pavol Hell, who taught me a great deal about math and how to write it up, and who also generously provided financial support. I would also like to thank Denis Hanson for sparking my interest in graph theory, and for some very timely advice and help. Thanks also to Benoit Larose and Claude Tardif for research discussions, and to my examining committee: Gary MacGillivray, Ladislav Stacho, Luis Goddyn and Tom Brown.

Next, there's my family. Thank you to my mom and my sister for encouraging me to do whatever I was interested in. Thanks also to all my extended family for the contributions you made to my life and for making science seem like a normal thing to do.

Thanks to the friends I've made in grad school who made up for the days when math was frustrating. In Ontario: Mark and Myra (who now have Martinique and Misha), Brad, Simal, Sean, Vine, Aaron, Ryan, Andrea, Tomi, Franklin, and Anne. In Vancouver: Laura, Jodie, Laurie, Warren, Kevin, Idris, Danny and Tara, Julia and Steve, Alan, Herre, Marni, John, Albert, and Mason.

Lastly, I want to thank my boyfriend Derek. It's hard to even say all I have to thank you for. Thanks for being the awesome person that you are, for being a part of my life for the last 9 years, for encouraging me to stand up for myself, and for patiently waiting for me to finally finish.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

A graph H is a *retract* of a graph G if H is a subgraph of G and there exists an edge preserving function from G to H that fixes each vertex of H. Such a map is called a *retraction.* Retractions on graphs were first studied by Hell in [40] and by Nowakowski and Rival in [56, 57], who were inspired by related work on posets. A question of great interest is "When do retractions exist?" There has been research concerning this question in regards to bipartite graphs, i.e., graphs without loops at any vertex and without odd cycles, see [3, 4, 6, 41, 42, 56]. This question has also been posed for reflexive graphs, i.e., graphs with loops at each vertex, see [4, 5, 9, 43, 45, 56, 57, 61]. It is retractions on reflexive graphs, and when they exist, that will be the focus of this thesis.

There are no known necessary and sufficient conditions for a graph H to be a retract of a graph G. We can however make classes out of graphs H for which the necessary conditions are sufficient. A graph H will be called an *absolute retract* with respect to necessary condition N if H is a retract of supergraph G whenever necessary condition N is satisfied. We will be studying absolute retracts with respect to different necessary conditions N.

Absolute retracts were first studied in topology [11], see [12, 47]. Let X and Y be (Hausdorff) spaces , where $X \subseteq Y$. Borsuk [11] points out that for X to be a retract of Y, X must be closed in Y; his definition of an absolute retract is based on this necessary condition. Absolute retracts have also be studied for posets. Nevermann and Rival [55] defined their version of an absolute retract based on the necessary condition of gap separation, a concept related to preserving holes, see Chapter 2.

To illustrate the approach of this thesis, we will present a simple necessary condition N for a retraction from a graph G to a subgraph H to exist, showing why it is necessary. Then we will define the class of absolute retracts with respect to N and provide a simple characterization of these absolute retracts.

A graph H is an *induced subgraph* of a graph G if H is a subgraph of G and if two vertices of H are end points of an edge e in G, then this edge e must also be an edge of H. We claim that if H is a retract of G, then H must be an induced subgraph of G.

Let H and G be (reflexive) graphs, and suppose that H is a retract of G. By the definition of retracts, H is a subgraph of G. Let θ be a retraction from G to H; thus θ is an edge preserving map from G to H such that $\theta(g) = g$ whenever g is also a vertex of H. Let x and y be vertices of H. Hence x and y are also vertices of G. If xy is an edge in G, then $\theta(x)\theta(y)$ must be an edge in H as θ is edge preserving. Moreover, as θ fixes each vertex of H, $\theta(x) = x$ and $\theta(y) = y$, and so xy must be an edge of H. Therefore, if H is a retract of G, H must be an induced subgraph of G. The graph in Figure 1.1 demonstrates that a graph H being an induced subgraph of a graph G is not a sufficient condition for H to be a retract of G.

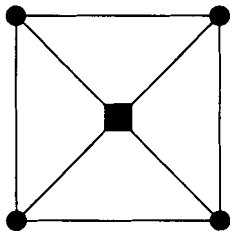


Figure 1.1: The graph on the round vertices is an induced subgraph of the graph on all the vertices, but the graph on the round vertices is not a retract of the graph on all the vertices.

Now we can construct a class of graphs for which being an induced subgraph is a sufficient condition for the existence of a retraction. We will restrict ourselves to connected graphs; the arguments that we make concerning connected graphs can be applied to each component of a disconnected graph. Let Z be the necessary condition that a connected graph H is an induced subgraph of a connected graph G. Then we can define the class of *absolute retracts with respect to* Z, denoted by $\mathcal{AR}_Z$, to be the set of all connected graphs H such that H is a retract of a connected supergraph G whenever H is an induced subgraph of G.

While the definition of $\mathcal{AR}_Z$ is simple, it is hard to recognize graphs that are in $\mathcal{AR}_Z$ based on the definition. Thus our next step is to try to classify the graphs in $\mathcal{AR}_Z$ in terms of properties we can recognize.

A *universal vertex* in a graph H is a vertex x that is adjacent to all vertices of H.

Theorem 1.1. *Let H be a connected graph. Then H is in $\mathcal{AR}_Z$ if and only if H has a universal vertex.*

Proof. Let H be a graph in $\mathcal{AR}_Z$. Let G be a connected supergraph of H such that the vertex set of G is the vertex set of H plus one other vertex w, where w is adjacent to all vertices of H. Clearly H is a induced subgraph of G as we have not added any edges between the vertices of H in creating G. Thus there exists a retraction θ from G to H because $H \in \mathcal{AR}_Z$. Consider the vertex w in G. By the way we defined G, wx is an edge of G for all vertices x of G; the vertex w is adjacent to all vertices of G including itself. Since θ is an edge preserving map, $\theta(w)\theta(x)$ is an edge of H for all vertices x of G. If x is also in H, then $\theta(x) = x$. Thus $\theta(w)x$ is an edge of H for all vertices x in H and so $\theta(w)$ is a universal vertex in H.

Let H be a connected graph that has a universal vertex, call it y. Let G be a connected graph such that H is an induced subgraph of G. Then define the map θ from G to H as follows:

$$\theta(g) = \begin{cases} g & \text{if } g \in V(H) \\ y & \text{otherwise.} \end{cases}$$

We claim that θ is a retraction. Clearly θ fixes every vertex of H. Now we need to show that θ is edge preserving. Let gg' be an edge of G. If both g and g' are vertices of H, then $\theta(g) = g$ and $\theta(g') = g'$. As H is an induced subgraph of G, gg' is also an edge of H. Hence $\theta(g)\theta(g')$ is an edge of H. Now suppose that g is a vertex of H and the g' is not. Then $\theta(g) = g$ and $\theta(g') = y$. Since y is a universal vertex of H, gy is an edge of H and so $\theta(g)\theta(g')$ is an edge of H. Lastly, assume that neither g nor g' is a vertex of H. Then $\theta(g) = \theta(g') = y$. Because H is a reflexive graph, $\theta(g)\theta(g')$ is an edge of H. Therefore θ is an edge preserving map.

$\square$

At this point, we have presented a necessary condition Z for the existence of a retraction, defined $\mathcal{AR}_Z$ and studied the graphs in $\mathcal{AR}_Z$. We will repeat this process for three stronger necessary conditions and their related absolute retracts.

In the remainder of this chapter, we present the needed graph theory definitions and some background material. In particular, we will present a well studied class of absolute retracts, we will justify our interest in chordal graphs and we will present some basic results about dismantlable graphs, graphs that admit near unanimity functions and chordal graphs.

In Chapter 2 we present the idea of a hole; this a constraint derived from unsatisfied distance requirements. We then show that 'preserving' all holes is a necessary condition for the existence of a retraction. After proving some basic results about holes and absolute retracts with respect to holes, we prove that holes imply the existence of certain isometric subgraphs in connected chordal graphs. Then we introduce the class of stretched graphs, whose definition is based on holes. We prove that the variety generated by connected chordal graphs is contained in the class of stretched graphs and we use stretched graphs to classify graphs that are in the intersection of the variety generated by connected chordal graphs, and absolute retracts with respect to holes. Lastly, we compare absolute retracts with respect to holes to graphs that admit near unanimity functions and dismantlable graphs.

In Chapter 3 we introduce tree obstructions; this is a constraint derived from trees with labeled leaves and partial functions that don't extend to homomorphisms. We then show the 'preserving' all tree obstructions is a necessary condition for the existence of a retraction. Next we prove some basic results about tree obstructions and absolute retracts with respect to tree obstructions. Then we relate tree obstructions to tree duality, an idea originating from homomorphisms on digraphs. As with holes, we study tree obstructions on connected chordal graphs. We then introduce the class of strongly stretched graphs, whose definition is based tree obstructions. We prove that the variety generated by connected chordal graphs is contained in the class of strongly stretched graphs and use strongly stretched graphs to conjecture a classification of the variety generated by connected chordal graphs. At the end of the chapter, we relate absolute retracts with respect to tree obstructions to graphs that admit near unanimity functions.

In Chapter 4, we rephrase the retraction problem as particular list homomorphism problem. Then we derive a necessary condition for the existence of a retraction based the idea of arc consistent lists. We present some basic facts about absolute retracts with respect to arc consistency and three classes of graphs that are equivalent to the class of absolute retracts with respect to arc consistency. Then we introduce a new class of graphs that generalizes connected chordal graphs and is contained in the class of absolute retracts with respect to arc consistency. Lastly, we prove that the class of absolute retracts with respect to arc consistency and the class of dismantlable graphs are incomparable.

1.1 Definitions

For any definitions not mentioned here, see Bondy and Murty [10] or Golumbic [37].

A *graph* H is an ordered pair $(V(H), E(H))$, where $V(H)$ is a finite non-empty set of *vertices* and $E(H)$ is a set of *edges*, where each edge is a subset of size 1 or 2 of $V(H)$. Let H be a graph and let e be an edge in $E(H)$. Then the vertex or vertices in e are called the *endpoint(s)* of e. If $e = \{x, y\}$, then e is an unordered pair and will sometimes be denoted by xy or yx. If $e = \{x\}$, then we call the edge e a *loop*, and e will sometimes be denoted by xx to remain consistent with the non-loop notation and to avoid confusion with the vertex x. For a vertex $x \in V(H)$, the loop xx is called the *loop at x*. Note that referring to an edge by its endpoint(s) causes no confusion as $E(H)$ is a set and so each edge is uniquely defined by its endpoint(s). A graph H is called *irreflexive* if H has no loops and H is called *reflexive* if xx is an edge of H for each vertex x of H, i.e., if there exists a loop at each vertex of H. For the rest of this thesis, all graphs are to be assumed to be reflexive unless otherwise specified. We omit the loops when drawing reflexive graphs to avoid clutter.

Let H be a graph. If $xy \in E(H)$, then we say that the vertices x and y are *adjacent* and that x is a *neighbour* of y and vice versa. The set of neighbours of x will denoted by $N_H(x)$, or $N(x)$ if the graph is clear from the context. Note that since we are assuming H is reflexive, $x \in N(x)$ for all vertices $x \in V(H)$. We call a vertex z of H a *nontrivial neighbour* of x if $z \in N(x) \setminus \{x\}$. For the purposes of this thesis, the *degree* of the vertex x, $\deg_H(x)$, is the number of nontrivial neighbours of x, i.e., $|N(x) \setminus \{x\}|$.

Let G be a graph. We call a graph H a *subgraph* of G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$; we denote this by $H \subseteq G$. Conversely, G is called a *supergraph* of H. Note that G is a subgraph of itself. The graph H is a *proper subgraph* G if H is a subgraph of G and $V(H) \subset V(G)$ or $E(H) \subset E(G)$ (or both); we then write $H \subset G$. For a set $S \subseteq V(G)$, the *subgraph of H induced by S* is the graph on vertex set S with edge set $\{xy \mid x, y \in S \text{ and } xy \in E(G)\}$. A subgraph H of G is an *induced subgraph* of G if H is the subgraph of G induced by $V(H)$. We may obtain other induced subgraphs of G by removing vertices of G. For $S \subseteq V(G)$, the graph $G \setminus S$ is the subgraph of G induced by $V(G) \setminus S$. Thus $G \setminus S$ is the subgraph of G we obtain by removing from G the vertices of S and all edges of G that have at least one endpoint in S. We may also remove edges of G to obtain a subgraph; for a set $E' \subseteq E(G)$, $G \setminus E'$ is the subgraph G' of G where $V(G') = V(G)$ and $E(G') = E(G) \setminus E'$. Observe that $G \setminus E'$ is not necessarily an induced subgraph of G. Let x be a vertex of G and e an edge of G. We will abuse notation slightly and write $G \setminus x$ and $G \setminus e$ instead of $G \setminus \{x\}$ and $G \setminus \{e\}$.

Let H and G be graphs which have at least one vertex in common. Then we define $H \cap G$ to be the graph with vertex set $V(H) \cap V(G)$ and edge set $E(H) \cap E(G)$. Note that $H \cap G$ is a subset of both H and G.

A graph on n vertices is called a *complete graph on n vertices*, denoted by K_n, if all vertices of K_n are pairwise adjacent. Hence, for each vertex $x \in V(K_n)$, $\deg(x) = n-1$ and $N(x) = V(K_n)$. Let H be a graph and let $S \subseteq V(H)$. The set S is a *clique* in H if the subgraph induced by S in H is a complete graph. Note that a graph, and hence a complete graph, may not have an empty vertex set. Therefore a clique contains at least one vertex. A set S is a *maximal clique* in H if S is a clique and if $S' \subseteq V(H)$ is not a clique whenever $S \subset S'$. Thus S is a maximal clique in H if the subgraph induced by S in H is complete and if $S \nsubseteq N(x)$ for all vertices $x \in V(H) \setminus S$. A vertex $x \in V(H)$ such that $N(x)$ is a clique in H is called *simplicial.*

A subset S of vertices in a graph H is an *independent set* if the subgraph in H induced by S has no edges other than the loops each of the vertices of S.

A *walk* in a graph H is a sequence of vertices of H $x_0x_1 \ldots x_k$ such that $x_ix_{i+1} \in E(H)$ for $i = 0, \ldots, k-1$; we call the vertices $x_1, \ldots, x_{k-1}$ the *internal* vertices of the walk. Moreover, we say that $x_0x_1 \ldots x_k$ is a walk from x_0 to x_k, or an $x_0 - x_k$ walk. A walk between vertices x and y in H can refer to either an $x - y$ walk or a $y - x$ walk. If x_i and x_j each appear once in the walk $P = x_0x_1 \ldots x_k$, $1 \leq i < j \leq k$, then $P[x_i, x_j]$ denotes the walk $x_ix_{i+1} \ldots x_{j-1}x_j$. The *length of a walk* is the number of edges in the walk; thus $x_0x_1 \ldots x_k$ is a walk of length k. A *path* in a graph H is a walk $x_0x_1 \ldots x_k$ in H in which all the vertices are distinct. The definitions for internal vertices, length, etc. in regards to walks carry over to paths. Lastly, a *cycle* in H is a walk $x_0x_1 \ldots x_k$ where the vertices $x_1, \ldots, x_k$ are distinct, $x_0 = x_k$ and $k \geq 3$. The *length of a cycle* is the number of edges in the cycle; thus $x_0x_1 \ldots x_{k-1}x_0$ is a cycle of length k. A cycle of length k will referred to as a k-cycle. Moreover, a cycle of length 3 is called a *trivial cycle* and all others are referred to as *nontrivial cycles.*

We call a graph G a *path* if G has vertex set $V(G) = \{x_0, x_1, \ldots, x_k\}$ and edge set $E(G) = \{x_ix_{i+1} \mid i = 0, 1, \ldots, k-1\} \cup \{x_ix_i \mid i = 0, 1, \ldots, k\}$. We call a graph G a *cycle* if it has vertex set $V(G) = \{x_1, x_2, \ldots, x_k\}$ and edge set $E(G) = \{x_ix_{i+1} \mid i = 1, 2, \ldots, k-1\} \cup \{x_kx_1\} \cup \{x_ix_i \mid i = 1, 2, \ldots, k\}$, $k \geq 3$; we often denote this graph by C_k. Moreover, if the graph C is a cycle and C is an induced subgraph of graph H, we say that C is an *induced cycle* in H. We define induced paths similarly.

An induced path will also be referred to as a *chordless* path.

A graph without induced cycles of size 4 or more is called *chordal*. Thus all induced cycles in chordal graphs are trivial cycles. Chordal graphs will be studied in detail in Section 1.2.2. A *wheel* on $k+1$ vertices, $k \geq 4$, is a graph, denoted by W_k, with vertex set $V(W) = \{r_1, \ldots, r_k, h\}$ and edge edge $E(W) = \{r_1r_2, r_2r_3, \ldots, r_{k-1}r_k, r_kr_1\} \cup \{hr_i \mid i = 1, \ldots, k\} \cup \{xx \mid x \in V(W)\}$. We call $r_1r_2 \ldots r_kr_1$ the *rim cycle* of W_k and the vertices $r_1, \ldots, r_k$ the *rim vertices* of W_k. The vertex h is the *hub* of W_k. We often refer to W_k as the wheel on k rim vertices.

Let H be a graph. We say that H is *connected* if for all vertices $x, y \in V(H)$ there exists a path between x and y in H and H is *disconnected* otherwise. A *component* of H is maximal connected induced subgraph of H; if J is a component of H and J' is another subgraph of H such that J' is connected and $J \subseteq J'$, then $J = J'$. We will also apply the term connected to sets; a subset S of $V(H)$ is *connected* if the subgraph of H induced by S is connected and S is *disconnected* otherwise. We call a subset $S \subseteq V(H)$ a *vertex cut set* if $H \setminus S$ is disconnected. The set S is a *minimal vertex cut set* there does not exist a proper subset S' of S such that $H \setminus S'$ is disconnected. A subset $S \subseteq V(H)$ is called an $x - y$ *separator* if there does not exist an $x - y$ path in $H \setminus S$ and $x, y \notin S$. The set S is called a *minimal* $x - y$ *separator* there does not exist a proper subset S' of S such that S' is an $x - y$ separator. The set S is called a *separator* if there exist vertices $x, y \in V(H)$ such that S is an $x - y$ separator. Moreover, S is a *minimal separator* if there exist vertices $x, y \in V(H)$ such that S is a minimal $x - y$ separator. Note that while every minimal vertex cut set is a minimal separator, a minimal separator may not be a minimal vertex cut set. Also, if H is disconnected, then $\emptyset$ is a vertex cut set.

A connected graph H that does not have a cycle as subgraph is called a *tree.*

Let H be a graph. The *distance* between two vertices x and y in H is the length of a shortest $x - y$ path. This will be denoted by $d_H(x, y)$, or just $d(x, y)$ if the the graph H is clear from the context. If there does not exist a path between x and y, then $d_H(x, y) = \infty$. The *diameter* of H is $\max\{d(x, y) \mid x, y \in V(H)\}$. A vertex $z \in V(H)$ is a *diametrical vertex* if there exists $y \in V(H)$ such that $d(z, y)$ is equal to the diameter of H; note that y is also a diametrical vertex.

We can use the concept of distance in graphs to define a new type of subgraph. Let G be a graph and let H be subgraph of G. Let x and y be two vertices of H and let P be an $x-y$ path in H of length $d_H(x,y)$. As H is a subgraph of G, P is also an $x-y$ path in G. Thus the length of P is bounded below by $d_G(x,y)$. Therefore $d_G(x,y) \leq d_H(x,y)$ for all vertices $x, y \in V(H)$. If $d_G(x,y) = d_H(x,y)$ for all vertices $x, y \in V(H)$, then we say that H is an *isometric subgraph* of G.

We can also use distances to define discs in graphs. Let H be graph. Given a non-negative integer k and a vertex $x \in V(H)$, we define the *disk centred at vertex x of H with radius k* to be the set $D_H(x,k) = \{y \mid d_H(x,y) \leq k\}$. Given a non-negative integer k and a non-empty subset $S \subseteq V(H)$, we define the *disk centred at subset S of $V(H)$ with radius k* to be the set $D_H(S,k) = \cup_{x \in S} D_H(x,k)$; in both cases we omit the subscript H when the graph H is understood from the context.

In the next paragraph, we will define a class of graphs based on permissible vertex orderings. As vertex orderings will be used frequently in this thesis, we will now set up some standard notation. To begin with, when listing elements of a set or ordering, we will write $x_1, \ldots, x_n$ instead of $x_1, x_2, \ldots, x_n$. Given a graph H and an ordering $h_1, \ldots, h_n$ of some or all of the vertices of H, we will use H_i to denote the graph $H \setminus \{h_1, \ldots, h_i\}$, $i = 1, \ldots, n$, with the following exception: if $\{h_1, \ldots, h_n\} = V(H)$, then H_n does not exist. For technical reasons, we set $H_0 = H$.

A vertex x in a graph H is *covered* by a vertex y in H, $y \neq x$, if the neighbourhood of y contains the neighbourhood of x, i.e., $N(x) \subseteq N(y)$. Note that as all vertices are self-adjacent, then x and y must also be adjacent. We will refer to x as a *dismantlable vertex* of H if there exists a vertex y of H that covers x. A graph H with n vertices is called *dismantlable* (or *cop win* [58]) if there exists an ordering $h_1, \ldots, h_n$ of $V(H)$ such that for each $i < n$, h_i is dismantlable in H_{i-1}. Note that if $n = 1$, then H is trivially dismantlable. The ordering $h_1, \ldots, h_n$ is call a *dismantling ordering* of H. If H has a dismantling ordering, then H must be connected since for each $i < n$, the vertex h_i has a neighbour h_j, $i < j$. We say that a graph H *dismantles to* a subgraph J if there exists a partial ordering $h_1, \ldots, h_k$ of vertices of H such that h_i is dismantlable in H_{i-1}, $i = 1, \dots, k$, and $H_k = J$. Observe that J is in fact an induced subgraph of H. A graph on at least two vertices is called *ramified* (or *stiff* [16]) if it has no dismantlable vertices.

A function from a graph G to a graph H is a function ϕ of $V(G)$ to $V(H)$; we denote this by $\phi : G \to H$. Such a function is call a *homomorphism* if for all edges $xy \in E(G)$, we have that $\phi(x)\phi(y) \in E(H)$. This is denoted by $G \xrightarrow{\phi} H$; we write $G \to H$ to indicate that $G \xrightarrow{\phi} H$ for some ϕ.

We will now define some common variations of homomorphisms.

Let G and H be graphs. For each vertex g of G, assign to g a subset of $V(H)$, which we will call the *list* of g and denote by $L(g)$. A homomorphism ϕ of G to H is called a *list homomorphism with respect to L* if $\phi(g) \in L(g)$ for all $g \in V(G)$. In this case we write $(G, L) \xrightarrow{\phi} H$; we write $(G, L) \to H$ if $(G, L) \xrightarrow{\phi} H$ for some ϕ. Given graph-list pairs (J, L') and (G, L), we say that (J, L') is homomorphic to (G, L) if there exists a homomorphism ϕ, $J \xrightarrow{\phi} G$, such that $L'(x) = L(\phi(x))$ for all vertices x of J; we denote this by $(J, L') \xrightarrow{\phi} (G, L)$. As before, we write $(J, L') \to (G, L)$ if $(J, L') \xrightarrow{\phi} (G, L)$ for some ϕ.

An *isomorphism* from a graph G to a graph H is a one-to-one, onto function $\phi : G \to H$ such that $xy \in E(G)$ if and only if $\phi(x)\phi(y) \in E(H)$. Thus an isomorphism is always a homomorphism, but not vice versa.

Let H and G be graphs such that H is subgraph of G. We call a homomorphism θ from G to H a *retraction* if $\theta(g) = g$ for all $g \in V(H)$ and we call the graph H a *retract* of G. It is retractions and retracts that are the focus of this thesis.

Let H and G be graphs such that H is a subgraph of G. Suppose that H has components $H_1, \dots, H_n$ and that G has components $G_1, \dots, G_m$, where H_i is a subgraph of G_i, for $i = 1, \dots, n$. Note that H is a retract of G if and only if H_i is a retract of G_i, for $i = 1, \dots, n$. If $m > n$, we can map the extra components of G to some vertex of H since H is reflexive. Therefore, when studying retractions and retracts, we will restrict our analysis to connected graphs.

Let H_i be a graph for $i = 1, \dots, k$. We define the *(categorical) product* of

$H_1, \ldots, H_k$, denoted by $H_1 \times H_2 \times \cdots \times H_k$ or by $\Pi_{i=1}^k H_i$, to be the graph with vertex set $V(H_1) \times V(H_2) \times \cdots \times V(H_k)$, where vertices $(x_1, \ldots, x_k)$ and $(x_1', \ldots, x_k')$ are adjacent if and only if x_i is adjacent to x_i' in H_i, for $i = 1, \ldots, k$. When referring to a vertex of $\Pi_{i=1}^k H_i$, we will either write $(x_1, \ldots, x_k)$ or use vector notation and write $\boldsymbol{x}$. The function $\pi_i : H \rightarrow H_i$ defined by

$$\pi_i(x_1, \ldots, x_k) = x_i$$

is the i^{th} *projection* of H, $i = 1, \ldots, k$. Clearly π_i is a homomorphism, $i = 1, \ldots, k$.

Let H be a graph and let $k \geq 1$ be an integer. We define H^k to be the graph $\Pi_{i=1}^k H_i$, where $H_1 = H_2 = \ldots = H_k = H$.

Let H be a graph and let $k \geq 3$ be an integer. A vertex $(x_1, \ldots, x_k)$ of H^k is called *constant* if $x_1 = x_2 = \ldots = x_k$ and $(x_1, \ldots, x_k)$ is called *nearly unanimous* if all the entries of $(x_1, \ldots, x_k)$ are the same except for one. We will use $\overline{x}$ to denote the constant vertex $(x, \ldots, x)$. A function $\eta : H^k \rightarrow H$ is called a *near unanimity function of arity* k if η is a homomorphism and $\eta(x_1, \ldots, x_k) = x$ whenever at least $k-1$ of the x_i's are equal to x. Thus if $\boldsymbol{x} = (x_1, \ldots, x_k)$ is constant or nearly unanimous, then $\eta(\boldsymbol{x})$ is the vertex of H that occurs at least $k-1$ times in $\boldsymbol{x}$. The graph H admits a near unanimity function if it admits a near unanimity function of some arity. A near unanimity function of arity 3 is commonly called a *majority function.*

Let $\mathcal{C}$ be a class of graphs. The class $\mathcal{C}$ is a *variety* if $\mathcal{C}$ is closed under retractions and products; thus for all $H \in \mathcal{C}$, all retracts H' of H are in $\mathcal{C}$ and for all $H_1, \ldots, H_k$ in $\mathcal{C}$, we also have that $\Pi_{i=1}^k H_i \in \mathcal{C}$. Let $\mathcal{C}$ be a class of graphs. Then the *variety generated by* $\mathcal{C}$ is the smallest variety that contains $\mathcal{C}$. For example, the variety generated by paths is the smallest variety that contains all graphs made of retractions and products of paths. Note that if $\mathcal{C}$ is a variety, then the variety generated by $\mathcal{C}$ is itself, $\mathcal{C}$. Moreover, when the class $\mathcal{C}$ is a variety, we will refer to as the *variety of* $\mathcal{C}$.

To prove that a class of graphs $\mathcal{C}$ is a variety, we must prove two properties. The first is that for all graphs $H \in \mathcal{C}$, each retract of H is also in $\mathcal{C}$. The second is that for any collection of graphs $H_1, \ldots, H_k \in \mathcal{C}$, we have that $\Pi_{i=1}^k H_i \in \mathcal{C}$. Since $\Pi_{i=1}^k H_i = H_1 \times \Pi_{i=2}^k H_i$, it suffices to prove that if $H', H'' \in \mathcal{C}$, then $H' \times H'' \in \mathcal{C}$.

1.2 Background work

We began our discussion on absolute retracts with a very simple example. We showed that if a graph H is a retract of a graph G, then H must be an induced subgraph of G. Next we defined a class of absolute retracts based on this necessary condition, $\mathcal{AR}_z$, and then we studied this class. Now we will show that if a graph H is a retract of graph G, then G must preserve the distances in H. This leads to absolute retracts with respect to isometry, which has been well studied. We will present a list of equivalences for this type of absolute retract, picking out the characteristics that we are interested in. In particular, we will justify our interest in chordal graphs. We will finish this section by presenting basic results concerning chordal graphs and the related graphs we are interested in, namely graphs that admit near unanimity functions and dismantlable graphs.

1.2.1 Absolute retracts with respect to isometry

In this section we present results concerning absolute retracts with respect to isometry and justify our interest in chordal graphs.

Let H be a graph and let G be a supergraph of H. Let x and y be vertices of H. Recall that $d_G(x, y) \leq d_H(x, y)$. Assume that H is a retract of G and let $\theta : G \rightarrow H$ be a retraction. Let $x = u_0 u_1 \dots u_p = y$ be a shortest $x - y$ path in G; thus $d_G(x, y) = p$. As θ is a homomorphism, $\theta(u_0), \theta(u_1), \dots, \theta(u_p)$ is a walk from $\theta(u_0) = x$ to $\theta(u_p) = y$ in H. Hence $d_H(x, y) \leq d_G(x, y)$ and so $d_H(x, y) = d_G(x, y)$. Thus H must be an isometric subgraph of G. Therefore a necessary condition for H to be retract of G is for G to preserve the distances in H, i.e., H must be an isometric subgraph of G. Now we can use this necessary condition to create a class of absolute retracts. As we mentioned when we gave the definition of a retraction, we will restriction our attention to connected graphs.

The class of *absolute retracts with respect to isometry*, denoted by $\mathcal{AR}_I$, is the set of all connected graphs H such that H is a retract of a connected supergraph G

whenever H is an isometric subgraph of G.

Note that preserving distances is not a sufficient condition for the existence of a retraction. Let H be the graph in Figure 1.2 that is induced by the round vertices, and let G be the entire graph. Then H is an isometric subgraph of G, but H is not a retract of G.

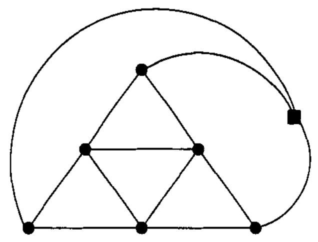


Figure 1.2: The graph on the round vertices is not a retract of the entire graph even though the graph on round vertices is an isometric subgraph of the entire graph.

Let H be a graph that is in $\mathcal{AR}_Z$. Then H is retract of a connected supergraph G whenever H is an induced subgraph of G. Let G a supergraph of H such that H is an isometric subgraph of G. Then H must obviously be an induced subgraph of G. Therefore H is a retract of G and hence $\mathcal{AR}_Z \subseteq \mathcal{AR}_I$. Recall that all graphs in $\mathcal{AR}_Z$ have a universal vertex by Theorem 1.1. Therefore all graphs with universal vertices are in $\mathcal{AR}_I$. For example, this implies that wheels are in $\mathcal{AR}_I$ as the hub of a wheel is a universal vertex.

Lemma 1.1. *Let W_k be a wheel on k rim vertices, $k \geq 4$. Then $W_k \in \mathcal{AR}_I$.*

As mentioned at the beginning of the chapter, research on absolute retracts has also been conducted for *irreflexive graphs*, i.e., graphs without loops at any vertex. In particular, much of the research in this area has been in with respect to *bipartite graphs*, irreflexive graphs that do not have odd cycles as subgraphs. We will discuss briefly some necessary conditions for retracts on irreflexive graphs.

Let H and G be irreflexive graphs such that H is subgraph of G. Clearly H being an isometric subgraph of G is still a necessary condition for H to be a retract of G; the reasoning we applied to reflexive graphs carries over as we made no use of loops. Since we are dealing with irreflexive graphs, homomorphisms may not send the endpoints of an edge to the same vertex. Thus chromatic numbers come into play as follows: suppose that H has a proper n-colouring. It is well known that an n-colouring of H is equivalent to $H \longrightarrow K_n$, where in this instance we mean by K_n the complete irreflexive graph on n vertices. If H is a retract of G, then $G \longrightarrow H$. Thus $G \longrightarrow K_n$, implying that G also admits a n-colouring. Conversely, if G admits an n colouring so does H, since H is a subgraph of G. Therefore H and G must have the same chromatic number. Now we can define n-chromatic absolute retracts; the class of *n-chromatic absolute retracts*, denoted by $\mathcal{ARI}_n$, is the set of all connected n-chromatic irreflexive graphs H such that H is a retract of a connected supergraph G whenever G has chromatic number n and H is an isometric subgraph of G. The class $\mathcal{ARI}_n$ has been studied in [8, 59, 60, 62]. It is well know that bipartite graphs are exactly those irreflexive graphs that admit a two colouring. The class of *absolute bipartite retracts with respect to isometry*, denoted by $\mathcal{ARI}_I$, is the set of all connected bipartite graphs H such that H is a retract of a connected bipartite supergraph G whenever H is an isometric subgraph of G. The class $\mathcal{ARI}_I$ has been studied in [3, 6, 40, 41, 42]. In [4], the authors explored the strong similarities between the two classes $\mathcal{AR}_I$ and $\mathcal{ARI}_I$, and their many characterizations. Pesch has written a monograph [61] studying absolute retracts, both reflexive and irreflexive, with respect to isometry.

Now we return to $\mathcal{AR}_I$, which has been investigated extensively [5, 9, 43, 44, 45, 56, 57, 61, 65]. We will present some of the results concerning $\mathcal{AR}_I$. But first, we need one last definition.

Let $\mathcal{F}$ be a finite family of finite sets. We say that $\mathcal{F}$ has the *Helly property* if for any subfamily $\mathcal{F}'$ of $\mathcal{F}$ the following holds: $\cap_{X\in\mathcal{F}'} X \neq \emptyset$ whenever $X \cap Y \neq \emptyset$ for all $X, Y \in \mathcal{F}'$. A graph H is *clique Helly* if the set of maximal cliques of H has the Helly property.

Theorem 1.2. *[5, 9, 43, 44, 57, 65] Let H be a connected graph. Then the following statements are equivalent:*

i.) $H \in \mathcal{AR}_I$.

ii.) the discs of H centred at vertices have the Helly property.

iii.) H is dismantlable and clique Helly.

iv.) H is in the variety generated by paths.

v.) H admits a majority function.

vi.) H has no k-holes, $k \geq 3$, see Chapter 2

vii.) for every diametrical vertex $x \in V(H)$, x is dismantlable and $H \setminus x \in \mathcal{AR}_I$.

The equivalence of statements *i*, *ii* and *vii* is proved in [5], the equivalence of statements *iii* and *ii* is proved in [9], the equivalence of statements *iv* and *vi* is proved in [57], the equivalence of statements *v* and *i* is proved in [44], and the equivalence of statements *vi* and *i* is proved in [43] (cf. [65]).

There are four items from the list of equivalences in Theorem 1.2 that we are particularly interested in. To begin with, as $\mathcal{AR}_I$ is the variety generated by paths, we know that $\mathcal{AR}_I$ is a variety. Also, all graphs in $\mathcal{AR}_I$ are dismantlable and all graphs in $\mathcal{AR}_I$ admit a near unanimity function of arity 3. Lastly, every graph in $\mathcal{AR}_I$ is made of paths. We will expand upon each of these points, explaining their significance.

We have defined $\mathcal{AR}_I$ to be the class of all connected graphs H such that H is a retract of a connected supergraph G whenever H is an isometric subgraph of G. Thus it is very logical that $\mathcal{AR}_I$ is closed under taking retracts. The reason why $\mathcal{AR}_I$ is closed under taking products is not as immediate, but it follows from projections being homomorphisms. In the following chapters, when studying $\mathcal{AR}_N$ for various necessary conditions N, we will provide direct proofs that $\mathcal{AR}_N$ is a variety.

The class of graphs $\mathcal{AR}_I$ is the intersection of dismantlable graphs and clique Helly graphs by Theorem 1.2. Thus $\mathcal{AR}_I$ is contained in the class of dismantlable graphs. As we look at $\mathcal{AR}_N$ for different necessary conditions N, we will investigate whether this containment is maintained.

The class of graphs $\mathcal{AR}_I$ is exactly the set of all connected graphs that admit a near unanimity function of arity 3. As will be seen in the next section, this implies that graphs in $\mathcal{AR}_I$ admit a near unanimity function of arity k, $k \geq 3$. If H is in $\mathcal{AR}_N$, for some necessary condition N, we will investigate whether H admits a near unanimity function; we will also investigate bounds on the arity of such a function, if it exists.

Lastly, Theorem 1.2 tells us that a graph $H \in \mathcal{AR}_I$ can be made of products and retracts of paths. For any class of graphs $\mathcal{C}'$ such that $\mathcal{C}'$ contains all paths, the variety generated by paths is contained in the variety generated by $\mathcal{C}'$. If in addition, $\mathcal{C}' \subseteq \mathcal{AR}_I$, then the variety generated by $\mathcal{C}'$ is a subset of $\mathcal{AR}_I$, as the variety generated by $\mathcal{AR}_I$ is $\mathcal{AR}_I$. Thus, for such a class $\mathcal{C}'$, the variety generated by $\mathcal{C}'$ would be equal to the variety generated by paths, by Theorem 1.2. We would like to find a class of graphs $\mathcal{C}$ such that $\mathcal{C}$ generalizes paths, has properties that make it easily recognizable but is not contained in $\mathcal{AR}_I$. Note that $\mathcal{C}$ containing the class of paths and $\mathcal{C} \setminus \mathcal{AR}_I \neq \emptyset$ implies that the variety generated by $\mathcal{C}$ strictly contains $\mathcal{AR}_I$. Now we need to find a such a class $\mathcal{C}$.

A very reasonable candidate for $\mathcal{C}$ is the class of all trees; all paths are trees and trees are easy to recognize. However, all trees are in $\mathcal{AR}_I$, [56, 65] and so we must look expand our search for $\mathcal{C}$. Note that trees are graphs that don't have cycles. *Interval graphs*, those graphs that are the intersection graphs of intervals on the real line, have cycles, but no induced nontrivial cycles (see [37]). Unfortunately, the class of connected interval graphs is still not large enough to be our desired class $\mathcal{C}$. This is demonstrated in the following proposition, which follows easily from Theorem 1.2.

Proposition 1.1. *Let H be a connected interval graph. Then H is in $\mathcal{AR}_I$.*

Proof. By Theorem 1.2, it is sufficient to prove that connected interval graphs are dismantlable and clique Helly.

Let $\{h_1, \ldots, h_n\}$ be the vertex set of H and let $I_1, \ldots, I_n$ be intervals on the real line such that $h_ih_j \in E(H)$ if and only if $I_i \cap I_j \neq \emptyset$. Let $r(I_i)$ be the right boundary point of the interval I_i.

Without loss of generality, we may assume that the intervals $I_1, \ldots, I_n$ are ordered such that $r(I_1) \leq r(I_2) \leq \ldots \leq r(I_n)$. Now consider the corresponding ordering $h_1, \ldots, h_n$ of the vertices of H. We claim that h_i is simplicial in H_{i-1}, $i = 1, \ldots, n$. Let h_j and h_k be neighbours of h_i, where $i < j < k$. Thus $I_i \cap I_k \neq \emptyset$, $I_i \cap I_j \neq \emptyset$ and $r(I_i) < r(I_j) < r(I_k)$. Therefore $r(I_i) \in I_j \cap I_k$, implying that $h_k h_j \in E(H)$. As h_i is simplicial in H_{i-1}, h_i is clearly dismantlable in H_{i-1}, $i = 1, \ldots, n-1$.

The maximal cliques of H can be ordered $S_1, \ldots, S_p$ such that if $S_i \cap S_j \neq \emptyset$, then $S_i \cap S_k \cap S_j \neq \emptyset$ for all k, $i \leq k \leq j$, [52]. Thus H must be clique Helly.

□

Our next candidate for $\mathcal{C}$ is the class of connected strongly chordal graphs. Let H be a graph. Then $h_1, \ldots, h_n$ is a *strong elimination ordering* of the vertices of H if $h_i h_k, h_j h_k \in E(H)$, where $i > j > k$, implies that $h_i h_j \in E(H)$, i.e., h_k is simplicial in H_{k-1}, and $h_i h_p, h_j h_p, h_j h_k \in E(H)$, where $i > j > k > p$, implies that $h_i h_k \in E(H)$. A graph that admits a strong elimination ordering is called *strongly chordal* [25]. Interval graphs are strongly chordal [25] and so we have a larger class of graphs. Strongly chordal graphs are also recognizable in polynomial time [25].

A graph J on $2n$ vertices, $n \geq 3$, is a *trampoline* [25] (or *complete sun* [19]) if the vertex set of J can be partitioned into two sets $W = \{w_1, \ldots, w_n\}$ and $U = \{u_1, \ldots, u_n\}$ such that W is independent, U is a clique and w_j is adjacent to u_i if and only if $i = j$ or $i \equiv j + 1 \bmod n$.

Theorem 1.3. *[19, 25] A graph H is strongly chordal if and only if H is chordal and has no induced trampoline.*

Unfortunately, there do not exist connected strongly chordal graphs outside of $\mathcal{AR}_I$; again, this follows easily from Theorem 1.2.

Proposition 1.2. *Let H be a connected strongly chordal graph. Then H is in $\mathcal{AR}_I$.*

Proof. By Theorem 1.2, it is sufficient to prove that connected strongly chordal graphs are dismantlable and clique Helly. It is easy to see that connected strongly chordal graphs are dismantlable; a strong elimination ordering of a connected graphs is also a dismantling ordering. Hence all that is left is to prove that strongly chordal graphs are clique Helly.

Let H be a strongly chordal graph and suppose that H is not clique Helly. Thus there exists maximal cliques $S_1, \ldots, S_k$ such that $S_i \cap S_j \neq \emptyset$, $1 \leq i, j \leq k$ and $\cap_{i=1}^{k} S_i = \emptyset$. We may assume that for $p = 1, \ldots, k$, there exits a vertex $u_p \in \cap_{i \neq p} S_i$. Since $S_1 \cap S_2 \neq \emptyset$ and $\cap_{i=1}^{k} S_i = \emptyset$, we must have that $k \geq 3$. Consider the vertices u_1, u_2 and u_3. The set $\{u_1, u_2, u_3\}$ is a clique as $u_i, u_j \in S_p$, where $i, j \neq p$. By the way we chose u_i, there exists a vertex $w_i \in S_i \backslash N(u_i)$, $i = 1, 2, 3$. Observe that $u_2, u_3, w_1 \in S_1$. Thus $u_2, u_3 \in N(w_1)$. Similarly, $u_i, u_j \in N(w_p)$, for $i, j \neq p$, $1 \leq i, j, p \leq 3$. If $w_i w_j \in E(H)$, then $u_i u_j w_i w_j u_i$ is an induced 4 cycle in H,$1 \leq i \neq j \leq 3$, which is a contradiction as strongly chordal graphs are chordal by Theorem 1.3. Thus $\{w_1, w_2, w_3\}$ is independent. Relabel the vertices w_1, w_2 and w_3 as follows, $w'_1 = w_3$, $w'_2 = w_1$ and $w'_3 = w_2$. Then the subgraph J of H induced by $\{u_1, u_2, u_3\} \cup \{w'_1, w'_2, w'_3\}$ is a trampoline, contradiction.

□

We still have not been able to find our desired class of graphs $\mathcal{C}$, but we do have the following result:

Corollary 1.1. *The following classes of graphs are the same:*

i.) $\mathcal{AR}_I$

ii.) the variety generated by paths.

iii.) the variety generated by trees.

iv.) the variety generated by connected interval graphs.

v.) the variety generated by connected strongly chordal graphs.

Now consider chordal graphs. All strongly chordal graphs are chordal by Theorem 1.3 and chordal graphs have many nice algorithmic properties, see [37], one of which is polynomial time recognition. The graph induced by the round vertices in Figure 1.2 is chordal, yet is not in $\mathcal{AR}_I$ as mentioned previously. Thus the variety generated by connected chordal graphs is not contained in $\mathcal{AR}_I$. Hence the variety generated by connected chordal graphs strictly contains $\mathcal{AR}_I$. We will study the intersection of the variety generated by connected chordal graphs and $\mathcal{AR}_N$ for various necessary conditions N. In the last chapter, we will present a class of graphs which generates a variety that strictly contains the variety generated by connected chordal graphs and we will prove that this new class is a particular type of absolute retract.

1.2.2 Dismantlability, near unanimity functions and chordal graphs

In this section, we will provide some basic facts about dismantlable graphs, graphs that admit near unanimity functions and chordal graphs; in particular, we will mention how these graph classes compare to each other. For further information on these graph classes and others, see [13].

Dismantlable graphs have been studied in [1, 9, 16, 26, 36, 58, 63, 65]. The following lemmas and theorem concerning dismantlable graphs closely resemble results on dismantlable posets [22].

Lemma 1.2. *If a graph H is not dismantlable, then H is ramified or there exists vertices $h_1, \ldots, h_k$ of H such that h_i is dismantlable in H_{i-1} for $i = 1, \ldots, k$ and H_k is ramified. Moreover, H_k is a retract of H.*

Proof. The lemma is trivial if H is ramified.

Assume that H is not ramified. Then H has a dismantlable vertex, call it h_1. Now consider H_1. If H_1 has a dismantlable vertex, call it h_2. Continue on in the manner until a partial ordering $h_1, \ldots, h_k$ has been created such that h_i is dismantlable in H_{i-1} for $i = 1, \ldots, k$ and H_k has no dismantlable vertices. As H is not dismantlable, H_k has at least two vertices and hence H_k is ramified.

As h_i is dismantlable in H_{i-1}, there exists a vertex $h_i' \in V(H_i)$ such that h_i' covers h_i in H_{i-1}, $i = 1, \ldots, k$. Let $\theta_i : H_{i-1} \to H_i$ be the map defined by

$$\theta_i(h) = \begin{cases} h & \text{if } h \neq h_i \\ h_i' & \text{if } h = h_i. \end{cases}$$

Clearly θ_i is a homomorphism. Let $\theta : H \longrightarrow H_k$ be the map $\theta = \theta_k \circ \theta_{k-1} \circ \ldots \circ \theta_1$. Then again θ must be a homomorphism and moreover $\theta(h) = h$ for all $h \in V(H_k)$. Thus θ is a retraction from H to H_k.

□

Lemma 1.3. *[49] Let H be a connected graph. Then H is dismantlable if and only if no retract of H is ramified.*

Proof. Let H be a dismantlable graph. We will prove that no retract of H is ramified by induction on the number of vertices of H.

Let H' be a retract of H and let $\theta : H \longrightarrow H'$ be a retraction. Let x be the first vertex of a dismantling ordering of H and let y be a vertex of H that covers x in H. If $x \notin V(H')$, then H' is clearly a retract of $H \setminus x$ and so H' is not ramified by induction. Therefore we may assume that $x \in V(H')$. If x is dismantlable in H', there is nothing to prove, and so assume that x is not dismantlable in H'. As $\theta(z) = z$ for all $z \in V(H')$ and as θ is a homomorphism, $\theta(y)$ must be adjacent to every neighbour of x in H'. Since we have assumed that x is not dismantlable in H', this implies that $\theta(y) = x$. In other words, $y \notin V(H')$. Let H'' be the subgraph of H induced by $(V(H') \setminus \{x\}) \cup \{y\}$. We claim that H'' and H' are isomorphic. To prove this, we need only prove that $N_{H''}(y) \setminus \{y\} = N_{H'}(x) \setminus \{x\}$. As y covers x in H, clearly $N_{H''}(y) \setminus \{y\} \supseteq N_{H'}(x) \setminus \{x\}$. Now let w be a nontrivial neighbour of y in H''. Thus $w \in V(H')$ and so $\theta(w) = w$. Since θ is a homomorphism, $yw \in E(H)$ implies that $\theta(y)\theta(w) \in E(H')$. As $\theta(y) = x$, w is a neighbour of x in H'. Therefore $N_{H''}(y) \setminus \{y\} \subseteq N_{H'}(x) \setminus \{x\}$. Thus H' and H'' are isomorphic. Define the map $\theta' : H \setminus x \longrightarrow H''$ as follows:

$$\theta'(z) = \begin{cases} y & \text{if } \theta(z) = x \\ \theta(z) & \text{otherwise.} \end{cases}$$

Clearly θ' is a retraction. Thus, by induction H'' is not ramified, and hence neither is H'.

Now assume that no retract of H is ramified. Then, by Lemma 1.2, H must be dismantlable.

□

Theorem 1.4. *The class of dismantlable graphs is a variety.*

Proof. We will first prove that the class of dismantlable graphs is closed under taking retractions, and then secondly we will prove that the class of dismantlable graphs is closed under taking products.

Let H be a dismantlable graph, and let H' be retract of H. If H' is not dismantlable, then H' has a ramified retract H'' by Lemma 1.2. As H' is a retract of H, so is H''. This contradicts Lemma 1.3. Hence H' must be dismantlable.

Let $H = H_1 \times H_2$, where H_i is dismantlable, $i = 1, 2$. Let $x_1, \ldots, x_{n_1}$ be a dismantling ordering for H_1 and let $y_1, \ldots, y_{n_2}$ be a dismantling ordering for H_2. Then it can be easily checked that the following ordering of the vertices of H is a dismantling ordering for H :

$$(x_1, y_1), (x_1, y_2), \ldots, (x_1, y_{n_2}),$$
$$(x_2, y_1), (x_2, y_2), \ldots, (x_2, y_{n_2}), \ldots, (x_{n_1}, y_1), (x_{n_1}, y_2), \ldots, (x_{n_1}, y_{n_2}).$$

□

Graphs admitting majority functions have been studied in [6, 56], and lately there has been interest in graphs that admit near unanimity functions of arity k, $k \geq 3$, see [15, 49]. There has long been interest in near unanimity functions in other areas such as posets see [38, 51, 64, 71] and universal algebras see [2, 7, 20, 46, 54]. We present some folklore results concerning near unanimity functions on graphs and varieties, plus two new theorems from [49]; one theorem we can use to identify graphs that admit near unanimity functions in polynomial time and the other theorem relates graphs that admit near unanimity functions to dismantlable graphs.

Let η be a near unanimity function of arity k on a graph H, where $k \geq 3$. Since $k \geq 3$, the value of $\eta(x, \ldots, x, y)$ is well defined; if $k = 2$, then both x and y would appear $k-1$ times in the k-tuple and η would not be well defined.

Lemma 1.4. *The class of graphs that admit a near unanimity function of arity k, $k \geq 3$, is a variety.*

Proof. We will proceed by proving that the class of graphs that admit a near unanimity function of arity k is closed under taking retractions, and then prove that it is closed under taking products.

Let H be a graph that admits a near unanimity function $\eta : H^k \longrightarrow H$, and let $\theta : H \longrightarrow J$ be a retraction. Let $\eta' : J^k \longrightarrow J$ be the function defined by $\eta'(\boldsymbol{x}) = \theta \circ \eta(\boldsymbol{x})$ for all vertices $\boldsymbol{x} \in V(J^k)$. It is easy to see that η' is a near unanimity function of arity k. Therefore, the class of graphs that admit a near unanimity function of arity k is closed under taking retractions.

Let H be the product of two graphs that admit near unanimity functions of arity k, say $H = H_1 \times H_2$. Note that a vertex of H^k is a vector of the form $((x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k))$, where $(x_1, \ldots, x_k)$ is a vertex of H_1^k and $(y_1, \ldots, y_k)$ is a vertex of H_2^k. Let $\eta_i : H_i^k \longrightarrow H_i$ be a particular near unanimity function on H_i, $i = 1, 2$. Now we can construct a function η of arity k on H as follows:

$$\eta((x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)) = (\eta_1(x_1, \ldots, x_k), \eta_2(y_1, \ldots, y_k)).$$

We will prove that η is a near unanimity function on H. The function η is clearly a homomorphism as η_i is a homomorphism for $i = 1, 2$. Suppose that at least $k-1$ of the pairs (x_i, y_i) are the pair (x, y). Then at least $k-1$ of the x_i's are x. Therefore $\eta_1(x_1, \ldots, x_k) = x$. Similarly $\eta_2(y_1, \ldots, y_k) = y$. Hence

$$\begin{aligned} \eta((x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)) &= (\eta_1(x_1, \ldots, x_k), \eta_2(y_1, \ldots, y_k)) \\ &= (x, y). \end{aligned}$$

Therefore η is a near unanimity function and the class of graphs that admit a near unanimity function of arity k is closed under taking products.

□

Lemma 1.5. *Let H be a graph that admits a near unanimity function of arity k, $k \geq 3$. Then H admits a near unanimity function of arity $k+1$.*

Proof. Let $\eta : H^k \longrightarrow H$ be a near unanimity function. Define the function $\hat{\eta} : H^{k+1} \longrightarrow H$ as follows:

$$\hat{\eta}(x_1, \ldots, x_{k+1}) = \eta(x_1, \ldots, x_k).$$

If the vertices $(x_1, \ldots, x_{k+1})$ and $(y_1, \ldots, y_{k+1})$ are adjacent in H^{k+1}, then the vertices $(x_1, \ldots, x_k)$ and $(y_1, \ldots, y_k)$ are adjacent in H^k. As η is a homomorphism, $\hat{\eta}(x_1, \ldots, x_{k+1})$ and $\hat{\eta}(y_1, \ldots, y_{k+1})$ are adjacent in H. Thus $\hat{\eta}$ is a homomorphism.

Let $(x_1, \ldots, x_{k+1})$ be a vertex of $V(H^{k+1})$ and suppose that at least k of the x_i's are the vertex $x \in V(H)$. Then at least $k-1$ of the x_i's in $(x_1, \ldots, x_k)$ are the vertex $x \in V(H)$. As η is a near unanimity function of arity k on H, $\eta(x_1, \ldots, x_k) = x$, and so $\hat{\eta}(x_1, \ldots, x_{k+1}) = x$. Therefore $\hat{\eta}$ is a near unanimity function of arity $k+1$ on H.

□

Theorem 1.5. *The class of graphs that admit a near unanimity function is a variety.*

Proof. We will prove that the class of graphs that admit near unanimity functions is closed under retractions and products.

Let H be a graph that admits a near unanimity function of arity k. Then by Lemma 1.4, all retracts of H also admit a near unanimity function of arity k.

Let $H = H_1 \times H_2$, where H_i admits a near unanimity function of arity p_i, $i = 1, 2$. Then by Lemma 1.5, H_1 and H_2 both admit a near unanimity function of arity $p = \max\{p_1, p_2\}$. Therefore H admits a near unanimity function of arity p, by Lemma 1.4.

□

Theorem 1.6. *[49] Let H be a connected graph and let $\tilde{H}$ be the copy of H in H^2 on the constant vertices. Then H admits a near unanimity function if and only if H^2 dismantles to $\tilde{H}$.*

Theorem 1.7. *[49] Each connected graph that admits a near unanimity function is dismantlable.*

Chordal graphs have also been studied under the following names: triangulated graphs, rigid-circuit graphs, monotone transitive graphs and perfect elimination graphs (see [13]). Orderings of the vertices of graphs will play an important role in much of the analysis we will do on chordal graphs. Let H be graph and let $h_1, \dots, h_n$ be a ordering of all of the vertices of H. Recall that $H_i = H \setminus \{h_1, \dots, h_i\}$ for $i = 1, \dots, n-1$ and that $H_0 = H$. If $h_1, \dots, h_n$ is an ordering of all the vertices of H such that h_i is simplicial in H_{i-1} for $i = 1, \dots, n$, then $h_1, \dots, h_n$ is a *perfect elimination ordering of H*. We say that a graph H *admits a perfect elimination ordering* if there exists a perfect elimination ordering of its vertices.

Theorem 1.8. *[21, 34, 37, 66] Let H be a graph. Then the following statements are equivalent:*

i.) H is chordal.

ii.) Every minimal vertex separator of every induced connected subgraph of H is a clique.

iii.) Every induced subgraph of H has a simplicial vertex.

iv.) H admits a perfect elimination ordering.

The equivalence of statements i and ii is proved in [21], the equivalence of statements i and iii is proved in [66], and the equivalence of statements i and iv is proved in [34]. The equivalence of i, ii and iv are presented in the book by Golumbic [37].

Proof. We will prove that the above statements are equivalent by showing $i \Rightarrow ii \Rightarrow iii \Rightarrow iv \Rightarrow i$.

$\underline{i \Rightarrow ii.}$ Assume that H is a chordal graph. Without loss of generality, assume that H is connected. Suppose there exists a minimal vertex separator K in H that is not a clique; thus K is a minimal $x - y$ separator for some $x, y \in V(H)$. Let H_x and H_y be the components of $H \setminus K$ that contain x and y respectively.

Suppose that there exists a vertex z in K that doesn't have a neighbour in H_x. Thus all paths from x to y in H must contain a vertex from $K' = K \setminus \{z\}$, implying that K' is a smaller $x-y$ separator in H, contradicting the minimality of K. Therefore each vertex of K has a neighbour in H_x. Similarly, each vertex of K has a neighbour in H_y.

As we have assumed that K is not a clique, it contains vertices u and v that are not adjacent in H. As u and v both have neighbours in H_x, there exists a chordless path P_x in H from u to v whose internal vertices are in $V(H_x)$. Similarly, there exists a path P_y in H from v to u whose internal vertices are in $V(H_y)$. Note that as H_x and H_y are disjoint, we can concatenate P_x and P_y to form a cycle C. Moreover, C contains at least 4 distinct vertices and must be chordless by construction. This is a contradiction as H is chordal. Hence every minimal vertex separator of every induced connected subgraph of H is a clique.

ii $\Rightarrow$ *iii.* Instead of proving *ii* $\Rightarrow$ *iii*, we will prove the following claim:

> If every minimal vertex separator of every induced connected subgraph of H is a clique, then if H is not a complete graph, H has two non-adjacent simplicial vertices [21].

Note that *ii* $\Rightarrow$ *iii* follows from the claim. Let H be a graph such that every minimal vertex separator of every induced connected subgraph of H is a clique. Then either H is a complete graph, and all vertices are simplicial, or H is not a complete graph, and by the claim, H has two non-adjacent simplicial vertices. Since any induced subgraph H' of H inherits the property that every minimal vertex separator of every induced connected subgraph of H is a clique, every induced subgraph H' of H has a simplicial vertex.

Let H be a graph such that every minimal vertex separator of every induced connected subgraph of H is a clique. Assume that H is not complete. Thus there exist non-adjacent vertices x and y in H. Let K be a minimal $x - y$ separator in H, and let H_x and H_y be the components of $H \setminus K$ that contain x and y, respectively. Let H' be the subgraph of H induced by $V(H_x) \cup K$. By induction on the number of vertices of H, either H' is complete or H' has non-adjacent simplicial vertices. If H' is complete, then all vertices of H' are simplicial (in H'). Now assume that H'

is not complete and thus it has two non-adjacent vertices. Since K is a clique, one of these simplicial vertices must be in $V(H_x)$. By construction, all the neighbours of vertices in $V(H_x)$ in H are contained in $K \cup V(H_x)$, and there exists a vertex, say x' in $V(H_x)$ that is simplicial H. Similarly, there exists a vertex, say y', in $V(H_y)$ that is simplicial in H. Again by construction, x' and y' can't be adjacent, and we have produced non-adjacent simplicial vertices in H, contradiction. Therefore, the claim must be true.

iii $\Rightarrow$ *iv.* Assume that every induced connected subgraph of H has a simplicial vertex. Thus, in particular, H has a simplicial vertex, call in h_1. The graph H_1 is an induced subgraph of H, and so H_1 has a simplicial vertex, call it h_2. We can continue in this manner, creating the ordering $h_1, \ldots, h_n$ of all the vertices of H, where h_i is simplicial in H_{i-1} for $i = 1, \ldots, n$. Therefore $h_1, \ldots, h_n$ is a perfect elimination ordering of the vertices of H.

iv $\Rightarrow$ *i.* Assume that H admits a perfect elimination ordering $h_1, \ldots, h_n$. Let C be an induced cycle in H, and let h_i be the vertex of C with lowest index. Thus $C \subseteq H_{i-1}$ and so the neighbours of h_i in C must be adjacent. Therefore C must be a trivial cycle and so H is chordal.

□

The following lemma is a well known property of chordal graphs.

Lemma 1.6. *(see [18]) Let C be a cycle of a chordal graph H. Then for each edge uv of C, there exists a common nontrivial neighbour w of u and v on C.*

The next theorem will be used when we present the idea of convexity in Chapter 2 with regard to chordal graphs.

Theorem 1.9. *[24] Let H be a chordal graph. Then every non-simplicial vertex of H lies on a chordless path between two simplicial vertices.*

Proof. Let z be a non-simplicial vertex of H. Thus z has non-adjacent neighbours x and y. Let K be a minimal $x - y$ separator in H; clearly $z \in K$. Let H_x and H_y be the components of $H \setminus K$ that contain x and y respectively. By the arguments in the proof of Theorem 1.8 *ii* $\Rightarrow$ *iii*, there exist vertices $x' \in V(H_x)$ and $y' \in V(H_y)$ such that x' and y' are simplicial in H. As $x \in V(H_x)$ and $y \in V(H_y)$, the vertex z has a neighbour in $V(H_x)$ and a neighbour in $V(H_y)$. Then there exists a chordless path P_x in H from x' to z such that all vertices on P_x are in $V(H_x) \cup \{z\}$. Similarly, there exists a chordless path P_y from z to y' such that all vertices on P_y are in $V(H_y) \cup \{z\}$. Now let P be the path from x' to y' in H formed by concatenating the paths P_x and P_y. The path P must also be a chordless path as there can be no edge from a vertex in $V(P_x \setminus z) \subseteq V(H_x)$ to an edge of $V(P_y \setminus z) \subseteq V(H_y)$.

□

We will finish by relating chordal graphs to dismantlable graphs and those graphs that admit near unanimity functions.

Let H be a connected chordal graph. By Theorem 1.8, the graph H admits a perfect elimination ordering $h_1, \ldots, h_n$. Since the neighbourhood of h_i is a clique in H_{i-1} for $i = 1, \ldots, n$ and since H is connected, the vertex h_i is covered in H_{i-1} for $i = 1, \ldots, n-1$. Thus a perfect elimination ordering is also a dismantling ordering, and hence connected chordal graphs are obviously dismantlable. The relationship between chordal graphs and those graphs that admit near unanimity functions is not as easy to derive, and so we only quote the following result:

Theorem 1.10. *[15] Let H be a chordal graph. Then H admits a near unanimity function of arity k, where $k \leq |V(H)|$*

Note that we can derive the dismantlability of connected chordal graphs from Theorem 1.10; since all chordal graphs admit a near unanimity function, and since all connected graphs that admit a near unanimity function are dismantlable by Theorem 1.7, all connected chordal graphs are dismantlable.

Chapter 2

Holes

A *distance constraint* on a connected graph H is a function f with domain $D_f \subseteq V(H)$, whose values are non-negative integers. A *complete filler* of the distance constraint f is a vertex a such that $d(a, x) \leq f(x)$ for all $x \in D_f$. The set of all complete fillers of f is denoted by $F_H(f)$ (or just $F(f)$ if H is clear from the context). Recall that $D_H(x, k) = \{y \in V(H) \mid d_H(x, y) \leq k\}$. Thus

$$F_H(f) = \bigcap_{x \in D_f} D_H(x, f(x)).$$

We also need to consider vertices that 'almost' satisfy a distance constraint f on a graph connected H. If $z \in D_f$, then a *z-relaxed filler of f* is a vertex a such that $d(x, a) \leq f(x)$ for all $x \in D_f \setminus \{z\}$. Of course, any complete filler of f is a z-relaxed filler of f, for all $z \in D_f$.

A distance constraint f on a connected graph H is *feasible* if $F(f) \neq \emptyset$, and f is *infeasible* otherwise. We can define a partial order of distance constraints on a given connected graph H as follows: for two distance constraints f and f' on H we say that $f' \leq f$ if $D_{f'} \subseteq D_f$ and $f'(x) \geq f(x)$ for all $x \in D_{f'}$. Moreover, we write $f = f'$ if $D_{f'} = D_f$ and $f'(x) = f(x)$ for all $x \in D_{f'}$, and we write $f' < f$ if $f' \leq f$ and $f' \neq f$.

In particular, if f_x is the distance constraint with $D_{f_x} = D_f$ defined by

$$f_x(z) = \begin{cases} f(z)+1 & \text{if } z = x \\ f(z) & \text{otherwise,} \end{cases}$$

then $f_x < f$. Additionally, if f' is a distance constraint on H where $D_{f'}$ is a proper subset of D_f and $f'(x) = f(x)$ for all $x \in D_{f'}$, then $f' < f$.

Proposition 2.1. *Let H be a connected graph and let f a distance constraint on H. For any distance constraint f' on H such that*

i.) $f' \leq f$, we have that $F(f) \subseteq F(f')$.

ii.) $f' < f$, there exists a vertex $x \in D_f$ such that $f' \leq f_x < f$.

Proof. i. Let f' be a distance constraint on H such that $f' \leq f$. Then by definition $D_{f'} \subseteq D_f$ and $f'(z) \geq f(z)$ for all $z \in D_{f'}$. Therefore $D(z, f(z)) \subseteq D(z, f'(z))$ for all $z \in D_{f'}$ and so

$$\bigcap_{z \in D_{f'}} D_H(z, f(z)) \subseteq \bigcap_{z \in D_{f'}} D_H(z, f'(z)),$$

which implies that

$$\bigcap_{z \in D_f} D_H(z, f(z)) \subseteq \bigcap_{z \in D_{f'}} D_H(z, f'(z)).$$

Therefore $F(f) \subseteq F(f')$.

ii. Let f' be a distance constraint on H such that $f' < f$. Since $f' \neq f$, there exits a vertex $x \in D_f$ such that either $x \notin D_{f'}$ or $f'(x) > f(x)$. By the definition of f_x, $D_{f_x} = D_f$ with $f_x(x) = f(x) + 1$ and $f_x(z) = f(z)$, $z \neq x$. Hence $D_{f'} \subseteq D_{f_x}$. If $x \notin D_{f'}$, then clearly $f' \leq f_x$. Thus suppose that $x \in D_{f'}$. Then $f'(x) \geq f(x) + 1 = f_x(x)$ and $f'(z) \geq f(z) = f_x(z)$ for $z \in D_{f'} \setminus \{x\}$. Therefore again we have that $f' \leq f_x$. We already know that $f_x < f$ by the construction of f_x. Thus $f' \leq f_x < f$.

□

Let H be a connected graph and let f be a distance constraint on H. We say that f is minimally infeasible, or a *hole*, if f is an infeasible distance constraint on H and all distance constraints f' on H such that $f' < f$ are feasible. Thus, for a hole f and for each $x \in D_f$, the distance constraint f_x is feasible. The cardinality $|D_f|$ is called the *size* of the hole f. A *k-hole* is hole of size k. We call a k-hole f a *degenerate hole* if $k = 2$ and a *non-degenerate hole* otherwise.

Holes have been studied previously for graphs and posets, but with a slight difference. Let f be a distance constraint on a connected graph H such that f is infeasible but any distance constraint f' on H is feasible when $D_{f'} \subset D_f$ and $f'(x) = f(x)$ for all $x \in D_{f'}$. Then f is a hole on H in [43], a minimal hole on H in [44] and a gap in H in [57]. Related structures in posets have been studied under the names zigzag [71], hole [55] and gap [22]. Consider graph in Figure 2.1. Regard the numbers labeling

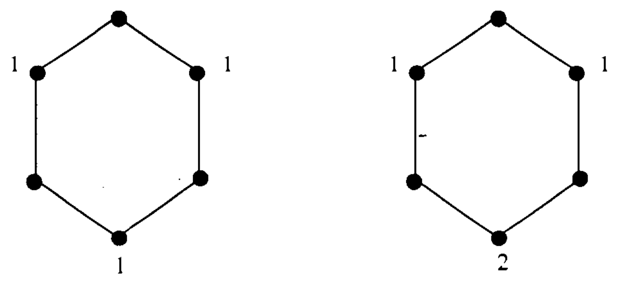


Figure 2.1: The difference between holes as defined in this thesis and related structures studied by others.

the vertices to be the values for distance constraints. Then both distance constraints are holes/minimal holes/gaps as defined in the papers mentioned above. However, only the distance constraint on the right is a hole as defined in this thesis.

Theorem 1.2 tells us that a connected graph $H \in \mathcal{AR}_I$ if and only if H has no k-holes, $k \geq 3$. This equivalence was originally stated using the definition of a hole as in [43]. It is not hard to see that the statement is still true using holes as defined in this thesis.

We end this section with more basic facts about holes on connected graphs.

Proposition 2.2. *Let H be a connected graph and let f be a hole on H.*

i.) Then for each $x \in D_f$, there exits an x-relaxed filler a_x such that $d(x, a_x) =$

$f(x)+1$.

ii.) If f is a degenerate hole, with say $D_f = \{x, x'\}$, then $d(x,x') = f(x)+f(x')+1$.

iii.) If f is a non-degenerate hole, then $d(x,x') \leq f(x)+f(x')$ for all distinct vertices $x, x' \in D_f$.

Proof. <u>i.</u> For each vertex x in D_f, consider the distance constraint f_x. Clearly $f_x < f$ with respect to the distance constraint poset on H, and so f_x must be feasible. Let a_x be a vertex in $F(f_x)$. Thus, $d(x,a_x) \leq f_x(x) = f(x)+1$ and $d(z,a_x) \leq f(z)$ for all $z \in D_f \setminus \{x\}$. Hence a_x is an x-relaxed filler of f in H. Since f is not feasible by definition, we must have $d(x,a_x) > f(x)$ and so $d(x,a_x) = f(x)+1$. Therefore, for each $x \in D_f$, there exits an x-relaxed filler a_x of f on H such that $d(x,a_x) = f(x)+1$.

<u>ii.</u> Suppose that f is a degenerate hole with $D_f = \{x, x'\}$. By the arguments in the previous paragraph, there exists a vertex a_x such that $d(x,a_x) = f(x)+1$ and $d(x',a_x) \leq f(x')$. Therefore $d(x,x') \leq f(x)+f(x')+1$. If $d(x,x') \leq f(x)+f(x')$, we could pick a vertex a on a shortest $x-x'$ path in H such that $d(x,a) \leq f(x)$ and $d(x',a) \leq f(x')$, contradicting the infeasibility of f. Hence $d(x,x') = f(x)+f(x')+1$.

<u>iii.</u> Suppose that f is a non-degenerate hole and let $x, x' \in D_f$ be distinct vertices. Let a be a z-relaxed filler for some vertex $z \in D_f \setminus \{x, x'\}$. Then $d(a,x) \leq f(x)$ and $d(a,x') \leq f(x')$. Thus $d(x,x') \leq f(x')+f(x)$.

□

For a non-degenerate hole f in a connected graph H, a vertex that is an x-relaxed filler for some $x \in D_f$ is called a *relaxed hole filler.*

Lemma 2.1. *Let f be a distance constraint on a connected graph H such that $f(x') = 0$ for some $x' \in D_f$. Then x' is the only possible x-relaxed filler of f for $x \in D_f \setminus \{x'\}$.*

□

Proposition 2.3. *Let f a non-degenerate hole on a connected graph H. Then the range of f is a set of positive integers.*

Proof. Let $D_f = \{x_1, \ldots, x_k\}$, and suppose that $f(x_1) = 0$. Note that $k \geq 3$ as f is a non-degenerate hole. Then by Lemma 2.1, x_1 is an x_i-relaxed filler for $i = 2, \ldots, k$. Specifically, x_1 is an x_2-relaxed filler and an x_3-relaxed filler; in other words, x is within the desired distance of each vertex of $D_f \setminus \{x_2\}$ and of $D_f \setminus \{x_3\}$. Thus x_1 is a complete filler. This contradicts the definition of a hole.

□

2.1 Absolute retracts with respect to holes

As the title of this section indicates, we will present a class of absolute retracts with respect to a necessary condition N, where N has to do with preserving holes. Let H be a connected graph and let G be a connected supergraph of H. Observe that a distance constraint f on H is also a distance constraint on G as $D_f \subseteq V(H) \subseteq V(G)$. We will show that if H is a retract of a G, then all holes on H must also be holes on G. We will then use this property to define the class of absolute retracts with respect to holes. Next we will present a proof that the class of absolute retracts with respect to holes is a variety. Lastly, given a graph H, we will construct a graph that in some sense "contains all the holes" of H. We will use this constructed graph to classify the graphs that are absolute retracts with respect to holes.

Proposition 2.4. *[45] Let H be a connected graph, and let G be a connected supergraph of H such that H is a retract of G. Let f be a distance constraint that is infeasible on H. Then f must also be infeasible on G.*

Proof. Suppose not; thus while $F_H(f)$ is empty, there exists a vertex $g \in F_G(f)$. By the definition of $F_G(f)$, $d_G(g, x) \leq f(x)$ for all $x \in D_f$. Let $\theta : G \to H$ be a retraction. Let $g = u_0u_1 \ldots u_p = x'$ be a shortest path in G from g to a vertex x' in D_f. Then $\theta(g) = \theta(u_0)\theta(u_1)\ldots\theta(u_p) = x'$ is a walk in H from $\theta(g)$ to x' of length $d_G(g, x')$. Therefore $d_H(\theta(g), x) \leq f(x)$ for all $x \in D_f$, contradicting the infeasibility of f on H.

□

Let H be a connected graph and let G be a connected supergraph of H. By Proposition 2.4, a necessary condition for H to be a retract of G is for each infeasible distance constraint on H to be an infeasible distance constraint on G. We will now show that each infeasible distance constraint on H being an infeasible distance constraint on G is equivalent to each hole on H being a hole on G.

Proposition 2.5. *Let H be a connected graph and let G be a connected supergraph of H. Then the following conditions are equivalent:*

i.) Each infeasible distance constraint on H is an infeasible distance constraint on G.

ii.) Each hole on H is a hole on G.

Proof. *$\underline{i \Rightarrow ii.}$* Assume that each infeasible distance constraint on H is an infeasible distance constraint on G. Let f be a hole on H. As f is an infeasible distance constraint on H, f is an infeasible distance constraint on G by assumption. Now we just have to prove that f is minimally infeasible on G. By Proposition 2.1, it is sufficient to prove that f_x is feasible on G for each $x \in D_f$. Since f is a hole on H, f_x is a feasible distance constraint on H for each $x \in D_f$. Hence there exists a vertex $a_x \in F_H(f_x)$. As H is a subgraph of G, we must have that $a_x \in F_G(f_x)$. Thus f_x is a feasible distance constraint on G for each $x \in D_f$.

$\underline{ii \Rightarrow i.}$ Assume that each hole on H is a hole on G. Suppose that there exists an infeasible distance constraint f on H that is feasible on G. There exists a hole f' on H such that $f' \leq f$ in the distance constraint poset on H. By Proposition 2.1, we have that $F_G(f) \subseteq F_G(f')$. Since f is feasible on G, there exits a vertex $g \in V(G)$ such that $g \in F_G(f)$, and so $g \in F_G(f')$. Therefore f' is feasible on G. This is a contradiction we assumed each hole on H is a hole on G, implying that f' must be infeasible on G.

□

Let H be a connected graph and let G be a connected supergraph of H. As mentioned after Proposition 2.4 a necessary condition for H to be a retract of G is for each infeasible distance constraint on H to be an infeasible distance constraint on G. By Proposition 2.5, it is equivalent to say that a necessary condition for H to be a retract of G for each hole on H to be a hole on G. This leads to the next class of absolute retracts.

The class of *absolute retracts with respect to holes*, denoted by $\mathcal{AR}_H$, is the set of all connected graphs H such that H is a retract of a connected supergraph G whenever each hole on H is also a hole on G.

Note that preserving holes is not a sufficient condition, as seen in Figure 2.6. The graph induced by the round vertices is not a retract of the entire graph even though each hole in the subgraph induced by the round vertices is a hole in the graph.

Let H be graph in $\mathcal{AR}_I$. Then H is a retract of a connected supergraph G whenever G preserves the distances in H. Let G be a supergraph of H such that each hole on H is a hole on G. Thus, in particular, each degenerate hole on H is a degenerate hole on G, and so H is an isometric subgraph of G. Therefore H is a retract of G and hence $\mathcal{AR}_I \subseteq \mathcal{AR}_H$. Hence, $\mathcal{AR}_H$ is a good generalization of $\mathcal{AR}_I$.

Similarly to [43], we will show that $\mathcal{AR}_H$ is a variety.

Theorem 2.1. *The class of graphs $\mathcal{AR}_H$ is a variety.*

Proof. We will first prove that $\mathcal{AR}_H$ is closed under taking retractions, and then secondly we will prove that $\mathcal{AR}_H$ is closed under taking products.

Let H be graph in $\mathcal{AR}_H$, and let H' be a retract of H; thus both H and H' are connected. Let G' be a connected supergraph of H' such that each infeasible distance constraint on H' is an infeasible distance constraint on G'. If we can prove that H' is a retract of G', then $H' \in \mathcal{AR}_H$ by Proposition 2.5.

We begin by noting that $H' \subseteq H \cap G'$, as H' is a subgraph of both H and G'. We may, without loss of generality, assume that in fact $H' = H \cap G'$ for the following reason; if there exists a vertex x in $H \cap G'$ that is not in H', we may replace H and G' with isomorphic graphs H_x and G_x, respectively, that are formed by replacing x with distinct vertices x_H and x_G that have the same neighbours as x in H and G', respectively.

Let $G = H \cup G'$. As H', H, and G' are all connected graphs, and as $H' = H \cap G'$, G is also connected. The graphs H', H, G' and G, and their relation to each other is depicted in Figure 2.2. If we can prove each infeasible distance constraint on H is an infeasible distance constraint on G, then H is retract of G by Proposition 2.5. We can use this to prove that H' is a retract of G'.

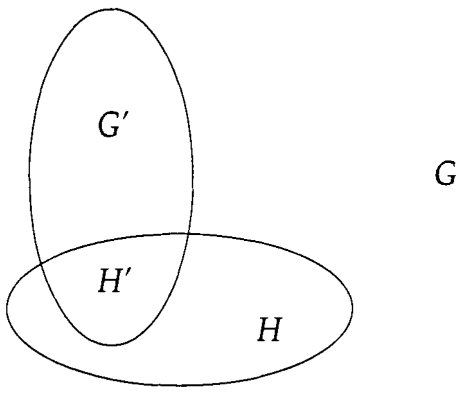


Figure 2.2: How the graphs H', H, G', and G relate to each other.

Let f be an infeasible distance constraint on H with $D_f = \{x_1, \ldots, x_k\}$. Suppose that there exists a vertex $a \in F_G(f)$. For each vertex x_i, choose a shortest $x_i - a$ path in G and let y_i be the first vertex of this path in H'; if $x_i \in V(H')$, then $x_i = y_i$. Let f' be the distance constraint on H' with $D_{f'} = \{y_1, \ldots, y_k\}$ where $f'(y_i) = f(x_i) - d_H(y_i, x_i)$ for $i = 1, \ldots, k$. As f is an infeasible distance constraint on H and H' is an isometric subgraph of H, we must have that $F_{H'}(f') = \emptyset$. By the way we defined G, the vertex a must be in G'. Thus $a \in F_{G'}(f')$, contradicting our choice of G'. Therefore each infeasible distance constraint on H is also an infeasible distance constraint on G. Hence H is a retract of G. Thus we have a homomorphism from G to H that fixes each vertex of H, and hence of H'. Since H' is a retract of H, we can obtain a retraction from G' to H' by composing a retraction from G to H and a retraction from H to H'.

Let $H = H_1 \times H_2$, where $H_1, H_2 \in \mathcal{AR}_H$. Let $\pi_i : H \to H_i$ be the i^{th} projection, $i = 1, 2$. Let G be a connected supergraph of H such that every infeasible distance constraint on H is an infeasible distance constraint on G. Let G_i be the graph we obtain from G by identifying all the vertices $h, h' \in V(H)$ such that $\pi_i(h) = \pi_i(h')$. Clearly H_i is a subgraph of G_i. We will prove that each infeasible distance constraint of H_i is also an infeasible distance constraint of G_i, thus implying existence of a retraction $\theta_i : G_i \longrightarrow H_i$, for $i = 1, 2$ by Proposition 2.5. We will then use these retractions to construction a retraction from G to H.

Let f_1 be an infeasible distance constraint on H_1 with domain $\{x_1, \ldots, x_k\}$. Let y be a vertex of H_2. Then the distance constraint f on H with domain $\{(x_1, y), \ldots, (x_k, y)\}$ such that $f(x_j, y) = f_1(x_j)$ for $j = 1, \ldots, k$ is infeasible on H. Suppose that there exists a vertex $a \in F_{G_1}(f_1)$. As $a \in V(G_1) \setminus V(H_1)$, we have that $a \in V(G)$ and so $a \in F_G(f)$, which contradicts the way we chose G. Therefore f_1 must also be an infeasible distance constraint on G_1. Hence there exists a retraction $\theta_1 : G_1 \longrightarrow H_1$. Similarly, there exists a retraction $\theta_2 : G_2 \longrightarrow H_2$.

Define a function $\theta : G \longrightarrow H$ as follows:

$$\theta(g) = \begin{cases} g & \text{if g } \in V(H) \\ (\theta_1(g), \theta_2(g)) & \text{otherwise} \end{cases}$$

We claim that θ is a retraction. It is obvious that θ fixes each vertex of H and that $\theta(G) \subseteq H$. We need to prove that θ is a homomorphism. Let gg' be an edge of G. If both or neither of the vertices are in H, then it is easy to see that $\theta(g)\theta(g')$ is an edge in H. Thus, without loss of generality, assume that g is in H, and that g' isn't. Then $\pi_i(g)$ and g are vertices of G_i with $\pi_i(g) \in V(H_i)$, and so $\pi_i(g)$ and $\theta_i(g')$ are adjacent in H_i, $i = 1, 2$. Therefore g and $\theta(g')$ are adjacent in H, and H is a retract of G. Thus $H \in \mathcal{AR}_H$ by Proposition 2.5.

$\square$

Let H be a graph. We will construct a supergraph of H called the *vector graph* [70] of H, denoted by $\mathcal{V}(H)$. The *distance vector* of $h \in V(H)$, denoted by $\boldsymbol{v}(h)$, is the vector indexed by the vertices of H such that

$$\boldsymbol{v}(h)_{h'} = d_H(h, h'), \quad \text{for all } h' \in H.$$

A vector $\boldsymbol{a}$ *satisfies* a vector $\boldsymbol{b}$, denoted by $\boldsymbol{a} \leq \boldsymbol{b}$, if $\boldsymbol{a}$ is less than or equal to $\boldsymbol{b}$ component-wise. The vertices of $\mathcal{V}(H)$ are $|V(H)|$-dimensional vectors $\boldsymbol{a}$ indexed by the vertices of H such that the entries of $\boldsymbol{a}$ are integers between 0 and the diameter of H and there exists a vertex $h \in V(H)$ such that $\boldsymbol{v}(h) \leq \boldsymbol{a}$. Two vertices $\boldsymbol{a}, \boldsymbol{b}$ of $\mathcal{V}(H)$ are adjacent if and only if

$$|\boldsymbol{a}_h - \boldsymbol{b}_h| \leq 1 \text{ for all } h \in V(H).$$

It is easy to see that $\mathcal{V}(H)$ has an induced copy of H on the distance vectors of the vertices of H; denote this subgraph of $\mathcal{V}(H)$ by $H_{\mathcal{V}}$. For a vertex $\boldsymbol{a}$ of $\mathcal{V}(H)$, let $f_{\boldsymbol{a}}$ denote the function from $V(H)$ to the non-negative integers defined by $f_{\boldsymbol{a}}(h) = \boldsymbol{a}_h$. Note that $f_{\boldsymbol{a}}$ is a distance constraint on H and that $F_H(f_{\boldsymbol{a}})$ is the set of all vertices in H whose distance vectors satisfy $\boldsymbol{a}$.

The following theorem is a corrected version of Theorem 3 in [43].

Theorem 2.2. *[70] Let H be a connected graph. Then $H \in \mathcal{AR}_H$ if and only if there exists a retraction from $\mathcal{V}(H)$ to $H_{\mathcal{V}}$.*

Proof. Assume that $H \in \mathcal{AR}_H$. As H and $H_{\mathcal{V}}$ are isomorphic, $H_{\mathcal{V}}$ is also in $\mathcal{AR}_H$. Let f be an infeasible distance constraint on $H_{\mathcal{V}}$ with $D_f = \{\boldsymbol{v}(x_1), \ldots, \boldsymbol{v}(x_k)\}$, $x_i \in V(H)$. Then the following distance constraint f' on H is infeasible; $D_{f'} = \{x_1, \ldots, x_k\}$ and $f'(x_i) = f(\boldsymbol{v}(x_i))$ for $i = 1, \ldots, k$. If we can prove that $F_{\mathcal{V}(H)}(f) = \emptyset$, then $H_{\mathcal{V}}$ is a retract of $\mathcal{V}(H)$ as $H_{\mathcal{V}}$ is in $\mathcal{AR}_H$ using Proposition 2.5. Suppose that there exists a vertex $\boldsymbol{a} \in F_{\mathcal{V}(H)}(f)$. By the definition of $\mathcal{V}(H)$, there exists a vertex a of H such that $\boldsymbol{v}(a) \leq \boldsymbol{a}$, i.e., $d_H(a, h) \leq \boldsymbol{a}_h$ for all $h \in V(H)$. Thus in particular,

$$d_H(a, x_i) \leq \boldsymbol{a}_{x_i} \leq f(\boldsymbol{v}(x_i)) = f'(x_i).$$

This implies that $a \in F_H(f')$, which contradicts the infeasibility of f'.

Assume that $H_{\mathcal{V}}$ is a retract of $\mathcal{V}(H)$. Let G be a connected supergraph of H such that all holes on H are holes on G. We will construct a homomorphism from G to $\mathcal{V}(H)$ and then use this homomorphism to construct a retraction from G to H.

Let $\phi : G \to \mathcal{V}(H)$ be the function defined by

$$\boldsymbol{\phi}(\boldsymbol{g})_h = \min\{\operatorname{diam}(H), d_G(g, h)\} \text{ for all } h \in V(H).$$

We must first prove that ϕ is well defined and that ϕ is a homomorphism.

Let g be a vertex of G. We will prove that $\phi(g)$ is a vertex of $\mathcal{V}(H)$ by showing that there exists a vertex h in H such that the distance vector of h satisfies $\phi(g)$. Let f_g be the distance constraint on H with domain $V(H)$ defined by $f_g(h) = d_G(g, h)$ for all $h \in V(H)$. Certainly $g \in F_G(f_g)$, i.e., f_g is a feasible distance constraint on G. By the way we chose G and by Proposition 2.5, f_g must also be feasible on H. Let h^* be a vertex in $F_H(f_g)$. Then

$$d_H(h^*, h) \leq f_g(h) = d_G(g, h) \text{ for all } h \in V(H).$$

The quantity $d_H(h^*, h)$ is certainly bounded above by the diameter of H for all $h \in V(H)$. Hence

$$d_H(h^*, h) \leq \min\{\text{diam}(H), d_G(g, h)\} \text{ for all } h \in V(H),$$

and so $\boldsymbol{v}(h^*)$ satisfies $\boldsymbol{\phi(g)}$. Therefore ϕ is well defined.

It is easy to see that ϕ is a homomorphism from G to $\mathcal{V}(H)$. If $gg' \in E(G)$, then for any vertex h of H, $|d_G(h, g) - d_G(h, g')| \leq 1$.

Let θ be a retraction from $\mathcal{V}(H)$ to $H_{\mathcal{V}}$ and let ϕ' be the isomorphism from $H_{\mathcal{V}}$ to H defined by $\phi'(\boldsymbol{v}(h)) = h$ for all vertices $\boldsymbol{v}(h)$ of $H_{\mathcal{V}}$. Then $\phi' \circ \theta \circ \phi$ is a retraction from G to H; clearly this function is a homomorphism and for all vertices h of H, $\phi' \circ \theta \circ \phi(h) = \phi' \circ \theta(h) = \phi'(\boldsymbol{v}(h)) = h$.

□

2.2 Holes on chordal graphs

In this section, we will be using the concept of convexity in chordal graphs to analyze holes on connected chordal graphs. In particular we will prove a lower bound on the distance between vertices in the domain of a hole on a connected chordal graph and we will prove that non-degenerate holes on connected chordal graphs imply the existence of a specific isometric subgraph.

The convexity we present here is a specific convex geometry (see [13, 23]). A *convex structure* [13] (or *alignment of* V [24]) is a pair $(V, \mathcal{C})$, where V is a set and $\mathcal{C}$ is a collection of subsets of V, called the *convex sets*, such that $\emptyset, V \in \mathcal{C}$ and $\mathcal{C}$ is closed under taking intersections. The *convex hull* of a set $S \subseteq V$ is the smallest convex set containing S. If $S \subseteq V$ is convex, an element $x \in S$ is an *extreme point* of S if $S \setminus \{x\}$ is also convex. Then a convex structure $(V, \mathcal{C})$ is a *convex geometry* if $(V, \mathcal{C})$ has the following additional property: Every convex set is the convex hull of its extreme points. This is referred to as the Minkowski-Krein-Milman property.

There have been two main types of convexity studied for graphs: geodesic convexity and monophonic convexity. We will be studying monophonic convexity exclusively and hence will refer to monophonically convex sets as convex, etc. The chapter on convexity in [13] contains many references and results concerning convexity in general, not just monophonic convexity. Let H be a graph. We say that a set $X \subseteq V(H)$ is *monophonically convex* if all chordless paths between vertices of X are contained in X. The authors of [24] proved that if H is a chordal graph and if $\mathcal{C}$ is the set that consists of the convex sets of H, then $(V(H), \mathcal{C})$ is a convex geometry. In Theorem 2.3 we present the proofs of properties of $(V(H), \mathcal{C})$ that we use in analyzing holes on connected chordal graphs.

A *partial perfect elimination ordering* of graph H is an ordering $h_1, \ldots, h_k$ of some, possibly all, of the vertices of H such that h_i is simplicial in H_{i-1}, for $i = 1, \ldots, k$. Recall that $H_i = H \setminus \{h_1, \ldots, h_i\}$ for $i = 1, \ldots, k$, where $i \neq |V(H)|$, and $H_0 = H$.

Theorem 2.3. *[24] Let H be a chordal graph. Then the following statements are true.*

i.) If $X, Y \subseteq V(H)$ are convex in H, then $X \cap Y$ is also convex in H

ii.) Let X be a connected subset of $V(H)$ and let $j \geq 1$ be an integer, then $D(X, j)$ is convex in H.

iii.) Let X be subset of $V(H)$. Then X is convex in H if and only if there exists partial perfect elimination ordering $h_1, \ldots, h_k$ of H such that $X = V(H) \setminus \{h_1, \ldots, h_k\}$.

Proof. i. Let $X, Y \subseteq V(H)$ be convex sets in H. If $|X \cap Y| \leq 1$, then we are done as sets of size at most one are obviously convex. Therefore we can assume there are distinct vertices h and h' in $X \cap Y$. Let P be any chordless path in H from h to h'. As X is convex in H, the vertices of P must be contained in X. Similarly, the vertices of P must be contained in Y. Therefore $X \cap Y$ is convex in H.

ii. Let $X \subseteq V(H)$ be connected and let $j \geq 1$ be an integer. Suppose that $D(X, j)$ is not convex in H. Thus there exist distinct vertices h, h' in $D(X, j)$ that have a chordless path P from h to h' such that P is internally disjoint from $D(X, j)$. Let H' be the subgraph of H induced by $D(X, j) \cup V(P)$. Clearly H' is chordal and it is also connected as $D(X, j)$ is connected and P is a path with endpoints in $D(X, j)$. By construction, $\{h, h'\}$ is a minimal cut set of H'. Thus by Theorem 1.8, $\{h, h'\}$ must be a clique, contradicting the way we chose h and h'. Therefore $D(X, j)$ must be convex.

iii. Let X be a subset of $V(H)$ that is convex in H. Let $h_1, \ldots, h_k$ be a partial perfect elimination ordering of H (that is possibly empty) such that any simplicial vertex of H_k is in X. Note that H_k is chordal as H_k is an induced subgraph of H and for the same reason, X is a convex set in H_k. Suppose there exists a vertex $h \in V(H_k) \setminus X$. Thus h is not simplicial in H_k by assumption. By Theorem 1.9 applied to H_k, the vertex h must lie on a chordless path P in H_k between two simplicial vertices x and x'. By assumption, $x, x' \in X$. But, by the definition of convexity, all vertices of P must be in X. This is a contradiction as we assumed that $h \notin X$. Therefore $V(H_k) = X$.

Let X be a subset of $V(H)$ such that there exists a partial perfect elimination ordering $h_1, \ldots, h_k$ of H where $X = V(H) \setminus \{h_1, \ldots, h_k\}$. Suppose that X is not convex. Then there exist vertices x, x' in X with a chordless path P from x to x' that is internally disjoint from X. Let h_i be the first vertex of P that occurs in the partial perfect elimination ordering $h_1, \ldots, h_k$. Then $P \subseteq H_{i-1}$. As h_i is simplicial in H_{i-1}, the neighbours of h_i on P must be adjacent. This is a contradiction as we assumed that P was chordless. Hence X is convex in H.

□

Note that statement iii of Theorem 2.3 also implies that all chordal graphs have perfect elimination orderings as $\emptyset$ is always a convex set.

Corollary 2.1. *Let H be a chordal graph. Then for any set of discs $D(x_i, j_i)$, $i = 1, \ldots, k$ in H, the set $\cap_{i=1}^{k} D(x_i, j_i)$ is convex. In particular, $F(f)$ is convex for all distance constraints f on H.*

Proof. Let $D(x_i, j_i)$, $i = 1, \ldots, k$ be a set of discs in H. By Theorem 2.3, $D(x_i, j_i)$ is convex if $j_i \geq 1$. If $j_i = 0$, then $D(x_i, 0) = \{x_i\}$ is trivially convex. Thus $\cap_{i=1}^{k} D(x_i, j_i)$ is convex, again by Theorem 2.3.

Let f be a hole on H. Then $F(f) = \cap_{x \in D_f} D(x, f(x))$ must be convex by our arguments in the preceding paragraph.

□

At this point, we have presented the results concerning convexity the we need. We will now start using these convexity results to analyze holes on connected chordal graphs.

Lemma 2.2. *Let H be a connected chordal graph, and let f be a non-degenerate hole on H with domain $D_f = \{x_1, \ldots, x_k\}$. Suppose that $A = \{a_1, \ldots, a_p\}$, $2 \leq p < k$ is a clique in H such that each a_i is an x_i-relaxed filler, $1 \leq i \leq p$. If a is an x_j-relaxed filler, where $p < j \leq k$, then a is equidistant to all vertices of A.*

Proof. Without loss of generality, assume that a is closer to a_1 than a_2. As A is a clique, a_1 and a_2 are adjacent and so a_1 is on a shortest path between a_2 and a. Thus by definition of convexity a_1 must belong to any convex set that contains both a and a_2. Note that both a_2 and a are within $f(x_1)$ of x_1. Hence, by Corollary 2.1 the vertex a_1 must also be within $f(x_1)$ of x_1. By assumption, a_1 is an x_1-relaxed filler. Hence, we in fact have that $a_1 \in \cap_{i=1}^{k} D(x_i, f(x_i))$. Thus a_1 is a complete filler of f, contradicting the definition of a hole. Therefore a is equidistant to all vertices of A.

□

Let P be a walk in a graph H. Recall that if a and b are vertices that each appear once on P, we denote by $P[a, b]$ the section of the walk from a to b.

The next theorem tells us that the existence of a non-degenerate hole on a connected chordal graph implies the existence of a clique of relaxed fillers. We will use this later to bound the size of holes, and the distance between elements in the domain of holes, on connected chordal graphs. In addition, we will use Theorem 2.4 to prove that the existence of a non-degenerate hole on a connected chordal graph implies the existence of a particular isometric subgraph.

Theorem 2.4. *Let f be a non-degenerate hole on a connected chordal graph H with domain $D_f = \{x_1, \ldots, x_k\}$. Then there exists a clique $A = \{a_1, \ldots, a_k\}$ such that each a_i is an x_i-relaxed filler of f, $i = 1, \ldots, k$, and*

$$d(x_i, a_j) = \begin{cases} f(x_i) & \text{if } j \neq i \\ f(x_i) + 1 & \text{if } j = i \end{cases}$$

Proof. The proof of this theorem will be given in two parts. First we will prove that we can choose an x_i-relaxed filler of f for $i = 1, 2, 3$, such that these three relaxed fillers are mutually adjacent. Second we will prove that if $\{a_1, \ldots, a_p\}$ is a maximal clique such that a_i is an x_i-relaxed filler of f for $i = 1, \ldots, p$, then $p = k$.

Choose an x_i-relaxed filler a_i of f for $i = 1, 2, 3$, such that $\sum_{1 \leq i < j \leq 3} d(a_i, a_j)$ is minimized. Note that a_1, a_2 and a_3 must be distinct, otherwise f is not a hole. If the vertices a_1, a_2 and a_3 that we have chosen are mutually adjacent, then we have accomplished the first part the the proof. Thus without loss of generality, assume that a_1 and a_2 are not adjacent.

Let $P_{i,j}$ be a shortest path from a_i to a_j, $1 \leq i < j \leq 3$ and let $P_{i,j}^{-1}$ denote the path $P_{i,j}$ in reverse order. Let u be the first vertex on $P_{1,3}$ that has a nontrivial neighbour on $P_{2,3}$. It is possible that $u = a_3$. Let v be the nontrivial neighbour of u on $P_{2,3}$ that is closest to a_2. Note that if $u = a_3$, then v is the nontrivial neighbour of a_3 on $P_{2,3}$. Let P' be the path $P_{1,3}[a_1, u]P_{2,3}^{-1}[v, a_2]$ from a_1 to a_2. Note that P' is chordless by the way we chose u and v. We have depicted the paths described above in Figure 2.3.

As the three paths P', $P_{1,3}$ and $P_{2,3}$ are chordless, we can use convexity to garner some information.

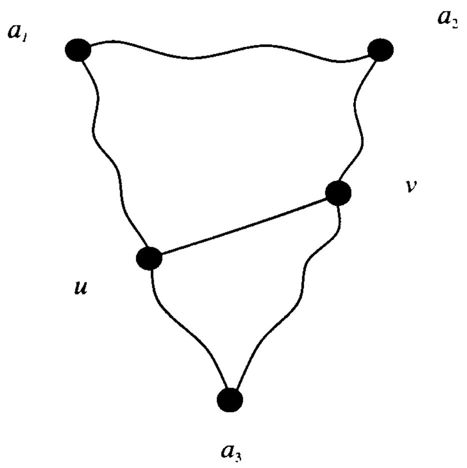


Figure 2.3: The paths $P_{1,2}$, $P_{1,3}$, $P_{2,3}$ and P'.

First consider the path P' from a_1 to a_2. By the definition of convexity, P' must be contained in any convex set that contains both a_1 and a_2. As a_i is an x_i-relaxed filler of f for x_i, $i = 1, 2$, both a_1 and a_2 are close enough to all vertices of D_f, except for possibly x_1 and x_2. This means that a_1 and a_2 are in $\cap_{x \neq x_1, x_2} D(x, f(x))$, which is a convex set by Corollary 2.1. Thus u and v are also in $\cap_{x \neq x_1, x_2} D(x, f(x))$ as $u, v \in V(P')$.

Now consider the path $P_{1,3}$ which contains u. As above, the path $P_{1,3}$ must be contained in any convex set that contains both a_1 and a_3. Then, by similar arguments, the path $P_{1,3}$, and hence u, is in the set $\cap_{x \neq x_1, x_3} D(x, f(x))$. We already know that u is also in the set $\cap_{x \neq x_1, x_2} D(x, f(x))$. Therefore u is in the set $\cap_{x \neq x_1} D(x, f(x))$ and so u is an x_1-relaxed filler of f.

Next consider the path $P_{2,3}$ which contains v. By repeating the same argument as above, we see that v is an x_2-relaxed filler of f. Thus we can replace a_1 by u and a_2 by v. However, this contradicts the way we chose a_1, a_2, and a_3 since, $d(u, v) + d(v, a_3) + d(u, a_3) < \sum_{1 \leq i < j \leq 3} d(a_i, a_j)$. Hence we may assume that there exists an x_i-relaxed filler a_i of f for $i = 1, 2, 3$ such that $\{a_1, a_2, a_3\}$ is a clique.

Let p be the largest integer such that there exists a clique $A_p = \{a_1, \ldots, a_p\}$, where a_i is an x_i-relaxed filler of f for $i = 1, \ldots, p$. By the above work, we know that $p \geq 3$. If $p = k$, then we are done. Thus suppose that $p < k$ and let a_{p+1} be an x_{p+1}-relaxed filler of f. By Lemma 2.2, a_{p+1} is equidistant to all vertices of A_p and so by the maximality of p, a_{p+1} has no neighbours in A_p. Let P be a shortest path from a_{p+1} to a_1 and let P' be a shortest path from a_2 to a_{p+1}. Clearly P and P' are of the same length. Thus the closed walk PP' contains a cycle C with the edge a_1a_2. By Lemma 1.6, a_1 and a_2 have a common nontrivial neighbour w on C. As P and P' are both shortest paths of the same length, we may assume without loss of generality that w is the neighbour of a_2 on P'; if w were a vertex on P' other than the neighbour of a_2, then $a_1P'[w, a_{p+1}]$ would be a path from a_1 to a_{p+1} that is shorter than P. By arguments similar to those in the first part of the proof, the path $a_1P'[w, a_{p+1}]$ is contained in $\cap_{i\neq 1,p+1} D(x_i, f(x_i))$ and P' is contained in $\cap_{i\neq 2,p+1} D(x_i, f(x_i))$. Hence $P'[w, a_{p+1}]$ is contained in the set of x_{p+1}-relaxed fillers of f. In particular, w is an x_{p+1}-relaxed filler of f with neighbours in A_p, and so by Lemma 2.2, w is an x_{p+1}-relaxed filler of f that is adjacent to all of A_p. This contradicts the maximality of p. Hence there is a clique $A = \{a_1, \dots, a_k\}$ with a_i an x_i-relaxed filler of f for $i = 1, \dots, k$.

By definition, $d(x_i, a_j) \leq f(x_i)$ for all $j \neq i$. If the inequality is strict for some j, then as A is a clique, we would have that $d(x_i, a_i) \leq d(x_i, a_j) + d(a_j, a_i) < f(x_i) + 1 \leq f(x_i)$, implying that a_i is a complete filler of f. Therefore we must have $d(x_i, a_j) = f(x_i)$ for all $j \neq i$ and $d(x_i, a_i) = f(x_i) + 1$ for all $i = 1, \dots, k$.

□

Corollary 2.2. *Let H be a connected chordal graph and let f be a non-degenerate hole H. Then $|D_f| \leq \lfloor |V(H)|/2 \rfloor$,i.e., the size of f is bounded above by $\lfloor |V(H)|/2 \rfloor$.*

Proof. By Theorem 2.4, there is a clique A in H that consists of exactly one x-relaxed filler of f for each $x \in D_f$. Each of the relaxed fillers of f in A must be distinct as a hole has no complete fillers. Thus $|A| = |D_f|$. We claim that A is disjoint from D_f.

Suppose there exists a vertex $x \in A \cap D_f$, and let a be the x-relaxed filler of f in A. Then as A is a clique, $d(x, a) = 1$, and so we must have $f(x) = 0$. This is a contradiction by Proposition 2.3.

Therefore $2|D_f| = |D_f \cup A| \leq |V(H)|$. As $|D_f|$ is an integer, $|D_f| \leq \lfloor |V(H)|/2 \rfloor$.
□

Let H be a connected graph and let f be a hole on H with $D_f = \{x_1, \ldots, x_k\}$. If f is a degenerate hole, i.e., if $k = 2$, then $d(x_1, x_2) = f(x_1)+f(x_2)+1$ by Proposition 2.2. There is a tightness here in that f is not feasible, and the vertices of the domain of f are just far enough away to cause f to not be feasible. If f is a non-degenerate hole, i.e., $k \geq 3$, we know that $d(x_k, x_j) \leq f(x_i) + f(x_j)$, again by Proposition 2.2. In Figure 2.4, the graph H has a hole f with domain $D_f = \{x_1, x_2, x_3\}$ where $f(x_i) = 1$ for $i = 1, 2, 3$. We see that $d(x_2, x_3) = 1 < f(x_2) + f(x_3) = 2$. Thus the tightness of the degenerate hole is missing. However, we shall show that, if we restrict ourselves to connected chordal graphs and the variety they generate, tightness is retained.

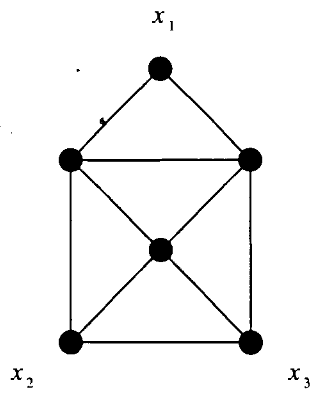


Figure 2.4: A graph with a squished hole.

Let H be a connected graph with hole f. A pair $x_1, x_2 \in D_f$ is *squished* if $d(x_1, x_2) < f(x_1) + f(x_2)$. A hole is called *squished* if it has a squished pair. Thus, the graph in Figure 2.4 has a squished hole. Consider a degenerate hole f' on a connected graph H with $D_{f'} = \{y_1, y_2\}$. Then $d(y_1, y_2) = f'(y_1) + f'(y_2) + 1$ by Proposition 2.2. Hence degenerate holes can't be squished. A connected graph that has no squished holes is called *stretched.* Recall that $\mathcal{AR}_I$, the class of absolute retracts with respect to isometry, are exactly those graphs that do not have non-degenerate holes by Theorem 1.2. Thus all graphs in $\mathcal{AR}_I$ are stretched.

Theorem 2.5. *Each connected chordal graph is stretched.*

Proof. Suppose that there exists a connected chordal graph H that has a squished hole f. Thus, there exist vertices $x_1, x_2 \in D_f$ such that $d(x_1, x_2) < f(x_1) + f(x_2)$. Then f must be a non-degenerate hole by Proposition 2.2, and so by Theorem 2.4, there exist x_i-relaxed fillers of f, vertices a_i, $i = 1, 2$, such that a_1 and a_2 are adjacent and $d(x_i, a_i) = f(x_i)+1$ for $i = 1, 2$ and $d(x_i, a_j) = f(x_i)$ for $i \neq j$, $i, j = 1, 2$.

Let P be a shortest path from x_1 to x_2. Let w be the vertex on P that is distance $f(x_1)$ from x_1 if the length of P is more than $f(x_1)$, and let w be x_2 otherwise. The length of $P[w, x_2]$ is at most $f(x_2) - 1$ as the length of P is at most $f(x_1) + f(x_2) - 1$. Let P' be a shortest path from x_2 to a_1. We know that P' has length $f(x_2)$ and so all the vertices of $P' \setminus a_1$ are within distance $f(x_2) - 1$ of x_2. Thus $P[w, x_2]P'a_2$ is a walk from w to a_2 such that all the vertices except a_2 and a_1 are within $f(x_2) - 1$ of x_2. This walk contains a chordless path P'' from w to a_2. The penultimate vertex of P'' must be a_1, else a_2 would have a neighbour within $f(x_2) - 1$ of x_2, implying $d(a_2, x_2) \leq f(x_2)$; this would contradict the infeasibility of f. Recall that $d(x_1, a_2) = f(x_1)$ and that we chose w such that $d(x_1, w) \leq f(x_1)$. Thus P'' is a chordless path between elements of $D(x_1, f(x_1))$ that is not contained in $D(x_1, f(x_1))$ as $d(x_1, a_1) = f(x_1) + 1$. This is a contradiction as $D(x_1, f(x_1))$ is convex by Theorem 2.3.

Therefore connected chordal graphs can not have squished holes and so connected chordal graphs are stretched.

□

Let f be a non-degenerate hole of connected graph H, and let $D_f = \{x_1, \ldots, x_k\}$. A *base* for f is an isometric subgraph J of H consisting of clique $\{a_1, \ldots, a_k\}$, vertices $\{z_1, \ldots, z_k\}$ such that z_i is adjacent to all of $\{a_1, \ldots, a_k\}$ except for a_i, and disjoint paths $P_1, \ldots, P_k$ where P_i is a path of length $f(x_i) - 1$ from z_i to a vertex x_i. There are no other edges in J, see Figure 2.5.

In [15], the authors proved that if a connected chordal graph H is not in $\mathcal{AR}_I$, then there exists a non-degenerate hole on H whose range is $\{1\}$ which has a base. In fact, any non-degenerate hole f on a connected chordal graph H has a base by the

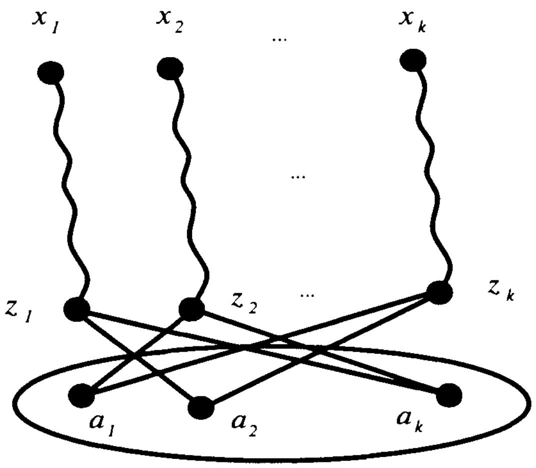


Figure 2.5: A generic hole base of a non-degenerate hole.

theorem below.

Theorem 2.6. *Let H be a connected chordal graph and let f be a non-degenerate hole on H. Then f has a base in H.*

Proof. Let f be non-degenerate hole on a connected chordal graph H. By Theorem 2.4, there exists a clique A in H that consists of one x-relaxed filler of f for each $x \in D_f$. Without loss of generality choose a vertex $x \in D_f$. We will construct a perfect elimination ordering with a particular distance property and then use this perfect elimination ordering to find an induced path of length $f(x) - 1$ in H from x to some vertex z that is adjacent to all of A but the x-relaxed filler. Note that $f(x) - 1 \geq 0$ by Proposition 2.3.

The disc $D_H(x, f(x) - 1)$ is convex in H by Theorem 2.3. Also by Theorem 2.3, there exists a partial perfect elimination ordering $y_1, \ldots, y_l$ of H such that $V(H_l) = D(x, f(x) - 1)$. The subgraph H_l of H is also chordal and $\{x\}$ is trivially convex in H_l. Hence there exists a partial perfect elimination ordering $y_{l+1}, \ldots, y_m$ of H_l such that $V(H_l) \setminus \{y_{l+1}, \ldots, y_m\} = \{x\}$.

Clearly $y_1, \ldots, y_l, y_{l+1}, \ldots, y_m, x$ is a perfect elimination ordering of H such that $D_H(x, f(x) - 1) = \{y_{l+1}, \ldots, y_m, x\}$. For technical reasons, let $y_{m+1} = x$.

Let a_x be the x-relaxed filler of f in A. By Theorem 2.4, $d_H(x,a) = f(x)$ for all $a \in A \setminus a_x$ and $d_H(x, a_x) = f(x) + 1$. Thus $A \subseteq \{y_1, \ldots, y_l\}$. Let a be the vertex of $A \backslash \{a_x\}$ in $y_1, \ldots, y_l$ of lowest index, i.e., $a = y_k$ and $A \backslash \{a_x\} \subseteq \{y_k, \ldots, y_l\}$. The vertex a has a neighbour z that is $f(x) - 1$ from x. Obviously z must be in $\{y_{l+1}, \ldots, y_{m+1}\}$ and z can't be adjacent to a_x. As the neighbours of a that appear after it in the perfect elimination ordering $y_1, \ldots, y_{m+1}$ form a clique, $(A \setminus \{a_x\}) \cup \{z\}$ is a clique. Let P be a path from z to x of length $f(x) - 1$. Thus we have produced a vertex z that is adjacent to each vertex of $A \setminus \{a_x\}$ such that $d(z, x) = f(x) - 1$. Observe that z and a_x are not adjacent since $d_H(z, x) = f(x) - 1$ and $d_H(a_x, x) = f(x) + 1$.

Let $D_f = \{x_1, \ldots, x_k\}$. By the above arguments, for each i, $1 \le i \le k$, there exists in H a chordless path P_i of length $f(x_i) - 1$ from some vertex z_i to x_i where z_i is adjacent to all of A but the x_i-relaxed filler of f in A, call it a_i.

Consider the walk Q in H from x_1 to x_2 we create by traveling backwards down P_1 from x_1 to z_1, then to a_3, then to z_2 and then traveling from z_2 to x_2 on P_2, i.e., $P_1^{-1} a_3 P_2$. This walk has length $f(x_1) + f(x_2)$ as the path P_i has length $f(x_i) - 1$ for $i = 1, 2$. Since f is not squished by Theorem 2.5, $d_H(x_1, x_2) = f(x_1) + f(x_2)$. Thus Q is in fact a shortest path and so Q is isometric in H. Therefore if J is the subgraph of H induced by $A \cup V(P_1) \cup \ldots \cup V(P_k)$, then J is an isometric subgraph of H.

Hence f has a base in H.

□

2.3 Stretched graphs and the variety generated by chordal graphs

We will use stretched graphs to study the relationship between $\mathcal{AR}_H$ and the variety generated by connected chordal graphs.

From Chapter 1, we know that the variety generated by connected chordal graphs strictly contains $\mathcal{AR}_I$, i.e., the variety generated by connected chordal graphs contains $\mathcal{AR}_I$ and there exists a graph that is in the variety generated by connected chordal graphs which is not in $\mathcal{AR}_I$. It turns out that there exist connected chordal graphs that are not in $\mathcal{AR}_H$, as seen in Figure 2.6; let H be the graph induced by the round vertices, and the let G be the entire graph. Even though each hole on H is also a hole on G, H is not a retract of G. However, there also exist graphs in $\mathcal{AR}_H$ that

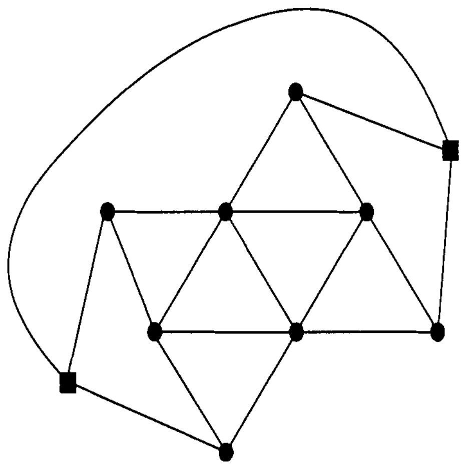


Figure 2.6: The subgraph induced by the round vertices is a chordal graph that is not in $\mathcal{AR}_H$.

are not in the variety generated by connected chordal graphs, as seen in Figure 2.4. The graph in this figure is in $\mathcal{AR}_H$ [43], but is not in the variety generated by chordal graphs; the graph is not stretched and the variety generated by connected chordal graphs is contained in the class of stretched graphs, by Corollary 2.3. Using stretched graphs, we can describe exactly the graphs that are both in $\mathcal{AR}_H$ and in the variety generated by connected chordal graphs.

Proposition 2.6. *The class of stretched graphs is a variety.*

Proof. To prove that the class stretched graphs is a variety, all we need to consider are two cases; when a graph H is the retract of a stretched graph and when a graph H is the product of stretched graphs.

If H is a retract of a stretched graph H', then clearly H can have no squished holes as H is an isometric subgraph of H'.

Now assume that H is the product of two stretched graphs, say $H = H_1 \times H_2$. Suppose that H has a squished hole f. Let $\pi_i : H \rightarrow H_i$ be the projection onto the i^{th} co-ordinate. The function f_i on $\pi_i(D_f)$ defined by $f_i(s) = \min_{\pi_i(\boldsymbol{z})=s} f(\boldsymbol{z})$ is certainly a distance constraint on H_i. Suppose that f_i is not feasible. Then there exists a hole f_i' on H_i such that $f_i' \leq f_i$. Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be a squished pair of f in H. As $\pi_i(\boldsymbol{x})$ and $\pi_i(\boldsymbol{y})$ can't be a squished pair of f_i' in H_i, we must have $\pi_i(\boldsymbol{x}) = \pi_i(\boldsymbol{y})$ or at least one of $\pi_i(\boldsymbol{x})$ and $\pi_i(\boldsymbol{y})$ is not in $D_{f_i'}$. Without loss of generality, we may assume that $D_{f_i'} \subseteq \pi_i(D_f \setminus \boldsymbol{y})$. Consider the set of $\boldsymbol{y}$-relaxed fillers of f in H. By assumption, there does not exist a vertex w_i in H_i such that

$$d_{H_i}(w_i, \pi_i(\boldsymbol{z})) \leq f_i'(\pi_i(\boldsymbol{z})) \quad \forall\, \boldsymbol{z} \in D_f \setminus \{\boldsymbol{y}\}\,.$$

Thus there can't be a vertex $\boldsymbol{w}$ in H such that

$$d_H(\boldsymbol{w}, \boldsymbol{z}) \leq f(\boldsymbol{z}) \quad \forall\, \boldsymbol{z} \in D_f \setminus \{\boldsymbol{y}\}\,.$$

Hence f has no $\boldsymbol{y}$-relaxed fillers. This contradicts the minimal infeasibility of f. Thus f_i has a complete filler w_i in H_i for $i = 1, 2$ and the vertex $\boldsymbol{w} = (w_1, w_2)$ will be a complete filler of f in H, contradicting the infeasibility of f in H. Thus H cannot have squished hole and so H is stretched.

Therefore the class of stretched graphs is a variety.

□

Corollary 2.3. *The variety generated by connected chordal graphs is a subset of the variety of stretched graphs.*

Proof. Any variety that contains connected chordal graphs must also contain the variety generated by connected chordal graphs. Since the class of stretched graphs is a variety by Theorem 2.5, the lemma is true.

□

The variety of stretched graphs is larger than the variety generated by connected chordal graphs, as seen by the graph in Figure 2.7; this graph is not in the variety generated by chordal graphs as it has a squished tree obstruction, a structure that

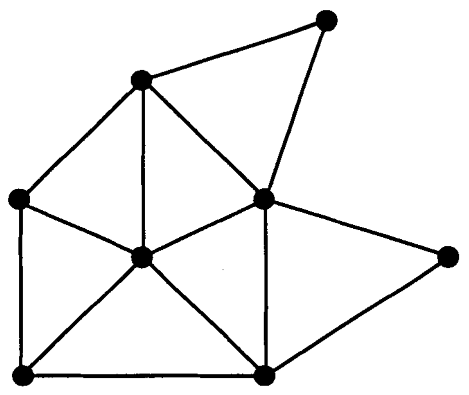


Figure 2.7: A stretched graph that is not in the variety generated by connected chordal graphs.

will be discussed in Chapter 3. Note that the graph in Figure 2.7 was obtained by adding an edge to the graph induced by the round vertices in Figure 2.6.

We will show that within $\mathcal{AR}_H$, the variety of stretched graphs and the variety generated by connected chordal graphs are in the same. Before proving this, we present a technical lemma concerning holes, products of graphs and projections.

Let H and J be connected graphs, and let γ be a homomorphism where $H \xrightarrow{\gamma} J$. Let f be a hole on H. We say that γ is a *preserving map* [45] of f if

$$\bigcap_{x \in D_f} D_J(\gamma(x), f(x)) = \emptyset.$$

Equivalently, we say that γ is a preserving map of f if $F_J(f^*) = \emptyset$, where $f^*(x') = \min_{\gamma(x)=x'} f(x)$. The terms hole separating [43] and gap preserving [57] have also been used.

Lemma 2.3. *[43] Let H be a fixed connected graph. If for each hole f of H there exists a connected graph J_f and a function $\gamma_f : H \to J_f$ such that γ_f is a preserving map of f, then H is isomorphic to a subgraph $\hat{H}$ of G, where*

$$G = \Pi \{J_f : f \text{ is a hole of } H\},$$

such that each hole on $\hat{H}$ is a hole on G.

Proof. The vertices of G are vectors indexed by the holes of H. Define a map $\phi : H \longrightarrow G$ by $\phi(h) = (\gamma_f(h))_f$ for each hole f of H and let $\hat{H} = \phi(H)$. Since γ_f is a homomorphism for each hole f on H, the function ϕ must also be a homomorphism. Let h and h' be distinct vertices of H, and let f' be the degenerate hole on H with $D_{f'} = \{h, h'\}$ such that $f'(h) = 0$ and $f'(h') = d_H(h, h') - 1$. Since there exists a graph $J_{f'}$ and a function $\gamma_{f'} : H \longrightarrow J_{f'}$ that is a preserving map of f', we have

$$d_{J_{f'}}(\gamma_{f'}(h), \gamma_{f'}(h')) \geq f'(h) + f'(h') + 1 = d_H(h, h'),$$

which implies that $d_G(\phi(h), \phi(h')) \geq d_H(h, h')$. Hence ϕ preserves distances, and so $\hat{H}$ is isomorphic to H.

Now we will prove that each hole on $\hat{H}$ is a hole on G. Suppose that there exists a hole $\hat{f}$ on $\hat{H}$ such that $F_G\left(\hat{f}\right) \neq \emptyset$. Let f be the corresponding hole on H. Therefore there exists a graph J_f and a function $\gamma_f : H \longrightarrow J_f$ that is a preserving map of f. Let $\pi_f : G \longrightarrow J_f$ be the projection onto J_f. For any vertex $h \in V(H)$, note that $\gamma_f(h) = \pi_f(\phi(h))$. In particular, this is true for any vertex $h \in D_f$. Hence the domain of $\hat{f}$ is exactly $\phi(D_f)$. If g is a vertex in $F_G\left(\hat{f}\right)$, then

$$d_G(g, \hat{h}) \leq \hat{f}(\hat{h}) \text{ for all } \hat{h} \in D_{\hat{f}}.$$

Therefore $d_{J_f}(\pi_f(g), \pi_f(\hat{h})) \leq \hat{f}(\hat{h})$ for all $\hat{h} \in D_{\hat{f}}$. By definition of $\hat{H}$ and since $D_{\hat{f}} = \phi(D_f)$,

$$d_{J_f}(\pi_f(g), \gamma_f(h)) \leq f(h) \text{ for all } h \in D_f,$$

which is a contradiction to the way we chose J_f. Therefore $\hat{f}$ is certainly an infeasible distance constraint on G. As in the proof on Proposition 2.5, it is easy to see that $\hat{f}$ must in fact be a hole on G.

□

Thus if H is in $\mathcal{AR}_H$ and G is a graph as constructed in Lemma 2.3, then there exists a retract of G which is isomorphic to H.

Theorem 2.7. *In $\mathcal{AR}_H$, the variety generated by connected chordal graphs and the variety of stretched graphs are the same. In other words, if we let X be the variety* *generated by connected chordal graphs and let Y be the variety of stretched graphs,*

$$\mathcal{AR}_H \cap X = \mathcal{AR}_H \cap Y.$$

Proof. If a graph H is in the variety generated by connected chordal graphs, then by Corollary 2.3, H is stretched. Thus this containment obviously holds when we restrict ourselves to $\mathcal{AR}_H$.

Now consider a stretched graph H in $\mathcal{AR}_H$. By Lemma 2.3, if for each hole f of H we can find a preserving map of f from H to a chordal graph J_f, then H is isomorphic to a retract of ΠJ_f. In particular, H would be in the variety generated by chordal graphs. Thus our task now is to find a chordal graph J for each hole f of H such that there is a preserving map of f from H to J.

Let f be a degenerate hole of H with $D_f = \{x, y\}$. Let J be a path $v_0, v_1, \ldots, v_p$ with $p = f(x) + f(y) + 1$. Then it is easy to see that a function $\gamma : H \to J$ defined by

$$\gamma(h) = \begin{cases} v_l & \text{if } d_H(h,x) = l \le p \\ v_p & \text{otherwise} \end{cases}$$

is a preserving map of f from H to J.

If H only has degenerate holes, there is nothing more to be done. Thus we may assume that H has a non-degenerate hole f with $D_f = \{y_1, \ldots, y_k\}$. Since H is stretched, it has no squished holes and so

$$d_H(y_i, y_j) = f(y_i) + f(y_j) \text{ for all } 1 \le i < j \le k. \tag{2.1}$$

Let J be a graph that consists of independent paths $P_1, \ldots, P_k$ and a clique $\{a_1, \ldots, a_k\}$, with $\{a_1, \ldots, a_k\} \cup V(P_i) \cup \ldots \cup V(P_k)$ a partition of $V(J)$. Each path P_i is a path of length $f(y_i) - 1$ from x_i to z_i where z_i is adjacent to all of $\{a_1, \ldots, a_k\}$ but a_i, for $i = 1, \ldots, k$. There are no other edges in J than those mentioned; see Figure 2.5. Note that $d_J(x_i, x_j) = f(y_i) + f(y_i)$ for $i \ne j$. Clearly, J is chordal and the distance constraint f' on J defined by $f'(x_i) = f(y_i)$ for all $i = 1, \ldots, k$ is a hole. In fact J itself is a base for f'.

Define a map $\gamma : H \to J$ by

$$\gamma(h) = \begin{cases} \text{the } j^{\text{th}} \text{ vertex of } P_i & \text{if } d_H(h, y_i) = j < f(y_i) \\ a_j, \text{ where } j \text{ is the smallest integer such that } d_H(h, y_j) > f(y_j) & \text{if } d_H(h, y_i) \geq f(y_i) \text{ for all } i = 1, \ldots, k \end{cases}$$

We claim that γ is a preserving map of f.

First note that γ is well defined. Let h be a vertex of H. If $d_H(h, y_i) < f(y_i)$ for some i, then $d_H(h, y_p) > f(y_p)$ for all $p \neq i$ by equation 2.1. Further, if $d_H(h, y_i) \geq f(y_i)$ for all $i = 1, \ldots, k$, there must some index p such that $d_H(h, y_p) > f(y_p)$ as f is not feasible.

Now we will show that γ is a homomorphism.

Let hh' be an edge of H with $h \neq h'$. Without loss of generality assume that $d_H(h, y_i) = j < f(y_i)$ for some $i = 1, \ldots, k$. Then $j - 1 \leq d_H(h', y_i) \leq j + 1 \leq f(y_i)$. If $d_H(h', y_i) < f(y_i)$, then clearly $\gamma(h)$ and $\gamma(h')$ are adjacent vertices of the path P_i. If $d_H(h', y_i) = d_H(h, y_i) + 1 = f(y_i)$, then $\gamma(h)$ is z_i and $\gamma(h')$ is some a_j, with $j \neq i$. Again $\gamma(h)$ and $\gamma(h')$ are adjacent vertices in J. Lastly, assume that $d_H(h, y_i), d_H(h', y_i) \geq f(y_i)$ for all $i = 1, \ldots, k$. Then $\gamma(h')$ and $\gamma(h)$ are vertices in $\{a_1, \ldots, a_k\}$. As $\{a_1, \ldots, a_k\}$ is a clique in J, $\gamma(h)$ and $\gamma(h')$ are adjacent. Therefore γ is a homomorphism from H to J.

By construction, $f'(x_i) = \gamma(f(y_i))$ for $i = 1, \ldots, k$ and $F_J(f') = \emptyset$. Hence γ is a preserving map of f from H to J.

□

Let H be a connected chordal graph and let f be a non-degenerate hole on H. In the proof of Theorem 2.7, we essentially showed that f has a base J, and J is a retract of H.

2.4 Near unanimity functions and dismantlability

In Chapter 1, we presented a theorem of equivalences for $\mathcal{AR}_I$, Theorem 1.2. In particular, Theorem 1.2 stated that $\mathcal{AR}_I$ is exactly the set of connected graphs that admit majority functions and that all graphs in $\mathcal{AR}_I$ are dismantlable. Now we will show the relationship between $\mathcal{AR}_H$ and the graphs that admit near unanimity functions and the graphs that are dismantlable.

Let H be a graph and let $k \geq 3$ be an integer. Let $\mathcal{M}_k(H)$ be the graph we obtain from H^k by identifying all $\boldsymbol{w}$ with $k-1$ components equal to x with the vertex $\overline{x} = (x, x, \ldots, x)$. More specifically, $\mathcal{M}_k(H)$ is the graph whose vertex set is

$$V(H^k) \setminus \{\boldsymbol{u} \mid \boldsymbol{u} \text{ is nearly unanimous}\}$$

and whose edge set is

$$\begin{aligned}&\{\boldsymbol{uw} \mid \boldsymbol{u}, \boldsymbol{w} \in V(\mathcal{M}_k(H)) \text{ and } \boldsymbol{uw} \in E(H^k)\} \cup \\ &\quad \{\overline{x}\boldsymbol{w} \mid x \in V(H) \text{ and } \exists\, \boldsymbol{uw} \in E(H^k), \text{ where } x \text{ occurs } k-1 \text{ times in } \boldsymbol{u}\}\end{aligned}$$

When we constructed $\mathcal{M}_k(H)$, we did not add any edges between the constant vertices of H^k. Thus the subgraph of $\mathcal{M}_k(H)$ induced by $\{\overline{x} \mid x \in V(H)\}$ is isomorphic to H; we will denote this subgraph by $H_{\mathcal{M}_k}$. We will show that H admits a near unanimity function of arity k if and only if $H_{\mathcal{M}_k}$ is a retract of $\mathcal{M}_k(H)$, and we will also show that all holes on $H_{\mathcal{M}_k}$ of size at most $k-1$ are also holes on $\mathcal{M}_k(H)$. Then, we use this information to relate $\mathcal{AR}_H$ and the graphs that admit a near unanimity function.

Proposition 2.7. *Let H be a graph and let $k \geq 3$ be an integer. Then H admits a near unanimity function of arity k if and only if $H_{\mathcal{M}_k}$ is a retract of $\mathcal{M}_k(H)$.*

Proof. Suppose that H admits a near unanimity function η of arity k. Define a function $\theta : \mathcal{M}_k(H) \to H_{\mathcal{M}_k}$ as follows:

$$\theta(\boldsymbol{w}) = \overline{\eta(\boldsymbol{w})} \text{ for all } \boldsymbol{w} \in \mathcal{M}_k(H).$$

We claim that θ is a retraction from $\mathcal{M}_k(H)$ to $H_{\mathcal{M}_k}$. The function θ is well defined as the vertex set of $\mathcal{M}_k(H)$ is a subset of $V(H^k)$. It is easy to see that θ fixes each vertex of $H_{\mathcal{M}_k}$, and that the range of θ is $V(H_{\mathcal{M}_k})$. It is also easy to verify that θ is a homomorphism.

Now suppose that there exists a retraction θ from $\mathcal{M}_k(H)$ to $H_{\mathcal{M}_k}$. Then define the function $\eta : H^k \longrightarrow H$ as follows:

$$\eta(\boldsymbol{w}) = \begin{cases} x & \text{if } \boldsymbol{w} \in V(\mathcal{M}_k(H)) \text{ and } \theta(\boldsymbol{w}) = \overline{x} \\ y & \text{if } y \text{ appears } k-1 \text{ times in } \boldsymbol{w}. \end{cases}$$

By definition, η sends all constant and nearly unanimous vertices of H^k to the appropriate vertices in H. Again, it is easy to see that η is a homomorphism, and hence a near unanimity function.

□

Let H be a connected graph and let p be an integer, $p \geq 3$. Before proving that every k-hole on $H_{\mathcal{M}_p}$ is a hole on $\mathcal{M}_p(H)$, where $2 \leq k \leq p-1$, we will first show that every 2-hole on $H_{\mathcal{M}_p}$ is a 2-hole on $\mathcal{M}_p(H)$, $3 \leq p$, i.e., $H_{\mathcal{M}_p}$ is an isometric subgraph of $\mathcal{M}_p(H)$.

Lemma 2.4. *Let H be a graph and $p \geq 3$ an integer. Then $H_{\mathcal{M}_p}$ is an isometric subgraph of $\mathcal{M}_p(H)$.*

Proof. Suppose that $H_{\mathcal{M}_p}$ is not an isometric subgraph of $\mathcal{M}_p(H)$; thus there exits a path P in $\mathcal{M}_p(H)$ from a vertex $\overline{x}$ to another vertex $\overline{y}$ of length $d_{\mathcal{M}_p(H)}(\overline{x},\overline{y})$, where $d_{H_{\mathcal{M}_p}}(\overline{x},\overline{y}) > d_{\mathcal{M}_p(H)}(\overline{x},\overline{y})$. Without loss of generality, we may assume that the internal vertices of P lie outside $H_{\mathcal{M}_p}$. Thus the internal vertices of P consist of vertices that are neither constant nor nearly unanimous; recall that we removed all nearly unanimous vertices when creating $\mathcal{M}_p(H)$. Let $P' = P \setminus \{\overline{x},\overline{y}\}$. Then, by the definition of $\mathcal{M}_p(H)$, there exists vertices $\boldsymbol{x}'$ and $\boldsymbol{y}'$ of H^p such that $\boldsymbol{x}'P'\boldsymbol{y}'$ is a path in H^p where x occurs at least $p-1$ times in $\boldsymbol{x}'$ and y occurs at least $p-1$ times in $\boldsymbol{y}'$; it is possible that $\overline{x} = \boldsymbol{x}'$ or $\overline{y} = \boldsymbol{y}'$. Note that P and $\boldsymbol{x}'P'\boldsymbol{y}'$ have the same length. Since $p \geq 3$, there exists an index q such that $\pi_q(\boldsymbol{x}') = x$ and $\pi_q(\boldsymbol{y}') = y$, where π_q is the q^{th} projection. Therefore $d_{H^p}(\boldsymbol{x}',\boldsymbol{y}') \geq d_H(x,y) = d_{H_{\mathcal{M}_p}}(\overline{x},\overline{y})$. This implies that the path P' in H^p has length at least $d_{H_{\mathcal{M}_p}}(\overline{x}, \overline{y})$, which is impossible since the path P has length at most $d_{H_{\mathcal{M}_p}}(\overline{x}, \overline{y}) - 1$ in $\mathcal{M}_p(H)$.

□

Lemma 2.5. *Let H be a connected graph and $p \geq 3$ an integer. If f is a hole of size k, $k \leq p-1$, on $H_{\mathcal{M}_p}$, then f is also a hole on $\mathcal{M}_p(H)$.*

Proof. Suppose that there exists a hole f on $H_{\mathcal{M}_p}$ of size k, $k \leq p-1$, such that f is a feasible distance constraint on $\mathcal{M} = \mathcal{M}_p(H)$. We assume that f is a **minimum** such hole on $H_{\mathcal{M}_p}$ in the following sense: if f' is a hole on $H_{\mathcal{M}_p}$ of size at most k such that $\sum_{\boldsymbol{y} \in D_{f'}} f'(\boldsymbol{y}) < \sum_{\boldsymbol{x} \in D_f} f(\boldsymbol{x})$ then f' is also a hole on $\mathcal{M}$. Note that by Lemma 2.4, we may assume that $2 < k \leq p-1$.

Since f is feasible on $\mathcal{M}$ but not on $H_{\mathcal{M}_p}$, there exists a vertex $\boldsymbol{a} \in F_{\mathcal{M}}(f) \setminus V(H_{\mathcal{M}_p})$. Let $D_f = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k\}$ and let P_i be a shortest path in $\mathcal{M}$ from $\boldsymbol{x}_i$ to $\boldsymbol{a}$, $i = 1, \ldots, k$. Then, P_i has length at most $f(\boldsymbol{x}_i)$, $i = 1, \ldots, k$.

We will prove that $\boldsymbol{x}_i$ is the only vertex on P_i that is in $H_{\mathcal{M}_p}$. Let $\boldsymbol{y}_i$ be the last vertex on P_i that is in $H_{\mathcal{M}_p}$, $i = 1, \ldots, k$. Suppose that there exists an index j such that $\boldsymbol{x}_j \neq \boldsymbol{y}_j$. As $H_{\mathcal{M}_p}$ is an isometric subgraph of $\mathcal{M}$ by Lemma 2.4, we may assume that $P_i[\boldsymbol{x}_i, \boldsymbol{y}_i]$ is in $H_{\mathcal{M}_p}$, $i = 1, \ldots, k$. Let f' be a distance constraint on $H_{\mathcal{M}_p}$ with domain $D_{f'} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k\}$ defined by $f'(\boldsymbol{y}_i) = f(\boldsymbol{x}_i) - d_{H_{\mathcal{M}_p}}(\boldsymbol{x}_i, \boldsymbol{y}_i)$ for $i = 1, \ldots, k$. Then f' is clearly a hole on $H_{\mathcal{M}_p}$ such that f a feasible distance constraint on $\mathcal{M}$. Since $\boldsymbol{x}_j \neq \boldsymbol{y}_j$, $\sum_{\boldsymbol{y} \in D_{f'}} f'(\boldsymbol{y}) < \sum_{\boldsymbol{x} \in D_f} f(\boldsymbol{x})$, we have a contradiction. Therefore $\boldsymbol{x}_i$ is the only vertex on P_i that is in $H_{\mathcal{M}_p}$.

Consider the vertex $\boldsymbol{x}_i$, $i = 1, \ldots, k$. Since $\boldsymbol{x}_i \in V(H_{\mathcal{M}_p})$, $\boldsymbol{x}_i$ is a constant vertex and so there exists a vertex $z_i \in V(H)$ such that $\boldsymbol{x}_i = \overline{z}_i$, $i = 1, \ldots, k$. Let $P_i' = P_i \setminus \boldsymbol{x}_i$, $i = 1, \ldots, k$. Because $\boldsymbol{x}_i$ is the only constant vertex on P_i and by the definition of $\mathcal{M}$, there exits a vertex $\boldsymbol{y}_i$ in H^p such that $\boldsymbol{y}_i P_i'$ is a path in H^p and z_i occurs at least $p-1$ times in $\boldsymbol{y}_i$, $i = 1, \ldots, k$. Note that the length of $\boldsymbol{y}_i P_i'$ is equal to the length of P_i, which is bounded above by $f(\boldsymbol{x}_i)$. Thus $d_{H^p}(\boldsymbol{y}_i, \boldsymbol{a}) \leq f(\boldsymbol{x}_i)$. Lastly, as $k \leq p-1$, there exists an index q such that $\pi_q(\boldsymbol{y}_i) = z_i$ for $i = 1, \ldots, k$, where π_q is the q^{th} projection. Then $d_H(z_i, \pi_q(\boldsymbol{a})) \leq d_{H^p}(\boldsymbol{y}_i, \boldsymbol{a}) = d_{\mathcal{M}}(\boldsymbol{x}_i, \boldsymbol{a}) \leq f(\boldsymbol{x}_i)$, $i = 1, \ldots, k$. Let

$\boldsymbol{a}^* = \overline{\pi_q(a)}$. Then

$$d_{H_{\mathcal{M}_p}}(\boldsymbol{x}_i, \boldsymbol{a}^*) \leq f(\boldsymbol{x}_i) \quad i = 1, \ldots, k,$$

contradicting the infeasibility of f on $H_{\mathcal{M}_p}$.

□

Thus, we have proved that if H is a connected graph and p is large enough, all holes on $H_{\mathcal{M}_p}$ are also holes on $\mathcal{M}_p(H)$. Using Proposition 2.7 and Lemma 2.5, we can now comment on H and near unanimity functions, when $H \in \mathcal{AR}_H$.

The following theorem is implied by [33] with they state that the k-Helly property is equivalent being of strict width k; strict width k is the same as admitting a near unanimity function of arity $k+1$.

Theorem 2.8. *Each graph H in $\mathcal{AR}_H$ admits a near unanimity function. In particular, there exists an integer k such that all holes on H are of size at most $k-1$ and H admits a near unanimity function of arity k.*

Proof. As the domain of any hole on H is a subset of $V(H)$, there exits an integer k such that all holes on H are of size at most $k-1$; indeed $k - 1 \leq |V(H)|$.

Consider the graph $H_{\mathcal{M}_k}$. As $H_{\mathcal{M}_k}$ and H are isomorphic, $H_{\mathcal{M}_k} \in \mathcal{AR}_H$ and all holes on $H_{\mathcal{M}_k}$ are of size at most $k-1$. Then, by Lemma 2.5, all holes on $H_{\mathcal{M}_k}$ are holes on $\mathcal{M}_k(H)$, implying that $H_{\mathcal{M}_k}$ is a retract of $\mathcal{M}_k(H)$. Therefore H admits a near unanimity function of arity k by Proposition 2.7.

□

Corollary 2.4. *If H is graph in $\mathcal{AR}_H$ and in the variety generated by connected chordal graphs, then H admits a near unanimity function of arity k, where* $k = \max\{3, \lfloor |V(H)|/2 \rfloor + 1\}$.

Proof. Let H be a graph in $\mathcal{AR}_H$ and in the variety generated by connected chordal graphs.

If H has only degenerate holes, then H is in $\mathcal{AR}_I$, and thus it admits a majority function, all by Theorem 1.2. Hence assume H has at least one non-degenerate hole.

If H is chordal, then by Corollary 2.2, the size of the largest hole of H is at most $\lfloor |V(H)|/2 \rfloor$. Thus by Theorem 2.8, H admits a near unanimity function of arity $\lfloor |V(H)|/2 \rfloor + 1$.

If H is in the variety generated by chordal graphs, then as seen in the proof of Theorem 2.7, there exist connected chordal graphs $J_1, \ldots, J_p$ such that each graph J_i has no more vertices than H and H is a retract of $\prod_{i=1}^{p} J_i$.

Note that if J_i admits a near unanimity function of arity k, $i = 1, \ldots, p$, then H also admits a near unanimity function of arity k as the class of graphs that admit a near unanimity function of arity k is a variety by Theorem 1.5. By the reasoning in the previous paragraphs, the chordal graph J_i admits a near unanimity function of arity $\max\{3, \lfloor |V(J_i)|/2 \rfloor + 1\}$, $i = 1, \ldots, p$. Therefore H admits a near unanimity function of arity $\max\{3, \lfloor |V(H)|/2 \rfloor + 1\}$.

□

In [15], the authors have the following bound for the arity of near unanimity functions on chordal graphs:

Theorem 2.9. *[15] Let H be a chordal graph on n vertices with maximum clique size ω, $\omega \geq 3$. Then H admits a near unanimity function of arity k, where $k \leq \min\{n - \omega + 2, n/(\omega - 1) + 1\}$. In particular, $k - 1 \leq n - \sqrt{n}$.*

Corollary 2.4 and Theorem 2.9 are both best possible. The chordal graph in Figure 1.2 is a chordal graph on 6 vertices that admits a near unanimity function of arity 4 by these results, but, by the following proposition, does not admit a majority function as the graph has a 3-hole.

Proposition 2.8. *[15] Let H be a connected graph with a hole of size $k \geq 3$. The H does not admit a near unanimity function of arity k.*

Proof. Let f be a hole of H of size k, $k \geq 3$. Let $D_f = \{x_1, \ldots, x_k\}$, and let a_i be an x_i-relaxed filler of f for $i = 1, \ldots k$. Now consider the vectors in the vertex set of H^k. Let $\boldsymbol{v}^{x_i}$ be the vector with a_i in its i^{th} position and x_i in all other positions, for $i = 1, \ldots, k$ and let $\boldsymbol{a}$ be the vector $(a_1, \ldots, a_k)$. It is clear that there is a path from $\boldsymbol{v}^{x_i}$ to $\boldsymbol{a}$ of length $f(x_i)$ in H^k. Thus there can't be near unanimity function of arity k, as $\boldsymbol{v}^{x_i}$ would be forced to be mapped to x_i, $i = 1, \ldots, k$, hence making it impossible to send $\boldsymbol{a}$ anywhere.

□

Corollary 2.5. *Let H be a graph in $\mathcal{AR}_H$ and let $k-1$ be the size of the largest hole of H. Then H admits a near unanimity function of arity k, but does not admit a near unanimity function of arity $k-1$.*

□

In Section 1.2.2, we presented basic properties of dismantlable graphs, and those graphs that admit near unanimity functions. In particular, we presented Theorem 1.7, stating that connected graphs that admit near unanimity functions are dismantlable. We can now use this to relate $\mathcal{AR}_H$ and dismantlable graphs.

Corollary 2.6. *Each graph in $\mathcal{AR}_H$ is dismantlable.*

Proof. Let H be a graph in $\mathcal{AR}_H$; thus H is connected and H admits a near unanimity function by Theorem 2.8. By Theorem 1.7, H is dismantlable.

□

Therefore, as we moved from $\mathcal{AR}_I$ to $\mathcal{AR}_H$, we have retained the following properties: admitting a near unanimity function and being dismantlable.

Chapter 3

Tree Obstructions

A *leaf-labeled tree* (T, ℓ) is a tree T, together with a labeling ℓ of its leaves; we denote by $\ell(x)$ the label of leaf x. Suppose T' is a subtree of T such that all leaves of T' are also leaves of T. Let ℓ' be labeling of the leaves of T' such that $\ell'(x) = \ell(x)$ for all leaves x of T'. Then we say that the leaf-labeled tree (T', ℓ') is a *leaf-labeled subtree* of the leaf-labeled tree (T, ℓ), denoted by $(T', \ell') \subseteq (T, \ell)$.

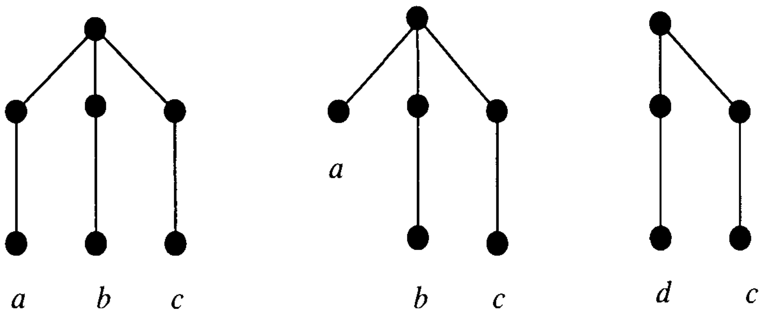


Figure 3.1: Let (T, ℓ) be the leaf-labeled tree on the left, (T', ℓ') the leaf-labeled tree in the middle and (T'', ℓ'') the leaf-labeled tree on the right, where the labels of the of leaves are as indicated in the figure.

Consider the leaf-labeled trees in Figure 3.1. While T' and T'' are subtrees of T, neither (T', ℓ') nor (T'', ℓ'') is a leaf-labeled subtree of (T, ℓ); there is a leaf of T' that is not a leaf of T, and while all leaves of T'' are leaves of T, there exists a leaf x of T

and T'' such that $\ell(x) \neq \ell''(x)$.

Recall that we *subdivide* an edge e of a graph H by removing e from H and replacing it with a path of length 2. A graph H' is called a *subdivision* of a graph H if H' can be obtained from H by a sequence of edge subdivisions. Note that H is a subdivision of itself by the empty sequence of edge subdivisions. Let (T, ℓ) a leaf-labeled tree. If T' is a subdivision of T, then we say that (T', ℓ) is a subdivision of (T, ℓ).

A *tree constraint* on a connected graph H is a leaf-labeled tree (T, ℓ), where the range of ℓ is a subset of $V(H)$. A tree constraint (T, ℓ) on a connected graph H is called *feasible* if there exists a homomorphism ϕ from T to H such that $\phi(x) = \ell(x)$ for each leaf of x of T; we say that ϕ *extends* ℓ. If such a homomorphism does not exist, then (T, ℓ) is called *infeasible*.

As with the distance constraints of Chapter 2, we will define a partial order on the set of tree constraints on a given connected graph H. Let (T', ℓ') and (T, ℓ) be two tree constraints on H. We say that $(T', \ell') \leq (T, \ell)$ if (T', ℓ') is a subdivision of some leaf-labeled subtree of (T, ℓ). Moreover $(T', \ell') = (T, \ell)$ if $T = T'$ and $\ell'(x) = \ell(x)$ for all leaves x of T'. We say that $(T', \ell') < (T, \ell)$ if $(T', \ell') \leq (T, \ell)$ and $(T', \ell') \neq (T, \ell)$. In particular, if T_e is the tree obtained by subdividing the edge e of tree T, then $(T_e, \ell) < (T, \ell)$.

Let H be a connected graph. Let (T, ℓ) a be tree constraint on H. We say that (T, ℓ) is a minimally infeasible tree constraint, or a *tree obstruction*, if (T, ℓ) is infeasible and all tree constraints (T', ℓ') on H such that $(T', \ell') < (T, \ell)$ are feasible. Thus, for each edge e of T, the tree constraint (T_e, ℓ) is feasible as $(T_e, \ell) < (T, \ell)$, and so there exists a homomorphism from T_e to H that extends ℓ; denote such a homomorphism by ϕ_e.

Related structures have been studied recently by others. Infeasible tree constraints have been studied by [15] under the name of *tree certificates*. More generally, let H and J be graphs, and let ℓ a partial labeling of the vertices of J with the vertices of H. Then (J, ℓ) is an *obstruction* [49] or a *conflict* [15] on H if there does not exist a homomorphism $\phi : J \to H$ that extends ℓ, but there does exist a homomorphism

$\phi : J' \to H$ that extends ℓ' for any proper subgraph J' of J, where $\ell'(x) = \ell(x)$ for all labeled vertices of J'.

Proposition 3.1. *Let H be a connected graph and let (T, ℓ) be tree constraint on H.*

i.) For any tree constraint (T', ℓ') on H such that $(T', \ell') < (T, \ell)$, there exists an edge e of T such that $(T', \ell') \leq (T_e, \ell) < (T, \ell)$.

ii.) Let $e = tt'$ be an edge of T. Then

$$d_H(\phi_e(t), \phi_e(t')) = 2,$$

where ϕ_e is a homomorphism from T_e to H that extends ℓ.

Proof. <u>i.</u> Let (T', ℓ') be a tree constraint on H such that $(T', \ell') < (T, \ell)$. Suppose that (T', ℓ') a leaf-labeled subtree of (T, ℓ) and that T' is a proper subgraph of T. Then there exists an edge $e \in E(T) \setminus E(T')$. Then $(T', \ell') < (T_e, \ell)$. Now suppose that there exists a leaf-labeled subtree (T'', ℓ'') of (T, ℓ) such that (T', ℓ') is subdivision of (T'', ℓ'') and $T' \neq T''$. Then there exists an edge $e \in E(T'')$ such that e was subdivided in the creation of (T', ℓ'). Then clearly $(T', \ell') \leq (T''_e, \ell'') \leq (T_e, \ell) < (T, \ell)$.

<u>ii.</u> As (T, ℓ) is a tree obstruction, (T_e, ℓ) is a feasible tree constraint on H. Therefore there exists a homomorphism $\phi_e : T_e \to H$ that extends ℓ. As ϕ_e is a homomorphism, $d_H(\phi_e(t), \phi_e(t')) \leq 2$. If $\phi_e(t)$ and $\phi_e(t')$ are adjacent in H, then the restriction of ϕ_e to $V(T)$ is a homomorphism from T to H that extends ℓ. This contradicts the infeasibility of (T, ℓ). Hence $d_H(\phi_e(t), \phi_e(t')) = 2$.

$\square$

Proposition 3.2. *Let H be a connected graph. Let (T, ℓ) be a tree obstruction on H and let x and y be two distinct leaves of T.*

i.) If T is a path, then

$$d_H(\ell(x), \ell(y)) = d_T(x, y) + 1.$$

ii.) If T is not a path, then

$$d_H(\ell(x), \ell(y)) \leq d_T(x, y).$$

Proof. $\underline{i.}$ Suppose that T is a path. If $d_H(\ell(x), \ell(y)) \leq d_T(x, y)$, then we could map the path T to a walk of length $d_T(x, y)$ in H between $\ell(x)$ and $\ell(y)$; the needed walk is a shortest $\ell(x) - \ell(y)$ path in H plus the vertex $\ell(y)$ repeated as many times as necessary. Since (T, ℓ) is infeasible, this is impossible, and so $d_H(\ell(x), \ell(y)) > d_T(x, y)$. Let e be a non-loop edge of T. As (T, ℓ) is minimally infeasible, the tree constraint (T_e, ℓ) is feasible on H and so there exists a homomorphism $\phi_e : T_e \longrightarrow H$ that extends ℓ. Thus, $d_H(\ell(x), \ell(y)) \leq d_{T_e}(x, y)$. We created T_e by subdividing the edge e of T, and so $d_{T_e}(x, y) = d_T(x, y) + 1$. Therefore $d_H(\ell(x), \ell(y)) = d_T(x, y) + 1$.

$\underline{ii.}$ Now suppose that T is not a path. Then there exists an edge e that is not on the unique $x - y$ path in T. As (T, ℓ) is a tree obstruction, (T_e, ℓ) is a feasible tree constraint on H and so there exists a homomorphism $\phi_e : T_e \longrightarrow H$ that extends ℓ. Therefore $d_H(\ell(x), \ell(y)) \leq d_{T_e}(x, y)$. Since we chose e to be an edge off the $x - y$ path in T, $d_T(x, y) = d_{T_e}(x, y)$. Hence $d_H(\ell(x), \ell(y)) \leq d_T(x, y)$.

□

Let T be a tree. A vertex $b \in V(T)$ of degree at least 3 is called a *branching point* of T. In addition, a leaf x of T is called *direct leaf* of the vertex b in T if b is the only branching point on the unique path from b to x in T.

At this point we would like to point out a correspondence between tree constraints on a connected graph H and the distance constraints (see Chapter 2) on H. This correspondence will demonstrate that tree constraints are a very natural generalization of distance constraints.

Proposition 3.3. *Let H be a connected graph.*

i.) For each distance constraint f on H there exists a tree constraint (T, ℓ) on H such that f is feasible if and only if (T, ℓ) is feasible.

ii.) For each tree constraint (T, ℓ) on H, where T has at most one branching point and distinct leaves of T have distinct labels, there exists a distance constraint f on H such that (T, ℓ) is feasible if and only if f is feasible.

Proof. $\underline{i.}$ Let f be a distance constraint on H with $D_f = \{x_1, \ldots, x_k\}$, $k \geq 2$. If $k = 2$, let T be a path of length $f(x_1) + f(x_2) = d_H(x_1, x_2) - 1$, with endpoints (leaves) y_1 and y_2. If $k \geq 3$, let T be a tree with branching point b, where b has direct leaves $y_1, \ldots, y_k$ such that $d_T(b, y_k) = f(x_k)$, $i = 1, \ldots, k$. In both cases, assign the leaf y_i the label $\ell(y_i) = x_i$, for $i = 1, \ldots, k$. Clearly f is feasible if and only if (T, ℓ) is feasible.

ii. Let (T, ℓ) be a tree constraint on H, where T has at most one branching point and distinct leaves of T have distinct labels. Let $\{y_1, \ldots, y_k\}$ be the leaves of T. Then let f be a distance constraint on H with $D_f = \{\ell(y_1), \ldots, \ell(y_k)\}$ defined by

$$f(\ell(y_1)) = 0 \text{ and } f(\ell(y_2)) = d_T(y_1, y_2) = d_H(\ell(y_1), \ell(y_2)) - 1$$

if $k = 2$, or defined by

$$f(\ell(y_i)) = d_T(y_i, b) \text{ for } i = 1, \ldots, k$$

if $k \geq 3$ and b the is branching point of T. Note that f is well defined as distinct leaves of T have distinct labels. Clearly (T, ℓ) is feasible if and only if f is feasible.

□

3.1 Absolute retracts with respect to tree obstructions

In this section, we present a new class of absolute retracts with respect to a necessary condition N, where N has to do with preserving tree obstructions. Let H be a connected graph and let G be a connected supergraph of H. As with distance constraints, each tree constraint (T, ℓ) on H is a tree constraint on G because $\ell(x) \in V(H) \subseteq V(G)$ for all leaves x of T. We will prove that if a graph H is a retract of a supergraph G, then all tree obstructions on H are also tree obstructions on G. We will then use this property to define the class of absolute retracts with respect to tree obstructions. Lastly, we will prove that the class of absolute retracts with respect to tree obstructions is a variety.

Proposition 3.4. *Let H be a connected graph, and let G be a connected supergraph of H such that H is a retract of G. Let (T, ℓ) be an infeasible tree constraint on H. Then (T, ℓ) must also be infeasible on G.*

Proof. Suppose not. Then there exists a homomorphism $\phi : T \rightarrow G$ that extends ℓ. Let $\theta : G \rightarrow H$ be a retraction. Then $\theta \circ \phi$ is a map from T to H such that for each leaf t of T, $\theta(\phi(t)) = \theta(\ell(t)) = \ell(t)$. Thus $\theta \circ \phi : T \rightarrow H$ is an extension of ℓ, contradicting the infeasibility of (T, ℓ) on H.

□

Thus, by Proposition 3.4, if a connected graph H is a retract of a connected supergraph G, it is necessary that each infeasible tree constraint on H is an infeasible tree constraint on G. We will now show that this is equivalent to each tree obstruction on H being a tree obstruction on G.

Proposition 3.5. *Let H be a connected graph and let G be a connected supergraph of H. Then the following conditions are equivalent:*

i.) Each infeasible tree constraint on H is an infeasible tree constraint on G.

ii.) Each tree obstruction on H is a tree obstruction on G.

Proof. $\underline{i \Rightarrow ii.}$ Assume that each infeasible tree constraint on H is an infeasible tree constraint on G. Let (T, ℓ) be a tree obstruction on H. By definition (T, ℓ) is an infeasible tree constraint on H, and thus (T, ℓ) is an infeasible tree constraint on G by assumption. Now all we have to do is prove that (T, ℓ) is minimally infeasible on G. By Proposition 3.1, it is sufficient to prove that (T_e, ℓ) is feasible on G for each $e \in E(T)$. Since (T, ℓ) is a tree obstruction on H, (T_e, ℓ) is feasible on H for each $e \in E(T)$. Thus there exists a homomorphism $\phi_e : T_e \rightarrow H$ that extends ℓ. As H is a subgraph of G, we have that $\phi_e : T_e \rightarrow G$ extends ℓ for each edge $e \in E(T)$. Therefore (T_e, ℓ) is a feasible tree constraint on G for each $e \in E(T)$.

$\underline{ii \Rightarrow i.}$ Assume that each tree obstruction on H is a tree obstruction on G. Suppose that there exists an infeasible tree constraint (T, ℓ) on H that is feasible on G. There exists a tree obstruction (T', ℓ') on H such that $(T', \ell') < (T, \ell)$. By assumption, (T', ℓ') is also a tree obstruction on G. Since (T, ℓ) is feasible on G, there exists a homomorphism $\phi : T \longrightarrow G$ that extends ℓ. Since $(T', \ell') < (T, \ell)$, there exists a leaf-labeled subtree (T'', ℓ'') of (T, ℓ) such that (T', ℓ') is a subdivision of (T'', ℓ''). If $(T'', \ell'') = (T', \ell')$, then the restriction of ϕ to T' is a homomorphism from T' to G that extends ℓ, which is a contradiction as (T', ℓ') is a tree obstruction on G. Thus (T', ℓ') is a subdivision of (T'', ℓ''). Hence there exists an edge e of T'' that we subdivided in the process of making T'. Let ϕ'' be the restriction of ϕ to T''. We can create a homomorphism from T''_e to G that extends ℓ''. Let x and y be the endpoints of e in T'', and let xzy be the new path of length 2 in T''_e. Then define $\phi''_e : T''_e \longrightarrow G$ as follows:

$$\phi''_e(t) = \begin{cases} \phi''(x) & \text{if } t = z \\ \phi''(t) & \text{otherwise.} \end{cases}$$

This is a homomorphism as ϕ'' is a homomorphism and because G is reflexive. It easy to see that $\phi''_e : T''_e \longrightarrow G$ extends ℓ''. Continuing in this manner, we can thus create a homomorphism $\phi' : T' \longrightarrow G$ that extends ℓ', contradiction.

□

Let H be a connected graph and let G be a connected supergraph of H. As mentioned after Proposition 3.4, a necessary condition for H to be a retract of G is for each infeasible tree constraint on H to be an infeasible tree constraint on G. By Proposition 3.5, this is equivalent to saying that a necessary condition for H to be a retract of G is for each tree obstruction on H to be a tree obstruction on G. Now we can define the class of absolute retracts with respect to tree obstructions.

The class of *absolute retracts with respect to tree obstructions*, denoted by $\mathcal{AR}_O$, is the set of all connected graphs H such that H is a retract of a connected supergraph G whenever each tree obstruction on H is also a tree obstruction on G.

Note that if H is connected graph that is a subgraph of another connected graph G, where each tree obstruction on H is a tree obstruction on G, H need not be a retract of G, see Figure 4.1. Let H be the graph induced by the round vertices, and let G be the entire graph, where the square vertices form a clique.

Let H be a graph in $\mathcal{AR}_H$. Then H is a retract of a connected supergraph G whenever each hole on H is a hole on G. Let G be a connected supergraph of H such that each tree obstruction on H is a tree obstruction on G. By Proposition 3.3, this implies that each hole on H is a hole on G and so H is a retract of G. Thus $\mathcal{AR}_H \subseteq \mathcal{AR}_O$.

Theorem 3.1. *The class of graphs $\mathcal{AR}_O$ is a variety.*

Proof. We will first prove that $\mathcal{AR}_O$ is closed under taking retractions, and then secondly we will prove that $\mathcal{AR}_O$ is closed under taking products.

Let H be graph in $\mathcal{AR}_O$, and let H' be a retract of H. Let G' be a connected supergraph of H' such that all infeasible tree constrainsts on H are infeasible tree constraints on G'. If we can prove that H' is a retract of G', then $H' \in \mathcal{AR}_O$ by Proposition 3.5.

As in the proof of Theorem 2.1, we may assume that $H' = H \cap G'$. Let $G = H \cup G'$. As H and G' are both connected, so is G. Thus, if we can prove the each infeasible tree constraint on H is an infeasible tree constraint on G, then H is a retract of G by Proposition 3.5. We can use this to prove that H' is a retract of G'.

Let (T, ℓ) be an infeasible tree constraint on H. Suppose that (T, ℓ) is feasible on G. Hence there exists a homomorphism $\phi : T \to G$ that extends ℓ. Let T' be the largest subtree of T such that $\phi(T') \subseteq G'$. For each leaf x' of T', assign x' the label $\ell'(x') = \phi(x)$. Since $H' = H \cap G'$, $G = H \cup G'$, and all paths from $G \setminus H$ to $H \setminus G$ pass through H', we have that $\ell'(x') \in V(H')$ for each leaf x' of T'. It is easy to see that (T', ℓ') is an infeasible tree constraint on H'. Then, by the way that we chose G', the tree constraint (T', ℓ') must infeasible on G' also. Yet if we restrict the homomorphism ϕ to $V(T')$, we have a homomorphism from T' to G' that extends ℓ', contradiction. Hence (T, ℓ) must be infeasible on G, and so H is a retract of G. Using the same arguments as in the proof of Theorem 2.1, we can now construct a retraction from G' to H'.

Let $H = H_1 \times H_2$, where $H_1, H_2 \in \mathcal{AR}_O$. Let $\pi_i : H \to H_i$ be the i^{th} projection, $i = 1, 2$. Let G be a connected supergraph of H such that every infeasible tree constraint on H is also an infeasible tree constraint on G. If we can prove that H is a retract G, then $H \in \mathcal{AR}_o$ by Proposition 3.5. Let G_i be the graph we obtain from G by identifying all the vertices $h, h' \in V(H)$ such that $\pi_i(h) = \pi_i(h')$. Clearly G_i is connected and H_i is a subgraph of G_i for $i = 1, 2$. We will prove that each infeasible tree constraint of H_i is also an infeasible tree constraint of G_i, thus implying existence of a retraction $\theta_i : G_i \to H_i$, for $i = 1, 2$, by Proposition 3.5. The retractions θ_1 and θ_2 can then be used to construct a retraction from G to H, as in the proof of Theorem 2.1.

Let (T_1, ℓ_1) be an infeasible tree constraint on H_1 where T_1 has leaves $\{x_1, \ldots, x_k\}$. Let y be a vertex of H_2. Then the tree constraint (T_1, ℓ) on H with $\ell(x_i) = (\ell_1(x_i), y)$, $i = 1, \ldots, k$, must be an infeasible tree constraint on H. Suppose that there exists homomorphism $\phi_1 : T_1 \to G_1$ that extends ℓ. Then the homomorphism $\phi : T_1 \to G$ defined by

$$\phi(t) = \begin{cases} (\phi_1(t), y) & \text{if } \phi_1(t) \in V(H_1) \\ \phi_1(t) & \text{otherwise} \end{cases}$$

extends ℓ, which contradicts the way we chose G. Therefore (T_1, ℓ_1) must also be an infeasible tree constraint on G_1, and so there exists a retraction $\theta_1 : G_1 \to H_1$. Similarly, there exists a retraction $\theta_2 : G_2 \to H_2$.

□

3.2 Tree duality

In [39], the authors introduced the idea of *tree duality*. An irreflexive digraph H is said to have tree duality if and only if the following two statements are equivalent for all irreflexive digraphs G:

i.) $G \not\to H$

ii.) There exists an oriented tree T such that $T \to G$ but $T \not\to H$.

We will adapt this idea to list homomorphisms and retractions to create list tree duality and retraction tree duality. We will then relate retraction tree duality and $\mathcal{AR}_o$.

Let G and H be graphs. Assign to each vertex g of G a list $L(g) \subseteq V(H)$. Recall from Section 1.1 that a homomorphism $\phi : G \to H$ is a list homomorphism with respect to L if $\phi(g) \in L(g)$ for all $g \in V(G)$ and we denote this by $(G, L) \xrightarrow{\phi} H$. List homomorphisms have been studied in [17, 28, 29, 30, 31, 32, 69]. Let J be a graph and assign to each vertex x of J a list $L'(x) \subseteq V(H)$. Recall also from Section 1.1, that (J, L') is homomorphic to (G, L) if there exists a homomorphism ϕ, $J \xrightarrow{\phi} G$, such that $L'(x) = L(\phi(x))$ for all vertices x of J. This is denoted by $(J, L') \xrightarrow{\phi} (G, L)$. We say that a graph H (of whatever sort) has *list tree duality* if and only if the following two statements are equivalent for all graphs G:

i.) $(G, L) \not\to H$.

ii.) There exists a tree T with lists L' such that $(T, L') \to (G, L)$ but $(T, L') \not\to H$.

Consider the list homomorphism question in this particular case; let H be a graph and let G be a supergraph of H, where the lists L are

$$L(g) = \begin{cases} \{g\} & \text{if } g \in V(H) \\ V(H) & \text{otherwise.} \end{cases}$$

It is easy to see that $(G, L) \to H$ if and only if there exists a retraction from G to H. Thus we may adapt list tree duality to retraction tree duality by interpreting the retraction problem as a particular type of list homomorphism problem. A connected graph H has *retraction tree duality* if and only if the following two statements are equivalent for all connected supergraphs G of H with list assignment $L(g) = \{g\}$ if $g \in V(H)$ and $V(H)$ otherwise:

i.) H is not a retract of G.

ii.) There exists a tree T with lists L' such that $(T, L') \to (G, L)$ but $(T, L') \not\to H$.

By definition, a connected graph H is in $\mathcal{AR}_O$ exactly when the following two statements are equivalent for all connected supergraphs G of H:

i.) H is not a retract of G.

ii.) There exists a tree obstruction (T, ℓ) on H that is feasible on G.

It turns out that a graph H has retraction tree duality if and only if H is in $\mathcal{AR}_O$.

Theorem 3.2. *Let H be connected graph. Then H has retraction tree duality if and only if H is in $\mathcal{AR}_O$.*

Proof. Let H be a connected graph. We will show that H has retraction tree duality if and only if $H \in \mathcal{AR}_O$ by showing that the second statements from the definition of retraction tree duality and the alternate definition of $\mathcal{AR}_O$ are equivalent.

Let G be a connected supergraph of H with lists L, where $L(g) = \{g\}$ if $g \in V(H)$ and $L(g) = V(H)$ otherwise. Suppose that there exists a tree T with lists L' such that $(T, L') \xrightarrow{\phi} (G, L)$ and $(T, L') \not\to H$ and assume that T is minimal with respect to this property; for any proper subgraph J of T $(J, L'_J) \to H$, where L'_J is the restriction of L' to the vertices of J, i.e., $L'_J(x) = L'(x)$ for all $x \in V(J)$. If we can prove that all leaves of T have lists of size one and all other vertices have lists equal to $V(H)$, then we can create an infeasible tree constraint (T, ℓ) on H, where $\ell(x)$ is the single vertex in $L'(x)$, for all leaves x of T. This tree constraint (T, ℓ) must then be feasible on G as $(T, L') \xrightarrow{\phi} (G, L)$. Then there exits a tree obstruction $(\hat{T}, \hat{\ell})$ on H, where $(\hat{T}, \hat{\ell}) \leq (T, \ell)$ on H; clearly $(\hat{T}, \hat{\ell})$ would be feasible on G.

Note that as $(T, L') \xrightarrow{\phi} (G, L)$, we have that $L'(t) = L(\phi(t))$ for all $t \in V(T)$ and so the list of each vertex in T either contains a single vertex from $V(H)$ or is $V(H)$.

Suppose that there exists a leaf x of T such that $L'(x) = V(H)$. Let $L''(t) = L'(t)$ for all $t \in V(T) \setminus \{x\}$. Then by the way that we chose (T, L'), we must have that $(T \setminus x, L'') \xrightarrow{\alpha} H$. Let y be the nontrivial neighbour of x in T and let h be a neighbour of $\alpha(y)$ in H. Then let $\alpha' : T \to H$ be the map

$$\alpha'(t) = \begin{cases} h & \text{if } t = x \\ \alpha(t) & \text{otherwise} \end{cases}$$

It is easy to see that $(T, L') \xrightarrow{\alpha'} H$, which is a contradiction. Therefore each leaf of T has a list of size one.

Now suppose that there exists a non-leaf vertex z of T such that $L'(z)$ has size one. Since all non-leaf vertices of trees are cut vertices, there exists subtrees T' and T'' of T such that $V(T') \cap V(T'') = \{z\}$ and $T' \setminus z$ and $T'' \setminus z$ are the components of $T \setminus z$. Let L' be the lists L restricted to $V(T')$, and let L'' be the lists L restricted to $V(T'')$. By the minimality of (T, L), we know that $(T', L') \xrightarrow{\alpha'} H$ and $(T'', L'') \xrightarrow{\alpha''} H$. Since $L(z) = L'(z) = L''(z)$ are lists of size one, α' and α'' agree on z. Now define the map $\alpha : T \to H$ as follows:

$$\alpha(t) = \begin{cases} \alpha'(t) & \text{if } t \in V(T') \\ \alpha''(t) & \text{if } t \in V(T''). \end{cases}$$

Clearly $(T, L) \xrightarrow{\alpha} H$, which is a contradiction. Therefore the list for each non-leaf vertex of T must be equal to $V(H)$.

Let G be a supergraph of H such that there exists a tree obstruction (T, ℓ) on H that is feasible on G. Hence there exists a homomorphism $\phi : T \to G$ that extends ℓ. Assign to each vertex $g \in V(G)$ the list $L(g)$ such that

$$L(g) = \begin{cases} \{g\} & \text{if } g \in V(H) \\ V(H) & \text{otherwise.} \end{cases}$$

Assign to each vertex $t \in V(T)$ the list $L'(t)$ such that $L'(t) = L(\phi(t))$. Clearly $(T, L') \xrightarrow{\phi} (G, L)$. Since $\phi : T \to G$ extends ℓ, $\phi(x) = \ell(x)$ for each leaf x of T. As (T, ℓ) is a tree constraint on H, $\ell(x) \in V(H)$ for each leaf x of T. Therefore $L'(x) = \{\ell(x)\}$ for each leaf x of T. Since (T, ℓ) is infeasible on H, then clearly $(T, L') \not\to H$.

□

3.3 Tree obstructions on chordal graphs

In Section 2.2, we used convexity to investigate the structure in connected chordal graphs implied by holes. In particular, we proved for any non-degenerate hole f on a connected chordal graph H, the vertices in D_f can't be 'too close'. This idea carries over to tree obstructions on connected chordal graphs; if (T, ℓ) is a tree obstruction on a connected chordal graph H, then the labels of the leaves of T can't be 'too close' in H. However, we do not use convexity to prove this; this method proved to be too unwieldy even when applied to tree constraints with only two branching points. Instead, we will rephrase tree obstructions as list homomorphism problems as was done in the previous section, and use a list homomorphism algorithm.

Given a tree constraint (T, ℓ) on a graph H, we say that (T, L_ℓ) is the *list homomorphism problem related to* (T, ℓ) on H, where

$$L_\ell(t) = \begin{cases} \{\ell(t)\} & \text{if } t \text{ is a leaf} \\ V(H) & \text{otherwise.} \end{cases}$$

Clearly (T, ℓ) is feasible if and only if $(T, L_\ell) \to H$. Furthermore, the lists of (T, L_ℓ) are connected. In [28], the authors proved that the connected list homomorphism problem is polynomial time solvable when the target graph, i.e., the graph being mapped to, is chordal. We will present the algorithm they use, and their proof that it is correct. Then we will apply the algorithm to (T, L_ℓ) to prove that (T, ℓ) can't be 'squished', when (T, ℓ) is a tree obstruction on a chordal graph H.

Algorithm 3.1. [28]

```
Input: A graph chordal graph $H$
     : A perfect elimination ordering $h_1, \dots, h_n$ of $H$.
     : A pair $(G, L)$, where $G$ is a graph and where $L(g) \subset V(H)$ is a
       connected list for each $g \in V(G)$.
Task: To find a list homomorphism from $G$ to $H$ if one exists.
Action: Process the vertices of $H$ using the perfect elimination ordering
        $h_1, \dots, h_n$. For each vertex $g$ of $G$
            * if $h_i \in L(g)$ and $|L(g)| \geq 2$, then remove $h_i$ from $L(g)$.
                  $L(g) \leftarrow L(g) \setminus \{h_i\}$
            ** if $L(g) = \{h_i\}$, then for each neighbour $g'$ of $g$ in $G$,
               remove from $L(g')$ the non-neighbours of $h_i$.
                  $L(g') \leftarrow L(g') \cap N_H(h_i)$
        (do this for $i = 1, \dots, n-1$ or until an empty list is produced)
      : If no empty list is produced, create $\phi : G \to H$ by setting $\phi(g)$ to be
        the single vertex in $L(g)$.
```

Consider the lists at each stage of Algorithm 3.1. The lists we start with are subsets of $V(H)$. Once we process the vertex h_1, each list is empty, a single vertex of $V(H)$ or a subset of $V(H_1)$. In general, once we have processed vertex h_i, $1 \leq i \leq n-1$, all lists are of size at most 1 or are subsets of $V(H_i)$. Thus once we have processed the vertex h_{n-1}, all lists are of size at most one or are subsets of $V(H_{n-1}) = \{h_n\}$. Therefore, once the vertex h_{n-1} has been processed, all lists are of size at most 1.

We have made one slight modification of Algorithm 3.1; we do not process the vertex h_n. Since we are studying reflexive graphs exclusively, h_nh_n is always an edge. The vertex h_n must be processed if H is an irreflexive graph as h_nh_n is never an edge.

We say that Algorithm 3.1 *fails* if an empty list is produced, and we say the Algorithm *succeeds* otherwise.

The following lemma is used in proof of the correctness of Algorithm 3.1.

Lemma 3.1. *[28] A graph H is chordal if and only if for any connected sets $X, Y \subseteq V(H)$, the set $X \cap D(Y, 1)$ is also connected.*

Proof. Without loss of generality, assume that H is connected; if H is not connected, perform the same analysis on each component. Now suppose that H is chordal, and let X and Y be connected vertex sets in H. Suppose that there exists distinct vertices u and v in $X \cap D(Y, 1)$. Since X is connected, there certainly exists a path P from u to v that is contained in X. In fact, we may assume that P is chordless. By Theorem 2.3, $D(Y, 1)$ is convex, and so any chordless path from u to v is contained in $D(Y, 1)$. Thus P must also be contained in $D(Y, 1)$. Hence $X \cap D(Y, 1)$ is connected.

Let H be a graph such that for any connected $X, Y \subseteq V(H)$, $X \cap D(Y, 1)$ is connected. Suppose that $u_1, \ldots, u_k u_1$ is an induced k-cycle in H. Let $Y = \{u_1\}$ and let $X = \{u_2, \ldots, u_k\}$. Clearly Y and X are connected. Thus, by assumption, $X \cap D(Y, 1) = \{u_2, u_k\}$ is connected. Therefore u_2u_k is an edge of H, and as we assumed $u_1, \ldots, u_k u_1$ was an induced cycle, $k = 3$. Therefore H must be chordal.

$\square$

Theorem 3.3. *[28] Let H be a connected chordal graph with perfect elimination ordering $h_1, \ldots, h_n$, and let G be graph such that each vertex g of G has been assigned a connected list $L(g) \subseteq V(H)$. Then $(G, L) \to H$ if and only if Algorithm 3.1 succeeds on (G, L) and H.*

Proof. Suppose that $(G, L) \to H$. We will prove that as we apply Algorithm 3.1 to (G, L) and the perfect elimination ordering $h_1, \ldots, h_n$, the lists assigned to the vertices of G remain connected, and a list homomorphism continues to exist with respect to the newly created lists.

Suppose that before we process the vertex h_i, $L(g)$ is connected for each $g \in V(G)$ and that $(G, L) \rightarrow H$. We will prove that this continues to hold after h_i has been processed. Let g be a vertex of G such that $h_i \in L(g)$.

If $|L(g)| \geq 2$, then we apply action $*$ to $L(g)$ and remove h_i from $L(g)$. Note that $L(g) \subseteq V(H)_{i-1}$ by our comments following Algorithm 3.1. Therefore if h_i has two neighbours in $L(g)$, they are adjacent since h_i is simplicial in H_{i-1}. Hence list connectivity is maintained. Now we must show that a list homomorphism continues to exist after we remove h_i from $L(g)$. Let ϕ be a list homomorphism that existed before removing h_i from $L(g)$, and let g' be a neighbour of g. If $\phi(g) \neq h_i$, there is nothing to prove, so assume that $\phi(g) = h_i$. Let $h_k = \phi(g')$. As $L(g)$ is connected and contains at least two vertices, h_i has a neighbour h_j in $L(g)$. Define a function $\phi^* : G \rightarrow H$ by $\phi^*(x) = \phi(x)$ if $x \neq g$ and $\phi^*(g) = h_j$. To prove that ϕ^* is a list homomorphism we need only prove that h_j and h_k are adjacent. If $k < i$, then action $**$ was performed on the list of g' when we processed h_k, thus ensuring that h_k is adjacent to all the vertices of $L(g)$. If $k \geq i$, then $h_k, h_j \in V(H_{i-1})$, and so h_k and h_j are adjacent as h_i is simplicial in H_{i-1}.

If $|L(g)| = 1$, then we apply action $**$. Let g' be a neighbour of g. By Lemma 3.1, $N(h_i) \cap L(g')$ is connected. Clearly, removing non-neighbours of h_i from $L(g')$ will not affect the existence of a list homomorphism.

Therefore, there exists a list homomorphism from G to H with respect to the final lists produced by Algorithm 3.1. By our comments following Algorithm 3.1, the final lists are singletons. Hence the function created in the last step of Algorithm 3.1 must be list homomorphism of (G, L) to H.

Now assume that Algorithm 3.1 applied to (G, L) and the perfect elimination ordering $h_1, \ldots, h_n$ succeeds. From our comments following Algorithm 3.1, we note that the final lists are singletons, and so the function ϕ created in the last step is well defined. Clearly $\phi(g) \in L(g)$ for all $g \in V(G)$. Let gg' be an edge of G. If $\phi(g)$ or $\phi(g')$ is not h_n, then action $**$ has been applied to g or g' (or both) at some point in application of Algorithm 3.1 to (G, L) and $h_1, \ldots, h_n$, ensuring that $\phi(g)\phi(g')$ is an edge of H. If $\phi(g) = \phi(g') = h_n$, then clearly $\phi(g)\phi(g')$ is an edge of H. Therefore

$(G, L) \xrightarrow{\phi} H$.

□

Recall that for any edge e of a tree T, T_e denotes the tree we obtain from T by subdividing the edge e.

Lemma 3.2. *Let H be a connected chordal graph with perfect elimination ordering $h_1, \ldots, h_n$. Let (T, ℓ) be a tree obstruction on H and let e be an edge of T. Then $(T_e, L_\ell) \rightarrow H$ and Algorithm 3.1 succeeds on (T_e, L_ℓ) and $h_1, \ldots, h_n$ producing a list homomorphism from (T_e, L_ℓ) to H.*

Proof. Since (T, ℓ) is minimally infeasible, (T_e, ℓ) must be a feasible constraint on H. Thus $(T_e, L_\ell) \rightarrow H$. By Theorem 3.3, Algorithm 3.1 succeeds on (T_e, L_ℓ) and $h_1, \ldots, h_n$, producing a list homomorphism from (T_e, L_ℓ) to H.

□

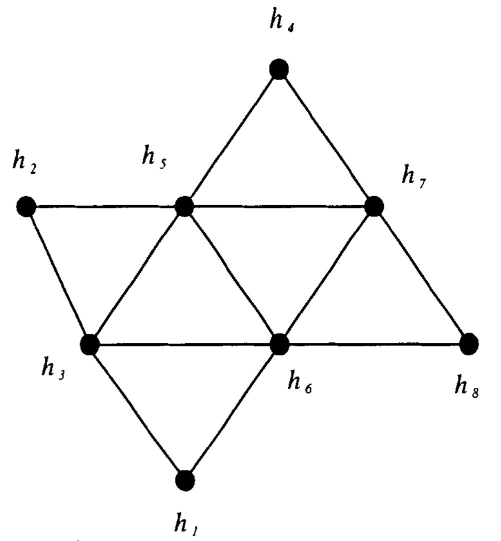
Let H be a connected chordal graph with perfect elimination ordering $h_1, \ldots, h_n$ and let (T, ℓ) be tree constraint on H. The following Lemma is a detailed analysis of Algorithm 3.1 applied to (T, L_ℓ) and the perfect elimination ordering $h_1, \ldots, h_n$.

Lemma 3.3. *Let H be a connected chordal graph and let (T, ℓ) be a tree obstruction on H. Let x be a leaf of T and let $t_0 t_1 \ldots t_m$ be a path in T with $t_0 = x$. Let e_i be the edge $t_i t_{i-1}$ in T and let T_i'' be the component of $T \setminus e_i$ that contains t_i, $i = 1, \ldots, m$. Then we have the following facts:*

i.) There exists a perfect elimination ordering $h_1, \ldots, h_n$ with $h_n = \ell(x)$.

ii.) When we apply Algorithm 3.1 to (T, L_ℓ) and the perfect elimination ordering $h_1, \ldots, h_n$, the algorithm stops when the list of x becomes empty. At this time, the list of each $t \in V(T \setminus x)$ is a singleton $\{\tilde{t}\}$. Moreover, $\ell(x) > \tilde{t}_1 > \ldots > \tilde{t}_m$, with respect to the perfect elimination ordering $h_1, \ldots, h_n$.

iii.) When we apply Algorithm 3.1 to (T_{e_i}, L_ℓ) and the perfect elimination ordering $h_1, \ldots, h_n$, the algorithm succeeds and produces a list homomorphism ϕ_{e_i}, $(T_{e_i}, L_\ell) \xrightarrow{\phi_{e_i}} H$, such that $\phi_{e_i}(t) = \tilde{t}$ for $t \in V(T_i'')$.



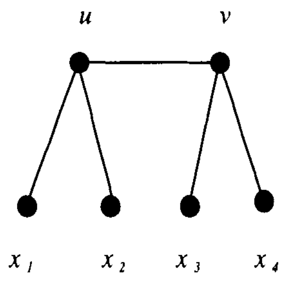


Figure 3.2: Let H be the graph on the left. Let T be the tree on the right, with leaf-labeling ℓ, where $\ell(x_1) = h_1$, $\ell(x_2) = h_2$, $\ell(x_3) = h_4$ and $\ell(x_4) = h_8$.

Before presenting the proof of Lemma 3.3, we will present an example to illustrate each statement of the lemma. Let H and (T, ℓ) be as defined in the caption of Figure 3.2. Let x_4 of T be the chosen leaf x from Lemma 3.3, and let x_4vux_1 be the path $t_0t_1 \dots t_m$. The ordering $h_1, \dots, h_8$ of the vertices of H shown in Figure 3.2 is a perfect elimination ordering of H. Thus there exists a perfect elimination ordering of H such that $\ell(x_4)$ is the last vertex in the ordering.

In Table 3.1, we show the lists of the vertices of T after each iteration when Algorithm 3.1 is applied to (T, L_ℓ) and $h_1, \dots, h_n$. The $*$ ($**$) in the upper right corner indicates that action $*$ ($**$) has been performed on the vertex in that iteration. We see that the list of x_4 becomes empty after action $**$ is performed on its only nontrivial neighbour, the vertex v. Thus Algorithm 3.1 applied to $h_1, \dots, h_8$ and (T, L_ℓ) stops when the list of x_4, our chosen leaf of T, becomes empty. Also, at this time the list of each vertex of T other than x_4 is a singleton. Moreover, $\ell(x_4) > \tilde{v} > \tilde{u} > \tilde{x_1}$ with respect to the perfect elimination ordering $h_1, \dots, h_8$.

Let e be the edge vx_4 in T. Let w be the new vertex in the tree T_e; thus vwx_4 is the path in T_e that replaces the edge e in T. In Table 3.2, we show the lists of

it.	$L(x_1)$	$L(x_2)$	$L(u)$	$L(x_3)$	$L(v)$	$L(x_4)$
0	$\{h_1\}$	$\{h_2\}$	$V(H)$	$\{h_4\}$	$V(H)$	$\{h_8\}$
1	$\{h_1\}^{**}$	$\{h_2\}$	$\{h_3, h_4\}^*$	$\{h_4\}$	$\{h_2, h_3, \dots, h_8\}^*$	$\{h_8\}$
2	$\{h_1\}$	$\{h_2\}^{**}$	$\{h_3\}$	$\{h_4\}$	$\{h_3, h_4 \dots, h_8\}^*$	$\{h_8\}$
3	$\{h_1\}$	$\{h_2\}$	$\{h_3\}^{**}$	$\{h_4\}$	$\{h_5, h_6\}^*$	$\{h_8\}$
4	$\{h_1\}$	$\{h_2\}$	$\{h_3\}$	$\{h_4\}^{**}$	$\{h_5\}$	$\{h_8\}$
5	$\{h_1\}$	$\{h_2\}$	$\{h_3\}$	$\{h_4\}$	$\{h_5\}^{**}$	$\emptyset$

Table 3.1: The lists of vertices of T at each iteration of Algorithm 3.1 applied to (T, L_ℓ) and $h_1, \dots, h_n$.

it.	$L(x_1)$	$L(x_2)$	$L(u)$	$L(x_3)$	$L(v)$	$L(w)$	$L(x_4)$
0	$\{h_1\}$	$\{h_2\}$	$V(H)$	$\{h_4\}$	$V(H)$	$V(H)$	$\{h_8\}$
1	$\{h_1\}^{**}$	$\{h_2\}$	$\{h_3, h_4\}^*$	$\{h_4\}$	$\{h_2, h_3, \dots, h_8\}^*$	$\{h_2, h_3, \dots, h_8\}^*$	$\{h_8\}$
2	$\{h_1\}$	$\{h_2\}^{**}$	$\{h_3\}$	$\{h_4\}$	$\{h_3, h_4 \dots, h_8\}^*$	$\{h_3, h_4 \dots, h_8\}^*$	$\{h_8\}$
3	$\{h_1\}$	$\{h_2\}$	$\{h_3\}^{**}$	$\{h_4\}$	$\{h_5, h_6\}^*$	$\{h_4, h_5, \dots, h_8\}^*$	$\{h_8\}$
4	$\{h_1\}$	$\{h_2\}$	$\{h_3\}$	$\{h_4\}^{**}$	$\{h_5\}$	$\{h_5, h_6, h_7, h_8\}^*$	$\{h_8\}$
5	$\{h_1\}$	$\{h_2\}$	$\{h_3\}$	$\{h_4\}$	$\{h_5\}^{**}$	$\{h_6, h_7\}^*$	$\{h_8\}$
6	$\{h_1\}$	$\{h_2\}$	$\{h_3\}$	$\{h_4\}$	$\{h_5\}^{**}$	$\{h_7\}^*$	$\{h_8\}$
7	$\{h_1\}$	$\{h_2\}$	$\{h_3\}$	$\{h_4\}$	$\{h_5\}^{**}$	$\{h_7\}^{**}$	$\{h_8\}$

Table 3.2: The lists of vertices of T_e at each iteration of Algorithm 3.1 applied to (T_e, ℓ) and $h_1, \dots, h_n$.

the vertices of T_e after each iteration when Algorithm 3.1 is applied to (Te, L_ℓ) and $h_1, \dots, h_n$. Let ϕ_e be the list homomorphism produced by Algorithm 3.1. Let T'' be the component of $T \setminus e$ that contains v; in this case, $T'' = T \setminus x_4$. By looking at the last row of each table, we see that $\phi_e(t)$ is indeed the vertex $\tilde{t}$ for each vertex $t \in V(T'')$.

Proof. The set $\{\ell(x)\}$ is convex in H. Therefore, by Theorem 2.3, there exists a partial perfect elimination ordering $h_1, \dots, h_{n-1}$ such that $\{\ell(x)\} = V(H) \setminus \{h_1, \dots, h_{n-1}\}$. If we set $h_n = \ell(x)$, then we have the desired perfect elimination ordering. For the rest of this proof, all applications of Algorithm 3.1 will use the perfect elimination ordering $h_1, \dots, h_n$ just described.

Since (T, ℓ) is a tree obstruction on H, $(T, L_\ell) \not\to H$ and so by Theorem 3.3, we know that Algorithm 3.1 fails on (T, L_ℓ). Now we need to prove that Algorithm 3.1 fails on (T, L_ℓ) when the list of x becomes empty.

Apply Algorithm 3.1 to (T, L_ℓ) and consider the actions taken on the lists of t_1 and x. As the initial list of x is $\{\ell(x)\}$ and $\ell(x) = h_n$, we do not process the vertex $\ell(x)$. Therefore the list of x has no effect on the list of t_1. Suppose that the list of t_1 has no effect on the list of x; i.e., even if we perform action $**$ on the list of t_1, no vertex is lost from the list of x. Therefore, as t_1 is the only nontrivial neighbour of x, when Algorithm 3.1 fails on (T, L_ℓ), the list of some vertex t', $t' \neq x$, becomes empty. Moreover, the list of t' will still become empty when Algorithm 3.1 is applied to $(T \setminus x, L)$, where $L(t) = L_\ell(t)$ for all $t \in V(T \setminus x)$. As $T \setminus x \subset T_{e_1}$ and $L(t) = L_\ell(t)$ for all $t \in V(T \setminus x)$, this implies that Algorithm 3.1 fails on (T_{e_1}, L_ℓ). This contradicts Lemma 3.2. Therefore when Algorithm 3.1 is applied to (T, L_ℓ), the list of t_1 does effect the list of x. Hence the list of t_1 must become a singleton say $\{\tilde{t}_1\}$, and when $\tilde{t}_1$ is processed, action $**$ is performed on the list of t_1, causing the list of x to loose a vertex; thus $\tilde{t}_1 < \ell(x)$ also. As the list of x was originally $\{\ell(x)\}$, this implies that the list of x becomes empty. Therefore when Algorithm 3.1 fails on (T, L_ℓ), it is because the list of x becomes empty.

Let t'_k be the new vertex added when we subdivide the edge e_k of T to form T_{e_k}. Thus the tree T_{e_k} contains the path $t_k t'_k t_{k-1}$.

By Lemma 3.2 we know that Algorithm 3.1 applied to (T_{e_1}, L_ℓ) produces a list homomorphism, say ϕ_{e_1}, where $(T_{e_1}, L_\ell) \xrightarrow{\phi_{e_1}} H$. Thus when Algorithm 3.1 stops on (T_{e_1}, L_ℓ), each vertex t of T has the singleton list $\{\phi_{e_1}(t)\}$. Consider the tree $T \setminus x = T''_1$; it is a subgraph of both T and T_{e_1}. If we can prove that the final lists for the vertices of $T \setminus x$ are the same when whether we apply Algorithm 3.1 to (T, L_ℓ) or to (T_{e_1}, L_ℓ), then we have proved that when Algorithm 3.1 stops on (T, L_ℓ), the list of t, $t \neq x$, is a singleton $\{\tilde{t}\}$ and more over that $\tilde{t} = \phi_{e_1}(t)$ for $t \neq x$.

Since $\ell(x) = h_n$, the vertex $\ell(x)$ is not processed when we apply Algorithm 3.1 to (T, L_ℓ) or to (T_{e_1}, L_ℓ). Therefore the list of x doesn't effect the list of t_1 when Algorithm 3.1 is applied to (T, L_ℓ) and the list of x doesn't effect the list of t'_1 when

Algorithm 3.1 is applied to (T_{e_1}, L_ℓ). As t_1 and $t_0 = x$ are the only nontrivial neighbours of t'_1 in T_{e_1}, the list of t'_1 can't become a singleton before the list of t_1 does, and so the list of t'_1 has no effect on the list of t_1 when Algorithm 3.1 is applied to (T_{e_1}, L_ℓ). Therefore the vertices of $T \setminus x$ must have the same final lists whether we apply Algorithm 3.1 to (T, L_ℓ) or to (T_{e_1}, L_ℓ).

We have already observed that $\tilde{t}_1 < \ell(x)$ and that the list of t_1 causes the removal of at least one vertex from the list of t_0 via action $**$ when Algorithm 3.1 is applied to (T, L_ℓ). Therefore we may assume that the following is true for $k - 1 \geq 1$:

- when we apply Algorithm 3.1 to (T, L_ℓ), the algorithm stops when the list of x becomes empty. At this time, the list of $t \in V(T \setminus x)$ is a singleton $\{\tilde{t}\}$. In fact, when the vertex $\tilde{t}_i$ was processed by the algorithm, action $**$ was applied to the list of t_i, causing at least one vertex to be removed from the list of t_{i-1}, $i = 1, \ldots, k-1$. Moreover, $\ell(x) > \tilde{t}_1 > \ldots > \tilde{t}_{k-1}$, with respect to the perfect elimination ordering $h_1, \ldots, h_n$.
- when we apply Algorithm 3.1 to (T_{e_i}, L_ℓ), list homomorphism ϕ_{e_i} is produced and $\phi_{e_i}(t) = \tilde{t}$ for $t \in V(T''_i)$ and $1 \leq i \leq k-1$.

We will now prove that the above statements hold for k.

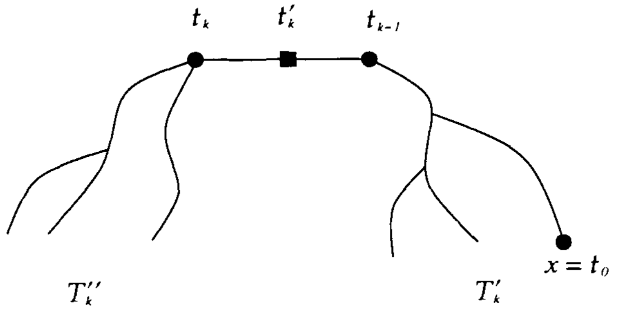


Figure 3.3: The tree T_{e_k}, and the subtrees T''_k and T'_k.

Consider the vertex t_k. Suppose that the list of t_k doesn't effect the list of t_{k-1} via action $**$ when Algorithm 3.1 is applied to (T, L_ℓ). As the vertex t_{k-1} is second vertex on the only $t_k - x$ path in T, we can remove the edge e_k and still have the list of x become empty. In other words, when we apply Algorithm 3.1 to $(T \setminus e_k, L_\ell)$, the list of x will still become empty. Let T_k' be the component of $T \setminus e_k$ that contains x and let L' be the restriction of L to the vertices of $V(T_k')$. Then Algorithm 3.1 will fail on (T_k', L'). Note that T_k' is a subtree of T_{e_k}. Therefore Algorithm 3.1 must fail on (T_{e_k}, L_ℓ), contradicting Lemma 3.2. Therefore action $**$ will be performed on the list of t_k when Algorithm 3.1 is applied to (T, ℓ), causing the removal of at least one vertex from the list of t_{k-1}.

Now we need to prove that $\tilde{t}_k < \tilde{t}_{k-1}$ with respect to the perfect elimination ordering $h_1, \ldots, h_n$; recall that $\tilde{t}_k$, and $\tilde{t}_{k-1}$, denote the single vertex in the list of t_k, t_{k-1} respectively, when Algorithm 3.1 fails on (T, ℓ). Let $L(t)$ be the list of $t \in V(H)$ just before the vertex $\tilde{t}_k$ is processed. Therefore $L(t_k) = \{\tilde{t}_k\}$ and there exists a vertex h in $L(t_{k-1})$ such that h is removed from $L(t_{k-1})$ when action $**$ is performed on $L(t_k)$; thus h is not adjacent to $\tilde{t}_k$. As the list of t_{k-1} will eventually become $\{\tilde{t}_{k-1}\}$, we know that $\tilde{t}_{k-1} \in L(t_{k-1})$. In fact $h \neq \tilde{t}_{k-1}$ as action $**$ ensures that $\tilde{t}_k$ and $\tilde{t}_{k-1}$ are adjacent. Therefore $|L(t_{k-1})| \geq 2$. By our comments following Algorithm 3.1, just before we process the vertex $\tilde{t}_k$, all lists of size 2 or more may not contain vertices preceding $\tilde{t}_k$ in the perfect elimination ordering $h_1, \ldots, h_n$. Therefore $\tilde{t}_k \leq \tilde{t}_{k-1}$. It remains to prove that $\tilde{t}_k \neq \tilde{t}_{k-1}$. This is clearly true if $\tilde{t}_k \notin L(t_{k-1})$. Therefore assume that $\tilde{t}_k \in L(t_{k-1})$. As noted in the proof of Theorem 3.3, lists remain connected through the running of Algorithm 3.1. Hence $L(t_{k-1})$ is connected and so there exists a neighbour h' of $\tilde{t}_k$ in $L(t_{k-1})$ such that h' is on a path from $\tilde{t}_k$ to h in H. Thus, even after removing the non-neighbours of $\tilde{t}_k$ from $L(t_{k-1})$, the resulting list contains the distinct vertices $\tilde{t}_k$ and h'. Hence when processing the vertex $\tilde{t}_k$, the vertex $\tilde{t}_k$ will be removed from the list of t_{k-1} via action $*$. Therefore $\tilde{t}_k \neq \tilde{t}_{k-1}$, and so $\tilde{t}_k < \tilde{t}_{k-1}$ with respect to the perfect elimination ordering $h_1, \ldots, h_n$.

Lastly, we need to prove that $\phi_{e_k}(t) = \tilde{t}$ for $t \in V(T_{e_k}'')$, where $(T_{e_k}, L_\ell) \overset{\phi_{e_k}}{\rightarrow} H$.

Observe that T_k'' is a subgraph of both T and T_{e_k}. If we can prove that the final lists of the vertices in $V(T_k'')$ are the same whether we apply Algorithm 3.1 to (T, L_ℓ) or to (T_{e_k}, L_ℓ), then $\phi_{e_k}(t) = \tilde{t}$ for $t \in V(T_k'')$, where $(T_{e_k}, L_\ell) \overset{\phi_{e_k}}{\rightarrow} H$.

By previous work, we know that when we apply Algorithm 3.1 to (T, L_ℓ), the action $**$ is performed on the list of t_k causing the removal of at least one vertex from the list of t_{k-1} at that time. Hence in this instance, the list of t_{k-1} has no effect on the list of t_k; when and if action $**$ is performed on the list of t_{k-1}, the list of t_k is already a singleton and as $t_k \neq x$, the list of t_k doesn't become empty.

Now consider what happens when we apply Algorithm 3.1 to (T_{e_k}, L_ℓ). The edge e_k of T has been replace by the path $t_k t'_k t_{k-1}$ and the vertex t'_k is assigned the list $V(H)$. Everything else is the same as (T, L_ℓ). The only way the list of t'_k could effect the list of t_k is if the list of t'_k became a singleton before the list of t_k did. This could only happen if action $**$ was performed on the list of t_{k-1} causing as least one vertex to be removed from the list of t'_k. In other words, the list of t_{k-1} must be come a singleton before the list of t'_k does and hence before the list of t_k does. Yet this would imply that the list of t_{k-1} becomes a singleton before the list of t_k becomes a singleton when we apply Algorithm 3.1 to (T, L_ℓ), which is a contradiction. Hence the list of t'_k doesn't effect the list of t_k when Algorithm 3.1 is applied to (T_{e_k}, L_ℓ). Therefore the final lists of the vertices in $V(T''_k)$ are the same whether we apply Algorithm 3.1 to (T, L_ℓ) or to (T_{e_k}, L_ℓ)

$\square$

Given a tree T, and two distinct vertices $x, y \in V(T)$, let $\rho_T(x, y)$ be the number of branching points internal to the unique path between x and y in T; if x or y is a branching point, then they are not counted by $\rho_T(x, y)$. If the tree is clear from the context, we omit the subscript.

Let H be a connected graph. Let (T, ℓ) be a tree obstruction of H and let x and y be distinct leaves of T. If T is a path, then $d_H(\ell(x), \ell(y)) = d_T(x, y) + 1$ by Proposition 3.2. On the other hand, if T is not a path, all we know is $d_H(\ell(x), \ell(y)) \leq d_T(x, y)$, again by Proposition 3.2. If H is a chordal graph and T has one branching point, then by Proposition 3.3 and Theorem 2.5, we have that $d_H(\ell(x), \ell(y)) = d_T(x, y)$. In other words, the hole corresponding to (T, ℓ) is not squished. Unfortunately, this tightness in the upper bound for $d_H(\ell(x), \ell(y))$ is not maintained when we consider trees T with more than one branching point, as we demonstrate below.

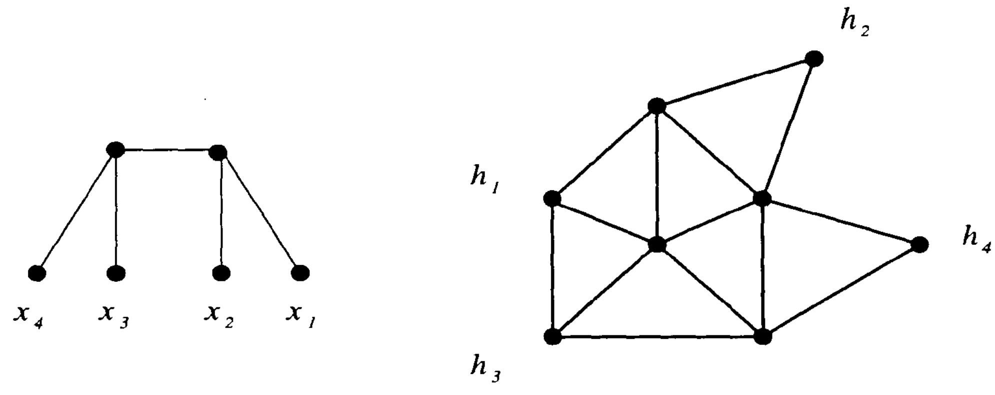


Figure 3.4: On the left is a leaf-labeled tree, (T, ℓ), where $\ell(x_i) = h_i$ for $i = 1, \ldots, 4$. This leaf-labeled tree is a squished tree obstruction for the graph on the right; the vertices h_1 and h_3 are too close together.

Let H be the graph on the right in Figure 3.4 and let $H' = H \setminus h_1h_3$. Observe that H is isomorphic to the graph in Figure 2.7 and that H' is isomorphic to the graph in Figure 2.6. Let (T, ℓ) be the tree obstruction in Figure 3.4, where $\ell(x_i) = h_i$, $i = 1, 2, 3, 4$. While $d_{H'}(\ell(x_1), \ell(x_2)) = d_T(x_1, x_2)$ and $d_{H'}(\ell(x_1), \ell(x_4)) = d_T(x_1, x_4)$, we have $d_{H'}(\ell(x_1), \ell(x_3)) = 2 < 3 = d_T(x_1, x_3)$. However, it is true that $d_{H'}(\ell(x_1), \ell(x_i)) \geq d_T(x_1, x_i) - \rho_T(x_1, x_i) + 1$ for $i = 2, 3, 4$. Note H' is chordal, and this distance property is violated by the non-chordal graph H; $d_H(\ell(x_1), \ell(x_3)) = 1 < 2 = d_T(x_1, x_3) - \rho_T(x_1, x_3) + 1$. Thus if we wish to define squished tree obstructions such that chordal graphs avoid them, we must take the number of branching points between leaves into consideration.

Let H be a connected graph and let (T, ℓ) be a tree obstruction on H. Let x and y be distinct leaves of T. Then x and y are called a *squished pair* if $d_H(\ell(x), \ell(y)) \leq d_T(x, y) - \rho_T(x, y)$. We call (T, ℓ) *squished* if it has a squished pair. Note that if T is a path, then (T, ℓ) can't be squished as $d_H(\ell(x), \ell(y)) = d_T(x, y) + 1$ by Proposition 3.2 and $d_T(x, y) - \rho_T(x, y) = d_T(x, y)$. A connected graph that has no squished tree obstructions is called *strongly stretched.* If f is a squished hole on H, then observe that the tree obstruction associated with f that we created in Proposition 3.3 is also squished. Therefore all strongly stretched graphs are stretched and so Theorem 2.5

is a special case of Theorem 3.4.

Theorem 3.4. *Each connected chordal graph is strongly stretched.*

Proof. Suppose that H does have a squished tree obstruction and let (T, ℓ) be a **minimum** squished tree obstruction on H in the sense that there does not exist another squished tree obstruction (T', ℓ') on H such that $|V(T')| < |V(T)|$. Pick a squished pair of leaves x and y of T. We know that $\rho_T(x, y) \geq 1$ as T can't be a path, by our comments following the definition of a squished tree obstruction.

By Lemma 3.3, there exists a perfect elimination ordering $h_1, \ldots, h_n$ of H such that $h_n = x$. For the rest of this proof, all applications of Algorithm 3.1 will use this particular perfect elimination ordering.

Let $t_m, t_{m-1}, \ldots, t_1, t_0$ be the path in T with $t_m = y$ and $t_0 = x$ and let e_i be the edge $t_i t_{i-1}$.

Apply Algorithm 3.1 to (T, L_ℓ). Again by Lemma 3.3, we know that the algorithm fails because the list of x becomes empty and also that at this time, for all vertices $t \in V(T) \setminus x$, list of t is a singleton $\{\tilde{t}\}$. Let $z = t_{m-1}$, the nontrivial neighbour of y in T. Let $T' = T \setminus \{t \mid z \text{ is an internal vertex of the unique } t - x \text{ path in } T\}$. Note that z is a leaf of T'. Define a leaf-labeling ℓ' on T' as follows:

$$\ell'(t) = \begin{cases} \tilde{z} & \text{if } t = z \\ \ell(t) & \text{otherwise} \end{cases}$$

The function ℓ' is well defined as z is the only leaf of T' that isn't also a leaf of T. Thus (T', ℓ') is certainly a tree constraint on H. If we can prove that (T', ℓ') is a squished tree obstruction on H, then we have a contradiction to the way we chose (T, ℓ) as $|V(T')| \leq |V(T) \setminus y| < |V(T)|$. This would then imply that H can not have a squished tree obstruction.

We will first prove that (T', ℓ') is a tree obstruction and then secondly prove that (T', ℓ') is squished.

To prove that (T', ℓ') is a tree obstruction, we must prove that (T', ℓ') is infeasible and moreover that (T', ℓ') is minimally infeasible.

Suppose that (T', ℓ') is feasible on H. Then we have $(T', L_{\ell'}) \xrightarrow{\phi'} H$. Note that in particular, $\phi'(z) = \ell'(z) = \tilde{z}$. Let e be the edge e_{m-1}. Since (T, ℓ) is a tree obstruction on H, (T_e, ℓ) must be feasible on H. Let ϕ_e be the function produced by Algorithm 3.1 applied to (T_e, L_ℓ). By Theorem 3.1, $(T_e, L_\ell) \xrightarrow{\phi_e} H$ and by Lemma 3.3, $\phi_e(z) = \tilde{z}$. Define a function $\phi : T \to H$ as follows:

$$\phi(t) = \begin{cases} \phi'(t) & \text{if } t \in V(T') \\ \phi_e(t) & \text{otherwise.} \end{cases}$$

The function ϕ is a homomorphism as ϕ' and ϕ_e are both homomorphisms and as $\phi'(z) = \tilde{z} = \phi_e(z)$. Moreover, $\phi : T \to H$ is an extension of ℓ. This contradicts the infeasibility of (T, ℓ) on H. Hence (T', ℓ') is infeasible on H.

Our next task is to prove that (T', ℓ') is minimally infeasible. Specifically, we need to prove that (T'', ℓ'') is a feasible tree constraint on H for all tree constraints (T'', ℓ'') on H such that $(T'', \ell'') < (T', \ell')$. By Proposition 3.1, we need only prove that (T'_e, ℓ') is feasible for each edge e of T'. Let e be a particular edge of T'. Then, since T' is a subgraph of T, e is also an edge of T. As (T, ℓ) is a tree obstruction on H, (T_e, ℓ) is a feasible tree constraint on H. Apply Algorithm 3.1 to (T_e, L_ℓ). By Theorem 3.3, the algorithm succeeds, and the function it produces, call it ϕ_e, is a list homomorphism from (T_e, L_ℓ) to H. There exists a path $p_j p_{j-1} \ldots p_0$ in T, where $p_0 = x$ and $p_j p_{j-1} = e$. Since z is a leaf of T', and since $e \in E(T')$, we have that $z \neq p_i$, $i = 0, 1, \ldots, j-1$. Therefore, by Lemma 3.3, $\phi_e(z) = \tilde{z}$. Note that T'_e is a subtree of T_e. Therefore ϕ_e is defined on each vertex of T'_e and moreover, if we let ϕ' be the restriction of ϕ_e to T'_e, then $\phi' : T'_e \to H$ is an extension of ℓ'. Therefore (T'_e, ℓ') is a feasible tree constraint on H and so (T', ℓ') is a tree obstruction on H.

Now we need to prove that (T', ℓ') is squished. In particular, we will prove that

$$d_H(\ell'(z), \ell'(x)) \leq d_{T'}(z, x) - \rho_{T'}(z, x).$$

Since $\ell'(z) = \tilde{z}$ and $\ell'(x) = \ell(x)$, this is equivalent to proving that

$$d_H(\tilde{z}, \ell(x)) \leq d_{T'}(z, x) - \rho_{T'}(z, x).$$

To do this, we will rely on the properties of perfect elimination orderings and Lemma 3.3.

By Lemma 3.3, there exists a list homomorphism ϕ_{e_1} from (T_{e_1}, L_ℓ) to H such that $\phi_{e_1}(t_i) = \tilde{t}_i$, $i = 1, \dots, m$, where $\{\tilde{t}\}$ is the final list of $t \in V(T \setminus x)$ when Algorithm 3.1 is applied to (T, ℓ). As ϕ_{e_1} is a homomorphism, $\tilde{t}_m \dots \tilde{t}_1$ is a walk in H, where $\tilde{t}_m = \ell(y)$. In fact, $\tilde{t}_m \dots \tilde{t}_1$ is a path that is strictly increasing with respect to the perfect elimination ordering $h_1, \dots, h_n$ as $\ell(x) > \tilde{t}_1 > \dots > \tilde{t}_m = \ell(y)$, also by Lemma 3.3. Denote this path $\tilde{t}_m \dots \tilde{t}_1$ by P.

Let Q be a shortest, and hence chordless, path from $\ell(y)$ to $\ell(x)$ in H. As x and y are a squished pair of the tree obstruction (T, ℓ), the length of Q is bounded above by $d_T(x, y) - \rho_T(x, y)$. Since Q is a shortest path that ends at $h_n = \ell(x)$, the vertices of Q are strictly increasing with respect to the perfect elimination ordering $h_1, \dots, h_n = \ell(x)$. Let h be the nontrivial neighbour of $\ell(y)$ on Q.

As P and Q are strictly increasing paths in H, h and $\tilde{z}$ are both neighbours of $\ell(y)$ that occur after $\ell(y)$ in the perfect elimination ordering $h_1, \dots, h_n$. Therefore h and $\tilde{z}$ are adjacent in H. Since the length of Q is bounded above by $d_T(x, y) - \rho_T(x, y)$, we have that $d_H(h, \ell(x)) \leq d_T(x, y) - \rho_T(x, y) - 1$, and so

$$d_H(\tilde{z}, \ell(x)) \leq d_T(x, y) - \rho_T(x, y). \tag{3.1}$$

As $d_{T'}(z, x) = d_T(x, y) - 1$, we in fact have

$$d_H(\tilde{z}, \ell(x)) \leq d_{T'}(z, x) + 1 - \rho_T(x, y). \tag{3.2}$$

Note that $\rho_{T'}(z, x) = \rho_T(x, y)$ if z is not a branching point of T and $\rho_{T'}(z, x) = \rho_T(x, y) - 1$ if z is a branching point of T.

Assume that z is a branching point in T. Thus $\rho_{T'}(z, x) = \rho_T(x, y) - 1$ and so inequality 3.2 becomes

$$d_H(\tilde{z}, \ell(x)) \leq d_{T'}(z, x) - \rho_T(z, x). \tag{3.3}$$

Therefore (T', ℓ') is squished if T' is not a path.

Suppose that T' is a path. We have already proved that (T', ℓ') is a tree obstruction. Then by Proposition 3.2,

$$d_H(\ell'(x), \ell'(z)) = d_{T'}(x, z) + 1. \tag{3.4}$$

As $\ell'(z) = \tilde{z}$, $\ell'(x) = \ell(x)$ and $\rho_{T'}(z,x) = 0$, inequality 3.3 becomes

$$d_H(\ell'(z), \ell'(x)) \leq d_{T'}(z,x),$$

contradicting equation 3.4. Therefore T' can't be a path and so (T', ℓ') is a squished tree obstruction.

Now assume that z is not a branching point of T. Then inequality 3.2 becomes

$$d_H(\tilde{z}, \ell(x)) \leq d_{T'}(z,x) + 1 - \rho_{T'}(z,x). \tag{3.5}$$

To prove that z and x are a squished pair, we must reduce this upper bound by 1.

Since z is not a branching point of T, its only nontrivial neighbours in T are y and t_{m-2}. By the proof of Lemma 3.3, we know that when we apply Algorithm 3.1 to (T, L_ℓ), the list of y will cause a vertex to be removed from the list of z via action $**$. As $\tilde{z} = \tilde{t}_{m-1} < \tilde{t}_{m-2}$, the list of t_{m-2} will not cause the removal of a vertex from the list of z. Hence, the list of y is the only list that will effect the list of z when Algorithm 3.1 is applied to (T, L_ℓ). Recall that h is the nontrivial neighbour of $\ell(y)$ on Q. If $\tilde{z} < h$ with respect to the perfect elimination ordering $h_1, \ldots, h_n$, then Algorithm 3.1 applied to (T, L_ℓ) would have removed the vertex $\tilde{z}$ from the list of z via action $*$, hence we may assume that $h \leq \tilde{z}$. Let h' be the nontrivial neighbour of h on Q other than $\ell(y)$. Then, by the definition of a perfect elimination ordering, we have that $\tilde{z}$ and h' are adjacent as they are neighbours of h that occur after h in the perfect elimination ordering $h_1, \ldots, h_n$. Since the length of Q is bounded above by $d_T(x,y) - \rho_T(x,y)$, we have that $d_H(h', \ell(x)) \leq d_T(x,y) - \rho(x,y) - 2$, and so

$$d_H(\tilde{z}, \ell(x)) \leq d_T(x,y) - \rho(x,y) - 1. \tag{3.6}$$

Now when we make the substitutions $d_T(x,y) = d_{T'}(z,x) + 1$ and $\rho_T(x,y) = \rho_{T'}(x,z)$ in inequality 3.1, we can conclude

$$d_H(\tilde{z}, \ell(x)) \leq d_{T'}(z,x) - \rho_{T'}(z,x). \tag{3.7}$$

Therefore (T', ℓ') is a squished tree obstruction on H.

Hence connected chordal graphs are strongly stretched.

□

Let H be a connected chordal graph, and let (T, ℓ) be a tree obstruction on H, where T is not a path. Let x and y be leaves of T. As leaves can't be branching points, $d_T(x, y) - \rho(x, y) + 1 \geq 2$, and so the bound tells us that $\ell(x)$ and $\ell(y)$ can't be adjacent in H.

We analyzed holes on connected chordal graphs using convexity in Section 2.2. Using this technique, we were able to prove that a non-degenerate hole f on connected chordal graph H implied the existence of particular isometric subgraph J in H; we called this graph the base of f in H. This graph J was in fact a connected chordal graph such that f was also a hole on J and each vertex of J was necessary; if any vertex of J was removed, f was no longer a hole on J. In some sense, there was one base for all holes with the same values on all connected chordal graphs. For example, if f is a hole on a connected chordal graph H, where $D_f = \{x_1, x_2, x_3\}$ and $f(x_i) = i$ for $i = 1, 2, 3$, then the base admitted by f in H is isomorphic to the graph in Figure 3.5. By Proposition 3.3, we know that for any connected graph H and

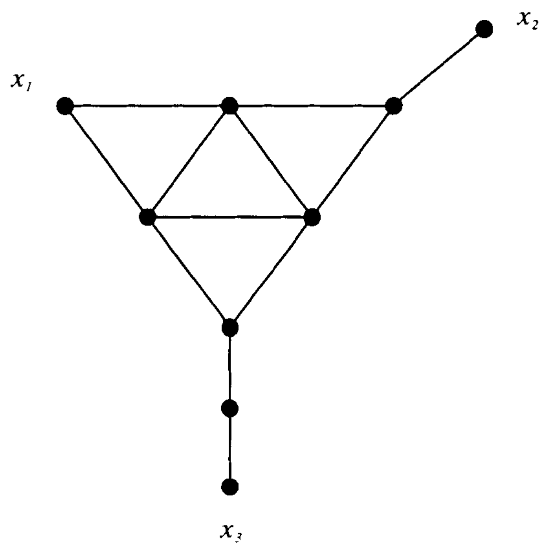


Figure 3.5: 'The' base for any 3-hole with values 1, 2, 3 on any connected chordal graph.

for each tree obstruction (T, ℓ) where T has at most one branching point and all the leaf-labels on T are distinct, there exits a corresponding hole f on H. Assume that H is chordal. Then as H is strongly stretched by Theorem 3.4, all tree obstructions on H must have distinct labels. Therefore if (T, ℓ) is a tree obstruction on H where T has one branching point, then there exits a corresponding hole f on H which admits a base by Theorem 2.6.

However, if H is a connected chordal graph and (T, ℓ) is a tree obstruction on H, where T has at least two branching points, we can't guarantee the existence of any such subgraph in H. Let H' be the chordal graph in Figure 3.6, let H'' be the chordal graph in Figure 3.7 and let T be the tree in Figure 3.8. Then (T, ℓ') is a tree obstruction on H', and (T, ℓ'') is a tree obstruction on H'', if we set $\ell'(x_i) = w_i$ and $\ell''(x_i) = v_i$, for $i = 1, 2, 3, 4$.

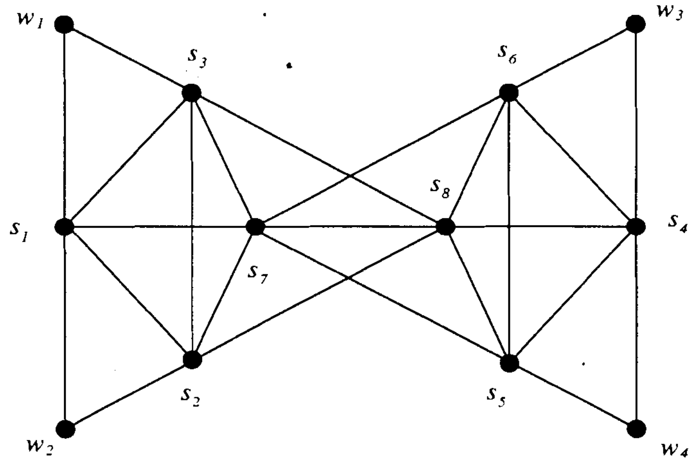


Figure 3.6: A chordal graph that is a potential tree obstruction base.

Notice that $d_{H'}(w_i, w_j) = d_{H''}(v_i, v_j)$ for $1 \leq i, j \leq 4$ and that each vertex of H' and each vertex of H'' is necessary in the following sense: if any vertex or edge of H' is removed, then (T, ℓ') is not a tree obstruction for the resulting graph. Similarly for H'' and (T, ℓ''). Thus there can't exist a (chordal) graph J with the following property: J is isomorphic to a subgraph J' of H' and to a subgraph J'' of H'' where (T, ℓ') is a tree obstruction on J' and (T, ℓ'') is a tree obstruction on J''.

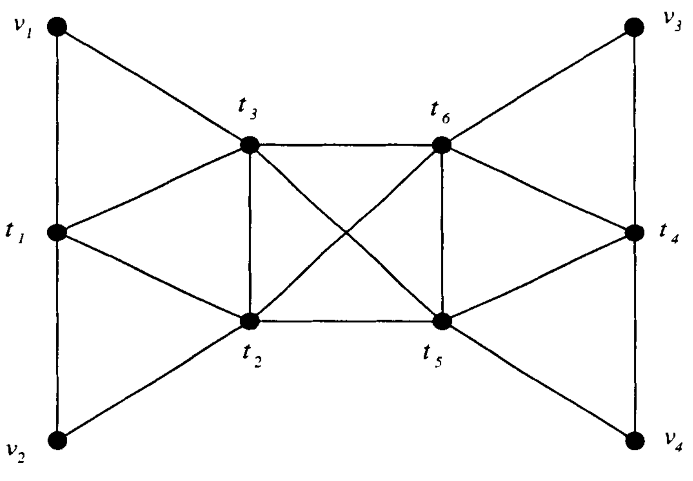


Figure 3.7: A chordal graph that is a potential tree obstruction base.

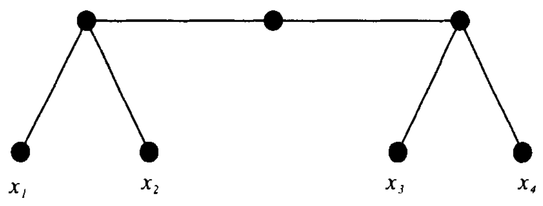


Figure 3.8:

Let γ be the homomorphism from H' to H'' defined by

$$\gamma(h) = \begin{cases} v_i & \text{if } h = w_i, i = 1, 2, 3, 4 \\ t_i & \text{if } h = s_i, i = 1, \ldots, 6 \\ t_3 & \text{if } h = s_7 \\ t_6 & \text{if } h = s_8. \end{cases}$$

The homomorphism γ 'preserves' the tree obstruction (T, ℓ') the sense that $(T, \gamma \circ \ell')$ is a tree constraint on H''. In fact, $(T, \gamma \circ \ell')$ is the tree obstruction (T, ℓ'') on H''.

Therefore, if (T, ℓ) is a tree obstruction on a connected chordal graph H, we can't guarantee the existence of a particular subgraph in H. Instead, we hope to guarantee the existence of a particular graph J **and** a particular homomorphism γ, $H \xrightarrow{\gamma} J$, such that J is chordal and γ 'preserves' the tree obstruction (T, ℓ). This is an area of future research.

3.4 Strongly stretched graphs and the variety generated by chordal graphs

In Section 2.3, we used stretched graphs to study the variety generated by connected chordal graphs. Now, we will used strongly stretched graphs in the same manner. We will prove that the variety generated by connected chordal graphs is a subset of the class of strongly stretched graphs and make a conjecture concerning strongly stretched graphs and the graphs that are in both $\mathcal{AR}_o$ and the variety generated by chordal graphs.

Theorem 3.5. *The class of strongly stretched graphs is a variety.*

Proof. Let H be graph that is a retract of a graph H', where H' is strongly stretched. Obviously, H must also be strongly stretched.

Let $H = H_1 \times H_2$, where H_i is strongly stretched, $i = 1, 2$. Suppose there exists a tree obstruction (T, ℓ) on H, where the tree obstruction has a squished pair x and y. Note that (T, ℓ_1) is a tree constraint on H_1, where $\ell_1(v) = \pi_1(\ell(v))$ for all leaves v of T. Since

$$d_{H_1}(\ell_1(x), \ell_1(y)) \leq d_H(\ell(x), \ell(y)) \leq d_T(x, y) - \rho_T(x, y) + 1,$$

and since H_1 is strongly stretched, (T, ℓ_1) is not a tree obstruction on H_1. Thus either (T, ℓ_1) is feasible or it is not minimally infeasible. Suppose that there exists a tree obstruction (T', ℓ') on H_1 such that $(T', \ell') < (T, \ell_1)$. Now consider the tree constraint (T', ℓ'') on H, where $\ell''(v) = \ell(v)$ for all leaves v of T'; this is well defined as all the leaves of T' are leaves of T by construction. Then since $\pi_i(\ell''(v)) = \ell'(v)$ for all leaves of T', (T', ℓ'') is an infeasible tree constraint on H such that $(T', \ell'') < (T, \ell)$, a contradiction. Hence (T, ℓ_1) is feasible. Similarly, the tree constraint (T, ℓ_2) on H_2 is feasible, where $\ell_2(v) = \pi_2(\ell(v))$ for all leaves v of T. Then it is easy to construct a homomorphism from T to H that extends ℓ, again a contradiction. Therefore H

must be strongly stretched.

□

Corollary 3.1. *The variety generated by connected chordal graphs is a subset of the variety of strongly stretched graphs.*

□

Let H be a connected graph and let (T, ℓ) be a tree obstruction of H. Note that we can regard ℓ as a function from the set of leaves of T to H. A homomorphism γ from H to a connected graph J is called a *preserving map* of (T, ℓ) if $(T, \gamma \circ \ell)$ is an infeasible tree constraint of J.

Lemma 3.4. *Let H be a fixed connected graph. If for each tree obstruction (T, ℓ) on H there exists a connected graph $J_{(T,\ell)}$ and a function $\gamma_{(T,\ell)} : H \longrightarrow J_{(T,\ell)}$ such that $\gamma_{(T,\ell)}$ is a preserving map of (T, ℓ), then H is isomorphic to a subgraph $\hat{H}$ of G, where*

$$G = \Pi \left\{ J_{(T,\ell)} : (T, \ell) \text{ is a tree obstruction of } H \right\},$$

such that each tree obstruction on $\hat{H}$ is a tree obstruction on G.

Proof. This lemma is an extension of Lemma 2.3, and accordingly, the proof is similar.

Note that the vertices of G are vectors that are indexed by the tree obstructions of H. Define the map $\phi : H \longrightarrow G$ by

$$\phi(g) = (\gamma_{(T,\ell)}(g))_{(T,\ell)},$$

for each tree obstruction (T, ℓ). Since each $\gamma_{(T,\ell)}$ is a homomorphism, so is ϕ. Choose a pair of distinct vertices h and h' in H. Let (T', ℓ') be a tree obstruction on H, where T' is a path of length $d_H(h, h') - 1$ with endpoints x and y such that $\ell(x) = h$ and $\ell(y) = h'$. Since $\gamma_{(T',\ell')} : H \longrightarrow J_{(T,\ell)'}$ is a preserving map of (T', ℓ'),

$$d_{J_{(T',\ell')}}(\gamma_{(T',\ell')}(h), \gamma_{(T',\ell')}(h')) \geq d_{T'}(x, y) + 1 = d_H(h, h'),$$

which implies that $d_G(\phi(h), \phi(h')) \geq d_H(h, h')$. Therefore ϕ preserves distances, and so $\hat{H}$ is isomorphic to H.

Now we will prove that each tree obstruction on $\hat{H}$ is a tree obstruction on G.

Suppose that there exists a tree obstruction $(\hat{T}, \hat{\ell})$ of $\hat{H}$ such that $(\hat{T}, \hat{\ell})$ is feasible on G. Thus, there exists homomorphism $\beta : \hat{T} \longrightarrow G$ such that for each leaf t of $\hat{T}$, $\beta(t) = \hat{\ell}(t)$. Let (T, ℓ) be the corresponding tree obstruction on H, i.e., $T = \hat{T}$ and $\hat{\ell} = \phi \circ \ell$. Thus for each leaf t of T, $\beta(t) = \phi(\ell(t))$. Let $\pi_{(T,\ell)} : G \longrightarrow J_{(T,\ell)}$ be the projection onto $J_{(T,\ell)}$. Consider the function $\pi_{(T,\ell)} \circ \beta : T \longrightarrow J_{(T,\ell)}$. This composition is a homomorphism as $\phi_{(T,\ell)}$ and β are each homomorphisms. Moreover, for each leaf t of T, $\pi_{(T,\ell)}(\beta(t)) = \pi_{(T,\ell)}(\phi(\ell(t))) = \gamma_{(T,\ell)}(\ell(t))$. Thus we have created a homomorphism from T to $J_{(T,\ell)}$ that extends $\gamma_{(T,\ell)} \circ \ell$, which is a contradiction to the way we chose $J_{(T,\ell)}$. Therefore $(\hat{T}, \hat{\ell})$ is not feasible on G. As in the proof of Proposition 3.5, $(\hat{T}, \hat{\ell})$ is in fact a tree obstruction on G.

□

Conjecture 1. *In $\mathcal{AR}_O$, the variety of strongly stretched graphs is exactly the variety generated by chordal graphs. In other words, if we let X be the variety generated by connected chordal graphs and let Y be the variety of strongly stretched graphs,*

$$\mathcal{AR}_O \cap X = \mathcal{AR}_O \cap Y.$$

The proposed method of proof for Conjecture 1 is the same method used in the proof of Theorem 2.7. Let H be a given graph in $\mathcal{AR}_O$. If H is in the variety generated by connected chordal graphs, then H is in the variety of strongly stretched graphs by Corollary 3.1. Now assume that H is a strongly stretch graph. By, Lemma 3.4, if for each tree obstruction (T, ℓ) of H we can find a separating map of (T, ℓ) from H to a chordal graph $J_{(T,\ell)}$, then H is isomorphic to a retract of $\Pi J_{(T,\ell)}$. In particular, H would be in the variety generated by connected chordal graphs. Thus our task now is to find a connected chordal graph J for each tree obstruction (T, ℓ) of H such that there is a preserving map of (T, ℓ) from H to J. As mentioned at the end of the previous section, this is an area for future research.

3.5 Near unanimity functions

In Section 2.4, we proved that each graph in $\mathcal{AR}_H$ admits a near unanimity function. The relation between $\mathcal{AR}_O$ and those graphs that admit near unanimity functions is not the same. Instead, we will prove that a graph $H \in \mathcal{AR}_O$, whose tree obstructions have a bounded number of leaves, admits a near unanimity function. Moreover, we will give an example of a graph in $\mathcal{AR}_O$ which has tree obstructions with an arbitrarily large number of leaves; these tree obstructions prevent the graph from admitting a near unanimity function of any arity.

Let H be a graph and let $k \geq 3$ be an integer. Recall the graphs $\mathcal{M}_k(H)$ and $H_{\mathcal{M}_k}$ from Section 2.4. We will use these graphs to relate the graphs in $\mathcal{AR}_O$ and those graphs that admit near unanimity functions.

Lemma 3.5. *Let H be a connected graph and $p \geq 3$ an integer. If (T, ℓ) is a tree obstruction on $H_{\mathcal{M}_p}$, where T has k leaves, $k \leq p-1$, then (T, ℓ) is also a tree obstruction on $\mathcal{M}_p(H)$.*

Proof. Suppose that there exists a tree obstruction (T, ℓ) on $H_{\mathcal{M}_p}$ such that T has k leaves and such that (T, ℓ) is a feasible tree constraint on $\mathcal{M} = \mathcal{M}_p(H)$. We assume that (T, ℓ) is a **minimum** such tree obstruction on $H_{\mathcal{M}_p}$ in the following sense: if (T', ℓ') is a tree obstruction on $H_{\mathcal{M}_p}$ with at most $p-1$ leaves such that $|V(T')| < |V(T)|$, then (T', ℓ') is also a tree obstruction on $\mathcal{M}_k(H)$.

Let $\{x_1, \ldots, x_k\}$ be the leaves of T. Then as (T, ℓ) is a tree obstruction on $H_{\mathcal{M}_p}$, $\ell(x_i) = \overline{z}_i$ for some vertex $z_i \in V(H)$, $i = 1, \ldots, k$. Let $\phi : T \rightarrow \mathcal{M}$ be a homomorphism that extends ℓ. We will show that for all vertices w of T, $\phi(w) \in V(H_{\mathcal{M}_p})$ if and only if w is a leaf of T.

Suppose that there exits a vertex w of T such that w is not a leaf and $\phi(w) \in V(H_{\mathcal{M}_p})$. Let $T_1, \ldots, T_j$, $j = \deg_T(w)$, be the subtrees of T such that for $i = 1, \ldots, j$, each tree T_i contains w, each $T_i \setminus w$ is a component of $T \setminus w$ and $T = \cup_{i=1}^{j} T_i$. Observe that w is a leaf of T_i and in fact w is the only leaf of T_i that is not also a leaf of T, $i = 1, \ldots, j$. Let (T_i, ℓ_i) be the tree constraint on $H_{\mathcal{M}_p}$ with leaf-labeling ℓ_i defined by

$$\ell_i(z) = \begin{cases} \phi(w) & \text{if } z = w \\ \ell(z) & \text{otherwise.} \end{cases}$$

As (T, ℓ) is an infeasible tree constraint on $H_{\mathcal{M}_p}$, at least one of $(T_1, \ell_1), \ldots, (T_j, \ell_j)$ must also be infeasible on $H_{\mathcal{M}_p}$. Without loss of generality, assume (T_1, ℓ_1) is infeasible on $H_{\mathcal{M}_p}$. Then, by the way that we chose (T, ℓ), the tree constraint (T_1, ℓ_1) must also be infeasible on $\mathcal{M}$. This is a contradiction as ϕ restricted to T_1 is homomorphism from T_1 to $\mathcal{M}$ that extends ℓ_1. Therefore only the leaves of T are mapped to $H_{\mathcal{M}_p}$.

Let w_i be the nontrivial neighbour of x_i in T. Let $T' = T \setminus \{x_1, \ldots, x_k\}$. Then by the work done in the previous paragraph, $\phi(T') \subseteq \mathcal{M} \setminus H_{\mathcal{M}_p}$. By the construction of $\mathcal{M}$, we also have that $\mathcal{M} \setminus H_{\mathcal{M}_p} \subseteq H^p$. Consider the edge $\phi(x_i)\phi(w_i)$ in $\mathcal{M}$. Since $\phi(x_i) = \ell(x_i) = \overline{z_i}$ and $\phi(w_i) \in V(H^p)$, there exists a vertex $\boldsymbol{y}_i$ in H^p such that $\boldsymbol{y}_i\phi(w_i)$ is an edge of H^p and z_i occurs at least $p-1$ times in $\boldsymbol{y}_i$, $i = 1, \ldots, k$. Since $k \leq p-1$, there exists an index q such that $\pi_q(y_i) = z_i$, for $i = 1, \ldots, k$, where π_q is the qth projection. Now define $\phi' : T \to H_{\mathcal{M}_p}$ as follows:

$$\phi'(t) = \begin{cases} \overline{z_i} & \text{if } t = x_i \\ \overline{\pi_q(\phi(t))} & \text{otherwise} \end{cases}$$

The function ϕ' is a homomorphism by construction and $\phi'(x_i) = \overline{z_i} = \ell(x_i)$, $i = 1, \ldots, k$. Therefore $\phi' : T \to H_{\mathcal{M}_p}$ is a homomorphism that extends ℓ, contradicting the infeasibility of (T, ℓ) on H.

$\square$

Theorem 3.6. *Let H be a graph in $\mathcal{AR}_O$. If there exits an integer k such that all tree obstructions on H have at most $k-1$ leaves, then H admits a near unanimity function of arity k.*

Proof. Let H be a graph in $\mathcal{AR}_O$ and suppose that there exists an integer k such that all tree obstructions on H have at most $k-1$ leaves. Then, as H and $H_{\mathcal{M}_k}$ are isomorphic, $H_{\mathcal{M}_k} \in \mathcal{AR}_O$ and all tree obstructions on $H_{\mathcal{M}_k}$ have at most $k-1$ leaves. By Lemma 3.5, all tree obstructions on $H_{\mathcal{M}_k}$ are tree obstructions on $\mathcal{M}_k(H)$. Therefore $H_{\mathcal{M}_k}$ is a retract of $\mathcal{M}_k(H)$, and so H admits a near unanimity function of arity k by Proposition 2.7.

□

The key difference between Theorem 2.8 and Theorem 3.6 is the following: the size of hole f on a graph H is bounded above by the number of vertices of H as $D_f \subseteq V(H)$, but the number of leaves in a tree obstruction (T, ℓ) on H has no inherent upper bound. We make use of this fact in constructing a graph in $\mathcal{AR}_O$ that has tree obstructions with arbitrarily large numbers of leaves. But first we will show that tree obstructions prevent near unanimity functions in the same way that holes prevent near unanimity functions.

Theorem 3.7. *Let H be a connected graph that admits a near unanimity function of arity k. Then all tree obstructions on H have at most $k-1$ leaves.*

Proof. Suppose that there exists a tree obstruction (T, ℓ) on H such that T has leaves $x_1, \ldots, x_p$, $p \geq k$. Let y_i be the nontrivial neighbour of x_i in T and let $e_i = x_i y_i$, $i = 1, \ldots, p$. Since (T, ℓ) is a tree obstruction, the tree constraint (T_{e_i}, ℓ) is feasible on H, $i = 1, \ldots, p$. Let $\phi_{e_i} : T_{e_i} \to H$ be a homomorphism that extends ℓ. Define the map $\phi_i : T \to H$ as

$$\phi_i(t) = \begin{cases} \text{a neighbour of } \phi_{e_i}(y_i) \text{ in } H & \text{if } t = x_i \\ \phi_{e_i}(t) & \text{otherwise} \end{cases}$$

Clearly ϕ_i is a homomorphism, $i = 1, \ldots, p$.

As H admits a near unanimity function of arity k, it must also admit a near unanimity function of arity p by Lemma 1.5. Let $\eta : H^p \to H$ be such a near unanimity function. Now define the map $\phi : T \to H$ by

$$\phi(t) = \eta(\phi_1(t), \ldots, \phi_p(t)).$$

As η and ϕ_i, $i = 1, \ldots, p$, are homomorphisms, ϕ must also be homomorphism. By the definition of ϕ_i, $\phi_i(x_j) = \ell(x_j)$ for $1 \leq i, j \leq p$ and $i \neq j$. Therefore, since η is a near unanimity function, $\phi(x_j) = \ell(x_j)$, $j = 1, \ldots, p$. Hence $\phi : T \to H$ is an extension of ℓ, contradiction.

Therefore all tree obstructions on H may have at most $k-1$ leaves.

□

Corollary 3.2. *Let H be a graph in $\mathcal{AR}_O$. If there exists an integer k such that there exists a tree obstruction on H with $k-1$ leaves, but there does not exist a tree obstruction on H with more than $k-1$ leaves, then H admits a near unanimity function of arity k, but does not admit a near unanimity function of arity $k-1$.*

□

We now present a graph that prevents $\mathcal{AR}_O$ from being contained in the class of graphs that admit a near unanimity function.

Theorem 3.8. *There exists a graph in $\mathcal{AR}_O$ that does not admit a near unanimity function of any arity.*

Proof. Let W be the graph in Figure 3.9. We will show that W is in $\mathcal{AR}_O$ in Section 4.4.1.

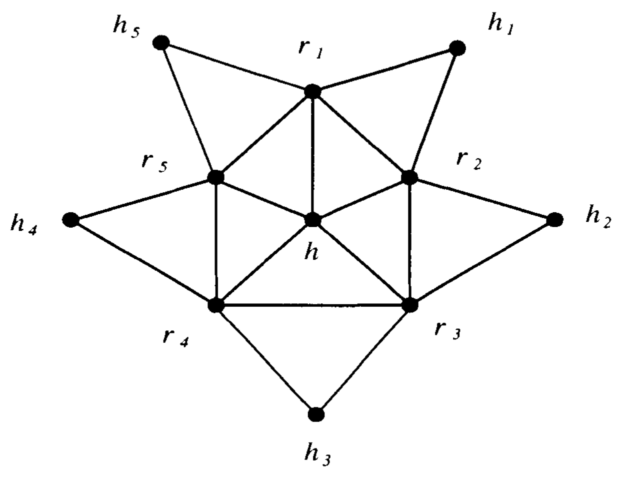


Figure 3.9: A graph in $\mathcal{AR}_O$ that does not admit a near unanimity function of any arity.

We claim that for any positive integer k, we can create a tree obstruction on W that has at least $5k$ leaves. Then by Theorem 3.7, W does not admit a near unanimity function of arity $5k$. This in turn implies that W does not admit a near unanimity function of arity p, $3 \leq p \leq 5k$, by Lemma 1.5.

Let T_k be a tree with $V(T_k) = \{x_1, \ldots, x_{5k}, y_2, \ldots, y_{5k-1}\}$, where $x_1y_2y_3\ldots y_{5k-1}x_{5k}$ is a path and x_iy_i is an edge for $i = 2, \ldots, 5k-1$; there are no other edges of T_k (except for the loops at each vertex). Observe that $\{x_1, \ldots, x_k\}$ is the set of leaves of T_k. Give T_k the following leaf-labeling: $\ell(x_1) = \ell(x_{5k}) = r_1$ and $\ell(x_i) = h_j$ when $i \equiv j \bmod 5$.

We will first prove that (T_k, ℓ_k) is infeasible, and then prove that it is minimally infeasible.

Suppose that there exists a homomorphism $\phi : T_k \to W$ that extends ℓ_k. Since we must have $\phi(x_1) = \ell(x_1) = r_1$ and $\phi(x_2) = \ell(x_2) = h_2$, this forces $\phi(y_2) = r_2$. Assume that $\phi(y_i) = r_j$, where $j \equiv i \bmod 5$, for $i \geq 2$ and consider y_{i+1}. Since $\phi(y_i) = r_j$, where $j \equiv i \bmod 5$ and $\phi(x_{i+1}) = \ell(x_{i+1}) = h_p$, where $p \equiv i+1 \bmod 5$, $j+1 \equiv p \bmod 5$ and so we must have $\phi(y_{i+1}) = r_{j+1}$, where $i \equiv j \bmod 5$. Hence, in particular, we have that $\phi(y_{5k-1}) = r_4$ and so $\phi(y_{5k-1})\phi(x_{5k})$ is not an edge of W as $\phi(x_{5k}) = \ell(x_{5k}) = r_1$. Therefore (T_k, ℓ_k) is an infeasible tree constraint on W.

Now we will prove that (T_k, ℓ_k) is minimally infeasible. By Proposition 3.1, it is enough to prove that any subdivision of (T_k, ℓ_k) is feasible. Note that in the argument in the previous paragraph, every edge was used once to force a value of ϕ (or in the case of $y_{5k-1}x_{5k}$ to cause the contradiction). Thus any subdivision of (T_k, ℓ_k) will be a feasible constraint on W.

□

When we introduced the definition of a tree obstruction, we remarked on a more general concept called an obstruction. We know that if H is in $\mathcal{AR}_O$, then H admits a near unanimity function of arity k if and only if all tree obstructions on H have at most $k-1$ leaves. However, if we consider obstructions, the same equivalency is true even when we don't restrict H to $\mathcal{AR}_O$. Namely, a graph H admits a near unanimity function of arity k if and only if all obstructions on H have at most $k-1$ labeled vertices [49, 15].

Chapter 4

Arc Consistency

In Chapter 3, we rephrased the retraction problem as a list homomorphism problem to relate $\mathcal{AR}_O$ to tree duality and to make use of Algorithm 3.1 when studying tree obstructions on chordal graphs. Now, we will use the list homomorphism problem version of the retraction problem to derive a necessary condition for the existence of a retraction.

Let H be a graph. Let (G, L) be a pair where G is a graph and where $L(g) \subseteq V(H)$ is a list for each $g \in V(G)$. Consider an edge gg' of G. We say that a vertex $h \in L(g)$ has *support* in $L(g')$ if there exists a vertex $h' \in L(g')$ such that $hh' \in E(H)$. If all vertices in $L(g)$ have support in $L(g')$ and all the vertices of $L(g')$ have support in $L(g)$, we say that $L(g')$ and $L(g)$ are *arc consistent.* Note that if there exists a list homomorphism ϕ, $(G, L) \xrightarrow{\phi} H$, then obviously $\phi(g) \in L(g)$ for all $g \in V(G)$, and for all vertices $g' \in N_G(g)$, $\phi(g)$ has support in $L(g')$. Thus if a vertex h in $L(g)$ has no support in $L(g')$, then we may as well remove h from $L(g)$ as g could never be sent to h by ϕ. Thus it is logical that when looking for a list homomorphism from G to H, we remove vertices from lists until the lists of adjacent vertices in G are arc consistent. We then say that the lists of G are arc consistent. This leads to the well known Arc Consistency Algorithm [53].

Algorithm 4.1.

Input: A graph H.

: A pair (G, L), where G is a graph with non-loop edges $\{e_1, \ldots, e_m\}$ and where $L(g) \subseteq V(H)$ is a list for each $g \in V(G)$.

Task: To produce arc consistent lists for the vertices of G.

Action: Process the non-loop edges of G. In iteration $i \geq 1$, we consider the edge e_j, $j \equiv i \bmod m$. If $e_j = gg'$, then we make $L(g)$ and $L(g')$ arc consistent by removing from $L(g)$ any vertex that has no support in $L(g')$ and vice versa.

$$L(g) \leftarrow \{h \in L(g) \mid \exists h' \in L(g'), hh' \in E(H)\}$$

$$L(g') \leftarrow \{h' \in L(g') \mid \exists h \in L(g), h'h \in E(H)\}$$

: Stop with failure if a list becomes empty.

: Stop with success if no list has been changed in the last m iterations.

Note that the algorithm ends as the input graphs are finite.

Our area of interest, however, is not the general list homomorphism problem " Does there exist a list homomorphism from (G, L) to H?", where H and G are graphs and $L(g) \subseteq V(H)$ for all $g \in V(G)$; instead, we are interested in the list homomorphism problem that is equivalent to the retraction problem.

Let H be a connected graph and let G be a connected supergraph of H. Assign to each vertex g of G the list $L(g)$, where

$$L(g) = \begin{cases} \{g\} & \text{if } g \in V(H) \\ V(H) & \text{otherwise.} \end{cases}$$

As mentioned in Section 3.2, H is a retract of G if and only if $(G, L) \to H$. We refer to (G, L) and H as the list homomorphism problem related to the retraction problem.

We can apply Algorithm 4.1 to (G, L) and H. Since the lists of G are derived from H, we can incorporate this into the Arc Consistency Algorithm when we apply it to this special case of the list homomorphism problem.

Algorithm 4.2.

Input:	A graph G with non-loop edges $\{e_1, \ldots, e_m\}$.
:	A subgraph H of G
Task:	To test for the existence of a retraction from G to H.
Action:	Initialize the lists of the vertices of G. For each vertex g of G, assign g the list $L(g)$ that consists of $\{g\}$ if $g \in V(H)$ and $V(H)$ otherwise.
:	Apply Algorithm 4.1 to (G, L) and H.

The Algorithm 4.2 will end as the input graphs are finite.

Hereafter, the original Arc Consistency Algorithm, Algorithm 4.1, will be referred to as the ACL Algorithm and the version of the Arc Consistency Algorithm modified for retractions, Algorithm 4.2, will be referred to as the ACR Algorithm.

4.1 Absolute retracts with respect to arc consistency

We will define the class of absolute retracts with respect to arc consistency. Next, we state that this new class is a variety, but we defer this proof to a later section. Lastly, we will present a graph that in some sense "contains all arc consistent lists" relative to H, which we use to classify the graphs that are absolute retracts with respect to arc consistency.

Let H be a connected graph and let G be a connected supergraph of H. Assign to each vertex g of G a list $L(g)$, where

$$L(g) = \begin{cases} \{g\} & \text{if } g \in V(H) \\ V(H) & \text{otherwise.} \end{cases}$$

In the previous section, we pointed out that $(G, L) \to H$ only if the ACL Algorithm applied to (G, L) and H succeeds. This is equivalent to saying that H is a retract of G only if the ACR Algorithm applied to H and G succeeds. Thus arc consistency provides us with a new necessary condition for the existence of retractions.

The class of *absolute retracts with respect to arc consistency*, denoted by $\mathcal{AR}_C$, is the set of all connected graphs H such that H is a retract of a connected supergraph G if and only if the ACR Algorithm succeeds when applied to H and G.

Note that the success of the ACR Algorithm is not a sufficient condition, as demonstrated by Figure 4.1.

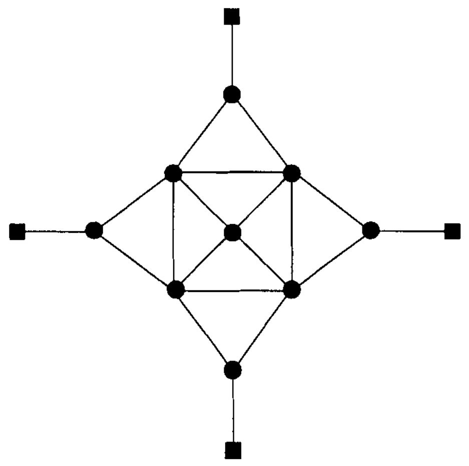


Figure 4.1: Let H be the graph induced by the round vertices. Observe that H is dismantlable. Let G be the entire graph where the square vertices form a clique. The ACR Algorithm succeeds when applied to H and G, but there is no retraction from G to H.

Theorem 4.1. *The class $\mathcal{AR}_C$ is a variety.*

The proof of Theorem 4.1 is deferred to Section 4.3.2.

Let H be a graph. Define the *power graph* of H, denoted by $\mathcal{P}(H)$, to be the graph whose vertex set is the set of all non-empty subsets of $V(H)$. Furthermore, two vertices X and Y of $\mathcal{P}(H)$ are adjacent if every vertex x in X has a neighbour in Y in the graph H, and every vertex y in Y has a neighbour in X in the graph H. In other words vertices X and Y of $\mathcal{P}(H)$ are adjacent if and only if every vertex of X has support in Y and every vertex of Y has support in X. Let x and y be two vertices of H. Then $\{x\}$ and $\{y\}$ are vertices of $\mathcal{P}(H)$. Let $\{x\} = Z_0Z_1 \dots Z_n = \{y\}$ be a path from $\{x\}$ to $\{y\}$ in $\mathcal{P}(H)$. By definition of adjacency in $\mathcal{P}(H)$, there is a neighbour of x in Z_1 in H. Call this neighbour z_1. For $i = 2, \dots, n$, let z_i be a neighbour of z_{i-1} in Z_i in H. Thus $xz_1 \dots z_{n-1}z_n = y$ is a walk from x to y in H. Clearly we must have $n \geq d_H(x, y)$. Hence $d_{\mathcal{P}(H)}(\{x\}, \{y\}) \geq d_H(x, y)$. On the other hand, if there is a path $x = w_0w_1 \dots w_k = y$ in H, there is a corresponding path $\{x\} = \{w_0\}\{w_1\} \dots \{w_k\} = \{y\}$ in $\mathcal{P}(H)$. Thus $d_H(x, y) = d_{\mathcal{P}(H)}(\{x\}, \{y\})$. Let $H_{\mathcal{P}}$ be the subgraph of $\mathcal{P}(H)$ induced by vertices that are sets of size one. Clearly $H_{\mathcal{P}}$ is isomorphic to H, and by our discussion, $H_{\mathcal{P}}$ is an isometric copy of H in $\mathcal{P}(H)$.

We are interested in when $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$. In [33], they also studied the idea of a power graph and investigated when $\mathcal{P}(H) \to H$, where H is irreflexive.

Proposition 4.1. *Let H be a connected graph. Then H is in $\mathcal{AR}_C$ if and only if $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$.*

Proof. Assume that H is in $\mathcal{AR}_C$. As H and $H_{\mathcal{P}}$ are isomorphic, $H_{\mathcal{P}} \in \mathcal{AR}_C$. Thus if the ACR Algorithm succeeds when applied to $H_{\mathcal{P}}$ and $\mathcal{P}(H)$, then $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$. Let X be a vertex of $\mathcal{P}(H)$. Then X is an element of the list assigned to it in the initialization phase of the ACR Algorithm. We may assume that X is an element of the list assigned to it up to iteration i for all vertices X of $\mathcal{P}(H)$. Let XY be the edge of $\mathcal{P}(H)$ that is processed in iteration i. By assumption X is in its own list, and so is Y. We just stated that XY is an edge, so neither X nor Y will be removed from their respective lists when the edge XY is processed. Therefore as we run the ACR Algorithm on $H_{\mathcal{P}}$ and $\mathcal{P}(H)$, the lists will never become empty and so the ACR Algorithm will stop with success. Therefore $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$.

Now assume that $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$ and let $\theta : \mathcal{P}(H) \to H_{\mathcal{P}}$ be a particular retraction. Let G be a connected supergraph of H such that the ACR Algorithm succeeds when applied to H and G. For each vertex g of G, let $L(g)$ be the final list assigned to g by the ACR Algorithm. As $L(g)$ is a non-empty subset of $V(H)$, $L(g)$ is a vertex of $\mathcal{P}(H)$. Thus we can defined a function ϕ from G to $\mathcal{P}(H)$ by sending the vertex g of G to the vertex $L(g)$ of $\mathcal{P}(H)$. The function ϕ is in fact a homomorphism from G to $\mathcal{P}(H)$; if vertices g and g' of G are adjacent, then all vertices of $L(g)$ must have support in $L(g')$ and vice versa. Thus $L(g)$ and $L(g')$ are adjacent in $\mathcal{P}(H)$. Then it is not hard to see that $\theta \circ \phi$ is a homomorphism from G to $H_{\mathcal{P}}$ such that $\theta \circ \phi(x) = \{x\}$ for all vertices x in H. As H is isomorphic to $H_{\mathcal{P}}$, there is a retraction from G to H.

□

The power graph has another very nice property; namely, if H is a connected graph, then $\mathcal{P}(H) \in \mathcal{AR}_C$, as proven in Proposition 4.2. We will use this property in Section 4.5 to describe a particular family of graphs in $\mathcal{AR}_C$.

Proposition 4.2. *[48] Let H be a connected graph. Then $\mathcal{P}(H)$ is in $\mathcal{AR}_C$.*

Proof. Let $\mathcal{P}^2(H) = \mathcal{P}(\mathcal{P}(H))$. The vertices of $\mathcal{P}^2(H)$ are of the form $\{X_1, \dots, X_p\}$, where X_i is non-empty subset of $V(H)$ for $i = 1, \dots, p$. The graph $\mathcal{P}(H)_{\mathcal{P}}$ is the subgraph of $\mathcal{P}^2(H)$ induced by the vertices $\{X\}$ where X is a vertex of $\mathcal{P}(H)$, i.e., X is a non-empty subset of $V(H)$. We will prove that $\mathcal{P}(H)_{\mathcal{P}}$ is a retract of $\mathcal{P}^2(H)$. Thus by Proposition 4.1, $\mathcal{P}(H)$ will be an element of $\mathcal{AR}_C$.

Define a function θ from $\mathcal{P}^2(H)$ to $\mathcal{P}(H)_{\mathcal{P}}$ by

$$\theta(\{X_1, \dots, X_p\}) = \left\{ \bigcup_{i=1}^{p} X_i \right\}.$$

Let $\{X_1, \dots, X_p\}$ and $\{Y_1, \dots, Y_q\}$ be adjacent vertices of $\mathcal{P}^2(H)$. By definition of adjacency in $\mathcal{P}^2(H)$, X_i is adjacent to some $Y_{j_i} \in \{Y_1, \dots, Y_q\}$ in $\mathcal{P}(H)$ for $i = 1, \dots, p$. Now as X_i and Y_{j_i} are adjacent vertices of $\mathcal{P}(H)$, each x in X_i must have a neighbour y in Y_{j_i} in H for $i = 1, \dots, p$. Thus each $x \in \cup_{i=1}^{p} X_i$ has a neighbour $y \in \cup_{i=1}^{q} Y_i$ in H. Similarly, each $y \in \cup_{i=1}^{q} Y_i$ has a neighbour $x \in \cup_{i=1}^{p} X_i$ in H. Therefore $\{\cup_{i=1}^{p} X_i\}$ and $\{\cup_{i=1}^{q} Y_i\}$ are adjacent in $\mathcal{P}(H)_{\mathcal{P}}$ and so f is a homomorphism. Let X be a non-empty subset of $V(H)$. Then $\{X\}$ is a vertex of $\mathcal{P}^2(H)$ and $f(\{X\}) = \{X\}$. Thus θ is a retraction from $\mathcal{P}^2(H)$ to $\mathcal{P}(H)_{\mathcal{P}}$.

□

4.2 Near unanimity functions and chordal graphs

The relation between absolute retracts with respect to necessary condition N and graphs that admit near unanimity functions, and chordal graphs has changed as we varied the necessary condition N. Starting with $\mathcal{AR}_I$, all graphs in $\mathcal{AR}_I$ admit a near unanimity function of arity 3 and are in the variety generated by connected chordal graphs, see Section 1.2. Next, in Chapter 2, we found that all graphs in $\mathcal{AR}_H$ admit a near unanimity function, but that the variety generated by connected chordal graphs does not contain $\mathcal{AR}_H$ and vice versa. Then, in Chapter 3, we found that there exists a graph in $\mathcal{AR}_O$ that does not admit a near unanimity function. In this section, we find that the relationship between $\mathcal{AR}_C$ and graphs that admit near unanimity functions, and chordal graphs is completely the opposite of the relationship between $\mathcal{AR}_I$ and the graphs that admit near unanimity functions and chordal graphs: the class $\mathcal{AR}_C$ contains all connected chordal graphs, and hence the variety generated by connected chordal graphs, and $\mathcal{AR}_C$ contains all connected graphs that admit a near unanimity function.

Theorem 4.2. *[27] Let H be a connected graph that admits a near unanimity function. Then H is in $\mathcal{AR}_C$.*

Theorem 4.3. *The class of graphs $\mathcal{AR}_C$ strictly contains the class of connected graphs that admit near unanimity functions.*

Proof. Let W be the graph that is in Figure 3.9. As shown in the proof of Theorem 3.8, the graph W does not admit a near unanimity function of any arity. We will prove in Section 4.4.1 that W is in $\mathcal{AR}_C$; the graph W is a wheel extension, as defined in Section 4.4.

□

Theorem 4.4. *[15, 28] Let H be a connected chordal graph. Then H is in $\mathcal{AR}_C$.*

The proof we present here is based on comments made in [28]. We have chosen this proof since we use a modified version of the techniques displayed further on in the thesis. Note that we can also use Theorem 4.2 to infer that connected chordal graphs are in $\mathcal{AR}_C$ as all chordal graphs admit near unanimity functions by Theorem 1.10.

Proof. Let H be connected chordal graph with perfect elimination ordering $h_1, \ldots, h_n$. Let G be a connected supergraph of H such that ACR Algorithm succeeds on H and G and let $L'(g)$ be the final list assigned to $g \in V(G)$ by the ACR Algorithm. We will prove that $L'(g)$ is connected and moreover that Algorithm 3.1 succeeds on (G, L') and $h_1, \ldots, h_n$.

Consider what happens when ACR Algorithm is applied to H and G. In the initialization step, each vertex g of G is assigned a list $L(g)$, where $L(g) = \{g\}$ if $g \in V(H)$ and $L(g) = V(H)$ otherwise. Thus the initial lists are connected. This connectivity is maintained as we process the non-loop edges of G. Assume all lists are connected before we process the non-loop edge $gg' \in E(G)$. When processing the edge gg' we replace the list of g with $L(g) \cap D(L(g'), 1)$ and vice versa. By Lemma 3.1, these new lists are connected. Therefore the final lists produced by the ACR Algorithm on H and G are connected.

Not only will we prove that Algorithm 3.1 succeeds on (G, L') and H, we will prove that action $**$ need never be performed.

As the lists produced by the ACR Algorithm on H and G are connected and arc consistent, we may assume the following when we apply Algorithm 3.1 to (G, L') and the perfect elimination ordering $h_1, \ldots, h_n$: before we process vertex h_i, all lists are connected and arc consistent. Moreover, action $**$ has not changed any lists. Let $L(g)$ be the list of g before we process the vertex h_i. Now process vertex h_i.

Let g be a vertex of G such that $h_i \in L(g)$.

Suppose $|L(g)| \geq 2$. Then action $*$ is applied to the list of $L(g)$. As in the proof of Theorem 3.3, $L(g) \setminus \{h_i\}$ is still connected. Let g' be a neighbour of g, and let h_k be a neighbour of h_i in $L(g')$. Again as in the proof of Theorem 3.3, the vertex

h_i has a neighbour h_j in $L(g)$, $i < j$. If $L(g') = \{h_k\}$, then h_k and h_j are adjacent as arc consistency has held up till now. If $|L(g')| \geq 2$, then $i \leq k$, and so h_j and h_k are adjacent as they are neighbours of h_i that follow h_i in the perfect elimination ordering. Hence action $*$ maintains arc consistency.

Now suppose that $L(g) = \{h_i\}$. As arc consistency has held till now, h_i is adjacent to all vertices in $L(g')$, for all neighbours g' of g in G. Hence action $**$ has no effect.

□

Corollary 4.1. *The variety generated by connected chordal graphs is contained in $\mathcal{AR}_C$.*

Proof. By Theorem 4.4, all chordal graphs are in $\mathcal{AR}_C$. Since $\mathcal{AR}_C$ is a variety by Theorem 4.1, $\mathcal{AR}_C$ contains the the variety generated by chordal graphs.

□

4.3 Equivalent classes

The previous section concerned classes that are strictly contained in $\mathcal{AR}_C$. In this section, we will prove that $\mathcal{AR}_C$ is equivalent to three other classes of graphs.

4.3.1 TSI graphs

While the graphs that admit near unanimity functions are strictly contained in $\mathcal{AR}_C$ by Theorem 4.3, there is another type of function that all graphs in $\mathcal{AR}_C$ do admit. In fact, the graphs in $\mathcal{AR}_C$ are characterized as being exactly those graphs that admit this new function.

Let A be a set and let f be a function from A^n to A, where $n \geq 1$ is an integer. The function f is a *totally symmetric function* if

$$f(a_1, \ldots, a_n) = f(b_1, \ldots, b_n) \text{ whenever } \{a_1, \ldots, a_n\} = \{b_1, \ldots, b_n\}, a_i, b_i \in A. \quad (4.1)$$

Notice that a totally symmetric function essentially acts on the non-empty subsets of A of size at most n. The function f is called an *idempotent function* if

$$\phi(a, \ldots, a) = a \text{ for all } a \in A. \tag{4.2}$$

Then ϕ is a *totally symmetric idempotent* function, or a TSI function, if equations 4.1 and 4.2 both hold.

Let H be a graph, and let $\phi : H^n \to H$ be a homomorphism, where $n \geq 1$. Then ϕ is called a *TSI homomorphism* if $\phi : V(H^n) \to V(H)$ is TSI function. We will show that the connected graphs that admit a TSI homomorphism of any arity are exactly the graphs in $\mathcal{AR}_C$. We note that a TSI homomorphism is closely related to the f-functions defined on templates in [33].

The following proposition is a simplification of Proposition 1 in [67] on algebras.

Proposition 4.3. *Let H be a graph.*

i.) If H admits a TSI homomorphism of arity $n \geq 2$, then H admits a TSI homomorphism of arity $n - 1$.

ii.) If H admits a TSI homomorphism of arity $n \geq |V(H)|$, then H admits a TSI homomorphism of arity $n + 1$.

Proof. i. Suppose that H admits a TSI homomorphism of arity $n \geq 2$; call it ϕ. Define a new function ϕ' of arity $n - 1$ on H as follows:

$$\phi'(x_1, \ldots, x_{n-1}) = \phi(x_1, \ldots, x_{n-1}, x_{n-1}).$$

The function ϕ' is well defined because $\{x_1, \ldots, x_{n-1}\} = \{x_1, \ldots, x_{n-1}, x_{n-1}\}$. If $(x_1, \ldots, x_{n-1})$ is a constant vector, so is $(x_1, \ldots, x_{n-1}, x_{n-1})$. Therefore ϕ' is idempotent as ϕ is idempotent. It easy to see that ϕ' must also be totally symmetric and a homomorphism. Therefore ϕ' is a TSI homomorphism of arity $n - 1$.

ii. Suppose that H admits a TSI homomorphism ϕ of arity $n \geq |V(H)|$. As H has at most n vertices, any $n + 1$ tuple of vertices of H will contain at most n distinct vertices. Thus we may define a function ϕ' of arity $n + 1$ as follows:

$$\phi'(x_1, \ldots, x_{n+1}) = \phi(y_1, \ldots, y_n) \text{ whenever } \{x_1, \ldots, x_{n+1}\} = \{y_1, \ldots, y_n\}.$$

Because ϕ is a TSI homomorphism, not only is ϕ' well defined, but ϕ' is also a TSI homomorphism.

□

The next proposition is a modification of Proposition 2.3 in [50] on posets.

Proposition 4.4. *Let H be a graph. Then H admits a TSI homomorphism of arity $|V(H)|$ if and only if $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$.*

Proof. Clearly $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$ if and only if the partial homomorphism $\phi : \mathcal{P}(H) \to H_{\mathcal{P}}$ defined by $\phi(\{x\}) = \{x\}$ extends to a homomorphism. Thus, instead of proving Proposition 4.4, we will prove the following claim:

> H admits a TSI homomorphism of arity $|V(H)|$ if and only if the partial homomorphism $\phi : \mathcal{P}(H) \to H_{\mathcal{P}}$ defined by $\phi(\{x\}) = \{x\}$ extends to a homomorphism.

Suppose that ϕ extends to a homomorphism ϕ'. Thus ϕ' is a homomorphism from $\mathcal{P}(H)$ to $H_{\mathcal{P}}$ that fixes the vertices of $H_{\mathcal{P}}$. Define a function $\phi'' : H^n \to \mathcal{P}(H)$ by

$$\phi''(x_1, \ldots, x_n) = \{x_1, \ldots, x_n\}$$

and a function $\psi : H_{\mathcal{P}} \to H$ by

$$\psi(\{x\}) = x.$$

Obviously, ϕ', ψ and ϕ'' are all homomorphisms. Then $\psi \circ \phi' \circ \phi''$ is a homomorphism from H^n to H that is also a TSI function of arity n.

Now assume that there exists a TSI homomorphism of arity $n = |V(H)|$ on H, call it α. Define a function $\phi'' : \mathcal{P}(H) \to H_{\mathcal{P}}$ by

$$\phi''(X) = \{\alpha(y_1, \ldots, y_n)\} \text{ where } \{y_1, \ldots, y_n\} = X.$$

Then ϕ'' is well defined homomorphism that extends ϕ.

□

The following equivalency is a modification of a poset result [51] and was brought to the attention of the author by Benoit Larose.

Theorem 4.5. *Let H be a connected graph. Then H admits a TSI homomorphism of arity k for all positive integers k if and only if H is in $\mathcal{AR}_C$.*

Proof. Note that H admits a TSI homomorphism of arity k for all k if and only if H admits a TSI homomorphism of arity $|V(H)|$ by Proposition 4.3 and H admits a TSI homomorphism of arity $|V(H)|$ if and only if $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$ by Proposition 4.4. Lastly, $H_{\mathcal{P}}$ is a retract of $\mathcal{P}(H)$ if and only if H is in $\mathcal{AR}_C$ by Proposition 4.1.

$\square$

4.3.2 Tree duality

We will prove that a graph H has retraction tree duality if and only if H is in $\mathcal{AR}_C$. Thus if a graph H is in $\mathcal{AR}_C$ and H is not a retract of some connected supergraph G, then there exists a tree T that is somehow preventing the retraction.

Recall from Section 3.2 that a connected graph H has retraction tree duality exactly when the following two statements are equivalent for all connected supergraphs G of H with list assignment $L(g) = \{g\}$ if $g \in V(H)$ and $V(H)$ otherwise:

i.) H is not a retract of G.

ii.) There exists a tree T with lists L' such that $(T, L') \to (G, L)$ but $(T, L') \not\to H$.

By the definition of $\mathcal{AR}_C$, a connected graph H is in $\mathcal{AR}_C$ exactly when the following two statements are equivalent for all connected supergraphs G of H:

i.) H is not a retract of G.

ii.) The ACR Algorithm fails on H and G.

Lemma 4.1. *Let H be a graph. Let T be a tree with lists L such that $L(t) \subseteq V(H)$ for all $t \in V(T)$. Then $(T, L) \to H$ if and only if the ACL Algorithm succeeds on (T, L) and H.*

Proof. Assume that $(T, L) \xrightarrow{\phi} H$. Then, when we apply the ACL Algorithm to (T, L) and H, the vertex $\phi(t)$ will remain in the list of t during all iterations of the ACL Algorithm. Hence no list may become empty, and so the ACL Algorithm succeeds on (T, L) and H.

Now assume that the ACL Algorithm succeeds on (T, L) and H, and let L' be the non-empty lists that result. If $L'(t)$ is a singleton for all $t \in V(T)$, then clearly $(T, L) \to H$. Thus, suppose that there exists a vertex $t_0 \in V(T)$ such that $|L'(t_0)| \geq 2$, and let h be a vertex in $L'(t_0)$. Assign the vertices of T new lists L'', where

$$L''(t) = \begin{cases} L'(t) \setminus \{h\} & \text{if } t = t_0 \\ L'(t) & \text{otherwise.} \end{cases}$$

If we can prove that the ACL Algorithm succeeds on (T, L'') and H, then we can continue in this manner, and eventually produce singleton arc consistent lists for the vertices of T, which would imply that $(T, L) \to H$.

Order the non-loop edges of T as follows: the edges incident with t_0, the edges xy such that $d(x, t_0) = 1$ and $d(y, t_0) = 2$, the edges xy such that $d(x, t_0) = 2$ and $d(y, t_0) = 3$, and so on. This is possible as T has no cycles. Now apply ACL Algorithm to (T, L'') and H with this ordering of the non-loop edges of T. When we process an edge incident with t_0, say $t_0 t_1$, then all vertices of $L''(t_0) = L'(t_0) \setminus \{h\}$ have support in $L''(t_1) = L'(t_1)$, but not all vertices of $L''(t_1)$ necessarily have support in $L''(t_0)$. Hence $L''(t_0)$ won't change. However the list $L''(t_1)$ may change, but doesn't become empty because each vertex of $L''(t_0)$ is supported by $L''(t_1)$. Let t_2 be a neighbour of t_1 at distance two from t_0. When we process the edge $t_1 t_2$, all the vertices in the new list of t_1 have support in the list of t_2, but not vice versa. Therefore the list of t_1 will not change, and while the list of t_2 may change, it will not become empty, and so on. Hence, once we have processed all the edges of T, using the described ordering, we end with non-empty arc consistent lists. Note that we pass through the non-loop edges of T twice when we apply the ACL Algorithm to (T, L'') and H; once to change all the lists appropriately, and the second time to fulfill the stopping condition of the ACL Algorithm.

□

Theorem 4.6. *A graph H has retraction tree duality if and only if H is in $\mathcal{AR}_C$.*

Proof. Let H be a connected graph and let G be a connected supergraph of H. Associate a list $L(g)$ with each vertex $g \in V(G)$ such that $L(g) = \{g\}$ if $g \in V(H)$ and $V(H)$ otherwise. We will prove that H has retraction tree duality if and only if $H \in \mathcal{AR}_C$ by showing that second statements from the definition of retraction tree duality and the alternate definition of $\mathcal{AR}_C$ are equivalent.

Suppose that there exists a tree T with lists L' such that $(T, L') \xrightarrow{\phi} (G, L)$, but $(T, L') \not\rightarrow H$. Then the ACL Algorithm fails on (T, L') and H by Lemma 4.1. Let G_T be the image of T under ϕ in G, and let L_T be the restriction of L to the vertices of G_T, i.e., $L_T(g) = L(g)$ for all vertices $g \in V(G_T)$. By the definition of $(T, L') \xrightarrow{\phi} (G, L)$, we have that $L'(t) = L(\phi(t))$ for all $t \in V(T)$. Let $e_1, \ldots, e_m$ the order in which the non-loop edges of T were processed when we applied ACL Algorithm to (T, L) and H. Apply ACL Algorithm to (G_T, L_T) and H, processing the non-loop edges of G_T in the order $\phi(e_1), \ldots, \phi(e_m)$, ignoring loops as necessary; note that the ACL Algorithm must fail in this instance also. Since the ACL Algorithm fails on (G_T, L_T) and H, and since $G_T \subseteq G$ where $L_T(g) = L(g)$ for all vertices $g \in V(G_T)$, the ACL Algorithm must also fail when applied to (G, L) and H. Therefore ACR Algorithm fails on H and G.

Now suppose that the ACR Algorithm fails on H and G. Then the ACL Algorithm fails on (G, L) and H. We construct a tree T with lists L' and a homomorphism $\phi : T \rightarrow G$ such that $(T, L') \xrightarrow{\phi} (G, L)$ and $(T, L') \not\rightarrow H$ by back-tracing through the ACL Algorithm applied to (G, L) and H.

Let g_0 be the vertex of G whose list became empty when the ACL Algorithm failed on (G, L) and H. Initialize the tree T to be the vertex t_0, set $\phi(t_0) = g_0$ and set $\mu(t_0)$ to be the iteration of the ACL Algorithm applied to (G, L) and H in which the list of g_0 became empty. Put t_0 at the end of the queue Q.

Repeat the following steps while Q is not empty: let t be the vertex at the beginning of the queue Q and let g be the vertex of G such that $g = \phi(t)$. For each neighbour g' of g such that the list of g' caused the list of g to loose a vertex in some iteration $i < \mu(t)$, create a neighbour t' of t, set $\phi(t') = g'$, set $\mu(t') = i$, set $L'(t') = L(g')$ and put t' at the end of the queue Q.

This process ends as we back-trace the actions in the ACL Algorithm applied to finite graphs. It is easy to see that $(T, L') \xrightarrow{\phi} (G, L)$. By construction, $(T, L') \not\to H$; apply ACL Algorithm to (T, L') and H, processing the edges of T in the reverse ordering in which they were created. Thus we have constructed (T, L') and ϕ, as advertised.

$\square$

Corollary 4.2. *Let H be a connected graph. Then the following statements are equivalent:*

i.) The graph H admits a TSI of arity k, for all $k \geq 1$.

ii.) The graph H has retraction tree duality.

iii.) The graph H is in $\mathcal{AR}_O$.

iv.) The graph H is in $\mathcal{AR}_C$.

Proof. Let H be a connected graph. Then H admits a TSI of arity k, for all $k \geq 1$ if and only if $H \in \mathcal{AR}_C$ by Theorem 4.5. Next, H is in $\mathcal{AR}_C$ if and only if H has retraction tree duality by Theorem 4.6. Lastly, H has retraction tree duality if only if H is in $\mathcal{AR}_O$ by Theorem 3.2.

$\square$

Proof of Theorem 4.1. Since $\mathcal{AR}_O = \mathcal{AR}_C$ by Corollary 4.2 and since $\mathcal{AR}_O$ is a variety by Theorem 3.1, the class of graphs $\mathcal{AR}_C$ is a variety.

$\square$

Since we now know that the classes $\mathcal{AR}_C$ and $\mathcal{AR}_O$ are in fact the same, we can comment further on how these classes relate to chordal graphs. We ended Section 3.4 with a conjecture concerning the intersection of the variety generated by connected chordal graphs and $\mathcal{AR}_O$. By Corollary 4.1, $\mathcal{AR}_C$, and hence $\mathcal{AR}_O$, contains the variety generated by connected chordal graphs. Therefore, we can rephrase Conjecture 1 as follows:

Conjecture 1′. *The variety generated by connected chordal graphs is exactly the intersection of the variety of strongly stretched graphs and $\mathcal{AR}_O$. In other words, if we let X be the variety generated by connected chordal graphs and let Y be the variety of strongly stretched graphs,*

$$X = \mathcal{AR}_O \cap Y.$$

Conjecture 1′ can also be stated with respect to the class $\mathcal{AR}_C$.

Conjecture 2. *The variety generated by connected chordal graphs is exactly the intersection of the variety of strongly stretched graphs and $\mathcal{AR}_C$. In other words, if we let X be the variety generated by connected chordal graphs and let Y be the variety of strongly stretched graphs,*

$$X = \mathcal{AR}_C \cap Y.$$

4.4 Wheeled graphs

Here we introduce the new class of graphs called wheeled graphs, which is the union of the two classes called wheel extensions and multi-wheel extensions; these graphs are formed by modifying a property of chordal graphs with respect to clique cut set decompositions. We then spend some time developing useful properties of wheel extensions and multi-wheel extensions. Next we prove that both classes are contained in $\mathcal{AR}_C$. The proofs are based on modified versions of Algorithm 3.1 and Theorem 4.4.

We will be using clique cut sets to decompose connected graphs; for disconnected graphs, we may apply the following analysis to each component. Recall from Section 1.1 that a vertex cut set of a graph H is a set $S \subseteq V(H)$ such that $H \setminus S$ is disconnected. For a graph H, we call a clique S in H such that $H \setminus S$ is disconnected a *clique cut set* of H. Recall from Section 1.1 that $\emptyset$ may never be a clique cut set as cliques always contain at least one vertex.

Suppose a connected graph H has a clique cut set K. Then H has a vertex partition $A \cup B \cup K$ such that A has no neighbours in B and $A \neq \emptyset \neq B$. We can then *decompose* H into two subgraphs H' and H'', *separated* by K, where H' is the subgraph of H induced by $A \cup K$ and H'' is the subgraph of H induced by $B \cup K$. Note that H' and H'' are also connected. By decomposing H' and H'' in the same way and repeating until no longer possible, we can decompose H into a set of connected induced subgraphs of H that have no clique cut sets, which are called *atoms* [35, 68]. Such a set is called a *clique cut set decomposition* of H. As remarked in [68], there can be more than one clique cut set decomposition of a graph. This is demonstrated by Figure 4.2. We will show that two distinct clique cut set decompositions for a graph H have the same set of atoms that are not complete graphs.

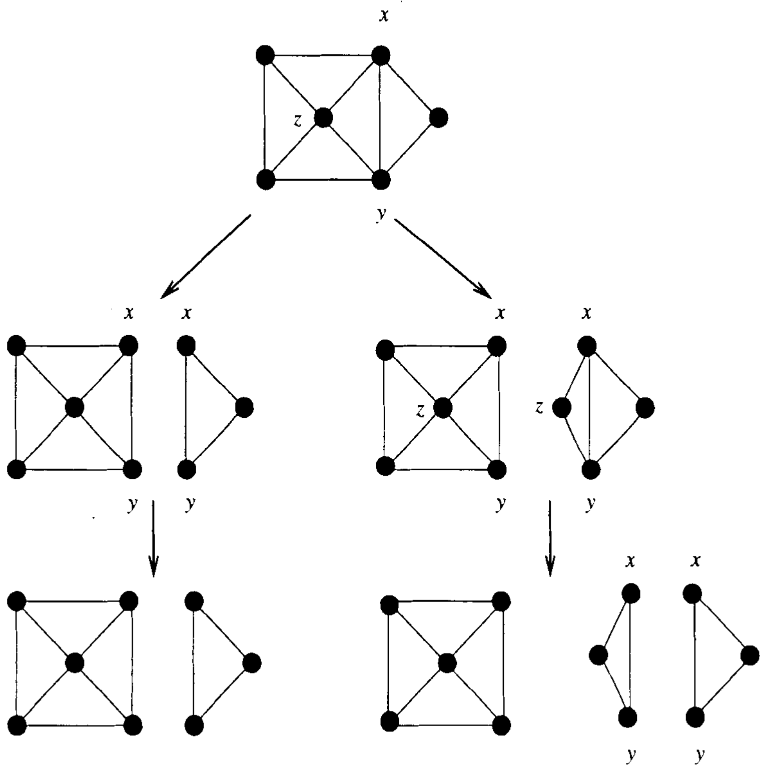


Figure 4.2: A graph with two different clique cut set decompositions; the decomposition on the left used the clique cut set $\{x, y\}$, and the decomposition on the right began with the clique cut set $\{x, y, z\}$ and then used the clique cut set $\{x, y\}$.

Lemma 4.2. *Let H be a connected graph and let J be a connected induced subgraph* *of H that does not have a clique cut set. Then for all clique cut set decompositions $\mathcal{D}$ of H, there exists an atom of $\mathcal{D}$ that has J as an induced subgraph.*

Proof. Let $\mathcal{D}$ be a clique cut set decomposition of H, and consider the induced subgraphs of H produced in decomposing H to create $\mathcal{D}$. Let G be a minimal induced subgraph of H produced in this process such that $J \subseteq G$. Thus if G has a clique cut set K and is decomposed into G' and G'', then neither G' nor G'' can have J as a subgraph. As J and G are both induced subgraphs of H, J must also be an induced subgraph of G. Hence G if is decomposed into G' and G'', then either J is a subgraph of G' or G'', or $K \cap V(J)$ must be a clique cut set of J. The first case contradicts the way we chose G, and the second case contradicts the fact that J doesn't have a clique cut set. Thus G must in fact be an atom of $\mathcal{D}$.

□

Proposition 4.5. *Let H be a connected graph. Let J be a connected induced subgraph of H such that J has no clique cut set and such that if J' is a connected induced subgraph of H with $J \subseteq J'$, then J' must have a clique cut set. Then J must be an atom of any clique cut set decomposition of H.*

Proof. Let $\mathcal{D}$ be a clique cut set decomposition of H. By Lemma 4.2, J is an induced subgraph of some atom $J' \in \mathcal{D}$. According to the definition of an atom, J' is a connected induced subgraph of H and J' has no clique cut set. Therefore $J = J'$ by the maximality of J, and so J is an atom of $\mathcal{D}$.

□

Lemma 4.3. *Let H be a connected graph and let $\mathcal{D}$ be a clique cut set decomposition of H. Suppose that there exists an induced subgraph J of H such that J is an induced subgraph of two distinct atoms of $\mathcal{D}$. Then J is a complete graph.*

Proof. Let G_1 and G_2 be two distinct atoms of $\mathcal{D}$ such that J is an induced subgraph of both. As in the proof of Lemma 4.2, we will make use of the induced subgraphs of H that are produced in creating $\mathcal{D}$. Let G be a minimal induced subgraph of H produced in creating $\mathcal{D}$ such that G_1 and G_2 are induced proper subgraphs of G.

As G_1 and G_2 are atoms of $\mathcal{D}$, we may assume that G has a clique cut set K and that induced subgraphs G' and G'' of G are produced such that G_1 is an induced subgraph of G', G_2 is an induced subgraph of G'' and $G_2 \not\subseteq G'$, $G_1 \not\subseteq G''$ (note that there may be some atom of $\mathcal{D}$ that contains both G_1 and G_2). By the way a clique cut set decomposition is made, $K = V(G') \cap V(G'')$. Therefore, as J is in the intersection of G_1 and G_2, which is contained in the intersection of G' and G'', J must be a complete graph.

□

Proposition 4.6. *Let H be a connected graph and let $\mathcal{D}$ and $\mathcal{D}'$ be two clique cut set decompositions of H. If there exists an atom $J' \in \mathcal{D}' \setminus \mathcal{D}$, then J' is a complete graph.*

Proof. Suppose there exists an atom J' in $\mathcal{D}'$ that is not in $\mathcal{D}$. The atom J' has no clique cut set by definition and so J' must be an induced subgraph of some atom J of $\mathcal{D}$ by Lemma 4.2. If $J' = J$ or if J is a complete graph, then there is nothing left to proof. Thus assume that J is not a complete graph and that J' is a proper induced subgraph of J.

Now consider the atom J. By definition of an atom of H, J has no clique cut set and J is a connected induced subgraph of H. Thus by Lemma 4.2, J must be an induced subgraph of some atom J'' of $\mathcal{D}'$. Hence J' must be a proper induced subgraph of J'' as $J' \subset J \subseteq J''$. Therefore J' is an induced subgraph of itself and J'', two distinct atoms of $\mathcal{D}'$ and so J' must be a complete graph by Lemma 4.3.

□

Let H be a connected graph and let $\mathcal{D}$ and $\mathcal{D}'$ be two clique cut set decompositions of H. By Proposition 4.6 we know that $\mathcal{D}$ and $\mathcal{D}'$ can only differ with regard to their complete atoms; the non-complete atoms of $\mathcal{D}$ and $\mathcal{D}'$ must be exactly the same:

$$\{J \in \mathcal{D} \mid J \text{ not a complete graph }\} = \{J' \in \mathcal{D}' \mid J' \text{ not a complete graph }\}.$$

Denote the set of non-complete atoms of the clique cut set decomposition $\mathcal{D}$ of H (and hence of any clique cut set decomposition of H) by $\widetilde{\mathcal{D}}(H)$. Clearly $\widetilde{\mathcal{D}}(H)$ is a subset of all clique cut set decompositions of H. Moreover, for any clique cut set decomposition $\mathcal{D}$ of H, $\mathcal{D} \setminus \widetilde{\mathcal{D}}(H)$ is either empty or consists solely of complete subgraphs of H. Let

J be a connected induced subgraph of H that has no clique cut set, is not complete and any connected induced subgraph of H that properly contains J has a clique cut set. By Proposition 4.5, J is an atom of any clique cut set decomposition of H. As we have specified that J is also not a complete graph, J must therefore be a member of $\widetilde{\mathcal{D}}(H)$. In fact, such graphs are exactly the elements of $\widetilde{\mathcal{D}}(H)$.

Proposition 4.7. *Let H be a connected graph. Then $\widetilde{\mathcal{D}}(H)$ is the set of all connected induced subgraphs J of H such that:*

i.) J is not a complete graph.

ii.) J does not have a clique cut set.

iii.) if J' is a connected induced subgraph of H such that $J \subset J'$, then J' must have a clique cut set.

Proof. Let J be a connected induced subgraph of H that has the above properties. Then J is an element of $\widetilde{\mathcal{D}}(H)$ by our discussion preceding Proposition 4.7.

Let J be an element of $\widetilde{\mathcal{D}}(H)$. Then by the definition of $\widetilde{\mathcal{D}}(H)$, J is not complete and J does not have a clique cut set. Let J' be any connected induced subgraph of H that properly contains J. We will now prove that J' must have a clique cut set.

Let $\mathcal{D}$ be a clique cut set decomposition of H. Consider the induced subgraphs of H created in the process of making $\mathcal{D}$. Let G be one of these induced subgraphs that is minimal with respect to containing J'. Either G has a clique cut set or G is an atom of $\mathcal{D}$. If G is an atom of $\mathcal{D}$, then J is an induced subgraph of two distinct atoms of $\mathcal{D}$, itself and G as $J \subset J' \subseteq G$. Therefore by Lemma 4.3, J must be complete, a contradiction. Hence G must have a clique cut set, say K. The clique cut set K separates G into two induced subgraphs G' and G'', neither of which contain J' by the way we chose G. Thus $K \cap V(J')$ is a clique cut set of J'.

□

Dirac proved that every minimal cut set of every induced connected subgraph of a chordal graph is a clique, see Theorem 1.8. Thus any clique cut set decomposition of any connected chordal graph consists solely of complete graphs [68]. In fact connected chordal graphs are exactly those graphs whose clique cut set decompositions consist of complete graphs. In this chapter we will be studying connected graphs that have clique cut set decompositions that consist of complete graphs and wheels. Recall from Chapter 1 that a wheel is graph with a nontrivial induced cycle (the rim cycle) and one vertex (the hub) adjacent to all the vertices of the induced nontrivial cycle. It is easy to see that wheels do not have clique cut sets.

Corollary 4.3. *Let H be a connected graph with a clique cut set decomposition $\mathcal{D}$ that consists of wheels and complete graphs. Then all clique cut set decompositions of H consist of wheels and complete graphs and $\widetilde{\mathcal{D}}(H)$ is exactly the set of induced wheels of H.*

Proof. Let $\mathcal{D}'$ be another clique cut set decomposition of H, if one exists. Then by Proposition 4.6, $\mathcal{D}' \setminus \mathcal{D}$ and $\mathcal{D} \setminus \mathcal{D}'$ may only contain complete subgraphs of H. Therefore $\mathcal{D}'$ must consist of all the wheels that are in $\mathcal{D}$, plus possibly some complete graphs, which may or may not be in $\mathcal{D}$.

Now consider $\widetilde{\mathcal{D}}(H)$. By definition, $\widetilde{\mathcal{D}}(H)$ is a subset of $\mathcal{D}$. In particular, $\widetilde{\mathcal{D}}(H)$ contains the atoms of $\mathcal{D}$ that are not complete graphs. Thus $\widetilde{\mathcal{D}}(H)$ is exactly the set of wheels in $\mathcal{D}$. Therefore if J is an element of $\widetilde{\mathcal{D}}(H)$, then J is wheel and as atoms of $\mathcal{D}$ are induced subgraphs of H, J must be an induced wheel of H.

Let W be an induced wheel in H. As W has no clique cut set, W must be an induced subgraph of some atom J of $\mathcal{D}$ by Lemma 4.2. By the choice of $\mathcal{D}$, J must either a complete graph or a wheel. Thus $J = W$, and $W \in \mathcal{D}$. Therefore $W \in \widetilde{\mathcal{D}}(H)$.

□

A connected graph H is said to be a *wheel extension* if there exists a clique cut set decomposition of H that consists of exactly one wheel on at least 5 rim vertices and possibly some complete graphs. In other words, H is a wheel extension if and only if $\widetilde{\mathcal{D}}(H)$ is exactly one wheel on at least 5 rim vertices. Clearly all wheels on at least 5 rim vertices are themselves wheel extensions.

Let H be the graph in Figure 4.3. It's easy to see that we can decompose H

using clique cut sets into the subgraph that is the five wheel and the various complete subgraphs of H.

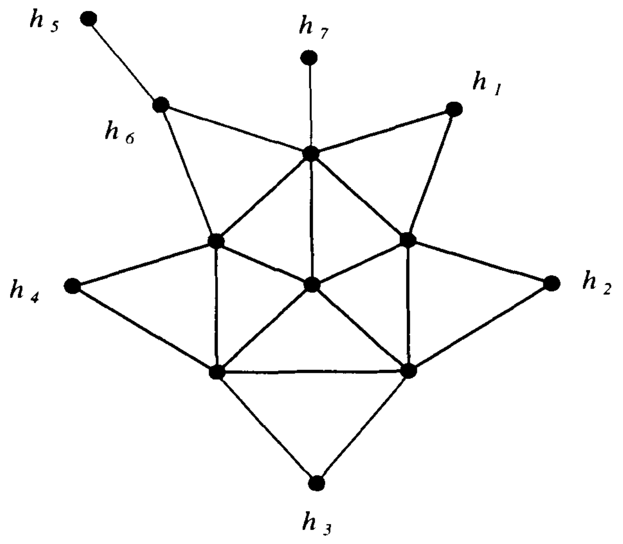


Figure 4.3: An example of a wheel extension.

Proposition 4.8. *Let H be a wheel extension. Then H has a unique induced wheel and the rim of this wheel is the only induced nontrivial cycle of H.*

Proof. As noted above, H is a wheel extension if and only if $\widetilde{\mathcal{D}}(H)$ is exactly one wheel on at least 5 rim vertices. By Corollary 4.3, $\widetilde{\mathcal{D}}(H)$ is exactly the set of induced wheels of H. Thus H has a unique induced wheel.

Let C be an induced nontrivial cycle of H. As C has no clique cut set, C must be a subgraph of some atom in each clique cut decomposition of H by Lemma 4.2. Each clique cut set decomposition of H consists of the unique induced wheel of H and possibly some complete graphs. Thus C must be the rim cycle of the unique induced wheel in H.

$\square$

Recall that if H is a graph with a partial ordering of its vertices, say $h_1, \ldots, h_n$, then $H_i = H \setminus \{h_1, \ldots, h_i\}$ for $i = 1, \ldots, n$, where $i \neq |V(H)|$, and $H_0 = H$.

Proposition 4.9. *Let H be a connected graph. Then H is a wheel extension if and only if there exists a partial perfect elimination ordering $h_1, \dots, h_n$ of H such that H_n is wheel on at least five rim vertices.*

Proof. Let H be a wheel extension. If H is also a wheel, then $H_0 = H$ is a wheel on at least 5 rim vertices and desired partial perfect elimination ordering is the empty ordering. Thus we may assume that H is not a wheel, and that all proper induced subgraphs of H that are wheel extensions have the desired partial perfect elimination ordering. As H is not a wheel, it must have a clique cut set K, that separates H into two connected induced subgraphs H' and H'' where $K = V(H') \cap V(H'')$. As the set of atoms of any clique cut set decomposition of H consists of one wheel on at least 5 rim vertices and some complete graphs, then without loss of generality, the set of atoms of any clique cut set decomposition of H' consist of complete graphs, and the set of atoms of any clique cut set decomposition of H'' consists of one wheel on at least 5 rim vertices and possibly some complete graphs. Thus H' is chordal and H'' is a wheel extension.

Consider the chordal graph H'. It contains the clique K by construction, and cliques are convex (see Section 2.2). Therefore by Theorem 2.3, there exists a partial perfect elimination ordering of H', say $x_1, \dots, x_k$, such that $K = V(H_k)$.

Now consider the graph H'', which also contains the clique K by construction. Since H'' is a proper induced subgraph of H that is a wheel extension, H'' has a partial perfect elimination ordering $y_1, \dots, y_l$ such that H''_l is a wheel on at least 5 rim vertices. Let $h_1, \dots, h_n$ be the partial ordering of H we obtain by concatenating $x_1, \dots, x_k$ and $y_1, \dots, y_l$ in that order.

We claim that $h_1, \dots, h_n$ is a partial perfect elimination ordering of H such that H_n is a wheel on at least 5 rim vertices. As $H_n = H \setminus (\{x_1, \dots, x_k\} \cup \{y_1, \dots, y_l\}) = H''_l$, we see that H_n is indeed a wheel on at least 5 rim vertices. Now we must prove that h_i is simplicial in H_{i-1}, $i = 1, \dots, n-1$. Consider a vertex h_i. If h_i is in H'', it is easy to see that h_i is simplicial in H_{i-1} since $h_i = y_p$ for some p and $H_{i-1} = H \setminus (\{x_1, \dots, x_k\} \cup \{y_1, \dots, y_p\}) = H''_p$. Thus assume that h_i is in $H' \setminus H'' = H' \setminus K$. Therefore $h_i = x_i$ and all the neighbours of x_i in H are contained in H'. Hence, as x_i is simplicial in

H'_{i-1} it must also be simplicial in $H_{i-1} = H\backslash h_1, \ldots, h_{i-1} = H\backslash x_1, \ldots, x_{i-1}$. Therefore h_i is simplicial in H_{i-1} for $i = 1, \ldots, n$.

Let H be a graph that has a partial perfect elimination ordering $h_1, \ldots, h_n$ such that H_n is a wheel on at least five rim vertices. If $h_1, \ldots, h_n$ is the empty ordering, then H is wheel on at least 5 rim vertices and clearly is a wheel extension. Thus assume that $n \geq 1$. Then the set of nontrivial neighbours of h_1 in $H = H_0$ form a clique cut set, separating H_1 and the subgraph of H_0 induced by $N_{H_0}(h_1)$ (recall that $h_1 \in N_{H_0}(h_1)$). Continuing on in this manner, we can create a clique cut set decomposition of H, whose atoms are the graph H_n and the complete subgraph of H_{i-1} induced by $N_{H_{i-1}}(h_i)$ for $i = 1, \ldots, n$. Therefore H must be a wheel extension.

□

Let H be a wheel extension. Then by Proposition 4.9, H has a partial perfect elimination ordering $h_1, \ldots, h_n$ such that H_n is a wheel on at least 5 rim vertices. Call such a partial perfect elimination ordering a *wheel extension ordering*; the graph in Figure 4.3 with the partial ordering as indicate is an example of wheel extension with a wheel extension ordering. Let $r_1 r_2 \ldots r_k r_1$ be the rim cycle of H_n and h the hub. Each vertex r_i is covered by h in H_n. Therefore $h_1, \ldots, h_n, r_1, \ldots, r_k, h$ is a dismantling ordering of H. We can summarize these observations in the following proposition.

Proposition 4.10. *Each wheel extension is dismantlable.*

Let H be a connected graph and let $\mathcal{E}$ be a subset of $E(H)$; for ease of discourse, we will use $V(\mathcal{E})$ to denote the set of vertices that are endpoints of the edges of $\mathcal{E}$. Then the pair $(H, \mathcal{E})$ is a *multi-wheel graph* if there exists a clique cut set decomposition $\mathcal{D}$ of H that consists of at least one wheel and possibly some complete graphs such that

i.) any clique cut set K used in decomposing H into the atoms of $\mathcal{D}$ is disjoint from $V(\mathcal{E})$.

ii.) each atom of $\mathcal{D}$ that is a wheel has exactly one edge e from $\mathcal{E}$, and e is a rim edge of that wheel.

iii.) each atom of $\mathcal{D}$ that is a complete graph is edge disjoint from $\mathcal{E}$.

Recall that we are assuming that the rim cycle of a wheel is always at least size 4.

Let H be the graph in Figure 4.4 and let $\mathcal{E} = \{e_1, e_2\}$. Note that the cliques $\{x, y\}$ and $\{y, z\}$ are disjoint from $V(\mathcal{E})$. We can use the clique cut set $\{z, y\}$ to decompose H into the subgraph induced by the clique $\{y, z, u\}$, call it H', and the subgraph induced by the two wheels, call it H''. We can then use the clique cut set $\{y, x\}$ to decompose H'' into the four wheel, call it W, and the five wheel, call it W'. The complete graph H' is edge disjoint from $\mathcal{E}$, and W and W' each contain exactly one edge from $\mathcal{E}$, and this edge is a rim edge. Thus $(H, \mathcal{E})$ is a multi-wheel graph.

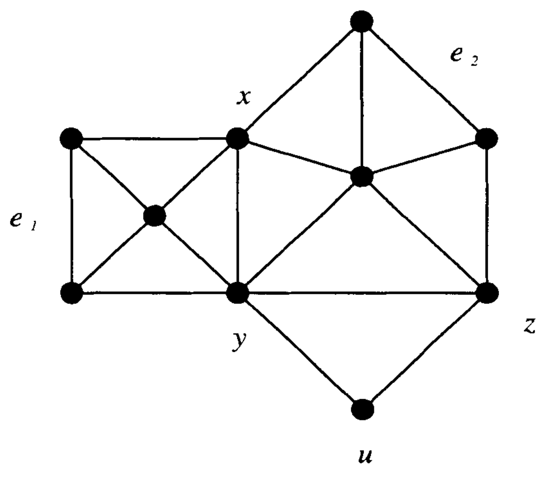


Figure 4.4: An example of a multi-wheel graph.

Note that by Corollary 4.3, if $(H, \mathcal{E})$ is a multi-wheel graph, then all the of clique cut set decompositions of H consist of wheels and complete graphs.

Proposition 4.11. *Let $(H, \mathcal{E})$ be a multi-wheel graph. Then*

i.) each induced wheel of H appears as an atom in any clique cut set decomposition of H.

ii.) each edge $e \in \mathcal{E}$ is contained in a unique induced wheel of H and so $|\mathcal{E}| = |\widetilde{\mathcal{D}}(H)|$.

iii.) each induced nontrivial cycle of H is the rim cycle of exactly one induced wheel of H.

iv.) if s is the endpoint of an edge in $\mathcal{E}$, then s is contained in a unique induced wheel W of H and the neighbourhood of s in H is contained in $V(W)$, i.e., $N_H(s) \subseteq V(W)$.

v.) $H \setminus \mathcal{E}$ is a chordal graph.

Proof. i. Let $\mathcal{D}$ be a decomposition of H that satisfies the requirements of the definition of a multi-wheel graph. In particular, $\mathcal{D}$ consists of wheels and complete graphs. By Corollary 4.3, $\widetilde{\mathcal{D}}(H)$ is exactly the set of induced wheels of H. As $\widetilde{\mathcal{D}}(H)$ is a subset of $\mathcal{D}$ by definition, statement i is true.

ii. By the definition of a multi-wheel graph, each wheel atom of $\mathcal{D}$, i.e., each element of $\widetilde{\mathcal{D}}(H)$, contains exactly one edge of $\mathcal{E}$, and so $|\widetilde{\mathcal{D}}(H)| \leq |\mathcal{E}|$. Suppose two wheels of $\widetilde{\mathcal{D}}(H)$, say W and W', share an edge $e \in \mathcal{E}$. At some point in the process of creating $\mathcal{D}$, W and W' must be separated; thus there exists induced subgraphs H' and H'' of H such that $W \subseteq H'$, $W \subseteq H''$ and $K = V(H') \cap V(H'')$ is a clique cut set used in the creation of $\mathcal{D}$. Obviously K must contain all vertices common to W and W'. Thus K must contain the end points of e, contradicting the requirement that $V(\mathcal{E}) \cap K = \emptyset$ in the definition of a multi-wheel graph. Therefore statement ii must be true.

iii. Let C be an induced nontrivial cycle of H. Clearly C has no clique cut set, and so C is an induced subgraph of some atom J in $\mathcal{D}$ by Lemma 4.2. The cycle C is nontrivial so J can't be complete. Thus J must be a member of $\widetilde{\mathcal{D}}(H)$, which is exactly the set of induced wheels of H, as noted previously. Therefore C must be the rim cycle of an induced wheel of H. If C is the rim cycle of two distinct wheel atoms of $\mathcal{D}$, then C is an induced subgraph of two distinct atoms of $\mathcal{D}$. Thus by Lemma 4.3, C must be a complete graph, which is a contradiction as C is not the 3-cycle. Therefore statement iii must hold.

iv. Let s be an endpoint of an edge $e \in \mathcal{E}$. Then by the definition of multi-wheel graphs, e is the rim edge of an induced wheel W in H. Suppose that s has a neighbour y in H that is not in $V(W)$. Let G be the last induced subgraph of H produced in creating $\mathcal{D}$ that contains both y and W. Then G must have a clique cut set K, where the subgraph $W \setminus K$ must be in one component of $G \setminus K$ and the vertex y must be in another. Thus s must be in K, which contradicts the definition of a multi-wheel graph and so all neighbours of s in H are contained in W. Therefore statement *iv* must hold.

v. Suppose that there exists an induced nontrivial cycle C in $H \setminus \mathcal{E}$. If C is also an induced cycle of H, then by statement *iii*, C is the rim cycle of a unique induced wheel of H. But by definition of a multi-wheel graph, there exists an edge of C that is in $\mathcal{E}$, a contradiction. Thus C is not an induced cycle of H. Hence there exist vertices s and s' in C that are not adjacent in C such that $ss' \in \mathcal{E}$. By statement *iv*, the vertex s is in the vertex set of a unique induced wheel W of H, and $N_H(s) \subseteq V(W)$. Consider the non-trivial neighbours of s. They are the vertex s', the other non-trivial neighbour of s on the rim cycle of W, call it r, and the hub h of W. As ss' is not an edge of C, r and h must be the nontrivial neighbours of s on the cycle C. As C is an induced nontrivial cycle in $H \setminus \mathcal{E}$, this implies that $rh \in \mathcal{E}$. By statement *ii*, W is the unique wheel that contains rh. This contradicts the definition of a multi-wheel graph as rh is not a rim edge of W. Therefore statement v is true.

□

The last statement of Proposition 4.11 shows how close to chordal multi-wheel graphs are; there exists one special edge in each induced cycle such that removing all these edges produces a chordal graph.

Let H be a graph and let $\mathcal{E}$ be subset of $E(H)$. Then $(H, \mathcal{E})$ is a *multi-wheel extension* if:

1. $(H, \mathcal{E})$ is a multi-wheel graph.

or

2. H has a simplicial vertex v with at most one neighbour in $V(\mathcal{E})$ such that $(H \setminus v, \mathcal{E})$ is a multi-wheel extension.

Thus, as opposed to multi-wheel graphs, if $(H, \mathcal{E})$ is a multi-wheel extension, then vertices in $V(\mathcal{E})$ may have neighbours outside the induced wheels that contain them.

Let H be the graph in Figure 4.5 and let $\mathcal{E} = \{e_1, e_2\}$. Let $H' = H \setminus \{x, y\}$. Then H' is the graph in Figure 4.4, and $(H', \mathcal{E})$ is a multi-wheel graph. Thus $(H', \mathcal{E})$ is also a multi-wheel extension. The vertex x is simplicial in $H \setminus y$, the vertex x has one neighbour in $V(\mathcal{E})$ and $(H \setminus \{x, y\}, \mathcal{E})$ is a multi-wheel extension. Thus $(H \setminus y, \mathcal{E})$ is a multi-wheel extension. The vertex y is simplicial in H, y has one neighbour in $V(\mathcal{E})$ and $(H \setminus y, \mathcal{E})$ is a multi-wheel extension. Therefore $(H, \mathcal{E})$ is a multi-wheel extension.

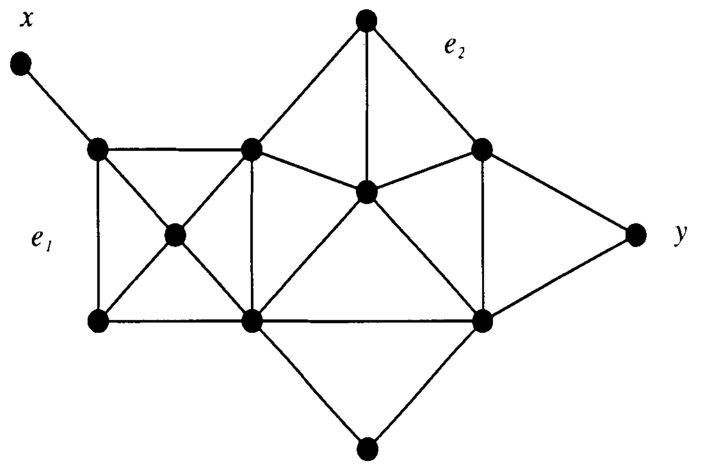


Figure 4.5: An example of a multi-wheel extension.

Proposition 4.12. *Let H be a graph and let $\mathcal{E}$ be a subset of $E(H)$. Then $(H, \mathcal{E})$ is a multi-wheel extension if and only if H has a partial perfect elimination ordering $h_1, \ldots, h_n$ such that*

i.) the neighbourhood of h_i in H_{i-1} contains at most one vertex from $V(\mathcal{E})$.

ii.) $(H_n, \mathcal{E})$ is a multi-wheel graph.

Proof. If $(H, \mathcal{E})$ is a multi-wheel graph, then the desired partial perfect elimination ordering is the empty ordering. Otherwise there exists a simplicial vertex, call it h_1, with at most one neighbour from $V(\mathcal{E})$ in $H = H_0$, such that $(H \setminus h_1, \mathcal{E}) = (H_1, \mathcal{E})$

is a multi-wheel extension. Either $(H_1, \mathcal{E})$ is a multi-wheel graph, or we can repeat the previous step. Therefore we can create a partial perfect elimination ordering $h_1, \ldots, h_n$ of H with properties *i* and *ii*.

Now assume that H is a graph with a subset $\mathcal{E}$ of edges such that properties *i* and *ii* hold. As $(H_n, \mathcal{E})$ is a multi-wheel graph, it is also a multi-wheel extension. Consider the graph H_{n-1}. The graph H_{n-1} contains the simplicial vertex h_n with at most one neighbour from $V(\mathcal{E})$ in H_{n-1} such that $(H_{n-1} \setminus h_n, \mathcal{E}) = (H_n, \mathcal{E})$ is a multi-wheel extension. Therefore $(H_{n-1}, \mathcal{E})$ is a multi-wheel extension. By repeating this argument, we prove that $(H_i, \mathcal{E})$ is a multi-wheel extension for $i = n-1, \ldots, 0$. Since $H_0 = H$, $(H, \mathcal{E})$ is a multi-wheel extension.

□

Note that Proposition 4.12 implies that multi-wheel extensions must be connected as wheel extensions are connected.

Proposition 4.13. *Let $(H, \mathcal{E})$ be a multi-wheel extension. If C is an induced nontrivial cycle in H, then C is the rim of a unique induced wheel in H. Moreover, $H \setminus \mathcal{E}$ is chordal.*

Proof. Let $h_1, \ldots, h_n$ be a partial perfect elimination ordering of H as described in Proposition 4.12. Then $(H_n, \mathcal{E})$ is a multi-wheel graph.

If C is an induced nontrivial cycle in H, then each vertex of C has non-adjacent neighbours, namely its neighbours on C. Therefore the vertex set of C and the set $\{h_1, \ldots, h_n\}$ are disjoint and so C must also be an induced subgraph of H_n. Then by Proposition 4.11, C is the rim of a unique induced wheel of H_n, and hence of H.

Suppose that C is an induced nontrivial cycle in $H \setminus \mathcal{E}$. If $V(C)$ is not disjoint from $\{h_1, \ldots, h_h\}$, then there exists a vertex h_i such that $C \subseteq H_{i-1}$. As h_i is simplicial in H_{i-1}, the nontrivial neighbours of h_i on C, say h and h', are adjacent in H_{i-1} and hence in H. Therefore, as C is nontrivial, $hh' \in \mathcal{E}$. This contradicts Proposition 4.12 as h_i is allowed to have at most one neighbour in $V(\mathcal{E})$. Hence $V(C)$ is disjoint from $\{h_1, \ldots, h_h\}$ and so C is an induced cycle of $H_n \setminus \mathcal{E}$. As $(H_n, \mathcal{E})$ is a multi-wheel graph, this contradicts Proposition 4.9.

□

In defining multi-wheel extensions, we have retained the property that made multi-wheel graphs nearly chordal; removing all special edges produces a chordal graph. Note that this property is lost if we allow a vertex in the partial perfect elimination ordering of Proposition 4.12 to have two neighbours that are endpoints of the special edges.

Proposition 4.14. *Let H be a multi-wheel extension. Then H is dismantlable.*

Proof. We will construct a dismantling ordering of H in three steps.

By Proposition 4.12, there exists a partial perfect elimination ordering $h_1, \ldots, h_p$ of H such that $(H_p, \mathcal{E})$ is a multi-wheel graph. The partial ordering $h_1, \ldots, h_p$ is the first part of the dismantling ordering of H; simplicial vertices are covered by all of their nontrivial neighbours.

Let $e = ss'$ be an edge of $\mathcal{E}$. By Proposition 4.11 , both s and s' are contained in a unique induced wheel W of H, and all the neighbours of s and s' in H are contained in W. Thus s and s' are covered by the hub of W. Let $\mathcal{E} = \{e_1, \ldots, e_k\}$ and let $e_k = s_k s'_k$. Then $s_1, s'_1, \ldots, s_k, s'_k$ is the second part of the dismantling ordering of H.

The graph $(H_p, \mathcal{E})$ is a wheel extension. Thus $H_p \setminus \mathcal{E}$ is chordal by Proposition 4.9 and so $H_p \setminus V(\mathcal{E})$ is clearly chordal. As $(H_p, \mathcal{E})$ is a wheel extension, H_p is connected. Hence, it is not hard to see that $H_p \setminus V(\mathcal{E})$ is both chordal and connected. Therefore $H_p \setminus V(\mathcal{E})$ admits a perfect elimination ordering, say $h_{p+1}, \ldots, h_n$, by Theorem 1.8. This is the last part of the dismantling ordering of H.

Thus, the dismantling ordering we have created for H is

$$h_1, \ldots, h_p, s_1, s'_1, \ldots, s_k, s'_k, h_{p+1}, \ldots, h_n.$$

□

Let H be graph such that either H is a wheel extension or $(H, \mathcal{E})$ is a multi-wheel extension for some $\mathcal{E} \subset E(H)$. We call such a graph a *wheeled graph.*

Theorem 4.7. *Let H be a wheeled graph. Then H is dismantlable.*

Proof. This follows from Proposition 4.10 and Proposition 4.14.

□

Proposition 4.15. *There exists a wheeled graph that does not admit a near unanimity function of any arity.*

The technique used is this proof is similar arguments used in [14] and [15].

Proof. Let H be the graph in Figure 4.6.

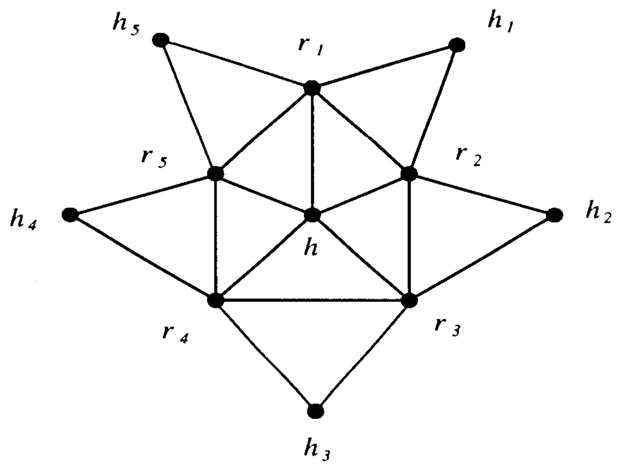


Figure 4.6: A wheeled graph that does not admit a near unanimity function of any arity.

The graph H is clearly a wheel extension. It is enough to prove that H does not admit a near unanimity function of arity $5k$ for all $k \geq 1$; by Lemma 1.5, if H does not admit a near unanimity function of arity $5k$, then H does not admit a near unanimity function of arity p, $3 \leq p \leq 5k$. In the proof of Theorem 3.8, we proved that H does admit have a near unanimity function of arity $5k$ by showing that H has a tree obstruction (T, ℓ) where T has $5k$ leaves; here, we present a direct proof.

Suppose there exists a near unanimity function $\eta : H^{5k} \longrightarrow H$ for some positive integer k. Define the vertex $\boldsymbol{v}_1$ of H^{5k} to be the vector

$$\boldsymbol{v}_1 = (\underbrace{r_1, \ldots, r_1}_{5k-1}, r_2)$$

and define the vertex $\boldsymbol{v}_{5k}$ of H^{5k} to be the vector

$$\boldsymbol{v}_{5k} = (r_5, \underbrace{r_1, \ldots, r_1}_{5k-1})$$

For $i = 2, \ldots, 5k-1$, define the vertex $\boldsymbol{v}_i$ of H^{5k} as follows:

$$\boldsymbol{v}_i = (\underbrace{r_i, \ldots, r_i}_{5k-i}, h, \underbrace{r_{i+1}, \ldots, r_{i+1}}_{i-1}),$$

where the indices of rim cycle vertices and the indices of the h_i's are to be interpreted modulo 5.

Since η is a near unanimity function, we must have $\eta(\boldsymbol{v}_1) = r_1$. Consider the image of $\boldsymbol{v}_2$ under η. In H^{5k}, the vertices $\boldsymbol{v}_2 = (\underbrace{r_2, \ldots, r_2}_{5k-2}, h, r_3)$ and $\boldsymbol{h}'_2 = (\underbrace{h_2, \ldots, h_2}_{5k-2}, h, h_2)$ are adjacent. By the definition of a near unanimity function, the image under η of the nearly unanimous vector $\boldsymbol{h}'_2$ is h_2. Thus $\eta(\boldsymbol{v}_2)$ must be adjacent to h_2 in H. Note also that $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are adjacent in H^{5k}. Hence, $\eta(\boldsymbol{v}_1) = r_1$ and $\eta(\boldsymbol{v}_2)$ must also be adjacent in H. The only vertex of H adjacent to both r_1 and h_2 is r_2. Thus we must have $\eta(\boldsymbol{v}_2) = r_2$.

Consider the vertex $\boldsymbol{v}_i$ of H^{5k}, $3 \leq i \leq 5k-1$. We may assume that $\eta(\boldsymbol{v}_j) = r_j$ for $1 \leq j < i$. The vertices $\boldsymbol{v}_{i-1}$ and $\boldsymbol{v}_i$ are adjacent in H^{5k} by construction, and by assumption $\eta(\boldsymbol{v}_{i-1}) = r_{i-1}$. Thus the image of $\boldsymbol{v}_i$ under η must be adjacent to r_{i-1}. The vertex $\boldsymbol{v}_i$ is also adjacent to the nearly unanimous vector $\boldsymbol{h}'_i = (\underbrace{h_i, \ldots, h_i}_{5k-i}, h, \underbrace{h_i, \ldots, h_i}_{i-1})$ in H^{5k}. The image of $\boldsymbol{h}'_i$ under η must be h_i. Thus the image of $\boldsymbol{v}_i$ under η must be adjacent to h_i. Therefore we can conclude that $\eta(\boldsymbol{v}_i) = r_i$.

By our arguments in the previous paragraph, we know that in particular $\eta(\boldsymbol{v}_{5k-1}) = r_{5k-1} = r_4$. The vertices $\boldsymbol{v}_{5k-1} = (r_4, h, \underbrace{r_5, \ldots, r_5}_{5k-2})$ and $\boldsymbol{v}_{5k} = (r_5, \underbrace{r_1, \ldots, r_1}_{5k-1})$ are adjacent in H^{5k}. For η to be a homomorphism, the image of $\boldsymbol{v}_{5k}$ must be adjacent to r_4.

Yet, by definition of a near unanimity function, the image of $\boldsymbol{v}_{5k}$ must be r_1, which is not adjacent to r_4. Thus we have a contradiction.

Therefore H does not admit a near unanimity function of arity $5k$ for all k.

□

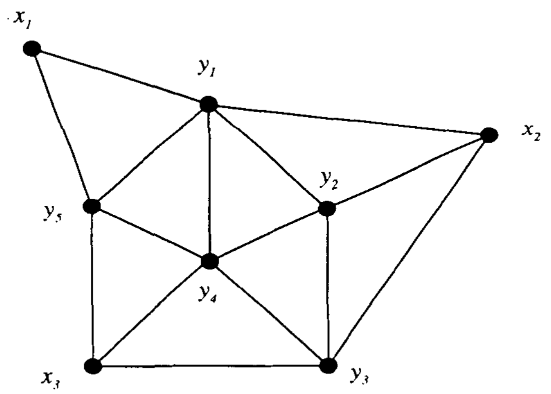


Figure 4.7: A graph which admits a near unanimity function that is not in the variety generated by wheeled graphs.

Notice that the graph in Figure 4.7 has the hole $\hat{f}$ with domain $D_{\hat{f}} = \{x_1, x_2, x_3\}$ where $\hat{f}(x_1) = 2$ and $\hat{f}(x_2) = \hat{f}(x_3) = 1$.

Lemma 4.4. *Let H be the graph in Figure 4.7 and let $\hat{f}$ be the hole on H as described above. If there exists a graph J and an onto homomorphism $\gamma : H \rightarrow J$ such that γ is a separating map of $\hat{f}$, then J is isomorphic to H.*

Proof. We will prove that γ is an isomorphism by showing that $\gamma(h) \neq \gamma(h')$, whenever $h \neq h'$. This is sufficient as we already know that γ is an onto homomorphism.

Let f^* be the distance constraint on J with domain $\gamma(D_{\hat{f}})$ where

$$f^*(s') = \min_{\gamma(s)=(s')} \hat{f}(s).$$

Then as γ is a separating map of $\hat{f}$, the distance constraint f^* is infeasible on J.

Let $X_1 = \{y_3\}$, $X_2 = \{y_4, y_5\}$ and $X_3 = \{y_1, y_2\}$. Then for each $z \in X_i$, z is an x_i relaxed filler such that $d_H(z, x_i) = f(x_i) + 1$, $i = 1, 2, 3$. Thus $d_J(\gamma(z), \gamma(x_i)) \leq f(x_i) + 1$ for $z \in X_i$ and $i = 1, 2, 3$. If there exists $i \in \{1, 2, 3\}$ and a vertex $z \in X_i$ such that $d_J(\gamma(z), \gamma(x_i)) \leq f(x_i)$, then $\gamma(z) \in F_J(f^*)$, contradicting the infeasibility of f^*. Thus $d_J(\gamma(z), \gamma(x_i)) = f(x_i) + 1$ for $z \in X_i$ and $i = 1, 2, 3$. Thus if hh' is a non-loop edge on a shortest $z - x_i$ path in H, $z \in X_i$, then $\gamma(h) \neq \gamma(h')$, $i = 1, 2, 3$. It is easy to check that all the non-loop edges lie on a shortest $z - x_i$ path in H for some $z \in X_i$ and some $i \in \{1, 2, 3\}$.

□

Proposition 4.16. *There exists a graph that admits a near unanimity function that is not in the variety of wheeled graphs.*

Proof. Let H be the graph in Figure 4.7. If we can prove that H^2 dismantles to its subgraph induced by the constant vertices, then H admits a near unanimity function by Theorem 1.6.

Note that in H, the vertex x_1 is covered by y_1, the vertex x_2 by y_2 and the vertex x_3 by y_4. Thus, in H^2, the vertex (x_1, z) is covered by (y_1, z) for all $z \in V(H)$, the vertex (x_2, u) is covered by (y_2, u) for all $u \in V(H)$ and the vertex (x_3, v) is covered by (y_4, v) for all $v \in V(H)$. Thus all vertices of the form (x_i, x_j) in H^2 are covered by some vertex. Let H' be the graph we obtain from H^2 be removing all vertices of the form (x_i, x_j), $i \neq j$. Clearly H^2 dismantles to H'. Let (y_i, y_j) be a vertex of H' such that (y_i, y_j) is not adjacent to $\overline{x}_k$, $k = 1, 2, 3$. Then (y_i, y_j) is covered by (y_4, y_4), when $i \neq j$. The set of all vertices (y_i, y_j) such that (y_i, y_j) is not adjacent to $\overline{x}_k$ for $k = 1, 2, 3$ is $\{(y_1, y_3), (y_1, y_4), (y_3, y_1), (y_4, y_1), (y_2, y_4), (y_2, y_5), (y_4, y_2), (y_5, y_2), (y_3, y_5), (y_5, y_3)\}$. Let H'' be the graph we obtain from H' by removing these vertices. Clearly H', and thus H^2, dismantles to H''. Then, by inspection, H'' dismantles to the graph induced by the constant vertices via the following partial ordering:

$$(y_1, y_5), (y_1, y_5), (y_4, y_5), (y_5, y_4), (y_1, y_2), (y_2, y_1), (y_2, y_3), (y_3, y_2), (y_3, y_4), (y_4, y_3).$$

Therefore H^2 also dismantles to its subgraph induced by the constant vertices.

Now we will show that H is not in the variety generated by wheeled graphs. Clearly H is not a wheeled graph itself; while x_1 is simplicial, H_1 is neither a wheel extension nor a multi-wheel graph. Thus, suppose that H is a retract of $G = \Pi_{i=1}^{k} J_i$, where J_i is a wheeled graph and J_i is not isomorphic to H, for $i = 1, \ldots, k$. Let π_i be the projection from G onto J_i. Let γ_i be the restriction of π_i to H. Note that we may assume $\gamma_i : H \longrightarrow J_i$ is onto. As H is a retract of G, we must have $F_G(f) = \emptyset$ for all holes f on H. Thus for each hole f on H, there must exist an index i such that $\pi_i : G \longrightarrow J_i$ and hence $\gamma_i : H \longrightarrow J_i$, is a separating map of f. Let $\hat{f}$ be the hole on H from Lemma 4.4. Without loss of generality, assume that $\gamma_1 : H \longrightarrow J_1$ is a separating map for $\hat{f}$. Then by Lemma 4.4, J_1 is isomorphic to H, which is a contradiction as H is not a wheeled graph.

□

Theorem 4.8. *The variety generated by wheeled graphs and the set of connected graphs that admit near unanimity function are not comparable.*

□

Theorem 4.9. *The variety generated by connected chordal graphs is a subset of the variety generated by wheeled graphs.*

Proof. We will prove the theorem by showing any connected chordal graph is the retract of some wheel extension; the class of connected chordal graphs being a subset of the variety generated by wheeled graphs implies that the variety generated by chordal graphs is a subset of the variety generated by wheeled graph.

Let H be a connected chordal graph. If H is a single vertex or a single edge, then clearly H is the retract of a wheel with 5 rim vertices. Thus assume H has at least 3 vertices. Let r_1r_2 be an edge of H. As $\{r_1, r_2\}$ is clearly a convex set, there exists a partial perfect elimination ordering $h_1, \ldots, h_n$ of H such that $V(H_n) = \{r_1, r_2\}$ by Theorem 2.3. Let G be the supergraph of H with vertex set $V(G) = V(H) \cup \{r_3, r_4, r_5, h\}$ where $\{r_1, \ldots, r_5, h\}$ induces a wheel in G with rim cycle $r_1r_2r_3r_4r_5r_1$ and hub h. Assume moreover that r_1 and r_2 are the only vertices of H that have neighbours in $G \setminus H$. Note that $h_1, \ldots, h_n$ is a partial perfect elimination ordering of G such that $G \setminus \{h_1, \dots, h_n\}$ is a wheel with 5. Thus G is a wheel extension by Proposition 4.9. Define the map $\theta : G \to H$ by

$$\theta(g) = \begin{cases} g & \text{if } g \in V(H) \\ r_1 & \text{otherwise.} \end{cases}$$

Clearly θ fixes each vertex of H. To prove that θ is a retraction, we only need to prove that θ is a homomorphism. Let gg' be an edge of G. If g and g' are both in $V(H)$ or both not in $V(H)$, it is clear that $\theta(g)\theta(g')$ is an edge H. Thus assume that $g \in V(H)$ and $g' \notin V(H)$. As r_1 and r_2 are the only vertices of H that have neighbours in $G \setminus H$, we must have that g is r_1 and r_2. As $\theta(g') = r_1$, $\theta(g)\theta(g')$ is an edge of H.

□

In the next two sections, we will show that wheel extensions and multi-wheel extension are each in $\mathcal{AR}_C$ by using modified forms of Algorithm 3.1 [28]. Thus we will conclude that the variety generated by wheeled graphs is contained in $\mathcal{AR}_C$ as $\mathcal{AR}_C$ is a variety by Theorem 4.1. Note that as the variety generated by connected chordal graphs is contained in the variety generated by wheeled graphs by Theorem 4.9, this is another proof that connected chordal graphs are in $\mathcal{AR}_C$.

Before proving that wheeled graphs are in $\mathcal{AR}_C$, we will review why connected chordal graphs are in $\mathcal{AR}_C$, stressing the useful qualities shared by connected chordal graphs and wheeled graphs.

Recall the proof we presented of Theorem 4.4, the theorem stating that connected chordal graphs are in $\mathcal{AR}_C$. Let H be a connected chordal graph with perfect elimination ordering $h_1, \dots, h_n$ and let G be a connected supergraph G of H such that ACR Algorithm succeeds on H and G. Let L' be the final lists produced by ACR Algorithm on H and G. We proved that we could find a retraction from G to H by only applying action $*$ of Algorithm 3.1 to (G, L') and $h_1, \dots, h_n$.

We would like to use a similar technique to prove that if H is a wheeled graph and if G is a connected supergraph of H such that the ACR Algorithm succeeds on H and G, then there exists a retraction from G to H. Thus our first step will be to prove that the ACR Algorithm produces connected lists when H is a wheeled graph. Next, we will consider wheel extensions and multi-wheel extension separately. In each case, we will create a partial ordering that contains most of the vertices of the graph under consideration and is close to a perfect elimination ordering. We will use action $*$ of Algorithm 3.1 to process these vertices. There will be a special operation or two to deal with the vertices that are not in the partial ordering, i.e., the vertices that prevent the graph from being chordal.

Lemma 4.5. *Let H be a wheeled graph. Let W be an induced wheel in H and let x and y be non-adjacent rim vertices of W. Then every path between x and y has an internal vertex in $V(W)$.*

Proof. Suppose not. Let H be a wheeled graph such that the lemma is false for H, but is true for any wheeled subgraph H' of H. Let W be an induced wheel of H such that W has non-adjacent rim vertices x and y that are connected by a path P in H that is internally disjoint from W. Moreover, assume that P is chordless. Consider the subgraph J of H induced by $V(W) \cup V(P)$. Note that J is neither a complete graph nor a wheel.

We claim that J has no clique cut set. Note that W and P are both induced subgraphs of J, and neither W nor P have a clique cut set. Thus if K is a clique cut set of J, $W \setminus K$ must in an component of $J \setminus K$ and $P \setminus K$ must be in another. This implies that the common vertices of W and P, namely x and y, must be in K. This is impossible as x and y are not adjacent.

As J is a connected induced subgraph of H with no clique cut set, J must be a subgraph of at least one atom of any clique cut set decomposition of H by Lemma 4.2. We noted earlier that J is neither a wheel nor a complete graph. Therefore H cannot be a wheel extension or multi-wheel graph. Therefore $(H, \mathcal{E})$ is a multi-wheel extension for some $\mathcal{E} \subset E(H)$. As we have stated that $(H, \mathcal{E})$ is not a multi-wheel graph, there must exist a simplicial vertex v of H such that $(H \setminus v, \mathcal{E})$ is a multi-wheel extension. Thus H has a simplicial vertex such that $H \setminus v$ is a wheeled graph. By the minimality of H, v must be a vertex of J ; J can't be a subgraph of $H \setminus v$. Any vertex of $V(W)$ in J has non-adjacent neighbours as W is an induced subgraph of J and W is a wheel. Therefore v must be a vertex internal to P. As we assumed that P was chordless, this implies that P has length 2 and that the endpoints of P, x and y, are adjacent, a contradiction.

□

Lemma 4.6. *Let H be a wheeled graph and let W be an induced wheel in H. Let x and y be two non-adjacent rim vertices of W. Let h be the hub of W. Let $B \subseteq V(H)$ be connected. Then if B supports x and y, B must also support h.*

Proof. Suppose that x and y have neighbours u and v in B. The vertices u and v are connected by a path P such that $V(P) \subseteq B$ as B induces a connected subgraph in H. We may assume that P is chordless and that x and y are only adjacent to u and v on P, respectively. By Lemma 4.5, any path from x to y must have an internal vertex in W. Thus P must contain a neighbour of h since xPy is a path from x to y. Therefore B supports h.

□

Proposition 4.17. *Let H be a wheeled graph, and let G be a connected supergraph of H. The lists produced by the ACR Algorithm applied to H and G are connected.*

Proof. We will analyze what occurs when the ACR Algorithm is applied to H and G.

We first initialize the list of every vertex g of G to either $\{g\}$ if $g \in V(H)$ or $V(H)$ otherwise. Certainly each list created initially is connected. We will prove that this property is maintained as we process the edges of G.

Assume that the list of every vertex of G is connected in all iterations before iteration i, and let gg' be the edge to be processed in iteration i. Note that only the lists of g and g' may change when we process the edge gg'. Hence all that is needed is to prove that the lists of g and g' retain their connectivity. This is trivially true if the lists of g and g' become empty in iteration i, at which point the algorithm would stop with failure. Hence we may assume that some vertex of $L(g)$, the list of g the beginning of iteration i, has support in $L(g')$, the list of g' beginning of iteration i. Let S be the vertices of $L(g)$ that have support in $L(g')$. If S is connected, there is nothing to prove. Thus suppose that S is not connected. Let S_1 and S_2 be distinct connected subsets of S. Hence there exists a path P in H whose vertices are contained in $L(g)$, where P is internally disjoint from S with one endpoint x in S_1 and the other endpoint y in S_2. We may assume that P is chordless. Thus x and y have support in $L(g')$, but no internal vertex of P does. Note that P contains at least 3 vertices as P has at least one internal vertex. Let u and v be the neighbours of x and y respectively in $L(g')$ such that the distance between u and v in the subgraph of H induced by $L(g')$ is minimum; it is possible that $u = v$. Let P' be a path from u to v in $L(g')$. Then $V(P) \cup V(P')$ induces a nontrivial cycle C in H. By Proposition 4.8 and Proposition 4.13, C is the rim of an induced wheel W in H. Let h be the hub of W. The vertex h obviously has support in $L(g')$ and it is a common neighbour of x and y. Thus h is not in $L(g)$. As $L(g)$ contains at least three vertices, the original list of g was $V(H)$ and so h must have been removed from the list of g in some iteration previous to the current one. Hence g must have a neighbour g'' such that the edge gg'' was processed in iteration j, $1 \leq j < i$ where the list of g'' at the beginning of iteration j had support for x and y, but not h. This contradicts Lemma 4.6 since we have assumed that all lists were connected in all iterations previous to iteration i. Therefore the vertices in $L(g)$ that have support in $L(g')$ form connected set, and vice versa.

□

4.4.1 Wheel Extensions

In this section we will prove that wheel extensions are in $\mathcal{AR}_C$ by using a modified version of Algorithm 3.1.

Let H be a wheel extension. By Proposition 4.9, H has a wheel extension ordering $h_1, \ldots, h_n$. Thus H_n is a wheel on at least 5 rim vertices. Let $r_1 r_2 \ldots r_k r_1$ be the rim cycle of H_n and let h be the hub of this wheel. As H_n is the unique induced wheel of H, we will also call $r_1 r_2 \ldots r_k r_1$ the *rim cycle* of H and h the *hub* of H. Note that $\{h_1, \ldots, h_n\} \cup \{r_1, \ldots, r_k\} \cup \{h\}$ is a partition of the vertex set of H.

Algorithm 4.3.

```
Input: A graph $G$.
     : A subgraph $H$ of $G$ which is a wheel extension with a wheel extension
       ordering $h_1, \ldots, h_n$, rim cycle $r_1, \ldots, r_k$ and hub $h$.
     : A non-empty list $L(g)$ for each $g \in V(G)$ produced by the ACR
       Algorithm applied to $H$ and $G$.
Task: To find a retraction from $G$ to $H$.
Action: Process the vertices $h_1, \ldots, h_n$. For each vertex $g$ of $G$
            * if $h_i \in L(g)$ and $|L(g)| \geq 2$, then remove $h_i$ from $L(g)$
              (do this for $i = 1, \ldots, n$).
                  $L(g) \leftarrow L(g) \setminus \{h_i\}$
      : Process the vertex $\{h\}$. For each vertex $g$ of $G$
            † if $h \in L(g)$ then remove all vertices but $h$ from $L(g)$.
                  $L(g) \leftarrow \{h\}$
      : Process the vertices $r_1, \ldots, r_k$. For each vertex $g$ of $G$
            †† if $r_i, r_{i+1} \subseteq L(g)$ for some $i$, where the indices are modulo
               $k$, then remove vertex $r_i$ from $L(g)$.
                  $L(g) \leftarrow L(g) \setminus \{r_i\}$
      : Create a retraction $\theta : G \rightarrow H$ by setting $\theta(g)$ to be the single vertex
        in $L(g)$.
```

Consider the lists at each stage of Algorithm 4.3. The lists provided by the ACR Algorithm applied to H and G are subsets of $V(H)$. Once we process the vertex h_1, each list is either a single vertex of $V(H)$ or is a subset of $V(H_1)$; note that processing the vertices of $h_1, \dots, h_n$ will never produce an empty list. In general, once we have processed vertex h_i, $1 \leq i \leq n$, all lists are singletons or subsets of $V(H_i)$. Next we process the the hub h of H by replacing any list that contains h with the list $\{h\}$. Thus once we have processed the vertices $h_1, \dots, h_n, h$, each list is a singleton or consists solely of rim vertices of H.

Theorem 4.10. *Let H be wheel extension, and let G be a connected supergraph of H. If the ACR Algorithm succeeds when applied to H and G, then Algorithm 4.3 will produce a retraction from G to H.*

Proof. As H is a wheel extension, H has a wheel extension ordering, a partial perfect elimination ordering $h_1, \dots, h_n$ such that H_n is wheel on at least 5 rim vertices by Proposition 4.9. Let $r_1 r_2 \dots r_k r_1$ be the rim cycle of H_n and h the hub. Let G be a connected supergraph of H such that the ACR Algorithm succeeds when applied to H and G. Thus the ACR Algorithm produces a non-empty list $L(g) \subseteq V(H)$ for each vertex g of G, and the lists are arc consistent. In addition, each list is connected by Proposition 4.17. We claim that list connectivity and arc consistency are maintained throughout the execution of Algorithm 4.3 applied to (G, L) and H.

For the rest of this proof, $L(g)$ will be the list of $g \in V(G)$ at the beginning of the action in question. Process the vertices of the wheel extension ordering $h_1, \dots, h_n$ of H. Assume that the claim is true before we process vertex h_i, $i \geq 1$.

Suppose that we are about to perform action $*$ on $L(g)$. Thus $|L(g)| \geq 2$. Since h_i is simplicial in H_{i-1} and since $L(g) \subseteq V(H_{i-1})$, action $*$ on $L(g)$ will perserve list connectivity as in the proof of Theorem 3.3 and action $*$ will preserve arc consistency as in the proof of Theorem 4.4.

Thus, once we have processed the vertices of the wheel extension ordering, all lists are still connected and arc consistent.

Process the hub h of H. Suppose that h is in $L(g)$ for some $g \in V(G)$ and that

$|L(g)| \geq 2$. Clearly $\{h\}$ is connected, thus we only need to prove that $\{h\}$ supports all the vertices $L(g)$ supports. Let g' be a neighbour of g in G. If the list of g', $L(g')$, is a singleton, then clearly arc consistency is maintained as this single vertex must be adjacent to all vertices of $L(g)$ and in particular is adjacent to h. Thus suppose that $L(g')$ is not singleton. Then $L(g') \subseteq V(H_n)$ by our comments following Algorithm 4.3. Recall that h is the hub of H_n, the unique induced wheel in H. Hence h is adjacent to every vertex of $L(g')$, and so arc consistency is preserved after h has been processed.

Therefore all lists of the vertices of G are connected and arc consistent after we have process $h_1, \ldots, h_n, h$.

Consider the lists at this time. Suppose that $|L(g)| \geq 2$, for some $g \in V(G)$. Thus $h \notin L(g)$. The vertex h is never removed from a list by any action of Algorithm 4.3, and so h must have been removed from the list of g in the running of the ACR Algorithm applied to H and G. Lemma 4.6 tells us that if a connected list doesn't support h, it can't support non-adjacent rim vertices of the wheel H_n. In the proof of Proposition 4.17, we demonstrated that the lists of the vertices of G were connected initially and remained so throughout all iterations of the ACR Algorithm. Thus if h was removed from the list of g in some iteration of the ACR Algorithm, the list of g at the end of that iteration did not contain non-adjacent rim vertices of H_n. As $L(g)$ is a subset of this previous list, $L(g)$ can not contain non-adjacent rim vertices of H_n. As noted in our comments after Algorithm 4.3, $L(g)$ contains only rim vertices of H. Hence $L(g)$ must consist of two adjacent rim vertices of H_n.

Therefore, for each list $L(g)$, $g \in V(G)$, either $L(g) = \{r_i, r_{i+1}\}$ or $L(g)$ is a singleton.

Lastly, process the rim vertices of H, $r_1, \ldots, r_k$. Let g and g' be adjacent vertices in G with non-singleton lists. Without loss of generality we may assume that $L(g) = \{r_p, r_{p+1}\}$ and $L(g') = \{r_q, r_{q+1}\}$, with $1 \leq p \leq q \leq k$ and the indices modulo k. Suppose that r_{p+1} and r_{q+1} aren't adjacent. As $L(g)$ and $L(g')$ are arc consistent, r_{p+1} must be adjacent to r_q and r_{q+1} must be adjacent to r_p. Recall that $r_1 r_2 \ldots r_k r_1$ is an induced cycle in H. Thus if r_p is adjacent to r_q, then $q = p + 1$ implying that r_{p+1} is adjacent to r_{q+1}. This contradicts our assumption. Hence $r_p r_{p+1} r_q r_{q+1} r_p$ is an induced 4-cycle in $H_n \setminus h$. But this is impossible as by the definition of a wheel extension, the rim cycle of H must be of size at least 5. Hence the vertices r_{p+1} and r_{q+1} are adjacent and so list connectivity and arc consistency are maintained once the rim vertices of H have been processed.

Thus, after processing all the vertices of $V(H)$, the lists of the vertices in G are still connected and arc consistent. Moreover, all lists are singletons. Therefore the map ϕ created in the last step of Algorithm 4.3 is well defined and it is a homomorphism. The function ϕ is in fact a retraction as the initial list assigned to $g \in V(H)$ by ACR Algorithm was $\{g\}$.

□

Corollary 4.4. *Each wheel extension is in $\mathcal{AR}_C$.*

□

4.4.2 Multi-wheel Extensions

In this section, we will prove that multi-wheel extensions are in $\mathcal{AR}_C$ by using a modified version of Algorithm 3.1. But first we need some additional structure lemmas which highlight the necessity of the 'special' edges in multi-wheel extensions.

Lemma 4.7. *Let $(H, \mathcal{E})$ be a multi-wheel extension and let s, s' be distinct vertices of $V(\mathcal{E})$. Then s and s' are contained in unique induced wheels W and W', respectively, in H, where $W = W'$ if and only if s and s' are adjacent. Moreover, if there exists a path P of length 2 or more from s to s', then P has internal vertices in both $V(W)$ and $V(W')$.*

Proof. By Proposition 4.12, there exists a partial perfect elimination ordering $h_1, \ldots, h_n$ of H such that $(H_n, \mathcal{E})$ is a multi-wheel graph and $|N_{H_{i-1}}(h_i) \cap V(\mathcal{E})| \leq 1$ for $i = 1, \ldots, n$. By Proposition 4.11, there exists unique induced wheels W and W' in H_n, such that $s \in V(W)$ and $s' \in V(W')$. Since H_n is an induced subgraph of H and as vertices of induced wheels can't be simplicial, W and W' are the unique induced wheels of H such that $s \in V(W)$ and $s' \in V(W')$.

We will use induction n to prove that any path P in H of length at least 2 between s and s' has internal vertices from $V(W)$ and $V(W')$. If $(H, \mathcal{E})$ is a multi-wheel graph, i.e., $n = 0$, s and s' only have neighbours in W and W' by Proposition 4.11. Thus we may assume that $n \geq 1$. Suppose there exists a path P in H with no internal vertices from W. Then we must have that h_1 is a vertex of P, otherwise $P \subseteq V(H_1)$. Let x and y be the neighbours of h_1 on P. Then $xy \in E(H)$ as h_1 is simplicial in H. Let P' be the path we get from P by replacing xh_1y with xy. Then $P' \subseteq V(H_1)$. Thus by induction, P' must be a path of length 1. This implies that, without loss of generality, $x = s$ and $y = s'$. Hence h_1 has two neighbours in $V(E)$ in H, contradiction. Therefore at least one internal vertex of P is from $V(W)$. Similarly, at least one internal vertex of P is from $V(W')$.

□

Lemma 4.8. *Let $(H, \mathcal{E})$ be a multi-wheel extension and let s, s' be distinct vertices of $V(\mathcal{E})$. Then s and s' are contained in unique induced wheels W and W', respectively, in H. Let h be the hub of W and h' the hub of W'. Let B be a connected subset of $V(H)$. Then if B supports s and s', it must also support h and h'.*

Proof. The existence induced wheels W and W' in H that contain s and s', respectively, follows from Lemma 4.7.

As B supports s and s', there exist vertices $u, v \in B$ adjacent to s and s' respectively. By assumption, there is a path P in H with $V(P) \subseteq B$ from u to v. Then sPs' is a walk from s to s'. By Lemma 4.7, P must contain vertices from W and from W', with the possibility that $W = W'$. Therefore B must also support h and h', the hubs of W and W'.

□

Let $(H, \mathcal{E})$ be a multi-wheel extension. Recall the dismantling ordering of H that we created in Proposition 4.14. This dismantling ordering consisted of three sections that we constructed separately. The first section, $h_1, \ldots, h_p$ is a partial perfect elimination ordering of H such that $(H_p, \mathcal{E})$ is multi-wheel graph. The third section of the dismantling ordering, $h_{p+1}, \ldots, h_n$, is a perfect elimination ordering of $H_p \setminus V(\mathcal{E})$. Call the partial ordering of the vertices of H $h_1, \ldots, h_p, h_{p+1}, \ldots, h_n$ a *multi-wheel* *extension ordering* of H. By construction, $\{h_1, \ldots, h_p\} \cup \{h_{p+1}, \ldots, h_n\} \cup V(\mathcal{E})$ is a vertex partition of H.

Algorithm 4.4.

```
Input:  A graph $G$.
     :  A subgraph $H$ of $G$ such that $(H, \mathcal{E})$ is a multi-wheel extension,
        for some $\mathcal{E} \subseteq E(H)$, with a multi-wheel extension ordering
        $h_1, \ldots, h_p, h_{p+1}, \ldots, h_n$.
     :  A non-empty list $L(g)$ for each vertex $g \in V(G)$ produced by the
        ACR Algorithm applied to $H$ and $G$.
Task:   To find a retraction from $G$ to $H$.
Action: Process the vertices $h_1, \ldots, h_p$. For each vertex $g$ of $G$
            * if $h_i \in L(g)$ and $|L(g)| \geq 2$, then remove $h_i$ from $L(g)$ (do
              this for $i = 1, \ldots, p$).
                    $L(g) \leftarrow L(g) \setminus \{h_i\}$
     :  Process the vertices of $V(\mathcal{E})$. For each vertex $g$ of $G$
            ‡ if $|L(g)| \geq 2$, remove all vertices of $V(\mathcal{E})$ from the list of
              $g$.
                    $L(g) \leftarrow L(g) \setminus V(\mathcal{E})$
     :  Process the vertices $h_{p+1}, \ldots, h_n$. Repeat action * for $i = p+1, \ldots, n$.
     :  Create a retraction $\theta : G \rightarrow H$ by setting $\theta(g)$ to be the single vertex
        in $L(g)$.
```

As with Algorithm 4.3, once we have processed the vertex h_i, $1 \leq i \leq p$, each list is either a singleton or a subset of $V(H_i)$. After processing the vertices of $V(\mathcal{E})$, each list of size at least two is disjoint from $V(\mathcal{E})$. Thus after processing the vertex h_i, $p+1 \leq i \leq n$, all non-singleton lists are subsets of $V(H_i) \setminus V(\mathcal{E})$.

Theorem 4.11. *Let $(H, \mathcal{E})$ be a multi-wheel extension and let G be a connected supergraph of H. If the ACR Algorithm succeeds when applied to H and G, then Algorithm 4.4 will produce a retraction from G to H.*

Proof. As H is a multi-wheel extension, H has a multi-wheel extension ordering $h_1, \ldots, h_p, h_{p+1}, \ldots, h_n$. Let G be a connected supergraph of H such that the ACR Algorithm succeeds when applied to H and G. Thus the ACR Algorithm produces a non-empty list $L(g) \subseteq V(H)$ for each vertex g of G, and the lists are arc consistent. By Proposition 4.17, the lists produced by the ACR Algorithm for the vertices in G are connected. We will prove that arc consistency and list connectivity are retained throughout the execution of Algorithm 4.4, with one exception; arc consistency may temporarily be lost in the midst of action ‡, but it will be regained once that action has been applied to all vertices of G.

List connectivity and arc consistency are maintained as we process the vertices $h_1, \ldots, h_p$, as in the proof of Theorem 4.10.

Let $L(g)$ denote the list of $g \in V(G)$ after we have processed the vertices $h_1, \ldots, h_p$, but before we perform action ‡ on any vertex of G. As we have just commented, these lists are connected and arc consistent. By our comments following Algorithm 4.4, if $|L(g)| \geq 2$ for some $g \in V(G)$, then $L(g) \subseteq V(H_p)$. Since $h_1, \ldots, h_p, h_{p+1}, \ldots, h_n$ is a multi-wheel extension ordering of H, $(H_p, \mathcal{E})$ is a multi-wheel graph. Thus by Proposition 4.11, for any vertex $s \in V(\mathcal{E})$, the only neighbours of s in H_p are those it has in the unique induced wheel of H_p that contains s.

Now we wish to process the vertices of $V(\mathcal{E})$. Let g be a vertex of G such that $L(g)$ contains vertices from $V(\mathcal{E})$ and let $\{s_1, \ldots, s_k\} = L(g) \cap V(\mathcal{E})$.

Suppose that $k \geq 2$. We know that arc consistency holds before we process the vertices of $V(\mathcal{E})$. Thus, if g' is a neighbour of g in G, then $L(g')$ must support $\{s_1, \ldots, s_k\}$. By Lemma 4.8, $L(g')$ must also support $\{h_{i_1}, \ldots, h_{i_k}\} \in V(H_p) \setminus V(\mathcal{E})$, where h_{i_j} is the hub of the induced wheel in H_p that contains s_j. As this support exists now, it must also have existed when the ACR Algorithm was applied to H and G. Therefore we must have $\{h_{i_1}, \ldots, h_{i_k}\} \subseteq L(g)$ now. Thus $L(g) \setminus V(\mathcal{E})$ is non-empty and in particular, as h_{i_j} covers all the neighbours of s_i in H_p, $L(g) \setminus V(\mathcal{E})$ is connected. In addition, if g' is a neighbour of g in G and if a vertex y of $L(g')$ had support in $L(g)$, it will still have support in $L(g) \setminus V(\mathcal{E})$; if y is the sole vertex of $L(g')$, then y is adjacent to all vertices of $L(g)$ and if $|L(g')| \geq 2$ and y is adjacent to s_i, then y is also adjacent to h_{i_k}. Therefore list connectivity and arc consistency are maintained when $k \geq 2$.

Now suppose that $k = 1$. By definition, there exists a vertex $s \in V(\mathcal{E})$ such that $s_1 s \in \mathcal{E}$. Moreover, there is a unique induced wheel W of H_p such that ss_1 is a rim edge of W by Proposition 4.11. As noted above, the only neighbours of s_1 in H_p are those it has in W. These would be s_1 itself, the neighbours of s_1 on the rim cycle of W, one of which is s, and the hub of W. Hence s_1 has two neighbours outside $V(\mathcal{E})$, and they are adjacent. As $L(g)$ is connected, s_1 has a neighbour x in $L(g)$. Moreover, since s_1 is the only vertex from $V(\mathcal{E})$ in $L(g)$, $x \in V(H_p) \setminus V(\mathcal{E})$, and so $L(g) \setminus V(\mathcal{E})$ is non-empty. Let y be the other neighbour of s_1 in $V(H_p) \setminus V(\mathcal{E})$. As remarked before, x and y are adjacent. Thus $L(g) \setminus s_1$ is connected as y is the only other possible neighbour of s_1 in $L(g)$. Let g' be a neighbour of g in G. As arc consistency has held till we process the vertices of $V(\mathcal{E})$, s_1 has a neighbour in $L(g')$. If this neighbour is x or y, then clearly it has support in $L(g) \setminus \{s_1\}$. Thus suppose this neighbour of s_1 is s', where $s' \in V(\mathcal{E})$. If $L(g') = \{s'\}$, then as arc consistency has held till now, s' is adjacent to all vertices of $L(g)$. If $|L(g')| \geq 2$, then s' will be removed from $L(g')$ when we apply the current action to $L(g')$. Therefore it doesn't matter if s_1 is the only neighbour of s' in $L(g)$. Thus list connectivity is maintained when $k = 1$, and if arc consistency is temporarily lost, it will be regained once action ‡ has been applied to all the vertices of G.

List connectivity and arc consistency are maintained as we process the vertices $h_{p+1}, \ldots, h_n$ for the same reasons as when we processed the vertices $h_1, \ldots, h_p$.

By our comments following Algorithm 4.4, all the lists are now singletons. Therefore the map ϕ created in the last step of Algorithm 4.4 is well defined. This map must be a homomorphism as we have just proved that the final lists produced by Algorithm 4.4 are arc consistent. The function ϕ is in fact a retraction as the initial list assigned to each $g \in V(H)$ by ACR Algorithm was $\{g\}$.

$\square$

Corollary 4.5. *Each multi-wheel extension is in $\mathcal{AR}_C$.*

$\square$

Corollary 4.6. *The class of wheeled graphs, and hence the variety generated by wheeled graphs, is contained in $\mathcal{AR}_C$.*

$\square$

4.5 Ramified graphs

We have examined many classes of graphs that are contained in $\mathcal{AR}_C$, such as $\mathcal{AR}_I$, $\mathcal{AR}_H$, the variety generated by wheeled graphs, and the connected graphs that admit a near unanimity function. All of these graphs are dismantlable. We naturally expected a nice relationship between the variety of dismantlable graphs and $\mathcal{AR}_C$. Unfortunately, $\mathcal{AR}_C$ does not contain the variety of dismantlable graphs, as shown by the graph in Figure 4.1, and the variety of dismantlable graphs does not contain $\mathcal{AR}_C$, as we will demonstrate shortly. We will in fact produce two families of ramified graphs that are contained in $\mathcal{AR}_C$.

Call a graph H a *net* if its vertex set can be partitioned into sets A and B where A induces a nontrivial k-cycle, and B induces a clique such that all adjacent pairs of vertices in A have a common neighbour in B if $k \geq 5$ and such that all pairs of vertices in A have a common neighbour in B if $k = 4$. If B is a single vertex, then H is a wheel, which is dismantlable as the hub of H covers all the other vertices of H. The graph in Figure 4.8 is a net that is not dismantlable.

Lemma 4.9. *Let H be a net with vertex partition $A \cup B$, where A induces a nontrivial cycle, and B is a clique. Let G be a connected supergraph H. If the ACR Algorithm*

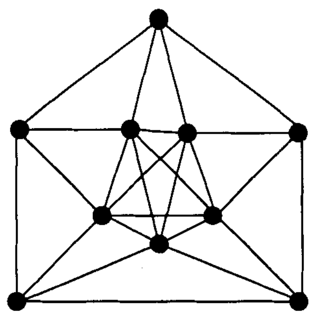


Figure 4.8: A ramified net with an induced 5 cycle.

succeeds when applied to H and G, then every non-singleton list produced by the ACR Algorithm contains at least one vertex from B.

Proof. We will analyze what occurs when we apply the ACR Algorithm to H and G.

We begin by assigning to each vertex g of G a list $L(g)$, where $L(g) = \{g\}$ if $g \in V(H)$ and $L(g) = V(H)$ otherwise. Thus:

$\diamondsuit$ Each non-singleton list contains at least one vertex from B.

We will prove that property $\diamondsuit$ is maintained as we process the edges of G.

Assume that property $\diamondsuit$ holds for all iterations of the ACR Algorithm up to the beginning of iteration $i \geq 1$. Let gg' be the edge to be processed in iteration i. If $|L(g)| = |L(g)| = 1$, then the property $\diamondsuit$ will also hold at the end of iteration i as neither list will change since we have assumed that the ACR Algorithm succeeds and so no empty lists are produced. Therefore we may assume, without loss of generality, that $|L(g)| \geq 2$. Thus, by assumption, $L(g)$ contains at least one vertex from B. If there exists a vertex of $L(g) \cap B$ that has support in $L(g')$, we are done. Therefore suppose that no vertex of $L(g) \cap B$ has support in $L(g')$. We will prove that this implies that $L(g')$ is of size one and that the list of g will be a singleton at the end of iteration i.

Since B is a clique and since no vertex of $L(g')$ supports a vertex in $L(g) \cap B$, we must have $L(g') \subseteq A$. As we have assumed property $\diamondsuit$ has held till the beginning of iteration i, this implies that $L(g) = \{a'\}$, where $a' \in A$. By our choice of G, the ACR Algorithm succeeds. Therefore a' has a neighbour $a \in L(g) \setminus B$. Thus $a \in A$. By the definition of a net, a and a' have a common neighbour $b \in B$. Clearly $b \notin L(g)$. Since $|L(g)| \geq 2$, the initial list of g must have been $V(H)$. Hence b was removed from the list of g in iteration j, $1 \leq j < i$. In other words, there exists a neighbour $g'' \in V(G)$ of g such that the edge gg'' was processed in iteration j, and b was removed from the list of g at this time. For the same reasons are before, the list of g'' at the beginning of iteration j must have been $\{a''\}$, with $a'' \in A$, such that a'' is not adjacent to b. In particular this means that a' and a'' are distinct vertices. Since a is in the list of g at the beginning of iteration i, a'' must be adjacent to a. Thus $a''aa'$ is a path in the k-cycle induced by A.

Suppose that a' and a'' have a common neighbour b' in B. Then, as before, $b' \notin L(g)$. Therefore there exists a neighbour g''' of g such the edge gg''' was processed in iteration p, $1 \leq p < i$, such that the list of g''' at the beginning of iteration p was $\{a'''\} \subset A$, where a''' is not adjacent to b', but is adjacent to a. This is impossible as the only neighbours of a in A are a' and a'', both of which are adjacent to b'. Hence a' and a'' can not have a common neighbour in B. In particular, this implies that A can't induce a 4 cycle in H by the definition of a net.

Since A induces at least a 5 cycle in H, a is the only common neighbour of a' and a'' in A. Along with the result from the previous paragraph, this implies that a is the only common neighbour of a' and a'' in the graph H. At the end of iteration j, the list of g was a subset of $N(a'')$. Therefore we must have that $L(g) \subseteq N(a'')$. Thus the list of g at the end of iteration i must be a subset of $N(a'') \cap N(a') = \{a\}$. We have chosen G such that the ACR Algorithm doesn't produce empty lists when it is applied to H and G. Therefore, the list of g as the end of iteration i is $\{a\}$.

□

Theorem 4.12. *Let H be a net and let G be a connected supergraph of H. If the ACR Algorithm succeeds when applied to H and G, there exists a retraction from G to H.*

Proof. Let H be a net with vertex partition $A \cup B$, where A induces a cycle and

B is a clique. Let G be a connected supergraph of H such that the ACR Algorithm succeeds when applied to H and G. Let $L(g)$ be the non-empty list produced by the ACR Algorithm for $g \in V(G)$. We claim that $\phi : H \to G$ is a retraction where

$$\phi(g) = \begin{cases} h & \text{if } L(g) = \{h\} \\ \text{any vertex of } B \cap L(g) & \text{otherwise.} \end{cases}$$

By Lemma 4.9, if $|L(g)| \geq 2$ for $g \in V(G)$, then $L(g) \cap B \neq \emptyset$. Thus ϕ is well defined. We will now show that ϕ is a homomorphism. Let gg' be a non-loop edge of G. If $L(g)$ consists of a single vertex, say h, then by arc consistency, h is adjacent to all vertices of $L(g')$, and so $\phi(g)\phi(g')$ is an edge of H. Thus assume that both $L(g)$ and $L(g')$ contain more than one vertex. Then, $\phi(g), \phi(g') \in B$. As B is a clique, $\phi(g)\phi(g')$ is an edge of H.

□

Corollary 4.7. *Each net is in $\mathcal{AR}_C$.*

□

Now that we know that all nets are in $\mathcal{AR}_C$, we will construct an infinite family of graphs such that each graph is a net and each graph is ramified;

Theorem 4.13. *There exists an infinite family of ramified nets in $\mathcal{AR}_C$.*

Proof. By Theorem 4.12, any net is in $\mathcal{AR}_C$. We will construct an infinite family of nets that are ramified.

Let N_k be the graph on $2k$ vertices, with vertex partition $A \cup B$ such that

i.) $A = \{a_1, \ldots, a_k\}$ and $B = \{b_1, \ldots, b_k\}$.

ii.) A induces a k-cycle, $a_1a_2 \ldots a_ka_1$.

iii.) B is a clique.

iv.) the only neighbours of a_i in B are b_{i-1} and b_i.

The graph in Figure 4.8 is N_5.

By statement *iv*, a_i and a_{i+1} are both adjacent to b_i. In other words, all pairs of adjacent vertices in A have a common neighbour in B. Thus N_k is a net for $k \geq 5$. We will now prove that N_k is ramified.

Consider the vertex a_i. It's only nontrivial neighbours in A are a_{i+1} and a_{i-1}, which are not adjacent as $k \geq 5$. Thus none of the nontrivial neighbours of a_i in A cover a_i. By statement *iii*, the only neighbours of a_i in B are b_{i-1} and b_i. The vertex b_{i-1} is not adjacent to a_{i+1} and the vertex b_i is not adjacent to a_{i-1}. Thus the neighbours of a_i in B don't cover a_i. Hence a_i is not dismantlable.

Now consider the vertex b_i. The neighbourhood of b_i is $B \cup \{a_i, a_{i+1}\}$. As b_i is the only common neighbour of a_i and a_{i+1} in B, no vertex of $B \setminus \{b_i\}$ covers b_i. In addition, neither a_i nor a_{i+1} covers b_i since neither a_i nor a_{i+1} is adjacent to all of B. Hence b_i is not dismantlable.

$\square$

Note that all nets have diameter at most 3 as all vertices of the induced nontrivial cycle have a neighbour in the central clique.

We will now construct a second family of ramified graphs in $\mathcal{AR}_C$. This construction will make use of the power graph defined in Section 4.1. Recall that if H is a graph, we denote the power graph of H by $\mathcal{P}(H)$ and we denote the copy of H in $\mathcal{P}(H)$ induced by the sets of size one by $H_{\mathcal{P}}$.

Let H be a graph and let J be a subgraph of H. Recall that we say H *dismantles to* J if there is a partial ordering $h_1, \ldots, h_k$ of H such that h_i is dismantlable in H_{i-1} for $i = 1, \ldots, k$, and $J = H_k$. Note that a graph trivially dismantles to itself via the empty ordering.

Theorem 4.14. *Let H be a connected ramified graph. Then there exists a connected ramified subgraph $\tilde{H}$ of $\mathcal{P}(H)$ such that $\mathcal{P}(H)$ dismantles to $\tilde{H}$ and $H_{\mathcal{P}}$ is an isometric subgraph of $\tilde{H}$.*

Proof. Note that since H is connected, so is $\mathcal{P}(H)$ and any retract of $\mathcal{P}(H)$. If $\mathcal{P}(H)$ is ramified, then the theorem is obviously true as we noted when we defined

$H_{\mathcal{P}}$ that $H_{\mathcal{P}}$ is an isometric subgraph of $\mathcal{P}(H)$. Thus let $\tilde{H}$ be a minimal subgraph of $\mathcal{P}(H)$ such that $\mathcal{P}(H)$ dismantles to $\tilde{H}$ and $H_{\mathcal{P}}$ is a subgraph of $\tilde{H}$. Therefore, either $\tilde{H}$ is ramified or any dismantlable vertex X of $\tilde{H}$ is of the form $X = \{x\}$, $x \in V(H)$. Suppose there exists a vertex $X = \{x\}$ of $\tilde{H}$ that is dismantlable. Thus $\{x\}$ is covered by some vertex Y in $\tilde{H}$. In particular, Y is a adjacent to $\{x\}$ in $\tilde{H}$. By the definition of adjacency in $\mathcal{P}(H)$, and hence in $\tilde{H}$, each vertex of H in Y must a neighbour of x in H. In other words, Y is a subset of the set of neighbours of x in H. As Y covers $\{x\}$ in $\tilde{H}$, Y is adjacent to $\{z\}$ in $\tilde{H}$ for all neighbours z of x in H and hence Y must be a subset of the set of neighbours of z in H. Thus any vertex in Y will cover x in H, a contradiction. Therefore $\tilde{H}$ must be ramified.

□

Corollary 4.8. *Let H be a connected ramified graph. Then there exists a connected ramified graph H' in $\mathcal{AR}_C$ such that H is isometric subgraph of H'.*

Proof. By Theorem 4.14, there exists a connected ramified graph $\tilde{H}$ such that $\mathcal{P}(H)$ dismantles to $\tilde{H}$ and $H_{\mathcal{P}}$ is an isometric subgraph of $\tilde{H}$. The graph $\tilde{H}$ is a retract of $\mathcal{P}(H)$ by Lemma 1.2. The graph $\mathcal{P}(H)$ is in $\mathcal{AR}_C$ by Proposition 4.2. As $\mathcal{AR}_C$ is a variety by Theorem 4.1, the retract of any graph in $\mathcal{AR}_C$ is again in $\mathcal{AR}_C$. Thus $\tilde{H}$ is in $\mathcal{AR}_C$. Since $H_{\mathcal{P}}$ is isomorphic to H and since $H_{\mathcal{P}}$ is an isometric subgraph of $\tilde{H}$, there exists a connected ramified supergraph H' of H such that H' is isomorphic to $\tilde{H}$ and H is an isometric subgraph of H'. Moreover, $H' \in \mathcal{AR}_C$ as H' and $\tilde{H}$ as isomorphic.

□

Call a graph that is the ramified retract of the power graph of some connected ramified graph a *ramified retract graph.*

Note that nontrivial cycles are ramified and connected, and so there exist ramified graphs of arbitrarily large diameter in $\mathcal{AR}_C$ by Corollary 4.8. Recall that nets have diameter at most 3. Thus the ramified retract graphs include graphs that are not nets, although the ramified retract of $\mathcal{P}(C_4)$ is a net. We conjecture that the ramified nets created in the proof of Theorem 4.13, N_k, $k \geq 5$, are not ramified retract graphs.

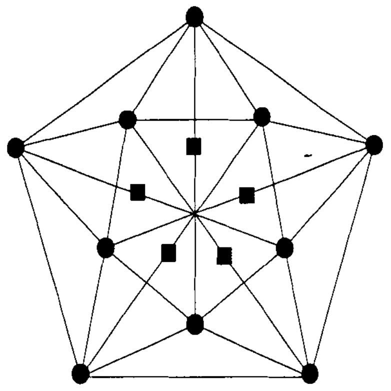


Figure 4.9: Let H be the graph in the above figure where the square vertices are a clique. Then H is the ramified graph to which $\mathcal{P}(C_5)$ dismantles.

Chapter 5

Summary

We end the thesis by reviewing how all the varieties studied relate to each other and by presenting a table of graphs that illustrate the differences between these varieties.

In Figure 5.1, we illustrate the relation between the various classes of absolute retracts (and their equivalent formulations), the variety generated by connected chordal graphs, the variety generated by wheeled graphs, the variety of connected graphs that admit near unanimity functions and the variety of dismantlable graphs.

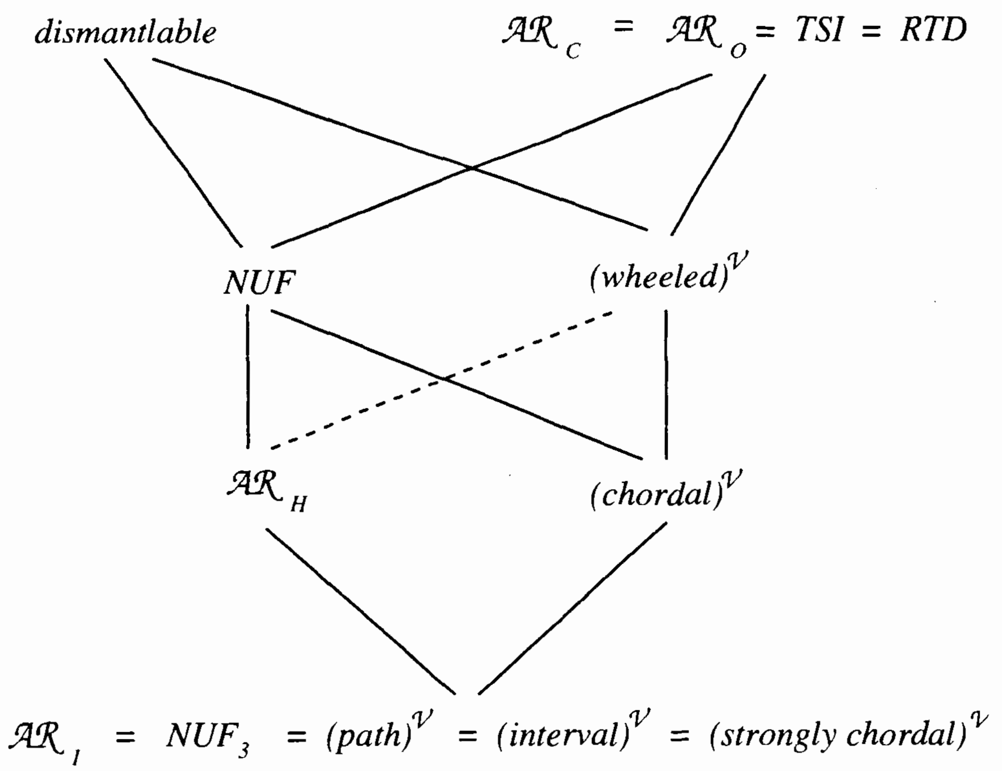

dismantlable
$\mathcal{AR}_C = \mathcal{AR}_O = TSI = RTD$
NUF
$(wheeled)^{\mathcal{V}}$
$\mathcal{AR}_H$
$(chordal)^{\mathcal{V}}$
$\mathcal{AR}_I = NUF_3 = (path)^{\mathcal{V}} = (interval)^{\mathcal{V}} = (strongly\ chordal)^{\mathcal{V}}$

Figure 5.1: A poset of some the varieties studied where the ordering is by inclusion. The superscript $\mathcal{V}$ indicates "the variety generated by (connected) ...", NUF indicates the variety of connected graphs that admit a near unanimity function, NUF$_3$ indicates the variety of connected graphs that admit majority functions, TSI indicates the class of connected graphs that admit a TSI of arity k for all $k \geq 1$ and RTD indicates the variety of graphs that have retraction tree duality. The dotted line indicates that we don't know if $\mathcal{AR}_H$ is contained in variety generated by wheeled graphs.

In Table 5.1, we illustrate the differences between the varieties in Figure 5.1.

Graph	$\mathcal{AR}_I$	$\mathcal{AR}_H$	$\mathcal{AR}_C$	(chordal)$^{\mathcal{V}}$	(wheeled)$^{\mathcal{V}}$	NUF	dismant.
$W_k, k \geq 4$	Y	Y	Y	Y	Y	Y	Y
Figure 1.2	N	Y	Y	Y	Y	Y	Y
Figure 2.6	N	N	Y	Y	Y	Y	Y
Figure 4.6	N	N	Y	N	Y	N	Y
Figure 4.7	N	N	Y	N	N	Y	Y
Figure 4.8	N	N	Y	N	N	N	N
Figure 5.2	N	N	N	N	N	N	Y

Table 5.1: The varieties of Figure 5.1 and graphs exhibiting their differences.

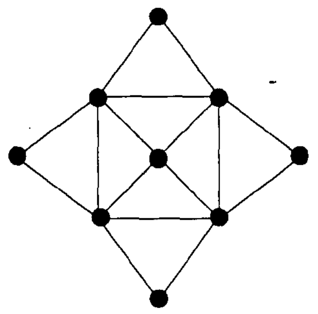


Figure 5.2: A dismantlable graph that is not in $\mathcal{AR}_C$.

When trying to classify the variety generated by connected chordal graphs, we introduced two new classes; stretched graphs and strongly stretched graphs. Both of these classes are varieties, and they both contain the variety generated by connected chordal graphs, as summarized in statement 5.1 (see Chapter 2.3 and Chapter 3.4):

$$(\mathit{chordal})^{\mathcal{V}} \subseteq \mathit{strongly\ stretched} \subset \mathit{stretched}. \tag{5.1}$$

Recall from Figure 5.1 that the superscript $\mathcal{V}$ indicates "variety generated by (connected) ...".

In Theorem 2.7, we were able to use stretched graphs to classify the variety generated by connected chordal graphs when restricted to $\mathcal{AR}_O$, i.e.,

$$(\mathit{chordal})^{\mathcal{V}} \cap \mathcal{AR}_H = \mathit{stretched} \cap \mathcal{AR}_H. \tag{5.2}$$

We were unable to prove the generalization of statement 5.2, and have left it as a conjecture, see Conjecture 1′ and Conjecture 2:

$$\begin{aligned} (\mathit{chordal})^{\mathcal{V}} &= \mathit{strongly\ stretched} \cap \mathcal{AR}_O && (5.3)\\ &= \mathit{strongly\ stretched} \cap \mathcal{AR}_C. && (5.4) \end{aligned}$$

Bibliography

[1] R. P. Anstee and M. Farber, *On bridged graphs and cop-win graphs*, J. Combin. Theory Ser. B **44** (1988), no. 1, 22–28.

[2] Kirby A. Baker and Alden F. Pixley, *Polynomial interpolation and the Chinese remainder theorem for algebraic systems*, Math. Z. **143** (1975), no. 2, 165–174.

[3] H.-J. Bandelt, A. Dählmann, and H. Schütte, *Absolute retracts of bipartite graphs*, Discrete Appl. Math. **16** (1987), no. 3, 191–215.

[4] H.-J. Bandelt, M. Farber, and P. Hell, *Absolute reflexive retracts and absolute bipartite retracts*, Discrete Appl. Math. **44** (1993), no. 1-3, 9–20.

[5] H.-J. Bandelt and E. Pesch, *Dismantling absolute retracts of reflexive graphs*, European J. Combin. **10** (1989), no. 3, 211–220.

[6] Hans-Jürgen Bandelt, *Graphs with edge-preserving majority functions*, Discrete Math. **103** (1992), no. 1, 1–5.

[7] Hans-Jürgen Bandelt and Gerasimos C. Meletiou, *An algebraic setting for near-unanimity consensus*, Order **7** (1990), no. 2, 169–178.

[8] Hans-Jürgen Bandelt and Erwin Pesch, *Efficient characterizations of n-chromatic absolute retracts*, J. Combin. Theory Ser. B **53** (1991), no. 1, 5–31.

[9] Hans-Jürgen Bandelt and Erich Prisner, *Clique graphs and Helly graphs*, J. Combin. Theory Ser. B **51** (1991), no. 1, 34–45.

[10] J. A. Bondy and U. S. R. Murty, *Graph theory with applications*, American Elsevier Publishing Co., Inc., New York, 1976.

[11] Karol Borsuk, *Sur les rétractes*, Fund. Math. **54** (1931), 152–170.

[12] ———, *Theory of retracts*, Monografie Matematyczne, Tom 44, Państwowe Wydawnictwo Naukowe, Warsaw, 1967.

[13] Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad, *Graph classes: a survey*, SIAM Monographs on Discrete Mathematics and Applications, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1999.

[14] R. Brewster, *Vertex colourings of edge-coloured graphs*, Ph.D. thesis, Simon Fraser University, 1993.

[15] Richard Brewster, Pavol Hell, Tomas Feder, Jing Huang, and Gary MacGillivray, *Near-unanimity functions and varieties of graphs*, in preparation.

[16] Graham R. Brightwell and Peter Winkler, *Gibbs measures and dismantlable graphs*, J. Combin. Theory Ser. B **78** (2000), no. 1, 141–166.

[17] A.A Bulatov, *Conservative constraint satisfaction problems are tractable*, in preparation.

[18] G. J. Chang and G. L. Nemhauser, *The k-domination and k-stability problems on sun-free chordal graphs*, SIAM J. Algebraic Discrete Methods **5** (1984), no. 3, 332–345.

[19] Gerard Chang, *k-domination and graph covering problems*, Ph.D. thesis, Cornell University, 1982.

[20] B. A. Davey, L. Heindorf, and R. McKenzie, *Near unanimity: an obstacle to general duality theory*, Algebra Universalis **33** (1995), no. 3, 428–439.

[21] G. A. Dirac, *On rigid circuit graphs*, Abh. Math. Sem. Univ. Hamburg **25** (1961), 71–76.

[22] Dwight Duffus and Ivan Rival, *A structure theory for ordered sets*, Discrete Math. **35** (1981), 53–118.

[23] Paul H. Edelman and Robert E. Jamison, *The theory of convex geometries*, Geom. Dedicata **19** (1985), no. 3, 247–270.

[24] M. Farber and R. E. Jamison, *Convexity in graphs and hypergraphs*, SIAM J. Algebraic Discrete Methods **7** (1986), no. 3, 433–444.

[25] Martin Farber, *Applications of linear programming duality to problems involving independence and domination*, Ph.D. thesis, Rutgers University, 1981.

[26] ______, *Bridged graphs and geodesic convexity*, Discrete Math. **66** (1987), no. 3, 249–257.

[27] T Feder and P Hell, *Width one and strict width 2 problems*, in preparation.

[28] Tomas Feder and Pavol Hell, *List homomorphisms to reflexive graphs*, J. Combin. Theory Ser. B **72** (1998), no. 2, 236–250.

[29] Tomas Feder, Pavol Hell, and Jing Huang, *List homomorphisms of graphs with bounded degrees*, in preparation.

[30] ______, *List homomorphisms and circular arc graphs*, Combinatorica **19** (1999), no. 4, 487–505.

[31] ______, *Bi-arc graphs and the complexity of list homomorphisms*, J. Graph Theory **42** (2003), no. 1, 61–80.

[32] Tomas Feder, Pavol Hell, and Kim Tucker-Nally, *List partitions and trigraph homomorphisms*, in preparation.

[33] Tomás Feder and Moshe Y. Vardi, *The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory*, SIAM J. Comput. **28** (1999), no. 1, 57–104 (electronic).

[34] D. R. Fulkerson and O. A. Gross, *Incidence matrices and interval graphs*, Pacific J. Math. **15** (1965), 835–855.

[35] Fănică Gavril, *Algorithms on clique separable graphs*, Discrete Math. **19** (1977), no. 2, 159–165.

[36] John Ginsburg, *Dismantlability revisited for ordered sets and graphs and the fixed-clique property*, Canad. Math. Bull. **37** (1994), no. 4, 473–481.

[37] M. C. Golumbic, *Algorithmic graph theory and perfect graphs*, Academic Press [Harcourt Brace Jovanovich Publishers], New York, 1980, With a foreword by Claude Berge, Computer Science and Applied Mathematics.

[38] László Hannák, *On the structure of many-valued logics*, Tanulmányok—MTA Számitástech. Automat. Kutató Int. Budapest (1984), no. 161, 134.

[39] P. Hell, J. Nešetřil, and X. Zhu, *Duality and polynomial testing of tree homomorphisms*, Trans. Amer. Math. Soc. **348** (1996), no. 4, 1281–1297.

[40] Pavol Hell, *Rétractions de graphes*, Ph.D. thesis, Université de Montréal, 1972.

[41] ______, *Absolute retracts in graphs*, Graphs and combinatorics (Proc. Capital Conf., George Washington Univ., Washington, D. C., 1973), Springer, Berlin, 1974, pp. 291–301. Lecture Notes in Math., Vol. 406.

[42] ______, *Graph retractions*, Colloquio Internazionale sulle Teorie Combinatorie (Roma, 1973), Tomo II, Accad. Naz. Lincei, Rome, 1976, pp. 263–268. Atti dei Convegni Lincei, No. 17.

[43] Pavol Hell and Ivan Rival, *Absolute retracts and varieties of reflexive graphs*, Canad. J. Math. **39** (1987), no. 3, 544–567.

[44] El Mostapha Jawhari, Maurice Pouzet, and Driss Misane, *Retracts: graphs and ordered sets from the metric point of view*, Combinatorics and ordered sets (Arcata, Calif., 1985), Contemp. Math., vol. 57, Amer. Math. Soc., Providence, RI, 1986, pp. 175–226.

[45] El Moustafa Jawhari, Maurice Pouzet, and Ivan Rival, *A classification of reflexive graphs: the use of "holes"*, Canad. J. Math. **38** (1986), no. 6, 1299–1328.

[46] Kalle Kaarli and Alden F. Pixley, *Polynomial completeness in algebraic systems*, Chapman & Hall/CRC, Boca Raton, FL, 2001.

[47] K. Kuratowski, *Topology. Vol. II*, New edition, revised and augmented. Translated from the French by J. Jaworowski, Academic Press, New York, 1966.

[48] Benoit Larose, *personal communication.*

[49] Benoit Larose, Cynthia Loten, and László Zádori, *A polynomial-time algorithm for near-unanimity graphs*, submitted.

[50] Benoit Larose and László Zádori, *The complexity of the extendibility problem for finite posets*, SIAM J. Discrete Math., to appear.

[51] ———, *Algebraic properties and dismantlability of finite posets*, Discrete Math. **163** (1997), no. 1-3, 89–99.

[52] C. G. Lekkerkerker and J. Ch. Boland, *Representation of a finite graph by a set of intervals on the real line*, Fund. Math. **51** (1962/1963), 45–64.

[53] A.K. Mackworth, *Consistency in networks of relations*, Artificial Intellegence **8** (1977), 99–118.

[54] Aleit Mitschke, *Near unanimity identities and congruence distributivity in equational classes*, Algebra Universalis **8** (1978), no. 1, 29–32.

[55] Peter Nevermann and Ivan Rival, *Holes in ordered sets*, Graphs Combin. **1** (1985), no. 4, 339–350.

[56] Richard Nowakowski and Ivan Rival, *Fixed-edge theorem for graphs with loops*, J. Graph Theory **3** (1979), no. 4, 339–350.

[57] ———, *The smallest graph variety containing all paths*, Discrete Math. **43** (1983), no. 2-3, 223–234.

[58] Richard Nowakowski and Peter Winkler, *Vertex-to-vertex pursuit in a graph*, Discrete Math. **43** (1983), no. 2-3, 235–239.

[59] E. Pesch, *Products of absolute retracts*, Discrete Math. **69** (1988), no. 2, 179–188.

[60] Erwin Pesch, *Minimal extensions of graphs to absolute retracts*, J. Graph Theory **11** (1987), no. 4, 585–598.

[61] ______, *Retracts of graphs*, Mathematical Systems in Economics, vol. 110, Athenäum Verlag GmbH, Frankfurt am Main, 1988.

[62] Erwin Pesch and Werner Poguntke, *A characterization of absolute retracts of n-chromatic graphs*, Discrete Math. **57** (1985), no. 1-2, 99–104.

[63] T Poston, *Fuzzy geometry*, Ph.D. thesis, University of Warwick, 1971.

[64] R. W. Quackenbush, I. Rival, and I. G. Rosenberg, *Clones, order varieties, near unanimity functions and holes*, Order **7** (1990), no. 3, 239–247.

[65] A Quilliot, *Homomorphismes, points fixes, rétractions et jeux de poursuite dans les graphs, les ensemble ordonnés et les espaces métriques*, Ph.D. thesis, Université de Paris VI, 1983.

[66] Donald J. Rose, *Triangulated graphs and the elimination process*, J. Math. Anal. Appl. **32** (1970), 597–609.

[67] Cs. Szabó and L. Zádori, *Idempotent totally symmetric operations on finite posets*, Order **18** (2001), no. 1, 39–47.

[68] Robert E. Tarjan, *Decomposition by clique separators*, Discrete Math. **55** (1985), no. 2, 221–232.

[69] Narayan Vikas, *Computational complexity of graph compaction*, Ph.D. thesis, Simon Fraser University, 1997.

[70] P. Winkler, *unpublished manuscript.*

[71] L. Zádori, *Monotone Jónsson operations and near unanimity functions*, Algebra Universalis **33** (1995), no. 2, 216–236.